WiPiEC Journal

Works in Progress in Embedded Computing Journal

Special Issue on selected papers of
Works in Progress Session within 27th Euromicro Conference on Digital System Design (DSD) and 50th Euromicro Conference Series on Software Engineering and Advanced Applications (SEAA), Paris, France, Aug. 28th – Aug. 30th, 2024.

August 2024 · Volume 10 · Issue 2 · ISSN 2980-7298

https://wipiec.digitalheritage.me

Message from the WiP Chair

It is my great pleasure to welcome you to the Works in Progress (WiP) Session within *the 27th Euromicro Conference on Digital System Design (DSD'2024*) and *the 50th Euromicro Conference on Software Engineering and Advanced Applications (SEAA'2024)*. I am honored to be a Chair of this Edition, which coincides with the 50th anniversary of SEAA events and that will be held in charming Paris, Sorbonne University, from 28th to 30th of August 2024. A feature of this year's WiP Session is the significantly increased interest of the authors in participating. This shows the justification of renewing this session, which we did last year.

This year we selected 19 works to be presented in the WiP Session, 9 from DSD and 10 from SEAA thematic areas.

A high-quality WiP program would not have been possible without the contribution of authors, enthusiastic colleagues from DSD'2024 and SEAA'2024 Committees as well as MECOnet, MANT and Euromicro staff who helped this edition to be published. We are especially grateful to the Jovan Djurkovic for his work on the technical editing and publishing of the material. Works in Progress in Embedded Computing Journal (WiPiEC) will traditionally publish the selected papers from this session in open access form.

I am already convinced that WiP Session will grow in the future and become especially popular among young researchers, who are significant factor for the future of our conferences.

Welcome to DSD'2024, SEAA'2024 and Works in Progress (WiP) Session!

Prof. Dr Radovan Stojanović
University of Montenegro, Montenegro
General Chair of WiP Session within DSD'2024 and SEAA'2024 events

Contents

Resistance of Radiation Tolerant TMR Shift Registers to Optical Fault Injections

Dmytro Petryk[1], Peter Langendörfer[1,2] and Zoya Dyka[1,2]
[1] *IHP – Leibniz-Institut für innovative Mikroelektronik,* Frankfurt (Oder), Germany
[2] *BTU Cottbus-Senftenberg,* Cottbus, Germany
{petryk, langendoerfer, dyka}@ihp-microelectronics.com

Abstract—**Protection of information is essential for IoT devices. They are often subject to lab analysis with the objective to reveal secret hidden information. One of the ways to reveal the cryptographic key is to perform optical Fault Injection attacks. In this work, we investigated the IHP radiation tolerant shift registers built of Triple Modular Redundant flip-flops. In our experiments, we were able to inject different transient faults into TMR registers using a single laser beam.**

Keywords — Fault Injection attack, laser, reliability, security, triple modular redundancy, standard library, rad-hard flip-flops.

I. INTRODUCTION

Confidentiality, data integrity, availability of services, and (mutual) authentication of communicating devices are the main goals which have to be implemented to guarantee a secure communication of the devices. It is ensured by using cryptographic algorithms. Cryptographic protocols are based on the secrecy of the used keys. The goal of many attacks is to extract the private/secret key. The cryptographic approaches are based on complex mathematical problems, which cannot be solved in a reasonable time if the used private/secret key is long enough. This changes dramatically if an attacker has physical access to the attacked device. Physical parameters such as the execution time of cryptographic operations, current drawn from the power supply, electromagnetic radiation, etc., called side-channel effects, can be measured during the execution of cryptographic operations and later analysed using statistical methods, signal processing methods or machine learning methods. Practically, many cryptographic algorithms process a private key or a secret scalar bit-by-bit. Many attacks concentrate on the analysis of registers' activity. Some attacks exploit the key-dependent hamming weight of the data, stored in registers, and require usually many traces for the analysis [1]. Other attacks exploit the distinguishability of the registers addressing and can be successful when analysing many traces [2] as well as a single trace only [3], [4]. The sensitivity of cryptographic chips to the environmental and operating parameters – temperature, voltage, light, etc. – can be exploited to reveal the cryptographic key too. A fluctuation of these parameters can cause fault(s). Cryptographic keys can be revealed by analysing such faults. Due to this fact, IoT devices have to be resistant to a broad spectrum of physical attacks such as Side-Channel Attacks (SCA) & Fault Injection (FI) attacks.

There are many different possibilities to increase the resistance against SCA attacks. The processed data can be hidden using different kinds of noise, or by applying randomization of inputs and the processing sequence. The main idea of the countermeasures is to make the measurable side-channel effects not dependable on the processed cryptographic key, or – at least – to reduce this dependability significantly. A common approach to increase resistance against FI attacks is to implement countermeasures based on redundancy techniques. The main idea of the countermeasures is to avoid successful manipulation of device parameters, or – at least – to reduce the influence of successful manipulations of the design.

In this work, we investigate the applicability of the radiation hard Triple Modular Redundancy (TMR) technique against optical FI attacks. Our first investigations were published in [24]. In this paper, we present a complete investigation of the resistance of radiation tolerant shift registers to optical FI attacks as well as provide the area of the investigated TMR flip-flops sensitive to laser illumination. The paper is structured as follows. Section II describes optical FI attacks, redundancy as a countermeasure against FI attacks and our previous works. Section III describes the attacked chip, our FI setup and attack scenarios. Section IV gives the results of our optical Fault Injection attacks against TMR shift registers. Section V concludes this work.

II. OPTICAL FAULT INJECTION ATTACKS

Optical FI attacks using lasers belong to the class of semi-invasive attacks and exploit the sensitivity of semiconductors to the visible light, i.e. internal photoelectric effect. This phenomenon describes the generation of "free" electrons in semi-conductor/dielectric material under light. A sufficient number of the additional "free" electrons generated in close vicinity to the transistor gate of the attacked closed transistor results in a current flow that can switch a transistor from the high resistance state to the low resistance state. If the attacked transistor changes its state from OFF to ON it can cause a change of the logic cell state. The pioneering work regarding optical FI attacks was published already in 2002 [5]. The attacks can be performed through the back-side (silicon) of the chip [22] or the front-side (metal layers) [17] or the lateral-side [25]. To implement the former, near-infrared (NIR) and infrared (IR) lasers are used. This is due to a low absorption of NIR and IR waves propagating through silicon [22]. The latter can be implemented with any kind of wavelength [17], but the optimal choice is a laser with 800 nm wavelength. An overview of FI attacks against implementations of cryptographic operations

using elliptic curves, including laser-based attacks, is given in [19]. An overview of optical FI attacks against different cryptographic and non-cryptographic implementations as well as used equipment is given in [6]. An optical FI attack against secured microcontrollers is reported in [14]. A multi-fault laser attack on an RSA-CRT implementation is described in [15].

A. H/W redundancy as a countermeasure against faults

Hardware redundancy is one of the means to reduce the success of FI [18]. We concentrate here on the hardware redundancy, especially on the TMR technique implementing registers. So-called standard NMR and full NMR are well-known types of hardware redundancy circuit architectures [11]. The standard TMR architecture for storing a single bit of information is based on a triplication of the flip-flop and implementation of a voter. The full TMR architecture is based on the triplication of each cell, i.e. flip-flop (FF), voter and other combinational logic cells have to be triplicated. The weak point of the standard TMR architecture is its voter, due to the fact that a fault injected into the voter manipulates the output value of the TMR-FF. To prevent this issue a full-TMR architecture can be used. The triplication of the voter increases the power consumption and area overhead of the TMR-FF.

B. Our previous works

In our previous publications we described successful attacks against different standard logic cells manufactured in IHP's 250 nm as well as in IHP's 130 nm technologies. We were able to inject transient as well as permanent faults into inverter-, NAND-, NOR-cells and FFs [7], [8]. Additionally, we investigated the possibility to inject a fault (using a red laser) into Resistive Random Access Memory (RRAM) cells [16] and radiation hard Junction Isolated Common Gate (JICG) flip-flops [10]. TABLE I gives a short overview of our results.

TABLE I. OVERVIEW OF RESULTS OF OPTICAL FI ATTACKS AGAINST DIFFERENT CELLS MANUFACTURED IN IHP'S TECHNOLOGIES

Red laser	Tech-nology, nm	Successfully attacked structures		
		Standard IHP library cells	RRAM	JICG
Single-mode	250	'1'→ '0' INV NOR NAND	([a]) 0↔1, 0↔US, US↔1, 0↔Stuck-at 1, US↔Stuck-at 1, 1↔Stuck-at 1	'1'→'0'; '0'→'1'.
Multi-mode	130	'0'→ '1' FF	Not investigated yet	Not investi-gated yet

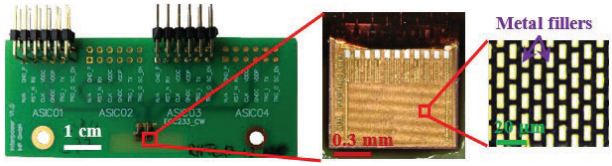
[a] SPECIFIC OF RRAM (4 DEFINED STATES [16]); US IS AN UNDEFINED STATE.

All our FI attacks were successful, even against radiation-hard JICG FF that utilizes duplication of non-standard transistors. These transistors are placed at a distance to ensure blocking capability against particle hit. Despite the implemented measures, we were able to illuminate a pair of duplicated transistors and manipulate the data stored. Details about successful attacks on JICG FFs can be found in [9], [23]. Since our attacks against the JICG flip-flops were successful we decided to evaluate the TMR technique available at IHP [20]. We experimented with shift registers that have been: designed to survive space missions, evaluated in single event effect measurements, verified as very radiation tolerant [21].

III. ATTACKED CHIP, OUR EXPERIMENTAL SETUP AND ATTACK SCENARIOS

A. IHP TMR shift register

We attacked radiation tolerant TMR shift registers designed at IHP and manufactured in IHP's CMOS 130 nm technology [21]. Two chips, each containing a single 1024-bit long TMR shift register, were bonded on a PCB, see **Fig. 1**. The attacked TMR circuit is based on the standard TMR architecture, i.e. there are 3072 flip-flops and 1024 voters, with two additional delay elements of 0.5 ns and 1.0 ns to filter possible transients at the inputs of the FFs, see **Fig. 2**. More details about the delay elements can be found in [21].



(a) *(b)*

Fig. 1. The attacked chips: *(a)* – PCB with 2 TMR shift registers; *(b)* – a TMR shift register chip, zoomed in.

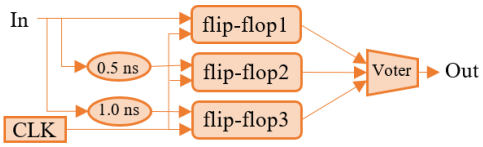


Fig. 2. Structure of the attacked TMR-FF.

B. FI setup and attack scenarios

In our experiments we used a setup that consists of: a modified Riscure Diode Laser Station (DLS), a VC glitcher, a PC with the Riscure Inspector software, a stable power supply, a signal generator and an oscilloscope. A detailed description of the experimental setup can be found in [23] and [23]. The original FI setup consists of a DLS that is equipped with two multi-mode lasers: a near-infrared (1064 nm) laser and a red (808 nm) laser [12], and with a single-mode laser from Alphanov [13]. In this work we applied the same equipment as in our experiments described in [7], [8] and [10]. The laser parameters, such as the laser pulse power and duration, were already evaluated experimentally in [23]. This allows to compare all our FI attack results.

In our experiments, we sent a constant input ('1' or '0') to the attacked shift register during the whole time of the experiment. We performed our experiments applying two clock signal frequencies: 10 MHz and 50 MHz. We performed attacks from the front-side since the decapsulation was not required. The placement of triplicated FFs in the layout of the attacked circuit is shown in **Fig. 3**. A triplicated flip-flop and its voter are placed in one row. The FFs are placed at different distances from each other. This distance reduces the probability of a fault occurrence caused by high energy particles due to the fact that at least two of FFs have to be influenced simultaneously. Due to the placement of the cells in the TMR-FF layout, we have three possible attack scenarios:

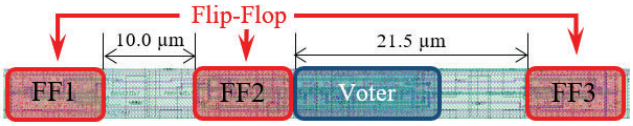


Fig. 3. Layout of the TMR-FF attacked.

Scenario1: attack illuminating logic cells of the voter.
Scenario2: attack illuminating FF1 and FF2 simultaneously.
Scenario3: attack illuminating the whole TMR-FF, i.e. illuminating three FFs and the voters simultaneously.

The issue is that during the attack according to *Scenario3* multi-faults can be injected. It can cause the "attack failed" case. For example, if faults are injected into two (or into all three) FFs and an additional fault is injected into the voter, the output of the TMR-FF will be equal to its fault-free state. Please note that additional difficulties in our experiments arise due to the so-called metal fillers. The metal fillers are small metal areas that are placed in different metal layers between the connection wires to maintain homogeneity of the etching process during manufacturing. Due to the metal fillers, the internal structure of the attacked TMR shift register is not visible through a microscope from the front-side, see **Fig. 1**. Nevertheless, in our previous experiments with the IHP standard logic cells we were able to induce faults successfully despite inserted metal fillers. Due to the fact that cells are unevenly covered by metal fillers, we expected that not all TMR-FFs will be successfully influenced.

IV. Experiments and Results

We performed our experiments according to *Scenario1* and *Scenario2* only, due to the difficulties of Scenario3 described above. We illuminated the last TMR-FF of the register first. Additionally, we attacked other randomly selected TMR-FFs.

Scenario1: We started our experiments using the single-mode red laser. The attacked register was clocked with a frequency of 10 MHz, i.e. the clock signal period is 100 ns. We started with 10 % laser power and 130[1] ns pulse duration using a 50× objective. We applied a 0.5 µm step size for the X- and Y-axis for the scanning of the voters' area. If we did not observe a fault during a scan of the whole area of the voter, we increased laser beam power stepwise up to 100 % and/or pulse duration up to 280 ns. After the experiments with the 50× objective we switched to the 100× objective and performed similar experiments. We did not observe a fault injection into the attacked TMR-FF. We attacked additional 34 TMR-FFs. All attacks were unsuccessful. Our experiments with the same laser parameters but with the clock frequency of 50 MHz[2], were not successful as well.

We also performed experiments using the more powerful multi-mode red laser. We started the experiments with 50 % laser beam power. All other parameters – the pulse duration, objective, etc. – were applied as for the experiments described above. After we reached the "maximum" parameter values, we switched to 100× objective and performed similar experiments. We attacked 24 TMR-FFs in total, but we did not observe a fault injection. We expected successful attacks using the multi-mode laser since we were able to influence standard library cells manufactured in IHP's 130 nm technology with metal fillers [8]. Nevertheless, all our FI attacks illuminating the voter of different FFs were not successful. It can be explained due to the difference regarding laser parameters between experiments described in [8] and here: in previous work we used a standard DLS from Riscure, after then the DLS was modified to allow using the Alphanov single-mode laser. We assume that the modification reduced the laser beam power.

Scenario2: In this attack scenario, we used the multi-mode laser to influence both FFs simultaneously. We started with 40 % power, 130 ns pulse duration, 20× objective and 1 µm step size for the X- and Y-axis. The register was clocked with a frequency of 10 MHz. We observed repeatedly successful bit-set ('0'→'1') and bit-reset ('1'→'0') faults. **Fig. 4** shows waveform of the measured signals demonstrating bit-reset fault into the last TMR-FF of the attacked TMR shift register.

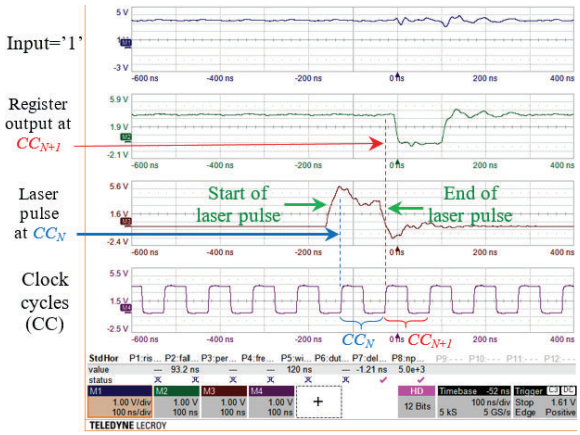


Fig. 4. Oscilloscope waveforms of laser pulse, clock, input and output of the attacked shift register measured demonstrating successful *bit-reset* fault into the last flip-flop.

Afterwards we performed similar experiments with a 5× objective and observed successful faults also. We attacked 24 TMR-FFs in total, of which 10 TMR-FFs were manipulated.

We also performed experiments with an increased laser power (up to 100%) and pulse duration (up to 280 ns) to assess the influence on the behaviour of the register. Neither *permanent*[3] nor *stuck-at*[4] faults were observed, even when applying maximum laser beam power. After the performed experiments, the register is fully functional, i.e. its

[1] We obtained this minimal pulse duration as a sum of the rise time of the single-mode laser to reach its peak power [13] and the clock signal period, i.e. 30 ns+100 ns=130 ns. The maximal pulse duration is as a sum of the rise time and the duration of 2.5 clock signal periods that results into 30 ns+2.5*100 ns=280 ns.

[2] We started with the pulse duration of 30 ns+20 ns=50 ns and increased it up to 30 ns+2.5*20 ns=80 ns.

[3] change of logic state is no more possible.

[4] change of logic state cannot be changed before the device shutdown.

surface is intact and the internal structure of the chip was not damaged. TABLE II gives an overview of our successful optical FI attacks against the IHP TMR registers for the *Scenario2*.

TABLE II. RESULTS OF ATTACKS AGAINST THE TMR REGISTERS

Register clock frequency, MHz	Register input	Magnification objective	Power, %[a]	Pulse, ns	Type of fault
50	'0' '1'	20×	80-100 90-100	80	*bit-set*[b]
				50-80	*bit-reset*
	'0' '1'	5×	90-100 60-100		*bit-set* *bit-reset*
10	'0' '1'	20×	40-100 45-100	130-280	*bit-set* *bit-reset*
	'0' '1'	5×	65-100 55-100		*bit-set* *bit-reset*

[a] POWER MEASUREMENT UNIT IN RISCURE SOFTWARE [12]. THE FIRST VALUE IN THE COLUMN "POWER" REPRESENTS THE MINIMAL POWER THAT RESULTS IN REPEATABLE FAULTS.
[b] THE FAULT IS NOT FULLY REPEATABLE.

According to the TMR shift register's layout, we observed successful FIs when the centre of a laser beam spot was over area marked yellow in **Fig. 5**. Due to the unknown profile of the

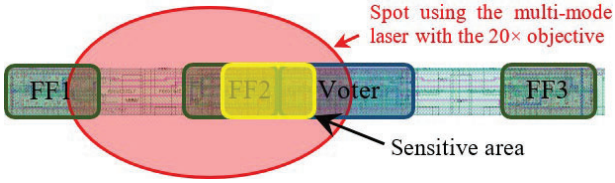


Fig. 5. Sensitive areas in the attacked TMR-FF.

laser beam intensity distribution of the used multi-mode laser and the fact that our precisely localised attacks on the voter (*Scenario1*) failed, we assume that faults were successfully injected in two adjacent FFs in the TMR-FF. Nevertheless, we do not exclude the fact that faults could be injected in each pair or even into three FFs simultaneously using the 5× objective since the laser beam spot allows to cover the whole TMR-FF.

V. CONCLUSION

In this work, we investigated the vulnerability of IHP radiation tolerant TMR-registers based on standard library flip-flops manufactured in 130 nm technology against front-side optical FI attacks. We performed attacks against the voter (*Scenario1*) and the triplicated FFs (*Scenario2*). We performed attacks using the Alphanov's single-mode red laser as well as the Riscure's multi-mode red laser. Attacks against voters were unsuccessful. Performing attacks against FFs we were able to inject transient repeatable *bit-set* as well as *bit-reset* faults into the IHP TMR shift registers using the multi-mode laser.

REFERENCES

[1] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis", in Advances in Cryptology — CRYPTO' 99, vol. 1666, Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 388–397.

[2] K. Itoh, T. Izu, and M. Takenaka, "Address-Bit Differential Power Analysis of Cryptographic Schemes OK-ECDH and OK-ECDSA", in Cryptographic Hardware and Embedded Systems, 2002, pp. 129–143.

[3] I. Kabin, Z. Dyka, D. Klann, and P. Langendoerfer, "Methods increasing inherent resistance of ECC designs against horizontal attacks", Integration, vol. 73, Jul. 2020, pp. 50–67.

[4] I. Kabin, Z. Dyka, and P. Langendoerfer, "Atomicity and Regularity Principles Do Not Ensure Full Resistance of ECC Designs against Single-Trace Attacks", Sensors, vol. 22, no. 8, Art. no. 8, Jan. 2022.

[5] S. Skorobogatov and R. Anderson, "Optical Fault Induction Attacks", Workshop on Cryptographic Hardware and Embedded Systems (CHES), USA, San Francisco, Aug, 13–15, 2002, pp. 2–12.

[6] D. Petryk et al., "Optical Fault Injections: a Setup Comparison", Proc. PhD Forum of the 8th BELAS Summer School, Estonia, Tallinn, June 20–22, 2018, pp. 1–5.

[7] D. Petryk, Z. Dyka and P. Langendörfer, "Sensitivity of Standard Library Cells to Optical Fault Injection Attacks in IHP 250 nm Technology", 2020 9th Mediterranean Conference on Embedded Computing (MECO), Montenegro, Budva, June 8–11, 2020, pp. 1–4.

[8] D. Petryk, Z. Dyka, J. Katzer and P. Langendörfer, "Metal Fillers as Potential Low Cost Countermeasure against Optical Fault Injection Attacks", 2020 IEEE East-West Design & Test Symposium (EWDTS), Bulgaria, Varna, Sept. 4–7, 2020, pp. 1–6.

[9] R. Sorge et al., "JICG MOS transistors for reduction of radiation effects in CMOS electronics", 2018 IEEE Topical Workshop on Internet of Space (TWIOS), USA, CA, Anaheim, Jan. 14–17, 2018, pp. 17–19.

[10] D. Petryk et al., "Optical Fault Injection Attacks against Radiation-Hard Shift Registers", 2021 24th Euromicro Conference on Digital System Design (DSD), Italy, Palermo, Sept. 1–3, 2021, pp. 371–375.

[11] V. Petrovic and M. Krstic, "Design Flow for Radhard TMR Flip-Flops", 2015 IEEE 18th International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS), 2015, pp. 203–208.

[12] Riscure. Diode Laser Station Datasheet, 2011. https://www.riscure.com/security-tools/inspector-hardware/

[13] Alphanov PDM laser sources. Rise time comparison. URL: https://www.alphanov.com/en/products-services/pdm-laser-sources

[14] Jasper G. J. van Woudenberg et al., "Practical optical fault injection on secure microcontrollers", 2011 Workshop on Fault Diagnosis and Tolerance in Cryptography, Japan, Nara, Sept. 28, 2011, pp. 91–99.

[15] E. Trichina and R. Korkikyan, "Multi Fault Laser Attacks on Protected CRT-RSA", 2010 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC), USA, CA, Aug. 21, 2010, pp. 75–86.

[16] D. Petryk et al., "Sensitivity of HfO2-based RRAM Cells to Laser Irradiation", Microprocessors and Microsystems, Volume 87, 2021, 104376, ISSN 0141-9331, pp. 1–20.

[17] S. de Castro et al., "Frontside Versus Backside Laser Injection: A Comparative Study", ACM Journal on Emerging Technologies in Computing Systems, 2016, 13 (1), pp. 7.

[18] M. Krstic, "Optimizing Design of Fault-Tolerant Computing Systems", Workshop on Hardware Design and Theory, Austria, Vienna, 2017.

[19] J. Fan et al., "State-of-the-art of secure ECC implementations: a survey on known side-channel attacks and countermeasures", 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), USA, CA, Anaheim, 2010, pp. 76-87.

[20] IHP – Leibniz-Institut für innovative Mikroelektronik. URL: https://www.ihp-microelectronics.com/about-us

[21] O. Schrape et al., "Design and Evaluation of Radiation-Hardened Standard Cell Flip-Flops", in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 68, no. 11, Nov. 2021, pp. 4796-4809.

[22] D. Lewis et al., "Backside Laser Testing of ICs for SET Sensitivity Evaluation", IEEE Transactions On Nuclear Science, Vol. 48, No. 6, Dec. 2001, pp. 2193-2201.

[23] D. Petryk, "Investigation of sensitivity of different logic and memory cells to Laser Fault Injections", Doctoral thesis, BTU Cottbus – Senftenberg, 2024. DOI: 10.26127/BTUOpen-6664

[24] D. Petryk et al., "Laser Fault Injection Attacks against Radiation Tolerant TMR Registers", 2022 IEEE 23rd Latin American Test Symposium (LATS), Montevideo, Uruguay, 2022, pp. 1-2.

[25] J. Rodriguez, A. Baldomero, V. Montilla and J. Mujal, "LLFI: Lateral Laser Fault Injection Attack", 2019 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC), Atlanta, USA, 2019, pp. 41-47.

Accelerating Cryptographic Algorithms on RISC-V cores using Carryless Multiplication

Simi Sukumaran*, Tripti S Warrier
CarS Lab, Department of Electronics, CUSAT
Cochin, India
simisukumaran@cusat.ac.in, tripti@cusat.ac.in

Babu P S, Neel Gala
Incore Semiconductors
Chennai, India
babu.ps@incoresemi.com, neelgala@incoresemi.com

Abstract— **Edge computing emerges as a critical paradigm in the wake of Internet of Things (IoT) and 5G New Radio (5GNR). It catalyzes the demand for energy-efficient devices that have resilient CPUs with lean physical footprints. Mitigating the security challenges in these networked devices necessitates Bit Manipulation Instruction (BMI) inclusive architectures to improve Galois Field (GF) arithmetic, which is a fundamental step for most cryptographic algorithms. All major Instruction Set Architectures (ISA), including RISC-V incorporate dedicated instructions for carryless multiplication, recognizing its significant contribution in cryptographic applications. Acknowledging the fact, this paper introduces a novel approach to enhance the performance of GF arithmetic using carryless multiplication. The approach presents a promising avenue by improving the execution cycle counts of a real-world cryptographic application like the Advanced Encryption Scheme (AES) and can be scaled to all GF-based cryptographic algorithms. The proposed GF algorithm effectively maps the Carryless Multiplication Instruction of the ratified RISC-V Zbc extension. Evaluations indicate about 4.5x performance improvement for multiple schemes of AES using an open-source RISC-V core (SweRV-EL2™ 1.3) without incurring any additional overhead in terms of area as well as compiler support.**

Keywords- Cryptography; Galois Field Arithmetic; RISC-V; AES;

I. INTRODUCTION

The Internet of Things (IoT) [1] and communication network topologies, such as 5G New Radio (5GNR) [2], are driving the transition from cloud to edge computing. The stringent architectural constraints of portable devices necessitate optimal performance at the least cost of area and power. This motivates researchers to derive the best from current designs through modifications to enhance crucial and frequent applications such as cryptography and error correction. Most measures that tackle the security and safety concerns according to today's device design trends rely predominantly on bit manipulation algorithms. The RISC-V community has been working on the introduction of BMI for the past couple of years and has been successful in ratifying four key BMI sub-extensions: Zba/b/c/s [3] for which GNU Compiler Collection (GCC) support is also available. The Zbc extension endows carryless multiplication (CLMUL) instructions, engaging which the GF arithmetic and any allied cryptographic algorithms like Advanced Encryption Standard (AES), Reed-Solomn and even post quantum cryptographic algorithms can be accelerated.

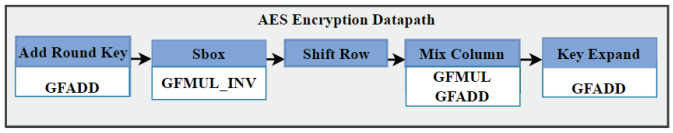


Figure 1: Datapath showing GF operations in AES encryption process

AES plays a critical role in minimizing the security vulnerabilities among networked IoT devices by guaranteeing secure encrypted communication. The functional breakdown of AES reveals strong reliance on GF arithmetic. Fig. 1 depicts the GF arithmetic units employed during different stages of AES encryption. Similarly, most of the cryptographic schemes have repeated instances of Galois Field Multiplication (*GFMUL*), Inversion (*GFMUL_INV*), and Addition (*GFADD*). This promotes the development of flexible galois field processors [4], dedicated hardware and hardware-software co-designs [5] to enhance the performance and security of edge devices. The *GFADD* represents a simple xor operation, whereas the *GFMUL* corresponds to a carryless multiplication followed by reduction logic [6], as illustrated in Fig. 2. Inversion uses the repeated square and multiply approach [7], which also aligns with the requirement for optimum Galois Field Multiplication.

The above facts demonstrate that optimized GF arithmetic employing CLMUL implementations provides the necessary impetus to boost cryptographic applications. Inspired by these observations, this work proposes a novel approach to modify the GFMUL algorithm using three CLMUL instructions, one for multiplication and the rest for polynomial reduction. Further, we simulate, validate and evaluate modified GFMUL and GFMUL_INV algorithms on SweRV-EL2 core with inbuilt Zbc extension [3] and measure the performance improvement

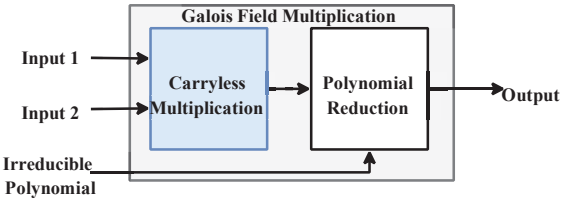


Figure 2: Galois Field multiplication using two stage logic

obtained on a real-world cryptographic application - AES. Our work stands distinct from the state-of -art works by exploiting the existing CLMUL instructions for polynomial reduction rather than custom instructions in RISC-V processors. Hence, it eliminates the modification of the RISC-V GNU toolchain and exempts any area overhead on the core for incorporating such custom instructions.

II. Proposed Work

The lower footprint requirement of lean embedded processors, along with the RISC-V GCC compiler's failure to render the CLMUL (carryless multiplication) instruction in embedded and crypto benchmarks emphasize the need to quantify the significance of these instructions. This section modifies the basic GF multiplication method utilizing carryless multiplication and maps it to AES- an essential cryptographic application.

A. Mathematical Basis of GF(2^m)

Popularly known as binary extension field, GF(2^m) represents elements with polynomial degrees up to m-1. The polynomial representation of GF(2^m) is shown in Equation (1). It involves a unit element (α^0), a zero element ($\alpha^{-\infty}$), and a primitive element (α). Equation (2) represents the irreducible polynomial f(x) associated with GF(2^m), where α (primitive element) forms its root.

$$F(\propto) = a_{m-1} \propto^{m-1} + \cdots . + a_1 \propto + a_0 ; \quad a_i \in GF(2), 0 \leq i \leq m-1 \tag{1}$$

$$f(x) = x_m + f_{m-1}x_{m-1} + \cdots + f_1 x + f0; \tag{2}$$

Arithmetic operations in GF(2^m) mandate a polynomial reduction, equivalent to the modulo using the irreducible field polynomial when the degree of the result exceeds m-1. All the arithmetic operations explained below use a(x) and b(x) represented by Equation (3) as inputs.

$$a(x) = a_{m-1}x^{m-1} + \cdots + a_1 x + a_0$$
$$b(x) = b_{m-1}x^{m-1} + \cdots + b_1 x + b_0 \tag{3}$$
$$a_{i,} \in GF(2), 0 \leq i \leq m-1$$

GFADD is a simple, yet most repeatedly used operation in cryptographic algorithms. It carries out direct bitwise XOR of the corresponding coefficients of the inputs as in Equation (4) and does not require polynomial reduction as it fits in the field.

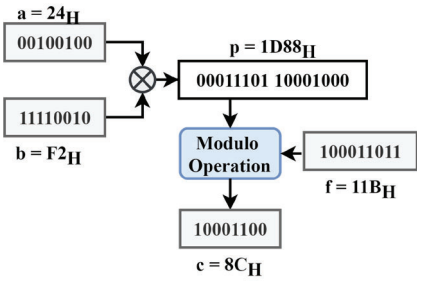


Figure 3: GF(2^8) Multiplication using standard logic

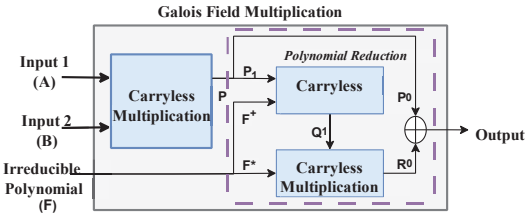


Figure 4: Modified Galois Field multiplication logic. Polynomial reduction is also broken down to make use of carryless multiplication. P, Q, and R are intermediate carryless multiplication results.

$$s(x) = (a_{m-1} \oplus b_{m-1})x^{m-1} + \cdots + (a_0 \oplus b_0) \tag{4}$$

GFMUL is a two-step process comprising of a carryless multiplication followed by polynomial reduction, as shown in Fig. 2. The result of carryless multiplication is of degree 2m-2 and should be reduced using the irreducible polynomial to keep the final result c(x)=p(x) mod f(x) within the field. Fig. 3 represents a GF(2^8) example of *GFMUL* with field polynomial $f(x) = x^8+x^4+x^3+x+1$ (11BH).

B. Modified Galois Multiplication Algorithm

Galois Multiplication being a complex but inevitable operation in cryptography is explored and implemented in different ways. Kuo et al. employs an appealing two-step process using carryless multiplication followed by polynomial reduction as shown in Fig. 2 [6]. The work in this paper modifies the *GFMUL* logic using three carryless multiplications as shown in Fig. 4. Here the polynomial reduction is also broken down into two carryless multiplications. The mathematical grounds of this mapping is derived from the explanations given by Gureon et al., but is not included here due to space considerations [8].

The Algorithm 1 gives a stepwise walk through this modified approach with two N-bit inputs (of degree N-1) and an N+1 bit field polynomial (of degree N). As long as the degrees of inputs (A and B) and field polynomial (F) are N-1 and N, respectively, F+ and F remain the same. Hence, using the proposed algorithm, the *GFMUL* result can be attained with three carryless multiplications and minor logic to extract the MSB and LSB of the intermediate results. Fig. 5 clearly demonstrates the steps depicted by Algorithm 1 using GF(2^8). This degree is chosen because many of the real world cryptographic applications, like AES make use of GF(2^8) with Field polynomial as $F(x)=x^8+x^4+x^3+x+1$ (11BH). Here, as F is of degree 8 and A, B are of degree 7, F^+ is same as F (11BH) and its lower significant Byte (lsB) forms F* (1BH). The results for the modified algorithm

Algorithm 1: Proposed GFMUL Algorithm

Input: A,B : N bits, Field Polynomial (F): N+1 bits
Output: C : N bits

STEP 1: P : [P1:P0] = carryless multiplication(A,B)
STEP 2: Q : [Q1:Q0] = carryless multiplication(P1,F^+)
where F^+ = result of X^{2N} divided by F
STEP 3: R : [R1:R0] = carryless multiplication(Q1,F*)
where F*= lower order N bits of F
STEP 4: Output : C = R0 xor P0

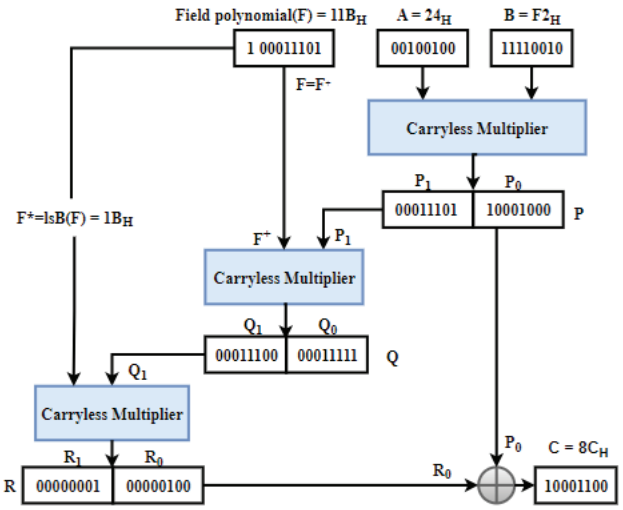


Figure 5: An eight bit *GFMUL* example using the modified algorithm

are verified for all possible 8-bit input combinations against the standard algorithm. The modified *GFMUL* algorithm when compiled for GF(2^8) using *RISC-V GCC compiler*, generates the instruction disassembly which gives three instances of CLMUL instruction as expected. This algorithm can be further extended to higher polynomial degrees but gets limited with the width of the RISC-V register.

C. Application mapping and evaluation on SWERV-EL2

The proposed *GFMUL* algorithm using CLMUL instruction can be used to accelerate various application that use GF(2^8) arithmetic operations. AES is the GF(2^8) based cryptographic application selected for the performance evaluation and is applied to multiple encryption schemes of AES like CBC, ECB and CTR for key-lengths 128, 192 and 256. SweRV-EL2 known as VeeR-EL2 at the time of writing is an open-source RISC-V core from Western Digital with support for the ratified bit manipulation extensions (Zba/b/c/s). Thus, by default, it has the hardware for carryless multiplication in the execution pipeline and deprecates any extra hardware for executing the proposed algorithm.

The notion of the work is to reduce the cycles of execution taken by the *GFMUL* in cryptographic applications using the CLMUL instructions. The evaluation strategy hence involves a standard variant that is compiled and executed using basic RISC-V instructions and a proposed variant, that requires the Zbc instruction support, that is available in SWERV-EL2.The AES algorithm in the repository mentioned in Table I has been modified to accommodate these two variants (Standard and Proposed) for three encryption Schemes (CBC, CTR and ECB) using multiple key lengths. Table I shows the record of tools, its configurations and the repositories used in this work. The simulation environment consists of a verilated model using SWERV-EL2 core connected to a virtual memory via AXI bus as shown in Fig. 6. AES is modified to have versions with and without CLMUL instruction based GF arithmetic, compiled by the RISC-V GNU toolchain and the resulting binary is fed into virtual memory for simulation. The verilated top module is modified to display the number of cycles executed and the total instructions retired in the console output for each run. This is accomplished by reading the values of MCYCLE and MINSRET controls status registers (CSR) of RISC-V ISA using the CSR read instruction. The counts corresponding to the standard and the proposed CLMUL implementation are extracted to calculate the percentage reduction in cycles executed. All the steps followed for the simulation here are available in the open-source SweRV repository [9].

TABLE I. RECORD OF TOOLS AND REPOSITORIES USED FOR EVALUATION

Core	
SWERV-EL2	Core™ 1.3: RISC-V from Western Digital Target= typical_pd [No ICCm, AXI4 bus interface] Frequency= 25 Mhz
Synthesis and Implementation	
Vivado	Version=v2023.2 (64-bit)
Target board	Nexys 4 DDR; Part: XC7A100T-1CSG324C
Simulation	
Verilator	v4.212
RISC-V GNU Toolcahin	32 bit GCC: version:13.2.0(gc891d8dc23e) Compiler flags: -mabi=ilp32 -march=rv32imac_zbc_zicsr -O3 -fomit-frame-pointer -fPIC -no-pie
Algorithms	
AES	Repository: https://github.com/kokke/tiny-AES-c

TABLE II. CLOCK CYCLES AND INSTRUCTION COUNTS FOR GFMUL

Function	Cycles		Instructions		Reduction (%)	
	Standard	Proposed	Standard	Proposed	Cycle	Instr
GFMUL						
GF(2^4)	128	57	64	21	55.47	67.19
GF(2^8)	175	63	106	21	64	80.19
GF(2^{16})	344	65	240	26	81.1	89.17
GFMUL_INV						
GF(2^4)	378	131	296	85	65.34	71.28
GF(2^8)	1250	274	1126	223	78.08	80.2
GF(2^{16})	6356	631	5928	573	90.07	90.33

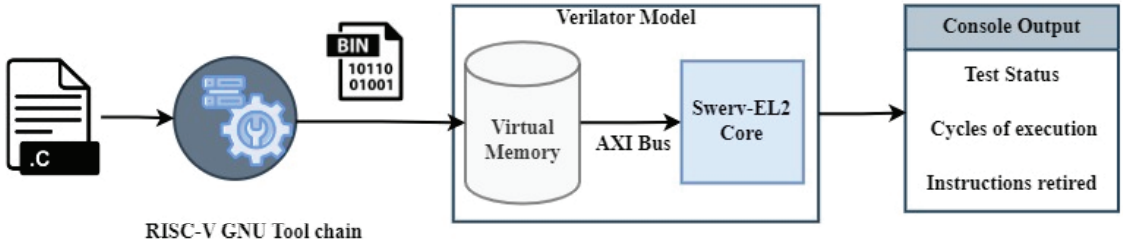


Figure 6: Verilated model of SWERV-EL2 for simulation of GF operations in AES encryption

III. RESULTS AND DISCUSSIONS

This section evaluates the performance of the proposed modification in Galois multiplication in terms of execution cycles and retired instructions. Though this work lays focus on GF(2^8) owing to the degree of field in AES , the proposed algorithm is scalable to other polynomial degrees as well. Table II shows the count of execution cycles and instructions returned count of *GFMUL* and *GFMUL_INV* for degrees 4, 8, and 16. It shows that the percentage of reduction in cycle count increases from 55% to 81% as the degree of polynomial or the number of bits in the arithmetic logic increases from 4 to 16. This trend makes it evident that the performance of the modified algorithm improves with higher degrees of Galois field arithmetic. Similarly, the inversion algorithm is modified using the proposed *GFMUL* and examined for multiple degrees. The results show cycle count reduction of 65%, 78% and 90% for degrees 4, 8 and 16 respectively over the standard execution.

CBC, ECB and CTR schemes of AES exhibits similar results of more than 75% reduction in execution cycles for encryption and decryption algorithms as portrayed by Fig. 7. The results marks a comparable performance with respect to the state-of-the-art (SOTA) solution with a small depreciation in cycle reduction [6]. This stems from the absence of dedicated hardware for polynomial reduction using custom instructions. In SOTA, the polynomial reduction logic as such is substituted with FFRED instruction (Custom), which incurs an area of 7% equivalent to 268 LUT's on NEXYS DDR4 FPGA board. On contrary, the proposed approach does not add any additional hardware and uses the already existing carryless multiplier twice for polynomial reduction. These additional instances of CLMUL instructions in the algorithm account for the aforementioned depreciation in the perfomance.

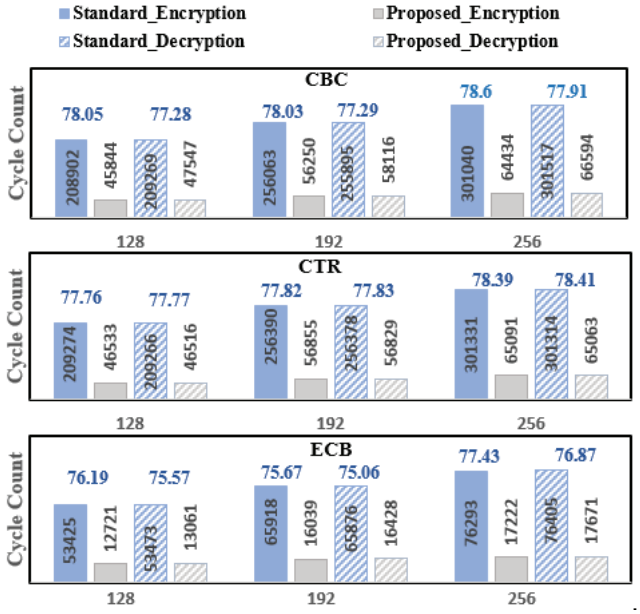


Figure 7: Plot of the Cycles counts for CBC, CTR and ECB schemes of AES for Standard and Proposed execution for various keylengths. The numbers on top of the bars indicate the percentage reduction in cycle count achieved by the proposed method corresponding to the absolute bar values.

Nevertheless, the proposed variant shows about 4.5x compared to the standard approach for AES, without any hardware modification to the existing core and also eliminates the need for compiler and ISA modifications and associated hardware overheads for custom instructions.

IV. CONCLUSION

The Security concerns on edge devices, initiate tailored optimizations of general-purpose CPUs for accelerated cryptographic applications. Galois field arithmetic - GF(2^m), is a critical step in pre and post-quantum encryption and error-correcting codes. Carryless multiplication enhances GF arithmetic, making the Zbc extension of RISC-V BMI valuable for accelerating GF-based cryptographic algorithms. This work proposes an approach to modify the Galois multiplication and related arithmetic using carryless multiplication to map the same to RISC-V ISA. Our analysis using an open-source RISC-V core revealed a significant decrease in clock cycles of over 75% for various AES schemes with a 25% decrease in static code size without the need for any additional hardware for custom instructions and related compiler overheads. Future work aims to explore carryless multiplier designs, incorporate them into an in-house RISC-V core, and evaluate more real-world applications like McEliece and Reed-Solomn.

ACKNOWLEDGMENT

This research is supported in part by C2S Project scheme of Ministry of Electronics and Information Technology (MeitY), Government of India vide project grant EE-9/2/2021-R&D-E.

REFERENCES

[1] L. Tan and N. Wang, "Future internet: The internet of things," in 2010 3rd international conference on advanced computer theory and engineering (ICACTE), vol. 5, pp. V5–376, IEEE, 2010.

[2] T. Huang, W. Yang, J. Wu, J. Ma, X. Zhang, and D. Zhang, "A survey on green 6g network: Architecture and technologies," IEEE access, vol. 7, pp. 175758–175768, 2019.

[3] R. International, "Risc-v bit-manipulation isa-extensions." https://github.com/riscv/riscv-bitmanip/blob/main/bitmanip/bitmanip.adoc, 2022.

[4] Y. Chen, S. Lu, C. Fu, D. Blaauw, R. Dreslinski Jr, T. Mudge, and H.-S. Kim, "A programmable galois field processor for the internet of things," in Proceedings of the 44th Annual International Symposium on Computer Architecture, pp. 55–68, 2017.

[5] W.-M. Lim and M. Benaissa, "Design space exploration of a hardware-software co-designed gf (2m) galois field processor for forward error correction and cryptography," in Proceedings of the 1st IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis, pp. 53–58, 2003.

[6] Y.-M. Kuo, F. Garcia-Herrero, O. Ruano, and J. A. Maestro, "Riscv galois field isa extension for non-binary error-correction codes and classical and post-quantum cryptography," IEEE Transactions on Computers, vol. 72, no. 3, pp. 682–692, 2023.

[7] X. Zhang, VLSI architectures for modern error-correcting codes. Crc Press, 2017.

[8] S. Gueron and M. Kounavis, "Efficient implementation of the galois counter mode using a carry-less multiplier and a fast reduction algorithm," Information Processing Letters, vol. 110, no. 14-15, pp. 549‑553, 2010.

[9] W. D. Corporation, "Risc-v swerv-el2 github repository." https://github.com/chipsalliance/Cores-SweRV-EL2, 2020.

opoSoM: A Modular Measurement Platform for Dynamic Power Consumption of SoCs

Kristóf Kanics, Meinhard Kissich
Graz University of Technology
Graz, Austria
{kristof.kanics,
meinhard.kissich}@tugraz.at

Gerhard Wirrer
Infineon Technologies AG
Neubiberg, Germany
Gerhard.Wirrer@infineon.com

Tobias Scheipel, Marcel Baunach
Graz University of Technology
Graz, Austria
{tobias.scheipel,
baunach}@tugraz.at

Abstract—**Software can have a significant impact on the electrical characteristics of the executing integrated circuit. The analysis of minor current consumption changes in a System-on-Chip reveals details about the executed instructions or the hardware's internal logic, potentially exposing sensible information. Despite careful design, glitches pose a further challenge that needs to be handled at hardware and software level.**

This paper introduces the concept of the open-source, modular *opoSoM* measurement platform that captures the dynamic power characteristics of System-on-Chips featuring external and on-chip measurement techniques. Due to the configurable measurement range and synchronous sampling at up to 250 MS/s, the platform provides valuable measurement data for investigating countermeasures against side-channel attacks and optimizing hardware and software towards lower dynamic power consumption.

***Index Terms*—System-on-Chip, measurement, supply voltage, power consumption, side-channel attacks**

I. Introduction

The externally observable dynamic power consumption may reveal valuable and sensitive information about the behavior of a processor core. Kocher et al. [1] showed very early that it is possible to extract information (e.g., cryptographic keys) from electronic devices based on power consumption measurements. Voltage sensors can be integrated into the implementation of a digital design [2] for Field Programmable Gate Arrays (FPGAs). Existing on-chip voltage sensors use the physical effect that the signal propagation depends on the device's supply voltage: Time-to-Digital Converters (TDCs) sample an input signal in a delay line with configurable length to probe how fast a signal transition propagates. Ring Oscillators (ROs) change their oscillation frequency depending on the delay in its individual stages, converting supply voltage transients into frequency deviations. Integrating on-chip voltage sensors into a device poses a significant threat when, e.g., an FPGA is shared among multiple users in cloud systems [3]. While service providers try to counteract attacks, by e.g., prohibiting combinatorial loops [4], such ROs have been used to crash an FPGA due to excessive power consumption by using only 12% of its lookup table resources [5]. As the power estimation of computer-aided design tools can significantly underestimate the power consumption [6] of such structures, a versatile and highly accurate research platform is required to detect and analyze malicious circuits.

Independent of their potentially malicious nature, fast transients in the dynamic power consumption (e.g., load jumps caused by unwanted signal activity) can compromise the device's own operation. The voltage on the supply pin might not be able to follow a rapid change in the current consumption due to the supply trace's parasitic inductance. Such supply anomalies can potentially cause device malfunctions or resets [7]. Optimizing the dynamic power consumption by e.g., minimizing the switching activity [8], is crucial when designing for side-channel attack resilience.

In this paper, we present a modular measurement platform that consists of a carrier board and pluggable modules that carry the System-on-Chips (SoCs) under test: *opo**SoM*** (optimizing power consumption for System on Modules) allows for cycle-accurate on-chip and external voltage measurements to analyze the dynamic power consumption of SoCs. It is designed to

- map voltage transients to instructions,
- compare voltage measurements that correspond to code sequences running on different implementations of the same architecture,
- investigate how the dynamic power consumption can be reduced by optimizing the hardware architecture, its implementation, and the software as a whole, and
- elaborate on the effectiveness of countermeasures against side-channel attacks.

When writing this paper, the first pluggable module is available, and the carrier board and further modules are under development.

The paper is structured as follows: **Section II** presents related work. **Section III** and **Section IV** describe the architecture of the carrier board and the necessary components of the modules, respectively. The suggested measurement techniques are presented in **Section V**, and **Section VI** closes with an outlook to future work.

II. Related Work

Fast voltage transients in an SoC have been investigated in several works, focusing on the current consumption of entire processes [9], [10], often using the built-in Analog-to-Digital Converters (ADCs) of the device under test [9], [11]. ChipWhisperer [12] offers a hardware and software environment to emulate and analyze side-channel attacks on

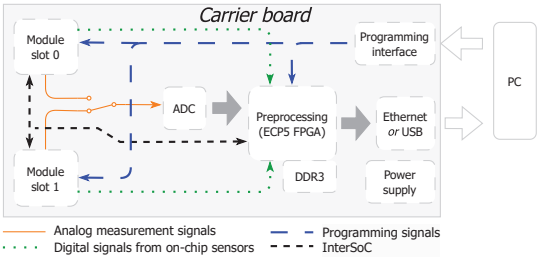


Fig. 1: Architecture of the *opo**SoM*** measurement platform.

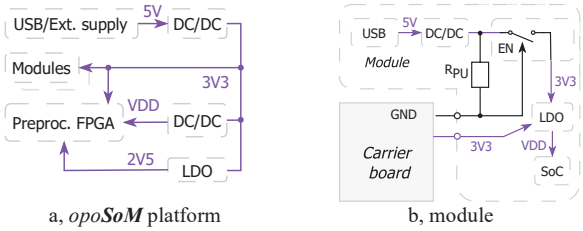
different circuit boards, using an external resistor as a current sensor. Voltage transients are often not easy to measure outside of an SoC since the on-chip and external decoupling capacitors flatten the transients. Therefore, *opo**SoM*** extends ChipWhisperer's approach by additionally applying on-chip sensor topologies with high voltage resolution and nanoscale range [13], [14], [15]. In contrast to the previously mentioned works [9], [10], [11], we are interested in how single instructions or short instruction sequences in Application Specific Integrated Circuit (ASIC) or FPGA-based soft processors can cause significant changes in the power consumption. We aim to perform cycle-accurate on-chip and external measurements to map voltage transients to executed instructions. Guiding the place and route tools enables to place sensors at different locations in the SoC to gather information about where exactly the voltage transients originate from or how different implementations of an architecture influence the measurement results. The measurement data is preprocessed in a separate device to avoid influencing the device under test itself. Localizing power peak sources at software and hardware levels helps to implement appropriate countermeasures to prevent side-channel attacks and optimize the system to reduce voltage drops that can lead to device malfunctions.

III. Carrier Board Architecture

The carrier board (cf. Fig. 1) has two slots for pluggable System-on-Modules (SoMs) that are built around the SoCs under test (cf. Section IV.). The measurement components are placed on the carrier board to provide an identical measurement environment for all SoCs. A Lattice ECP5 FPGA [16] is used to buffer and preprocess the measured sensor data. The data is then forwarded to a PC through Ethernet or USB. By generating a clock signal using the on-board ECP5 or an external clock source, the ADC and the SoC under test on a module can be clocked synchronously. Synchronous clocking is essential to keep track of code execution and voltage transients simultaneously. The following sections describe the individual building blocks of the carrier board in more detail.

A. Power supply

Fig. 2a summarizes the power supply scheme. The carrier board can be powered either via USB or by an external power supply, in case USB is insufficient to operate the on-board components *and* the plugged modules. An on-board step-down DC/DC converter provides the necessary 3V3 I/O voltage for the preprocessing FPGA and the plugged modules. The 1V2 core voltage of the preprocessing FPGA is generated by a second



a, *opo**SoM*** platform b, module

Fig. 2: Power supply scheme.

step-down converter. The 2V5 voltage level with low current flow is provided by a Low Drop-Out (LDO) regulator.

B. Programming interface

To avoid using multiple USB cables, the on-board FPGA and the SoCs on the modules can be programmed via an on-board USB hub and a subsequent JTAG converter. Configuration data are stored in each SoC's corresponding flash memory.

C. Data preprocessing and PC interface

For data preprocessing (e.g., filtering, compression) and buffering, the board features 1 GiB of DDR3 memory. The preprocessed data can be transmitted to a PC via Gigabit Ethernet or using a 5 Gbps USB peripheral controller.

D. Analog-to-digital converter

The analog signals from the modules are fed into a two-channel differential ADC. The two ADC channels are driven by two fully differential amplifiers. The amplifier circuits provide linearity in a wider frequency range compared to baluns. For conversion, an ADS4229 [17] converter was selected based on its 250 MS/s maximum sample rate, 550 MHz analog input bandwidth, and moderate price. To be compliant with the Nyquist-Shannon sampling theorem, the frequency of the input signal must not be higher than 125 MHz. An anti-aliasing filter ensures the band-limitation of the input signal.

IV. Module Architecture

Each module features an SoC that contains an arbitrary analog or digital design to be evaluated. The effects on the dynamic power consumption of an instruction sequence are measured utilizing the on-chip voltage sensors and the on-board passive components. The measurement signals are routed directly to the module's interface connectors, and are not processed by the SoC under test to avoid negative effects on the signal quality. The modules can be plugged onto the carrier board described in the previous section. As a second option, the modules are designed to be operable in stand-alone mode when no measurements are performed, allowing to work on different modules simultaneously.

When using the module plugged onto the carrier board, it is supplied via the carrier board's 3V3 power rail, and the on-module step-down converter is disabled to eliminate its switching noise. The ground potential of the carrier board is used as a disable signal for a power switch (cf. Fig. 2b), preventing reverse current flow into the output of the step-down converter of the module. In the absence of the carrier board, R_{PU} pulls the enable pin of the power switch high to keep the switch on. The

core voltage of the SoC under test and other necessary voltage levels are generated directly on the module. For this purpose, LDOs are preferred over switched-mode regulators to produce less noise. The USB signals of the module are accessible by the carrier board, which allows to program the module via the carrier board's USB connector and hub. The modules connect to the carrier board via a specifically tailored interface (InterSoC) and using Serializer/Deserializer (SerDes) if available on the SoC under test. SerDes signals are directly accessible on the module via four SMA connectors as the example design shows (Fig. 3).

In stand-alone mode, the modules are supplied via USB and the on-board voltage regulators (cf. Fig. 2b). The modules can be programmed via the same USB connector and an on-module USB-to-JTAG converter.

V. Measurement Concept

Fig. 4 sketches the measurement concept of the *opo**SoM*** platform. To capture fast transients of the core supply, techniques *A.* to *C.* are applied:

A. Voltage drop on current-sense resistor

The voltage drop on a current-sense resistor R_S is proportional to the current consumption *I*. Small (10 nF to 100 nF) noise filtering capacitors placed close to the device pin are not depicted in Fig. 4. Depending on the SoC under test, they may or may not be necessary for a normal SoC operation. On each module, we populate as few of them as required to keep the device operational while not absorbing the transients. Decoupling capacitors C_{dec} (100 nF to 10 µF) are placed in front of R_S so that the current flowing from them into the device pin can be captured. With fast signal changes, package parasitics become more significant. The capacitor C_{comp} is placed in parallel to R_S to compensate for the parasitic inductance of R_S. For evaluating the measurements with an external oscilloscope and for calculating the compensation afterwards, the potential difference on R_S is accessible on SMA connectors. To the measurement signals on R_S, an anti-aliasing (AA) filter is applied, and the signals are amplified on the carrier board before feeding them into the ADC.

Current sensing on a serial resistor is a common measurement technique [10] but it requires a current-sense resistor (and eventually a compensation capacitor) as close as possible to the SoC's supply pins. The optimal position is under the device, on the back of the circuit board, which is already a component-dense area due to decoupling capacitors. The value of R_S has to be chosen based on the current consumption of the SoC under test. The higher the resistor value and the current consumption, the larger the induced voltage drop and therefore the easier it is to capture. On the other hand, a high voltage drop

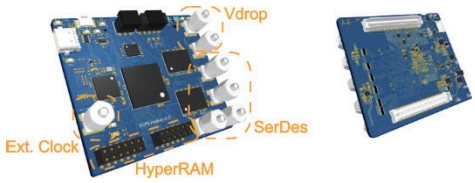


Fig. 3: 3D view of an example module with an ECP5 FPGA.

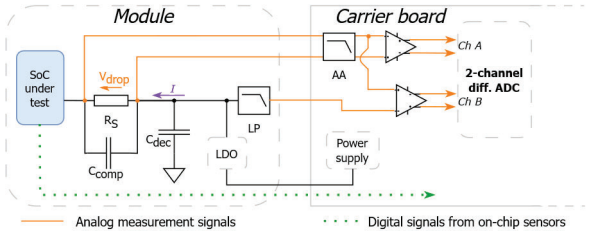


Fig. 4: Measurement concept.

can reduce the supply voltage on the device pin below the device's specification.

B. Direct voltage measurement

In contrast to the current-sense method, this technique is prone to noise coupling and ground bouncing. Short, impedance-controlled traces mitigate these issues. High resolution can be achieved by reducing the measured voltage range to the range of interest using the voltage regulator's low-pass (LP) filtered output as a potential reference. Single-ended voltage measurement does not require additional components, and a higher current consumption does not lead to a more difficult transient capture. If the evaluation of this measurement technique results in a comparable accuracy as the current-sense technique, further modules can be designed implementing only this technique, reducing manufacturing costs. The accuracy of this technique is high due to the differential signaling and is limited by package and trace parasitics.

C. Measurement with on-chip sensors

On-chip voltage level sensors (TDCs or ROs) can be implemented close to the source of transients, directly inside a target FPGA, to provide the most accurate measurement data among techniques *A.* to *C.* The sensors' nanoscale resolution allows the detection of transients even above the device's operation frequency due to the FPGA's internal power delivery network parasitics [15]. The carrier board receives signals from two 10-bit on-chip sensors of each plugged module. Since these sensors are directly implemented in the device under test, this measurement technique is not applicable for hard cores (e.g., microcontrollers) unless the sensors are already part of the implemented design. The on-chip sensors' resolution depends on their implementation (e.g., sampling rate).

D. Additional considerations

The carrier board can accommodate two pluggable SoMs with one SoC under test on each of them. The analog interfaces are multiplexed in a way that the ADC can process two arbitrary analog interfaces coming from any of the modules simultaneously. The external techniques' measurement range strongly depends on the static *and* dynamic current consumption of the SoC under test, and it has to be calibrated for each SoC under test by selecting an appropriate R_S. To log the temperature while performing measurements, the carrier board can read the module's temperature using external temperature sensors attached onto the SoC under test.

Synchronizing the ADC's sampling clock and the device's clock allows for the detection of transients that occur aligned with the clock edge. Without synchronous sampling, the

detection requires a sampling rate far over the device's operation frequency. ChipWhisperer [12] shows that the results of synchronous sampling at 96 MS/s are comparable to those of an asynchronous sampling at 2 GS/s. With 250 MS/s synchronous sampling, our platform is designed to detect voltage transients on devices running at maximum 250 MHz. Synchronous sampling also provides a reliable timestamp that allows for an efficient mapping of voltage transients to executed instructions.

VI. Conclusion and Future Work

In this paper, we presented the architecture of a modular measurement platform to track the dynamic power consumption of a wide range of SoCs in the same measurement environment. The external and on-chip measurement techniques provide a flexible measurement range for devices running at up to 250 MHz. The requirements for further modules with new SoCs to be evaluated and the corresponding interfaces between a module and the carrier board are described.

On each module, a calibration process needs to be performed: We have to determine the values of the noise filtering capacitors on the SoC's supply pins and the value of the current-sense resistor R_S experimentally. Current-sense and direct voltage measurements need to be compared using an external oscilloscope to cancel out package parasitics for R_S. If the comparison yields a significant difference between the two measurement techniques, the worse-performing technique can be optimized out for further modules in order to reduce manufacturing costs.

Future research demands a measurement dataset that we can obtain via running benchmark code on an open architecture that we have full control of. Suitable candidates are RISC-V soft cores due to their growing popularity and a number of existing implementations (e.g., CV32E40P [18], VexRiscv [19], FazyRV [20]). These first datasets help to identify which instructions or instruction sequences influence the dynamic power characteristics the most. Based on the gathered knowledge, we aim to develop software techniques to detect ambiguous instruction sequences at compile time and improve the digital logic to mitigate the effect of specific instructions (e.g., via adjusting the soft core's pipeline [21]). We are also interested in how the voltage transients differ from core to core, especially from a soft core to a hard core of the same architecture implementation. Another interesting research question is how these transients can be attenuated by applying different place-and-route techniques. Finally, we aim to explore other optimization strategies to mitigate side-channel attacks originating both within and outside of an SoC.

The source files of the *opo**SoM*** platform are available open source on *https://github.com/EAS-DSD/opoSoM*.

Acknowledgment

This work has been supported by the FFG under contract No.881844: "Pro2Future" (Products and Production Systems of the Future), Graz University of Technology (TU Graz), and Infineon Technologies AG in the joint research project CompEAS.

References

[1] P. Kocher et al., "Differential Power Analysis," Springer Berlin Heidelberg, 1999.

[2] K. M. Zick et al., "Sensing nanosecond-scale voltage attacks and natural transients in FPGAs," in *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, Monterey, 2013.

[3] O. Glamočanin et al., "Are Cloud FPGAs Really Vulnerable to Power Analysis Attacks?," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2020.

[4] aws/aws-fpga, *AWS EC2 FPGA HDK+SDK Errata, GitHub,* 2021. (accessed 2024-04-26).[Online]. Available: https://github.com/aws/aws-fpga/blob/master/ERRATA.md

[5] D. R. E. Gnad et al., "Voltage drop-based fault attacks on FPGAs using valid bitstreams," in *2017 27th International Conference on Field Programmable Logic and Applications (FPL)*, 2017.

[6] K. Matas et al., "Power-hammering through Glitch Amplification – Attacks and Mitigation," in *IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, 2020.

[7] A. Boutros et al., *Neighbors From Hell: Voltage Attacks Against Deep Learning Accelerators on Multi-Tenant FPGAs,* 2020.

[8] M. Kubica et al., "Logic Synthesis Strategy Oriented to Low Power Optimization," *Applied Sciences,* vol. 11, 2021.

[9] Feinstein, David Y. et al., "System-on-Chip Power Consumption Refinement and Analysis," in *6th IEEE Dallas Circuits and Systems Workshop on System-on-Chip*, 2007.

[10] T. Doebbert et al., "Precision measurement of the application-dependent current consumption of a wireless transceiver chip in the time and frequency domain," *Journal of Sensors and Sensor Systems,* 2022.

[11] Nakutis, Žilvinas, "A consumption current measurement approach for FPGA based embedded systems," in *IEEE International Instrumentation and Measurement Technology Conference Proceedings*, 2012.

[12] O'Flynn, Colin and Chen, Zhizhang, "ChipWhisperer: An Open-Source Platform for Hardware Embedded Security Research," in *Lecture Notes in Computer Science*, Springer International Publishing, 2014.

[13] X. Xu et al., "A High-Resolution Nanosecond-Scale On-Chip Voltage Sensor for FPGA Applications," *IEEE Transactions on Instrumentation and Measurement,* vol. 72, 2023.

[14] S. Moini et al., "Understanding and Comparing the Capabilities of On-Chip Voltage Sensors against Remote Power Attacks on FPGAs," in *2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2020.

[15] J. Gravellier et al., "High-Speed Ring Oscillator Based Sensors for Remote Side-Channel Attacks on FPGAs," *2019 International Conf. on ReConFigurable Computing and FPGAs (ReConFig),* 2019.

[16] Lattice Semiconductor, *ECP5 and ECP5-5G Family, Datasheet*

[17] Texas Instruments, *ADS4229 Dual-Channel, 12-Bit, 250-MS/S Ultralow-Power ADC,* June 2011 – Revised May 2015.

[18] OpenHW Group, *cv32e40p,* GitHub, (accessed 2024-05-23). [Online]. Available: https://github.com/openhwgroup/cv32e40p.

