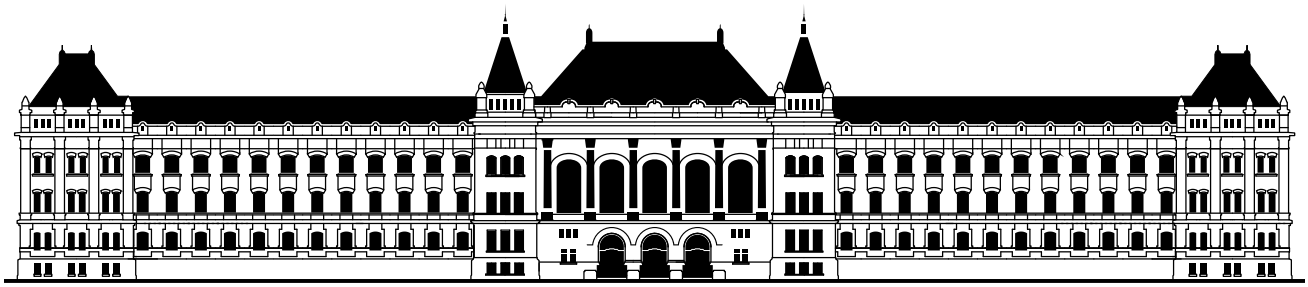
PROCEEDINGS OF THE 31ST MINISYMPOSIUM

OF THE

DEPARTMENT OF MEASUREMENT AND INFORMATION SYSTEMS
BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS

(MINISY@DMIS 2024)

FEBRUARY 5–6, 2024
BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS



BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
DEPARTMENT OF MEASUREMENT AND INFORMATION SYSTEMS

ISBN 978-963-421-951-4

Head of Department:
Tamás Dabóczi

General Chair:
Balázs Renczes

Scientific Chairs[1]:
Tadeusz Dobrowiecki
András Pataricza
Gábor Péceli

Local Chairs:
Lóránt Tibor Csőke
Attila Ficsor
Gábor Révy
András Wiesner

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, as well as of research reports that were presented at the Minisymposium:
Péter Antal, Balázs Bank, Tamás Bartha, Bence Bolgár, Bence Bruncsics, Márton Elekes, András Földvári, András Gézsi, László Gönczy, Bence Graics, Gábor Hullám, Attila Klenik, Imre Kocsis, István Majzik, Zsolt Kollár, Kristóf Marussy, András Millinghoffer, Vince Molnár, György Orosz, Béla Pataki, Vilmos Pálfi, István Ráth, Gábor Révy, Péter Sárközy, Oszkár Semeráth, András Vörös

Foreword

On behalf of the Organizing Committee, I welcome you to the 31st Minisymposium of the Department of Measurement and Information Systems at the Budapest University of Technology and Economics. It is a pleasure to announce that, similar to last year's practice, the Symposium is held as a part of VIK Inference, the Conference of the Faculty of Electrical Engineering and Informatics.

We have witnessed an increasing number of PhD students over the last couple of years. As a result, we will have two complete days with more than thirty presentations.

Besides the regular sessions, participants will also have the opportunity to discuss the research topics with the talented IMSc students of the Department.

The presentation topics show the diverse research areas of the Department: we will have talks on Digital Signal Processing, Embedded Systems, Artificial Intelligence, and Fault-Tolerant Systems.

I hope the 31st Minisymposium will help the participants get acquainted with the topics covered at the Department while generating further research conversations.

Budapest, February 5. 2024

Balázs Renczes
General Chair

Contents

Design Optimization of a Current Sensing Trace with respect to Skin Effect by FEM Simulations

Ákos Ferenc Hegedűs, Tamás Dabóczi
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
hegedus43akos@yahoo.com, daboczi@mit.bme.hu

***Abstract*—We have developed a new current sensing method, CSRTRI (Current Sensing by Real-Time Resistance Identification), the feasibility of which was demonstrated in [1], where we reported an achieved accuracy of 0.93%...1.10% and a bandwidth of DC…2 MHz. The working principle is based on the in-situ determination of the current-conducting element's temperature dependent, thus continuously changing resistance with the utilization of an auxiliary inductive sensor signal. The resistance identification takes place at AC, typically at the fundamental PWM-frequency, while the calculated value is applied over the whole current signal spectrum. Therefore, it is a prerequisite of the feasibility of the CSRTRI, that the resistance's frequency-dependence is negligible, from DC up to the first couple of ripple-frequency harmonics. In this paper we investigate this frequency dependence for two design variants: first, that of a single Cu-trace with rectangular cross-section, second an antiparallel low-inductance trace-pair. We conducted AC-magnetics FEM-simulations to assess the influence of the skin-effect on the current trace's resistance. Based on the simulation results presented herein, the antiparallel variant's resistance increase over the DC-30 kHz frequency range is as low as 0.1%, which is acceptable, considering the 2 kHz applied fundamental frequency. The single trace, on the other hand, is prone to a 4.28% resistance rise over the same frequency range, implying that it would have been clearly an unacceptable solution. The prototype sensor system including the optimized low-inductance antiparallel trace, as well as the auxiliary coil, the test setup and measurement results detailed in [1] were also summarized.**

Keywords—current, current control, current measurement, inductive coupling, inductive transducers, resistive transducers, sensor systems and applications, simulation, skin effect

I. Introduction

Electrification is one of the major global technological megatrends in the 21st century. The processes of generation and utilization of electric energy shall be optimized to mitigate the dual problem of fossil fuels' depletion and worsening climate change. Consequently, the efficiency of energy converters is being pushed ever higher. Being a crucial part of the solution, low cost, high bandwidth, high accuracy current sensing methods have increasingly got to the center of scientific and technical interest nowadays [1].

Shunts: The most widespread of these methods is the current measurement with shunts, i.e., converting the current I into a voltage signal U using a shunt resistor R in the current path according to Ohm`s law: $U = I \cdot R$. Shunts are renowned for their best-in-class accuracy (typically under 1%) and bandwidth (2 GHz or more). The low TCR (thermal coefficient of resistance) of the shunt material (e.g., ± 10 ppm/K for Zeranin®), and the additional Kelvin-contacts are key to their outstanding accuracy [2]. The required additional galvanic isolation, high self-heating and cost are the general disadvantages of high-power shunts.

Hall Sensors: State of the art magnetic field current sensing is typically realized by silicon-based Hall-effect sensor ICs [3]. This method utilizes the $U_{Hall} = R_H \cdot I_{bias} \cdot B_z/d$ Hall-voltage signal proportional to the B_z out-of-plane magnetic field component generated by the current. Here the R_H material parameter is called the Hall-coefficient, I_{bias} and d stand for the bias current and the thickness of the Hall-plate, respectively. Piecewise calibrated Hall-sensors typically reach a moderate accuracy of 2-3% and offer inherent galvanic isolation and negligible insertion resistance. Moreover, they are DC-capable, and their bandwidth is between 100 kHz and a few MHz [4],[5]. The limited accuracy and bandwidth are the main disadvantages here.

Magnetoresistance: Magnetoresistive sensor technologies like AMR, GMR and TMR (Anisotropic-, Giant- and Tunneling Magnetoresistance) are also both AC- and DC-capable and exploit the resistance change of a thin film structure induced by the magnetic field of the current. The new and therefore expensive TMR current sensors can reach up to 1 MHz bandwidth and an accuracy better than 1% [6].

Inductive Current Sensors (ICSs): Another widespread current sensing method, utilized by ICSs, is based on the phenomenon of magnetic induction. These sensors include a ferrite- or air-core coil, positioned in proximity to the current conductor. In ICSs, a voltage signal is being induced proportional to the time-derivative of the current's magnetic field, hence to the slope of the current itself according to Faraday's law: $U_{ind} = M \cdot dI/dt$. Here M denotes the mutual inductance, dI/dt denotes the time-derivative of the current. As a direct consequence, this sensor-type is fully insensitive to DC-currents, which constitutes its largest disadvantage. The raw signal is usually integrated by an analog integrator circuit, to get a voltage output, representing the current. They have inherent galvanic isolation and high upper bandwidths up to $f_{max,-3\,dB} = 200$ MHz [7].

II. Challenges in Cu-Trace Current Sensing

The idea to utilize the resistance of a conductor element for current measurement, e.g., that of a rectangular PCB Cu-trace instead of a shunt, arises naturally. With appropriate dimensioning the required DC-resistance value R can be set:

$$R = \frac{\rho \cdot l}{w \cdot d}. \quad (1)$$

Here ρ denotes the resistivity of the copper, l is the trace's length, w is its width, d is its thickness. For a fixed R value

DOI https://doi.org/10.3311/MINISY2024-001

one might increase for example both l and w, with this improving the cooling, and mitigating the self-heating. Furthermore, in power modules ceramic substrate based DCB (Direct Copper Bonded) Cu-traces are in excellent thermal-coupling with the heatsink also. Unfortunately, one runs into some difficulties when trying to implement the concept [8].

Self-Inductance: First, the longer the trace, the larger its self-inductance L is, which contributes to the output voltage signal for time-varying currents. So, the trace can be modelled with a $Z = R + i\omega L$ impedance instead of an ideal resistor R, where i is the imaginary unit, ω is the angular frequency. One possible solution here is to apply antiparallel low-inductance design to shrink the sensing loop:

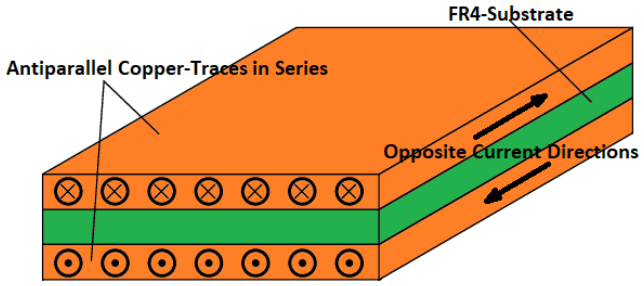


Fig. 1. Low-inductance, PCB-based, current-conductor design.

The other countermeasure makes use of an analog RC-lowpass filter, made of the resistor R_1 and capacitor C_1, where the filter is designed so, that $R_1C_1 = L/R$ is fulfilled, so the dynamic inductive term is cancelled out:

$$U_{filter,out} = I \cdot (R + i\omega L) \cdot \frac{1}{1 + i\omega R_1 C_1} = I \cdot R. \qquad (2)$$

Skin Effect: Second, due to skin effect, in a wide flat conductor with permeability μ, and resistivity ρ, the current density j of frequency f decays into the surface exponentially according to (3) with the characteristic length equal to the skin depth δ_{skin}, assuming the conductor's thickness is severalfold of it:

$$j_z = j_0 \cdot e^{-\frac{z}{\delta_{skin}}}, \qquad (3)$$

where j_0 and j_z are the current density values directly on the surface, and in depth z, respectively, and:

$$\delta_{skin} = \sqrt{\frac{\rho}{\pi \cdot f \cdot \mu}}. \qquad (4)$$

Consequently, the effective conductor cross section decreases over frequency, which causes the resistance to increase. To calculate the frequency dependence of R_{AC}/R_{DC} for conductors of rectangular cross-section, the empirical Haefner-formulae can be applied [9]. For a 16 mm × 35 µm Cu-trace of arbitrary length, with $f = 2$ kHz, one gets 1.23% resistance increase relative to DC, for example.

Manufacturing Tolerances: Third, the dimensions of a specific conductor section in general are subject to manufacturing tolerances, which have a direct effect on its resistance in the order of 10%-20% [1]. This initial uncertainty of R can be overcome by a single End of Line-calibration step. On top of that, corrosion and abrasion may increase the resistance during the product-lifetime.

Thermal Drift: Finally, metals have significant TCR (Thermal Coefficient of Resistance), thereby their resistivity increases strongly over T (temperature). For copper it is $\alpha_{Cu} = 3900$ ppm/K, which implies a large T-dependence both due to varying ambient temperature and self-heating. The T-driven resistance drift can be partially compensated by direct on-trace T-sensing, the accuracy of which largely limits the current-measurement's accuracy itself [10].

The above effects make a pure Cu-trace unfeasible for accurate current measurement in general. As a possible solution, in [1] we presented a practical method for an accurate, high speed, in-situ resistance-identification of the Cu-trace, especially suitable for PWM-controlled systems.

III. The New Method: CSRTRI (Current Sensing through Real-Time Resistance Identification)

The idea can be illuminated as follows: let us consider a trace of resistance R, and an inductive current sensor without the integrator stage, i.e., a coil, in proximity of the trace.

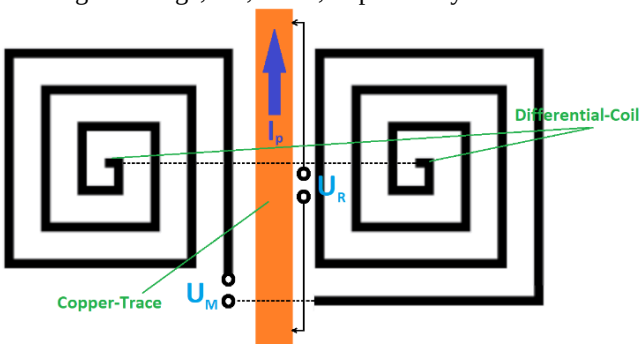


Fig. 2. The new sensing concept.

Let us denote the mutual inductance between the trace and the coil by M. The coil can be arranged symmetrically to the trace, as shown in Fig. 2, so the homogeneous background fields are suppressed. For a current waveform $I = I(t)$ we can write the following equations for the voltage signals:

$$U_R = R \cdot I, \qquad (5)$$

$$U_M = M \cdot \frac{dI}{dt} \equiv M \cdot \dot{I}. \qquad (6)$$

Here U_R denotes the output voltage of the resistive trace, U_M is the induced voltage of the coil, $\dot{I}$ is the time-derivative of the current. Let us take the time derivative of equation (5):

$$\dot{U}_R = \dot{R} \cdot I + R \cdot \dot{I}. \qquad (7)$$

It can be shown, that in practical PWM-controlled systems, where the current waveform includes a current ripple of several kHz frequency, the ratio of the first and the second term in (7) is well under 10^{-3}, so the former is negligible:

$$\dot{U}_R \approx R \cdot \dot{I}. \qquad (8)$$

Excluding the case of the pure, ideal DC-current, i.e., $\dot{I} \neq 0$, we may divide the equation (8) by (6):

$$\frac{\dot{U}_R}{U_M} \approx \frac{R \cdot \dot{I}}{M \cdot \dot{I}} = \frac{R}{M}. \qquad (9)$$

It means, that in effect we can identify the actual resistance value in situ, assuming negligible noise:

$$R(t) \approx M \cdot \frac{\dot{U}_R(t)}{U_M(t)}. \qquad (10)$$

Here, according to [1], the T-dependence of M is negligible. The pure $\dot{U}_R$ and U_M signals differ only in a slowly drifting scaling-factor according to (9), so applying the same bandpass filter to each, leaves their ratio unchanged.

Finally, the actual current value can be determined in real-time, according to (5), as shown by (11) and Fig. 3.

$$I(t) = \frac{U_R(t)}{R(t)} \approx \frac{1}{M} \cdot \frac{U_R(t) \cdot U_M(t)}{\dot{U}_R(t)}. \qquad (11)$$

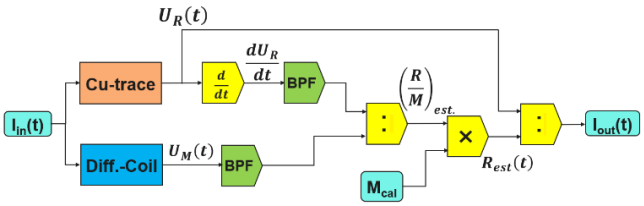


Fig. 3. Flow chart illustrating the CSRTRI-concept.

IV. DESIGN ENHANCEMENT BY FEM SIMULATION

As pre-indicated in chapter II, we expect a significant resistance increase of a simple Cu-trace of dimensions $l = 16$ cm; $w = 16$ mm; $d = 35$ µm due to skin effect already at a few kHz frequency. To quantify the effect, we ran AC-magnetics FEM-simulations of an envisioned single-trace from 1 Hz to 3 MHz. We extracted the $j(x)$ current density distribution across the trace and the $Z = R + j\omega L$ impedance for each frequency value. The simulation results justified the hypothesis suggested by the analytical estimations of the Haefner-curves, specifically, that the frequency-dependence of the resistance is unacceptably high. At 100 kHz for example the R_{AC} value is 18.6% above R_{DC}, which makes an accurate, high bandwidth current measurement impossible. Although the serial inductance remains stable over frequency, its value is relatively large. As is widely known, the low-inductance design can essentially suppress these effects, since the magnetic fields largely cancel each other out in both conductor sections. Therefore, we modified the current trace geometry to the antiparallel one by utilizing both the top- and bottom-Cu layer of the PCB to overcome the above disadvantages. Finally, we repeated the FEM-simulations with the new model geometry, which verified our design choice. The explanation is that at higher frequencies the current density increases on the inner sides of the traces for opposite current directions. Thus, with the low inductance design, at 100 kHz, the R_{AC} value is only 0.84% higher than R_{DC} due to skin effect, so R may be considered frequency independent in equations (5)-(11) with good approximation. Figs. 4 and 5 illustrate the current density maps for 1 mm thick current bars. Fig. 6 depicts $j(x)$, Fig. 7 the R(f) and $L(f)$ curves for the two analyzed PCB-designs. The series equivalent parameters over frequency are summarized in Table I.

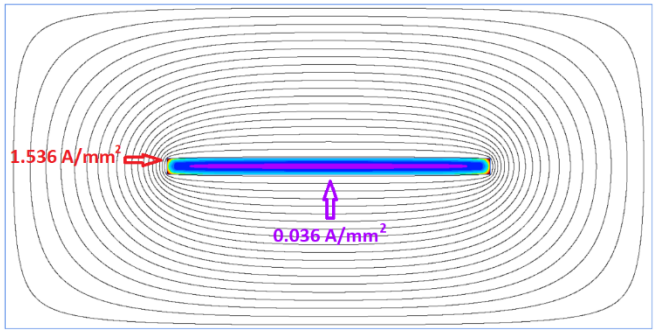


Fig. 4. Current density in a single Cu-bar at $f = 1$ MHz cross-sectional view of 160 mm x 20 mm x 1mm current rail

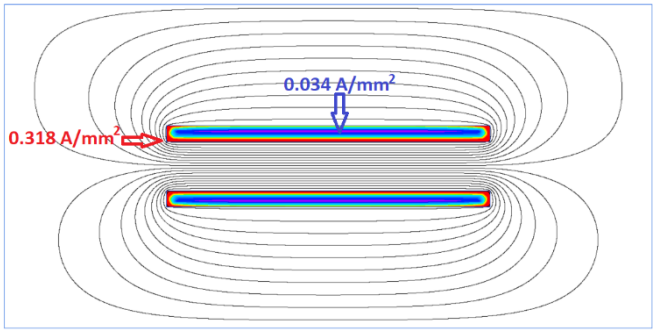


Fig. 5. Current density in antiparallel Cu-bar at $f = 1$ MHz cross-sectional view of 160 mm x 20 mm x 1mm current rails

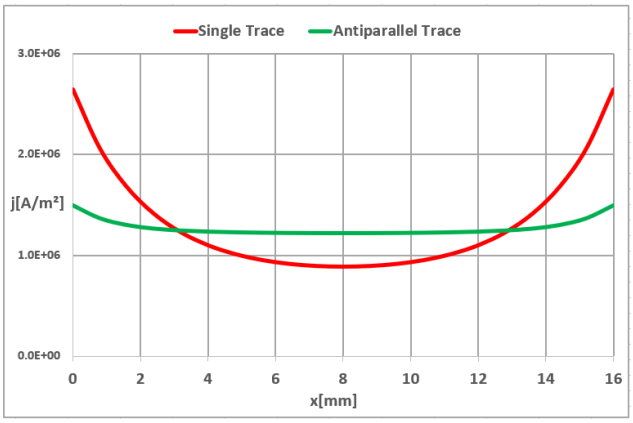


Fig. 6. Current density plots over the two analyzed Cu-traces of 16 mm x 35 µm dimensions at $f = 100$ kHz

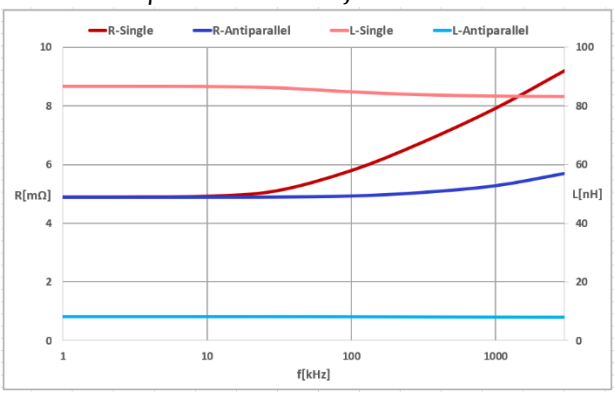


Fig. 7. The series R and L parameters' frequency dependence

To implement the dual sensing-element described in chapter III, we designed a 4-layer, standard PCB with 120 mm x 90 mm x 1.55 mm dimensions, containing the optimized antiparallel Cu-trace formed on the top- and bottom Cu-layers for the resistive signal and a differential-coil, consisting of 2x4 planar, square-shaped spirals, making

use of all the 4 Cu-layers for homogen background-field suppression. The coils are coupled to the 4-layer 8-shaped conductor path (Fig. 8).

