# PROCEEDINGS OF THE 32ND MINISYMPOSIUM

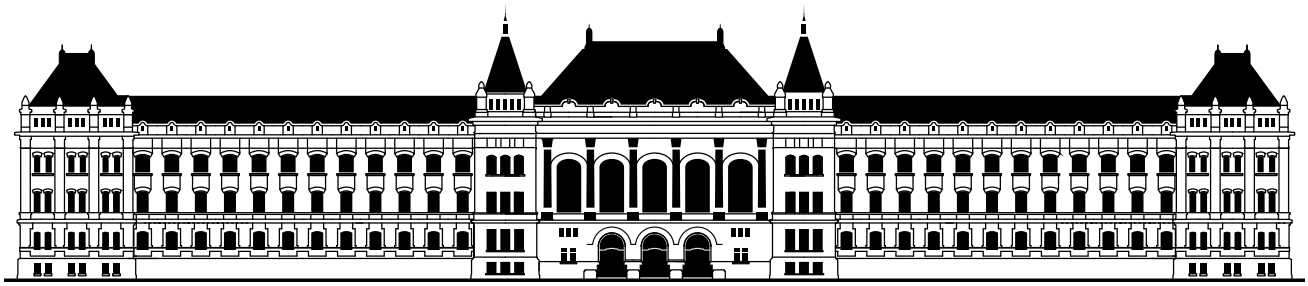
OF THE

## DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND SYSTEMS ENGINEERING BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS

## (MINISY@DMIS 2025)

## FEBRUARY 3–4, 2025

## BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS



BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
DEPARTMENT OF ARTIFICIAL INTELLIGENCE
AND SYSTEMS ENGINEERING

ISBN 978-963-421-989-7

Head of Department:
László Gönczy

General Chair:
Balázs Renczes

Local Chairs[1]:
Zsófia Ádám
Milán Mondok
Bertalan Zoltán Péter
Domonkos Pogány
Dániel Sándor

Homepage of the Conference:
`http://minisy.mit.bme.hu/`

Sponsored by:
Schnell László Foundation

[1]This list contains all the Reviewers who participated in the review process of papers published in this proceedings:
*Péter Antal, Bence Bruncsics, Bence Cseppentő, András Gézsi, László Gönczy, Bence Graics, Dániel Hadházi, Gábor Hullám, Attila Klenik, Zsolt Kollár, István Majzik, Kristóf Marussy, Tamás Mészáros, András Millinghoffer, Zoltán Micskei, Milán Mondok, György Orosz, András Palkó, Béla Pataki, András Pataricza, Balázs Renczes, Oszkár Semeráth, András Vörös*

**Foreword**

On behalf of the Organizing Committee, I am pleased to welcome you to the 32nd Minisymposium of the Department of Artificial Intelligence and Systems Engineering at the Budapest University of Technology and Economics. This year, we are honored to once again host the Minisymposium as part of VIK Inference, the official conference of the Faculty of Electrical Engineering and Informatics.
In recent years, we have seen a significant increase in the number of PhD students within our department. As a result, this year's event will span two full days, featuring over thirty presentations.
The symposium program reflects the breadth and depth of research conducted within the department, encompassing topics such as Digital Signal Processing, Embedded Systems, Artificial Intelligence, and Fault-Tolerant Systems.
We hope that the 32nd Minisymposium serves as a valuable platform for knowledge exchange, fostering new research discussions and collaborations among participants.
I wish you an engaging and inspiring symposium!

Budapest, February 3, 2025

Balázs Renczes
General Chair

**Contents**

# Automatic pulmonary vessel network labeling on thoracic CT scans based on partially labeled data

Gábor Révy, Gábor Hullám, Dániel Hadházi
Budapest University of Technology and Economics,
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {revy, gabor.hullam, hadhazi}@mit.bme.hu

***Abstract*—Accurate pulmonary vessel segmentation, and the separation into arterial and venous networks helps doctors by guiding them during surgical planning, particularly in thoracic procedures. Despite significant advancements, the automation of this task is still currently in progress. Many studies show promising results, yet they often rely on high-quality CT scans, which are not always available in routine clinical practice. Specifically, these algorithms typically depend on contrast-enhanced CT scans with thin slice spacing, precise timing of contrast injection, and the application of filtering algorithms during reconstruction to minimize streaking artifacts.**

**In our previous work we presented a set of algorithms to improve an existing system for artery-vein separation, with the aim of making it more robust on CT scans of typical quality. Despite evaluating a wide variety of algorithms, it became clear that an explicit, model-based approach was necessary to achieve the desired accuracy and reliability in artery-vein separation. However, developing such a model requires a substantial amount of properly labeled data, which is difficult to obtain due to the time-consuming nature of manual labeling.**

**We investigated the dataset of the PARSE2022 (Pulmonary Artery Segmentation) challenge with incomplete annotations: only arteries are segmented on the CT scans. This poses a considerable obstacle for training a robust model. To address this, we developed an algorithmic solution based on bilateral filtering to automatically complete the segmentation with the venous labels, ensuring that the dataset was sufficiently annotated for model training.**

***Index Terms*—image processing, CT, vessel segmentation, pulmonary vessel separation.**

## I. Introduction

Along with CADe (computer-aided detection) and CADx (computer-aided diagnosis) systems, surgical planning systems have been important research areas in recent decades. They often require the incorporation of the former two in order to automate as many steps as possible. This usually requires researchers to decide between two primary approaches. They can (1) either design algorithms based on area-specific domain knowledge or (2) develop models based on labeled data. The former one - called expert algorithms - can be useful when there is only a little amount of labeled data or no labeled data is available. The latter approach is usually applied if properly labeled data of sufficient size are available.

This research was funded by the Josef Heim Medical Innovation Scholarship (Josef Heim Alkotói Ösztöndíj).

Pulmonary vessel segmentation in the context of CT scans refers to the process of deciding for each voxel whether or not it belongs to the vascular network in the lung. The next level is the separation into artery and vein (A/V) classes. This means that the segmented vessels must be assigned to the arterial or venous vascular network. There are several challenges regarding the two tasks. In many cases, contrast agent is not used in clinical practice. This makes the boundary between the vessels hard to detect, especially vessels entering the lungs, close to the hilar region. Even if contrast agent is used during the scanning process, if the filtering algorithms are not (properly) applied, streaking artifacts may appear on the scans, ruining density values of the vessels, creating ”artificial boundaries” between them. Slice spacing is also an important factor influencing the accuracy of the algorithms. Many algorithms build on the ”traceability” of the vessels. This means that the segments of the same vessel on multiple slices are connected in 3D and segments of different vessels are not connected. CT scans with larger slice spacing (above $\sim 2$ mm) tend not to meet this condition in cases where two vessels are running close to each other in the superior-inferior direction. This results in a segmentation where the voxels of two separate vessels can be found in the same 3D connected component. Algorithms that build on the ”traceability” feature mislabel vessels by leaking the segmentation of one class (A/V) into the other type of vessel during the separation. For example, marching algorithms that march from voxel to voxel and spread a seed label might step from one type of vessel to the other one. Therefore, without incorporating other information, the vessels will be mislabeled.

In our previous work [1] we presented extensions of expert algorithms which aimed to make them more robust. Our extensions mainly built on a semiautomatic, fuzzy distance transform-based [2] iterative algorithm designed by Saha *et al.* [3] called multiscale topomorphologic opening. Although the algorithm was designed to mitigate the impact of the partial volume effect and performs this function fairly well, there are specific locations where the algorithm seems to be missing a priori information that is hard to encode into such an algorithm. A typical mistake made by the algorithm is the leakage of labeling between the right pulmonary artery and the right superior pulmonary vein. Figure 1 shows an example of this location. In the case of non-contrast CT scans, the border

DOI https://doi.org/10.3311/MINISY2025-001

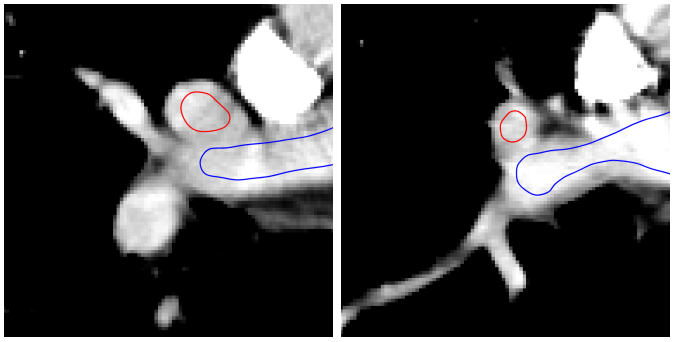


Fig. 1. A location where it is hard for the algorithms to differentiate between A/V due to the lack of countour between them: the right pulmonary artery (blue) and the right superior pulmonary vein (red).

between vessels running close to each other is not consistently visible, therefore a marching algorithm is prone to creating a leakage at that point. Since this location is not deep inside the lungs and the marching algorithm progresses from the heart toward the alveoli inside the lungs, this results in mislabeling several vessel branches.

Neural networks are known for having good performance in pattern recognition problems [14]. Currently, we do not know of any type of neural network that would be capable of learning A/V separation in an end-to-end manner, architecturally enforcing the continuity of the blood vessel segmentation, therefore this approach would not be realistic. Our aim is to address this problem by a model-based deep learning approach [15] where the local patterns are detected by a neural network, and this is complemented by a conventional expert algorithm.

In order to train a neural network, a sufficient amount of properly labeled data is required. Although this is a problem studied by many researchers [5], [16]–[18], no fully publicly available dataset suitable for the task is currently available. The main reason for this is that A/V labeling of a CT scan is a time-consuming task, and the resulting dataset is of great value.

Table I summarizes possibly relevant datasets based on our literature research. Datasets DS8 (HiPaS [11]) and DS9 ( [12]) are not currently available and despite extensive searches, the DS5 (ISICDM2021 [8]) could not be located from any accessible or public repositories. Datasets DS1 (VESSEL12 [4]), DS3 (APCTP [6]) and DS7 (MICCAI'23 PTR [10]) are not applicable for the reasons detailed in the table. In DS2 (CARVE14 [5]) 10 CT scans are fully annotated, however, they are only segmented inside the lungs, thus part of the entry points of the vessels is not labeled. DS4 (CHD68 [7]) might be utilized later to make the separation algorithm more robust. Its scans, however, were made using a short exposure time, multi-row CT scanner without the use of $\beta$-blockers, and it is likely that a low-pass filtering kernel was applied in filtered backprojection during the reconstruction. This resulted in a scan with significantly reduced perceived in-plane resolution. DS6 (PARSE2022 [9]) showed significant potential, however, only the arteries are segmented in the scans, with the veins not being included. However, the PARSE2022 dataset is the best option because of its size, quality, and level of appropriateness. This required the creation of the venous vascular network's segmentation, ideally as automatically as possible.

## II. Method

In this section, the steps leading to the creation of the venous segmentation are introduced. The process consists of 3 main steps. First, (1) a vessel segmentation is created, then (2) a basic vein segmentation is created based on the difference of the arterial and the full vessel segmentation. Finally, (3) the vein segmentation is refined.

### A. Vessel segmentation

Vessel segmentation includes the segmentation of the vessels inside the hilum and the lungs.

Vessel segmentation inside the hilum is determined by thresholding the HU values. First, the hilar area is determined based on the lung segmentation. Then the threshold value is calculated based on the percentile of the HU values.

The vessel segmentation inside the lungs is created by combining the vessel segmentation of the TotalSegmentator [19] tool and a Frangi-like vesselness filter.

Our Frangi-like filter calculates a vesselness score according to the equation:

$$I^{vesselness}(x) = \sqrt{\max(-\lambda_0(x), 0) \cdot \max(-\lambda_1(x), 0)} \quad (1)$$

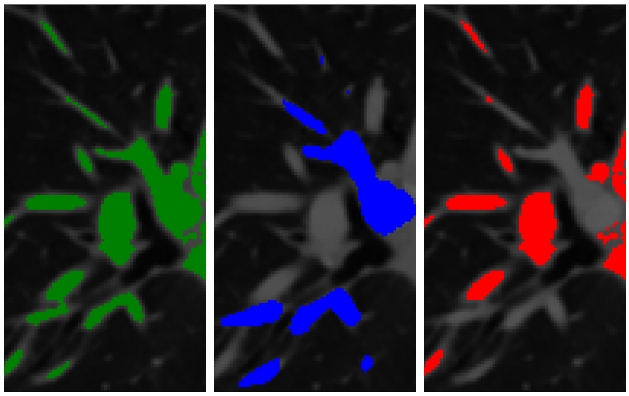
Here, $\lambda_0(x)$ and $\lambda_1(x)$ denote the largest and the second largest absolute value of eigenvalues of the Hessian matrix after applying Gaussian filtering. The vessel segmentation is created by thresholding the vesselness score. Based on our experiments, the filter is able to assign a nonzero score to the vessel segments; however, it has two main shortcomings. Since only one scale is used (1) the small vessels are oversegmented. This is solved by applying guided filtering [20] using the mask as input and the CT scan as a guidance image. We also found that near the branching points, the second largest absolute value eigenvalue is positive, which resulted in some parts of the network not being enhanced. This is addressed by connected component analysis using another vessel mask created by TotalSegmentator.

TotalSegmentator is a nnU-Net [21] (3D U-Net) based automatic segmentation tool for medical (CT and MR) images. While investigating its output we found that (1) its vessel segmentation is strictly restricted to the lungs, missing the large vessels entering the lungs, and (2) sometimes misses a few clearly visible vessels.

By combining our Frangi-like filtering method with the one created by TotalSegmentator a robust vessel segmentation approach is achieved.

TABLE I
DATA SETS IDENTIFIED THROUGH A LITERATURE SEARCH THAT ARE RELEVANT TO THE TASK. URLS ACCESSED: NOVEMBER 26, 2024

| DS id | name | cite | availability | comment |
|---|---|---|---|---|
| 1 | VESSEL12 | [4] | grand-challenge<br>zenodo | Contains vessel segmentation only without A/V differentiation. |
| 2 | CARVE14 | [5] | grand-challenge<br>zenodo | Vessels are only segmented inside the lungs (no entry).<br>Contains 55 non-contrast scans from which 10 are fully annotated.<br>For the remaining scans, annotations are provided on 50-50 randomly selected vessels. |
| 3 | APCTP | [6] | zenodo | Synthetic dataset. |
| 4 | CHD68 | [7] | kaggle | 68 CT images showing different congenital heart diseases. |
| 5 | ISICDM2021 | [8] | N/A | Not available. |
| 6 | PARSE2022 | [9] | grand-challenge | 100 training, 70 test and 30 validation cases.<br>Contains artery segmentation only (no vein segmentation). |
| 7 | MICCAI'23 PTR | [10] | github | A/V segmentations only, the corresponding CT scans<br>and pixel spacing values are not available. |
| 8 | HiPaS | [11] | N/A | 315 CTPA volumes and 758 non-contrast CT volumes with A/V segmentation.<br>Currently being commercialized, not yet published,<br>but the authors plan to publish the datasets and pre-trained models<br>once the paper is accepted. |
| 9 | (*LIDC A/V labeled*) | [12] | N/A | 120 LIDC [13] CT scans labeled. Their method is in the process of commercialization.<br>The relevant code and data is not available to the public. |



(a) vessel (b) artery (c) vein

Fig. 2. Automatically generated vessel segmentation, manual artery segmentation and the baseline vein segmentation as the difference between the two.

### *B. Baseline vein segmentation*

A baseline vein segmentation is created by subtracting the artery segmentation from the vessel segmentation. Since the manual labeling is a result of an approach unknown to us, this difference-based segmentation does not perfectly equal to the true vein segmentation. Some of the errors are difficult to correct algorithmically. Specifically, certain vessels are segmented as veins; however, in the context of the separation task, they are not referred to as arteries or veins. This includes vessels such as the ascending aorta and the superior vena cava.

### *C. Bilateral filtering-based refinement*

There is a type of error that was alleviated using bilateral filtering. Due to the different approaches of the manual and the automatic segmentation, their borders do not align. As a result, the difference-based segmentation may erroneously classify portions of arteries as veins, particularly along the borders and different types of vessels running close to each other.

The bilateral filter is defined as:

$$I^{\text{filtered}}(x) = \frac{1}{W(x)} \sum_{x_i \in \Omega_x} I(x_i) f_r(\|G(x_i) - G(x)\|) g_s(\|x_i - x\|) \tag{2}$$

Here, $I$ denotes the input image: the baseline A/V segmentation encoded as $-1, 0$ and $1$ for artery, background, and vein, respectively. $G$ is the input (CT) image, $\Omega_x$ is a window centered around $x$. $f_r$ and $g_s$ are the range and spatial kernels assigning weights to the distance between voxels in the spatial domain and the intensity differences in the guidance image, respectively, determining how much each neighboring pixel contributes to the filtered result based on its proximity and similarity to the center voxel:

$$f_r(x) = exp(-\frac{x^2}{2\sigma_r^2}) \quad \text{and} \quad g_s(x) = exp(-\frac{x^2}{2\sigma_s^2}) \tag{3}$$

where $\sigma_r$ and $\sigma_s$ control the decay of the weighting functions. $W(x)$ is a normalization factor ensuring that the weights sum to 1:

$$W(x) = \sum_{x_i \in \Omega_x} f_r(\|G(x_i) - G(x)\|) g_s(\|x_i - x\|) \tag{4}$$

Figure 3 shows an example for the effect of bilateral filtering. As can be seen in the image, filtering has two visible effects: (1) the two segmentations are better separated, and (2) the segmentations are more closely aligned with the density values.

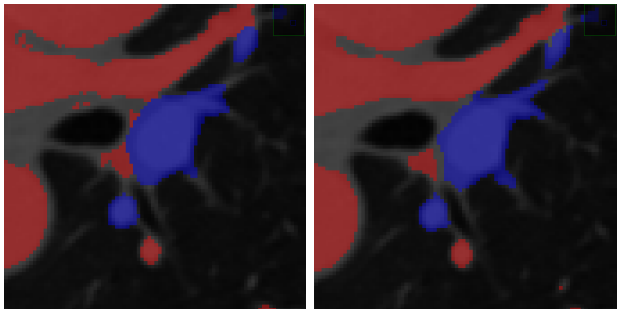


Fig. 3. Artery/vein segmentation (blue/red) before and after applying bilateral filtering.

## III. Assessment

Since the goal of our work is to create a ground truth for other algorithms, no quantitative evaluation can be performed. The results of the procedure have been visually examined. It can be argued that the labeling is sufficient to create a basic segmentation model: the segmentation of vessels entering the lungs, near the hilum is - apart from minor errors - sufficient enough to train a neural network. Minor errors include the following. (1) The soft tissue around the vessels might be segmented due to its radiodensity being similar to that of the vessel in particular contrast phases. This is the most common error of the algorithm (multiple patches per CT scan). (2) In some cases, the tracheal wall is included in the segmentation. (3) Finally, thin connections between vessel components might get suppressed by the bilateral filtering, resulting in multiple components. This can be addressed in the future by connected component analysis and incorporating the direction of the vessel.

## References


[1] G. Révy and A. Bodnár, "Towards pulmonary vessel separation," in *Proceedings of the 31st Minisymposium*. Budapest University of Technology and Economics, 2024, p. 13–18. [Online]. Available: http://dx.doi.org/10.3311/MINISY2024-003

[2] P. K. Saha, F. W. Wehrli, and B. R. Gomberg, "Fuzzy distance transform: theory, algorithms, and applications," *Computer Vision and Image Understanding*, vol. 86, no. 3, pp. 171–190, 2002.

[3] P. K. Saha, Z. Gao, S. K. Alford, M. Sonka, and E. A. Hoffman, "Topomorphologic separation of fused isointensity objects via multiscale opening: Separating arteries and veins in 3-d pulmonary ct," *IEEE transactions on medical imaging*, vol. 29, no. 3, pp. 840–851, 2010.

[4] R. D. Rudyanto, S. Kerkstra, E. M. Van Rikxoort, C. Fetita, P.-Y. Brillet, C. Lefevre, W. Xue, X. Zhu, J. Liang, I. Öksüz *et al.*, "Comparing algorithms for automated vessel segmentation in computed tomography scans of the lung: the vessel12 study," *Medical image analysis*, vol. 18, no. 7, pp. 1217–1232, 2014.

[7] X. Xu, T. Wang, Y. Shi, H. Yuan, Q. Jia, M. Huang, and J. Zhuang, "Whole heart and great vessel segmentation in congenital heart disease using deep neural networks and graph matching," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part II 22*. Springer, 2019, pp. 477–485.

[8] W. Tan, L. Zhou, X. Li, X. Yang, Y. Chen, and J. Yang, "Automated vessel segmentation in lung ct and cta images via deep neural networks," *Journal of X-ray science and technology*, vol. 29, no. 6, pp. 1123–1137, 2021.

[9] G. Luo, K. Wang, J. Liu, S. Li, X. Liang, X. Li, S. Gan, W. Wang, S. Dong, W. Wang *et al.*, "Efficient automatic segmentation for multi-level pulmonary arteries: The parse challenge," *arXiv preprint arXiv:2304.03708*, 2023.

[10] Z. Weng, J. Yang, D. Liu, and W. Cai, "Topology repairing of disconnected pulmonary airways and vessels: Baselines and a dataset," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2023, pp. 382–392.

[11] Y. Chu, G. Luo, L. Zhou, S. Cao, G. Ma, X. Meng, J. Zhou, C. Yang, D. Xie, R. Henao *et al.*, "Deep learning-driven pulmonary arteries and veins segmentation reveals demography-associated pulmonary vasculature anatomy," *arXiv preprint arXiv:2404.07671*, 2024.

[12] J. Pu, J. K. Leader, J. Sechrist, C. A. Beeche, J. P. Singh, I. K. Ocak, and M. G. Risbano, "Automated identification of pulmonary arteries and veins depicted in non-contrast chest ct scans," *Medical image analysis*, vol. 77, p. 102367, 2022.

[13] S. Armato III, G. McLennan, L. Bidaut, M. McNitt-Gray, C. Meyer, A. Reeves, B. Zhao, D. Aberle, C. Henschke, E. Hoffman *et al.*, "Data from lidc-idri [data set]. the cancer imaging archive," 2015.

[14] J. Wu, "Convolutional neural networks," in *Essentials of Pattern Recognition*. Cambridge University Press, Nov. 2020, pp. 333–364.

[15] N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis, "Model-based deep learning," *Proceedings of the IEEE*, vol. 111, no. 5, pp. 465–499, 2023.

[16] C. Payer, "Separation of arteries and veins in pulmonary ct images," Master's thesis, Graz University of Technology, 2015.

[17] P. Nardelli, D. Jimenez-Carretero, D. Bermejo-Pelaez, G. R. Washko, F. N. Rahaghi, M. J. Ledesma-Carbayo, and R. S. J. Estépar, "Pulmonary artery–vein classification in ct images using deep learning," *IEEE transactions on medical imaging*, vol. 37, no. 11, pp. 2428–2440, 2018.

[18] Z. Zhai, M. Staring, X. Zhou, Q. Xie, X. Xiao, M. Els Bakker, L. J. Kroft, B. P. Lelieveldt, G. J. Boon, F. A. Klok *et al.*, "Linking convolutional neural networks with graph convolutional networks: application in pulmonary artery-vein separation," in *Graph Learning in Medical Imaging: First International Workshop, GLMI 2019, Held in Conjunction with MICCAI 2019, Shenzhen, China, October 17, 2019, Proceedings 1*. Springer, 2019, pp. 36–43.

[19] J. Wasserthal, H.-C. Breit, M. T. Meyer, M. Pradella, D. Hinck, A. W. Sauter, T. Heye, D. T. Boll, J. Cyriac, S. Yang *et al.*, "Totalsegmentator: robust segmentation of 104 anatomic structures in ct images," *Radiology: Artificial Intelligence*, vol. 5, no. 5, 2023.

[20] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 6, pp. 1397–1409, 2012.

[21] F. Isensee, T. Wald, C. Ulrich, M. Baumgartner, S. Roy, K. Maier-Hein, and P. F. Jaeger, "nnu-net revisited: A call for rigorous validation in 3d medical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2024, pp. 488–498.

# Giving Some Pointers for Abstraction-Based Model Checking

Levente Bajczi, Dániel Szekeres, Csanád Telbisz, Vince Molnár
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {bajczi, szekeres}@mit.bme.hu, csanadtelbisz@edu.bme.hu, molnarv@mit.bme.hu

***Abstract*—Abstraction-based software model checkers often rely on external analyses or unbounded SMT arrays to reason about pointers, arrays, and dynamic heap manipulation. External analyses are precise but often require the modification of existing verification algorithms, while SMT arrays provide a native solution for solver-based verifiers but require strict type safety often forgone in real-world programs. We propose a novel way of integrating a precise pointer and array analysis as a plug-in for abstraction-based model checking, which does not require the modification of the underlying algorithms. Our solution keeps track of arbitrary predicates over potentially abstract memory locations, moving toward more efficient verification of software code by allowing a fine-grained and precise abstraction of memory accesses.**

## I. Introduction

Verifying software remains a complex yet necessary task in modern systems engineering. One source of complexity comes from the use of dynamic data structures such as heap-allocated arrays. , and their formal analysis remains a vital part of practical software verification, especially as more and more applications rely on formal methods for safety guarantees. In this paper, we refer to Satisfiability Modulo Theories (SMT) solver-aided verification [1] as *software model checking*.

State-of-the-art software model checking tools generally fall into two categories in terms of heap analysis. They either encode the heap as unbounded SMT arrays [2] (which can be done either for each array individually, or monolithically per type [3]), or use an auxiliary analysis such as symbolic memory graphs (SMGs) [4] to keep track of pointer (and thus, array) information. However, for *abstraction-based* analyses [1], both solutions are problematic. Auxiliary analyses often result in both heightened complexity and a performance penalty, monolithic heap encoding as a single SMT array per type is also often costly as the solver needs to reason about the heap as a whole, while direct SMT-encoding of individual arrays often lacks support for the type conversions necessary for verifying real-world programs.

Consider for example a C function that takes a `void*` pointer, and depending on external factors, dereferences it as either an `int` or a `float`. Given their sizes are equivalent (e.g., in the `ILP32` architecture[1]), this is valid in C, and sometimes, such as when passing an argument to a new thread via `pthread_start`, even necessary. An SMT encoding, however, cannot implement this without somehow relying on a different array expression for each type. Therefore, every pointer used as an array has to have multiple encodings, which must also be kept in sync when updated.

We propose a solution that integrates directly into existing Counterexample-Guided Abstraction Refinement (CEGAR) [5] frameworks (sparing the complexity of standalone analyses), while also providing an easy and performant way of relying on heap information in the abstract state space. We also extend this approach to handle stack-based arrays, structs, and references to variables. We also discuss how our approach can be integrated with analyses relying on a dependency relation among transitions (e.g., partial order reduction).

## II. Informal example

Consider the program in Fig. 1. The variant in Fig. 1a declares two variables $a$ and $b$, both initialized to 0, then based on a nondeterministic choice (such as user input) sets the pointers $c$ and $d$ to each point to different ones. Afterwards, the program sets the memory location at address $c$ to 1, and the memory location at address $d$ to 2. It is required that at least $a$ or $b$ has the value 1, because we know that

1) $c$ points to $a$ or $b$, and
2) $d$ cannot alias $c$, thus cannot overwrite the value 1.

Therefore, we expect a program analysis verdict of *safe*.

### A. Model Checking without Abstraction

First, we construct a second variant of the same program Fig. 1a, seen in Fig. 1b – $a$ and $b$ are now pointers themselves, initialized to unique memory address via the call to `malloc`, and instead of references to $a$ or $b$, $c$ and $d$ simply alias exactly one of them, each. The requirement is the same: at the memory location pointed to by either $a$ or $b$, we must find the value 1. The advantage of this form is the lack of the `&` (reference) operator, thus we only need to handle the $\star$ (dereference) operator when we encode the data flow in an SMT formula.

This research was partially funded by the EKÖP-24-{2,3} New National Excellence Program under project numbers EKÖP-24-2-BME-118, EKÖP-24-3-BME-213 and EKÖP-24-3-BME-159, and the Doctoral Excellence Fellowship Programme under project numbers 400434/2023 and 400443/2023; funded by the NRDI Fund of Hungary.

[1]https://unix.org/whitepapers/64bit.html

```
1 int a;
2 int b;
3 a = 0; b = 0;
4 int cond = nondet();
5 int *c =  cond?&a:&b;
6 int *d = !cond?&a:&b;
7 *c = 1;
8 *d = 2;
9 assert(a==1||b==1);
```

(a) Before preprocessing

```
1 int *a = malloc();
2 int *b = malloc();
3 *a = 0; *b = 0;
4 int cond = nondet();
5 int *c =  cond?a:b;
6 int *d = !cond?a:b;
7 *c = 1;
8 *d = 2;
9 assert(*a==1||*b==1);
```

(b) After preprocessing

Fig. 1: Program showcasing pointer aliasing

```
1 int arr[2];
2 arr[0] = 0;
3 arr[1] = 0;
4 int cond = nondet();
5 int c =  cond?0:1;
6 int d = !cond?0:1;
7 arr[c] = 1;
8 arr[d] = 2;
9 assert(
      arr[0]==1||arr[1]==1);
```

Fig. 2: Aliasing via offsets

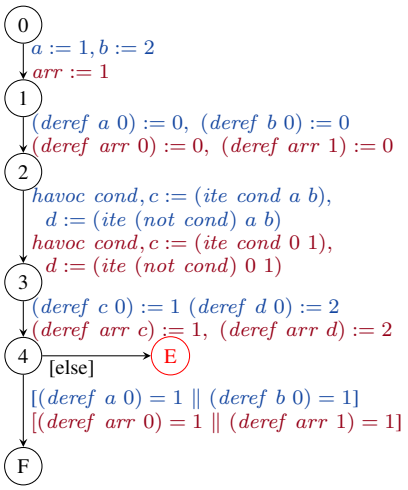



Fig. 4: CFA of Fig. 1b and Fig. 2

```
1 (declare-fun deref (Int Int Int) Int) ; (ptr, offset, idx) -> value
  (declare-fun arr () Int) (declare-fun c () Int) (declare-fun d () Int)
  (declare-fun cond () Bool)                          ; havoc cond
2 (assert (= arr 1))                                  ; unique arbitrary address
3 (assert (= (deref arr 0 1) 0)) (assert (= (deref arr 1 1) 0))
4 (assert (= c (ite cond 0 1))) (assert (= d (ite (not cond) 0 1)))
5 (assert
    (let ((idx1 (+ 1 (ite (= c 1) 1 (ite (= c 0) 1 0)))))
    (let ((idx2 (+ 1 (ite (= d c) idx1 (ite (= d 1) 1 (ite (= d 0) 1 0))))))
      (and (= (deref arr c idx1) 1)                   ; previous idx + 1
           (= (deref arr d idx2) 2)                   ; previous idx + 1
6          (let ((idx3 (ite (= 0 d) idx2 (ite (= 0 c) idx1 1))))
           (let ((idx4 (ite (= 1 d) idx2 (ite (= 1 c) idx1 1))))
             (or (= (deref arr 0 idx3) 1)             ; previous idx
                 (= (deref arr 1 idx4) 1))))))))      ; previous idx
```

Fig. 3: SMT-encoding of Fig. 2 over the trace from 0 to $F$ in Fig. 4

| *ptr* | *off* | *idx* | (*deref ptr off idx*) |
|---|---|---|---|
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 |
| 1 | 0 | 2 | **1** |
| 1 | 1 | 2 | **2** |

Fig. 5: One of the models for *deref* in Fig. 3 (final results in **bold**)

Then, we construct a Control Flow Automaton (CFA) [6] seen in Fig. 4[2] where we also show the CFA of the program in Fig. 2 (shown in a different color on the edges), which is functionally the same as Fig. 1b but with a single stack-allocated array and two literal offsets instead of $a$ and $b$. $a$, $b$, and $arr$ are initialized to unique but unimportant values (here we used a small natural number for each), representing the memory address. For the dereferences in both cases (either via the $\star$ or the `[]` operator), we use an uninterpreted SMT function `deref`, taking two values: a *basis* pointer address and an *offset*. For $*a$ and $*b$ in Fig. 1b the bases are unique addresses returned by `malloc` and the offsets are 0, while for $arr[0]$ and $arr[1]$ in Fig. 2 the bases are the same ($arr$), and the offsets are 0 and 1.

Because the programs are *safe*, encoding the path from the *initial* location to the *error* location should be `UNSAT`, therefore, for demonstration purposes, let us encode the path to the *final* location instead in Fig. 3, which should be *SAT*. We use the array variant in Fig. 2.

After the function declarations in line 1, we assign a unique arbitrary address (here, 1) to $arr$ in line 2. Then based on the nondeterministic value of $cond$, we set the values of $c$ and $d$ to either 0 or 1 in line 4. The writes to $arr$ are encoded in line 5, and the reads for the `assert` are encoded in line 6.

Notice using a three-parameter `deref` in the SMT encoding instead of the two-parameter one in the CFA. The idea is similar to the static single assignment (SSA) [8] used for regular variable encoding for SMT queries: constants are created by associating indexes with variable usages, where each time a variable is updated, its corresponding index is incremented. This allows us to reason about changing values at a memory address, rather than a constant. We encode this index symbolically in the third (index) parameter of the `deref` function. In line 3, we know that the array updates are the first in the program, therefore we can use a literal 1 index. In line 5 however, we do not know if $arr[c]$ overwrites $arr[0]$ or $arr[1]$, so we must construct an expression that evaluates to the correct index instead of a literal. We use the parameter `idx1` for this, which must be 1 greater than the previous index, because we overwrite the value. The previous index is 1 if $c = 1$ or $c = 0$. Otherwise, the previous index is 0, referring to the uninitialized memory at the beginning of execution. We can also construct `idx2` the same way, but we must also consider the case where $d = c$, in which case the previous index is `idx1`. We use the same logic to construct `idx3` and `idx4` in line 6, but we do not increment the previous indices because these are *read* accesses.

By querying an SMT solver with the input in Fig. 3, we can retrieve a model for the `deref` function (see Fig. 5). Each entry in the table refers to a value in the memory throughout the execution of the program, and entries with the highest *idx* value for any given *(ptr, off)* pair represent the last write to the specific memory address, thus representing the state of the memory at the end of execution.

[2]We show the CFA in a partial Large Block Encoding (LBE) [7] for readability, but single block encoding or full LBE could also be used

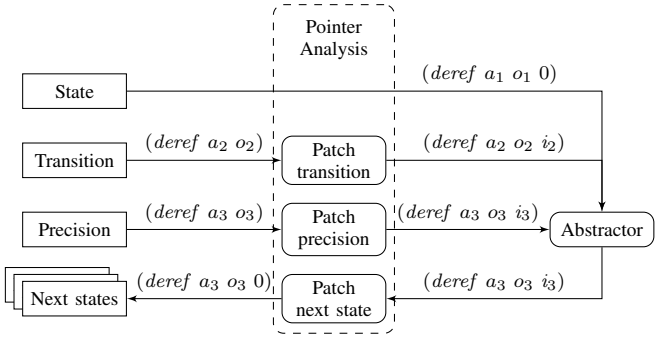



Fig. 6: Integration into abstraction frameworks. Labels denote example subexpressions in the in- and output of the abstractor.

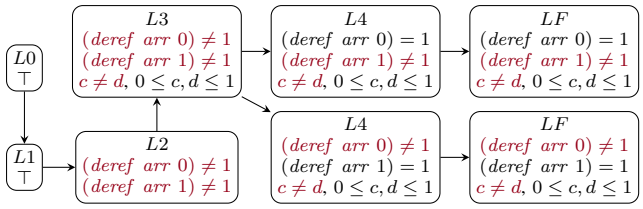



Fig. 7: ARG for Fig. 2 with $\Pi = \{(deref\ arr\ 0) = 1;\ (deref\ arr\ 1) = 1;\ c = d;\ 0 \leq c, d \leq 1\}$

### B. Model Checking with Abstraction

Instead of using a monolithic SMT formula to represent a trace in the program, we can rely on abstraction to reason about the reachable state space of the program. Let us consider the same program in Fig. 2, using predicate abstraction [6] with the predicates $\{arr[0] = 1; arr[1] = 1; c = d; 0 \leq c, d \leq 1\}$ and explicit location tracking.

We assume we can rely on an existing predicate abstractor (that takes a state, a transition, and a precision and returns a list of successor states), and only create a *pointer analysis* adapter that patches the input and output of that abstractor, as seen in Fig. 6.

Because we want to represent the state of the memory in each abstract state, we use 0 as the index to the three-parameter `deref` in the state labels, and re-start indexing in every successor state calculation. Therefore we do not need to adapt the state's expression when invoking the abstractor. We do need to patch the transition, however, because on the labels of the CFA all *deref* calls had two parameters only. We calculate a suitable index for every dereference as demonstrated in Fig. 3, and pass it to the abstractor. We do not use dereferences preceding the input state in the index calculation, only those that come afterward – should a read not find a suitable previous write in the transition, it will read the 0-index value included in the state.

We also insert an additional read access at the end of the transition for every dereference present in the precision, thus querying the resulting values in the memory, including those (patched to use 0 as indices) in the next states when the truth value of a predicate is established.

Using this method, we can construct the abstract reachability graph (ARG) [6] in Fig. 7 containing no reachable abstract error state. We can thus conclude that the program is *safe*. Had we found an abstract counterexample, we would have checked its conretizability to avoid spurious results by using the approach in Sect. II-A. If a spurious counterexample is reached, we refine the precision using its refutation (e.g., via interpolation), remembering to remove the indices from all dereferences.

### C. Advantages of `Deref`

While the presented encoding is similar to other pointer handling techniques, our approach combines and extends their advantages as follows:

**Performance** We only keep track of "interesting" abstract places in the memory (ensured by precision refinement)

**Precision** We can rely on precise aliasing information (similar to dedicated analyses [4]) but only when deemed necessary by the refiner, thus sparing overly detailed tracking

**Array polymorphism** Arrays can be polymorphic and require no explicit conversion when placing different types in an array (a trait of uninterpreted SMT functions)

**Flat struct support** Due to the aforementioned polymorphism, structs can be modeled as arrays, even when containing elements of different types

To expand on the struct support and polymorphism, consider a mapping from `struct A{char c; float f;}` to the `deref` function. While structs by themselves can be mapped to individual variables via a preprocessing step, when combined with pointers (especially when arrays-of-structs are used), this transformation can be hindered. Therefore, we deal with structs as if they were polymorphic arrays themselves. When accessing a value, the base pointer can be an abstract memory location $a$, and the offset can be the index of the field in the type (1 for `c` and 2 for `f`). Because these indices never change, $(deref\ a\ 1)$ will always refer to a 1-byte bitvector, and $(deref\ a\ 2)$ a single-precision float. Special care needs to be taken in cases when the semantics of the source language (such as C) passes the struct by value rather than by pointer or reference, as a deep copy of the values must be made in these cases (so that any modification will not be reflected in the original struct).

## III. Assumptions and Consequences

We assume the following about the input program.

*a) Program Traits:* We assume the input is a control flow automaton (CFA) [6] with variables $V = \{v_1, v_2, \ldots, v_n\}$ over domains $D_{v_1}, D_{v_2}, \ldots, D_{v_n}$, locations $L$, and edges $E \subseteq L \times Ops \times L$, where $Ops$ can have $assume(expr)$, $assign(expr_{lhs}, expr_{rhs})$, and $havoc(v \in V)$ instructions for guards, variable updates, and nondeterministic value assignments, respectively. Sequential combination of operations are permitted in the complex operation $seq(op_1, ...)$, which is a shorthand for a sequence of edges and locations with individual operations. Domains of variables are subsets of domains

in SMT. Expressions can be variables, literals, and domain-specific operations over expressions. Besides conventional operators, we also use a distinguished binary operator `deref`, which maps a base pointer and offset pair to an in-memory value. Pointers and offsets are integers. Conventionally, the left-hand side of assignments could only be variable references, but we allow `deref` expressions as well, to facilitate address-based updates. While executing, the state of the CFA is identified by a valuation over $V$, the current location $l \in L$, and the state of the global memory, given by a collection of base-offset-value triples (we use a 2D memory model [2]). We assume that a memory location is identified uniquely by its base- and offset (an array cannot "index into" another array).

*b) Pointer Semantics:* We further assume that input programs follow the C standard [9] for pointer-, array- and heap-manipulating instructions. Furthermore, all memory accesses are *correct*, because our primary goal is answering error state reachability queries, for which we can assume no undefined behaviour may occur on trajectories leading up to the error state. Therefore we use no bounds-checking, nor do we check for invalid `free` usages.

*c) Eliminating References:* We rely on the CFA not containing variable references, provided by a pre-transformation step that transforms all referred variables to dynamically allocated 1-element arrays. Therefore, references will become variable accesses (see Fig. 1), and previous variable accesses will become dereferences. Because variables can only be referenced a finite amount of times, this pre-transformation step is always possible, and we can always rely on the CFA not containing any references.

*d) Transforming Dereferences:* During analysis (i.e., calculating successor states from a state and a CFA edge), we transform the binary `deref` instruction to ternary `deref` expressions for the SMT solver. The third argument is an index. The index 0 refers to the value in memory before any known writes (either uninitialized, or referring to the value in the previous state). A lower-index `deref` expression is considered overwritten by a higher-index one, if their base and offset values match. Indices are incremented when the corresponding `deref` expression is present on the left-hand side of an assignment. The highest index expression will appear in the successor state as the in-memory value for the corresponding base and offset.

*e) Considerations for Dependency-Based Analyses:* Algorithms used for the verification of concurrent software often require identifying dependency between interacting program operations from different threads [10]. Without pointers, this can be easily performed by a syntactic check on shared variable accesses. However, with pointers, this is insufficient: we also have to check if the operations access the same memory location. Since we may not know the exact basis and offset values of parallel memory accesses, we can use an SMT solver to decide whether they can point to the same memory location provided the information in the current (abstract) state. This is potentially a very expensive operation, but can provide accurate information in the abstract state space. A possible alternative is to use a safe overapproximation, like a static *points-to analysis* [11], which can detect dependencies of CFA edges as a pre-processing step.

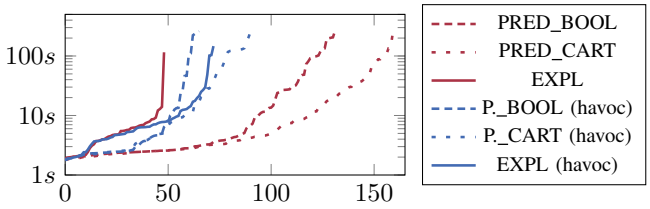



Fig. 8: Quantile plot of results

## IV. Practical Evaluation

To evaluate the proposed approach, we developed a proof-of-concept implementation in the Theta model checking framework [6]. We introduced an adapter layer between the XCFA frontend [12] and the CEGAR backend [6], which implements the necessary transformations of the expressions exchanged between the two regarding pointers. We aimed to answer the following research questions:

**RQ1** For verification tasks containing pointers, can the proposed approach solve more tasks than using the *havoc* memory model (where all reads are nondeterministic)?

**RQ2** For verification tasks containing pointers, which abstract domain is better: explicit value, or predicate analysis?

Answering **RQ1** helps in establishing the rationale for implementing the pointer analysis extension for the abstractor. With the *havoc* memory model, all reads are nondeterminstic, thus they force coarser abstraction and may hinder safety proofs. Answering **RQ2** can result in suggestions for potential users on abstract domain choice when faced with verification problems containing pointers.

To facilitate the experiments, we selected 633 error label reachability tasks containing pointers from the sv-benchmarks repository[3] which can be parsed by Theta. We ran Theta with predicate and explicit value abstraction, using sequential interpolation in the refinement step [6]. We executed these tests on 2 dedicated AMD EPYC 7352 cores each, with 16GB of memory and 5 minutes of timeout. We used BenchExec [13] for precise resource measurements. The results can be seen in Fig. 8, with axis $x$ displaying the number of solved tasks and axis $y$ the time necessary to achieve that many solutions. There were no incorrect results.

**RQ1** and **RQ2** can both be answered directly from the plot: for the explicit value analysis, it is *not worth* using the proposed new technique as it presumably introduces such an overhead that previously solved tasks now become unsolvable within the time limit. Predicate abstraction with the new approach, however, outperforms the *havoc* memory model with either abstract domain. Therefore, we can conclude that

[3]https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks

there is value in implementing the technique for predicate abstraction-based analyses. However, the performance differences between the two predicate abstraction strategies require further investigation.

For the explicit value analysis, we theorize that the lack of explicit *memory state* value information leads to the diminished performance. Therefore, we will attempt to realize this addition to the explicit abstract domain in the future, and re-evaluate the proposed approach.

## REFERENCES


[1] D. Beyer, M. Dangl, and P. Wendler, "A Unifying View on SMT-Based Software Verification," *J. Autom. Reason.*, vol. 60, no. 3, pp. 299–335, 2018. [Online]. Available: https://doi.org/10.1007/s10817-017-9432-6

[2] H. Tuch, "Formal memory models for verifying C systems code," Ph.D. dissertation, University of New South Wales, Sydney, Australia, 2008. [Online]. Available: http://handle.unsw.edu.au/1959.4/41233

[3] R. Burstall, "Some techniques for proving correctness of programs which alter data structures. Machine Intelligence 7," *American Elsevier, New York*, 1972.

[4] D. Baier, "Implementation of Value Analysis over Symbolic Memory Graphs in CPAchecker," Master's thesis, Ludwig-Maximilians-Universität München, Institut für Informatik, 2022.

[5] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, "Counterexample-Guided Abstraction Refinement," in *Computer Aided Verification, 12th International Conference, CAV 2000, Chicago, IL, USA, July 15-19, 2000, Proceedings*, ser. Lecture Notes in Computer Science, E. A. Emerson and A. P. Sistla, Eds., vol. 1855. Springer, 2000, pp. 154–169. [Online]. Available: https://doi.org/10.1007/10722167_15

[6] Á. Hajdu and Z. Micskei, "Efficient Strategies for CEGAR-Based Model Checking," *J. Autom. Reason.*, vol. 64, no. 6, pp. 1051–1091, 2020. [Online]. Available: https://doi.org/10.1007/s10817-019-09535-x

[7] D. Beyer, A. Cimatti, A. Griggio, M. E. Keremoglu, and R. Sebastiani, "Software model checking via large-block encoding," in *Proceedings of 9th International Conference on Formal Methods in Computer-Aided Design, FMCAD 2009, 15-18 November 2009, Austin, Texas, USA*. IEEE, 2009, pp. 25–32. [Online]. Available: https://doi.org/10.1109/FMCAD.2009.5351147

[8] R. Cytron, J. Ferrante, B. K. Rosen, M. N. Wegman, and F. K. Zadeck, "Efficiently computing static single assignment form and the control dependence graph," *ACM Trans. Program. Lang. Syst.*, vol. 13, no. 4, p. 451–490, oct 1991. [Online]. Available: https://doi.org/10.1145/115372.115320

[9] "Programming languages — C," International Organization for Standardization, International Electrotechnical Commission, International Standard, Dec. 2010.

[10] P. Abdulla, S. Aronis, B. Jonsson, and K. Sagonas, "Optimal dynamic partial order reduction," *ACM SIGPLAN Notices*, vol. 49, no. 1, pp. 373–384, 2014.

[11] B. Steensgaard, "Points-to analysis in almost linear time," in *Proceedings of the 23rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, 1996, pp. 32–41.

[12] L. Bajczi, Z. Ádám, and V. Molnár, "C for yourself: comparison of front-end techniques for formal verification," in *Proceedings of the IEEE/ACM 10th International Conference on Formal Methods in Software Engineering*, ser. FormaliSE '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 1–11. [Online]. Available: https://doi.org/10.1145/3524482.3527646

[13] D. Beyer, S. Löwe, and P. Wendler, "Reliable benchmarking: requirements and solutions," *Int. J. Softw. Tools Technol. Transf.*, vol. 21, no. 1, p. 1–29, feb 2019. [Online]. Available: https://doi.org/10.1007/s10009-017-0469-y

# Explainable Graphical Model-Based Transcriptomic Clock from Single Cell Gene Expression Data

Dániel Sándor, Péter Antal
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {sandor, antal}@mit.bme.hu

***Abstract*—Predicting the biological age of an organism is a critical task, and multiple accurate models currently exist. However, to extend both lifespan and more critically, healthspan, it is essential to identify points of intervention and understand how to address them. Effective treatments, whether through medication or other methods, require the ability to diagnose the underlying causes of reduced lifespan, like the up or down-regulation of certain genes.**

**In this study, we evaluate the performance of a modern structure-learning algorithm in constructing a graphical model-based clock for predicting and analyzing biological age. Using a single-cell gene expression atlas, we develop a probabilistic graphical model capable not only of predicting biological age but also of identifying potential gene-level interventions. We compare the performance of various Bayesian and non-Bayesian algorithms for age prediction, assessing them based on their predictive accuracy and ability to elucidate complex biological processes associated with aging in Caenorhabditis elegans.**

***Index Terms*—structure learning, probabilistic graphical models, biological clock, gene expression**

## I. Introduction

Biological aging clocks predict the biological age of an organism, which is a measurement of health [22]. They usually measure cellular aging within the organism. Biological age differs from chronological age, however it is strongly correlated. This metric is influenced by a multitude of factors, like diet, exercise, or environmental stress. Of course, measurement of biological age is a difficult task, thus we try to model it by the condition of the organism. It is an important measurement because if we can get an accurate picture of the condition, we might gain insights into the possible interventions or treatments.

Many different methods have emerged to predict the biological age, primarily but not necessarily using molecular level data, e.g. epigenetics, telomere length, or transcriptomic predictors [3]. However, non-molecular approaches are also available [8], and if they can be combined we can hope to get a more accurate descriptor (for recent reviews see: [23], [24]). To combine aspects we need models that support multimodal transfer learning to efficiently learn across domains. In this study we focus on the most widely used transcriptomic clocks [1], [4] that use RNA-seq data, to model biological age based on the expression levels of genes.

We find that most state-of-the-art methods using predictive approaches are not explainable and are hard to validate with experiments [4], [7], [8]. We attempt to construct a Probabilistic Graphical Model of aging in Caenorhabditis elegans, to create an explainable model, which also has predictive capabilities using Bayesian inference. The choice of c. elegans was made, because of the extensive data available on the organism including cell type-specific aging patterns [1]. Thus validation of the learned biological clocks becomes easier.

Our main contributions are the following:

- We create a biological clock, that has predictive value on novel, established data from single-cell RNA-seq experiments on c. elegans.
- We demonstrate the power of structure learning using scalable structure learning of Directed Acyclic Graphs.
- We validate the explainable model based on existing findings on c. elegans aging.

The structure of the paper is the following: In section II we present the related works of structure learning and biological clocks. In section III we present the methodologies of data and model, with an emphasis on the structure learning algorithm. In section IV we discuss our findings on the explainable network-based transcriptomic clocks. We end in section V with a conclusion on the usability of biological clocks.

## II. Related work

Recently, multiple aging atlases have been created for many different organisms [1], [5]–[7]. In all these studies, subjects' genes are sequenced and examined in different age groups. Different studies highlight certain aspects, but many have claimed to identify relevant genes that are up or down-regulated with time. Thus they can be associated with the process of aging.

### A. Transcriptomic markers of aging

Roux et al. [1] find that aging in c. elegans depends heavily on the cell type examined, as different cells differ in their stress signature. They find that gei-3, a gene responsible for enabling DNA binding transcription factors, is upregulated with age in a broad range of cell types, thus they conclude that it is

PROJECT NO. 2024-2.1.2-EKÖP-KDP-2024-00005 HAS BEEN IMPLEMENTED WITH THE SUPPORT PROVIDED BY THE MINISTRY OF CULTURE AND INNOVATION OF HUNGARY FROM THE NATIONAL RESEARCH, DEVELOPMENT AND INNOVATION FUND, FINANCED UNDER THE EKÖP_KDP-24-1-BME-15 FUNDING SCHEME.

universally associated with biological age. de Magalhaes et al. [7] examined human genes, using mice as model organisms, and created a network of aging based on associated genes related to DNA metabolism. Aging Atlas [5] is a collection of multiple genetic and other biomarkers related to aging, its notable RNA-seq module is responsible for the findings on the human FOXO3 and CLOCK genes, both of which change significantly with aging. However, these markers are only predicted by associations, and so far few causal approaches exist to find relevant regulators in the aging process.

### B. Biological clocks

Biological clocks measure the aging of the organism with mixed predictive performance and use a wide array of modalities [2], [3], [8], [21]. In addition to RNA-seq, the most prominent predictors of biological age are epigenetic markers [2], [10]. The length of the telomere is also mentioned in models with high predictive performance [9]. But extreme cases also exist, such as the prediction of biological age from photos [8]. Some promising approaches based on DNA methylation include XAI-AGE [21], where Prosz et al. have developed an explainable clock with predictive performance comparable to state-of-the-art black-box models. Transcriptomic clocks differ from most of these, in the sense that they can measure variables, that can directly influence aging through gene expression levels.

### C. Transcriptomic clocks

Shokhirev and Johnson [11] model multiple clocks based on different markers, using regression techniques to weigh the effect of each gene. This creates extremely accurate clocks with an $R^2$ value of 0.96. The study demonstrates that simple linear models are often enough to predict biological age; however, this result also relies on the association of genes with age, without regarding causality. Holzscheck et al. [12] use neural networks to predict age, with a smaller dataset, however, they manage to incorporate a priori knowledge in their system. In humans, they claim to achieve a mean absolute error of 4.7 years. Our most important comparison is BiT age [4], which is based on the Elastic Net Regression, which also achieved an $R^2$ value of 0.96, where they claim that it is close to the limit of predictive performance.

### D. Network-based modeling

While explainable models do exist in the field of biological aging clocks, these are mostly focused on the post-hoc explainability of blackbox models [13], [21]. To use graphical model-based solutions we find methods in the field of causal discovery. The goal of causal discovery is to learn the structure of independences between our variables. With the help of this, we can construct a regulatory network of genes, where we can incorporate age as a variable, that genes can influence. An important point to make is that we are using a prior, where age is determined from the genetic patterns, thus age cannot have outgoing edges, as it would violate this principle i.e. processes in the organism cause biological age, and biological age cannot be the cause of genes being expressed differently. However modeling a DAG still makes sense, as necessary interventions can be made at different variables with varying effects.