[19] C. Papon, *VexRiscv,* 2018., GitHub, 2018, (accessed 2024-05-16). [Online]. Available: https://github.com/SpinalHDL/VexRiscv

[20] M. Kissich and M. Baunach, "FazyRV: Closing the Gap between 32-Bit and Bit-Serial RISC-V Cores with a Scalable Implementation," in *Proc. of the 21st ACM International Conference on Computing Frontiers (CF '24)*, 2024.

[21] T. Scheipel et al., "moreMCU: A Runtime-reconfigurable RISC-V Platform for Sustainable Embedded Systems," in *2022 25th Euromicro Conference on Digital System Design (DSD)*.

On Exploiting PSOP Decomposition for Quantum Synthesis

Anna Bernasconi
Dipartimento di Informatica
Università di Pisa, Italy
anna.bernasconi@unipi.it

Valentina Ciriani, Gianmarco Cuciniello, Asma Taheri Monfared
Dipartimento di Informatica
Università degli Studi di Milano, Italy
{valentina.ciriani, asma.taheri}@unimi.it

***Abstract*—The synthesis strategy for quantum oracles is based on a reversible logic synthesis and a quantum compilation step. In reversible logic synthesis it is important to obtain a compact reversible circuit in order to minimize the size of the final quantum circuit. Projected Sum Of Product, PSOP, decomposition is an EXOR based technique that can be applied to any Boolean function as a very fast pre-processing step for further minimizing the circuit area in standard logic synthesis. In this paper, we exploit PSOP decomposition in quantum synthesis. In particular, we describe a new technique for the quantum synthesis of PSOP decomposed functions. The experimental results validate the proposed pre-processing method in quantum synthesis, showing an interesting gain in area, within the same time limit.**

***Index Terms*—Circuit decomposition, reversible logic, quantum circuits**

I. Introduction

The recent technological improvement in quantum architectures has led to renewed and growing interest in quantum computing and in the design of secure cryptographic protocols. Therefore, the research in the field of quantum logic synthesis has attracted considerable new attention. In particular, many quantum algorithms, including Grover's search algorithm, usually require computing *oracles* [13], i.e., subroutines given as classical logic functions. The standard method for synthesizing quantum oracles generally consists of two steps: reversible logic synthesis and quantum compilation. This is due to the fact that, in general, the evolution of quantum systems is described by reversible unitary operators.

Recently, new techniques and tools have been proposed for quantum synthesis [10], [14], [18]. Moreover, several pre-processing methods have been proposed for further enabling reversible synthesis and quantum compilation. Indeed, such methods exploit structural regularities of the input function [1], [4]. Therefore, just "regular functions" can benefit from this pre-processing strategies (i.e., autosymmetric and D-Reducible functions). In this paper we propose a new pre-processing strategy that have the following characteristics:

1) The method can be exploited for *any* Boolean function (not just regular ones);
2) The method consists in a decomposition of the original function f and a re-composition after the quantum compilation;
3) The decomposition procedure has a linear-time complexity and the re-composition phase is constant in time.

These characteristics make the proposed method a possibly useful and fast pre-processing strategy before reversible synthesis and quantum compilation. The method is based on a structural non-disjoint decomposition of the input function called *Projected Sum of Products* (PSOP), which is an EXOR based non-disjoint decomposition. PSOP forms are a generalization of the standard Shannon decomposition. We consider this particular decomposition since it is EXOR based (i.e., the reconstruction has a very low quantum cost), and the decomposition process is vary fast (i.e., exhibits a linear time complexity). Figure 3 shows the standard quantum synthesis methods and the new proposed one.

The theoretical part of this paper describes the proposed pre-processing method and the reconstruction strategy. In particular, we show that after the PSOP decomposition and the quantum synthesis of the components, it is possible to reconstruct the original function f adding a constant number of Toffoli gates (2 or 3) that correspond to 8 or 12 T-gates [10].

Finally, we test the quantum synthesis of PSOP decomposed functions and we compare the results with the ones obtained by the classical quantum compilation. The experimental results show that the proposed pre-processing phase gives better results for the 61% of the benchmarks with an average gain of about 22% in terms of T-gates, using the XAG-based quantum compilation described in [10].

The paper is organized as follows: the preliminary concepts of projections of functions, PSOP forms, along with reversible circuits and quantum compilations, are outlined in Section II. Our proposed new pre-processing strategy for quantum reversible synthesis is presented in Section III. We discuss the evaluation of the proposed method and report our experiments on a set of benchmarks in Section IV. Finally, the conclusion of this work is given in Section V.

II. Preliminaries

A. Projections of Functions

PSOP decomposition, originally introduced for logic synthesis in CMOS technology, can be exploited to enable quantum compilation, as proposed and discussed in Section III. In this preliminary section, we review some basic concepts of Boolean space partitioning and we present the projections exploited in this particular decomposition.

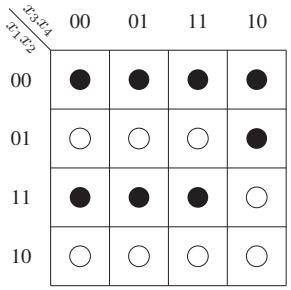


Fig. 1. The Boolean space $\{0,1\}^4$ partitioned into the two distinct sets $B_{x_1=p}$ (black points) and $B_{x_1\neq p}$ (white points), with $p = x_2 \cdot (\overline{x}_3 + x_4)$.

Let us consider the Boolean space with variables $x_1, x_2, \ldots, x_n$ and let x_i be one of these variables. Let p represent a function over a subset of variables excluding x_i, denoted by $\{x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n\}$. The Boolean space $B = \{0,1\}^n$ can be partitioned into two distinct subsets: the set $B_{x_i=p}$, where x_i equals the function p, and the set $B_{x_i\neq p}$, where x_i is not equal to the function p. Formally, we have $B_{x_i=p} = \{(v_1, \ldots, v_n) \in \{0,1\}^n \mid v_i = p(v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n)\}$, and $B_{x_i\neq p} = \{(v_1, \ldots, v_n) \in \{0,1\}^n \mid v_i \neq p(v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n)\}$, respectively.

Example 1: Let us consider the function $p = x_2 \cdot (\overline{x}_3 + x_4)$ and the variable x_1 in the Boolean space $\{0,1\}^4$, which can be partitioned into two sets. The first set consists the subspace where $x_1 = p$ and the second one consists of the subspace where $x_1 \neq p$. Figure 1 depicts a Karnaugh map illustrating these two sets, the black points correspond to $B_{x_i=p} = \{0000, 0001, 0010, 0011, 0110, 1100, 1101, 1111\}$, while the white points correspond to $B_{x_i\neq p} = \{0100, 0101, 0111, 1000, 1001, 1010, 1011, 1110\}$.

It is noteworthy that the Boolean space divides evenly into these two sets, demonstrating the following general property [3]: When x_i is a Boolean variable and p is a function represented as $p : \{0,1\}^{n-1} \rightarrow \{0,1\}$ on variables $\{x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n\}$, the sets $B_{x_i=p}$ and $B_{x_i\neq p}$ are such that:

1) $B_{x_i=p} \cup B_{x_i\neq p} = \{0,1\}^n$
2) $|B_{x_i=p}| = |B_{x_i\neq p}| = 2^{n-1}$
3) $B_{x_i=p} \cap B_{x_i\neq p} = \emptyset$

In the Boolean space $B = \{0,1\}^n$, the simplest partitioning occurs when p equals 1 (or 0). In this scenario, $B_{x_i=1}$ and $B_{x_i\neq 1}$ represent the subspaces of B where x_i equals 1 and x_i equals 0, respectively. The characteristic functions x_i and $\overline{x}_i$ can represent these subspaces [3]. Observe that $B_{x_i=1}$ and $B_{x_i\neq 1}$ represent the basic partitions of the *classical Shannon decomposition*

$$f = x_i f|_{x_i=1} + \overline{x}_i f|_{x_i\neq 1}$$

where $f|_{x_i=1}$ and $f|_{x_i\neq 1}$ denote the two cofactors obtained from f replacing x_i with 1 and 0, respectively.

In [6], [8], a Boolean functional decomposition method is presented, generalizing the classical Shannon decomposition: $f = (\overline{x}_i \oplus p) f|_{x_i=p} + (x_i \oplus p) f|_{x_i\neq p}$. This method projects the function f onto the two complementary subsets $B_{x_i=p}$ and $B_{x_i\neq p}$ of the Boolean space $B = \{0,1\}^n$. The expressions $\overline{x}_i \oplus p$ and $x_i \oplus p$ denote the characteristic functions of $B_{x_i=p}$ and $B_{x_i\neq p}$, respectively. It should be noted that the Shannon decomposition is a specific case of this partition, where $x_i \oplus 1$ equals $\overline{x}_i$ and $\overline{x}_i \oplus 1$ equals the variable x_i.

Example 2: Figure 2 illustrates the Karnaugh map of the function $f = \{0000, 0001, 0111, 1011, 1100, 1101, 1111\}$, in the right side. Consider $B_{x_1=p}$ and $B_{x_1\neq p}$ as two projecting sets with $p = x_2 \cdot (\overline{x}_3 + x_4)$. As shown in the left side of this figure, the function f can be projected onto the two spaces $B_{x_1=p}$ and $B_{x_1\neq p}$. The resulting projected functions depend on x_2, x_3, x_4 and can be represented by $f|_{x_1=p} = \{000, 001, 100, 101, 111\}$ and $f|_{x_1\neq p} = \{011, 111\}$, respectively.

It is important to point out that *Hamming distances* can change when points are projected onto different subspaces. As a result, they may be combined into larger terms and may reveal new implications not present in the original function. For instance, consider the two points in Example 2 represented by the minterms $\overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4$ and $x_1x_2\overline{x}_3\overline{x}_4$. We can notice that their Hamming distance is equal to 2. After the projection onto the space where $x_1 \neq p$, their Hamming distance is reduced to 1 as they become more similar; thus it is possible to merge them into the larger product $(\overline{x}_3)$ in $f|_{x_1\neq p}$.

B. PSOP forms

The PSOP decomposition and synthesis approach involves constructing a circuit for f by exploiting p and the two projected functions $f|_{x_i=p}$ and $f|_{x_i\neq p}$. The synthesis for the projected functions is simpler, compared to f, as they involve at least one fewer variable, leading often to more compact circuits. The functions p, $f|_{x_i=p}$ and $f|_{x_i\neq p}$ can be synthesized using any logic minimization methodologies, including SOP synthesis and also *quantum* synthesis. When represented as sums of products, they form the *Projected Sum of Products* form, abbreviated as PSOP(f) and defined as follows [3].

Definition 1: Let $f|_{x_i=p}$ and $f|_{x_i\neq p}$ indicate the projections of f onto $B_{x_i=p}$ and $B_{x_i\neq p}$, respectively. The *PSOP* of f with respect to p is expressed as

$$\text{PSOP}(f) = (\overline{x}_i \oplus p) f|_{x_1=p} + (x_i \oplus p) f|_{x_1\neq p}\,,$$

where p, $f|_{x_i=p}$, and $f|_{x_i\neq p}$ are expressed as SOP forms.

It is worth mentioning that in order to minimize the overall form, it is possible to further minimize the SOP for p as well as the two projected SOP forms $f|_{x_i=p}$ and $f|_{x_i\neq p}$ after projection.

Definition 2: Let $\tilde{p}$ be a minimal SOP form for the function p and let the *minimal SOP expressions* for the projections of f onto the sets $B_{x_i=p}$ and $B_{x_i\neq p}$ be represented by $\tilde{f}|_{x_i=p}$ and $\tilde{f}|_{x_i\neq p}$, respectively. The *minimal PSOP* of f with respect to p is expressed as:

$$\text{PSOP}(f) = (\overline{x}_i \oplus \tilde{p}) \tilde{f}|_{x_1=p} + (x_i \oplus \tilde{p}) \tilde{f}|_{x_1\neq p}\,.$$

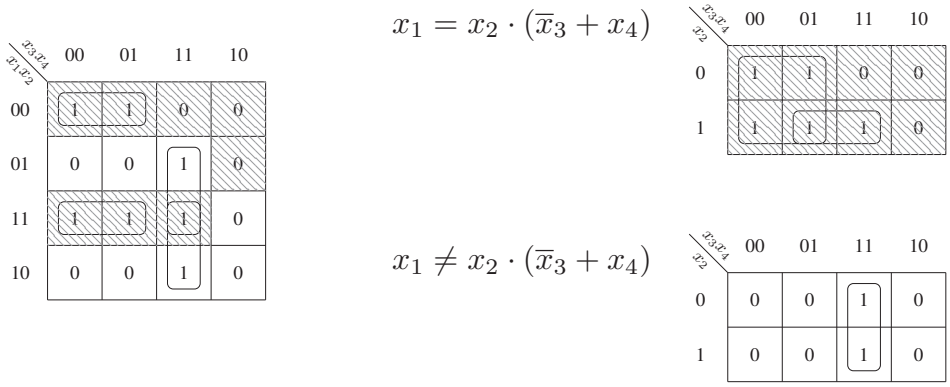


Fig. 2. Karnaugh maps of a function f (left) and its corresponding projections onto $f|_{x_1=p}$ and $f|_{x_1\neq p}$ (right), with $p = x_2 \cdot (\overline{x}_3 + x_4)$..

Example 3: Consider the function $f = \overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4 + \overline{x}_1\overline{x}_2\overline{x}_3x_4 + \overline{x}_1x_2x_3x_4 + x_1\overline{x}_2x_3x_4 + x_1x_2\overline{x}_3\overline{x}_4 + x_1x_2\overline{x}_3x_4 + x_1x_2x_3x_4$ described in Example 2. The minimal SOP form of this function is $\overline{x}_1\overline{x}_2\overline{x}_3 + x_2x_3x_4 + x_1x_2\overline{x}_3 + x_1x_3x_4$. As shown in Figure 2, the function f is projected onto the two sets. The minimal SOP forms for the sets $B_{x_1=p}$ and $B_{x_1\neq p}$, can be represented by $\tilde{f}_{x_1=p} = \overline{x}_3 + x_2x_4$ and $\tilde{f}|_{x_1\neq p} = x_3x_4$, respectively. As the minimal SOP form of p is $\tilde{p} = x_2\overline{x}_3 + x_2x_4$, the overall minimal PSOP form for f is then $(\overline{x}_1 \oplus (x_2\overline{x}_3 + x_2x_4))(\overline{x}_3 + x_2x_4) + (x_1 \oplus (x_2\overline{x}_3 + x_2x_4))(x_3x_4)$.

Another useful form is the *Pr-SOP* for the function f, which is also known as the *PSOP with remainder* [3]. It includes a *remainder*, containing all products in the SOP expression of f that intersect both projection sets. These products are called *crossing products*, whereas products entirely included in one of the two projection sets are called *non-crossing products*.

Definition 3: Let $f|_{x_i=p}$ and $f|_{x_i\neq p}$ denote the projections of all non-crossing products in a SOP representation of f, and let r denote the sum of all crossing products. The *Pr-SOP* of f with respect to p is expressed as:

$$\text{Pr-SOP}(f) = (\overline{x}_i \oplus p)f|_{x_i=p} + (x_i \oplus p)f|_{x_i\neq p} + r\,.$$

Minimizing all SOP expressions, we derive a *minimal PSOP with remainder*.

Definition 4: Let $\tilde{p}$ and $\tilde{r}$ be minimal SOPs form for the function p and the remainder r, respectively. Let the minimal SOP expressions for the projections of all non-crossing products of f onto $B_{x_i=p}$ and $B_{x_i\neq p}$ be represented by $\tilde{f}|_{x_i=p}$ and $\tilde{f}|_{x_i\neq p}$, respectively. The *minimal Pr-SOP* of f with respect to p is expressed as:

$$\text{Pr-SOP}(f) = (\overline{x}_i \oplus \tilde{p})\tilde{f}|_{x_1=p} + (x_i \oplus \tilde{p})\tilde{f}|_{x_1\neq p} + \tilde{r}\,.$$

Algorithms and heuristic methods for minimizing PSOP expressions have been proposed and analyzed in [3], [5].

C. Reversible Circuits and Quantum Compilation

Reversible circuits have one-to-one correspondence between their inputs and outputs, ensuring that the number of outputs is always equal to the number of inputs. In order to assess the efficiency of such circuits, metrics like *the number of ancilla inputs and garbage outputs* are crucial. The number of ancilla inputs refers to the number of additional input bits required to make logic gate irreversibles. The number of garbage outputs signifies the outputs generated to maintain one-to-one mappings but contain unimportant values. Decreasing these important features enhances the efficiency of designing reversible circuit.

Reversible circuits are normally based on *Mixed-polarity Multiple Control (MPMC) Toffoli gates*. All functions can be implemented with these reversible gates. In Figure 6, the realization of three *MPMC Toffoli gates* are illustrated, the notation $\oplus$ indicates the target line, while the notations $\bullet$ and $\circ$ denote positive and negative control connections, respectively. This gate is known as a *NOT gate* when there are no control connections. It is classified as a *Controlled-NOT (CNOT) gate* when there is only one positive connection, and as a *Multiple-Control Toffoli gate* when there are only positive connections [13].

The synthesis of quantum circuits begins with the design of reversible circuits. An additional quantum compilation step is then required to transform reversible circuits, which utilize gates like *MPMC Toffoli gates*, into quantum circuits with functionally equivalent gates.

This process involves decomposing each reversible gate into elementary quantum gates, based on *standard quantum gate libraries* [9], [12]. Adding a quantum compilation step to a reversible circuit with *MPMC Toffoli gates* can transform it into a functionally equivalent quantum circuit that can be implemented on quantum hardware. A quantum circuit as a result of this transformation maintains the same logical operations as the original reversible circuit while using quantum gates.

In this work, this mapping will be based on the *Clifford+T library*. It includes *Pauli, Hadamard, and CNOT gates* as well as *T-gate* [13]. Since the T-gate is considered as the most expensive quantum gate, the cost efficiency of quantum circuits is evaluated in terms of the number of T-gates required.

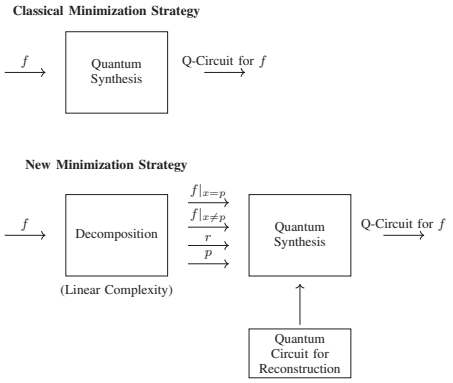


Fig. 3. Classical and new minimization strategies *without* SOP synthesis.

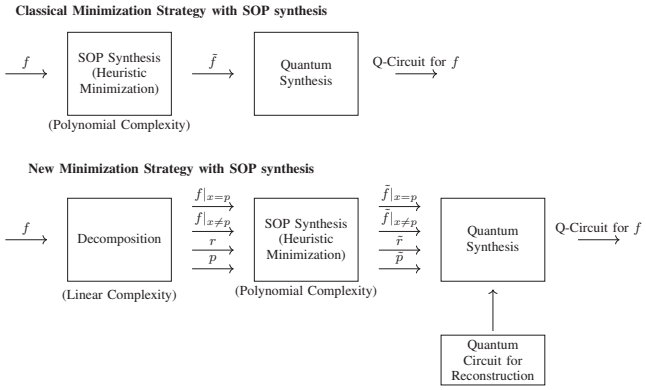


Fig. 4. Classical and new minimization strategies *with* SOP synthesis.

The classical cost in terms of *T-gates* for the realization of a 2-*controlled MPMC Toffoli gates* is 4, in accordance with the algorithm described in [10].

A detailed information on reversible circuits as well as an overview of efficient quantum compilation methods can be found in [10], [18].

III. Quantum circuits synthesis based on PSOP decomposition

In this section, we describe how the PSOP decomposition of a Boolean function f can be exploited to ease its quantum compilation. More precisely, we show how to combine quantum circuits for the two projected functions $f_{|x_i=p}$ and $f_{|x_i\neq p}$, the function p, and the remainder r, if present, in order to derive a quantum circuit for the original function f, following the strategies depicted schematically in Figure 3. Potential benefits of this approach are a reduced compilation time, and a final quantum circuit of reduced area with respect to the quantum circuit derived compiling directly the function f without leveraging its PSOP decomposed forms.

As already observed, this new quantum compilation strategy can be applied to any Boolean function, after the fast PSOP decomposition step, whose cost is linear in the initial SOP representation of the target function.

Let f be a Boolean function depending on n binary variables, and let PSOP(f) and Pr-SOP(f) denote its PSOP forms, without and with remainder r:

$$\text{PSOP}(f) = (\overline{x}_i \oplus p) f_{|x_i=p} + (x_i \oplus p) f_{|x_i\neq p}\,,$$
$$\text{Pr-SOP}(f) = (\overline{x}_i \oplus p) f_{|x_i=p} + (x_i \oplus p) f_{|x_i\neq p} + r\,,$$

where x_i is one input variable, p is a function that does not depend on x_i, $f_{|x_i=p}$ and $f_{|x_i\neq p}$ are the two projections of the SOP expression of f, and r is the remainder. Recall that both forms can be derived in linear time.

Before the quantum synthesis step, a heuristic SOP minimization step could be performed to facilitate quantum compilation and possibly derive more compact circuits, as shown in Figure 4. This step can be performed by applying polynomial time SOP heuristics on all components of the PSOP(f) and Pr-SOP(f) expressions, which are generally smaller functions, that depend on fewer variables and contain fewer minterms than the target function f. Notice that a similar step in the standard quantum compilation flow, not based on decomposition, would require the more costly heuristic SOP minimization of the whole function f. This preliminary minimization is not mandatory and can be avoided in case of large benchmarks, whose SOP minimization could result too time demanding.

After the optional SOP minimization step, quantum compilation is applied independently onto the subfunctions p, $f_{|x_i=p}$, $f_{|x_i\neq p}$, and the remainder r (if present).

Finally, we derive a quantum circuit for the overall function f using the quantum circuits for p, $f_{|x_i=p}$, $f_{|x_i\neq p}$, and the remainder r as building blocks, as shown in Figures 5 and 6.

Before describing how to derive a quantum circuit for a function f from PSOP decomposition, we state and prove a proposition that allows to ease the reconstruction strategy.

Proposition 1: Let f be a Boolean function depending on n binary variables, and let PSOP(f) and Pr-SOP(f) be its PSOP decomposition without and with remainder, respectively. The disjunction between the first two terms in both algebraic expressions can be replaced with an Exclusive Or:

$$\text{PSOP}(f) = (\overline{x}_i \oplus p) f_{|x_i=p} \oplus (x_i \oplus p) f_{|x_i\neq p}\,,$$
$$\text{Pr-SOP}(f) = \big((\overline{x}_i \oplus p) f_{|x_i=p} \oplus (x_i \oplus p) f_{|x_i\neq p}\big) + r\,.$$

Proof. Observe that the first two terms in both PSOP(f) and Pr-SOP(f) represent disjoint sets of points. Indeed, the two subspaces $B_{x_i=p}$ and $B_{x_i\neq p}$ do not intersect, and the product of their characteristic functions $(\overline{x}_i \oplus p)$ and $(x_i \oplus p)$ is the zero function. This immediately implies that the disjunction can be replaced with an exclusive OR. ■

This result is important for the reconstruction procedure since an EXOR can be easily implemented in a quantum circuit using a CNOT instead of a Toffoli gate.

We now describe the reconstruction procedure of a quantum circuit for f, considering first the case of PSOP decomposition without remainder.

The overall quantum circuit for f in this case is obtained concatenating the two quantum subcircuits for the projections, that depend on all variables but x_i, with the quantum circuit for $(x_i \oplus p)$, possibly depending on all input variables. The

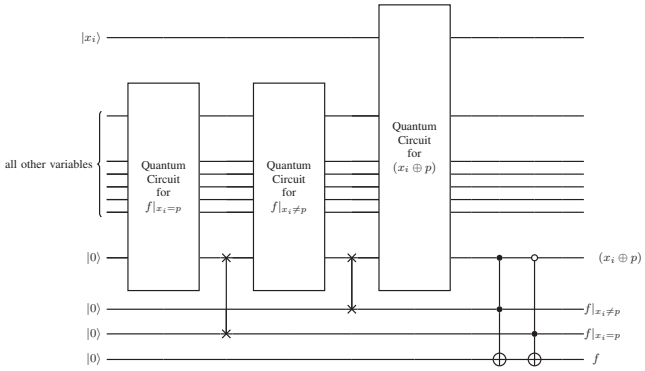


Fig. 5. Quantum circuit based on PSOP without remainder.

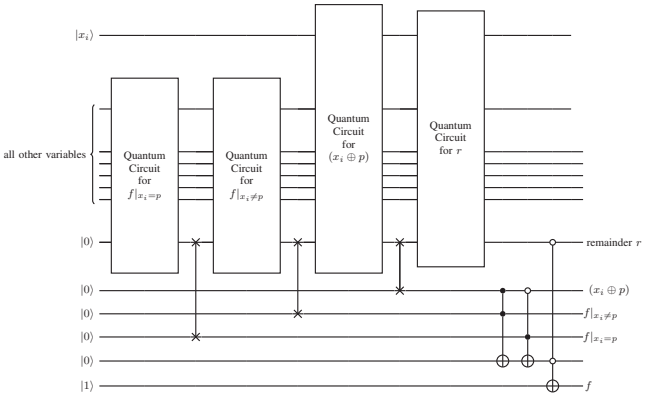


Fig. 6. Quantum circuit based on PSOP with remainder.

quantum circuit for $(x_i \oplus p)$ can be derived inserting a CNOT, controlled by x_i, on the output line of a quantum circuit for p. Note that four additional lines (and therefore four new qubits) are needed: one for $f_{|x_i=p}$, one for $f_{|x_i\neq p}$, one for $(x_i \oplus p)$ and finally one output line for f. The overall circuit structure is shown in Figure 5, where two swap gates are used to bring the qubits for the intermediate results closer to the corresponding subcircuits. Eventually, two Toffoli gates are inserted for computing the AND between the projections and the corresponding subspaces, one described by the subcircuit for $(x_i \oplus p)$ and the other by its complement. Both Toffoli gates act on the output line for f, thanks to the fact that the OR operator in the PSOP expression has been replaced with an EXOR. The overall methodology is summarized in the algorithm in Figure 7, and its cost in terms of elementary quantum T-gates is discussed in the following proposition.

Proposition 2: The number of T-gates required to synthesize the PSOP-based quantum circuit for f is given by the overall number of T-gates occurring in the subcircuits for $f_{|x_i=p}$, $f_{|x_i\neq p}$, and $(x_i \oplus p)$, plus 8 additional T-gates.

Proof. Observe from Figure 5 that the three quantum subcircuits for $f_{|x_i=p}$, $f_{|x_i\neq p}$, and $(x_i \oplus p)$ are combined using only two additional swap gates and two Toffoli gates. Since swap gates are implemented using CNOTs, only 8 additional T-gates are required, four for each Toffoli gate [10]. ■

Figure 6 shows the circuit for f based on the PSOP decomposition with remainder. As already noted in Proposition 1, the first disjunction can be replaced by an EXOR. Moreover, using De Morgan's laws, we can replace the remaining OR with a NAND. Thus the form becomes

$$\text{Pr-SOP}(f) = \overline{\left((\overline{x}_i \oplus p) f_{|x_i=p} \oplus (x_i \oplus p) f_{|x_i\neq p}\right)} \wedge \overline{r}$$

The overall quantum circuit for f is thus obtained concatenating the subcircuits for the projections, for the characteristic function $(x_i \oplus p)$ of the projection subspace, and for the remainder r, possibly depending on all input variables.

This time, six additional lines are used: two for $f_{|x_i=p}$ and $f_{|x_i\neq p}$, one for $(x_i \oplus p)$, one for the remainder, one for storing the intermediate result $(\overline{x}_i \oplus p) f_{|x_i=p} \oplus (x_i \oplus p) f_{|x_i\neq p}$, and one as output line for f. As before, swap gates are used to bring the qubits for the intermediate results closer to the corresponding subcircuits.

Two Toffoli gates, both acting on the same line, are then used for computing the EXOR of the products between the projections and the corresponding subspaces. A third Toffoli gate on the output line for f, inizialized with a qubit in state $|1\rangle$, is finally used to compute the NAND between the complement of the EXOR of the two products on the second to last line, and the complement of the remainder r.

The overall methodology, summarized in the algorithm in Figure 8, requires a constant number of additional T-gates for combining the four quantum subcircuits, as stated and proved in the following proposition.

Proposition 3: The number of T-gates required by the quantum circuit based on PSOP decomposition with remainder is given by the overall number of T-gates occurring in the subcircuits for $f_{|x_i=p}$, $f_{|x_i\neq p}$, $(x_i \oplus p)$, and r, plus 12 additional T-gates.

Proof. Observe from Figure 6 that the four quantum subcircuits for $f_{|x_i=p}$, $f_{|x_i\neq p}$, $(x_i \oplus p)$, and r are combined using three additional swap gates, implemented using CNOT gates only, and three Toffoli gates. Thus, only 12 additional T-gates are required, four for each Toffoli gate [10]. ■

The overall computational cost of the proposed approach includes the cost of the projections (linear in the initial SOP of f), the cost of the optional heuristic SOP minimization of $f_{|x_i=p}$, $f_{|x_i\neq p}$, p, and r (polynomial), the cost of their quantum compilation, and the (constant) cost for combining the quantum subcircuits into a quantum circuit for f.

The cost of the standard quantum compilation would include the cost of the optional heuristic SOP minimization of f and the cost of its quantum compilation.

IV. Experimental Results

In this section we evaluate the effectiveness of the proposed method for the quantum synthesis of PSOP-decomposed functions. We, then, present the computational results achieved by constructing PSOP expressions for Boolean functions, and comparing these expressions to their standard quantum synthesis forms. In order to assess reversible circuits derived

INPUT
f /* Function in SOP form depending on n variables $\{x_1, \ldots, x_n\}$ */
x_i /* An input variable */
p /* Function in SOP form depending on all input variables, but x_i */
$f_{|x_i \neq p}$ /* Projection of f onto the subspace $(x_i \oplus p)$ */
$f_{|x_i = p}$ /* Projection of f onto the subspace $(\overline{x}_i \oplus p)$ */

OUTPUT
Q /* Quantum circuit for f */

OPTIONAL: Heuristic SOP minimization of p, $f_{|x_i \neq p}$, $f_{|x_i = p}$;
$Q_{f_{\neq}}$ = QuantumSynthesis($f_{|x_i \neq p}$);
$Q_{f_{=}}$ = QuantumSynthesis($f_{|x_i = p}$);
Q_p = QuantumSynthesis($x_i \oplus p$);
Q = Toffoli($Q_{f_{\neq}}$, Q_p) $\oplus$ Toffoli($Q_{f_{=}}$, $\overline{Q}_p$);
return Q

Fig. 7. Quantum synthesis based on PSOP decomposition without remainder.

INPUT
f /* Function in SOP form depending on n variables $\{x_1, \ldots, x_n\}$ */
x_i /* An input variable */
p /* Function in SOP form depending on all input variables, but x_i */
$f_{|x_i \neq p}$ /* Projection of the non-crossing products of f onto the subspace $(x_i \oplus p)$*/
$f_{|x_i = p}$ /* Projection of the non-crossing products of f onto the subspace $(\overline{x}_i \oplus p)$*/
r /* Sum (OR) of the crossing products of f */

OUTPUT
Q /* Quantum circuit for f */

OPTIONAL: Heuristic SOP minimization of p, $f_{|x_i \neq p}$, $f_{|x_i = p}$, r;
$Q_{f_{\neq}}$ = QuantumSynthesis($f_{|x_i \neq p}$);
$Q_{f_{=}}$ = QuantumSynthesis($f_{|x_i = p}$);
Q_p = QuantumSynthesis($x_i \oplus p$);
Q_r = QuantumSynthesis(r);
Q_1 = Toffoli($Q_{f_{\neq}}$, Q_p) $\oplus$ Toffoli($Q_{f_{=}}$, $\overline{Q}_p$);
Q = 1 $\oplus$ Toffoli($\overline{Q_r}$, $\overline{Q_1}$);
return Q

Fig. 8. Quantum synthesis based on PSOP decomposition with remainder.

from PSOP decomposition and compare them with standard synthesis derived circuits, we have measured their number of qubits and also evaluated their cost in terms of elementary quantum T-gates. Specifically, we mapped each MPMC Toffoli gate into elementary quantum gates. This mapping was performed based on the Clifford+T library and the algorithm detailed in [10]. Since the T-gate is considered the most expensive gate in the library, usually the cost of a Toffoli gate is expressed in the number of T-gates needed for its realization. For this reason we report the number of T-gates in the tables.

All computational experiments have been run on a Intel i7-8550U CPU of 1.80GHz with 16GB of RAM. The benchmarks utilized are classical benchmarks in PLA form (classical Espresso and LGSynth'89 benchmark suite [19]). This choice is due to the fact that the computation of the function p described in [2] derives from a statistical analysis of the initial SOP (or PLA) form. The benchmarks in other classical sets (such as EPFL benchmark suite [16], [17]) are, unfortunately, not given in PLA form. We further discuss this point in the concluding section. As representative indicators of our experiments, we report only a significant subset of the functions.

The experiments has been conduct using the SOP minimization as described in the strategy depicted in Figure 4, using ESPRESSO [7] in the heuristic mode for the SOP synthesis. The experimental results are obtained by applying the XAG-based quantum compilation heuristic proposed in [11]. In particular, we are interested in evaluating experimentally whether this recent technique could benefit from the PSOP decomposition of the target function.

The decomposition phase is extremely fast, coherently with the linear time complexity of the corresponding algorithm [2]. Therefore, the computational times of the standard minimization and the decomposed one are extremely similar. For this reason the comparison of computational times is not interesting, and we do not report them in the tables.

In Table I, the names of a significant set of benchmarks, included in our experiments, are listed in the first column. The following four columns provide details on the number of T-gates, which determine the cost, and the number of qubits required for the quantum circuits obtained from standard synthesis and PSOP expressions of the benchmarks. As can be seen, we investigate three scenarios for PSOP expressions: the projection of f with respect to p is first explored as an AND of two variables (PSOP with AND), secondly as a simple Boolean variable (PSOP with variable), and lastly as an EXOR of two variables (PSOP with EXOR). In each scenario, we examine PSOP expressions both with and without remainder.

As shown in Table I, it is clear that some benchmarks experience significant advantages from PSOP expressions in terms of the number of T-gates and the number of qubits compared to standard synthesis. For instance, the benchmarks *rd73* and *sym10* achieve a significant reduction in T-gates and qubits when utilizing PSOP with EXOR (with remainder) and PSOP with EXOR (without remainder), respectively. Specifically, *rd73* shows a 49% reduction in T-gates and a 45% reduction in qubits, while *sym10* shows a 47% reduction in T-gates and an 46% reduction in qubits. However, in some cases, standard synthesis results in circuits with fewer T-gates and qubits compared with PSOP-based synthesis. For example, the *addm4* benchmark.

Overall, we can note that the best strategy seems to be the one where p is a single variable (with or without remainder). Moreover, the one that uses p as an AND gate is less useful. This is probably due to the fact that an AND gate has an expensive (in terms of T-gates) quantum representation. Meanwhile, the single variable or the EXOR gates require less expensive quantum gates.

Table II reports a subset of all the benchmarks used in our experiments. The first column lists the name of each benchmark. The next group of 2 columns detail the cost and the number of qubits for the the quantum circuits derived from standard synthesis and the best PSOP expressions of the benchmarks. Finally, the last column reports the gain in the number of T-gates.

According to the results shown in Table II, it is evident that some benchmarks benefit greatly from the proposed strategy. For instance, the benchmarks *newapla* and *newxcpla1* achieve a 71% and 75% reduction in T-gates, respectively. However, the gain is much less significant for some benchmarks, such as *in0* and *in2*. In some cases, the best PSOP strategy results in circuits with a higher number of T-gates, for example *adr4*

TABLE I
COMPARISON BETWEEN THE COMPILATION HEURISTIC PROPOSED IN [11] (STANDARD SYNTHESIS) APPLIED AFTER ESPRESSO IN THE HEURISTIC MODE, AND THE PROPOSED STRATEGY WITH DIFFERENT OPTIONS OF p WITH AND WITHOUT REMAINDER.

			PSOP with AND				**PSOP with variable**				**PSOP with EXOR**			
	Standard synthesis		**Without remainder**		**With remainder**		**Without remainder**		**With remainder**		**Without remainder**		**With remainder**	
Benchmark	**T -count**	**# qubits**	**T -count**	**# qubits**	**T -count**	**# qubits**	**T -count**	**# qubits**	**T -count**	**# qubits**	**T -count**	**# qubits**	**T -count**	**# qubits**
addm4	**1680**	429	2048	521	2156	548	2240	569	2244	570	2352	597	2352	597
adr4	**108**	35	264	74	120	38	216	62	216	62	352	96	440	118
amd	1244	325	**1096**	288	1128	296	1124	295	1132	297	1140	299	1140	299
apla	**404**	111	776	204	824	216	724	191	748	197	616	164	632	168
b3	1220	338	1320	363	**1200**	333	1304	359	1340	368	1272	351	1272	351
b10	1520	396	**1424**	372	1428	373	1436	375	1468	383	1552	404	1524	397
b12	280	85	344	101	260	80	260	80	**236**	74	408	117	256	79
bench	**228**	63	280	76	296	80	264	72	260	71	332	89	332	89
br1	504	138	580	157	496	136	**444**	123	504	138	524	143	524	143
br2	**348**	99	**348**	99	388	109	360	102	368	104	432	120	436	121
co14	192	62	244	75	224	70	192	62	192	62	**172**	57	180	59
dc2	**328**	90	424	114	364	100	**328**	90	**328**	91	424	114	424	115
exp	**1132**	292	1280	329	1284	330	1192	307	1208	311	1208	311	1272	327
f51m	454	121	412	111	**400**	108	508	135	508	135	420	113	416	112
fout	**820**	211	932	239	928	238	916	235	916	235	900	231	900	231
gary	1716	444	1792	463	1488	387	1612	418	1592	413	1692	438	**1440**	375
in0	1720	445	**1656**	429	1668	432	1712	443	1708	442	1720	445	1744	451
in2	1352	357	1632	427	1340	354	**1316**	348	1352	357	1328	351	1332	352
in3	1284	356	**1164**	326	1256	349	1224	341	1272	353	1280	355	1240	345
in4	1344	368	1384	378	1288	355	1372	375	1380	377	1344	368	**1320**	363
in5	1216	328	1160	314	1072	293	1052	287	**956**	264	1168	316	1088	297
in7	480	146	536	160	348	114	592	174	332	110	432	134	**316**	106
inc	**364**	98	444	118	444	118	380	102	384	103	388	104	388	104
m3	1024	264	936	242	948	245	1008	260	1052	271	**888**	230	912	236
m4	2128	540	1692	431	**1592**	406	1680	428	1836	467	1944	494	1932	491
max128	1392	356	**1196**	307	1232	316	1220	313	1260	323	1496	382	1496	382
mlp4	**1264**	324	1336	342	1308	335	1388	355	1388	355	1316	337	1316	337
newapla	136	46	196	61	72	31	140	47	**40**	23	200	62	76	32
newcpla1	336	93	464	125	248	72	280	79	320	90	364	100	**260**	75
newcpla2	228	64	216	61	**148**	45	236	66	168	49	240	67	152	46
newxcpla1	508	136	528	141	184	56	368	101	**128**	42	584	155	184	56
p3	632	167	736	193	720	189	**584**	155	592	157	628	166	620	164
p82	**292**	78	328	87	320	85	320	85	328	87	368	97	372	98
rckl	520	162	864	248	568	175	544	168	**296**	107	528	164	568	175
rd73	352	95	344	93	344	93	388	104	312	85	300	82	**180**	52
root	516	137	520	138	524	139	**452**	121	**452**	121	468	125	**452**	121
spla	**2536**	650	2720	696	2828	723	3356	855	3464	882	3308	843	3484	887
sqr6	404	108	408	109	416	111	**392**	105	**392**	105	440	117	428	114
sym10	1420	365	1080	280	1080	280	804	211	804	211	**748**	197	756	199
t1	**568**	163	792	219	792	219	572	164	592	169	816	225	816	225
t3	252	75	244	73	**228**	69	276	81	272	80	320	92	272	80
tms	744	194	828	215	840	218	704	184	744	194	**680**	178	692	181
vg2	444	136	440	135	360	116	512	153	364	116	348	112	**292**	99
x6dn	1112	317	1224	345	1180	334	**1080**	309	1092	312	1248	351	1308	366
x9dn	408	129	428	134	348	115	436	136	348	115	320	107	**256**	92
Z5xp1	456	121	380	102	392	105	512	135	520	137	356	96	**352**	95
Z9sym	828	216	804	210	788	206	692	182	692	182	**680**	179	**680**	179

and *apla*. Overall, the T cost of the best PSOP-based quantum circuit is significantly lower than that of circuits derived from standard synthesis.

It is also crucial to minimize the number of qubits in quantum circuit design. As can be observed in Table II, it is clear that the best PSOP strategy has a significant effect on some benchmarks in terms of the number of qubits. For example, the benchmark *newapla* and *newxcpla1* experiences more that 50% reduction in qubit numbers. However, the improvement is much smaller for some benchmarks like *b3* and *sqr6*. In some instances, the best PSOP strategy leads to circuits with a higher number of qubits, such as in the case of *adr4*. In general, we can see that the number of qubits in quantum circuits based on the best PSOP is notably fewer compared to the circuits obtained through standard synthesis.

In summary, we have that the proposed strategy gives better results for the 61% of the benchmarks with an average gain of about 22% in terms of T-gates, within the same time limit. Some benchmarks particularly benefit from this strategy, since their cost gain is more than the 70%.

V. CONCLUSION

This paper has described a pre-processing procedure and a reconstruction method to ease quantum synthesis. The proposed strategy is given by a PSOP decomposition based on the expression $x_i \oplus p$. Moreover, the algorithms have been experimentally tested on decompositions where p is a variable, an AND of variables and an XOR of variables, validating the proposed approach.

The synthesis method based on PSOP decomposition gives interesting results. Nevertheless, the decomposition is applied to SOP forms, since the PSOP decomposition is based on some statistics on the variables appearing in the starting SOP. This means that the input Boolean function must be represented with a PLA or any 2 level logic representation.

The future works on this topic should study new methods for deriving PSOP forms starting from other representations as AND inverter graphs or ROBBDs. Moreover, another interesting new direction is the study of the use of other possible decompositions for easing quantum compilation. For example, it would be interesting to study "Projected Exclusive Sum of Products" forms as a starting point for reversible

TABLE II
COMPARISON BETWEEN THE COMPILATION HEURISTIC PROPOSED IN [11] (STANDARD SYNTHESIS) APPLIED AFTER ESPRESSO IN THE HEURISTIC MODE, AND THE BEST SOLUTION OF THE PROPOSED STRATEGY.

	Standard synthesis		Best PSOP		Gain
Benchmark	**T -count**	**# qubits**	**T -count**	**# qubits**	**(T-gates)**
addm4	**1680**	429	2048	521	–
adr4	**108**	35	216	62	–
amd	1244	325	**1096**	288	12%
apla	**404**	111	616	164	–
b3	1220	338	**1200**	333	2%
b10	1520	396	**1424**	372	6%
b12	280	85	**236**	74	16%
bench	**228**	63	260	71	–
br1	504	138	**444**	123	12%
br2	**348**	99	**348**	99	–
co14	192	62	**172**	57	10%
dc2	**328**	90	**328**	90	–
exp	**1132**	292	1192	307	–
f51m	454	121	**400**	108	12%
fout	**820**	211	900	231	–
gary	1716	444	**1440**	375	16%
in0	1720	445	**1656**	429	4%
in2	1352	357	**1316**	348	3%
in3	1284	356	**1164**	326	9%
in5	1216	328	**956**	264	21%
in7	480	146	**316**	106	34%
inc	**364**	98	380	102	–
m3	1024	264	**888**	230	13%
m4	2128	540	**1592**	406	25%
max128	1392	356	**1196**	307	14%
mlp4	**1264**	324	1308	335	–
newapla	136	46	**40**	23	71%
newcpla1	336	93	**260**	75	23%
newcpla2	228	64	**148**	45	35%
newxcpla1	508	136	**128**	42	75%
p3	632	167	**584**	155	8%
p82	**292**	78	320	85	–
rckl	520	162	**296**	107	43%
rd73	352	95	**180**	52	49%
root	516	137	**452**	121	12%
spla	**2536**	650	2720	696	–
sqr6	404	108	**392**	105	3%
sym10	1420	365	**748**	197	47%
t1	**568**	163	572	164	–
t3	252	75	**228**	69	10%
tms	744	194	**680**	178	9%
vg2	444	136	**292**	99	34%
x6dn	1112	317	**1080**	309	3%
x9dn	408	129	**256**	92	37%
Z5xp1	456	121	**352**	95	23%
Z9sym	828	216	**680**	179	18%

logic synthesis, instead of Exclusive Sum of Products (ESOP) expressions [15].

ACKNOWLEDGMENT

This study was carried out within the National Centre on HPC, Big Data and Quantum Computing - SPOKE 10 (Quantum Computing) and received funding from the European Union Next-GenerationEU - National Recovery and Resilience Plan (NRRP) – MISSION 4 COMPONENT 2, INVESTMENT N. 1.4 – CUP N. I53C22000690001.

This work was supported in part by project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the Italian MUR. Neither the European Union nor the Italian MUR can be held responsible for them.

The first two authors are members of the Gruppo Nazionale Calcolo Scientifico-Istituto Nazionale di Alta Matematica (GNCS-INdAM), which provided partial support for this work.

REFERENCES

[1] A. Bernasconi, A. Berti, V. Ciriani, G. M. D. Corso, and I. Fulginiti, "XOR-AND-XOR logic forms for autosymmetric functions and applications to quantum computing," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 42, no. 6, pp. 1861–1872, 2023.

[2] A. Bernasconi, V. Ciriani, and R. Cordone, "On Projecting Sums of Products," in *11th Euromicro Conference on Digital Systems Design: Architectures, Methods and Tools*, 2008.

[3] ——, "The optimization of kEP-SOPs: Computational complexity, approximability and experiments," *ACM Trans. Design Autom. Electr. Syst.*, vol. 13, no. 2, 2008.

[4] A. Bernasconi, V. Ciriani, A. T. Monfared, and S. Zanoni, "Compact quantum circuits for dimension reducible functions," in *26th Euromicro Conference on Digital System Design, DSD 2023, Golem, Albania, September 6-8, 2023*. IEEE, 2023, pp. 776–781.

[5] A. Bernasconi, V. Ciriani, G. Trucco, and T. Villa, "On decomposing boolean functions via extended cofactoring," in *2009 Design, Automation & Test in Europe Conference & Exhibition*. IEEE, 2009, pp. 1464–1469.

[6] J. C. Bioch, "The complexity of modular decomposition of boolean functions," *Discrete Applied Mathematics*, vol. 149, no. 1-3, pp. 1–13, 2005.

[7] R. Brayton, G. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*. Kluwer Academic Publishers, 1984.

[8] P. Kerntopf, "New generalizations of shannon decomposition," in *Int. Workshop on Applications of Reed-Muller Expansion in Circuit Design*, 2001, pp. 109–118.

[9] D. Maslov, G. W. Dueck, and D. M. Miller, "Synthesis of fredkin-toffoli reversible networks," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, no. 6, pp. 765–769, 2005.

[10] G. Meuli, M. Soeken, E. Campbell, M. Roetteler, and G. De Micheli, "The role of multiplicative complexity in compiling low t-count oracle circuits," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2019, pp. 1–8.

[11] G. Meuli, M. Soeken, E. Campbell, M. Roetteler, and G. D. Micheli, "The role of multiplicative complexity in compiling low t-count oracle circuits," in *Proceedings of the International Conference on Computer-Aided Design, ICCAD*, D. Z. Pan, Ed. ACM, 2019, pp. 1–8.

[12] D. M. Miller, D. Maslov, and G. W. Dueck, "A transformation based algorithm for reversible logic synthesis," in *Proceedings of the 40th annual Design Automation Conference*, 2003, pp. 318–323.

[13] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge university press, 2010.

[14] B. Schmitt, F. Mozafari, G. Meuli, H. Riener, and G. D. Micheli, "From boolean functions to quantum circuits: A scalable quantum compilation flow in C++," in *Design, Automation & Test in Europe Conference & Exhibition*. IEEE, 2021, pp. 1044–1049.

[15] B. Schmitt, M. Soeken, G. D. Micheli, and A. Mishchenko, "Scaling-up ESOP synthesis for quantum compilation," in *2019 IEEE 49th International Symposium on Multiple-Valued Logic (ISMVL), Fredericton, NB, Canada, May 21-23, 2019*. IEEE, 2019, pp. 13–18.

[16] E. Testa, M. Soeken, H. Riener, L. Amaru, and G. D. Micheli, "A logic synthesis toolbox for reducing the multiplicative complexity in logic networks," in *2020 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2020, pp. 568–573.

[17] E. Testa, M. Soeken, L. G. Amarù, and G. D. Micheli, "Reducing the multiplicative complexity in logic networks for cryptography and security applications," in *Proceedings of the 56th Annual Design Automation Conference 2019, DAC 2019, Las Vegas, NV, USA*, 2019, p. 74.

[18] R. Wille, S. Hillmich, and L. Burgholzer, "Efficient and correct compilation of quantum circuits," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020, pp. 1–5.

[19] S. Yang, "Logic synthesis and optimization benchmarks user guide version 3.0," Microelectronic Center, User Guide, 1991.

COSOI: True Random Number Generator Based on Coherent Sampling using the FD-SOI technology

Licinius Benea, Florian Pebay-Peyroula, Mikael Carmona
Univ. Grenoble Alpes, CEA, Leti,
Grenoble, France
{licinius-pompiliu.benea, florian.pebay,
mikael.carmona}@cea.fr

Romain Wacquez
CEA-Leti, Mines Saint-Etienne
Gardanne, France
romain.wacquez@cea.fr

Abstract— **This work presents a proof of concept of the implementation of a Coherent Sampling Ring Oscillator TRNG (COSO-TRNG) using the Fully Depleted Silicon On Insulator (FD-SOI) technology. COSO-TRNG appears as one of the best structures optimizing the throughput per area trade-off and having a model for its entropy source. The back-biasing capability of the FD-SOI technology is proved here to be a very simple and efficient technique for the ring oscillator frequency calibration needed for the coherent sampling method. This is the first demonstration of feasibility of COSO-TRNG validated on ASIC FD22nm. A throughput of 3.36 Mbits/s was obtained, equivalent to results in the literature.**

Keywords— True random number generator, coherent sampling, phase noise, jitter, ring oscillator, Allan variance, noise models, Fully Depleted Silicon On Insulator

I. Introduction

True Random Number Generators (TRNG) are an essential brick for any cryptographic system. In contrast to PRNGs (Pseudo Random Number Generators), which use a known nonce to generate random numbers, TRNGs are based on physical phenomena. As a consequence, current standards imposed by BSI (Bundesamt für Sicherheit in der Informationstechnik) [1], NIST (National Institute of Standards and Technology) [2] and ISO (International Organization for Standardization) [3] demand a stochastic model capable of estimating the entropy of the generated bits. The model should be based on the known characteristics of the physical noise source.

For this reason, ring oscillators (RO) are a widespread choice due to their well-established models. Random number generation originates from phase noise, which represents the difference in time between the expected and the measured clock signal. The position of this variation, called jitter, translates into random values. In the literature, there is a great variety of proposed designs and their respective models: the Elementary Ring Oscillator TRNG [4], the Coherent Sampling Ring Oscillator TRNG [5], [6], the Transition Effect Ring Oscillator TRNG [7], [8], the Edge Sampling TRNG [9] and the Multi-Ring Oscillator TRNG [10], [11].

All principles considered, the Coherent Sampling Ring Oscillator (COSO) TRNG is an interesting choice due to its simplicity, low area and relatively good output [12]. Moreover, the COSO has the unique digitizing architecture that contains the total failure alarm by design. Nevertheless, its functionality is conditioned upon a perfectly controlled ratio between the frequencies of the two ring oscillators, which varies on the basis of the local mismatch inherent to all contemporary silicon technologies. This has thus so far hindered the widespread use of the COSO TRNG. In order to overcome this issue, multiple solutions were proposed. Peetermans *et al.* [13] introduced a dynamic calibration mechanism using a combination of four multiplexers (MUXs) configured statically to switch between a multitude of possible paths, in order to achieve the desired precision on the RO frequencies. Other architecture proposed by Tang *et al.* [14] uses a trimming capacitor bank, which is connected to all inverter stages of the ring oscillators. Despite their benefits, these approaches change the RO architecture for which the existing ring oscillator phase models may not apply directly.

This article proposes a new approach based on the Fully Depleted Silicon on Insulator (FD-SOI) technology specificities. This technology allows the use of a secondary transistor gate (back gate) in order to tune transistor characteristics. As such, the back gate biasing is used here in order to modify the threshold voltage of the transistors constituting the ring oscillator, and, as a consequence the ring oscillator nominal frequency without modifying the RO architecture.

The article is organised in the following way: Section 2 presents a description of the FD-SOI technology, of the working principle of COSO-TRNG and of the utilized ASIC structure. In Section 3, the results obtained on an ASIC structure are presented: inherent technology variability and its impact on the output signal of COSO-TRNG, influence on parameters as a consequence of back-gate voltage variation and the implications on entropy. Finally, conclusion and perspectives are presented in the last section.

II. Methods

This section provides an all level description of the device studied in this work, beginning with general information about the FD-SOI transistor, exposing the working principle of the COSO-TRNG and, finally, a description of the experimental setup.

A. FD-SOI technology

The FD-SOI technology is semiconductor fabrication technique that offers significant advantages over traditional bulk CMOS (Complementary Metal-Oxide-Semiconductor) processes. Its main characteristic is related to the ultra-thin insulating layer called BOX (Buried Oxide), which physically delimits the transistor channel (Fig. 1). This reduces leakage currents enhancing energy efficiency and enables very efficient electrostatic control of the channel at lower gate length. The N-Well and P-Well doped regions below the thin BOX can act a second gate with a very efficient coupling to the channel forming a very well-established and already adopted option for power management in the market of connectivity. Indeed, the back-biasing capability of FD-SOI is particularly noteworthy, as it allows for dynamic adjustment of the threshold voltage, providing a flexible trade-off between performance and power savings. Moreover, the difference between the capacitances of the front-gate and back-gate allow a precise adjustment of the transistor threshold voltage. Studies in the literature show a threshold voltage tuning capability of the order of 100 mV/V [15]. This signifies that for every 1 V applied on the back-gate, the threshold voltage of the transistor shifts 100mV.

B. COSO-TRNG

The COSO-TRNG working principle is based upon sampling one ring oscillator (RO1) with another ring oscillator (RO0), see Fig. 2 (a). When the signal from RO1 is in advance with respect to the signal of RO0, the D flip-flop generates a "1". Respectively, when the RO1 is behind RO0, the D flip-flop generates a "0" (Fig. 2 (b)). The length of the resulting signal (called "beat") is inversely proportional to the difference in periods of the two ring oscillators according to the following formula:

$$N_{mean} = \frac{T_0}{|T_1 - T_0|} \quad (1)$$

where T_0, T_1 are the average periods of RO0 and RO1, respectively.

The length of the beat signal and the variations around it represent the framework allowing the generation of bits through the LSB (Least Significant Bit) and the characterization of the noise source through, for example, the variance. The mutualisation of bit generation, total failure test and entropy source characterization on a single output variable constitute one of the key advantages of the structure.

C. Our setup

The device under test (DUT) is fabricated using the 22 nm FD-SOI technology. The results presented in this article use a structure with two ring oscillators made up of one NAND gate for the enable/disable function and 216 inverter stages amounting to a nominal frequency of 269 MHz. The physical implementation of the two ROs are fully similar and the difference in frequency is only explained by local variability of the technology. The ROs are designed with LVT transistors, with flipped well, meaning that the application of a back bias to these transistors lead to FBB (Forward Body Bias) which increases the frequencies of the ring oscillators. As the goal of the designed structure was to make a proof of concept, the back bias voltage (V_{BB}) is applied externally on the RO1 oscillator. For the sampling oscillator (RO0) the back bias voltage is set to 0 V. A counter after the D flip-flop measures the length (in periods of RO0) of the beat signal.

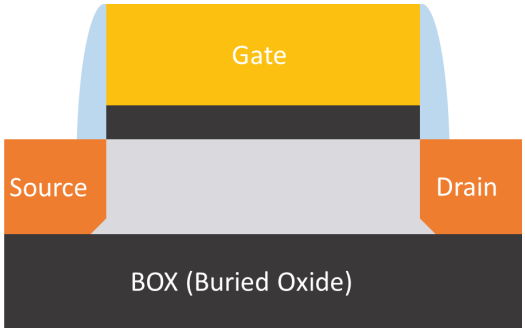


Fig. 1. Schematic representation of a FD-SOI transistor

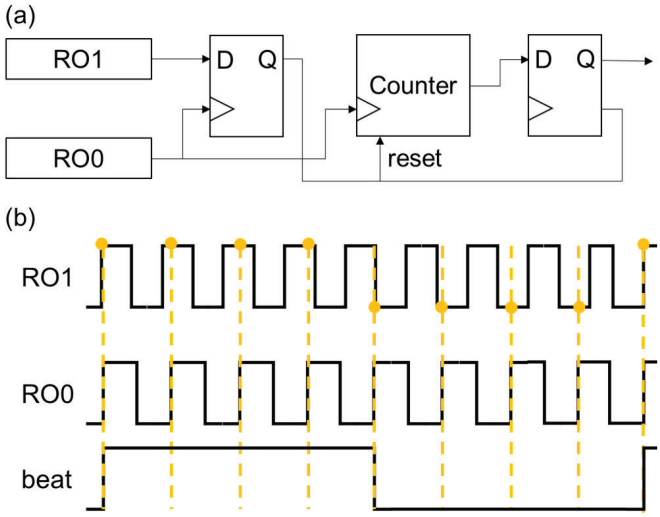


Fig. 2. Architecture of the COSO-TRNG (a) and schematic representations of the signals (b)

III. Results

In this section, we present the results and analysis of the obtained on the structure described in the previous section.

A. Inherent variability and effect on N_{mean}

By developing equation (1), the inverse for N_{mean} can be determined as:

$$\frac{1}{N_{mean}} = \left|\frac{T_1}{T_0} - 1\right| \quad (2)$$

By assuming a Gaussian distribution of T_0 and T_1 as a result of fabrication, their quotient also follows a Gaussian distribution according to [16]. Therefore, N_{mean} follows an inverse Gaussian, or Wald distribution. Fig. 3 presents the results obtained from 42 identically fabricated ring oscillator pairs on 3 separate chips. By

determining distribution parameters $\mu = 161$ and $\lambda = 170$, one can simulate a Wald distribution results using the invgauss function of the Python scipy library [17]. Results presented in Fig. 3 fit measured data. We conclude that N_{mean} follows a Wald distribution with an average value of 161 for measured devices.

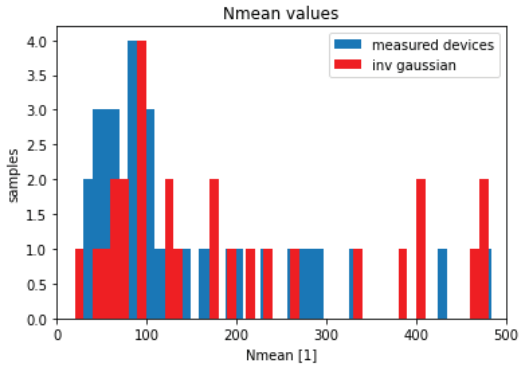


Fig. 3. Histogram of measured Nmean values on 42 devices (blue) and simlulated Wald distribution based on extracted parameters (red).

B. Counter variation against V_{BB}

Body biasing modifies the frequency of RO1, and, therefore the difference in period between RO0 and RO1. This in turn, modifies the value of N_{mean}.

The histograms corresponding to the measured counter values obtained for different V_{BB} varying from 0 V to 1.6 V are traced in Fig. 4. The mean value of the histograms increases as the frequencies of RO0 and RO1 are closer and then decreases when they come apart. Additionally, the histograms corresponding to higher N_{mean} approach more a Gaussian behaviour. This is due to a better measurement precision, which is in this case $1/N_{mean}$. The histograms are, in fact, discrete representations of jitter for different accumulation intervals corresponding to N_{mean} periods. A higher N_{mean} accounts for a better measurement precision of jitter, but also for a higher accumulation time.

For the same sample, Fig. 5 presents the variation of N_{mean} for different V_{BB}. As observed from Fig. 4, the mean counter value increases as the frequencies of the two ring oscillators are closer and decreases when their frequencies are further apart. For the presented device, the maximum value of 121.43 is reached for $V_{BB} = 0.3V$.

The average counter value gives access to the difference in frequency between RO0 and RO1 (equation 1). Assuming a nominal frequency of 269 MHz, one can determine the difference in frequency. The results are presented in Fig. 6 and show a quadratic behaviour. According to [18], the frequency of a ring oscillator is quadratically dependent on the overdrive. The latter represents the difference between the voltage applied to the transistor gate and the threshold voltage of the transistor. Assuming a linear dependency between the threshold voltage and the back-bias voltage, as shown in [15] for the range of values used in this case, the frequency of RO1, and thus the difference in frequency between RO1 and RO0 should vary quadratically depending on V_{BB}. The quadratic fit of the measured values presented in Fig. 6 proved the validity of this assumption. More importantly, we observe that for a large spectrum of low V_{BB} values (inferior to 0.8V), there is a very low variation of the difference in frequency, enabling a fine tuning so that the frequencies of the two ring oscillators are as close as needed.

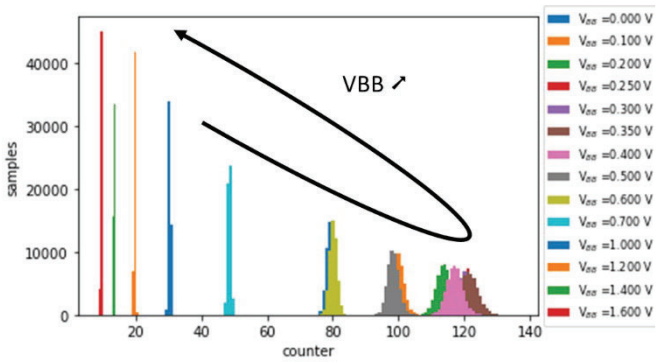


Fig. 4. Histogram of counter values mesured for different V_{BB}

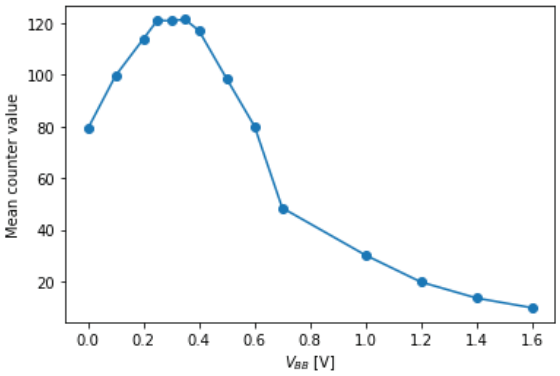
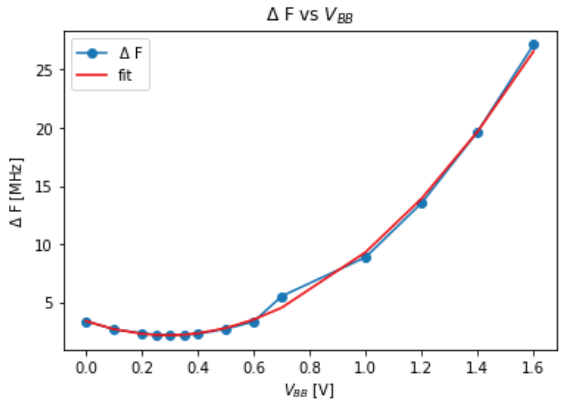


Fig. 5. Variation of N_{mean} in function of V_{BB}



a.

Fig. 6. Calculated difference in frequency $\Delta F = |F_{RO1} - F_{RO0}|$ calculated for different V_{BB}

C. Variance of counter values

The variance of jitter is the property which directly determines the entropy of the TRNG [4]. The variations of counter values are a discrete representation of jitter sampled with a precision of 1 bit. The physical origins of jitter are thermal noise and flicker noise [18]. Thermal noise is completely random and uncorrelated which makes it an ideal source for TRNG applications. On the other hand, flicker noise originates from two major phenomena [19]: carrier number fluctuation due to presence of traps at the interface between the gate oxide and silicon, and mobility fluctuation due to Coulomb scattering. This type of jitter introduces correlations in the bit series extracted from jitter and can decrease the quality of randomness. The variance of the accumulated jitter varies linearly in the case of thermal noise jitter (predominant for low accumulation times) and quadratically for flicker noise jitter (predominant for higher accumulation times) [18]. Depending on the number of accumulation periods N, the variance of jitter follows a quadratic law:

$$\sigma_N^2 = a_0 + a_1 \cdot N + a_2 \cdot N^2 \quad (3)$$

Where a_0, a_1, a_2 are the corresponding coefficients attributed to jitter resulting from quantization, thermal and flicker noise, respectively.

The variance of counter values for different mean counter values (obtained by varying V_{BB}) is traced in Fig. 7. The latter accounts here for the number of accumulated periods. The curve begins with a plateau with values close to 1/12, which corresponds to the quantization noise measured with a precision of 1 bit [20]. The curve has a quadratic behaviour, as expected theoretically.

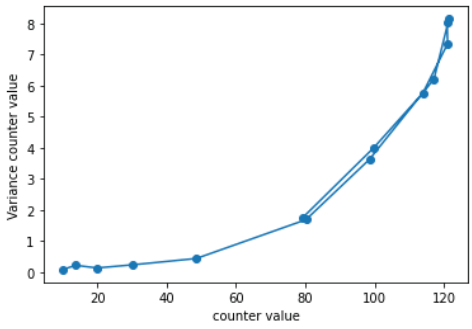


Fig. 7. Variance of counter values for different mean counter values.

In order to further investigate the amplitude of the measured thermal and flicker contributions to jitter, variance was also traced for counter values corresponding to all V_{BB} values (Fig. 8). This time, the accumulation periods are multiples of N_{mean}. Moreover, in this case, the Allan variance [21] was used a proposed in [22], [23]. The curves in a log-log scale show a quadratic behaviour, with slopes equal to 1 for low accumulation times, accounting for the thermal dominant region, and with a slope equal to 2 for higher accumulation times, highlighting the effect of flicker noise. However, the calculated values of Allan variance are higher for curves corresponding to higher N_{mean}. This effect can be observed for equivalent accumulation times, where, in some cases, even an order of magnitude of difference can be seen. Moreover, the curves corresponding to low N_{mean} values present a predominantly linear behaviour, whereas the curves corresponding to high N_{mean} values show a quadratic behaviour. An analysis of the correct noise amplitudes will be carried out in the followings.