TABLE I
COMPARISON OF THE SIMULATED EQUIVALENT SERIES R AND L PARAMETERS OVER FREQUENCY

f [kHz]	R-Single [mOhm]	L-Single [nH]	R-Antiparallel [mOhm]	L-Antiparallel [nH]
0.001	4.884	86.770	4.884	8.246
1	4.884	86.770	4.884	8.247
3	4.887	86.764	4.884	8.247
10	4.912	86.698	4.885	8.246
30	5.093	86.244	4.889	8.243
100	5.790	84.816	4.925	8.212
300	6.722	83.810	5.041	8.142
1000	7.917	83.332	5.277	8.076
3000	9.200	83.166	5.695	8.044

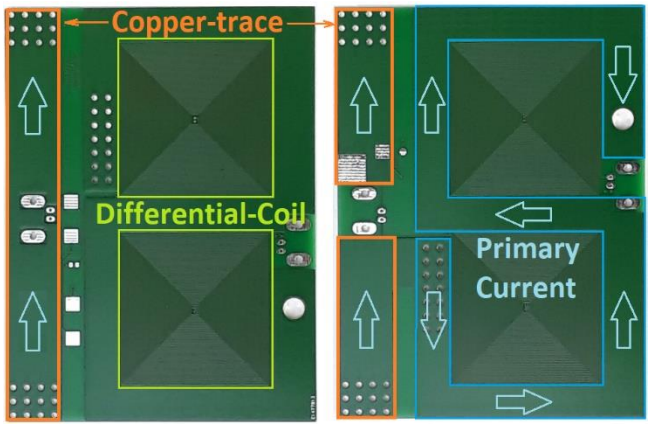


Fig. 8. Sensor-PCB, top and bottom side.

The nominal DC-resistance of the trace is $R_{Cu,25°C} \approx 4.90\ \text{m}\Omega$ considering $\rho_{25°C} = 1.71 \cdot 10^{-8}\ \Omega\text{m}$ Cu-resistivity at 25°C, and the dimensions $l = 160.6\ \text{mm}; w = 16\ \text{mm}; d = 35\ \mu\text{m}$. The analytically estimated mutual inductance value is $M \approx 5.92\ \mu\text{H}$ [1]. Both R and M were determined accurately during system calibration [1].

V. EXPERIMENTAL SETUP

For testing we applied unipolar triangle-like current waveforms up to 150 A peak value and several kHz. We set the amplitude, the offset, and the frequency using an amplifier based current signal generator (Figs. 9 and 10).
The TL081-type amplifier is driven by a Hameg HM-8030-5 function generator in non-inverting topology. The sense-resistor of the current source is a high precision, high power RUG-Z-R010-0.1-TK10-type shunt of $R_{shunt} = 10\ \text{m}\Omega$ value, 0.1% tolerance and better than $\pm 1\ \text{ppm/K}$ TCR from Isabellenhütte [11]. The amplifier forces the shunt's voltage to be equal to the $U_{in}(t)$ through the gate-emitter voltage of the IGBT. The main circuit is powered by a 45 Ah, 12 V Polaris car-battery of 360 A maximal current. The sensor-PCB is placed in series with the shunt.

By measuring the shunt voltage signal directly, the actual output current can be measured with high precision, and used as the reference current waveform in each round:

$$I_{ref}(t) = \frac{U_{shunt}(t)}{R_{shunt}} \approx \frac{U_{in}(t)}{R_{shunt}}. \qquad (12)$$

Similarly to the $U_{shunt}(t)$, the $U_R(t)$, the $U_M(t)$ and the $U_{NTC}(t)$ temperature sensor output were also measured by a PicoScope 3206 PC Oscilloscope with 200 MHz bandwidth from Pico Technology. Knowing the mutual inductance, M from the calibration step, with the CSRTRI algorithm based on equation (11), we could calculate the measured $I(t)$ and compare it against $I_{ref}(t)$.

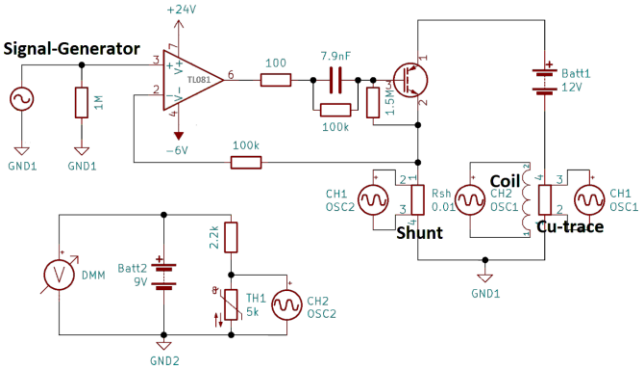


Fig. 9. Schematic of the experimental setup, including current signal generator and sensor-PCB.

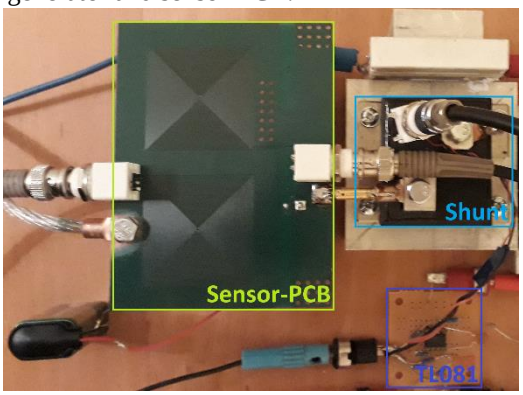


Fig. 10. The sensor-PCB in the test-setup in series with the reference-shunt.

VI. SIGNAL PROCESSING ALGORITHM

We acquired the raw signals: the $U_{shunt}(t)$, $U_R(t)$, and $U_M(t)$ for 5 ms interval and plotted them in Octave (Fig. 11).

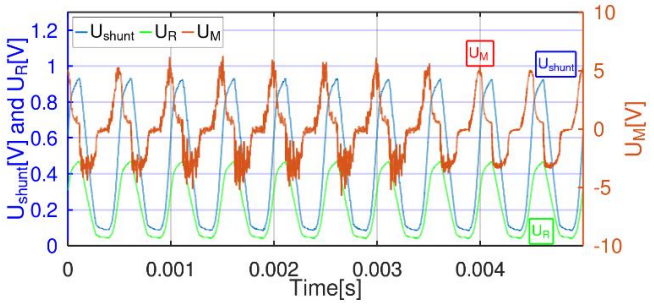


Fig. 11. Acquired raw signal waveforms for 2 kHz triangle-like excitation at $T_{Cu} \approx 50°C$.

We calculated $\dot{U}_R(t)$, the time derivative of $U_R(t)$ using the backward difference-method. Knowing the mutual inductance M, applying (10) would lead directly to the actual resistance value R in a noise-free scenario. Physical and quantization noises are greatly amplified during the process, particularly when calculating the time-derivative and the

quotient. This effect must be suppressed, and to this end we applied noise filtering in two consecutive steps. First a band-selective digital filtering of the $\dot{U}_R(t)$ and $U_M(t)$ waveforms is introduced at the typically known fundamental-frequency of the PWM-signal, before the division, to optimize the SNR of the R-estimate:

$$\hat{R}(t_0) = M \cdot \frac{\dot{U}_R(t_0,\tau) * h_{BP}(t)}{U_M(t_0,\tau) * h_{BP}(t)} = \\ = M \cdot \frac{\{DFT(\dot{U}_R)\}(f)\cdot H_{BP}(f)}{\{DFT(U_M)\}(f)\cdot H_{BP}(f)} = M \cdot \frac{\{DFT(\dot{U}_R)\}(f_0)}{\{DFT(U_M)\}(f_0)}, \quad (13)$$

where * denotes convolution, $\{DFT(U)\}(f)$ denotes the discrete Fourier transform (DFT) of signal $U(t_0,\tau)$. To speed up the algorithm we calculated only the f_0 component of the DFT. The change of R over time was considered in the following way. We calculated $\hat{R}(t)$ from (13) by applying a sliding window and calculate the DFTs on those segments of the signal recursively. To further improve SNR, we applied a median filter to $\hat{R}(t)$, which resulted in $R_{median}(t)$.

In our concrete implementation, at $f_0 = 2$ kHz ripple-frequency, we applied $f_s = 12.5$ MHz sampling frequency and calculated the DFT in $N_s = 6250$ points. The median filter had the window also equal to the triangle-wave's period.

Finally, we calculated the actual $I(t)$ from Ohm's law (14) and compared it against $I_{ref}(t)$. (Figs. 12, 14, 16)

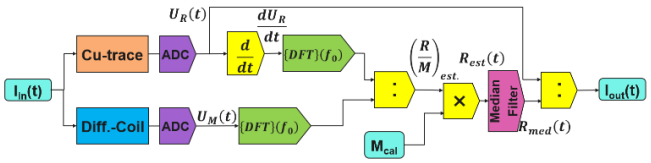


Fig. 12. Flow chart illustrating the CSRTRI implementation.

$$I(t) \approx \hat{I}(t) = \frac{U_R(t)}{R_{median}(t)} \quad (14)$$

Design Considerations: Due to the bandpass-filtering effect of the DFTs in the RTRI-process, electromagnetic disturbances of frequencies other than f_0, affecting both the Cu-trace and the coil potentially, are largely suppressed. By median-filtering the R-estimate, residual noise can be further decreased without significant signal-distortion. It is of paramount importance also, that the parasitic inductance of the $U_R(t)$-sensing wires is minimized [8].

VII. MEASUREMENT RESULTS AND EVALUATION

After the calibration, we conducted a set of measurement rounds, to validate our CSRTRI-method. Around $T_{Cu} \approx 20°C \ldots 25°C$, we excited the system with a unipolar triangle-like current waveform, including both significant AC- and DC-content, and acquired the raw voltage signal waveforms: the $U_{shunt}(t)$ shunt-voltage, the resistive $U_R(t)$ signal of the Cu-trace, and the inductive $U_M(t)$ signal of the differential-coil for 5 ms with 2 kHz fundamental ripple-frequency. The applied I_{AMP}/I_{DC} AC-to-DC ratio was between 60%-100% [1]. I_{AMP} denotes the amplitude of the current ripple. We monitored the PCB-temperature with the onboard T-sensor, i.e., with the NTC-thermistor with $\pm 2°C$ accuracy. As Figs. 13 and 15 show, the corresponding unfiltered identified resistance values at 16.9°C and 62.1°C respectively, have up to $\pm 2\%$ variability due to quantization noise mainly, which can be largely suppressed by median filtering. Figs. 14 and 16 depict the measured $I(t)$ and $I_{ref}(t)$ waveforms and their differences over time, i.e., the current measurement errors on separate panes. The latter ones consist mainly of noise, and has an RMS-value, which is 0.93% ... 1.10% of the whole current signal's RMS-value.

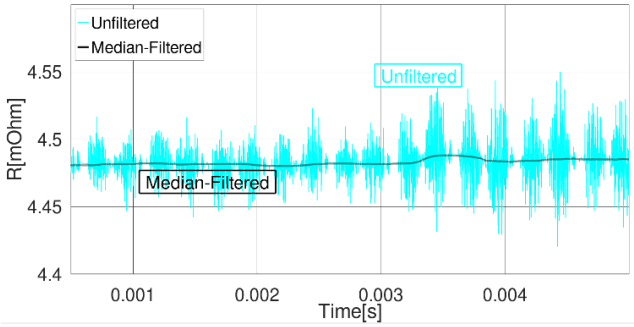


Fig. 13. Resistance Identification of the Cu-trace for 2 kHz triangle-like excitation at $T_{Cu} \approx 17°C$.

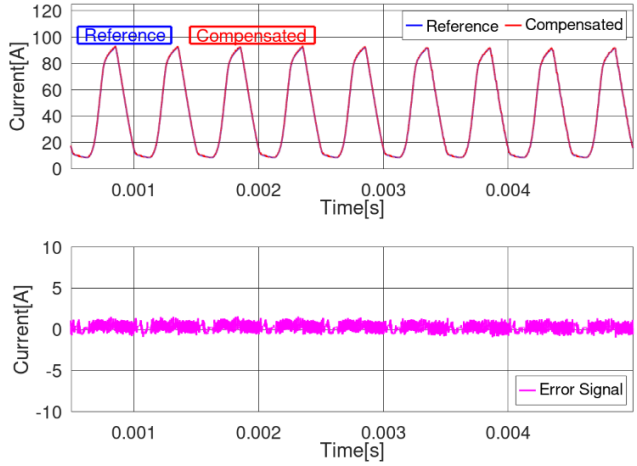


Fig. 14. Validation of CSRTRI for 2 kHz at $T_{Cu} \approx 17°C$.

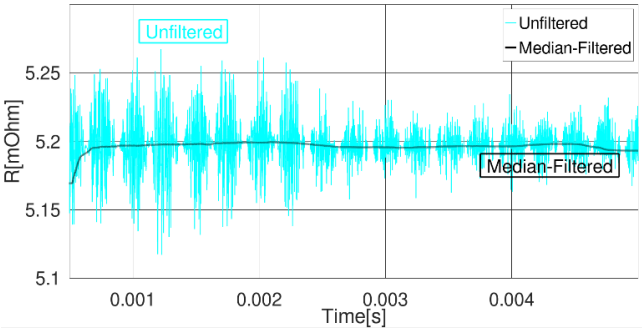


Fig. 15. Resistance Identification of the Cu-trace for 2 kHz triangle-like excitation at $T_{Cu} \approx 62°C$.

In addition to the main shunt-based validation, for completeness, we conducted measurement rounds with $f_0 = 2$ kHz, with controlled and directly measured Cu-trace temperatures, to correlate the RTRI's results, according to (13), with the expectable resistance-values ensuing from the direct temperature measurement results [1]. To that end, we set five different PCB-temperature levels by an IR-lamp. To

minimize the effect of the primary current's self-heating and the resulting temperature-inhomogeneity, we applied current pulses of approximately 100 ms duration. In Table II, we summarized the results.

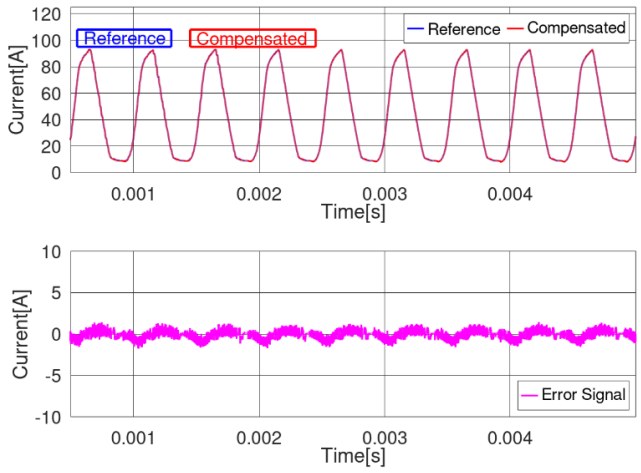


Fig. 16. Validation of CSRTRI for 2 kHz at T_{Cu}≈62°C.

TABLE II
CORRELATING THE IDENTIFIED RESISTANCE VALUES WITH DIRECT T-MEASUREMENTS [1]

WF	f [kHz]	I_{AMP} [A]	I_{DC} [A]	I_{RMS} [A]	ΔI_{RMS} [A]	$\lvert\Delta I_{RMS}/I_{RMS}\rvert$	R_{RTRI} [mΩ]	T_{NTC} [°C]	R_{direct} [mΩ]	$\lvert\Delta R/R\rvert$
#6	2.00	42.2	43.1	53.4	0.52	0.97%	4.483	16.9	4.445	0.85%
#7	2.00	42.4	43.2	53.5	0.51	0.95%	4.696	31.5	4.706	0.21%
#8	2.00	42.3	43.3	53.6	0.59	1.10%	4.869	41.2	4.880	0.23%
#9	2.00	42.5	43.2	53.6	0.50	0.93%	5.048	51.0	5.056	0.15%
#10	2.00	42.5	43.2	53.6	0.53	0.99%	5.197	62.1	5.254	1.10%

VIII. Conclusion

In [1] we proposed an innovative, wideband current sensing method based on real-time resistance identification of a low-inductance Cu-trace, located in the current-path, using an auxiliary coil-system and an efficient band selective signal processing algorithm. The main advantage of this approach is the perspective of replacing the shunts with a current trace or bar, which is already present in the system and easier to manage thermally.

Using AC-magnetics FEM-simulation, we optimized, designed, and built a PCB-based prototype sensor system. Despite the skin effect, due to the antiparallel trace-pair, the resistance value, identified at 2 kHz ripple frequency, could be considered as frequency independent with good approximation, and applied in equations (11) and (14): at 100 kHz we got 0.84% resistance increase, which is acceptably low for the fiftieth harmonic. We calibrated the sensor system and tested its current-sensing accuracy by placing a high precision shunt in series with the PCB for reference measurements: we demonstrated, 0.93% ... 1.10% accuracy with 0.17% uncertainty [1]. The achieved overall accuracy is acceptable, considering that it is roughly up to 15-times lower, than the 14.4% temperature drift of the Cu's resistivity over the 37°C temperature difference between 25°C to 62°C.

Finally, in Table III we briefly compare the CSRTRI under investigation with state-of-the-art current sensing methods, i.e., Hall-sensors, inductive current sensors, and shunts in general.

TABLE III
COMPARISON WITH STATE-OF-THE-ART CURRENT SENSING METHODS [1]

Sensing Method	Accuracy	Bandwidth	DC-Capability	Galvanic Isolation	Insertion Resistance
Hall	good	up to 1 MHz	yes	yes	low or zero
TMR	good	up to 1 MHz	yes	yes	low or zero
Inductive	good	up to 200 MHz	no	yes	zero
Shunts	very good	up to 2.2 GHz	yes	no	significant
CSRTRI	good	2 MHz	yes	no	low or zero

Acknowledgment

T. Dabóczi acknowledges the financial support of Project no. 2019-1.3.1-KK-2019-00004, provided from the National Research, Development, and Innovation Fund of Hungary, financed under the 2019-1.3.1-KK funding scheme, and Project no. RRF-2.3.1-21-2022-00009, titled National Laboratory for Renewable Energy, provided by the Recovery and Resilience Facility of the European Union within the framework of Program Széchenyi Plan Plus.

References

[1] Á. F. Hegedűs and T. Dabóczi, "A Wideband Current Sensing Method Based on Real-Time Resistance Identification" in IEEE Transactions on Instrumentation and Measurement, Accepted on 14.11.2023, doi: 10.1109/TIM.2023.3338684.

[2] W. Zhang, Z. Zhang, F. Wang, E. V. Brush and N. Forcier, "High-Bandwidth Low-Inductance Current Shunt for Wide-Bandgap Devices Dynamic Characterization," in *IEEE Transactions on Power Electronics*, vol. 36, no. 4, pp. 4522-4531, April 2021, doi: 10.1109/TPEL.2020.3026262.

[3] M. Motz et al., "A miniature digital current sensor with differential Hall probes using enhanced chopping techniques and mechanical stress compensation," SENSORS, 2012 IEEE, Taipei, Taiwan, 2012, pp. 1-4, doi: 10.1109/ICSENS.2012.6411161.

[4] Asahi Kasei Microdevices (AKM), "Current Sensor ICs | PRODUCTS | Asahi Kasei" 2021. [Online]. Available: https://www.akm.com/content/dam/documents/products/current-sensor/cq3300/cq3300-en-datasheet.pdf. [Accessed 11 March 2023].

[5] Allegro Microsystems, "ACS370: 1 MHz Bandwidth, Galvanically Isolated Current Sensor," 2021. [Online]. Available: https://www.allegromicro.com/en/Products/Sense/Current-Sensor-ICs/Zero-To-Fifty-Amp-Integrated-Conductor-Sensor-ICs/ACS730. [Accessed 11 March 2023].

[6] Helmuth Lemme, "The Universal Current Sensor" 2023. [Online]. Available: https://www.fierceelectronics.com/components/universal-current-sensor. [Accessed 12 March 2023].

[7] Pearson Electronics, "Wide Band Current Monitors" 2023. [Online]. Available: https://pearsonelectronics.com. [Accessed 12 March 2023].

[8] S. Ziegler, R. C. Woodward, H. H. -C. Iu and L. J. Borle, "Investigation into Static and Dynamic Performance of the Copper Trace Current Sense Method," in IEEE Sensors Journal, vol. 9, no. 7, pp. 782-792, July 2009, doi: 10.1109/JSEN.2009.2021803

[9] Alan Payne, "The AC Resistance of Rectangular Conductors." 2021. [Online]. Available: https://www.researchgate.net/publication/351307928. [Accessed 17 March 2023].

[10] P. Weßkamp, J. Melbert, "High Performance Current Measurement with Low-Cost Shunts by means of Dynamic Error Correction", 18. GMA/ITG-Fachtagung Sensoren und Messsysteme, 2016 January, doi:10.5162/sensoren2016/3.4.4

[11] Distrelec, "RUG-Z-R010-0.1-TK1 - Power Resistor 10mOhm 0.1% 250W, Isabellenhütte" 2023. [Online]. Available: https://www.distrelec.de/en/power-resistor-10mohm-250w-isabellenhuette-rug-r010-tk1/p/16057583 [Accessed 8 April 2023].

Evaluation of a graph distance metric to assess the diversity of timed automata

Rebeka Farkas
Budapest University of Technology and Economics,
Department of Measurement and Information Systems,
Budapest, Hungary
Email: farkasr@mit.bme.hu

***Abstract*—Reliable testing and benchmarking of modelling tools require diverse inputs. However, *diversity* has no precise definition in this context, nor is there a given level of acceptance. While there is one diversity metric [1] that can be used to assess diversity for structural models, there is no such metric for behavioural models, like timed automata.**

In our research, we work on adapting the structural diversity metrics presented in [1] to behavioural models. We evaluate the structural diversity of timed automata by first transforming the models to a unified, structural format and then applying the structural diversity metric.

In this paper, we present a way to adapt the distance metric to timed automata and we apply it to an existing benchmark suite. We evaluate the metric both manually – checking whether models that are similar according to the metric actually show similar behaviours –, and automatically – checking whether verification algorithms perform similarly well/poorly (with respect to other algorithms) on 'similar' models. We show that – despite only considering structural differences – the metric can be useful for finding models with similar behaviours among timed automata.

***Index Terms*—timed automata, distance metric, benchmark**

I. Introduction

Model diversity aims to quantify similarities between models. Diversity metrics (often referred to as distance metrics) can be useful in solving a number of problems, such as finding diverse test cases, similar data points (clustering), or choosing the best design alternative. This paper focuses on diversity as a requirement of reliable benchmarking – specifically in the case of *timed automata*.