Learning gene regulatory networks by causal discovery has proven to work in multiple scenarios [14]–[16]. The most prominent real benchmark dataset is the Sachs protein signaling [17] in the field. We present two methods Generative flow networks (GFlow) [19] and BayesDAG [18], both of which are able to solve the Sachs dataset with acceptable accuracy. The choice of Bayesian methods is necessary if we want to conduct further analysis. By sampling directly from the posterior, we can find certain and uncertain parts of our networks, and improve them with prior knowledge.

## III. Methods

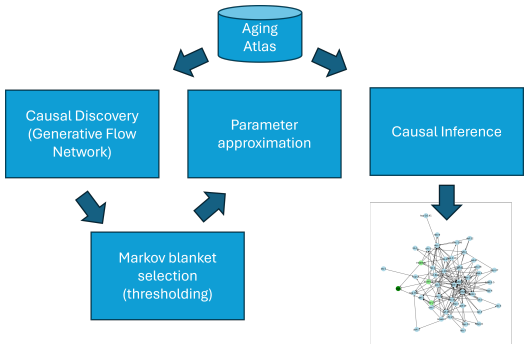



Fig. 1: Overview of methodologies.

We perform end-to-end examination of the aging atlas from Roux et al. [1]. Our primary goal is not to achieve the highest predictive accuracy of age. However, predictive power is an important requirement for the model. We want to draw causal conclusions from the model, that is why we use Causal Discovery and Causal Inference along the process. To tighten the scope we do not analyze the whole dataset, but focus on its most visible changes (the aging of amphid neurons). An overview of the process can be seen in Fig. 1

### A. Aging atlas

Our goal was to demonstrate that causal prediction can be drawn from the collected data. We examined the results of the aging atlas and tried to narrow the search, for parts where meaningful conclusions can still be drawn but the number of variables is small enough, that learning a Generative Flow Network over DAG space was still feasible. The training of the network was conducted on the Single-cell Variational Inference transformed 'denoised' dataset.

### B. Gene selection

The original paper mentions more than 4000 genes by name, and that is too large in the state space, for most DAG learning strategies to be applicable. As we wanted to prove the most important results of the paper we curated the list of genes to the 50 we deemed most important. The complete list of genes and the reasoning behind their choices can be found in

Appendix A. The most important principles in choice were the following: We chose known indicators of healthspan in c. elegans, like skn-1, daf-16, hlh-30, fkh-7, fkh-9, daf- 12, dpy-27, gei-3. We included the hsp-70 group (hsp-70, hsp-16.41, F44E5.4, F44E5.5, hsp-110) as they are mentioned to be important cytosolic chaperones, which are upregulated in neurons. Ultimately we included every gene that is described in detail and affects the aging process. Thus our goal is not the selection of relevant genes regulating aging, but finding the best predictors and explaining their relationships.

### C. Amphid neurons

Amphid neurons were key samples in the learning process. Roux et al. find that they exhibit the most changes with aging from all cell clusters of c. elegans. To denoise the dataset we filtered for the samples from amphid neurons, thus making it easier for an estimator to find the connections between variables. Furthermore, all genes mentioned to be up or downregulated in these cells are included in the gene set.

### D. Generative flow network

We use the Generative Flow Network [19] as a base estimator for the DAG posterior. The method uses an iterative approach to building a DAG by adding edges and measuring the probabilities of DAG states by the flow going through them in each state. Flow is defined over the graph of DAGs, where the amount of flow going through a DAG state is proportional to the state's posterior probability. We sample the DAG posterior to get an accurate distribution, on which we can learn the parameters. In the cases where point estimates are given, the estimate is calculated by averaging 64 samples from the posterior, in no cases were the acyclicity constraint violated by averaging.

### E. Selecting the Markov Blanket

The Markov Blanket of age is equivalent to its parents in this case, as the prior forbids any outgoing edges from the variable. The only relevant question is where to threshold the edge probabilities to get a desirable Markov Blanket set (MBS). The thresholding is done on the posterior probabilities of the edges in the result of the GFlow algorithm. The desirable MBS is non-empty and contains less than the total number of variables, for explainability and predictive performance.

In most of our experiments, the threshold was set to 0.5, as this contained all edges that were more likely to be in the DAG than not, based on the edge posteriors. To include more variables we conducted experiments with MBS threshold of 0.46 (a threshold determined by the iterative lowering until we got the desirable amount of variables), however, these do not seem to enhance the predictive probabilities of the models. In cases where we trained the model on the full dataset, the threshold is set to 0.6 to account for the noise introduced with the new samples.

### F. Parameter learning

On the thresholded MBS we conduct parameter learning, to create a Bayesian Network. In most cases it is simple as the target variable has a small number of parents in the BN, thus the conditional probability tables (CPD) will in turn also be small. As most approaches only support discrete values, for the parameter learning we discretize the parents of the age (which is discrete in the original dataset, into quartiles). We use maximum likelihood estimation to find the best parameter for each value of the CPD.

### G. Causal inference

In practice the inference is simple [20]. For each sample of the test dataset, we take the discretized version and predict age, by taking the weighted average of the resulting distributions. If we do not care about point estimates we can generate the posterior over the age variable, which is a more meaningful measure, as this incorporates any uncertainties the model might carry.

## IV. Results

In this section, we discuss our findings. The two setups for experimentation only differed in the number of samples used: In the full data setup we used 37938 samples to train our structure and parameter models, and in the amphid neurons setup we used 79 samples for training.

### A. Results of structure learning

In the full data case a more dense DAG was found after averaging the models, however, to get a clearer picture we used a higher threshold of 0.6 the resulting DAG structure can be seen in Fig. 3. To validate the structure we turn to the Aging Atlas. Roux et al. report, that lin-1, which is one of the parents of age in the DAG, is a transcription factor that is associated with increased movement and healthspan, thus it makes sense that it affects biological age, however, it is not the most indicative of lifespan according to Roux et al. The other parent of age is F46A8.13, which is one of the top-ranked genes, that is upregulated across cell types in aging, however, interestingly this is only in the 17th place in the dataset. Although not all genes in the ranking are covered by our examination, this might suggest that further refinement of the method is required.

In the case of the amphid neurons, the threshold can be left at 0.5, this keeps only edges, that most of the DAGs in the sampled posterior contain. The point estimate of the DAG can be seen in Fig. 4 In this case the only parent of the age variable is F44E5.4, which is one of the cytosolic chaperones from the hsp-70 family, responsible for protein refolding. Its parent is hsp-16.2, also from the same family. Roux et al. show, that these genes are heavily upregulated with age in all neuron classes, but especially in amphid neurons. Further examination of the hsp-70 family's regulatory network might be necessary to prove the regulatory relationships between them and aging, but right now we can show, that this family influences the cell type's aging, as we can see from the predictive performances as well.

### B. Predictive performance

When it comes to the accuracy of the clocks, the best practice we can do to evaluate them is to examine their correlations with chronological age. We do not expect them to be fully identical, as biological age is influenced by a multitude of factors, but this is the best indicator for the performance of aging clocks. To compare these we share the $R^2$ value for each clock, and compare them to existing approaches.

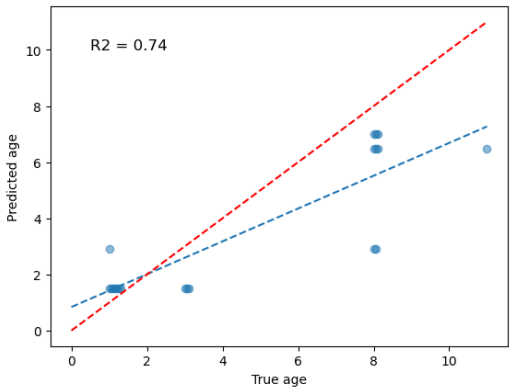



Fig. 2: Correlation of predicted biological age and true chronological age in amphid neurons.

For the prediction of the age variable we use causal inference, and to make it efficient we only include evidence belonging to the Markov blanket of the variable. In each case, we discretized the test data, based on the quartiles of training data, and predicted on the test set. The test data in all cases is a randomized 10% of the relevant dataset. We present the values in Table I

TABLE I: $R^2$ values of different clocks

| **Method** | $R^2$ |
|---|---|
| Full Dataset GFlow + MLE | 0.46 |
| Amphid Neurons GFlow + MLE | 0.74 |
| BiT age reported | 0.96 |

We can see that all our methods significantly correlate with the chronological age variable, which is a remarkable result if we consider the discrete dataset that was used for parameter learning. From our clocks, the amphid neuron case performs best, with the 0.5 threshold, increasing or decreasing the threshold all yield worse results, as more variables introduced more noise in the inference. We can see the model's performance on the whole test set in Fig. 2. We find that most test samples are predicted to be younger than their chronological age, but potentially all this difference can be explained by the discretization of the test dataset, and the uncertainty introduced by the small size of the dataset.

## V. Conclusion

To summarize we presented the idea of an explainable network-based aging clock based on transcriptomic data. We created the clock, which although lags in predictive power from the theoretical limit, is widely usable for causal discovery and further research in aging-related questions. Both the found genes (F44E5.4, lin-1, F46A8.13) are good predictors of aging, and are worthy of further genetic experiment-based examination, together with the hsp-70 family. It is also mentioned by Roux et al. and confirmed by our finding of the structure of the gene regulatory network.

## Appendix A
## List of genes

The complete list of genes for the construction of the clocks is the following: gei-3, daf-16, skn-1, hlh-30, fkh-7, fkh-9, daf-12, nhr-23, nhr-25, dpy-27, hsp-70, hsp-16.41, hsp-16.2, hsp-4, cup-2, pdi-6, hsp-3, hrdl-1, cnx-1, crt-1, ctc-3, nduo-6, pck-1, gpd-2, tpi-1, rpl-3, rps-9, rps-28, mec-12, let-607, stc-1, clec-166, tag-353, unc-73, flp-24, ceh-36, crh-1, daf-19, F44E5.4, F44E5.5, F46A8.13, F57F5.1, Y94H6A.10, xbp-1, nmad-1, efl-2, elt-7, lin-1, hsp-110

## References


[1] Roux, A. E., Yuan, H., Podshivalova, K., Hendrickson, D., Kerr, R., Kenyon, C., & Kelley, D. (2023). Individual cell types in C. elegans age differently and activate distinct cell-protective responses. Cell Reports, 42(8).

[2] Poganik, J. R., Zhang, B., Baht, G. S., Tyshkovskiy, A., Deik, A., Kerepesi, C., ... & Gladyshev, V. N. (2023). Biological age is increased by stress and restored upon recovery. Cell Metabolism, 35(5), 807-820.

[3] Jylhävä, J., Pedersen, N. L., & Hägg, S. (2017). Biological age predictors. EBioMedicine, 21, 29-36.

[4] Meyer, D. H., & Schumacher, B. (2021). BiT age: A transcriptome-based aging clock near the theoretical limit of accuracy. Aging cell, 20(3), e13320.

[5] Aging Atlas: a multi-omics database for aging biology. Nucleic acids research, 2021, 49.D1: D825-D830.

[6] Kedlian, V. R., Wang, Y., Liu, T., Chen, X., Bolt, L., Tudor, C., ... & Zhang, H. (2024). Human skeletal muscle aging atlas. Nature aging, 1-18.

[7] de Magalhaes, J. P., & Toussaint, O. (2004). GenAge: a genomic and proteomic network map of human ageing. FEBS letters, 571(1-3), 243-247.

[8] Zalay, O., Bontempi, D., Bitterman, D. S., Birkbak, N., Shyr, D., Haugg, F., ... & Aerts, H. J. (2023). Decoding biological age from face photographs using deep learning. medRxiv.

[9] Vaiserman, A., & Krasnienkov, D. (2021). Telomere length as a marker of biological age: state-of-the-art, open issues, and future perspectives. Frontiers in genetics, 11, 630186.

[10] Horvath, S., & Raj, K. (2018). DNA methylation-based biomarkers and the epigenetic clock theory of ageing. Nature reviews genetics, 19(6), 371-384.

[11] Shokhirev, M. N., & Johnson, A. A. (2021). Modeling the human aging transcriptome across tissues, health status, and sex. Aging cell, 20(1), e13280.

[12] Holzscheck, N., Falckenhayn, C., Söhle, J., Kristof, B., Siegner, R., Werner, A., ... & Kaderali, L. (2021). Modeling transcriptomic age using knowledge-primed artificial neural networks. npj Aging and Mechanisms of Disease, 7(1), 15.

[13] Qiu, W., Chen, H., Kaeberlein, M., & Lee, S. I. (2023). ExplaiNAble BioLogical Age (ENABL Age): an artificial intelligence framework for interpretable biological age. The Lancet Healthy Longevity, 4(12), e711-e723.

[14] Feng, K., Jiang, H., Yin, C., & Sun, H. (2023). Gene regulatory network inference based on causal discovery integrating with graph neural network. Quantitative Biology, 11(4), 434-450.

[15] Foraita, R., Friemel, J., Günther, K., Behrens, T., Bullerdiek, J., Nimzyk, R., ... & Didelez, V. (2020). Causal discovery of gene regulation with incomplete data. Journal of the Royal Statistical Society Series A: Statistics in Society, 183(4), 1747-1775.

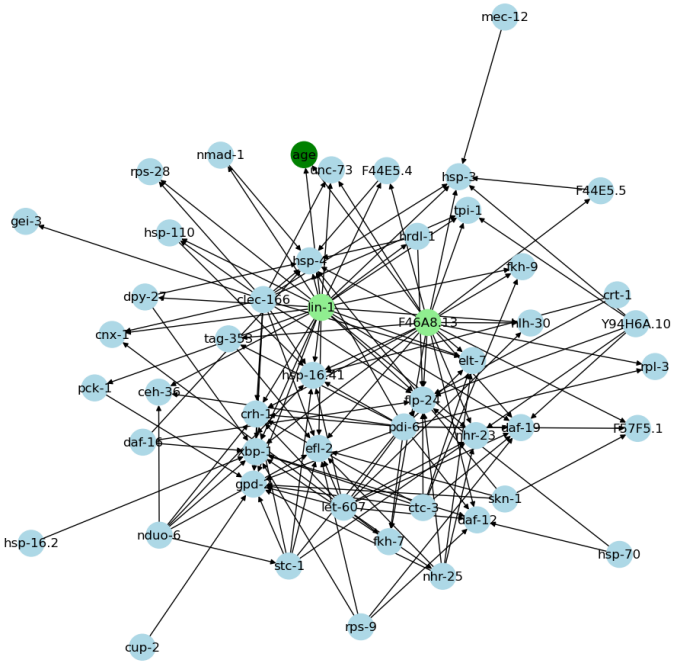



Fig. 3: Structure of the Bayesian Network learned from the full Aging Atlas (average of 64 DAGs, thresholded at 0.6), the Markov Blanket of age is highlighted with green

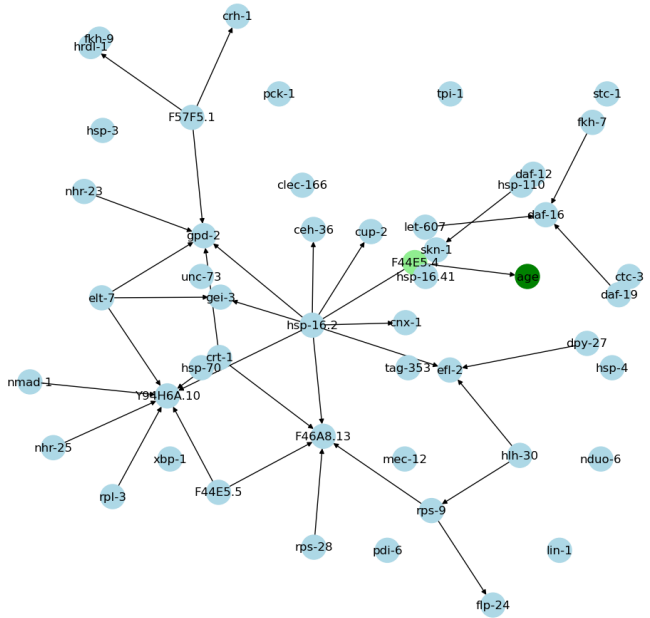



Fig. 4: Structure of the Bayesian Network learned from the amphid neurons from Aging Atlas (average of 64 DAGs, thresholded at 0.5), the Markov Blanket of age is highlighted with green

[16] Sándor D.& Antal P. (2024). Systematic evaluation of continuous optimization approaches for causal discovery of gene regulatory networks. Proceeding of 31th Minisymposium of the Department of Measurement and Information Systems, 6, 50-54.

[17] Sachs, K., Perez, O., Pe'er, D., Lauffenburger, D. A., & Nolan, G. P. (2005). Causal protein-signaling networks derived from multiparameter single-cell data. Science, 308(5721), 523-529.

[18] Annadani, Y., Pawlowski, N., Jennings, J., Bauer, S., Zhang, C., & Gong, W. (2023). Bayesdag: Gradient-based posterior inference for causal discovery. Advances in Neural Information Processing Systems, 36, 1738-1763.

[19] Deleu, T., Góis, A., Emezue, C., Rankawat, M., Lacoste-Julien, S., Bauer, S., & Bengio, Y. (2022, August). Bayesian structure learning with generative flow networks. In Uncertainty in Artificial Intelligence (pp. 518-528). PMLR.

[20] Pearl, J. (2010). Causal inference. Causality: objectives and assessment, 39-58.

[21] Prosz, A., Pipek, O., Börcsök, J., Palla, G., Szallasi, Z., Spisak, S., & Csabai, I. (2024). Biologically informed deep learning for explainable epigenetic clocks. Scientific Reports, 14(1), 1306.

[22] Levine, M. E. (2013). Modeling the rate of senescence: can estimated biological age predict mortality more accurately than chronological age?. Journals of Gerontology Series A: Biomedical Sciences and Medical Sciences, 68(6), 667-674.

[23] Li, Z., Zhang, W., Duan, Y., Niu, Y., Chen, Y., Liu, X., ... & Chen, X. (2023). Progress in biological age research. Frontiers in public health, 11, 1074274.

[24] Bafei, S. E. C., & Shen, C. (2023). Biomarkers selection and mathematical modeling in biological age estimation. npj Aging, 9(1), 13.

# Inductive Learning-Based Qualitative Fault Diagnosis in Distributed Systems

Márton Tarnay, András Földvári, Bertalan Zoltán Péter
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {tarnay.marton, andras.foldvari, bpeter}@edu.bme.hu

***Abstract*—The growing complexity of microservice systems poses significant challenges in diagnosing faulty systems. Traditional monitoring techniques often fall short due to the distributed and dynamic nature of these systems. This paper presents a novel model-based diagnostics framework that uses multimodal observability data for accurate fault detection and localization in microservice environments.**

**The diagnostic process uses Answer Set Programming (ASP), a declarative programming language that leverages logic reasoning over a qualitative multimodal data model to provide insights into the system's state. The presented approach introduces an inductive learning solution for extracting the diagnostic rules, utilizing Inductive Learning of Answer Set Programs (ILASP) to derive explainable diagnostic rules from labeled historical datasets automatically.**

**The approach was evaluated on a benchmark microservice application dataset with promising results compared to existing fault detection and diagnostic solutions.**

***Index Terms*—fault diagnosis, distributed systems, logic reasoning, microservices, qualitative modeling, distributed tracing, observability, answer set programming, inductive learning**

## I. Introduction

Microservice architecture is a modern software development paradigm that has emerged as the prominent way to design rapidly evolving cloud-native applications. Microservices (Figure 1) are small, independently deployed services that communicate over lightweight APIs and are built around business domains [1].

While being highly scalable and flexible, microservice systems introduce significant challenges in observability and understanding system behavior due to their highly complex, decentralized, and modular design [2]. The heterogeneous nature of the architecture and constantly changing topology make traditional monitoring and fault localization techniques difficult. The reliability and availability requirements in these systems necessitate approaches that can tackle multimodal observability data and provide operators with explainable diagnostic results.

The Cloud Native Computing Foundation (CNCF) [3] describes observability as:

The work of Bertalan Z. Péter supported by the Doctoral Excellence Fellowship Programme (DCEP) is funded by the National Research Development and Innovation Fund of the Ministry of Culture and Innovation and the Budapest University of Technology and Economics under a grant agreement with the National Research, Development and Innovation Office.

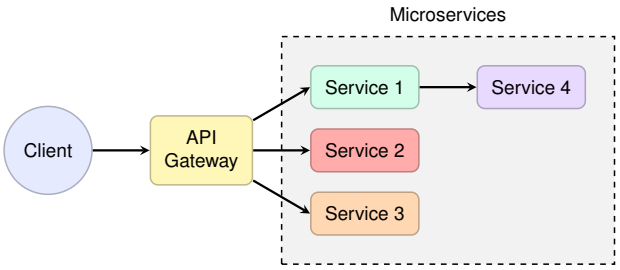



Figure 1. Simplified Logical Structure of Microservice Architecture

> *"A system property that defines the degree to which the system can generate actionable insights. It allows users to understand a system's state from these external outputs and take (corrective) action."*

Observability is essential for understanding the health and performance of complex systems, particularly in microservice architectures. At its core, observability is built upon three main pillars [4]: logs, metrics, and traces. These data types provide a multi-aspect approach to monitoring and diagnosing system behavior. Each one offers unique insights that come together to give a complete picture of how a system works.

The goal of the paper is to address two main challenges concerning fault diagnosis and observability in microservice systems [5]: multimodal telemetry data representation and creating explainable diagnostic results.

The paper presents a novel model-based diagnostics framework for microservice systems, utilizing a joint metamodel representing multimodal telemetry data. The solution leverages Answer Set Programming (ASP) [6], a logic-based reasoning engine, and Inductive Learning of Answer Set Programs (ILASP) [7], an inductive logic programming (ILP) approach, to analyze this fused data, allowing for the identification of faults across interconnected microservice components. This approach tackles the challenges of fault localization in microservice systems by generating explainable diagnostic results and providing actionable insights for operations teams, ultimately improving system reliability and reducing operational workload.

## II. Inductive Learning

The goal of inductive learning in the solution is to extract general diagnostic rules from historical multimodal data. These interpretable rules can be applied in the diagnostic process to precisely identify the location and type of faults.

Inductive learning derives rules from observations that cover the patterns without overgeneralizing, ensuring that the rules are neither too loose – potentially covering irrelevant cases – nor too strict, which might exclude relevant ones.

In our approach, we used ILASP [7] to extract rules from the ASP representation of observations, ensuring that the rules accurately diagnose faults while minimizing the number of false positive results.

### A. Answer Set Programming

Answer Set Programming (ASP) is a form of declarative programming oriented towards difficult search problems [6]. The approach originates from the field of logic programming and non-monotonic reasoning. ASP utilizes the concept of stable model semantics to represent and solve problems. Problems are encoded as logic programs, and the solutions to the problems correspond to answer sets.

In ASP, we start with a finite set $\mathcal{A}$ of propositional atoms. A *literal* is either an atom $a \in \mathcal{A}$ or a default-negated atom $\text{not } a$. A *body* is a set of such literals, often split into the positively included atoms $B^+$ and the default-negated atoms $B^-$. A *head* is a set of atoms $\{a_1, \ldots, a_k\} \subseteq \mathcal{A}$. An ASP *rule* $r$ has the form $H \leftarrow (B^+, \text{ not } B^-)$, where $H$ is the head and $B^+ \cup B^-$ is the body. A *fact* is simply a rule with an empty body, indicating that the head holds unconditionally. Finally, an ASP *program* $P = \{r_1, r_2, \ldots, r_n\}$ is a finite set of such rules, encompassing facts and rules, and, optionally, *constraints* (i.e., rules with empty heads).

An *answer set* is a consistent set of literals that represent a possible solution that satisfies all rules and constraints of the program. $AS(P)$ is the operator that, given a logic program $P$, defines the procedure for computing all of its answer sets.

### B. Inductive Logic Programming

The paradigms of inductive logic programming (ILP) and ILASP are used in this paper to support inductive diagnostic rule generation described in detail in section IV. The advantage of ILP systems is their ability to learn rules that can be explained and interpreted [8]. Compared to other state-of-the-art artificial intelligence and machine learning techniques, which function like black-boxes, ILP systems work transparently with algorithms that can be fully explained.

Formally, an ILP task is a tuple $T = \langle B, S_M, \langle E^+, E^-, C \rangle \rangle$ where $B$ is an ASP program, $S_M$ is a set of ASP rules, $E^+$ and $E^-$ are the finite sets of positive and negative *examples*, and $C$ is the optional context of the example in the form of an answer set program.. $H \subseteq S_M$ hypotheses is an inductive solution of $T$ iff.:

$$\forall e^+ \in E^+ : \exists A \in AS(B \cup H \cup C) \quad : A \text{ extends } e^+ \quad (1)$$

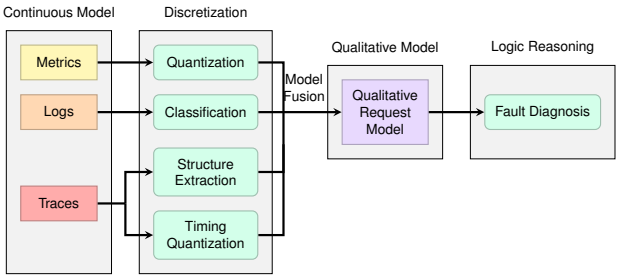



Figure 2. Model Extraction and Reasoning

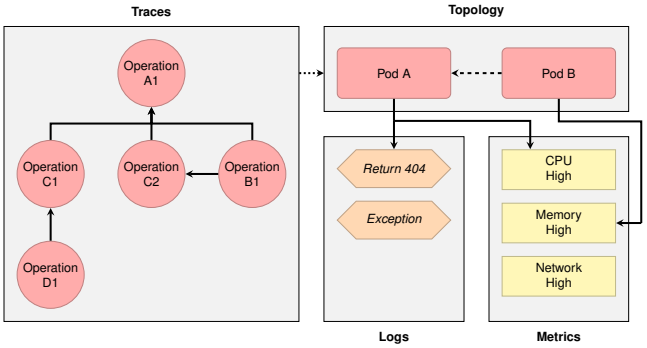



Figure 3. Aggregate Model of the Telemetry Data

and

$$\forall e^- \in E^- : \nexists A \in AS(B \cup H \cup C) \quad : A \text{ extends } e^- \quad (2)$$

(1) means that for all positive examples, there must be a rule that covers the example in the set of answer sets of the union of the background knowledge and the hypothesis. (2) means that for all negative examples, there must not be any rule in the answer sets that cover the negative example [7].

Inductive Learning of Answer Set Programs (ILASP) is a novel system for machine learning of ASP programs from examples [7]. ILASP works by creating a meta-level representation of the candidate hypotheses: rules to be learned in the form of mode declarations. Mode declarations specify the allowed structure of rules, which are used to generate all possible rule structures. Examples and context are translated into constraints. These constraints are used to test the candidate's hypotheses, as the solution must satisfy the positive examples and not violate the negative examples.

## III. System Diagnosis with Multimodal Data

The first step of the model extraction and diagnostic reasoning process (Figure 2) is to create a unified model representation from multimodal data. This unified representation enables the generation of a diagnostic code (in our case ASP), which subsequently facilitates the system diagnostic task through the use of logic reasoning.

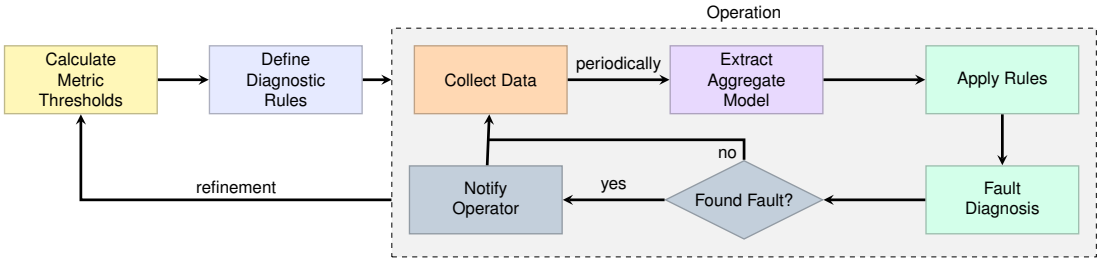



Figure 4. Diagnostic Workflow

### A. Heterogeneous Aggregated Metamodel

For the model created from multimodal data to be viable, an efficient, automatic process is required for the creation of models from the collected telemetry data. The properties are extracted from log, metric, and trace data. The current solution directly generates ASP code from the metamodel. However, there is a possibility for an intermediate representation allowing for the use of other toolchains.

The qualitative representation of data supports the creation of an aggregated model, enabling the compact representation of large datasets.

An example of the reduced and aggregated model can be seen in Figure 3, as a visual representation of the ASP program of the model. This model includes the operation graph. The aggregated logs and metrics are displayed, summarizing the significant anomalies in the time period.

This aggregated data model allows for faster inductive learning without sacrificing diagnostic performance. Using this method also allows for greater scalability, both in the rule extraction and operation phase.

### B. Diagnostics

Combining the ASP program of the aggregate model with the diagnostic rules yields a result that contains a set of fault literals representing the identified faults in the system. These fault literals capture the root cause of the diagnosed fault, while the relevant anomalies are also provided for an interpretable result.

The complete workflow of the created diagnostic approach can be seen in Figure 4. Metric thresholds are created from previously collected data, and the rules are defined before operation. During operation, the system periodically extracts the aggregate model based on the request models representing collected data. The diagnostic rules are evaluated over the aggregate model, and the resulting diagnosis is relayed to the operators in case a fault is detected. The diagnosis may also be used to execute automated corrective actions on the system.

## IV. Rule Extraction with Inductive Learning

Manually defining diagnostics rules is time-consuming and complex, as it requires deep knowledge of the system architecture and behavior, potential failure modes, and even operational nuances. The application of inductive learning offers a solution to this challenge by automatically creating diagnostic rules. With the analysis of labeled historical data, inductive algorithms are able to identify the correlations and patterns needed for fault diagnosis in the system.

### A. Task Definition

First, let us establish the following notations.

- Let $\mathcal{O}$ denote the set of all operations within the system, and let us define $\mathcal{OG} \subseteq \mathcal{O}^2$ where every $(o, o_{\text{parent}}) \in \mathcal{OG}$ pair represents an execution path going from the parent to the child operation.
- Let $\mathcal{S}$ denote the set of services in the system.
- Let $\mathcal{M} \subseteq \mathcal{MT} \times \mathcal{MV} \times \mathcal{S}$ be the set of metric observations in the system where $\mathcal{MT} \overset{\text{eg}}{=} \{\texttt{Memory}, \texttt{CPU}, \ldots\}$ is the set of observed metric types, and $\mathcal{MV} \overset{\text{eg}}{=} \{\texttt{Low}, \texttt{High}, \ldots\}$ is the set of qualitative observation values.
- Let $\mathcal{L} \subseteq \mathcal{LT} \times \mathcal{S}$ be the set of log observations extracted from the system where $\mathcal{LT} \overset{\text{eg}}{=} \{\texttt{HTTP\_404}, \texttt{EXCEPTION}, \ldots\}$ is the set of log types.
- Let $\mathcal{F} \subseteq \mathcal{FT} \times \mathcal{S}$ be the set of faults in the system where $\mathcal{FT} \overset{\text{eg}}{=} \{\texttt{CPU\_CONTENTION}, \texttt{CPU\_CONSUMPTION}, \ldots\}$ is the set of known fault types.
- We partition $\mathcal{F}$ into disjoint subsets $\mathcal{F}^+$ and $\mathcal{F}^-$, comprising the active and inactive faults in the context of the evaluation, respectively.

To create inductive rules, we propose to encode this problem as *ILP tasks* as follows. Let the rule extraction problem be ILP task tuples of the form $T = \langle B, S_M, \langle E^+, E^-, C \rangle \rangle$, where $S_M$ is the hypothesis space defined by the head and body mode declarations $M = M_h \cup M_b$ that are used to generate the hypothesis space.

The heads are defined as $M_h = \mathcal{F}$. The bodies can be defined with the observation sets: $M_b = \mathcal{M} \cup \mathcal{L} \cup \mathcal{OG}$.

The context $C$ contains all observations from the aggregation as $C = \mathcal{L} \cup \mathcal{M} \cup \mathcal{OG}$, in the form of ASP facts.

Finally, $E^+ = \mathcal{F}^+$ and $E^- = \mathcal{F}^-$.

### B. Workflow

This section describes the workflow of inductive diagnostic rule extraction in detail, focusing on the creation of ILASP programs from a fault-labeled monitoring dataset. The complete workflow is shown in Figure 5.

The first step in the process is extracting all faults and their relevant data from a labeled dataset with faults in them. For

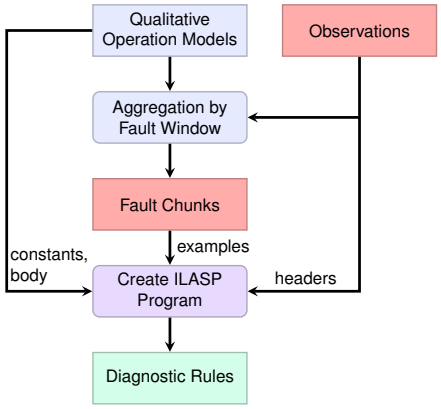



Figure 5. Rule Extraction Workflow

each active fault, the impacted timeslice, which encompasses the duration in which the fault occurred, must be processed; request models shall be created and aggregated into an aggregate model. Following data collection and aggregate model extraction, an ILASP program is created for each microservice.

Each ILASP program is tailored to a specific microservice in the system. The mode headers reflect only the faults injected into the particular microservice assigned to them, meaning that the results of the program will only create rules for that microservice relevant to the context of that service.

Each example in the program is a fault-affected timeslice; all timeslices have to be included across all ILASP programs. This is critical because it allows for the analysis of examples where the inspected microservice is fault-free, providing a more robust dataset for inductive learning. The results of the execution of these ILASP programs are the fault diagnosis rules. These rules encapsulate the learned relationships and patterns identified by inductive learning.

### C. Example

Listing 1 shows two examples from a simple ILASP program. The program contains a single fault, named `exampleFault`. There are two partial interpretations, `aggregate1` and `aggregate2` – the fault is active in the former and inactive in the latter. The only context given is a metric, named `someMetric`, which has a `high` anomaly in `aggregate1`. After running this simple program, the result is a rule that states that the example fault is active when `someMetric` is `high`. This demonstrates the basic principle of how partial interpretations work for diagnostic rules. The result of Listing 1 is:

```
fault(exampleFault) :- someMetric(high).
```

Listing 2 shows an excerpt from an ILASP program with two positive interpretations: `aggregate1` and `aggregate2`. `aggregate1` has a weight of 10 – this penalty value is calculated based on the relative relevance of the example in the diagnosis rule extraction. The value is higher if the example contains an actual fault in the service. It is also influenced by the amount of anomalies present in the context

```
#modeh(fault(exampleFault)).
#modeb(someMetric(const(metricLevel))).
metricLevel(high).
metricLevel(low).


#pos(aggregate1,
    { fault(cpuConsumed) },
    {},
    { someMetric(high) }
).
#pos(aggregate2,
    {},
    { fault(cpuConsumed) },
    {}
).
```

Listing 1: Example ILASP Program for Fault Rule Extraction

```
#pos(aggregate1@10,
    { fault(cpu_contention, frontend) },
    { fault(return, frontend) },
    { cpuMetric(high, frontend).
      memMetric(high, frontend). }
).
#pos(aggregate2@15,
    { fault(return, frontend) },
    { fault(cpu_contention, frontend) },
    {
        log(400, frontend).
        log(error, frontend).
        netMetric(high, frontend).
        has_parent(
            frontend_ErrorHandler,
            frontend_Open
        )
    }
).
```

Listing 2: Example ILASP Program with Penalties

aggregate model. The *positive example* part of the program is the `cpu_contention` fault localized in the `frontend` service. The *negative example* part is the `return` fault also localized in the `frontend` service. The *context* is the `high` CPU metric and `high` memory metric in the `frontend` service. `aggregate2` has a penalty of 15 – this value is higher because it contains more anomalies in the aggregate model. The *positive example* part is the `return` fault in the `frontend` service, and the *exlusions* part is the CPU contention fault also in the `frontend` service. The *context* describes two logs – a 400 HTTP response and an error log – and a high network latency metric. It also contains an operation relationship: the `Frontend_ErrorHandler` operation is called by the `Frontend_Open` operation.

This example shows how different penalties can be set for partial interpretation based on how heavily they should influence the rule learning process. This is essential, as real-world data usually contains noise, either from measurement errors or fault mislabeling. The example also shows how all kinds of multimodal observability data can be used to give context to the fault slices, allowing for the generation of more accurate diagnostic rules.

## V. Evaluation

The Online Boutique dataset was used for the evaluation of the developed framework. This dataset was chosen for its use

of traces, metrics, and logs, as well as its previous usage for benchmarking contemporary diagnostic tools [9]. The dataset was extracted from a real, commonly employed benchmark system for microservice research and development.

Online Boutique [10] is a microservices demo application created by Google to demonstrate their cloud-native tools and services, focusing on Kubernetes and Google Cloud Operations. This application is built as an online store; the primary user journey in the application consists of the user selecting an item, adding it to their cart, and making the purchase.

### A. Dataset

The dataset contains several kinds of faults injected into several different services. The following fault modes are used:

**CPU contention** occurs when multiple services are competing for CPU resources, leading to decreased performance because the CPU is overloaded with tasks.

**CPU consumption** means that a single service consumes an excessive amount of CPU resources, leading to performance degradation.

**Network delay** introduces latency in network communication between services. It causes delays in data transmission, leading to slower response times and timeouts.

**Exception** involves unexpected errors or exceptions occurring within a service's execution. This causes the service to crash or behave unpredictably.

**Return** refers to a service returning with unexpected behavior, which can be due to logic errors, incorrect configuration, or data corruption, affecting the functionality of dependent services.

The first two faults represent node-level performance issues in a system. *Network delay* is intended to show faults in the communication network of the system. *Exception* and *Return* faults are application-level faults, which resemble configuration issues and software bugs. By using these five fault modes, the approach can be tested regarding its viability in diagnosing many areas of network, infrastructure, and application-level faults while minimizing the complexity of the results.

### B. Results

Inductive rule extraction was performed on the training dataset from the Online Boutique application. This yielded 13 service-specific rules and 4 general rules. There was significant variety in the execution times of the ILASP programs for specific services, ranging from around 10 to 100 seconds, depending on the complexity of the service and the examples.

The diagnostic evaluation was done by taking each fault injection window (the timeslice where a specific fault is active) in the dataset as the input for fault diagnosis. Specific diagnostic rules were learned for each service in the system, while general diagnostic rules were aggregated from the results of multiple services.

The results were evaluated using the *Accuracy@1 (A@1)* – also known as Top-1 Accuracy – and *Accuracy@3 (A@3)* statistics, commonly used to evaluate the performance of models in classification tasks, such as recommender systems and multi-class classification problems.

Table I
RESULTS OF THE ONLINE BOUTIQUE BENCHMARK. COMPARISON DATA FROM NEZHA [9]

| Approach | A@1 | A@3 |
|---|---|---|
| MicroScope [11] | 12.5 | 41.07 |
| MicroRCA [12] | 16.07 | 62.5 |
| SBLD [13] | 19.64 | 23.21 |
| LogFaultFlagger [14] | 19.64 | 21.42 |
| MicroRank [17] | 41.07 | 48.21 |
| TraceAnomaly [15] | 30.35 | 33.92 |
| PDiagnose [16] | 41.07 | 73.21 |
| Nezha [9] | 92.86 | 96.43 |
| **Presented Solution** | **37.5** | **50** |

*Accuracy@1* measures the percentage of test cases where the correct answer was the top prediction. *Accuracy@3* measures the percentage of test cases where the top three predictions were correct. As the developed solution does not rank the possible diagnosis results but may present multiple answers, the correct *A@1* test cases are considered when the only given answer is correct. *A@3* test cases are considered correct if there are at most three possible diagnosis results, and one of them is correct.

After running the diagnostics framework on the benchmark dataset, the A@1 result was $37.5\,\%$, while the A@3 result was $50\,\%$. These results were compared to results [9] from other diagnostic and root cause analysis tools, using the same dataset, as seen in Table I. The developed solution is superior to MicroScope [11], MicroRCA [12], SBLD [13], LogFaultFlagger [14], and TraceAnomaly [15] in Top-1 accuracy. However, the results are slightly inferior to PDiagnose [16], a multimodal approach. Nezha [9], a novel multimodal approach, shows a significant accuracy difference between the solution presented in this paper, but also to other state-of-the-art approaches. It is clear that there is a great room for further development. However, compared to the imperative methodologies commonly used by other approaches, the declarative, logic-based approach presented here allows for enhanced scalability and flexibility. These results prove that the developed framework has potential, even in this early state.

### C. Scalability

The time required to solve an ILP problem depends heavily on the size of the search space and the complexity of the background knowledge. For example, choice rules generate multiple models, each of which has to be evaluated. The number of examples also plays a smaller factor in the runtime; however, this was barely noticeable at the scale of the benchmark datasets used in this evaluation. The main concern in terms of scalability is increasing the complexity of the mode declarations. Using more types of variables and constants in these literals leads to an increase in the hypothesis space.

In the presented solution, the background knowledge does not affect the process since the program does not include choice rules. However, to decrease the number of constant types in the head and body declarations, we chose to decompose and extract rules based on services. This allowed for shorter execution times, facilitating rapid prototyping with different models and configurations necessary for research.

## VI. Conclusion

This paper aimed to address two main challenges concerning fault diagnosis and observability in microservice systems: multimodal telemetry data representation and creating explainable diagnostic results. The following contributions were made to tackle these challenges.

We presented a metamodel and aggregate model capable of handling heterogeneous observability data, integrating multiple types of observability data (metrics, logs, and traces) to facilitate a comprehensive view of system behavior and enable further analysis. Leveraging the power of the created metamodel, a modular diagnostic framework using ASP integrates the processing of telemetry data, qualitative model extraction, and the application of diagnostic rules, enabling precise and interpretable diagnostics in microservice environments.

To automate diagnostic rule extraction, a novel solution was presented that uses ILP to learn rules from historical data. This technique is able to systematically derive diagnostic rules by analyzing fault-containing datasets, enabling the discovery of complex relationships and causalities.

The developed framework was evaluated using a fault-injected dataset based on Google's Online Boutique system. The solution was able to outperform several contemporary diagnostic tools, showing the potential for accurate, interpretable diagnostics in microservice systems. The developed models and framework lays the groundwork for further research in the field of observability and fault diagnosis.

### *Future Work*

Other applications of the created metamodel, model extraction, and inductive reasoning framework should be explored. The developed approaches could have further uses ranging from auto-scaling, fulfilling extra-functional requirements, and even analyzing user behavior.

Improving the qualitative diagnostic process could yield more robust and accurate results. The use of clustering algorithms for metrics, qualitative knowledge discovery [18], and incorporating quantitative elements should be considered. Employing error propagation analysis (EPA) could enable the diagnosis of more complex faults [19].

Using a microservice test environment would allow for evaluating the solution and further improvements over a more controllable and diverse set of data.

## References


[1] S. Wells, *Enabling Microservice Success*. O'Reilly, 2024, ISBN: 9781098130794.

[2] L. Giamattei, A. Guerriero, R. Pietrantuono, *et al.*, "Monitoring tools for devops and microservices: A systematic grey literature review," *Journal of Systems and Software*, vol. 208, p. 111 906, 2024, ISSN: 0164–1212. DOI: https://doi.org/10.1016/j.jss.2023.111906.

[3] *Cloud native glossary: Observability*, https://glossary.cncf.io/observability/, Accessed: 2024-10-22.

[4] M. Usman, S. Ferlin, A. Brunstrom, and J. Taheri, "A survey on observability of distributed edge & container-based microservices," *IEEE Access*, vol. 10, pp. 86 904–86 919, 2022. DOI: 10.1109/ACCESS.2022.3193102.

[5] S. Zhang, S. Xia, W. Fan, *et al.*, *Failure diagnosis in microservice systems: A comprehensive survey and analysis*, 2024. arXiv: 2407.01710. [Online]. Available: https://arxiv.org/abs/2407.01710.

[6] V. Lifschitz, "What is answer set programming?" In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, ser. AAAI'08, Chicago, Illinois: AAAI Press, 2008, pp. 1594–1597, ISBN: 9781577353683.

[7] M. Law, A. Russo, and K. Broda, "Inductive learning of answer set programs," in *Proceedings of the 14th European Conference on Logics in Artificial Intelligence - Volume 8761*, Berlin, Heidelberg: Springer-Verlag, 2014, pp. 311–325. DOI: 10.1007/978-3-319-11558-0_22.

[8] S. Muggleton and L. de Raedt, "Inductive logic programming: Theory and methods," *The Journal of Logic Programming*, vol. 19–20, pp. 629–679, 1994, Special Issue: Ten Years of Logic Programming. DOI: https://doi.org/10.1016/0743-1066(94)90035-3.

[9] G. Yu, P. Chen, Y. Li, H. Chen, X. Li, and Z. Zheng, "Nezha: Interpretable fine-grained root causes analysis for microservices on multi-modal observability data," in *ESEC/FSE 2023*, ACM, 2023.

[10] Google, *Online boutique application*, https://github.com/GoogleCloudPlatform/microservices-demo, Accessed: 2024-10-23.

[11] J. Lin, P. Chen, and Z. Zheng, "Microscope: Pinpoint performance issues with causal graphs in micro-service environments," in *Service-Oriented Computing*, C. Pahl, M. Vukovic, J. Yin, and Q. Yu, Eds., Cham: Springer International Publishing, 2018, pp. 3–20, ISBN: 978-3-030-03596-9.

[12] L. Wu, J. Tordsson, E. Elmroth, and O. Kao, "Microrca: Root cause localization of performance issues in microservices," in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–9. DOI: 10.1109/NOMS47738.2020.9110353.

[13] C. M. Rosenberg and L. Moonen, "Spectrum-based log diagnosis," in *Proceedings of the 14th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, ser. ESEM '20, Bari, Italy: Association for Computing Machinery, 2020, ISBN: 9781450375801. DOI: 10.1145/3382494.3410684.

[14] A. Amar and P. C. Rigby, "Mining historical test logs to predict bugs and localize faults in the test logs," in *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, 2019, pp. 140–151. DOI: 10.1109/ICSE.2019.00031.

[15] P. Liu, H. Xu, Q. Ouyang, *et al.*, "Unsupervised detection of microservice trace anomalies through service-level deep bayesian networks," in *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*, 2020, pp. 48–58. DOI: 10.1109/ISSRE5003.2020.00014.

[16] C. Hou, T. Jia, Y. Wu, Y. Li, and J. Han, "Diagnosing performance issues in microservices with heterogeneous data source," in *2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*, 2021, pp. 493–500. DOI: 10.1109/ISPA-BDCloud-SocialCom-SustainCom52081.2021.00074.

[17] G. Yu, P. Chen, H. Chen, *et al.*, "Microrank: End-to-end latency issue localization with extended spectrum analysis in microservice environments," in *Proceedings of the Web Conference 2021*, ser. WWW '21, Ljubljana, Slovenia: Association for Computing Machinery, 2021, pp. 3087–3098, ISBN: 9781450383127. DOI: 10.1145/3442381.3449905.

[18] G. Kern-Isberner, M. Thimm, and M. Finthammer, "Qualitative knowledge discovery," in *Semantics in Data and Knowledge Bases*, K.-D. Schewe and B. Thalheim, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 77–102, ISBN: 978-3-540-88594-8.

[19] M. Tarnay, *Managing operational uncertainties in deployed systems using logic reasoning*, https://tdk.bme.hu/conference/VIK/2023/sessions/distsys/paper/Mukodesi-Bizonytalansagok-Kezelese-Elosztott, 2023.

# Effects of Noisy Occupancy Data on an Auction-based Intelligent Parking Assignment

Levente Alekszejenkó, Tadeusz Dobrowiecki
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {alelevente, dobrowiecki}@mit.bme.hu

***Abstract*—Smartphones and cloud services can provide sophisticated parking assignment in modern intelligent cities. These solutions aim to guide drivers to vacant parking lots near their destination, reducing the necessary cruising for parking. Hence, they can smoothen the traffic flow and mitigate harmful emissions. Moreover, auction-based assignment can also dynamically optimize the actual parking prices, benefiting drivers, and parking lot operators.**

**To operate such a system, we shall know the actual occupancy of the supervised parking lots. This data can come from various sources, e.g., crowdsourcing, parking lot operators, or third-party data providers. Sensing and fusing these records might lead to inaccurate input for the assignment method. In this paper, we analyze the impact of such noise on the performance of an auction-based parking lot assignment system. The results indicate that accurate information is crucial for perfect operation, but current state-of-the-art solutions provide sufficient input to benefit from the system.**

***Index Terms*—auctions, noisy data, parking assignment**

## I. Introduction

The advent of autonomous vehicles (AVs) could affect the appearance and use of future cities. Regardless of their ownership model, AVs will reduce parking demand; hence, nowadays parking facilities can be reconstructed into attractive areas, e.g., parks, playgrounds or community buildings [1]. However, recent financial processes negatively influenced the automotive industry, i.e., overall car production dropped to a decade-old level [2]. Consequently, car manufacturers have to decrease their investments in research and development, certainly postponing the introduction of AVs.

Moreover, traffic congestion is surprisingly expensive; for example, a single congestion event costs more than 50.000€ to the society [3], while a driver spent 110 extra hours in 2024 in Budapest, Hungary, according to TomTom's TrafficIndex[1] due to rush hours. In rush hours, many drivers cruise for parking [4], slowing down traffic and adding to congestion. Without AVs that minimize parking demand, we might be able to optimize parking usage to eliminate cruising and mitigate the harmful effects of traffic congestion.

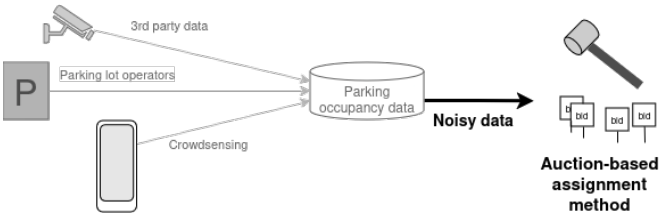



Fig. 1: System architecture of the assumed system.

A demand-based, market-priced curbside parking could effectively alleviate cruising for parking [4]. Therefore, we assume that a municipality deploys an intelligent parking assignment and pricing system based on auctions. However, this system requires constant monitoring of the available parking spaces. These monitoring data can come from various independent sources, including parking lot operators and video surveillance cameras. Moreover, drivers can also report the occupancy status of parking lots using a crowd-sourcing application; see Fig. 1. Naturally, combining these originally inaccurate observations leads to a noisy input for the auction-based assignment system, possibly impeding its performance. Thus, we will get two sets of free parking spaces. There is a set of *real* free parking spaces on the road, and there is another set of free parking spaces *represented* in the auction mechanism. If we had perfect information, the set of real and represented free parking lots would coincide. Considering a real-world free parking lot occupancy detector, there might be misclassifications; for example, it can mark an occupied parking lot free, or a free parking lot as occupied. Furthermore, the validity of the measurement data can also expire (someone occupies a parking lot or leaves it after the measurement); consequently, the set of free parking lots might not be identical. The problem addressed here is similar to an auction-based task allocation system in which we cannot know whether a task exists in the real world or whether all the real world tasks are represented on the actions. To our knowledge, no one has investigated yet how such noisy inputs affect an auction-based parking assignment system. In this paper, our aim is to fill this research gap. By microscopic traffic simulation of a central business district, we will assess how noisy observations

This research was supported by project no. EKÖP-24-3-BME-319, implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the EKÖP-24-3 funding scheme; and by project no. TKP2021-EGA-02, implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the TKP2021-EGA funding scheme.

[1]https://www.tomtom.com/traffic-index/budapest-traffic/ (accessed: 24/01/2025)

reduce the traffic smoothing capabilities of such a parking assignment system. We will also check how successful parking reservations are when the assignment system does not receive perfect information. Finally, we also investigate how noisy observations influence the pricing mechanism and what the individual economic outcomes of operating such a system are.

## II. Related Works

As [4] concludes, parking pricing shall reflect market demand in a temporal-spatial way. For example, in San Francisco, USA, the *SFPark* project implements a demand-based pricing scheme. That project aims to keep the occupancy ratio in the 60-80% range. If the occupancy rate does not reach this level in particular parking spaces, the parking prices decrease and if they exceed 80%, the municipality increases the parking fee. In the *SFPark* project, price recalculation follows a strict rule; hence, it is a reactive system that requires many iterations to reach the target state [5].

Auction-based calculations can provide an optimal solution incomparably faster than *SFPark*. For example, a sealed bid Vickrey-Clarke-Groves mechanism only needs to compute the final results without testing the actual prices in the real world [6]. Unfortunately, this method is not budget-balanced, but with a restriction, it can avoid running with deficits [7]. However, participant drivers shall still report their valuation of the parking lots to a central agent. It might pose a privacy threat, as it reveals information about a driver's destination or financial status. To avoid sharing these pieces of information, instead of sealed bid auctions, one could implement an ascending auction mechanism for multiple items [8], as it only requires a single bit of information, whether or not a driver is willing to pay a particular price for a parking lot. The local greedy bidding (LGB) strategy, which means that each participant bids for items that maximize its utility function, can obtain, e.g., at most 1 item in simultaneously running independent ascending bid (online) auctions (SIA) [9]. Similarly to [10], we implemented an ascending bid auction system for parking assignment [11].

Hypothetically, individual parking operators can run the algorithms necessary for these independent auctions. In practice, even simpler intelligent parking solutions require tens of millions of dollars of infrastructural development [12]. Considering that municipalities cannot afford such investments, we assume that auctions run on remote servers. In addition to the bids of the drivers, the operation of this server requires the knowledge of the number of available parking lots. Recent studies have introduced various vision-based parking lot occupancy detectors. Some of these state-of-the-art solutions require a stationary surveillance camera on a high vantage point, facing a larger parking facility. A video-based solution, QuickSpot [13], can achieve 97.8-99.2% average detection accuracy, and a single-frame-based detection system can reach 89-95% accuracy [14]. Similarly to these results, a third stationary camera-based solution in [15] provides a detection rate of 96.43-100%.