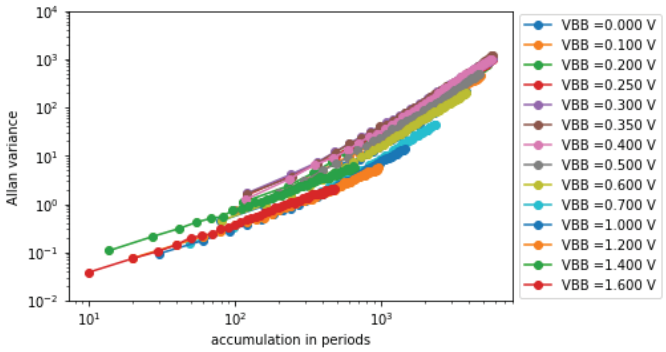


Fig. 8. Allan variance vs. accumulation times corresponding to multiples of N_{mean} for V_{BB} varying from 0V to 1.6V

The amplitudes of the corresponding thermal and flicker contributions to jitter can be determined by fitting the variance curves and extracting the a_1, a_2 coefficients, which correspond to thermal and flicker noise amplitudes, respectively. As the accumulation times cover multiple orders of magnitude, a normalized regression method is needed. A classical method would grant disproportionately higher weights to larger values. The Least-Squares Normalized Error (LSNE) method was used for its simplicity and effectiveness [24].

The obtained coefficients determined from the curves in Fig. 8 are represented in Fig. 9 and Fig. 10 for flicker and thermal noise respectively. One can observe that in both cases, there is an increase in coefficient values, which is correlated with the increase in Nmean, especially for V_{BB} values where Nmean is greater than 100 (V_{BB} between 0.2V and 0.4V). For a_1, we consider those values as an overestimation due to the fact that the curves are essentially in a quadratic domain. Also, the values corresponding to V_{BB} greater than 1.2V also present higher values, which may be attributed to an overestimation of thermal noise due to a poor measurement precision (high quantization noise) as observed in [25]. Consequently, the thermal noise amplitude coefficient can be determined by averaging the obtained coefficients, excluding the values from the ranges mentioned in previous paragraphs. The obtained value of a_1 is $2.73 \cdot 10^{-3}$.

D. Entropy and output

Due to intrinsic correlated behaviour of flicker noise, only the thermal component of jitter is used as a legitimate entropy source.

The COSO-TRNG model presented in [13] makes the assumption that the distribution of counter values is Gaussian, without making any distinction about the origin of the accumulated jitter. The min-entropy is determined as:

$$H_\infty = -\log_2\left(max(p_0, p_1)\right) \quad (4)$$

Where p_0, p_1 are the probabilities that the counter value is even or odd, respectively. i.e. the LSB of the counter value is 0 or 1, respectively.

A model which takes into consideration only the thermal contribution of jitter in order to determine a minimum bound for entropy is presented in [4]. The equation describing the entropy is:

$$H_{min} = 1 - \frac{4}{\pi^2 \ln 2} \cdot exp\left(-4 \cdot \pi^2 \cdot \left(\frac{{\sigma_T}^2 \cdot N}{{T_1}^2}\right) \cdot \frac{T_0}{T_1}\right) \quad (5)$$

Where σ_T is the thermal jitter, T_0, T_1 the periods of the sampling and sampled RO, respectively and N the accumulation time in periods of RO0.

This model is conceived for the Elementary RO TRNG, but the formula can be adapted to the COSO-TRNG by replacing equivalent terms:

$$H_{min} = 1 - \frac{4}{\pi^2 \ln 2} \cdot exp\left(-4 \cdot \pi^2 \cdot a_1 \cdot N \cdot \frac{N_{mean}+1}{N_{mean}}\right) \quad (5)$$

Where $a_1 \cdot N = \frac{{\sigma_T}^2 \cdot N}{{T_1}^2}$ is the normalized variance of jitter coming from thermal noise and $\frac{N_{mean}+1}{N_{mean}} = \frac{T_0}{T_1}$ is the adjustment coefficient equivalent to the ratio of the periods of the two ring oscillators.

The obtained results for the min-entropy estimation are presented in Fig. 11. While the results obtained from [13] are discrete, as they are obtained from counter values, the model in [4] allows obtaining a continuous function based on the value a_1 determined in the previous section. In order to obtain an entropy greater than 0.9998, the minimal accumulation times in periods of RO0 are N > 80 for [13] and N > 74 for [4]. In order to accommodate both conditions and considering the frequency of RO0 at 269 MHz, the calculated output of the TRNG is 3.63MHz. This is equivalent to results obtained in the literature [12], [13].

E. Discussion about noise composition

The point of equivalence between thermal and flicker noise contributions (N_C) can be determined by calculating the ratio a_1/a_2. The obtained values for different V_{BB} are presented in Fig. 12. The results prove that thermal domain is limited to accumulation times lower than N = 100 periods of RO0. This may explain the cause of the greater values of variance observed for N_{mean} greater than 100.

By using the coefficients for the worst case at V_{BB} = 0.2V, the thermal noise and flicker noise composition is traced in Fig. 13. One can observe that even for low accumulation times, flicker noise plays an important part in the mix. For example, a combination of 10% flicker and 90% thermal is already reached at N=11 periods. This might be enhanced by the use of the advanced 22nm FD-SOI technology. This shows that for newer technological nodes, flicker noise represents a major part of the mixture and its influence on entropy estimation needs to be understood [26].

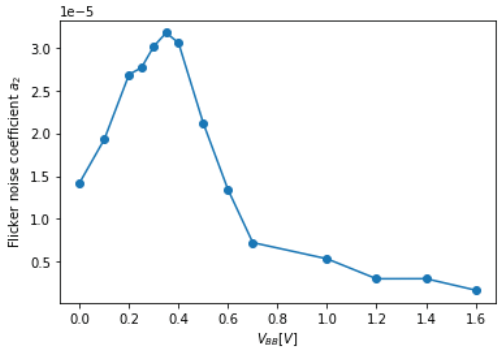


Fig. 9. Flicker noise coefficient for different V_{BB} values

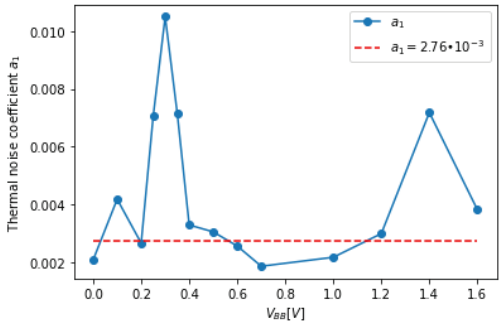
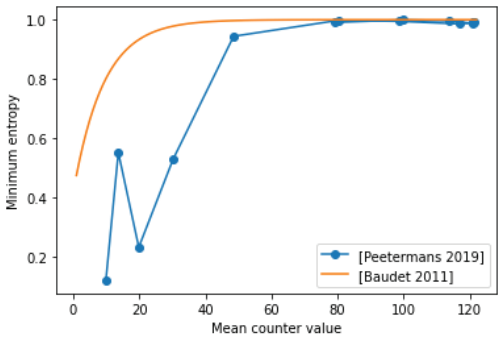


Fig. 10. Thermal noise coefficient for different V_{BB} values



b.

Fig. 11. Measured minimum entropy estimation of our TRNG according to [13] and [4]

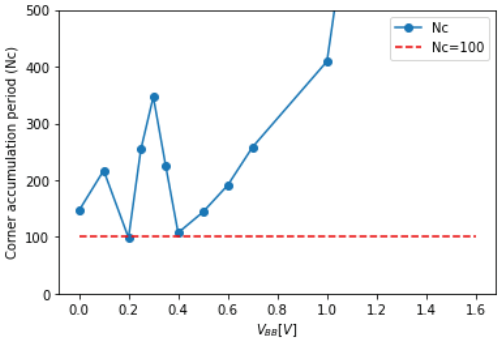


Fig. 12. Accumulation time in periods of RO0 (N_C) at thermal-flicker equivalence.

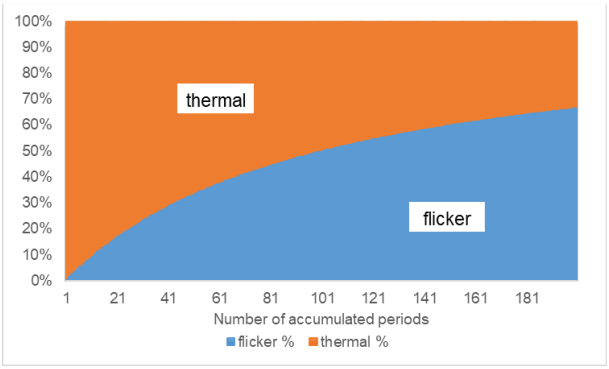


Fig. 13. Noise composition for V_{BB} = 0.2V

IV. Conclusions

This work presents a proof of concept of the implementation of a COSO-TRNG using the FD-SOI technology. The back-biasing technique, specific to this technology, proved to be well suited for the ring oscillator frequency calibration needed for the coherent sampling method.

The isolation of thermal noise needed to determine the entropy proved to be a complex issue. A deep analysis needs to be done by varying different parameters to obtain the correct estimation. By applying two distinct models, a throughput of 3.36 Mbits/s was obtained, equivalent to results in the literature.

Further work need to be realized on the study of the different noise sources present in the architecture, optimisation of the design, improvement of the figures of merit and on the statistical tests adapted to the COSO-TRNG.

References

[1] M. Peter and W. Schindler, 'A Proposal for Functionality Classes for Random Number Generators', *02.06.2023*, [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Certification/Interpretations/AIS_31_Functionality_classes_for_random_number_generators_e_2023.html?nn=910324

[2] M. S. Turan, E. Barker, J. Kelsey, K. A. McKay, M. L. Baish, and M. Boyle, 'Recommendation for the entropy sources used for random bit generation', National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-90b, Jan. 2018. doi: 10.6028/NIST.SP.800-90B.

[3] 'ISO/IEC JTC 1/SC 27. Test and analysis methods for random bit generators 541 within ISO/IEC 19790 and ISO/IEC 15408', ISO. [Online]. Available: https://www.iso.org/standard/68296.html

[4] M. Baudet, D. Lubicz, J. Micolod, and A. Tassiaux, 'On the Security of Oscillator-Based Random Number Generators', *J Cryptol*, vol. 24, no. 2, pp. 398–425, Apr. 2011, doi: 10.1007/s00145-010-9089-3.

[5] P. Kohlbrenner and K. Gaj, 'An embedded true random number generator for FPGAs', in *Proceeding of the 2004 ACM/SIGDA 12th international symposium on Field programmable gate arrays - FPGA '04*, Monterey, California, USA: ACM Press, 2004, p. 71. doi: 10.1145/968280.968292.

[6] F. Bernard, V. Fischer, and B. Valtchanov, 'Mathematical model of physical RNGs based on coherent sampling', *Tatra Mountains Mathematical Publications*, vol. 45, no. 1, pp. 1–14, Dec. 2010, doi: 10.2478/v10127-010-0001-1.

[7] M. Varchola and M. Drutarovsky, 'New High Entropy Element for FPGA Based True Random Number Generators', in *Cryptographic Hardware and Embedded Systems, CHES 2010*, vol. 6225, S. Mangard and F.-X. Standaert, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 351–365. doi: 10.1007/978-3-642-15031-9_24.

[8] P. Haddad, V. Fischer, F. Bernard, and J. Nicolai, 'A Physical Approach for Stochastic Modeling of TERO-Based TRNG', in *Cryptographic Hardware and Embedded Systems -- CHES 2015*, vol. 9293, T. Güneysu and H. Handschuh, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 357–372. doi: 10.1007/978-3-662-48324-4_18.

[9] B. Yang, V. Rožic, M. Grujic, N. Mentens, and I. Verbauwhede, 'ES-TRNG: A High-throughput, Low-area True Random Number Generator based on Edge Sampling', *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 267–292, Aug. 2018, doi: 10.13154/tches.v2018.i3.267-292.

[10] B. Sunar, W. Martin, and D. Stinson, 'A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks', *IEEE Trans. Comput.*, vol. 56, no. 1, pp. 109–119, Jan. 2007, doi: 10.1109/TC.2007.250627.

[11] D. Lubicz and V. Fischer, 'Entropy Computation for Oscillator-based Physical Random Number Generators', *J Cryptol*, vol. 37, no. 2, p. 13, Feb. 2024, doi: 10.1007/s00145-024-09494-6.

[12] O. Petura, U. Mureddu, N. Bochard, V. Fischer, and L. Bossuet, 'A survey of AIS-20/31 compliant TRNG cores suitable for FPGA devices', in *2016 26th International Conference on Field Programmable Logic and Applications (FPL)*, Lausanne, Switzerland: IEEE, Aug. 2016, pp. 1–10. doi: 10.1109/FPL.2016.7577379.

[13] A. Peetermans, V. Rozic, and I. Verbauwhede, 'A Highly-Portable True Random Number Generator Based on Coherent Sampling', in *2019 29th International Conference on Field Programmable Logic and Applications (FPL)*, Barcelona, Spain: IEEE, Sep. 2019, pp. 218–224. doi: 10.1109/FPL.2019.00041.

[14] Q. Tang, B. Kim, Y. Lao, K. K. Parhi, and C. H. Kim, 'True Random Number Generator circuits based on single- and multi-phase beat frequency detection', in *Proceedings of the IEEE 2014 Custom Integrated Circuits Conference*, San Jose, CA, USA: IEEE, Sep. 2014, pp. 1–4. doi: 10.1109/CICC.2014.6946136.

[15] C. Navarro, M. Bawedin, F. Andrieu, B. Sagnes, F. Martinez, and S. Cristoloveanu, 'Supercoupling effect in short-channel ultrathin fully depleted silicon-on-insulator transistors', *Journal of Applied Physics*, vol. 118, no. 18, p. 184504, Nov. 2015, doi: 10.1063/1.4935453.

[16] J. M. Hernández-Lobato, 'Balancing Flexibility and Robustness in Machine Learning: Semi-parametric Methods and Sparse Linear Models'.

[17] 'SciPy -'. [Online]. Available: https://scipy.org/

[18] A. Hajimiri, S. Limotyrakis, and T. H. Lee, 'Jitter and phase noise in ring oscillators', *IEEE J. Solid-State Circuits*, vol. 34, no. 6, pp. 790–804, Jun. 1999, doi: 10.1109/4.766813.

[19] G. Ghibaudo, O. Roux, Ch. Nguyen-Duc, F. Balestra, and J. Brini, 'Improved Analysis of Low Frequency Noise in Field-Effect MOS Transistors', *physica status solidi (a)*, vol. 124, no. 2, pp. 571–581, 1991, doi: 10.1002/pssa.2211240225.

[20] W. R. Bennett, 'Spectra of Quantized Signals', *Bell System Technical Journal*, vol. 27, no. 3, pp. 446–472, Jul. 1948, doi: 10.1002/j.1538-7305.1948.tb01340.x.

[21] D. W. Allan, 'Statistics of atomic frequency standards', *Proc. IEEE*, vol. 54, no. 2, pp. 221–230, 1966, doi: 10.1109/PROC.1966.4634.

[22] P. Haddad, Y. Teglia, F. Bernard, and V. Fischer, 'On the assumption of mutual independence of jitter realizations in P-TRNG stochastic models', in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014*, Dresden, Germany: IEEE Conference Publications, 2014, pp. 1–6. doi: 10.7873/DATE.2014.052.

[23] E. Noumon Allini, M. Skórski, O. Petura, F. Bernard, M. Laban, and V. Fischer, 'Evaluation and Monitoring of Free Running Oscillators Serving as Source of Randomness', *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. Volume 2018, pp. 214-242 Pages, Aug. 2018, doi: 10.13154/TCHES.V2018.I3.214-242.

[24] B. E. Grantham and M. A. Bailey, 'A Least-Squares Normalized Error Regression Algorithm with Application to the Allan Variance Noise Analysis Method', in *2006 IEEE/ION Position, Location, And Navigation Symposium*, Coronado, CA: IEEE, 2006, pp. 750–756. doi: 10.1109/PLANS.2006.1650671.

[25] L. Benea, M. Carmona, F. Pebay-Peyroula, and R. Wacquez, 'On the Characterization of Jitter in Ring Oscillators using Allan variance for True Random Number Generator Applications', in *2022 25th Euromicro Conference on Digital System Design (DSD)*, Maspalomas, Spain: IEEE, Aug. 2022, pp. 534–538. doi: 10.1109/DSD57027.2022.00077.

[26] L. Benea, M. Carmona, V. Fischer, F. Pebay-Peyroula, and R. Wacquez, 'Impact of the Flicker Noise on the Ring Oscillator-based TRNGs', *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2024, no. 2, Art. no. 2, Mar. 2024, doi: 10.46586/tches.v2024.i2.870-889.

Towards Sustainable Electronic Design Automation Flow: A Joint Approach Based on Complexity Metrics

Virginie Delalot
INNOVA Advanced Technologies
2 Rue Emile Augier
38000 Grenoble, France
virginie@innova-advancedtech.com

Chouki Aktouf
INNOVA Advanced Technologies
2 Rue Emile Augier
38000 Grenoble, France
chouki@innova-advancedtech.com

Gilles Fritz
INNOVA Advanced Technologies
2 Rue Emile Augier
38000 Grenoble, France
gilles@innova-advancedtech.com

Bastien Gratreaux
INNOVA Advanced Technologies
2 Rue Emile Augier
38000 Grenoble, France
comm@innova-advancedtech.com

Nermine Ali
Universite Paris-Saclay, *CEA, List*
Palaiseau, France
nermine.ali@cea.fr

Lilia Zaourar
Universite Paris-Saclay, *CEA, List*
Palaiseau, France
lilia.zaourar@cea.fr

Abstract—**This paper addresses sustainability criteria and Electronic Design Automation (EDA) needs. We aim to optimize the operational stages of an EDA Flow and address a series of investigations to reduce carbon footprint. Our main purpose is to optimize the design flow, considering sustainability criteria to reduce the environmental impact of EDA tools. First, metrics correlating sustainability and design project complexity are provided and implemented as part of an EDA design solution, which INNOVA Advanced Technologies proposes. Second, the INNOVA design solution provides job scheduling based on sustainability criteria. Typical case studies are provided in this paper.**

Index Terms—EDA, Sustainability, Eco-design.

I. INTRODUCTION

In recent years, the global drive toward an environmentally sustainable society has increasingly relied on advancements in the semiconductor industry. Historically, this industry has prioritized making chips smaller, faster, and more energy efficient, often overlooking environmental impacts.

The rapid increase in chip production is responsible for considerable environmental effects, including the environmental impacts of the different design phases [1]. Hence, the ecological dimension of a chip design is becoming strategic. Therefore, the association of eco-design metrics with traditional power/performance/area (PPA) design metrics is no longer optional for next-generation integrated circuits.

As computing technology and digital accessibility continue to expand, the carbon footprint of the Information and Communication Technology (ICT) sector—measured in *CO*2 emissions—is projected to exceed its current 1.2-2.4% share of global emissions [2]. In 2022 alone, there was a significant rise in digital adoption, with 192 million new internet users, 85 million new mobile phone users, and 300 additional hyperscale data centers expected to be operational by 2024 [3].

Indeed, in digital and electronics in general, it is known that the most polluting stages are the extraction and production of raw materials. The production of most of our mobile phones, computers, and televisions accounts for almost 80% of the greenhouse gas emissions that these devices will produce during their lifetime [4]. Some studies deal with the life cycle as a whole and focus on these stages [5] [6] [7] [8] or on recycling and reuse [9] [10]. However, upstream, the design of electronic components also has an impact, and notions of ecological impact can also guide the choices made for this design. Indeed, there are no small savings when it comes to environmental footprint.

Despite advances in technology scaling and Electronic Design Automation (EDA) that have led to the creation of energy efficient VLSI systems, the overall environmental impact has continued to rise over the past decade. This increase is primarily due to the carbon emissions associated with chip design and manufacturing, known as the embodied carbon footprint. To reduce the environmental footprint of electronics and computing devices, it is essential to develop new tools that enable designers to make informed sustainability decisions throughout the design process. Every step, from EDA tools and design flows to manufacturing, is crucial in enhancing sustainability.

The future of electronics lies in sustainable EDA and manufacturing steps. This approach prioritizes reducing environmental impact and maximizing energy efficiency throughout the product life cycle. Incorporating eco-friendly principles into EDA tools empowers engineers to design

electronic circuits with lower power consumption, sustainable materials, and optimized resource utilization. This approach directly tackles the pressing issue of electronic waste (e-waste) and associated carbon footprints, ensuring adherence to evolving environmental regulations and promoting responsible business practices. Moreover, sustainable EDA facilitates environmentally conscious and socially responsible product development.

Today, typical System on Chips (SoC) design projects can involve the efforts of tens to even hundreds of engineers, with an average project design time of nine to twelve months [11]. Costly design resources are usually required: design automation tools and libraries, third-party intellectual-property cores, and computing server farms for data and design jobs processing. Another equally critical component is the increased focus on sustainability and, more precisely, design resource management: the need to adopt more environmentally responsible design practices. In parallel, during the design of complex chips, cost reduction is becoming a real challenge for small, medium, and large companies. Resource management is a key to containing design cost [12].

However, despite the growing pressure among stakeholders to control the environmental effects of chip design, no standards or international agreements to formalize these mitigation efforts have yet been established. Nevertheless, in line with ISO environmental recommendations [13], eco-design in the microelectronics industry must consider the impact of the full life cycle and adopt a systematic approach to design, especially for complex systems such as SoCs.

With climate change accelerating, it is important to measure and reduce the environmental impact of all areas of human activity. This is particularly true for the production of electronic components, as the number of electronic devices has increased enormously in recent decades; this trend looks set to continue [4] [14], and the planet's resources are limited. Recently, various studies have presented several sustainability methods and approaches in electronics and semiconductors. However, most of the scientific contributions target electronic product life cycles.

The main methodology and criteria for measuring the impact of a product or service is Life Cycle Assessment (LCA). This involves analyzing various stages in their life, from design to end-of-life, via the extraction of raw materials, production, distribution, and use. It enables us to identify the environmental impact stages (consumption of fossil resources, consumption of rare earths, greenhouse gas emissions, water consumption, pollution of natural environments, etc.) and to find solutions to reduce this impact. Life-cycle analysis involves many players throughout the product value chain, such as the various service providers and suppliers, and even beyond, by including consumers or users, right through to recovery and recycling. Nevertheless, these studies need to address the impact of EDA tools and flow for chip design on the process itself.

Inspired by the principles of environmental sustainability—namely *"reduce"* and *"reuse"*—this work explores the potential for creating a Sustainable EDA flow. We propose tools incorporating sustainability considerations in chip design through optimal design space exploration with environmental impact as a key criterion and within the EDA flow. Our approach combines complexity metrics with strategies to reduce the computational effort involved in designing each component of the SoC.

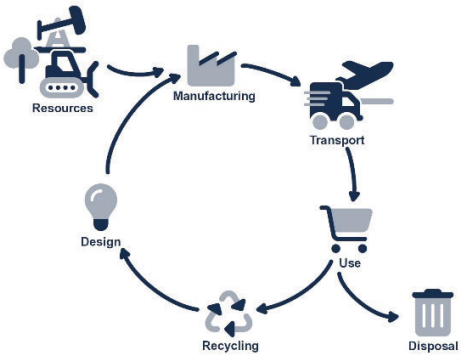


Fig. 1. Life Cycle Assessment.

We introduce *SoC Planner*, a concept to develop a new generation of design exploration solutions for modern System on Chips in a collaborative project. The main objective is to push the design one step further through automation and optimal planning for efficiency in the design process. This project intends to help in the automatic exploration and optimization of SoC design configurations based on several criteria, such as power, performance, and area, by considering the ecological dimension during the design process.

Energy consumption analysis directly impacts climate change. Such analysis correlates to selecting and assigning design resources when starting a new design project. This is particularly true for integrated circuits with large production volumes. During the chip design process, computing servers are used that have a high calculation capacity and significant power consumption of several megawatts per design flow execution. Each tool used for design, verification, or simulation might need to be used for several days, significantly impacting power consumption. Further, many tasks — up to 20% — can fail without yielding convincing results. Climate change mitigation requires using all possible means to avoid wasting energy, especially on tasks that may prove useless.

This paper is organized as follows. Section II presents an in-depth SoC Planner project detailing its primary tools and functionalities. Section III exposes the INNOVA EDA solution for metrics evaluation, followed by Section IV, which discusses the scheduling and allocation of resources based on eco-design criteria. Finally, Section V concludes the paper and discusses future works.

II. SOC PLANNER

With the miniaturization of components, transistor density is increasing faster than the productivity of system-on-chip designers and the improvement of EDA tools. Therefore, the

SoC Planner Project aims to provide system-on-chip designers with a solution to the explosion in design costs, particularly for advanced nodes. This project intends to develop an innovative and indispensable reference software suite for all hardware designers, enabling automatic configuration and planning of the next generation of system-on-chips, guaranteeing the best compromise regarding cost, performance, and eco-design criteria.

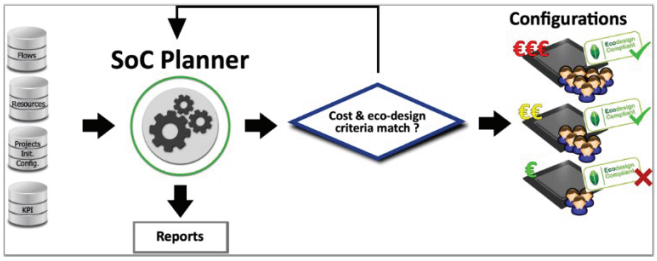


Fig. 2. SoC Design Planning with Sustainability Measures.

The main novelty of the *SoC Planner* project is to limit the number of possible configurations for a complex system-on-chip from the very start of a new design project. It is usual to have hundreds of thousands or even millions of possible configurations for a single project. *SoC Planner* innovation will drastically reduce this number to a set of relevant configurations in line with the scope of the designer specifications at early stages. This will significantly reduce the number of design cycles (iterations of the product development) and, consequently, in development cost savings. An eco-design dimension is also taken into account to enable precise measurement of environmental impact at the start of a new system-on-chip design project, and the minimization of this criterion in the same way as the usual performance criteria (consumption, time, surface area).

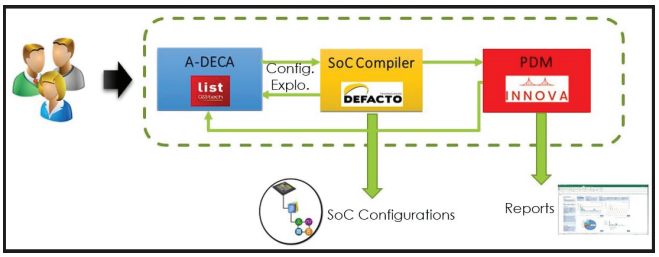


Fig. 3. SoC Planner design flow.

SoC Planner addresses sustainability at the early stages of SoC design by integrating eco-friendly principles throughout every step of the design process. It starts with the ADECA (Automated Design space Exploration for Computing Architectures) tool, which automates the exploration and optimization of computing architecture designs to find the most suitable configurations and guides designers toward more energy-efficient and resource-aware solutions. Next, thanks to the SoC Compiler tool, we automate the generation of RTL and the integration of IPs into a larger SoC, streamlining this process and reducing errors. Finally, the design flow optimization is achieved through INNOVA Project Management (PDM), the design management platform that helps reduce costs by simplifying flow and resource management for SoC and complex electronic systems. The following sections detail the tools: SoC Compiler, A-DECA, PDM.

A. A-DECA framework:

A-DECA aims to address the complexity involved in designing many-core processors by using a modular and automated methodology for the exploration of design parameters [15].

This methodology concentrates on several key aspects by breaking down the challenges of designing and optimizing a complex computing architecture into distinct domains, each equipped with suitable representations and tools to address its specific sub-problem. A-DECA is implemented to be modular and versatile, not restricted to a specific set of parameters imposed by the simulator. Its modularity allows it to adapt to different architectural requirements, ensuring its relevance across diverse applications from High-Performance Computing to Artificial Intelligence IPs and embedded systems. Integrating eco-design principles emphasizes sustainability, guiding the design process to minimize environmental impact while maintaining high performance. Here are the key features of its implementation:

A-DECA places a strong emphasis on sustainability by incorporating eco-design principles. A-DECA inherently promotes sustainability and environmental friendliness by focusing on configurations that minimize area and reduce energy consumption. This approach ensures that the resulting processors are powerful and efficient and align with eco-friendly design principles, reducing their overall ecological footprint and aligning technological advancements with environmental stewardship.

B. SoC Compiler

Defacto Technologies proposes a front-end SoC integration tool called SoC Compiler [16]. It is a software gamechanger for front-end SoC design. This platform addresses the challenges of integrating increasingly complex SoCs with a high level of automation, all before the logic synthesis stage. It excels at managing a multidimensional design flow, considering various aspects like RTL code, pre-designed IP blocks (described by IP-XACT), timing constraints, power consumption, physical layout plans, and test requirements. This unified approach ensures that the SoC design considers all these factors, leading to a more optimized and robust end product at the early stages. This tool provides significant advantages, including achieving optimal Power-Performance-Area (PPA) targets within tight schedules. It accomplishes this through extensive automation of SoC integration tasks before logic synthesis. SoC Compiler promotes design reuse of up to 90% by leveraging existing RTL code and design collateral. Its integration with existing EDA

tool flows is facilitated through support for high and low-level APIs in different programming languages (Tcl, Python, Perl, C++, Java). This software enables SoC creation even in incomplete design blocks or views. SoC Compiler is used most of modern chip design, offering a sophisticated and efficient path to design highly integrated, performance-optimized, and reliable circuits for various applications, including automotive systems and industrial devices. Its ability to establish complex design processes is a key element in advancing integrated circuit technology.

C. Project and Design Management (PDM) solution

INNOVA's Project and Design Management (PDM) is a unified software platform integrating project management presented in figure 4, design flow, and resource management into a single software environment. PDM is thus multi-user, opening to a wide audience: design project managers, design engineers, purchasing departments, and human resources. INNOVA's PDM provides the infrastructure to track the usage of different resources (EDA tools, servers, engineering resources, libraries, etc.). It can be securely plugged into any existing IT environment and inter-operates with standard project, license, and server management tools [17].

INNOVA approach is to provide automated measures, which help detect and filter design and resource configuration that directly impact power consumption and design project sustainability in general. INNOVA PDM design solutions measure if sustainability conditions are fulfilled and help differentiate between different configurations. Configuration means design options, resource options, and design flow altogether. Therefore, the functionalities and access rights to PDM are distributed between these users. PDM key features are summarized as follows:

- It is scalable or extensible, that is to say, it is easily interfaced with existing information systems;
- It ensures consistent data throughout the life of a project, including design information, design assets, design flows, and proprietary project management tools;
- It ensures real-time synchronization with design data.

Being fully compatible with existing software, such as user selected design software and directory management software, PDM serves as a single portal, thereby reducing the complexity of using dedicated design tools and environments. This platform is based on a unique representation model of the different entities handled (software, flows, servers, project, human resources), which ensures interoperability between the three key areas: management of design flows, management of resources, and project management. In addition, this technology can integrate both software aspects, such as design software licenses, and hardware aspects, such as calculation servers, particularly regarding flow management and control. Design entities can include a project, a design stream, design data, or a design resource, such as a server or a license for design software. The design process for a complex SoC requires unified project management, with strong links among the entities involved (see figure 4). Such a management process allows full traceability of the use of design resources.

Therefore, integration with PDM contributes to achieving the overall design flow optimization goals, as presented in the next section.

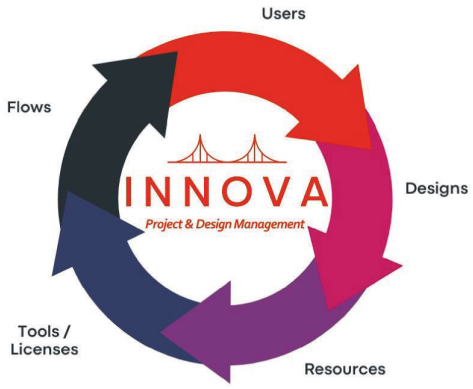


Fig. 4. Innova Unified Design and Project Management design environment.

III. ECO-DESIGN AND SUSTAINABILITY METHODOLOGY BASED ON PDM

INNOVA initiated the work to develop the eco-design and sustainability methodology, which will be validated and adopted during the "*SoC Planner*" project. This methodology is based on the following structuring elements:

- The development of an ecological footprint that will represent a decision-making aid for designers of complex SoCs;
- The prediction of failure or success of design tasks using a learning engine which will limit the waste of resources due to the launch of design tasks whose probability of success is quite low;
- The allocation of resources taking into account eco-design criteria;
- Mapping and Scheduling of design tasks taking into account eco-design criteria.

A. Generation of an ecological footprint/score

The generation of an ecological footprint accompanying the different design stages is at the heart of the eco-design methodology of INNOVA'PDM. Such a footprint will help design engineers make task launch decisions related to the impact of resource consumption and, therefore, energy. It is noteworthy that PDM already provides several complexity metrics that target design configuration, design flow configuration, design resource configurations, etc. An adequate combination of these metrics reflects a project's complexity. It provides a meaningful score to deem an SoC project highly,

medium, or low complex. This directly impacts the project's sustainability and its related impact on power consumption. We illustrate these metrics here through the typical use case presented below.

B. Typical Case Study – Flow complexity metrics

The flow complexity metric is part of other metrics of an overall sustainability score. This score is considered to compare the complexity of different design projects. Here, the focus is on design flows and their related complexity. The main objective is to be able to compare different versions of the same design flow. In other words, for one design flow, how complex are the changes if we decide to split or partition a complex SoC design into 2, 3, 4, or 5 partitions/parts towards logic synthesis.

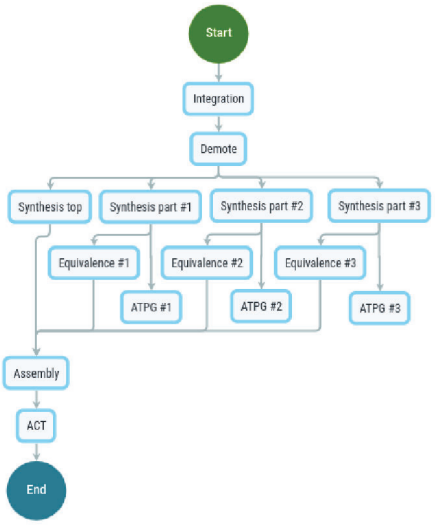


Fig. 5. Typical flow use cases for multi-partitions SoC design.

Designers know this as parallel vs. serial synthesis. Two different metrics are computed: cyclomatic complexity and N-path metrics. These two metrics reflect the graph complexity for design flows. Table I below provides the related complexity values. The first column gives the number of steps, and the next two are the metrics Cyclomatic complexity and N-Pat, respectively.

TABLE I

DESIGN FLOW : COMPLEXITY METRICS.

Metrics:	Cyclomatic complexity	N-Path
2 parts	7	10
3 parts	9	22
4 parts	11	50
5 parts	13	114

As we can see in the graph of Figure 6 below, several parts directly impact the flow complexity: as expected, cyclomatic complexity shows that we increase the number of linearly independent paths - each new part adds two linearly independent paths; N-path shows that we have increased number of possible paths exponentially. Indeed, a new step has two potential paths, success or failure, multiplied by other possible paths. Finally, the weighted average shows that the needed resources are directly proportional to the number of parts.

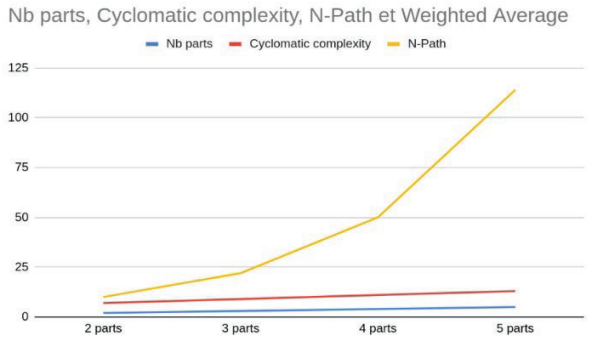


Fig. 6. Evolution of complexity compared to number of parts and number of paths.

IV. SCHEDULING AND ALLOCATION OF RESOURCES BASED ON ECO-DESIGN CRITERIA

Traditionally, design tasks are executed according to the rule of *"First In, First Out"* without any consideration of the availability of resources (servers, design software licenses, etc.). INNOVA PDM shortens the execution times of design tasks thanks to optimized management of resources (licenses, servers), reducing waiting times to access such resources. Thus, by having an overall view of the tasks to be executed and their need in terms of resources, it is possible to schedule these tasks in such a way as to reduce blockage scenarios and bottlenecks due to lack of resources, as illustrated in the figure 7 below. Indeed, the first schedule takes the tasks in the arrival order (red, blue, then yellow) and executes them as soon as resources are available. The second one identifies that executing the red tasks first induces long waiting times and will focus on implementing the blue tasks first. One of the key points for task scheduling to be most effective is estimating the duration of a task. INNOVA's PDM, through automatic learning from previous executions, is a central element for the automatic estimation of task execution time.

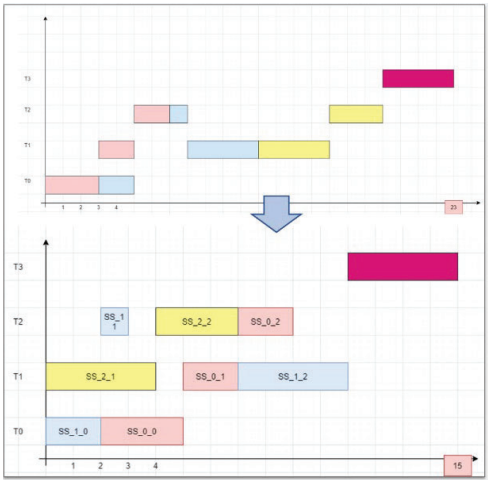


Fig. 7. Automated management of design tasks and jobs based on resource availability.

As illustrated in figure 7, when design jobs are executed based on available resources, the resources consumed during a design project will be positively impacted. For example, a longer execution for the logic synthesis design step (usually called silicon compilation) can mobilize up to ten servers, which results in a consumption of around $40kWh$. Consumption management at all levels will thus enable significant energy performance for complex digital integrated circuits. With a global view of the design tasks and the resource requirements, design decisions can be made based on resource availability, yielding more predictable project outcomes. Such visibility opens new options for design and resource allocation and scheduling, with the ability to modify the list of design tasks and shift or cancel unnecessary ones. Managers can now order design tasks based on resource availability and define the priorities for each task. Design managers and decision-makers can focus design teams on those tasks related to higher-priority and strategic projects.

V. CONCLUSION

The future of electronics lies in sustainable EDA and manufacturing steps. In this work, we incorporate eco-friendly principles into EDA tools to empower engineers in the design of electronic circuits with lower power consumption, sustainable materials, and optimized resource utilization. In a collaborative project, we introduce *SoC Planner*, a concept to develop a new generation of design exploration solutions for modern System on Chips. The main objective is to push the design one step further through automation and optimal planning for efficiency in the design process. Modern project management methodologies are required to address current resource management challenges during the SoC design process, enabling design entities to manage design projects jointly and with the resources necessary. Beyond cost, eco-design is a dimension of SoC design that EDA tools can no longer ignore. This work has presented a new EDA design approach and solution that considers sustainability key when qualifying an SoC design project. The strategy is structured around complexity metrics and job scheduling based on resource availability criteria. As a perspective for this work, new metrics will be added to the current INNOVA design solution.

ACKNOWLEDGMENT

The SoC Planner Project has received funding from the French institution BPI France.

REFERENCES

[1] G. Marinova and A. Bitri, "Challenges and opportunities for semiconductor and electronic design automation industry in post-covid-19 years," *IOP Conference Series: Materials Science and Engineering*, vol. 1208, p. 012036, 11 2021.

[2] M. G. Bardon, P. Wuytens, L.-A. Ragnarsson, G. Mirabelli, D. Jang,˚ G. Willems, A. Mallik, A. Spessot, J. Ryckaert, and B. Parvais, "Dtco including sustainability: Power-performance-area-costenvironmental score (ppace) analysis for logic technologies," in *2020 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 2020, pp. 41–4.

[3] J. C. Herlin, "Kaupunkimarkkinointi tiktokissa," 2022.

[4] (2020 - 2023) Ademe-arcep study: assessment of the digital environmental footprint in france in 2020, 2030 and 2050. [Online]. Available: https://en.arcep.fr/uploads/tx gspublication/press-kit-study-Ademe-Arcep-lot3 _march2023.pdf

[5] T. Pirson, T. P. Delhaye, A. G. Pip, G. Le Brun, J.-P. Raskin, and D. Bol, "The Environmental Footprint of IC Production: Review,Analysis, and Lessons From Historical Trends," *IEEE Transactions on Semiconductor Manufacturing*, vol. 36, no. 1, pp. 56–67, Feb. 2023, conference Name: IEEE Transactions on Semiconductor Manufacturing. [Online]. Available: https://ieeexplore.ieee.org/document/9979766

[6] B. Li, R. B. Roy, D. Wang, S. Samsi, V. Gadepally, and D. Tiwari, "Toward Sustainable HPC: Carbon Footprint Estimation andEnvironmental Implications of HPC Systems," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, Nov. 2023, pp. 1–15, arXiv:2306.13177 [cs]. [Online]. Available: http://arxiv.org/abs/2306.13177

[7] S. B. Boyd, *Life-cycle assessment of semiconductors*. University of California, Berkeley, 2009.

[8] F. Taiariol, P. Fea, C. Papuzza, R. Casalino, E. Galbiati, and S. Zappa, "Life cycle assessment of an integrated circuit product," in *Proceedings of the 2001 IEEE International Symposium on Electronics and the Environment. 2001 IEEE ISEE (Cat. No.01CH37190)*, 2001, pp. 128–133.

[9] U. Gupta, M. Elgamal, G. Hills, G.-Y. Wei, H.-H. S. Lee, D. Brooks, and C.-J. Wu, "ACT: designing sustainable computer systems with an architectural carbon modeling tool," in *Proceedings of the 49th Annual International Symposium on Computer Architecture*, ser. ISCA '22. New York, NY, USA: Association for Computing Machinery, Jun. 2022, pp. 784–799. [Online]. Available: https://doi.org/10.1145/3470496.3527408

[10] C. Fitzpatrick, J. Walsh, and I. Grout, "Environmentally superior implementation of electronic hardware through modular programmable logic devices & eco design," in *Proceedings of the 2006 IEEE International Symposium on Electronics and the Environment, 2006.*, 2006, pp. 228– 232.

[11] J. A. Darringer, R. A. Bergamaschi, S. Bhattacharya, D. Brand, A. Herkersdorf, J. K. Morrell, I. I. Nair, P. Sagmeister, and Y. Shin, "Early analysis tools for system-on-a-chip design," *IBM Journal of Research and Development*, vol. 46, no. 6, pp. 691–707, 2002.

[12] T. Li, J. Hou, J. Yan, R. Liu, H. Yang, and Z. Sun, "Chiplet heterogeneous integration technology—status and challenges," *Electronics*, vol. 9, no. 4, 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/4/670

[13] International Organization for Standardization (ISO): Geneva, Switzerland, "Iso 14001:2015; environmental management systems - requirements with guidance for use." 2015. [Online]. Available: https://www.iso.org/standard/60857.html

[14] F. Bordage. (2019) The environmental footprint of the digital world. [Online]. Available: https://www.greenit.fr/wp-content/uploads/2019/11/GREENIT EENM etude EN accessible.pdf

[15] L. Zaourar, A. Chillet, and J.-M. Philippe, "A-DECA : an Automated Design space Exploration approach for Computing Architectures to develop efficient high-performance many core processors," in *DSD/SEAA 2023 - 26th Euromicro Conference Series on Digital System Design (DSD) and 49th Euromicro Conference Series on Software Engineering and Advanced Applications (SEAA)*, Durres, Albania, Sep. 2023, pp. 756–763. [Online]. Available: https://cea.hal.science/cea-04224485

[16] Front-end soc integration. [Online]. Available: https://defactotech.com/products-solutions/soc-integration-at-rtl

[17] [Online]. Available: https://www.innova-advancedtech.com/

An Open-source HLS Fully Parameterizable Matrix Multiplication Library for AMD FPGAs

Angelos Athanasiadis
School of Electrical and Computer Engineering,
Aristotle University of Thessaloniki,
54124, Thessaloniki, Greece
angelathan@ece.auth.gr

Nikolaos Tampouratzis
Department of Industrial Engineering and Management,
International Hellenic University,
57400, Sindos, Greece
ntampouratzis@ihu.gr

Ioannis Papaefstathiou
School of Electrical and Computer Engineering,
Aristotle University of Thessaloniki,
54124, Thessaloniki, Greece
ygp@ece.auth.gr

Abstract—**One common characteristic of High-Performance Computing (HPC) and Cyber-Physical Systems (CPS) is their need for heterogeneous energy-efficient solutions. In this work we present a library for FPGA-accelerated dense matrix multiplication which is flexible, open-source, written in purely synthesizable C and has no dependencies on the actual hardware implementation tools. Our library is designed so as to support arbitrary array sizes and accuracy, making it a versatile and adaptable solution that meets the diverse computational requirements of applications all the way from CPS to HPC. Our approach provides an adaptable solution that efficiently exposes the flexibility and performance of the FPGAs to both novice and expert developers which is not the case with the black-box libraries provided by the FPGA manufacturers. Our approach has been evaluated in a number of state-of-the-art AMD FPGAs; the end results demonstrate that the presented implementations can achieve 9x, 34x and 3x gains, in terms of energy efficiency, when compared with embedded, high-end CPUs and GPUs respectively. Moreover, our solution matches or slightly outperforms the most advanced similar FPGA-tailored approach while also being much more flexible and designer-friendly while also library-independent.**

Keywords-High-Performance; Computing; Neural Networks; Matrix Multiplication; AMD FPGA; Vitis

I. INTRODUCTION

In recent years, the complete computing continuum has undergone a significant transformation, fueled by the growing demand for computing power in a range of domains, such as scientific simulations, machine learning, and data analytics. At the same time FPGAs have become a captivating solution to tackle the increasing demand for computing power at a relatively low power envelope, thanks to their parallel processing capabilities and flexibility.

One very widely used function in numerous CPS and HPC applications is the multiplication of dense matrices (e.g. crucial for linear algebra computations). The efficient execution of dense matrix multiplication on FPGAs has been the subject of extensive research [1], [2] because it has a direct significant impact on the overall computational throughput and energy efficiency of FPGA-based HPC and embedded systems.

Although several optimization techniques for both CPU and GPU architectures have been presented [3], [4], a comparable set of guidelines and principles for code optimizations in High-Level Synthesis (HLS) design flows has yet to be established. Furthermore, due to the low clock frequency, lack of cache, and fine-grained configurability of FPGAs, naive HLS implementations, very often, have low performance and relatively high-power consumption and require significant transformations so as to surpass the multi/many core CPUs and GPUs in terms of speed and/or energy efficiency.

This paper presents a generic open-source solution for HLS design flows allowing for the implementation of high-performance dense matrix multiplication on modern AMD FPGAs. Specifically, the contribution of this paper can be summarized as follows:

- An **open-source library**[1] designed to accelerate dense matrix multiplication of any size and datatype by extracting parallelism in two dimensions. This purely Synthesizable C library provides very high configurability and flexibility in order to take full advantage of the resources of modern AMD FPGAs **without any dependency** on the hardware implementation tool (e.g. version of the tools) or any **external library**.
- An innovative flow that enables designers to easily develop FPGA-accelerated applications, that involve matrix handling, using the presented fundamental structures minimizing the development and verification time while achieving high performance and energy efficiency.
- The effectiveness and performance of this library have been rigorously and comprehensively evaluated when implemented on both small and high-end FPGAs and it has been proved that our solution outperforms CPUs, GPUs and even relevant FPGA-tailored approaches.

The presented library will be part of a larger library for efficient matrix handling (which is under development) while it has already been used in a full-precision Convolutional Neural Network (CNN) acceleration framework implemented in multiple FPGAs.

II. RELATED WORK

Ahmad and Pasha [2] investigated several optimization techniques for hardware-accelerated general matrix multiplication on FPGAs, with a specific focus on CNNs. In contrast to [2], our approach provides a comprehensive library

[1] https://github.com/angelosathanasiadis/Gemm-HLS-Fully-Parametrizable

that empowers users to customize FPGA-based matrix multiplication to their unique computational requirements, irrelevant of the application domain. Although Haghi et al. [5] present a reconfigurable FPGA assistant for in-network computations, with an accompanying case study on distributed matrix multiplication, their study focuses on reconfigurable compute-in-the-network FPGA assistance for collective support. Our work is orthogonal to this since it mainly aims at taking full advantage of the hardware resources of a single FPGA while providing adaptability for i) implementation in different FPGA devices and ii) easy integration with design flows which focus on distributed FPGAs.

De Fine Licht et al. [1], [6] presented their research results on transformations of HLS code for HPC and flexible communication respectively. However, their approach has certain limitations : i) they are dependent on the HW implementation tool (e.g. they can be used only up to AMD Vitis 2021.1[2]), which limits their use and ii) they rely on a hardware library developed by the authors, tailored to the FPGA implementation tool, which introduces a certain tool dependency and thus undermining the adaptability and scalability of their approach, while they do not exploit state-of-the-art HBM2 memories. In comparison to those studies, we present a methodology that is independent of specific hardware tools and libraries and is implemented purely in C with HLS pragmas. We, thus, establish a more robust and sustainable solution for enhancing computational performance in HLS designed FPGA systems.

Moving to the GPUs area, several studies have explored the use of cuBLAS for efficient matrix multiplication on GPUs, as well as optimizations for power efficiency and fault tolerance [3]. For AMD GPUs, rocBLAS high-performance library has been developed for matrix operations exploiting the specific architectural features of AMD hardware. Authors in [4] present a robust framework built on rocBLAS that efficiently handles batched matrix multiplications, even with unbalanced input sizes, showcasing rocBLAS's flexibility and efficiency. While our performance metrics indicate slower computation times compared to GPU-based solutions such as cuBLAS and rocBLAS, our approach demonstrates significantly higher energy efficiency. This makes our approach particularly advantageous in power-constrained environments, where energy consumption is critical.

III. Implementation

A. Reference Implementation

In order to demonstrate our optimization flow, we start with a basic reference matrix multiplication implementation, in which we adopt a simple effective HLS pipeline approach aiming at maximizing the computational efficiency on the targeted FPGAs. The algorithm comprises of nested loops that meticulously traverse matrices A, B, and C, and calculate the dot product of a row from matrix A with a corresponding column from matrix B. The key element is the HLS pipeline directive, which allows the execution of one multiplication and one addition in a single clock cycle achieving a relatively high throughput and performance. However, the main drawback of this method is the excessive memory accesses to external DRAMs & HBMs (i.e. in total M*N*K accesses in all arrays) preventing further parallelization.

B. Vectorization

In traditional FPGA matrix multiplication implementations, the frequent accesses to external DRAMs have been a serious performance bottleneck since each DRAM access adds latency and consumes valuable energy. In our approach we propose the use of 512bit vector elements so as to dramatically reduce the number of DRAM accesses. This reduction is achieved by aggregating multiple data points into a single, wide uint512_t element, effectively consolidating data transfers and parallel computations. The use of uint512_t elements results in improved FPGA Block RAM (BRAM) utilization since by using wider data elements we increase the utilization of each BRAM memory. In addition, since modern FPGAs are connected to multiple memory banks with dedicated channels (e.g., multiple DRAM modules or HBM lanes), we split data transfers into a parameterizable number of memory banks/lanes to increase the effective external memory bandwidth.

C. Innovative Optimization flow

Moreover, as illustrated in Figure 1, we utilize the internal BRAMs in conjunction with HLS streams, to optimize also the on-chip memory access patterns and further increase the computational efficiency. We allocate BRAM resources for matrix BRAM_B and use HLS streams for matrices A and C. Stream_A and Stream_C_in are used to read matrices A and C respectively from external memories (DRAMs/HBMs) and Stream_C_out to transfer the data from the internal computation modules back to the external memories.

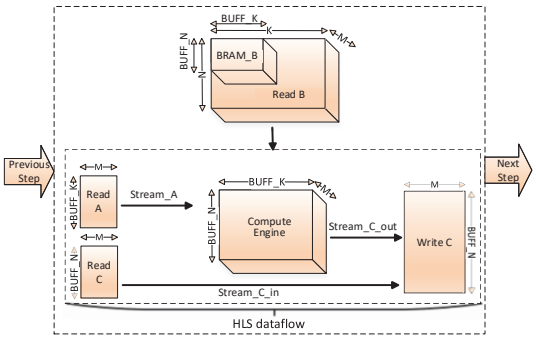


Figure 1: Architecture of Innovative Optimization flow

The aforementioned technique leads to a significant decrease in the effective latency associated with off-chip memory retrieval, thereby diminishing idle time caused by external memory latency; Listing 1 presents the optimized external memories total accesses which are at least 3 orders of magnitude less than the reference implementation. In addition, streams enable a continuous flow of data between processing elements, without the need for intermediary storage in BRAMs, minimizing the latency and resource overhead. By leveraging streams, data are transferred directly between producer and consumer processes, facilitating pipelined execution and

[2] https://github.com/spcl/gemm_hls/issues/25

enhancing parallelism. This direct transfer mechanism reduces the need for on-chip memory for temporary storage, leading to more efficient resource utilization and higher throughput. Furthermore, streams can handle variable data rates more effectively ensuring that processes are not overwhelmed by the rate of data production.

Our optimization approach is based on four (4) fully configurable parameters: buffer sizes BUFF_K and BUFF_N, which are used for the on-chip memory utilization for arrays B and C, as well as loop unrolling parameters UNROLL_N and UNROLL_K, which show the level of unroll that will happen in each dimension. These parameters enable us to customize the architecture to fully utilize any FPGA architecture/size and memory technology/topology. Through the careful selection of buffer sizes and unrolling factors, we can enhance memory access patterns and computational parallelism, achieving an optimal balance between resource usage, in case there are additional modules that are also placed on the same device, and throughput.

```
initialize BRAM_B[BUFF_K][BUFF_N]
initialize BRAM_C[BUFF_N]

for ex_k from 0 to K with step BUFF_K:
  for ex_n from 0 to N_512 with step BUFF_N:
    // Read B - K*N/VECTOR Accesses
    for k from 0 to BUFF_K, n from 0 to BUFF_N:
      BRAM_B[k][n] = DRAM_B512[];
#pragma HLS DATAFLOW
    // Read A - M*K*N/(VECTOR*BUFF_N) Accesses
    for m from 0 to M, k from 0 to BUFF_K/VECTOR:
        stream_A << DRAM_A512[]
    // Read C_in - M*K*N/(VECTOR*BUFF_K) Accesses
    for m from 0 to M, n from 0 to BUFF_N:
        stream_C_in << DRAM_C512[]
    // Main Loop
    for m from 0 to M:
      for k from 0 to BUFF_K/VECTOR:
      #pragma HLS pipeline
      //Perform Efficient Parallel MM Computation
      //UNROLL_N*UNROLL_K*VECTOR elem. in 1 cycle
      for n from 0 to BUFF_N:
        stream_C_out << BRAM_C[n]
    // Write C
    for m from 0 to M, n from 0 to BUFF_N:
        DRAM_C512[] << stream_C_in + stream_C_out
```

Listing 1: Innovative Memory Access

To provide further clarification, as illustrated in Figure 2, Stream_A is read in uint512_t quantities in order to reduce latency even further. After that, Stream_A and BRAM_B are fed to the computational part in order to calculate the outcomes. The outcomes for the whole BUFF_K dimension are stored in BRAM_C array and then streamed through Stream_C_out in WriteC-sized blocks.

Moreover, we strategically utilize the #pragma HLS pipeline in the matrix multiplication computation tile. This directive plays a critical role in enabling fully pipelined processing for the entire tile. In terms of the implemented Synthesizable C code, each iteration of the loops becomes a distinct processing step executed within a single clock cycle. In that respect we reduce pipeline stalls, guaranteeing an uninterrupted stream of data to the multiplication and addition hardware and thus achieving efficient parallel fully-parameterizable processing of **UNROLL_N*UNROLL_K*VECTOR elements in a single (1) clock cycle**. Finally, our library can further parallelize the kernel operations by allocating matrices among the different Sliced Logic Regions (SLRs) of the recent AMD Alveo devices achieving even better performance and energy efficiency.

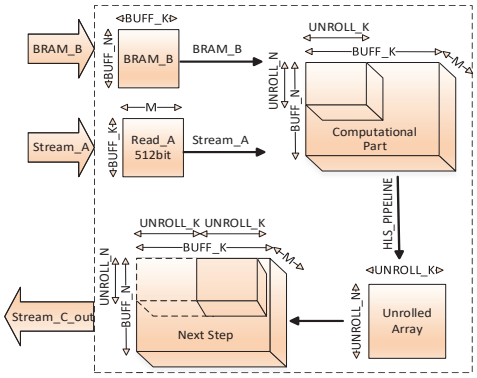


Figure 2: Architecture of Compute Engine

IV. EVALUATION

Table 1: Results with multiple matrix dimensions

M	K	N	U55 [GFLOPS]	U55 [Exec.]	KR260 [GFLOPS]	KR260 [Exec.]
512	1024	2048	207	0.0104s	24	0.89s
1024	2048	4096	215	0.08s	22.6	0.76s
1024	4096	2048	219	0.078s	22.6	0.76s
2048	2048	2048	215	0.08s	22.3	0.77s
2048	4096	16384	229	1.2s	22	12.5s
2048	16384	4096	249	1.1s	22	6.25s
4096	4096	4096	229	0.6s	22	12.5s

We have evaluated the efficiency of our open-source, fully parameterizable, purely-C library for dense matrix multiplication when implemented in numerous FPGA boards. In order to comprehensively evaluate the efficacy and robustness of our proposed matrix multiplication approach, multiple experimental runs were executed across varying dimensions as presented in Table 1. The experimental results demonstrate that we can achieve high performance in diverse dimensional configurations which highlights the wide applicability of our approach. The maximum performance achieved is 249 GFLOPS for matrix sized of M=2048, K=16284 and N=4096 on a high-end FPGA (AMD Alveo U55C) and 24 GFLOPS for matrix sized of M=512, K=1024 and N=2048 on an embedded one (Kria KR260), while the performance is also relatively high in smaller matrix sizes, in contrast to the GPU implementations.

To demonstrate further the effectiveness of the presented approach we compare the non-optimized reference model, the fully optimized model on both a high-end FPGA (AMD Alveo U55C) and an embedded one (Kria KR260), the parallel execution of matrix multiplication using OpenMP in a multicore CPU and using CUDA on an NVIDIA T4 GPU. In all experiments the array dimensions which are M=2048, K=4096, and N=16384 are selected so, as to be different from our optimal configuration, demonstrating also the flexibility of our approach. Furthermore, the results obtained from numerous experiments with different dimensions are fully inline with those presented in Figure 3. As illustrated in Figure 3, our implementation in the embedded FPGA is two orders of magnitude faster than the reference implementation and 9x times faster compared to the fully parallelized algorithm executed on an embedded ARM 4-core CPU (Cortex-A53); those numbers include all the memory accesses and the external memory technologies and topologies are exactly the same in both cases. Similarly, our fully optimized approach when implemented on the Alveo U55C board is appro-

Table 2: Comparison to previous FPGA implementations

	Device	Logic Util. BRAM	Logic Util. DSPs	Perf. FP32 [GFLOPS]	Perf. FP64 [GFLOPS]	Power Effic. [GFLOPS/W]	Tool/Library Independency	Open Source
D'Hollander [7]	Zynq 7000	32%	99%	5	-	-	×	×
Guan [8]	Stratix V	67%	17%	100	-	2.92	×	×
gemm_hls [6]	Alveo U200	58%	44%	211	74	8.49	×	✓
This work	Alveo U55C	56%	44%	229	80	9.02	✓	✓
This work	Kria KR260	94%	60%	22	10.1	2.44	✓	✓

ximately three orders of magnitude faster than the reference implementation and 10x times faster compared to the fully parallelized algorithm executed on an Intel Xeon E5-2620 v4 (8 cores). In order to further compare the overall efficiency of the presented approach with that triggered by the software implementation, the GFLOPS/Watt for each implementation were also measured. Based on our measurements we achieve 34x and 9x higher energy efficiency than the best CPU parallel implementation in an Alveo U55C and a Kria KR260 respectively. Moreover, our design, when implemented on the Alveo, is 3x more power efficient than the CUDA implementation on an NVIDIA T4 GPU which is also implemented on a better CMOS technology (12nm for T4 vs 16nm for U55c).

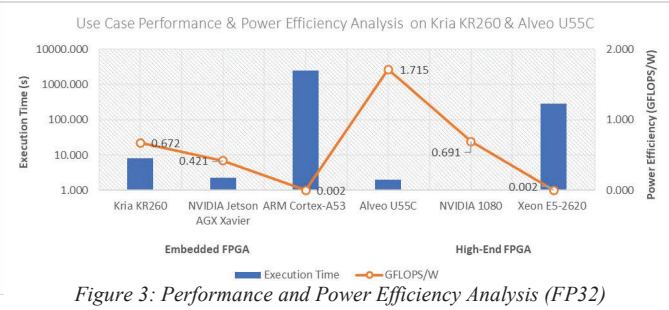


Figure 3: Performance and Power Efficiency Analysis (FP32)

Table 2 presents the comparison of our approach with other FPGA-tailored similar systems. From those, the only open source widely used library for dense Matrix Multiplication is the one in [6] so we tried to implement it in our reference modern FPGAs. However, even though it is an open-source library, it has rather limited applicability because it requires AMD Vitis v2021.1 or older as also referred in Section II. As a result, we implemented it in the largest FPGA supported by this version of the tool we had available (i.e. Alveo U200). In addition, [6] utilizes a specialized hardware library, which also makes it less flexible, than our purely C-based approach. As shown in the table, our design achieves 9.02 GFLOPS/Watt, while gemm_hls' implementation triggers 8.49 GFLOPS/Watt for the same utilization percentage (229 GFLOPS compared to 211 GFLOPS from gemm_hls).[3]

More importantly, our methodology has been developed with a focus on simplicity and wide compatibility, eliminating the need for any specific external libraries and HW implementation tools. The designer-friendliness and the flexibility of our approach enables even non-advanced designers to seamlessly develop matrix-multiplication modules, probably within broader systems (e.g. CNNs or Deep Neural Networks).

V. Conclusions and Future Work

This paper presents an open-source, FPGA-tailored library for dense matrix multiplication that is implemented in purely synthesizable C, thus providing very high flexibility and adaptability. The easily customizable parameters allow users to maximize resource utilization and performance and/or energy efficiency for any given FPGA device from low resources to high-end ones. Our evaluation demonstrates that our library outperforms both embedded and high-end multi-threaded CPUs and GPUs. The pioneering nature of the tool is highlighted by the absence of a similar open-source solution, positioning it as a valuable resource for those looking for easily programmed, yet high-performing FPGA acceleration.

VI. Acknowledgements

This work is supported by the Key Digital Technologies Joint Undertaking (KDT-JU) under the powers delegated by the European Commission and its members, including the top-up funding by National Authorities in the context of the REBECCA (Reconfigurable Heterogeneous Highly Parallel Processing Platform for safe and secure AI) project (grant agreement #101097224).

VII. References

[1] J. de Fine Licht, M. Besta, S. Meierhans, and T. Hoefler, "Transformations of high-level synthesis codes for high-performance computing," IEEE Trans. Parallel Distrib. Syst., vol. 32, no. 5, pp. 1014–1029, May 2021.

[2] A. Ahmad and M. Pasha, "Optimizing hardware accelerated general matrix-matrix multiplication for cnn on fpgas," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 67, pp. 1–1, 2020.

[3] S. Wu, Y. Zhai, J. Liu, J. Huang, Z. Jian, B. Wong, and Z. Chen, "Anatomy of a high-performance semi-automatic fine-tuned tolerance on gpus," in Proceedings of the 27th International Conference on Supercomputing, ser. ICS '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 360–372.

[4] R. Wang, Z. Yang, H. Xu, and L. Lu, "A high-performance batched matrix multiplication framework for gpus under unbalanced input distribution," The Journal of Supercomputing, vol. 78, no. 2, p. 1741–1758, Jun 2021.

[5] P. Haghi, A. Guo, T. Geng, J. Broaddus, D. Schafer, A. Skjellum, and M. Herbordt, "A reconfigurable compute-in-the-network fpga assistant for high-level collective support with distributed matrix multiply: case study," IEEE Conference on Field Programmable Technology.

[6] J. de Fine Licht, G. Kwasniewski, and T. Hoefler, "Flexible communication avoiding matrix multiplication on fpga with high-level synthesis," ser. FPGA '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 244–254.

[7] E. H. D'Hollander, "High-level synthesis optimization for blocked floating-point matrix multiplication," SIGARCH Comput. Archit. News, vol. 44, no. 4, p. 74–79, Jan 2017.

[8] Y. Guan, H. Liang, N. Xu, W. Wang, S. Shi, X. Chen, G. Sun, W. Zhang, and J. Cong, "Fp-dnn: An automated framework for mapping deep neural networks onto fpgas with rtl-hls hybrid templates," 04 2017, pp. 152–159.

[3] It is very difficult to compare the actual resources in each FPGA since AMD is listing differently the resources for the Alveo U200 and the U55c boards (i.e. CLBs and Registers vs System Logic Cells).