A timed automaton [2] is a simple formalism that uses *clock variables* to represent the elapse of time. It can be used to model time-dependent behavior, such as communication protocols with timeouts. Just like other behavioural models, timed automata are often used for formal verification of system behavior, which suffers from state space explosion. In the literature, there are many algorithms and tools for model checking timed automata, but in order to be able to choose the best tool for a given model, reliable benchmarks are necessary.

One of the requirements of reliable benchmarks [3], [4] is *diversity*: in order to be able to analyze a wide range of possible behaviors, the input models have to be diverse. For simple data structures, like numbers and dates, there are methods for choosing diverse inputs (e.g. equivalence partitioning). However, diversity is not well-defined in this context for complex structures like models. While there are various model diversity metrics in the literature, they generally target small differences between two very similar models [5]–[7]. Therefore, they are not suitable for this use case.

On the other hand, the metric presented in [1] has been used to assess the diversity of test suites – among other use cases – but it is defined for *structural models*. To the best of our knowledge, there is no diversity metric for timed automata that can be used to evaluate diversity in benchmark suites.

This paper presents a way to adapt the structural distance metric to timed automata described in Uppaal's XTA format [8]. Before the distance metric is applied, the timed automaton model is transformed into a simpler model that applies abstraction to some of the described behaviour (the expressions) but expands structural features. Due to the nature of behavioural models, the structural distance metric is not expected to find all the similarities and differences between the operation of timed automaton models. However, in this paper, we show that it can find larger, more significant differences and similarities between the types of behaviours shown by two models.

The presented distance metric was applied to a collection of timed automata (the XtaBenchmarkSuite [4]) and its effectiveness in finding similarities and differences between model was checked. First, we manually checked wether the models that are the most similar according to the metric actually demonstrate similar behaviour. Then, in order to evaluate the usefulness of the metric for analyzing benchmark suites, we executed a benchmark over the models and checked whether different algorithms perform similarly (relative to each other) over similar models when exploring the complete state space.

II. Background

A. Timed Automata and extensions

A *timed automaton* [2] extends a finite automaton (state machine) with *clock variables*, which are special variables whose value constantly increases as time elapses. During operation, the values of the clock variables can be checked in guards to enable transitions and updated (*reset*) during state changes. However, the value of a clock variable continues to increase after the reset.

The basic timed automaton formalism can be extended to increase the understandability or the expressive power of the

DOI https://doi.org/10.3311/MINISY2024-002

language. One of the most popular extensions is Uppaal's XTA (extended timed automaton) format [8], that not only allows the definition and usage of *data variables* and *networks of automata*, but also comes with a language that makes the design process very efficient.

Data variables are an extension, where besides clock variables, discrete variables (e.g. integers, bools, etc.) can be defined and used in constraints to enable transitions (*data guards*) and can be modified by transitions (*update*). However, clock variables are not allowed to appear in data guards or updates. Data variables increase the expressive power of the formalism to that of program code, which increases the complexity of model checking. To prevent this, modelling tools often force lower and upper bounds on integers values.

A *network of timed automata* [8] is the parallel composition of a set of timed automaton models. Communication is possible by shared variables or hand-shake synchronization using synchronization channels. Constructing networks of timed automata does not increase expressive power, as a network of timed automata can be transformed into an equivalent timed automaton, but it increases the understandability of the model.

Other than these extensions, the XTA language allows the use of complex structures known from programming languages (such as arrays and function calls), and it provides automaton *templates* that have parameters and can be instantiated (in a so-called *system declaration*) multiple times using different parameter bindings. Moreover, the templates can have local variables which get instantiated along with the automaton and may only be used by the corresponding automaton.

For space considerations, we omit the formal definition and list of model elements of the extended timed automaton formalism. For more information, the interested reader is referred to [8].

B. XtaBencmarkSuite

The XtaBenchmarkSuite [4] is a set of timed automaton models – described using the XTA format – for benchmarking timed automaton verification algorithms. It includes models of 24 systems, including three models with multiple versions (e.g. enhancements and fixes) and six models with parameters determining the number of automata in the network.

The models are categorized according to the types of systems they represent (e.g. circuit, protocol) or the type of the corresponding verification problem (e.g. shortest/fastest path).

C. Distance Metrics and Model Distance

A function $d : S \times S \to \mathbb{R}_{\geq 0}$ over a set S is *distance metric* iff for all $x, y, z \in S$

1) $d(x, x) = 0$,
2) if $x \neq y$, then $d(x, y) > 0$ (*positivity*),
3) $d(x, y) = d(y, x)$ (*symmetry*),
4) $d(x, z) \leq d(x, y) + d(y, z)$ (*subadditivity*).

In practice, for more complex structures, such as graphs *pseudodistances* are used, which are similar to distance metrics, except without the *positivity* criterion.

The pseudo distance metric d_G presented in [1] determines distances between pairs of *labelled graph models*. A labelled graph G is a tuple $\langle V, E, T\rangle$ over a set of labels L, where (V, E) is a directed graph, and $T : V \cup E \to L$ is the *labelling function* assigning labels to the graph elements.

The distance d_G of two graphs are computed using *neighbourhood shapes*: for every vertex v of a graph, the neighbourhood shape $N(v)$ describes the labels of elements within a given range. Afterwards, *shapes* are computed, which characterize graphs by the neighbourhood shapes of its vertices: $S(G)$ is a multiset of neighbourhood descriptors of $V(G)$.

The shapes of the two graphs are encoded into *shape vectors* as arrays of multiplicities, where the dimensions are the neighbourhood shapes of all vertices of both graphs. The distance $0 \leq d \leq 1$ is then computed from the angle α of the two vectors: $d_G(G_1, G_2) = \sqrt{1 - \cos \alpha(\overline{S(G_1)}, \overline{S(G_2)})}$.

D. Kendall Distance for Ranking

The Kendall tau rank distance is a distance metric for rankings that counts the number of pairwise disagreements between two ranking lists. That is, the Kendall distance of two rankings of the same items is the number of pairs of items whose order differs in the two rankings. For example, the Kendall distance d of rankings ***ABCD*** and ***BADC*** is $d = 2$ because the orders of the pair $\langle A, B\rangle$ is different at the two rankings (A comes first in ***ABCD*** but B comes first in ***BADC***) as well as the pair $\langle C, D\rangle$, but for the rest of the pairs $\langle A, C\rangle, \langle A, D\rangle, \langle B, C\rangle, \langle B, D\rangle$ their order is the same.

This paper uses a modified version of this distance that considers rankings where two items can have the same rank. In this metric, the order of two items A and B can be ***AB*** and ***BA*** like before, as well as ***(AB)***, which we use to denote that A and B have the same rank. The orders of a pair can be the same ($d = 0$) and reverse ($d = 2$) like before, but also *semisimilar* ($d = 1$), when the pair has the same rank in one of the rankings but not in the other. As before, the distance of the two rankings is the sum of the pairwise order distances. As an example, the distance of ***ABCD*** and ***BA(CD)*** is $d = 3$ since the order of the pair $\langle A, B\rangle$ is reversed and the orders of $\langle C, D\rangle$ is semisimilar.

III. Computing Diversity Between Timed Automata

In this section we present the way to adapt the structural diversity metric to timed automaton models, by first preprocessing the automata – transforming them into a unified, structural form (a labelled graph) – and then applying the structural diversity metric. Formally, the distance d_m of timed automaton models m_1, m_2 is $d_m(m_1, m_2) = d_G(G(m_1), G(m_2))$, where $G(m)$ is the a labelled graph created from m.

Model transformation aims to create a labelled graph of a unified format, thereby eliminating *purely syntactic* differences. An example of purely syntactic differences is *constant integers*: the XTA format allows the definition of constant integers, which facilitates the design process. However, it also creates different syntax for the same semantics – that is, $c < 5$

and $t = 5 \wedge c < t$ means the same, but the corresponding structural representations (expression trees) are very different.

For similar reasons, structural distance metrics are inapplicable for expressions (e.g. in data guards) and other textual model elements (program code) of extended timed automata. Because of this, the presented distance metric does not consider all possible model elements provided by the XTA format.

A. Target Model

In order to create a unified structural format, the following principles were followed.

- *Constants* (including constant values, constant variables and initial values) are supported but not transformed to the target model – as otherwise, the structural diversity metric would be vulnerable to the exact values.
- *Expressions* in guards, invariants and updates are supported but they aren't transformed. Instead, the referenced variables are identified and stored in the target model.
- The target model only has *simple, uniform elements* – e.g. global variables, local variables and variables of arrays are all transformed into separate, global variables.
- Automaton templates are *instantiated* – that is, for each template in m, $G(m)$ contains as many automata as many times the template is instantiated in the system declaration. This creates the complete structure of the network, with exact connections (through synchronization channels) between the components, allowing the structural diversity metric to explore all the specific details.

Figure 1 shows the metamodel of the target model in which the original XTA models are transformed – that is, the vertex and edge labels and the possible relationships in $G(m)$. (The metamodel is presented in UML format, which has special features, – e.g. multiplicity and containment edges – but they are irrelevant to model transformation.)

The central element of the metamodel is the *XtendedTimedAut* type, which is connected to all the variables (only simple, global variables), automata and communication channels. The automata are connected to their locations and transitions, and the transitions are connected to their source and target locations. Guards of transitions and invariants of locations are sets of *atomic constraints* (that is, the conjunctions are separated in the target model), which are connected to the variables whose value affects the truth value of the constraint. However, the precise expressions are not stored in the target model. Updates are similarly handled, except the updated variable is connected through a different edge type from those whose values affect its new value. Transitions are also connected to the synchronization channels which they use.

The target model also supports many of the XTA-specific elements, such as committed and urgent locations, and broadcast channels (these are additional vertices with specific labels in the labelled graph). However, complex data and control structures (such as enumerations and function calls) are not yet supported. Instead, they have to be manually replaced with equivalent but simpler structures before transformation.

B. Model Transformation

The transformation of the supported elements are as follows. First, the *XtendedTimedAut* vertex of the graph is created, with vertices corresponding to global variables and channels and the connections are established. For each array a of length l, l vertices (with the label corresponding to the type of the array) are created, and an index is assigned (but not denoted in the graph) to each of them. Modifiers, such as integer bounds and broadcast channels, are also handled.

Next, automaton templates are transformed. Automaton vertices are created and connected to the *XtendedTimedAut* vertex – the number i of instances is determined based on the system declaration in the original XTA model. For each local variable, i variable vertices are created and connected to the *XtendedTimedAut* vertex in the target model, and a separate vertex is assigned (but not connected in the graph) to each automaton instance.

The automaton structure (locations and transitions) is already described as a graph in the XTA, but transitions are denoted with edges. In $G(m)$ transitions are also represented by vertices which makes it possible to connect them to vertices representing guards, updates and synchronization. The transition direction is represented by the *source* and *target* edges. All elements of the automaton structure are replicated in the graph i times and additional edges (e.g. from the corresponding *automaton* vertex) are created.

Expressions in constraints and updates are separated into atomic expressions. Then, for each atomic expression, an *atomicConstraint* or *update* vertex is created and replicated i times – each of them connected to the corresponding *transition* or *location* vertex representing an element of a *different* automaton instance. Edges are created pointing towards the graph nodes representing the variables referenced by the expression in the original model. If a referenced variable is a *local* variable, the graph node assigned to the corresponding automaton instance is used. For arrays, the corresponding variable vertex is determined from the array index.

Synchronizations are transformed by adding *send* or a *receive* edges between transition and channel vertices. Channel arrays are often indexed by variables, but usually only one channel vertex is involved in a synchronization. However, sometimes an automaton in the original model represents a central component in the system, sending signals over an array of channels, iteratively changing the index of the array. In this case, the transition in the target model is connected to all corresponding channels.

Once the target model is created, the distance computation of the created graphs can start.

C. Distance Computation

We applied the presented model transformation to the models in the XtaBenchmarkSuite. Three models were excluded due to having unsupported model elements, such as complex data structures and function calls. Distances for the rest were computed.

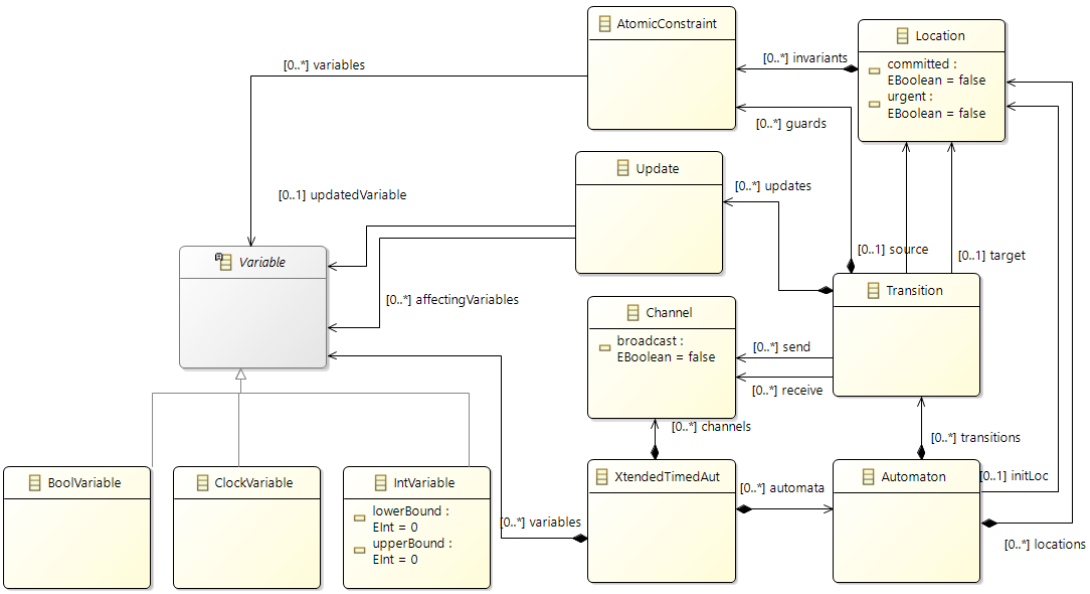


Fig. 1. Metamodel of the target model

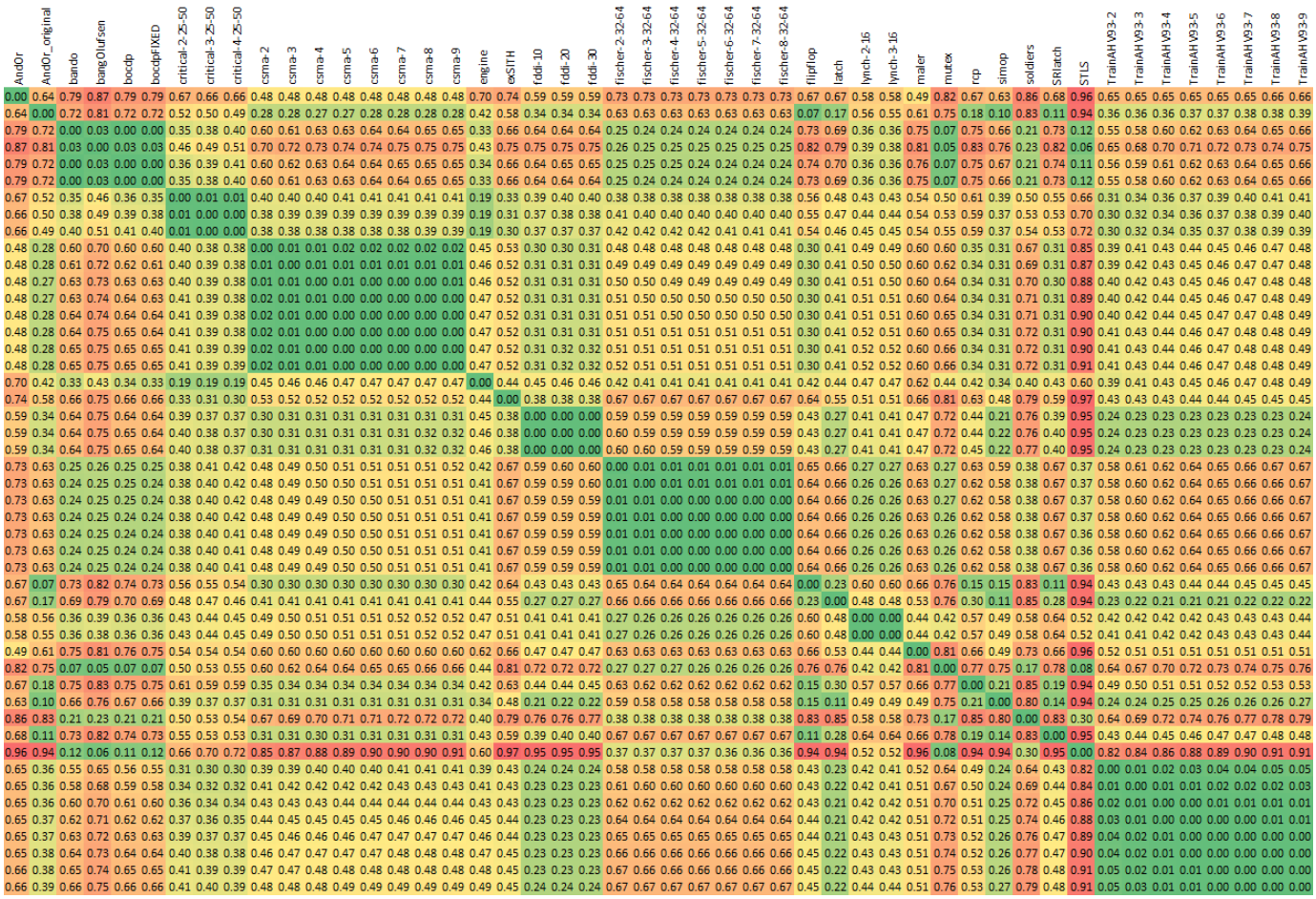


Fig. 2. Table of model distances

Figure 2 depicts a table containing all computed distances (rounded to two decimal places) between all models. The cells are coloured according to the values, ranging from dark green (0 – 0.05) to dark red (0.95 – 1). The horizontal and the vertical order of the models are the same. The dark green cells in and around the diagonal represent that the models are very similar to themselves, as well as different parameterizations or other versions of the same model (e.g *bodcp* and *bodcpFIXED*). The only exception is *AndOr* and *AndOr_original*, where the difference between the two versions is that the data variables in *AndOr_original* are encoded in the locations, which makes the automata structurally very different.

IV. Evaluation

Evaluating the metric is a challenge because there is no existing alternative to compare the results. Instead, we identify the expectations of a useful metric and check the measured distances, both manually and by performing a benchmark.

A. Expectations of Model Distance Metrics

A behavioural model distance metric should be able to find similar behaviours in models: different versions of the same model (e.g. parametrization) are generally expected to be found similar, and – in a loser interpretation of behavioural similarity – models coming from the same domain are expected to demonstrate some similarities. For instance, while the models of the group *circuit* represent different circuit elements, they all demonstrate signal processing delays.

Additionally, in order for a metric to be useful in assessing benchmark suites, it should be able to differentiate between behaviours that model checkers handle differently: model checkers have strengths and weaknesses corresponding to the types of behaviours they can verify efficiently or inefficiently, but they have similar performance over models with similar behaviours. Distance metrics may not necessarily check such behaviours explicitly, but model checkers should perform similarly well or poorly – relative to each other – over models that are similar according to the metric. Formally, let m_1 and m_2 be models and t_1 and t_2 model checkers, and let us denote the performance of a model checker on a model by $p(t, m)$. If m_1 and m_2 are similar and $p(t_1, m_1) < p(t_2, m_1)$ then $p(t_1, m_2) < p(t_2, m_2)$ should hold.

Note: the reverse statement is not necessarily true – efficient model checkers perform well over various, different models.

B. Manual Exploration

We put model pairs from Figure 2 in ascending order and checked whether the corresponding models demonstrate similar behaviours. The results are as follows.

The smallest distances ($d_m < 0.05$) (depicted dark green in Figure 2) are between models by themselves, different versions of the same models, and an additional group: There were four models in the XtaBenchmarkSuite (*bando, bangOlufsen, bodcp* and *bodcpFIXED*) determined similar to each other according to the metric. While the models originate from three different sources, their description are also very similar: Bang & Olufsen (collision detection) protocol, and the process and variable names in the models are also very similar. Therefore, all distances $d_m < 0.05$ according to the metric do point to very similar models.

Increasing the limit to $d_m < 0.2$, similarities are found between the models of the *circuit* group, most non-parametric models of the *protocol* group and some additional findings. *STLS* (short for *Single Track Line Segment*, originally in the *system* group of the benchmark suite) was found to be similar to some models of the *protocol* group. Upon inspection of the model description we have found that the *STLS* model describes a mutual exclusion protocol in the context of railways. Another interesting finding is the similarity between *soldiers* (high school mathematics problem involving a bridge that can only hold at most two soldiers at a time) and *mutex* (a mutual exclusion protocol), since the bridge can be considered as a special type of critical section that allows two participants.

However, there are findings where the explanations are not as straightforward. Some models of the *circuit* group were found to be similar to *simop* – which comes from the same source but describes a networked automation system – and *rcp* – which is a model of a root connection protocol. Additionally, *engine* – a running engine — is similar to *critical* – a mutual exclusion protocol, according to the metric.

Increasing the limit to $d_m < 0.3$, some additional similarities between models of the same group (*protocol* or *circuit*) are discovered but the unexpected similarities also increase, e.g. between *circuit* models and *protocols*.

Since the largest possible distance is $d_m = 1$, we do not consider distances $d_m > 0.3$ of any particular interest.

In conclusion, the metric is able to find similarities between different versions of the same models, models from the same domain and even some hidden similarities (e.g. *soldiers* and *mutex*). On the other hand, finding similar behaviours between some of the pairs that are similar according to the metric is not straightforward. Moreover, some models that were expected to be similar (e.g. different parametric mutual exclusion protocols) were not found to be similar by the metric.