Despite stationary surveillance cameras, parking occupancy detector solutions can also use images from dashcams in moving vehicles. The system described in [16] recognizes vacant parking spaces with 97% recall, and 86% classification accuracy. Besides individual recognition systems, a vision-based vehicular crowdsensing system of [17] provides a 83% average accuracy.

Crowdsensing systems usually run auction methods for orchestration [18], [19], or even to optimize the accuracy of the obtained data [20]. In this paper, we assume that the noise of the input data is independent of the auction mechanism; hence the system cannot enhance its accuracy. Moreover, there might be an uncertainty in the number of bidders, their valuation of the objects, the number of objects in auctions [21]–[23]. We note, that in our problem, the auction mechanism lacks these uncertainties as in our model the list of real free parking lots and the represented parking lots could be misaligned independently of the auction mechanism.

In the following, we will investigate how inaccurate parking occupancy detection influences the performance of an auction-based parking lot assignment system.

## III. The Auction Method

Following [9], we implemented[2] an SIA method for parking assignment. In this approach, dedicated auctions control the occupancy of every free parking space. Vehicles bid for a parking space in these auctions following the LGB strategy. Naturally, a driver aims to minimize the combination of required walking and parking fees. Hence, denoting $d_{i,j}$ the driving distance (which is an overestimate of the walking distance) from the original destination of vehicle $j$ to parking lot $i$, let $d_{j,max} = \max_i d_{i,j}$, $\rho_i$ the current hourly price of parking lot $i$ and $\rho_{max} = \max_i \rho_i$, then the $c_{i,j}$ parking cost of vehicle $j$ at parking $i$ is calculated as:

$$c_{i,j} = 0.5 \cdot \frac{d_{i,j}}{d_{j,max}} + 0.5 \cdot \frac{\rho_i}{\rho_{max}}. \tag{1}$$

As $c_j \in [0.0, 1.0]$, and the vehicles aim to minimize parking costs, to maximize their utility, they shall maximize $1 - c_{i,j}$ over $i$. This yields, vehicle $j$ shall prefer the $\mathcal{P}_j = \arg\max_i(1 - c_{i,j})$ auction.

To ensure that a vehicle gets assigned to at most one parking lot, it shall only bid on an auction if it is overbid in every other auction. In our experiments, bidders have a 2.43 €[3] upper bound of hourly parking price that they are willing to pay for a parking lot.

## IV. Simulations

To test the described auction method with inaccurate input data, we created a simulation scenario of an abstract central business district (CBD). The road network of this CBD consists of $6 \times 6$ perpendicular road segments which are 100 m

[2]Source codes are available on Github: https://github.com/alelevente/bprof_mi_multiagent.

[3]2.43 €was equivalent to 1000 HUF (Hungarian Forint) at the writing of this paper, on 11th of November, 2024.

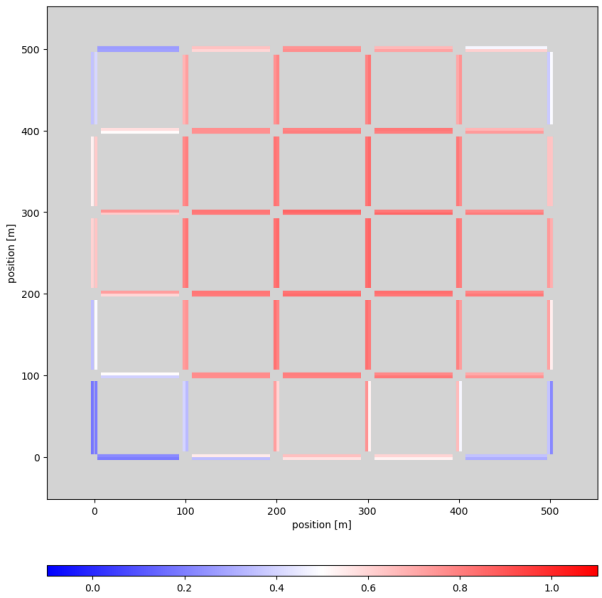



Fig. 2: Average parking lot occupancy rate in the uncoordinated (unc.) case. As turning right is generally easier in a right-hand side driving system, the top right corner of the road network is slightly more preferred than other corners of the road network.

long, see Figure 2. There are $t = 15$ parking spaces on the curb on both sides of each road.

We assume people come to the CBD, do their business there, and leave after a short stay (i.e., evenly distributed between 20 and 120 minutes). We assume that the demand for parking in the central part of the CBD is higher than in the outskirts. Therefore, we generated traffic for the simulation aiming at the central part of the grid 9 times more likely than the outer parts. Traffic demand decreases evenly as we move farther away from the center. We also assume that parking lot operators have already set parking pricing accordingly, i.e., central parking lots cost 1.1 €/h, while the most distant parking lots have a 0.12 €/h hourly price. The auction method uses these prices as starting prices.

We ran microscopic traffic simulations of this scenario in Eclipse SUMO [24]. In the simulation, we placed `ParkingLotRerouters` at the end of each road segment. These rerouters redirect vehicles to the nearest parking lots, ensuring that they find a vacant parking space. With induction loop-type traffic detectors placed at 60th m of each road segment, we measured the traffic flow in the simulation in vehicles/h ([veh/h]) unit. We also acquire the exact $n_i$ number of each $i$ free parking lot from Eclipse SUMO. We added different amounts of discrete noise $\delta$ to these values to achieve a predefined detection accuracy rate. To ensure experimenting with an expected target level of accuracy $g$, we sampled distortions from a uniform integer distribution $\mathcal{U}\big(0, 2(t-tg)\big)$. Hence, the $\tilde{n}_i$ distorted capacity of parking lot $i$ will be:

$$\tilde{n}_i = n_i \pm \delta. \tag{2}$$

In this way, (2) defines an unbiased noise model with an expected accuracy of $g$.

In addition to simulating the traditional *uncoordinated* (unc.) parking search method, we also experimented with auction-based parking assignment. In the latter case, every 15th seconds of the simulation, we run auctions for the parking spaces. Vehicles that have departed since the last auction runs shall participate in these auctions to reserve a parking space. Then, we instruct Eclipse SUMO to reroute the cars to their assigned parking lots. However, some vehicles (approximately 1.5%) complete their route faster than 15 s; consequently, we cannot reroute them according to the reservation mechanism. In addition, inaccurate parking lot occupancy data can lead to overreserved parking lots. In this case, if the demand for a parking lot exceeds its capacity, excess vehicles shall find another vacant parking space via the traditional cruising method as a fall-back algorithm. To ensure that users of the auction-based parking lot assignment mechanism do not lose money, they shall only pay the auctioned parking fees if the parking reservation is successful. Otherwise, they will pay the original price of the parking space they managed to occupy. To handle stochasticity, we repeated each simulation for 10 times.

## V. Results of the Inaccurate Parking Occupany

In the following, we analyze the results obtained from the simulations. Firstly, we check how the auction method influences macroscopic traffic and whether it can mitigate congestion by improving the flow of traffic. Secondly, we present what an individual driver would experience using the auction-based parking assignment method. Finally, we check how inaccurate data influences the auction method.

### A. Impact on Macroscopic Traffic

The auction-based intelligent parking assignment system improves traffic flow; see Fig. 3. The traditional, uncoordinated (unc.) system provides the lowest average traffic flow, and the perfect information achieves the highest. However, the amount of non-zero noise in the parking occupancy data has a negligible effect on macroscopic traffic flow.

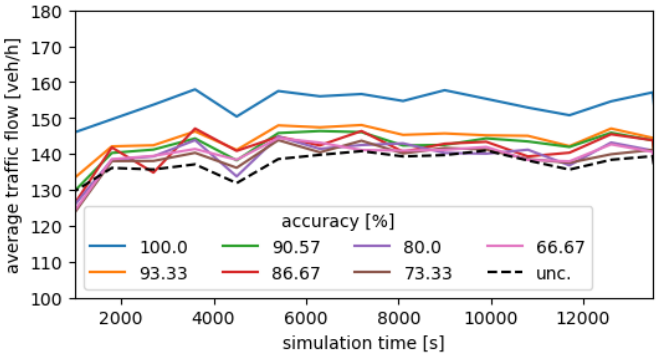



Fig. 3: Traffic flow with different accuracy levels.

On the other hand, parking habits change significantly due to the auction method. Previously, drivers solely preferred to park close to their destination, resulting in a high occupancy rate in the central part of the simulated road network; see Fig. 2. In addition to traditional walking distance, auction-based assignment also optimizes parking costs using the (1) cost function when bidding with the LGB strategy. That leads vehicles to cheaper, farther away parking spaces. Fig. 4 groups parking lots by their Euclidean distance from the center of the road network. While the traditional uncoordinated (unc.) system provides a larger demand in the center of the road network, the novel system fills the more distant parking lots instead. When parking lot occupancy data are less accurate, more and more drivers must revert to the traditional parking cruising method by deviating from new parking habits.

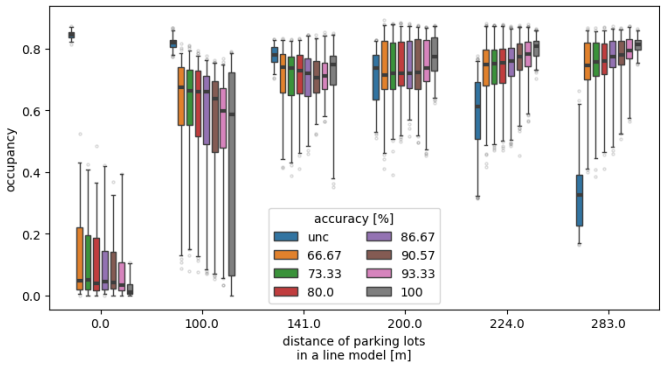



Fig. 4: Parking occupancy rates at different distances from the center with different accuracy levels. Uncoordinated (unc.) case shows a completely different parking habit compared to the auction-based parking assignment system.

### B. Impact on Microscopic Traffic

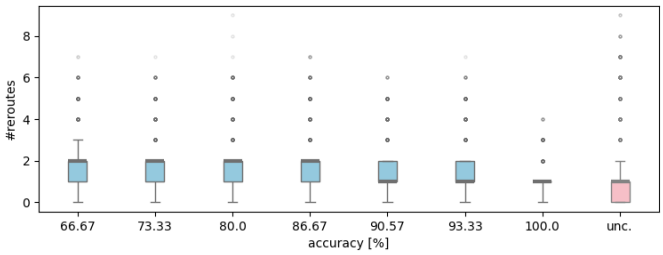



Fig. 5: Number of reroutes in Eclipse SUMO with different accuracy levels. Medians significantly decrease at ≈90% accuracy.

The assignment system influences drivers in two ways. We call the first *number of reroutes*. There are two reasons to experience a rerouting event. Firstly, when a vehicle gets an assigned parking lot from the system, it changes its target to that parking space, similar to a route recalculation of a modern navigation system. Secondly, if the vehicle cannot park in a parking lot, it keeps cruising and looking for a vacant

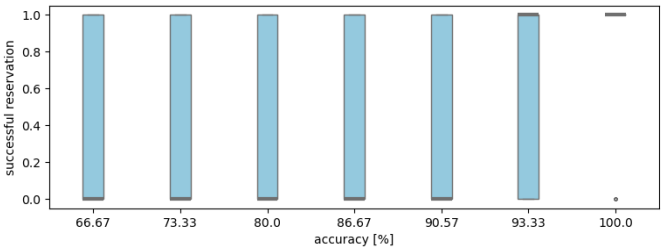



Fig. 6: Rate of successfully occupied parking lots offered by the auction method. Medians significantly increase at ≈93% accuracy.

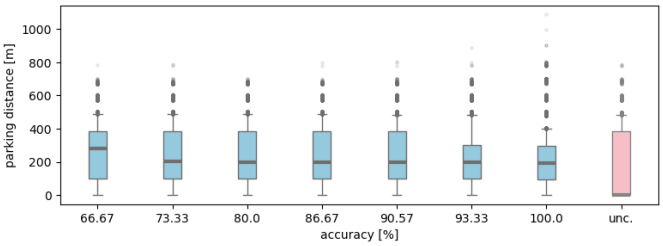



Fig. 7: Driving distance between the occupied parking lot and a driver's original destination.

parking space. It also triggers a reroute event if a car passes an intersection during this cruising. Fig. 5 shows the number of reroutes. The traditional uncoordinated (unc.) system can even force vehicles to cruise on up to 9 streets until finding a free parking space. If the auction system has perfect input data, it naturally needs one (recalculating route-type) reroute. Outliers are due to the mechanism mentioned in section IV. With decreasing accuracy, the number of reroutes increases. Its reason is that the noisier input data of the assignment system reduces the success rate of occupying the assigned parking lot; see Fig. 6. Consequently, most vehicles cannot occupy the parking space reserved by the auction method below ≈93% of accuracy.

The second effect of the auction-based parking assignment system is the distance between the parking lot and the driver's original destination, called *parking distance*. As walking distance is less than equal to the driving distance (obtainable from the Eclipse SUMO simulation), it gives an upper estimate of the walking distance. Fig. 7 shows a decreasing trend in the parking distance as accuracy increases. At higher accuracy levels, the deviation of the parking distance distribution also decreases. Fig. 5 can explain these observations, as more accurate input data decrease the number of reroutes, the length of cruising for parking, and, according to Fig. 6, help drivers occupy their assigned parking lot. Naturally, as the uncoordinated mechanism has a different goal function that only minimizes the parking distance, it mainly achieves shorter parking distances than the auction mechanism.

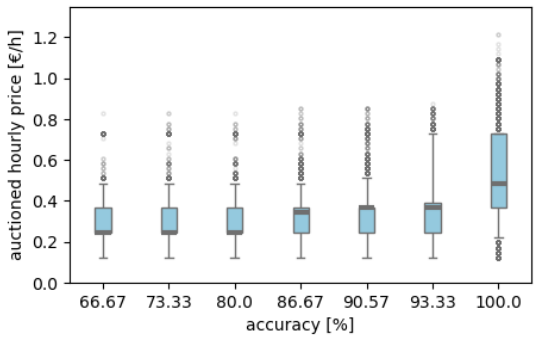



Fig. 8: Hourly parking prices provided by the auction-based assignment method.

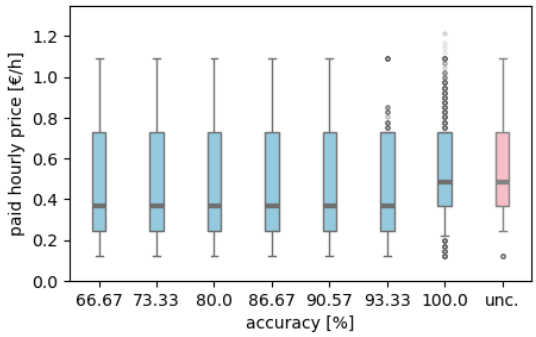



Fig. 9: Actually paid hourly parking prices.

### C. Impact on Auction Method

Finally, we analyze how imperfect data affects the auction method itself. To this end, we check the resulting auctioned prices (determined by the number of bids) that reflect the competition between vehicles for parking spaces. According to Fig. 8, prices increase as the accuracy of the input data increases. It indicates that the competition decreases if the number of free parking spots is overestimated due to the (implied) larger supply. On the other hand, competition also decreases if we underestimate the number of free parking spots because vehicles would choose other parking alternatives rather than competing. Consequently, exact knowledge improves the competition, resulting in higher auctioned hourly parking pricing, converging to the market value of the curbside parking pricing.

As drivers are only required to pay the auctioned parking price if they can occupy the reserved parking space, the actually paid hourly prices can differ from the auctioned prices. Fig. 9 summarizes the actual hourly prices. Due to the changed parking habits, the auction mechanism reduces the users' parking costs compared to the uncoordinated case. We can also see that, below the accuracy of $\approx$93%, any further inaccuracy has no significant effect on the paid prices.

## VI. Conclusion

In this paper, we analyzed how inaccurate parking lot occupancy data can affect an SIA auction-based intelligent parking lot assignment method. Simulation results indicate that the system provides most of its merits with perfect information. However, approximately 93% accuracy is enough to experience the positive effects of the auction-based parking assignment system. According to our literature overview, state-of-the-art camera-based parking occupancy detectors can provide the expected data accuracy to operate such an intelligent parking coordination system. With further investment in infrastructure, as in the case of the SFPark project [12], more precise data would be available to enjoy all the benefits such a system can provide.

Furthermore, following the analogy of section I, the presented problem is a concrete case of an auction-based task assignment mechanism without exact knowledge of the existence of tasks and without any guarantee of the completeness of the task list. In other similar scenarios, typically in search and rescue missions, an imperfect list of tasks might also hinder the system performance. Therefore, further research is needed to exploit all the merits of auction-based multi-agent systems.

## References

[1] E. González-González, S. Nogués, and D. Stead, "Parking futures: Preparing european cities for the advent of automated vehicles," *Land Use Policy*, vol. 91, p. 104 010, 2020, ISSN: 0264-8377. DOI: 10.1016/j.landusepol.2019.05.029.

[2] J. Probst. "The decline of car manufacturing is hurting the German economy," Recruitonomics. (2023), [Online]. Available: recruitonomics.com/the-decline-of-car-manufacturing-is-hurting-the-german-economy/ (visited on 11/25/2024).

[3] E. Struyf, C. Sys, E. Van de Voorde, and T. Vanelslander, "Calculating the cost of congestion to society: A case study application to Flanders," *Research in Transportation Business Management*, vol. 44, p. 100 573, 2022, Challenges and solutions for current freight transport and logistics, ISSN: 2210-5395. DOI: 10.1016/j.rtbm.2020.100573.

[4] D. Shoup, "Pricing curb parking," *Transportation Research Part A: Policy and Practice*, vol. 154, pp. 399–412, 2021, ISSN: 0965-8564. DOI: 10.1016/j.tra.2021.04.012.

[5] E. Inci, "A review of the economics of parking," *Economics of Transportation*, vol. 4, no. 1, pp. 50–63, 2015, Special Issue on Collective Contributions in the Honor of Richard Arnott, ISSN: 2212-0122. DOI: 10.1016/j.ecotra.2014.11.001.

[6] P. Cramton, Y. Shoham, and R. Steinberg, *Combinatorial Auctions*. MIT Press, Boston, 2006.

[7] M. Cheng, E. Inci, S. X. Xu, and Y. Zhai, "A novel mechanism for private parking space sharing: The vickrey–clarke–groves auction with scale control," *Transportation Research Part C: Emerging Technologies*, vol. 150, p. 104 106, 2023, ISSN: 0968-090X. DOI: 10.1016/j.trc.2023.104106.

[8] G. Demange, D. Gale, and M. Sotomayor, "Multi-item auctions," *Journal of Political Economy*, vol. 94, no. 4, pp. 863–872, 1986, ISSN: 00223808, 1537534X.

[9] V. Bansal and R. Garg, "Simultaneous independent online auctions with discrete bid increments," *Electronic Commerce Research*, vol. 5, pp. 181–201, 2005. DOI: 10.1007/s10660-005-6156-1.

[10] S. R. Rizvi, S. Zehra, and S. Olariu, "MAPark: A multi-agent auction-based parking system in internet of things," *IEEE Intelligent Transportation Systems Magazine*, vol. 13, no. 4, pp. 104–115, 2021. DOI: 10.1109/MITS.2019.2953524.

[11] L. Alekszejenkó and T. Dobrowiecki, "Auction based parking lot assignment and empty cruising limitation of privately owned autonomous vehicles in a simple city model," in *2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops), Nagoya, Japan*, 2021, pp. 335–341. DOI: 10.1109/IVWorkshops54471.2021.9669251.

[12] T. Lin, H. Rivano, and F. Le Mouël, "A survey of smart parking solutions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 12, pp. 3229–3253, 2017. DOI: 10.1109/TITS.2017.2685143.

[13] E. Màrmol and X. Sevillano, "QuickSpot: a video analytics solution for on-street vacant parking sport detection," *Multimedia Tools and Applications*, vol. 75, pp. 17 711–17 743, 2016. DOI: 10.1007/s11042-016-3773-8.

[14] J. Nyambal and R. Klein, "Automated parking space detection using convolutional neural networks," in *2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech)*, 2017, pp. 1–6. DOI: 10.1109/RoboMech.2017.8261114.

[15] K. Shaaban and H. Tounsi, "Parking space detection system using video images," *Transportation Research Record*, vol. 2537, no. 1, pp. 137–147, 2015. DOI: 10.3141/2537-15.

[16] M.-C. Wu and M.-C. Yeh, "Early detection of vacant parking spaces using dashcam videos," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 9613–9618, Jul. 2019. DOI: 10.1609/aaai.v33i01.33019613.

[17] T. Higuchi and K. Oguchi, "Monitoring live parking availability by vision-based vehicular crowdsensing," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–5. DOI: 10.1109/GLOBECOM42002.2020.9348221.

[18] A. I. Middya and S. Ray, "Truthful double auction based incentive mechanism for participatory sensing systems," *Peer-to-Peer Networking and Applications*, vol. 17, pp. 2137–2166, 2024. DOI: 10.1007/s12083-024-01681-3.

[19] Z. Feng, Y. Zhu, Q. Zhang, L. M. Ni, and A. V. Vasilakos, "Trac: Truthful auction for location-aware collaborative sensing in mobile crowdsourcing," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, 2014, pp. 1231–1239. DOI: 10.1109/INFOCOM.2014.6848055.

[20] T. Ni, Z. Chen, G. Xu, S. Zhang, and H. Zhong, "Differentially private double auction with reliability-aware in mobile crowd sensing," *Ad Hoc Networks*, vol. 114, p. 102 450, 2021, ISSN: 1570-8705. DOI: 10.1016/j.adhoc.2021.102450.

[21] S. Lauermann and A. Speit, "Bidding in common-value auctions with an unknown number of competitors," *Econometrica*, vol. 91, no. 2, pp. 493–527, 2023. DOI: 10.3982/ECTA17793.

[22] D. Aycinena and L. Rentschler, "Auctions with endogenous participation and an uncertain number of bidders: Experimental evidence," *Experimental Economics*, vol. 21, pp. 924–949, 2018. DOI: 10.1007/s10683-017-9558-8.

[23] S. Fatima, M. Wooldridge, and N. R. Jennings, "Sequential auctions in uncertain information settings," in *Agent-Mediated Electronic Commerce and Trading Agent Design and Analysis*, J. Collins, P. Faratin, S. Parsons, *et al.*, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 16–29, ISBN: 978-3-540-88713-3.

[24] P. Alvarez Lopez, M. Behrisch, L. Bieker-Walz, *et al.*, "Microscopic traffic simulation using SUMO," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, Nov. 2018, pp. 2575–2582.

# Localization of the Lungs on PA chest X-ray images using deep CNN-s

Adam Tumay, Daniel Hadhazi, Gabor Hullam
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {tumay, hadhazi, gabor.hullam}@mit.bme.hu

***Abstract*—Cardiovascular diseases (CVDs) are among the leading causes of deaths worldwide. Most of these diseases are difficult to diagnose in time; however, many of the lesions connected to them can be seen on PA chest radiographs. Although this is not the primary modality for the identification of such lesions, it is the most commonly used in daily medical practice. In a previous research we proposed a method for the accurate and robust segmentation of the heart. This method uses a classical image processing algorithm to localize the lung area as a preprocessing step. In this paper, we present a deep neural approach to tackle this problem aiming for a more accurate segmentation. This is necessary, as a more accurate lung localization model may improve the performance of our heart segmentation model, furthermore if both accurate and robust heart and lung segmentations are available, it is possible to calculate the cardiothoracic ratio (CTR), which is a common metric used to decide whether a patient has cardiomegaly. In our research we train and assess a modified UNet architecture on the Japanese Radiological Society (JSRT), Montgomery and Shenzen datasets and qualitatively evaluate the model on a private dataset coming from daily medical practice. The last step is important, as publicly available datasets often lack the variance and come from a different distribution than the data acquired in daily medical practice.**

***Index Terms*—neural networks, deep learning, lung segmentation, medical image processing, UNet**

## I. Introduction

Nowadays, cardiovascular diseases (CVDs [1]) are among the leading causes of deaths worldwide. Most of these diseases are difficult to diagnose in time as symptoms can only be observed at more advanced stages where the probability of successful treatment drops significantly while mortality rates increase. However, many of the lesions connected to these diseases can be observed on PA chest radiographs. Although this is not the primary modality to diagnose such diseases, it is among the most commonly used in daily radiological practice, thus automatic filtering for them may come with many benefits. One of the diseases is Cardiomegaly, which correlates well with the patient's Cardiothoraic ratio (CTR [2]) - the quotient of the width of the heart and lung. To calculate this metric, it is necessary to segment the heart and lung regions of the image accurately. Furthermore, segmentations to said areas come with many other benefits such as filtering the input image to key areas as a preprocessing step, etc. This was the case in our previous work too [3], where we segmented the heart area through a model based deep learning method. We found that the accuracy and robustness of the algorithm largely improves if we crop the input image to the target area, by segmenting the lung using a traditional image processing method. In this paper we explore another option using pure deep learning to tackle the problem. While we found this approach insufficient to segment the heart area accurately [3], in this case more data is available (200 vs 700 training samples), coming from a more varied distribution (including cases with TB for example). Related works also show that this approach is able to generalize well for samples coming from the distribution of the training datasets (e.g. Gaál et al. [4] on JSRT with ∼250 samples), achieving quantitatively satisfactory results, thus we are also curious how our method performs qualitatively on a dataset coming from daily medical practice.

The research presented here was funded by the Josef Heim scholarship

## II. Related Works

The segmentation of the lungs is often a critical preprocessing step for many medical image processing algorithms working with PA chest X-ray images. The main approaches are either using traditional image processing algorithms as Reamaroon et al. [5] did using a Total Variation Active Contour filtering (TVAC) method, achieving a Dice score [6] (DSC) of $0.95 \pm 0.03$ on the Japanese Radiological Society [7] (JSRT) dataset, $0.96 \pm 0.03$ for the Montgomery County dataset, and $0.86 \pm 0.04$ on a dataset featuring critical care patients. While Candemir et al. [8] proposed using anatomical atlases with nonrigid registration achieving a DSC of $0.967 \pm 0.008$ on the JSRT and $0.960 \pm 0.018$ on the Montgomery dataset. The main approach used nowadays however, is training neural networks for the task on the data available instead of creating models based on expert knowledge. Most commonly architectures used for this task are some modifications of the UNet [9] architecture. Gaál et al. [4] segmented the lung and heart areas together using Attention U-Net Based Adversarial architectures achieving a dice score of $0.976 \pm 0.005$ on the JSRT dataset. Liu et al. [10] proposed adding residual connections to UNets for improving performance achieve a Dice score of $0.979$ and $0.977$ on the JSRT and Montgomery datasets respectively. What studies often lack however, is evaluating the trained models' performance on more diverse datasets featur-

DOI https://doi.org/10.3311/MINISY2025-006

ing lesions that make segmentations more difficult. Reamaroon et al. [5] did so with samples coming from critical condition patients measuring a significant drop in performance. Another relatively unexplored territory of the task is measuring the robustness of segmentations by taking the locality of the errors into account (by measuring the Hausdorff distance between the segmented and ground truth masks for example), as non-local errors can heavily affect the usability of the predictions.

## III. Datasets

For training and evaluating our models we used three publicly available datasets which were manually annotated with segmentation masks coming from experts. To qualitatively evaluate the model's generalization capability across datasets we used another private dataset coming from daily medical practice.

### A. JSRT dataset

The dataset coming from the Japanese Radiological Society [7] is one of the most common to be used in studies. It is annotated with both heart and lung masks. The dataset contains 247 analog radiographs, which we randomly partitioned into two disjoint sets: One containing 200 images for training, and another 47 for validation and tests. It is worth noting that the patients on the datasets are well-aligned and that no lesions are present which would make the segmentation task more difficult, by obscuring parts of the lung for example.

### B. Shenzen dataset

A dataset coming from the hospitals of Shenzen, China [11]. The dataset contains analog 663 PA Chest X-ray images in total, out of which 566 are annotated manually with lung masks, we only use these radiographs for training and evaluation. The images show great variety in terms of contrast, the positioning of patients as well as lesions visible in the lung area. In fact 287 out of the 566 annotated images are TB positive cases. Pediatric cases are also present. This dataset was also partitioned randomly into two disjoint subsets, a training set containing 400 images, and a test and validation set with 166 samples.

### C. Montgomery dataset

A dataset provided by the US National Institute of Health (NIH) [11], collected in Montgomery County, Maryland, USA. The images were acquired as part of a TB screening program, thus similar to the Shenzen dataset, some of the radiographs show more extensive lesions. The dataset consists of 138 annotated X-ray images in total, out of which 58 cases are TB positive. The lung areas are annotated manually, the left and right lungs are available on separate images. Similar to the previous datasets, we randomly partitioned the set into two disjoint sets of 100 and 38 elements for training and evaluation.

### D. Combined dataset

To train and evaluate our model quantitatively, we combine the previous three datasets into a single set, containing 700 training and 251 testing samples. This way we aim to create a dataset that is sufficiently diverse for the ANN to achieve good generalization ability on the background distribution of PA CXR images. To further help generalization, we implement data augmentation transformations on the training data, such as randomized affine transformations, where the image may be translated, rotated, and re-scaled before being passed to the network. In addition, high frequency noise (i.e. in terms of the rate of change of intensity values) coming from a white Gaussian process is also added to the images to prevent overfitting on such components. As a further defensive measure we also used blurring with 2 pixels as the sigma parameter of the 2d Gaussian weight function as a pre-processing step on all images to filter out high-frequency components as we found it improves the robustness of neural networks, while such features are unimportant in the context of localizing the lungs.

### E. Private dataset for qualitative evaluation

To see how the evaluated model performs on data in practice, we used a private dataset acquired in Hungarian hospitals during daily medical practice. The dataset consists of 23 digital radiographs, where some may show more extensive lesions such as cardiomegaly, the usage of pacemakers, asymmetric diaphragm etc. Annotations for the dataset are not available, only qualitative evaluation is possible. This is still important however, as while neural networks often generalize well for their training datasets, show poor generalization capabilities across datasets, seriously impairing their practical usability.

## IV. Proposed method

In our research we trained a classical UNet [9] architecture, using the training partition of the Combined dataset as training data. To evaluate our model we used Dice score to measure the overall quality of the predictions, as well as Hausdorff distance to check the method's robustness.

### A. Preprocessing

The format of the data available in the datasets was in the form of pre-normalized .png images, thus further normalization deemed unnecessary. In case of .DICOM inputs we would use a robust min-max normalization, taking the 5th and 95th percentiles of the image's sorted intensity values as the minimum and maximum. The intensities of the loaded .png images are re-scaled to be in the range of $[0, 1]$. The images are resized to a resolution of 384x384 pixels, and then previously mentioned preprocessing steps of applying affine transformations, Gaussian blur and noise are executed. The parameters of the affine transformation are a uniformly random rotation between $\pm 30$ degrees, a uniform translation between $\pm 38,4$ pixels independently on both coordinates, and uniform scaling with a scale factor between $0.8$ and $1.2$. The Gaussian kernel used for blurring has a size of 51x51 pixels, and a sigma

parameter of 2. The high frequency noise also comes from a Gaussian distribution with 0 as the expected value, and it's standard deviation is also Gaussian with a sigma parameter of 0.025. After applying the augmenting transformations (if necessary), the intensities of the image are clamped in the $[0, 1]$ range.

### B. Architecture

The architecture used for the neural network was a conventional UNet [9], interpreting the segmentation task as pixel-level classification. We use this architecture, as it is the most commonly used for such tasks, and generally performs well according to literature [12], [13]. A schematic outline of the architecture can be seen in Fig. 1. The network has one input channel containing the grayscale input image, and one output channel for the segmented lungs. (We do not differentiate between the left and right lungs, as not all datasets are annotated that way, furthermore it is not necessary for our purposes.) Since we interpret the task as classification of the pixels, the last layer of the network is a sigmoid function, to create probabilities from the predicted logits.

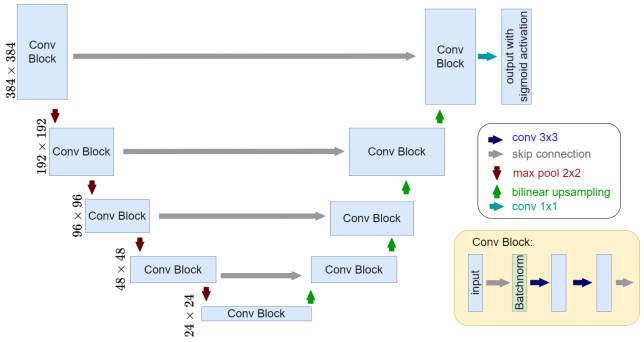



Fig. 1. A schematic outline of the UNet architecture. The descending part of the "U-shaped" architecture consists of consecutive pairs of convolutional blocks and max pooling layers, while the ascending part consists of pairs of convolutional blocks and bilinear upsampling layers. The descending and ascending branches of the network are connected with skip connections. Each convolutional block contains a batch normalization and two convolutional layers.

### C. Loss function

As we perform pixel level classification for the segmentation task, and there is only one class, meaning a binary decision, we use binary cross entropy in Eq. (1) as the loss function. This enables the network to predict values with a high confidence for pixels, resulting in strong contours for the segmented masks, assuming a Bernoulli distribution for the errors made. As the size of the lung is comparable with the size of the rest of the image, correction for class imbalance is not necessary.

$$L = \frac{1}{N}\sum_{i=1}^{N} -\log(p(y_i)) \cdot y_i - \log(1 - p(y_i)) \cdot (1 - y_i) \quad (1)$$

Binary cross entropy used as the loss function, where $L$ corresponds to the loss, $y$ to the flattened vector of ground truth pixels, and $p(y)$ the flattened vector of network predictions. $N$ denotes the total number of pixels in the image.

### D. Training

We trained the network for 50 epochs with a batch size of 12, and ADAM as the optimizer. We used a learning rate of $8 \cdot 10^{-5}$ and a weight decay of $10^{-3}$. Early stopping was used to avoid over-fitting on the train dataset, however, by monitoring the train and validation losses, we haven't observed the tendency of that happening at the expense of validation performance. As the network converged and no significant overfitting was observed, we used the validation set also as the test set to evaluate model performance. Since the network still converged well through the epochs, we can deduce that the expressive power of the network is in balance with the available training samples. The evolution of loss values through the iterations can be seen in Fig. 2.

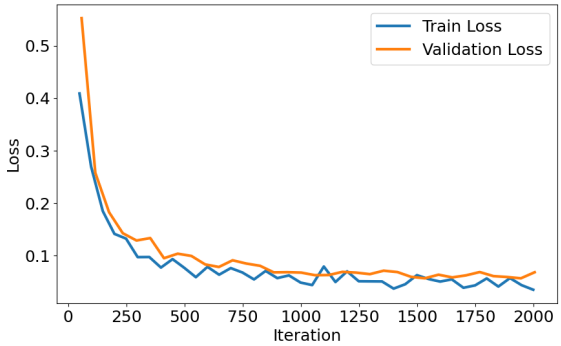



Fig. 2. The evolution of loss values through training iterations.

## V. Results

For evaluation, the predictions of the neural network are binarized using a threshold of 0.5. This can be done because the sigmoid output layer and binary cross entropy loss function grants high confidence outputs around either 0 or 1, thus the exact level of thresholding not significant.

### A. Evaluation metrics

To evaluate the performance of our model quantitatively, we used Sorensen-Dice coefficient and Hausdorff distance to measure the accuracy as well as the locality or robustness of the predictions.

*1) Sorensen-Dice coefficient:* The Sorensen-Dice coefficient measures the overlap between the predicted mask and the ground truth task by taking the harmonic mean of precision and recall [14]. Formally:

$$DSC = \frac{2|PRED \cap GT|}{|PRED| + |GT|} \quad (2)$$

Where $DSC$ denotes the dice coefficient, $PRED$ the set of the coordinates of the mask predicted by the neural network, and $GT$ the set of the coordinates of the ground truth mask

of the lung. $|A|$ denotes the number of elements in set $A$, and $\cap$ denotes the intersection operator.

*2) Hausdorff Distance:* For a segmentation to be useful, high similarity with the target area is not sufficient, it is also important that the errors made are local - they are around the target area. This is important because if the segmentation is used as a preprocessing step, or as the input of some other image processing algorithm, non-local errors may have a detrimental effect on quality of further processing steps. For measuring the locality of the errors a commonly used metric is the Hausdorff distance the measures the largest distance between ground truth and predicted masks. Formally:

$$Hd = \max\left(\max_{x \in PRED} \min_{y \in GT} d(x,y), \max_{x \in GT} \min_{y \in PRED} d(x,y)\right) \tag{3}$$

Where $Hd$ denotes the Hausdorff distance, $PRED$ the prediction of the neural network, $GT$ the ground truth mask, and $d$ representing the Euclidean distance of two points.

### B. Empirical Results

First we evaluated our model quantitatively on the test partitions of the datasets also used for training, which can be seen in Table I. In terms of DSC we managed to achieve similar results to the "state of the art" (SOTA) methods, however, by observing the Hausdorff distance of the predictions we come to a different conclusion: The Hausdorff distance varies between 3-10% of the resolution of the image, and the standard deviation from it being the same. This means that the predictions are not sufficiently robust, non-local errors are frequent. Furthermore, it is also worth noting, that in terms of quantitative results there are huge differences between the datasets. The model generalized to the Montgomery set the best, which was also the smallest in size, followed by JSRT with slightly worse results, finally performing on the Shenzen dataset the worst, in spite of having the largest sample size. The difference in dice errors between the Shenzen and Montgomery sets is twofold, and this is the case with Hausdorff distances as well. Based on this and the distribution of lesions across the datasets we conclude that the model is able to generalize well, even between datasets, however, for more complex lesions, the amount of training data is still not enough to achieve robust generalization.

TABLE I
MEASUREMENT RESULTS

| Dataset | Dice Score | Hausdorff (pixels) |
|---|---|---|
| JSRT Dataset | $0.976 \pm 0.010$ | $18.181 \pm 16.240$ |
| Shenzen Dataset | $0.956 \pm 0.034$ | $28.458 \pm 33.186$ |
| Montgomery Dataset | $0.977 \pm 0.007$ | $13.124 \pm 12.161$ |

By qualitatively evaluating the resulting segmentation masks, we come to a similar conclusion, while the Montgomery dataset contains TB positive patients, both halves of the lungs are clearly visible on all of the images, segmentations are accurate, non-local errors are relatively rare. The case is similar with the JSRT dataset, where pacemakers also occasionally confuse the network. When looking at samples from the Shenzen dataset, we can see that the lower performance of the network stems from two root causes: Some of the annotations are inconsistent, meaning that parts of the heart shadow are also annotated as lungs or the lung mask itself being inaccurate. However, in many cases lesions making the segmentation more difficult are also present. Examples from the different datasets can be seen in Fig. 3.

(a) Montgomery (b) JSRT (c) Shenzen

Fig. 3. Example segmentations from the different datasets. The ground truth segmentations use the red, predictions use the green color channels. Notice how the performance of the predictions degrade as the segmentation task becomes more difficult. An example for inconsistent annotation is the last image, as part of the heart shadow was also included in the mask.

Finally, evaluating the model on a dataset coming from daily clinical practice, we can observe similar performance to the more difficult samples coming from the Shenzen dataset - overflows at the boundary of the lungs are common and so are non-local hallucinations by the network on lower density locations. Some lesions and artifacts, such as pacemakers may also confuse the network, examples can be seen in Fig. 4.

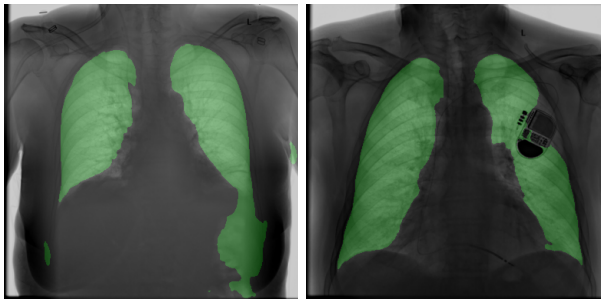


Fig. 4. Example segmentations from the private dataset. Notice how the segmentation overflows at the patient's left diaphragm in the first case. Such errors are difficult to correct even by using expert knowledge based post-processing.

## VI. DISCUSSION

Overall, we found the performance of the model inadequate to serve as a preprocessing step for other algorithms as the segmentations lack the robustness necessary to be used as such. Some may argue that by using traditional image processing methods, the segmented masks may be corrected. While this is true to an extent, some of the hallucinations might be filtered out by selecting the two largest connected components for example, some error modes are still difficult to detect and avert, like over-/underflows near the boundary of the lungs. It is also important to point out that while on

some of the benchmark datasets end-to-end neural networks show decent performance, generalization across datasets is rarely investigated, thus their practical performance may be significantly lower.

## REFERENCES


[1] R. C. Woodruff, X. Tong, S. S. Khan, N. S. Shah, S. L. Jackson, F. Loustalot, and A. S. Vaughan, "Trends in cardiovascular disease mortality rates and excess deaths, 2010-2022," *American Journal of Preventive Medicine*, vol. 66, no. 4, pp. 582–589, 2024.

[2] K. Truszkiewicz, R. Poreba, and P. Gać, "Radiological cardiothoracic ratio in evidence-based medicine," *Journal of Clinical Medicine*, vol. 10, no. 9, p. 2016, 2021.

[3] A. Tumay, D. Hadhazi, and G. Hullam, "Heart segmentation on pa chest x-ray images by model-based deep learning approach," in *2024 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, pp. 1–6, 2024.

[4] G. Gaál, B. Maga, and A. Lukács, "Attention u-net based adversarial architectures for chest x-ray lung segmentation," *arXiv 2020, arXiv:2003.10304.*, 2020.

[5] N. Reamaroon, M. W. Sjoding, H. Derksen, E. Sabeti, J. Gryak, R. P. Barbaro, B. D. Athey, and K. Najarian, "Robust segmentation of lung in chest x-ray: applications in analysis of acute respiratory distress syndrome," *BMC Medical Imaging*, vol. 20, no. 1, p. 116, 2020.

[6] A. Carass, S. Roy, A. Gherman, J. C. Reinhold, A. Jesson, T. Arbel, O. Maier, H. Handels, M. Ghafoorian, B. Platel, A. Birenbaum, H. Greenspan, D. L. Pham, C. M. Crainiceanu, P. A. Calabresi, J. L. Prince, W. R. G. Roncal, R. T. Shinohara, and I. Oguz, "Evaluating white matter lesion segmentations with refined sørensen-dice analysis," *Scientific Reports*, vol. 10, May 2020.

[7] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, T. Kobayashi, K.-i. Komatsu, M. Matsui, H. Fujita, Y. Kodera, and K. Doi, "Development of a digital image database for chest radiographs with and without a lung nodule," *American Journal of Roentgenology*, vol. 174, pp. 71–74, 01 2000.

[8] S. Candemir, S. Jaeger, K. Palaniappan, J. Musco, R. Singh, Z. Xue, A. Karargyris, S. Antani, G. Thoma, and C. Mcdonald, "Lung segmentation in chest radiographs using anatomical atlases with nonrigid registration," *IEEE Transactions on Medical Imaging*, vol. 33, pp. 577–590, 02 2014.

[9] O. Ronneberger, P. Fischer, and T. Brox, *U-Net: Convolutional Networks for Biomedical Image Segmentation*, p. 234–241. Springer International Publishing, 2015.

[10] W. Liu, J. Luo, Y. Yang, W. Wang, J. Deng, and L. Yu, "Automatic lung segmentation in chest x-ray images using improved u-net," *Scientific Reports*, vol. 12, p. 8649, May 2022.

[11] S. Jaeger, S. Candemir, S. Antani, Y.-X. J. Wáng, P.-X. Lu, and G. Thoma, "Two public chest x-ray datasets for computer-aided screening of pulmonary diseases," *Quant Imaging Med Surg*, vol. 4, pp. 475–477, Dec. 2014.

[12] R. Azad, E. K. Aghdam, A. Rauland, Y. Jia, A. H. Avval, A. Bozorgpour, S. Karimijafarbigloo, J. P. Cohen, E. Adeli, and D. Merhof, "Medical image segmentation review: The success of u-net," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, p. 10076–10095, Dec. 2024.

[13] N. Siddique, S. Paheding, C. P. Elkin, and V. Devabhaktuni, "U-net and its variants for medical image segmentation: A review of theory and applications," *IEEE Access*, vol. 9, p. 82031–82057, 2021.

[14] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, p. 861–874, June 2006.

# xLSTM Architectures in Reinforcement Learning

Mátyás Antal, András Gézsi
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {antalm, gezsi}@mit.bme.hu

***Abstract*—Long Short-Term Memory (LSTM) architectures have recently seen significant advancements through innovations such as exponential gating and modified memory structures, reigniting interest in their potential for modern sequence-based tasks. While xLSTM models have demonstrated strong performance in language modeling, their suitability for reinforcement learning (RL) tasks has yet to be fully explored. In this work, we investigate the application of xLSTM in RL environments, focusing on classic control tasks that are commonly employed as benchmarks. This comparison provides a starting point for understanding the differences between xLSTM and LSTM in the context of reinforcement learning.**

***Index Terms*—xLSTM, reinforcement learning, machine learning**

## I. Introduction

Reinforcement Learning (RL) has emerged as a powerful paradigm for training agents to make sequential decisions in complex environments [2], [7], [13]. Central to the success of RL in tasks such as robotics [12] and autonomous driving [10] is the agent's ability to capture and utilize temporal dependencies within the environment. These temporal dependencies often manifest as long-term relationships between actions and their consequences, necessitating sophisticated memory architectures to effectively learn and adapt. Additionally, many RL environments are partially observable, meaning that the agent must infer hidden states from a sequence of observations, further highlighting the need for robust memory mechanisms. [6], [9]

Long Short-Term Memory (LSTM) [3] networks have been a cornerstone in addressing temporal dependencies due to their ability to maintain and update internal states over extended sequences. Since their introduction in the 1990s, LSTMs have demonstrated remarkable performance across various sequence-based tasks, including language modeling, time-series prediction, and more recently, reinforcement learning [4], [8], [16]. However, despite their strengths, traditional LSTMs exhibit limitations in scalability and efficiency, particularly when applied to large-scale models and complex tasks.

The advent of Transformer architectures, characterized by their parallelizable self-attention mechanisms, has overshadowed LSTMs in many applications, especially in natural language processing. In response to these challenges, Extended Long Short-Term Memory (xLSTM) architectures have been proposed as an enhancement to traditional LSTMs [3]. xLSTM introduces two primary modifications: exponential gating and novel memory structures. These innovations aim to mitigate the inherent limitations of LSTMs by enabling more efficient memory manipulation and better scalability. Specifically, xLSTM encompasses two variants—sLSTM and mLSTM—that incorporate scalar and matrix memory updates, respectively, thereby enhancing parallelizability and memory capacity.

This paper explores the applicability and performance of xLSTM architectures within the realm of reinforcement learning. We aim to address the following research questions:

1) How do xLSTM architectures compare to standard LSTM networks in RL tasks?
2) Can xLSTMs improve performance in environments characterized by long-term temporal dependencies and partial observability?

To achieve these objectives, we implement necessary modifications to accommodate non-zero initial hidden states, enabling more flexible and dynamic learning in RL settings. We conduct a systematic evaluation of xLSTM models on classic control tasks, which are commonly used to evaluate early versions of RL algorithms.

Our contributions are twofold:

1) **Architectural Enhancements**: We adapt xLSTM architectures to support non-zero initial hidden states during parallel execution.
2) **Initial Evaluation**: We perform the first comparison of xLSTM and standard LSTM models across multiple RL environments, giving a first look into their strengths and weaknesses.

Through this investigation, we aim to shed light on the potential of xLSTM architectures to advance the state-of-the-art in reinforcement learning, offering new avenues for developing more capable and efficient RL agents.

## II. Related Work

### A. LSTM in Reinforcement Learning

Long Short-Term Memory networks have been extensively utilized in reinforcement learning to address challenges posed by temporal dependencies and partial observability. Traditional RL algorithms often assume fully observable environments, but many real-world tasks require agents to infer hidden states from sequences of observations. Recurrent Reinforcement Learning (RRL) integrates LSTM layers into RL frameworks, enabling agents to maintain and update internal representations based on past experiences.

DOI https://doi.org/10.3311/MINISY2025-007

### B. Advancements in LSTM: Extended LSTM (xLSTM)

Traditional LSTMs, though effective, encounter scalability and efficiency issues when managing large parameter spaces or complex tasks requiring extensive memory. To address these challenges, Extended Long Short-Term Memory architectures introduce two key innovations: exponential gating and novel memory structures. Exponential gating offers a more flexible control of information flow, enabling finer updates and greater training stability. xLSTM variants include sLSTM, which uses scalar memory and updates with memory mixing via heads, and mLSTM, featuring a matrix memory with a fully parallelizable covariance update rule.

These enhancements not only increase the expressive power of LSTMs but also improve their parallelization capabilities, positioning xLSTM alongside state-of-the-art models like Transformers and State Space Models in performance and scalability. Additionally, integrating xLSTM into residual block architectures allows for stacking xLSTM blocks, further enhancing their ability to handle complex sequence-based tasks.

### C. Challenges in Reinforcement Learning

Reinforcement learning inherently involves several challenges that necessitate sophisticated architectural solutions. Key challenges include [1]:

- **Partial Observability**: Many RL environments do not provide full state information, requiring agents to infer hidden states from a history of observations.
- **Sparse Rewards**: In some tasks, rewards are infrequent or delayed, making it difficult for agents to associate actions with outcomes.
- **Long-Term Credit Assignment**: Effective learning requires agents to correctly attribute rewards to actions taken many steps in the past, necessitating the ability to maintain and utilize long-term dependencies.

### D. Summary of Related Work

The integration of LSTM networks into reinforcement learning has significantly advanced the field [4], [8], [16] by enabling agents to handle temporal dependencies and partial observability. However, the scalability and efficiency of traditional LSTMs present ongoing challenges, particularly in large-scale and complex environments. While xLSTM has shown promise in domains like language modeling and time-series prediction, its application in reinforcement learning remains largely unexplored. This paper aims to bridge this gap by systematically evaluating xLSTM architectures in RL settings, providing new insights into their effectiveness and potential advantages over standard LSTM models.

## III. Methodology

In this section, we outline the methodologies employed to evaluate the effectiveness of Extended Long Short-Term Memory architectures in reinforcement learning environments. We begin by providing an overview of the Proximal Policy Optimization (PPO) algorithm [15], which serves as the foundation for our RL framework. Subsequently, we discuss the integration of LSTM networks into PPO, highlighting how recurrent architectures modify standard PPO computations. Finally, we detail the modifications made to the mLSTM architecture to enable flexible hidden state handling when sampling sequences during training.

### A. Proximal Policy Optimization

Proximal Policy Optimization [15] is a state-of-the-art on-policy RL algorithm that optimizes the policy while ensuring stable and efficient updates. PPO uses a clipped surrogate objective to limit large deviations from the current policy, balancing exploration and exploitation during training. The PPO objective is defined as:

$$L^{CLIP}(\theta) = \mathbb{E}_t \left[ \min \left( r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right], \tag{1}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the probability ratio, $\hat{A}_t$ is the estimated advantage at time step $t$, and $\epsilon$ is a hyperparameter controlling the clipping range.

PPO also incorporates Generalized Advantage Estimation (GAE) [14] to balance bias and variance in the advantage calculation and performs multiple epochs of gradient ascent on the same batch of collected trajectories to improve sample efficiency.

### B. On-Policy RL with LSTM

The integration of LSTM networks into the PPO framework enables the agent to maintain a memory of past observations, which is critical for environments with temporal dependencies and partial observability. However, the use of recurrent layers introduces unique challenges to the standard PPO implementation, particularly during the training process.

*1) Recurrent Policy and Value Networks:* In our setup, both the policy and value networks are augmented with LSTM layers. These layers allow the network to process sequences of observations $(s_1, s_2, \ldots, s_T)$ and update their internal hidden states $h_t$ and $c_t$ over time. The outputs of the recurrent networks are then used to compute the policy $\pi(a_t|s_t, h_t)$ and the value function $V(s_t, h_t)$:

$$h_t, c_t = \text{LSTM}(s_t, h_{t-1}, c_{t-1}), \tag{2}$$

$$\pi(a_t|s_t, h_t) = \text{PolicyNetwork}(h_t), \tag{3}$$

$$V(s_t, h_t) = \text{ValueNetwork}(h_t), \tag{4}$$

where $h_t$ and $c_t$ represent the hidden and cell states of the LSTM at time step $t$.

*2) Handling Hidden States During Training:* In reinforcement learning, trajectories of experience are collected in batches, often spanning multiple episodes. These trajectories are split into fixed-length sequences $(s_t, \ldots, s_{t+K})$ for gradient updates using Backpropagation Through Time (BPTT). To ensure proper training of recurrent models, it is important that the hidden states (such as $h_t$ and $c_t$) at the start of each

training segment are consistent with the hidden states at that point in the original uninterrupted sequence. This consistency ensures that the model learns from the correct context without losing track of the information carried over from previous time steps.

### C. Parallelized mLSTM with Non-Zero Initial States

While the mLSTM cell naturally accommodates non-zero initial states through its recursive formulation, its parallelized implementation—designed to process all time steps simultaneously—originally assumes zero-initialized states. In standard language modeling or sequence processing tasks where full trajectories are known a priori, zero initialization at the start of each sequence is suitable. However, in on-policy reinforcement learning, trajectories are sampled in segments that must preserve the continuity of hidden states across sequence boundaries. To address this requirement, we modify the parallelized mLSTM forward pass to accept and incorporate non-zero initial states.

The mLSTM maintains three sets of hidden states: 1) a matrix memory $\mathbf{C} \in \mathbb{R}^{d\times d}$, 2) a normalization vector $n \in \mathbb{R}^d$, and 3) a stabilization scalar $m \in \mathbb{R}$. In the original parallelized version, these states are implicitly assumed to be zero at the start of each processed batch. Here, we explicitly incorporate $\mathbf{C}_0$, $n_0$, and $m_0$ as initial states, enabling the model to continue from previously computed states, thus preserving temporal information across sampled segments.