Immersive Environments with Haptic Technology for the Control of an Industrial Robotic Arm

Wilman Paucar
Ingeniería Electrónica
Universidad Politécnica Salesiana
Quito, Ecuador
wpaucarp@est.ups.edu.ec

Gustavo Caiza*
Ingeniería Electrónica
Universidad Politécnica Salesiana
Quito, Ecuador
*gcaiza@ups.edu.ec

William Oñate
Ingeniería Electrónica
Universidad Politécnica Salesiana
Quito, Ecuador
wonate@est.ups.edu.ec

Morelva Saeteros
Institute of Technology
University of Deusto
Bilbao, España
morelvasaeteros@opendeusto.es

***Abstract*— The technological advancement of Industry 4.0 and the Internet of Things has brought significant changes and benefits to the industry, allowing observation, control, and decision-making in industrial processes from anywhere in the world and in real-time. This article presents the power of a Mitsubishi RV-2AJ Industrial Robotic Arm using haptic technology "Senso Glove" and real-time 3D visualization, enabling users to interact and control the arm's movements. For the implementation, data is processed in Unity and sent to the "Firebase" cloud, where this data is requested by an embedded system to subsequently be converted into commands for arms control, specifically in the coordinates "X", "Y", "Z," and in the opening or closing of the gripper. The results demonstrated that the arm can be controlled from the gloves, opening various applications. Additionally, an accuracy of 87.8% was achieved in the conducted tests, with an average communication time of 815 milliseconds.**

Keywords*— *Senso Glove, Mitsubishi RV-2AJ Robotics Arm, Internet of Things, Unity.

I. Introduction

There is an overwhelming technological advancement and computational systems [1] that strengthen the Internet of Things, cloud computing, Big Data analytics, cybersecurity, simulation, digital transformation, and artificial intelligence, among others, which are integral to progress. These directly impact the final cost of our product [2]. Thus, Industry 4.0 aims for sustainability and improvement in production processes.

Robotic arms are also constantly evolving to provide their best services based on needs [3], handling hazardous tasks for operators such as managing radioactive elements [4], and ensuring precision and accuracy in certain processes, for example, in the pharmaceutical or automotive industry. In their basic form, they are composed of links, joints, and their end effector, the latter being a highly versatile tool that depends on the task the arm performs [5]. Motors in the joints allow the rotation or translation of the links. With the advancement of computing, they become more robust, acquiring diverse capabilities and designs, enabling the creation of complex geometries with high precision [6], and implementing advanced control techniques.

An example is the development of the modeling of the 5-axis Mitsubishi RV-2AJ robotic arm in SolidWorks by [7]. It stands out for not risking the actual robotic arm to conduct functional tests and make necessary corrections for its optimal operation. Another way to control the arm is remote, as implemented by [8], who, through the internet and an internal network within the laboratories, achieve manipulation of the RV-2AJ arm to conduct practical exercises for students. In real-time, students have a video feed and can observe the arm's manipulation based on parameters they input into the software externally. The human-robot communication interface has been increasingly developed [9]. They devised an interface where sensors can capture electromyographic signals from the human body's bioelectrical signals, allowing the generation of commands for robotic arm manipulation.

Wireless gloves (Senso Glove) enable the detection of every hand movement [10]. They contain sensors that transmit information about the current state of the fingers, palm, and wrist. These gloves are continually being updated, with the information sent via radio frequency to the computer to be extracted and simulated in a virtual environment.

Currently, the Internet of Things is booming, referring to the interconnection of intelligent devices to a global network that links the physical and digital worlds. There is a diversity of interconnected smart devices [11], leading to the emergence of smart homes. In these homes, appliances, sensors, and actuators are monitored and controlled remotely through the Internet [12].

Thus, the industrial RV-2AJ arm is controlled using wireless gloves (Senso Glove) and its respective simulation. The computer receives data wirelessly transmitted by the gloves, which is then processed in simulation software (Unity) and sent to the cloud (Firebase). Subsequently, the data is retrieved by an

electronic card (ESP32) to determine the instruction to be sent to the RV-2AJ arm and execute the corresponding action. This action may involve movement along the vertical or horizontal axis or maintaining a stationary position. Simultaneously, the virtual arm is configured to replicate the same movements as the physical arm.

II. Previous Concepts

A. Mitsubishi Melfa RV-2AJ Robotic Arm

It is of the angular or articulated type, small, compact, and powerful, featuring five joints, each with a specific range of motion [13]. The robotic arm is equipped with brushless AC servo motors, and its end effector is a pneumatic gripper designed to hold objects weighing up to 2 kg [14]. It finds applications in the medical field for sample handling and in education for tasks like palletizing or recreating a two-dimensional drawing, among others. There are three ways to control it [15]: through manual control using the R28TB, by executing a Melfa IV program in the arm's controller, or externally by sending commands through RS-232 serial communication.

B. Senso Glove

It is a hardware device (wireless glove) that enables human-robot interaction by capturing hand movements [10]. Utilizing inertial measurement unit sensors, it features eight sensors communicating at a speed of 400 Hz, with a latency of 15 ms. Additionally, it includes a controller for virtual reality or augmented reality and vibration motors for user feedback [16]. The sensors incorporate a gyroscope, allowing the measurement of orientation in three dimensions, including the pitch, roll, and yaw angles for each finger, palm, and wrist. Moreover, they provide information about the speed at which our fingers or the entire hand is moving.

C. Firebase

It is one of Google's cloud-related products that provides services such as [17] storage, hosting, instant messaging, user self-identification, and real-time NoSQL database. In the case of the NoSQL database functionality, it synchronizes all clients in real time, providing each with an instance to automatically receive updates. It is used in multiple applications, including web services, mobile applications, and sensor data, implementing data encryption security [18]. This involves the use of a digital certificate that authenticates identity, enabling a secure connection.

D. Internet Of Things

Technological advancement is overwhelming, being in the fourth industrial revolution [19], defined as the connection of sensors and actuators to the internet, sending real-time information to assess changes and control objects or phenomena in both physical and digital processes. Additionally, objects interact with each other, and their information is collected and evaluated. To achieve this, high-speed internet, data storage, and computing power are essential [20], leading to the use of cloud computing technology. Its applications are diverse, ranging from video security in communications, automobile navigation systems, telemedicine, production systems, home automation [21], and more. It is a comprehensive multidisciplinary system, ensuring the security and reliability of data transmission. While the initial implementation cost is high, it provides accessibility from anywhere in the world.

III. Methodology

A. Developed Proposal

Perform control of the Mitsubishi RV-2AJ arm through the Senso Glove (Figure 1). The wireless glove operates in the 868 MHz ISM band. The information it sends to Unity (version 2021.3.19f) is processed based on the arm's movements and sent to the cloud using one of the Firebase Realtime Database services that utilizes the WebSocket protocol. At the moment it is sent, it is also processed for the digital arm, which depends on this data for its movement. Subsequently, the sent information will be read by the ESP32 electronic card, which interprets and sends commands serially for the movement or activation of the gripper of the RV-2AJ arm. Below is a detailed breakdown of each configuration segment to achieve the proposed objectives.

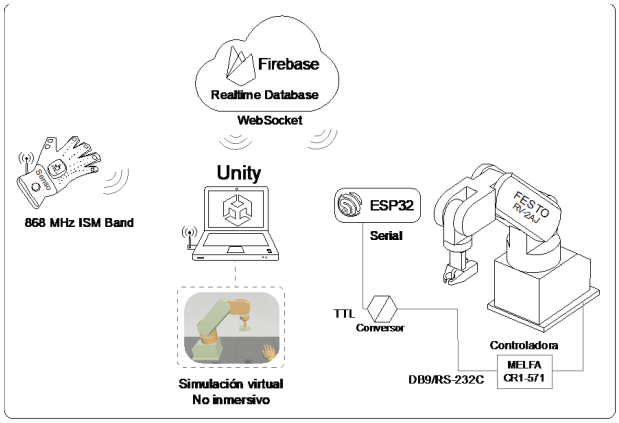


Fig. 1. Design of control and simulation using the Senso Glove for the Mitsubishi RV-2AJ arm.

B. Virtual Design

In this case, a 3D CAD design software is used to create a virtual arm that replicates the key features of the physical arm, including its base, links, and end effector, while adhering to its dimensions to ensure similarity.

Upon importing it into the Unity scene, it will be scaled using the Transform scale function: 0.05 for proper visualization. It is important to note that to emulate the movement of the physical arm, it must be constructed in a way that allows its joints to be programmatically controlled. For this purpose, the SolidWorks program was used, which enables the design of each movable piece individually and then assembling them. This assembly is saved in STL format to be imported into Blender software, version 3.1.2, and subsequently exported as an FBX file without any modifications.

Finally, the virtual arm is integrated into the glove scene, creating a scenario for the simulation of both at the same time, as shown in Figure 2

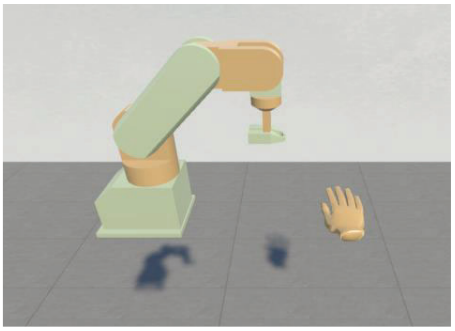


Fig. 2. Virtual integration of the RV-2AJ Arm and the Senso Glove hand, initial positions.

C. Glove Calibration

Firstly, to acquire data, communication is established between the glove and the computer through the Senso_DK3_GUI software. The calibrated glove and its connection, receiving real-time sensor data, can be observed. This communication occurs via radio frequency in the 868 MHz band. Upon initiating the software, it allows the user to calibrate the glove in case it is misconfigured, ensuring that hand movements align coherently with the received data through an animated visualization.

D. Communication

In the following Figure 3, the information flow is identified, starting from the gloves that have wireless communication with the PC. The data is processed within the Unity program through its C# scripts, structured for interpretation, and then sent to the Firebase Realtime Database. Subsequently, the electronic board will constantly evaluate the changes in the database and process them into corresponding commands for the RV-2AJ arm. It is worth noting that there is electronic hardware to accommodate the communication voltage levels between the ESP32 (0 to 3.3 V) and the Melfa CR1-571 arm controller (-15 to 15 V). This results in an action on the arm, whether it is to move, stop, or open and close the gripper.

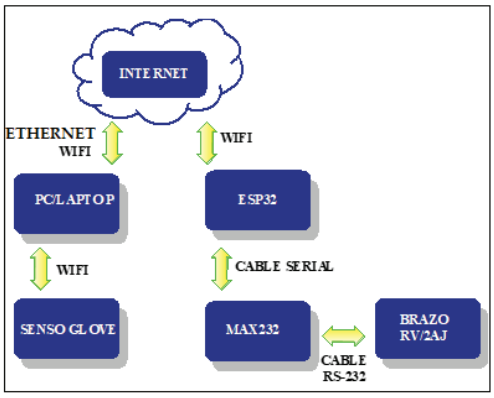


Fig. 3. Communication Diagram for RV-2AJ Arm Control.

E. Programming

In the Unity software, using the C# language, classes, methods, and variables from the freely accessible Senso gloves package (SensoHandSDK.unitypackage) are interpreted. These elements help obtain information sent by the gloves, allowing for the creation of new scripts based on specific needs. In this case, the scripts are developed to control both the physical RV-2AJ arm and its digital counterpart. The obtained ranges are associated with movements in two axes and the actions of the end effector. These relationships also guide the movements of the digital arm, such as moving from left to right, from top to bottom, or vice versa, and opening or closing the end effector.

In the following (see Figure 4), the MoveXY class diagram illustrates the analysis of data obtained from the SensoHandExample class, specifically data related to the palm and middle finger. The acquired information is segmented to conceptualize it according to our requirements, which, in this case, determines an action in the RV-2AJ arm. This allows for control of movements along the "Y" and "Z" coordinate axes and the opening and closing of the end effector gripper.

Additionally, the corresponding data is sent to the Firebase database to be read as needed. For the MoveArm class, control was implemented for the digital arm, which mimics the movement of the physical arm. Necessary calculations are performed to utilize Euler functions, ensuring the normal development of joint and link movements. Unity's nesting object property is also emphasized during this process.

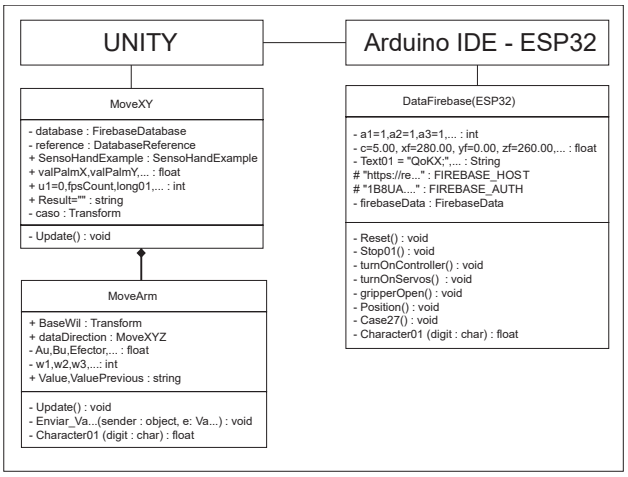


Fig. 4. Control System, Classes for Sending Information of the Digital and Physical Arm.

The real-time location of the physical arm can be requested, and this information is sent by the arm in a character string containing the corresponding values of its current location. The respective functions are used to obtain these values, making them usable locally or externally.

IV. Results

A. Data Presentation

In Table I, two segments of 30 samples each are summarized, one when communication is optimal and the other when there is an initial delay in the received signal intensity. These data were extracted from the Senso_DK3_processing Windows console, which is activated when the gloves communicate with the computer.

TABLE I. Quality of Received Signal Intensity

Sample	FPS	rssi	Delay	Wi-Fi RSSI
1ra.	67/68	-63	2,013	Very Good
2da.	48/68	-79	4,420	Good

Next, with the information obtained in the Unity software from the inertial IMU sensors includes the interpretation of movement concerning the coordinate axes "X," "Y," and "Z" of the rotation angles Pitch, Roll, and Yaw, respectively.

B. Communication Times

In Figure 5, option a) shows the gripping actuation sampling time from the moment the signal is sent from the Senso Glove, with an average of 850 ms. The data is transmitted from the gloves, processed in Unity, sent to the Firebase cloud, and then read by the ESP32, which finally encodes it to open or close the gripper. For option b), the box-and-whisker plot illustrates the median with a value of 800 ms.

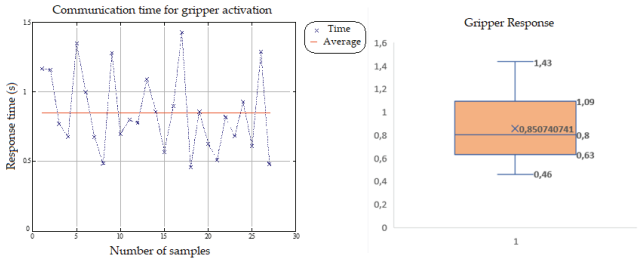


Fig. 5. Gripper Response from the Data Processing by Unity.

For option a) in Figure 6, the time it takes to react to movement along the "Y" or "Z" axes is shown, with an average of 780 ms. Similarly, the data is sent to the Firebase cloud to be later read by the ESP32 and transformed into the respective command for movement. For option b) in Figure 6, the median is observed with a value of 760 ms.

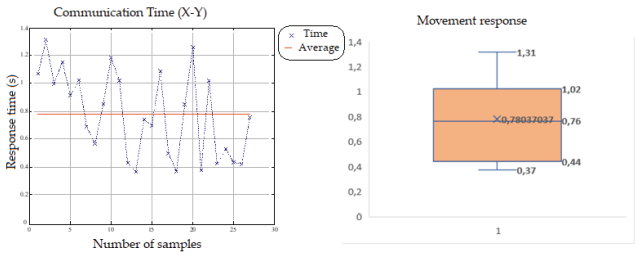


Fig. 6. Response of the Arm Movement when the Data is Processed from Unity.

Table 2 shows the gripper opening with an effectiveness of 92%, based on a sample of 25 consecutive repetitions. Likewise, for the respective movements along both axes, there is an average of 83% effectiveness between them.

TABLE II. EFFECTIVENESS IN THE MOVEMENT AND GRIPPER OPENING BASED ON 25 REPETITIONS.

Position	Up	Down	Left	Right	Gripper
Number of times obtained.	22	21	20	20	23
Effectiveness.	88%	84%	80%	80%	92%

C. Results

In Figure 7, the non-immersive virtual environment of the arm and glove with their corresponding physical parts can be observed. In this case, the arm is moving to the right, both physically and digitally, due to the interaction of the Senso Glove, which is in the right position from its initial point.

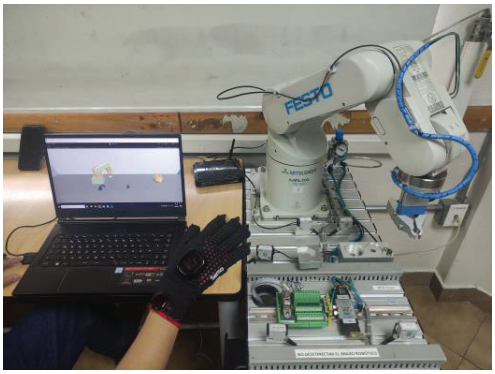


Fig. 7. Overall Visualization of the Physical and Virtual Environment.

Figure 8 displays the physical and virtual environment of the robotic arm for different positions. The end effector, with its digital and physical gripper opening, is observed. It is controlled from the glove, enabling the operator to perform arm movements based on hand motion.

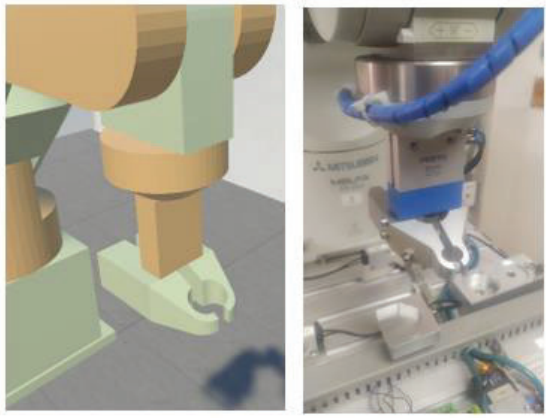


Fig. 8. Initial Position of the RV-2AJ Arm and Gripper Opening.

V. DISCUSSION

In this study, control of an industrial robotic arm was achieved through Haptic technology, cloud communication, and visualization in a virtual environment. It was observed that the use of Haptic technology provides a more user-friendly control, and the system exhibits high precision in various movement tests. It was observed that the calibration process of the Senso Glove is crucial to determine the working environment and thus ensure the proper functioning of the arm, which can then be visualized in the virtual environment.

The results indicated that there is a variability of 70 ms in the response times for both movement and gripper, with the averages in both cases. The medians are below the averages, indicating asymmetric box plots. This time depends on the processor of the machine running the programs. The focus of

this research is on the Internet of Things (IoT) with outcomes for non-critical processes, leading to a considerable latency based on the obtained results.

VI. Conclusions

Controlling a robotic arm through haptic technology and visualizing its digital representation in real-time is achievable through the integration of various technologies such as IoT, 3D simulation, haptic technology, and communication protocols. There is a delay in the data flow from Unity to ESP32 due to the transmission or reception to the Firebase cloud and the repetition loops in both Unity and ESP32, resulting in an average of 850 ms for gripper actuation and 780 ms for response to movement on its axes. It should be noted that the time is directly related to the processing power of the machine and can be improved by using a PC with better specifications.

In the control system, specifically in opening or closing the gripper, there is an average effectiveness of 87.8%, with higher values obtained in samples of 15 or 20 consecutive repetitions. This value depends on the movements given to the arm because interaction involves multiple axes, and the response times of the arm are slower than those of the hand due to its manufacturing characteristics and degrees of freedom.

References

[1] Kumar, V., Vrat, P. & Shankar, R. (2022). Factors Influencing the Implementation of Industry 4.0 for Sustainability in Manufacturing. Glob J Flex Syst Manag 23, 453–478. https://bibliotecas.ups.edu.ec:2582/10.1007/s40171-022-00312-1

[2] García, C. A., Caiza, G., Guizado, D., Naranjo, J. E., Ortiz, A., Ayala, P. & Garcia, M. (2023). Visualization of Key Performance Indicators in the Production System in the Context of Industry 4.0. IFAC-PapersOnLine, 56(2), 6582-6587. https://www.sciencedirect.com/science/article/pii/S2405896323006663

[3] Yenorkar, R. & Chaskar, U. (2018). GUI Based Pick and Place Robotic Arm for Multipurpose Industrial Applications. Second International Conference on Intelligent Computing and Control Systems (ICICCS), 200-203.

[4] Barrientos, A., Peñín, L. F., Balaguer, C. & Aracil, R. (2007). Fundamentos de Robótica. Madrid. McGRAW-HILL, 2da. Edición.

[5] Harikrishnan, M., Ghanshsyam, T., Chourasia, G., Das, A., Shrivastava, A. & Bhatt, Z. (2018). Flamen − 7 DOF Robotic Arm to Manipulate a Spanish Fan. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 4152-4157.

[6] Park, J. & Jung, S. (2023). Form-Finding to Fabrication: A Parametric Shell Structure Fabricated Using an Industrial Robotic Arm with a Hot-Wire End-Effector. Nexus Network Journal, 1522-4600.

[7] Eshah, N. S., Osman, K., & Zainuddin N. (2020). 5-Axis of Mitsubishi RV-2AJ Robotics Arm Modelling Using Solidworks. 16th IEEE International Colloquium on Signal Processing & Its Applications (CSPA), 207-212.

[8] Buitrago J. A., Giraldo F. D. & Lamprea J. A. (2011). Remote access lab for Mitsubishi RV-2AJ robot. IX Latin American Robotics Symposium and IEEE Colombian Conference on Automatic Control, 1-7.

[9] Agurto, J., Bonilla, V., Moya Cajas, M., Mosquera, G., Paredes, R. & Litvin, A. (2019). Mitsubishi RV-2AJ Robot Controled by Electromyographic Signals. International Conference on Information Systems and Computer, 77-82.

[10] Weber P., Rueckert E., Calandra R., Peters J. & Beckerle P. (2016). A low-cost sensor glove with vibrotactile feedback and multiple finger joint and hand motion sensing for human-robot interaction. IEEE International Symposium on Robot and Human Interactive Communication, 99-104.

[11] Peralta, J. & Smarsly, K. (2022). An Introduction and Systematic Review on Machine Learning for Smart Environments/Cities: An IoT Approach. Machine Learning for Smart Environments/Cities. Intelligent Systems Reference Library, V 121. https://doi.org/10.1007/978-3-030-97516-6_1

[12] Ghoul, Y. & Naifar, O. (2023). IoT based applications for healthcare and home automation. Multimed Tools Appl, https://bibliotecas.ups.edu.ec:2582/10.1007/s11042-023-16774-z

[13] Cajamarca, J. A. & Portilla, A. D. (2016). Implementación de un controlador para la cinemática inversa del brazo robot Mitsubishi RV-2AJ a través de una tarjeta ARM y Matlab. Universidad Politécnica Salesiana. 17-18.

[14] Ayob, M. A., Zakaria, W. N., & Jalani, J. (2014). Forward kinematics analysis of a 5-axis RV-2AJ robot manipulator. Electrical Power, Electronics, Communications, Control and Informatics Seminar (EECCIS), 87-92.

[15] Crainic, M. & Preitl, S. (2016). Development of a special setup for path tracking of RV-2AJ robot. IEEE 11th International Symposium on Applied Computational Intelligence and Informatics (SACI), 217-220.

[16] Fahmi, F., Nainggolan, F., Siregar, B., Gaweng, S. & Zarlis, M. (2020). User experience study on crane operator erection simulator using senso glove in a virtual reality environment. IOP Conference Series: Materials Science and Engineering, V.851, 012023.

[17] Le, G.T., Tran, N.M., Tran, T.V. (2021). IoT System for Monitoring a Large-Area Environment Sensors and Control Actuators Using Real-Time Firebase Database. Intelligent Human Computer Interaction, Vol 12616. 3-20.

[18] Moroney, L. (2017). The Firebase Realtime Database. In: The Definitive Guide to Firebase, https://doi.org/10.1007/978-1-4842-2943-9_3

[19] Bucsai, S., Kučera, E., Haffner, O. & Drahoš, P. (2020). Control and Monitoring of IoT Devices Using Mixed Reality Developed by Unity Engine. Cybernetics & Informatics (K&I), 1-8.

[20] Caiza, G., Saeteros, M., Oñate, W. & García M. (2020). Fog computing at industrial level, architecture, latency, energy, and security: A review. Heliyon, V. 6(4).

[21] Caiza, G., Chiliquinga, S., Manzano, S. & García, V. (2020). Software-Defined Network (SDN) Based Internet of Things within the context of low-cost automation. IEEE 18th International Conference on Industrial Informatics (INDIN), 587-591.

Culsans: An Efficient Snoop-based Coherency Unit for the CVA6 Open Source RISC-V application processor

Riccardo Tedeschi
DEI
University of Bologna
Bologna, Italy
riccardo.tedeschi6@unibo.it

Luca Valente
Gianmarco Ottavi
DEI
University of Bologna
Bologna, Italy
{gianmarco.ottavi2, luca.valente}@unibo.it

Enrico Zelioli
Nils Wistoff
IIS
ETH Zurich
Zurich, Switzerland
{ezelioli, nwistoff}@iis.ee.ethz.ch

Massimiliano Giacometti
Abdul Basit Sajjad
PlanV Tech
Munich, Germany
{massimiliano.giacometti, abdul.basit}@planv.tech

Luca Benini
IIS, DEI
ETH Zurich, University of Bologna
Zurich, Switzerland
lbenini@iis.ee.ethz.ch

Davide Rossi
DEI
University of Bologna
Bologna, Italy
davide.rossi@unibo.it

Abstract— **Symmetric Multi-Processing (SMP) based on cache coherency is crucial for high-end embedded systems like automotive applications. RISC-V is gaining traction, and open-source hardware (OSH) platforms offer solutions to issues such as IP costs and vendor dependency. Existing multi-core cache-coherent RISC-V platforms are complex and not efficient for small embedded core clusters. We propose an open-source SystemVerilog implementation of a lightweight snoop-based cache-coherent cluster of Linux-capable CVA6 cores. Our design uses the MOESI protocol via the Arm's AMBA ACE protocol. Evaluated with Splash-3 benchmarks, our solution shows up to 32.87% faster performance in a dual-core setup and an average improvement of 15.8% over OpenPiton. Synthesized using GF 22nm FDSOI technology, the Cache Coherency Unit occupies only 1.6% of the system area.**

Keywords- cache coherency; RISC-V; tightly coupled; CVA6; Culsans; ACE;

I. INTRODUCTION

Symmetric Multi-Processing based on Cache Coherency is critical for computing platforms in high-end embedded systems, such as those used in automotive applications. In this field, RISC-V is rapidly gaining acceptance, and Open-Source Hardware (OSH) platforms based on RISC-V have great potential for overcoming several issues, such as IP cost barriers, supply chain constraints, vendor captivity concerns, and non-recurring engineering (NRE) costs.

Current multi-core cache-coherent open-source RISC-V platforms use custom on-chip communication protocols and automated HDL generation, complicating the integration into third-party Systems on Chip (SoCs). Research platforms like OpenPiton [1] and ESP [2] use directory-based coherence to scale to many cores (> 4). However, the complexity of a distributed directory-based system is overkill for small embedded core clusters, leading to inefficiencies and area overheads. Rocket [3] offers a tightly coupled solution but relies on the Chisel hardware construction language to generate the HDL description, making it hard to develop a verification and integration strategy for SoCs where most of the IPs and system interconnect are based on industry-standard HDLs (e.g., SystemVerilog).

Thus, an open Cache Coherency Unit (CCU) designed for low overhead and high efficiency with a small core count (2-4), easy integration into custom SoCs, and full compatibility with commercial EDA flows has yet to be released. To close this gap, we propose Culsans, an open-source SystemVerilog implementation of a lightweight snoop-based tightly-coupled cache-coherent cluster of Linux-capable CVA6 cores [4], and we demonstrate its integration into Cheshire [5], an open RISC-V platform for domain-specific accelerators plug-in. Our solution implements the MOESI cache coherency protocol via the industry-standard Arm's AMBA ACE protocol, which extends the AXI protocol already supported in CVA6 with additional signals and channels aimed at memory coherency.

This work was supported by the Italian National Centre for HPC, Big Data and Quantum Computing – HPC (CN00000013).

Table 1. MOESI and ACE states mapping to status flags

MOESI	ACE	Valid	Shared	Dirty
Modified	UniqueDirty	1	0	1
Owned	SharedDirty	1	1	1
Exclusive	UniqueClean	1	0	0
Shared	SharedClean	1	1	0
Invalid	Invalid	0	X	X

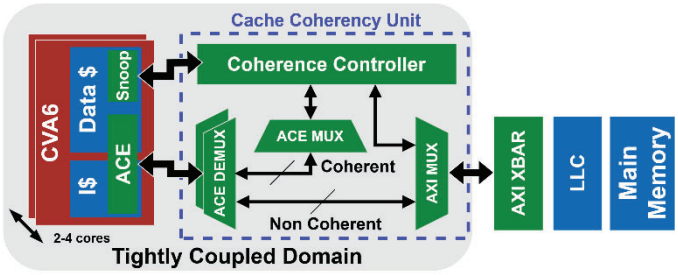


Figure 1. System Level view of the tightly coupled cluster of cores

A CCU was implemented to handle multiple outstanding memory requests in a pipelined fashion. The CVA6 Write-Back (WB) cache subsystem was updated to support AMBA ACE on top of AXI.

Our solution was evaluated using Splash-3 benchmarks [6] against OpenPiton. Under similar cache configurations, the tightly coupled cluster of cores proved to be up to 32.87% faster in a dual-core setup, with an average improvement of 15.8%. Moreover, the system was synthesized using GF 22nm FDSOI technology, and the area occupation of the Cache Coherent Unit amounted to only 1.6% of the overall system.

II. ARCHITECTURE

A. *Memory Hierarchy*

The proposed implementation of a snoop-based tightly coupled cache-coherent cluster of CVA6 cores is reported in Figure 2. The memory hierarchy features L1 Data and Instruction caches interacting with the CCU, while a Last Level Cache (LLC) shared among all the cores is inserted between the tightly coupled domain and the main memory.

B. *CVA6 Data Cache*

On the core side, the pre-existing CVA6 WB data cache was extended to support ACE on top of AXI. The cache line status comprises three flags: valid, shared, and dirty. These additional flags are stored in the cache SRAM along with the cache line tag and data. The combination of the status three bits encodes the MOESI/ACE states, as shown in Table 1.

The WB cache comprises multiple controllers that handle the requests issued by the core, as shown in Figure 2. In particular, the Page Table Walker (PTW), the Load Unit, the Accelerator, and the Store Unit have a dedicated controller each. In addition, the Miss Handler is responsible for miss requests towards the next memory level, Atomic Memory Operations, cache flushes, and writeback operations since it acts as an initiator on the AXI interface of the core. The arbitration on the SRAM's single port read/write port is handled via a statically assigned priority. The

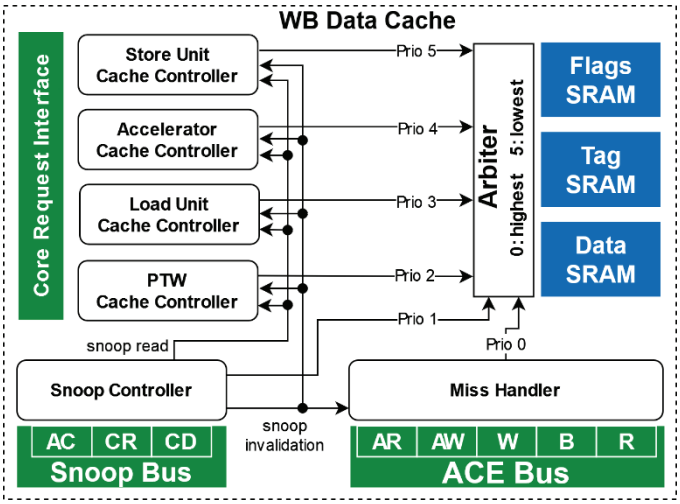


Figure 2. WB Data Cache Controllers with priority scheme and snoop control signals

Miss Handler has the highest precedence, followed by the PTW, the Load Unit, the Accelerator, and the Store Unit.

An additional Snoop Controller was added to the Cache Subsystem to handle transactions on the snoop bus, namely the snoop request channel AC, the snoop response channel CR, and the snoop data channel CD as defined in the ACE protocol. The other cores use this additional snoop interface to access and invalidate cache lines. The Snoop Controller was assigned a priority second only to the Miss Handler to ensure that cache line status updates needed to safeguard coherency are served before any request issued by the core. Additional snoop control signals are propagated to the Miss Handler and the other Cache Controllers to indicate an external invalidation or read request on a given cache line. A Cache Controller requesting unique access (i.e. shared flag equal to 0) to a cache line must ensure that no snoop read is performed concurrently, which indicates a transition to a shared condition of the cache line. Similarly, the Miss Handler must monitor the same event when fetching a cache line for unique access. In addition, a cache line might be invalidated by the Snoop Controller while a Cache Controller has ongoing operations on it, thus a snoop invalidation signal is used to propagate the information on the address being invalidated.

Lastly, the Miss Handler was updated to handle coherent and non-coherent requests via the fields added to the traditional AXI channels to encode the ACE defined transactions. This controller generates both data and invalidation requests towards the CCU.

C. *CVA6 Instruction Cache*

The coherence of the Instruction Cache is required in specific applications where a core can generate instructions to be executed on a different core, such as in the Bao embedded hypervisor [7]. To accommodate this need, the Instruction Cache can be configured via an RTL-level parameter to generate coherent fetch requests and ensure coherency with the data caches of the clustered cores. Otherwise, this feature can be turned off to avoid additional snoop traffic.

D. Cache Coherency Unit

Figure 2 shows the interconnect architecture of the CCU, which was developed as a completely new IP. The memory operations issued by the cores are routed depending on whether they are non-coherent or coherent by the ACE DEMUX block. In the first situation, the request is directly forwarded to the memory interface of the CCU. In the second case, the coherent requests issued by different cores are ordered in the ACE MUX depending on the arrival time according to a round-robin policy and are processed serially by the Coherence Controller, which is also the initiator of the Snoop Bus. Both coherent and non-coherent requests are eventually serialized towards the system crossbar by the AXI MUX block.

The internal organization of the controller is depicted in Figure 3. Three main blocks are present: Decoder, Memory Unit, and Snoop Unit. The Decoder processes the initiator core's write (AW channel) and read (AR channel) requests. Starting from the request address and the ACE coherency transaction issued by the initiator core, the Decoder generates the appropriate snoop transaction towards the remaining cores through the snoop request (AC) and response (CR) channels. The generation of the AC request and the processing of the ensuing CR response are decoupled and mutually nonblocking; thus, a new AC request can be generated without waiting for the previous CR response. A FIFO keeps track of the response order since no transaction ID is associated with the Snoop channels.

Suppose a data snoop is triggered, and one or more cores can provide the needed cache line. In that case, the first responder passes the data to the CCU via the CD data channel, and the response is buffered in the Snoop Unit, which forwards it to the initiator core by generating a burst response on the read response R channel. Similarly, if a snooped core issues a write-back, the CD data is stored in a FIFO inside the memory unit for later processing. The memory unit handles the memory interface of the CCU, either by serving memory operations generated by the initiator core or by generating AXI transactions once a snooped core issues a write-back.

The requests are moved across the different blocks in a pipelined fashion, and control flow is ensured via asynchronous handshakes between the Snoop Unit, the Memory Unit, and the Decoder. Moreover, the inherent channel parallelism of the AXI is leveraged to decouple requests from responses. Serialization of requests is enforced only when multiple requests are targeted to the same cache line. An associative table is used as a Collision Checker to keep track of currently accessed cache lines, and stalling happens if a collision is detected upon a lookup by the Decoder.

III. RESULTS

We evaluated our solution using Splash-3 benchmarks, comparing its performance to OpenPiton in a dual-core setup. Both are based on the CVA6 core, but in our implementation the CVA6 core employs a WB data cache, while in OpenPiton it

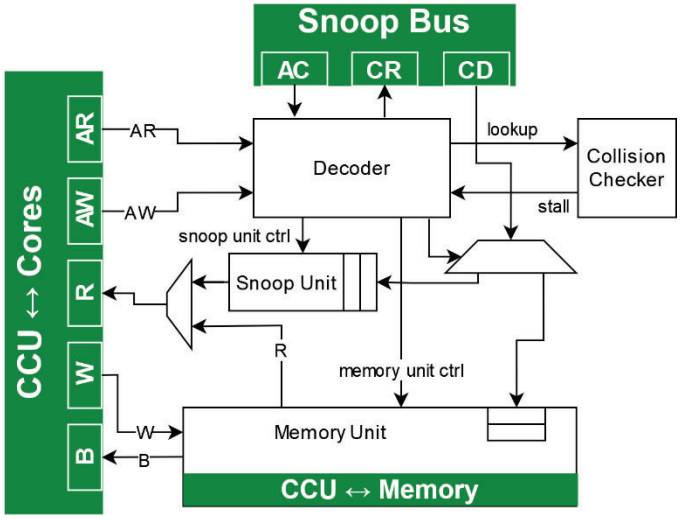


Figure 3. Internal block diagram of the Coherence Controller

uses a Write Through (WT) configuration. As shown in Figure 4, our solution is up to 32.87% faster in a dual-core setup, with an average improvement of 15.8%. These results are mainly due to our solution's tightly coupled design, which incurs less latency than directory-based platforms intended for many cores.

The performance advantage of the CCU varies depending on the benchmark. Table 2 reports the profiling of pipeline stalls and memory operations normalized to the total number of instructions across the different benchmarks. Specific tests, namely FFT and RADIX, do not show improved performance with our snoop-based approach compared to the directory-based OpenPiton because of both their significant number of stalls and fewer memory operations than the other benchmarks. Thus, numerous operand-related pipeline stalls occur, and no advantage ensues from faster coherence transactions. On the contrary, benchmarks such as OCEAN or LU NC are characterized by a significant number of stalls along with a higher number of memory operations and benefit from the proposed changes.

In a first attempt to profile WB and WT caches implemented in CVA6 in a single core setup, we observed that the WB appears to be less performing than the WT, despite expecting from a theoretical point of view a performance advantage of the WB policy over the WT one. A possible explanation stems from the observation that several implementation bottlenecks are present in the available WB cache, leading to sub-optimal handling of transactions. Moreover, the CVA6 core has limited support for multiple outstanding transactions.

We synthesized the design in GF 22nm FDSOI technology using topographical synthesis to assess the area overhead of the additional coherence logic. The CCU area occupation is 1.6% of the total design area, and the coherence logic does not limit the multi-core cluster maximum frequency.

Table 2. Stalls and Memory Operations normalized to the total number of instructions on Splash-3 benchmarks

Benchmark	**Ocean**	**Barnes**	**Chol.**	**Rad.**	**FMM**	**Wat. nsqrd**	**LU NC**	**Wat. Spatial**	**LU Cont**	**FFT**	**Radix**
Stalls/Instr	3.05	0.67	1.07	0.48	0.60	0.90	3.66	0.69	0.97	1.50	2.35
Mem. Op/Instr	0.35	0.44	0.28	0.35	0.21	0.31	0.36	0.32	0.36	0.28	0.23

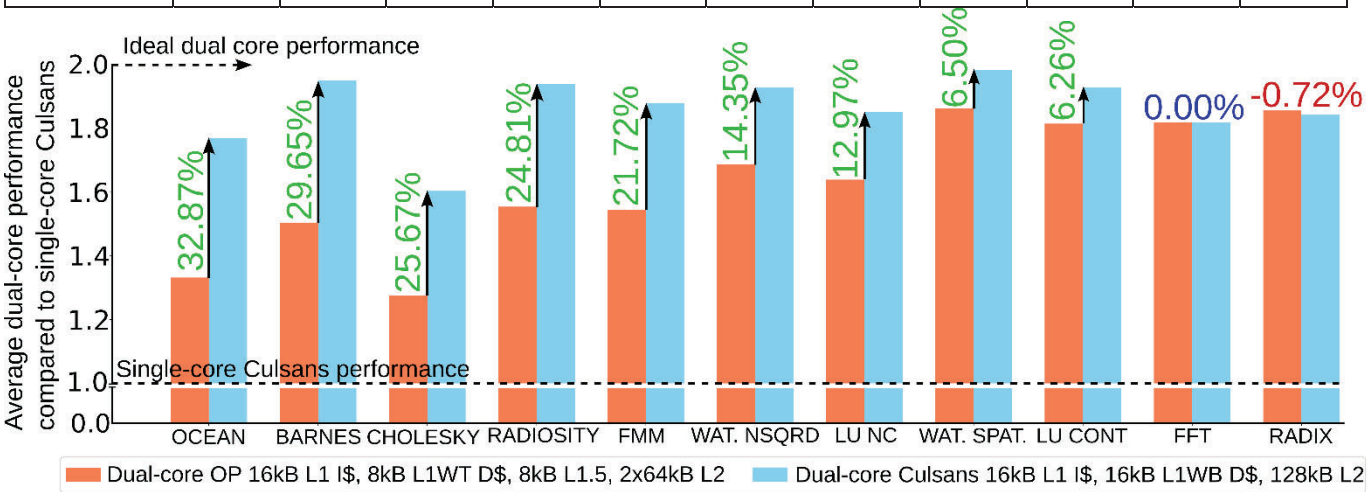


Figure 4. Performance comparison with respect to OpenPiton on the Splash-3 benchmarks

IV. Conclusion

We presented Culsans[1], a snoop-based coherency unit for a tightly coupled cluster of CVA6 Open Source RISC-V application processors. The MOESI protocol is implemented via the industry-standard Arm's AMBA ACE protocol. The proposed architecture was fully integrated into the Cheshire platform and was evaluated against OpenPiton using the Splash-3 benchmarks. Our solution is up to 32.87% faster in a dual-core setup, with an average improvement of 15.8%. The area occupation of the new CCU is less than 2% of the entire dual-core system.

Future developments will focus on supporting advanced non-blocking and performance-oriented caches, such as the HPDCache [8], and more powerful cores, e.g. T-Head 910 [9], thanks to the use of the standardized AMBA ACE protocol. In addition, the Power, Performance, and Area analysis will be extended to larger clusters (4-8 cores). The hardware developed in this work is open-source to support an innovation ecosystem for high-performance, safety-critical embedded systems. Further work focusing on reliability and predictability in an embedded tightly coupled cluster of cores will be enabled by the availability of the proposed platform.

References

[1] J. Balkind, M. McKeown, Y. Fu, T. Nguyen, Y. Zhou, A. Lavrov, M. Shahrad, A. Fuchs, S. Payne, X. Liang and others, "OpenPiton: An open source manycore research framework," ACM SIGPLAN Notices, vol. 51, p. 217–232, 2016.

[2] P. Mantovani, D. Giri, G. Di Guglielmo, L. Piccolboni, J. Zuckerman, E. G. Cota, M. Petracca, C. Pilato and L. P. Carloni, "Agile SoC development with open ESP," in Proceedings of the 39th International Conference on Computer-Aided Design, 2020.

[3] K. Asanovic, R. Avizienis, J. Bachrach, S. Beamer, D. Biancolin, C. Celio, H. Cook, D. Dabbelt, J. Hauser, A. Izraelevitz and others, "The rocket chip generator," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17, vol. 4, p. 6–2, 2016.

[4] F. Zaruba and L. Benini, "The Cost of Application-Class Processing: Energy and Performance Analysis of a Linux-Ready 1.7-GHz 64-Bit RISC-V Core in 22-nm FDSOI Technology," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, pp. 2629-2640, November 2019.

[5] A. Ottaviano, T. Benz, P. Scheffler and L. Benini, "Cheshire: A Lightweight, Linux-Capable RISC-V Host Platform for Domain-Specific Accelerator Plug-In," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 70, pp. 3777-3781, 2023.

[6] C. Sakalis, C. Leonardsson, S. Kaxiras and A. Ros, "Splash-3: A properly synchronized benchmark suite for contemporary research," in 2016 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2016.

[7] J. Martins and S. Pinto, "Bao: A modern lightweight embedded hypervisor," in Proc. Embedded World Conf., 2020.

[8] C. Fuguet, "HPDcache: Open-source high-performance L1 data cache for RISC-V cores," in Proceedings of the 20th ACM International Conference on Computing Frontiers, 2023.

[9] C. Chen, X. Xiang, C. Liu, Y. Shang, R. Guo, D. Liu, Y. Lu, Z. Hao, J. Luo, Z. Chen, C. Li, Y. Pu, J. Meng, X. Yan, Y. Xie and X. Qi, "Xuantie-910: A Commercial Multi-Core 12-Stage Pipeline Out-of-Order 64-bit High Performance RISC-V Processor with Vector Extension : Industrial Product," in 2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA), 2020.

[1] Repository URL: https://github.com/pulp-platform/culsans

Development of a Firearms and Target Weapons Recognition and Alerting System Applying Artificial Intelligence

Jhonny Jimenez
Electronic Engineering
Universidad Politécnica Salesiana
Quito - Ecuador
jjimenez@est.ups.edu.ec

Jhonatan Toscano
Electronic Engineering
Universidad Politécnica Salesiana
Quito - Ecuador
jtoscano@ups.edu.ec

William Oñate
Electronic Engineering*)*
Universidad Politécnica Salesiana
Quito - Ecuador
wonate@ups.edu.ec

Gustavo Caiza
Electronic Engineering
Universidad Politécnica Salesiana
Quito - Ecuador
gcaiza@ups.edu,.ec

***Abstract*—Nowadays, surveillance systems that make use of security cameras are indispensable to ensure the protection and security of companies and organizational entities. These systems operate through monitoring by trained individuals. The progress of a system that uses artificial intelligence to identify and recognize firearms and knives is based on the implementation of machine learning techniques and real-time image and video analysis. The main goal is to increase public safety by accurately and quickly detecting the presence of weapons in various environments. The purpose of this research is to improve public safety through the early detection of threats and more effective responses by security forces. To achieve this, convolutional neural networks have been used. During the development of the system, a database has been created using images and videos containing firearms or knives, based on the "You Only Look Once" (YOLO) algorithm, particularly YOLOv5s.**

Keywords—Deep learning, convolutional neural networks, algorithm, YOLO.

I. Introduction

In modern times, surveillance using Closed Circuit Television (CCTV) video surveillance systems is widely used to avoid dangerous situations, such as robberies involving weapons. However, it has several notable disadvantages, such as a high price, the complexity of its maintenance, and the need for operators to constantly monitor the camera recordings [1]; after an uninterrupted 12-minute period of constant monitoring, a study [2] in "Security Oz Magazine" found that an operator is likely to lose up to 45% of screen activity and failure rate after 22 minutes of monitoring. increases to 95% since inspecting images with human intervention would be labor-intensive as well as being a slow process [3]. This can lead to the waste of vulnerable situations due to fatigue, lack of attention, or the omission of crucial information in surveillance recordings [4] Some of these studies found no impact after the installation of CCTV, while others revealed a significant decrease in the incidence of violent crime [5]; Object classification is the first step in object detection, followed by identification of weapons in images or videos [6]. The focus is to provide security to citizens and use this vital tool for crime prevention and citizen protection, proposing to help reduce the occurrence of violent crimes and provide a video surveillance solution, which will go to a new level, reducing statistics in armed crimes.

Artificial intelligence has challenges that have been developed with automatic solutions focused on the detection and recognition of dangerous elements of video surveillance, with the advantage of precision, consistency, and speed [7]. Technology is clinging to every step of our daily lives, from our phones [8]; to the computer in the identification and detection of objects, for which artificial intelligence has been rede-fined to society in the detection of objects in the world and direct time that tries to improve the work capacity of the comparison of correct and incorrect annotations resulting better than human surveillance. On the other hand [9] presented a model that works in different steps, the initial step is the acquisition of images and the detection of motion, and the last one is the detection of weapons, they carried out experiments in three different models, the first is the model based on Convolutional Neural Networks (CNN), the second is the model based on Fast R, the third is the model based on MobileNet, to analyze its behavior and trends (YOLO) these algorithms being the basis for object detection because of the rapid progress in computer vision skills [10][11][12] as well as having two completely connected layers at the end.

YOLO is like CNN in that it uses convolutional layers and max-pooling layers [13]. It is considered the best and fastest algorithm currently available globally, it is preferred by most AI researchers and enthusiasts thanks to its diversity and faster inference time [14]. The algorithm only analyzes images once, requiring forward programming [15]. From the above, this article aims to delve deeper into the uses of artificial intelligence for weapons recognition systems at the same time, with characteristics that present real-time data and implement machine learning algorithms, to contribute to improving the security of society

II. Methodology

To guarantee the safety of citizens, it is essential to conceive a design for the communication structure that we will use in the creation of a detection system. This process requires the development of a large database for model training, as is

done by virtually all systems that use machine learning as an analytical approach. The procedure begins by obtaining access to Firebase [16] since it is necessary to have a database hosted in the cloud that stores and synchronizes data in real time.

A complete set of data has been prepared for training the weapons detection model, which was obtained from different sources on the web, to provide training information. All this was carried out following the MakeSense protocol, as it offers an extremely intuitive graphical interface [17]. A representation of the image selection process with appropriate quality must be provided. After analyzing the images, it was determined that their quality is ideal for training the network, taking into consideration aspects such as focus, resolution, and luminosity, which allows a more detailed visualization of all the characteristics present in the image.

A. Database

This choice was chosen to provide training examples to the neural network that resemble the situations that can be found during its use, in which the firearm or knife is displayed in complex environments with the presence of numerous nearby objects. The original database consists of 10,354 images and is divided into two classes (firearms and edged weapons), each with approximately 50% of the samples.

First Class – Firearms. For the first class, images are given, which represent illustrations of people participating in shooting exercises, showing common postures of individuals carrying a firearm. These postures turned out to be similar to those found in real robbery situations. Including these images in an environment different from that of a real robbery allows the system to obtain more generalized data, avoiding over-training with the same data.

Second Class – Edged Weapons. The second category of the database comprises images of individuals carrying knives, collected from different web places where these weapons were involved in robbery or were simply found in various positions and contexts. Using the suggested approach, a set of 10,354 images was generated, and from these, the corresponding "tags" were extracted. In these tags, the detection target is a small region of the image in a complex environment, specifically when the firearm or knife is adjacent to people and not in other areas of the image.

Through the suggested alterations, it was possible to reduce images of large proportions to smaller images, preserving the crucial information, which in this context is people carrying firearms. A total of 5,179 images were then supplied to the firearms detection model, while 5,176 images were provided to the knife detection model, depending on the classes that needed to be identified. In both instances, the training process encompassed 70% of the data, while the testing comprised the remaining 30%.

B. Training.

Encryption, replay, and training of the element and image detection model were performed in the Google Colab interactive environment. The programming also included the YOLO v5s algorithm. With the help of these tools, it was easy to tune the model parameters to previously labeled images and store a substantial data set. The YOLO v5s network was then trained for convergence through Images, Obtaining Label iterations of the additional Class A: Firearms and Class B: White Weapons classes using 70% of the images. The model was then evaluated using the remaining 30% of the images using the Python text editor, the programming we relied on for training is shown in Unified Modeling Language (UML) diagrams in Figure 1. On the other hand, retraining of the neural model was used to achieve convergence if the accuracy was between 80% and 81%. Ultimately, if the result was positive, the algorithm was corroborated by the analysis carried out by the video surveillance system.

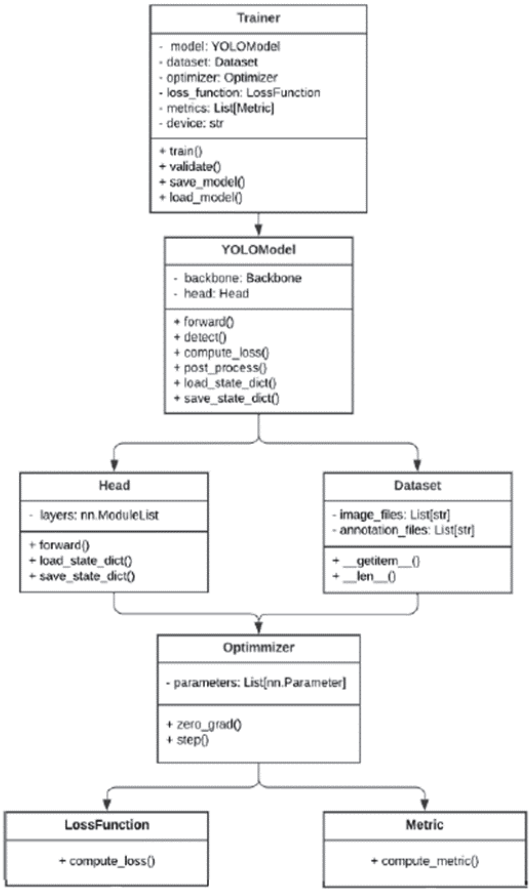


Fig. 1. Basic structure for YOLO training on UML diagrams.

C. Configuration

The system was expanded using Raspberry Pi 4 [18], specific parameters serve as a solid foundation for system operation. The 5 Mpx camera (Raspberry Pi 4) was used for trial and error and in the implementation the 12 Mpx webcam was chosen and at the same time the Raspbian OS system because it works with the hardware requirements of the Raspberry Pi 4 model. Real-time imaging system camera integrated with, Firebase Cloud, Android App and YOLOv5s with its two class types, Class A Class B, is a crucial part, and the detection model processes the information of that image to determine if a weapon may or may not be present.

D. Firebase Integration

To manage and store data in applications, Firebase offers several functions and services when used as a cloud solution. These logs contain information about detection events and their associated details, are stored in real-time, and are kept accessible for later analysis and consultation. The architectural representation in Figure 2 demonstrates how its use in the system allowed these functionalities to be leveraged and ensure an effective approach to the management of data generated by weapons detection.

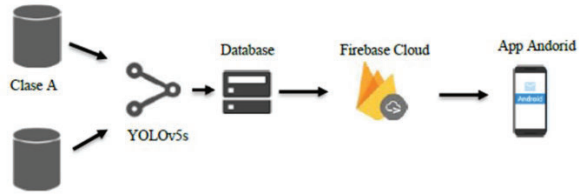


Fig. 2. Integration with Firebase.

To facilitate smooth and effective interaction between the system and a connected mobile application such as Android Studio, Firebase also takes care of communication with the mobile application, offering a collection of real-time communication tools. By ensuring that only authorized users can access the data generated by the system, the authentication and security mechanisms of your platform add an extra layer of protection to the information we store.

III. Results

The checks carried out during the training process of the neural network using a dimension or batch size of 940 images with a validation accuracy of 0.45% for knives and 0.771% for firearms, the image size is set to 640 having the precision of a total of classes of 0.61% in mAP (Mean Average Precision) with 0.5% being 98%, for YOLOv5s it can incredibly detect objects with a speed of 140 fps (frames per seconds), which is 180% faster for obtaining training loss and accuracy. You get the Google Colab configuration with "batch" 16 and "epoch" 75.

For the execution of tests, the YOLOv5s architecture mentioned in section 2 was used, it was verified through the train.py file that the model correctly selected the weapons to be detected shown in Figure 3, from these we proceed with the analysis and optimal system performance.

Fig. 3. Weapon selection using Google Colab training.

The results obtained in the precision-recovery curve graphs observed in Figure 4 are detailed. This identifies the precision values of firearms with 0.681%, and knives with 0.699%. With these values, we can evaluate the performance of the model in object detection which is an accepted value for the performance of the implemented system.

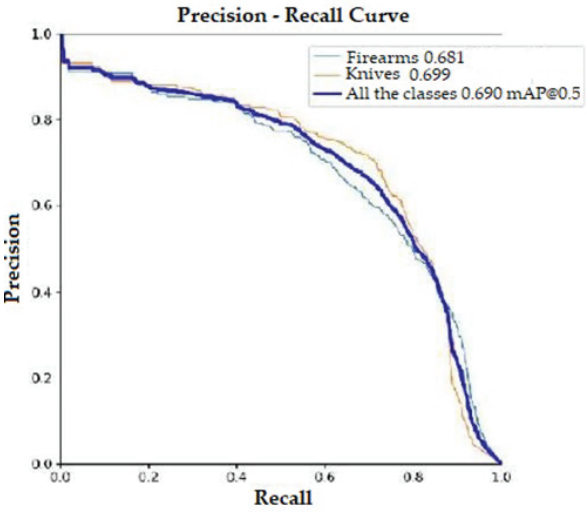


Fig. 4. Precision-recall curve.

The performance of the YOLOv5s model is measured in terms of Mean Average Precision (mAP). The higher the mAP value, the more accurate the model is. The model gives 0.95% as the Average Precision. See Figure 5.

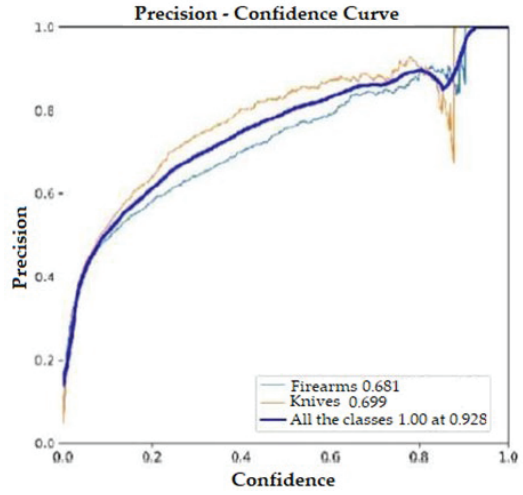


Fig. 5. Precision-confidence curve.

The value of precision and recall varies as the number of epochs increases. In epoch 75, the precision reached 0.98% and the recall was 0.87%, showing a good sign of performance. See Figure 6.

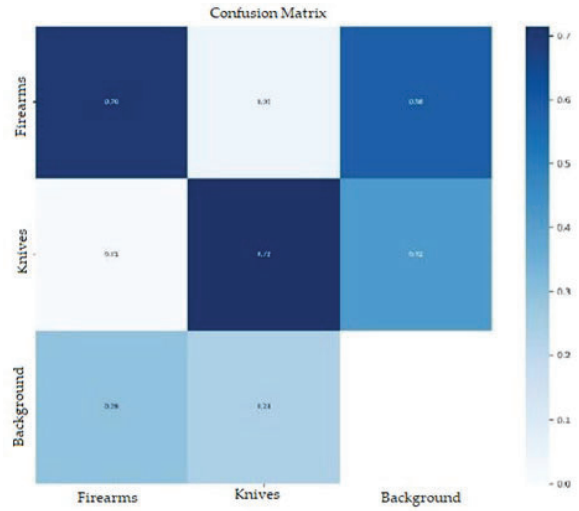


Fig. 6. Confusion matrix.

According to the results obtained from the training of the model, the levels of pressure and loss can be distinguished in the analysis of the images since these are on par in the precision curves, and do not present much difference since they remain with stable values in the confusion matrix; The improved susceptibility of the model which was obtained from the training is noted. With the values obtained in the precision curves, the values are optimal for the correct functioning of the detection system since it does not present drawbacks in its use. At the time of detecting the object, the image is saved and subsequently sent to Firebase, the database that was used in this stores the image in Storage and turns it into the real-time database, in the latter it is saved in a link with the date, and time of the entry of the image, which is displayed in the application.

When considering the times of saving, sending, and displaying the image of the detected object, we can consider that they are optimal for a detection system since these occur almost instantly and do not exceed 4 seconds in detecting and sending the information to the user. With the data obtained, the results in the detection of weapons and knives will be displayed in the application, which is indicated in Figure 7, and in turn, the interface is indicated, which consists of two buttons, the first shows the detections and the second button calls emergencies. By clicking the "Show detection" button, the detections achieved are indicated and the date and time of these in their different classes are detailed.

Fig. 7. Recognition of edged weapons and firearms.

IV. Conclusions

The multi-class object detection model was even less accurate than the single-class object detection model. Despite the infrastructure environments demonstrating the need for a technological application of great relevance in the field of security, even though some small elements can still confuse the system and cause erroneous identifications, the operation is efficient. The camera detects weapons held by certain individuals, weapons that could go unnoticed by a human being who sometimes has multiple monitors around or does not have images of sufficient quality for adequate perception. In addition to having superior observation capacity, the system also re-acts more quickly; In less than a tenth of a second, it could warn of the presence of weapons and save human lives.

The weapon detection period is 400 ms, that is, immediately, and the latency metric for sending data from detected weapons to the mobile application is 48 seconds, indicating that this is the communication time between the cloud and the mobile application. The use of "YOLO" in the identification system made it possible to provide the developed model with the parts of the image that are relevant to detecting firearms and edged weapons. This allows the system to focus exclusively on areas where weapons are found and ignore other regions of the image where firearms are not expected to be present. and knives, this helps to make the criminal activities that occur in the environment irrelevant to current common scenarios.

References

[1] J A Hidalgo, E. (2012). Sistema cctv (circuito cerrado de televisión) entre edificios, para la seguridad y vigilancia en el aeropuerto internacional cotopaxi. Universidad Técnica de Ambato.

[2] R. K. Tiwari and G. K. Verma, "A Computer Vision based Framework for Visual Gun Detection Using Harris Interest Point Detector," Procedia., vol. 54, pp. 703– 712, 2015.

[3] Jain, Harsh y Vikram, Aditya y Mohana, Mohana y Kashyap, Ankit y Jain, Ayush. (2020). Detección de Armas usando Inteligencia Artificial y Aprendizaje Profundo para Aplica- ciones de Seguridad. 193-198. 10.1109/ICESC48915.2020.9155832.

[4] GF Shidik, E. Noersasongko, A. Nugraha, PN Andono, J. Jumanto y EJ y Kusuma, "Una revisión sistemática de la videovigilancia de

inteligencia: tendencias, técnicas, marcos y con- juntos de datos". Acceso IEEE, vol. 7,págs. 457–473, 2019.

[5] Lai, Y. L., Sheu, C. J., & Lu, Y. F. (2019). El esquema de circuito cerrado de televisión monitoreado por la policía realmente importa en la reducción del crimen, 63(1), 101-134. doi:10.1177/0306624X18780101.

[6] Redmon et al., Solo se mira una vez: Detección unificada de objetos en tiempo real. IEEE CVPR, 2016.

[7] D. Schuller y BW Schuller, "La era de la inteligencia emocional artificial", Computadora IEEE, vol.51, número 9, pp. 38-46, septiembre de 2018.

[8] S. Narejo, B. Pandey, C. Rodríguez, MR Anjum et al., "Detección de armas usando yolo v3 para sistema de vigilancia inteligente,"Problemas Matemáticos en Ingeniería, vol. 2021, 2021.

[9] Elmir, Yourssef, Sid Ahmed Laouar y Larbi Hamdaoui. "Aprendizaje profundo para la de- tección automática de pistolas en secuencias de video".JERI.2019.

[10] ST Ratnaparkhi, A. Tandasi y S. Saraswat, "Detección y reconocimiento de rostros para el sistema de identificación de delincuentes", 202111.ª (Confluence), 2021, págs. 773-777, doi :10.1109/Confluence51648.2021.9377205.

[11] Alex Krizhevsky, Ilya Sutskever and Geoffrey E. Hinton. 2012. Clasificación de ImageNet con redes neuronales convolucionales profundas. En Actas de la 25.ª Conferencia Internacional sobre Sistemas de Procesamiento de Información Neural - Volumen 1 (NIPS'12). Curran Associates Inc., Red Hook, Nueva York, EE. UU., 1097–1105.

[12] GF Shidik, E. Noersasongko, A. Nugraha, PN Andono, J. Jumanto y EJ Kusuma, "Una re- visión sistemática de la videovigilancia de inteligencia: tendencias, técnicas, marcos, (CCECE) pp. 1-6, abril de 2012 y conjuntos de datos", en IEEE Access, vol. 7, págs. 170457-170473, 2019, doi: 10.1109/ACCESS.2019.2955387.

[13] R. Girshick, J. Donahue, T. Darrell y J. Malik, "Jerarquías de características ricas para la detección precisa de objetos y la segmentación semántica", IEEE, págs. 580–587, 2014.PyTorch, "YOLOv5"

[14] U. Nepal y H. Eslamiat, "Comparación de yolov3, yolov4 y yolov5 para la detección autó- noma de puntos de aterrizaje en vehículos aéreos no tripulados defectuosos" Sensores, vol. 22, núm. 2, pág. 464, 2022.

[15] Rothe, Rasmus y Guillaumin, Matthieu y Van Gool, Luc. (2015). Supresión no máxima para la detección de objetos al pasar mensajes entre ventanas. LNC. 9003.10.1007/978-3-319- 16865-4_19.

[16] Zhao, Z., Zheng, P., Xu, S., & Wu, X. (2019). Detección de objetos con aprendizaje pro- fundo: una revisión. IEEE Transactions on Neural Networks and Learning Systems, 30(11), 3212-3232. doi: 10.1109/TNNLS.2018.2876865.

[17] P. Kapica, "Implementing YOLO v3 in Tensorflow (TF-Slim) – ITNEXT," 2018. [Online]. Available: https://itnext.io/implementing-yolo-v3-in- tensorflow-tf-slim-c3c55ff59dbe. [Accessed: 13-Dec-2018]

[18] G. J. Caiza and M. V. García, "Implementación de sistemas distribuidos de bajo costo bajo norma IEC-61499, en la estación de clasificación y manipulación del MPS 500," Ingenius, vol. 18, pp. 40–46, 2017, doi: 10.17163/ings.n18.2017.

Comparing Approaches for Prioritizing and Selecting Scenarios in Simulation-based Safety Testing of Automated Driving Systems

Fauzia Khan
Institute of Computer Science
University of Tartu
Tartu, Estonia
fauzia.khan@ut.ee

Hina Anwar
Institute of Computer Science
University of Tartu
Tartu, Estonia
hina.anwar@ut.ee

Dietmar Pfahl
Institute of Computer Science
University of Tartu
Tartu, Estonia
dietmar.pfahl@ut.ee

***Abstract*—Simulation-based safety testing provides a cost-effective method for testing Automated Driving Systems (ADS) across diverse scenarios. However, prioritizing or selecting test scenarios for simulation-based safety testing remains challenging due to the infinite variety of scenarios that ADS may encounter. In this study, we conducted a literature review to identify approaches for selecting or prioritizing scenarios for ADS safety testing. We compare the six identified approaches in a tabular form across various aspects. We discuss one approach in detail, illustrating how it could complement the other selected approaches through an example. Our ongoing work involves extending the comparative analysis to cover all approaches comprehensively.**

Keywords - Automated Driving System (ADS); Simulation-based Testing; Safety Testing, Scenario Selection; Scenario Prioritization.