C. Model distance and performance rankings

In order to evaluate the corresponding property on the presented metric, we executed a benchmark on different configurations (*LU* [9], *FWITP, BWITP* [10]) of the timed automaton verification algorithm in the Theta [11] framework and we compared the performance rankings of the configurations for all pairs of models where at least one of the configurations was able to explore the complete state space within the time limit of 10 minutes.

The algorithms *FWITP* and *BWITP* are improvements of *LU* which use interpolants over data variables. Due to the similar nature of the Theta algorithms, they often explore the state space of a given model in the exact same way. Because of this, when comparing the performances, the size of the explored state space (i.e. abstract reachability graph) is considered instead of the runtime. Because of this, different algorithm configurations for a given model may have the same

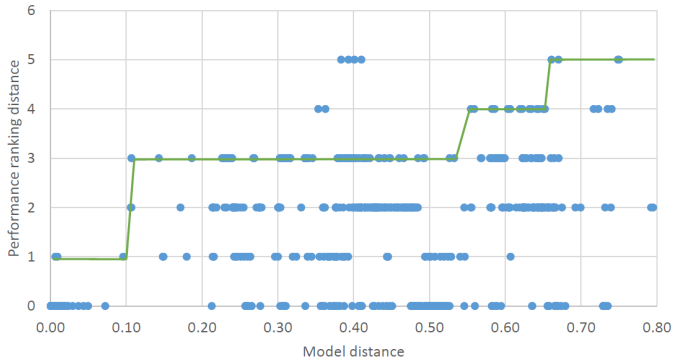


Fig. 3. Model and performance rank distances

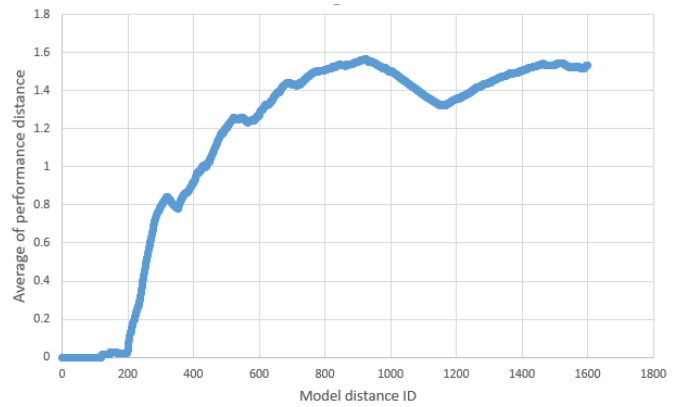


Fig. 4. Average performance distance over number of ascending model distances

performance rank. Therefore, we use the modified Kendall distance d_p to compute the distances of the performance ranks.

Figure 3 depicts a scatterplot showing the model distance and the performance ranking distances for all pairs of models. The green line depicts that aside from a small cluster at model distance $d_m \approx 0.4$ where there are some data points where the performance ranking distance $d_p > 3$, the range of the performance distance is growing with the model distance: for $d_m < 0.1 \rightarrow d_p \leq 1$, for $d_m < 0.55 \rightarrow d_p \leq 3$ (aside from a few data points), for $d_m < 0.65 \rightarrow d_p \leq 4$, for $d_m > 0.65 \rightarrow d_p \leq 5$. The data shows correlation between the two distances, which meets the (one-sided) expectation about tools performing similarly over similar models.

Figure 3 depicts the change in the average value of performance distance as we add the model pairs in ascending order of model distance. The figure shows that the average almost continuously increases for the first ≈ 900 pairs, which shows that – until that point – not only the range of the performance distances increase with the increasing model distances, but the larger performance distances also appear in a larger rate.

After the increasing slope, the average starts descending, which is not the expected behavior. However, almost all of the corresponding models are parametric models of different protocols, which coincides with the previous observation about the presented metric not recognizing the similarities between such models. This weakness can be caused by the metric focusing on structural similarity.

Aside from a few descending parts, the almost continuously increasing slope of average Kendall distances shows that – while there are models with large model distances and small Kendall distances – in general, larger model distances are expected to yield larger Kendall distances.

V. Conclusions and Future Work

The presented timed automaton distance metric is able to find similarities between different versions and different parametrizations of the same model, as well as models of the same domain. It has also discovered hidden similarities between the behaviours of different models. Furthermore, the presented model distance correlates with the performance ranking distance of algorithms of the Theta verification framework. On the other hand, the presented distance metric was ineffective in finding similarities among parameterized models of different protocols.

In the future, we plan to support more XTA model elements. We also plan to improve the presented metric by including behavioural information, and we will perform a more exhaustive benchmark with a wider range of verification algorithms. We are also going to create metrics for other types of behavioural models, such as Petri Nets.

Acknowledgment

The author wishes to thank Vince Molnár and Mihály Dobos-Kovács for their help.

References

[1] O. Semeráth, R. Farkas, G. Bergmann, and D. Varró, "Diversity of graph models and graph generators in mutation testing," *STTT*, vol. 22, no. 1, pp. 57–78, 2020.

[2] R. Alur and D. L. Dill, "The theory of timed automata," in *Real-Time: Theory in Practice, REX Workshop, Mook, The Netherlands, June 3-7, 1991, Proceedings*, 1991, pp. 45–73.

[3] J. Gray, Ed., *The Benchmark Handbook for Database and Transaction Systems (2nd Edition)*. Morgan Kaufmann, 1993.

[4] R. Farkas and G. Bergmann, "Towards reliable benchmarks of timed automata," in *Proceedings of the 25th PhD Minisymposium of the Department of Measurement and Information Systems*. Budapest University of Technology and Economics, 2018, pp. 20–23.

[5] T. A. Henzinger, R. Majumdar, and V. S. Prabhu, "Quantifying similarities between timed systems," in *International Conference on Formal Modeling and Analysis of Timed Systems*. Springer, 2005, pp. 226–241.

[6] P. Černý, T. A. Henzinger, and A. Radhakrishna, "Simulation distances," *Theoretical Computer Science*, vol. 413, no. 1, pp. 21–35, 2012.

[7] U. Fahrenberg, C. Thrane, and K. Larsen, "Distances for weighted transition systems: Games and properties," *Electronic Proceedings in Theoretical Computer Science*, vol. 57, 07 2011.

[8] J. Bengtsson, K. G. Larsen, F. Larsson, P. Pettersson, and W. Yi, "UPPAAL - a tool suite for automatic verification of real-time systems," in *Hybrid Systems III: Verification and Control, Proceedings of the DIMACS/SYCON Workshop on Verification and Control of Hybrid Systems, October 22-25, 1995, Ruttgers University, New Brunswick, NJ, USA*, 1995, pp. 232–243.

[9] F. Herbreteau, B. Srivathsan, and I. Walukiewicz, "Lazy abstractions for timed automata," in *CAV 2013*, 2013, pp. 990–1005.

[10] T. Tóth and I. Majzik, "Lazy reachability checking for timed automata with discrete variables," in *SPIN 2018*. Springer, 2018, pp. 235–254.

[11] T. Tóth, A. Hajdu, A. Vörös, Z. Micskei, and I. Majzik, "Theta: a framework for abstraction refinement-based model checking," in *Proceedings of the 17th Conference on Formal Methods in Computer-Aided Design*, D. Stewart and G. Weissenbacher, Eds., 2017, pp. 176–179.

Towards pulmonary vessel separation

1st Gábor Révy
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
0000-0002-4547-3923

2nd Anna Bodnár
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
anna.bodnar@edu.bme.hu

3rd Dániel Hadházi
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
0000-0002-6233-5530

4th Gábor Hullám
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
0000-0002-4765-2351

***Abstract*—Pulmonary artery-vein segmentation and separation plays an important role by guiding doctors' surgical planning. However, automation of this process has not yet been fully achieved. There are plenty of studies that achieve fairly accurate results. However, most of the available algorithms rely on high-quality CT scans, which are not always available in everyday practice. High quality for CT means (1) contrast-enhanced CT scans with (2) well-timed contrast material injection, (3) thin slice thickness, and (4) the application of filtering during the reconstruction to reduce the amount of streaking artifacts.**

In this work, we present 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. Furthermore, their results are investigated, discussing their strengths and current difficulties.

***Index Terms*—pulmonary vessel separation, ct, image processing**

I. Introduction

CT-based computer-aided pre-operative surgical planning tools are becoming increasingly popular. An important part of the planning is the localisation of organs and body parts, i.e. voxel-level labelling on the CT scan. To plan pulmonary surgery, it may be necessary to segment blood vessels in the lungs, and even to separate arterial and venous networks for some types of surgery. There exist numerous studies on the algorithmization of separation, but they suffer from real-world circumstances. The main requirement for these algorithms to work properly is a thin slice thickness. Based on our experience, these algorithms do not work as intended for CT scans with a slice spacing of 2.5 mm. Another complicating factor is the lack of contrast material, or even if the scan is contrast-enhanced, artifacts can be present due to the inappropriate timing and the use of inappropriate filters during reconstruction. During our work we aimed to improve a segmentation-separation algorithm that is working reasonably well even in real-life conditions.

This research was funded by the Josef Heim Medical Innovation Scholarship (Josef Heim Alkotói Ösztöndíj).
This research was supported by the National Research, Development, and Innovation Fund of Hungary under Grant TKP2021-EGA-02, and the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.

II. Related work

A. Prior work

There are several approaches for pulmonary vessel segmentation and separation. Payer [1] uses an optimally oriented flux-based [2] multi-scale tubularity filter to enhance tubular structures for vessel extraction. In the next step a local maxima graph is created, in which local maxima (vertices) close to each other are connected. The connections are represented by a 4D path, which in addition to the coordinates also includes the radius of the vessel. Finally, two integer programs were proposed (1) for the extraction of disjoint subtrees and (2) for the labeling of the subtrees. The algorithm uses different descriptors to distinguish between artery and vein, *e.g.*, a so-called "arterialness measure", which leverages the anatomical property that arteries, unlike veins, typically run closely and parallel the bronchi. His method achieved a mean Dice score of 0.941 with a range of 0.85 to 0.987. However, it is worth noting that their validation dataset consisted of images with a slice thickness of ~ 0.6 mm.

Nardelli *et al.* [3] designed a neural network-based solution. Their algorithm consists of three main steps. First, vessel candidates are extracted using a Frangi filter [4]. Then, patches around the vessel of interest are classified by a CNN. They investigated the use of different types of neural networks and input types: five different CNNs processing 2D and 3D patches with and without additional local information. This initial local classification is then refined by a graph cut-based optimization method. Their algorithm reached an accuracy of 95.3 ± 2.3 (mean ± std) with a sensitivity of 95.8 ± 3.1 and specificity of 94.5 ± 2.2 on mild COPD cases. They used CT scans with a voxel size of $0.6 - 0.75$ mm for their study.

The algorithm by Charbonnier *et al.* [5] uses anatomical knowledge for the classification. After segmenting the vessels, a centerline representation is created, from which a geometric graph is constructed. In this graph the arterial and venous

vessels may appear to intersect, thus a pruning algorithm is applied. This results in subtrees that must be classified as artery or vein. Finally, these subtrees are connected and classified (1) on the basis of the classification of individual components and (2) using the anatomical knowledge that venous vascular networks have a larger volume than arterial ones. They evaluated their algortihm on CT scans with a slice thickness of 1.0 mm. Their system achieved an accuracy of 92%.

B. Multiscale topomorphologic opening

Since a large part of this work is based on the multiscale topomorphologic opening (MSTMO) algorithm designed by Saha *et al*. [6], first, an outline is given of how the algorithm works. The semiautomatic algorithm was designed to be able to reduce the impact of partial volume effect. This effect results in a blurred border between vessels running close to each other. Based purely on the Hounsfield (HU) values, the border cannot be determined. The strength of this algorithm, is that it is able to produce a fairly good vessel separation even on CT scans with large slice thickness. Furthermore, with a little modification, the result can be manually corrected using the same algorithm. The algorithm consists of three major steps, from which the second two is repeated until no change is detected in the separation. The first step is the (1) calculation of the *fuzzy distance transform*. The two alternating steps are the (2) calculation of *fuzzy morphoconnectivity strength* and (3) *morphological reconstruction*. The algorithm requires at least two seed points: one each in the arterial and venous networks.

1) Fuzzy distance transform: Conventional distance transform (DT) is calculated on a binary mask and returns the distance to the closest backgound (*i.e*. non-object) voxel for each voxel. This value equals to the radius of the minimal size spherical kernel at each point of the object that, when eroding the figure, makes the point belong to the background. If the mask is not binary, but each voxel has a continuous membership value ($\mu_O : \mathbb{Z}^3 \rightarrow [0,1]$), it is called fuzzy distance transform (FDT). The fuzzy distance between two neighboring (p and q) voxels is defined as:

$$\frac{1}{2}(\mu_O(p) + \mu_O(q))\|p - q\|,$$

where $\|.\|$ denotes the Euclidean distance. Then, the distance ω_O between any two points is defined as the total distance Π_O of the shortest path between them:

$$\Pi_O(\pi) = \sum_{i=1}^{N-1} \frac{1}{2}(\mu_O(p_{i+1}) + \mu_O(p_i))\|p_{i+1} - p_i\|$$
$$\omega_O(p,q) = \min_{\pi \in P(p,q)} \Pi_O(\pi),$$

where $P(p,q)$ contains all paths between p and q.

The value of the FDT Ω_O in a given voxel p is the distance ω_O from the nearest background (non-object) point to the voxel:

$$\Omega_O(p) = \min_{q \in \overline{\Theta(O)}} \omega_O(p,q) \quad (1)$$

where $\Theta(O) = \{p | \mu_O(p) > 0\}$ denotes the object-voxels and $\overline{\Theta(O)}$ the non-object voxels. The FDT can be performed by a dynamic programming-based algorithm described by Saha *et al*. [7].

2) Fuzzy morphoconnectivity strength: The first of the two alternating steps in the iterative algorithm is the calculation of the fuzzy morphoconnectivity strengths (FMS). The FMS of a path is defined as the minimum FDT value along the path:

$$\Gamma_O(\pi) = \min_{p \in \pi} \Omega_O(p)$$

The FMS between any two points is defined as the maximum FMS of the paths between the two points:

$$\gamma_O(p,q) = \max_{\pi \in P(p,q)} \Gamma_O(\pi)$$

The morphoconnectivity strength value (in case of Euclidean distance transform) between 2 points can therefore be thought of as the minimum radius of a sphere with which, if erosion is performed, there will be no path between the two points passing through solely object-voxels.

The idea behind the labeling algorithm is that FMS assigns two values to each voxels: the connectivity strength to the arterial and the venal seed points. To whichever the voxel is stronger connected, it is labeled accordingly (R_A – labeled as artery, R_V – labeled as vein, S_A – artery seed points, S_V – vein seed points):

$$R_A = \left\{ p \middle| \max_{a \in S_A} \gamma_A(a,p) > \max_{v \in S_V} \gamma_V(v,p) \right\}$$
$$R_V = \left\{ p \middle| \max_{v \in S_V} \gamma_V(v,p) > \max_{a \in S_A} \gamma_A(a,p) \right\}$$

3) Morphological reconstruction: Points with an FDT value lower than the FDT value in the "narrowest cross-section" leading to the seed points cannot be classified in the previous step, because the connectivity strengths from the starting points of the two classes will be the same at these points. These points are usually located on boundary of the object and at the border between two different types of vessel. Thus another step is required to classify these object boundary points as well.

The morphological environment of a class is the set of points from which a point of that class can be reached via monotonically increasing FDT values:

$$N_O(R_O) = \{p | \exists q \in R_O : \omega_O(p,q) < \Omega_O(q) \wedge \exists \pi \in P(p,q) \text{ path of increasing FDT values}\}$$

During the reconstruction, R_A will be "extended" with the points that fall within its morphological environment and are

closer to R_A than to R_V, and the same procedure is followed for the extension of R_V:

$$M_O(R_A) = R_A \cup \Big\{ p | p \in N_O(R_A) \wedge \big(\min_{q \in R_A} \omega_A(p,q) < \min_{q \in R_V} \omega_V(p,q) \big) \Big\}$$

$$M_O(R_V) = R_V \cup \Big\{ p | p \in N_O(R_V) \wedge \big(\min_{q \in R_V} \omega_V(p,q) < \min_{q \in R_A} \omega_A(p,q) \big) \Big\}$$

These additional points are not explicitly labeled by the algorithm, but used to modify the FDT values. The other purpose of the modification of FDT values is to prevent the algorithm from relabeling points that have already been labeled. To do this, the FDT values of all points already classified or reconstructed in another class must be zeroed out:

$$\Omega_A(p) = \begin{cases} 0 & \text{if } p \in N_O(R_V) - M_O(R_A) \\ \Omega_O(p) & \end{cases}$$

$$\Omega_V(p) = \begin{cases} 0 & \text{if } p \in N_O(R_A) - M_O(R_V) \\ \Omega_O(p) & \end{cases}$$

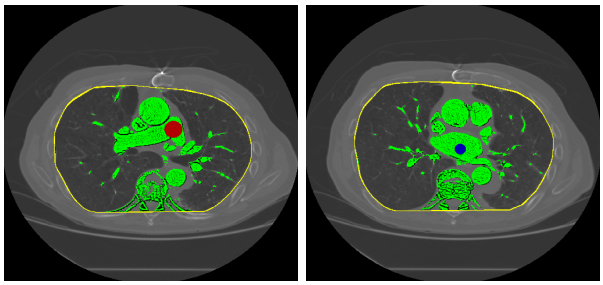
The classified points (R_A and R_V) are used as seed points (S_A and S_V) in the next iteration. The algorithm continues with the FMS calculation and repeats until there is change.

III. Methods

In this section the proposed improvements and examined algorithms are described.

A. Manual seed point selection

The algorithm is very sensitive to the selection of the seed points. If a seed point with low FDT value is chosen the final segmentation may be ruined. This follows from the voxel selection property of the labeling algorithm: suppose that only a single voxel with low FDT ($\Omega_O(p) < K$) is labeled as artery. Then, by definition $\gamma_A(p,q) < K$ for all voxels of the volume. In this case only those voxels are labeled as artery, that can be reached from that seed point with a γ_O value lower than this low K value. This results in only a fraction of artery voxels being labeled as artery and the rest as vein. However, it cannot be expected from the user to know the FDT values of the voxels to be labeled. Therefore, a GUI was implemented for the seed point selection step. This UI is shown in Figure 1. If the user moves the mouse pointer over a voxel on the axial slice, a sphere with the same radius as the DT value at that point appears at the cursor position (considering the anisotropic resolution). This way, the user is only required to seek the sphere with the largest radius in the vicinity of the voxel to be selected. To make this step more robust, not only the center voxel is labeled but all the voxels covered by the sphere. Since we have tried to minimize manual interaction, two points must be selected: one in the bifurcation of the pulmonary artery (as artery) and another in the left atrium (as vein). Of course, the more points are given, the better the result the user will obtain.



(a) Arterial seed point selection (b) Venous seed point selection

Fig. 1. GUI during the selection of the seed points: a sphere of radius corresponding to the DT value of the given voxel appears in the cursor position.

B. Correction of the segmentation

The original algorithm does not provide a solution to correct the separation result afterwards. But re-running with new seed points would be time consuming. Therefore, a correction algorithm was designed using as many of the already well-labelled points as possible.

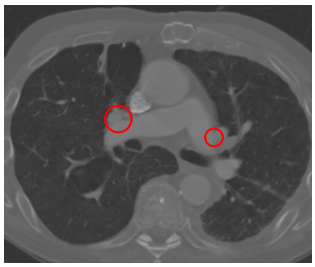
Since the label of a voxel already labeled does not change later, for each labeled voxel the iteration when it was labeled can be recorded (iteration index). During the corrections phase the same seed point selection method is used as in the initial selection. After labeling the voxels from which points the segmentation is mislabeled, the lowest iteration number of voxels covered by the sphere is chosen. Voxels with a smaller iteration index are extended by the newly annotated voxels and are used as seed points for the MSTMO algorithm. This way the number of voxels to be labeled is greatly reduced.

C. Enhancing borders between vessels

The boundary between side-by-side running blood vessels is often difficult to establish due to the so-called "partial volume" effect and the low intra-slice resolution as it can be seen in Figure 2a. If, however, a small difference in density is observed at the boundary between the two, this can be amplified to derive the density-based membership values with a low membership value at the boundary. The boundaries are emphasized by one-sided LoG (Laplacian of Gaussian) filtering:

$$I_{emph} = I - \alpha \cdot \max\{\text{LoG}(I), 0\} \tag{2}$$

Here, the positive and negative "valleys" are enhanced by the LoG filter, and the image is modified by a positive scalar (α) multiple of the *negative* direction valleys (positive values after LoG filtering). Based on this, global thresholds can now be used to scale the enhanced HU values between 0 and 1 to obtain membership (μ_O) values within the heart and lungs as shown in Figure 2b. Both filtering with 3D and 2D LoG filters was examined. We found, that with the typically large slice thickness, filtering in the superior-inferior direction might degrade the quality of the enhancement, thus we opted for 2D filtering.



(a) Original axial slice, with blurred boundaries marked in red.

(b) Corresponding membership values, with boundaries enhanced using a one-sided LoG filter.

Fig. 2. Effect of applying a one-sided LoG filter on the image to emphasize blurred borders between the vessels.

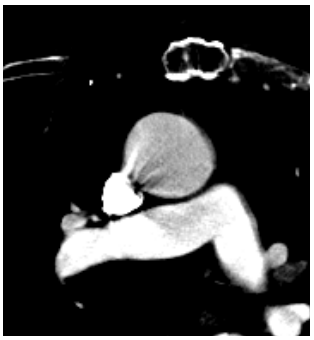
D. Reducing the impact of streaking artifact

Improper timing and the use of inappropriate reconstruction algorithms often lead to streaking artifacts in contrast enhanced CT scans. An example for streaking artifact is shown in Figure 3a. Since the membership (μ_O) values are calculated based on the densities, this phenomenon often causes problems near the pulmonary artery bifurcation, next to the superior vena cava, where the contrast agent usually accumulates. Namely, the streaking artifact causes radial streaks to appear with variations in density and the high density area as center. This results in spoiling the calculation of the FDT values. The algorithm, partially mitigating this effect, is described in the followings.