Next, we present a parallelized formulation of the mLSTM forward pass, with newly introduced terms for handling non-zero initial states highlighted in red. Let $X \in \mathbb{R}^{T\times d}$ be the input sequence of length $T$. The mLSTM cell applies linear transformations to $X$ to obtain the query, key, value, and pre-activation output gate matrices, $\mathbf{Q}, \mathbf{K}, \mathbf{V}, \tilde{\mathbf{O}} \in \mathbb{R}^{T\times d}$, as well as the pre-activation forget and input gate vectors, $\tilde{f}, \tilde{i} \in \mathbb{R}^T$. In the standard forward pass, these gate activations are arranged to form a lower-triangular structure. A matrix $D$ is then introduced to represent the compounded influence of forgetting and input signals over time, with exponential and logarithmic transformations applied for numerical stability. To incorporate non-zero initial states, the initial forget factor and memory updates are adjusted, ensuring that the forward pass accurately reflects the desired initial conditions.

$$F_{ij} = \begin{cases} 0 & \text{if } i < j, \\ 1 & \text{if } i = j, \\ \prod_{k=j+1}^{i} \sigma(\tilde{f}_k) & \text{if } i > j, \end{cases} \tag{5}$$

$$\tilde{I}_{ij} = \begin{cases} 0 & \text{if } i < j, \\ \tilde{i}_j & \text{if } i \geq j, \end{cases} \tag{6}$$

$$D = F \odot \exp(\tilde{I}). \tag{7}$$

We define $f \in \mathbb{R}^d$, as the vector of forget gate activations for the initial states, $n_0$ and $\mathbb{C}_0$. The stabilized version of the activations also includes the initial stabilization scalar, $m_0$.

$$f_i = m_0 \prod_{k=1}^{i} \sigma(\tilde{f}_k). \tag{8}$$

The activations need to be stabilized due the exponential gating functions:

$$\tilde{D} = \log D = \log(F) + \tilde{I}, \tag{9}$$

$$\tilde{f} = \log f, \tag{10}$$

$$m = \max_{\text{row-wise}} \tilde{D}, \tag{11}$$

$$m' = \max_{element-wise}(m, \tilde{f}), \tag{12}$$

$$\hat{D} = \exp(\tilde{D} - m'), \quad \hat{f} = \exp(\tilde{f} - m'). \tag{13}$$

With these stabilized values, we compute the hidden states for all $T$ steps. We can use the stabilized initial forget gate activation vector $\hat{f}$, to add the terms of the initial states:

$$\tilde{\mathbf{C}} = \frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}} \odot \hat{D}, \tag{14}$$

$$\tilde{\mathbf{H}} = \mathbf{Q}\mathbf{C}_0 \odot \hat{f} + \tilde{\mathbf{C}}\mathbf{V}, \tag{15}$$

$$n = \left| n_0 \odot \hat{f} + \sum_{j=1}^{T} \tilde{\mathbf{C}}_{ij} \right|, \tag{16}$$

$$\hat{\mathbf{H}} = \frac{\tilde{\mathbf{H}}}{\max(n, \exp(-m'))}, \tag{17}$$

$$\mathbf{H} = \sigma(\tilde{\mathbf{O}}) \odot \hat{\mathbf{H}}. \tag{18}$$

These modifications allow the parallelized mLSTM to incorporate non-zero initial states in an on-policy RL setting. By doing so, we preserve the temporal continuity of the hidden states across sampled sequence segments, ensuring that the model accurately processes partial trajectories and maintains essential information for effective reinforcement learning.

## IV. Experiments

In this section, we detail the experimental setup used to evaluate the performance of various network architectures within reinforcement learning environments. We describe the models trained, the environments selected for testing, and the evaluation metrics employed to assess performance, stability, convergence speed, and sample efficiency.

### A. Model Architectures

We trained five distinct models using the Proximal Policy Optimization algorithm across different RL environments. The architectures evaluated are as follows:

1) **Multilayer Perceptron (MLP)**: A fully connected neural network without recurrent connections.
2) **LSTM**: Incorporates LSTM layers to capture temporal dependencies.
3) **xLSTM with a Single Block**:
   - **sLSTM Only**: Utilizes the scalar LSTM variant.
   - **mLSTM Only**: Utilizes the matrix LSTM variant.

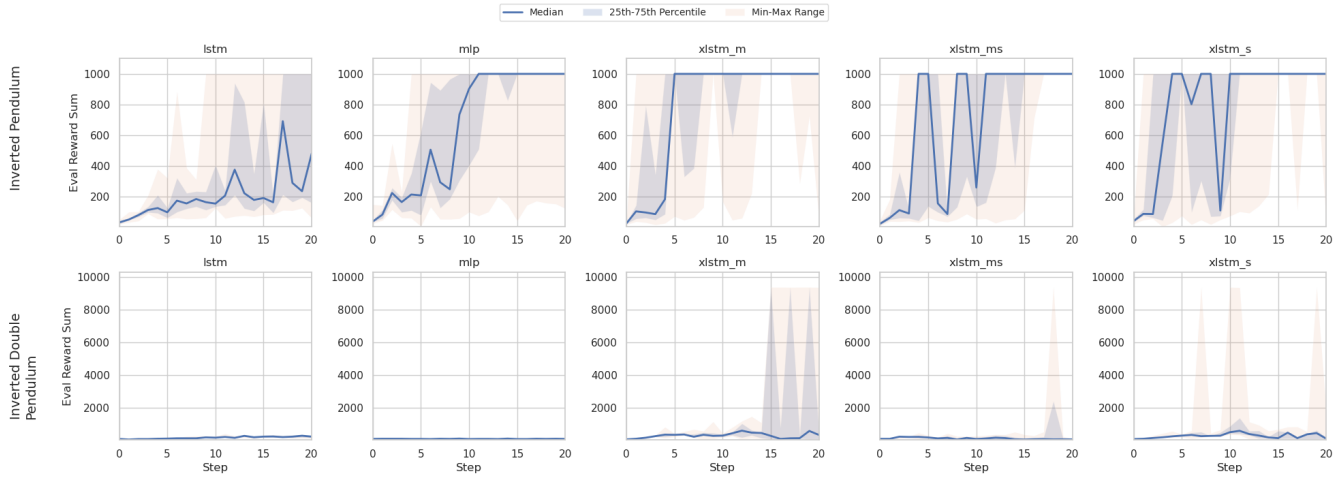



Fig. 1. Cumulative reward distributions across different training runs. Each column represents a different model, and each row corresponds to a different environment. The top row shows results for the Inverted Pendulum environment, while the bottom row corresponds to the Inverted Double Pendulum environment. The xLSTM variants consistently outperform the MLP and LSTM models, achieving higher cumulative rewards and demonstrating faster convergence.

4) **xLSTM with Two Blocks**: Combines both mLSTM and sLSTM blocks in sequence, with the mLSTM block followed by the sLSTM block.

Table I summarizes the number of parameters for each model architecture.

TABLE I
PARAMETER COUNTS FOR DIFFERENT MODEL ARCHITECTURES

| Model | Number of Parameters |
|---|---|
| MLP | 135,427 |
| LSTM | 595,971 |
| sLSTM | 498,947 |
| mLSTM | 485,387 |
| xLSTM | 914,443 |

### B. Environments

We evaluated the models on two classic control tasks: the Inverted Pendulum and the Inverted Double Pendulum. These tasks are fundamental benchmarks in the field of control theory and reinforcement learning [5], [11], [15], providing a robust framework for assessing the performance and stability of various algorithms. The Inverted Pendulum task involves balancing a single pendulum in the upright position by applying forces at the base, challenging the model to maintain equilibrium under dynamic conditions. In contrast, the Inverted Double Pendulum introduces an additional joint, significantly increasing the complexity and requiring more sophisticated control strategies to manage the inherently unstable two-link system. By selecting these environments, we aim to test our models' ability to learn effective control policies in both simple and more intricate settings, thereby demonstrating their robustness and adaptability in handling varying levels of control complexity.

Both environments employ a limited range continuous reward structure and evaluate the agent over a maximum of 1,000 steps. Rewards are granted based on how upright the pendulums remain at each step. In the Inverted Pendulum task, the agent receives a reward of +1 for every step in which the pendulum angle stays within a specified limit. In contrast, the Inverted Double Pendulum utilizes a more intricate reward formula that accounts for both the angles and velocities of the two pendulum links, with rewards ranging from 0 to 10 per step. Consequently, the maximum cumulative rewards achievable are 1,000 for the Inverted Pendulum and 10,000 for the Inverted Double Pendulum.

### C. Experimental Procedure

For each environment, we conducted multiple training runs to assess the stability and consistency of each model architecture. The key aspects of the experimental procedure are as follows:

*1) Training Setup:* Each model was trained using the PPO algorithm with the following configuration. Values for different environments are detailed in Table II.:

- **Total Frames** ($N_{\text{total}}$): The total number of frames processed during training.
- **Evaluation Interval** ($N_{\text{eval}}$): The frequency at which the agent's performance is evaluated.
- **Multiple Runs**: RL algorithms are often unstable, especially when not using highly optimized hyper-parameters. Due to this instability we run 10 independent training for all environments.

*2) Evaluation Metrics:* To comprehensively evaluate the performance of each model, we employed the following metrics:

- **Median Cumulative Reward**: The median of cumulative rewards obtained across multiple runs, offering a robust

TABLE II
TRAINING PARAMETERS FOR DIFFERENT ENVIRONMENTS

| Environment | $N_{\text{total}}$ | $N_{\text{eval}}$ | $N_{\text{runs}}$ |
|---|---|---|---|
| Inverted Pendulum | 100,000 | 5,000 | 10 |
| Inverted Double Pendulum | 100,000 | 5,000 | 10 |

measure less susceptible to outliers. In each training iteration, the model that achieves the highest cumulative reward is identified and counted as the result of the training.

- **Stability**: Assessed by the variance in cumulative rewards across different runs.
- **Convergence Speed & Sample Efficiency**: The number of frames required for the model to achieve near-optimal performance.

Additionally, Figure 1 illustrates the cumulative rewards achieved by each model throughout the training process.

### D. Results

The calculated metrics for the Inverted Pendulum environment are presented in Table III. All model types successfully achieve the maximum score of 1000 for this relatively simple task. However, the sLSTM model achieves the perfect score with the fastest convergence speed, requiring fewer samples to master the task. This suggests that the enhanced structure of the sLSTM contributes to a quicker learning process.

TABLE III
PERFORMANCE METRICS FOR INVERTED PENDULUM

| Model | Median Reward | Reward Std. Dev. | Convergence Speed |
|---|---|---|---|
| MLP | 1000 | 0 | 6.7 |
| LSTM | 1000 | 195 | 12.1 |
| sLSTM | 1000 | 0 | 2.7 |
| mLSTM | 1000 | 0 | 3.5 |
| xLSTM | 1000 | 0 | 3.9 |

The calculated metrics for the Inverted Double Pendulum environment are shown in Table IV. Unlike the Inverted Pendulum task, this environment poses a greater challenge, leading to notable differences in model performance. The mLSTM model achieves the highest median reward of 9354 while maintaining a high reward standard deviation. This indicates that the mLSTM model is capable of high performance but exhibits variability across runs. Additionally, the xLSTM model demonstrates faster convergence compared to other models, showcasing its efficiency in learning despite not achieving the highest score.

*1) Performance Analysis:* The experimental results indicate that xLSTM architectures consistently outperform standard LSTM and MLP models across classic control tasks, but the two-block xLSTM architecture, which combines mLSTM and sLSTM blocks, doesn't demonstrate superior stability and faster convergence compared to single-block variants and traditional models, this is due the simple nature of the test

TABLE IV
PERFORMANCE METRICS FOR INVERTED DOUBLE PENDULUM

| Model | Median Reward | Reward Std. Dev. | Convergence Speed |
|---|---|---|---|
| MLP | 129 | 18 | 8.4 |
| LSTM | 316 | 39 | 15.8 |
| sLSTM | 1128 | 4193 | 12.2 |
| mLSTM | 9354 | 4218 | 14.4 |
| xLSTM | 330 | 3924 | 7.2 |

environments. The enhanced memory structures of xLSTM enable better handling of long-term dependencies and partial observability, leading to improved sample efficiency and higher cumulative rewards.

## V. CONCLUSION AND FUTURE WORK

In this study, we explored the application of Extended Long Short-Term Memory architectures within reinforcement learning environments, specifically targeting classic control tasks. Our experiments compared five distinct models: Multilayer Perceptron, standard Long Short-Term Memory networks, single-block xLSTM variants (sLSTM only and mLSTM only), and a two-block xLSTM architecture combining mLSTM and sLSTM blocks.

**Summary of Findings:** Our experimental results indicate that xLSTM architectures consistently outperform both traditional LSTM and MLP models across all tested environments. The one-block mLSTM, which consist of a single mLSTM block, demonstrated the highest median cumulative rewards, but the single block sLSTM displayed the fastest convergence. The performance of the xLSTM blocks can be attributed to the advanced memory structures and exponential gating mechanisms, which facilitate better handling of long-term dependencies and partial observability inherent in RL tasks.

**Implications:** The superior performance of xLSTM architectures suggests that they are highly effective in environments requiring robust temporal reasoning and state inference. By maintaining and utilizing complex internal states, xLSTM enables RL agents to achieve higher performance levels, making it a promising architecture for a wide range of RL applications, including those with intricate temporal dynamics and partial observability.

**Limitations:** Despite the promising results, xLSTM models exhibit a higher computational cost compared to standard LSTMs due to their increased number of parameters and more complex memory structures. Additionally, while xLSTM showed significant improvements in the tested environments, its scalability and effectiveness in more complex and high-dimensional environments remain to be thoroughly evaluated.

**Future Work:** Future research will focus on several key areas to further enhance and validate the effectiveness of xLSTM in reinforcement learning:

- **Broader Environment Testing**: Extend the evaluation to a wider variety of RL environments, pixel-based partially observable environments, such as the atari57 benchmark,

to assess the generalizability and robustness of xLSTM architectures.

- **Theoretical Analysis**: Conduct a deeper theoretical analysis of xLSTM's memory dynamics and gating mechanisms to better understand the underlying factors contributing to its superior performance.

In conclusion, xLSTM architectures represent a significant advancement in the design of memory-augmented networks for reinforcement learning. Their ability to effectively manage temporal dependencies and partial observability opens new pathways for developing more capable and efficient RL agents. We anticipate that continued exploration and refinement of xLSTM models will contribute to further breakthroughs in the field of reinforcement learning.

## REFERENCES


[1] Andrew G Barto. Reinforcement learning: An introduction. by richard's sutton. *SIAM Rev*, 6(2):423, 2021.

[2] Jakob Bauer, Kate Baumli, Feryal Behbahani, Avishkar Bhoopchand, Nathalie Bradley-Schmieg, Michael Chang, Natalie Clay, Adrian Collister, Vibhavari Dasagi, Lucy Gonzalez, et al. Human-timescale adaptation in an open-ended task space. In *International Conference on Machine Learning*, pages 1887–1935. PMLR, 2023.

[3] Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xlstm: Extended long short-term memory. *arXiv preprint arXiv:2405.04517*, 2024.

[4] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[5] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[6] Dibya Ghosh, Jad Rahme, Aviral Kumar, Amy Zhang, Ryan P Adams, and Sergey Levine. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability. *Advances in neural information processing systems*, 34:25502–25515, 2021.

[7] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[8] Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019.

[9] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

[10] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):4909–4926, 2021.

[11] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.

[12] Athanasios S Polydoros and Lazaros Nalpantidis. Survey of model-based reinforcement learning: Applications on robotics. *Journal of Intelligent & Robotic Systems*, 86(2):153–173, 2017.

[13] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[14] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[15] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[16] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

# Towards Integrating Abstraction and Partial Order Reduction in Probabilistic Model Checking: Survey of Challenges and Opportunities

Dániel Szekeres, István Majzik
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: szekeresdani@edu.bme.hu, majzik@mit.bme.hu

***Abstract*—Partial order reduction (POR) and abstraction are successful techniques for mitigating state space explosion in model checking, but combining them in a sound and efficient way is non-trivial. While they have been successfully combined in non-probabilistic model checking, their integration for probabilistic models remains largely unexplored. We review the related literature, identify research gaps, and suggest potential strategies for integrating abstraction and POR in probabilistic model checking.**

***Index Terms*—Probabilistic Model Checking, Abstraction, Partial Order Reduction, Markov Decision Processes**

## I. Introduction

Probabilistic model checking is a widespread formal verification technique for complex systems exhibiting both stochastic and non-deterministic behavior. Its application areas include reliability analysis, where stochastic behavior results from random hardware failures, and performance evaluation, where random arrival and service times must be modeled. However, its scalability is limited by state space explosion, a problem caused by different aspects of non-determinism: the order in which concurrent processes act, non-deterministic environmental inputs, and non-deterministic control flow [1].

We focus on two state space reduction techniques aiming to address this: *abstraction*, which omits unnecessary state information, and *partial order reduction (POR)*, which eliminates redundant interleavings of concurrent processes [2]. Their sound and efficient combination is challenging [3]–[5], even without leveraging synergies, and synergistic combinations present further difficulties [6], [7]. While they have been successfully combined in non-probabilistic model checking, their integration for probabilistic models remains largely unexplored. This paper reviews the related literature, identifies research gaps, and explores potential methods for integrating abstraction and POR in probabilistic model checking. A complete review of general POR and trace reduction techniques would warrant a full survey by itself, we only give an overview of its major directions, focusing on those that have been applied in probabilistic or abstraction-based contexts.

This research was partially funded by the EKÖP-24-3 New National Excellence Program under project number EKÖP-24-3-BME-159, and the Doctoral Excellence Fellowship Programme under project number 400443/2023; funded by the NRDI Fund of Hungary.

Our contributions are two-fold: we categorized the surveyed literature based on the underlying POR approach and the context, and identified potential future directions based on our findings. First, We briefly explain the necessary background on POR, abstraction, probabilistic model checking, and how games are related to the abstraction of probabilistic models, then present our categorization of the relevant literature. After this, each section is dedicated to one POR approach. We describe the main idea of each technique and our findings for that technique. Finally, we identify some research gaps and promising directions for future research based on our findings.

## II. Background

*Probabilistic Model Checking: Markov Decision Processes (MDPs)* are low-level mathematical models that are able to represent both proper non-determinism by offering a choice between multiple actions in each state and probabilistic behavior by sampling from a distribution of successor states after choosing an action. They are often used in probabilistic model checking to model randomized distributed algorithms, where concurrency introduces non-deterministic behavior through the ordering in which the distributed components act [1]. Apart from concurrency, the lack of statistical information about aspects of the system or the environment might also necessitate using proper non-determinism in the model.

Probabilistic model checking also takes the property to check as input, which can take several possible forms: the maximal probability of reaching a specified set of states, linear-time temporal properties specifying constraints for traces, branching-time temporal properties constraining the whole computation tree of the model, etc. [1]. When developing state-space reduction techniques, we must make sure the reduced state space is equivalent to the original one w.r.t. the property of interest – preserving branching time properties necessitates stricter conditions for keeping traces than linear time and reachability [8], [9].

Most MDP analysis algorithms are based on the *value function* mapping each state to a real number depending on the property. For example, it maps each state to the probability of reaching a target state from that state for reachability queries,

DOI https://doi.org/10.3311/MINISY2025-008

and to the expected value of the total collected reward starting from that state for reward-based queries.

Instead of explicitly specifying states and transitions, MDP models are often given in a high-level, symbolic form, using state variables and actions modifying them. The actions can be enabled based on the current value of the variables, and each of them has an associated distribution of assignments changing the variables. After an enabled action is chosen, an assignment is sampled from its distribution and applied to the variables to compute the next state. Structural features like control flow locations and composition of multiple components are also widely used in such description languages, which are useful for implementing state-space reduction techniques.

*Abstraction: Abstraction* reduces the size of the state space by merging multiple original states into abstract states, ignoring some information about them. *Abstract interpretation* can be regarded as both an implementation of abstraction-based analysis and as a general framework for abstraction [10], defining the notions of *abstract domains* for specifying how information is ignored, and *abstract transformers*, functions for applying actions on the level of abstract states.

As instantly finding the appropriate abstraction for a model is hard, iterative techniques emerged. *Counterexample-Based Abstraction Refinement (CEGAR)* [11] starts with a very coarse conservative abstraction of the original model and refines it if it finds an abstract counterexample for the property which cannot be mapped to the original model. *Lazy abstraction* techniques, like the algorithms of BLAST [12] and IMPACT [13] interleave the abstract state-space exploration and the refinement steps of CEGAR and refine only the necessary parts of the abstract state space. Such techniques often utilize a *covering* relation: as lazy refinement makes the precision of the abstraction non-homogeneous throughout the state space, some abstract states can cover all concrete states represented by another abstract state, making further exploration from the covered abstract state unnecessary.

*MDP abstraction and stochastic games:* Several approaches exist for abstraction-refinement-based MDP analysis. An important aspect for categorizing them is the formalism used for the abstract model: while it can also be an MDP [14]–[16], other approaches instead opt for separating the original and abstraction-induced non-determinism by using *stochastic games* and assigning the task of resolving them to different players [17]–[22]. The latter choice has the advantage that both a lower and an upper bound can be computed for the value function by making the players either cooperate or compete. This gives information about how good the current abstraction is and shows the precision of each abstract state with respect to the value function, enabling more focused refinement.

*Partial Order Reduction: partial order reduction (POR)* builds on the observation that in most concurrent systems, the order in which some actions are executed does not always matter. The main underlying concept is the definition of equivalence classes for traces, where the ordering of such actions is ignored. These equivalence classes can be straightforward to define declaratively, especially with reference to the whole state space, but the aim of POR is exactly to avoid exploring the unnecessary parts of the state space. Operational definitions of specific POR algorithms mostly rely on computing a sufficient subset of actions to explore in each state. There is a trade-off between the price of computing such sets and their reduction power (i.e. how close they are to exploring exactly only the minimum necessary number of states).

POR algorithms can be classified as *static* or *dynamic*. *Static POR (SPOR)* [23] completely relies on a preprocessing step operating on the high-level formal description of the concurrent processes [24], while *Dynamic POR (DPOR)* uses an approach based on a depth-first search to generate the action sets to explore on-the-fly. DPOR can operate with more precise independence relations, but it needs a dedicated integration into the verification algorithm, making it harder to combine with other state-space reduction techniques [1].

## III. Overall survey results

After surveying the related literature, we identified the main underlying POR approaches used in the context of abstraction-based analysis, probabilistic systems, or games. Table I shows each related publication categorized by this and its context.

| | Basic | Abstract | Probabilistic | Game |
|---|---|---|---|---|
| Ample | [25] | | [8], [24], [26]–[28] | |
| Persistent | [29] | [30], [31] | | |
| Stubborn | [29], [32] | [3], [5] | [33] | [34]–[38] |
| Source | [39], [40] | [41] | | |
| Confluence | [42] | | [43]–[47] | |
| Unfolding | [48]–[50] | [51], [52] | | |
| Pre-reduction | [53] | | [54]–[56] | |
| Other | [57] | [4], [7], [58] | [59] | [60], [61] |

TABLE I: Categorization of papers related to our survey.

The *Basic* column shows examples of relevant publications for each approach without abstraction or probabilistic aspects.

The *Abstract* column lists the papers we found on combining abstraction and POR in non-probabilistic model checking. Overall, the existing works show that POR and abstraction interact in several ways: abstraction creates and eliminates dependence between actions [51], covering interacts with the backtracking step of DPOR [41], and the soundness of how the concrete and abstract independence relations are used during exploration must be ensured [7].

The *Probabilistic* column gives a list of the publications we found about applying POR in the context of probabilistic models, categorized by the type of POR adapted. Although quite a lot of different reduction techniques have been explored for MDPs, there is still a very rich set of methods and ideas to adapt from the toolbox of non-probabilistic model checking.

Given the success of game-based abstraction schemes, we explored the literature for applying POR to stochastic games but found no relevant results. However, some POR techniques have been proposed for non-stochastic games; these are listed in the *Game* column. The number of works in this area remains limited compared to other contexts.

The following sections introduce each major POR approach (rows of Table I), and discuss our findings for that category.

## IV. Ample sets

*Ample set methods* [25] build on an *independence relation*: two actions can be considered independent if they cannot disable each other, and firing them in any order leads to the same state. The independence relation used for partial order reduction need not be the largest such relation – most methods use a safe approximation that can be computed efficiently.

An *ample set* for a state is a subset of enabled actions satisfying multiple conditions, the most important one being the *independence condition*: for an enabled sequence of actions starting from the state such that none of the actions are in the ample set, each action in the sequence must be independent with[1] every action in the ample set. This makes sure that an action depending on the ample set cannot occur before some action from the set has occurred first [1], [9]. Another relevant condition regarding the adaptation for MDPs is the *cycle condition*: if there is a cycle in the state space, any action enabled somewhere in the cycle must be in the ample set of some state in the cycle. POR explores a reduced state space by firing only the actions in a chosen ample set of the state.

*Probabilistic:* The first POR technique to be used in the context of MDPs was based on ample sets [8], [27], changing the ample set conditions to account for end components (sub-MDPs of the original model where it is possible to remain for infinitely many steps with probability 1) instead of deterministic cycles and extending them with additional constraints handling probabilistic actions. Constraints for preserving probabilistic reachability, linear time, and branching time properties have been proposed [8]. This approach has later been extended to reward-based properties [26] and adapted to the case of distributed schedulers [28], where system components may use only partial information for their decisions. A static POR algorithm for MDPs was later proposed [24], overapproximating the dependence relation to compute ample sets based only on syntactic information.

## V. Persistent sets

*Persistent set methods* [29] use the notion of *independence in a state*, giving an independence relation for each state instead of defining a single global relation. This makes the independence relation more fine-grained, resulting in higher reduction power. The definition is similar to the global one, but the conditions do not need to hold in *every* state, only in the state where independence is being checked.

*Sleep set methods* [29] exploit information about the past of the search and are often used together with persistent sets for further reduction. The sleep set of a state is a set of transitions that are enabled in it but will *not* be executed. To compute the sleep set for a new state, we start with the sleep set of its parent, add all actions already explored from the parent before the action leading to the new state, and then remove actions dependent with the action that led to the new state.

[1]To emphasize the symmetric relation, actions are often said to be (in)dependent *with* each other in POR literature.

*Abstract:* A combination of *trace abstraction* [62], [63] and persistent- and sleep-set POR has been proposed [30]. Instead of merging states, trace abstraction represents the abstraction of the original program as a set of traces by viewing the control flow graph as an automaton and traces as words accepted by it. The authors argue that this integration is easier and more natural than with state-based ones like predicate abstraction. This observation makes it a perspective avenue to explore whether trace abstraction can be adapted to the probabilistic setting, as we have not found any existing publication thereon. On the other hand, this makes a similar approach less likely to work with existing MDP abstraction algorithms, which are generally state-based.

The lazy abstraction algorithm of Henzinger et al. [12] has been used together with a persistent- and sleep-set POR in the *Explicit Scheduler Symbolic Thread (ESST)* algorithm [31]. ESST uses lazy abstraction separately for each thread and explicit model checking techniques to explore the scheduling choices between them. POR can be used to reduce the number of possible scheduling choices. As such, the application of POR and abstraction is mostly orthogonal in this framework.

## VI. Stubborn sets

*Stubborn set methods* [32] are often defined without an explicit independence relation, directly referring to the commutativity of actions instead [9] (sometimes the relation is still considered important for their practical implementation [33]). Unlike in the previously mentioned methods, a stubborn set can also contain disabled actions, but must contain at least one enabled action if there is one. Apart from this, the main condition for stubborn sets says that for any enabled action in the stubborn set and any enabled sequence of actions *not* in the stubborn set, the action in the set must remain enabled after firing the sequence, and firing it after the sequence must lead to the same state as firing it before the sequence [9].

*Probabilistic:* After the successful adaptation of ample sets to MDPs, a stubborn-set approach has also emerged, where fairness assumptions are also taken into account [33].

*Game:* POR based on stubborn sets has been adapted to non-stochastic games: [37] extended it to parity games (where players aim to either end the game in a vertex marked with a number of a given parity, or make the highest infinitely often seen number be of that parity), while [34] focused on reachability games (where the goal of one player is reaching a set of target states, and the other aims to prevent this). Interestingly, both of these were originally published in an incorrect version and were later corrected [35], [38]. This shows the non-triviality of correctly adapting POR for games while preserving the winner. As such, utmost care has to be taken when introducing stochastic behavior as well.

## VII. Source sets and trace optimality

The *Optimal DPOR* algorithm [64] introduced another family of sufficient action sets called *Source sets* [39]. We refrain from giving a definition of source sets here as it builds on multiple other definitions. It has been proven that exploring all

elements of a source set from a state is not only a *sufficient*, but also *necessary* condition for the correctness of POR, making source sets a generalization of earlier techniques [40].

Optimal DPOR also introduced *trace optimality*: provably exploring exactly one element of each equivalence class. While this minimizes explored traces to match the number of equivalence classes, further reductions are possible by defining coarser equivalence relations that preserve the property of interest [57], [65]. Developing more efficient trace optimal algorithms is another important avenue of research [50], [66].

*Abstract:* The lazy abstraction algorithm Impact [13] has been combined with source-set DPOR [41]. This work highlights the interaction between the covering relation and the on-the-fly dependence calculation of DPOR, and proposes an efficient way to make the combination sound.

## VIII. Confluence reduction

*Confluence reduction* [42] looks for transitions that do not interfere with any observable behavior: an action enabled before a confluent transition should still be enabled after it, and the system should end up in the same state, regardless of whether the other transition is taken before or after the confluent transition. Such *confluent* transitions can be prioritized and always fired if they are enabled before firing any non-confluent ones, eliminating non-deterministic choices [44].

*Probabilistic:* Confluence reduction has been adapted to MDPs [43], and Markov Automata (extending MDPs with exponentially distributed timed transitions) [45]. Confluence reduction is originally presented with an action focus, but rephrasing it in a state-focused way, similar to POR, has enabled comparison to ample-set POR [44], showing that confluence reduction has strictly higher reduction power.

It was also used to enable statistical model checking (SMC) of a special subset of MDPs [47]. For sound SMC, the model must exhibit only deterministic and probabilistic behavior. For models with only spurious non-determinism (i.e. any strategy has the same outcome), confluence reduction might eliminate it, making SMC applicable. Combining this with POR [46] enabled SMC-based analysis of even more models.

## IX. Unfolding

*Unfolding* approaches were originally proposed for Petri-nets, constructing a Petri-net without cycles and without any two transitions outputting to the same place based on the original general Petri-net [48]. These constraints led to a partial order on the transitions and places such that it is impossible to fire a transition without firing all of its predecessors in the order first. Unfolding-based algorithms compute prefixes of an unfolding using different techniques to make it complete (in the sense that there is a "witness" for every original state and transition) and finite [49]. This has been adapted to more general concurrent systems and combined with POR [50].

*Abstract:* The idea of Petri-net unfolding has been utilized in an abstraction-refinement loop using Petri-nets as abstractions of concurrent programs, analyzing them using the unfolding technique, then refining the net on-demand [52]. Closer to our aim is a paper combining unfolding-based POR with abstract interpretation [51]. Apart from giving a specific algorithm for this combination and proving its soundness, the authors also give examples of the abstraction creating and eliminating dependence between actions and introduce the notion of independence of transformers, which are more generically applicable contributions.

## X. Pre-reduction

Preprocessing the model to create a reduced version is another promising angle of attack at the explosion of interleavings. The underlying is often similar to POR, but instead of reducing the state space during exploration, the model is transformed to another one equivalent with respect to the property of interest but easier to analyze.

In standard software verification, an automatic search for effective reductions has been proposed recently [53]: it searches for a proof of the property and a sound reduction that it can prove safe simultaneously. Adapting this to MDPs could lead to an efficient method for finding an equivalent but easier to analyze MDP, but finding a notion analogous to proofs in non-probabilistic verification is a significant challenge.

*Probabilistic: Syntactic Partial Order Compression* [56] merges actions of a component into atomic blocks when a syntactic analysis of the composite system can prove that this does not affect the overall behavior. This can be seen as a kind of partial order reduction where the representative of an equivalence class is always chosen to execute as many actions of the same component as possible before yielding to another component. This is much less computationally costly than "actual" POR options, and it can be easily combined with any other state space reduction technique, as the preprocessed model is equivalent to the original one and can be analyzed using any MDP analysis technique. The disadvantage is that much less reduction is possible than when the representative interleaving is not constrained this way.

Organizing the components into communication-closed layers [54] is another approach in this category, transforming from a distributed representation to a layered one. This has been extended to abstract probabilistic automata specifications [55], where transitions can have *may*, *must*, and *must not* modalities.

## XI. Other POR techniques

There are other POR techniques in all of our investigated contexts that did not fit into any of the previous categories, as they build directly on the commutativity of actions instead of referring to the types of sufficient action sets.

*Abstract:* Impact [13] has been combined with a POR algorithm developed directly for it [4], highlighting the interaction of covering and DPOR.

Another work integrates POR and abstraction for analyzing timed automata [58]. The focus on zone abstraction for clock variables makes the specific results hard to apply to our target context, but it highlights the importance of considering POR techniques for special model subsets, and the observations on how abstraction affects commutativity can be relevant.

Most of the previously mentioned works concentrated on using abstraction and POR together in a sound way, without exploiting the fact that abstraction can break the dependence of actions. Although we found that this is a much less studied direction, some works on *abstraction-aware* partial order reduction have also been published: abstract commutativity and its sound combination with concrete commutativity has been proposed [7], and a synergistic combination of state interpolation and POR has also been developed [6].

*Probabilistic:* Instead of aiming for general MDPs, developing a POR-like method that works for a constrained, but practically useful subset of models can also be fruitful, as was also observed in the non-probabilistic case [67]. A work focusing on *causally deterministic MDPs* has proved that for this subset of MDPs, considering greedy complete strategies is sufficient for reachability properties referring only to a single process, eliminating the need for checking all schedulings [59].

*Game:* A POR preserving linear- and branching-time temporal properties extended with reasoning about distributed knowledge (i.e. specifying the partial information known by each agent) has been proposed [60]. It was later adapted to Alternating Temporal Logic in a partial information context [61]. The latter one is unsound with perfect information, making it unfit for integration with game-based MDP abstraction.

## XII. Research Gaps and Potential Directions

We have not found any publications on integrating POR and abstraction in *probabilistic* model checking, although this combination has already been studied in non-probabilistic model checking, and both techniques have been adapted to the probabilistic context separately.

To summarize the relationships between the basic POR techniques that this paper considered: ample and stubborn set techniques are often considered to be included in persistent set techniques [29], while other works still consider stubborn sets as a distinct technique depending on the exact definitions [9]. Source sets have been proved to be trace optimal and as such supersede earlier approaches in terms of its peak potential [39], [64], but because of its recency, it has not been adapted to probabilistic models yet. Confluence reduction has been shown to have higher reduction power than ample set POR when defined in a state-focused way, both for probabilistic and non-probabilistic models [44]. The different abstraction techniques cannot be precisely ordered in terms of their reduction power.

We identified the following directions for integrating probabilistic model checking, ordered from most to least promising, based on potential impact and how advanced current research is toward achieving full integration:

- Most components for integrating stubborn-set POR with abstraction for MDPs are already in place. If a game-based algorithm is chosen, a stubborn-set-based POR [35], [38] for games could be adapted to *stochastic* games, inspired by the adaptation of the non-game version to MDPs [33]. However, abstraction games have a special structure: scheduling choices are concentrated on one specific player, and only one player has probabilistic actions in most versions [17]–[19]. Developing a POR algorithm tailored to this subset could be more effective, as seen in other specialized algorithms [58], [59], [67].
  If MDPs are used as abstract models, the existing stubborn-set POR for MDPs [33] could be used, while ensuring soundness of the integration in a similar way as in non-probabilistic verification [3], [5]. If covering is used to prune the abstract state space [14], its interaction with DPOR must be handled [4], [41].
  In both cases, reduction can be further enhanced using abstract commutativity [7].
- Trace-optimality [64], [66] is still relatively new and needs to be adapted to abstraction-based and probabilistic settings, along with coarser equivalence relations [57], [65]. In the long run, pursuing this direction can lead to provably optimal algorithms, but more steps are needed than for the previous alternative.
- Confluence reduction has been shown to be generally better than ample sets in both probabilistic verification [44], but we found no previous work on whether it can be easily combined with abstraction in a sound way. This is also a perspective future direction.
- Reduction through pre-processing [54]–[56] is orthogonal to abstraction-based verification steps, allowing straightforward joint usage. Since they alter the model structure, a comparative evaluation could assess whether abstraction performs better or worse on reduced models. We include this last as it represents a different approach to the problem compared to the focus of this paper.

In non-probabilistic model checking, these methods often made previously intractable models analyzable, so we expect them to be impactful in the probabilistic context as well. Each of these directions can be pursued separately; they do not depend on each other in any way.

## XIII. Conclusions

This paper presented the results of our literature review on integrating partial order reduction and abstraction techniques to address the state space explosion problem in probabilistic model checking. While several foundational elements are already in place, no published work has yet explored their comprehensive integration within the context of MDP analysis. We identified promising directions for future research to advance this area based on our findings.

## References


[1] C. Baier and J. Katoen, *Principles of model checking*. MIT Press, 2008.

[2] E. M. Clarke, T. A. Henzinger, H. Veith, R. Bloem *et al.*, *Handbook of model checking*, 2018, vol. 10.

[3] H. Hansen, "Abstractions for transition systems with applications to stubborn sets," in *Concurrency, Security, and Puzzles - Essays Dedicated to Andrew William Roscoe on the Occasion of His 60th Birthday*, 2017.

[4] D. Kroening, S. Sharma, and B. Wachter, "Abpress: Flexing partial-order reduction and abstraction," *CoRR*, 2014.

[5] H. Hansen, S. Lin, Y. Liu, T. K. Nguyen, and J. Sun, "Diamonds are a girl's best friend: Partial order reduction for timed automata with abstractions," in *CAV 2014*, 2014.

[6] D. Chu and J. Jaffar, "A framework to synergize partial order reduction with state interpolation," in *HVC 2014*, 2014.

[7] A. Farzan, D. Klumpp, and A. Podelski, "Stratified commutativity in verification algorithms for concurrent programs," *Proc. ACM Program. Lang.*, no. POPL, 2023.

[8] M. Größer and C. Baier, "Partial order reduction for markov decision processes: A survey," in *FMCO 2005*, 2005.

[9] A. Valmari and H. Hansen, "Stubborn set intuition explained," *Trans. Petri Nets Other Model. Concurr.*, 2017.

[10] P. Cousot and R. Cousot, "Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints," in *POPL 1977*, 1977.

[11] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, "Counterexample-guided abstraction refinement," in *CAV 2000*, 2000.

[12] T. A. Henzinger, R. Jhala, R. Majumdar, and G. Sutre, "Lazy abstraction," in *POPL 2002*, 2002.

[13] K. L. McMillan, "Lazy abstraction with interpolants," in *CAV 2006*, 2006.

[14] D. Szekeres, K. Marussy, and I. Majzik, "A lazy abstraction algorithm for markov decision processes," in *ASMTA 2024*, 2025.

[15] R. Chadha and M. Viswanathan, "A counterexample-guided abstraction-refinement framework for markov decision processes," *ACM Trans. Comput. Log.*, no. 1, 2010.

[16] H. Hermanns, B. Wachter, and L. Zhang, "Probabilistic CEGAR," in *CAV 2008*, 2008.

[17] M. Kattenbelt, M. Z. Kwiatkowska, G. Norman, and D. Parker, "A game-based abstraction-refinement framework for Markov decision processes," *Formal Methods Syst. Des.*, no. 3, 2010.

[18] J. Esparza and A. Gaiser, "Probabilistic abstractions with arbitrary domains," in *SAS 2011*, 2011.

[19] B. Wachter and L. Zhang, "Best probabilistic transformers," in *VMCAI 2010*.

[20] F. S. Vira and J. Katoen, "Tight game abstractions of probabilistic automata," in *CONCUR 2014*, 2014.

[21] F. Sher and J. Katoen, "Compositional abstraction techniques for probabilistic automata," in *TCS 2012*, 2012.

[22] J. Katoen and F. Sher, "Modal stochastic games - abstraction-refinement of probabilistic automata," in *Models, Algorithms, Logics and Tools - Essays Dedicated to Kim Guldstrand Larsen on the Occasion of His 60th Birthday*, 2017.

[23] R. P. Kurshan, V. Levin, M. Minea, D. A. Peled, and H. Yenigün, "Static partial order reduction," in *TACAS 1998*, 1998.

[24] Á. F. Díaz, C. Baier, C. B. Earle, and L. Fredlund, "Static partial order reduction for probabilistic concurrent systems," in *QEST 2012*, 2012.

[25] D. A. Peled, "Ten years of partial order reduction," in *CAV 1998*, 1998.

[26] M. Größer, G. Norman, C. Baier, F. Ciesinski, M. Z. Kwiatkowska, and D. Parker, "On reduction criteria for probabilistic reward models," in *FSTTCS 2006*, 2006.

[27] F. Ciesinski, C. Baier, M. Größer, and J. Klein, "Reduction techniques for model checking markov decision processes," in *QEST 2008*, 2008.

[28] S. Giro, P. R. D'Argenio, and L. M. F. Fioriti, "Partial order reduction for probabilistic systems: A revision for distributed schedulers," in *CONCUR 2009*, 2009.

[29] P. Godefroid, *Partial-order methods for the verification of concurrent systems: an approach to the state-explosion problem*, 1996.

[30] F. Cassez and F. Ziegler, "Verification of concurrent programs using trace abstraction refinement," in *LPAR 2015*, 2015.

[31] A. Cimatti, I. Narasamdya, and M. Roveri, "Software model checking with explicit scheduler and symbolic threads," *Log. Methods Comput. Sci.*, vol. 8, no. 2, 2012.

[32] A. Valmari, "Stubborn sets for reduced state space generation," in *Advances in Petri Nets 1990*, 1989.

[33] H. Hansen, M. Z. Kwiatkowska, and H. Qu, "Partial order reduction for model checking markov decision processes under unconditional fairness," in *QEST 2011*, 2011.

[34] F. M. Bønneland, P. G. Jensen, K. G. Larsen, M. Muñiz, and J. Srba, "Partial order reduction for reachability games," in *CONCUR 2019*, 2019.

[35] ——, "Stubborn set reduction for two-player reachability games," *Log. Methods Comput. Sci.*, vol. 17, no. 1, 2021.

[36] ——, "Stubborn set reduction for timed reachability and safety games," in *FORMATS 2021*, 2021.

[37] T. Neele, T. A. C. Willemse, and W. Wesselink, "Partial-order reduction for parity games with an application on parameterised boolean equation systems," in *TACAS 2020*, 2020.

[38] T. Neele, T. A. C. Willemse, W. Wesselink, and A. Valmari, "Partial-order reduction for parity games and parameterised boolean equation systems," *Int. J. Softw. Tools Technol. Transf.*, vol. 24, no. 5, pp. 735–756, 2022.

[39] P. A. Abdulla, S. Aronis, B. Jonsson, and K. Sagonas, "Source sets: A foundation for optimal dynamic partial order reduction," *J. ACM*, no. 4, 2017.

[40] ——, "Comparing source sets and persistent sets for partial order reduction," in *Models, Algorithms, Logics and Tools - Essays Dedicated to Kim Guldstrand Larsen on the Occasion of His 60th Birthday*, 2017.

[41] D. Kroening, S. Sharma, and B. Wachter, "Abpress: Flexing partial-order reduction and abstraction," *CoRR*, 2014.

[42] S. Blom and J. van de Pol, "State space reduction by proving confluence," in *CAV 2002*, 2002.

[43] M. Timmer, M. Stoelinga, and J. van de Pol, "Confluence reduction for probabilistic systems," in *TACAS 2011*, 2011.

[44] H. Hansen and M. Timmer, "A comparison of confluence and ample sets in probabilistic and non-probabilistic branching time," *Theor. Comput. Sci.*, 2014.

[45] M. Timmer, J. Katoen, J. van de Pol, and M. Stoelinga, "Confluence reduction for markov automata," *Theor. Comput. Sci.*, 2016.

[46] A. Hartmanns and M. Timmer, "Sound statistical model checking for MDP using partial order and confluence reduction," *Int. J. Softw. Tools Technol. Transf.*, no. 4, 2015.

[47] ——, "On-the-fly confluence detection for statistical model checking," in *NFM 2013*, 2013.

[48] K. L. McMillan, "Using unfoldings to avoid the state explosion problem in the verification of asynchronous circuits," in *CAV 1992*, 1992.

[49] B. Bonet, P. Haslum, V. Khomenko, S. Thiébaux, and W. Vogler, "Recent advances in unfolding technique," *Theor. Comput. Sci.*, 2014.

[50] C. Rodríguez, M. Sousa, S. Sharma, and D. Kroening, "Unfolding-based Partial Order Reduction," in *CONCUR 2015*, 2015.

[51] M. Sousa, C. Rodríguez, V. V. D'Silva, and D. Kroening, "Abstract interpretation with unfoldings," in *CAV 2017*, 2017.

[52] D. Dietsch, M. Heizmann, D. Klumpp, M. Naouar, A. Podelski, and C. Schätzle, "Verification of concurrent programs using petri net unfoldings," in *VMCAI 2021*, 2021.

[53] A. Farzan and A. Vandikas, "Reductions for safety proofs," *Proc. ACM Program. Lang.*, no. POPL, 2020.

[54] M. Swaminathan, J. Katoen, and E. Olderog, "Layered reasoning for randomized distributed algorithms," *Formal Aspects Comput.*, no. 4-6, 2012.

[55] A. Sharma and J. Katoen, "Layered reduction for abstract probabilistic automata," in *ACSD 2014*, 2014.

[56] G. Fox, D. Stan, and H. Hermanns, "Syntactic partial order compression for probabilistic reachability," in *VMCAI 2019*, 2019.

[57] K. Chatterjee, A. Pavlogiannis, and V. Toman, "Value-centric dynamic partial order reduction," *Proc. ACM Program. Lang.*, no. OOPSLA, 2019.

[58] R. Govind, F. Herbreteau, B. Srivathsan, and I. Walukiewicz, "Abstractions for the local-time semantics of timed automata: a foundation for partial-order methods," in *LICS 2022*, 2022.

[59] S. Akshay, T. Meggendorfer, and P. S. Thiagarajan, "Causally deterministic markov decision processes," in *CONCUR 2024*, 2024.

[60] A. Lomuscio, W. Penczek, and H. Qu, "Partial order reductions for model checking temporal-epistemic logics over interleaved multi-agent systems," *Fundam. Informaticae*, vol. 101, no. 1-2, pp. 71–90, 2010.

[61] W. Jamroga, W. Penczek, T. Sidoruk, P. Dembinski, and A. W. Mazurkiewicz, "Towards partial order reductions for strategic ability," *J. Artif. Intell. Res.*, 2020.

[62] M. Heizmann, J. Hoenicke, and A. Podelski, "Refinement of trace abstraction," in *SAS 2009*, 2009.

[63] ——, "Software model checking for people who love automata," in *CAV 2013*, 2013.

[64] P. A. Abdulla, S. Aronis, B. Jonsson, and K. Sagonas, "Optimal dynamic partial order reduction," in *POPL 2014*, 2014.

[65] E. Albert, M. G. de la Banda, M. Gómez-Zamalloa, M. Isabel, and P. J. Stuckey, "Optimal dynamic partial order reduction with context-sensitive independence and observers," *J. Syst. Softw.*, vol. 202, 2023.

[66] F. Herbreteau, S. Larroze-Jardiné, G. Point, and I. Walukiewicz, "Revisiting stateful partial-order reduction," *arXiv preprint arXiv:2411.16921*, 2024.

[67] A. Valmari, "Stop it, and be stubborn!" *ACM Trans. Embed. Comput. Syst.*, vol. 16, no. 2, pp. 46:1–46:26, 2017.

# Towards Configurable Coordination for Distributed Reactive Systems

Richárd Szabó, Dóra Cziborová, András Vörös
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {szabor,cziborova,vori}@mit.bme.hu

***Abstract*—Modern cyber-physical systems (CPS) present unique challenges as they are distributed real-time systems used in many critical application domains, such as automotive or railway systems. However, modeling and verifying the distributed and timed aspects of the system are challenging tasks. We need a precise description of the possible orderings of the components' execution, and a formal representation to be able to run formal verification. In this paper, we investigate the possible extension of a modeling and verification framework to support the flexible, configurable description of the coordination of distributed critical systems. We present an extension of the timed automata formalism, the coordination automata formalism, and show the applicability of the extension on a motivating example.**

***Index Terms*—coordination, distributed systems, reactive systems, case-study, modeling**

## I. Introduction

The modeling and verification of modern cyber-physical systems (CPS) present several unique challenges that arise due to the integration of distributed heterogeneous components. These systems integrate physical systems with HW/SW components across varying platforms and locations. One challenge is to account for the constraints of the system's environment, particularly the timing of distributed components, which can be influenced by the laws of physics (measuring physical properties, e.g., the angle of the steering column or the rotational speed of the wheels), and the communication network between the distributed components (messages can be delayed or lost).

CPSs are often used in critical use-cases, e.g. the railway or the automotive industry. One way to ensure the reliability and correctness of these systems is formal verification using formal descriptions of the system's behavior. These descriptions enable the application of formal verification techniques like model checking, which systematically explores the state space of the system to verify properties such as safety, liveness, and correctness under all possible scenarios. Exploring the whole state space of a system through model checking can be computationally expensive, thus the formal description of the system must be created in such a way that it constrains the possible state space. The constraints of the state space must be defined in such a way that the formal model still conforms to the modeled system. To constrain the behavior of the system we define the *coordination of the system* as the possible and allowed interactions between the subsystems.

In distributed CPSs, the underlying formal models of the communication and the subsystems often vary based on the application domain, network architecture, and system requirements, thus a configurable solution is needed to model the timing constraints of the system.

In this paper, we investigate how the coordination of the subsystems can be modeled in a configurable way. We also show the applicability of our approach through a motivating example.

## II. Background

### A. Modeling Complex Distributed Systems

There are several widely used modeling languages for defining the architecture and behavior of a system. These languages operate at either higher-level, such as UML, SysML, and AADL, or lower-level, such as the timed automata formalism of UPPAAL and our TXSTS formalism (see Subsection II-B). Each of these languages has varying levels of precision in their semantics. To use mathematical tools for the systematic examination of a system's design, a formal model of the system with mathematically precise semantics is needed [1].

One tool focusing on bridging the gap between higher-level (engineering) models and lower-level (formal) models is the *Gamma Statechart Composition Framework* [2], expressly designed for modeling and verifying component-based reactive systems. The framework provides semantic preserving model transformations between higher-level and lower-level modeling languages. The framework supports the hierarchical composition of components, which enables the engineers (1) to break down the system into smaller, reusable subsystems, or (2) connect distributed components to provide higher-level system functionalities.

### B. TXSTS Modeling Formalism

The *timed extended symbolic transition system* (TXSTS) formalism (proposed in [3] as an extension of [4] and [5]) is an intermediate modeling formalism with high-level language constructs suitable for representing complex engineering models. Nevertheless, it is compatible with model checking algorithms that are usually based on low-level formal models.

TXSTS models contain *data variables* to represent data-dependent behavior, while timed behavior is modeled by *clock variables*. Clock variables are continuous, non-negative

DOI https://doi.org/10.3311/MINISY2025-009

variables. They are initialized to zero and incremented equally but can be reset individually.

A *timed extended symbolic transition system* is a tuple $TXSTS = \langle V_D, V_C, V_{ctrl}, val^0, init, env, tran \rangle$ where

- $V_D$ and $V_C$ are finite sets of data variables and clock variables;
- $V_{ctrl} \subseteq V_D$ is a set of *control variables* that may be handled differently by the algorithms;
- $val^0$ is the initial valuation over $V_D$ that maps each variable $x \in V_D$ to the initial value of the variable, or $\top$ if unknown;
- $init \subseteq \mathcal{O}$ is a set of operations representing the *initialization operation set*, it describes more complex initialization that cannot be described by $val^0$;
- $env \subseteq \mathcal{O}$ is a set of operations representing the *environment operation set*, it describes the interactions of the system with the environment;
- $tran \subseteq \mathcal{O}$ is a set of operations representing the *internal operation set*, it describes the internal behavior of the system.

A *state* of a TXSTS model is a tuple $\langle \langle val_D, val_C \rangle, \tau \rangle$, where $val_D$ is a valuation over $V_D$, $val_C$ is a valuation over $V_C$ and $\tau \in \{init, env, tran\}$ is an operation set, which is the only operation set that can be executed in this state.

The operation sets consist of one or more operations taken from a set of operations $\mathcal{O}$. When executing an operation set, the operation to be executed is selected from the operation set in a nondeterministic manner. The set of operations $\mathcal{O}$ contains the following types of operations:

- *Assumptions* have the form $[\varphi]$, where $\varphi$ is a Boolean combination of predicates over $V_D$ and *clock constraints* over $V_C$, with clock constraints being formulas of the form $c_i \sim k$ or $c_i - c_j \sim k$, where $c_i, c_j \in V_C$, $\sim \in \{<, \leq, =, \geq, >\}$, and $k \in \mathbb{Z}$;
- *Data assignments* have the form $x := \varphi$, where $x \in V_D$, and $\varphi$ is an expression of the same type as $x$, containing variables of $V_D$ and $V_C$ (clock variables are restricted to appear only in clock constraints, e.g. in assignments to Boolean variables);
- *Clock resets* have the form $c := n$, where $c \in V_C$ and $n \in \mathbb{N}$;
- *Havoc operations* are denoted by $havoc(x)$, which is a nondeterministic assignment to a data variable $x \in V_D$;
- *Delays* are denoted simply by $delay$, it is a nondeterministic but equal incrementation of all clocks;
- *No-op*: denoted by $skip$;
- *Sequences*: $op_1, op_2, \ldots, op_n$, where $op_i \in \mathcal{O}$ for all $1 \leq i \leq n$, the operations are executed one after the other;
- *Nondeterministic choices*: $\{op_1\}\, or\, \{op_2\}\, or\, \ldots\, \{op_n\}$, where $op_i \in \mathcal{O}$ for all $1 \leq i \leq n$, exactly one operation is executed, chosen in a nondeterministic manner;
- *Conditional operations*: $if(\varphi)\, then\, \{op_1\}\, else\, \{op_2\}$, where $\varphi$ is a Boolean combination of predicates over $V_D$ and clock constraints over $V_C$, and $op_1, op_2 \in \mathcal{O}$;
- *Loops*: $for\ i\ from\ \varphi_a\ to\ \varphi_b\ do\ \{op\}$, where $i$ is an integer variable, $\varphi_a$ and $\varphi_b$ are expressions that evaluate to integers, serving as the lower and upper bound for the loop variable $i$, and $op \in \mathcal{O}$.