I. INTRODUCTION

Automated Driving Systems (ADS) represent a much-advanced technology that has the potential to revolutionize the automotive and transportation industry [1]–[3]. An ADS is a safety-critical system that can perform Dynamic Driving Tasks (DDTs) without human driver assistance [4], [5]. An ADS is loaded with advanced technologies that can independently control the vehicle and make decisions regarding steering, acceleration, deceleration, braking, maneuver planning, and object detection [6], [7]. These control decisions are operational and depend upon the level of automation[1]. As automation levels increase, human driver involvement in driving decreases, with complete ADS control achieved at level 5.

Despite many advancements in ADS, thorough testing and validation remain critical for ensuring their safety [9], [10]. Moreover, in the context of safety, even a single failure made by one manufacturer affects its reputation and undermines public trust in the entire automotive industry, diminishing confidence in ADS. Thus, rigorous testing is required to build and maintain trust in ADS.

Generally, there are two common methodologies for testing ADS: real-world testing [11], [12] and simulation-based testing [13]–[15]. In real-world testing, the ADS would have to be driven hundreds of billions of miles to prove that it is safer than a typical human-driven vehicle (HDV) [16], which is costly and risky. In contrast, simulation-based testing is a cost-effective way to assess ADS's safety in various scenarios. [17]–[19]. A scenario refers to a specific situation the ADS under test (typically called an 'ego vehicle') might encounter during its operation. A scenario is defined through various components, including the ego vehicle itself, the static environment (such as roads, buildings, and infrastructure), the dynamic environment (such as other vehicles, pedestrians, and objects), and specific conditions (such as weather light, etc.) [20], [21]. These scenario components have specific elements and sub-elements [22]. For example, in the scenario, "An ego vehicle changes lanes on a highway during the night in rainy weather", the sub-elements associated with the dynamic environment are the "ego vehicle" and "changes lanes". "Highway" is the sub-element associated with the static environment while "rainy" and "night" are sub-elements associated with conditions.

An ADS could encounter infinite scenarios depending upon various factors, such as dynamic objects (pedestrians, vehicles, animals, etc.), static objects (traffic lights, sign boards, etc.), road typologies, weather, and lighting conditions. Combining these factors leads to infinite scenarios, and testing every scenario is not feasible, even in a simulator.

[1] The Society of Automotive Engineers defines six levels of automation for on-road automated vehicles. These levels are described in SAE safety standard J3016 [8]

In this study, we aim to identify and compare the approaches defined in the literature for prioritization or selection of test scenarios to see how these approaches could help select test scenarios for simulation-based safety testing of ADS. This comparison could benefit both the automotive industry and academic research. The contributions of our study are:

- A literature review to identify the studies related to the selection or prioritization of test scenarios for simulation-based safety testing of ADS.
- A comparison of selected approaches w.r.t the purpose and scope of testing, input data type, input data content type, datasets used, number of steps involved, scenario components, supported simulator, tool support, open source, and limitations.
- A preliminary comparison via an illustrative example of one of the selected approaches (SSTSS) could complement other selected approaches. Detailed comparisons among the selected approaches are a work in progress.

The rest of this paper is organized as follows. Section II presents the related work. Section III presents the methodology. Section IV presents the results of the comparative analysis. Section V presents the discussion, and the conclusion and future work are given in Section VI.

II. RELATED WORK

To the best of our knowledge, no similar study exists in the literature whose goal is to compare the approaches of scenario selection or prioritization for safety testing of ADS. However, in this section, we present related secondary studies focused on the testing of ADS. We also highlight how the automotive industry tests ADS safety and compares it to human drivers as a benchmark.

A. Simulation-based Safety Testing of ADS

Tang et al. [23] presented a survey focused on module and system-level testing of ADS. They analyzed how different modules are affected by various technical factors and highlighted issues during ADS's development or deployment. Khan et al. [5] conducted a systematic literature review to identify safety features, testing methods, tools, and datasets used for the safety testing of ADS. Lou et al. [24] presented a comprehensive study to understand current ADS testing practices and needs by conducting interviews with companies and developers. They further analyzed the gap between ADS research and practitioners' and suggested future directions, such as developing test reduction techniques to accelerate simulation-based testing of ADS. [25] presented a survey focusing on the algorithms and tools used to generate critical scenarios in automated driving. They further identified challenges in existing approaches regarding safety-critical scenario generation approaches.

B. Industry Practices for ADS Safety Testing

In the automotive industry, simulation-based testing of ADS is widely used as an alternative to real-world testing [24]. In a recent study[2], Waymo compared the simulated performance of its autonomous Waymo driver to human drivers involved in fatal crashes in Chandler, Arizona, between 2008-2017. They modeled and simulated a non-impaired human driver (NIEON). The Waymo driver consistently did better than this high benchmark by avoiding 100% of crashes, except when struck from behind. Additionally, in collision simulations, the Waymo driver surpassed the NIEON model, avoiding 75% of crashes and reducing severe injury risk by 93%.

Cruise[3] also evaluated the safety performance of ADS by comparing their collisions with those of human drivers. However, assessing human driving performance remains challenging due to limited data. Particularly, there's a lack of information regarding significant collisions or violations despite the abundance of data on AV safety collected through advanced sensors and data logging. Our study aims to identify and compare scenario selection or prioritization approaches for simulation-based safety testing of ADS.

III. METHODOLOGY

In this section, we present our methodology to identify studies related to scenario selection or prioritization approaches for simulation-based safety testing of ADS and their comparison. We formulate our research question and conduct a literature review to find relevant studies. We explain the formulation of the search string, the execution of the search query, and the exclusion/inclusion criteria. These steps were guided by K. Petersen et al. [26]. Finally, we present the criteria for comparing the approaches developed for test scenario selection or prioritization for simulation-based safety testing of ADS.

A. Research Questions

The goals of the study are to identify relevant literature on scenario selection or prioritization for simulation-based safety testing of ADS and to compare these approaches to identify strengths and weaknesses. To end this, we formulated two research questions (RQs) as follows:

RQ1: *What scenario selection or prioritization approaches exist in simulation-based safety testing?*

RQ2: *How do the identified approaches compare to one another?*

B. Search Query

We searched the online repositories (IEEE, ACM, Springer, Science Direct) to find publications to answer the research questions. To maximize the coverage of the literature, we selected search terms from the research questions. We used the keyword "AND" for the concatenation of search terms and the keyword "OR" for combining synonyms of the search terms to

[2] https://waymo.com/blog/2022/09/benchmarking-av-safety

[3] https://getcruise.com/news/blog/2023/cruises-safety-record-over-one-million-driverless-miles/

maximize coverage. Since we are only interested in the prioritization or selection of test scenarios for simulation-based safety testing of ADS, therefore we used search terms such as "Test scenario selection", "Test reduction", and "Test scenario prioritization". We also used the search terms "Autonomous Driving Systems" and "Self-driving cars" as synonyms to ensure that the search query produces results related to testing scenarios for ADS. We did not explicitly use the term "On-road testing" or "Test scenario generation" with the keyword "NOT" to filter the search query results because we noticed that it would filter out many relevant publications. Search terms related to our research questions, e.g., "Simulation", were also not added to the search query because they make the research query too specific and might not capture relevant publications. We also used "*" with some search terms to capture all variations of search terms. We show the general search query below[4]:

((Test scenario select*) OR (Test reduction) OR (Test scenario prioritize*)) AND ((Autonomous Driving Systems) OR (Self-driving car*))

C. Additional Filters

We entered the search query into the advanced search of the online databases and applied the filters mentioned in Table I. We consider primary studies published in journals and conferences from the year 2000 onwards.

TABLE I: FILTERS APPLIED TO SEARCH QUERY RESULTS

Filter	Value
Year	2000-2024
Content Type	Journal Articles and Conference Papers
Language	English
Discipline	Computer Science, Engineering

D. Exclusion/Inclusion Criteria

To select the publications focused on the selection or prioritization of test scenarios for simulation-based safety testing of ADS, we performed the following steps: (i) We read the titles, abstracts, and keywords to analyze the relevance of the publications. If the decision is not made, we read the introduction, methodology, results, and conclusions and download the relevant ones. (ii) We made an Excel sheet of the downloaded papers and removed the duplicates. If a duplicate is found, we include the most recent and comprehensive versions of duplicated papers. (iii) We included (I1) journal articles or conference papers only whose primary goal is to select, prioritize or reduce test scenarios for simulation-based safety testing of ADS. We excluded all publications meeting at least one of the following criteria: (E1) the publication is written in a language other than English, (E2) the publication is not accessible or unavailable, (E3) the publication lacks sufficient details such as methods, contributions, etc., (E4) the publication falls into categories such as magazine, doctoral symposium paper, thesis, secondary studies position paper, keynote presentation, or abstract.

E. Data Extraction

RQ1 can be answered based on the final set of selected studies because each study will describe at least one scenario selection or prioritization approach.

Related to RQ2, we need to extract data items from the selected studies that inform the comparison of existing approaches. The following extracted data items were used as criteria for the comparison:

1) Purpose of Testing: We compare the selected approaches based on their testing purpose as follows: (i) compare ego car vs human-driven car accident, (ii) compare ego car vs human-driven car without accidents, (iii) explore ego car behavior for testing new/improved autonomy software features (for developers), (iv) expose ego car behavior to the scenario typically used in driver tests, (v) expose simulate ego car behavior to provide feedback for the real-world test plan. ***2) Scope of Testing:*** We compare the scope of testing based on whether the selected testing approach is specific to test (i) full ADS, (ii) specific ADS feature or its subset. ***3) Input Data Type of Approach:*** We also extract the information regarding the initial input(s) data type of each selected approach, which includes: (i) text, (ii) video, (iii) image, (iv) audio, (v) combination of any of above. ***4) Input Data Content of Approach:*** We further compare the content of input data used in each selected approach, which includes: (i) human-driven car accident data, (ii) human-driven car data without accidents, (iii) simulator recorded ADS driving, (iv) ADS from real-world recording. ***5) Number of Steps:*** To determine the ease of application of each selected approach, we compare the number of steps involved in each method ***6) Scenario Component:*** We extract the information regarding the scenario components considered as a proof of concept in selected approaches. (i) Static Environment, (ii) Dynamic Environment, (iii) Weather, (iv) Light conditions. ***7) Supported Simulator:*** Considering that each simulator has its unique limitations and capabilities, we extract the information about which simulator each approach supports. ***8) Tool Support:*** We compare the maturity of each approach based on tool availability. ***9) Open Source:*** We also check whether the selected approach is open source, making it easier for researchers and practitioners to access, utilize, and replicate the approach. ***10) Limitation:*** Finally, we present each selected approach's possible limitation(s) for ADS simulation-based safety testing.

To extract data items used in selected approaches, we read each selected publication's abstract, introduction, methodology, and results sections and organized the data in a tabular format. Each table row recorded the data for predefined criteria, as discussed above. During data extraction, conflicts (if any) are discussed and resolved among the authors. We identified the key terms and their synonyms. We removed duplicate terms and

[4] The query was entered in databases on 22 April 2024

applied the bottom-up merging method to merge key terms to get high-level categories.

IV. RESULTS

In this section, we present the results of each RQ. For RQ1, the results of the selection procedure, selected publications, and their approaches are presented. For RQ2, we present the comparison of selected approaches based on predefined criteria (cf. Section III-E).

RQ:1 *What scenario selection or prioritization approaches exist in simulation-based safety testing?*

A. Results of the Selection Procedure

The number of publications after each selection step is given in Table II. We found 1438 publications identified in the initial search at step 1. In step 2, we downloaded 31 publications and found one duplicate in step 3. In step 4, we selected only six publications after applying inclusion and exclusion criteria. The results were recorded in an Excel sheet available online[5]. Table III shows the count of conference papers and journal articles selected from each online repository.

TABLE II: NUMBER OF PUBLICATIONS FOR EACH STEP IN THE SELECTION PROCEDURE

Screening Steps	No. of Publications
Initial Search	1438
Downloaded Publications	31
Duplicate Removal	1
Included Publications	6

TABLE III: DISTRIBUTION OF SELECTED PUBLICATIONS PER REPOSITORY

Database	Conference Paper	Journal Article	Total Publications
IEEE	2	0	2
ACM	1	1	2
Springer	2	0	2
Science Direct	0	0	0

B. Selected Publications and their Approaches

Table IV shows the selected publication and their approaches for selecting or prioritizing scenarios for simulation-based testing for safety testing of ADS. We presented an overview of each approach shown in Table IV.

1) **SSTSS**: stands for (simulation-based safety testing scenario selection). This approach uses a human driver car crash dataset to prioritize and select test scenarios through the following eight steps: *(i) Scenario catalog selection:* Initially, a publicly available scenario catalog that is comprehensive and published by a reputable organization is chosen such as National Highway Traffic Safety Administration - (NHTSA)[6], Euro NCAP[7]. *(ii) Enumerating ODD of ADS:* enumerate the ODD[8] of Vehicle Under Test (VHT) in terms of spatial, temporal, and environmental conditions[9]. *(iii) Filtering scenarios based on ODD of ADS:* exclude scenarios that do not fall within the ODD enumerated in the previous step. (iv) *Scenario grouping:* categorize the scenarios into groups based on similar critical actions of the ego vehicle or the target object, such as pedestrians, cyclists, animals, etc. (v) *Removing duplicates within a scenario group:* identify the duplicate scenarios within scenario groups. *(vi) Prioritizing scenario groups:* prioritize the scenario based on common crash scenario statistics and assign the highest priority to scenario groups where more accidents occur. *(vii) Filtering scenarios based on simulator limitations:* choose a simulator that can replicate the real world closely and analyze data on performance metrics such as travel time, crashes, etc. After simulator selection, exclude the scenarios that cannot be implemented due to simulator limitations. *(viii) Prioritizing and selecting scenarios for testing ADS* - assign a score based on the scoring technique to each scenario within the scenario group based on the availability of car crash data statistics for scenario each element, such as actors, weather, etc. The final output of the SSTSS process is a ranked list of prioritized scenario groups in descending order of priority. Testing starts with the scenarios from the top-prioritized scenario group in sequence. After all scenarios within the top-prioritized scenario group are tested, the process proceeds to subsequent prioritized scenario groups.

TABLE IV: SELECTED PUBLICATIONS OVERVIEW

Sr.#	Publication	Approach	Online Repository
1	[27]	SSTSS	Springer
2	[28]	SDC-Scissor	IEEE
3	[29]	SDC-Prioritizer	ACM
4	[30]	STRaP	ACM
5	[31]	SPECTRE	Springer
6	[32]	J. Bach et al.	IEEE

2) **SDC-Scissor:** stands for self driving cars-cost-effective test selector. This approach [28] utilizes machine learning (ML) models to select scenarios based on five components, which are as follows: *(i) SDC-Test Generator:* generates random test using Frenetic and AsFault tool [36], [37]. *(ii) SDC-Test Executor:* executes the generated simulation-based tests and records the resultant output to categorize tests as safe or unsafe. *(iii) SDC-Features Extractor:* extracts various road features (distance, turns, angles, etc.) and road statistics from driving scenarios. *(iv) SDC-Benchmarker:* uses these features and corresponding labels to train the ML models and determine the most effective model for predicting test outcomes.

[5] https://github.com/ScenarioSelectionApproaches

[6] https://rosap.ntl.bts.gov/view/dot/6281/dot_6281_DS1.pdf?

[7] Euro NCAP

[8] The specific conditions and environments under which a particular driving automation system is designed to operate [33]

[9] ODD covers spatial (geography, road types, lanes, speed limits, etc.), temporal (day/night), and environmental conditions (weather). [8], [34], [35]

(v) SDC-Predictor: identifies simulation-based test scenarios that are unlikely to detect faults in the ego vehicle and excludes them from execution.

3) **SDC-Prioritizer**: This approach [29] is similar to SDC-Scissor; however, it uses single-objective and multi-objective genetic algorithms to prioritize test scenarios. The steps are as follows: (i) Initially, a random test scenario is generated using the tool AsFault [37]. Each generated test scenario has information (JSON file) on the start and destination points for the ego car on the map, the entire road network, and the chosen driving path. (ii) Two sets of road features are extracted. i.e., road characteristics (road length and direct distance between the start and destination points, number of turns, angles) and the road segment statistics to identify safe and unsafe scenarios even before executing them. (iii) The extracted features serve as inputs to two black-box scenario prioritization approaches, i.e., SO-SDC-Prioritizer and MO-SDC-Prioritizer, which utilize single and multi-objective GA to prioritize test cases.

4) **STRaP:** stands for Scenario-based Test Reduction and Prioritization. This approach [30] is developed to reduce and prioritize the testing scenarios using data recordings of previous versions of ADS under test. The steps are as follows: (i) the semantic information from each frame of ADS video recordings is extracted via the ADS's communication channels under test and converted into vectors. The semantic information includes static (e.g., traffic lights, crosswalks, etc.) and dynamic objects (e.g., vehicles, cyclists, actions of vehicles, etc.). (ii) The driving recordings are divided into continuous and redundant segments to cut length. (iii) The remaining segments are prioritized based on their coverage of driving scene elements and the rarity of elements such as traffic lights, pedestrians, etc.

5) **SPECTRE:** stands for selection and prioritization of test scenarios for autonomous driving systems. This multi-objective search-based approach [31] was developed to minimize testing costs for newer versions of ADS by utilizing historical test data from previous versions. The steps are as follows: (i) Initially, each scenario is characterized by a set of properties of the ADS under test, such as speed and environmental factors such as weather and obstacles. (ii) Each scenario is executed in simulations, which yields four key output values, resulting in four objective functions: collision occurrence, collision probability, ADS demand, and scenario diversity. (iii) Four optimization objectives are defined based on the results obtained in the previous step. (iv) Finally, multi-objective evolutionary algorithms are used as optimization techniques to prioritize driving scenarios.

6) **J. Bach et al.:** In this study [32], a two-step selection concept was introduced for selecting scenarios for the safety testing of ADS. The steps are as follows: (i) Initially, scenarios are categorized into abstract groups based on specified system-level requirements. This includes segmentation by geolocation (e.g., different countries with varying traffic rules) or road category (e.g., motorways, rural roads, urban streets), as well as criteria such as the ego vehicle's state and surrounding traffic density. A classification-tree approach is employed to systematically preselect scenarios, ensuring coverage of relevant use cases for the targeted regions. (ii) Repetitive information and situations are removed by looking at the two-dimensional histograms of frames. These histograms are visual sources for how scenarios are spread out and connected. Only scenarios that fill empty areas in the histograms are kept, while others are removed.

RQ2: *How do the identified approaches compare to one another?*

To answer RQ2, we compare the selected approaches w.r.t the purpose and scope of testing, input data type, input data content type, datasets used, number of steps involved, scenario components, supported simulator, tool support, open source, and limitations. Table V compares six selected approaches per the criteria defined in Section III-E. All the selected approaches prioritize test scenarios to reduce the number of testing scenarios in simulation-based testing. However, the testing purpose and the scope of the selected approaches differ. Therefore, it would be interesting to discuss if these approaches could complement one another (by combining testing from different perspectives) for a more comprehensive, effective, and efficient testing of ADS.

In this study, we present the preliminary results of the comparison. Using an illustrative example, we demonstrate how SSTSS could complement or be combined with selected approaches. Detailed comparisons among selected approaches are a work in progress.

Comparison of the SSTSS process (via illustrative example): The selected five approaches differ from SSTSS as they are based on simulation data. The SSTSS process would complement simulation-based approaches in different ways.

(i) The SSTSS process could be used as the first step for the pre-selection of scenarios before employing other approaches. Since the SSTSS process prioritizes statistically significant scenarios using real-world car crash statistics, it could be used to optimize the initial selection of scenarios and driving routes. The STRaP and SPECTRE approaches select testing scenarios from available driving routes within the simulator. However, solely relying on a specific driving loop may overlook various real-world situations, leading to potential gaps in scenario coverage. Furthermore, there could be scenarios of lower significance in a specific driving loop, and allocating resources to test them might not be cost-effective. Once the driving route containing prioritized testing scenarios is selected using the SSTSS process, STRaP or SPECTRE approaches could be used to reduce the test scenarios further. Using such a hybrid approach could reduce the number of test scenarios, making the testing more efficient, cost and resource-effective.

Let's consider an example. Suppose we want to test a BOLT car (ego vehicle) from the Autonomous Driving Lab at the University of Tartu, Estonia. The BOLT car has the following ODD: It can follow the traffic flow, detect pedestrians, and give way to them. Given the ODD, we apply the SSTSS process to select prioritized scenarios for simulation-based testing from a list of 111 scenarios. Given the sorted list of scenarios generated from the SSTSS process (see a sample output in Table VI), a specific driving route could be selected. In the sample output shown in Table VI, the top three prioritized scenario groups are Following Lead Vehicle[10], Crossing Path[11], and Pedestrian Interaction[12]. As a result, the "Tartu loop" could be selected, which includes seven regulated pedestrian tracks, four unregulated pedestrian tracks, two regulated intersections, one right turn, and one roundabout, and multi-lane roads where the ego vehicle can closely follow another vehicle and encounter pedestrians. Furthermore, it includes intersections to encounter crossing paths scenarios. For the selected "Tartu loop", a digital twin is available online[13], which can be used as input for STRaP. The STRaP extracts the semantic information from video recordings of the selected "Tartu loop" to reduce the number of testing scenarios. Then, the driving recordings are segmented to

TABLE V: COMPARISON OF APPROACHES

Sr.#	Features	SSTSS	SDC-Scissor	SDC-Prioritizer	STRaP	SPECTRE	J. Bach et al.
01	Purpose of Testing	compare ego car vs human-driven car accidents	explores ego car behavior to test new/improved autonomy software features	explore ego car behavior to test new/improved autonomy software features	explore ego car behavior to test new/improved autonomy software features	explore ego car behavior to test new/improved autonomy software features	expose simulate ego car behavior to provide feedback for the real-world test plan
02	Scope of Testing	Full ADS	Specific Feature (LDW)	Specific Feature (LDW)	Full ADS	Full ADS	Specific Feature (PCC)
03	Input DT	Text	Video	Video	Video	Video	Video
04	Input Data Content of Approach	Human-driven car accident data	Simulator recorded ADS driving	Simulator recorded ADS driving	Simulator recorded ADS driving	Simulator recorded ADS driving	Read-world and Simulator recorded ADS driving
05	# of steps	8	5	3	3	4	3
06	Scenario Components	Static & Dynamic Env., Weather & Light condition.	Static Environment (only Roads)	Static Environment (only Roads)	Static & Dynamic Env., Weather	Static & Dynamic Env., Weather & Light conditions	Static Environment (Roads and Location)
07	Sup. Simulator	Carla	BeamNG	BeamNG	LGSVL	LGSVL	-
08	Tool Sup.	No	Yes	No	Yes	No	No
09	OS	Yes	Yes	Yes	Yes	Yes	No
10	Limitations	-Solely relies on the availability of human-driver accident dataset, overlooking ADS data -Requires Manual effort to execute each step.	-Solely relies on simulation driving data for scenario selection, thus potentially missing critical scenarios that arise from real-world human driving behavior - Handles limited driving scenarios, e.g., road shapes only	-Solely relies on simulation driving data for scenario selection, thus potentially missing critical scenarios that arise from real-world human driving behavior - Handles limited driving scenarios, e.g., road shapes only	Solely relies on simulation driving data for scenario selection, thus potentially missing critical scenarios that arise from real-world human driving behavior -Requires manual feature extraction of scenario elements from driving videos	-Solely relies on simulation driving data for scenario selection, thus potentially missing critical scenarios that arise from real-world human driving behavior -Relies on optimization objectives, which could lead to overfitting	Relies on predefined selection criteria, which could overlook critical scenarios that do not fit within predefined criteria, which may limit the selection of diverse scenarios

(LDW = Lane Departure Warning, PCC = Predictive Cruise Control, DT = Data Type, Env.= Environment, Sup.= Support, OS = Open Source)

[10] The Following Lead Vehicle scenario group includes the situations where the ego vehicle is driving behind another vehicle referred to as the "lead vehicle"

[11] The Crossing Path scenario group includes the situations where the paths of ego vehicle or more vehicles intersect or cross each other at an intersection or junction.

[12] The Pedestrian Interaction scenario group includes the situation where ego vehicle encounters pedestrians while driving.

[13] https://adl.cs.ut.ee/lab/simulation

eliminate redundancy and excessive length. The remaining segments are prioritized based on the coverage and the rarity of driving scenes to select the crucial test scenarios for effective and efficient testing.

(ii) The SSTSS process could be combined with other approaches to test specific features of an ADS. Since the SSTSS process prioritizes scenario groups, and each scenario group corresponds to specific ADS feature(s), indicating a need to focus on testing these features. The prioritized scenario groups using SSTSS could suggest which ADS features to focus on, and further thorough testing of these features can be performed using feature-specific approaches such as SDC-Scissor, SDC-prioritizer, etc. For example, in the lane change [14] scenario group, approaches like SDC-Scissor or SDC-prioritizer could be further used to select or prioritize safe or unsafe scenarios for testing ADS's Lane Departure Warning (LDW) feature.

In ADS testing, one potential question is determining how safe is safe enough. One answer could be that if an ADS is safer than a human driver, it is secure enough to deploy. In line with this, the SSTSS process uses human driver car crash datasets (as one of the inputs) to select test scenarios. Testing the ADS in these scenarios can directly compare its safety performance to that of human drivers. Also, it could be helpful for stakeholders to confirm if the ADS meets or exceeds the safety standards set for human drivers. Therefore, the SSTSS process could complement other approaches for the initial selection of test scenarios.

The main strength of the SSTSS approach is that it uses real-world car crash statistics for scenario selection, which prioritizes statistically significant yet often overlooked scenarios. However, the reliance on the availability and quality of car crash datasets may restrict the applicability in scenarios not well-represented in the data or geographic regions with sparse data collection. If the data is not up-to-date, infrastructure, technological developments, or other changes in driving behavior might not be reflected in the data and could affect the output of the process.

The SSTSS process selects and prioritizes scenarios based on the ODD of ADS under test (as one of the inputs). By considering ODD, on the one hand, SSTSS ensures that the selected scenarios reflect the intended use cases and environments for which the ADS under test is designed. This increases the probability of identifying potential safety-critical issues in those specific operating conditions. On the other hand, if manufacturers do not clearly define or convey the ODD, the selection of relevant scenarios may be less accurate.

V. THREATS TO VALIDITY

In this section, we discuss the possible threats to the validity of our study and the measures taken to mitigate them.

TABLE VI: A SAMPLE SSTSS PROCESS OUTPUT

Sr. #	Prioritized Scenario Groups	Selected Scenario Identifier	Scenario Priority
01	Following Lead Vehicle	S17	1
		S18	2
		S19	3
		S20	3
02	Crossing Path	S25	4
		S29	4
		S28	5
		S30	6
		S31	6
		S33	6
		S34	6
		S26	7
03	Pedestrian Interaction	S27	8
		S32	8
		S36	9
		S37	9
		S41	9
		S45	9
		S39	10
		S40	10
		S42	11
		S43	11
		S38	12
		S44	13

The description of scenario identifiers is available online[15]

Researcher bias: The first two authors chose the keywords and selected publications, which might have introduced subjective bias. We tried to mitigate this threat by explicitly defining inclusion and exclusion criteria for selecting studies to select publications objectively.

Search string validity: The search query string either generates too few results (false negatives) or too many (false positives). For both conditions, we tried to mitigate threats. We used wild card (*) to widen the coverage and the keyword "AND" to minimize the false positive. We tried to keep the search query the same across all online repositories, but we slightly varied it due to the constraints of each repository. We reviewed the search query results to see if they contained studies we knew already. We were aware of 4 relevant papers, and the search string retrieved 3 of them. However, we found the missing paper's conference version in another searched online repository. Furthermore, the results of the search strings were also manually checked, and false positives were removed. We did not manually add the missing publications to the results because we wanted to follow the procedure defined for strictly reviewing the literature. Furthermore, the studies using the

[14] The lane change scenario includes the situation where the ego vehicle or other adjacent vehicle is merging or switching lanes in the same direction without maintaining an appropriate distance and speed with adjacent vehicle.

[15] https://github.com/ScenarioSelectionApproaches

synonyms of the keywords used in the search query in the title might have been missed. We mitigate this threat by using synonyms of the keywords used.

VI. CONCLUSION AND FUTURE WORK

ADS requires exhaustive testing before its deployment on real roads. Simulation-based testing provides a cost-effective approach to test ADS and requires test scenarios. In this study, we identify the approaches used to select or prioritize test scenarios for simulation-based safety testing of ADS and compare them. We also illustrate an example of how one approach complements or could be combined with other approaches for improving testing effectiveness and efficiency. In the future, we aim to demonstrate via examples how the remaining five selected approaches can complement or combine with each other to enhance testing effectiveness and efficiency.

ACKNOWLEDGMENT

The research reported in this paper has been partly funded by BMK, BMAW, and the State of Upper Austria in the frame of the SCCH competence center INTEGRATE [(FFG grant no. 892418)] part of the FFG COMET Competence Centers for Excellent Technologies Programme, as well as by the European Regional Development Fund, grant PRG1226 of the Estonian Research Council and by Bolt Technology OU.

REFERENCES

[1] C.-Y. Chan, "Advancements, prospects, and impacts of automated driving systems," International journal of transportation science and technology, vol. 6, no. 3, pp. 208–216, 2017.

[2] B. Schoettle and M. Sivak, "A survey of public opinion about autonomous and self-driving vehicles in the us, the uk, and australia," tech. rep., University of Michigan, Ann Arbor, Transportation Research Institute, 2014.

[3] R. Pfeffer, G. N. Basedow, N. R. Thiesen, M. Spadinger, A. Albers, and E. Sax, "Automated driving-challenges for the automotive industry in product development with focus on process models and organizational structure," in 2019 IEEE International Systems Conference (SysCon), pp. 1–6, IEEE, 2019.

[4] X. Zhang, J. Tao, K. Tan, M. Torngren, J. M. G. S¨ anchez, M. R. Ramli,´ X. Tao, M. Gyllenhammar, F. Wotawa, N. Mohan, et al., "Finding critical scenarios for automated driving systems: A systematic literature review," arXiv preprint arXiv:2110.08664, 2021.

[5] F. Khan, M. Falco, H. Anwar, and D. Pfahl, "Safety testing of automated driving systems: A literature review," IEEE Access, 2023.

[6] E. Zablocki, H. Ben-Younes, P. P´ erez, and M. Cord, "Explainability of´ deep vision-based autonomous driving systems: Review and challenges," International Journal of Computer Vision, vol. 130, no. 10, pp. 2425–2452, 2022.

[7] J. M. Anderson, K. Nidhi, K. D. Stanley, P. Sorensen, C. Samaras, and O. A. Oluwatola, Autonomous vehicle technology: A guide for policymakers. Rand Corporation, 2014.

[8] SAE, "3016—taxonomy and definitions for terms related to on-road motor vehicle automated driving systems." https://www.sae.org/standards/ content/j3016 202104/, 2021. Accessed: 15-07-2023.

[9] K. Bimbraw, "Autonomous cars: Past, present and future a review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology," in 2015 12th international conference on informatics in control, automation and robotics (ICINCO), vol. 1, pp. 191–198, IEEE, 2015.

[10] Rodionova, I. Alvarez, M. S. Elli, F. Oboril, J. Quast, and R. Mangharam, "How safe is safe enough? automatic safety constraints boundary estimation for decision-making in automated vehicles," in 2020 IEEE Intelligent Vehicles Symposium (IV), pp. 1457–1464, IEEE, 2020.

[11] D. Sportillo, A. Paljic, and L. Ojeda, "On-road evaluation of autonomous driving training," in 2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp. 182–190, IEEE, 2019.

[12] X. Zhao, V. Robu, D. Flynn, K. Salako, and L. Strigini, "Assessing the safety and reliability of autonomous vehicles from road testing," in 2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE), pp. 13–23, IEEE, 2019.

[13] S. Baltodano, S. Sibi, N. Martelaro, N. Gowda, and W. Ju, "The rrads platform: a real road autonomous driving simulator," in Proceedings of the 7th International Conference on Automotive User Interfaces and Interactive Vehicular Applications, pp. 281–288, 2015.

[14] Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in Conference on robot learning, pp. 1–16, PMLR, 2017.

[15] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Mozeiko,ˇ E. Boise, G. Uhm, M. Gerow, S. Mehta, et al., "Lgsvl simulator: A high fidelity simulator for autonomous driving," in 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), pp. 1–6, IEEE, 2020.

[16] N. Kalra and S. M. Paddock, "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?," Transportation Research Part A: Policy and Practice, vol. 94, pp. 182–193, 2016.

[17] T. D. Son, A. Bhave, and H. Van der Auweraer, "Simulation-based testing framework for autonomous driving development," in 2019 IEEE International Conference on Mechatronics (ICM), vol. 1, pp. 576–583, IEEE, 2019.

[18] D. J. Fremont, E. Kim, Y. V. Pant, S. A. Seshia, A. Acharya, X. Bruso, P. Wells, S. Lemke, Q. Lu, and S. Mehta, "Formal scenario-based testing of autonomous vehicles: From simulation to the real world," in 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), pp. 1–8, IEEE, 2020.

[19] Z. Zhong, Y. Tang, Y. Zhou, V. d. O. Neves, Y. Liu, and B. Ray, "A survey on scenario-based testing for automated driving systems in highfidelity simulation," arXiv preprint arXiv:2112.00964, 2021.

[20] H. Elrofai, D. Worm, and O. Op den Camp, "Scenario identification for validation of automated driving functions," in Advanced Microsystems for Automotive Applications 2016: Smart Systems for the Automobile of the Future, pp. 153–163, Springer, 2016.

[21] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," in 2015 IEEE 18th international conference on intelligent transportation systems, pp. 982–988, IEEE, 2015.

[22] E. de Gelder, O. O. den Camp, and N. de Boer, "Scenario categories for the assessment of automated vehicles," CETRAN, Singapore, Version, vol. 1, 2020.

[23] S. Tang, Z. Zhang, Y. Zhang, J. Zhou, Y. Guo, S. Liu, S. Guo, Y.-F. Li, L. Ma, Y. Xue, et al., "A survey on automated driving system testing: Landscapes and trends," ACM Transactions on Software Engineering and Methodology, vol. 32, no. 5, pp. 1–62, 2023.

[24] G. Lou, Y. Deng, X. Zheng, M. Zhang, and T. Zhang, "Testing of autonomous driving systems: where are we and where should we go?," in Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 31–43, 2022.

[25] W. Ding, C. Xu, M. Arief, H. Lin, B. Li, and D. Zhao, "A survey on safety-critical driving scenario generation—a methodological perspective," IEEE Transactions on Intelligent Transportation Systems, 2023.

[26] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in 12th international conference on

evaluation and assessment in software engineering (EASE), BCS Learning & Development, 2008.

[27] F. Khan, H. Anwar, and D. Pfahl, "A process for scenario prioritization and selection in simulation-based safety testing of automated driving systems," in International Conference on Product-Focused Software Process Improvement, (Austria), pp. 89–99, Springer, 2023.

[28] C. Birchler, N. Ganz, S. Khatiri, A. Gambi, and S. Panichella, "Costeffective simulation-based test selection in self-driving cars software," Science of Computer Programming, vol. 226, p. 102926, 2023.

[29] C. Birchler, S. Khatiri, P. Derakhshanfar, S. Panichella, and A. Panichella, "Single and multi-objective test cases prioritization for self-driving cars in virtual environments," ACM Transactions on Software Engineering and Methodology, vol. 32, no. 2, pp. 1–30, 2023.

[30] Y. Deng, X. Zheng, M. Zhang, G. Lou, and T. Zhang, "Scenario-based test reduction and prioritization for multi-module autonomous driving systems," in Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 82–93, 2022.

[31] C. Lu, H. Zhang, T. Yue, and S. Ali, "Search-based selection and prioritization of test scenarios for autonomous driving systems," in International Symposium on Search Based Software Engineering, pp. 41–55, Springer, 2021.

[32] J. Bach, J. Langner, S. Otten, E. Sax, and M. Holzapfel, "Test scenario¨ selection for system-level verification and validation of geolocationdependent automotive control systems," in 2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC), pp. 203– 210, IEEE, 2017.

[33] Iso, "Pas 21448-road vehicles-safety of the intended functionality," Int. Organization for Standardization, 2019.

[34] Mercedes-Benz, "Safety first for automated driving (safad)." https://group.mercedes-benz.com/innovation/case/autonomous/ safety-first-for-automated-driving-2.html, 2019. Accessed: 19-072023.

[35] Waymo, "Waymo safety report: On the road to fully self-driving." https: //g.co/kgs/N6nC77u, 2018. Accessed: 18-07-2023.

[36] E. Castellano, A. Cetinkaya, and P. Arcaini, "Analysis of road representations in search-based testing of autonomous driving systems," in 2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS), pp. 167–178, IEEE, 2021.

[37] Gambi, M. Muller, and G. Fraser, "Asfault: Testing self-driving car¨ software using search-based procedural content generation," in 2019 IEEE/ACM 41st International Conference on Software Engineering:Companion Proceedings (ICSE-Companion), pp. 27–30, IEEE, 2019.

Log Frequency Analysis for Anomaly Detection in Cloud Environments at Ericsson

Prathyusha Bendapudi
Blekinge Institute of Technology
Karlskrona, Sweden
prbe21@student.bth.se

Vera Simon
Ericsson
Aachen, Germany
vera.simon@ericsson.com

Deepika Badampudi
Blekinge Institute of Technology
Karlskrona,
Sweden
deepika.badampud
i@bth.se

***Abstract*—Log analysis monitors system behavior, detects errors and anomalies, and predicts future trends in systems and applications. However, with the continuous evolution and growth of systems and applications, the amount of log data generated on a timely basis is increasing rapidly. This causes an increase in the manual effort invested in log analysis for error detection and root cause analysis. The current automated log analysis techniques mainly concentrate on the messages displayed by the logs as one of the main features. However, the timestamps of the logs are often ignored, which can be used to identify temporal patterns between the logs which can form a key aspect of log analysis in itself. In this paper, we share our experiences of combining log frequency based analysis with log message based analysis, which thereby helped in reducing the volume of logs which are sent for manual analysis for anomaly detection and root cause analysis.**

***Index Terms*—Log Analysis, Log Frequency Patterns, Anomaly Detection, Machine Learning, Cloud Environments**

I. INTRODUCTION

Software logs have been widely employed in a variety of reliability assurance tasks because they are some of the main sources of information available that record software run-time information. System logs are essential for various reasons, especially for monitoring and troubleshooting in a given computing environment [7]. However, the exponential growth of log data presents challenges for manual analysis, leading to the development of automated log analysis tools that enhance efficiency and effectiveness.

Traditional methods of detecting anomalies in logs involve using manual techniques such as searching for specific keywords (such as "failed", "exception", and "error") or match- ing rules to identify potentially problematic logs associated with system issues [2]. However, with the growing volume and variety of logs generated by modern software systems, these manual approaches are becoming increasingly impractical due to their labor-intensive nature [2].

Automated log analysis tools for different phases such as logging, log compression, log parsing, and log mining exist [5]. In the log mining phase, statistics, data mining, and machine learning techniques can be employed for automatically exploring and analyzing large volumes of log data to obtain meaningful patterns and informative trends. In existing log analysis techniques, the log analysis is primarily based on log messages [5]. Little consideration is given to the timestamps of the logs, which might also serve as an important source of information with regard to the temporal pattern displayed by a set of logs [1]. Few solutions focused on identifying temporal patterns using deep learning methods in log analysis [3, 1]. However, the time frequency patterns displayed by the logs were not correlated to the processes indicated by these logs and the messages displayed by the logs. Baril et al. developed a new model that captures temporal deviations by means of a sliding window data representation [1]. However, the model they developed used a semi-supervised learning approach, where new temporal patterns were compared to past temporal patterns, which were devised to be 'normal' and anomalies were predicted based on the deviation between old and new temporal patterns. Due to the continuous evolution of software processes and ever-changing patterns in log occurrences, an unsupervised learning approach could be beneficial for log-based anomaly detection. Ericsson developed an AI-assisted tool, Lexicon, for automated log based anomaly detection and root cause analy- sis in 5G Cloud network infrastructure. By employing ma- chine learning algorithms like clustering and TF-IDF (term frequency-inverse document frequency), Lexicon automates anomaly detection and root cause identification based on log messages (identifier-based approach). It operates in two modes: monitoring and troubleshooting, facilitating proactive issue detection, and retrospective analysis.

Lexicon takes logs generated from different processes in different cloud environments of Ericsson (Kubernetes pods creation, VM creations, health checks, application tests, etc.), and analyzes these logs primarily based on the log message. Lexicon also detects erroneous logs based on the messages displayed by the logs. It categorizes all the input logs into different groups (log types) based on the similarity of their message and the process that generated these logs. This categorization is done for both normal and erroneous logs. Lexicon assigns a 'normal value' to each log, which is on a scale of -1 to 10 based on how likely the log is to be a normal log without any anomaly or error. Here, an anomaly refers to any unexpected behaviour or error in the associated cloud environment. The details on the normal value in the context of Lexicon are provided in Table I. Lexicon sends the logs with very low normal values to system experts for

manual analysis to understand the root cause of the error that the log might represent. Although Lexicon reduces the manual effort involved in error detection and root cause analysis to a great extent, the combination of the timestamp-based analysis approach with Lexicon's existing identifier-based approach can prove to be beneficial in further reducing the manual effort involved in anomaly detection and root cause analysis.

The aim of this study is to streamline log analysis for root cause detection in Ericsson's cloud environment by reducing the system experts' effort involved in log analysis. This is achieved by combining the two approaches of log analysis: Identifier-based and Timestamp-based. Once Lexicon classifies the logs based on an identifier-based approach, we further classify them using a timestamp-based approach, which includes identifying frequency patterns in logs corresponding to different processes and flagging deviant logs as anomalies.

II. RELATED WORK

Previous research includes developing new approaches for automated log analysis, however, the existing ap- proaches consider only one approach of analysis, i.e., either identifier(message)-based or timestamp-based [6]. We wanted to see if we can improve the efficiency of log analysis-based anomaly detection and root cause analysis by combining both approaches. Moreover, the different log analysis tools available currently, such as splunk, logSayer [13], log assist [11], logFlash [9] provide automated log analysis solutions, but none of them explored the timestamp-based approach yet. Most automated log analysis tools concentrated on identifier-based methods, demonstrating a gap in explor- ing the timestamp-based aspect. Prior research, such as that by Zaman et al. [12], presented tools like SCMiner that localize system-level concurrency faults through log partitioning. While effective in detecting concurrent system failures, SCMiner primarily addresses known system failures and lacks consideration for detecting anomalies based on their time of occurrence. Similarly, LogAssist by Chen et al. [11] focus on summarizing logs to condense large log files into informative summaries through log partitioning. However, it does not consider the detection of anomalies based on their time frequency patterns. The proposed log analysis in this paper advances log partitioning by combining it with feature extraction, forming a good base for further anomaly detection. Grund et al. [4]'s approach, grounded in clustering and statistical methods, abstracted logs automatically. In comparison, this paper builds on these methods

III. STUDY DESIGN

In this section, we describe the steps followed to streamline log analysis for root cause detection in Ericsson's cloud environment. The steps conducted in this study are illustrated in Figure 1.

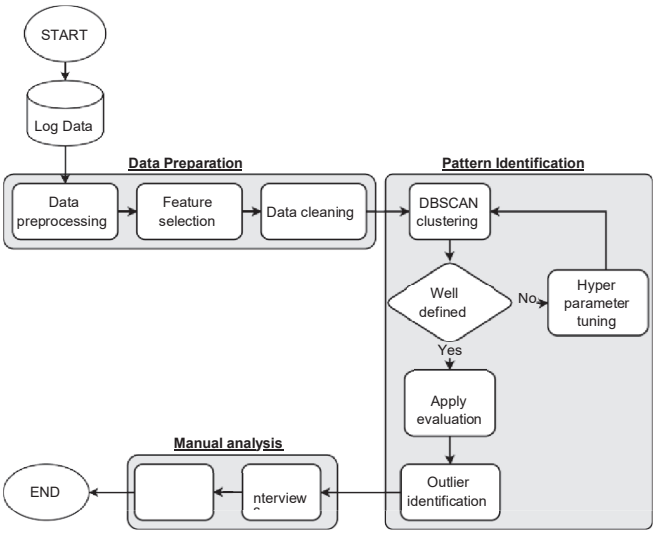


Fig. 1. Research workflow

A. Data Preparation

The initial dataset comprises seven JSON files, each encompassing a complete day's worth of system logs. The dataset contains an extensive collection of approximately 10 million logs which were initially analysed by Lexicon using the identifier based approach, offering a rich source of data for analysis. In each log, several fields were captured in a structured manner, including a unique log uuid, log type, log input type, timestamp, message, and cleaned message, among others. In order to prepare the initial dataset for comprehensive analysis, a series of data preprocessing steps were implemented. The seven JSON files, each representing a single day's log data, were combined into a single dataset to prepare the logs to be grouped on the basis of their log type. A Python script was written to extract and transform the structure of the log entries and store all these entries in a data frame. Each column within the data frame corresponds to a specific log field, such as log uuid, log type, message, timestamp, and many more.

Since it is computationally complicated to deal with all the fields of a huge number of logs, necessary fields were extracted before proceeding with the analysis. Log ID, log type, timestamp, normal value, and message were the required fields for the frequency analysis. Important fields from the log data were selected for the analysis based on what would support the frequency analysis and what log fields would be needed for the system experts to analyze the logs efficiently. A Python script was written to extract the necessary fields from the data set and store them in a separate data frame. While forming this data frame, the logs were grouped according to their log types. Approximately 2000 log files were generated, each of which contained logs pertaining to a particular process in the cloud environment.

The log data contained no null values or significant anoma-

lies that needed removal or replacement. For our analysis, we excluded some log files that contained fewer than 100 logs. This was because we had sufficient examples of log types with more data, which would lead to a more representative result for this kind of analysis. Also, one of the targets of this research is to reduce the amount of logs sent for manual analysis. It is more effective if the analysis is run on the log types that appear several hundreds of times.

The first author converted 7 JSON files, each representing logs from one day of the week, into approximately 2000 files. Each of these new files contained logs of one log type. Then the first author excluded files with less than 100 logs from the 2000 files.

The final dataset consists of the fields listed in table I.

TABLE I
LOG FIELDS

Log uuid	It is a unique identifier for each log.
timediff	It is the time difference between the current log and its preceding log in seconds.
log type	It is a hash ID that is unique to each log type, which depicts a particular process occurring in the cloud environment.
normal value	It is a value assigned to a particular log type by Lexicon according to the degree of normalcy shown by the log occurrences on a scale of -1 to 10, where 0 means that the log is abnormal/erroneous, and 10 means that the log is normal. A normal value of -1 indicates that the log hasn't been seen before and is new to Lexicon.
message	The raw message displayed by the log

B. Pattern Identification

The prepared data set consisted of log files, each log file consisting of logs corresponding to one log type, which represents a particular process in the cloud environment. It was necessary to identify the temporal patterns of these logs in order to proceed with the study. DBSCAN algorithm was used to identify the frequency patterns in each log file. Traditional clustering methods like K-means and KNN may struggle with identifying time-based patterns, especially when clusters occur at irregular intervals [10]. In these clustering algorithms, the number of clusters needs to be mentioned explicitly, and since there was no prior information on how many temporal patterns could be identified in one log file, it was not possible to use them for this analysis. DBSCAN, however, is useful in detecting clusters based on temporal proximity.

Each log file, representing a specific log type, was sequentially loaded for clustering. Parameters such as epsilon(ε) and min_samples were meticulously calibrated to optimize clustering results, with an iterative hyperparameter tuning process ensuring alignment with the temporal dynamics inherent in the log data. The importance of each hyperparameter in the context of log frequency analysis is as follows:

- **Epsilon (ε) -** Defining Neighbourhood Radius: The epsilon parameter served as a pivotal determinant in delineating the radius within which data points were considered neighbors. In the context of log frequency analysis, epsilon played a crucial role in capturing the temporal proximity of log occurrences. A range of epsilon values were explored with meticulous tuning to strike a balance between granularity and inclusivity.
- **Min_samples -** Determining Cluster Density: This parameter emerged as a vital factor in shaping the clustering process. It dictated the minimum number of data points required to constitute a cluster, a critical consideration given the varied frequencies of log occurrences. For example, some log files can have 100 logs but some other can have 500 logs. Also, in each log file, a number of logs can have one temporal pattern and some other logs can have a different temporal pattern. In log frequency analysis, fine-tuning min_samples is crucial to adapt the algorithm's sensitivity to density fluctuations displayed by different log types.
- **Custom Distance Metric -** Tailoring for Log Characteristics: The custom distance metric was intricately crafted to encapsulate the temporal dissimilarity between log occurrences, a fundamental aspect in log frequency analysis. The main challenge faced while implementing DBSCAN on the log data was that each log type had different time intervals between consecutive logs, and it was not possible to use the same set of hyperparameters for all the log types and achieve the same efficiency in clustering. A fixed inter-cluster distance could not be used for all the clusters in all the log files. For a cluster in which the time difference between consecutive logs is 1 second, a distance of 1 second will be huge. On the contrary, for clusters where the time difference is in minutes or hours, 1 second distance will be insignificant. Usage of this custom distance made it possible to use constant values for the hyperparameters in every cluster, especially for epsilon, which otherwise would have to be tuned for every log type. Using a custom distance metric enabled normalizing the variance within the clusters based on the time difference between consecutive logs.

The identified patterns were systematically analyzed, considering cluster statistics and visualizations such as scatter plots. We considered evaluation metrics to serve as objective criteria for selecting log files for in-depth analysis. Details about the evaluation metrics:

- Mean by Standard Deviation value: This metric is used to measure the dispersion among the data points in a cluster. A higher dispersion means that the clusters are not well defined, and the gap between data points is bigger. A low mean by standard deviation value indicates that the clusters are well-defined.
- Outlier by Datapoint count ratio: This metric is the ratio of the number of outliers in a log file to the total number of logs in the log file.
- Silhouette score: This metric measures how well a dataset is clustered by measuring intra-cluster similarity and inter-cluster discrimination.

Subsequently, outlier logs were extracted for further manual analysis, aimed at facilitating error detection within the log data.

C. Manual Analysis

The manual analysis phase commenced after applying the DBSCAN clustering algorithm and identifying outliers us- ing predefined evaluation metrics. These metrics guided the selection of files that could be sent for detailed analysis. The manual examination aimed to reveal the relation between specific log messages and temporal patterns shown by them, providing insights into error detection and root cause analysis implications. All the logs that deviated from the identified frequency patterns were termed outlier logs. The legitimacy of the outliers was supported by using evaluation metrics for the DBSCAN clustering algorithm. Some of the outlier logs were taken and manually analyzed to identify the cause of the deviation. The outlier logs were then sent to system experts to learn more about these logs and to identify if these outliers indicated anomalies in the processes.

Semi-structured interviews [8] were conducted with system experts to extract insights into manual log analysis. Practitioners from Ericsson were selected for the interviews based on expertise in log analysis and domain knowledge. The interviews aimed to understand the relevance of log frequency analysis for reducing manual effort and to explore connections between log processes and temporal pat- terns. A collaborative effort with manual log analysts followed, involving a curated selection of log files shared for analysis. Analysts were briefed on the current functioning of Lexicon and the proposed improvements to log analysis before conduct- ing a thorough manual analysis. Subsequent semi-structured interviews enabled analysts to share their observations and insights, focusing on potential errors indicated by logs, outlier significance, and the approach's feasibility of reducing man- ual workload. Interview topics ranged from the professional backgrounds of the practitioners to the methodologies used for log analysis and encountered challenges. The aim of the interviews was to understand the impact of the proposed approach of log frequency analysis, reviewing outlier logs, and envisioning its integration into existing workflows. In addition, the intention was to get feedback to refine the approach and foster collaboration between automated techniques and human expertise for effective anomaly detection and manual root cause analysis in cloud environments.

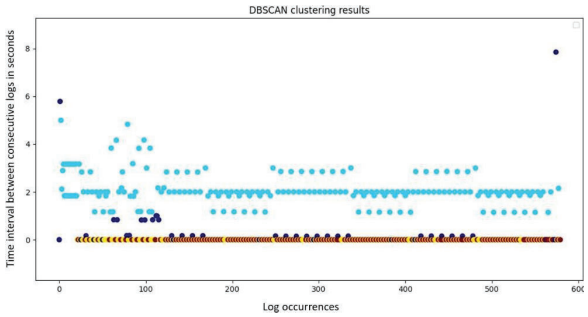
IV. RESULTS AND FINDINGS

DBSCAN clustering algorithm was implemented on the log files to identify the frequency patterns in the form of clusters. To verify that the clusters formed were accurate enough to depict the frequency patterns, hyperparameter tuning was employed to maximize the accuracy and efficiency of the algorithm and make the algorithm suitable for all log files at once. After iterative tuning of the hyperparameters, the final hyperparameters were:

- Epsilon value of 0.5
- Min_samples value of 20% of the total number of logs in each log file

Fig. 2. Logs with definite temporal patterns

- Custom distance metric: abs(x - y) / mean([x, y]), where x and y are two consecutive points in the log dataset.



A. Outcome of the timestamp-based log analysis:

After implementing the clustering algorithm, it was observed that while most of the log types displayed temporal patterns, a few log types had logs that did not adhere to a time-frequency pattern. Visual representation of some clusters is shown in Figures 2, 3, 4 where each dot on the graph represents one log in the log type. The variation in the colors of the points represents different clusters, where each cluster has logs that adhere to a specific temporal pattern. Figure 2 represents a log file which contains logs that are related to egress in a particular cloud cluster, i.e., the process where containers reach and communicate with external resources to get a particular task done in Ericsson's CNF Cloud environment. According to the clusters, the most common time intervals between the logs were recorded to be between 1-5 seconds, with a domination of 2-second time intervals. The graphical representation also shows that many logs had a time interval of 0 seconds, i.e., many logs occurred in a burst with no time difference among each other. According to domain experts, these logs are supposed to occur at regular time intervals, as the process depicted by these logs is a timed process. Thus, all the logs that fell in the red cluster (0 seconds time interval) were deduced as erroneous. There are some outlier points (depicted in navy blue in the figure) that do not follow any of the identified temporal patterns. They might be alarming, especially if it is important for the logs to occur in a timely manner. Upon being sent for further analysis, the logs associated with the outliers were found to be erroneous. This analysis was carried out as a part of the interview process. For example, the log represented by the navy blue dot on the top of the graph, which has an eight-second time interval with its preceding log, was found to be problematic after further analysis.

Figure 3 denotes a clustering result of a specific log file containing logs that do not follow any temporal pattern. The logs occur with varying time intervals ranging from 0

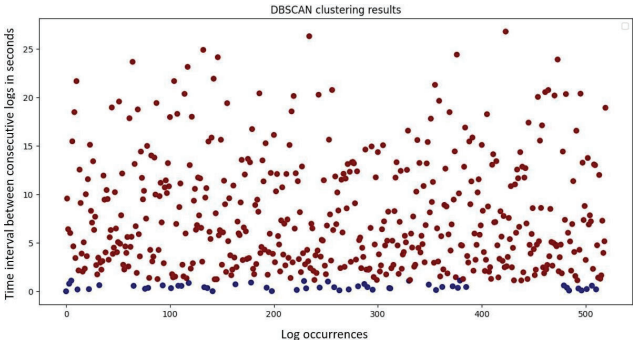


Fig. 3. Logs with no definite temporal pattern

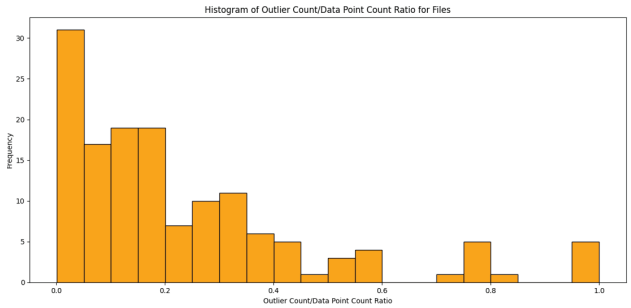


Fig. 5. Histogram representing outlier/datapoint count ratio for different log files

to 30 seconds, with most logs occurring below 15 seconds after the preceding log. However, a proper cluster depicting regular time intervals could not be formed by the clustering algorithm, as there weren't enough log occurrences that followed a regular pattern. The dots in blue represent the logs that occur immediately after their preceding log. In contrast, the dots in red represent the logs that occur after a certain time interval after their preceding log. The logs depicted in the above graph are related to a manually triggered query event. Thus, it is natural that the logs which represent this process will not occur in a timely manner.

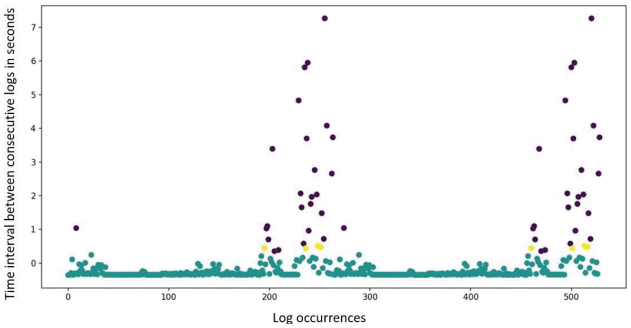


Fig. 4. Logs with temporal patterns and outliers

Figure 4 denotes a clustering result of logs that have both - temporal patterns and scattered logs. On analyzing the messages displayed by these logs, it was found that these logs belong to a process related to load balancing of a containerized application. The message displayed by the log did not show any error, but the logs were occurring continuously without any gap (represented by the sea green colored points). This immediate occurrence of a burst of logs is alarming with respect to the domain of these logs, and thus, it might indicate an error. On the contrary, the logs shown by the purple points have a consistent pattern with varying time intervals, which shows that the load balancing in the container is running well. It is noticeable from the graph that the pattern represented by the purple points repeats after 200 log occurrences. It is not necessarily a pattern with regular log occurrences, as the time differences between consecutive logs ranged from 1 second to 7 seconds in this pattern. Still, it was interesting to send the logs of this log type for further manual analysis to identify the domain of the logs and see why this pattern has occurred. In a similar manner, many other log files were analyzed using DBSCAN clustering. Here, the logs of each log type were fed to the clustering algorithm, and time patterns in each log file were identified. Most of the identified patterns helped in analyzing the logs based on their timestamps and enabled the detection of hidden errors and abnormal behavior.

Our algorithm showed promising clustering results, as it could identify temporal patterns in the logs of different log types. The best way to validate the clustering results and the overall idea of log frequency analysis was to involve human evaluation in the clustering algorithm results. As it was not possible to send all the files for manual analysis due to the limited availability of system experts and a huge number of log files, certain log files had to be selected that showed promising clusters and an optimal number of outliers. To decide which files are most suitable for further manual analysis, a few evaluation metrics, such as *outlier/datapoint count ratio* and *silhouette score* were used.

The selection criteria employed by the evaluation metrics to select the log files for outlier analysis and manual log analysis are:

- Low Mean/Std value
- Outlier/datapoint count ratio lower than 10%
- Silhouette score close to 1.0 (for files with multiple clusters)

Figures 5 and 6 provide an overview of the evaluation metrics, which helped in evaluating the clustering efficiency and choosing the files for further manual analysis. These histograms were generated to analyze the distribution of log files that fall under different values of the evaluation metrics, i.e., *outlier by data point count ratio* and *silhouette score*. The processed data set consisted of more than 200 log files, and sending all these files for manual analysis was impractical. Evaluation metrics were therefore used to select the files that can be sent for manual analysis. Having a visual representation of these metrics further facilitated a calculated selection of log files.

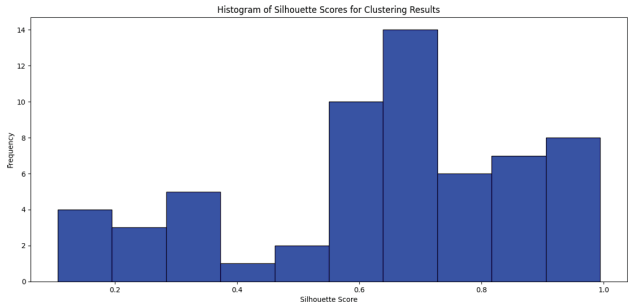


Fig. 6. Histogram representing silhouette scores of the log clusters

Figure 5 shows the distribution of the ratio of the number of outliers to the total number of logs in each file. The X-axis represents the outlier count/Data point count ratio. Here, the ratio of 1.0 indicates that all data points were outliers, and 0 indicates the absence of outliers. Keeping the log files with no outliers aside, the majority of the log files had a ratio of less than 20% (0.2 on the x-axis). Meaning that 20% of the logs in a log file were outliers. Such logs would be more appropriate to consider for anomaly detection using log frequency analysis. Therefore, we considered focusing on logs with less than 20% outliers for manual analysis.

Figure 6 shows the distribution of the silhouette score in log files containing more than one cluster. The histogram shows that the majority of the log files had a silhouette score of 0.7. However, a silhouette score close to 1 indicates distinct clusters. Hence, a combination of files that had silhouette scores of 0.7-0.9 was considered for manual analysis.

We conducted interviews involving Ericsson practitioners having experience with domain knowledge of the CNF cloud environment and manual log analysis for error detection and root cause analysis. Before the interviews, relevant log files and identified outliers were sent to the participants. Partic- ipants were provided with background information on the log analysis tool, the research on frequency analysis, and the specific context of the study.

The interview consisted of 4 practitioners with varying experiences and responsibilities to consider different perspectives on the log frequency analysis and its impact. The information of the participants is listed in Table II.

TABLE II
INTERVIEW DEMOGRAPHICS

Questions	**Participant 1**	**Participant 2**	**Participant 3**	**Participant 4**
Experience	9 Years	4 years	5 years	2 years
Role	Lead Cloud Solutions Architect	Solutions Architect	Senior Cloud Technology Developer	Cloud Technology Developer
Tasks	Higher level supervision of log-based error detection	Manual log analysis	Development of services in cloud environment	Development of Lexicon tool

B. Feedback on the timestamp-based log analysis:

The System experts, after analyzing the outlier logs of each log type, were able to find that the outlier logs indeed indicated anomalous behavior in the associated process. They felt that instead of analyzing all the logs pertaining to the failed process, only analyzing the logs that deviated from the frequency pattern reduced the effort involved in error detection by condensing the number of logs sent for root cause analysis to a great extent. This approach can also be used for proactive system monitoring and predictive maintenance of systems.

The interview structure along with the results consisting of practitioners' perceptions of the revised log analysis is provided below:

1) **Interviewee summary:** We started the interview by asking participants about their experience, roles, and the tasks they are involved in. Table II summarizes the introduction of the interviewees.
2) **Understanding the current process of Log analysis:** Participants were asked about the typical process followed to analyze logs and the problems faced while analyzing these logs for error detection analysis before proceeding with the explanation of our approach.
   Responses from the Participants: The experts analyze log messages, such as '*debug´* and '*error´* from a set of logs associated with a particular process. However, they also note that sometimes, *"the logs might not be showing any suspicious message, which makes it difficult to comprehend where the actual error is coming from"*. A Cloud Solutions Architect added that "*Another problem that we face regularly is the huge number of logs we need to analyze to get to the root cause of the error. it gets tedious sometimes.*"
3) **Explaining the revised Frequency Analysis Approach:** Participants were given a briefing about the revised log frequency analysis approach, after which they were asked if they understood the new approach, and if they perceived any benefits or limitations regarding revised the frequency analysis.
   Responses from the participants: During the interview, the participants were informed about the frequency analysis approach and its potential integration into the current workflow of the Lexicon. The solution architects who are involved in supervision of log analysis highlighted that it is not practical to manually examine the timestamps of a large number of logs. Hence, they are rarely considered in the manual analysis. However, they added that "*not looking at timestamps of the logs might lead to neglecting some logs which might be erroneous, but do not show any error message*". Furthermore, A senior cloud technology developer mentioned that "*I feel this frequency analysis approach will be beneficial in reducing the number of logs that we need to deal with for error detection because identifying timestamp-based patterns and eliminating the logs that adhere to a pattern can enable better efficiency of log analysis.*"

4) **Review of Outlier Logs:** The log files and their outliers, which were identified as a result of the clustering process, were shared with the practitioners two weeks before the interviews for them to have a thorough understanding of the logs. During the interviews, they were asked about their understanding of the domain of these logs and their experience with detecting errors using the outlier logs alone.
   Responses of the participants: Participants analyzed the log files and their corresponding outlier logs. All interviewees mentioned that they were able to get to the root cause of the error by just analyzing the outlier logs in most cases. One interviewee further elaborated that *"Out of the 5 log files provided to me, in 4 files, I could trace the error by just analyzing the outlier logs, and analyzing the temporal logs [not outliers] did not yield in the identification of extra errors."* In some cases, they could not identify the root cause but could find the error by analyzing the logs that occur immediately before and after the outlier logs, in the given cloud system environment. X added that "*the neighboring logs showed problematic behavior in the cloud environment, which led to delay of the selected [outlier] logs. In fact, this way we could catch hold of some errors which might not be visible otherwise."*
5) **Error Detection and Correlation:** Participants were asked to discuss the extent to which the outlier logs helped in error detection.
   Responses from the participants: The participants deemed that most log files' outliers were associated with errors. However, they felt that the correlation of error detection with the outliers of frequency analysis would also depend on the domain of the process depicted by the logs. For example, it is important that logs pertaining to automated processes such as health checks follow a time pattern, but the same is not true for logs depicting manually triggered events. X further elaborated "*Out of the files sent to me, one file was related to fetching data from a database deployed in one of the containers in the CNF environment. The logs mostly followed a temporal pattern, but it was not necessarily a timed process, because it can be manually triggered too. Thus, analyzing the outliers in such cases wouldn't help much, as the logs not following a time pattern in such cases don't necessarily mean that there is an error.*"
6) **Impact on Manual Analysis:** Participants were asked whether frequency analysis could reduce the manual analysis workload and how it might fit into their existing workflows.
   Responses from the participants: The participants felt that this approach would indeed help in reducing the manual effort, as in most cases, analyzing the outliers alone was enough to detect the errors. X added "*If all the logs were to be analyzed, it was a very tedious task for us. We get to the root cause eventually, but the work involved takes a lot of time. For example, one of the files which I analysed as a part of this interview, contained close to 150 logs. whereas, the outlier logs in this file were only 9. I could detect the error just by analysing these 9 logs, and tracing their neighbour logs helped me reach the root cause of the error. I checked the remaining 150 logs which followed a temporal pattern, to see if there are any underlying errors indicated by them. I could not find anything new which was not found by analysing these 9 outlier logs. So I would say this would increase the efficiency of my daily workflow to a great extent.*"
7) **Suggestions for Improvement:** Participants were asked to offer suggestions for enhancing the accuracy of the approach and additional features.
   Responses of the participants: The participants were satisfied with the frequency analysis approach and suggested integrating it into the Lexicon pipeline. A cloud technology developer further added that "*it is better to deduce a way so that you can implement it in Lexicon, which will cater to a lot of analysts, and will improve the efficiency of Lexicon in root cause analysis of errors.*"
8) **Future Considerations:** Participants were asked to share insights on potential applications beyond error detection and challenges in implementing the approach.
   Responses from the participants: During the interview, one of the participants working as a senior cloud solution architect shared an interesting use case of frequency analysis to predict maintenance needed in a cloud envi- ronment. He added "*I think since we are going to use this approach for error detection, we can go a step forward and also use this approach for predictive maintenance of the associated cloud environments. This way, we can predict arising errors before their occurrence and take proactive measures to prevent them. This will further help us, as we will not have to deal with such a huge amount of errors if they can be prevented beforehand.*"

C. Impact on manual analysis effort:

Before conducting frequency analysis, experts would have to analyze 2252 logs for anomaly detection across 20 log files. However, after performing the log frequency analysis, the number was reduced to 167 logs. These logs were identified as outliers and were helpful in tracing errors. The remaining 2085 logs, identified as normal (non-outlier) logs, did not lead to any errors in the associated processes. We found that the manual effort involved in analyzing the logs in our sample was reduced by 92.6%. This percentage is calculated by comparing the number of logs analyzed before (2252) and after (167) the introduction of log frequency analysis.