First, contrast agent accumulation is detected (*i.e.* high-density areas) in the heart. This is obtained by applying thresholding to the HU values inside a simple whole heart segmentation mask on the axial slices. Then, the center of the area is determined by taking the median of its points' coordinates. This is marked as the focus point of the rays. In addition, linear structures are detected using the Frangi filter [4] (see Figure 3c). The result of this filtering method is a "ridgeness" map, which has values between 0 and 1. This map is further processed in polar coordinate representation with the previously determined focus point as center. The polar representation is thresholded to determine "ridge" and "non-ridge" voxels. From this center point rays are cast and the number of non-ridge voxels is accumulated along the rays. The ray stops, when a given number of non-ridge voxels is reached. This way streaks, starting from the center point can be filtered and their impact reduced (shown in Figure 3d). Although this algorithm was able to reduce the effect of the streaking artifact slightly, it did not achieve a significant improvement.

E. 2D-3D FDT

Originally, the FDT values are calculated in 3D. However, with the typical large slice thickness, due to the partial volume effect, the border between vessels on different slices is usually not visible, thus the membership value is not low. This usually results in a segmentation leakage between the two vessel types. Furthermore, after placing new seed points for correction,



(a) CT scan with streaking artifact

(b) Result of Frangi filter applied to the axial slice

(c) Artifact detected by the algorithm

(d) Ridges after reducing the impact of artifacts

Fig. 3. Main steps of the algorithm reducing the effect of streaking artifact.

there is no guarantee, that the seed points will not cause leakages. FDT is calculated by using the physical dimensions of the voxels (pixel spacing and slice thickness). By increasing the slice thickness dimension, the label propagation in the superior-inferior direction can be regulated. Although, different levels of dimensional change have been examined, 2D FDT calculation has been chosen. This way the algorithm prefers the propagation of the labels inside the axial slices and the propagation between the axial slices is greatly reduced. This approach greatly helped by reducing the errors resulting from between-slice leakages.

F. FDT recalculation

After the morphological reconstruction step (described in Section II-B3) the FDT values of object *A* are zeroed out at the voxels labeled as object *B* and vice versa. This, however results in an FDT map inconsistent with the current state of the algorithm, since the FDT values in the vicinity of the newly labeled voxels are incorrect. The original paper does not mention the recalculation of the FDT values. The dynamic programming based algorithm allows the restricted recalculation of the FDT values by applying the modification only from the zeroed out points.

IV. Evaluation

The segmentation and separation algorithm was evaluated on a private dataset containing pulmonary chest CT scans.

The difficulty of the precise and expressive evaluation lies in the lack of properly annotated dataset and a sufficiently descriptive metric. In our dataset only the pulmonary arterial vessels are annotated. As a result, not only the separation but also the segmentation of the vessels contributes to the computed accuracy.

For example, in the manual segmentation, voxels with significantly low HU value (well below -700 HU) are segmented as artery on the edge of the vessel. This makes the manual segmentation smooth but inaccurate. These voxels, however, are not segmented as vessel by the semiautomatic algorithm, which is based on density thresholding.

It is also worth pointing out that there are parts of the heart that are not segmented as part of the arterial network in the dataset, but are directly connected to it. For example the right ventricle is directly connected to the pulmonary artery bifurcation but is not labeled as artery in the dataset. This results in an oversegmentation heavily penalized by the generally applied evaluation metrics, such as intersection over union (*iou*) and Dice similarity coefficient (*dsc*). The initial segmentation reached a mean *iou* and *dsc* of 0.2368 and 0.3764 respectively. This result would indicate a very low accuracy, however the results are influenced by the circumstances described above.

During our research we have endeavored to minimize the need for human intervention. This is why two seed points were opted for the initial segmentation. Being a semiautomatic algorithm, this result can be improved to a seemingly near perfect vessel separation by using only a few correction seed points. This was tested on one CT scan and resulted in an improvement from 0.2431 (*iou*) and 0.3911 (*dsc*) to 0.5349 and (*iou*) 0.697 (*dsc*).

Subfigures 4c (initial result) and 4b (result after correction) show examples for segmentations, where the initial segmentation required manual correction. Some of the errors are highlighted with green arrows. The error e_1 is clearly caused by the partial volume effect: the vessels cross each other seemingly on the same slice, thus the segmentation from the arterial vessel network can propagate to the venous network. The errors e_2 and e_3 are caused by the same common mistake of the algorithm: the left superior pulmonary vein coming from the left atrium is bypassing the left pulmonary artery from inferior direction. In most cases the boundary between the two is not visible thus the venous vessel is mislabeled as arterial vessel. This is represented by the error e_5. This error propagating further results in e_2, e_3 and e_6. With one or two properly placed correction seed points, this can be corrected. Two might be necessary not to mislabel the arterial vessels already correctly labeled.

V. Conclusion

In this paper, we have proposed improvements and extensions to the fuzzy distance transform (FDT) based pulmonary artery-vein separation algorithm called multiscale tomomorphologic opening introduced by Saha *et al*. [6]. The algorithm uses a thoracic CT scan and two manually defined (one arterial and one venous) seed points as its input. The current algorithm is explicit model-free, however, in order to improve its initial results and further minimize the required user interactions a model of some kind would be required. The output of the algorithm can be manually improved by the user, however it requires a sufficient level of medical knowledge. Feedback from medical professionals suggests that this is a useful tool, which in its current form is part of a pre-operative medical software.

References

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

[2] M. W. Law and A. C. Chung, "Three dimensional curvilinear structure detection using optimally oriented flux," in *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part IV 10*. Springer, 2008, pp. 368–382.

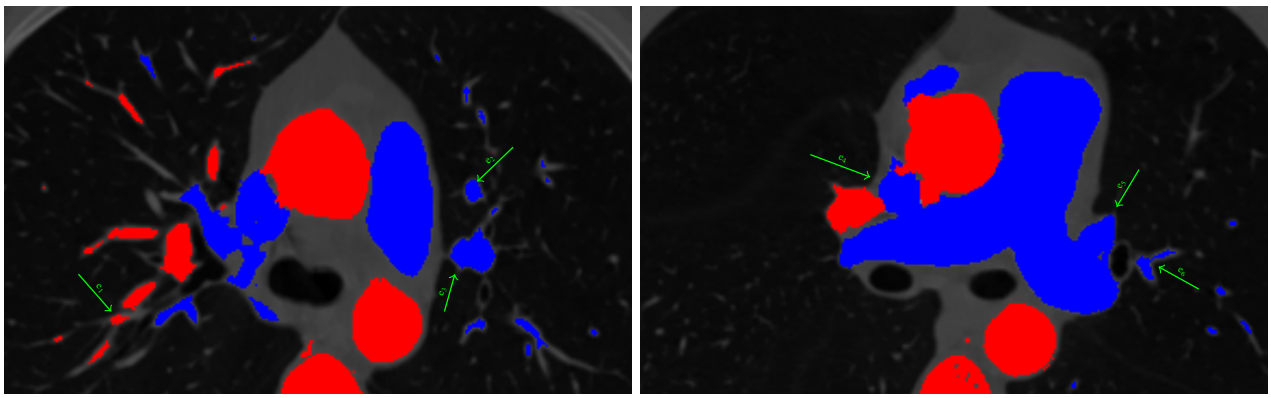
[3] P. Nardelli, D. Jimenez-Carretero, D. Bermejo-Pelaez, G. R. Washko, F. N. Rahaghi, M. J. Ledesma-Carbayo, and R. S. J. Estépar, "Pulmonary artery–vein classification in ct images using deep learning," *IEEE transactions on medical imaging*, vol. 37, no. 11, pp. 2428–2440, 2018.

[4] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever, "Multiscale vessel enhancement filtering," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI'98: First International Conference Cambridge, MA, USA, October 11–13, 1998 Proceedings 1*. Springer, 1998, pp. 130–137.

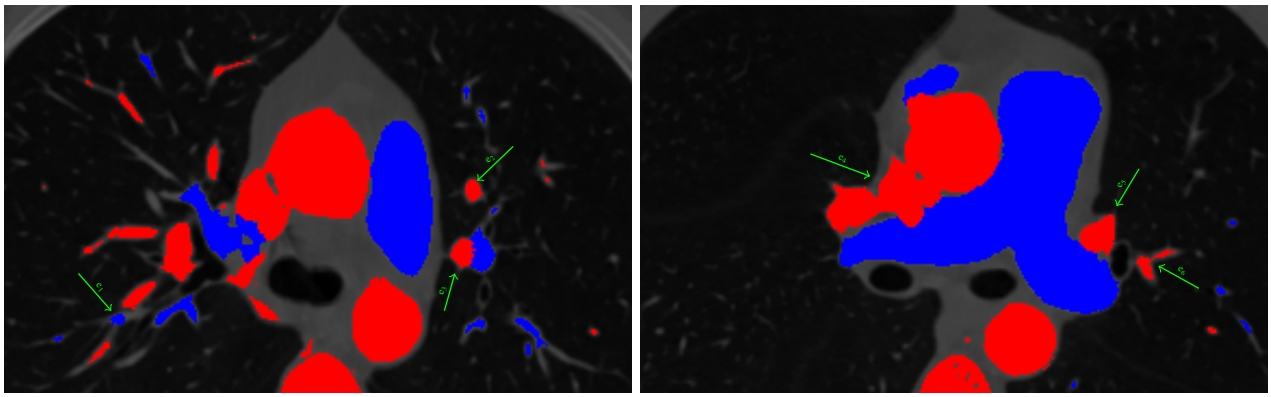
[5] J.-P. Charbonnier, M. Brink, F. Ciompi, E. T. Scholten, C. M. Schaefer-Prokop, and E. M. Van Rikxoort, "Automatic pulmonary artery-vein separation and classification in computed tomography using tree partitioning and peripheral vessel matching," *IEEE transactions on medical imaging*, vol. 35, no. 3, pp. 882–892, 2015.

[6] 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.

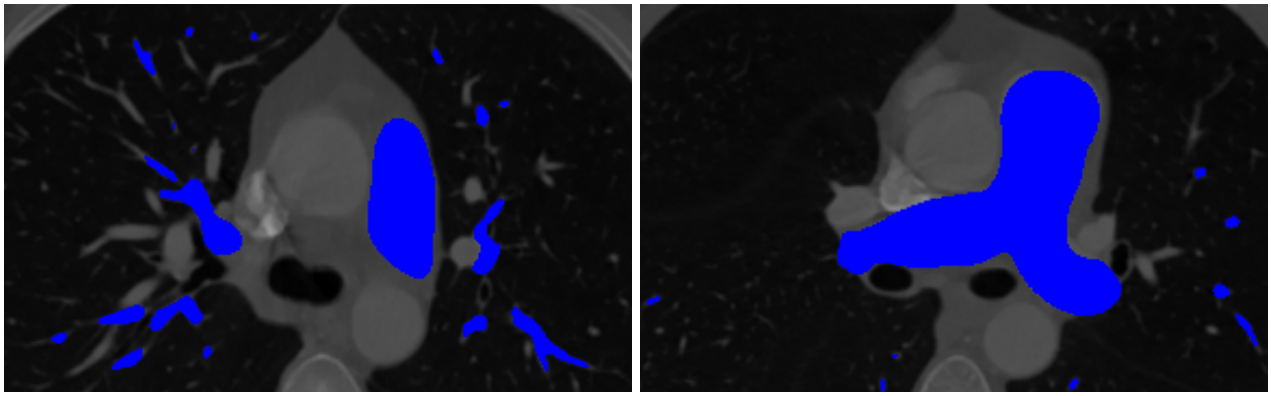
[7] P. K. Saha, F. W. Wehrli, and B. R. Gomberg, "Fuzzy distance transform: theory, algorithms, and applications," *Computer Vision and Image Understanding*, vol. 86, no. 3, pp. 171–190, 2002.



(a) Initial segmentation results by the semiautomatic method (using two seed points)



(b) Semiautomatic segmentation results after the correction step.



(c) Manual segmentation from the dataset (arterial vessels only).

Fig. 4. Pulmonary vessel segmentation on a CT scan: initial (top), corrected (middle) and manual (bottom) segmentations. Blue indicates artery and red indicates vein. Some of the errors are highlighted with green.

Segmentation on PA chest x-ray images

Ádám Tumay
Budapest University of Technology and Economics,
Department of Measurement and Information Systems
Budapest, Hungary

Dániel Hadházi
Budapest University of Technology and Economics,
Department of Measurement and Information Systems
Budapest, Hungary

***Abstract*— In the field of medical image processing, object detection techniques play a key role for Computer Assisted Diagnosis (CAD), as well as for feature extraction tasks for other algorithms. One such practical problem is the detection of lung nodules on PA chest X-ray images which can, for example, help to increase the detection of lung cancer at an early stage. In this paper our aim is to compare the performance of different Convolutional Neural Network architectures, such as simple feed-forward networks and their combination with YOLO V1's head and UNET-s combined with Densely Convolutional (DENSE) blocks on this problem. Furthermore, we provide insight into techniques used for fitting these networks on smaller datasets, by training and testing our solutions on the JSRT dataset, which only consists of 247 images. While we don't always manage to achieve a good fit, by utilizing the proposed augmentation and preprocessing techniques, we manage to substantially decrease the loss on the validation dataset, as well as get qualitatively better results. Finally, X-ray images are often only provided in an unnormalized DICOM format, where the choice of the utilized normalization method of the input images often becomes crucial in regard of the performance of neural networks. For this task we also analyze multiple methods, such as min-max normalization with the possibility of detecting outlier intensities, histogram equalization, and normalization by creating a simple, rough segmentation of the lung through traditional image processing methods and observing intensities in that area.**

Keywords—neural networks, CNN, YOLO, UNET, DenseUNET, nodule segmentation, lung segmentation, dicom normalization

I. Introduction

In the field of medical image processing, one of the major research areas today is the automated generation of diagnoses and the facilitation of medics' work by providing various auxiliary information. Although with the advent of neural networks advancements have been made in this area, the problem remains unsolved. The difficulty lies in the slow and costly process of creating large datasets with detailed and consistent annotation. The latter is often a near impossible task, as even human experts in the field will have different ideas about the nature and margin of lesions in the image, while precise validation of diagnoses (biopsy) is only possible in a small minority of cases. The practicality of the problem lies in not so much in fully automating the diagnosis process, but in helping different experts to make better decisions. In the course of this paper, our aim is to find models that are able to generalize on the available smaller size and often inconsistent datasets as well as test various traditional image processing techniques to help these models fit. The scope of our research is to find lung nodules on PA chest X-ray images using various neural network models. According to our talks with expert radiologists the exact localization of these lesions is just as important as classifying these images; furthermore, it improves the explainability aspect of diagnoses generated by the models. However, most datasets available only have partial information about the locality of the lesions, as often their exact shapes (such as masks) are not provided, only the central (pixel) coordinates on the images and a bounding box are usually given. This was also the case with the JSRT dataset used in our study, which consists of 247 PA chest X-ray images. We experimented with various models and techniques to see which ones could best fit and generalize on such a small number of samples, such as CNN-s, trying out YOLO V1's head, and UNET's DenseUNET variation. In this study, as well as in previous projects, we found that proper normalization of the inputs is key to achieve a good performance with neural networks. In case of X-ray inputs this often proves to be a challenge as they are often only available in unnormalized DICOM formats, where metadata about the normalization window is missing, but low contrast is often a problem in the case of analogue X-ray scans too. For this problem we try out various approaches such as min-max, and outlier filtered min-max normalization, as well as a method based on segmenting the lung first using the apparatus of traditional image processing and observing intensities in that area.

II. Related works

A. Neural Network Architectures

The family of Convolutional Neural Network (CNN) architectures is the most popular choice for image processing tasks since, with the help of convolutional layers, one can train neural networks where the number of parameters is independent of the input resolution, helping greatly to avoid overfitting. For our fully convolutional feed-forward baseline we used an architecture similar to the feature extraction part of the VGGNet [1], which has already proven its capabilities on various image processing tasks. A more complex, but also successful, fully convolutional architecture is DenseNet [2], which is able to look back at the output of all of the previous layers through skip connections greatly helping the propagation of gradients in deeper architectures. For coarse object localization (localization without finding the exact boundaries or shape of the object), bounding box regression is a common solution, where the YOLONet [3] family provides an accurate and computationally efficient solution. When accurate segmentation of the regions is desired, most architectures rely on a structure similar to encoder-decoder networks. In the field of medical image processing one such proven architecture is UNET [4], which we also used. A combination of this network with the aforementioned DenseNet is the DenseUnet [5] model, where the basic convolutional blocks of the UNET have been replaced with DenseBlocks.

DOI https://doi.org/10.3311/MINISY2024-004

B. Normalization

During our research, we tested different normalization methods, one of them being adaptive histogram equalization [6], a method which enhances contrast while also taking account localities of the image. One of our proposed normalization methods requires a coarse segmentation of the lung area. For this task we use a Total Variation based procedure proposed in [7], which relies on total variation filtering to extract the contours of the lung, and then utilises an active contour model [8] to fit an initial segmentation to the contours found.

III. Datasets

A. JSRT

To train our models for nodule detection we used a dataset produced by the Japanese Society of Radiological Technology [9], which has annotations for this task, but has also been the basis for a broad spectrum of projects in the field of medical image processing. The dataset consists of 247 analogue PA chest x-ray images, of which 154 contain tumours and 93 have no such lesions. The images are provided in already intensity normalized 2048x2048 png files, where one pixel corresponds to a length of 0.175 mm. The annotation for the images is available in csv format, containing the following information: tumour detectability graded from 1 to 5, tumour diameter in mm, patient gender and age, the position of the lesion in pixel coordinates, the doctor's diagnosis, and whether the tumour is benign or malignant. The quality of the annotation is outstanding in the sense that all positive samples were subsequently confirmed using 3D modality, but difficulties can arise as no more than one tumour is marked per image. Another problem in implementing learning algorithms is the low sample size, which makes generalisation difficult, and increases the risk of overfitting. Selecting a representative validation and test set might also prove to be a problem due to the number of samples. For our purposes splitting 1/8th of the dataset for validation and using the rest for training was sufficient.

B. Vindr-CXR

Since our training dataset consisted of already normalized images, for the exploration of the different normalization techniques, we used a dataset consisting of 15,000 PA images provided in DICOM format [10]. These were recorded in Vietnamese hospitals, including labels of 28 different diseases, 22 of which were also localized at a bounding box level. Since all images were assessed by 3 radiologists, we consider a given lesion positive if it is marked by at least one of the radiologists. The dataset is diverse in pathologies, nodules being one of them, but pixel-level segmentation cannot be inferred from the available information, as we found the given bounding box annotations inconsistent in size, as well as multiple nodules in a group are often classified as a single lesion.

IV. Our Work

As segmentation masks for the nodules to be detected were not available, we experimented with localization on a coarse resolution. Each nodule was approximated with a circle, the center of which was aligned with the nodule's center, and also had the same diameter. We normalized our inputs to a resolution of 512x512. To calculate the outputs, we first divided the input image to non-overlapping blocks of 16x16 pixels and measured the amount of intersection with the previously approximated circles relative to the area of these blocks. This way we get 32x32 images to train on. To train and evaluate our models binary cross entropy was used with additional weighting between the classes. The motivation of the weighting in the loss function is that mostly only a few percent of the image area contains a nodule, while there are many more negative regions, which could cause the models to converge to a near 0 output during training (i.e. the model learns only the average of the labels instead of generalization).

$$L(y,d) = -\sum_i w_{i,1} d_i log(y_i) + w_{i,2}(1-d_i) log(1-y_i) \quad (1)$$

Here, y_i is the output of the network for input i, d_i is the expected output, $w_{i,1}$ is the weight for observation i being a false negative, and $w_{i,2}$ is the penalty for false positives. The weighing step is also justified by the fact that during the annotation process, at most one nodule was flagged for an image, so we can expect that there are also untagged lesions. Furthermore, if we think of the models as region of interest filtering that detects nodules, then outputs with false positives are more informative for physicians, or for other algorithms, than outputting the zero mean. Not to mention that, for medical diagnostic applications, false positives are typically a better outcome than the failure to detect a nodule.

A. Simple CNN architecture

For our first, baseline attempt we tried a fairly simple feed-forward fully convolutional architecture inspired by the feature extraction part of the VGGNet (Fig. 1.). The network was trained with a batch size of 20 and a learning rate of 10^{-6}, using the Adam optimizer. A weight of 50 was chosen for false negatives, while the weight for false positives remained one.

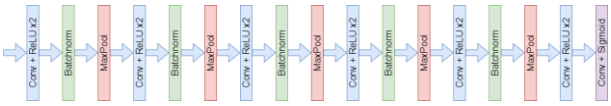


Fig. 1. The initial baseline model

B. DenseNet

To improve upon the previously introduced solution, we first tried making the architecture deeper as well as using skip connections between layers. The methods used here were motivated by DenseNet, a popular model in the field of medical image processing. The idea is that in the forward step, the outputs of all previous blocks are concatenated to a common feature map, and subsequent blocks can select from these outputs (of course, as the resolutions of the processing blocks are reduced through subsequent layers, these feature maps are also downscaled). Since the number of channels increases linearly with the depth of the network, to avoid an explosion in the number of trainable parameters, a 1x1 convolution is applied at the beginning of each Dense block reducing the number of channels. The peculiarity of the architecture is that the skip connections implemented this way allow the creation of even deeper networks by supporting the propagation of gradients to previous layers, effectively reducing the gradient vanishing problem. Another advantage of the solution is the reusability of features, i.e., the activations on the output of one block can be used by several subsequent blocks. This allows for using processing blocks with less channels, thus using fewer parameters as well as memory. The exact architecture of the processing blocks used and the overall network is illustrated in Fig. 2.

While training the network, the higher number of feature maps lead to an increased VRAM usage. Consequently, had

to reduce the batch size to 10. Additionally, the static learning rate has been replaced with a learning rate scheduler that starts at 10^{-5} and gradually decreases with the number of iterations until it reaches 10^{-7}.