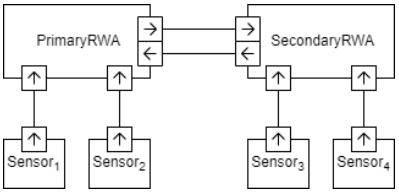



Fig. 1. Architecture of the SbW system

Let $[\![op]\!](\langle \langle val_D, val_C \rangle, \tau \rangle)$ denote the result of applying the operation $op \in \mathcal{O}$ on state $\langle \langle val_D, val_C \rangle, \tau \rangle$. We use the same notation for the result of applying $op$ on some components of a state, e.g. $[\![op]\!](\langle val_D, val_C \rangle)$ denotes applying $op$ on the pair of valuations $\langle val_D, val_C \rangle$.

With $\tau$ denoting the only operation set that can be executed in a state $\langle \langle val_D, val_C \rangle, \tau \rangle$, $[\![op]\!](\langle \langle val_D, val_C \rangle, \tau \rangle) = \emptyset$ if $op \notin \tau$. Otherwise, $[\![op]\!](\langle \langle val_D, val_C \rangle, \tau \rangle) = [\![op]\!](\langle val_D, val_C \rangle) \times [\![op]\!](\tau)$.

The order of the execution of the operation sets is fixed in TXSTS models. The operation set $init$ is executed only once, in the initial state. Sets $env$ and $tran$ are executed in an alternating manner, but only after $init$. In accordance with this consecution of operation sets, $[\![op]\!](init) = \{env\}$, $[\![op]\!](env) = \{tran\}$ and $[\![op]\!](tran) = \{env\}$.

The semantics of operations regarding the $\langle val_D, val_C \rangle$ component of states is straightforward in most cases, therefore we only give the semantics of some operations:

- Assumptions: $[\![[\varphi]]\!](\langle val_D, val_C \rangle) = \{\langle val_D, val_C \rangle\}$ if $\langle val_D, val_C \rangle$ satisfies $\varphi$, otherwise $[\![[\varphi]]\!](\langle val_D, val_C \rangle) = \emptyset$;
- Havoc operations: $[\![havoc(x)]\!](\langle val_D, val_C \rangle) = \{\langle val'_D, val_C \rangle \mid val'_D(x) \in D,\ \forall x' \in V_D \backslash \{x\} :\ val'_D(x') = val_D(x')\}$, where $D$ is the domain of $x$;
- $[\![delay]\!](\langle val_D, val_C \rangle) = \{\langle val_D, val_C^{\Delta} \rangle \mid \Delta \in \mathbb{R}_{\geq 0}\}$ where $val_C^{\Delta}(c) = val_C(c) + \Delta$ for all clocks $c \in V_C$;
- Nondeterministic choices: $[\![\{op_1\}\, or\, \{op_2\}\, or\, \ldots\, or\, \{op_n\}]\!](\langle val_D, val_C \rangle) = [\![op]\!](\langle val_D, val_C \rangle)$ such that $op \in \{op_1, op_2, \ldots, op_n\}$.

## III. Motivating Example: Steer-by-Wire

In our previous work [6], we presented an approach to model time-dependent behaviors in complex distributed systems. We presented the applicability of our approach with a Steer-by-Wire (SbW) system inspired by our industrial partner, first presented in [7].

In this paper, we reuse some components of the SbW case study and extend the case study with additional aspects to motivate our research.

In a SbW system to provide steering functionality the road wheels are actuated via a control loop. Unlike classic Electric Power Assisted Steering (EPAS) systems, in the case of an SbW system, there are no direct mechanical connections

between the steering wheel and the road wheels. The angle of the steering wheel is measured by multiple sensors, and the road wheels are actuated by the *Road Wheel Actuator (RWA)* subsystems, based on the measurements. Since actuating the road wheel is a critical function, a Master Selection Protocol (MSP) was developed to control the actuation of the road wheels. The MSP assigns which of the redundant RWAs must be used to control the road wheels, based on the availability and correctness of their sensor measurements. It must be verified that the MSP cannot select both RWAs as master or that it cannot select a failed RWA as the master, since this selection can lead to accidents.

In the presented case study there are two RWAs and four sensors as depicted in Figure 1.

### A. Challenges in Modeling and Verifying Distributed Systems

Modeling tools with mathematically precise semantics provide the semantics of the modeled components' execution including execution order or scheduling. In the case of Gamma Framework [2] the execution orders of the composition modes are the following [8], [9]:

- The *Synchronous-reactive* composition provides a scheduling where each component is executed simultaneously on a given cycle, samples, and processes their inputs, like a hardware circuit. This scheduling *constrains the state space* of the system but *leaves out important behaviors* of distributed systems, where the clocks of the distributed components differ or are executed one after another.
- The *Cascade* and the *Scheduled asynchronous-reactive* compositions, provide a scheduling, where the components are executed one after another, like a pipeline. This scheduling also *constrains the state space* of the system but *leaves out important behaviors* of distributed systems, where the execution of given components can overlap or happen simultaneously.
- The *Asynchronous-reactive* composition provides *free execution* of the components, there are no constraints on the possible execution order. The lack of a predefined execution order is close to the execution of truly heterogeneous distributed systems but leads to an unmanageable state space, and *introduces unwanted and spurious behaviors* like executing a single component a million times and starving out the rest of the system.

As the examples above show, using the built-in execution orders of the modeling tools can leave out important behaviors of the system or can introduce unwanted and spurious behaviors that are unfeasible or irrelevant in the real system. Introducing the constraints to model the correct system in each modeling tool's language can be a tedious and challenging task.

## IV. Formal Modeling of the Motivating Example

In this section, we propose a new formalism, the coordination automata, to ease the challenges of modeling the possible execution orders of the system. We present the semantics of the coordination automata using the TXSTS modeling formalism, which is an extension of the inner modeling formalism used by the Gamma Framework. Finally, we present the mapping of the coordination automaton of the motivating example to the TXSTS formalism.

### A. Coordination Automata

We propose a new *coordination automata* formalism, which extends timed automata [10] extended with notations referencing other formal models, e.g., TXSTS models. These models can be referenced on the edges of the coordination automaton, meaning the given component is scheduled for execution. Additionally, multiple components can be referenced on a single edge with the use of either the keyword $unord$ or $seq$, denoting whether the referenced models should be executed in an unordered way, or sequentially in the given order.

A coordination automaton uses a set of clocks $C$, which is disjoint from the clock variables of the referenced formal models, i.e., $C \cap V_{Ci} = \emptyset$ for all $1 \leq i \leq n$ in the case of TXSTS models $\{TXSTS_1, TXSTS_2, \ldots, TXSTS_n\}$ with clock variables $V_{C1}, V_{C2}, \ldots, V_{Cn}$, respectively. For a set of clocks $C$, let $\mathcal{G}(C)$ denote the set of clock constraints in the form of $c_i \sim z$ or $c_i - c_j \sim z$, where $c_i, c_j \in C$, $\sim \in \{<, \leq, =, \geq, >\}$, and $z \in \mathbb{Z}$. Furthermore, let $\mathcal{A}(C)$ denote the set of clock assignments in the form of $c_i := z$, where $c_i \in C$ and $z \in \mathbb{Z}$. Then, a coordination automaton over the set of clocks $C$ and the set of model references $\mathcal{M}$ is a tuple $CA = \langle N, n_0, E \rangle$, where

- $N$ is a finite set of nodes;
- $n_0 \in N$ is the initial node;
- $E \subseteq N \times 2^{\mathcal{G}(C)} \times 2^{\mathcal{M}} \times \{unord, seq\} \times \mathcal{A}(C) \times N$ is a set of directed edges.

### B. Network of TXSTS models

As the models referenced on the edges of a coordination automaton form a system of interacting components, time should advance at the same rate in all of the models. Therefore, we define the semantics for a network of TXSTS models, where time advances equally. The states of the network $\langle TXSTS_1, TXSTS_2, \ldots, TXSTS_n \rangle$ are of the form $S_N = \langle S_1, S_2, \ldots, S_n \rangle$, where $S_i$ is a state of $TXSTS_i$. In the following, we give an informal description of the semantics.

In a state $\langle S_1, S_2, \ldots, S_n \rangle$, an operation *op* can be executed only if there exists an $i$ such that $S_i = \langle \langle val^i_D, val^i_C \rangle, \tau^i \rangle$ and $op \in \tau^i$. Executing this operation changes the $\tau$ of only $TXSTS_i$, even if there are multiple candidates for $i$. If no such $i$ exists, then $[\![op]\!]_N(\langle S_1, S_2, \ldots, S_n \rangle) = \emptyset$, where $[\![op]\!]_N(S_N)$ denotes the result of applying an operation *op* on state $S_N$ of the network of TXSTS models.

In the following, we give the semantics of operations regarding the projection of $S_N$ that contains only the data and clock valuations of $S_1, S_2, \ldots, S_n$. For this, let $val(S_i)$ denote $\langle val^i_D, val^i_C \rangle$ if $S_i = \langle \langle val^i_D, val^i_C \rangle, \tau^i \rangle$, and $val(S_N)$ denote $\langle val(S_1), val(S_2), \ldots, val(S_n) \rangle$.

- if *op* is an assumption, data or clock assignment, then *op* is executed on all TXSTS models of the network, i.e., $[\![op]\!]_N(val(S_N)) = \{\langle \langle val^1_D, val^1_C \rangle, \ldots, \langle val^n_D, val^n_C \rangle \rangle \mid$

$\forall\, 1 \leq i \leq n\colon \langle val_D^i, val_C^i \rangle \in [\![op]\!](val(S_i))\}$, to keep shared variables synchronized;

- $[\![havoc(x)]\!]_N(val(S_N))$ is the set of all valuations $\langle\langle val_D^1, val_C^1\rangle, \ldots, \langle val_D^n, val_C^n\rangle\rangle$ such that $val_D^1(x) = val_D^2(x) = \cdots = val_D^n(x) = v$, $v \in D$, where $D$ is the domain of $x$, and $val_D^i$, $val_C^i$ are not changed otherwise for all $1 \leq i \leq n$;
- $[\![delay]\!]_N(val(S_N))$ is the set of all valuations $\langle\langle val_D^1, val_C^1\rangle, \ldots, \langle val_D^n, val_C^n\rangle\rangle$ such that the same delay is applied on all $val_C^i$, $1 \leq i \leq n$;
- for compound operations, such as sequences, nondeterministic choices, conditional operations and counting loops, the semantics is the same as in a simple TXSTS model, e.g., for a nondeterministic choice $[\![\{op_1\}\, or\, \{op_2\}\, or\, \ldots\, or\, \{op_n\}]\!]_N(val(S_N)) = [\![op]\!]_N(val(S_N))$ such that $op \in \{op_1, op_2, \ldots, op_n\}$.

### C. Semantics of Coordination Automata

With the semantics of a network of TXSTS models defined, we can define the semantics of coordination automata, which is similar to that of timed automata. With the set of clocks $C$ and the set of TXSTS models $\{TXSTS_1, TXSTS_2, \ldots, TXSTS_n\}$, the semantics of the corresponding coordination automaton is defined by a transition system with a set of states $\mathcal{S}_{CA} \subseteq N \times Val_C \times \mathcal{S}_N$ where $Val_C$ is the set of clock valuations over $C$, and $\mathcal{S}_N$ is the set of states of the network of TXSTS models $\langle TXSTS_1, TXSTS_2, \ldots, TXSTS_n\rangle$.

In the transition system there are two kinds of transitions:

- An action transition $\langle n, val_C, S_N\rangle \xrightarrow{G,M,o,A} \langle n', val'_C, S'_N\rangle$ where $G \subset \mathcal{G}(C)$, $M \subseteq \mathcal{M}$, $o \in \{unord, seq\}$, $A \subset \mathcal{A}(C)$ is enabled iff the following conditions are satisfied:
  - $\langle n, G, M, o, A, n'\rangle \in E$,
  - $val_C$ satisfies all clock constraints in $G$,
  - $S'_N$ is the result of executing each $\tau^i$ selected by $M$ on $S_N$, in the order given by $M$ if $o = seq$ and in an undefined order if $o = unord$ (a $\tau^i$ is selected by $M$ if $TXSTS_i \in M$ and the state of $TXSTS_i$ in $S_N$ is $\langle\langle val_D^i, val_C^i\rangle, \tau^i\rangle$),
  - $val'_C(c) = z$ if $(c := z) \in A$, otherwise $val'_C(c) = val_C(c) + \Delta$ where $\Delta$ is the sum of all delays applied on $S_N$ to obtain $S'_N$.
- A delay transition $\langle n, val_C, S_N\rangle \xrightarrow{\Delta} \langle n, val'_C, S'_N\rangle$ is enabled iff:
  - $val'_C(c) = val_C(c) + \Delta$ for all $c \in C$, and
  - $S'_N$ is obtained from $S_N$ by replacing each clock valuation $val_C$ in $S_N$ by $val_C^{\Delta}$, similarly to a delay operation.

### D. Modeling Coordination for Distributed Reactive Systems

The coordination automaton of the case study system is presented in Figure 2. At first, the sensor components run and process their inputs according to their possible failure modes. This happens in an unordered way, as they are deployed separately and they cannot influence each other's outputs. After all the sensors produced outputs, either the PrimaryRWA runs first and the SecondaryRWA runs second or vice versa. This ordering can influence the cross-checking between the RWAs and possibly the MSP.

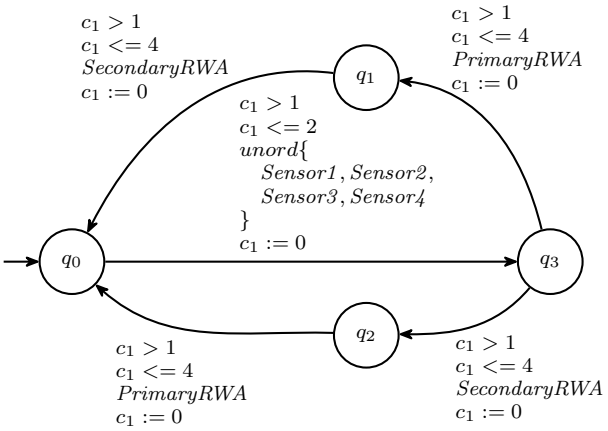



Fig. 2. Coordination Automaton of the SbW System

The TXSTS models representing the components of the system are connected in a single TXSTS model that already implements the coordination described by the given coordination automaton.

The mapping of the coordination automaton to the TXSTS formalism introduces two new variables to the $V_D$ and $V_{ctrl}$ data and control variable sets of the resulting TXSTS. The new variable $scheduled$ represents the component that should be executed in the current execution of the $env$ operation set and the next execution of the $tran$ operation set. The second new variable, $coordState$, represents the current state of the coordination automaton, with the initial value $q_0$ in our case.

In nodes $q_1$ and $q_2$ of the coordination automaton, there is only one possible execution order; in these cases, the mapping to TXSTS is straightforward, as shown in Listing 1.

In node $q_3$ of the coordination automaton there are multiple edges to other nodes, referencing single TXSTS models. The mapping of this case to TXSTS is done using a nondeterministic choice for the possible outcomes, as shown in Listing 2.

For the unordered execution at the edge from $q_0$ to $q_3$ of the coordination automaton we introduce some more variables in the TXSTS. The variable $unord_{q_0,q_3}$ can take values of the references of models on the edge from $q_0$ to $q_3$, i.e., $Sensor1$, $Sensor2$, $Sensor3$ and $Sensor4$. We also introduce a new Boolean variable for each of these components, namely $unord_{q_0,q_3}Sensor1$, $unord_{q_0,q_3}Sensor2$, $unord_{q_0,q_3}Sensor3$ and $unord_{q_0,q_3}Sensor4$. These represent whether the given component was already scheduled at the edge from $q_0$ to $q_3$. All of these newly introduced variables are also included in the set of control variables. The scheduled component is chosen by a nondeterministic assignment to $unord_{q_0,q_3}$, but considering only the components that were not yet scheduled, based on the newly introduced Boolean variables. The coordination automa-

```
if (coordState == q1) {
    assume c1 > 1 && c1 <= 4;
    scheduled := SecondaryRWA;
    c1 := 0;
    coordState := q0;
}
if (coordState == q2) {
    assume c1 > 1 && c1 <= 4;
    scheduled := PrimaryRWA;
    c1 := 0;
    coordState := q0;
}
```

Listing 1. TXSTS representation of deterministic scheduling of components in nodes $q_1$ and $q_2$ of the coordination automaton

```
if (coordState == q3) {
    choice {
        assume c1 > 1 && c1 <= 4;
        scheduled := PrimaryRWA;
        c1 := 0;
        coordState := q1;
    } or {
        assume c1 > 1 && c1 <= 4;
        scheduled := SecondaryRWA;
        c1 := 0;
        coordState := q2;
    }
}
```

Listing 2. TXSTS representation of nondeterministically choosing the scheduled component in node $q_3$ of the coordination automaton

```
ctrl var unord_q0q3 : enum{Sensor1, Sensor2, Sensor3,
     Sensor4}
ctrl var unord_q0q3_Sensor1 : boolean = false
ctrl var unord_q0q3_Sensor2 : boolean = false
ctrl var unord_q0q3_Sensor3 : boolean = false
ctrl var unord_q0q3_Sensor4 : boolean = false

if (coordState == q0) {
    assume c1 > 1 && c1 <= 2;
    havoc unord_q0q3;
    choice {
        assume unord_q0q3 == Sensor1 && !unord_q0q3_Sensor1;
        unord_q0q3_Sensor1 := true;
        scheduled := Sensor1;
    } or {
        assume unord_q0q3 == Sensor2 && !unord_q0q3_Sensor2;
        unord_q0q3_Sensor2 := true;
        scheduled := Sensor2;
    } or {
        ... // Sensor3
    } or {
        ... // Sensor4
    }
    if (unord_q0q3_Sensor1 && unord_q0q3_Sensor2 &&
     unord_q0q3_Sensor3 && unord_q0q3_Sensor4) {
        c1 := 0;
        coordState := q3;
        unord_q0q3_Sensor1 := false;
        unord_q0q3_Sensor2 := false;
        ... // Sensor3, Sensor4
    }
}
```

Listing 3. TXSTS representation of unordered scheduling of sensors in node $q_0$ of the coordination automaton

ton completes the transition to $q_3$ only when all referenced components were already scheduled. Until then, it stays in $q_0$. The TXSTS mapping of this unordered execution can be seen in Listing 3.

Although sequential executions are not present in our example, we propose a method for handling them as well. They

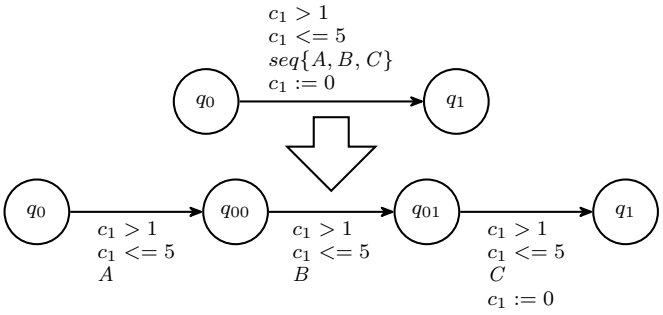



Fig. 3. Transformation of the sequential ordering

should not necessarily be translated directly to the TXSTS formalism, instead, they can be handled by a transformation of the coordination automaton. The transformation substitutes sequential execution edges with a path between the original source and target nodes, where exactly one component is referenced on each edge, in the original sequential order. The original clock constraints are added to all intermediate edges, and the original clock assignments are added only to the last edge. An example transformation can be seen in Figure 3.

The main structure of the resulting TXSTS model can be seen in Listing 4. The above-described mapping of the coordination automaton to the TXSTS formalism is embedded at the beginning of the single operation contained by the $env$ operation set. This is followed by the actual operations of the individual components, in the form of conditional operations $if(scheduled == comp)\,then\,\{env_{comp}\}\,else\,\{skip\}$ selecting the scheduled component, where $env_{comp}$ is the $env$ operation set of component $comp$. If the operation set in question contains multiple operations, then the contained operations are wrapped in a nondeterministic choice $\{env_1\}\,or\,\{env_2\}\,or\,\ldots\,or\,\{env_n\}$ for $env_{comp} = \{env_1, env_2, \ldots, env_n\}$, to represent them as a single operation without changing the possible behaviours of the model.

In the $env$ operation set, delays are applied before and after the execution of the coordination automaton, since time can advance between an internal (or initialization) step of a scheduled component and a step of the coordination automaton, as well as between a step of the coordination automaton and an environment step of a scheduled component.

The $tran$ operation sets of the individual TXSTS models are mapped similarly, using $if(scheduled == comp)\,then\,\{tran_{comp}\}\,else\,\{skip\}$ operations, where $tran_{comp}$ is the $tran$ operation set (or a nondeterministic choice containing all operations of $tran$) of component $comp$.

In the $tran$ operation set, a delay is applied at the beginning, as time can also advance between an environment step and an internal step of a scheduled component.

## V. Related Work

*High-level models:* Lingua Franca [11] is a coordination language to model concurrent reactive components (reactors).

```
ctrl var coordState : enum{q0, q1, q2, q3} = q0
ctrl var scheduled : enum{Sensor1, Sensor2, Sensor3,
    Sensor4, PrimaryRWA, SecondaryRWA}
ctrl var unord_q0q3 : enum{Sensor1, Sensor2, Sensor3,
    Sensor4}
ctrl var unord_q0q3_Sensor1, unord_q0q3_Sensor2, ...
...
env {
   __delay;
   if (coordState == q0) { ... }
   if (coordState == q1) { ... }
   if (coordState == q2) { ... }
   if (coordState == q3) { ... }
   __delay;
   if (scheduled == Sensor1) { <env of Sensor1> }
   if (scheduled == Sensor2) { <env of Sensor2> }
   ... // Sensor3, Sensor4, PrimaryRWA, SecondaryRWA
}
tran {
   __delay;
   if (scheduled == Sensor1) { <tran of Sensor1> }
   if (scheduled == Sensor2) { <tran of Sensor2> }
   ... // Sensor3, Sensor4, PrimaryRWA, SecondaryRWA
}
init {
   <init of Sensor1>
   <init of Sensor2>
   ... // Sensor3, Sensor4, PrimaryRWA, SecondaryRWA
}
```

Listing 4. Structure of a TXSTS model that schedules multiple TXSTS components based on a coordination automaton

Lingua Franca works with both logical time (discrete ordering of events) and physical time (timestamps). The goal of Lingua Franca is to provide a deterministic model of the system by utilizing the priorities of the reactors, the dependencies between the reactors, and the logical and physical timing of events. Even though the goal of the author of Lingua France is to model deterministic systems, nondeterminism is allowed if explicitly required by the engineer. Our framework however builds on nondeterminism to model the coordination of distributed systems in a fair way.

*Low-level formal models:* Similarly to the coordination automata formalism presented in Subsection IV-A, [12] Norström et al. presents an extended timed automata. Their formalism, the task automata, enables the schedulability analysis of tasks with reachability analysis. In [13], the authors present a timed labeled transition system with invariants to define controlling and priorities of applications. The requirements of the system are mapped to specific transitions, thus requiring a white-box model of the system. Our proposed formalism considers the coordinated subsystems as black-box components since the transitions of the subsystems are not modified.

*Coordination models:* According to [14], the goal of a coordination model is to integrate heterogeneous components to provide some higher-level functionality than the individual components. Our goal is similar: providing a coordination of the subsystems in such a way that the integration of the components satisfies the safety goals, provided by the customers and the standards. Furthermore [14], classifies the coordination models as either data-driven or control-driven, and the proposed formalism satisfies the conditions of a control-driven coordination model.

## VI. Conclusion and Future Work

In this paper, we presented the coordination automata formalism as an extension of the timed automata formalism, to model the possible interactions of the distributed subsystems and presented the network of TXSTS models to define the semantics of the coordination automata. We presented the applicability of our modeling formalism on a motivating example inspired by our industrial partner.

The presented coordination automata formalism is flexible and configurable: the coordinated subsystems can be modeled using arbitrary formal or engineering modeling languages.

As a continuation of our work, we investigate the possibilities of extending the coordination automata with more complex parallelism, to define coordination with overlapping component executions.

## References


[1] C. Baier and J. Katoen, *Principles of model checking*. MIT Press, 2008.

[2] V. Molnár, B. Graics, A. Vörös, I. Majzik, and D. Varró, "The Gamma Statechart Composition Framework: design, verification and code generation for component-based reactive systems," in *ICSE '18*. ACM, 2018, pp. 113–116.

[3] D. Cziborová, "Abstraction-based model checking for real-time software-intensive system models," Scientific Students' Association Report, Budapest University of Technology and Economics, 2023.

[4] B. Graics, M. Mondok, V. Molnár, and I. Majzik, "Model-based testing of asynchronously communicating distributed controllers," in *Formal Aspects of Component Software*, J. Cámara and S.-S. Jongmans, Eds. Cham: Springer Nature Switzerland, 2024, pp. 23–44.

[5] B. Graics, M. Mondok, V. Molnár, and I. Majzik, "Model-based testing of asynchronously communicating distributed controllers using validated mappings to formal representations," *Science of Computer Programming*, p. 103265, 2025.

[6] D. Cziborová and R. Szabó, "Modeling of time-dependent behavior in fault-tolerant systems," in *31th PhD Minisymposium of the Department of Measurement and Information Systems*. Budapest University of Technology and Economics, 2024.

[7] R. Szabó, D. Szekeres, S. J. Nagy, Z. Thimár, I. Majzik, Z. Micskei, and A. Vörös, "Iterative exploration of distinct requirement violation scenarios for fault-tolerant architectures," *Submitted*.

[8] B. Graics, V. Molnár, A. Vörös, I. Majzik, and D. Varró, "Mixed-semantics composition of statecharts for the component-based design of reactive systems," *Software and Systems Modeling*, vol. 19, no. 6, pp. 1483–1517, 2020.

[9] B. Graics and I. Majzik, "Integration test generation and formal verification for distributed controllers," in *30th PhD Minisymposium of the Department of Measurement and Information Systems*. Budapest University of Technology and Economics, 2023.

[10] R. Alur and D. Dill, "Automata for modeling real-time systems," in *Automata, Languages and Programming: 17th International Colloquium Warwick University, England, July 16–20, 1990 Proceedings 17*. Springer, 1990, pp. 322–335.

[11] M. Lohstroh, C. Menard, S. Bateni, and E. A. Lee, "Toward a lingua franca for deterministic concurrent systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 20, no. 4, pp. 1–27, 2021.

[12] C. Norstrom, A. Wall, and W. Yi, "Timed automata as task models for event-driven systems," in *Proceedings Sixth International Conference on Real-Time Computing Systems and Applications. RTCSA'99 (Cat. No. PR00306)*. IEEE, 1999, pp. 182–189.

[13] K. Altisen, G. Gößler, and J. Sifakis, "Scheduler modeling based on the controller synthesis paradigm," *Real-Time Systems*, vol. 23, pp. 55–84, 2002.

[14] G. A. Papadopoulos and F. Arbab, "Coordination models and languages," in *Advances in computers*. Elsevier, 1998, vol. 46, pp. 329–400.

# Model-Driven Method for Data Quality Assurance

Nada Akel, László Gönczy
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: nada.akel@edu.bme.hu, gonczy.laszlo@vik.bme.hu

***Abstract*—Ensuring data quality through validation against structural and semantic constraints defined by a specific use case is critical, mainly when data is used to train machine learning models or make accurate analyses and decisions. In this paper, we proposed the usage of Refinery, a framework for generating well-formed models to define data quality constraints, ensuring consistency between them and validating data against them. The proposed approach was evaluated using event trace data formatted in Experience API (xAPI), a learning technology interoperability specification designed to track and share learner activity and experiences. Its triple structure (actor, verb, object), domain profile features, and predefined vocabulary make the xAPI data a good fit for validating the proposed approach.**
***Index Terms*—Data Quality, Event Traces, Experience API, Refinery**

## I. Introduction

Data space is an interoperable framework based on common governance principles, standards, practices, and enabling services. That enables trusted data transactions between participants [1]. The data space should comply with legal frameworks for data sharing, like the European Data Governance Act [2], which enhances trust in voluntary data sharing. Data quality is a key issue in data space frameworks, mainly when the shared data is used to train machine learning models or make decisions. In a data space, data quality requirements are often derived from multiple sources, highlighting the need for a robust data validation framework [3].

A data validation framework ensures data quality by validating datasets against constraints derived from predefined requirements. It identifies issues within the data and provides guarantees or proofs of quality for data exchange and interoperability. In this paper, we propose using the Refinery framework [4] to check the validity of the data against the domain-specific requirements and to check the coherence of these requirements themselves. Additionally, we utilize the framework's graph generation capabilities to generate diverse and consistent instance models that can be used as valid/ invalid test cases, which differs from the methods presented in related literature. The approach is validated using learning experience data in xAPI format to support the data veracity component of the EDGE Skills project [5].

The remainder of this paper is structured as follows. Section 2 overviews the xAPI data and domain modeling with Refinery. Then, we present the proposed application of Refinery in data validation tasks in Section 3. In Section 4, we present the conclusions and future work.

## II. Background overview

### A. Related Work

To validate data against semantic constraints, several tools and approaches can be used, such as Ontology with Semantic Web Rule Language (SWRL), Cognipy with Controlled Natural Language (CNL), and Shapes Constraint Language (SHACL). Rabelo et al. [6] conducted a comparative study of four xAPI validation tools and found SmartLAK, an ontology-based approach, to be the most effective. Vidal et al. [7] proposed an ontology for xAPI data using OWL and SHACL to define constraints and validate data. Their ontology captures the semantics of the xAPI specification with nearly 150 axioms and integrity constraints to ensure correctness. Pareti et al. [8] reviewed SHACL as a constraint language, explaining its basic concepts, constructs, components, and interactions for validating RDF graphs.

### B. Learning Record Data Format (Experience API)

The Experience API (xAPI) standard [9] provides an interoperable way to record and share information about formal and informal learning experiences that happened online and offline. It outlines a format for describing these experiences and explains how the descriptions can be exchanged electronically. The xAPI enables the exchange of learner actions, referred to as "Statements." A statement is a data structure that provides evidence of an experience or event being tracked in xAPI as a Learning Record [9]. A set of statements, each representing a specific event at a given time, can be combined to track the full details of a learning experience. The quality of xAPI data is primarily driven by the requirements defined in the profile.

### C. Domain Modeling with Refinery

Refinery [4] is a novel open-source software framework that automatically generates diverse, consistent domain-specific graph models. The Refinery framework provides a high-level specification language for defining the domain and controlling the range of graphs users request, a semantically well-founded graph generation approach based on refining partial models with 4-valued logic, and a modern cloud-based architecture that includes a partial modeling editor, a partial model reasoner, and a graph solver. We will use Refinery's features to generate models satisfying specific domain data requirements.

DOI https://doi.org/10.3311/MINISY2025-010

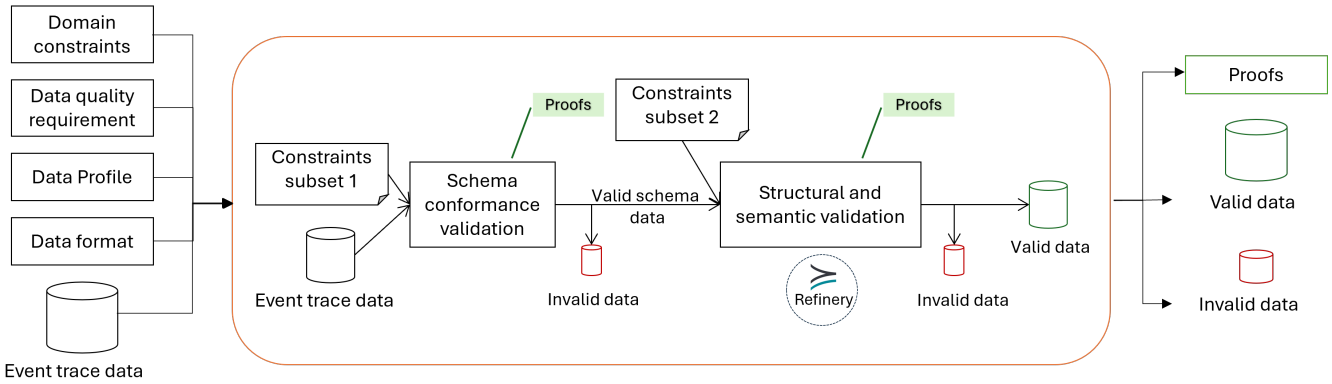



Fig. 1. Event trace data quality framework

## III. Application of Refinery Framework in Data Validation Tasks

This section will first explore Refinery's potential use in data validation tasks, concentrating on proposing a framework to support those use cases to validate event trace data formatted in the xAPI standard. We will detail the metamodel and the integrity constraints that formalize the provided xAPI profile specification. In this paper, we will focus on the core elements of the xAPI statement, and later, we can expand to include all the model elements.

### A. Potential Usage of Refinery in Data Validation Tasks

The following are three key objectives of using Refinery, each supporting a different step in building a data quality framework.

- **Goal 1: Checking Data Requirements Consistency.** Refinery helps us determine if data requirements are consistent and contradictions-free by mapping each to model constraints. If contradictions exist, Refinery identifies the issue as unsatisfiable because it cannot generate models that satisfy the conflicting constraints. While it does not show the exact cause of the problem, it indicates that the data requirements and rules are inconsistent.
- **Goal 2: Supporting the Validation of Data Veracity Component.** The Refinery framework can generate example models that conform to the rules of a given metamodel and predefined requirements. The generated models can serve as resources for creating valid and invalid test cases, which can be used to evaluate data quality solutions or validate the functionality of different versions of data veracity components where the primary use case of Refinery is synthesizing graphs as test cases [4].
  Although Refinery supports the evaluation of numerical attribute constraints to determine their compliance with specified rules, the ability to generate these numerical attributes remains an area of ongoing research [10].
- **Goal 3: Model Validation.** Once the domain specification is provided to Refinery and a concrete model is defined, the Refinery framework ensures that the model complies with the defined metamodel and meets all predefined constraints, representing specific data requirements. Additionally, the framework identifies errors or inconsistencies within the model. This ability to find the mistakes in models helps maintain the quality and reliability of the data.

### B. Event Trace Data Quality Framework

Figure 1 provides an overview of our proposed data quality framework for validating event trace data. This framework takes three main inputs: (1) the trace of the event, which represents the data itself; (2) the data format, which can be any commonly used format for representing event trace; and (3) multiple sources of data quality requirements. Where those requirements can arise from different sources, such as constraints enforced by the data format, domain-specific rules, insights from data profiling, data quality standards like ISO/IEC 25000 [11], ISO 8000 [12], or those explicitly defined in data exchange agreements. These requirements create a comprehensive set of criteria the data must satisfy to ensure adherence.

Our proposed framework uses Refinery to validate trace event data. To achieve this, we first have to provide the domain specification to Refinery [4] by defining the metamodel of the xAPI statement and the set of structural and semantic predicates and constraints that ensure the xAPI data conforms with our profile requirements. Then, we will validate the data of the statements against the defined metamodel and constraints. It is worth noting that before starting the validation process, we assume our xAPI data is syntactically valid, which means it is a well-formed JSON object, as shown in Figure 1.

### C. Event Trace Validation with Refinery

*1) Statement graph model:* **Statement** forms the main structure of the xAPI data format. Each statement should contain three core required elements: an **Actor** is an individual or group; Who the statement is about or who made the action; a **Verb** is the action taken; and an **Object** is something with which an actor interacted. The object could be an **Activity**,

**Agent**, **SubStatement**, or another statement that is the object of the statement. The statement can also contain additional properties like **Context**, including further details, which gives the statement more meaning; **Result**, representing a measured outcome; **Attachments** that are part of the learning record **Authority** and which means the agent or group asserting this statement is true [9]. A set of statements can collectively describe a learning scenario if they share contextual identifiers that define their relationship. However, two statements from the same actor may represent actions in different scenarios. Figure 2 shows the proposed metamodel of the xAPI statement. Each metamodel class represents a specific element of the statement, and we used references to describe the relationship between the classes. We also used multiplicity constraints to constrain the required and optional fields of the statement with their cardinality.

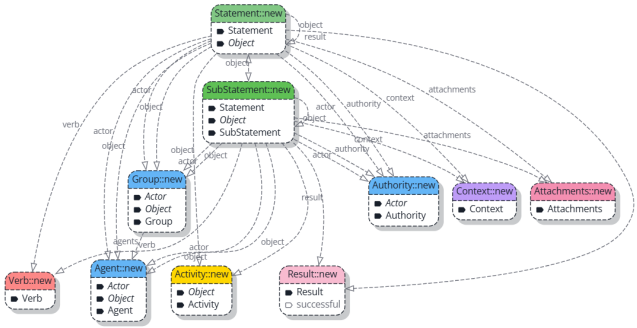



Fig. 2. xAPI statement metamodel in Refinery.

*2) Integrity constraints:* To comply with the xAPI specification, we must introduce certain restrictions to our metamodel to ensure that the generated models satisfy all structure and semantics data requirements.

- **Structural Requirements:** The following structural requirements defined in the xAPI specification GitHub repository [13] serve as key examples:
  - Req1: a SubStatement MUST specify an "objectType" property with the value SubStatement.
  - Req2: a SubStatement MUST be validated as a Statement and other SubStatement requirements.
  - Req3: a SubStatement MUST NOT have the "authority" properties.
  - Req4: a SubStatement MUST NOT contain a SubStatement of its own.

  Requirements **1** and **2** can be satisfied using the Refinery **inheritance hierarchy** during the metamodel development. A substatement declaring the object class as a superclass means a substatement could serve as an object for another statement. At the same time, the substatement inherits the statement class. Ensuring that substatement is validated against all statement requirements.
  Requirements **3** and **4** can be enforced by applying constraints to the metamodel with Refinery **error predicates**, where each requirement is mapped to a specific error predicate. This allows further restriction of the range of valid graphs where a consistently generated model, such as error predicates, should have no matches [4]. This satisfies the generation of valid test cases, as outlined in Refinery's **goal 2**. Following is an example of the error predicates that satisfy data requirements 3 and 4; figure 3 illustrates the errors generated when defining a concrete model that violates these structural requirements. Specifically, errors are highlighted when a SubStatement contains the "authority" property or includes another SubStatement as its object. This example aligns with the model validation outlined in Refinery's **goal 3**.

```
error pred 'Sub-statements
cannot have authority: '
    (SubStatement sub) <->
    authority (sub,_).

error pred 'Sub-statement
cannot be nested:' (SubStatement sub,
    SubStatement o ) <->
    object(_,sub),
    object(sub,o),
    SubStatement(o).
```

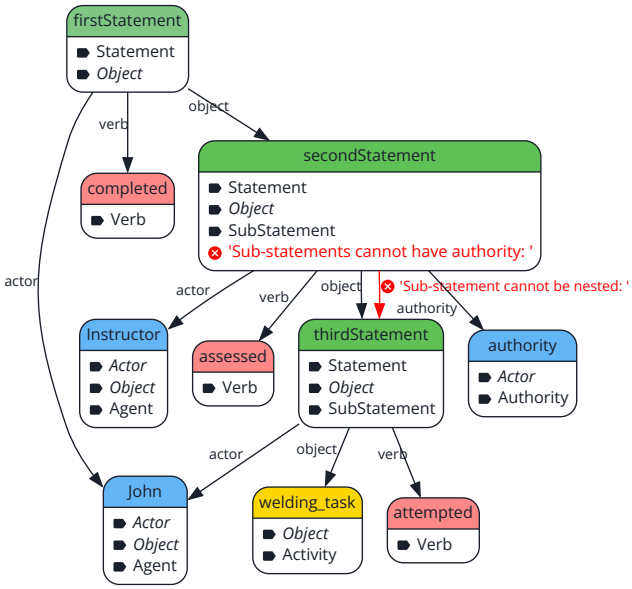



Fig. 3. Example of Structural Constraints Check.

- **Semantic Requirements:** In addition to structural compliance, semantic data requirements ensure that the logical meaning of data aligns with the outcomes of learning activities. Consider a scenario involving learning traces from VR activities that capture, record, and analyze user interactions within a virtual environment. Automated analysis methods are increasingly important to evaluate these interactions, making physical training safer and cheaper. For example, in a welding learning experience, each xAPI statement represents a specific activity per-

formed during the session. A semantic data requirement for such a scenario could be:

- Req5: A welding activity marked as ”completed” with a successful result must ensure that no catastrophic events (e.g., ”fire” or ”explosion”) were recorded during the session.

This requirement implies that any model containing a welding activity marked as ”completed” with a successful result must ensure that no catastrophic events were recorded during the session. To achieve this using Refinery, we must list catastrophic verbs representing unsafe or dangerous situations. Then, we need to write the error predicate that prevents this in cases. Following is the error predicate that satisfies this semantic data requirement; Figure 4 below illustrates an error generated when defining a model for an activity marked as completed despite the student causing a fire. This inconsistency highlights the importance of aligning activity outcomes with the actual events recorded during the session. The described scenario represents another example of the model validation outlined in Refinery's **goal 3**.

```
error pred 'Error: Success is not possible
after a catastrophic action:'
(Statement s1, Statement s2) <->
    result(s1,r),
    successful(r),
    verb(s2,v),
    CatastrophicVerb(v),
    actor(s1, a),
    actor(s2, a).
```

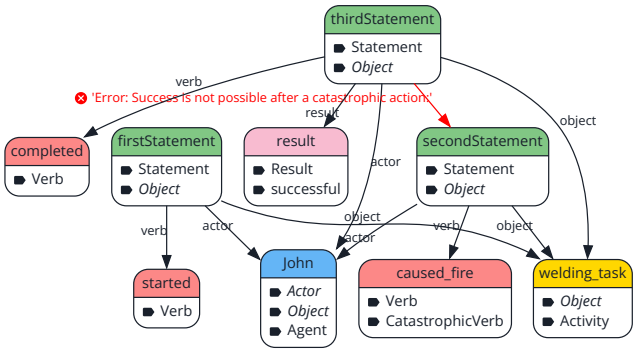



Fig. 4. Example of Semantic Constraints Check.

## IV. Conclusion and Future Work

This paper examined how data can be validated against constraints derived from data quality requirements arising from diverse sources (figure 1), ensuring compliance with these requirements. We have explored the potential of Refinery with graph queries in data validation. We have presented the xAPI metamodel's main elements, including predicates and constraints that capture the semantic structure of the core xAPI statement specification. We illustrated how Refinery can validate a concrete model representing the xAPI statement data against the predefined metamodel and error predicates, which implements goal 3 of Refinery usage. Goals 1 and 2 are part of ongoing work. We intend to extend the metamodel with additional xAPI rules that provide the conformance of an xAPI model, that is, indicate if the model is non-conformant, against best practices, or completely conformant, exploring a hybrid framework to combine Refinery's strengths with complementary approaches to build a complete data syntactic, structural, and semantic validation, Integrate our work with existing standards in the skills and education domain, such as taxonomies and ontologies like ESCO [14], these could help specify, e.g., the necessary/forbidden verbs.

## V. Acknowledgements

This work has been partially supported by the European Union in the frame of the project European Dataspace for Growth and Education – Skills, short: EDGE Skills (101123471)

## References

[1] CEN-CENELEC Workshop Agreement, “CWA 18125:2024 - Trusted Data Transaction,” CEN-CENELEC, Tech. Rep., 2024, accessed: 2025-01-21. [Online]. Available: https://www.cencenelec.eu/media/CEN-CENELEC/CWAs/RI/2024/cwa18125_2024.pdf

[2] (n.d.) Data governance act. European Commission. Accessed: 2024-12-18. [Online]. Available: https://digital-strategy.ec.europa.eu/en/policies/data-governance-act

[3] M. Altendeitering, S. Dübler, and T. M. Guggenberger, “Data quality in data ecosystems: Towards a design theory.” in *AMCIS*, 2022.

[4] K. Marussy, A. Ficsor, O. Semeráth, and D. Varró, “Refinery: Graph solver as a service: Refinement-based generation and analysis of consistent models,” in *Proceedings of the 2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings*, 2024, pp. 64–68.

[5] Prometheus-X Association, “Data veracity design document,” 2024, accessed: 2024-12-18. [Online]. Available: https://github.com/Prometheus-X-association/data-veracity/blob/main/docs/design-document.md

[6] T. Rabelo, M. Lama, J. C. Vidal, and R. Amorim, “Comparative study of xAPI validation tools,” in *2017 IEEE Frontiers in Education Conference (FIE)*. IEEE, 2017, pp. 1–5.

[7] J. C. Vidal, T. Rabelo, M. Lama, and R. Amorim, “Ontology-based approach for the validation and conformance testing of xAPI events,” *Knowledge-Based Systems*, vol. 155, pp. 22–34, 2018.

[8] P. Pareti and G. Konstantinidis, “A review of SHACL : from data validation to schema reasoning for RDF graphs,” *Reasoning Web International Summer School*, pp. 115–144, 2021.

[9] IEEE, *9274.1.1 xAPI Base Standard Overview*, n.d., accessed: 2024-12-15. [Online]. Available: https://opensource.ieee.org/xapi/xapi-base-standard-documentation/-/blob/main/9274.1.1%20xAPI%20Base%20Standard%20Overview.md

[10] O. Semeráth, A. A. Babikian, A. Li, K. Marussy, and D. Varró, “Automated generation of consistent models with structural and attribute constraints,” in *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*. New York, NY, USA: Association for Computing Machinery, 2020, p. 187–199.

[11] ISO, *ISOIEC 25024:2015 Standard: Systems and Software Engineering-Systems and Software Quality Requirements and Evaluation (SQuaRE)-Measurement of Data Quality*. ISOIEC, 2015.

[12] *ISO/TS 8000 Standard, “Data Quality and Enterprise Master Data https://www.iso.org/committee/54158/x/catalogue”*.

[13] Advanced Distributed Learning Initiative, “xAPI specification - statement properties,” 2024, accessed: 2024-12-15. [Online]. Available: https://github.com/adlnet/xAPI-Spec/blob/master/xAPI-Data.md#24statement-properties

[14] M. le Vrang, A. Papantoniou, E. Pauwels, P. Fannes, D. Vandensteen, and J. De Smedt, “Esco: Boosting job matching in europe with semantic interoperability,” *Computer*, vol. 47, no. 10, pp. 57–64, 2014.

# Evaluation of Embedded AI Through Model Difference Analysis

András Gergely Deé-Lukács, András Földvári, András Pataricza
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {andrasgergely.dee-lukacs, andras.foldvari}@edu.bme.hu
patericza.andras@vik.bme.hu

***Abstract*—The growing reliance on embedded AI components in critical systems demands robust mechanisms for explainability and reliability. These systems often integrate highly complex, opaque models whose decision-making processes are difficult to interpret, posing significant challenges to debugging and trustworthiness. This paper introduces an approach that allows examining regions identified through model comparisons, specifically focusing on areas where interpretable surrogate models and opaque models diverge or produce inconsistencies. By analyzing these regions, the paper provides actionable insights for identifying edge cases and mitigating risks associated with model inaccuracies.**

**This paper leverages qualitative abstraction techniques to translate complex model behavior into comprehensible representations, enabling systematic evaluation of discrepancies. By focusing on the intersection of model behavior and system-level impact, the proposed methodologies offer a scalable approach for enhancing both the dependability and interpretability of AI-enabled systems. The findings advance the state of explainable AI and contribute to the development of safer, more transparent applications in critical domains.**

***Index Terms*—explainable AI, qualitative reasoning, qualitative model extraction**

## I. Introduction

In the rapidly advancing field of artificial intelligence (AI), integrating explainability into complex systems has emerged as a critical requirement. Embedded AI components are increasingly utilized in systems where their outputs significantly impact decision-making processes, posing challenges such as difficulty in debugging, limited trust due to lack of transparency, and the risk of performance issues in critical scenarios. However, the opacity of these opaque models poses challenges for debugging, trust-building, and performance evaluation. To address this, surrogate models and qualitative reasoning techniques have gained prominence, offering interpretable approximations of complex AI systems.

This paper focuses on analyzing regions identified through model comparison, specifically examining areas where interpretable surrogate models and complex, opaque models converge, diverge, or exhibit inconsistencies better to understand their significance for system-level reliability and improvement. The research explores how these insights can inform system evaluation, debugging, and improvement by examining these regions- where models agree, disagree, or fail. The primary goal is understanding how these regions can be leveraged to enhance system performance, identify edge cases, and mitigate risks. By leveraging qualitative abstraction and surrogate modeling, this paper introduces a framework for evaluating model differences and understanding model behavior in a systematic and interpretable manner. The methods highlighted in this paper provide actionable strategies for addressing discrepancies.

This research was founded by DigitalTech EDIH DIGITAL-2021-EDIH-01 and DOSS HORIZON-CL3-2022-CS-01 projects.

## II. Qualitative Abstraction

Qualitative Modeling [1] (QM) represents and reasons about continuous aspects (quantities, motion, space, time) in a symbolic, human-like way. Unlike differential equations demanding precise numerics, QM abstracts information to intuitively understand changes without specific numeric knowledge. QM aims to enable symbolic reasoning about continuous aspects of systems, bridging human intuition and formal mathematical models. A key objective is clustering continuous values into discrete categories, making reasoning more intuitive and interpretable.

### A. Clustering Continuous Values

Let $X \subseteq \mathbb{R}$ be the domain of a continuous feature, Q the *qualitative domain* of the clustered feature so that $Q = \{q\}, Q \subset \mathbb{Z}$, and $I = I_1, I_2, ..., I_n \subset \mathbb{IR}$ real intervals [2]. These intervals represent *partitions* of the feature domain:

$$\forall I_i, I_j : I_i \cap I_j = \emptyset; \bigcup_i I_i = X \tag{1}$$

These partitions are selected in a way that the values they contain represent a *similar behavior* in the system. This means these elements only differ in aspects that are not as relevant. Relevance is defined by the task at hand to be resolved using the qualitative model. There's an *interval membership function* that simply assigns each number to an interval.

$$mb : X \longmapsto I : mb(x) = I_i \iff x \in I_i \tag{2}$$

The $cf$ clustering function assigns a qualitative value to an element, the value corresponding to a one-on-one mapping with an interval partition of the feature domain.

$$x \in X : cf(x) \in Q \wedge cf(x_1) = cf(x_2) \iff \exists I_i : mb(x_1) = mb(x_2) \quad (3)$$

## III. Explainable AI

Machine learning models are increasingly complex, leading to opaque decision-making. The need for explainability arises from legal "right to explanation" [3], ethical concerns, trust-building, and model debugging. These factors drive *Explainable AI* (xAI) and *Explanatory Model Analysis* [4], aiming to create verifiable AI systems [5]. Exploratory Model Analysis systematically examines a model's structure, outputs, and behavior. Explanation methods include *interpretable-by-design* models (e.g., decision trees [6], linear/logistic regression, rule-based methods [7], [8]), *model-agnostic* approaches (e.g., LIME [9], SHAP [10]), and *model-specific* techniques (e.g., feature importance in random forests).

### A. Surrogate Models

A surrogate model is an abstract interpretable-by-design xAI model that approximates the behavior of an advanced, opaque model and describes it using a ruleset. Surrogate modeling is similar to a formal method called *abstract interpretation* [11] that approximates a program's behavior by mapping its concrete states to abstract domains.

*1) Decision Tree:* The decision tree [6] (DT) is one of the simplest and most popular ML algorithms created in the early days. Although it is simple, it is surprisingly accurate and very versatile. It can be ideally used to solve both classification and regression problems. A decision tree (DT) recursively splits a dataset until a stopping condition (e.g., depth limit or a subset with only one label) is met, creating a tree whose vertices represent split conditions, edges represent true/false branches, and leaves represent predictions.

A *node* $v \in V_{tree}$ represents a point in the tree where a decision is made. Each node is associated with a *feature* $feature(v)$, which is an attribute of the dataset used to make the split, and a *value* $val(v)$, which determines the threshold for splitting. A *split condition* at node $v$ is $splitcondition(v) := feature(v) < val(v)$. Its children $v_{\text{true}}, v_{\text{false}}$ split samples based on whether $feature(v)$ is less than $val(v)$.

A *path condition* of node $v$ is the conjunction of edge conditions (split condition on true edge, negated otherwise) from the root to $v$. A *ruleset* is the disjunction of path conditions of leaves that predict a positive label:

$$ruleset(T) = \bigvee_{\substack{v_i \in V_{tree} \\ deg(v_i)=1}} pathcondition(v_i). \quad (4)$$

*2) Boolean Rules via Column Generation:* The Boolean Rules via Column Generation (BRCG) algorithm [7], implemented in the AIX360 toolkit [12], efficiently generates CNF or DNF rules from a data set by framing Boolean decision rule learning as a linear program. It minimizes the number of false positive and false negative classifications.

Given a training dataset $D = \{(x, y)^n\}$, that can be partitioned into $P \cup Z$, where $P$ and $Z$ contains the indices of the positive and negative samples. Let $K$ denote the collection of all possible clauses to be used in the ruleset, and $K_i$ denote the clauses satisfied by sample $i$. $\xi_i \in \{0, 1\}$ denotes if sample $i$ was misclassified, $w_k$ denotes whether clause $k$ was used in the ruleset, and $c_k$ denotes the complexity of clause $k$. $C$ is a parameter that bounds the complexity of the ruleset to prevent overfitting. The *master integer program* (MIP) minimizes:

$$z_{MIP} = \textbf{min} \quad \sum_{i \in P} \xi_i + \sum_{i \in Z} \sum_{k \in K_i} w_k \quad (5)$$

subject to constraints ensuring positive sample coverage, bounded complexity, and binary clause usage.