True Positives and False Positives: According to the manual analysis by system experts, 142 out of 167 outlier logs indicated anomalies, resulting in a true positive percentage of approximately 85% and a false positive percentage of approximately 15%.

V. LIMITATIONS AND THREATS TO VALIDITY

The dataset used for log frequency analysis consisted of logs that occurred in a specific time frame, which limits the results of this analysis to the sample considered in the study.

A. External Threats

The log files used for manual analysis of logs in our study were limited, as it was not possible for system experts to analyze a large amount of logs. The number of log types identified in the initial data set was ~2000, of which some types were excluded because the number of logs that these files had was not sufficient to identify frequency patterns. Out of more than 200 log files that made it to the final step of the automated analysis, 20 files were selected that could be sent to system experts for further manual analysis based on the evaluation metrics. All the files used for the manual analysis had an outlier log ratio of 10-20%, and log files that had varying percentages of outliers were not explored for manual analysis. The extent of effort reduction in manual log analysis can change with changing log data and thus cannot be generalized. However, we expect that the logs that have 10- 20% outliers based on timestamps can have similar results in terms of reduced manual effort. However, analyzing more log files manually may lead to better insights into the effectiveness of the log frequency analysis.

B. Internal Threats

The logs used in this Thesis were limited to one cloud environment of Ericsson, which makes the results of this analysis domain specific. Further research on log frequency patterns needs to be conducted to generalize the validity of this research to other domains in the field of Computing.

VI. CONCLUSIONS AND FUTURE WORK

Our experience has shown that analyzing logs for error detection and root cause analysis can be made more efficient by using a log frequency analysis approach that combines a timestamp-based and identified-based method. Collaborat- ing with system experts through interviews and manual log analysis has been instrumental in validating the efficacy of this approach. This collaborative approach has the potential to reduce manual analysis efforts and improve anomaly detection. In the future, the research could focus on refining outlier selection criteria and incorporating advanced machine learning models to automate the process of outlier extraction using log frequency analysis. The outlier logs can be identified continuously in streaming log data for the proactive identification of anomalous behavior in the associated process. This way, this analysis can be extended to all log types without restricting it to the log files that have a certain percentage of outlier logs.

REFERENCES

[1] Xavier Baril, Oihana Coustie´, Josiane Mothe, and Olivier Teste. 2020. Application Performance Anomaly Detection with LSTM on Temporal Irregularities in Logs. In *Proceedings of the 29th ACM International Conference on Information & Knowl- edge Management* (Virtual Event, Ireland) *(CIKM '20)*. Associa- tion for Computing Machinery, New York, NY, USA, 1961–1964. https://doi.org/10.1145/3340531.3412157

[2] Zhuangbin Chen, Jinyang Liu, Wenwei Gu, Yuxin Su, and Michael R. Lyu. 2021. Experience Report: Deep Learning-based System Log Analysis for Anomaly Detection. https://arxiv.org/abs/2107.05908

[3] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. 2017. DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (Dallas, Texas, USA) *(CCS '17)*. Association for Computing Machinery, New York, NY, USA, 1285–1298. https://doi.org/10.1145/3133956.3134015

[4] Mostafa Farshchi, Jean-Guy Schneider, Ingo Weber, and John Grundy. 2015. Experience report: Anomaly detection of cloud application operations using log and cloud metric correlation analysis. In *2015 IEEE 26th International Symposium on Software Reliability Engi- neering (ISSRE)*. 24–34. https://doi.org/10.1109/ISSRE.2015.7381796

[5] Shilin He, Pinjia He, Zhuangbin Chen, Tianyi Yang, Yuxin Su, and Michael R. Lyu. 2021a. A Survey on Automated Log Analysis for Reliability Engineering. *ACM Comput. Surv.* 54, 6, Article 130 (jul 2021), 37 pages. https://doi.org/10.1145/3460345

[6] Shilin He, Pinjia He, Zhuangbin Chen, Tianyi Yang, Yuxin Su, and Michael R. Lyu. 2021b. A Survey on Automated Log Analysis for Reliability Engineering. *ACM Comput. Surv.* 54, 6, Article 130 (jul 2021), 37 pages. https://doi.org/10.1145/3460345

[7] Shilin He, Xu Zhang, Pinjia He, Yong Xu, Liqun Li, Yu Kang, Minghua Ma, Yining Wei, Yingnong Dang, Saravanakumar Rajmohan, and Qingwei Lin. 2022. An Empirical Study of Log Analysis at Microsoft. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Singapore, Singapore) *(ESEC/FSE 2022)*. As- sociation for Computing Machinery, New York, NY, USA, 1465–1476. https://doi.org/10.1145/3540250.3558963

[8] S.E. Hove and B. Anda. 2005. Experiences from conducting semi-structured interviews in empirical software engineering research. In *11th IEEE International Software Metrics Symposium (METRICS'05)*. 10 pp.–23. https://doi.org/10.1109/METRICS.2005.24

[9] Tong Jia, Yifan Wu, Chuanjia Hou, and Ying Li. 2021. LogFlash: Real-time Streaming Anomaly Detection and Diagnosis from System Logs for Large-scale Software Systems. In *2021 IEEE 32nd International Symposium on Software Reliability Engineering (ISSRE)*. 80–90. https://doi.org/10.1109/ISSRE52982.2021.00021

[10] Hari Kanagala and V.V. Krishnaiah. 2016. A comparative study of K-Means, DBSCAN and OPTICS. 1–6. https://doi.org/10.1109/ICCCI.2016.7479923

[11] Steven Locke, Heng Li, Tse-Hsun Peter Chen, Weiyi Shang, and Wei Liu. 2022. LogAssist: Assisting Log Analysis Through Log Summarization. *IEEE Transactions on Software Engineering* 48, 9 (2022), 3227–3241. https://doi.org/10.1109/TSE.2021.3083715

[12] Tarannum Shaila Zaman, Xue Han, and Tingting Yu. 2019. SCMiner: Localizing System-Level Concurrency Faults from Large System Call Traces. In *2019 34th IEEE/ACM Interna- tional Conference on Automated Software Engineering (ASE)*. 515–526. https://doi.org/10.1109/ASE.2019.00055

[13] Pengpeng Zhou, Yang Wang, Zhenyu Li, Xin Wang, Gareth Tyson, and Gaogang Xie. 2020. LogSayer: Log Pattern-driven Cloud Component Anomaly Diagnosis with Machine Learning. In *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*. 1–10. https://doi.org/10.1109/IWQoS49365.2020.9212954

Many a Little Makes a Mickle: On Micro-Optimisation of Containerised Microservices

Zheng Li
School of Electronics, Electrical Engineering and Computer Science
Queen's University Belfast
Belfast, United Kingdom
ORCID: 0000-0002-9704-7651

Abstract— **Performance optimisation is a key to the success of microservices architecture. Correspondingly, many studies have been conducted on optimising orchestration or composition of multiple microservices within different application contexts. Unlike the existing efforts on the global optimisation, we are concerned with the internal optimisation of individual microservices. Considering the loosely coupled nature of individual microservices, their performance improvements could be independent of each other and thus would naturally bring benefits to their composite applications. Driven by such intuitive ideas together with the de facto tech stack, we have been working on micro-optimisation of containerised microservices at the Operation side (i.e., Ops-side optimisation) against the Development side. Based on both theoretical discussions and empirical investigations, our most recent work delivered three micro-optimisation principles, namely *just-enough containerisation*, *just-for-me configuration*, and *just-in-time compilation (during containerisation)*. Our current research outcomes have not only offered new ideas and practical strategies for optimising microservices, but they have also expanded the conceptual scope and the research field of software micro-optimisation.**

Keywords-containerisation; DevOps; micro-optimisation; microservice; Ops-side optimisation; performance engineering

I. INTRODUCTION

Since microservice-based applications may suffer from intrinsic performance penalties due to their distributed nature [1], it has been identified that performance optimisation is a key to the success of microservices architecture [2]. Unlike the existing studies that directly aim at application-level optimisation (e.g., through resource provisioning [3] or microservice placement [4]), we wonder if there are opportunities to focus on every single microservice to optimise. Intuitively, and depending on the application topology, a many-microservice application may obtain enormous benefits if every microservice contributes a little performance improvement. Even if the eventual application benefits are only "small efficiencies" [5], the microservice-level optimisation will still be worth it, because modern (Web) applications are generally performance sensitive, e.g., "every drop of 20 ms... latency will result in a 7–15% decrease in page load times" [6].

Driven by these intuitive ideas, we defined the following research question to unfold concrete investigations and to distinguish from the application-level optimisation studies:

RQ: Can we, and if yes, how do we conduct application-agnostic optimisations to improve the performance of individual microservices?

Inspired by the single-purpose feature of microservices, our current investigation efforts are paid to the micro-optimisation opportunities. Traditionally, low-level optimisation may not be considered as a good idea because it tends to be platform-centric [7]. When it comes to the containerised microservices, the containerisation mechanism encapsulates microservice components and their runtime environment all together, which means that the microservices' platform details (e.g., the OS kernel, word size, and CPU instruction set architecture) are all settled in advance. Then, there will be nothing wrong to conduct platform-centric optimisations in this situation.

Moreover, given the convergence of infrastructure as code and container technologies [8], containerised microservices should be able to enjoy more micro-optimisation opportunities, not only at the Development side (i.e., Dev-side optimisation in source code files) but also at the Operation side (i.e., Ops-side optimisation in Dockerfile, shell scripts, and machine-readable definition files). To verify this new idea, we further narrow down our focus to micro-optimisation with respect to containerisation. At the time of writing, our ongoing work has developed three micro-optimisation principles, and we name them as *just-enough containerisation*, *just-for-me configuration*, and *just-in-time compilation (during containerisation)*. By reporting and justifying these principles, this paper makes a twofold contribution:

- In theory, this work expands the conceptual scope of, and reveals new research opportunities in the field of, software micro-optimisation. To our best knowledge, this is the first study that advocates and investigates the low-level, Ops-side optimisation.
- In practice, this work suggests new strategies to optimise containerised microservices. These strategies would be increasingly applicable, along with the fast-growing DevOps ecosystem that heavily leverages infrastructure as code and container technologies.

II. RELATED WORK

A. Microservices Optimisation

Exploring "microservice(s) optimisation" in the literature will bring a tremendous number of studies on optimising

orchestration or composition of multiple microservices, within different application contexts. By distinguishing between scalable (e.g., cloud) and restricted (e.g., user device) runtime environments, the existing studies generally formulate microservices optimisation either as resource-provisioning problems or as microservice-placement problems, with different objectives and different solution proposals.

In particular, we have observed a broad range of optimisation objectives including, for example, minimising response time/latency [4], resource consumption/cost [3], [9], energy consumption [10], failure rate [10], etc.; and maximising reliability [4], [9], resource utilisation efficiency [3], network throughput [10], load balancing [4], [9], etc. Correspondingly, the proposed solutions also vary hugely, such as particle swarm optimisation, non-dominated sorting genetic algorithm III (NSGA-III), fine-tuned sunflower whale optimisation algorithm, ant colony algorithm, knowledge-driven evolutionary algorithm, Lagrangian multipliers, etc.

It is worth noting that although some researchers claim to have worked on the performance tuning [11] and configuration adjustment [12] of individual microservices, their research work still measures and refers to the application indicators to conduct the optimisation. In contrast, our research focuses on the application-agnostic optimisation of microservices; and more distinctively, we aim to optimise microservices during their containerisation process before the runtime execution.

B. Software Micro-Optimisation

Since we have not found any study on micro-optimisation of microservices, we consider the generic software micro-optimisation as a related topic to our research.

Software micro-optimisation is generally defined as the source code-level optimisation (e.g., using StringBuilder instead of String in Java) without changing the software architecture, design, and algorithms [13]. According to the literature, the community seems to have opposite opinions about software micro-optimisation. By citing Sir Tony Hoare's famous quote "premature optimisation is the root of all evil" (popularised by Donald Knuth) [5], "we should forget about small efficiencies" has been argued as a best practice of software engineering and even as a rule of programming [7]. Except for the unawareness of software micro-optimisation by some practitioners, the main concern is that micro-optimisation may not be a worthwhile investment of time compared to macro-optimisation [13], [14].

In the meantime, there are also advocates of software micro-optimisation. A direct response to the aforementioned concern is that it is always worth investing developers' time to save software users' time [5]. Another investigation empirically justifies how worthwhile the micro-optimisation can be, by tweaking a piece of code that is responsible for a substantial proportion of the execution time [14]. After all, given the software crisis behind the continuously growing CPU power, it is never enough to emphasise the optimisation of software systems.

We are also convinced of the value of software micro-optimisation, because "any (even small) performance improvement will matter" in modern computing paradigms (e.g., IoT, edge, fog, cloud) [6]. Particularly, we are further concerned with the micro-optimisation at the Ops side instead of Dev side.

III. Three Micro-Optimisation Principles

To better introduce the three micro-optimisation principles, we particularly highlight the justification for each principle description, in separate subsections.

A. Just-enough Containerisation

1) Principle Description: When wrapping up target functionalities into a containerised microservice, the containerisation should include just-enough software components and minimise the installation of just-in-case programs and middleware.

2) Justification: When it comes to deploying a microservice-based application, it has been widely discussed that the co-located microservices in a multi-tenant environment can interfere with and slow down each other due to the competition for non-partitionable resources, and the resource competition may eventually cause unpredictable behaviour and performance degradation of the microservice-based application [15], [16].

In fact, if zooming into an individual microservice, we can also expect to see the resource competition among its co-located components. Recall that each microservice is a (single-purpose) software application and controls its own data [17]. Since a single-purpose application is still composed of multiple components (e.g., codebase modules and a database management system), a containerised microservice can include multiple containers, and each container encapsulates a software component together with the component's entire runtime environment. Therefore, such a multi-container microservice should naturally be recognised as a multi-tenant system.

Furthermore, to support the main functionalities, each containerised software component may also install its enabling services, registry entries, background tasks, drivers, shared libraries, etc. on the fly during the image building process. Thus, the containerisation of any software component would incur extra performance overhead. Even if there is no resource competition by those inactive software components at runtime, the unneeded installations will unnecessarily increase the size of the corresponding microservice, and it has been empirically identified that the unused stuff is the major cause of memory waste [13].

B. Just-for-me Configuration

1) Principle Description: Without changing the codebase and the predefined tech stack, it is worth customising the configurations of microservice components during the containerisation, to better support the microservice's non-functional features.

2) Justification: Aligning with the single-responsibility principle "do one thing and do it well", the loosely coupled microservices are supposed to work (largely) independently on different single purposes, even within the same application context. Since each microservice may have its own and unique runtime characteristics, there does not exist a one-size-fits-all configuration to maximise the potentials of different microservices. For example, without understanding the data needs of individual microservices and accordingly planning respective strategies, the efficient cache configurations for one

microservice (e.g., Django's caching framework and MySQL query cache) could result in inefficient caching and performance degradation for another microservice [18].

Another typical example is when containerising microservices that involve database systems, a microservice can achieve higher performance by optimising its database performance against its specific workload and dataset [19]. For instance, we can create additional indexes to expedite data retrieval for stateless queries. Besides the existing database tuning tips, in this paper we particularly report our experience in tweaking read-only database containers to exemplify and justify the micro-optimisation's effectiveness and efficiency.

According to Docker, the official mechanism of containerising a read-only database is to use the read-only parameter to specify a mounted data volume as read-only.[1] However, mounting data volumes "by punching a hole through the container" has been considered unreliable [20], and thus it has been argued that containers cannot be a secure candidate solution for database [21]. In our project, we advocate pre-baking read-only data into container images to avoid mounting external volumes. By making such a customisation, this unofficial mechanism can not only intrinsically enable read-only databases (because the pre-baked data are immutable in image) but also improve the reliability of database containers (because no "hole" exists through the container).

We have also compared the data retrieval performance between these two mechanisms of containerising read-only databases. The testbed was set up on a clean HP OMEN laptop of model 17-an003la (with Intel's four-core CPU Core™ i7-7700HQ at 2.8 GHz base frequency). After re-installing 64-bit Ubuntu 20.04 as the operating system (OS), we further installed Docker 20.10.2, Python 3.8.5, and Jupyter Notebook 3.8.5. To facilitate observation, we intentionally employed large-size datasets (up to about 160 MB) to measure the latency of data retrieval from MySQL 5.7.34, in order to magnify the performance difference.

Given the experimental results[2] as shown in Figure 1, it is clear that pre-baking data into image can even achieve performance advantage over mounting data volumes. In other words, this micro-optimisation can bring multiple non-functional benefits to read-only database containers as well as their supported microservices

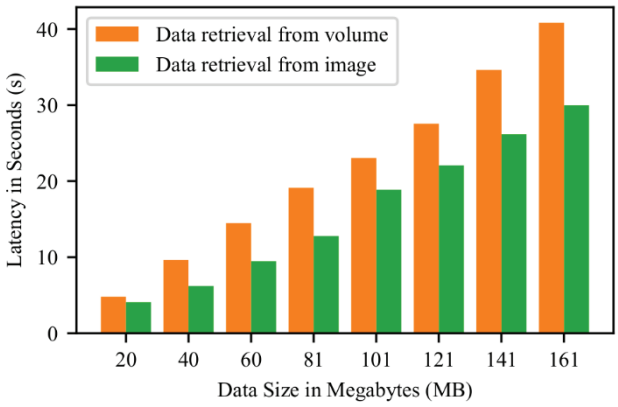


Figure 1. Data retrieval performance comparison between two mechanisms of read-only database containerisation.

C. *Just-in-time Compilation (during Containerisation)*

1) Principle Description: When applicable, it is worth compiling the interpreted microservice components during the containerisation process, to avoid (or at least minimise) the interpretation overhead in the runtime of containerised microservices.

2) Justification: Driven by the needs of reducing development costs cross heterogeneous platforms, "write once, run anywhere" has become a standard practice in software industry. Despite various implementations of this standard practice, the essential enabling technique is to equip the development with a pervasively installed middleware (e.g., a runtime framework or a code interpreter) that abstracts the underlying details of different platforms. Nevertheless, constrained by the No-Free-Lunch theorem of optimisation [22], it is also well known that the middleware-aided code translation will impose a performance penalty whenever it runs.

To alleviate the performance penalty, there emerges a runtime compilation strategy, i.e., translating source code or bytecode into machine code during the first-time execution of a program. However, this will meanwhile introduce a warm-up latency to the executables due to the extra computational overhead for interpreting, compiling, and linking the code.

Recall that one of the best practices of containerisation is to not only import an OS base image but also specify the OS version [23], while the container images must have targeted one or more specific platforms (processor architectures) [24]. In other words, a container's production environment is already (pre-)fixed when preparing its image. Therefore, we shall be able to compile the to-be-containerised microservice components just in time when building container images. In this way, we can further avoid the aforementioned warm-up latency of the relevant microservice components, and ultimately improve the overall microservice performance at runtime.

In addition to the codebase components, the database part of a microservice also has similar micro-optimisation opportunities. Conventionally, given a SQL query, the database system will first convert the query into an execution plan (i.e., a sequence of data access steps) and then execute the plan via interpretation. Since the performance of modern query engines is increasingly dominated by the memory access and CPU usage, there is an emerging trend in dropping interpretation in favour of compilation [25]. As demonstrated by a quantitative study on compiling a set of selected benchmark queries [26], although the performance advantage varies case by case, the query execution after compilation generally outperforms the query execution via interpretation.

Similarly, the execution performance advantage also comes with an extra overhead of compilation, which may make the overall query processing take even longer time. To address this problem, unlike the current research efforts that dominantly aim

[1] https://docs.docker.com/storage/volumes/

[2] The experimental source files and documentation are shared at https://doi.org/10.5281/zenodo.8341856

to expedite runtime and session-specific compilation of dynamic user queries [27], we argue to natively compile the pre-known microservice-specific queries just in time during containerisation, because this will extinguish the runtime compilation overhead when executing the containerised queries. In fact, we have seen promising techniques that are aligned with this idea, e.g., SQL Server supports native compilation of tables and stored procedures into DLLs.[3]

IV. Conclusions and Future Work

Performance optimisation is crucial and valuable for the implementation of microservices architecture. In addition to globally optimising microservice-based applications by adjusting resource provisioning and/or microservice placement, we argue that the internal micro-optimisation of individual microservices would also bring enormous benefits to microservice-based applications. Although still at an early stage, our theoretical discussions and empirical trials have led to a set of micro-optimisation principles with initial validation of their effectiveness and efficiency, at least at the Operation side for containerised microservices.

On the other hand, there are clearly needs to enrich empirical evidence for strengthening our developed principles and for proposing new ones. Therefore, this early-stage research points out two directions toward the immediate future work. Firstly, it is worth keeping trying different micro-optimisation techniques and quantitatively studying their effects in the context of a single microservice. Secondly, it will be helpful to use the third-party application benchmarks [4] to observe and investigate the combined and overall effects of micro-optimising multiple microservices.

References

[1] S. Baškarada, V. Nguyen, and A. Koronios, "Architecting microservices: Practical opportunities and challenges," *J. Comput. Inf. Syst.*, vol. 60, no. 5, pp. 428–436, 2020.

[2] Firdavs, "Quality Attribute Analysis in Microservices Architectures" https://dev.to/firdavsm1901/quality-attribute-analysis-in-microservices-architectures-33li, 13 Apr. 2024.

[3] M. Kumar, J. K. Samriya, K. Dubey, and S. S. Gill, "QoS-aware resource scheduling using whale optimization algorithm for microservice applications," *Softw.: Pract. Exper.*, vol. 54, no. 4, pp. 546–565, Apr. 2024.

[4] G. Fan, L. Chen, H. Yu, and W. Qi, "Multi-objective optimization of container-based microservice scheduling in edge computing," *Comput. Sci. Inf. Syst.*, vol. 18, no. 1, pp. 23–42, Jan. 2021.

[5] R. Hyde, "The fallacy of premature optimization," *ACM Ubiquity*, vol. 10, no. 3, Feb. 2009, art. no. 1.

[6] Z. Li and J. Galdames-Retamal, "On iot-friendly skewness monitoring for skewness-aware online edge learning," *Appl. Sci.-Basel*, vol. 11, no. 16, Aug. 2021, art. no. 7461.

[7] B. Smaalders, "Performance anti-patterns: Want your apps to run faster? here's what not to do." *ACM Queue*, vol. 4, no. 1, pp. 44–50, Feb. 2006.

[8] J. Das, "Making infrastructure as code a better framework with containers," https://blog.aspiresys.com/infrastructure-managed-services/making-infrastructure-as-code-a-better-framework-with-containers/, 28 Sept. 2022.

[9] M. Lin, J. Xi, W. Bai, and J. Wu, "Ant colony algorithm for multi-objective optimization of container-based microservice scheduling in cloud," *IEEE Access*, vol. 7, pp. 83 088–83 100, Jun. 2019.

[10] A. Samanta and J. Tang, "Dyme: Dynamic microservice scheduling in edge computing enabled IoT," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6164–6174, Jul. 2020.

[11] V. M. Mostofi, D. Krishnamurthy, and M. Arlitt, "Fast and efficient performance tuning of microservices," in *Proc. CLOUD 2021*. Chicago, IL, USA: IEEE Press, 05-10 Sept. 2021, pp. 515–520.

[12] H. Dinh-Tuan, K. Katsarou, and P. Herbke, "Optimizing microservices with hyperparameter optimization," in *Proc. MSN 2021*. Exeter, United Kingdom: IEEE Press, 13-15 Dec. 2021, pp. 685–686.

[13] M. Linares-Vásquez, C. Vendome, M. Tufano, and D. Poshyvanyk, "How developers micro-optimize Android apps," *J. Syst. Soft.*, vol. 130, pp. 1–23, Aug. 2017.

[14] A. Trotman and M. Crane, "Micro- and macro-optimizations of SaaT search," *Softw.: Pract. Exper.*, vol. 49, no. 5, pp. 942–950, May 2019.

[15] Y. D. Barve, S. Shekhar, A. Chhokra, S. Khare, A. Bhattacharjee, Z. Kang, H. Sun, and A. Gokhale, "FECBench: A holistic interference-aware approach for application performance modeling," in *Proc. IC2E 2019*. Prague, Czech Republic: IEEE Press, 24-27 Jun. 2019, pp. 211–221.

[16] D. Masouros, S. Xydis, and D. Soudris, "Rusty: Runtime interference-aware predictive monitoring for modern multi-tenant systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 1, pp. 184–198, Jan. 2021.

[17] S. Li, H. Zhang, Z. Jia, Z. Li, C. Zhang, J. Li, Q. Gao, J. Ge, and Z. Shan, "A dataflow-driven approach to identifying microservices from monolithic applications," *J. Syst. Softw.*, vol. 157, Nov. 2019, art. no. 110380.

[18] S. K. Shivakumar, "Web performance monitoring and infrastructure planning," in *Modern Web Performance Optimization: Methods, Tools, and Patterns to Speed Up Digital Platforms*. Berkeley, CA: Apress, Nov. 2020, ch. 7, pp. 175–212.

[19] K. Kanellis, R. Alagappan, and S. Venkataraman, "Too many knobs to tune? towards faster database tuning by pre-selecting important knobs," in *Proc. HotStorage 2020*. USENIX Association, 13-14 July 2020, art. no. 16.

[20] J. Tobin, "Are docker containers good for your database?" https://www.percona.com/blog/2016/11/16/is-docker-for-your-database/, 16 Nov. 2016.

[21] S. Shirinbab, L. Lundberg, and E. Casalicchio, "Performance evaluation of containers and virtual machines when running Cassandra workload concurrently," *Concurrency Comput. Pract. Exper.*, vol. 32, no. 17, Feb. 2020, art. no. e5693.

[22] F. Rabhi, M. Bandara, A. Namvar, and O. Demirors, "Big data analytics has little to do with analytics," in *ASSRI 2015, ASSRI 2017: Service Research and Innovation, ser*. Lect. Notes Bus. Inf. Process., A. Beheshti, M. Hashmi, H. Dong, and W. E. Zhang, Eds. Cham: Springer, Mar. 2018, vol. 234, pp. 3–17.

[23] Y. Wu, Y. Zhang, T. Wang, and H. Wang, "Characterizing the occurrence of dockerfile smells in open-source software: An empirical study," *IEEE Access*, vol. 8, pp. 34 127–34 139, Feb. 2020.

[24] A. Mouat, "Multi-platform docker builds," https://www.docker.com/blog/multi-platform-docker-builds/, Mar. 2020.

[25] T. Neumann, "Evolution of a compiling query engine," *Proc. VLDB Endow.*, vol. 14, no. 12, pp. 3207–3210, Jul. 2021.

[26] A. Rayabhari, "Compilation-based execution engine for a database," https://www.cs.cornell.edu/courses/cs6120/2020fa/blog/db-compiler/, 18 Dec. 2020.

[27] H. Funke and J. Teubner, "Low-latency compilation of SQL queries to machine code," *Proc. VLDB Endow.*, vol. 14, no. 12, pp. 2691–2694, Jul. 2021.

[3] https://learn.microsoft.com/en-us/sql/relational-databases/in-memory-oltp/native-compilation-of-tables-and-stored-procedures?view=sql-server-ver16

[4] https://github.com/delimitrou/DeathStarBench

Towards Real-time Object Detection for Safety Analysis in an ML-Enabled System Simulation

RTOD for Safety Analysis in an ML-Enabled System Simulation

Jubril Gbolahan Adigun[⍟△♦], Patrick Aschenbrenner[¶], Michael Felderer[⍟♠⊥]

[⍟] Department of Computer Science, University of Innsbruck, Austria
Email: {firstname}.{lastname}@uibk.ac.at

[¶] Institute for Astro- and Particle Physics, University of Innsbruck, Austria
Email: {firstname}.{lastname}@student.uibk.ac.at

[△] Ainnov8 Technologies Ltd, Nigeria
Email: {firstname}@ainnov8.com

[♦] Center for Artificial Intelligence (AI) Research Nepal, Nepal
Email: {firstname}.{lastname}@cair-nepal.org

[♠] German Aerospace Center (DLR), Germany
Email: {firstname}.{lastname}@dlr.de

[⊥] University of Cologne, Germany
Email: {firstname}.{lastname}@uni-koeln.de

Abstract — **Machine learning (ML)-equipped critical systems such as collaborative artificial intelligence systems (CAISs), where humans and intelligent robots work together in a shared space are increasingly being studied and implemented in different domains. The complexities of these systems raise major concerns for safety risks because decisions for controlling the dynamics of the robot during the interaction with humans must be done quickly driving the detection of potential risks in form of collision between a robot and a human operator using information obtained from sensors such as camera or LIDAR. In this work, we explore and compare the performance of two You Only Look Once (YOLO) models - YOLOv3 and YOLOv8 - which rely on convolutional neural networks (CNNs) for real-time object detection in a case study collaborative robot system simulation example. The preliminary results show that both models achieve high accuracy (≥ 98%) and real-time performance albeit requiring a GPU to run at such speed as 40FPS. The results indicate the feasibility of real-time object detection in a CAIS simulation implemented with CoppeliaSim software.**

Keywords - collaborative robot, object detection, simulation, machine learning, risk analysis

I. INTRODUCTION

Artificial intelligence (AI), inculcating its fast-growing branch, machine learning (ML) has become a mainstay in many aspects of human life. From its integration in simple non-intrusive systems such as a movie recommendation system to those deployed to reduce human effort, for example, GitHub Copilot[1] which can simplify tedious programming tasks. Moreover, AI models dynamically evolve by learning from large datasets obtained from sensor readings, text corpus as well as continuous interaction with humans. Therefore, their development varies significantly from conventional software systems [1].

This presents a new challenge as software engineers and researchers consider AI engineering as a standalone endeavour from software engineering process. Surely, software engineering is broader and involves tasks such as designing, developing, testing, and maintaining software applications across many application domains e.g., education, finance, healthcare, entertainment among others. Yet, AI engineering, which is a specialism within software engineering, is dedicated to the development of systems equipped with human-like intelligence and that are capable of learning from data and making informed decisions or predictions based on that data [2].

Furthermore, in conventional software systems, defects and malfunctions can and do occur, they are not harmful. However, malfunctions due to the application of ML solutions in safety-critical domains can have devastating impacts on lives, property and systems, like robotic systems that work together with humans in a shared physical space to reach a common goal i.e., collaborative artificial intelligence systems (CAISs) [3]. These systems rely on ML components to process external data, decide on the next action and learn from observations.

Several properties, such as safety, robustness and accuracy are crucial in CAISs [4]. Safety risk assessment in industrial collaborative robots is guided by the ISO/TS 15066 which specifies safety requirements for collaborative industrial robot systems and the work environment. The standard provides four modes of operation to ensure safety within the system: safety-rated monitored stop, hand guiding, speed and separation monitoring, and power and force limiting. The safety risk management of CAISs covered by the ISO 10218-1/ISO

[1] https://github.com/features/copilot

This work was partially supported by the Austrian Science Fund (FWF), under grant I 4701-N

102182/TS 15066 standards [5] aligns with the tenets of Industry 5.0 [6] which aims at a human-centric industry. This can be achieved with CAISs by ensuring that workers are supported by autonomous intelligent machines in a safe manner.

Testing and safety analysis of these complex systems requires new approaches that are both efficient and effective because testing takes up a large chunk of software system development budgets. Simulation testing [7] has been used extensively to this end to conduct quick and less costly analysis of different complex systems since simulators present a simpler and more inexpensive way to test systems.

To address the challenges above, we demonstrate an approach that leverages a simulated CAIS where a human operator works alongside a roof-mounted ML-enabled robotic arm. Since the simulation is within a virtually controlled environment, the experimentation can be done with no risk to human life. Compared to a real-world CAIS system, it is way cheaper, changes and modifications to the system can be added quickly and most importantly, there is no risk for real persons and the physical system. For safety risk management in the simulated system, we employ a safety-rated monitored stop in which the identification of a hazard – in the form of a human hand close to a moving robot arm – initiates an emergency stop (IEC 60204-1 category 2 stop and IEC 61800-5-2 SOS)[2].

The robotic arm is equipped with a vision sensor mounted at its tip for obtaining continuous image streams and detecting potential risks for the human operator and stops the robot, if necessary, via a control script. As a risk measure in the simulation, the distance between the operator's hand and the tip of the robotic arm is used. Specifically, two different YOLO [8] models for object detection methods are applied and both rely on a deep convolutional neural network (CNN) architecture trained from scratch as well as via transfer learning i.e., using a testing a model pre-trained on real life data on data generated from the simulation and compared in terms of speed and prediction accuracy.

CONTRIBUTIONS. Through this paper, we contribute to the existing literature in the safety analysis on ML-enabled systems through the following:

- We highlight the need for real-time object detection as a safety function based on ISO 10218-1/ISO 10218-2/TS 15066 safety risk management standards in collaborative ML-enabled systems.
- We explore the potential of transfer learning of existing object detection models from real to virtual as way to quickly validate the models.
- We present an argument for the use of simulators in safety analysis of ML-enabled systems.
- We carry out an experimental campaign on a simulated industrial CAIS to demonstrate our approach which to the best of our knowledge is a pioneer example in the safety analysis of ML-enabled systems.
- We evaluate the performances of the selected object detection approaches.

The remaining part of the paper is structured as follows: Section II introduces the theoretical background of the applied methods, in Section III, our approach is explained in detail, a brief discussion of the preliminary results ensues in Section IV and finally, in Section V, a summary of this work and an outlook for the future are highlighted

II. BACKGROUND

Making sense of a scene, real or virtual, comes rather naturally to humans and they can decipher objects within an image more easily. Teaching a computer to identify objects usually requires thousands of training samples. With object detection, machines can localize objects within an image. Indeed, a high predictive performance and real-time discernment of objects are necessary for human-like object detection in computers. Accordingly, real-time means that the objects in an image must be detected within the time it takes until another frame surfaces (i.e., consecutive frames of a video stream [9]).

AI researchers have developed different approaches for object detection. Section II-A provides the basis for a popular network in artificial intelligence on which complex systems such as convolutional neural networks (Section II-B), learning and decision-making tasks are built. Section II-C introduces to the usage of (convolutional) neural networks for object detection

A. Neural network

The first proposal of artificial neurons predates the invention of electrical computers and goes back to McCulloch and Pitts [10]. Their neurons have a set of input and control signals, and one output signal. This simple neuron only works with binary data. It is possible to create logical AND, OR and NOT gates, which build a functionally complete set. Therefore, every Boolean function can be built from McCulloch-Pitts neurons.

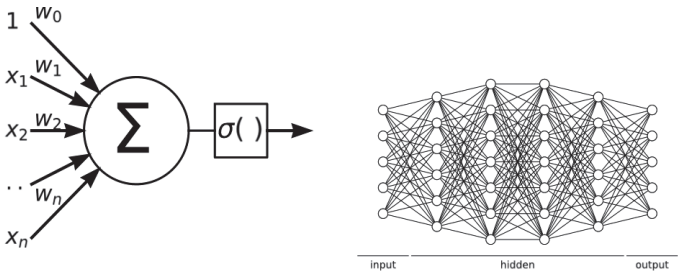


Figure 1. Neuron and fully connected neural network. *Left:* Model of a single neuron. The output of the neuron is the activation of the weighted sum of all inputs $gma(\Sigma_{i=0}^{n} x_i \omega_i)$. *Right:* Example of a fully connected neural network. Each circle represents a neuron. The network has one input layer, four hidden layers and one output layer.

A neural network can be constructed from multiple neurons by arranging them in layers and using the outputs from one layer as the input for the next layer. The first layer is called the input

[2] https://www.controlengeurope.com/article/109959/EN-61800-5-2--more-than-just-Safe-Torque-Off.aspx

layer, the last one output layer and all layers in between are hidden layers. More details about how neural networks work can be found in Werbos [11] and Rumelhart et al. [12].

A simple network is shown in Figure 1. It is called fully connected since every output of a layer is an input of every neuron in the next layer.

The evaluation of a fully connected neural network is done via forward propagation (i.e. evaluate the first layer, use the output to evaluate the second one and so on). The output $\mathbf{o_i}$ of layer i is calculated as follows:

$$\begin{aligned} o_1 &= \sigma_1\left(W^{(1)}x\right) \\ o_2 &= \sigma_2\left(W^{(2)}o_1\right) \\ &\dots \\ o_i &= \sigma_i\left(W^{(i)}o_{i-1}\right) \end{aligned} \tag{1}$$

Note that $W^{(i)}$ is the matrix obtained by stacking all the (row) weight vectors for each neuron ($\mathbf{w}$ in (1)) in the i-th layer and that $\mathrm{gma}_i(\cdot)$ is applied element-wise. The activation function for different layers can be different. Typical nonlinear activation functions that are used are:

$$\begin{aligned} \sigma(x) &= \frac{1}{1+e^{-x}}: \text{logistic sigmoid} \\ tanh(x) &= \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}: \text{hyperbolic tangent} \\ &\max(0, x): \text{rectified linear unit (ReLU)} \\ H(x) &= \begin{cases} 1 & x > 0 \\ 0 & x \le 0 \end{cases}: \text{Heaviside step function} \end{aligned} \tag{2}$$

If a linear activation function is used at each layer, for example, the identity function, then one can always find an equivalent neural network without a hidden layer, since all the matrix multiplications can be simplified to one.

B. *Convolutional neural network*

The task of image recognition can also be performed with a fully connected network, where the different pixel values are used as input for the neural network. However, in images nearby pixels are stronger related than distant ones and object features are present in local parts of an image. Therefore, fully connected networks have many unnecessary connections. Furthermore, the exact position of an object within the image is most of the time irrelevant. Building those principles into a neural network was first done by LeCun et al. [13], where they trained a neural network to recognize handwritten digits in zip codes.

To extract the local features in an image, a convolutional layer is used. In difference to a fully connected layer, it is only connected locally in a regular pattern. The result after the convolution is a large feature vector. Since the exact location is often not important, a pooling layer can be used for simplification, where values are aggregated over a neighbourhood, e.g. via max- or mean-pooling. Neural networks which include one or more convolutional layers are called convolutional neural networks (CNNs).

Over the past decade, the accuracy of CNNs has been greatly improved by adding more layers, creating so-called deep CNNs. One of those models, AlexNet by Krizhevsky et al. [14], won the ImageNet Large Scale Visual Recognition Challenge [15] with a big lead over the second place. Since then, classification tasks are dominated by neural networks over the classical approaches.

C. *Object detection with convolutional neural networks*

Different approaches for object detection with CNNs have been developed and improved, each trying to outperform the other. Prominent examples are YOLO (you only look once) [8], SSD (single shot detector) [16] and RetinaNet [17]. Improvements made to YOLO (YOLOv3) [18] led to a huge performance increase, outperforming the other networks [18] as shown in Figure 2. Over the years, many modifications have been added to the YOLO architecture, with the most recent version being YOLOv8 introduced by Jocher et al. [19].

However, many implementations still use YOLOv3 due to its reliability.

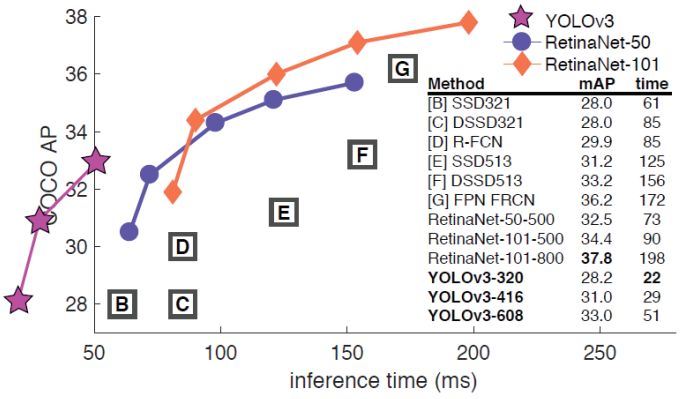


Figure 2. Runtime comparison of different CNN models for object detection. Figure taken from [18].

Object detection is that the latter finds the bounding box of an object within the image. In the case of YOLO, the bounding box annotations are defined in a text file, each line contains 5 numbers (without <>):

$$< label > \quad < c_x > \quad < c_y > \quad < w_x > \quad < w_y > \tag{3}$$

The *label* is an integer starting from 0. It is up to the user to keep track of what real label (e.g. dog, cat, hand, ...) the number corresponds to. Usually, all labels are defined in an extra text file, in the first line is the name of the 0*th* object, in the second line of the 1*st* object and so on. The floating-point numbers c_x and c_y are the bounding box centre coordinates, normalized by the image width and image height respectively. w_x and w_y are floating point numbers of the bounding box width and height, again normalized by the image width and image height respectively.

III. APPROACH

The workflow applied in this work is illustrated in Figure 3. As shown, the tester (domain expert) defines relevant domain features that would be serve as parameters for a test case generator (a random search algorithm). Next, the search algorithm creates a set of randomly generated test data that is fed into the simulator to create different scenarios for which the vision sensor my capture different image frames of the scene. When the simulation is run, the vision sensor captures images of within its point-of-view (POV) and saves them. From the images, a training dataset is created which is used to train a model for object detection. This model is then used to stop the simulation if an emergency is detected.

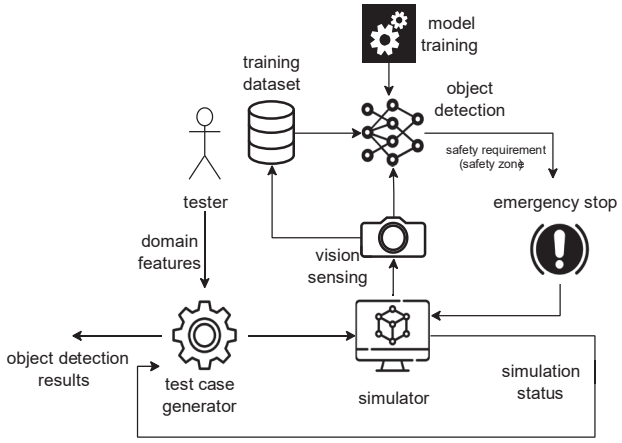


Figure 3. Illustration of the workflow used to implement the real-time object detection campaign in a simulator.

Overall, there are two major parts, the simulator (explained in Section III-B) where the simulation itself runs and the control package, from where the flow of the simulation is controlled. In the latter the object detection is implemented, see Section III-C for details.

A. Runtime System Specification

All the programs and libraries used for the implementation are available for Windows and Linux operating systems. Table I shows the system properties and libraries inculcated in the implementation used. NOTE: the libraries may depend on additional requirements.

Table I. System and Simulation Environment Properties

Feature	Description/Component
Operating system	Windows 10 Pro (64bit)
CPU	Intel Core i7-2600, 3.40GHz
GPU	NVIDIA GeForce GTX 970 (4GB)
Simulation software	CoppeliaSim Edu 4.4.0
Control environment	Python 3.9.0; NumPy1.24.2; OpenCV 4.7.0.72; PyTorch 2.0.1+cu118; ImageAI 3.0.3; PyZMQ 25.0.0; Ultralytics 8.0.111

B. Simulation

The simulation itself is implemented with the robot simulation software CoppeliaSim [20]. A snapshot of the working example of the simulated system is shown in Figure 4. The CAIS in the simulation consists of two protagonists, the human operator (called Bill) and a robotic arm. Both have the task to grab a green cube that lies on the conveyor belt. The robotic arm has a visual sensor at its tip and takes images with a resolution of 256 × 256 px.

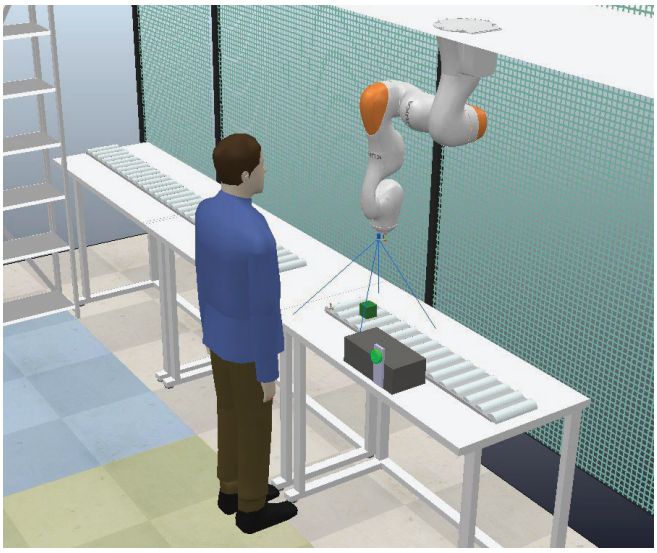


Figure 4. Overview of the simulated CAIS. Bill, the human operator stands next to a table with a conveyor belt. A green cube is placed on top of it. On the ceiling is a robotic arm mounted. At its tip, a vision sensor is placed. The viewing frustum of the vision sensor is outlined by the blue lines.

1) Simulation environment: The simulation environment typically has properties and parameters defined as part of a scene and runs in discrete time steps that may be altered e.g., dt = 50ms. By default, a scene must have a main script (programmed with Lua [21]) and contains the fundamental code that allows a simulation to run through a collection of four callback functions: *sysCall init*, *sysCall actuation*, *sysCall sensing* and *sysCall cleanup*. The main script can interact with a simulation "control" script also directly programmed within the simulation scene as a child script or as an external script. The two main actors in the simulation are Bill and the robotic arm.

Both actors aim to pick and move the cube away from the collaborative space, as they proceed on each timestep. In total, seven (7) free parameters (domain features) in the simulation are randomly assigned values from an automated control script:

1: Actuation delay for Bill
2: Hand movement speed for Bill
3: Actuation delay for the robotic arm
4: Movement speed of the robotic arm
5-7: RGB colour values of the ambient light

2) Control system: The simulation parameters and evaluation of a run are implemented in Python programs. To connect to CoppeliaSim the ZeroMQ[3] remote API is used.

[3] https://zeromq.org/

ZeroMQ is a high-performance asynchronous messaging library which is available for many different programming languages and operating systems. The simulation software opens a TCP port (default 23000) which can then be accessed to get the state of the simulation, set parameters and advance the time steps. Since the communication is TCP based, the control environment does not have to run on the same system as the simulation; an own dedicated machine can be used.

C. Implementation

Two versions of the YOLO object detection models[4] were deployed in our work. They were implemented as part of the control script in Section III-B2 and linked to the simulation from where continuous images are sent from a vision sensor connected to the tip of the robotic arm.

1) Dataset: The first step towards training both methods was to generate two datasets of images. One for training and one for testing. To generate the images, the simulation was run 500 times with random parameters and the images captured by the vision sensor were saved. Then the images were sorted manually into the ones with a hand present and those without a hand. From the total number of available images, 2960 images were selected for the training set and 500 for the test set each with and without a hand respectively.

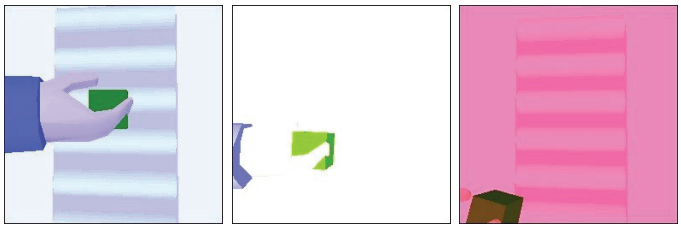


Figure 5. Three example images from the training for different light conditions and hand positions. The images have a resolution of 256 × 256px.

The neural network approach only needs the training samples with a hand, called positive samples. However, those images must be labelled according to the YOLO label format (see (3)). LabelImg [22] was used to label the images since it provides a convenient graphical user interface (GUI) to quickly label a set of images. A rectangular box can be drawn around the object one wants to label, and the label of the object can be selected - in our case, the object of focus is Bill's hand within an image frame.

The labels are saved into a text file with the same name as the image (except for the file extension). During the labelling process, a few misclassified images were noticed and removed from the data set. This resulted in the datasets listed in Table II. Examples of the images are shown in Figure 5. In total, the labelling took roughly 6 hours.

Table II. Number of images in the training dataset and the testing dataset.

	Images with hand	**Images without hand**
Training dataset	2960	2960
Testing dataset	500	500

2) Object detection using YOLOv3: For YOLOv3, the ImageAI [23] Python library was used and relies on PyTorch [24]. The library implements the YOLOv3 architecture and has pre-trained models available. There are different "sizes" of YOLO available with different numbers of weights. In this work, only the tiny-YOLOv3 version was used because GPU resource constraints. For the training process, the dataset is required to follow a strict directory structure where the generated images are named similarly as their annotations as in *imagexxx.jpg* should have a corresponding *imagexxx.txt*.

The names of the parent directory and the images do not matter. However, the annotations must have the same name as the images, except for the file extension. Note that the images for training and validation are both from the training dataset described in Section III-C1, 20% of the images are taken for the validation process and 80% for training. The tiny-YOLOv3 model was trained with an upper limit of 1000 epochs with a batch size of 16.

The YOLOv3 source code was modified to stop model training since there is no method for early stopping (and to prevent overfitting) of the training process in the ImageAI implementation of YOLOv3 if the model performance on the validation set no longer increases for 50 consecutive epochs. While training a neural network from scratch can take quite some time, transfer learning may be used leverage the knowledge of already trained networks by initializing the weights of a new network with the solution of a different network (that solves a similar problem) [25] - in our case, moving from real to virtual. A more recent discussion can be found in the work of Zhuang et al. [26]. To investigate the different behaviours, the networks were both trained from scratch and as well implemented in a transfer learning paradigm.

3) Object detection using YOLOv8: YOLOv8 is the most recent version of the YOLO model available in a Python library from Ultralytics [19]. Its implementation is similar to YOLOv3, as described in the previous section. Two small differences for the training process are, that the directory annotations must be renamed to labels and that the location of the training images must be specified in a separate YAML file [27]. Otherwise, the same training images, batch size and number of training epochs were used. YOLOv8 also comes in different model sizes: nano, small medium, large and extra-large. In this work, the nano and small versions were trained, both from scratch and with transfer learning.

[4] The object detection models have been packaged as a Python library at https://github.com/PatrickAschenbrenner/cais_rtod.

D. Object Detection Integration

The A simple example of how different detection methods is shown in Listing 1.

```
1 from cais_rtod.detector import YOLOv3, YOLOv8
2
3 img_file = "random/image.jpg"
4 # image can also be an in-memory opencv image
5 # img_file = cv2.imread("random/image.jpg")
6
7 # YOLOv3
8 yolov3_detector = YOLOv3()
9 prediction = yolov3_detector.predict(img_file)
10 if prediction == 1:
11         print("Hand detected with YOLOv3")
12
13 # YOLOv8
14 yolov8_detector = YOLOv8()
15 prediction = yolov8_detector.predict(img_file)
16 if prediction == 1:
17          print("Hand detected with YOLOv8")
```

Listing 1. Example of how the different detectors can be used in Python to predict if a hand is present in an image. Note that the detectors should only be called once (lines 8 and 14 in the code above) if multiple images are analysed. The library can also be used to draw bounding boxes around the detected hands, an example can be found in the repository containing the Python library.

IV. PRELIMINARY EVALUATION

In this section, we demonstrate how the different approaches performed with the testing dataset using common evaluation metrics [28], [29] according to the following image classification definitions:

- True positive (TP): Number of images with hand, correctly classified
- True negative (TN): Number of images with no hand, correctly classified
- False positive (FP): Number of images with no hand, incorrectly classified
- False negative (FN): Number of images with hand, incorrectly classified

The values resulting from determining how the different images are classified by the object detection models are then to calculate the metrics in (4).

$$\begin{aligned} \text{True negative rate (TNR):}\ & \frac{TN}{TN+FP} \\ \text{Precision:}\ & \frac{TP}{TN+FP} \\ \text{Recall (True positive rate (TPR)):}\ & \frac{TP}{TP+FN} \\ \text{Accuracy:}\ & \frac{TP+TN}{TP+FP+TN+FN} \end{aligned} \tag{4}$$

The above metrics are useful for classification tasks. However, in object detection, one wants to also know how well the predicted bounding box fits. To compare the true box with the predicted one, the intersection over union (IoU) is used. It is defined as the ratio of the intersection of the two bounding boxes over the area of their union.

A common metric to measure how well bounding boxes are predicted is the mean average precision (mAP, mAP@50 if IoU ≥ 0.5 is chosen as the detection threshold) over all classes. A detailed explanation of how it is calculated can be found in [30]. However, since our focus is on assessing safety risk, it is sufficient that the models identify an arm fully or in part within an image frame to trigger an emergency stop.

A. YOLOv3

The YOLOv3 network was trained from scratch and with transfer learning. For the former, the network was trained from scratch for 2000 epochs without early stopping. This number was chosen as it is significantly higher than the number chosen for early stopping and training could still be completed in a reasonable amount of time. However, training for more epochs decreased the accuracy of the network. The training time for early stopping was below 1h, while the full 2000 epochs took 30h. The evaluation of the full test dataset was about 22.3 ± 0.3s on the GPU and 129.1 ± 0.6s on the CPU. For the latter, the weights were initialized with those from a pre-trained model on the COCO dataset [31] provided by ImageAI[5].

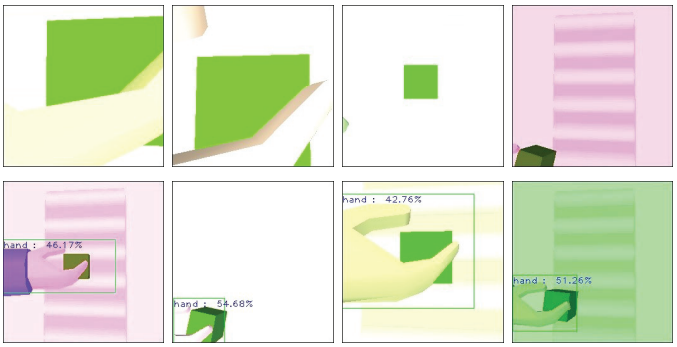


Figure 6. YOLOv3 example images. *Top:* Four false negatives from the test sample. *Bottom:* Selection of correctly identified images.

The results for the different models are listed in Table III. The minimum confidence level for detection is set to 40%. The best performance is obtained from the transfer learning model. From the test set only five images with a hand were misclassified (FN = 5) and none of the images without a hand (FP = 0). Examples of detected hands are shown in Figure 6.

B. YOLOv8

Similar to YOLOv3, the YOLOv8 networks were trained from scratch and with transfer learning, both with early stopping enabled. The nano and small implementations of the YOLOv8 architecture were used to achieve the best framerate. For transfer learning the pre-trained model provided by Ultralytics, which was originally trained on the COCO dataset, was used. The minimum confidence level for detection is set to 40%. Compared to YOLOv3 the training time per epoch is faster. The

[5] https://github.com/OlafenwaMoses/ImageAI/releases/download/3.0.0-pretrained/tiny-yolov3.pt

evaluation of the complete test dataset for nano-YOLOv8 (12.0 ± 0.1s) and small-YOLOv8 (13.0 ± 0.1s) models are approximately twice as fast as that of the tiny-YOLOv3 model as shown in Table III. Expectedly, the nano models have a higher frame rate than the small models. The small pre-trained model achieved the best accuracy, with $FN = 3$ and $FP = 7$. In Figure 7 some example images are shown.

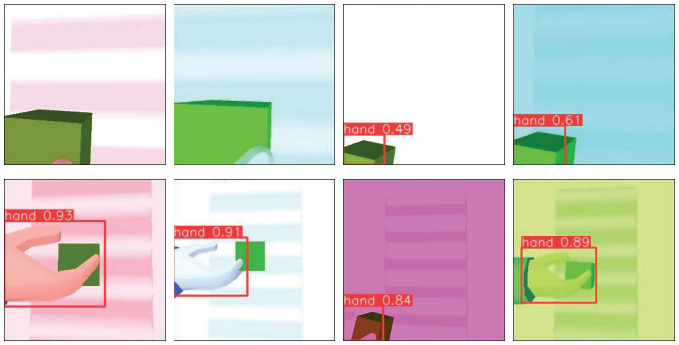


Figure 7. YOLOv8 example images. *Top:* Misclassified images. From left to right are two false negatives and then two false positives *Bottom:* Selection of correctly identified images.

C. *Implication of the Results*

In terms of the actual performances of the models, the results are consistent with findings from a similar work on real-world datasets which compares variants of YOLOv3 and YOLOv5 [32]. Although in our implementation there weren't significant differences in the accuracy, precision and recall of the models, the differences are more noticeable in terms of speed of detection whereby YOLOv8 variants have faster detection rates that YOLOv3.

The results in the previous sections showcase promising potential for proactive identification of potential safety hazards before deploying the CAIS in the real world allowing for adjustments to be made to the system or the environment to mitigate risks. For instance, if the performances of the models were to be low in general or lower than a certain set threshold, additional models can quickly be implemented, or the existing models modified and tested until a desired outcome is obtained.

Additionally, the simulation testing process can be extended for real world systems with minimal cost implications either financially or in terms of human effort needed. For instance, while in this work, we have relied on using transfer learning of object detection models trained on real-world data, for use in a virtual domain, the reverse is also possible i.e., training the models on synthetic data and then transferring them for use in real-world systems with little adjustments made to system.

Furthermore, it is important to bear in mind that simulated environment may not perfectly reflect the real world. Several environmental and system factors like lighting variations, sensor noise, and unexpected object appearances may be difficult to obtain in a simulator. Therefore, discrepancies may occur between simulation results and real-world performance, and these must be accounted for.

V. CONCLUSION & OUTLOOK

In this work, we argue for the need for safety risk analysis in critical ML-enabled systems using simulations. Accordingly, we present two different YOLO models for real-time object detection: YOLOv3 and YOLOv8 which rely on a deep CNN in a CAIS simulation. The models have been composed into an easy-to-use Python library and can be added to an existing simulation control environment without the need to rewrite the existing code. The simulation showcases a collaboration between a human operator and a robotic arm equipped with vision sensor detect a potential risk for the operator (Bill), completing a simple pick and move task.

Both models achieved similar accuracy (≥ 98%) and precision (≥ 96%). However, YOLOv8 achieves higher detection rate of about 83FPS (12.0ms inference time) nearly twice the evaluation speed of YOLOv3 with 44FPS (22.7ms inference time) although both required dedicated GPU for training and testing. For early stopping, both models were trained in approximately 1-2 h. Noticeably, longer training did not result in better performance.

For future use cases, the implementations must be tested for robustness. This work only considered 7 domain features and the models were trained based on the assigned of random values to the various features. Additional parameters and changes in the scenery may alter the look of the captured images to an extent where they differ too much from the training images.

One may also use the size information from the predicted bounding box to estimate the distance between the robotic arm and the hand of the operator so that a more fine-grained risk estimation can be applied in combination with applicable safety assessment standards. For example, a speed and separation monitoring (SSM) based risk management process may be used instead of the safety-rated monitored stop.

Future work would explore if a model trained on synthetic data only can be tested on real-world data by initially training with a large set of simulated data, and afterwards, only a small training sample from the real world is needed. However, additional parameters such as shadows and occlusion which are practical phenomena in the real world must be considered.

ACKNOWLEDGMENT

This study is partially supported by the Austrian Science Fund (FWF) for the project SafeSec, grant agreement No: I 4701 Internationale Projekt.

TABLE III. Summary of the different metrics for the YOLOv3 & YOLOv8 implementation. The models are trained with transfer learning and from scratch. The training stopped if the performance (mAP@50 of the validation dataset) did not increase for 50 epochs. For comparison, one YOLOv3 model was trained for 2000 epochs.

	tiny-YOLOv3 pretrained	**tiny-YOLOv3 pretrained**	**tiny-YOLOv3 scratch**	**nano-YOLOv8 scratch**	**small-YOLOv8 pretrained**	**nano-YOLOv8 scratch**	**small-YOLOv8 scratch**
Training time	54min	55min	30.6h	0.81h	1.2h	1.9h	1.7h
Epochs	59	55	2000	117	118	274	177
Model size	33.1MB	33.1MB	33.1MB	5.9MB	21.4MB	5.9MB	21.4MB
FPS (GPU)	44.8	44.8	44.8	83.1	76.7	83.1	76.7
FPS (CPU)	7.7	7.7	7.7	18.5	8.8	18.5	8.8
TNR	1	0.996	0.972	0.968	0.986	0.976	0.98
Precision	1	0.996	0.972	0.969	0.986	0.976	0.98
Recall	0.99	0.982	0.99	0.998	0.994	0.996	0.99
Accuracy	0.995	0.989	0.981	0.983	0.99	0.986	0.985

REFERENCES

[1] L. E. Lwakatare, A. Raj, J. Bosch, H. H. Olsson, and I. Crnkovic, "A taxonomy of software engineering challenges for machine learning systems: An empirical investigation," in Agile Processes in Software Engineering and Extreme Programming (P. Kruchten, S. Fraser, and F. Coallier, eds.), (Cham), pp. 227–243, Springer International Publishing, 2019.

[2] Jellyfish, "AI Engineer vs. Software Engineer," 2023.

[3] M. Camilli, M. Felderer, A. Giusti, D. T. Matt, A. Perini, B. Russo, and A. Susi, "Risk-Driven Compliance Assurance for Collaborative AI Systems: A Vision Paper," pp. 123–130, 2021.

[4] J. G. Adigun, M. Camilli, M. Felderer, A. Giusti, D. T. Matt, A. Perini, B. Russo, and A. Susi, "Collaborative artificial intelligence needs stronger assurances driven by risks," Computer, vol. 55, no. 3, pp. 52– 63, 2022.

[5] Universal Robots, "New Technical Specification on Collaborative Robot Design," 2018.

[6] E. Commission, D.-G. for Research, and Innovation, Industry 5.0: human-centric, sustainable and resilient. Publications Office, 2021.

[7] C. Birchler, S. Khatiri, B. Bosshard, A. Gambi, and S. Panichella, "Machine learning-based test selection for simulation-based testing of self-driving cars software," Empirical Software Engineering, vol. 28, p. 71, Apr. 2023.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 7, pp. 138–148, 2015.

[9] M. A. Santana, R. Calinescu, and C. Paterson, "Risk-aware realtime object detection," in 2022 18th European Dependable Computing Conference (EDCC), pp. 105–108, 2022.

[10] W. McCulloch and W. Pitts, "A logical calculus of ideas immanent in nervous activity," Bulletin of Mathematical Biophysics, vol. 5, pp. 127–147, 1943.

[11] P. Werbos, Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. PhD thesis, Harvard University, Cambridge, MA, USA, 1974.

[12] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, no. 6088, pp. 533– 536, 1986.

[13] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," Neural Computation, vol. 1, no. 4, pp. 541–551, 1989.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 6, pp. 84–90, 2012.

[15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," International Journal of Computer Vision (IJCV), vol. 115, no. 3, pp. 211–252, 2015.

[16] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," European Conference on Computer Vision, vol. 9905, pp. 21–37, 2016.

[17] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for´ dense object detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 2, pp. 318–327, 2018.

[18] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018.

[19] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," Jan 2023.

[20] Coppelia Robotics, "CoppeliaSim."

[21] R. Ierusalimschy, "Lua," 1993.

[22] Tzutalin, "LabelImg." Git code, 2015.

[23] Moses, "Imageai, an open source python library built to empower developers to build applications and systems with self-contained computer vision capabilities," mar 2018–.

[24] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, HighPerformance Deep Learning Library," in Advances in Neural Information Processing Systems 32 (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alche Buc, E. Fox, and R. Garnett, eds.), pp. 8024–8035, Curran´ Associates, Inc., 2019.

[25] S. Bozinovski and A. Fulgosi, "The influence of pattern similarity and transfer learning upon the training of a base perceptron B2.," Proceedings of Symposium Informatica, vol. 3-121-5, 1976.

[26] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A Comprehensive Survey on Transfer Learning," 2020.

[27] O. Ben-Kiki, C. Evans, and B. Ingerson, "YAML ain't markup language (YAML) (tm) version 1.2," 2009.

[28] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," Information Processing Management, vol. 45, no. 4, pp. 427–437, 2009.

[29] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A survey on performance metrics for object-detection algorithms," in 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 237–242, 2020.

[30] D. Shah, "Mean Average Precision (mAP) Explained: Everything You Need to Know," 2022.

[31] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollar, "Microsoft coco:´ Common objects in context," 2014.

[32] K. Liu, H. Tang, S. He, Q. Yu, Y. Xiong, and N. Wang, "Performance validation of yolo variants for object detection," in Proceedings of the 2021 International Conference on Bioinformatics and Intelligent Computing, BIC 2021, (New York, NY, USA), p. 239–243, Association for Computing Machinery, 2021.