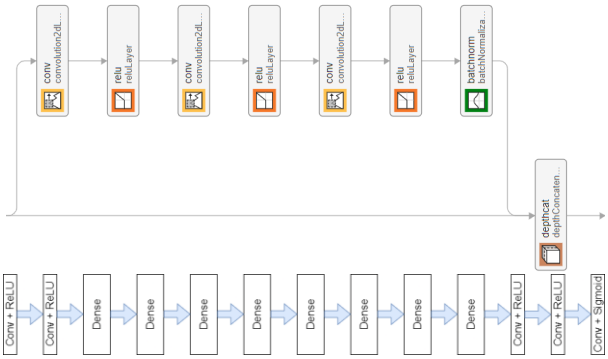


Fig. 2. The architecture of a simple Dense block (top), and the proposed network used for localization (bottom)

C. *Knowledge fusion with fully connected layers*

So far the effective sensitivity area of a given output region is not much larger than the 16x16 block it represents on the original image (in practice, the area is larger than that due to the overlapping convolutional kernels, however, pixels far from a given region on the output have only a marginal effect on the output activations of that region). This means that our models would have to determine whether there is a nodule in a given region based only on this very local information, making them much more difficult fit and generalize. If we interpret the task not as generating classifications for specific regions, but as segmentation of the nodules on a much lower resolution, techniques used for this purpose can be applied. One possibility was to use fully connected layers – which were also utilized in YOLO V1 networks – to produce the output. This way, the sensitivity area of the output neurons is extended to the whole image, allowing them to calculate the output for a region while also taking the predictions for other regions into account. The usage of fully connected layers greatly increases the number of trainable parameters, making the network prone for overfitting and also causing other stability issues due to the potential redundancy. To combat these, we have decreased the number of input channels before the fully connected layers, added dropout with a probability of 0.5 for more robust operation in parallel with a strong L2 regularization. Unfortunately, the number of trainable parameters are still in the range of millions, which greatly increases the memory consumption of the model, thus slowing down the training significantly. The new network head can be seen in Fig. 3., which is applied after the DenseNet described in the previous chapter.

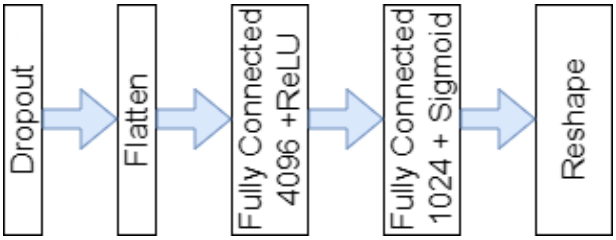


Fig. 3. The proposed architecture of the new head with fully connected layers

D. *DenseUNET*

Another option is motivated by the following: suppose that the architecture presented as DenseNet only produces a lower resolution image on which we want to perform the segmentation. We can process this output with UNET, a multi-level encoder-decoder network where the levels of the same resolution are connected by skip connections. We also used Dense blocks as the processing blocks for the UNET part of the network, allowing any given block to use the outputs of any of the previous blocks. The resulting new head can be seen on Fig. 4., which we apply after the network explained in the DenseNet chapter.

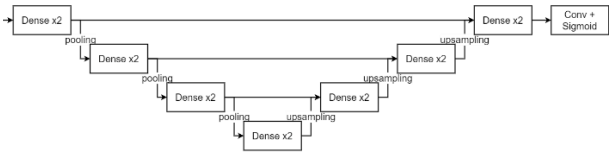


Fig. 4. The proposed architecture of the new head using DenseUNET as the base idea

E. *Augmentation Techniques*

During training, due to the low number of samples we observed significant overfitting for most models. To remedy this, we relied on the virtual increase of the number of samples – augmentation. The technique consists of performing transformations on the input data which should result in an invariant behavior of the network response. In practice it is important, that the transformed versions of the same image are different enough so that the network cannot overfit on them, but the resulting variations sample the same background distribution as the original samples (because of the expected invariant behavior). To achieve this, we applied the following transformations on the samples and their corresponding expected outputs:

- Normally distributed noise is added to the input intensities to make overfitting on high frequency components more difficult.
- A small amount of random rotation on both inputs and outputs, as convolutional filters are not rotation invariant. This also makes overfitting more difficult as well as the prediction more robust for inputs which are not well aligned.
- Random shifting on both inputs and expected outputs. Although convolutional filters are invariant for shifted inputs, it is theoretically possible to learn the distance of a region from the edge of the image due to padding, and the invariant property is also spoiled by pooling layers, making room for overfitting. By using shifts, the possibility of this kind of overfitting is ruled out. This step again also helps making the models more robust when having to predict on poorly aligned images.
- Random rescaling of the input and output, since the convolutional operation is not scale invariant, this transformation should also help against overfitting.

F. *Normalization Techniques*

While for training, the input images are in an already normalized png format, if we want to reliably predict for new images, the preprocessing of DICOM files becomes necessary. Since normalization windows are often not

annotated, an automatized way is necessary. In most cases ordinary min-max normalization is not enough since radiographs often contain outlier values.

1) Histogram-based methods

Histogram equalization works by trying to match the distribution of the image intensities to a predefined distribution by applying a monotone increasing function on the intensities pixel-wise. This can either be the histogram of another image's intensities, or uniform distribution if we want to maximize contrast. A drawback of using these methods is that the value set of the target distribution is usually much smaller to facilitate the matching of the two histograms, which results in a loss of information. Another disadvantage is that this method is also prone to amplify noise on otherwise homogenous areas. An improved version of the algorithm is adaptive histogram equalization [6], which consists of dividing the input into regions, performing the smoothing on each of these regions separately, and then producing the output from these regions. The advantage of this method is that it is suitable for enhancing contours in local regions with small intensity differences, furthermore, contrast limitation can also be used to lessen the noise amplification.

2) Filtering outliers

Another possibility is to normalize by trying to bring areas of the same radiodensity to a common intensity. If we live with the assumption that any two lungs consist of the same materials, min-max normalization would be a feasible solution. However, this assumption does not hold in most cases, due to scanning issues, as well as artificial objects such as pacemakers. A more robust solution would be to assume that a portion of the most and least dense substance of the images approximately have the same densities. This way we can create a normalization method more resilient to outliers by sorting the points by intensity and taking the intensity measured at the 10^{th} and 90^{th} percentiles as the new minimum and maximum. This is based on the observation that both the darkest and lightest areas typically occupy more than 10% of the image surface area, while areas where anomalies occur typically occupy less than that, thus there is a high probability that we will approximate the real 10^{th} and 90^{th} percent of the distribution accurately.

3) Normalization based on lung segmentation

The efficiency of the previous method could be greatly improved if we could only take pixel intensities in the lung area into account and perform the normalization based only on these pixels, since this way we can ensure that the selected intensities represent the same material. Although both conventional and neural solutions for segmentation exist, since robustness is key for a preprocessing step, we chose a conventional image processing method based on [7] that uses the following steps:

i. The input DICOM image intensities are first normalized by adaptive histogram equalization.

ii. The resulting images are blurred using the total variation denoising (TV) algorithm.

iii. Blurred images are binarized by thresholding and then divided into connected components. The components are filtered to remove smaller areas treating them as noise. Two elements of the remaining set which are the most likely to represent the lung halves are selected as candidates.

iv. Candidates are compared along several well-formedness constraints to check whether the segmentation is accurate. If not, the thresholding is performed at a higher level, and we jump back to step iii. In case no components remain after thresholding we return with an error.

v. If candidates are found to be accurate, their convex envelope is constructed and refined with an active contour model aligning its contour to the pleura on the original image.

The idea of the total variation denoising step is to create an image where the energy of the gradients is minimal (i.e. the edges are blurred as much as possible). A trivial solution to the problem would be to create an image using only a single intensity, thus an extra regularization term is also necessary that penalizes the difference between the original and the new image.

$$\arg min_u \left\{ \lambda \int |\nabla u| + \frac{1}{2} \int (f-u)^2 \right\} \qquad (2)$$

Here the integration is performed on the pixels of the images, f is the original, u is the resulting image, λ represents the regularization hyperparameter of the algorithm (in our case λ=2 proved to be optimal).

Lung candidates are selected by taking the two connected components that are closest to the points in the first and third quarters of the image horizontally and the top third of the image vertically. For these candidates, the following metrics are examined for each candidate separately: eccentricity, equivalent diameter, area relative to the image, and whether the components are adjacent to the image boundary. If more than one constraint is violated for any component, candidates are classified invalid. (One constraint might be violated to allow possibilities like the edge of the lung hanging out of the image boundary.) Once candidates passed these constraints, we also examine them relative to each other to make sure that their diameters do not differ too much, or that they do not occupy too large an area of the original image. If all tests are successful, we accept the candidates. After creating an initial convex envelope for the lungs, a series of shrinking and expansion steps follow using the active contour model. However, we found that using a single expansion step is computationally more efficient and stable without compromising accuracy significantly.

During our initial experiments with the algorithm, we found that in some cases it is prone to falsely classify the background as one of the lung components, as well as some light stripes on the side of images might occur due to the way images were scanned, which can make it difficult to segment the background. To combat these issues, we used morphological opening and closing to eliminate the stripes created during the scanning process. The next step is finding and filtering out the background of the image by first binarizing the image at the 30^{th} percentile, finding connected components, and classifying them as background if the length of their boundary adjacent to the boundary of the image is longer than a portion of the circumference of the image. (in our case that portion is 10%)

After creating the segmentation of the lung this way, we can use the previously mentioned outlier filtered min-max normalization on the segmented area for an even more robust operation.

V. Results

A. Simple CNNs

Neither the proposed architecture similar to VGGNet nor DenseNet could achieve a good fit. The first network was not able to generalize even on the training set, while the second architecture was not much more successful either, only being able to localize the lung area with its predictions. Results suggest that the representational ability of the network is below what the task would require, as it cannot even overfit on the small number of training samples. We identify the small sensitivity area of the output regions as the cause of the phenomenon, as latter models showed better performance.

B. Fully connected and DenseUNET heads

In this case both networks were able to overfit on the training set successfully, as expected, however due to the small number of samples, they could not generalize on the validation set. To overcome this issue, we used augmentation in subsequent trainings.

C. Augmentation

Using augmentation, the network using a head with fully connected layers, although no longer overfit, was not able to generalize on the validation set. In contrast, the DenseUnet architecture converged to a lower loss, and the effect of regularization is clearly visible on the loss functions as training and validation losses have similar values. While qualitatively assessed, the output of the model is still unsatisfactory, it is able to find nodules in general, although oversegmenting them as well as giving many false positives. However, by observing the location of such incorrect outputs, they mostly occur at points where there is some irregularity or where such lesions are often formed. Furthermore, by observing the segmentation results qualitatively, we noticed that if there was a nodule in a given image, higher activations in the output could be observed, thus to some extent the output energy correlates with the binary classification of the images, even if the localization is incorrect. The resulting training losses and examples of the segmentation results can be seen on Fig. 5.

D. Normalization techniques

1) Histogram equalization

It was also noted in our study that histogram equalization can produce higher contrast images than those created through basic min-max normalization. It is essential to note, though, that while histogram-based methods offer more easily interpreted outcomes for humans, the coarse intensity levels and the introduction of high frequency noise can negatively affect the results of neural networks or conventional image processing algorithms. A comparison of the results with min-max filtering can be seen on Fig. 7.

2) Outlier filtering and lung segmentation

We evaluated the performance of the segmentation algorithm in IV.F.3) by running it on the Vindr-CXR dataset and for each sample we calculated the proportion of the image area that was declared as lung and observed the resulting histogram. based on this metric, in around 2% of the cases the algorithm was not able to find the lungs. By further examining the segmentations manually, we found that in addition to the clearly erroneous results mentioned above, a further 10-20% of the results showed significant under-segmentation or wrong selection for lung candidates. Some examples of the segmentation can be seen in Fig. 6.

VI. Discussion

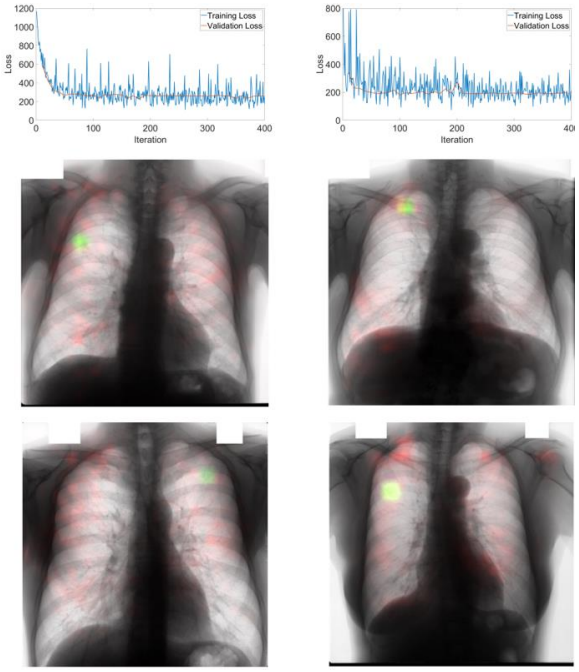


Fig. 5. Training results using augmentation. The left column shows the results using fully connected layers in the head of the network, while the right column shows the results obtained with DenseUnet. The first row shows the evolution of the loss functions, the second row shows the prediction of the models on the train dataset, and the third row shows prediction for a sample in the validation part of the dataset. The red colour channel indicates the prediction of the network while green represents the expected output. Lesions found are shown in yellow.

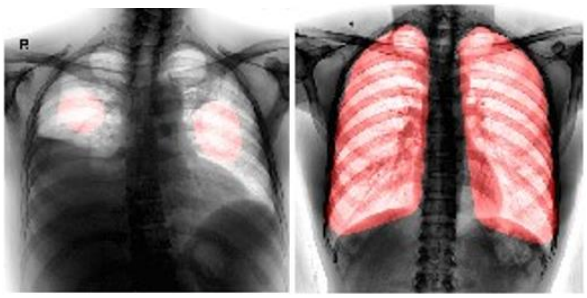


Fig. 6. An unsatisfactory (left), and a usable (right) segmentation of the lung area using the proposed algorithm.

Fig. 7. A comparison between different normalization methods. Min-max (left), outlier filtered min-max (center), histogram equalization (right), note the different anomalies introduced using the last method.

A. Nodule segmentation architectures

The quality of the predictions did not justify quantitative analysis, hence, our evaluation of the performance of the networks was primarily qualitative. Based on the experiments, we derived the following conclusions about the task:

1) The problem requires neural networks with more representational power than we first expected as most architectures struggled to overfit even on a small number of samples.

2) For an accurate evaluation of a region, it is not sufficient to observe only the pixels within that region, instead, an architecture must be considered that has a much wider sensitivity area than the region examined.

3) A larger dataset is necessary, because although augmentation has been able to make current architectures generalize, a network with a larger representational capability would require stronger transformations, which might lead to the augmented images describing a completely different background distribution than the original samples. (This might have already been observed during the DenseUnet training, where the validation loss runs slightly lower than the training loss, which may suggest an over-regularisation effect of the augmentation or a necessary increase to the number of trainable parameters.)

B. Normalization techniques

Because of the different anomalies histogram equalization is prone to produce, we do not recommend using inputs normalized this way for training or for prediction without further processing. While the proposed segmentation algorithm works well on most samples, it is not robust enough to be used for normalization. Furthermore, it is difficult to avoid some commonly observed fault modes of the segmentation model, such as misaligned patients on some radiographs, or where one or more of the lobes are filled with fluid. In the end we find that the best method for normalizing DICOM images is the outlier filtered min-max normalization as it is able to work robustly in various scenarios without producing major anomalies.

ACKNOWLEDGMENT

This research was supported by the National Research, Development, and Innovation Fund of Hungary under Grant TKP2021-EGA-02, and the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.

REFERENCES

[1] Karen Simonyan, Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), 2015.

[2] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger, "Densely Connected Convolutional Networks," in *CVPR 2017*, 2017.

[3] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *MICCAI 2015*, 2015.

[5] Xiaomeng Li, Hao Chen, Xiaojuan Qi, Qi Dou, Chi-Wing Fu, Pheng Ann Heng, "H-DenseUNet: Hybrid Densely Connected UNet for Liver and Tumor Segmentation From CT Volumes," *IEEE Transactions on Medical Imaging*, vol. 37, no. 12, 2018.

[6] S. M. Pizer, E. P. Amburn, J. D. Austin, et al, "Adaptive Histogram Equalization and Its Variations," *Computer Vision, Graphics, and Image Processing*, vol. 39, pp. 355-68, 1987.

[7] Narathip Reamaroon , Michael W. Sjoding, Harm Derksen, Elyas Sabeti, Jonathan Gryak, Ryan P. Barbaro, Brian D. Athey and Kayvan Najarian, "Robust segmentation of lung in chest x-ray: applications in analysis of acute respiratory distress syndrome," *BMC Medical Imaging*, 2020.

[8] Michael Kass, Andrew Witkin, Demetri Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321-331, 1988.

[9] "Japanese Society of Radiological Technology," [Online]. Available: http://db.jsrt.or.jp/eng.php. [Accessed 06 10 2023].

[10] Ha Q. Nguyen et al., "VinDr-CXR: An open dataset of chest X-rays with radiologist's annotations," 2020.

Nonparametric Statistical Testing of Functional Connectivity in EEG Data

Mihály Vetró
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
vetro@mit.bme.hu

Gábor Hullám
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
gabor.hullam@mit.bme.hu

***Abstract*—The use of nonparametric permutation-based statistical tests for the analysis of functional neuroimaging data (mainly EEG, MEG, and fMRI) is a common approach for comparing observations under different conditions, or in different subject groups. There are variations of these methods to analyze brain signals obtained along the spatial and temporal dimensions, and also across frequency, which accounts for oscillatory activity. However, there is no well-known method for nonparametric testing of the measured functional connectivity between different areas of the brain. In this paper, we introduce a modified version of an existing nonparametric testing framework, which we adapted for use with functional connectivity data. By using this method, we will show that there are brain areas between which patients with Alzheimer's disease have reduced functional connectivity in an eyes-closed resting state, based on EEG data.**

***Index Terms*—Statistics, Bioinformatics, Electroencephalography, Functional connectivity**

I. Introduction

It is a common case during the analysis of Electroencephalography (EEG) data, that the possible spatial-, temporal- and spectral extent of the effect to be detected is large, or even unknown to the researchers. In these cases, the utilization of cluster-based nonparametric permutation tests is a common solution, which provide some insight into the extent of the detected effect (if any), while also handling the multiple comparisons problem (MCP). These cluster-based tests exploit the fact that the individual data-points in space, frequency and time (within an EEG recording) are usually not independent of each other, as they tend to correlate with adjacent data-points. This dependence within the data provides the opportunity to exempt the results of the statistical tests conducted over the neighbouring data-points (in space, frequency and time) from the usual approaches that are meant to compensate for multiple comparisons. However, as the related literature presented in Section II show, these methods still provide a way to control the false positive rate, by restricting the definition of dependence between the tested variables to those data-points, which are directly adjacent. Consequently, the application of such a cluster-based test requires a definition of some type of adjacency along all dimensions of the data. This definition is not evident for some representations, and according to the current literature, there is no well-known method to define adjacency for functional connectivity data, which is a representation that provides insight into the synchronization between different areas of the brain across time. A solution to define adjacency within functional connectivity data is proposed in Section III, then a modified version of an existing cluster-based permutation testing framework is demonstrated using this adjacency definition in Section V. Finally, further insights are provided in Section VI, and possible improvements and additional research directions in Section VI-A.

This research was supported by the National Research, Development, and Innovation Fund of Hungary under Grant TKP2021-EGA-02, and the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.

II. Related work

There are two main topics regarding this paper which need introduction to properly position the presented approach: (1) the concept of the cluster-based nonparametric permutation test, along with the concrete testing framework which was used as the basis of the approach, and (2) some examples in the current literature for the statistical analysis of functional connectivity data.

A. Cluster-based nonparametric permutation test

When analyzing EEG data, most research aims fall into two categories: either to distinguish two (or more) mental states of a subject by comparing different time segments of the same EEG recording (typically evoked responses to external stimuli), or to identify the differences between two (or more) subject groups based on their EEG data (also typically evoked responses). Both of these cases involve comparing samples with a high amount of variables, the direct (and uncontrolled) comparison of which can result in a high false positive rate.

To solve this problem, Maris and Oostenveld proposed a method [1] based on the cluster-based nonparametric testing paradigm, which can detect and localize significant differences between samples of EEG data while also solving the problem of multiple comparisons, therefore controlling the aforementioned false positive rate. This method is nonparametric in the sense that it can be applied using any test statistic that is capable of comparing two (or more) independent samples[1].

[1]Some modified versions of this method also exist for one-sample and paired test statistics, that are not covered in this paper.

DOI https://doi.org/10.3311/MINISY2024-005

The method proposed by Maris and Oostenveld, that will be referenced here as the Spatio-Temporal Cluster Test (STCT) – in line with the terminology used in the MNE Python framework [2] – makes the singular assumption that there is some (maybe unknown) probability distribution, that the samples were drawn from, then considers the null-hypothesis that all samples have the same probability distribution. It should be noted – according to Maris and Oostenveld – that this is the same null-hypothesis that is made by most of the well-known statistical tests, like the t-test or the F-test, but without assuming normality or equal variance.