The objective function (5) has two components. The first component it tries to minimize is the number of misclassified positive samples, i.e., the false negative samples. The second component to be minimized represents the number of clauses that satisfy negative samples, in other words, the number of clauses that can lead to false positive predictions.

Since solving the MIP directly is computationally infeasible for real-world datasets, a *column generation* (CG) approach is used. Initially, a *restricted master linear* problem (RMLP), a linear programming (LP) relaxation of the MIP, is solved with a small subset of simple clauses. A *pricing problem* then iteratively identifies and adds clauses with the most *negative reduced cost* to improve the RMLP objective. This process repeats until no clause with a negative reduced cost remains, yielding the optimal ruleset.

### B. Interpretable Model Differencing

To compare the approximate model with the embedded model, a solution is needed that can reveal the differences between the two models in an understandable and illustrative manner. This is particularly important because, in the case of complex machine learning models, differences often remain hidden, making it challenging to determine where and why their decisions diverge. To address this, the Interpretable Model Differencing (IMD) algorithm [13] enables model comparison by constructing a *Joint Surrogate Tree* (JST), which simultaneously represents both models, visually identifies agreements and disagreements, and presents contradictions in the form of rules.

The process begins by training two models $M_1, M_2 : X_{data} \longmapsto Y_{data}$ with the same dataset. Their outputs are used to label the data as $Y_{M_1}, Y_{M_2}$. An IMD instance, a binary classifier $imd : X_{data} \times Y_{M_1} \times Y_{M_2} \longmapsto Y_{imd}$, is trained to predict whether the models contradict each other for a given sample based on input features and model predictions.

A JST is a decision tree with standard decision nodes and special *OR-nodes*, where the models diverge in their behavior. Decision nodes can be *common* (shared split conditions) or model-specific. Subtrees of OR-nodes no longer share conditions. Leaves in the JST hold model predictions, which may be *pure* (single label) or *impure* (mixed labels, resolved by

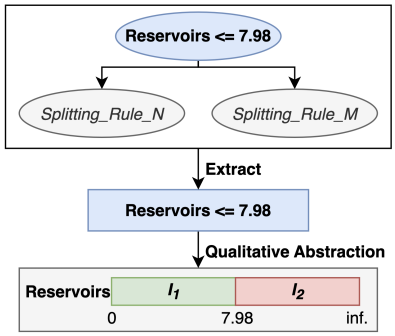



Fig. 1. Example Qualitative Model Extraction from Tree Structures

majority voting). The JST is built recursively. At each node, the algorithm chooses between creating a common node or an OR-node based on impurity measures. An OR-node is created if the following condition fulfills:

$$impurity(X_{data}, Y_{M_1}) + impurity(X_{data}, Y_{M_2}) \leq \alpha \cdot impurity(X_{data}, Y_{M_1}, Y_{M_2}) \quad (6)$$

where $\alpha \leq 1$ controls the trade-off.

A difference rule can be extracted from a JST by selecting an OR-node, two leaves from its two subtrees with different labels, and conjunction the path conditions of these leaves.

$$\begin{aligned} diffrule = \{&pathcondition(leaf_1) \wedge \\ &pathcondition(leaf_2) : \\ &leaf_i \in leaves(v_{OR}), i = 1, 2 \\ &label(leaf_1) \neq label(leaf_2)\} \end{aligned} \quad (7)$$

### C. Ruleset to Qualitative Model

The core of xAI modeling lies in combining simple rules that accurately approximate the dataset. Our research builds on the observation that widely used models rely on combinations of simple inequalities to partition behavioral domains using logical expressions. Transforming these inequalities into qualitative variables serves as a general principle for deriving qualitative counterpart models. More specifically, the explanations of these models can be represented as a tree graph, where each node corresponds to a decision rule expressed as a split condition (an inequality). These inequalities can be directly converted into a qualitative model.

Each value in these inequalities in a ruleset represents a *qualitative landmark*. This means the continuous domains of the features can be clustered by partitioning them into intervals using interval logic [2] along the split values. After that, each partition is assigned a *qualitative value*, which is named using domain knowledge. This process is called *qualitative abstraction*. Let $feature \in \mathbb{R}_+$ be a feature and $R$ be a simple rule containing one split condition: $R = \{feature < val\}$. In this case, the domain can be partitioned into $I_1 = (-\inf, val)$ and $I_2 = [val, +\inf)$. To these intervals, $q_1$ and $q_2$ qualitative values can be assigned.

As an example, in a ruleset built by training a rule-based model on the MetroPT-3 dataset [14], there is a condition: $[Reservoirs <= 7.98]$. After qualitative abstraction using the process described above, we can say that the $Reservoirs$ feature could have two qualitative values: $LOW$ and $HIGH$.

Converting a ruleset into a qualitative model means we apply this process to all of the variables using all of the conditions in the ruleset. The clustering of overlapping conditions is solved using interval logic.

## IV. Model Evaluation with Difference Model

The aim is to examine the primary model using a surrogate model that matches the data precisely. Next, we employ a model difference method to evaluate the performance of the primary model by comparing its outputs with those of the surrogate model and the labels of the test dataset.

Through this comparison, we can pinpoint the specific regions or conditions in which the primary model underperforms. Identifying these areas of weakness helps us determine which corrective or improvement measures to apply. We also take into account both the magnitude of the errors and the severity of their potential consequences, ensuring that any refinements target the most critical issues effectively.

### A. xAI Surrogate Model

Advanced *opaque models* often dominate in production because they typically outperform simpler, more interpretable models. They excel at handling complexity by learning intricate patterns, including nonlinear relationships in high-dimensional data. However, their complexity makes it hard to understand how decisions are reached and to debug errors when they arise. A classic example is a neural network with fine-tuned parameters trained on clean, standardized data.

Analyzing complex models is challenging due to limited interpretability. We address this by approximating their behavior with a simpler, interpretable surrogate model. This approach replaces expected behavior representations with an abstract surrogate model that covers positive samples in n-dimensional space (Figure 2). This hybrid, model-agnostic method provides a clearer understanding of the original model's behavior.

In our research, we adopt an *interpretable-by-design xAI model* as the surrogate. Apart from interpretability — which allows transformation into a qualitative model — high sensitivity is critical for covering all positive samples. Achieving 100% coverage is typically unrealistic, so techniques like oversampling, boosting, or cost-sensitive learning are employed to enhance model sensitivity.

### B. Interpretation of Model Differences

As a further step of the analysis, a difference model can be built on top of the surrogate and the primary models. If we evaluate this model while also taking into account the labels of the training dataset, the samples can be divided into

three categories: 1) Certain solutions; 2) Uncertain solutions; 3) False solutions.

*Certain solutions* represent the so called *stable zones*, where both models predict correctly. These scenarios form the basis of the normal operation and the stability of xAI surrogate models. In our research, we have not paid much attention to them. However, they can be used to examine the *characteristic samples* found in regions or clusters where models are performing confidently or consistently well. This analysis can be useful in examining which are the best learned rules.

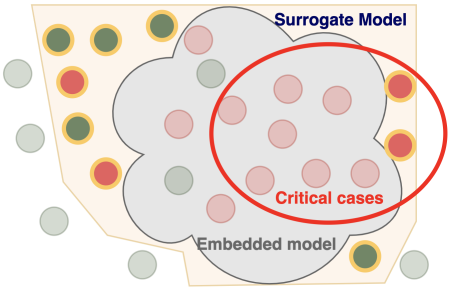



Fig. 2. Surrogate Model and Boundary Region

*Uncertain solutions* contain samples where the models contradict each other. They help identify the edge cases, representing the boundary region between the positive and negative samples (Fig. 2). These can be partitioned into two subcategories: a) *Surrogate Divergence* represents those samples where the surrogate model produces a misprediction and the primary model is correct. They are mostly the consequence of an abstract or overly simplified surrogate model. If the number of these cases is too high, the level of abstraction can be directly tuned by controlling the complexity of the surrogate model. (e.g. setting the depth of a DT). The ideal and most probable scenarios represented by these are *false alarms* (false positive samples), as the goal of the surrogate model is to cover all positive samples. The occurrence of false negative predictions of the surrogate model is very rare. Ideally, there should be none. The existence of these samples is because there is no perfectly sensitive model in practice. However, the question of to what extent a perfectly sensitive surrogate model is needed can be further discussed.

b) *Primary Model Divergence* represents those cases where the primary model mispredicts, and the surrogate model is correct. These cases must be analyzed as these carry the greatest risks in a real application. These scenarios represent the most of the escaping faults, it must be examined whether they are critical. Less often, but it may happen that the real label is disputed, and the embedded model predicts (partly) right. This phenomenon is called *data drift* [15].

*False Solutions* contains samples where both models mispredict. A high number of these cases can often indicate data or feature problems. These cases can also result from model problems such as underfitting, overfitting, or poor generalization. In this case, we assume that the labels in the training and test datasets are correct. This paper does not address scenarios where errors occur during the labeling process.

### C. Evaluation of the Difference Regions

Our goal of model evaluation using difference models is, firstly, to set up priorities regarding the samples, i.e., the critical cases are given special attention, and secondly, to incorporate feedback into the development cycle. This means an improved cost function that makes the model more accurate to the critical cases, the examination of the underrepresented samples, and the minimization of the data drift. Examining each region is a multi-step process: 1) Identify regions relevant from the perspective of errors or inconsistencies (e.g., JST leaves, data slicing, clustering). 2) Understand why and what types of errors occur in those regions. 3) Assess the risk (evaluate the cost or potential hazard associated with the specific type of error). 4) Develop an action plan (e.g., data cleaning, model fine-tuning, introducing new features, modifying surrogates).

To identify the regions, the difference model can be fitted, and its results compared with the original labels of the test dataset. These results can then be used to segment the dataset.

Various methods can be employed to evaluate the regions, depending on the type of region in question. By fitting qualitative rules to the segmented data points, it is possible to analyze the types of conditions that occur in those regions based on the operational model. This empirical approach helps domain experts gain a general understanding of the region.

Additionally, by comparing the qualitative rules across multiple regions, potential overlaps can be identified. When an operational domain appears in multiple regions, it may indicate an issue with the model or suggest that the engineering model is incomplete.

Once these regions are identified, they can be used for exploratory data analysis (EDA). EDA helps visualize whether the region truly differs from others (e.g., extreme values, rare events, clusters with varying densities). It also allows the detection of *outlier* points or patterns where the model responds incorrectly. This type of anomaly detection contributes to understanding the error profile of the regions (e.g., records typically having missing values or extreme feature values).

During EDA, whether the regions (e.g., leaf nodes in a JST) are sharply separated or exhibit a continuous transition can be examined. This helps assess the artificial boundaries between regions (e.g., created by simple decision rules) and how much they reflect genuine, distinct segments within the data.

Regions critical from a dependability perspective (e.g., extreme values, high-cost errors) can be easily visualized using EDA, allowing domain experts to gain insights quickly. For instance, it can be assessed whether a specific anomalous region truly results from extreme behavior in the actual process or is instead due to measurement or annotation errors.

Error Propagation Analysis (EPA) [16] offers the capability to evaluate the model within the context of a larger system. Since the model's decisions propagate within the system (e.g., supporting specific decisions), it is necessary to assess whether

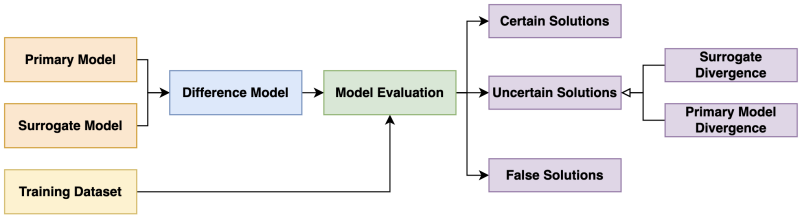



Fig. 3. Solutions of the Model Difference Evaluation

a correct or incorrect decision in a given system state could lead to critical system-wide failures.

EPA provides the opportunity to handle errors with weighted importance. For example, a "False Negative" in a region with few occurrences but high risk (e.g., fraud, hazardous situations) can be more severe than a frequently occurring but low-risk error. Beyond the frequency of errors, the cost or critical impact of the errors can also be incorporated into the analysis.

During the evaluation, the domain expert can assign risk to different regions and operational modes, considering the results of both EDA and EPA. This list can then be used to guide efforts on improving the model and determining which safeguards should be implemented in the system to prevent the model's errors from causing system-wide failures. The insights gained from the evaluation can be used to improve the models by identifying operational regions where performance was suboptimal. Qualitative rules enable impact analysis of the model within its embedding system context, allowing these results to be integrated into the refinement process.

Furthermore, understanding the operational domains creates opportunities to enhance the data collection process or to artificially adjust the existing dataset (e.g., in cases of imbalanced datasets). This ensures that errors are mitigated in subsequent training iterations, reducing the likelihood of incorrect predictions.

## V. Example

The example is demonstrated using the MetroPT-3 dataset [14], which contains data from the Air Production Unit (APU) component of a metro train, including attributes such as pressure, temperature, and motor current. The dataset labels indicate whether the component was faulty at a given time.

As a first step, we trained a neural network tasked with predicting errors in the APU. In the second step, we built a surrogate model based on the training data, designed to accurately capture the faulty cases. Finally, we used the IMD algorithm to compare these models and matched the results against the labels in the test dataset. The resulting regions were then evaluated, and qualitative rules were derived using the JST, providing a foundation for further analysis.

### A. Training a Neural Network with Preprocessed Data

As a highly advanced model, a neural network was used as a failure prediction model trained on the MetroPT-3 dataset. This neural net consists of two hidden layers having 64 neurons and the activation function of *relu* and an output layer of a single neuron using the *sigmoid* activation function. This model was then trained on clean (subsampled and removed outliers) standardized data.

### B. Surrogate Model Creation

Our method to build a sensitive model was to iteratively train a DT with cost-sensitive learning, boost the weight of false negative (FN) samples, and repeat. We trained a DT using class weights of $1/11$ for negative samples and $10/11$ for positive samples, then multiplied the weight of FN samples by $1.5$ and repeated the cycle $10$ times.

The resulting model has an extraordinarily high sensitivity, covering all positive samples except one. However, the side effect of low specificity also can be seen as the false positive rate increased by $268\%$.

### C. Regions from Difference Model

Table I summarizes the number of samples that fall into each solution region. From the table, it is evident that the majority of samples belong to a certain solution category. The remaining samples represent cases where either one or both models made incorrect predictions. In the following, we will examine the data points that fall into the "Primary divergence" region, as these points may indicate operational domains where the primary model needs to perform well to ensure coverage of multiple operational modes, particularly in critical system applications. Additionally, we will analyze the cases where both the surrogate and the primary models produced incorrect predictions.

### D. Evaluation

We fit a RIPPER model to explain those scenarios where the predictions of the primary model diverge with a rule set in CNF form. For instance, one rule describes these system states as:

$$\begin{aligned}(P_{discharge\ valve} \geq -0.01) \ \wedge (P_{discharge\ valve} < 1.684)\\ (T_{oil} \geq 66.6) \ \wedge (T_{oil} < 70.225)\\ (P_{pneumatic\ panel} < 6.86)\end{aligned} \tag{8}$$

The qualitative model of this rule is shown in Figure 4.

We used EDA to investigate the False region in order to identify operational domains where the model underperforms. During the analysis, the `H1` variable emerged as a notable

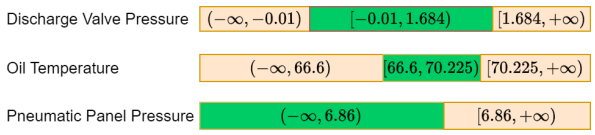



Fig. 4. Qualitative Divergence Rules

TABLE I
SURROGATE MODEL EVALUATION - REGIONS

| Solution type | Primary | Surrogate | Label | # of samples |
|---|---|---|---|---|
| Certain | 0 | 0 | 0 | 292282 |
| | 1 | 1 | 1 | 5955 |
| Uncertain Primary divergence | 1 | 0 | 0 | 47 |
| | 0 | 1 | 1 | 90 |
| Uncertain Surrogate divergence | 0 | 1 | 0 | 3736 |
| | 1 | 0 | 1 | 1 |
| False | 1 | 1 | 0 | 1279 |
| | 0 | 0 | 1 | 0 |

factor. In Figure 5, the distribution of `H1` across the entire dataset is shown in red, while its distribution within the False region is highlighted in blue. It is evident that the blue values are concentrated in the lower range. This pattern was observed for several other variables as well, enabling us to delineate the problematic domain using this method.

## VI. CONCLUSION AND FUTURE WORK

This paper presented a systematic approach for analyzing and evaluating embedded AI components in critical systems through model differencing and surrogate modeling. The approach focuses on identifying and addressing areas of divergence between interpretable surrogate models and opaque primary models, providing actionable insights for improving system performance and dependability. By leveraging qualitative abstraction and explainable AI techniques, the proposed framework enables domain experts to identify edge cases, assess risks, and refine models to mitigate critical issues. The case study utilizing the MetroPT-3 dataset demonstrates the application of the approach.

In future work, we will examine the use of the results from explanatory model analysis [4] for validating and improving the rule set derived from the surrogate model. In this paper, we neglect the ideal requirement for absolute sensitivity (no false negatives) combined with high specificity (few false alarms) and adopt a best-effort approach, as they do not affect the core algorithm. Another approach under consideration is Inselberg's nested cavities [17], which can ensure perfect sensitivity—at least for the training set—and then gradually refine specificity. The method employs a convergent series of upper and lower approximations driven by Reed-Mueller normal form logic rather than the usual AND-OR constructs.

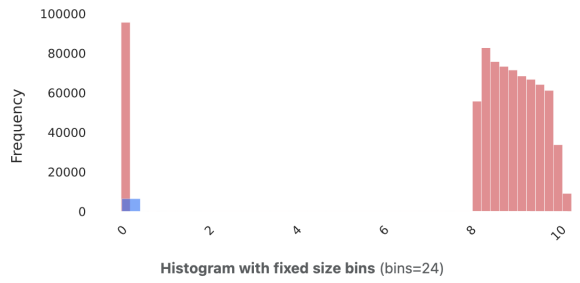



Fig. 5. Evaluation of the H1 Feature

## REFERENCES

[1] K. D. Forbus, "Chapter 9 qualitative modeling," in *Handbook of Knowledge Representation*, ser. Foundations of Artificial Intelligence, F. van Harmelen, V. Lifschitz, and B. Porter, Eds. Elsevier, 2008, vol. 3, pp. 361–393. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S157465260703009X

[2] B. Kubica, *Interval Methods for Solving Nonlinear Constraint Satisfaction, Optimization and Similar Problems: From Inequalities Systems to Game Solutions*, 01 2019.

[3] B. Goodman and S. Flaxman, "European union regulations on algorithmic decision making and a "right to explanation"," *AI Magazine*, vol. 38, no. 3, p. 50–57, Sep. 2017. [Online]. Available: http://dx.doi.org/10.1609/aimag.v38i3.2741

[4] P. Biecek and T. Burzykowski, *Explanatory Model Analysis*. Chapman and Hall/CRC, New York, 2021. [Online]. Available: https://pbiecek.github.io/ema/

[5] D. Gunning, E. Vorm, J. Y. Wang, and M. Turek, "Darpa's explainable ai (xai) program: A retrospective," *Applied AI Letters*, vol. 2, no. 4, p. e61, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ail2.61

[6] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, pp. 81–106, 1986.

[7] S. Dash, O. Günlük, and D. Wei, "Boolean decision rules via column generation," 2020.

[8] W. Cohen, "Fast effective rule induction," *Twelfth International Conference on Machine Learning: 1995*, vol. 95, 10 2000.

[9] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," 2016. [Online]. Available: https://arxiv.org/abs/1602.04938

[10] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 4765–4774. [Online]. Available: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[11] P. Cousot, "Abstract interpretation based formal methods and future challenges," ser. Lecture Notes in Computer Science, R. Wilhelm, Ed. Springer-Verlag, 2001, pp. 138–156.

[12] V. Arya, R. K. E. Bellamy, P.-Y. Chen, A. Dhurandhar, M. Hind, S. C. Hoffman, S. Houde, Q. V. Liao, R. Luss, A. Mojsilović, S. Mourad, P. Pedemonte, R. Raghavendra, J. Richards, P. Sattigeri, K. Shanmugam, M. Singh, K. R. Varshney, D. Wei, and Y. Zhang, "One explanation does not fit all: A toolkit and taxonomy of ai explainability techniques," 2019. [Online]. Available: https://arxiv.org/abs/1909.03012

[13] S. Haldar, D. Saha, D. Wei, R. Nair, and E. M. Daly, "Interpretable differencing of machine learning models," 2023. [Online]. Available: https://arxiv.org/abs/2306.06473

[14] V. B. R. R. Davari, Narjes and J. Gama, "MetroPT-3 Dataset," UCI Machine Learning Repository, 2023, DOI: https://doi.org/10.24432/C5VW3R.

[15] S. Ackerman, O. Raz, M. Zalmanovici, and A. Zlotnick, "Automatically detecting data drift in machine learning classifiers," *arXiv preprint arXiv:2111.05672*, 2021.

[16] A. Földvári and A. Pataricza, "Handling uncertainity in error propagation analysis," in *Proceedings of the 30th Minisymposium*. Budapest University of Technology and Economics, 2023, p. 29–32. [Online]. Available: http://dx.doi.org/10.3311/minisy2023-008

[17] A. Inselberg, *Parallel Coordinates: Visualization, Exploration and Classification of High-Dimensional Data*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 643–680. [Online]. Available: https://doi.org/10.1007/978-3-540-33037-0_25

# Extending Bayesian Networks with Large Language Models for Interactive Semantic Explanations

Benedek Ágota, Tamás Mészáros
Budapest University of Technology and Economics,
Department of Artificial Intelligence and Systems Engineering,
Budapest, Hungary
Email: agotabenedek@edu.bme.hu, meszaros@mit.bme.hu

***Abstract*— Bayesian Networks (BN) in medical diagnostics have proven successful. However, domain experts often struggle to interpret them and their results, which limits practical adoption. Previous solutions aiming to overcome this issue failed to provide semantic explanations with dynamic interactivity. This paper presents a new, LLM-based method to augment Bayesian Networks that relies on an earlier BN explanation algorithm and semantic annotations to overcome these issues. Users can input evidence and query the BN as to why it came to certain results. Based on the query, an explanation is generated, and relevant semantic information is gathered from the annotations to enrich the explanation. This is then passed as context to the LLM to answer the user's query similarly to the method of retrieval augmented generation. Based on the method, the authors implemented a prototype system with a BN for dementia diagnosis and evaluated its ability to convey the BN's knowledge and results.**

***Keywords—Bayesian Networks, Large Language Models, Semantic Explanations, Retrieval Augmented Generation***

## I. Introduction

Bayesian Networks (BN) are quite popular for medical diagnostic tasks. Their success is not surprising, as BNs can combine expert knowledge and learning from data to create probability distributions and probabilistic reasoning capabilities [1].

BNs are quite transparent due to their white-box nature. However, many find the reasoning of BNs counterintuitive and the models overly complex [2], [3], [4]. These factors lead to mistrust towards BNs' diagnostic results, and that can cause underutilization of BNs. While there have been numerous efforts [2] made to solve the explainability problem of BNs through various BN explanation methods, there are still unresolved issues in the sphere.

There has been a turn toward natural language explanations in recent years, as visual explanation methods have not led to significant results [5]. Textual explanations also have the benefit of not requiring the user to know anything about Bayesian Networks making these approaches more general and approachable. Despite the promising possibilities, results produced by textual explanation systems have still not been fully satisfactory, as current approaches offer limited interactivity and poor, rigid verbalization of results. [2]

Adjacent to the explainability problem is that BNs are often built with serious amounts of expert knowledge, which is seldom exploited after the model has been finished [6]. This is unfortunate, as semantic information related to BNs could be beneficial to explanatory solutions. There have been previous attempts at mitigating these issues. However, these solutions remain underutilized. [6]

Another tool domain experts can turn to are large language models (LLM). LLMs offer the ease of natural language interactions through chat interfaces and answers tailored to the user's specific query. However, LLMs struggle to consistently produce reliable information, and as a solution, the technique of retrieval augmented generation (RAG) has emerged where the LLM is extended with an information retrieval (IR) system [7], [8].

To leverage the diagnostic capabilities and domain knowledge of BNs, we propose a method along the lines of RAG to combine BNs and LLMs. The goal is to make BNs and their probabilistic results interpretable, utilize their domain knowledge, and provide natural language interaction for them.

## II. Related Works

### A. Bayesian Network explanations

#### 1) Taxonomy

Lacave and Diez [5] summarized the early research of BN explanations and laid out a comprehensive taxonomy for the explanation of Bayesian networks that is still used today:

**Explanation of the model**

Explanation of the structure, connections, and parameters of the network. This can be particularly useful since Bayesian Networks are often built with the help of domain experts and probabilities so that domain knowledge can be encoded in the model. This implicit knowledge might not be evident to all users and developers.

**Explanation of evidence**

Which configuration of unobserved variables is most likely to have resulted in a given set of evidence? Usually, this is done by finding the most probable explanation (MPE), meaning the configuration of variables that likely resulted in the observations.

**Explanation of reasoning**

How a certain result was obtained in the network and the reasoning behind it. This can be extended to include why certain results were not obtained by the Bayesian Network. Hypothetical reasoning can also be explained, which entails how the BN could have come to different conclusions given a different set of evidence.

#### 2) Algorithms

Early attempts focused on general explanations of BN's and their inferences in various textual and visual forms such as BANTER, Elvira, B2, DIAVAL [5]. These approaches have mostly been deemed insufficient [2]. Modern algorithms tend to focus more on specific variables of interest and posteriors [9]. One of the latest notable algorithms in the medical domain was presented by Kyrimi et al. [4]. It is a three-level explanation method for medical BNs. In the first level of the explanation, significant evidence variables are identified given variables of interest; in the second level, how the information flows through intermediate variables to the target from the evidence is analyzed; and on the third level,

DOI https://doi.org/10.3311/MINISY2025-012

the influence of evidence on the intermediate variables is explained. The results are in natural language along with numerical information. This incremental explanation method has been evaluated by medical professionals with mixed results. However, it showed that the system provides similar reasoning to that of clinicians and that the system influences their decision-making while being mostly clear.

This is a good example of an explanation method that accurately shows the BN's results. However, these are still static explanations lacking deeper semantics. They cannot be focused on specific areas of interest or tailored to a domain expert's individual needs.

### B. Bayesian Networks and annotations

BNs capture and formalize certain elements of domains; however, a lot of domain knowledge is left out of these models that is semantic and often in textual form. This knowledge could be valuable for users and knowledge engineers working with these systems. To resolve this challenge, Antal et al. [6] introduced the annotated Bayesian Network (ABN). The ABN is BN extended with an ontology and textual information (e.g., annotations, documents) attached to relations, variables, values, probability distributions, and various parts of the model. Thus, the ABN can be viewed as a semantic BN and document store. ABNs are able to aid in the generation of explanations. When a prediction is made using the BN, annotations can be accessed for the most active evidence variables and paths of influence. The found information thus provides an explanation for the BN's reasoning. ABNs are also applicable in information retrieval tasks. ABNs can be applied in query transformations as they provide a domain ontology capable of specializing and generalizing search terms. In the ABN-IR method, the ABN model was further extended to support context creation for information retrieval, enabling the indexing of ABN elements and their quantitative combination using the ABN-IR language to generate contexts for IR [10]. The ABN-IR method further defines keyword-based indexing of the domain documents, and the ABN-IR language can be used to query those documents.

### C. Features of good natural language explanations

A good natural language explanation is written with a clear purpose and narrative structure for a well-defined audience while addressing uncertainties and data issues as outlined by Reiter [11].

The explanation having a clear purpose means it has a communicative goal. Such a goal could be to help users understand how the AI came to its conclusions and build trust in the system. In essence, the purpose is what the explanation wants to change about how the user thinks.

A defined audience is important because different sets of users will interpret an explanation in other ways. This usually means using terminology, content, and style the users are already familiar with. For example, a medical doctor might not be familiar with the term acyclic graph, but she understands the cause and effect between a disease and a symptom.

Narrative structure means that the explanation consists of key messages, and these messages are linked by causal, argumentative, or other discourse relations. This way of communicating is better understood by humans than listing numbers and facts. For example, for a BN, an accurate explanation is listing all the evidence and the diagnosis result, but it is hardly comprehendible to humans. Instead, we should communicate the important evidence and the way they contributed to the result.

Communicating uncertainty entails making the user understand what certain probabilities mean beyond black-and-white, yes-and-no thinking. This is a notoriously hard problem with no clear solutions yet [11]. It is also important to communicate how the model was built and on what data since it could mean that certain biases can be present in the model. For example, a BN that assesses the risk of dementia developed in the US based on American data might be quite different from a model developed in South Africa, as certain environmental factors are quite different.

### D. Retrieval Augmented Generation

One of the biggest limitations LLMs face is generating factually incorrect content known as hallucination. The problem becomes even more significant in knowledge-intensive and domain-specific tasks [7]. To overcome this issue, the method of RAG has emerged and was able to remedy the issue.

RAG overcomes the challenge of generating factually incorrect content by supplying the LLM additional information at inference by employing some sort of information retrieval system. So, when the user queries the LLM, the first relevant information is fetched and given to the LLM as context to answer the user's question, like an open book exam.

There are countless different RAG architectures ranging from simple to highly advanced that involve multiple steps in how this information is found and how the final answer is generated. RAG typically works with textual data like articles and wikis, usually referred to as documents. Supplying entire documents to LLMs might not be suitable for all LLMs, as they might not fit into the LLM's context window. To overcome this challenge, documents are chunked based on a chunking strategy, and then these chunks are indexed, stored, and retrieved. Fig. 1 shows a simple RAG architecture that is able to retrieve these document chunks.

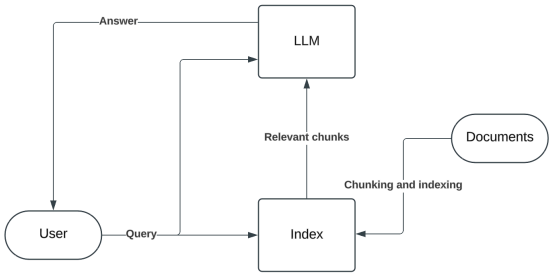



Fig. 1. A general RAG architecture.

A very popular approach to indexing and retrieval is to turn chunks into embeddings with an embedding model. This means the text is turned into a high-dimensional vector representation. Embeddings are useful since these high-dimensional vectors model semantics like different topics and relations well. When looking for relevant information, a user's query is also embedded, and with various algorithms, relevant chunks can be retrieved based on similarity to the query. Embeddings are stored and retrieved with vector databases. Chunks can also be stored and searched with more classical search platforms like Solr [12]. Search engines index the documents based on their keywords and the frequency of those and retrieve chunks based on keyword matches from the query. Hybrid approaches use keyword and vector-based search together.

A subtype of RAG called Graph RAG works with graph data or graph-based indexes [13]. In text-only RAG, usually, entire documents or chunks of various sizes are retrieved. With graph data, a more diverse set of objects can be retrieved, as graphs have many different components like nodes and edges. Indexing can also be complicated, but a simple solution is to convert the graph's parts into textual representations, and then those can be stored and searched like document-based RAG.

## III. A Method for Interactive Semantic Explanations

Our method for interactive semantic explanations of BNs relies on a BN explanation algorithm like the one described in section II.A.2 and uses its output as context so an LLM can better customize the explanation to fit a user's need and provide interactivity. This approach is similar to RAG in the way that it relies on an external system to create text based on which an LLM can respond. However, by itself, this approach will not be able to provide deeper insights into the semantic reasons behind a diagnosis due to the limitations of BN explanation algorithms. To overcome this issue, the method presented has to further enhance explanations with semantic context.

Explanation algorithms only describe how a model came to its conclusions but not necessarily why it came to those. This means that while a generated explanation can reveal to the user statements such as "the patient likely has Huntington's disease because young-onset dementia was previously present in the family" it cannot tell the user why that is. All types of BN explanations (model, reasoning, evidence) require semantic extensions to be able to answer these questions. A semantically enhanced answer to the previous one might go like this "the patient is likely to develop Huntington's disease as she has a family history of young-onset dementia, which is a major indicator of risk because Huntington's is caused by certain gene mutations that are hereditary". However, this type of knowledge is left out of BN models. Nonetheless, BNs can be augmented with semantic knowledge, as seen with annotated BNs [6] (ABN) described in more detail in section II.B. In our method, annotations form the basis of a RAG solution that finds relevant information, similar to the ABN-IR [10] approach, to semantically enhance explanations. Explanations provide the probabilistic reasoning logic, and RAG enriches it with semantic information. This enhanced context is used by an LLM to give valid explanations of reasoning and communicate to clinicians with semantic understanding. Fig. 2 illustrates how such a method can improve BN explanations. Systems based on this method would be able to generalize to any audience since, via prompting, LLMs can easily tailor their outputs to different user groups.

The method defines an Explainer and a Retriever component for generating explanations and semantically enhancing them, respectively. These components rely on an ABN and an LLM to realize their functions and produce their results. The proposed method realizes ABNs' potential for generating explanations [6] and updates the ABN-IR method for modern RAG.

Evidence can be input in structured form for the BN. The user can input natural language queries, which are first handled by the Explainer and then Retriever. A high-level overview of this architecture can be seen in Fig. 3.

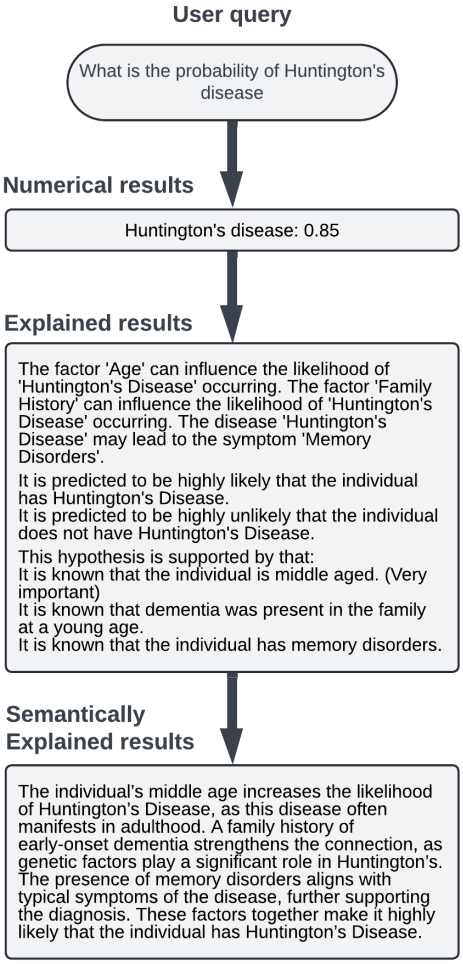



Fig. 2. How explanations can make the numerical results of BNs more knowledge-dense and how semantic explanation enhances them further.

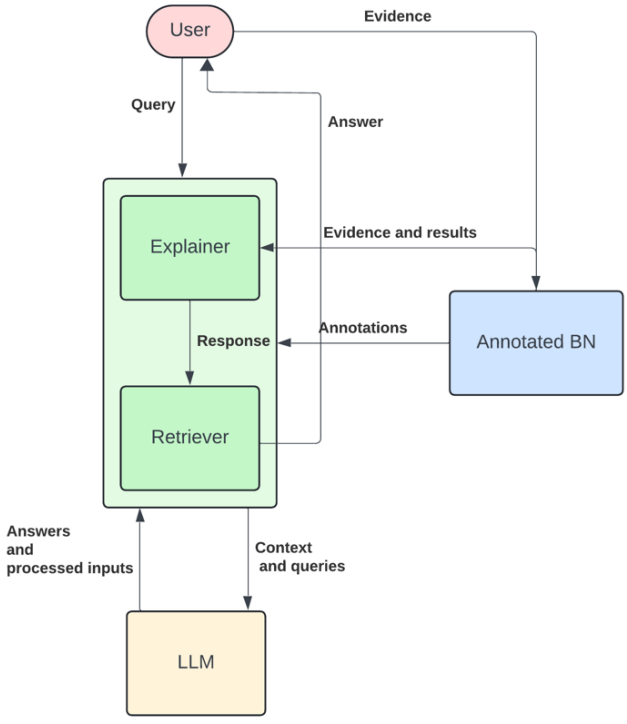



Fig. 3. Architecture of a system based on the method.

**Example #1:**

**Query:** “What is the likelihood of Huntington's disease?”

**Evidence:** "Lifestyle": "High risk", "Family History": "Dementia was present at a young age"

**Explainer context**: “ […] It is predicted to be highly likely that the individual has Huntington's Disease. This hypothesis is supported by that: It is known that the individual has movement disorders. [...]”

**Explainer response:** “The individual's family history of dementia at a young age and their middle age are significant risk factors that strongly support the likelihood of Huntington's Disease. [...]”

**Retrieved information:** “Huntington's disease, which is inherited in an autosomal dominant manner. This means that if one parent carries the gene responsible for the disease, there is a 50% [...]”

**Retriever response:** “Based on the provided information, the likelihood of the patient having Huntington's disease is assessed as high. This conclusion is drawn from the following key factors: Family History: A family history of early dementia significantly increases the risk of genetic conditions like Huntington's disease, which is inherited in an autosomal dominant manner [...]”

**Example #2:**

**Query:** “What examination should I run next?”

**Evidence:** "None"

**Explainer context:** “CT is unobserved and is an examination. MRI is unobserved and is an examination [...]”

**Explainer response:** “The factor 'Imaging Examination' can influence the likelihood of 'Dementia' being diagnosed. The factor 'Structural Brain Changes' can influence the likelihood of 'Dementia' being detected. dementia [...]”

**Retrieved information:** “Imaging examinations in dementia diagnosis help detect structural brain changes, such as atrophy in the hippocampus [...]”

**Retriever response:** “I recommend running imaging examinations like CT and MRI because they haven't been carried out yet and they are essential for detecting structural brain changes, such as hippocampal atrophy, commonly associated with dementia. These scans help rule out other potential causes of cognitive decline […]”

Fig. 4. Examples of the inputs and outputs of the different components.

### A. Annotations

The annotations are defined by the ABN method. Additionally, the annotations must include helper templates for the Explainer component's verbalization process, like the ones in Fig. 5. How the templates are formulated is largely dependent on the Explainer’s underlying explanation algorithm.

**Examples of relation templates:**
The examination '{target}' can provide evidence regarding the presence of '{parent}'.
The factor '{parent}' can influence the likelihood of '{target}' occurring.
The disease '{parent}' may cause the symptom '{target}'.
The symptom '{parent}' could be indicative of '{target}'.

**Examples of evidence templates:**
Memory Disorders:
- Yes: "It is known that the individual has memory disorders."
- No: "It is known that the individual does not have memory disorders."

Family History of Dementia:
- Positive Evidence: "It is known that dementia was present in the family at a young age."
- Negative Evidence: "It is known that dementia was not present in the family at a young age."

Fig. 5. Examples of helper templates.

### B. Explainer

The Explainer is responsible for generating explanations of the BN with the help of an algorithm that is capable of generating explanations for the BN and the LLM.

The Explainer gets the user’s query as input and identifies which variable the user is interested in. This selection is done by the LLM relying on an ontology defined above the nodes of the BN. The ontology is part of the BN’s annotations, it defines classes and relations between the nodes. This makes it possible to handle ambiguous queries like “What is the probability of all diseases?” where no individual variable can be identified, but using the ontology, the variables with the type of “disease” then can be queried.

In the next step, either the probability of the selected nodes is calculated or, if they are present as evidence, their state is “retrieved”. Then a textual description (explanation) is generated as to what could have led to these obtained results. This is achieved by an explanation algorithm (like the one presented in section II.A.2) that is modified to rely on helper templates from the annotations illustrated in Fig. 5. We do not constrain what algorithms can be adapted for our method as long as they can generate explanations for individual nodes. As to what is described and what type of explanations (model, evidence, reasoning) are supported, it is dependent upon the underlying explanation algorithm; for example, the one discussed in section II.A.2 only supports explanations of reasoning. Additionally, the generated description is independent of the BN inference methods and is verbalized in a way that people not familiar with BNs could understand.

In the last step, the original query and the output of the explanation algorithm are passed to the LLM as context to construct the Explainer’s response. Fig. 4 shows examples of the Explainer’s context and final output generated for different inputs. While these explanations are already more understandable, the Retriever is needed for semantically rich and natural explanations that incorporate knowledge beyond the BN model.

### C. Retriever

The Retriever’s responsibility is to semantically enhance the initial explanation provided by the Explainer. It achieves this by relying on an ABN and reworking the ABN-IR method described in section II.B, for a modern RAG-based approach.

The original ABN-IR method relied on an extended keyword-based query language, we instead rely on the natural language understanding capabilities of modern LLMs and text-embedding models to find relevant nodes. The ABN-IR method retrieved entire documents, in RAG, document chunks are retrieved for reasons discussed in section II.D.

The Retriever gets the user’s query and the response of the Explainer as inputs. Based on these, it selects nodes that could hold relevant document chunks. This is done by the LLM using an ontology defined above the BN similar to how variable selection is carried out in the Explainer.

This initial selection of nodes is expanded by their relevant neighboring nodes. A neighbor is deemed relevant based on the similarity between the textual embeddings of the connecting edge's name and the query. This set is not expanded further.

In the next step, document chunks are retrieved from the selected nodes. For finding relevant chunks, various search and retrieval augmented generation techniques can be used, such as keyword- or embedding-based search and hybrid solutions.

Finally, the retrieved information and response of the Explainer are given as context to the LLM to answer the user’s query, Fig. 4 shows examples of this.

## IV. Evaluation

The authors constructed a BN for dementia diagnosis that consists of 17 variables, based on expert knowledge from a medical textbook. It was annotated with an ontology for

diagnostics and medical articles from the web. This only served as a "mock" ABN, which, while easy to annotate, does not fully capture the true nature of a highly complex BN with many variables. However, for such BNs, there are no already existing annotations and evaluation datasets for our purposes, and creating large diagnostic ABNs was outside the scope of this paper. Furthermore, the evaluation did not focus on the accuracy or validity of the underlying dementia diagnosis BN itself. Evaluation was carried out from the perspective of how approachable the method makes the BN and how accessible it makes the knowledge stored inside the BN and its annotations.

Where it was possible, already established metrics and automatic methods were used, but in other areas, we had to rely on human assessment by the authors. Evaluation by domain experts and users is left as future work. While human assessment is error-prone and hard to scale, developing evaluation methodologies for the Explainer component is a substantial effort, which falls outside the scope of this paper, as there are no ready-to-use datasets for evaluating ABNs, their textual explanations, or natural language explanations in general.

To provide a comprehensive overview, the two components were evaluated separately and in combination. For the Explainer the explanation algorithm discussed in section II.B.2 was used, and GPT-4o-mini[14] was used as the LLM for the prototype.

### *A. Explainer*

The results of the BN must be communicated in a way that is easily understood and accurately represents the BN's results. Defining good metrics for the evaluation of explanations is not a trivial task, and it is still an active research area [11]. While trustworthy automatic metrics do not yet exist, we relied on criteria for good natural language explanations already discussed in section II.C. Based on these criteria, a checklist was compiled to evaluate the responses of the system by the authors. Individual responses to queries can be checked whether they conform to the criteria in the list in a yes-or-no manner. From that, an average can be calculated for the correct answers for each criterion to showcase how well the Explainer performs on them. Modern LLMs are expected to excel in communicative and information-extraction tasks like this. For deeper evaluation, a more complex ABN and dataset would be required.

**Checklist:**

- **Agreement:** The explanation has to be in agreement with the numeric results of the BN.
- **Goal achieved:** The user's query is successfully addressed.
- **Appropriate language:** The terms and "language" of the domain are being used.
- **Narrative:** The output text is narratively communicated.
- **Uncertainty communicated:** Probabilities are clearly communicated.

For the evaluation, a dataset consisting of diagnostic questions with scenarios was composed. Scenarios are made up of sets of evidence and questions. The dataset contains 7 scenarios about diagnosing 7 types of dementias and 2 questions about which diseases are most likely and what are the expected results of unobserved variables. Furthermore, we asked 4 more questions in different scenarios, comparing different possible diagnoses. The complete dataset consists of 18 question and evidence pairs. These evidence-question pairs are similar to the ones in Fig. 4.

In all cases, the explanation was in agreement with the BN, it used appropriate language and communicated probabilistic uncertainty in a straightforward manner. The output was 8 times in a narrative form and employed a lot of listing of significant variables, although the explanation elaborated on how each of them contributed to the results. As the Explainer does not rely on deeper semantic knowledge, the lack of semantics we found made it harder to interpret the results of the BN. Furthermore, using only the Explainer it cannot be guaranteed that the LLM will not hallucinate, as it has been seen to come up with domain information one time that cannot be directly linked to the BN explanation. The results of this evaluation can be seen in Table I.

TABLE I. RESULTS OF THE EXPLAINER EVALUATION

| | |
|---|---|
| **Agreement** | 100% |
| **Goal achieved** | 100% |
| **Appropriate language** | 100% |
| **Narrative** | 66% |
| **Uncertainty communicated** | 100% |

### *B. Retriever*

The Retriever subcomponent is responsible for semantic enhancement of explanations. It is essentially a RAG solution, so it is fitting to apply RAG evaluation methods to it.

The evaluation was carried out with the help of a popular Python library called RAGAS [15] (RAG Assessment) developed specifically for testing RAG applications. Utilizing question-answer pairs to compare the answers of the system to RAGAS can compute the following metrics that are relevant to the current use case:

- **Faithfulness:** How faithful the answer is to the retrieved context.
- **Semantic Similarity:** Cosine similarity between the embedding of the reference and generated answers.
- **Factual Correctness:** How many of the statements present in the generated answer were in the reference.
- **Response Relevancy:** How well does the response address the query.
- **Context Recall:** How much of the statements present in the reference answer can be attributed to the retrieved context.

For the evaluation dataset, 159 question-answer pairs were generated from the documents in the annotations with GPT-4o-mini [14], a model well suited for these types of tasks. An example from this dataset is in Table II.

TABLE II. EXCERPT FROM THE RETRIEVER EVALUATION DATASET

| Question | Reference answer |
|---|---|
| „ What is pseudodementia?" | "Pseudodementia is a clinical condition characterized by cognitive symptoms that closely resemble those of dementia [...]" |

The results of the evaluation with RAGAS can be seen in Table III compared to how well GPT-4o-mini and Llama 3.1 70b [16] did on the same questions without context. In Factual correctness and Semantic similarity, the Retriever outperforms both. This means that Retriever more consistently provides trustworthy information than the LLMs.

TABLE III. RESULTS OF THE RETRIEVER EVALUATION COMPARED TO GPT-4O-MINI AND LLAMA 3.1 70B

| | Retriever | GPT-4o-mini | Llama 3.1 70b |
|---|---|---|---|
| **Answer relevancy** | 91% | **94%** | 92% |
| **Factual correctness** | **58%** | 41% | 38% |
| **Semantic similarity** | **95%** | 93% | 91% |
| **Faithfulness** | 91% | Nan | Nan |
| **Context recall** | 92% | Nan | Nan |

### C. *Complete system*

The complete system's performance was assessed in the same manner as the Explainer's as the emphasis is on how well the results are communicated. The dataset consisted of 10 complex queries based on the previous evaluation datasets.

By combining both components, the system scored perfectly on the explanation evaluation checklist. This means that when the Retriever semantically enriches explanations, those become narrative in a semantic sense and not just a pure listing of causes and influences, and we found these to be much easier to comprehend. However, the same limitations noted in the Explainer evaluation in section IV.A apply here too. Table IV shows the results of this evaluation.

TABLE IV. RESULTS OF THE COMPLETE EVALUATION

| | |
|---|---|
| **Agreement** | 100% |
| **Goal achieved** | 100% |
| **Appropriate language** | 100% |
| **Narrative** | 100% |
| **Uncertainty communicated** | 100% |

## V. CONCLUSIONS

This paper presented a novel method for creating interactive semantic explanations for Bayesian Networks. The method defines an Explainer and a Retriever component for generating explanations of BNs and enhancing them with semantic knowledge respectively. These components rely on an ABN [6] and an LLM. The Retriever component updates the ABN-IR [10] method to a modern RAG-based approach. The Explainer realizes ABN-based explanations. Previous BN explanation methods were too rigid and offered very limited interactivity and underutilized information about the model's domain.

Furthermore, based on this method, a prototype was implemented in the domain of dementia diagnosis. The system was able to communicate the knowledge of the underlying BN in a manner that is easy to understand.

Future research could focus on finding reliable ways of evaluating natural language explanations and further refining context-creation techniques for both the Explainer and the Retriever.

The method could be extended to facilitate natural language input of evidence and be used for generating various medical documents based on a BN's diagnostic results and the rich semantics of the annotations.

The Explainer-Retriever architecture could be adapted to other white-box models, thus enabling LLM-based interactive semantic explanations for various other modalities of data, such as visual and time series. Furthermore, this could allow the generation of text corpora and synthetic data for the pre-training and fine-tuning of LLMs and multimodal LLMs. This could enable LLMs to incorporate the capabilities and knowledge inside these models.

## REFERENCES


[1] S. McLachlan, K. Dube, G. A. Hitman, N. E. Fenton, and E. Kyrimi, "Bayesian networks in healthcare: Distribution by medical condition," *Artif. Intell. Med.*, vol. 107, p. 101912, Jul. 2020, doi: 10.1016/j.artmed.2020.101912.

[2] C. Hennessy, A. Bugarín, and E. Reiter, "Explaining Bayesian Networks in Natural Language: State of the Art and Challenges," in *2nd Workshop on Interactive Natural Language Technology for Explainable Artificial Intelligence*, J. M. Alonso and A. Catala, Eds., Dublin, Ireland: Association for Computational Linguistics, 2020, pp. 28–33. Accessed: Sep. 19, 2024. [Online]. Available: https://aclanthology.org/2020.nl4xai-1.7

[3] A. Barredo Arrieta *et al.*, "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Inf. Fusion*, vol. 58, pp. 82–115, Jun. 2020, doi: 10.1016/j.inffus.2019.12.012.

[4] E. Kyrimi, S. Mossadegh, N. Tai, and W. Marsh, "An incremental explanation of inference in Bayesian networks for increasing model trustworthiness and supporting clinical decision making," *Artif. Intell. Med.*, vol. 103, p. 101812, Mar. 2020, doi: 10.1016/j.artmed.2020.101812.

[5] C. Lacave and F. Dez, "A Review of Explanation Methods for Bayesian Networks," *Knowl. Eng. Rev.*, vol. 17, May 2001, doi: 10.1017/S026988890200019X.

[6] P. Antal, B. De Moor, T. Mészáros, and T. Dobrowiecki, "Annotated Bayesian Networks: A Tool to Integrate Textual and Probabilistic Medical Knowledge.," presented at the Proceedings of the IEEE Symposium on Computer-Based Medical Systems, Jan. 2001, pp. 177–182. doi: 10.1109/CBMS.2001.941717.

[7] Y. Gao *et al.*, "Retrieval-Augmented Generation for Large Language Models: A Survey," Mar. 27, 2024, *arXiv*: arXiv:2312.10997. doi: 10.48550/arXiv.2312.10997.

[8] P. Lewis *et al.*, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," Apr. 12, 2021, *arXiv*: arXiv:2005.11401. Accessed: Apr. 23, 2024. [Online]. Available: http://arxiv.org/abs/2005.11401

[9] R. Butz, R. Schulz, A. Hommersom, and M. van Eekelen, "Investigating the understandability of XAI methods for enhanced user experience: When Bayesian network users became detectives," *Artif. Intell. Med.*, vol. 134, p. 102438, Dec. 2022, doi: 10.1016/j.artmed.2022.102438.

[10] P. Antal, B. De Moor, D. Timmerman, T. Meszaros, and T. Dobrowiecki, "Domain knowledge based information retrieval language: an application of annotated Bayesian networks in ovarian cancer domain," in *Proceedings of 15th IEEE Symposium on Computer-Based Medical Systems (CBMS 2002)*, Jun. 2002, pp. 213–218. doi: 10.1109/CBMS.2002.1011379.

[11] E. Reiter, "Natural Language Generation Challenges for Explainable AI," in *Proceedings of the 1st Workshop on Interactive Natural Language Technology for Explainable Artificial Intelligence (NL4XAI 2019)*, J. M. Alonso and A. Catala, Eds., Association for Computational Linguistics, 2019, pp. 3–7. doi: 10.18653/v1/W19-8402.

[12] "Welcome to Apache Solr." Accessed: May 26, 2024. [Online]. Available: https://solr.apache.org/index.html

[13] B. Peng *et al.*, "Graph Retrieval-Augmented Generation: A Survey," Sep. 10, 2024, *arXiv*: arXiv:2408.08921. doi: 10.48550/arXiv.2408.08921.

[14] "GPT-4o mini: advancing cost-efficient intelligence." Accessed: Aug. 31, 2024. [Online]. Available: https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/

[15] "🚀 Get Started - Ragas." Accessed: Oct. 31, 2024. [Online]. Available: https://docs.ragas.io/en/stable/getstarted/index.html

[16] "Introducing Llama 3.1: Our most capable models to date," Meta AI. Accessed: Nov. 03, 2024. [Online]. Available: https://ai.meta.com/blog/meta-llama-3-1/

# On Expressing Blockchain Financial Operations in BPMN Models

Damaris Jepkurui Kangogo, Balázs Ádám Toldi, Imre Kocsis
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {dkangogo, balazs.toldi}@edu.bme.hu, ikocsis@vik.bme.hu

***Abstract*—Business Process Model and Notation (BPMN) is a widely adopted standard for modeling business processes across various domains. BPMN, particularly starting from version 2.0, provides extensibility through custom elements. Existing research demonstrates that domain-specific extensions have been developed to address gaps in representing specialized processes, e.g., in IT security, healthcare workflows, and manufacturing. However, in the context of financial workflows, there is a lack of systematic approaches to adequately represent payment processes. BPMN modeling often simplifies payment steps and their side effects as generic activities or simply uses boundary error events for failed payment transactions. With the increasing adoption of blockchain for workflow execution, traditional ways of modeling payment tasks in BPMN prove inadequate for capturing the complexity of financial primitives and the joint atomicity of payment and workflow steps. To address these limitations, we propose to leverage the extensibility offered by BPMN to model complex payment steps. We also formulate new safeness and soundness concepts and begin to lay out a state-based joint modeling approach for process and associated distributed ledger.**

***Index Terms*—BPMN, Blockchain, Financial Workflows, Payment Process Modeling, Domain-Specific Modeling.**

## I. Introduction

Business Process Model and Notation (BPMN) [1] is a standard for modeling business processes, widely adopted by industry and academia. BPMN was initially proposed in 2004 by the Business Process Modeling Initiative, recognized as a standard in 2013, and has since been maintained by the Object Management Group. It presents an intuitive graphical representation that enhances the comprehension of business processes for all involved stakeholders. Additionally, the availability of a large pool of supporting tools on the market has further facilitated BPMN's wide adoption [2].