Tapping in a Remote Vehicle's onboard LLM to Complement the Ego Vehicle's Field-of-View

Malsha Ashani Mahawatta Dona[1], Beatriz Cabrero-Daniel[1], Yinan Yu[2], Christian Berger[1]
[1]*University of Gothenburg* and [2]*Chalmers University of Technology*
Gothenburg, Sweden
{malsha.mahawatta,beatriz.cabrero-daniel,christian.berger}@gu.se, yinan@chalmers.se

***Abstract*—Today's advanced automotive systems are turning into intelligent Cyber-Physical Systems (CPS), bringing computational intelligence to their cyber-physical context. Such systems power advanced driver assistance systems (ADAS) that observe a vehicle's surroundings for their functionality. However, such ADAS have clear limitations in scenarios when the direct line-of-sight to surrounding objects is occluded, like in urban areas. Imagine now automated driving (AD) systems that ideally could benefit from other vehicles' field-of-view in such occluded situations to increase traffic safety if, for example, locations about pedestrians can be shared across vehicles. Current literature suggests vehicle-to-infrastructure (V2I) via roadside units (RSUs) or vehicle-to-vehicle (V2V) communication to address such issues that stream sensor or object data between vehicles. When considering the ongoing revolution in vehicle system architectures towards powerful, centralized processing units with hardware accelerators, foreseeing the onboard presence of large language models (LLMs) to improve the passengers' comfort when using voice assistants becomes a reality. We are suggesting and evaluating a concept to complement the ego vehicle's field-of-view (FOV) with another vehicle's FOV by tapping into their onboard LLM to let the machines have a dialogue about what the other vehicle "sees". Our results show that very recent versions of LLMs, such as GPT-4V and GPT-4o, understand a traffic situation to an impressive level of detail, and hence, they can be used even to spot traffic participants. However, better prompts are needed to improve the detection quality and future work is needed towards a standardised message interchange format between vehicles.**

***Index Terms*—Pedestrian Detection, Cyber-Physical Systems (CPS), Cooperative Intelligent Transportation Systems (C-ITS), Large Language Models, Generative AI, Vehicle-to-Vehicle, V2V**

I. Introduction

In recent years, advanced technologies have been integrated into vehicles turning them into intelligent Cyber-Physical Systems (CPS) to improve comfort and safety. The development of such intelligent systems aims to enhance safety and efficiency while providing an overall improved driving experience. CPS represents a paradigm where computational platforms are integrated with control algorithms, which are capable of handling heterogeneous distributed systems [1] as we face them in automotive systems. CPS uses computational resources to interact with physical processes; modern vehicles, for example, perceive their surroundings with cameras, radars and ultrasound devices and even road-side units (RSUs) to act safely in their context.

Current technological advancements potentially influencing automotive systems and CPS include Large Language Models (LLMs) that leverage natural language processing (NLP) techniques to understand and respond even to complex multi-modal data inputs. LLMs such as Generative Pre-trained Transformers (GPT) excel in various domains due to their exceptional language understanding and generation capabilities [2]. Hence, they have shown potential applicability in various domains such as health care, education, research, etc. [3] to provide better experiences and additional services to the users efficiently.

A. Problem Domain and Motivation

For today's ADAS to interact with other traffic participants, such as other vehicles or pedestrians within the defined operational design domains (ODDs), they typically need to be within a sensor's field of view (FOV). When relaxing the constraint of requiring other traffic participants to be in a vehicle's field-of-view (FOV) to explore opportunities and challenges beyond today's generation of ADAS or even towards automated driving (AD), technical concepts typically suggest either the use of V2I via RSUs or to share sensor data or object information between vehicles by streaming data from one vehicle to another one (V2V) to complement a vehicle's own field-of-view [4]. However, RSUs do not see a broad availability to support such scenarios, and streaming other vehicles' data is consuming a substantial amount of cellular network bandwidth while only a fraction of the information in the streamed data is of relevance for an ADAS or AD system.

With the recent advancements in the automotive context, as, for example, seen in the SOAFEE framework [5], vehicle system architectures are trending towards powerful centralized processing units that include CPUs and GPUs. Anticipating the presence of an LLM in such a vehicle could be used to improve, for instance, the passengers' in-car experience with better voice assistants processing even complex natural language-based dialogues between the human and the vehicle. In fact, some global automotive manufacturers are already working on deploying state-of-the-art foundational models within the vehicles to assist the in-car experience for passengers [6], [7].

We suggest a concept to exploit the anticipated availability of such technologically advanced smart vehicles to circumvent the aforementioned relaxed constraints to enrich the own

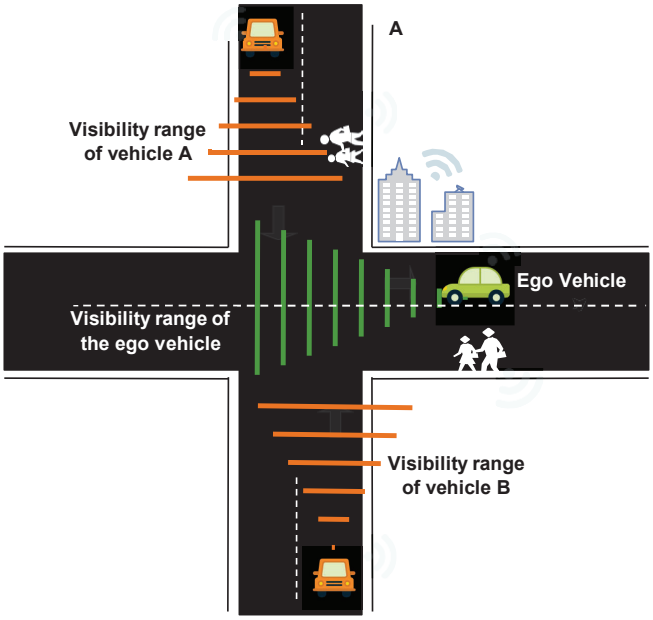


Fig. 1: Graphical representation of a looking around the corner problem: The ego vehicle is approaching an intersection together with vehicles A and B. Each vehicle has its own FOV represented by their respective colours, and that information could help each other understand the complete traffic situation at hand.

vehicle's FOV for practical traffic situations such as looking around the corner at intersections [8]: As shown in Fig. 1, the ego vehicle is approaching an intersection where other vehicles A and B are in proximity as well. All vehicles have not entered their respective line-of-sight and hence, information about their own surroundings could help the other vehicles to better understand the traffic situation. Instead of letting vehicles A and B stream their data to the ego vehicle, or being dependent on an RSU for distributing sensor data between vehicles, we envision that the ego vehicle is having a *dialogue* with other vehicles via their respective LLMs to enrich its own understanding of the surrounding traffic.

Compared to streaming images over a VANET with 1Mbps network link including 10% overhead, a 218.6KB image would require 1.98 seconds to complete the transmission. In contrast, the same amount of data corresponds to a dialog fitting on approx. 124 pages of text. Hence, we can assume that even a detailed back-and-forth dialogue would require less data compared to transmitting images. Therefore, we envision that our proposed approach will bring tremendous advantages over the current technical concepts suggested in the scientific literature: (A) the amount of data to be shared can be reduced substantially as only small texts are exchanged as we will illustrate and evaluate in our study; (B) no intellectual property (IP)-sensitive information about hardware capabilities of the data-supplying vehicles need to be exchanged as the LLM is acting as a communication interface within the CPS abstracting such details; and (C) no IP-sensitive information about the software or machine learning (ML) models such as ML-weights need to be exchanged.

B. Research Goal and Research Questions

Our main research goal is to evaluate the effectiveness of using an LLM to serve as a communication interface between vehicles to detect pedestrians that are not in direct line of sight. We address the following research questions:

RQ-1: To what extent can a state-of-the-art LLM be used as a communication interface to qualitatively identify pedestrians?

RQ-2: To what extent can a state-of-the-art LLM be used to reliably detect the location of pedestrians?

C. Contributions and Scope

We explore the possibility of applying an LLM for pedestrian detection by conducting a set of experiments. Our main contribution is a qualitative assessment of a state-of-the-art LLM to detect pedestrians while using a minimalistic dialogue for a curated dataset containing single pedestrians on a crosswalk. While our experiments unveil impressive fine-granular details, the LLM's quality to locate a pedestrian as well as the execution times need to be improved further.

D. Structure of the Paper

The remainder of the paper is organised as follows: In Section II, we review the related work. Section III provides a detailed description about the adopted methodology that addresses the research goal and the research questions for our study. In Section IV, we present our results in detail, and Section V provides the analysis and discussion of the findings. We conclude the paper in SectionVI.

II. Related Work

When we look at the feasibility of deploying LLMs within a CPS for better utilization of resources in handling complex tasks, studies such as Xu et al. [9] and Yang et al. [10] show the potential of using LLMs to handle various tasks in dynamic environments. Xu et al. coined a concept called "Penetrative AI", introducing LLMs to interact with and reason about the physical environment through various types of sensors [9]. The study's findings showcase that LLMS such as ChatGPT are proficient in interpreting IoT sensor data and reasoning about them according to the tasks in the physical world. Furthermore, the study of Yang et al. also demonstrate that LLMs can be employed in physical environments as a dynamic solution, illustrating the potential of integrating LLMs into CPS to enhance their productivity [10]. These studies motivate delving deeper into the concept of integrating LLMs with CPS.

The problem of enriching a vehicle's FOV as in scenarios like looking around the corner [8] has been addressed for a while to find an effective and feasible solution that can be applied in Cooperative Intelligent Transportation Systems (C-ITS). Isele et al. [11] explored the effectiveness of Deep Reinforcement Learning (RL)-based approaches for intersection

handling. They report that Deep Q-Network-based approaches show better task efficiency and success rates compared to traditional rule-based methods. The authors collect a point cloud through a combination of six Light Detection and Ranging (LiDAR) sensors from an autonomous vehicle to simulate a real-world unsignaled T-junction scenario in SUMO to test the Deep-Q-Network-based approaches and traditional Time-To-Collision (TTC) algorithms. However, in the event of occlusions within the intersection, the deep Q-networks resulted in collisions.

Kilani et at. [12] propose a framework to improve the driver's FOV to detect road obstacles using LiDAR point cloud data. The proposed method can be used to identify obstacles such as buildings, vegetation, or any other roadside infrastructure that occludes the driver's FOV. The latter part of the proposed method investigates how visibility assessment data can be used in achieving intersection safety by mitigating collisions. The study shows that the higher blockage rates increase the collision proportions indicating a higher risk of collisions in intersections.

Zhou et al. [13] present a framework called EAVVE as a vehicle-to-everything system that utilises edge servers to provide vehicular vision to mitigate collisions. The authors have evaluated the framework against real-world road testing in different traffic densities using different infrastructures. The prototype of EAVVE works on positional data and images captured by sender vehicles in a given geographical area. The image data will be processed through object detectors that are located in either the sender vehicle itself or at the edge of the network, like in RSUs. The detection results and the direct positional data will be used to provide real-time emergency detection and notifications to the client vehicles. EAVVE has shown significant performances in both real-world testing scenarios as well as in simulation-based environments.

The previously mentioned studies have contributed to addressing the challenge of occluded traffic situations. However, respective limitations and gaps still remain to cover various traffic scenarios in the context of heterogeneous system architectures for vehicles.

III. Methodology

Our methodology to address the overall research goal is comprised of several parts that we describe in detail in the following. We included a preliminary exploratory study to determine the general feasibility of our idea to identify (RQ-1) and to locate (RQ-2) pedestrians with today's state-of-the-art LLMs, as technical advancements happen regularly and rapidly in this area. Afterward, we decided on a dataset that we used for the subsequent systematic experiments. These experiments entailed to determine to what degree LLMs can identify pedestrians at all and sub-experiments to determine the quality of an LLM's response to locate a pedestrian reliably.

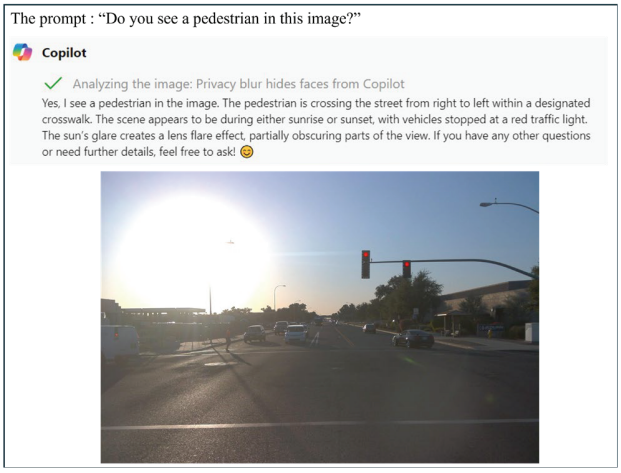


Fig. 2: Detailed answers obtained from Microsoft Copilot. The question "Do you see a pedestrian in this image?" was used as the prompt. The input image is taken from the Waymo dataset [16] and represents a sunset.

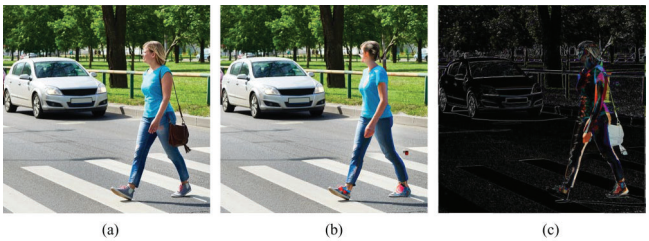


Fig. 3: Exploring an LLM's capabilities not only to describe but also to locate a pedestrian: (a) shows the input image (taken from [17]), (b) the DALL.E 2 generated image (b), and differences between both.

A. Preliminary Exploratory Studies

We conducted a series of preliminary feasibility studies with Microsoft Copilot, DALL.E 2 and GPT-4. Our initial studies unveiled that using generative AI models for object detection is working impressively well already in a zero-shot setting [14]. Fig. 2 shows a detailed description generated by Microsoft Copilot as an answer for the prompt "Do you see a pedestrian in this image?" However, localising the pedestrian posed a challenge for Microsoft Copilot, resulting in unwanted hallucinations [15] occurring unpredictably from time to time. We investigated as part of our pre-studies DALL.E 2 with the goal of to what extent DALL.E 2 is capable of not only detecting but also locating a pedestrian by generating a properly annotated image as a response. The difference between the original image supplied to the LLM and its generated response was calculated and analysed to determine to what degree the LLM could locate a pedestrian. We observed that the image edit feature of DALL.E 2 works best when supplying an image mask to guide the LLM where to search for the pedestrian as shown in Fig. 3. While experimenting with different image

masks seems to enhance an LLM's response, this approach was not further pursued as it is contrary to our research goal to let the LLM locate a pedestrian on its own where the requesting vehicle has no information at all where the pedestrians and other vehicles would be. We completed our preliminary exploratory studies with the conclusion that only GPT-4 with vision (GPT-4V) introduced in September 2023 [18] and GPT-4o ("o" for "omni") introduced in May 2024 [19] are able to detect objects with promising results.

B. Datasets

For our main experiments, we used the Waymo open dataset [16] and, in particular, the labelled objects on pedestrians. The Waymo dataset has been collected in urban and suburban areas in the USA in 2021, covering over 100,000 different day and night scenarios. The dataset was extracted with a sample rate of 10 Hz, focusing on the front camera images in the training set. The extraction contained 15,947 images, out of which the first 5000 images were considered to curate our dataset.

The curated dataset contains 126 images with pedestrian labels to control potentially influencing factors for our experiment. We selected images having only one single pedestrian on a crosswalk in the centre part of the image. Another subset of 138 images was selected where pedestrians were not visible on crosswalks as a control group. These two sets of images were used throughout all our main experiments as described below. We used the VGG Image Annotator [20] to retrieve the bounding box coordinates of pedestrians that we later used as the ground truth for evaluation.

C. Experiment Design

After assessing the preliminary results, we decided to choose the recent release of GPT-4 with vision, also known as GPT-4V or gpt-4-vision-preview, and GPT-4o for all three experiments as these two models could analyse and reason about images in a promising manner. The obtained results will allow a comparison between two state-of-the-art LLMs.

1) Experiment 1: Binary Pedestrian Detection:

We designed this experiment to address RQ-1 by assessing to what extent the LLM can detect pedestrians. We used our curated dataset with the GPT-4V and GPT-4o models and the following prompt:

> **Is there a human pedestrian in this image? Answer only either "yes" or "no".**

2) Experiment 2: Bounding Box Generation:

We designed this experiment to address RQ-2 assessing to what extent GPT-4V and GPT-4o can locate a pedestrian using a prompt that requests the LLM to return the coordinates of the area occupied by the pedestrian. Details about the reference coordinate system to be used were fed to the models along with the prompt to standardise the coordinate system of pedestrian detection. All images in the curated dataset were checked against the prompt. These coordinates generated by GPT models were compared together with the ground truth coordinates retrieved from the Waymo Dataset.

> **Given the reference system where (0,0) is the top-left corner and (1,1) is the bottom-right corner of the image, provide the coordinates (X,Y), (X',Y') representing the precise location of a person in the image. Ensure the coordinates accurately delineate the complete area occupied by the person. Return coordinates ONLY using this template: (X,Y), (X',Y')**

3) Experiment 3: Comparative Evaluation of Prompts:

We expanded the previous experiment by systematically checking the curated dataset three times against the three prompts listed in Tab. I to conduct the comparative evaluation across prompts. The multiple runs across the same prompt facilitated analysing the consistency of the responses to identify potential hallucinations as suggested in literature [21]. The initial prompt used in experiment 2 was refined repeatedly based on the feedback of a generative AI model. The generated bounding box details, respective ground truth coordinates, and processing times were recorded for each prompt. Furthermore, the results recorded under both models were compared against each other for performance evaluation.

IV. Results

We report the results from the three experiments in the following. For replication purposes, we share the necessary code and raw results as Supplementary Material in a GitHub repository [22].

A. Experiment 1: Binary Pedestrian Detection

In this experiment, GPT-4V and GPT-4o both were prompted using the images from our curated subset. The two models were then asked to determine whether a pedestrian was present in each of the individual images. The generations were then contrasted to the manual labels for each of the images, and the instances where the models correctly predicted the labels were counted.

As shown in Tab. II, both models GPT-4V and GPT-4o can correctly identify the pedestrians in all images (high recall) at the expense of wrongly labelling images with no pedestrians (false positives). The trade-off between recall and other metrics, as reported in Tab. II, aligns with the goal of enriching a vehicle's FOV to reliably detect pedestrians when they are indeed present.

The same confusion matrix was obtained when re-running the experiment. However, it is important to note that even though none of the values for the metrics in Tab. II changed, the false positives were occurring in different images in the different runs.

TABLE I: List of prompts that were used in the Experiment 3

P1	Given the reference system where (0,0) is the top-left corner and (1,1) is the bottom-right corner of the image, provide the coordinates (X,Y), (X',Y') representing the precise location of a person in the image. Ensure the coordinates accurately delineate the complete area occupied by the person. Return coordinates ONLY using this template: (X,Y), (X',Y')
P2	In the image, identify the person's location using a coordinate system where (0,0) is the top-left corner and (1,1) is the bottom-right corner. Provide the coordinates (X,Y), (X',Y') that encapsulate the entire area occupied by the person. Use this format only: (X,Y), (X',Y')
P3	Using (0,0) is the top-left corner and (1,1) is the bottom-right corner, provide the coordinates (X,Y), (X',Y') of the location of a person in the image. The coordinates should contain the complete area occupied by the person. Return coordinates ONLY using the template: (X,Y), (X',Y')

The Tab. III contains a comparative evaluation between the two models GPT-4V and GPT-4o to report about recall, specificity, and precision.

Even though the OpenAI API documentation [23] states that GPT-4V is not ready to accurately answer questions about the positions of objects, our preliminary tests showed that it can understand whether a pedestrian is present, including detailed natural language explanations of where the pedestrian is and where they are headed to. This natural language explanation capability was also shown by GPT-4o, the most recent release of GPT-4 model. For instance, about the image used in Fig. 6, GPT-4V stated that "The individual is located in the centre of the image, standing in the middle of a crosswalk on the road [. . .] The position of the pedestrian is directly between the two lanes of the road" whereas GPT-4o stated that "The pedestrian is in the middle of a crosswalk, facing to the right, and holding a "STOP" sign." Similarly, detailed descriptions of the position and predictions of the intentions of the pedestrians were obtained from other images support the use of LLMs to obtain precise information about traffic.

B. Experiment 2: Bounding Box Generation

The second experiment further explored the ability of the models to locate the identified pedestrians, addressing **RQ-2**, by asking the LLM to provide coordinates using a specific reference system. To understand the overall performance, the following measures based on the differences between the generated and the ground truth bounding boxes were used:

- Number of unions: Percentage of images, in which the generated bounding box overlaps with the manual bounding box (as seen in Fig. 5).
- Recall: Percentage of the manual bounding box that overlaps with the generated one (Fig. 5 shows an example with recall close to 100% while the images that do not share any overlapping between the manual bounding box and generated one reported 0% recall).
- Intersection over union: Relation between the overlapping between the generated and manual bounding boxes and the intersection between the two. This metric penalises big bounding boxes that would achieve 100% recall.

We computed these metrics for all images across three runs, and we obtained that the generated bounding boxes overlapped with the manually annotated ones in only 30.42% of the tests with GPT-4V, whereas it was recorded as 42.18% for GPT-4o. In the cases where the manual and generated bounding boxes did overlap, the recall was recorded as 37.38% and 49.76% for both GPT4-V and GPT-4o. This means that only around a third of the area occupied by a pedestrian is covered, on average, in only a third of the images containing a pedestrian. Moreover, the standard deviation across images and runs was 28.81% and 35.77% for both models, respectively, which is substantial. This means that there may be some images where the overlap is bigger, as in Fig. 5, and some images where there is very little to no overlap. To explore this variance, we analysed the performance of GPT-4V across images. The distribution of the recall for all images in the dataset containing a pedestrian can be seen in the left-most box in Fig. 7. Fig. 4 represents the percentages of Intersection over Union (IoU) of the bounding boxes generated by GPT-4V considering the images where an overlapping was reported. However, the average IoU in all images is smaller. This is due to taking into account the area of the generated bounding box and penalises too large areas (covering the whole picture would lead to a recall of 100%, but it would not be informative).

The individual test results, as seen in Fig. 6 show that the performances of both models are not consistent across runs, which makes the LLMs unreliable (low recall) out of the box, as we further discuss in Sec. V. The first row in Fig. 6 depicts the behaviour of both models for the prompt considered in Experiment 2. Further analysis of the results also shows that some of the images are indeed challenging for the LLM. We conducted subsequent data analysis to figure out the reasons behind the incorrect responses produced by both the LLMs. The three main categories of error messages were identified as (A) No pedestrians detected, (B) Partial coordinates and (C) Ambiguous descriptions.

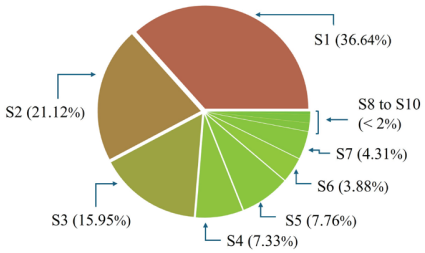
The majority of the error messages caused by GPT-4V fell under the partial coordinates category, where the constraint of the maximum token list may have hindered the generation of the response. However, had the model been behaving according to the prompt and only providing the coordinates, the selected maximum token count would have been sufficient. GPT-4o provided more "no pedestrian detected" error messages compared to GPT-4V. However, we carefully checked all images where the LLMs were unable to locate the pedestrians and observed that 52.94% (more than half of the sample) are captured during dusk or sunset or within a shady environment or else when there is a large solar glare captured in the image.

C. Experiment 3: Comparative Evaluation of Prompts

In our final experiment, we studied whether changing the prompt could increase the performance for the two tasks

TABLE II: Confusion matrix and relevant performance metrics for GPT-4V and GPT-4o in pedestrian detection over the dataset.

	Ground Truth: True	**Ground Truth: False**	
GPT-4V Predicted True	120 (TP)	5	$Recall_{4V}$ = 95.24%
GPT-4V Predicted False	6	129 (TN)	$Specificity_{4V}$ = 96.27%
GPT-4o Predicted True	125 (TP)	11	$Recall_{4o}$ = 99.21%
GPT-4o Predicted False	1	127 (TN)	$Specificity_{4o}$ = 92.03%



Slice	IoU percentage upper limit
S1	6.67%
S2	13.33%
S3	20.00%
S4	26.67%
S5	33.33%
S6	40.00%
S7	46.67%
S8-S10	>46.67 %

Fig. 4: Intersection-over-Union (IoU) percentages for the images that share an overlapping between the GPT-4V generated and the ground truth bounding boxes, zooming in the overlapping IoUs. The pie chart contains 15 slices.

Statistic	**GPT-4V**	**GPT-4o**
Recall	95.24%	**99.21%**
Specificity	**96.27%**	92.03%
Precision	**96.00%**	91.91%
Negative Predictive Value	95.56%	**99.22%**
False Positive Rate	3.73%	**7.97%**
False Discovery Rate	4.00%	**8.09%**
False Negative Rate	**4.76%**	0.79%
Accuracy	**95.77%**	95.45%
F1 Score	**95.62%**	95.42%
Matthews Correlation Coefficient	**91.53%**	91.18%

TABLE III: Comparative evaluation of GPT-4V and GPT-4o

Fig. 5: Images from Waymo Dataset [24] with a GPT-4V and GPT-4o generated bounding box (red) almost completely covering the ground truth area (green). The selected images show day and night scenarios. The results (a) and (b) are retrieved from the GPT-4V model, and the (c) and (d) are retrieved from the GPT-4o model.

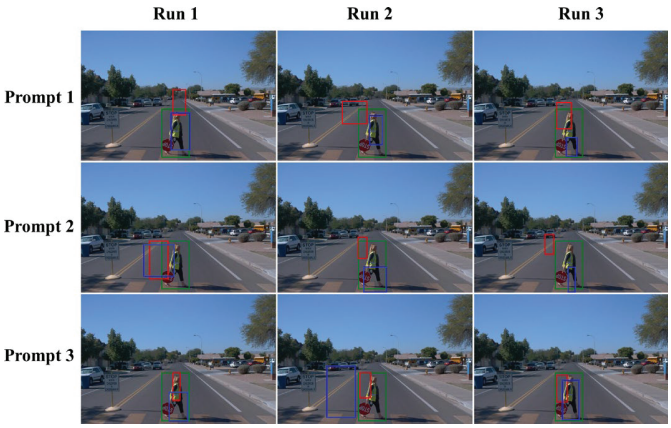


Fig. 6: The behaviour of models GPT-4V (blue), GPT-4o (red) against the ground truth (green) in multiple runs across each prompt

above. We refined the prompt based on the metrics reported in Tab. II and tested it with GPT-4V. The new prompt emphasised the need to avoid false positives by adding, "It is very important for you to say 'no' if there is no pedestrian close by," following the recommendations by Cheng et al. [25]. Experiment 1 was re-run and the recall dropped to 98.18% since one of the images containing a pedestrian was mislabelled as false negative. This is an undesired result in the intended application context (i.e., missing out on a real pedestrian where we must not). However, the accuracy raised to 88.41%, given that fewer images were misclassified in total.

To determine whether a different choice of prompt would also improve the generation of coordinates, we refined also the original prompt (marked as P1 in Tab. I) used for Experiment 2. The goal was to increase the average recall across images

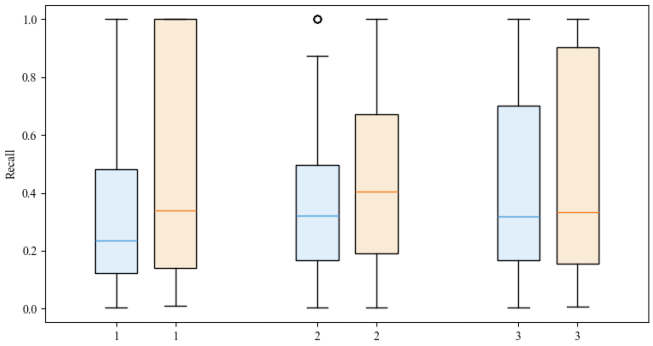


Fig. 7: Distribution of recall values among images (for three runs) tested with GPT-4V (blue) and GPT-4o (blue), for the three alternative prompts in Tab. I.

and GPT-4V was asked to generate alternative prompts to this end. Two alternative prompts as listed in Tab. I were also tested three times for each image in the selected dataset.

Comparable to prompt P1, the bounding boxes generated by prompt P3 overlapped with the manual ones in 33.82% of the tests in GPT-4V, whereas it was 43.9% for GPT-4o. However, the recall was 47.04% ± 34.32% and 47.6% ± 36.21% on average for GPT-4V and GPT-4o, respectively. These results are a bit higher than that of prompt P1 as seen in Fig. 7. Prompt P2 also achieved higher average recall values (47.65% ± 34.64% for GPT-4V and 45.01% ± 32.35% for GPT-4o) across all tests. Even though there were slight differences in the number of unions going as high as 47.76% and 45.01% for P2, the difference is not statistically significant as seen in Fig. 7, and the high standard deviation highlighted the different performance across images.

V. Analysis and Discussion

The first question in this study sought to determine how well an LLM can act as a communication interface to detect pedestrians for ADAS. To answer that question, further information about the robustness of LLMs in detecting and locating pedestrians was needed. Therefore, the experiments in this study set out with the aim of assessing whether GPT can detect a pedestrian in an image, as described in Sec. IV-A, and describe the area of the image where they are. Further tests as described in Sec. IV-B attempted to force the GPT model to use a standard format (e.g., 2D coordinates) to describe the area of an image containing a pedestrian.

The results of the first experiment determined that GPT-4V is indeed capable of labelling an image whether it contains a pedestrian or not, reaching a recall of 95.24% and an accuracy of 95.77%. GPT-4o showcased slightly better results with a near-perfect recall of 99.21% and an accuracy percentage of 95.45%. These results show how well the state-of-the-art LLMs perform complex tasks such as pedestrian detection. Within this study, we consider the recall value to be the most crucial metric that indicates the sensitivity of the LLMs. In pedestrian detection, higher recall values indicate fewer false negatives (missed pedestrians), which ultimately helps the system to increase safety by minimizing accidents.

Moreover, the accuracy can be further improved by changing the prompt at the expense of a different balance between precision and recall. This might be interesting for scenarios other than the looking-around-the-corner problem, where recall is of the utmost importance.

Another finding that stands out from our results is that while GPT-4V and GPT-4o can provide natural language descriptions about the position and heading of the pedestrian in the images at impressive details, it did not perform well in providing 2D coordinates to describe the bounding boxes around them. In two out of three cases, bounding boxes generated by GPT-4V did not overlap with the area occupied by the pedestrians at all, as reported in Sec. IV-B. GPT-4o results were slightly better than GPT-4V where, in an average of 45% of the tests, the generated bounding boxes overlapped with the area occupied by the pedestrians according to the ground truth data.

Moreover, our results show that (i) the variability of the recall across images is large, and that (ii) the accuracy is not consistent across different runs using the same image either. This undesired behaviour makes GPT-4V and GPT-4o not ready yet to determine the location of pedestrians; however, as reported in the OpenAI API documentation [23], future updates of the model might address this limitation.

Following this experiment, as presented in Sec. IV-C, the prompt was updated and the performance of GPT-4V and GPT-4o was evaluated. The behaviour of each model was arbitrary in each prompt under different runs, and the performances were better on average. However, the null hypothesis could not be rejected, and studying alternative formulations for the prompt is out of the scope of this study.

We discuss threats to validity based on Feldt and Magazinius [26]. During the study, we used the LLMs Copilot, DALL.E 2, GPT-4V and GPT-4o, in which we have no control over model-level updates. Had there been any model updates during the experiment period, the results and performance evaluations would have been changed. We consider this as the main internal threat of the study. The selection of the dataset, curating a subset of images for the experiments, and formulating the curation criteria could be considered subjective to the authors' knowledge and might cause a potential bias. Although a preliminary study was conducted with a sample of grey-scale images, it is not evident whether there is a significant difference in accuracy between colour and grey-scale images. Therefore, further studies are required, and results may have an impact based on the image type. In addition to that, technological advancements in the field of automotive, including different sensor technologies and hardware equipment, might limit the generalisability of the results. The ethical concerns, rules, and regulations related to LLMs and Artificial Intelligence may also have an impact on the application level, limiting the generalisability. Therefore we consider them as the external threats that may cause an impact on the study.

VI. Conclusion and Future Work

We have suggested and systematically evaluated a novel approach to enrich a vehicle's understanding of its surroundings to overcome limitations when being dependent on V2I with RSUs or streaming a lot of data in V2V setups. Our proposed concept addresses situations where the ego vehicle's FOV is partially occluded, such as in looking-around-the-corner situations.

Our approach adopts LLM as a communication interface for connected, cooperative (automated) mobility (C-CAM) where the vehicles can have a "dialogue" about a traffic situation to abstract from the underlying sensor hardware and IP-sensitive software implementations. Furthermore, practical issues from interoperability would be substantially reduced with an LLM-based abstraction. Our experiments show that LLMs exhibit impressive capabilities to understand fine-granular details of a situation reliably. However, the current generation of LLMs has insufficient performance in locating objects reliably. Nevertheless, we consider the balance of precision and recall as reported in Tab. II acceptable in the context of our research goal for this study to reliably detect pedestrians when there are indeed pedestrians present. Other problems, however, might rely on different metrics (e.g., specificity, accuracy, or F1-score) to determine whether GPT-4V and GPT-4o are performing superiorly compared to dedicated ML models.

While LLM-based object detection is in its infancy and has not yet been optimised to answer with exact locations of objects [23], our results show the potential to expect for image detection and localisation. However, further improvements and research work need to be conducted on various complex scenarios that are not addressed yet by this single study, for instance, scenarios where multiple vehicles are prompting, complex road scenarios where multiple pedestrians and cyclists are involved, etc. Summing up to the questions that remain such as to what extent the LLM's performance can scale with the input data and the complexity of the environment. Furthermore, exploring the performance of locally running LLMs needs is left open as well.

In addition to that, the authors are already working on evaluating the trustworthiness of the proposed pedestrian detection mechanism as a further study focusing on hallucination detection and mitigation.

Acknowledgments

This work has been supported by the Swedish Foundation for Strategic Research (SSF), Grant Number FUS21-0004 SAICOM and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

[1] S. Chakraborty, M. A. Al Faruque, W. Chang, D. Goswami, M. Wolf, and Q. Zhu, "Automotive cyber–physical systems: A tutorial introduction," *IEEE Design & Test*, vol. 33, no. 4, pp. 92–108, 2016.

[2] T. Brown *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper _files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[3] M. Sallam, "Chatgpt utility in healthcare education, research, and practice: Systematic review on the promising perspectives and valid concerns," *Healthcare*, vol. 11, no. 6, 2023. [Online]. Available: https://www.mdpi.com/2227-9032/11/6/887

[4] M. N. A. Kun-Feng Wu and W.-J. Ye, "An evaluation scheme for assessing the effectiveness of intersection movement assist (ima) on improving traffic safety," *Traffic Injury Prevention*, vol. 19, no. 2, pp. 179–183, 2018, pMID: 28812374. [Online]. Available: https://doi.org/10.1080/15389588.2017.1363891

[5] *SOAFEE - Scalable Open Architecture for Embedded Edge*, 2024, accessed: 2024-01-31. [Online]. Available: https://www.soafee.io/

[6] M. R. A. H. Rony, C. Suess, S. R. Bhat, V. Sudhi, J. Schneider, M. Vogel, R. Teucher, K. E. Friedl, and S. Sahoo, "Carexpert: Leveraging large language models for in-car conversational question answering," 2023.

[7] "Bmw intelligent personal assistant powered by the alexa large language model (llm)," 2024, accessed: 2024-02-26. [Online]. Available: https://tinyurl.com/BMWweb

[8] A. Furda and L. Vlacic, "An object-oriented design of a world model for autonomous city vehicles," in *2010 IEEE Intelligent Vehicles Symposium*, 2010, pp. 1054–1059. [Online]. Available: https://doi.org/10.1109/IVS.2010.5548138

[9] H. Xu, L. Han, Q. Yang, M. Li, and M. Srivastava, "Penetrative ai: Making llms comprehend the physical world," in *Proceedings of the 25th International Workshop on Mobile Computing Systems and Applications*, 2024, pp. 1–7.

[10] H. Yang, M. Siew, and C. Joe-Wong, "An llm-based digital twin for optimizing human-in-the loop systems," 2024.

[11] D. Isele, R. Rahimi, A. Cosgun, K. Subramanian, and K. Fujimura, "Navigating occluded intersections with autonomous vehicles using deep reinforcement learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 2034–2039.

[12] O. Kilani, M. Gouda, J. Weiß, and K. El-Basyouny, "Safety assessment of urban intersection sight distance using mobile lidar data," *Sustainability*, vol. 13, no. 16, 2021. [Online]. Available: https://www.mdpi.com/2071-1050/13/16/9259

[13] P. Zhou, T. Braud, A. Zavodovski, Z. Liu, X. Chen, P. Hui, and J. Kangasharju, "Edge-facilitated augmented vision in vehicle-to-everything networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 10, pp. 12 187–12 201, 2020.

[14] V. S. Dorbala, J. F. M. Jr, and D. Manocha, "Can an embodied agent find your "cat-shaped mug"? llm-based zero-shot object navigation," *IEEE Robotics and Automation Letters*, pp. 1–8, 2023.

[15] Z. Ji, T. Yu, Y. Xu, N. Lee, E. Ishii, and P. Fung, "Towards mitigating LLM hallucination via self reflection," in *Findings of the Association for Computational Linguistics: EMNLP 2023*, H. Bouamor, J. Pino, and K. Bali, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 1827–1843. [Online]. Available: https://aclanthology.org/2023.findings-emnlp.123

[16] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[17] *Woman crossing the road - Image*, 2015, accessed: 2024-01-28. [Online]. Available: https://www.istockphoto.com/se/foto/woman-crossing-street-at-pedestrian-crossing-gm478525366-67201453

[18] OpenAI, "Gpt-4v(ision) system card," 2023. [Online]. Available: https://cdn.openai.com/papers/GPTV_System_Card.pdf

[19] *Hello GPT-4o*, 2024, accessed: 2024-05-15. [Online]. Available: https://openai.com/index/hello-gpt-4o/

[20] A. Dutta and A. Zisserman, "The VIA annotation software for images, audio and video," in *Proceedings of the 27th ACM International Conference on Multimedia*, ser. MM '19. New York, NY, USA: ACM, 2019. [Online]. Available: https://doi.org/10.1145/3343031.3350535

[21] K. Ronanki, B. Cabrero-Daniel, and C. Berger, "Chatgpt as a tool for user story quality evaluation: Trustworthy out of the box?" in *International Conference on Agile Software Development*. Springer, 2022, pp. 173–181.

[22] *Github Repository - Use of LLM for pedestrian detection*, 2019, accessed: 2024-01-29. [Online]. Available: https://github.com/MalshaMahawatta/UseofLLM_AirDnD

[23] *GPT-4V - Vision Preview*, 2024, accessed: 2024-01-29. [Online]. Available: https://platform.openai.com/docs/guides/vision

[24] A. Geiger, C. Wojek, and R. Urtasun, "Joint 3d estimation of objects and scene layout," in *Advances in Neural Information Processing Systems (NIPS)*, 2011.

[25] C. Li, J. Wang, Y. Zhang, K. Zhu, W. Hou, J. Lian, F. Luo, Q. Yang, and X. Xie, "Large language models understand and can be enhanced by emotional stimuli," 2023.

[26] R. Feldt and A. Magazinius, "Validity threats in empirical software engineering research - an initial survey." 01 2010, pp. 374–379.

Optimizing End-to-End test execution: Unleashing the Resource Dispatcher - WiP

Cristian Augusto [*1], Jesús Morán [2], Claudio de la Riva [3] and Javier Tuya [4]
Computer Science Department, University of Oviedo
Campus de Viesques, Gijón, Asturias
[1] augustocristian@uniovi.es
[2] moranjesus@uniovi.es
[3] claudio@uniovi.es
[4] tuya@uniovi.es

Abstract—**Continuous integration practices have transformed software development, but executing test suites of modern software developments addresses new challenges due to its complexity and its huge number of test cases. Certain test levels, like End-to-end testing, are even more challenging due to long execution times and resource-intensive requirements, moreover when we have many End-to-end test suites. Those E2E test suites are executed sequentially and in parallel over the same infrastructure and can be executed several times (e.g., due to some tester consecutive contributions, or version changes performed by automation engines). In previous works, we presented a framework that optimizes E2E test execution by characterizing Resources and grouping/scheduling test cases, based on their compatible usage. However, the approach only optimizes a single test suite execution and neglects other executions or test suites that can share Resources and lead to savings in terms of time and number of Resource redeployments. In this work, we present a new Resource allocation strategy, materialized through a Resource Dispatcher entity. The Resource Dispatcher centralizes the Resource management and allocates the test Resources to the different test suites executed in the continuous integration system, according to their compatible usage. Our approach seeks efficient Resource sharing among test cases, test suites, and suite executions, reducing the need for Resource redeployments and improving the execution time. We have conducted a proof of concept, based on real-world continuous integration data, that shows savings in both Resource redeployments and execution time**

End-to-End Testing; E2E Testing; Test Suite Optimization; System Testing; Testing Resources; Test Optimization; Testing

I. INTRODUCTION

Continuous integration (CI) practices that integrate and test software code automatically are widely adopted in both academia and industry, reducing the duration of development cycles from years/months to weeks, or even days [1]. These shortened cycles have impacted critical development stages such as software testing, where validating modern software developments is even more complex, due to longer and costly test suites comprised of thousands of test cases that are executed frequently [2], [3].

Apart from the challenges faced by CI, the testing level End-to-End testing (E2E) present additional challenges due their high cost. E2E testing validates from the user iteration to the low-level layers like persistence or networks and are costly due to long execution times, expensive test Resources[1] [4] or requiring the entire system up for their execution. The Resources are the physical (e.g. a mobile device or a physical sensor), logical (e.g. a database or a webserver) or computational (e.g. a lambda function or a container provided in Azure containers) entities that are required by a test suite during its execution. Although there are techniques that aim to optimize the test suite execution, such as test prioritization, selection, and minimization [5] that are effective in other testing levels [6], [7]. However, in E2E testing these traditional techniques are less effective because they continue to require the same expensive Resources/system for their execution.

In previous works, we introduced RETORCH: Resource-Aware End-to-End Test Orchestration [4], a framework that optimizes the E2E test execution through a characterization of the Resources used, a grouping and scheduling of the test cases according to their usage. The groups of compatible test cases are scheduled and executed against their Resources, exclusively deployed and tear down for them. This grouping and scheduling of test cases reduces the execution time and the number of unnecessary Resource redeployments to execute the suite as it enables test parallelization and concurrent execution over the Resources. However, when several executions of the same test suite (e.g., some repository changes committed closely, pull requests opened to test several configurations or dependency updates) are executed in the same CI system, there is still room for improvement.

Throughout the life of a software project, these additional Resource deployments during hundreds or even thousands of CI executions impact the total project budget, especially in a Cloud environment where you only pay for what you use—but you pay for everything you use. [8].

In this paper, we propose an approach that extends the RETORCH framework with a Resource Dispatcher that enables

[1] Henceforth, we will use the term "Resources" (capitalized) when referring to the ones required by the E2E test suite.

E2E test Resource sharing between different test cases and suite executions (e.g., two consecutive commits or pull requests opened in the repository). The objective of this Resource Dispatcher is to take advantage of already deployed Resources and share those Resources between the test cases of different test suites or test suite executions, whenever the tests perform a compatible Resource usage (e.g., test cases that do not modify the Resource or restore its original state after its execution). To achieve this, we propose a Resource Dispatcher, that is integrated with the RETORCH approach, centralizing the Resource management, deploying and tearing down the necessary Resources for the entire continuous integration system.

The use of this Dispatcher has several scenarios. For example, it can assign an already deployed Resource in the CI (e.g., an ELK stack or a Selenoid Instance), one Resource that previously belonged to another TJob and make a compatible usage (e.g., a database that was cleaned before its usage). Another feasible strategy could be allocate a Resource that is being used concurrently by other TJobs belonging to other execution plans. The Execution plans [4] are TJobs scheduled in sequential or parallel aimed to reduce the execution time and the number of Resource redeployments during the E2E test execution. The Resource Dispatcher is applied over a case study with the CI data of a real demonstrator that showcases differences in Resource redeployments and execution time.

The rest of the paper is structured as follows: Section II provides the necessary background, Section III presents the approach, Section IV presents the proof of concept with the real demonstrator CI data, and finally, Section V presents the conclusions.

II. BACKGROUND

The RETORCH orchestration approach [4] is composed of four processes: Resource identification, grouping, scheduling, and deployment. In the Resource identification process, the tester performs a smart characterization of the Resources with different static and dynamic attributes that describe the Resources and show how they are used by the test cases. For instance, examples of these attributes are the maximum number of Resources, their cost, the hierarchy relationships, or the specific access mode (e.g., read, write, read-write). The output of the Resource identification is used in the grouping and scheduling processes that group and schedule the test cases with the containerized SUT in the so-called TJobs that are arranged sequentially and in parallel in the Execution Plan. The Execution Plan is deployed during the deployment phase generating the necessary pipelining and scripting code (e.g., Jenkins Jenkinsfile, GitHub actions YAML files, or Travis travis.yml).

The TJobs execution follows a lifecycle composed of different phases. First, a set-up is performed in which various actions are required to deploy and configure the environment. This set-up is followed by test execution (onwards exec), during which one or several test cases with compatible Resource usage are executed together. Finally, the tear-down phase performs cleaning and release actions, as well as saving results and other debugging information, such as different logs.

In RETORCH, the Resource management is handled by each TJob, which is responsible for deploying and releasing Resources. This strategy is efficient if the Execution Plan is executed alone in the CI system and not executed frequently, but there is room for improvement when the CI system has already deployed Resources, other Execution Plans, or executions of the same plan sequentially or in parallel. Some of the Resources already deployed can potentially be used by other TJobs that belong to another Execution Plan executed later or in parallel.

III. RESOURCE DISPATCHER FOR E2E TESTING

The new approach aims to enable Resource sharing between different Execution Plans, minimize the number of Resource redeployments, and also to reduce the execution time, because uses deployed Resources and not wait for its instantiation.

To enable this Execution Plan Resource sharing, the concept of Resource is refined to include the possibility of sharing them between different TJobs. In other words, the TJobs can take advantage of already deployed Resources whereby their usage does not impact their or other TJobs execution.

We propose to decouple this Resource management (deployment and tear-down) of the TJob, through a Resource Dispatcher (henceforth referred to as Dispatcher). The Dispatcher introduces the role of test Resource manager who is responsible for managing the Resources used within the CI environment. The general process is depicted in Fig. 1, which gives as input several pull requests (PR) opened with their Execution Plans (that can be different or several executions of the same plan). When the CI starts with the plan, the different TJobs start their execution sequentially and in parallel, requesting different Resources from the Dispatcher. The CI executes the different TJobs until the last has ended, which continues until the last TJob has finished.

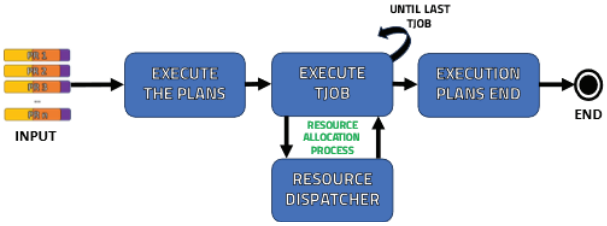


Figure 1 General overview of the process

The Resource allocation process using the Dispatcher is depicted in Fig. 2. *T*he process starts with the TJob set-up, on which the Resources are requested. If a compatible Resource is already available, it is allocated. If not, a new Resource is instantiated. Then the Resource is used during the test execution phase and released before the end of the TJob in the tear-down phase.

When a TJob starts its execution, it enters into the set-up phase (in yellow), which requests a Resource from the Dispatcher (1), who checks whether the Resource is already deployed by other Execution Plans (2) by reviewing the

Resource Pool (3). If the Resource is not deployed, the Dispatcher deploys a new Resource and registers it in the Resource Pool (5), along with the type of access mode performed and its attributes, e.g., if it is possible to be shared with other TJobs or it can be accessed concurrently.

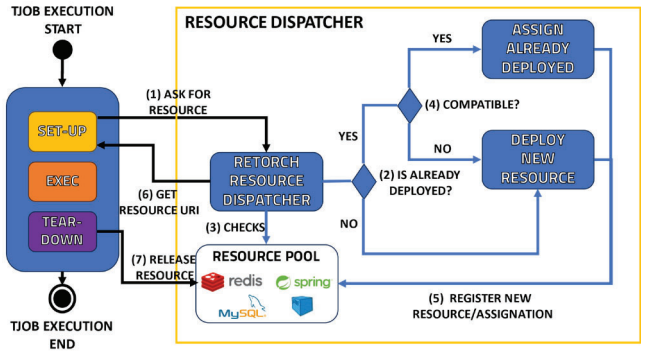


Figure 2 Resource Allocation process

If the Dispatcher has an already deployed Resource, it verifies whether the intended usage by the TJob is compatible with it (4). For example, two TJobs that use the same Resource without modifying it are compatible and can share it for their execution, saving the time of a new deployment. However, if TJob 1 modifies and pollutes the Resource, it must be executed with different Resources. If not, the Dispatcher proceeds with the instantiation of a new Resource (e.g. turns on the hardware device if it's a physical Resource, instantiates it on the air if it's a logical Resource, or asks the service provider for a computational Resource) and registers it within the Resource Pool with the usage that is performing the TJob (5). Conversely, if the Resource is compatible, it is assigned mapping this assignation in the Resource of the Resource Pool (5). The Dispatcher answers the TJob with the Resource (6), allowing the TJob to continue with its execution (in orange). When the TJob ends, during its tear-down phase (in violet), it notifies the Dispatcher (7) that this Resource is no longer required, giving to the Dispatcher the responsibility for the tear-down or its reasignation to another TJob.

IV. Evaluation

To assess the viability of the Resource Dispatcher, we carried out a proof of concept using the FullTeaching test suite [9], part of a demonstrator belonging to ElasTest Horizon 2020 European Project [10]. FullTeaching [11] is an online education platform designed to simplify the creation of courses and virtual classrooms, making remote teaching more accessible. FullTeaching is composed of several Resources, such as web and multimedia servers, relational databases, or web browsers.

RETORCH, with the information provided by the annotated Resources and access modes in the test cases, provides an Execution Plan composed of 12 TJobs deployed in parallel in groups of 5 TJobs. For this proof of concept, we focused on three different TJobs of this Plan: TJob-A, TJob-B, and TJob-C, whose Resources, number of test cases, and access modes performed are depicted in Table I:

TABLE I
TJobs and Resources access modes

	Access Modes		
TJob	**Database**	**Web Server.**	**Mult. Server**
A	R-W	R	R
B	R-W	R	R
C	R-W*	R	No-Access

All the TJobs **A, B, and C** modify the database, but **TJob-C** restores its state before concluding (R-W*), allowing the database to be used by subsequent test cases (albeit not concurrently). TJob C is also characterized by not accessing to the multimedia server, making it possible to mock it to provide only the health check, which is lighter than the entire Resource.

We employ the continuous integration data of the Friday 1st of March at 0:00 a.m. on which 5 pull requests were opened by DependaBot with different version updates in the GitHub repository [9]. This repository is integrated into our Jenkins continuous integration system, executing the Execution Plan for each new pull request created. The average times on which the different TJob lifecycle phases start and end all pull requests are shown in *Table II*:

TABLE II
Average times for PR Execution and deployment time (Resources)

	Resources			TJobs					
ID	**Database**	**Web Server**	**Multimedia Serv,**	**Set-up-start**	**Set-up-end**	**Exec-start**	**Exec-end**	**Tear-down-start**	**Tear-down-end**
TJob-A	29	30	46	1	48	49	120	121	126
TJob-B	29	30	46	1	47	48	149	150	153
TJob-C	29	30	46	1	49	50	121	122	124

Each Resource set-up individually takes on average in the different TJobs 28-29 seconds for the BD, 30.5 seconds for the multimedia server, and 46.36 seconds for the webserver. Fig. 3 depicts the difference in execution time with 3 parallel executions of the 5 different pull requests (from PR1 to PR5), using RETORCH's original approach (in blue) against RETORCH with the Dispatcher (in green).

For PR 2 and PR 3 executions, the Dispatcher alternative takes the same amount of time to set up. Part of this time is spent preparing the PR resources (indicated in yellow, with a 28-29 second BD setup wait), while the remaining 17-18 seconds are spent waiting for the readiness of the PR1 Resources (Multimedia Server). The reuse of the same Resources in TJob A only requires instantiating/cleaning the database in the different TJobs. PR 4 and PR 5 do not require waiting and the execution time is reduced. On the other hand, the RETORCH alternative requires instantiating the Resources for each TJob and tearing them down when it finishes, using more time than the other alternative in both phases and leading to a higher total

execution time, saving 21 seconds. These time savings are even more important in the CI systems where the test suites are executed hundred or even thousands of times at each repository change.

Figure 3 Duration of the different PR

Fig. 4 depicts the number of Resource redeployments of both alternatives during the whole PR, in blue RETORCH, and in green the RETORCH + Dispatcher alternative.

The Dispatcher alternative redeploys 70% fewer Resources (45 Resource re-deployments with RETORCH against 14 with RETORCH + Dispatcher) to execute the PRs because the TJob B and C share the multimedia and web servers of the first PR and clean the database for each TJob, while the TJob A instantiates its Resources in the first execution and reuses them in the subsequent executions. The reduction of Resource instantiations is useful when the Resources are limited (e.g. physical limitations like the number of available devices) or when the testing is carried out over the Cloud and each instantiation impacts the total project budget.

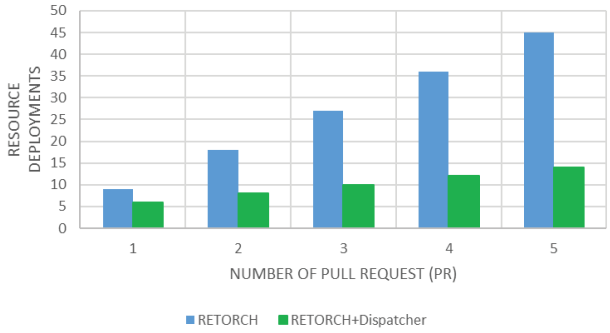


Figure 4 Number of Resource redeployments

V. CONCLUSIONS

We have proposed the Resource Dispatcher, that evolves the current RETORCH Resource allocation approach into a centralized way to manage the Resources. The Dispatcher enables Resource sharing between different Execution Plans or executions of the same Execution Plan itself, which can be performed when a pull request or several consecutive contributions arrive at the repository.

Through a proof of concept using real-world continuous integration data, we show the feasibility of our proposed approach, showcasing benefits in terms of Resource savings and execution time. This remarks the practical applicability of our solution in addressing the challenges posed by modern software development practices, where the complexity of test suites and the need for frequent testing require innovative solutions to streamline the testing process and continue improving and enhancing the software quality.

As future work, we plan to evaluate our approach in more demonstrators and test suites. Additionally, we intend to combine RETORCH with other optimization techniques such as test-batching. Another research line involves incorporating the Resource Dispatcher into a bot engine and integrating it with the RETORCH platform

ACKNOWLEDGMENTS

This work was supported by the project PID2022-137646OB-C32 under Grant MCIN/AEI/10.13039/501100011033/FEDER, UE

REFERENCES

[1] M. Meyer, "Continuous integration and its tools," *IEEE Softw.*, vol. 31, no. 3, pp. 14–16, 2014, doi: 10.1109/MS.2014.58.
[2] H. Esfahani *et al.*, "CloudBuild: Microsoft's distributed and caching build service," in *Proceedings - International Conference on Software Engineering*, in {ICSE} '16. ACM, 2016, pp. 11–20. doi: 10.1145/2889160.2889222.
[3] A. Memon *et al.*, "Taming google-scale continuous testing," in *Proceedings - 2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track, ICSE-SEIP 2017*, IEEE, May 2017, pp. 233–242. doi: 10.1109/ICSE-SEIP.2017.16.
[4] C. Augusto, J. Morán, A. Bertolino, C. de la Riva, and J. Tuya, "RETORCH: an approach for resource-aware orchestration of end-to-end test cases," *Softw. Qual. J.*, vol. 28, no. 3, pp. 1147–1171, Sep. 2020, doi: 10.1007/s11219-020-09505-2.
[5] S. Yoo and M. Harman, "Regression testing minimization, selection and prioritization: A survey," *Software Testing Verification and Reliability*, vol. 22, no. 2. John Wiley and Sons Ltd., pp. 67–120, Mar. 2012. doi: 10.1002/stv.430.
[6] G. Rothermel, M. J. Harrold, J. Ostrin, and C. Hong, "Empirical study of the effects of minimization on the fault detection capabilities of test suites," in *Conference on Software Maintenance*, 1998, pp. 34–43. doi: 10.1109/icsm.1998.738487.
[7] W. E. Wong, J. R. Horgan, A. P. Mathur, and A. Pasquini, "Test set size minimization and fault detection effectiveness: A case study in a space application," *J. Syst. Softw.*, vol. 48, no. 2, pp. 79–89, 1999, doi: 10.1016/S0164-1212(99)00048-5.
[8] A. Eivy, "Be Wary of the Economics of 'Serverless' Cloud Computing," *IEEE Cloud Comput.*, vol. 4, no. 2, pp. 6–12, 2017, doi: 10.1109/MCC.2017.32.
[9] C. Augusto, J. Morán, C. de la Riva, and J. Tuya, "FullTeaching E2E Test Suite." 2023. [Online]. Available: https://github.com/giis-uniovi/retorch-st-fullteaching
[10] B. Garcia *et al.*, "A proposal to orchestrate test cases," in *Proceedings - 2018 International Conference on the Quality of Information and Communications Technology, QUATIC 2018*, 2018, pp. 38–46. doi: 10.1109/QUATIC.2018.00016.
[11] ElasTest EU Project, "Fullteaching: A web application to make teaching online easy." Universidad Rey Juan Carlos, 2017. Accessed: Aug. 10, 2023. [Online]. Available: https://github.com/pabloFuente/full-teaching

Project Security Using Analytical Time Evaluation Techniques

Maksim Goman
LIT Secure and Correct Systems Lab
Johannes Kepler University Linz
Linz, Austria
maksim.goman@jku.at

***Abstract*—Time planning of IT Projects is difficult due to a large number of uncertainties in the software development process. Mistakes in software development planning open ways to architectural, functional, and integrative deficiencies of the product and surrounding processes. Using a simple synthetic project, we compare three analytic techniques to evaluate project duration. These three techniques can work with uncertain input parameters, represent projects as stochastic activity networks, and use simple computations to approximate distributions of the start time and end time of all tasks. Most importantly, they address the problem of merge event bias. We observe that a method with most relaxed assumptions performs better than others in comparison with a simulated ideal solution.**

Keywords-project management; software development; project time estimation; merge event bias

I. INTRODUCTION

Time planning in software development projects has always been problematic due to uncertainty of required resources, time, specifications, and unreliable human estimates. A recent review on the development of evaluation techniques of project time is given in [1]–[3]. Modern software projects contain uncertainty in virtually all possible parameters, the project environment is usually unstable and changeable (including requirements specification), and there are no established tools to help project managers with time analysis. Simple techniques that can work with roughly estimated input data seem useful for this situation in industry.

Program evaluation and review technique (PERT) [4, p. 269] is always mentioned in the literature for practitioners as an analytical probabilistic method for project time analysis [2,3]. This technique searches for only one critical path (CP), and have very restrictive assumptions for applicability. Another way of reasoning is based on approximations of the effect of merge event bias (MEB) [4, p.296] in each node of the stochastic network. MEB shifts the mean of the time distributions (TD) to the right, i.e. to a longer duration.

Although the issue of MEB is not well communicated in today's literature, a practical technique should include MEB into consideration, because it is a major model factor of underestimation of project time [1]–[4] beyond estimating *uncertain parameters*. As we are working on a new analytical method for project time analysis for very uncertain projects, our objective is to evaluate performance of simple techniques which address MEB problem. We have found no recent traces of such a comparison of any analytical techniques for project time analysis. We have identified three analytical project evaluation techniques that can operate under high uncertainty of task duration estimates, and they include a MEB correction procedure. These are modified PNET algorithm [4], a technique for estimating the distribution of a stochastic project makespan by Cohen and Zwikael [5] (hereinafter C-Z method), and the Pessimistic project evaluation and review technique (PPERT) [3].

The PNET method is an extension to classical PERT technique based on estimates and initial task TD of classical PERT. As in PERT, the normal distribution is assumed for eventual path duration. It considers paths in the stochastic network of the project but introduces a MEB correction step. The task durations are not necessarily statistically independent. A heuristic is given to decrease the number of paths involved in computations. Then, they select a set of representative paths determined using a proposed linear correlation coefficient between pairs of project paths and a given threshold 0.5. If the correlation coefficient of two paths is less than 0.5, then the two paths are assumed independent and representative. Otherwise, the "longer" path is selected to represent both paths and the shorter path is removed from consideration. The representative paths are assumed statistically independent and hence having few or no common tasks. It is required to generate the list of paths (starting with the CP). The cumulative distribution function (CDF) of the project completion time is estimated as a product of CDFs of the representative paths.

The C-Z method can work with diverse *symmetric* distribution types, and it approximates the CDF of project time in each network node using discretization, interpolation, and extrapolation. The main point is that the maximum distribution in nodes differs from the distribution of the longest expected time among the predecessor tasks. Assuming independence of predecessor tasks of a node, it is possible to estimate (bound) the CDF of the event in the node as a product of CDFs of end TD of its immediate predecessor tasks connecting the current node and preceding nodes. The TD of the immediate predecessor task is the TD of the preceding node plus the random duration of the predecessor task. Thus, this algorithm does not consider paths and needs only one traverse through

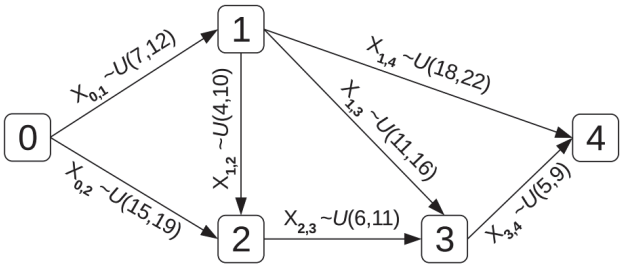
the project network to compute all approximations of end TD of nodes. The meth adds quantiles of distributions of consecutive tasks, and multiplies quantiles of parallel collapsing tasks.

The method PPERT needs uniform distributions of task times as the input. Additionally to assumed task dependencies, other non-statistical dependencies (logical, technical, procedural, etc.) may be known as background information. Therefore, accepting the high uncertainty of the outcome of time of each task, a symmetric triangular distribution is used for the approximation of TD in nodes of the stochastic network as a MEB correction procedure. The chance constraint is used to compare two distributions to determine if MEB correction is required or the "smaller" path can be dropped, and the distribution of the longer "dominating" path can be set as the TD in the node. One traverse through the network is required. A simple correction of the upper bound of the final TD of the end node is required if MEB corrections were applied.

We evaluate the application of the methods using an artificial generic network taken from [2]. We will compare CDFs of the project duration obtained with the three methods and classical PERT to CDF of a simulated makespan. The network includes two paths of relatively equal probabilistic duration and generates MEB in three nodes.

II. Analysis of Example Project

The project network is given in Fig. 1 in activity-on-arc notation, and input data are given in Table I. Random time variables of nodes i are denoted as X_i, $i = 0,1,2,3,4$ and random time of tasks $X_{i,j}$,is indicated as $X_{i,j}$, where i and j are their begin and end nodes. Their respective CDF are F_X_i and $F\_X_{i,j}$. For paths, the indices of the origin, intermediary nodes, and the end node are used with hyphens, e.g. random time of path X_{0-1-3}.



The network of the example project

Initial time estimates of tasks are uniformly distributed. These data are used directly in C-Z and PPERT techniques. Beta distributions for classical PERT and, respectively, PNET algorithm, were obtained with simple approximations from PERT theory: the mean time is *(o+4m+p)/6* and standard deviation is *[p-o]/6* [5], where optimistic (o), pessimistic (p) and most likely (m) values coincide with the minimum, maximum and mean parameters of given uniform distributions.

We are searching for an approximated maximum time distribution in nodes 2, 3, and 4. The distribution in node 4 is also project makespan.

TABLE I. Input Data

Task	Min	Max	Mean	Variance	Std. dev.
$X_{0,1}$	7	12	9.5	0.6944	0.8333
$X_{0,2}$	**15**	**19**	**17**	**0.4444**	**0.6667**
$X_{1,2}$	4	10	7	1	1
$X_{2,3}$	**6**	**11**	**8.5**	**0.6944**	**0.8333**
$X_{1,3}$	11	16	13.5	0.6944	0.8333
$X_{1,4}$	18	22	20	0.4444	0.6667
$X_{3,4}$	**5**	**9**	**9**	**0.4444**	**0.6667**

Arcs related to CP of the classical PERT are marked in bold in Table I. The solution of the classical PERT for all nodes is given in Table II. The outcome is obvious. The PNET solution is shown in Table III, the outcome of the C-Z method is given in Table IV, and PPERT solution is in Table V.

A. PNET solution

As paths 0-2 and 0-1-2 are not correlated using the correlation coefficient, the PNET solution for node 2 is trivial: CDF $F\_X_2 \sim N$(mean, std. dev.) $= N(17,\ 0.6667) \cdot N(16.5,\ 1.3017)$. Solutions for nodes 3 and 4 are presented in Table III. Paths from the origin to node 3, respectively, node 4, are ranked in descending order of the mean path duration in the left part of the table. In the right part of the table are correlation coefficients for each pair of significant paths. Following the given heuristic rule for node 3, path P13 is not considered as its mean differs from the first (critical) path by more than twice the larger of its standard deviations, i.e. $25.5 - 23 = 2.5 > 2.3570 = 1.1785 \cdot 2$. In the same way, for node 4, path P4 is not considered as its mean differs from the first (critical) path by more than twice the larger of its standard deviations, i.e. $32.5-29.5 = 3 > 2.5166 = 1.2583 \cdot 2$. The respective correlation coefficients appear in italics. For node 4, the correlation coefficient between paths P1 and P2 is 0.5377>0.5, therefore the path P1 represents both paths and the path P2 is removed. PNET solution for node 3 is $F\_X_3 \sim N(25.5,\ 1.0672) \cdot N(25,\ 1.5456)$, and for node 4 is $F\_X_4 \sim N(32.5,\ 1.2583) \cdot N(30,\ 1.3540)$.