To define the STCT method, Maris and Oostenveld first introduces the concept of a permutation test, that serves as the basis of their approach. This test is supposed to determine if two (or more) independent random samples have the same distribution. Although their definition only considered two samples (i.e. two types of stimuli or two subject groups), it can be concluded, that the test can be applied for any number of samples (at least two), by using a test statistic capable of handling multiple samples. The general steps of the permutation test (following the mentioned insights) are the following:

1) Measure a test statistic for the samples, let the resulting value of this statistic be denoted by τ.
2) Permute the samples by randomly redistributing the data between them, while maintaining the original size of every sample, then calculate the test statistic for the resulting permuted samples.
3) Repeat step 2 K times, and let the value of the statistic calculated after repetition k for $\{k \in \mathbb{R} | 1 \leq k \leq K\}$ be denoted by $\hat{\tau}_k$.
4) Let the set of k-values for which $\tau < \hat{\tau}_k$ holds be denoted by $\boldsymbol{S} = \{k \in \mathbb{R} | 1 \leq k \leq K, \tau < \hat{\tau}_k\}$.
5) Calculate an approximate p-value $\hat{p}$ for the null-hypothesis (which states that all the samples have the same distribution) by dividing the cardinality of $\boldsymbol{S}$ with the number of permutations: $\hat{p} = |\boldsymbol{S}|/K$.
6) If for some arbitrary critical α-level (typically, $\alpha = 0.05$) $\hat{p} < \alpha$ holds, then conclude that the distributions of the samples are significantly different.

After introducing the above permutation test, Maris and Oostenveld proceeds to describe the cluster-based test statistic, first proposed by Bullmore et. al. [3] for use with structural MRI data. In this introduction, Maris and Oostenveld only considers the signal of a single electrode, then they elaborate on how this test statistic can be adapted for use over the data of multiple electrodes and the time-frequency representation. In this paper, the general (extended) procedure within the STCT method to calculate said cluster-based test statistic is described, which consists of the following steps:

1) For every variable (a point in space, time and possibly frequency) compare the value of that variable in the different samples by means of an arbitrary statistic (e.g. a t-test for two samples, or ANOVA for multiple samples).
2) Select all variables whose test statistic exceeds some threshold (e.g. a threshold for the t-value of a t-test can be computed from the sample size and an arbitrary α critical value, typically $\alpha = 0.05$).
3) Create a graph in which the selected variables are the nodes, and they have an edge connecting them if and only if they are adjacent (on the basis of spatial, temporal and possibly spectral adjacency).
4) Cluster the selected variables in such a way that the number of clusters is equal to the number of components[2] of the graph defined in step 3, and for every distinct component there is a cluster that consists exactly of its nodes (variables).
5) Calculate the cluster-level statistic for every cluster, which is the sum of the previously calculated statistic values of every variable within that cluster.
6) Take the cluster with the largest cluster-level statistic, and the cluster-level statistic of that cluster will be the resulting statistic of the cluster-based test.

The STCT method uses this cluster-based test as the test statistic while performing the above-defined permutation test. The result of this test consists of a p-value for the null hypothesis that the samples have the same distribution (from the permutation process), and a cluster of adjacent (locally significant) variables for which the cluster-based test statistic is highest (from the process of calculating the cluster-based test statistic). The validity and formal correctness of the STCT method is explained by Maris and Oostenveld in their paper.

B. Statistical analysis of functional connectivity data

Functional connectivity in the brain is a concept defined in a broad sense by Friston [4] as the "temporal coincidence" between spatially distant neurophysiological activities. The main concept behind functional connectivity is that areas of the brain are presumed to be coupled, or to be part of the same – temporarily – coherent network if their activities show consistent temporal or spectral correlation within a given time period.

A multitude of metrics exist which aim to quantify functional connectivity, and a common property of them is that they all assume a fully connected graph, in which the nodes are points in space where brain activity is measured, and the functional connectivity values are the edge-weights between these nodes. Therefore, functional connectivity can be defined as an attribute of every pair of spatial points. Although these points can be defined in multiple ways (e.g. sensors of the EEG/MEG device or reference points in a source estimate), only the EEG electrodes will be considered as spatial points in this paper, for the sake of simplicity. For similar reasons, only symmetric functional connectivity metrics will be considered, which don't carry any information about directionality.

In the statistical testing of functional connectivity data, the greatest difficulty is often the sheer number of variables (connectivity values): for N spatial points (in this case: EEG

[2]A component of a graph is a connected subgraph that is not part of any larger connected subgraph.

electrodes) the number of possible – symmetric – connectivity values C is defined as the number of edges in an – undirected – complete graph with N nodes: $C = N(N-1)/2$, therefore the number of connectivity values C is quadratic with regards to the number of spatial points N.

Based on previous research, there are three main approaches through which statistical tests are conducted on functional connectivity data in the literature: either (1) only a small subset of connectivity values are compared, (2) a large number of connectivity values are compared independently, or (3) some aggregate of the connectivities is used for comparison.

As an example of the first approach, Rodinskaia et. al. performed statistical tests [5] on the coherence measured between 4 electrode-pairs, to examine the cross-hemisphere connectivity of patients with Alzheimer's Disease (AD) and Mild Cognitive Impairment (MCI) compared to healthy controls during various mental tasks.

Illustrating the second approach, Clark et. al. performed statistical tests [6] comparing all measured coherence values of AD and MCI patients with healthy controls while performing a memory task, then applied a Bonferroni-like correction (which was originally intended to correct for multiple independent tests) in order to control the false positive rate.

The third approach is utilized by Cea-Cañas et. al. [7], who used the Connectivity Strength metric (which is the mean phase-locking value between all electrode-pairs) to measure task-related modulation of the overall connectivity in the brain in patients with schizophrenia and bipolar disorder compared with healthy controls, while performing an auditory tone-detection task.

Finally, also using the third approach, Fodor et. al. [8] used the mean coherence between the electrodes, and also a graph-based connectivity metric called the maximum Betweenness Centrality of the Minimum Spanning Tree[3] in the connectivity graph, in order to distinguish between patients with MCI and healthy controls.

There are distinct problems with all three presented approaches: approach (1) requires prior knowledge (or assumptions) for the localization of the effect, approach (2) assumes independence between the connectivity values, which leads to overcompensation for the false positive rate, and finally, the aggregation used in approach (3) mitigates localized effects, making them harder to detect, due to a worse signal-to-noise ratio.

III. Description of the Method

As it was discussed at the end of section II-B, the current analyses of connectivity data used in the literature require either prior knowledge of the extent of the effect, or the use of some type of aggregated metric. If none of these two conditions are met, then the false positive rate is often overcompensated in the results of the performed tests. The STCT framework (detailed in section II-A) provides a solution for all of these problems, however its clustering step requires some type of adjacency to be defined between the variables of the data, along all dimensions. Temporal- and spectral adjacency can be defined intuitively as the simple rule, that "subsequent variables in time and/or frequency are adjacent". Also, adjacency in space between voltage or power variables (as in one point in time and/or the frequency spectrum) in an EEG measurement is usually defined in one of two ways: (1) either using a simple distance threshold between the points of measurement (a typical threshold is 40 mm for EEG electrodes, but it depends on the electrode density and overall number of electrodes), or (2) by using Delaunay triangulation between the electrodes on the spherical head surface that has been laid out on the 2D plain. However, none of these two methods can be applied directly to functional connectivity measurements, as there is no evident spatial position to a functional connectivity variable. In this section, an algorithm is introduced, to define the adjacency of functional connectivity variables in space based on empirical observations, so the application of the STCT framework to functional connectivity data becomes possible.

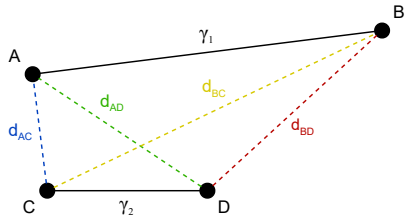


Fig. 1. An example case of two functional connectivities. γ_1 means the functional connectivity value between electrodes A and B, γ_2 similarly represents the functional connectivity value between electrodes C and D. The spatial distances of all possible electrode-pairs X and Y (that are not part of the same connectivity measurement) are denoted by d_{XY}.

A. Spatial distance between connectivity variables

As it was discussed in section II-B, the variables in functional connectivity data can be interpreted as edge-weights of a complete graph, where the nodes of the graph are points of measurement in space, which are EEG electrodes in this case. To ease the definition of a spatial distance metric between functional connectivity measurements, the following assumption is made: two functional connectivity values are more likely to correlate if they are defined between similar brain areas. For example, two connectivities are likely to correlate, if both of them have one of their electrodes in the frontal lobe, and the other in the occipital lobe. After this assumption, the spatial distance between two connectivity values (edges) can be defined as a function of the spatial distances between the node-pairs belonging to the connectivity variables.

For the adjacency method, two distance functions were devised between functional connectivity variables γ_1 and γ_2, which are $d_{min}(\gamma_1, \gamma_2)$ and $d_{max}(\gamma_1, \gamma_2)$. These functions are

[3]In their approach, the spanning tree with the maximum total connectivity value (sum of edge-weights) was selected, therefore the name can be misleading. The naming choice is most probably related to the fact that the term "Minimum Spanning Tree" is widely accepted in graph theory.

defined the following way, using the terminology introduced in figure 1:

$$d_{min}(\gamma_1, \gamma_2) = min(d_{AC}, d_{AD}, d_{BC}, d_{BD}) \tag{1}$$

$$d_{max}(\gamma_1, \gamma_2) = \begin{cases} d_{AC}, & \text{if } d_{min}(\gamma_1, \gamma_2) = d_{BD} \\ d_{AD}, & \text{if } d_{min}(\gamma_1, \gamma_2) = d_{BC} \\ d_{BC}, & \text{if } d_{min}(\gamma_1, \gamma_2) = d_{AD} \\ d_{BD}, & \text{otherwise} \end{cases} \tag{2}$$

The distance functions Eq. 1 and Eq. 2 are defined in such a way, that d_{min} is always the smallest pairwise distance in space between the electrodes of the two connectivities (disregarding the distance between electrodes connected by the functional connectivities in question), and then the value of d_{min} directly determines the value of d_{max}, which is always the "other" electrode pair, compared to the pair between which d_{min} is measured. For example, if from the distances $\{d_{AC}, d_{AD}, d_{BC}, d_{BD}\}$ in figure 1 d_{AC} is minimal, then d_{min} becomes d_{AC}, which disregards the distance between electrodes B and D, hence determining the value of d_{max} to be d_{BD}. From now on, d_{min} will be referred to as the minimum distance between two connectivities, and d_{max} as the maximum distance between connectivities. It is useful to notice, that if electrodes A and C or electrodes B and D are the same, then $d_{min} = 0$.

B. Adjacency of functional connectivity variables

In the current approach, the adjacency of functional connectivity variables are defined as a function of d_{min} and d_{max} (introduced in section III-A), which determines if variables γ_1 and γ_2 are adjacent. This adjacency function works by defining an upper threshold for both d_{min} and d_{max}, however defining a universal threshold that provides reliable results in every setting is most likely impossible. Because of this reason, the following algorithm is used to determine the thresholds for d_{min} and d_{max} in an empirical way, based on the available data for the functional connectivity variables:

1) Take all the functional connectivity variables, and compute the Pearson correlation coefficient[4] between every variable-pair (based on their occurring values), thus creating a correlation matrix.
2) For every correlation coefficient (between every variable-pair), based on the value of the coefficient and the amount of data available, compute the p-value for the null hypothesis stating that the variables in the variable-pair are not correlated.
3) Using the Bonferroni-corrected critical value $\alpha = 0.05/N_c$ (where $N_c = N_v(N_v - 1)/2$ is the number of comparisons, if N_v denotes the number of connectivity variables) determine which variable-pairs are correlated and which are not.

[4]Only a possible linear connection is considered, which is a stricter criterion than statistical dependence.

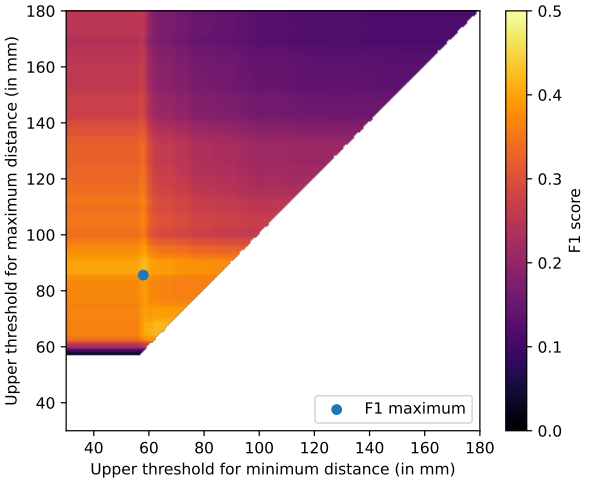


Fig. 2. The F1 scores computed for the different threshold-combinations t_{min} and t_{max}, as well as the selected threshold (blue dot) where the F1 score is maximal. The connectivities extracted from the dataset described in section IV were used for calculation, where one subject was equal to one data-point for every variable. Note, that the minimum physical distance between any electrode pair in this particular setup is 57 mm.

4) As the result of the test conducted in step 3, let the correlated variable-pairs be considered as "positive" and the uncorrelated variable-pairs as "negative" records.
5) The thresholds t_{min} and t_{max} will be upper thresholds for the value of d_{min} and d_{max} respectively, so the thresholding operation will predict the variable-pair correlated ("positive") if and only if both $d_{min} < t_{min}$ and $d_{max} < t_{max}$ hold, otherwise the variable-pair will be predicted as not correlated ("negative").
6) Based on the true correlations defined in step 4, and the predicted correlations defined in step 5, set the value of t_{min} and t_{max} in such a way that the maximal F1 score (which is the harmonic mean of the precision and the recall) is achieved.

An example result of this threshold-finding algorithm is presented in figure 2. Using the thresholds t_{min} and t_{max}, the adjacency function is the following: functional connectivity variables γ_1 and γ_2 are deemed adjacent if and only if both $d_{min}(\gamma_1, \gamma_2) < t_{min}$ and $d_{max}(\gamma_1, \gamma_2) < t_{max}$.

IV. Data used

For purposes of demonstration, a dataset [9] containing eyes-closed resting state recordings of 88 subjects were utilized, of which 36 were diagnosed with Alzheimer's Disease (AD group), 23 with Frontotemporal Dementia (FTD group), and and the remaining 29 subjects are healthy controls (Control group). The recording was performed using a Nihon Kohden EEG 2100 clinical device with 19 electrodes arranged according to the international 10-20 system, and a sampling frequency of 500Hz. No additional preprocessing was performed on the raw data, besides the preprocessing pipeline

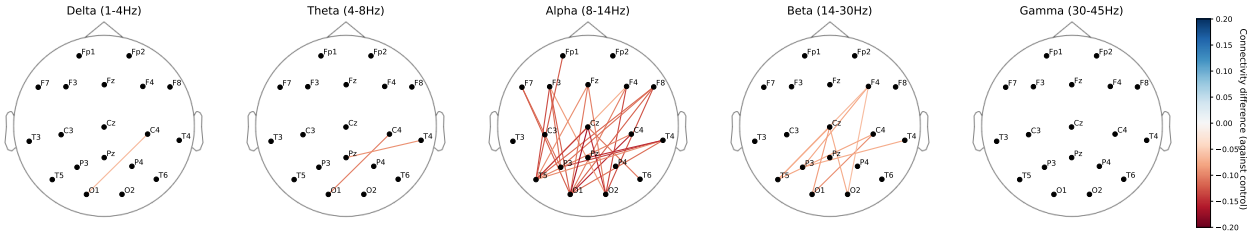


Fig. 3. The difference in connectivity of the AD group compared to the control group. Only those connectivity variables (edges) are shown, which are part of the significant cluster found by the STCT method, with a p-value of less than 0.0001. The electrodes are denoted according to the international 10-20 system. Red edges indicate decreased functional connectivity compared to controls.

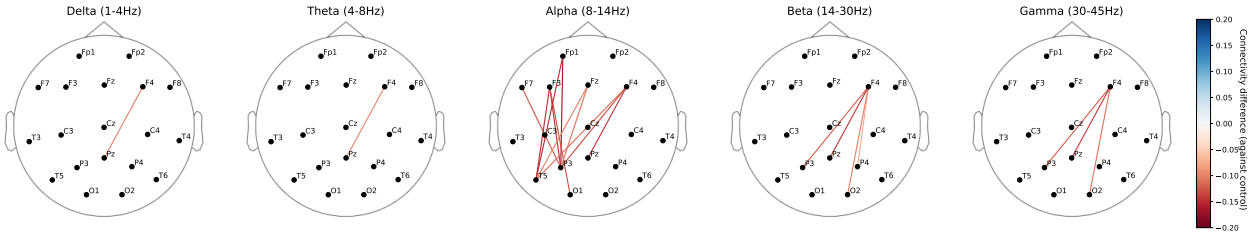


Fig. 4. The difference in connectivity of the FTD group compared to the control group. Only those connectivity variables (edges) are shown, which are part of the significant cluster found by the STCT method, with a p-value of 0.0003. The electrodes are denoted according to the international 10-20 system. Red edges indicate decreased functional connectivity compared to controls.

that was already applied by the authors of the dataset, as it is detailed in their corresponding paper [10].

From the preprocessed EEG recordings, the functional connectivity of all electrode-pairs were computed for every subject using the Phase-Locking Value (PLV) [11] for the Delta (1-4Hz), Theta (4-8Hz), Alpha (8-14Hz), Beta (14-30Hz), and Gamma (30-45Hz) frequency bands. Since the PLV metric can only be exactly computed for one specific frequency, all of the five frequency bands were sampled uniformly at 20 frequency values, then the average PLV for the given frequency bands were acquired. The resulting connectivity data contained 171 connectivity values (from the number of electrodes) for the 5 frequency bands for every subject.

V. Practical results

To determine if there is a significant difference in functional connectivity between AD or FTD patients compared to healthy controls, the Spatio-Temporal Cluster Test (STCT, described in section II-A) was utilized. As the statistic that is used to form clusters within the cluster-based test of section II-A, Welch's t-test was utilized, for which equal variance of the samples is not assumed. Two separate tests were conducted, one for the comparison of AD patients to healthy controls, and the other for the comparison of FTD patients to healthy controls. In terms of dimensionality, the data of every subject contained two dimensions: functional connectivity values in space and frequency. Along the spectral dimension a simple adjacency was used based on ordinality, while along the spatial dimension, the novel algorithm described in section III-B was utilized to determine adjacency. The resulting F1 scores for the considered thresholds, as well as the selected threshold values (where the F1 score is maximal) are presented in figure 2.

As a result, both conducted tests returned a significant cluster, the first one (AD) having a p-value[5] under 0.0001, and the second one (FTD) having a p-value of 0.0003. The difference in connectivity in the detected cluster for the AD group is shown in figure 3, and in the detected cluster for the FTD group in figure 4.

In both cases, a significant decrease in connectivity can be observed between the frontal and parietal/occipital lobe considering the given condition (either AD or FTD), mainly in the Alpha (8-14Hz) frequency band.

Additionally, the thresholds selected to determine the adjacency between the connectivity variables, as well as the F1 scores of all threshold-combinations t_{min} and t_{max} are shown in figure 2.

VI. Discussion

In this paper, a novel algorithm was introduced (in section III-B) as our main contribution, to determine the adjacency between the variables in functional connectivity data, so the STCT method (discussed in section II-A) can be applied to conduct tests over samples of said data-type. Using this adjacency, the STCT framework was applied to show that patients with Alzheimer's Disease and Frontotemporal Dementia have significantly reduced functional connectivity between the frontal lobe and the parietal/occipital lobes. Additional research is required to investigate this result from multiple aspects. It is also worth noting, that the p-value returned by

[5]A permutation count of 10.000 was utilized, therefore $1/10000 = 0.0001$ is the lowest measurable p-value in this case.

the STCT method is only indicative of the existence of *some* effect, while the certainty regarding the extent of the detected effect is unknown, according to Sassenhagen and Draschkow [12]. However, our results coincide with the decreased long-distance coherence between the frontal and parietal-occipital lobes in Alzheimer's Disease, shown by Liu et al. based on functional MRI measurements [13].

A. Future work

Although, the computation in practice of the adjacency introduced for functional connectivity data was only presented for connectivity variables derived from EEG recordings, in theory, this adjacency can be generalized for connectivities derived from Magnetoencephalographic (MEG) or functional Magnetic Resonance Imaging (fMRI) data, or even source-estimates based on EEG and MEG. The main reason for this, is that the only data-specific requirement of the adjacency algorithm is the existence of some type of strictly positive distance metric defined between all points of measurement. The practical application of this method to functional connectivity variables extracted from MEG- or fMRI data, as well as source estimates should be examined as additional research.

There are also functional connectivity metrics which – besides the temporal coincidence – also estimate the direction of influence between two points of measurement in space. For these directional connectivity metrics, a new adjacency should be defined, which is also a possible target of future research.