Since BPMN 2.0[1], the modeling language provides extensibility through custom elements, enabling domain-specific adaptations. This has led to numerous extensions in areas requiring specialized representations [3]. However, financial workflows, particularly those involving payment transactions, have not received the same attention. Payment processes are frequently modeled in BPMN using simplified constructs, such as generic tasks or subprocesses, and boundary error events for modeling failed payment steps. While these elements suffice for basic scenarios, they fail to capture domain-specific details of financial transactions.

These limitations become particularly significant given the increased adoption of blockchain technology in enabling collaborative process execution. This is because transactions often involve complex financial primitives such as escrows, smart contracts, and the joint atomicity between payment and workflow steps. For instance, in a scenario where payment is made through an escrow and funds are deposited in a secure intermediary account with clear dependencies between transaction success and workflow outcomes, if payment is successful, the funds are distributed appropriately. But in the event of failure, the escrow is freed up and the funds are returned to the original sender. Expressing these relationships visually using traditional BPMN constructs is a challenge, especially since the workflows must provide joint guarantees on performing the payment and related business tasks.

## II. Background

BPMN is widely used across industries such as logistics, finance, banking, manufacturing, and healthcare to document processes with well-defined sequences of recurring activities. The standard includes four diagram (model) types: *process*, *collaboration*, *choreography*, and *conversation* diagrams, and the core set of BPMN elements can be categorized into four main groups: *flow objects*, *connecting objects*, *swimlanes*, and *artifacts*. An exhaustive description of these elements can be found in the official specification of the standard [4].

Currently, BPMN presents more than 200 distinct graphical notation elements [2]. Furthermore, BPMN 2.0.2 includes an *"extension by addition"* mechanism that enables users to incorporate new elements into BPMN models. This allows domain-specific elements to be attached to the core elements of the language by addition, facilitating the reuse of the core functionality while taking advantage of its benefits, such as standardization and tool compatibility [3]. Additionally, developing BPMN extensions is generally less costly as compared to developing an entirely new domain-specific modeling language from scratch.

### A. Decentralized Orchestration

In the past few years, several tools have been developed to assist in process modeling and execution. Currently, more

[1]BPMN 2.0.2 is the latest version released in December 2013 https://www.omg.org/spec/BPMN/2.0.2 formally published by ISO as the 2013 edition standard: ISO/IEC 19510

DOI https://doi.org/10.3311/MINISY2025-013

than 70 tools support BPMN [2]. The more advanced ones often use a BPMN model to track and orchestrate activities, collect activity-related data, and make decisions on the further progression of the processes.

When processes extend beyond a single entity, centralized orchestration poses challenges as it necessitates a trusted third party to oversee process coordination. The emergence of blockchain and distributed ledger technologies has offered promising opportunities to decentralize various elements of cross-organizational collaborations. Smart contracts deployed on the blockchain make it possible to monitor the state of collaboration and enforce the obligations and permissions related to the activities of involved parties without the need for a centralized trusted third party. Thus, blockchain ensures an immutable and tamper-resistant logging of transactions.

The role of blockchain in enhancing workflows has been increasingly explored in literature [5]. Tools exist for translating BPMN models to smart contracts using Model-Driven Engineering (MDE). Waber et al. [6] proposed the first approach to integrating blockchain into the choreography of processes. Their solution involved generating a Solidity smart contract from a BPMN choreography to monitor or coordinate business processes. Ever since then, several other approaches have been proposed; notable ones are Caterpillar [7] – an open-source BPMN-to-Solidity compiler; Lorikeet [8] – which extends the BPMN 2.0 specification with support for asset registries; and Chorchain [9] and its successors Multi-Chain [10] and FlexChain [11].

In the above approaches, the orchestrator smart contract logic is generated from the BPMN model automatically, with the model serving as a specification. These smart contracts ensure the automatic enforcement and execution of rules and conditions agreed upon by all parties, providing trust in a trust-limited environment.

### B. Modelling Payment in BPMN

An increasing number of BPMN extensions have been proposed and implemented in the literature [3] to represent domain-specific business processes in various domains such as healthcare [12], Internet of Things (IoT) [13], and manufacturing [14]. Limited studies exist that propose an extension of the BPMN modeling language to represent payment steps in financial workflows. For instance, Waber et al. [6] extend the BPMN standard to include payments by introducing a choreography monitor, a smart contract that can also act as an escrow for conditional payment at designated points. Panduwinata and Yugopuspito [15] utilize Hyperledger Composer to model business processes and implement smart contracts that manage parking reservations. For payments, the approach introduced a smart contract task for automated execution of payment when specific conditions are met. The study [16] presents two conceptual models using BPMN: one depicting the existing South African real estate transaction process and another illustrating a blockchain-based approach where smart contracts manage payment transfers and a distributed ledger to keep transaction records.

## III. Motivation and problem statement

The established approach for expressing payment and other financial side effects on BPMN diagrams is to use (uninterpreted) payment tasks and transfer-representing message flows. However, we argue that for blockchain-orchestrated BPMN execution as well as BPMN execution meaningfully relying on financial side effects implemented by decentralized functions (blockchain primitives and smart contracts), the established modeling approaches are inadequate.

- For the financial side effects which are more complicated than ”sending money” but very much utilized on blockchains, standard BPMN seems to fail in its role of specification for documentation and human understanding; it is hard to decipher the ”financial intentions” of models.
- Uninterpreted and ad-hoc modeling of financial side effects hinders reasoning about and enforcing novel model and process execution properties that directly flow from the non-revocable nature of distributed ledger transactions.

### A. Atomic Swap as an Example

As an example, consider the model in Figure 1, which represents the ”Energy support use case” – a central bank digital currency (CBDC) prototype developed by BME and MNB, the Central Bank of Hungary, for the Bank of International Settlements (BIS) Rosalind Phase 2 competition[2]. A distributed ledger system maintains pseudonymized CBDC accounts (with optional sub-accounts) for citizens and businesses, managed by private-sector Payment Interface Providers (PIPs). While the transaction set is limited, it includes two-party and three-party locks, a ”drawdown” mechanism (permissioned outbound transfer allowance), and ”hash time-locked contracts” (HTLCs) in addition to CBDC transfers.

In the example, the support office, a governmental body, applies fine-grained policies in real time for citizen energy bill support decisions, incorporating energy-saving incentives as well as in-need considerations to support vulnerable groups. Citizens use a mobile app to record meter readings with the energy supplier, who issues them energy bills. The citizen then turns to the support office to acquire indicative support eligibility for the bill.

Using a double HTLC mechanism, unsupported and supported portions of a bill are funneled into a specialized ”proxy account”. This proxy account restricts outgoing transfers to the energy supplier as a PIP service. The two HTLCs enable an atomic swap: either the citizen’s payment is settled with the support office and the full amount transferred to the proxy account, or funds remain with both parties if the transaction fails.

While alternative ”modeling styles” can be used (and have been explored by us), it seems hard to argue that the financial intentions of the collaboration can be modeled ”cleanly” in an easy-to-grasp way using standard tasks, messages, and events.

[2]https://www.bis.org/innovation_hub/projects/rosalind_phase2.pdf

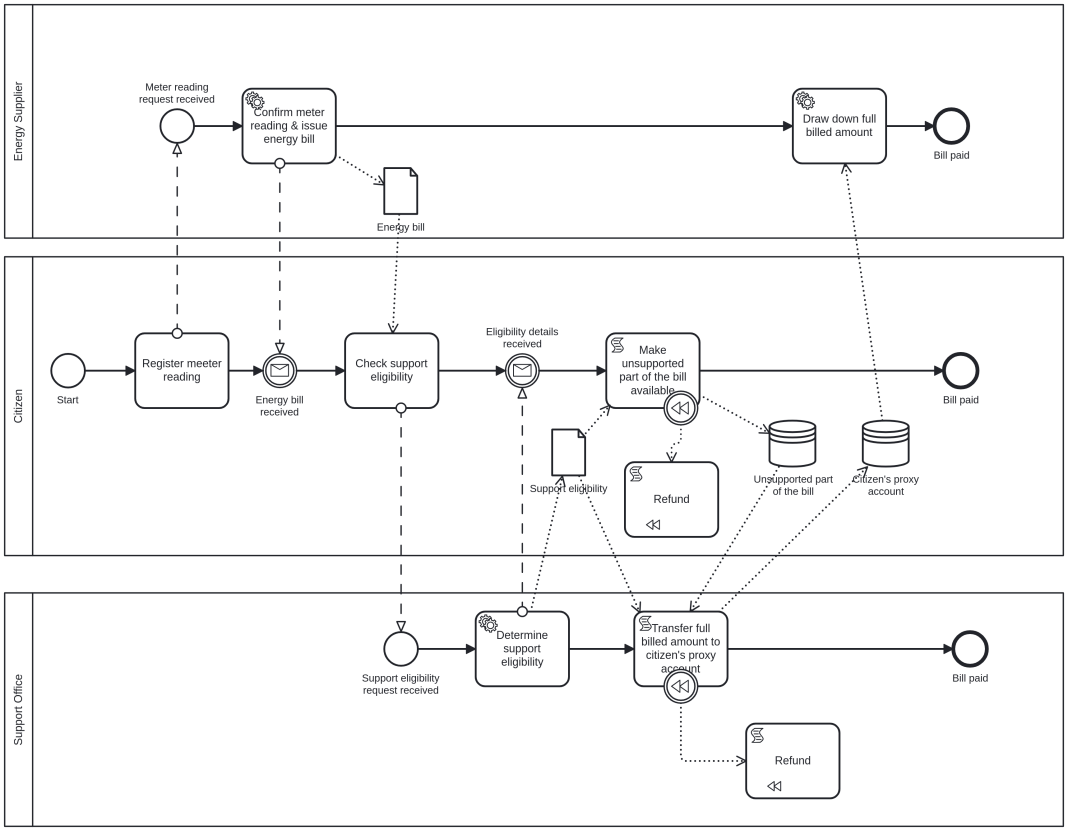



Fig. 1. The energy use case collaboration flow modeled with standard BPMN constructs

We are assembling a number of visual extensions for BPMN that communicate financial (joint) activity intent over distributed ledgers much better than the established modeling styles; due to limited space here, these will be elaborated in later publications. One (incrementally BPMN-extending) graphical ”style” is to introduce a ”blockchain-pool” (such as shown in Figure 2), where activities on the funds of participants (and outcomes) can be represented in dedicated lanes.

However, expressive modeling shines light on a deeper problem, too: i.e., how can we ensure that process execution and blockchain state *jointly* conform to our expectations? Even with far better graphical notation, it is not necessarily apparent what the ”financial end states” of the process can be or whether a ”financial deadlock” cannot happen due to the parties beginning the process without sufficient funding.

Verification of BPMN models is not a new notion. Various formal and tool-based approaches exist [17] exploring properties of BPMN models such as soundness, reachability, liveness, completeness, and compliance.

### B. Extending notions of BPMN safety and liveness

Formal notions of *safeness* and *soundness* for BPMN collaborations under a standard token passing semantics already exist [18] – expressing bounds on token markings and the proper ”finishability” of collaborations. While there’s a large corpus of BPMN verification [17] which focuses on mostly ad-hoc, case-specific properties, there are examples where domain and execution environment concerns serve to extend the basic verification targets; e.g., [19] extends BPMN (graphically and textually) with the notion of performing actions in space and works out extended spatial properties.

Using a similar train of thought, we propose that it is an important challenge to be able to create models that, in addition to clearly communicating the financial intent, can be shown to fulfill key financial properties under a given blockchain-based execution regimen.

## IV. Proposed Approach

We propose introducing a novel *joint state space semantics* of process execution and attached blockchain state evolution. That is, to interpret process state as a $S_P = (M, BC, T)$ tuple, where $M$ is a BPMN state marking, $BC$ is the blockchain state (e.g., in an Ethereum-style system, accounts, contracts, and balances) and $T$ is the set of outstanding blockchain transactions (”transaction pool”). The transition function $\delta$ then acts on this state notion:

$$\delta : (S_P \times \{SmartContractCall \cup ProcessStep\}) \rightarrow S_P$$

That is, the joint state evolves either by performing the blockchain calls (abstracting away consensus for now) or by

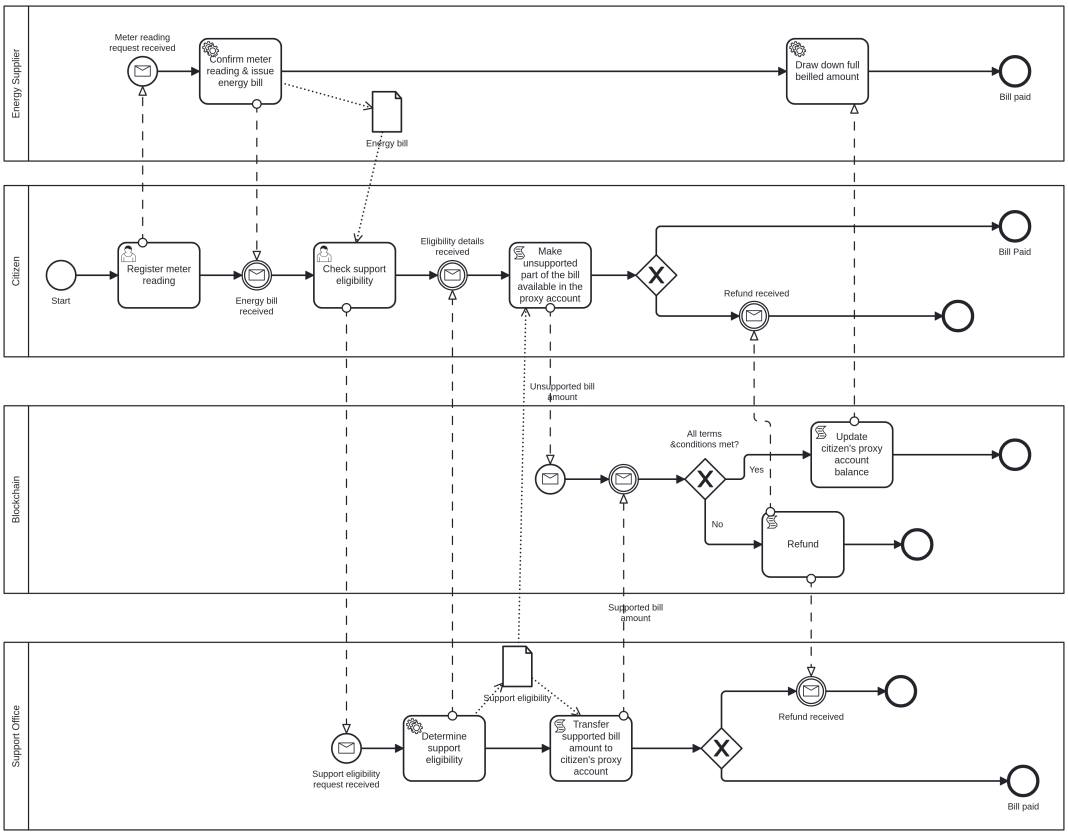



Fig. 2. To-be energy use case collaboration diagram

stepping the associated process. The benefits of this approach are the following:

- Both BPMN and distributed ledgers have well-developed state-based modeling approaches, with the possibility of symbolic execution modeling, making practical realizations for the purposes of verification, validation, and runtime property enforcement feasible at the model scales, which are typical for BPMN.
- It enables the consistent handling of a wide gamut of operational scenarios, from an unorchestrated process simply using a blockchain as a payment rail to full smart contract-based orchestration with contract-integrated financial activities.
- It enables the joint checking of invariants, path properties, and temporal properties, which is important in light of the properties identified earlier.

This simple formalism facilitates formulating *financial* notions of process safeness, soundness, and liveness.

- **Financial safeness**: similarly to the definition of process safeness based on token flow, financial safeness should cover aspects such as no double spending of tokens/assets, no unaccounted assets in transit, and no overlapping financial claims.
- **Financial soundness**: as for a process soundness expresses proper completion and no dead tokens, financial soundness should encompass that all obligations are settled, there are no ”stranded” assets on completion, value is conserved, and there are no unresolved debts.
- **Financial liveness**: no financial deadlocks (funds prevent progress), every obligation has a settlement path, and there is no infinite waiting for financial conditions.
- **Financial consistency**: balance sheet consistency across parties, deterministic financial outcomes for all execution paths.
- **Temporal-financial properties**: time-bound settlement guarantees, deadline-linked financial state validity.

## V. Conclusion

In this paper, we have identified the need to introduce a dedicated visual formalism for expressing blockchain financial operations in BPMN models. We formulated novel notions of safeness and soundness and proposed the joint state-based modeling of process and blockchain state as an integrative approach for V&V and runtime assurance planning.

## References

[1] Object Management Group, “About the Business Process Model and Notation Specification Version 2.0.2 — omg.org.” https://www.omg.org/spec/BPMN/2.0.2, 2014. [Accessed 27-11-2024].

[2] I. Compagnucci, F. Corradini, F. Fornari, and B. Re, “A Study on the Usage of the BPMN Notation for Designing Process Collaboration, Choreography, and Conversation Models,” *Business & Information Systems Engineering*, vol. 66, pp. 1–24, 06 2024.

[3] K. Zarour, D. Benmerzoug, N. Guermouche, and K. Drira, "A systematic literature review on BPMN extensions," *Business Process Management Journal*, vol. 26, no. 6, pp. 1473–1503, 2020.
[4] OMG, "Business Process Model and Notation (BPMN) Version 2.0.2," 2013. [Online; accessed 2024-12-12].
[5] H. Taherdoost and M. Madanchian, "Blockchain and Business Process Management (BPM) Synergy: A Comparative Analysis of Modeling Approaches," *Information*, vol. 15, no. 1, p. 9, 2023.
[6] I. Weber, X. Xu, R. Riveret, G. Governatori, A. Ponomarev, and J. Mendling, "Untrusted business process monitoring and execution using blockchain," in *Business Process Management: 14th International Conference, BPM 2016, Rio de Janeiro, Brazil, September 18-22, 2016. Proceedings 14*, pp. 329–347, Springer, 2016.
[7] O. López-Pintado, L. García-Bañuelos, M. Dumas, I. Weber, and A. Ponomarev, "Caterpillar: A business process execution engine on the Ethereum blockchain," *Software: Practice and Experience*, vol. 49, pp. 1162 – 1193, 2018.
[8] Q. Lu, A. Binh Tran, I. Weber, H. O'Connor, P. Rimba, X. Xu, M. Staples, L. Zhu, and R. Jeffery, "Integrated model-driven engineering of blockchain applications for business processes and asset management," *Software: Practice and Experience*, vol. 51, no. 5, pp. 1059–1079, 2021.
[9] F. Corradini, A. Marcelletti, A. Morichetta, A. Polini, B. Re, F. Tiezzi, *et al.*, "ChorChain: A model-driven framework for choreography-based systems using blockchain," in *ITBPM@ BPM*, pp. 26–32, 2021.
[10] F. Corradini, A. Marcelletti, A. Morichetta, A. Polini, B. Re, E. Scala, and F. Tiezzi, "Model-driven engineering for multi-party business processes on multiple blockchains," *Blockchain: Research and Applications*, vol. 2, no. 3, p. 100018, 2021.
[11] F. Corradini, A. Marcelletti, A. Morichetta, A. Polini, B. Re, and F. Tiezzi, "Flexible execution of multi-party business processes on blockchain," in *Proceedings of the 5th International Workshop on Emerging Trends in Software Engineering for Blockchain*, pp. 25–32, 2022.
[12] M. Szelagowski, P. Biernacki, J. Berniak-Woźny, and C. R. Lipinski, "Proposal of BPMN extension with a view to effective modeling of clinical pathways," *Business Process Management Journal*, vol. 28, no. 5/6, pp. 1364–1390, 2022.
[13] Y. Kirikkayis, F. Gallik, M. Winter, and M. Reichert, "BPMNE4IoT: a framework for modeling, executing and monitoring iot-driven processes," *Future Internet*, vol. 15, no. 3, p. 90, 2023.
[14] V. Ribeiro, J. Barata, and P. Rupino da Cunha, "A BPMN Extension to Model Inter-Organizational Processes in Industry 4.0," 2021.
[15] F. Panduwinata and P. Yugopuspito, "BPMN approach in blockchain with hyperledger composer and smart contract: Reservation-based parking system," in *2019 5th international conference on new media studies (CONMEDIA)*, pp. 89–93, IEEE, 2019.
[16] J. L. Tilbury, E. de la Rey, and K. van der Schyff, "Business Process Models of Blockchain and South African Real Estate Transactions," in *2019 International Conference on Advances in Big Data, Computing and Data Communication Systems (icABCD)*, pp. 1–7, 2019.
[17] T. Lopes and S. Guerreiro, "Assessing business process models: a literature review on techniques for BPMN testing and formal verification," *Business Process Management Journal*, vol. 29, no. 8, pp. 133–162, 2023.
[18] F. Corradini, C. Muzi, B. Re, and F. Tiezzi, "A Classification of BPMN Collaborations based on Safeness and Soundness Notions," in Proceedings Combined 25th International Workshop on *Expressiveness in Concurrency* and 15th Workshop on *Structural Operational Semantics,* Beijing, China, September 3, 2018 (J. A. Pérez and S. Tini, eds.), vol. 276 of *Electronic Proceedings in Theoretical Computer Science*, pp. 37–52, Open Publishing Association, 2018.
[19] R. Saddem-Yagoubi, P. Poizat, and S. Houhou, "Business Processes Meet Spatial Concerns: The sBPMN Verification Framework," in *Formal Methods* (M. Huisman, C. Păsăreanu, and N. Zhan, eds.), (Cham), pp. 218–234, Springer International Publishing, 2021.

# From Language to Causality: Extracting Causal Relations from Large Language Models

Márk Marosi, Kristóf Váradi, Péter Antal
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {marosi, antal}@mit.bme.hu, kristofvaradi@edu.bme.hu

***Abstract*—This research introduces a novel framework for constructing causal networks by leveraging the causal reasoning abilities of multiple Large Language Models (LLMs). We instruct LLMs to extract explicit causal links from their internal knowledge representations regarding specific topics. We explore methods for consolidating these graphs, addressing conflicts, and determining the strength and directionality of causal links. Evaluated across various domains using the Qwen 2.5 model family (0.5B to 14B parameters), the framework demonstrates the ability of language models to generate meaningful causal networks from complex queries. Our findings suggest that fusing causal knowledge from multiple LLMs significantly enhances causal discovery from natural language, though practical application benefits from human oversight and domain expertise to ensure accuracy and reliability. We also highlight the potential of integrating probabilistic approaches to quantify uncertainty within the extracted causal relationships.**

***Index Terms*—Large Language Models, Natural Language Processing, Bayesian Networks, Causal Discovery, Probabilistic Graphical Models**

## I. Introduction

Causal Bayesian networks, represented as directed acyclic graphs (DAGs), have become indispensable tools for modeling causal relationships across diverse scientific disciplines, including medicine, engineering, and social sciences [12]. Traditionally, constructing these models relies on expert elicitation, statistical analysis of observational data, or a combination thereof [4]. However, the emergence of Large Language Models (LLMs) offers a compelling new paradigm: extracting causal knowledge directly from their vast latent representations. This approach shares common goals with foundational work on automated knowledge discovery from scientific literature [13, 14], and extends text-mining and natural language processing methods that integrate textual information with other data modalities for knowledge synthesis [9, 5, 8, 1, 2]. While our approach similarly aims for extracting causal relations from scientific literature used in LLMs training, it differs fundamentally in that we are not limited to relations that are explicitly mentioned in the text but leverage the LLM's ability to generate plausible causal relations not seen in the data. This hypothesized capacity of LLMs' latent representations correspond to the causal level of their semantic compositionality.

We introduce a framework that leverages multiple LLMs to infer causal relations and complete causal diagrams from their latent representations, as expressed through their textual outputs. Our methodology, visualized in Figure 1, comprises three key stages. First, we prompt multiple LLMs to generate causal subgraphs related to a specific topic, encouraging diverse perspectives and mitigating individual model biases. We guide the generation process toward tree-like structures initially to reduce the risk of cyclical dependencies. Second, we employ various post-processing techniques to refine and unify the extracted graphs. We develop strategies to reconcile conflicting causal assertions, leveraging the collective wisdom of multiple LLMs. Semantic similarity measures, based on embedding vectors, are used to merge semantically equivalent nodes, thereby reducing redundancy. We also implement graph consolidation procedures that further refine the network representation. Finally, we investigate methods for assessing the existential certainty and directionality of causal relationships using the aggregated knowledge of the LLMs.

Recent research has begun to explore the potential of LLMs for causal reasoning tasks [6, 16, 15, 10, 7, 3]. Our work distinguishes itself by focusing on the explicit construction and refinement of causal network structures by synthesizing knowledge from multiple LLMs. We present a comprehensive evaluation across multiple domains using the Qwen 2.5 model family (0.5B to 14B parameters) [17]. Our results confirm the emergence of causal compositionality in LLM latent representations and demonstrate the utility of this approach for automated causal knowledge discovery and highlight key challenges and future research directions at the intersection of LLMs, causal discovery, and causal inference. We emphasize the importance of human-in-the-loop validation and the advantage of incorporating probabilistic reasoning to manage uncertainty in the extracted causal relations.

DOI https://doi.org/10.3311/MINISY2025-014

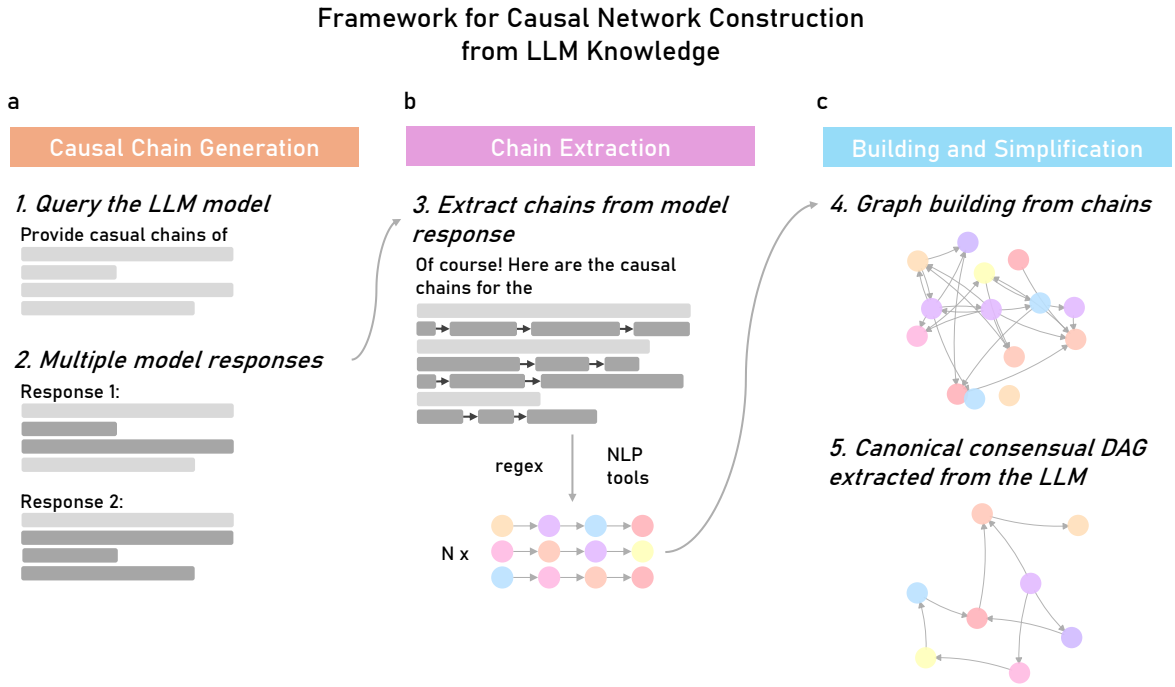



Fig. 1. Framework for constructing causal networks from large language models. The process comprises three stages: (**a**) **Causal Chain Generation**: The LLM is queried with a prompt to elicit multiple responses detailing causal relationships. (**b**) **Chain Extraction**: Natural language processing (NLP) tools and regular expressions (regex) are employed to systematically extract structured causal chains from the LLM's textual outputs. (**c**) **Building and Simplification**: The extracted chains are used to construct an initial causal graph. This graph is then simplified to obtain a canonical directed acyclic graph (DAG), representing the consensus causal structure derived from the LLM's knowledge. The framework leverages natural language understanding, graph-based representation, and consensus-building methods to generate a robust causal network.

## II. Background

A Bayesian Network (BN) is a probabilistic graphical model that represents a set of random variables $\{X_1, X_2, \ldots, X_n\}$ and their conditional dependencies via a directed acyclic graph. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a directed acyclic graph (DAG), where: $\mathcal{V} = \{X_1, X_2, \ldots, X_n\}$ is the set of nodes, each representing a random variable and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of directed edges, indicating direct dependencies between variables.

The joint probability distribution of the variables $\{X_1, X_2, \ldots, X_n\}$ in the network factorizes according to the structure of the graph $\mathcal{G}$:

$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid \text{Pa}(X_i)),$$

where $\text{Pa}(X_i)$ denotes the set of parent nodes of $X_i$ in the graph $\mathcal{G}$. Each node $X_i$ is associated with a conditional probability distribution (CPD) $P(X_i \mid \text{Pa}(X_i))$, which specifies the probabilities of $X_i$ given its parent nodes.

Bayesian Networks satisfy the Parental Markov property, which states that a node $X_i$ is conditionally independent of its non-descendants given its parent nodes:

$$X_i \perp\!\!\!\perp \text{NonDescendants}(X_i) \mid \text{Pa}(X_i).$$

Although Bayesian Networks are often used to model causal relationships, a directed edge from $u$ to $v$ does not necessarily imply that $X_v$ is causally dependent on $X_u$ [11]. A causal network is a Bayesian Network that explicitly encodes causal relationships (mechanisms) [12]. In a causal network, performing an intervention, denoted by $\text{do}(X = x)$, modifies the network by removing the edges into $X$ and setting $X$ to the value $x$. This allows for predicting the effects of external interventions using the induced truncated probability distributions.

## III. Methods

### A. Network Extraction

To address the challenge of cycle creation and incoherence in LLM-generated Bayesian Networks, we introduce a network extraction approach using directed subgraphs. We

prompt LLMs to produce causal "chains" (e.g., A $\rightarrow$ B $\rightarrow$ C) with few-shot examples guiding structured output like `Cause -> Effect -> ...`. While we prompt the LLMs to generate chains in a structured $A \rightarrow B \rightarrow C$ format, variations in responses can occur. To ensure robust extraction, we employ a regular expression, specifically '`([\w\s.-]+)\s*->\s*([\w\s.-]+)`', designed to capture cause-effect pairs even if minor deviations from the prompted format are present. This regex accommodates variable names containing spaces and special characters. This regular expression is crucial because, despite few-shot prompting, LLMs may not always adhere strictly to the desired output structure. The regex provides a reliable mechanism to extract the intended causal relationships from the LLM's free-form text. These pairs are then merged into an unweighted causal graph, where branching chains (e.g., A $\rightarrow$ B, A $\rightarrow$ C) are unified under a single root.

### B. Existential Uncertainty of Causal Relations

To quantify the degree to which an LLM supports a given causal relation $A \rightarrow B$, we introduce a **certainty score** derived from the model's token-level output probabilities. This score reflects the model's confidence in $B$ being a consequence of $A$. We treat the effect, B, as a sequence of tokens because we are measuring the model's confidence in generating B given A. By summing the log probabilities of each token in B, we estimate the overall likelihood of the model producing B in response to A. The cause, A, is part of the input prompt and is not treated as a sequence of tokens for this calculation.

Formally, given a causal relationship $A \rightarrow B$, we construct a context string:

$$\text{context} = \text{concatenate}(\text{prompt}, A, \text{" } \rightarrow \text{ "})$$

and then measure the log probability of generating the sequence of tokens in $B = (b_1, b_2, \ldots, b_m)$. The certainty score for the directed edge $A \rightarrow B$ is:

$$\text{Certainty}(A \rightarrow B) = \sum_{i=1}^{m} \log p\big(b_i \mid \text{context}, b_1, \ldots, b_{i-1}\big).$$

Here, $p(b_i \mid \ldots)$ is the conditional probability the LLM assigns to token $b_i$ given the prefix. The certainty score for a given edge A -¿ B is defined as the sum of the log probabilities of the tokens in B, given the context (prompt + A + " $\rightarrow$ ") and the preceding tokens of B.

In cases where multiple extractions from different chains or models yield the same pair $(A, B)$, we retain the maximum certainty score among duplicates. This reflects the strongest association between $A$ and $B$ as a likely cause-effect pair according to the LLM's learned representations.

### C. Conflict Resolution for Opposite Edges

Occasionally, our pipeline uncovers contradictory statements such as $A \rightarrow B$ and $B \rightarrow A$. To resolve these conflicts, we aggregate the certainty scores for each direction. If one direction has a substantially higher score, we select that direction. If the scores are comparable, we label the edge as *ambiguous* and flag it for human review. When merging nodes results in conflicting edges (e.g., $A \rightarrow B$' and $D \rightarrow B$' after merging $B$ and $C$), we aggregate the certainty scores as follows. For each direction, we sum the certainty scores of the original edges that contributed to the merged edge.

In the example [1] $A \rightarrow B \rightarrow D \rightarrow E$; [2] $A \rightarrow C \leftarrow D \rightarrow E$ ; $B + C = B'$, we would calculate:

- Certainty($A \rightarrow B'$) = Certainty($A \rightarrow B$)
- Certainty($B' \leftarrow D$) = Certainty($C \leftarrow D$)

We then compare these aggregated scores to determine the final edge direction, following the same procedure as for non-merged conflicting edges. If the certainty scores are not significantly different, we mark the edge $A \leftrightarrow B$ as ambiguous.

### D. Post-processing

After extracting cause–effect pairs, many nodes represent the same concept but use different wording. We reduce redundancy by merging nodes whose cosine similarity exceeds a threshold $\tau$. We obtain embedding vectors for node labels using the pre-trained Sentence-BERT model, specifically the `'all-mpnet-base-v2'` model, which has demonstrated strong performance in semantic similarity tasks. Throughout our experiments, we set $\tau = 0.8$.

We find that increasing $\tau$ above $0.9$ under-merges nodes (e.g., "global warming" and "climate change" might remain separate), while lowering $\tau$ below $0.7$ tends to over-merge semantically related but distinct concepts (e.g., "greenhouse gases" might be merged with "climate change"). We set the cosine similarity threshold $\tau$ to 0.8 based on empirical evaluation and common practice in textual similarity tasks. The method's sensitivity to $\tau$ is moderate; small variations around 0.8 do not drastically alter the results, but larger deviations can lead to significant under- or over-merging, impacting the final network structure.

## IV. Experiments

### A. Network Complexity

To evaluate LLMs' ability to identify causal relationships across diverse subjects, we develop a suite of 50 few-shot tasks spanning multiple domains with various topics and corresponding target variables. Our evaluation considers 5 advanced topics in each of *Healthcare, Environmental Science, Computer Science, Physics, Biology, Chemistry, Mathematics, Psychology, Engineering*, and *Astronomy*. In this work, we focus on evaluating the size and complexity

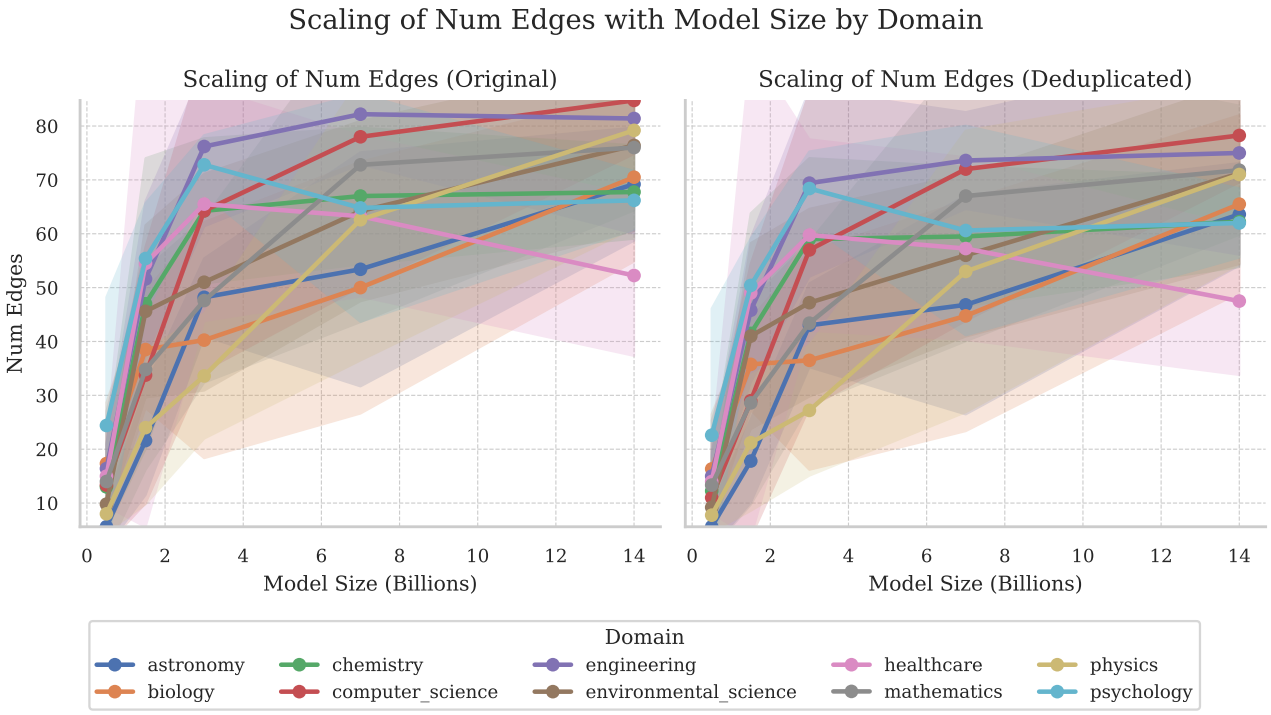



Fig. 2. Scaling of the number of edges in causal graphs with increasing model size across different domains. The figure presents two line plots illustrating the relationship between the size of the LLM used (in billions of parameters) and the number of edges in the resulting causal graphs. The left plot shows the scaling for the "original" graphs, while the right plot depicts the scaling for the "deduplicated" graphs. The shaded area around each line represents the variance across topics runs. A general trend of increasing edge count with larger model sizes is observed across most domains in both the original and deduplicated graphs, suggesting that larger models extract more causal relationships. The rate of increase, however, varies by domain and tends to plateau for larger model sizes, particularly in the deduplicated graphs. The deduplicated graphs (right plot) exhibit a more pronounced plateauing effect and generally lower edge counts than the original graphs (left plot), indicating the effectiveness of the deduplication process in removing redundant information.

of the generated causal graphs. The variation in graph complexity across these domains is illustrated in Figure 3, which compares the average number of edges in the original and deduplicated graphs for each domain.

### *B. Scaling Behavior*

We assess the benchmark tasks using the Qwen 2.5 family of language models, with parameter sizes ranging from 0.5B to 14B. These models are the state-of-the-art series of LLMs developed by Alibaba Cloud. Specifically, we evaluate models ranging from 0.5 billion to 14 billion parameters, allowing us to analyze the impact of model scale on causal knowledge extraction. These models are transformer-based and have been pre-trained on a massive corpus of text and code, equipping them with broad world knowledge and strong language understanding capabilities. Our findings reveal a positive correlation between the number of parameters and the size of the extracted causal graphs, indicating that larger LLMs are capable of identifying more complex causal relationships. This scaling trend is visually evident in Figure 2, which demonstrates the increase in the number of edges in the generated causal graphs as the model size grows across various scientific domains.

## V. Conclusion

This work introduces a framework to extract and refine causal knowledge from LLMs' latent representations in the form of Bayesian Networks. We demonstrate that by prompting models to generate causal chains and employing post-processing techniques to merge similar nodes and resolve conflicts, we can construct increasingly complex and accurate causal graphs. Our experiments using the Qwen 2.5 model family show that larger models construct more complex causal explanations.

However, LLMs can produce plausible yet factually inaccurate explanations and may exhibit biases derived from their training data. These biases are particularly pronounced in specialized domains. Addressing these challenges requires a multi-faceted approach. Future work will focus on analyzing and mitigating these biases, potentially by integrating domain-specific knowledge bases to verify generated explanations against established facts. We will also explore

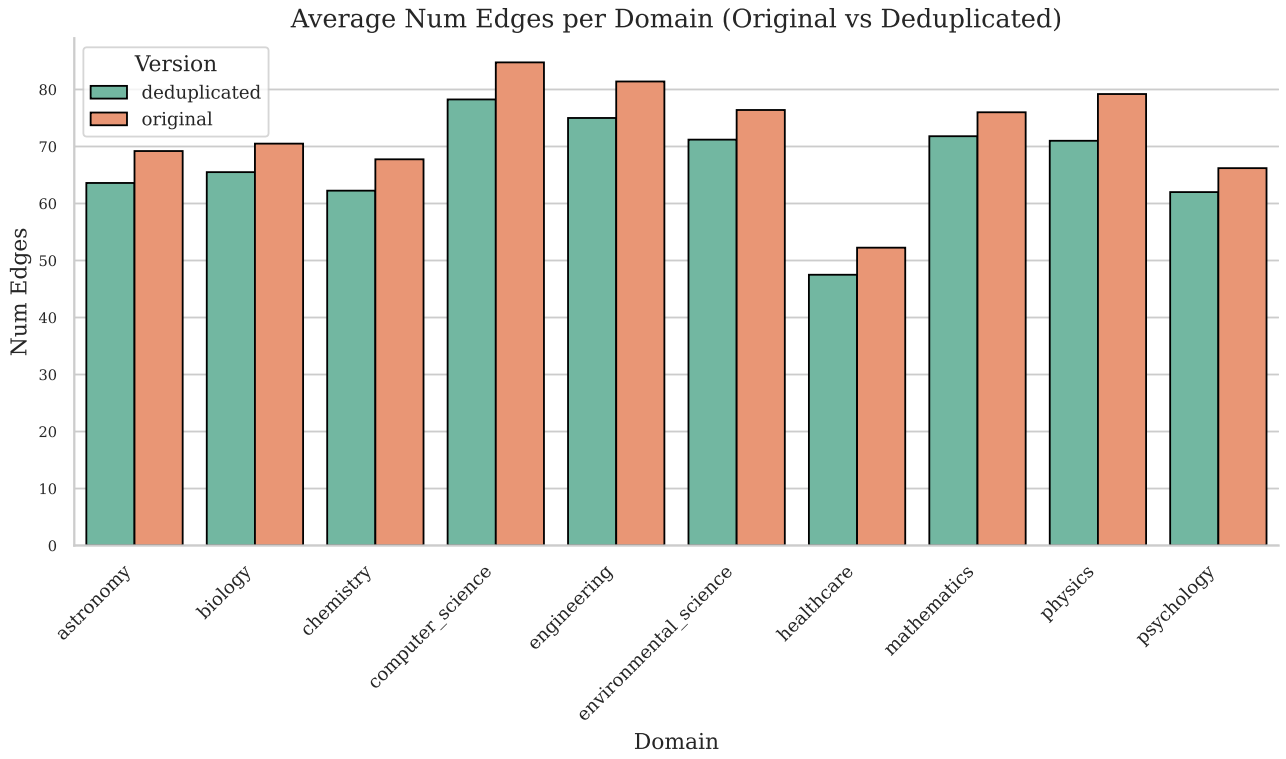



Fig. 3. Average number of edges per domain in causal graphs before and after deduplication. The bar chart compares the average number of edges in causal graphs generated from various scientific domains, contrasting the original graphs with their deduplicated versions. The "original" version (coral bars) represents the initial graph constructed from the extracted causal chains, while the "deduplicated" version (teal bars) reflects the graph after removing redundant edges.

more advanced natural language understanding techniques to handle complex statements involving multiple causes and conditional relationships. One limitation of the current framework is its inability to capture complex contextual relationships where multiple variables jointly influence an outcome under specific conditions. This is due to the chain-based extraction method, which primarily focuses on pairwise causal links. Future work will explore incorporating more sophisticated natural language understanding techniques capable of identifying and representing such context-dependent relationships, potentially by allowing for more complex graph structures beyond simple chains during the initial extraction phase. For example, we could query the LLMs specifically about how different factors interact to produce certain effects or prompt them to describe scenarios where a cause-effect relationship holds only under certain conditions.

## REFERENCES

[1] Peter Antal, Geert Fannes, Dirk Timmerman, Yves Moreau, and Bart De Moor. Using literature and data to learn Bayesian networks as clinical models of ovarian tumors. *Artificial Intelligence in medicine*, 30(3):257–281, 2004.

[2] Péter Antal and András Millinghofer. Learning causal bayesian networks from literature data. *Periodica Polytechnica Electrical Engineering (Archives)*, 50(3-4):201–221, 2006.

[3] Haoang Chi, He Li, Wenjing Yang, Feng Liu, Long Lan, Xiaoguang Ren, Tongliang Liu, and Bo Han. Unveiling Causal Reasoning in Large Language Models: Reality or Mirage? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

[4] David Heckerman, Dan Geiger, and David M Chickering. Learning bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20:197–243, 1995.

[5] Robert Hoffmann and Alfonso Valencia. Life cycles of successful genes. *Trends in genetics*, 19(2):79–81, 2003.

[6] Zhijing Jin, Yuen Chen, Felix Leeb, Luigi Gresele, Ojasv Kamal, LYU Zhiheng, Kevin Blin, Fernando Gonzalez Adauto, Max Kleiman-Weiner, Mrin-

maya Sachan, et al. Cladder: Assessing causal reasoning in language models. In *Thirty-seventh conference on neural information processing systems*, 2023.

[7] Emre Kıcıman, Robert Ness, Amit Sharma, and Chenhao Tan. Causal reasoning and large language models: Opening a new frontier for causality. *arXiv preprint arXiv:2305.00050*, 2023.

[8] Michael Krauthammer, Charles A Kaufmann, T Conrad Gilliam, and Andrey Rzhetsky. Molecular triangulation: bridging linkage and molecular-network information for identifying candidate genes in Alzheimer's disease. *Proceedings of the National Academy of Sciences*, 101(42):15148–15153, 2004.

[9] Michael Krauthammer, Pauline Kra, Ivan Iossifov, Shawn M Gomez, George Hripcsak, Vasileios Hatzivassiloglou, Carol Friedman, and Andrey Rzhetsky. Of truth and pathways: chasing bits of information through myriads of articles. In *ISMB*, pages 249–257, 2002.

[10] Jing Ma. Causal inference with large language model: A survey. *arXiv preprint arXiv:2409.09822*, 2024.

[11] Christopher Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 403–410, 1995.

[12] Judea Pearl. The seven tools of causal inference, with reflections on machine learning. *Communications of the ACM*, 62(3):54–60, 2019.

[13] Don R Swanson. Undiscovered public knowledge. *The Library Quarterly*, 56(2):103–118, 1986.

[14] Don R Swanson and Neil R Smalheiser. An interactive system for finding complementary literatures: a stimulus to scientific discovery. *Artificial intelligence*, 91(2):183–203, 1997.

[15] Guangya Wan, Yuqi Wu, Mengxuan Hu, Zhixuan Chu, and Sheng Li. Bridging causal discovery and large language models: A comprehensive survey of integrative approaches and future directions. *arXiv preprint arXiv:2402.11068*, 2024.

[16] Linying Yang, Vik Shirvaikar, Oscar Clivio, and Fabian Falck. A Critical Review of Causal Reasoning Benchmarks for Large Language Models. In *AAAI 2024 Workshop on"Are Large Language Models Simply Causal Parrots?"*, 2024.

[17] Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, and Shanghaoran Quan. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024.

# Framework for Automated Worst-Case Analysis

Pál Weisz, György Orosz
Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
Budapest, Hungary
Email: {weisz, orosz}@mit.bme.hu

***Abstract*—In this paper, a framework along with an electronic circuit database is presented. The framework can be used to test advanced worst-case circuit analysis algorithms. It has two main components: a software environment and a circuit database.**

**The software environment is capable of running the circuit simulations and returning the results. The framework also provides a software interface between the circuit simulator and the data processing algorithms by transferring and converting data between different software components. Finally, the simulation results can be post-processed at different levels in time and frequency domains.**

**The heart of the framework is the circuit database. The database is intended to be used as a benchmark and test environment for advanced analysis algorithms. The circuits are collected and categorized to support promising research directions. Such research fields are: automatic decomposition of electrical circuits into independent or loosely coupled subcircuits; extreme value search in the case of complex circuits; identification of characteristic behavior of circuits where the system has different operation modes or some parameter constellations can result in different behavior.**

***Index Terms*—worst-case analysis, test environment, circuit model, schematic, framework, simulation**

## I. Introduction

For safety-critical systems, incompatibility with the required specifications can lead to accidents and environmental damage, so such applications are designed and verified with great care in the industry. The design is based on methods that comply with various standards and specifications. During this stage, different test cases are defined for the verification process. With the help of testing and verification, the system's performance becomes measurable.

It is more cost-effective and less hazardous to first test a model of the circuit in a simulated environment on a computer. A physical prototype is only realized after all the required test cases are satisfied.

When assessing compliance with requirements, it is important to know how a circuit's operation depends on the electrical components' parameters, different environmental influences, and failure modes. The analysis of the effect of extreme conditions on the circuit operation is called worst-case analysis (WCA) [1] [2]. In this type of test, the subjects of interest are extreme phenomena and their causes, where the circuit operation deviates as much as possible from the specification. Since a complex circuit may contain several hundreds or thousands of analysis tasks, the efficiency of the WCA solution is crucial.

WCA is typically performed using circuit simulation software (e.g. LTspice, OrCAD, Tina). These software tools provide some basic and traditional analysis methods for performing WCA, e.g. Extreme Value Analysis (EVA), Monte Carlo Analysis, and sensitivity analysis [1].

However, these programs do not always provide a fast and efficient solution. Simulator programs numerically solve system equations to provide solutions for various systems. Sometimes, it may be possible to describe a problem in analytical form and solve it more quickly than with simulators using numerical algorithms [3].

Several methods have been proposed in recent decades to solve WCA tasks efficiently. Some examples are interval arithmetic [4] or affine arithmetic [5] [6]. If the system is described in analytical form, it also has the potential advantage of applying advanced extreme value search techniques [3] [7] [8] [9].

The above discussion shows that there is no standard method in the worst-case analysis process and that there are several open issues. Promising research fields could be, for example, the design of intelligent analysis algorithms that involve the characteristic properties of electronic circuits as a priori knowledge, providing more explainable and interpretable results that can support the root cause finding when a requirement fails. In order to support this research, a framework is suggested in this paper capable of running and evaluating electronic circuit simulations and contains benchmark and test circuits.

The paper is structured as follows. Section II provides an overview of the motivation behind the importance of the proposed framework. Section III presents the functional architecture and the software environment's main components, as well as the design considerations behind them. Section IV introduces some examples from the circuit database suitable for testing and further research. These highlighted examples also serve to illustrate typical test methods.

## II. Objective

The proposed framework was motivated by the needs of industrial practice. Experience has shown that moving the data generated during the analysis process between the different analysis and design environments is cumbersome. Not only is it necessary to manually move data from one computer program to another, but in many cases, it also takes extra time to convert the data formats. This is necessary because

DOI https://doi.org/10.3311/MINISY2025-015

several development and test software environments cannot work effectively in a cascade.

Fig. 1 shows the role of the framework in the analysis process along with the different related components: circuits, computational tools, and higher-level algorithms.

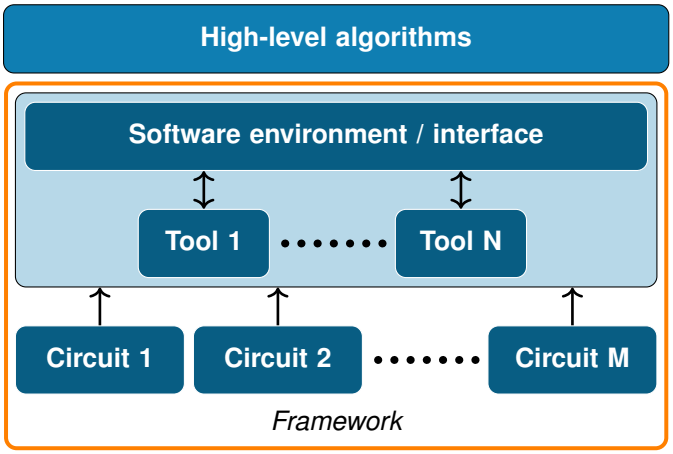



Fig. 1. Parts of the framework

Finding the least favorable behavior of a circuit is a challenging task, especially for more complex circuits, since several factors and parameters affect the circuit simultaneously. Many design and development steps precede the worst-case analysis. The process itself can be quite complex and time-consuming.

The proposed framework aims to optimize the entire worst-case analysis process and make it more efficient. On the other hand, it provides a consistent interface to output and input channels to support automation and greater flexibility in the different tools used for the analysis process.

## III. Software Environment

One of the fundamental elements of the framework is a software environment. The proposed software environment can produce the results of the circuit model evaluation. There are several options in order to do this.

In order not to be limited to the processing of simulation results, the software environment provides a uniform interface to the completely different methods that produce the results of the models.

### A. General Structure

The overall architecture of the proposed software environment is shown in Fig. 2.

The software is built architecturally in layers. The essential task of the bottom layer is to evaluate the circuit model by simulation or other computational methods. This can be done by calling simulator programs or various solver algorithms that provide results based on the circuit model.