TABLE II. PERT Solution

Node	Path	Distribution
1	0-1	*N* (9.5, 0.8333)
2	0-2	*N* (17, 0.6667)
3	0-2-3	*N* (25.5, 1.0672)
4	0-2-3-4	*N* (32.5, 1.2583)

B. C-Z solution

According to the algorithm, we begin with tasks $X_{0,1}$ and $X_{0,2}$ as those without predecessors and set their start-time to zero. The maximum time in node 1, and start time of

Table III. PNET SOLUTION

Node 3									
Path id.	***Path***	***Mean***	***Variance***	***Std. dev.***	***/***	***P11***	***P12***	***P13***	
P11	0-2-3	25.5	1.3389	1.0672	P11	1	0.4210	0	
P12	0-1-2-3	25	2.3889	1.5456	P12	–	1	*0.3812*	
P13	*0-1-3*	*23*	*1.3889*	*1.1785*	P13	–	–	1	
Node 4									
Path id.	***Path***	***Mean***	***Variance***	***Std. dev.***	***/***	***P1***	***P2***	***P3***	***P4***
P1	0-2-3-4	32.5	1.5833	1.2583	P1	1	**0.5377**	0.2609	0
P2	0-1-2-3-4	32	2.8333	1.6833	P2	–	1	0.4997	0.3866
P3	0-1-3-4	30	1.8333	1.3540	P3	–	–	1	*0.4806*
P4	*0-1-4*	*29.5*	*1.1389*	*1.0672*	P4	–	–	–	1

TABLE IV. SOLUTION WITH C-Z

CDF/Days	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**
F_X_4Tc	0	0	0	0.002	0.004	0.112	0.214	0.329	0.443
	34	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
	0.543	0.631	0.715	0.793	0.855	0.911	0.955	0.999	1
F_X_4Ti	0	0	0	0	0	0.050	0.131	0.230	0.351
	34	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
	0.495	0.594	0.676	0.741	0.799	0.850	0.900	0.950	1

immediately following tasks $X_{1,2}$ and $X_{1,3}$ is CDF of the respective predecessor task s $X_{0,1}$.

For node 2, start-time CDFs are discretized, and sums or convolution are computed as completion TD for the task $X_{1,2}$ is TD of path 0-1-2. CDF of maximum time in node 2 (F_X_2Tc, F_X_2Ti) and start time of immediately following task $X_{2,3}$ are computed, and required quantiles are extrapolated. Computations for nodes 3 and 4 are similar. The approximate discretized distributions of makespan are in Table IV.

C. *PPERT solution*

Initially, the end time of tasks $X_{0,1}$ and $X_{0,2}$ are known. The start and end time of all other tasks are set to 0, and TD in all nodes, start and end time of the rest of the tasks are assigned "unset". The steps are given in Table V. We go from the source to the sink of the network computing approximate distributions in nodes searching the distribution of maximum project time (DMPT) in each node. Node distributions depend on end time distributions of incoming tasks and determine start time distributions of outgoing tasks.

As the end TD of task $X_{0,1}$ is known. It is equal to TD of the node 1 and start time of tasks $X_{1,2}$ and $X_{1,3}$. End time of task $X_{1,2}$ is a convolution of TD in node 1 and the duration of the task. Now, knowing the end TD of tasks $X_{1,2}$ and $X_{0,2}$, we can determine the TD in node 2 where two sub-paths are met in step 1 (see Table V).

In node 2, the domination of one sub-path A measured with probabilistic comparison operation (PCO) A>B (and B>A) is less than chosen reliability coefficient α = 0.9. Therefore, we perform MEB correction with a symmetric triangular distribution with $P_1 = h(A, B) \sim Tri(15, 18.5, 22)$ and continue with this triangular distribution as the distribution P1 of the node 2 in the next step.

In step 2 we compare distributions of two collapsing subpaths A: *Tri*(15, 18.5, 22) and *U*(6, 11), and B: *U*(7, 12)∗ *U*(11, 16) in the next node 3. The former distribution is larger than the latter compared with PCO with the given α. Therefore, we continue with *Tri*(15, 18.5, 22)∗*U*(6, 11) under the name P2 as the distribution of the node 3 in step 3.

Step 3 is performed by analogy. P3 with support [26, 42] is the approximation of the makespan. Because there was MEB correction in node 2, the upper-bound is adjusted by (1−α)%: (42-26)*0.9 = 14.4, so, the new support is [26, 40.4]. The distribution in node 4 is symmetric triangular *Tri*(26, 33.2, 40.4).

III. Conclusion

A summary of the analysis is given in Fig. 2. Approximate CDF of makespan depends on the method. Although the mean is almost the same with all methods, of interest is naturally the probability *above the mean.* This is a manifestation of a longer possible delay. A simulated makespan is a "perfect" time to which the methods should be close to.

As indicated previously (e.g. [2], [4]), classical PERT underestimates probabilities of longer duration. Although PNET corrects it, the improvement is small due to reliance on paths instead of analysis of nodes where MEB emerges and accumulates. On the opposite, the C-Z method overestimates the duration. Moreover, overestimation is larger if one uses by design only discretized TD and interpolation (extrapolation) to calculate distributions of successor tasks without application of sometimes available original CDF or their convolutions. It follows from Fig. 2 where lines C-Z1 and C-Z2 show the results with Tc and Ti suffixes respectively.

The approximation of PPERT is *closer* to the simulated result despite its simple type of approximating function. This simple example shows that under- and overestimation can be visible even in a small network. There is no data on the behavior of unsophisticated heuristics in larger stochastic networks. However, the error can become larger. Although comparison to simulated distributions is a usual way of performance comparison, a better way were to apply the methods to data from real projects and compare results to recorded end times of each task and makespan. The latter is our incentive for future research in this area.

Table V. Processing with PPERT, $\alpha = 0.9$

Step	A	B	P(A>α B)	DMPT	MEB Corr.	Next step
1	$X_{0,2}$	$X_{0,1}*X_{1,2}$	0.43	$P_1 = h(A,B)$	Yes	2
2	$P_1*X_{2,3}$	$X_{0,1}*X_{1,3}$	0.91	$P_2 = A$	No	3
3	$P_2*X_{3,4}$	$X_{0,1}*X_{1,4}$	0.93	$P_3 = A$	No	–

References

[1] P. Ballesteros-Pérez, G. D. Larsen, and M. C. González-Cruz, "Do Projects really end late? On the shortcomings of the classical scheduling techniques," J. Technol. Sci. Educ., vol. 8, no. 1, p. 17, Mar. 2018.

[2] M. Goman and J. Pecerska, "Merge Event Bias in Project Evaluation Techniques – Problems and Directions," in 2020 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS), Oct. 2020, pp. 1–6.

[3] M. Goman, "A Heuristic Technique for Project Time Analysis in Conditions with High Uncertainty," in Business Modeling and Software Design, B. Shishkov, Ed., Cham: Springer International Publishing, 2021, pp. 363–373.

[4] J. J. Moder, C. R. Phillips, and E. W. Davis, Project management with CPM, PERT, and precedence diagramming, 3rd ed. New York: Van Nostrand Reinhold, 1983.

[5] Y. Cohen and O. Zwikael, "A New Technique for Estimating the Distribution of a Stochastic Project Makespan:," International Journal of Information Technology Project Management, vol. 1, no. 3, pp. 14–27, Jul. 2010.

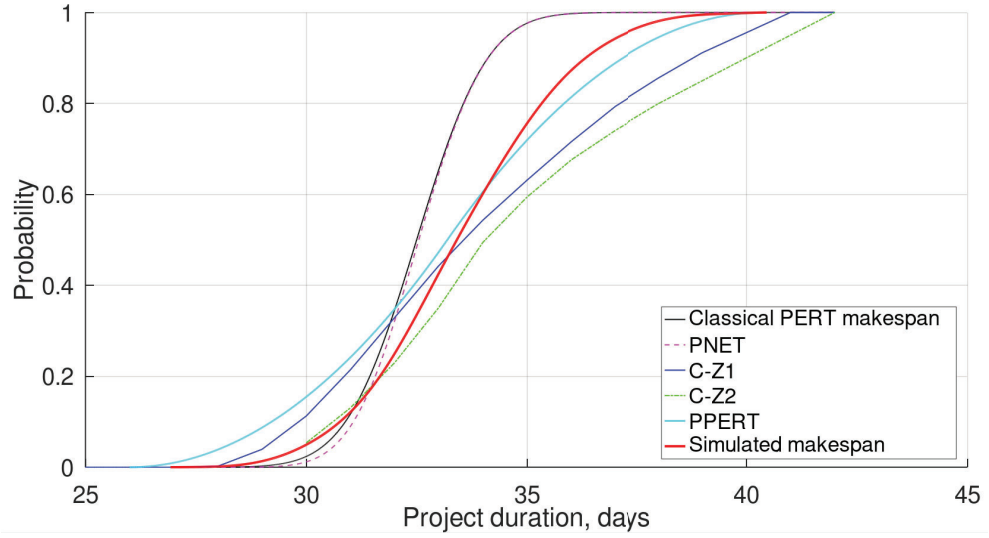


Figure 2. Approximated CDF of the project duration obtained with three methods

A Systematic Analysis of MLOps Features and Platforms

Leonhard Faubel
University of Hildesheim
Institute of Computer Science
Software Systems Engineering
Hildesheim, Germany, 31141
faubel@uni-hildesheim.de

Klaus Schmid
University of Hildesheim
Institute of Computer Science
Software Systems Engineering
Hildesheim, Germany, 31141
faubel@uni-hildesheim.de

Abstract—**While many companies aim to use Machine Learning (ML) models, transitioning to deployment and practical application of such models can be very time-consuming and technically challenging. To address this, MLOps (ML Operations) offers processes, tools, practices, and patterns to bring ML models into operation. A large number of tools and platforms have been created to support developers in creating practical solutions. However, specific needs vary strongly in a situation-dependent manner, and a good overview of their characteristics is missing, making the architect's task very challenging. We conducted a systematic literature review (SLR) of MLOps platforms, describing their qualities, features, tactics, and patterns. In this paper, we want to map the design space of MLOps platforms. We are guided by the Attribute-Driven Design (ADD) methodology. In this way, we want to provide software architects with a tool to support their work in the platform area.**

Keywords-MLOps, Design space, ML platforms

I. INTRODUCTION (HEADING 1)

While machine learning (ML) becomes increasingly important to company success, the continuous transition from model development to deployment in the field becomes increasingly important. Companies may take a month or even more to deploy an ML model. One reason is that data scientists often lack knowledge of Software Engineering (SE) and computer science in general. While they are able to solve business problems with analytics and ML algorithms, they often struggle to deploy these algorithms into software systems in the real world. MLOps engineers fill this gap with additional knowledge in software engineering and automation [1]. On the other hand, SE processes have to be adapted to ML lifecycles [2] to work properly. ML brings in novel characteristics into existing SE methods: It requires trained models, is often unpredictable, and copes with constantly changing data.

MLOps provides processes, tools, practices, and patterns to close this gap. It aims at increasing quality attributes like correctness or reliability [3]. MLOps relies on DevOps principles, an attempt to automate the deployment processes of new software versions in software engineering and extents this by integrating ML relevant adaptations of the processes. MLOps provides best practices and guiding principles around ML, including collaborative work, reproducibility, continuous delivery, testing, and monitoring [4]. It aims to increase quality, simplify the management process, and automate the deployment process of bringing ML models into production [5]. Digital platforms can be defined as products, services, or technologies that serve as a basis for a large number of companies and offer complementary products, services products, services, and technologies [6]. MLOps platforms are platforms that manage ML pipelines [7] and try to satisfy the abovementioned gaps and capabilities. Since most platforms have many overall features in common, they are mainly different in their particular application [4]. This paper discusses the design space of such MLOps platforms from a technical software engineering perspective. It categorizes and provides an overview as decision support for platform developers and software architects who choose platforms as an integration component of more extensive applications. Initially, 70 MLOps platforms were compared based on a search on Google and Google Scholar to gain a general understanding and define terms. Based on that knowledge, an SLR was conducted as described in Section III. The results of our analysis led to a thorough description of characteristics of functions relevant to MLOps as well as example tools that implement them. Together with descriptions of relevant tactics this provides the core of our analysis of the design space of MLOps platforms as given in Section IV. In Section V, we particularly describe some major platforms, while Section VI provides an analysis and Section VII concludes.

A. Background

Traditional systems are static and often meet a company's needs in terms of speed, reliability, and the ability to predict its outcome [3]. ML models, in contrast, are different: They have inferences whose function cannot always be verified. Sometimes, their behavior is neither predictable nor fully repeatable. Data, code, and models are cross-dependent, and data, pre-processing, and training are interrelated in very complex ways. In production, input changes may cause inadvertent behavior. Changes in data or parameters cause changes in the inference [8], so focusing on mitigation strategies is especially important.

MLOps as a collection of techniques, tools, practices, or processes for ML deployment in production [9], [10], [11].

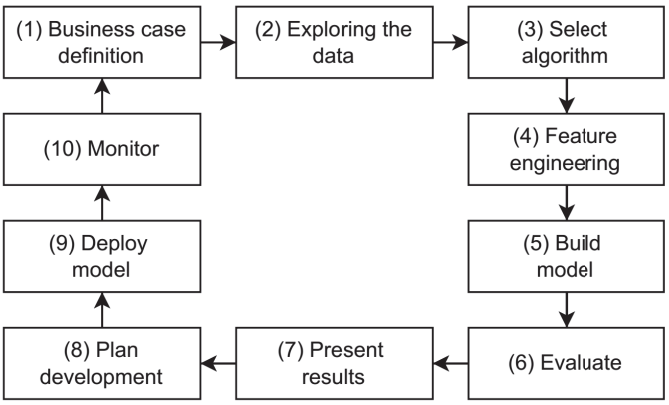


Figure 1: ML life-cycle adapted from [17].

aims to increase the level of automation [12], [13], [14]. A (1) business problem must be solved. Initially, the problem has to be fully understood. Further, the problem should be specified. Data should be acquired or made available for further steps [15]. For this, raw data must be processed to extract information, remove erroneous data, and bring it to a reasonable shape [16]. Then, the data needs to be (2) explored [15]. This involves finding correlations in the data by data scientists in collaboration with business domain experts to find technical challenges and first correlations in the data. Then, a fitting algorithm to the problem is (3) selected. (4) Feature selection means checking for similarities and impact on the prediction goals and eventually transforming raw data into other representations. A model is (5) built using the selected algorithm with appropriate features to use the correlation or patterns to predict. This model is (6) evaluated by checking how well the model performs using data that has not been used for training. The results are (7) presented and should be compared to other methods. This includes an analysis of the advantages: does its usage exceed the benefits or the cost? Then follows the (8) design of a pipeline that suits the needs of prior steps. After the model is (9) deployed as a module with interfaces (serving the model), it is (10) monitored by controlling the model behavior over time: often, there are underlying problems like drift under real-world
conditions, leading to anomaly and misbehavior of the model. Consequently, a typical ML life-cycle contains the activities shown in Figure 1.

II. Related Work

This section reviews literature related to the design space of MLOps platforms. We focus on architectural descriptions, implemented MLOps platforms, and systematic reviews. Several publications discuss MLOps architecture based on functional components, tools, and software infrastructures [18], [19], [20], [21]. Some implementations of MLOps platforms consider functional aspects as components to showcase the benefits of introducing MLOps tools [22], [23]. The architectures of such implementations are often domain-specific, e.g., Nia et al. [24] present an MLOps infrastructure for model deployment for telecom data that has been implemented and evaluated using metrics. Furthermore, authors reveal their design decisions by comparing features offered by different platforms, e.g., Zarate et al. [25] compare different tools based on predefined features, and Recupito et al. [26] conducted a multivocal literature review of the most common MLOps tools in which they examined features and functionalities. Compared to other literature reviews that address features, in this paper we look at 70 MLOps platforms and consider features from an SE perspective as a solution approach for software architects. Architectural design solutions are described. These include reference architectures based on components [18], [27], software architecture patterns [28], and cloud computing design patterns for MLOps [29].

III. Method

The method for this systematic literature review (SLR) followed the guidelines for performing SLRs in Software Engineering [30] and is described in more detail in a technical report [31]. At first, a search on the topic was carried out via Google and Google Scholar. Based on the results, an overview of MLOps platforms and an initial selection of published studies were obtained. This initial automated search helped to develop an initial understanding of the topic and to define terms and expressions that served as the basis for the search in this SLR. A comprehensive search strategy was developed to identify relevant studies. The search was performed in academic search engines including ACM Digital Library, IEEE Explore, and Science Direct using the keywords related to "MLOps" or "Machine Learning Operations". As a result, we included 47 white and grey literature studies and 95 self-published documents.

A. Research Questions

RQ1 How do MLOps platforms differ in their design?

RQ1.1 What features are commonly found in MLOps platforms?

RQ1.2 How do MLOps platforms support different qualities or characteristics in the context of their design and implementation?

RQ2 What are the current MLOps platforms?

RQ2.1 What can be learned from the current research results?

B. Selection Criteria

We included published research studies in English, such as conferences, journals, magazine papers, books, reports, and white papers using the term MLOps or Machine Learning Operations and dealing with MLOps platforms, automation, or software engineering published between January 2012 and May 2022. Self-published documents such as vendor homepages and blogs support our findings if no studies are available. Excluded are duplicate versions of studies, studies in languages other than English or German, studies without

reference to the research questions, talks without available information like protocols or notes, and posters.

Studies were excluded if they:

- Did not meet the inclusion criteria
- Were duplicates
- Were published in languages other than English

Different studies related to MLOps or Machine Learning Operations were excluded from the analysis. This was because MLOps can have various meanings, such as minimum linear operations or Multi-objective Lexicographic Optimization Problems. Additionally, the term Machine Learning Operations refers to supporting methods for ML, processing operations, or ML methods.

C. Data Extraction, Analysis, and Synthesis

Data extracted from the selected studies include item type, author(s), publication year, title, DOI, abstracts, and references. A thematic analysis approach was used to synthesize the findings and identify patterns across the selected studies. The quality of the included studies was assessed by checking whether the results were based on evidence and arguments, whether the study presented a research project, and whether the study's objectives were clearly defined. Studies that met these quality criteria were included in the final analysis.

D. Threats to Validity

Search on Google utilizes user preferences and factors such as search history and location, especially when no action is taken. Since MLOps is in the author's area of interest and he regularly searches for it concerning his work, we assume the bias could benefit our case. The Google and Google Scholar searches were incomplete, as there were too many results to review. However, the most relevant articles are listed first. During the assessment process, we identified several potential sources of bias that might have impacted the integrity of the findings. First, the identification of studies relied exclusively on selecting electronic databases. The initial search on Google and Google Scholar enabled us to check whether we had omitted any relevant studies. Further, bias could arise from only one author reviewing the articles, leading to subjective decisions. Another source of bias is the initial search of MLOps platforms, which helped select keywords and created an understanding of the study's vocabulary but did not exclude the inclusion of developer and commercial vendor articles that were not based on scientific methods. However, the information gathered there was purely technical and was supported by information from the subsequent SLR and our experiences on the topic.

IV. Design Space

This chapter describes features and design decisions in MLOps platforms. Features, which we consider as function groups, represent design decisions, while the quality aspects and criteria provide a rationale for these decisions.

Here we cover the quality aspects, technical features, and functionalities required in MLOps platforms. These include functionalities that support the activities of the MLOps lifecycle and features that increase the quality of MLOps. In some platforms, these features are implemented directly, while in others, they are made possible by openness and adaptability.

A. Quality Attributes

This section outlines context-specific quality attributes of MLOps platforms. Significantly more quality attributes are described in our technical report [31]. Figure 2 provides an overview of the exemplary properties described in detail below.

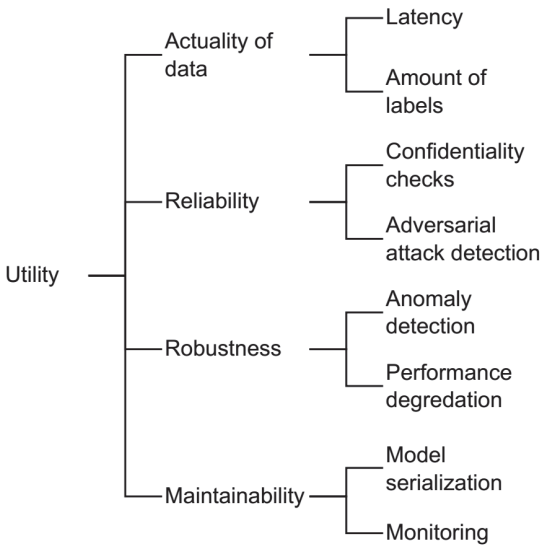


Figure 2: Context-specific overview of qualititesbased on a utility tree.

1) Actuality of Data: The higher the amount of labeled data, the higher the model's quality. Automatic labeling or label UIs can help gather new data. It may be beneficial to check if the correct labels exist and add a category for data that is not supposed to be predicted by the model to allow completeness. Additionally, the data should include other representations of each label. Standards in the data collection and labeling process allow consistency in terms of the expected data format and volume. However, labeling mechanisms have weaknesses since data must meet the intended needs and requirements to be relevant. This is not always granted due to human or machine error and must be checked frequently. That is why data cleaning is crucial. It helps to avoid typos, duplicate entries, measuring inaccuracies, missing features or values, and inaccurate labels. There may be a latency until the data appears in the database when information about a particular data point is recorded. Timeliness is essential when the system requires the latest and up-to-date data [17], [32]. Here, real-time environments for distributed processing of large datasets, data warehouses, and distributed search- engines [33] can help. Some of these environments support the high-performance processing of semi-structured in-memory data [34].

2) Reliability: Reliability is granted when a system continues to work correctly [35] and perform at the desired level, even in the face of adversarial attacks, hardware, software, or human errors [36]. Integrity and confidentiality checks and adversarial attack detection can face some of these challenges. CI, CD, and CT can avoid human errors [37].

3) Robustness: Robustness [38] is granted when a model can cope with an anomaly, performance degradation, and/or misbehavior.

4) Maintainability: Easing adapting to new data and maintaining the system behavior makes a system maintainable. Incorporating user feedback [39], [40], model serialization, monitoring, logging, model formatting, and documentation can help here [18].

B. Functional Groups

Several features must be available in the form of functional groups to support the MLOps lifecycle. We consider process, analytics, deployment, operations, and data storage as general function groups. These are described below for specific functions in design concept catalogs [41].

1) Process and Analytics: Data exploration, algorithm selection, feature engineering, model creation, and model performance evaluation are mandatory steps in machine learning, and they need to be supported by an MLOps platform (Table 1).

Table 1: Process and Analytics Catalog.

Specific Function	Duties	Example Tools
Data exploration	Visualization	Superset
		Grafana
	Data cleaning	pandas
	Data labeling	Doccano
		Label studio
Algorithm selection	AutoML	pycaret
		AutoGluon
		TransmogrifAI
	Hyperparameter tuning	Hyperopt
		Katib
		talos
Feature engineering	Feature extraction	Featuretools
		tsfresh
Model building	Machine learning	sklearn
		OpenCV
		MediaPipe
		Tensorflow
		Pytorch
		OpenNN

Data exploration: Identifying correlations in data and ensuring its suitability for analysis.

- **Visualization:** Representing data effectively to identify correlations and patterns. Transforming raw data for feature engineering and inference [15].
- **Data cleaning:** Detecting and correcting corrupt or inaccurate data values [42].
- **Data labeling:** Annotating or tagging data to enhance prediction accuracy [43].

Algorithm selection: Choosing an appropriate algorithm for the given problem.

- **AutoML:** Building, optimizing, and evaluating machine learning algorithms and pipelines, often with automated algorithm selection [44].
- **Hyperparameter tuning:** Automating the optimization of parameters [9].

Feature engineering: Identifying and extracting meaningful features from data.

- **Feature extraction:** Utilizing automated toolboxes to extract features from various types of datasets [45].

Model building: Constructing models using selected algorithms and learned features.

- **Machine learning:** Leveraging machine learning libraries for building and evaluating models.

2) Deployment and Operations: Further, there are deployment and operational features (Table 2). Some platforms allow models to be deployed in the cloud on the web, while others allow models to be deployed on-premise or on edge.

Table 2: Deployment and Operations Catalog.

Specific Function	Duties	Example Tools
Model as a Service	Model serialization	MLeap
		MLflow models
		pickle
		H20
	Model formatting	PMML
		PFA
		ONNX
	Model serving	spring
		tomcat
		django
		flask
Platform as a Service	Packaging and containers	Docker
		Kubernetes
		Kubeflow
Infrastructure as a Service	CPU cluster	CUDA
	GPU cluster TPU	OpenCL
	FPGA	Vitis AI
Communication	Service-oriented architecture	flask-RESTful
		gRPC
	Distributed event streaming	kafka
		RabbitMQ
Observability	Monitoring and Logging	Wavefront
		splunk
		datadog

Model as a Service (MaaS): Providing a fully functioning service for deploying models.

- **Model serialization:** Converting a trained model into a format or execution engine that can be easily saved, transferred, and reloaded without retraining.
- **Model formatting:** Structuring and arranging a serialized model to include metadata and apply necessary formats.
- **Model serving:** Serving a model as a service or dependency [46].

Platform as a Service (PaaS): Providing a platform for software development.

- **Packaging and containers:** Utilizing microservices for data pipelines [13]. Containers package code and depen-

dencies in an isolated environment, often managed by container orchestration platforms.

Infrastructure as a Service (IaaS): Providing computing resources as a foundation for cloud computing.

- **CPU Cluster:** Enabling CPUs to collaborate for computationally intensive ML tasks.
- **GPU Cluster:** Providing dynamic compute power like GPU clusters for training and scaling specific ML algorithms [47], [48].
- **TPU support:** Accelerating ML tasks with Tensor Processing Units (TPUs) for faster processing times [46].
- **FPGA support:** Using Field Programmable Gate Arrays (FPGAs) for ML tasks with faster processing times [49].

Communication: Enabling components and services to communicate effectively.

Service-oriented architecture (SOA): Creating loosely coupled services [50].

- **Distributed event streaming:** Managing the storage, distribution, and balancing of event streams across services [51].

Observability: Understanding system components for performance monitoring and issue detection.

- **Monitoring and logging:** Tracking model behavior over time, detecting data drift, anomalies, performance degradation, or misbehavior [15].

3) Data Storage: Repositories store data, code, models, and metadata. Specific databases are frequently used for various kinds of data and performance requirements (Table 3).

Table 3: Data storage catalog.

Specific Function	Duties	Example Tools
Versioning	Data repository	DVC
	Model repository	MLflow model repository
		DVC
	Code repository	GitHub
	Metadata repository	DVC
	Feature store	feast
NoSQL Databases	Reasoning and semantic	Neo4j
	Timescale	Cassandra
		MongoDB
		InfluxDB
	Caching and Queuing	Redis

Versioning: Tracking data, models, code, and metadata due to cross-dependencies.

- **Data repository:** Storing data along with its origin, acquisition details, and aggregations, often using different databases based on data type and performance requirements [15].
- **Model repository:** Storing mathematical models in machine-readable formats, possibly including validity checks [52].
- **Code repository:** Using distributed version control systems to manage code changes [37].

Metadata repository: Storing crucial metadata such as model lineage, versioning, aliasing, tagging, and annotations. Functions to store metadata are often included in data, code, and model registries [53].

- **Feature store:** Aiding feature engineering by storing and selecting suitable features, often including information about raw data transformations, aggregation, and management of feature values [54], [15].

Databases: Utilizing specialized databases for performancecritical applications.

- **Reasoning and semantic database:** Storing and interpreting relationships and meanings in data.
- **Timescale database:** Scaling queries for time-series data flexibly.
- **Caching and queuing database:** Enhancing performance and managing resource utilization.

C. Tactics

Tactics such as proven design strategies can improve various qualities [41]. We collected data on the quality aspects mentioned in the reviewed publications. These qualities can be categorized into data, models, and software. We could also extract tactics to address quality issues for most of the qualities. We summarized the qualities and constraints in Table 4. Attributes for which no reliable tactics were found in the literature are listed without naming suitable tactics. More detailed descriptions are provided in our report [31].

Table 4: Overview of quality attributes and constraints of the platforms.

Attribute	Tactics
Data Quality	
Accuracy	Avoid typos and duplicate entries
Completeness	
Consistent	Fill/remove missing entries, check for outliers
Currentness	Stream/Batch processing, update datasets frequently
Relevance	Collect data for the specific use case
Model Quality	
ML Performance	Monitor metrics for model evaluation
Explainability	
Software Quality	
Reliability	Monitoring, container management (dealing with errors in the distributed system)
Scalability	Distribution: containerization
Security	Encryption, software infrastructure updates
Performance	Latency: in-memory operations, model compression, reduce hardware requirements via model compression or frequency and number of calculations
Evolvability	Feature store, model repository
Maintainability	ML pipelines
Operability	Monitoring, retraining
Traceability	Storing and versioning of data, models, code, metadata, and their relationship
Constraints	
Cost	Costs can be reduced via software performance optimization
Development Time	Use automated setups
Model update time	ML pipelines, AutoML

Table 5: Platform categories.

Category	Description
Platform	ML supporting platforms and fully managed ML platforms.
Operating System	Entire operating system that is optimized for the operation and performance of ML.
Model-based	End-to-end platforms providing high-level scripting languages or graphical user interfaces.
Workbench	Offering managed programming environments with built-in integrations that help set up end-to-end MLOps production environments.
Package/IDE	Focused on the ML part, providing capabilities to put models into production.

V. Machine Learning Platforms

This section overviews infrastructure categories, licensing, and domain-specific platforms. In the initial search, we identified 70 MLOps infrastructures and categorized them as described in Table 5.

1) Licensing:: Commercial platforms such as *SageMaker*, *Azure MLOps*, *GCP*, and *valohai* are offered and managed by cloud providers. These platforms frequently cover the whole MLOps life-cycle and have AutoML tools embedded [15]. Models developed on the platforms of such cloud providers can be deployed on-demand using serverless technology, such as *Google Cloud Functions* and *Azure Functions*. However, commercial platforms often have the disadvantage of not being particularly customizable and open to technology variants. In addition, developers often struggle with vendor lock-in.

On the other hand, open-source licenses are free of charge and allow the software to be adapted and extended [19]. Further, developers who are concerned about putting their data in the cloud or being dependent on another company are implementing their own in-house MLOps platforms. Usually, they compose open-source platforms, tools, and libraries for that purpose, e.g., *MLFlow*, *Sacred*, and *DVC* (data version control) [55]. Tool overviews [56], [57], [34] can be used to find adequate options for such a compilation.

2) Domain-specific platforms:: Depending on the requirements, domain-specific platforms for container provisioning, automated ML, and big data can be useful. Many platforms use containerized workflows. Specialized container platforms for **container provisioning** platforms [58] are, e.g., *Docker*, *Kubernetes*, *OpenShift*, *AWS Elastic Container*, and *AWS Lambda*. Some platforms cover and **automate machine learning** development and deliver ready-to-production models [59], e.g., *Lobe AI*, *Google teachable machine*, and *Clarifai*. For **big data** processing, some of the reviewed platforms built up on platforms like *Databricks*, *Hadoop/Spark*, *Snowflake*, *Amazon Elastic Map Reduce* and *Google Big Query* [15].

VI. Discussion

Many developers use fully managed MLOps platforms provided by cloud service providers as those care for the entire infrastructure, servers, networks, security, and updates, allowing MLOps personnel to focus on their applications and data without worrying about the infrastructure. These platforms frequently offer services like virtualization, storage, and machine learning. Users can easily scale their resources up or down as needed but pay only for the resources they use [60].

In cases where companies have concerns about cost or about outsourcing data and using it in a cloud, in-house solutions are used to perform these tasks. This is also sometimes the case for addressing latency or bandwidth issues, e.g., in the case of Industry 4.0 scenarios [61]. With today's open-source libraries and tools, organizations can effectively compose platforms for their own purpose.

In our study, we identified features, tactics, design patterns, and reference architectures that are covered to varying degrees by the literature. In the existing literature, features for individual MLOps tools [25], [26] are described. Past literature, however, lacks features for MLOps platforms in specific. Therefore, our study summarizes MLOps **features** for platforms to fill that gap. Furthermore, we have extracted novel **tactics** to build such platforms. Not all quality attributes were addressed in the literature, so we could not provide a tactic for each of them, which leaves room for refinement in the future. **Design patterns** have been described by several sources [62] in the past. Moreover, reference architectures for MLOps [18], [27] and MLOps with specific requirements such as explainable AI and feedback in industrial use cases [63] are present in the existing literature. In summary, we provide an overview of the design space that can help in the design of MLOps platforms, but it does not cover all types of problems. Area-specific problems may occur that are not fully covered by this overview.

In our study, we found that the existing literature partially addresses challenges related to hardware and software optimization and containerization. Workflow management platforms grant scalability and help manage, scale, and deploy complex infrastructure. Searching the existing literature, we could not find any platform design concepts that reflect specific Machine Learning issues of large language models, e.g., encapsulating the multi-tier nature of fine-tuning aspects. We assume that a second tier is necessary for this purpose in addition to the training tier, which also requires high computing power and scalability at the user interface level. The first tier would be base model training with extensive datasets on high-performance clusters with GPU support. The second tier specializes the models for specific tasks and datasets. For example, scalable cloud instances could be accessed for webbased interfaces.

There is already some basic understanding and templates [64] on how to build MLOps platforms. These templates need adaptions for specific requirements. However, based on the existing literature, the question of how to build MLOps platforms is only partially answered from a software engineering perspective. With our analysis, we contribute to this knowledge and open up possibilities for future research. Future studies should extend this knowledge concerning integrating explainability and feedback into MLOps platforms and explore how to evaluate them.

VII. Conclusion

MLOps platforms frequently have a similar architecture [3]. This indicates that there are common qualities, features, and tactics relevant to their design. In this paper, we analyzed features of existing platforms as well as relevant design concepts like existing platforms and tools that can be used in the systematic creation of specialized MLOps platforms. We also discussed relevant qualities and potential tactics to achieve them.

We believe that the various design concepts that we identified are rather useful both for designers of innovative or customized MLOps platforms, as well as for potential users, who can use it as a basis to identify appropriate platform characteristics to select among potential alternative platform providers.

Acknowledgements

This work is supported by the project EXPLAIN, funded by the German Federal Ministry of Education under grant 01—S22030E. Any opinions expressed herein are solely by the authors and not the funding agency.

References

[1] M. Przybyla, "Data scientist vs machine learning ops engineer. here's the difference.", [Online]. Available: https://towardsdatascience.com

[2] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software engineering for machine learning: A case study," in *IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), 2019, pp. 291–300.*

[3] V. Maya and A. Felipe, "The state of MLOps," 2021. [Online]. Available: https://repositorio.uniandes.edu.co/handle/1992/51495

[4] Valohai, "MLOps - machine learning operations." [Online]. Available: https://valohai.com/mlops/

[5] MLOps: What it is, why it matters, and how to implement it. [Online]. Available: https://neptune.ai/blog/mlops-what-it-is-why-itmatters-and-how-to-implement-it-from-a-data-scientist-perspective

[6] A. Hein, M. Schreieck, T. Riasanow, D. S. Setzke, M. Wiesche, M. Bohm, and H. Krcmar, "Digital platform ecosystems," ¨ *Electronic Markets*, vol. 30, no. 1, pp. 87–98, 2020.

[7] S. Oladele, "Best end-to-end MLOps platforms: Leading machine learning platforms that every data scientist need to know." [Online]. Available: https://neptune.ai/blog/end-to-end-mlops-platforms

[8] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, and M. Young, "Machine learning: The high interest credit card of technical debt," in *SE4ML: Software Engineering for Machine Learning (NIPS 2014 Workshop)*, 2014.

[9] J. George and A. Saha, "End-to-end machine learning using kubeflow," in *5th Joint International Conference on Data Science; Management of Data*. Association for Computing Machinery, 2022, pp. 336–338, event-place: Bangalore, India. [Online]. Available: https://doi.org/10.1145/3493700.3493768

[10] K. K. Gupt, M. A. Raja, A. Murphy, A. Youssef, and C. Ryan, "GELAB – the cutting edge of grammatical evolution," *IEEE Access*, vol. 10, pp. 38 694–38 708, 2022.

[11] Z. Sun, L. Sandoval, R. Crystal-Ornelas, S. M. Mousavi, J. Wang, C. Lin, N. Cristea, D. Tong, W. H. Carande, X. Ma, Y. Rao, J. A. Bednar, A. Tan, J. Wang, S. Purushotham, T. E. Gill, J. Chastang, D. Howard, B. Holt, C. Gangodagamage, P. Zhao, P. Rivas, Z. Chester, J. Orduz, and A. John, "A review of earth artificial intelligence," *Computers and Geosciences*, vol. 159, 2022.

[12] T. D. Akinosho, L. O. Oyedele, M. Bilal, A. Y. Barrera-Animas, A.-Q. Gbadamosi, and O. A. Olawale, "A scalable deep learning system for monitoring and forecasting pollutant concentration levels on UK highways," *Ecological Informatics*, vol. 69, 2022.

[13] D. De Silva and D. Alahakoon, "An artificial intelligence life cycle: From conception to production," *Patterns*, p. 100489, 2022.

[14] G. Symeonidis, E. Nerantzis, A. Kazakis, and G. A. Papakostas, "MLOps - definitions, tools and challenges," in *IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, 2022, pp. 0453–0460.

[15] N. Gift and A. Deza, *Practical Mlops: Operationalizing Machine Learning Models*. O'Reilly Media, Inc, USA, 2021.

[16] S. Alla and S. K. Adari, *Beginning MLOps with MLFlow: Deploy Models in AWS SageMaker, Google Cloud, and Microsoft Azure*, 1st ed. Apress, 2020.

[17] V. Lakshmanan, S. Robinson, and M. Munn, *Machine Learning Design Patterns: Solutions to Common Challenges in Data Preparation, Model Building, and MLOps*. O'Reilly UK Ltd., 2020.

[18] D. Kreuzberger, N. Kuhl, and S. Hirschl, "Machine learning operations ¨ (MLOps): Overview, definition, and architecture," *arXiv:2205.02302 [cs]*, 2022.

[19] S. Choudhary, "Kubernetes-based architecture for an onpremises machine learning platform," 2021. [Online]. Available: https://aaltodoc.aalto.fi:443/handle/123456789/110516

[20] A. B. Kolltveit and J. Li, "Operationalizing machine learning models: a systematic literature review," in *SE4RAI '22: Proceedings of the 1st Workshop on Software Engineering for Responsible AI*. Association for Computing Machinery, 2022, pp. 1–8.

[21] A. Lima, L. Monteiro, and A. P. Furtado, "Mlops: Practices, maturity models, roles, tools, and challenges-a systematic literature review." *ICEIS (1)*, pp. 308–320, 2022.

[22] Y. Zhou, Y. Yu, and B. Ding, "Towards MLOps: A Case Study of ML Pipeline Platform," in *International Conference on Artificial Intelligence and Computer Engineering (ICAICE)*. IEEE, 2020, pp. 23–25.

[23] R. Min˜on, J. Diaz-de Arcaya, A. I. Torre-Bastida, and P. Hartlieb, ´ "Pangea: An MLOps Tool for Automatically Generating Infrastructure and Deploying Analytic Pipelines in Edge, Fog and Cloud Layers," *Sensors*, vol. 22, no. 12, p. 4425, 2022.

[24] A. H. Nia, F. J. Kaleibar, F. Feizi, F. Rahimi, and H. Kashfi, "Unlocking the Power of Data in Telecom: Building an Effective MLOps Infrastructure for Model Deployment," in *2023 7th Iranian Conference on Advances in Enterprise Architecture (ICAEA)*. IEEE, 2023, pp. 15–16.

[25] G. Zarate, R. Mi ´ n˜on, J. D ´ ´ıaz-de Arcaya, and A. I. Torre-Bastida, "K2e: Building mlops environments for governing data and models catalogues while tracking versions," in *IEEE 19th International Conference on Software Architecture Companion (ICSA-C)*. IEEE, 2022, pp. 206– 209.

[26] G. Recupito, F. Pecorelli, G. Catolino, S. Moreschini, D. Di Nucci, F. Palomba, and D. A. Tamburri, "A Multivocal Literature Review of MLOps Tools and Features," in *2022 48th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 2022, pp. 2022–02.

[27] I. Kumara, R. Arts, D. Di Nucci, W. J. Van Den Heuvel, and D. A. Tamburri, "Requirements and Reference Architecture for MLOps:Insights from Industry," *Authorea Preprints*, 2023.

[28] H. Washizaki, H. Uchida, F. Khomh, and Y.-G. Gueh ´ eneuc, "Studying ´ software engineering patterns for designing machine learning systems," in *2019 10th International Workshop on Empirical Software Engineering in Practice (IWESEP)*, 2019, pp. 49–495.

[29] R. Subramanya, P. Rais ¨ anen, S. Sierla, and V. Vyatkin, "Cloud comput-¨ ing design patterns for mlops: applications to virtual power plants," in *IECON 2023-49th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2023, pp. 01–07.

[30] B. Kitchenham, "Procedures for performing systematic reviews," *Keele, UK, Keele University*, vol. 33, pp. 1–26, 2004.

[31] L. Faubel and K. Schmid, "An mlops platform comparison," *Hildesheimer Informatik Berichte*, no. 1/2024, SSE 1/24/E, 2024.

[32] E. Raj, *Engineering MLOps: Rapidly build, test, and manage productionready machine learning life cycles at scale*. Packt Publishing, 2021.

[33] A. Choudhury, "Top 8 alternatives to apache spark." [Online]. Available: https://analyticsindiamag.com/top-8-alternatives-to-apache-spark/

[34] Y. Gavrilova, "The best open-source MLOps tools you should

[35] D. Meedeniya and H. Thennakoon, "Impact factors and best practices to improve effort estimation strategies and practices in DevOps," in

The 11th International Conference on Information Communication and Management. Association for Computing Machinery, 2021, pp. 11–17.
[36] S. Rahman and E. Kandogan, "Characterizing practices, limitations, and opportunities related to text information extraction workflows: A human-in-the-loop perspective," in *CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3491102.3502068
[37] S. Garg, P. Pundir, G. Rathee, P. Gupta, S. Garg, and S. Ahlawat, "On continuous integration / continuous delivery for automated deployment of machine learning models using MLOps," in *2021 IEEE Fourth International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, 2021, pp. 25–28.
[38] W. Wu and C. Zhang, "Towards understanding end-to-end learning in the context of data: Machine learning dancing over semirings codd's table," in *Proceedings of the Fifth Workshop on Data Management for End-To-End Machine Learning*. Association for Computing Machinery, 2021.
[39] H. Jayalath and L. Ramaswamy, "Enhancing performance of operationalized machine learning models by analyzing user feedback," in *4th International Conference on Image, Video and Signal Processing*. Association for Computing Machinery, 2022, pp. 197–203.
[40] A. Serban, K. van der Blom, H. Hoos, and J. Visser, "Adoption and effects of software engineering best practices in machine learning," in *Proceedings of the 14th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. Association for Computing Machinery, 2020.
[41] H. Cervantes and R. Kazman, *Designing Software Architectures: A Practical Approach*. Boston, MA, USA: Addison-Wesley Professional, 2016.
[42] S. Giannakopoulou, M. Karpathiotakis, B. Gaidioz, and A. Ailamaki, "CleanM: An optimizable query language for unified scale-out data cleaning," *Proc. VLDB Endow.*, vol. 10, no. 11, pp. 1466–1477, 2022.
[43] C. Poss, T. Irrenhauser, M. Prueglmeier, D. Goehring, F. Zoghlami, V. Salehi, and O. Ibragimov, "Enabling robot selective trained deep neural networks for object detection through intelligent infrastructure," in *Proceedings of the 4th International Conference on Automation, Control and Robotics Engineering*. Association for Computing Machinery, 2019.
[44] N. O. Nikitin, P. Vychuzhanin, M. Sarafanov, I. S. Polonskaia, I. Revin, I. V. Barabanova, G. Maximov, A. V. Kalyuzhnaya, and A. Boukhanovsky, "Automated evolutionary approach for the design of composite machine learning pipelines," *Future Gener. Comput. Syst.*, vol. 127, pp. 109–125, 2022, place: NLD Publisher: Elsevier Science Publishers B. V.
[45] E. Raj, D. Buffoni, M. Westerlund, and K. Ahola, "Edge MLOps: An automation framework for AIoT applications," in *2021 IEEE International Conference on Cloud Engineering (IC2E)*, 2021, pp. 191–200.
[46] H. Liu, Q. Gao, J. Li, X. Liao, H. Xiong, G. Chen, W. Wang, G. Yang, Z. Zha, D. Dong, D. Dou, and H. Xiong, "JIZHI: A fast and costeffective model-as-a-service system for web-scale online inference at baidu," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery ; Data Mining*. Association for Computing Machinery, 2021, pp. 3289–3298.
[47] A. Bose and A. Aggarwal, "MLOps – "Why is it required?" and "What it is"? - KDnuggets," 2022. [Online]. Available: https://www.kdnuggets.com/2020/12/mlops-why-required-what-is.html
[48] Z. Li, X.-Y. Liu, J. Zheng, Z. Wang, A. Walid, and J. Guo, "FinRLpodracer: high performance and scalable deep reinforcement learning for quantitative finance," in *Proceedings of the Second ACM International Conference on AI in Finance*. Association for Computing Machinery, 2021, pp. 1–9.
[49] Z. Azad, R. Sen, K. Park, and A. Joshi, "Hardware acceleration for DBMS machine learning scoring: Is it worth the overheads?" in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2021, pp. 243–253.
[50] O. E. Oluyisola, S. Bhalla, F. Sgarbossa, and J. O. Strandhagen, "Designing and developing smart production planning and control systems in the industry 4.0 era: A methodology and case study," *J. Intell. Manuf.*, vol. 33, no. 1, pp. 311–332, 2022.
[51] M. Oplenskedal, P. Herrmann, and A. Taherkordi, "DeepMatch2: A comprehensive deep learning-based approach for in-vehicle presence detection," *Information Systems*, vol. 108, p. 101927, 2022.
[52] J. N. van Rijn, B. Bischl, L. Torgo, B. Gao, V. Umaashankar, S. Fischer, P. Winter, B. Wiswedel, M. R. Berthold, and J. Vanschoren, "OpenML: A collaborative science platform," in *Advanced Information Systems Engineering*. Springer Berlin Heidelberg, 2013, vol. 7908, pp. 645–649.
[53] B. Brik, K. Boutiba, and A. Ksentini, "Deep learning for b5g open radio access network: Evolution, survey, case studies, and challenges," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 228–250, 2022.
[54] M. van der Goes, "Scaling enterprise recommender systems for decentralization," in *Fifteenth ACM Conference on Recommender Systems*. Association for Computing Machinery, 2021, pp. 592–594.
[55] D. L. Visengeriyeva, A. Kammer, I. Bar, A. Kniesz, and M. Pl ¨ od, ¨ "ml-ops.org." [Online]. Available: https://ml-ops.org/
[56] "LF AI & Data Landscape," 2024, [Online; accessed 5. Apr. 2024]. [Online]. Available: https://landscape.lfai.foundation
[57] K. S. d. Prado, "Awesome MLOps," original-date: 2020-05-25T22:53:26Z. [Online]. Available: https://github.com/kelvins/awesomemlops
[58] D. Muiruri, L. E. Lwakatare, J. K. Nurminen, and T. Mikkonen, "Practices and infrastructures for ML systems – an interview study," 2021, publisher: TechRxiv.
[59] L. Silva and F. Osorio, "Flowi: A platform for ML development and management," 2021, [Online; accessed 14. May 2024]. [Online]. Available: https://proceedings.science/wmecai/wmecai-2021/papers/flowi-aplatform-for-ml-development-and-management?lang=en
[60] L. Cardoso Silva, F. Rezende Zagatti, B. Silva Sette, L. Nildaimon dos Santos Silva, D. Lucredio, D. Furtado Silva, and H. de Medeiros Caseli, ´ "Benchmarking machine learning solutions in production," in *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2020, pp. 626–633.
[61] L. Faubel, K. Schmid, and H. Eichelberger, "MLOps Challenges in Industry 4.0," *SN Comput. Sci.*, vol. 4, no. 6, pp. 828–11, Oct. 2023.
[62] H. Washizaki, H. Uchida, F. Khomh, and Y.-G. Gueh ´ eneuc, "Machine ´ learning architecture and design patterns," *IEEE Software*, vol. 8, p. 2020, 2020.
[63] L. Faubel, T. Woudsma, L. Methnani, A. G. Ghezeljhemeidan, F. Buelow, K. Schmid, W. D. van Driel, B. Kloepper, A. Theodorou, M. Nosratinia, and M. Bang, "Towards an mlops architecture for xai in ˚ industrial applications," 2023.
[64] Valohai, "Valohai | Take ML places it's never been," Feb. 2023, [Online accessed 23. Feb. 2023]. Available: https://valohai.com

Using GPT-4 for Source Code Documentation

Magdalena Kneidinger, Markus Feneberger, Reinhold Plösch
Institute of Business Informatics - Software Engineering
Johannes Kepler University
Linz, Austria
k12019988@students.jku.at, {markus.feneberger,reinhold.ploesch}@jku.at

***Abstract*—Writing good software documentation imposes significant effort. Large Language Models (LLMs) could potentially streamline that process, though. So, the question arises whether current LLMs are able to generate valid code documentation for classes and methods on basis of the bare code. According to literature, various such models have the capability to generate documentation that is on par with or even superior to reference documentation. In our experimental study using zero-shot prompting, we found that the model GPT-4 by OpenAI leads to poor results when measuring similarity to the reference documentation on class level. Thus, GPT-4 is not yet usable for generating class documentation. On method level, however, the model achieved higher similarity ratings and can be considered applicable for the use case.**

***Index Terms*—LLM, Large Language Model, GPT-4, Software Documentation, Class Documentation, Method Documentation**

I. Introduction

The documentation of software is of crucial importance in the software development process. It contributes significantly to the comprehensibility, maintainability and further development of software projects [1]. However, the creation of high-quality documentation tends to involve considerable effort. This often means that documentation is neglected or does not achieve the desired level of quality [2]. In recent years, large language models (LLMs) have emerged as promising technologies. The application *ChatGPT*, based on OpenAI's GPT model series, in particular has attracted a lot of attention [3]. These models are able to generate human-like text and respond to requests in natural language [3]. In addition, some LLMs are able to handle source code, which is attracting increasing interest for use in software development tasks including the generation of software documentation [4]–[6].

A. Motivation & Objective

Many programming languages (or, their tooling, actually) offer a way to document code through structured comments. Examples for this include Java's *Javadoc* or JavaScript's *JSDoc* comment formats, or Python's *Docstring* system which utilises string literals. Poor documentation is common in industry and little effort is put in improving it [7]. With the current speed of developing newer LLMs, automation of code documentation might be on its way, too. This lead us to the following research question: What results does Java class and method *documentation generated by GPT-4* achieve, compared to manually written *reference documentation*?

To answer this question, we investigated GPT-4's capability to generate class and method documentation for Java code, i.e., Javadoc comments for classes and methods, in an experiment. Other large language models, including previous versions of the GPT model, have already been surveyed for this use case.

B. Related Work

Table I presents an overview of recent work on code documentation generation with LLMs. It can be seen that the automated generation of method level documentation has already been investigated rather extensively. Concerning the class level, however, scientific literature is currently lacking.

II. Research Method

To answer the asked research question, we conducted an experiment on the quality of GPT-generated source code documentation. In this section, the experiment setup is explained in detail, including a description of the used model and the used Java classes. This is followed by the created prompt designs and an overview of the applied evaluation methods.

A. Datasets

As test data, we used eleven Java classes to have the AI model generate Javadoc comments for class level, and method level, for a selection of five methods per class. This leads to a total of eleven classes and 55 methods, including seven classes from the Java Development Kit (JDK), one from the EJML[1] package and three from the JHotDraw[2] package.

B. Applied LLM

OpenAI's latest model, GPT-4, has come out in March 2023 [14]. As per Table I and how new the model still is, few research papers on it have been published yet. For that reason, we used GPT-4 in this experiment, to address this research gap and investigate the model's effectiveness in terms of code documentation. GPT-4 is able to process visual as well as textual input; it outperforms many existing LLMs in a number of NLP tasks and eclipses the vast majority of SOTA models [14].

C. Prompt Design

To test the basic performance of GPT-4, we chose a zero-shot prompting approach, consisting of two prompts: one for the method-level and one for the class-level documentation. Thus, the same prompts have been used for all test data for

[1]http://ejml.org
[2]https://www.randelshofer.ch/oop/jhotdraw/

TABLE I
PREVIOUS STUDIES ON LLM-GENERATED CODE DOCUMENTATION

Citation	Model(s)	Language(s) (Dataset(s))	Layer(s)	Evaluation
[3]	ChatGPT (GPT 3.5), compared with CodeBERT & CodeT5	Python (CSN)	Method	BLEU, METEOR, ROUGE-L
[8]	GPT-3.5, GPT-4, Bard, Llama2, Starchat	Python	Inline, Method, Package	Manual
[9]	GPT-3.5	Java (Funcom)	Method	Manual
[10]	CodeX, compared with CodeBERT variants, CodeT5, few-shot prompting focus	Various[3](CodeXGLUE)	Method	Smoothed BLEU-4
[11]	CodeX Few-shot prompting and ICL[4]focus	Java (Funcom, TLC)	Method	BLEU, ROUGE-L, METEOR, Manual
[12]	CodeBERT, GraphCodeBERT	Java, Python	-	BLEU-4
[6], [13]	CodeT5+, compared with RoBERTa, CodeBERT, UniXcoder, PLBART, CodeT5	Various[1] (CodeSearchNet)	Method	Smoothed BLEU-4

both documentation layers, to ensure comparability between test classes and methods. After a few trial runs, the prompts have been constructed as follows:

- System Prompt: The model has been assigned the role of a helpful assistant for Javadoc code summarising.
- Task: The model has been given the task with clear instructions. Three quality criteria were mentioned: Correctness, Completeness and Conciseness as well as the code level to generate documentation for.
- Datasets: The whole, uncommented Java class was given to the model.
- Output: The model was instructed to return the Java code with the newly inserted Javadoc comments.

Taking all these aspects into account, we went with these two prompts:

- Add accurate, complete and concise Javadoc-documentation to the class and all methods and fields of this java-class: {javaCode}. The output should contain the whole source code of the given java-class with all generated Javadoc-documentation.
- Add accurate, complete and concise Javadoc-documentation to the following methods of this java-class: {javaCode}. The methods are: [Names of the five selected methods]; please also consider the annotations. The output should contain the code of these java-methods with the generated javadoc-documentation.

D. Evaluation

The quality of the generated code documentation was evaluated based on different metrics. As automated metrics, Smoothed-BLEU-4 and ROUGE-1 have been used. Both metrics are commonly used for similar experiments [10], [11], [13], [15], see Table I. They both measure the similarity of two texts [16], [17]. The improved version of BLEU, *Smoothed-BLEU*, attempts to improve on some problems as the original only correlates weakly with human judgement.

In addition to BLEU and ROUGE, we performed a manual evaluation as well. Since the two metrics only evaluate the lexical discrepancy between the generated and the reference documentation, they are not sufficient to detect semantic differences [11]. Furthermore, we cannot assume that the original documentation is perfect, an aspect that has to be considered in the comparison of generated documentation. For that reason, the generated documentation was checked for correctness, completeness and conciseness, which are three common quality criteria [9]. To allow quantitative evaluation, we defined a four-part scale for each criterion:

III. RESULTS

In this section, the results of the conducted experiment are presented, grouped by documentation level and type of evaluation.

A. Class-level documentation

Figure 1 shows the results of the manual evaluation of the class documentation. The points have been averaged across all classes, by criterion. The bar chart presents the evaluation points of the reference documentation (light) and the generated documentation (dark) pairwise.

Considering **correctness**, the diagram shows that both the reference documentation and the generated documentation have reached the full three points, i.e., do not include any false information. This is not the case for **completeness**, where the generated documentation has performed significantly worse at only 1.3 pts. Overall, GPT-4 tended to generate shorter, more general descriptions that miss key elements that are included in the reference documentation. This also leads to GPT-4 almost matching the reference documentation in **conciseness**, though.

Applying the metrics ROUGE-1 and Smoothed-BLEU-4 showed that GPT-4 did not generate class documentation that is similar to the reference documentation:

- ROUGE-1: 17 %
- Smoothed-BLEU-4: 3.38 %

The above-mentioned shorter comments from GPT are partly responsible for this. On five classes (i.e., almost half of them), the BLEU result was less than 0.1 per cent. The highest BLEU score was achieved on `java.lang.Integer` at 16.6 per cent. While being generally higher, the ROUGE score correlated strongly (Pearson correlation coefficient of 0.9433). The

[3]Six languages, including Java and Python
[4]In-context Learning

TABLE II
CRITERIA OF THE MANUAL EVALUATION

Criterion	Points	Definition
Correctness	0	The documentation is incorrect as a whole or missing entirely
	1	The documentation contains more than one piece of false information. What is false?
	2	The documentation contains one piece of false information. What is false?
	3	The documentation is correct.
Completeness	0	The documentation is missing entirely.
	1	The documentation lacks multiple important pieces of information. What is missing?
	2	The documentation is missing one piece of important information. What is missing?
	3	The documentation is complete.
Conciseness	0	The documentation only contains unnecessary information or is missing entirely.
	1	The documentation contains more than one unnecessary piece of information. What is unnecessary?
	2	The documentation contains one unnecessary piece of information. What is unnecessary?
	3	The documentation is concise and hence contains no unnecessary information.

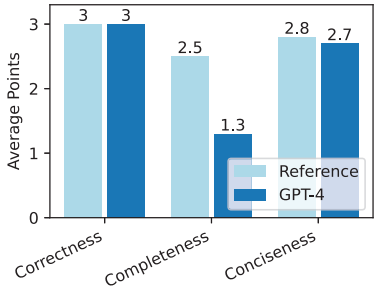


Fig. 1. Average Results on Class level

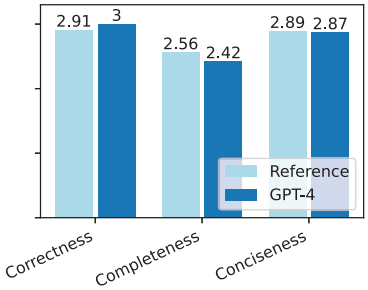


Fig. 2. Average Results on Method level

correlation of both scores with the manual evaluation is much weaker, as to be expected from the raw data: ROUGE and Manual correlate at 0.0953, BLEU and Manual at 0.1592 per cent. Both indicate weak, but not reliable, positive correlation [see 18].

B. Method-level documentation

The results of the manual evaluation of the method documentation are shown in Figure 2, in the same way as the class results in Figure 1. It can be seen that, unlike at the class level, GPT-4 (3 pts.) has managed to surpass the reference documentation (2.91 pts.) quality in correctness. Also in the two other criteria, the model was able to almost match it. This means that GPT-4 produced (significantly) more complete method documentation than class documentation. On several methods, GPT-4 has also outperformed the reference documentation in completeness.

Bearing the previously learned correlation in mind, the generated documentation can also be expected to be more similar (i.e., higher BLEU and ROUGE scores) to the reference documentation. This is indeed the case:

- ROUGE-1: 42.47 %
- Smoothed-BLEU-4: 18.38 %

The Pearson correlation coefficient was also calculated for the method data:

- ROUGE with BLEU: 0.9112
- ROUGE with Manual: 0.1938
- BLEU with Manual: 0.2075

So, both similarity metrics strongly correlate among the method documentation as well. The correlation of each metric with the manual evaluation results is stronger than at the class level but still quite weak. Thus, a higher similarity indicates a slightly higher probability that a generated Javadoc method comment is correct, complete and concise. The major *threat to validity* is besides the limited number of investigated documentation the fact that the quality of the reference documentation and of the generated documentation was judged by two experts, only. Nevertheless, we think that only using similarity metrics is not convincing, but needs expert judgement from point of view of software developers.

IV. CONCLUSIONS

To evaluate whether the large language model GPT-4 is capable of generating Javadoc comments for classes and methods, we conducted a two-part experiment.

At class level, both the generated and the reference documentation received maximum scores in terms of accuracy. However, the generated documentation showed significant deficits in completeness compared to the reference documentation, as they were shorter and less detailed and often lacked important details and standard Javadoc tags. In terms of conciseness, the generated and reference documentation scored similarly. In general, the generated documentation could not match the quality of the reference documentation. On two of the eleven classes, however, it could be considered on par.

At the method level, GPT-4 performed better than the reference documentation in terms of correctness, as GPT-4 did not generate any incorrect information, as was the case at the class level. However, the generated documentation had a slightly lower completeness score, particularly due to missing '`@see`' tags and references to special cases or IEEE standards. Conversely, the reference documentation often lacked descriptions of return and parameter values. The ratings for the conciseness of the method documentation were almost identical, which indicates that both the generated and the reference documentation largely contained only relevant information. The aggregated results across all classes show that in most cases the reference documentations scored slightly higher than the generated documentations, but without significant differences, indicating an effective performance of GPT-4.

Since there is no evaluation of class-level documentation in literature yet, we could not make direct comparisons. Nevertheless, higher-level (such as class-level) documentation imposes a challenge for large language models [19] in general. Concerning the method level, though, our results fit with previous insights in that the generated documentation almost matches the reference documentation in quality. Other studies have even found GPT-4 producing *better* method documentation than the reference documentation [8], but we do not know about the quality of the latter. GPT mostly relying on the control flow instead of method and variable names [3] is a conclusion that cannot be drawn from our results [see 3], [12]. We currently conduct a study on the migration of a larger hardly documented software system where we will use GPT-4 for generating documentation - in an attempt to have a better input basis for code transformation, unit test enhancements, etc.

References

[1] A. Y. Wang, D. Wang, J. Drozdal, *et al.*, "Documentation Matters: Human-Centered AI System to Assist Data Science Code Documentation in Computational Notebooks," en, *ACM Transactions on Computer-Human Interaction*, vol. 29, no. 2, pp. 1–33, Apr. 2022, ISSN: 1073-0516, 1557-7325. DOI: 10.1145/3489465.

[2] R. S. Geiger, N. Varoquaux, C. Mazel-Cabasse, and C. Holdgraf, "The Types, Roles, and Practices of Documentation in Data Analytics Open Source Software Libraries: A Collaborative Ethnography of Documentation Work," en, *Computer Supported Cooperative Work (CSCW)*, vol. 27, no. 3-6, pp. 767–802, Dec. 2018, ISSN: 0925-9724, 1573-7551. DOI: 10.1007/s10606-018-9333-1.

[3] W. Sun, C. Fang, Y. You, *et al.*, *Automatic Code Summarization via ChatGPT: How Far Are We?* arXiv:2305.12865 [cs], May 2023. DOI: 10.48550/arXiv.2305.12865.

[4] J. Cao, M. Li, M. Wen, and S.-c. Cheung, "A study on Prompt Design, Advantages and Limitations of ChatGPT for Deep Learning Program Repair," 2023. DOI: 10.48550/ARXIV.2304.08191.

[5] Z. Feng, D. Guo, D. Tang, *et al.*, "CodeBERT: A Pre-Trained Model for Programming and Natural Languages," en, in *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online: Association for Computational Linguistics, 2020, pp. 1536–1547. DOI: 10.18653/v1/2020.findings-emnlp.139.

[6] Y. Wang, H. Le, A. D. Gotmare, N. D. Q. Bui, J. Li, and S. C. H. Hoi, *CodeT5+: Open Code Large Language Models for Code Understanding and Generation*, arXiv:2305.07922 [cs], May 2023.

[7] M. Kajko-Mattsson, "A Survey of Documentation Practice within Corrective Maintenance," en, *Empirical Software Engineering*, vol. 10, no. 1, pp. 31–55, Jan. 2005, ISSN: 1382-3256. DOI: 10.1023/B:LIDA.0000048322.42751.ca.

[8] S. S. Dvivedi, V. Vijay, S. L. R. Pujari, S. Lodh, and D. Kumar, *A Comparative Analysis of Large Language Models for Code Documentation Generation*, arXiv:2312.10349 [cs], Dec. 2023.

[9] C.-Y. Su and C. McMillan, *Distilled GPT for Source Code Summarization*, arXiv:2308.14731 [cs], Aug. 2023. DOI: 10.48550/arXiv.2308.14731.

[10] T. Ahmed and P. Devanbu, "Few-shot training LLMs for project-specific code-summarization," en, in *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, Rochester MI USA: ACM, Oct. 2022, pp. 1–5, ISBN: 978-1-4503-9475-8. DOI: 10.1145/3551349.3559555.

[11] M. Geng, S. Wang, D. Dong, *et al.*, "Large Language Models are Few-Shot Summarizers: Multi-Intent Comment Generation via In-Context Learning," 2023. DOI: 10.48550/ARXIV.2304.11384.

[12] A. H. Mohammadkhani, C. Tantithamthavorn, and H. Hemmati, *Explainable AI for Pre-Trained Code Models: What Do They Learn? When They Do Not Work?* arXiv:2211.12821 [cs], Aug. 2023.

[13] Y. Wang, W. Wang, S. Joty, and S. C. Hoi, "CodeT5: Identifier-aware Unified Pre-trained Encoder-Decoder Models for Code Understanding and Generation," en, in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021, pp. 8696–8708. DOI: 10.18653/v1/2021.emnlp-main.685.

[14] OpenAI, J. Achiam, S. Adler, *et al.*, *GPT-4 Technical Report*, arXiv:2303.08774 [cs], Mar. 2024.

[15] T. Kajiura, N. Souma, M. Sato, M. Takahashi, and K. Kuramitsu, "An additional approach to pre-trained code model with multilingual natural languages," in *2022 29th Asia-Pacific Software Engineering Conference (APSEC)*, ISSN: 2640-0715, Dec. 2022, pp. 580–581. DOI: 10.1109/APSEC57359.2022.00090.

[16] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: A method for automatic evaluation of machine translation," en, in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*, Philadelphia, Pennsylvania: Association for Computational Linguistics, 2001, p. 311. DOI: 10.3115/1073083.1073135.

[17] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *Text Summarization Branches Out*, Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 74–81.

[18] E. Reiter, "A Structured Review of the Validity of BLEU," en, *Computational Linguistics*, vol. 44, no. 3, pp. 393–401, Sep. 2018, ISSN: 0891-2017, 1530-9312. DOI: 10.1162/coli_a_00322.

[19] S. A. Rukmono, L. Ochoa, and M. R. Chaudron, "Achieving High-Level Software Component Summarization via Hierarchical Chain-of-Thought Prompting and Static Code Analysis," in *2023 IEEE International Conference on Data and Software Engineering (ICoDSE)*, Toba, Indonesia: IEEE, Sep. 2023, pp. 7–12, ISBN: 9798350381382. DOI: 10.1109/ICoDSE59534.2023.10292037.

Author Index

Supported by:

ISSN 2980-7298