In the next layer, the software environment provides an interface between the circuit model solver/simulator and the data processing system, which can transfer and convert data between each part. This interface allows the software to read

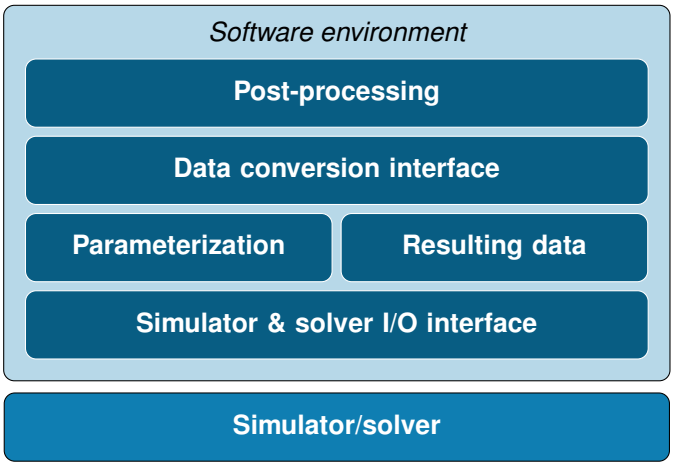



Fig. 2. Software Architecture.

the data and also to parameterize the model circuit in an automated way.

The results are produced in a format depending on the method used. The resulting data must, therefore, be converted into a format that the software can handle consistently.

Finally, the data obtained can be post-processed at different levels in time and frequency domains. The proposed software environment can be used to test various high-level analysis methods.

The framework uses LTspice as a circuit simulator and MATLAB for mathematical modeling and further computations. These choices are sufficiently versatile that they do not limit the overall usability of the framework.

### B. Functional Description

The low-level input-output interface has basic functionalities: modifying model parameters, starting the simulation, and retrieving results. In this particular case, the responsibility of the simulator I/O interface is to parameterize the circuit model and run the simulation by invoking LTspice commands.

The first important improvement presented in this paper is the choice of the way the parameters are set. As shown in Fig. 3, there are three ways of running the simulations for more than one parameter set:

- simulator program is started for every single parameter setting: general method but slow;
- simulator program is started for a batch of parameter set: more efficient, but requires parameter values in the actual batch in advance;
- analytical equations are generated from the schematic and parameters are substituted into equations: most efficient, but analytical form does not exist for each schematic.

When an iterative method of a worst-case analysis is being performed, there is a serious overhead in runtime to repeatedly restart the simulation. In order to mitigate this impact, the root causes need to be understood.

One reason for the runtime overhead is that the simulation software must be started (highlighted in bold font in Fig. 3)

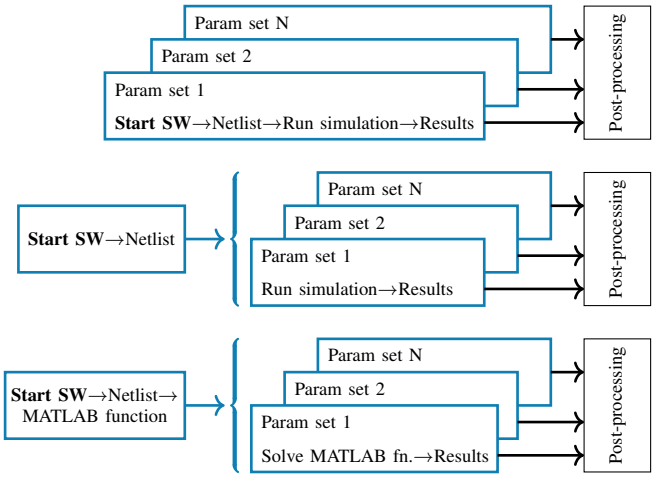



Fig. 3. Different approaches to computation methods

which produces a SPICE netlist. Generally, parameters are defined as constants in the circuit model's schematic diagram. The other option is to read the parameters from an external file. During simulation in LTspice, as a first step, the schematic diagram is transformed into a netlist. The simulation is being performed using this standard netlist file.

The simulation runtime can thus be reduced by omitting this transformation step and introducing the parameters only after the netlist has been generated (as shown in the second row of Fig. 3). From a less technical aspect, on the other hand, in some cases, it is possible to introduce multiple parameters at the same time to perform repeated analysis.

This means the simulation must be started less often to reduce the related overhead. However, suppose the parameters need to be changed frequently (e.g. to perform gradient-based extreme value finding, which the simulator cannot). In that case, it is advisable to manipulate the netlist in contrast with reading from an external file or change parameters in the schematic. While extra file operations are required in the first case, the netlist generation cannot be avoided in the latter case. If any option is chosen, an increase in runtime is expected.

In certain cases, it is possible to create an analytical representation of a circuit model [3]. If this option is available, excluding the simulation step from the process is possible.

The data conversion interface extracts the data from the simulation output files. It also selects their relevant records and converts them to the required format for higher-level use. The simulator calculates the results of all variables, but not all data is required. From the resulting data series, the ones that are relevant for data processing must be selected.

The top layer of the architecture implements the post-processing of the data. At this point, data in a format suitable for high-level analysis processes is available. Post-processing can be directly integrated with the worst-case analysis methods mentioned in Section I. At this point, the traditional mathematical apparatus can be applied.

## IV. Circuit Database

The purpose of the circuit database is to support promising research fields in worst-case circuit analysis. Even advanced methods apply standard analysis steps, simple statistical methods, or blind evaluation of system equations without considering the electronic circuits' properties. In order to develop intelligent analysis algorithms, several test cases, illustrative examples, and benchmark circuits are needed.

The analysis can be performed more efficiently if the characteristic properties of the circuit are taken into account, as in a human analysis.

One of the suggestions for an intelligent analysis is that it is important to recognize the characteristic behaviors of the circuits. Some illustrative examples are distinguishing between operation modes (e.g. saturation, normal mode), separation of different behaviors in the frequency domain (e.g. high-pass, low-pass band, or different slope), identifying phases of a time-domain signal (e.g. rise-time, ringing, steady-state), or checking disjoint parameter constellations which can result in different operation modes (e.g. some load capacitance can result in the oscillation of an amplifier).

Decomposing the system into independent or loosely coupled subsystems could also be advantageous because it reduces the complexity of solving problems. So far, decomposition is done mainly based on human decisions, but it could be prone to error or influenced by subjective decisions. Hence, this paper also presents examples of where decomposition algorithms can be tested.

Traditional worst-case analysis algorithms often consider the linear approximation of the error surface [1]. Computationally efficient algorithms generally use only analytical formulas and could provide too conservative extreme value [4]–[6]. Numerical methods are promising [10], but there are several choices, and choosing the most effective is not trivial. So, finding the extreme value in complex systems is not trivial. Hence, problems with nontrivial extreme value are important, and such examples are also enumerated in the proposed database [11].

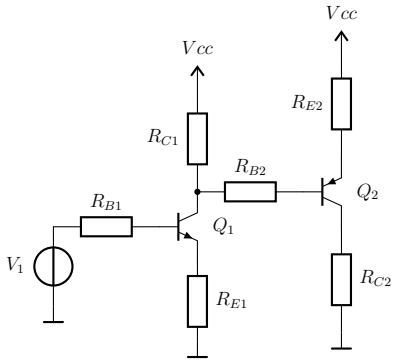
The electronic circuit database consists of different kinds of circuits based on real-life examples on which different analysis methods can be performed. Its schematic diagram defines a particular circuit model. A computational or simulation model of the circuit is used during the analysis.

In order to test different characteristic behaviors of a particular model, it is important to parameterize the circuits correctly to ensure that the desired properties are emphasized.

Some circuits from the database are listed below. An extended list of circuits can be found in [11]. A schematic diagram of the circuit and a representative simulation result for each example are presented. The simulation results are a qualitative representation of the circuit behavior under investigation, with a particular setting of the component parameters.

The first example is a switching circuit consisting of two bipolar junction transistors.

The schematic diagram and the simulation result are shown in Fig. 4.




| $V_1$ | $R_{C1}$ | $R_{B1}$ | $R_{E1}$ | $R_{C2}$ | $R_{B2}$ | $R_{E2}$ |
|---|---|---|---|---|---|---|
| 5 V | 2 kΩ | 10 kΩ | 470 Ω | 18 kΩ | 1 kΩ | 1.5 kΩ |

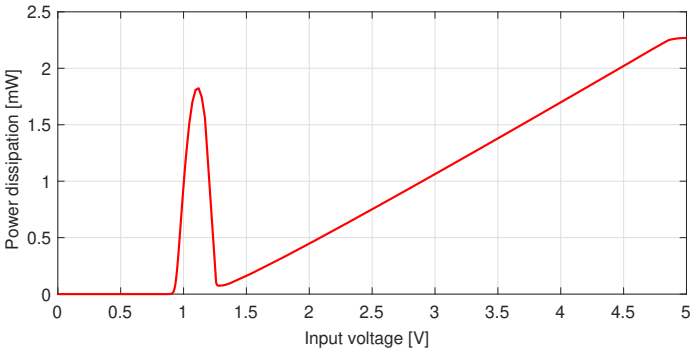



Fig. 4. Switching circuit using two transistors. Time-domain response to linearly increasing input voltage.

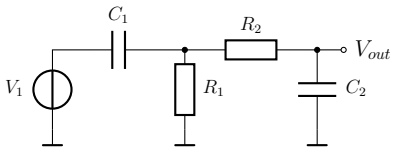
The simulation aims to highlight the particular behavior of the PNP transistor labeled $Q_2$. The quantity tested is the power dissipation of $Q_2$, depending on the linear sweep of the input voltage $V_1$ over time. According to the diagram, an initial peak in the power dissipation curve is experienced as a characteristic feature. The specialty of the circuit is that the dissipation shows a very nonlinear nature, and transistors have different operation modes during the input sweep. It is possible to test how the characteristic operation regions can be detected and how the worst-case values can be found on this circuit.

The next example represents a second-order band-pass filter shown in Fig. 5.

The resulting transfer function is displayed on a Bode plot. In this case, the high-pass and low-pass bands could be identified automatically, and at specific parameter settings, the high-pass and low-pass stages are highly independent; they do not influence each other considerably.

The following example demonstrates an amplitude-stabilized Wien-bridge oscillator circuit shown in Fig. 6.

The example also illustrates the framework's post-processing capability. The upper graphs show the output signal, with red circles indicating the zero crossing points determined by interpolation. The specialty of this circuit is that the operational amplifier's slew rate limit has effect only at particular parameter constellation so that it can be tested, e.g. how the distortion is influenced in certain parameter subspaces,




| $V_1$ | $C_1$ | $R_1$ | $R_2$ | $C_2$ |
|---|---|---|---|---|
| 1 V | 1 µF | 1 kΩ | 100 kΩ | 100 pF |

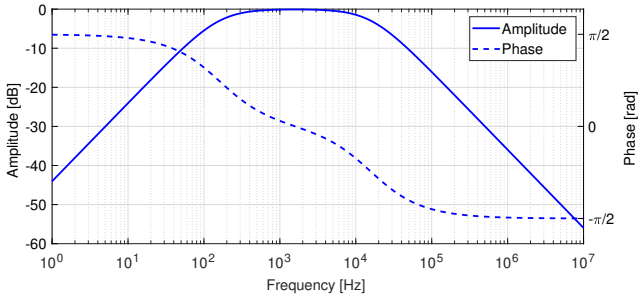
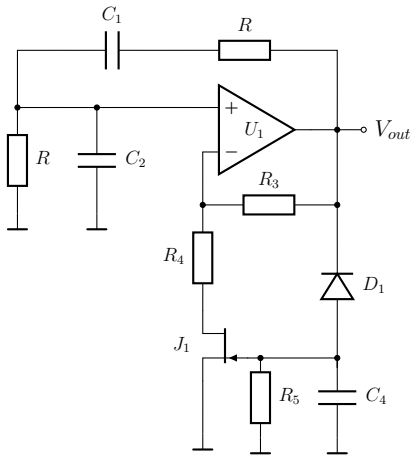



Fig. 5. Band pass filter and its transfer function




| $R$ | $C_1$ | $C_2$ | $R_3$ | $R_4$ | $J_1$ | $D_1$ | $R_5$ | $C_4$ | $U_1$ SR |
|---|---|---|---|---|---|---|---|---|---|
| 10 kΩ | 100 nF | 100 nF | 10 kΩ | 4.7 kΩ | 2N3819 | 1N4148 | 10 kΩ | 100 nF | 0.26 V/µs |

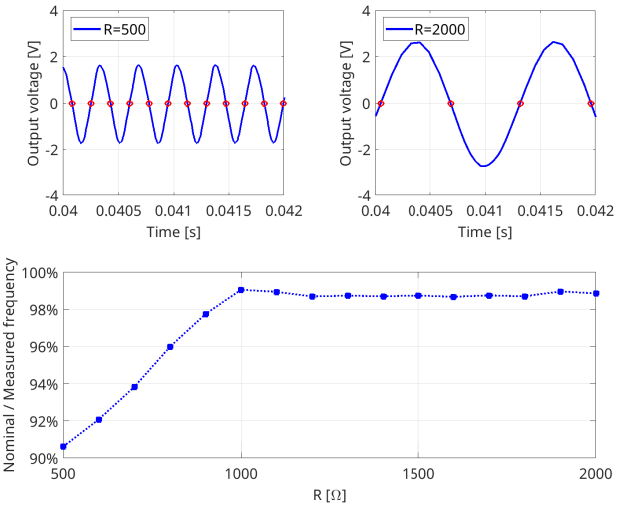
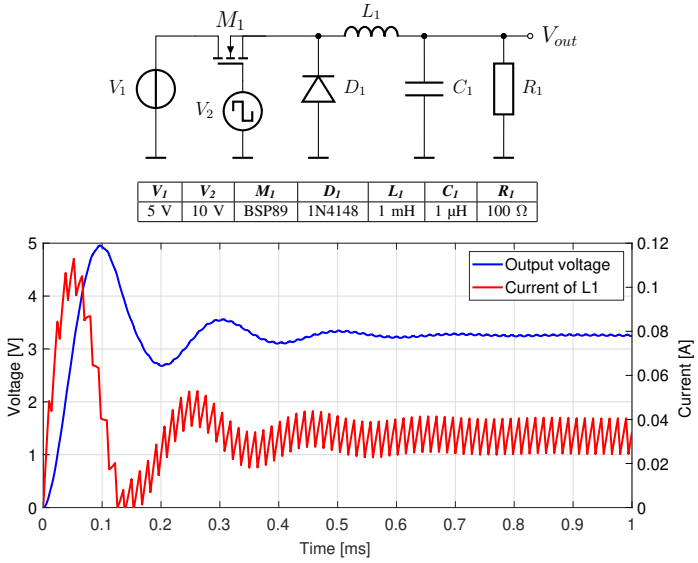



Fig. 6. Wien-bridge oscillator

and the output frequency sensitivity for different parameters is also influenced by the fact whether the slew rate limit is achieved or not. The bottom plot shows the frequency limiting effect of the slew rate for different values of resistor $R$.

Figure 7 shows a simplified model of a switching mode power supply. It is a buck converter intended to output a lower DC voltage than the input.




| $V_1$ | $V_2$ | $M_1$ | $D_1$ | $L_1$ | $C_1$ | $R_1$ |
|---|---|---|---|---|---|---|
| 5 V | 10 V | BSP89 | 1N4148 | 1 mH | 1 µH | 100 Ω |

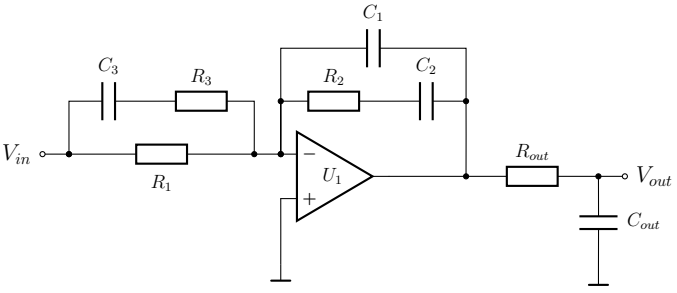
Fig. 7. Simple buck converter and its power-on transient simulation

The simulation represents the power-on transient of this buck converter. The overshoot in both the output voltage and the inductor current can be observed after power-on as well as some oscillations with the voltage and current ripple that persists in a steady state. This circuit allows to test how to distinguish between the high-frequency oscillations and the normal settling of the envelope, the overshoot in the initial transient and the steady state. Generally, the electronic components' parasitic parameters can also affect the resulting signal shapes so that they can be tested during worst-case analysis.

Figure 8 serves as an example of a separable circuit.

Breaking the system into smaller subcircuits reduces the complexity of parameter space, so the worst-case value is easier to find. The example is a linear system; thus, the circuit equations will yield a transfer function in the s-domain, which can be expressed as the fraction of two polynomials. The circuit parameters can be partitioned into disjoint subsets affecting the poles and zeros in the transfer function. Finally, the resulting subcircuits can be evaluated separately based on the parameter groups. There are similar circuits where the overall computation time can be significantly reduced thanks to independent parameters [3].

Figure 9 shows a current limiter with fold-back characteristics. Its primary purpose is to reduce the short circuit current while allowing full output current during regular operation. Its main characteristics are the maximum current and voltage, the slope, and the region near the folding point. In this particular case, the sensitivity to the component parameters proved to be particularly important.




| $R_1$ | $R_2$ | $R_3$ | $R_{out}$ | $C_1$ | $C_2$ | $C_3$ | $C_{out}$ |
|---|---|---|---|---|---|---|---|
| 10 kΩ | 30 kΩ | 560 | 10 kΩ | 6p pF | 1.1 nF | 3.3 nF | 6 pF |

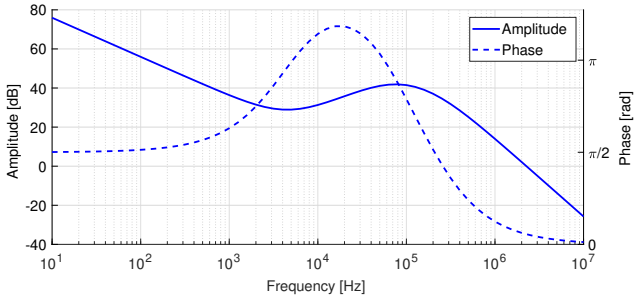
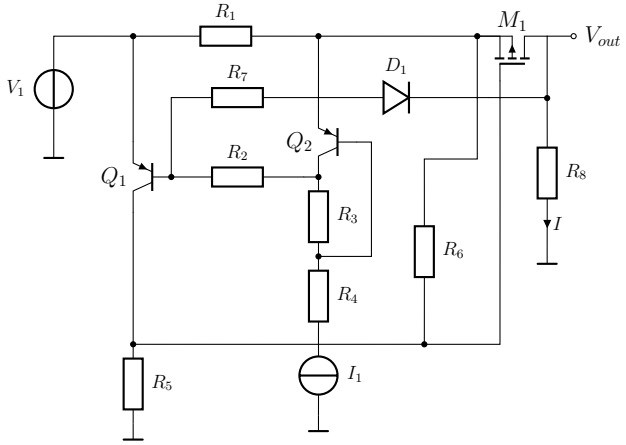



Fig. 8. Active filter circuit for testing separability




| $R_1$ | $R_2$ | $R_3$ | $R_4$ | $R_5$ | $R_6$ | $R_7$ |
|---|---|---|---|---|---|---|
| 0.25 Ω | 2.2 kΩ | 400 Ω | 2.4 kΩ | 47 kΩ | 470 kΩ | 100 kΩ |
| $R_8$ | $Q_1$ | $Q_2$ | $V_6$ | $I_7$ | $D_8$ | $M_1$ |
| 1...201 Ω | MMBT2907A | MMBT2907A | 5 V | 0.43 mA | 1N4148 | IRF4905 |

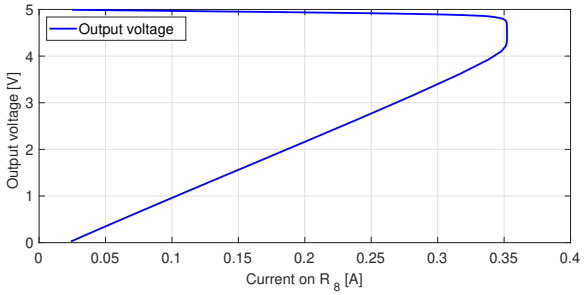



Fig. 9. Current limiter with fold-back characteristics

TABLE I
EXAMPLES OF CIRCUITS WITH TYPICAL BEHAVIOR

| **Example circuit** | **Special property** | | |
|---|---|---|---|
| | ***Nontrivial extreme value*** | ***Separability*** | ***Characteristic features*** |
| Transistor switch | ★ | | ★ |
| Band-pass filter | | ★ | ★ |
| Simple buck converter | ★ | | ★ |
| Wien-bridge oscillator | | | ★ |
| Active filter | ★ | ★ | |
| Current limiter | ★ | | ★ |

Table I summarizes the above discussion. It shows, how the circuits can be categorized according to the research goals.

In addition to the examples described above, other circuits of varying complexity are also included in the database. Even for structurally simple circuits, the number of parameters increases significantly if complex component models are used. Although it is indeed important to mention how the circuits' complexity affects the software environment's computational requirements, this aspect is not discussed in the paper, as further research is needed to obtain quantitative results. The research goals include the possibility of automatically decomposing complex circuits to some extent, thus reducing the computational complexity.

## V. CONCLUSIONS

This paper presented a framework designed to improve Worst-Case Analysis processes for electronic circuits. The framework integrates a software environment with a circuit database, addressing challenges in circuit analysis by facilitating automation, optimizing simulation workflows, and supporting a range of analysis methodologies.

The framework provides tools to assist in developing and validating WCA algorithms. The individual software components in the framework are integrated into a coherent system that supports diverse methodologies and advanced analytical approaches, enhancing analysis efficiency and flexibility while runtime optimization is also considered.

The main scientific contribution of this paper is a set of circuits. This database is intended to be used as a test collection to develop advanced analysis methods that can automatically identify typical properties of circuits and apply them in the analysis process. An important consideration in the design of the circuits was the choice of parameters such that the circuits exhibit the desired properties.

Potential extensions of the framework include incorporating machine learning techniques to enhance predictive capabilities and feature extraction. The presented collection of circuits can also be further extended. However, even in this form, they can also serve as a reference and benchmark for testing many analysis methods, as this paper illustrates with examples. Additional features like advanced visualization tools could further support analysis and design tasks.

Overall, the framework provides a structured approach for advancing WCA methods and fosters innovation in automated circuit analysis.

## REFERENCES


[1] B. Johanson, D. Russell, W. Swavely, Reliability Analysis Center, "Worst case circuit analysis application guidelines," Reliability Analysis Center, 1993

[2] B. A. Lenertz, "Electrical design worst-case circuit analysis: guidelines and draft standard (REV A)," AEROSPACE REPORT NO. TOR-2013-00297, The Aerospace Corporation, June 3, 2013.

[3] B. Bak, P. Weisz, Gy. Orosz, "Worst-Case Circuit Analysis using Schematic Conversion and Decomposition," *2024 25th International Carpathian Control Conference (ICCC)*, Krynica Zdrój, Poland, 22-24 May 2024, doi: 10.1109/ICCC62069.2024.10569978

[4] C. M. Rocco, "Variability analysis of electronic systems: classical and interval methods," in *Annual Reliability and Maintainability Symposium*, Philadelphia, PA, USA, 1997, pp. 188-193, doi: 10.1109/RAMS.1997.571704.

[5] N. Femia and G. Spagnuolo, "True worst-case circuit tolerance analysis using genetic algorithms and affine arithmetic," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 9, pp. 1285-1296, Sept. 2000, doi: 10.1109/81.883323.

[6] T. Ding, L. Zhang, R. Trinchero, I. S. Stievano and F. G. Canavero, "Worst-Case analysis of electrical and electronic equipment via affine arithmetic," in *2017 International Conference on Electromagnetics in Advanced Applications (ICEAA)*, Verona, 2017, pp. 991-993,

[7] M. W. Tian and R. C.-J. Shi, "Worst case tolerance analysis of linear analog circuits using sensitivity bands," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 8, pp. 1138-1145, Aug. 2000

[8] M. Sunun and S. Uatrongjit, "Improvement of sensitivity band technique for worst case tolerance analysis of linear circuits," in *2008 5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, Krabi, Thailand, 2008, pp. 713-716

[9] E. Niculescu, D-M. Purcaru, M. Niculescu, "An approach to worst-case circuit analysis," *WSEAS Transactions on Electronics*, ISSN: 1109-9445, vol. 4, no. 10, Oct. 2007., pp. 237-244

[10] K. Horváth, B. Bank, Gy. Orosz, "Towards Automated Worst-Case Analysis of Circuits: Selecting Initial Values for Global Optimization," *30th Minisymposium, Budapest, Hungary, 2023, 4 p.*

[11] P. Weisz, "Circuit Database for Testing Worst-Case Circuit Analysis Methods" mit.bme.hu https://home.mit.bme.hu/~weisz/circuit-database (accessed Jan. 23, 2025).

# Novel Organ-on-a-Chip device for high throughput drug candidate screening

Norman Sepsik[*], Ferenc Ender[*†], György T. Balogh[‡]
[*] Budapest University of Technology and Economics, Department of Electron Devices
Budapest, Hungary
Email: sepsik.norman@edu.bme.hu, ender.ferenc@vik.bme.hu
[†] Spinsplit LLC
Budapest, Hungary
[‡] Budapest University of Technology and Economics, Department of Chemical and Environmental Process Engineering
Budapest, Hungary
Email: balogh.gyorgy@vbk.bme.hu

***Abstract*— Organ-on-a-Chip microreactors are in vitro microfluidic devices capable of modelling the function of individual organs of a living organism. The developed Liver-on-a-chip device models certain aspects of liver metabolism. One of the important steps of pharmacokinetic studies related to the preclinical phase of drug research is the in vitro modelling of the liver-bound metabolism of candidate molecules. Drug metabolism in the living organism usually occurs by oxidative transformation catalysed by the cytochrome P450 (CyP450) family of enzymes, which are present in larger quantities in the liver. In the practice of (industrial) pharmaceutical research, a biological model based on liver microsomes (endoplasmic reticulum of hepatocytes) containing the above enzyme is used, but this system has many known limitations. As part of our research, we are developing a new nanocomposite system that is a biomimetic alternative to the currently used in vitro biological model. In the biomimetic system, instead of a liver microsome, we use a metalloporphyrin organocatalyst system immobilised on a nanoparticle and embedded in a polymer nanofiber, which, as a structural analogue of the active site of the CyP450 enzyme family, models the transformation carried out by the microsome, and a potential device for cheap, high throughput metabolite screening.**

***Keywords— drug metabolism, electrospinning, microfluidics, nanofibers, organ-on-a-chip, liver-on-a-chip***

## I. Introduction

The process of drug discovery and development is a costly and time-consuming process, involving the screening and research of tens of thousands of Active Pharmaceutical Ingredient (API) candidates to find the right drug substance, which on average takes 10-12 years to bring a drug to market, with the cost of drug development typically ranging between $0.8 and $1.8 billion [1]. The process consists of distinct phases, involving different activities, carried out according to different rules, but building on each other, and their order determined by their effectiveness [2]. Drug candidates are investigated and sorted on the basis of Absorption, Distribution, Metabolism and Excretion (ADME) studies. The aim of these studies is to model the time course of drug concentration in the circulation and in organs and tissues of the body, and to estimate or measure bioavailability [3]. Orally ingested drug passes first through the stomach and then through the initial part of the intestinal tract (duodenum). From there, the majority of the active substance is transported to the liver by the so-called portal circulation (about *70%*), while a smaller proportion is excreted (without being utilized) through the intestinal tract. The active substance that reaches the liver is metabolically modified or degraded in the liver before entering the systemic circulation. As a consequence, only a fraction of the original dose (about *15%*) drug is available at the site of action, moreover the substance might be chemically modified due to the oxidation in the liver [4],[5]. Liver-dependent metabolism involves so-called first-phase reactions such as oxidation, reduction and hydrolysis [6]. Oxidative metabolism as the primary way of metabolism, is mediated by heme group-containing proteins belonging to the superfamily of cytochrome P450 (CYP450) enzymes, which are predominantly present in the endoplasmic reticulum of hepatocytes [7]. The by-products of metabolism are called metabolites [8] and can be pharmacologically classified into the following groups [9]: 1) metabolites that have become inactive; 2) fully or partially active metabolites with the intended effect; 3) active but toxic metabolites. One of the main objectives of the research and preclinical phases of drug discovery is to investigate the metabolism of drug candidates, first and foremost to exclude group 1) and 3) metabolites while keeping candidates from group 2) [10]. Several assays are currently used for the above screening process, such as hepatocyte cell culture [11], liver microsomes [2] and recombinant human enzymes (CYP450) [12]. Each of these biological assay has advantages, disadvantages and different throughput, which are all summarised in Table. I. Metalloporphyrins, acting as the catalytic centre of CYP450 enzyme can be synthesized and used directly to mimic the oxidative reaction of CYP450, promising a cheap and high-throughput alternative for recombinant human enzyme assays. Porphyrins are ring compounds built up by four pyrrole heterocyclic molecules linked by methylene bridges [13]. They are able to form complexes with metal ions, these are called metalloporphyrins [14].

Organ-on-a-Chip (OoC) microfluidic devices incorporating cell culture can mimic the physicochemical microenvironment of tissues in the human body under controlled conditions, using only small amounts of fluids in the nanolitre range. Among the many applications, the most important are drug development and the study of the effects of drugs on specific organs [15]. One of the important organs

DOI https://doi.org/10.3311/MINISY2025-016

*Table. I. Summary of the biological assays used in drug screening and development*

| ***Assay*** | ***Purpose of the assay*** | ***Advantages*** | ***Disadvantages*** | ***Throughput (molecule/day)*** | ***Ref.*** |
|---|---|---|---|---|---|
| Hepatocytes | Examination of the accumulation and excretion of toxic materials and xenobiotics, as well as the stability of the metabolite showing an exact in-vitro - in-vivo correlation | They contain the enzymes and other cofactors belonging to the first and second phase metabolic reactions in a physiological concentration, which is why they show a good in-vitro - in-vivo correlation | Maintaining the function of hepatocytes in in vitro conditions is difficult, so their use is limited if, for example, metabolites are formed slowly<br>The throughput is low | Low | [11]<br>[20] |
| Liver microsome | Quantitative metabolic catalysis, metabolite stability study | Higher throughput, simpler use, lower cost, easier storage compared to the hepatocyte model | The enzyme concentration and number of cofactors do not correspond to the physiological conditions, so the in-vitro - in-vivo correlation is weaker than in the case of the hepatocyte model | 10-100 | [20]<br>[21]<br>[22] |
| Recombinant human enzymes | Enzyme kinetics, qualitative analysis of specific metabolite catalysis | The selective production of metabolites, as well as the fact that they are easier to use and have a higher throughput compared to the microsome system | They are not suitable for examining the effect of the activity of metabolites on the physiological cell system | 100-1000 | [21]<br>[23] |
| Metalloporphyrin | Exploiting the so-called biomimetic property to create metabolites in one step | Creating metabolites in one step without the presence of a complex biological matrix | They have relatively low stability in homogeneous oxidative systems.<br>Their automation for active substance screening is not a solved problem | Could be higher than recombinant human enzymes | [18]<br>[24] |

under investigation is the liver, whose function can be mimicked by OoC devices called Liver-on-a-chip (LoC). These can provide a 3D environment instead of the previously presented liver metabolism assay methods [16]. In recent years, several approaches have been developed to design a suitable 3D microenvironment and improve the metabolic function of hepatocytes in vitro [17]. In the present study, a LoC microreactor was developed containing a metalloporphyrin (FeTPPS) based biomimicking compound embedded in nanofibers. Demonstration measurements were carried out with the fabricated device to prove its biomimetic performance for a prospective future use as a high throughput drug candidate screening device.

## II. MATERIALS AND METHODS

### *A. Production of FeTPPS-MNP/PLA nanocomposite*

The composition of the precursor solution used for fibre formation and the parameters for fibre formation were chosen based on the work of Balogh-Weiser et al [18].

Polylactic acid (PLA, *230 kDa*) was dissolved in *8 wt%* dichloromethane (DCM) and N,N-dimethylformamide (DMF) (DCM:DMF, *V/V, 6:1*) in a *10 ml* glass vial and stirred continuously at room temperature on a magnetic stirring table at room temperature at *450 RPM* until the PLA was completely dissolved. Then, magnetic nanoparticles (MNPs) with immobilized FeTPPS (5,10,15,20-Tetrakis(4-sulfonatophenyl) porphyrinato Iron (III), Chloride) molecules were added to the PLA solution *1.25 w/w%* and formed into nanofibers by means of electrospinning. The process is described in details elsewhere [18]. MNPs were suspended in *400 µl* DCM:DMF (*V/V, 6:1*) and sonicated the solutions for *10 min*, after mixing with the PLA polymer solution, the final precursor solution was vortexed for another one minute.

For the preparation of nanofibrous composite (PLA/MNP-FeTPPS), we used a horizontal electrospinner system (Spincube, Spinsplit LLC, Budapest Hungary, Figure 1) with a feed rate of *18 µl* $min^{-1}$, and accelerating voltage of *19 - 21 kV* and a capillary diameter of *1.2 mm* (*22 G*). The precursor solutions were placed in a *3 ml* syringe. The distance between emitter and collector plate was *15 cm*. Fibre formation was carried out at room temperature with a relative humidity of *40-60 RH%*. First, an aluminium foil was fixed on the flat collector plate, then 6 glass fibre mesh of *180 x 10 mm* were placed transversely under each other and the fibres were formed on them. Using the glass fibre support surfaces, the fibre web can be easily wrapped around the inserts to be placed in the microreactor. The mass of the collector before and after fibre formation was measured on a precision laboratory balance to ensure consistency between the samples.

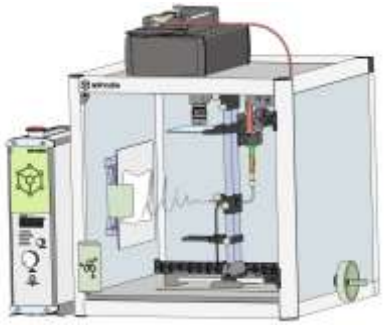


*Figure 1. Schematic view of the electrospinning setup for the nanocomposite formation*

*B. Biocompatibility assessment of raw materials*

The aim of the biocompatibility study was to select from the available 3D printing materials those that do not react, i.e. behave inertly, with the reagents that we plan to use in the first testing of the LoC device, such as the drug molecule amlodipine, the MeOH:Ac (Methanol:Acetate) buffer, the oxidizing agent *t*BuOOH (*tert*-butyl hydroperoxide) and the catalyst FeTPPS. The measurements were carried out using Ultra-violet visible (UV-Vis) spectroscopy and high-performance liquid chromatography with UV detection coupled with electrospray ionization tandem mass spectrometry (HPLC-UV-MS) analytical techniques to investigate the biocompatibility of the polymers. Available polymers used in 3D printing were investigated for their biocompatibility were: 1) ABS (Acrylonitrile Butadiene Styrene), 2) ASA (Acrylonitrile Styrene Acrylate), 3) CPE-HG 100 (Co-polyesther based filament), 4) PLA, 5) Conductive polymer, 6) Resin used in stereolithography (SLA resin), 7) PP (Polypropylene), all purchased from 3DJake niceShops GmbH, Paldau, Germany.

For the UV-Vis measurement, experiments were performed in *1.5 ml* screw glass tubes and the total volume of the reaction mixtures was *0.5 ml* in each case. The reference samples contained *486 µl* methanol and *14 µl* FeTPPS (*0.735 mM*, dissolved in methanol). The test samples contained *486 µl* of methanol and *14 µl* of FeTPPS (*0,735 mM*, dissolved in methanol), as well as the polymer samples. *10 mg* of each polymer filament and *30-40 mg* of the SLA resin sample were measured. The reaction mixtures were shaken for *72 h* at *37 °C* at *1100 RPM* (Eppendorf Thermomixer Comfort shaker Eppendorf GmbH, Germany) after one hour and then *72 hours* of continuous shaking, samples of the reaction mixtures were taken and UV-Vis spectra were recorded in the wavelength range from *200 nm* to *700 nm* (Multiskan SkyHigh measuring tray spectrophotometer, Thermo Fisher Scientific, Waltham, MA, USA). The measurements were repeated on 3 replicate samples.

To assess the effect of the material candidates on the biocatalytic reaction, reaction mixtures were prepared, and the reaction was followed by HPLC-UV-MS. Reference samples contained *490 µl* of methanol and *10 µl* of *10 mM* drug stock solution. Blank samples contained *384 µl* methanol, *10 µl 10 mM* drug stock solution, *6 µl t*BuOOH stock solution (*147 mM*) and *100 µl* aqueous buffer solution *pH 4.5* (*64 mM*). The biomimetic nature of the reaction was confirmed for the blank sample. The samples tested contained *370 µl* of methanol, *10 µl* of *10 mM* drug stock solution, *14 µl* of FeTPPS (*0.735 mM*, dissolved in methanol) solution, *6 µl* of *t*BuOOH stock solution (*147 mM*) and *100 µl* of aqueous buffer solution (*64 mM*) at *pH 4.5*. Measurements were performed using a Waters Micromass Quattro Ultima Pt tandem quadrupole mass spectrometer coupled to a Waters 2690 liquid chromatography system (Milford, MA, USA). Detection was performed with a Waters 2487 Dual λ Absorbance detector. The analysis was performed at *40 °C* on a Kinetex XB C18 column (*150x4.6 mm, 2.6 µm*) (Phenomenex, Torrance, CA, USA) at a mobile phase flow rate of *1 ml min*$^{-1}$. The composition of eluent A used was *0.1 vol %* formic acid dissolved in water and eluent B was a 95:5 mixture of MeCN:H2P to which *0.1 vol %* formic acid was added. The gradient elution program: a linear gradient of *5-100%* B was used in the range *0-11 min*, and an isocratic gradient of *100%* B was used in the range *11-13 min*. This was followed by a *2-min* equilibration period before the next injection using a *5%* B formulation. The injection volume was adjusted to *5 µl* and the chromatographic profile was recorded at a wavelength of *240 (± 4) nm*. The mass spectrometry conditions were as follows: ion source ESI, positive ion mode, scanning ion mode (*150 - 600 m/z*), drying gas ($N_2$) temperature *350 °C*, flow rate *5.5 l min*$^{-1}$, nebulizer gas ($N_2$) pressure *6 bar*, quadrupole temperature *120 °C*, capillary voltage *2500 V*, fragmentor voltage *60 V*. Data were processed using MassLynx 4.0 software.

*C. Production of microreactor cassette*

The microreactor cassettes were produced using Fused Deposition Modelling (FDM) 3D printing technology from the selected polymeric material considered to be biocompatible. The cassettes were prepared in several iterations. The first 3 versions were made with the CraftBot Plus printer, the others with the Prusa i3 MK3 printer. The main considerations for the design were ease of assembly, simple design, and adequate internal chamber size to accommodate the necessary nanofiber composite.

*D. Demonstration of the LoC microreactor*

The experiment was performed using the microreactor design shown in Figure 2. Demonstration measurements of the microreactor cassette were carried out in the arrangement shown in Figure 3. The experimental set-up consisted of an HPLC pump (1), a stock solution containing the drug substance (2) and a microreactor (3). Samples were collected from the output of the microreactor every *4 minutes* for *1 minute* into *5 ml* vials (4). Samples were then pipetted into *1 ml* containers suitable for subsequent HPLC analysis and stored in a sample container (5), refrigerated at *5 °C* until analysis by HPLC.

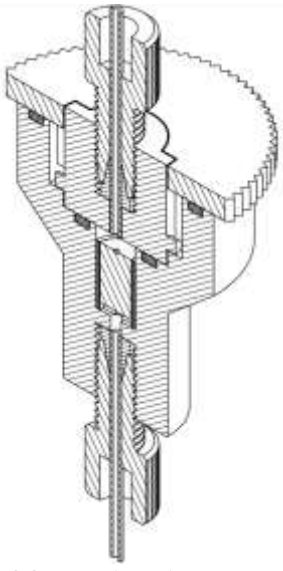


*Figure 2. The LOCV5 version microreactor used during the demonstration measurements*

Flow rate settings relied on our previous studies can be found elsewhere [19]. The system was flushed with distilled water between the two wiring modes, without changing the charge. The experiments were carried out for 45 minutes.


*Figure 3. Experimental set-up for the demonstration measurements*

The experiments were performed out to determine the concentration of amlodipine and its human metabolite, dehydroamlodipine, in samples taken from the microreactor outlet as a function of sampling time by HPLC-UV-MS.

## III. Results and Discussion

### A. Biocompatibility of the polymer materials

The resulting spectra of UV-Vis analysis can be seen in Figure 4. It can be observed that, compared to the reference FeTPPS spectrum, PLA, CPE, conductive polymer and PP show negligible differences (therefore their curves coincide), indicating no influence on FeTPPS, nor dissolution into the reaction mixture. For SLA resin and ASA, the intensity of the peak ($\lambda = 393$ *nm*) characteristic of FeTPPS changed indicating a decrease in the amount of FeTPPS, also new peaks were observed presumably due to the dissolution of the given substances into the reaction mixture. ABS found to be biocompatible for short term use only (1 hour residence time), while after *72 hours* a new spectral peak (at a wavelength of approx. $\lambda = 251$ *nm*) indicated a possible dissolution.

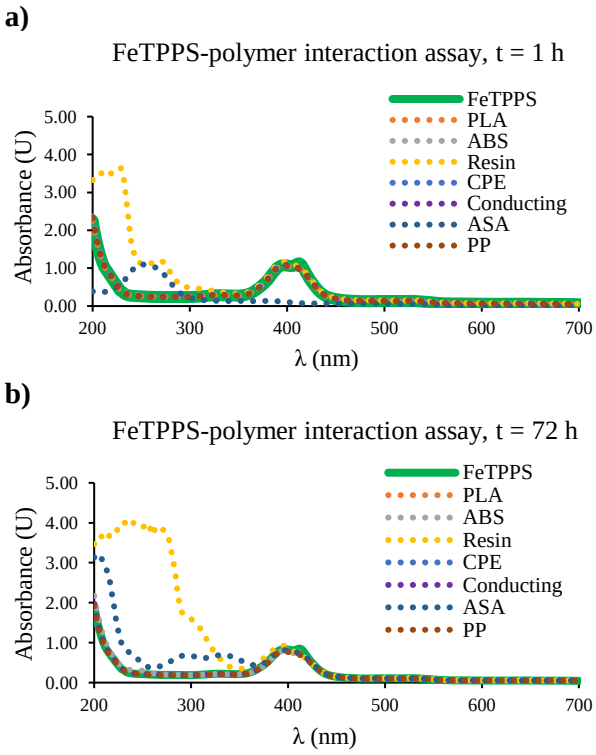



*Figure 4. Result spectra of the FeTPPS-polymer interaction assay after 1 hour (a) and after 72 hours (b)*

Four polymers PLA, CPE, Conducting, PP were investigated in presence of biomimetic oxidation, compared to a reference case (without polymer) using the HPLC-UV-MS method. It can be observed that the value of the conversion compared to the reference reaction ("Biomimetic", *86.72%*) slightly increased in the presence of PP (*87.58%*), so it had a favourable effect on the process of biomimetic oxidation. The conducting polymer, on the other hand, completely inhibits the reaction (*0%*), because it catalysed the complete decomposition of the oxidizing agent, and in the case of CPE we can also see a significant inhibition (*38.61%*). In the case of PLA, the conversion of the parent compound (*73.67 %*) is similar to the case of PP, but presumably PLA catalyses the decomposition of the oxidant to a small extent or adsorbs it on its surface. The results can be seen in Figure 5. Since PLA is a material that can be used much more universally during 3D printing compared to PP, we carried out further experiments with this material, assuming that a final construction should (also) be implemented on a PP basis. Furthermore, PLA does not react with the amlodipine drug molecule used later and does not hinder biomimetic oxidation.

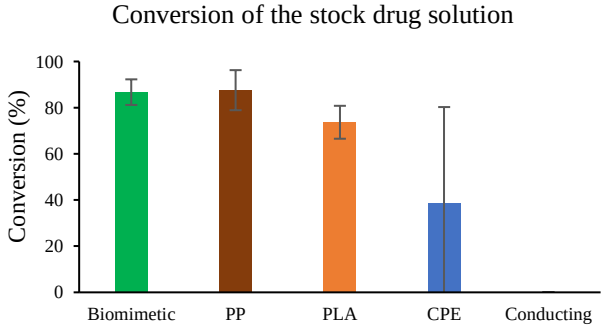



*Figure 5. Conversion of the drug solution in the presence of polymers that were declared inert during UV-Vis measurement*

### B. *Demonstation of the LoC microreactor*

The tested microreactor did not show any signs of leakage until the first four samplings (20 minutes), however in the second half of the experiment minimal leakage cannot be excluded due to liquid found in the bayonet lock track following the disassembly of the reactor. Changes in the conversion and concentration of dehydroamlodipine over time can be seen in Figure 6. Figure 6 a) shows the change of concentration in time both for amlodipine and its human metabolite. The concentration of both materials was found more or less unchanged in the first 40 minutes, however a rapid fall of amlodipine, and, respectively, an elevation of dehydroamlodipine concentration was found between 40 and 50 minutes. Accordingly, the reactor showed low conversion in the first 40 minutes, following that the conversion ratio increased rapidly until it reached *53.04 %* (Figure 6 b).

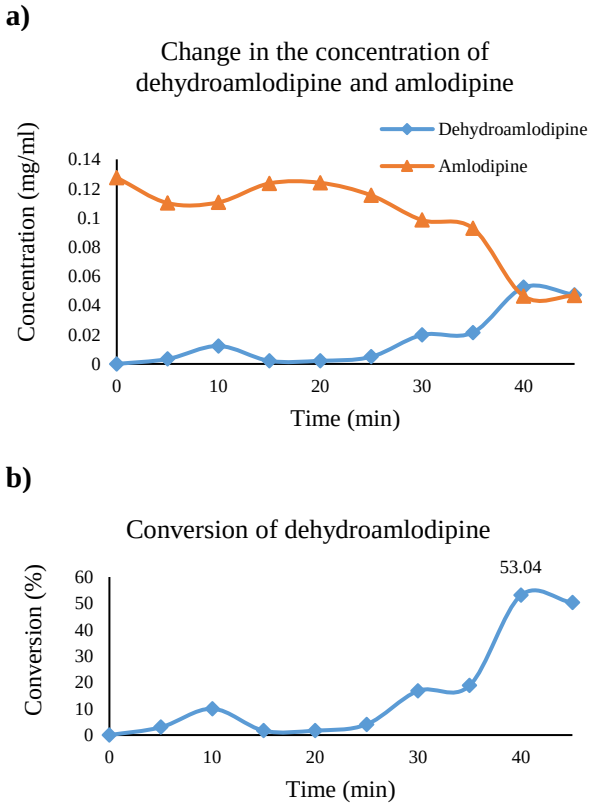



*Figure 6. The change in concentration of amlodipine and its metabolite dehydroamlodipine (a) and the conversion of the drug stock solution (b) during the course of the measurement*

## IV. Summary and Conclusion

The previously developed nanofiber composite system was integrated into a microreactor cassette designed for this purpose. The biomimetic operation of the microreactor was demonstrated and significant conversion rate was achieved in the biomimetic oxidation of amlodipine to dehydroamlodipine.

The conversion reached its maximum after about 40-45 minutes which might be a consequence of internal residual time conditions within the microreactor and requires further investigation. The current microreactor cassette can be a good candidate for further upscaling and demonstration for high throughput drug candidate screening.


## Acknowledgment

This work was prepared in the framework of 2021-1.1.4-GYORSÍTÓSÁV-2022-00059 project with the support of the Ministry of Culture and Innovation, from the National Research, Development and Innovation Fund, Hungary.


## References


[1] B. Shaker, S. Ahmad, J. Lee, C. Jung, and D. Na, "In silico methods and tools for drug discovery," Oct. 01, 2021, *Elsevier Ltd*. doi: 10.1016/j.compbiomed.2021.104851.

[2] G. M. Keserű, Ed., *A gyógyszerkutatás kémiája*. Akadémiai Kiadó, 2015. doi: 10.1556/9789630590761.

[3] A. Reichel and P. Lienau, "Pharmacokinetics in drug discovery: an exposure-centred approach to optimising and predicting drug efficacy and safety," in *Handbook of Experimental Pharmacology*, vol. 232, Springer New York LLC, 2016, pp. 235–260. doi: 10.1007/164_2015_26.

[4] D. Choudhary, H. Goykar, D. Kalyane, R. K. Tekade, and N. Sreeharsha, "Prodrug design for improving the biopharmaceutical properties of therapeutic drugs," in *The Future of Pharmaceutical Product Development and Research*, Elsevier, 2020, pp. 179–226. doi: 10.1016/B978-0-12-814455-8.00006-2.

[5] X. He, "Chapter 18 - Integration of Physical, Chemical, Mechanical, and Biopharmaceutical Properties in Solid Oral Dosage Form Development," in *Developing Solid Oral Dosage Forms*, Y. Qiu, Y. Chen, G. G. Z. Zhang, L. Liu, and W. R. Porter, Eds., San Diego: Academic Press, 2009, pp. 407–441. doi: https://doi.org/10.1016/B978-0-444-53242-8.00018-7.

[6] M. Feghali, R. Venkataramanan, and S. Caritis, "Pharmacokinetics of drugs in pregnancy," Nov. 01, 2015, *W.B. Saunders*. doi: 10.1053/j.semperi.2015.08.003.

[7] D. Iacopetta *et al.*, "Impact of Cytochrome P450 Enzymes on the Phase I Metabolism of Drugs," May 01, 2023, *MDPI*. doi: 10.3390/app13106045.

[8] O. A. Almazroo, M. K. Miah, and R. Venkataramanan, "Drug Metabolism in the Liver," Feb. 01, 2017, *W.B. Saunders*. doi: 10.1016/j.cld.2016.08.001.

[9] F. J. Dowd, "Pharmacokinetics: The Absorption, Distribution, and Fate of Drugs," in *Pharmacology and Therapeutics for Dentistry: Seventh Edition*, Elsevier, 2017, pp. 15–43. doi: 10.1016/B978-0-323-39307-2.00002-3.

[10] I. P. Nnane and X. Tao, "DRUG METABOLISM | Metabolite Isolation and Identification," in *Encyclopedia of Analytical Science (Second Edition)*, Second Edition., P. Worsfold, A. Townshend, and C. Poole, Eds., Oxford: Elsevier, 2005, pp. 305–311. doi: https://doi.org/10.1016/B0-12-369397-7/00109-6.

[11] D. Zhang, G. Luo, X. Ding, and C. Lu, "Preclinical experimental models of drug metabolism and disposition in drug discovery and development," *Acta Pharm Sin B*, vol. 2, no. 6, pp. 549–561, Dec. 2012, doi: 10.1016/j.apsb.2012.10.004.

[12] K. Schroer, M. Kittelmann, and S. Lütz, "Recombinant human cytochrome P450 monooxygenases for drug metabolite synthesis," Aug. 01, 2010, *John Wiley and Sons Inc.* doi: 10.1002/bit.22775.

[13] F. Bryden and R. W. Boyle, "Metalloporphyrins for Medical Imaging Applications," in *Advances in Inorganic Chemistry*, vol. 68, Academic Press Inc., 2016, pp. 141–221. doi: 10.1016/bs.adioch.2015.09.003.

[14] E. V Antina and N. Sh Lebedeva, "Molecular Porphyrin and Metalloporphyrin Complexes," 2001.

[15] A. E. Danku, E. H. Dulf, C. Braicu, A. Jurj, and I. Berindan-Neagoe, "Organ-On-A-Chip: A Survey of Technical Results and Problems," Feb. 10, 2022, *Frontiers Media S.A.* doi: 10.3389/fbioe.2022.840674.

[16] N. Azizipour, R. Avazpour, D. H. Rosenzweig, M. Sawan, and A. Ajji, "Evolution of biochip technology: A review from lab-on-a-chip to organ-on-a-chip," *Micromachines (Basel)*, vol. 11, no. 6, pp. 1–15, Jun. 2020, doi: 10.3390/mi11060599.

[17] T. Messelmani *et al.*, "Liver organ-on-chip models for toxicity studies and risk assessment," Jun. 01, 2022, *Royal Society of Chemistry*. doi: 10.1039/d2lc00307d.

[18] D. Balogh-Weiser *et al.*, “Novel biomimetic nanocomposite for investigation of drug metabolism,” *J Mol Liq*, vol. 368, p. 120781, Dec. 2022, doi: 10.1016/J.MOLLIQ.2022.120781.

[19] D. Balogh-Weiser *et al.*, “Magnetic nanoparticles with dual surface functions—efficient carriers for metalloporphyrin-catalyzed drug metabolite synthesis in batch and continuous-flow reactors,” *Nanomaterials*, vol. 10, no. 12, pp. 1–16, Dec. 2020, doi: 10.3390/nano10122329.

[20] A. P. Li, “The scientific basis of drug-drug interactions: Mechanism and preclinical evaluation,” 1998.

[21] L. Di, E. H. Kerns, S. Q. Li, and G. T. Carter, “Comparison of cytochrome P450 inhibition assays for drug discovery using human liver microsomes with LC-MS, rhCYP450 isozymes with fluorescence, and double cocktail with LC-MS,” *Int J Pharm*, vol. 335, no. 1–2, pp. 1–11, Apr. 2007, doi: 10.1016/j.ijpharm.2006.10.039.

[22] S. Asha and M. Vidyavathi, “Role of human liver microsomes in in vitro metabolism of drugs-A review,” Mar. 2010. doi: 10.1007/s12010-009-8689-6.

[23] M. Ooka, C. Lynch, and M. Xia, “Application of in vitro metabolism activation in high-throughput screening,” Nov. 01, 2020, *MDPI AG*. doi: 10.3390/ijms21218182.

[24] F. Li, Y. Li, Y. Wan, H. Lv, X. Gao, and Y. Yu, “Metalloporphyrin-Based Biomimetic Catalysis: Applications, Modifications and Flexible Microenvironment Influences (A Review),” Jan. 01, 2023, *Pleiades Publishing*. doi: 10.1134/S1070363223010255.