REFERENCES

[1] E. Maris, R. Oostenveld, "Nonparametric statistical testing of EEG- and MEG-data", vol. 164, pp. 177-190, Journal of Neuroscience Methods, 2007. DOI: 10.1016/j.jneumeth.2007.03.024

[2] A. Gramfort, M. Luessi, E. Larson, D.A. Engemann, D. Strohmeier, C. Brodbeck, R. Goj, M. Jas, T. Brooks, L. Parkkonen, M.S. Hämäläinen, "MEG and EEG Data Analysis with MNE-Python", vol. 7, pp. 1-13, Frontiers in Neuroscience, 2013. DOI: 10.3389/fnins.2013.00267

[3] E.T. Bullmore, J. Suckling, S. Overmeyer, S. Rabe-Hesketh, E. Taylor, M. Brammer, "Global, voxel, and cluster tests, by theory and permutation, for a difference between two groups of structural MR images of the brain", vol. 18, pp. 32-42, IEEE Transactions on Medical Imaging, 1999. DOI: 10.1109/42.750253

[4] K. Friston, "Functional and effective connectivity in neuroimaging: A synthesis", vol. 2, pp. 56-48, Human Brain Mapping, 1994. DOI: 10.1002/hbm.460020107

[5] D. Rodinskaia, C. Radinski, J. Labuhn, "EEG coherence as a marker of functional connectivity disruption in Alzheimer's disease", vol. 2, pp. 100098, Aging and Health Research, 2022. DOI: 10.1016/j.ahr.2022.100098

[6] R. A. Clark, K. Smith, J, Escudero, A. Ibáñez, M. A. Parra, "Robust Assessment of EEG Connectivity Patterns in Mild Cognitive Impairment and Alzheimer's Disease", vol. 1, pp. N/A, Frontiers in Neuroimaging, 2022. DOI: 10.3389/fnimg.2022.924811

[7] B. Cea-Cañas, J. Gomez-Pilar, P. Núñez, E. Rodríguez-Vázquez, N. de Uribe, A. Díez, A. Pérez-Escudero, V. Molina, "Connectivity strength of the EEG functional network in schizophrenia and bipolar disorder", vol. 98, pp. 109801, Progress in Neuro-Psychopharmacology and Biological Psychiatry, 2020. DOI: 10.1016/j.pnpbp.2019.109801

[8] Zs. Fodor, A. Horváth, Z. Hidasi, A.A. Gouw, C.J. Stam, G. Csukly, "EEG Alpha and Beta Band Functional Connectivity and Network Structure Mark Hub Overload in Mild Cognitive Impairment During Memory Maintenance", vol. 13, pp. N/A, Frontiers in Aging Neuroscience, 2021. DOI: 10.3389/fnagi.2021.680200

[9] A. Miltiadous, K.D. Tzimourta, T. Afrantou, P. Ioannidis, N. Grigoriadis, D.G. Tsalikakis, P. Angelidis, M.G. Tsipouras, E. Glavas, N. Giannakeas, A.T. Tzallas, "A dataset of EEG recordings from: Alzheimer's disease, Frontotemporal dementia and Healthy subjects", vol. N/A, pp. N/A, OpenNeuro, 2023. DOI: 10.18112/openneuro.ds004504.v1.0.6

[10] A. Miltiadous, K.D. Tzimourta, T. Afrantou, P. Ioannidis, N. Grigoriadis, D.G. Tsalikakis, P. Angelidis, M.G. Tsipouras, E. Glavas, N. Giannakeas, A.T. Tzallas, "A Dataset of Scalp EEG Recordings of Alzheimer's Disease, Frontotemporal Dementia and Healthy Subjects from Routine EEG", vol. 8, pp. N/A, Data, 2023. DOI: 10.3390/data8060095

[11] P. Tass, M.G. Rosenblum, J. Weule, J. Kurths, A. Pikovsky, J. Volkmann, A. Schnitzler, H.-J. Freund, "Detection of $n : m$ Phase Locking from Noisy Data: Application to Magnetoencephalography", vol. 81, pp. 3291-3294, Physical Review Letters, 1998. DOI: 10.1103/PhysRevLett.81.3291

[12] J. Sassenhagen, D. Draschkow, "Cluster-based permutation tests of MEG/EEG data do not establish significance of effect latency or location", vol. 56, pp. N/A, Psychophysiology, 2018. DOI: 10.1111/psyp.13335

[13] Y. Liu, C. Yu, X. Zhang, J. Liu, Y, Duan, A. F. Alexander-Bloch, B. Liu, T. Jiang, E. Bullmore, "Impaired long distance functional connectivity and weighted network architecture in Alzheimer's disease", vol. 24, pp. 1422-1435, Cerebral Cortex, 2014. 10.1093/cercor/bhs410

Adversarial Localization Algorithms in Indirect Vehicle-to-Vehicle Communication

Levente Alekszejenkó, Tadeusz Dobrowiecki
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
{alelevente, dobrowiecki}@mit.bme.hu

***Abstract*—Communicating autonomous vehicles (CAVs) can obtain direct measurements from their sensors or indirectly receive them via Vehicle-to-Vehicle (V2V) communication. As the CAVs are expected to share a part of their measurements, it can naturally pose a privacy threat by possibly revealing the route of the sender vehicle. Consequently, we shall assess the risks of sharing a dataset that is a mixture of direct and indirect measurements. However, a wide variety of papers focus on localization attacks for direct measurements; incorporating indirect measurements opens a new horizon for these researches.**

In this paper, we analyze a couple of localization algorithms for mixture datasets with applicable performance metrics. We have evaluated the algorithms in an Eclipse SUMO-based simulation. We consider these results as the baseline of future research.

***Index Terms*—indirect measurements, V2V communication, localization attack, localization performance**

I. Introduction

With the emergence of communicating autonomous vehicles (CAVs), vehicle-to-vehicle (V2V) communication will open new horizons in crowdsensing. When two CAVs meet, they can exchange measurement data to support real-time decision-making and tactical or strategic planning. In this paper, we denote the sender vehicle as *Ego* and the receiver as *Alter*.

As these vehicles receive data from each other, their knowledge base is a mixture of their own *direct* measurements and *indirect* records received from fellow vehicles, see Fig. 1. However, exchanging data is fruitful for various use cases; it also poses security issues [1]. For example, an honest but curious *Alter* might conduct a passive localization or tracking attack against *Ego* by inferring its route according to the exchanged dataset.

This paper presents three heuristic adversarial localization algorithms that try to identify *Ego*'s route by analyzing the dataset it sent. We also define abstract, road-network independent performance metrics to evaluate the success rate of these algorithms. Consequently, we consider the results shown in this paper as baseline measures for qualitative analysis of privacy-preserving V2V data-sharing techniques.

This research was supported by the National Research, Development, and Innovation Fund of Hungary under Grant TKP2021-EGA-02, and the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.

Supported by the ÚNKP-23-3-II-BME-233 New National Excellence Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund.

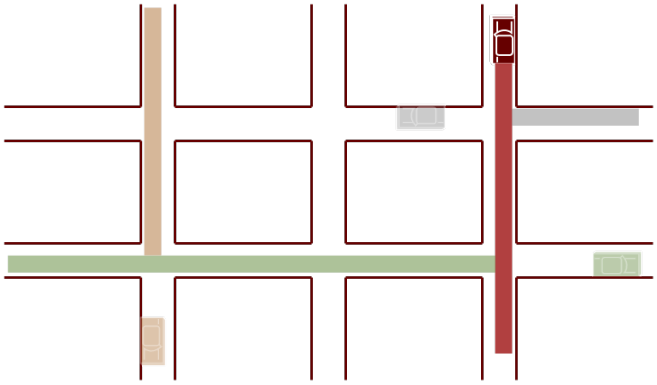


Fig. 1: Example of the mixed dataset collected by a vehicle. *Ego* is the dark red vehicle, its direct measurements are depicted as a dark red bar. Other bars represent the indirect measurements received from the gray and the green vehicles. The green vehicle also shared the maple-colored car's data with *Ego*.

A. Related Works

Preserving the privacy (i.e., the route) of CAV users is a severe security issue [2], especially in a computational fog environment [3]. However, the data exchange in vehicular communication is similar to traditional mobile crowdsensing [4]; V2V communication introduces a new communication layer. Hence, already existing privacy-preserving mechanisms [5] do not focus on securing this channel.

To fill this research gap, we shall take the following steps:

1) We shall define attacker models against the V2V crowdsourcing scheme.
2) Make countermeasures against localization attacks.
3) Assess the performance of the attacker to evaluate the proposed countermeasures.

This paper focuses on the 1) and 3) items and proposes a framework with comparative baseline measurements for analyzing countermeasures. For performance evaluation, many papers define privacy or trajectory similarity measures in the function of geographical distance [6], [7] although it might depend on concrete road networks and traffic scenarios. To mitigate possible side effects, we use abstract performance and similarity measures besides endpoint distances.

DOI https://doi.org/10.3311/MINISY2024-006

II. Problem Formulation

A tracking algorithm of the *Alter* vehicle is equivalent to the following challenge: *Ego* vehicle (being on the λ_e street) shares dataset $\mathcal{D}$ with *Alter*. *Ego* has already passed through an R route on a P set of (directed) streets. Given a finite map of a city as a directed graph $M(J, L)$ with J junctions and L streets, *Alter* has to label each $\lambda \in L$ street to indicate whether or not *Ego* has moved along that particular λ street. Therefore, *Alter's* challenge is a binary classification, in which $\widehat{I}(\lambda) = 1$ indicates *Ego* has moved along street λ, and $\widehat{I}(\lambda) = 0$ indicates that *Ego* has not been on street λ. The resulted $\widehat{P} = \{\lambda_i | \widehat{I}(\lambda_i) = 1, \text{for } \forall \lambda_i \in L\}$ estimate is the *localization attack* of *Alter* against *Ego*. We assume that *Alter* can only perform one single attack.

Dataset $\mathcal{D}$ consists of $\{\lambda, \tau, \nu\}$ triplets, where $\lambda \in L$ is the location of a measurement, τ is the measurement time, and ν is the measured value. Moreover, we assume that *Alter* knows the $\mathbf{T}$ transition matrix of which $\mathbf{T}[i, j] = \mathbb{P}(\overrightarrow{\lambda_i \lambda_j})$ element corresponds to the turning probabilities of the traffic for each $\lambda_i, \lambda_j \in L$ street pairs. Using this information, *Alter* can order a $\widehat{P}$ set to have a topologically ordered $\widehat{R}$ list of edges, which is the estimated route of *Ego*.

To evaluate such $\widehat{R}$ estimates, we can define performance metrics. There are three simple approaches to the definitions. The first idea is to measure the aerial distance $d(R_0, \widehat{R}_0)$ between the origins R_0 and $\widehat{R}_0$. In the case of localization, it is more meaningful to use aerial distance instead of driving distance, since it is symmetric, i.e., $d(R_0, \widehat{R}_0) = d(\widehat{R}_0, R_0)$.

Considering *Alter's* challenge is fundamentally a binary classification problem, we can use classical, confusion matrix-based performance metrics. For this, we define the number of *positive* samples as: $\mathrm{P} = |P|$, and negative samples as: $\mathrm{N} = |L \setminus P|$. Number of true-positives (TP) are the number of correctly identified streets along *Ego's* route:

$$\mathrm{TP} = \sum_{\lambda_i \in P} \widehat{I}(\lambda_i) \tag{1}$$

Similarly, false-positive (FP) are those samples that Alter mistakenly identifies as a part of *Ego's* route:

$$\mathrm{FP} = \sum_{\lambda_i \in L \setminus P} \widehat{I}(\lambda_i) \tag{2}$$

Following the traditional definition, the true-positive-rate (TPR) and false-positive-rate (FPR) is defined as:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} \tag{3}$$

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} \tag{4}$$

The third approach to evaluate *Alter's* performance originates in natural language processing (NLP). One can treat the $\lambda_i \in P$ streets, which *Ego* has visited, as letters; hence, we can consider R as a word. Consequently, *Alter's* $\widehat{R}$ route estimate is also a word. Since we aim to compare R to $\widehat{R}$, we can use traditional NLP similarity metrics to evaluate the accuracy of *Alter's* algorithm.

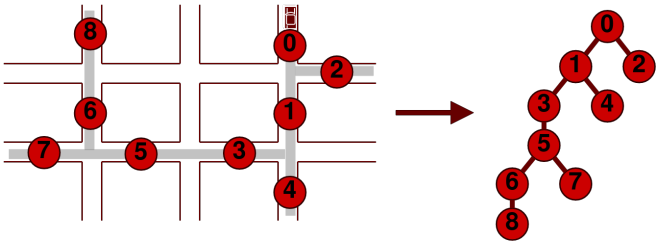


Fig. 2: Construction of the BFS–data-tree. (Edge directions are not shown.)

One of these similarity measures is the $\mathcal{J}(R, \widehat{R})$ Jaccard index [8]. In this paper, we define the Jaccard index as:

$$\mathcal{J}(R, \widehat{R}) = \frac{\mathrm{TP}}{\mathrm{P} + |\widehat{R}| - \mathrm{TP}}. \tag{5}$$

Another well-known similarity metric is the cosine similarity: $S_c(R, \widehat{R})$. In our case, the $S_c(R, \widehat{R})$ can be defined as:

$$S_c(R, \widehat{R}) = \frac{\mathrm{TP}}{\sqrt{\mathrm{P}} \cdot \sqrt{|\widehat{R}|}}. \tag{6}$$

III. Localization Attack Algorithms

We will discuss in the following how *Alter* can compute $\widehat{R}$, the estimate of *Ego's* route. However, more sophisticated algorithms could exist; we will define three simple methods in this paper.

A. The BFS-Tree

Given the $M(J, L)$ map and the $\mathcal{D}$ dataset, *Alter* can build a topologically sound tree from the $\lambda_e \in L$ position. As a first step, *Alter* shall restrict the search space only on those λ_i edges which are in the received $\mathcal{D}$ dataset: $P_d = \{\lambda_i | \forall \lambda_i \in \mathcal{D}\}$. We note there is no restriction on $\mathcal{D}$; therefore, P is not necessarily a subset of P_d, and vice versa. Hence, *Alter* shall build a tree on the filtered $M(J, L \cap P_d)$ map graph. Secondly, *Alter* shall construct the $M(J, \widehat{L \cap P_d})$ reverse of the map. It can be calculated by the Bayes rule, by taking the $\mathbf{T}^\top$ transpose of the original transition matrix and normalizing it row by row. The resulted $\mathbf{Q}$ matrix is the so-called time-reversed version of the original $\mathbf{T}$ transition matrix, similarly to [9]. While the $\mathbf{T}$ matrix describes which are the λ_j possible next streets to go to for a CAV being on the λ_i street; $\mathbf{Q}$ represents which are those λ_j streets from which a CAV could come if it is currently on street λ_i.

Hence, the $\mathcal{F}$ *data tree* can be constructed by the search tree of a breadth-first-search [10] on the $M(J, \widehat{L \cap P_d})$ graph from the λ_e root as illustrated in Fig. 2. After obtaining the $\mathcal{F}$ BFS–data-tree, *Alter* has to mark a path in this tree that will be its $\widehat{R}$ estimate. The following algorithms describe three ways to label such a path.

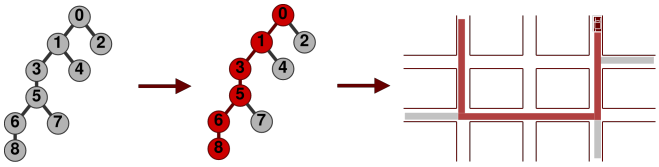


Fig. 3: An example of the *river* algorithm.

B. River Algorithm

Selecting the longest path, i.e., the highest subtree in $\mathcal{F}$, might be the most natural $\widehat{R}$ estimate. One can assume that *Ego's* movement connects the indirect measurements in the $\mathcal{D}$ dataset. Hence, *Ego's* R route shall form the *trunk* of the $\mathcal{F}$ tree. This *trunk* is a path from the root to one of the deepest leaves. As the resulting algorithm is similar to the naming traditions of rivers in Geography, this algorithm is called the *river alogrithm*.

Algorithm 1 presents, and Fig. 3 illustrates the behavior of the algorithm.

Algorithm 1: River Algorithm.

```
input: F, λ_e
output: R̂

function get_highest_subtree(λ)
begin
    if λ ∈ leaves(F) then:
        return λ, 1
    else:
        t ← [], h ← 0
        foreach λ_i ∈ children(λ):
            t_λi, h_λi ← get_highest_subtree(λ_i)
            if h_λi > h then:
                t ← t_λi, h ← h_λi
    return insert(t, λ), h + 1
end

begin
    R̂, h ← get_highest_subtree(λ_e)
    return R̂
end
```

C. Time-bounded River Algorithm

Ego's direct measurements, besides possibly forming the tallest subtree $\mathcal{F}$, are ordered correctly by its movement both in space and time. Although, the *river* algorithm only calculates with locations. To also handle the time of the measurements, we must modify the *river* algorithm.

The first modification is to add time label τ_i to each $\lambda_i \in \mathcal{F}$ node, see Fig. 4. However, containing direct and also indirect measurements, $\mathcal{D}$ may contain more time entries for a λ_i location. Direct measurements of *Ego* are likely the most recent ones[1]; hence, in the labeling, we will choose the latest possible τ_i value. Let us denote the latest time entry from a λ_i street $\tau(\lambda_i)$.

Secondly, with the $\tau(\lambda_i)$ function, we will make the following restriction in the highest subtree searching subroutine. If λ_i is the parent of λ_j in $\mathcal{F}$ then $\tau(\lambda_i) \geq \tau(\lambda_j)$ shall be satisfied.

[1]However, sometimes *Ego* functions like a transmitter, see Fig. 4. In such cases, *Ego* can have indirect measurements that are more recent than its direct ones.

With this modification, we can modify the *river* algorithm to obtain the *time-bounded river algorithm (triver)* as presented in Algorithm 2.

Algorithm 2: Time-bounded River Algorithm.

```
input: F, λ_e, τ(·)
output: R̂

function get_highest_timed_subtree(λ)
begin
    if λ ∈ leaves(F) then:
        return λ, 1
    else:
        t ← [], h ← 0
        foreach λ_i ∈ children(λ) ∧ τ(λ) ≥ τ(λ_i):
            t_λi, h_λi ← get_highest_timed_subtree(λ_i)
            if h_λi > h then:
                t ← t_λi, h ← h_λi
    return insert(t, λ), h + 1
end

begin
    R̂, h ← get_highest_timed_subtree(λ_e)
    return R̂
end
```

D. Minimal Probability Route Algorithm

We may observe that the previously presented *river* and *triver* algorithms suppose that *Ego* has already traveled a long route, which characterizes the trunk of $\mathcal{F}$. However, *Ego* can come from a rarely-visited nearby street. This case is quite interesting because the (reason of) uniqueness of *Ego's* data varies along its R route. Let us assume the following scenario of Fig. 5: *Ego* comes from a dead-end street being in communication range to an arterial way. After departing, *Ego* goes through some local roads. Here, having data from the dead-end street might not be unique as many vehicles can be close enough to have measurements from this part of the road network. After that, *Ego* turns into a collector way. On the collector way, vehicles rarely have data from the dead-end street, as they usually do not visit its neighborhood. Therefore, *Ego* has some unique data records. Finally, after joining the traffic of the arterial way, having data from the dead-end street is normal as it is in communication range. However, *Ego's* dataset is unique as most CAVs on the artery do not have data from the local streets out of the communication range.

After receiving $\mathcal{D}$, *Alter* can reason that if *Ego* has some records from an *unlikely* street, its origin might be this particular street. However, we shall define *unlikely* street carefully. Having measurements from the local roads of the above example is *unlikely* for a vehicle moving on the artery. But it is *not* among fellow vehicles on the local roads.

To express the (un)likelihood of a λ street, we can use the $\mathbf{Q}$ time-reversed transition matrix. We know that *Ego* is currently on the λ_e street. Let us denote π_s the vector that indicates the position of *Ego* $s = \{0, 1, 2, \dots\}$ steps ago[2]. Hence, the π_0 vector is a deterministic unit vector of which e^{th} coordinate is 1.0, indicating *Ego* is currently on the λ_e street. For any $s \geq 1$, the i^{th} coordinate of the π_s vector expresses the probability

[2]In this paper, s was limited to the $0 \geq s < 20$ range, as longer routes were unlikely.

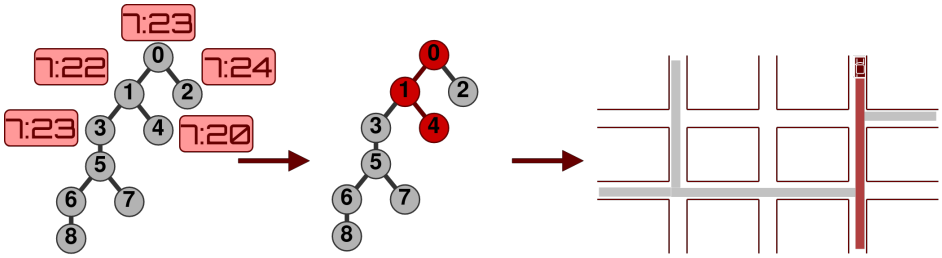


Fig. 4: An example of the *time-bounded river (triver)* algorithm.

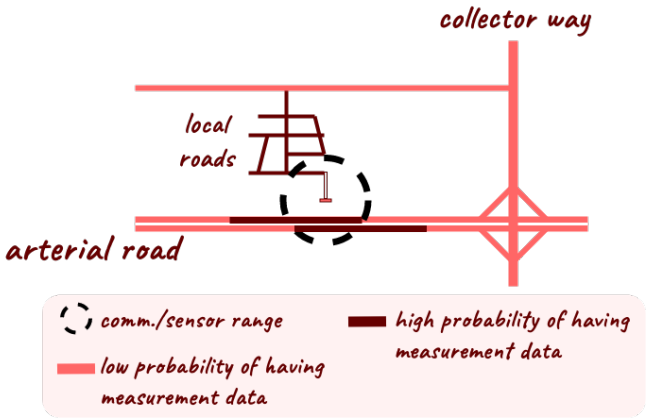


Fig. 5: Road network with typical hierarchy. The *Ego* vehicle can start from a dead-end road, and after traveling through some local and collector ways, it joins the traffic in the arterial road. Along the way, there are different possibility that the vehicles in the traffic have data from the dead-end street.

that *Ego* was on the λ_i street s steps ago. Using $\mathbf{Q}$, *Alter* can calculate this probability as:

$$\pi_s = \pi_0 \mathbf{Q}^s \qquad \forall s \in \{1, 2, 3, \dots\}. \tag{7}$$

For a particular $\lambda_i \in P_d$ street, *Alter* can calculate its $\delta(\lambda_i)$ depth from the root in the $\mathcal{F}$ data tree. As *Alter* assumes that *Ego* has a (bounded) rationality in route planning, this depth corresponds to the s steps that *Ego* had to move between the λ_i and λ_e streets.

For a more compressed expression, let us denote $p\big(\lambda_i, \delta(\lambda_i)\big)$ the probability that *Ego* was on the λ_i given λ_i is in $\delta(\lambda_i)$ depth in $\mathcal{F}$.

Therefore, *Alter* can label each $\lambda_i \in \mathcal{F}$ edges with corresponding probabilities: $p\big(\lambda_i, \delta(\lambda_i)\big)$. The *minimal probability route (minp)* algorithm will select the least probable $\lambda_{\min_p} = \arg\min_{\lambda_i} p\big(\lambda_i, \delta(\lambda_i)\big)$ edge in $\mathcal{F}$, and simply returns the way from the λ_e root to $\lambda_{\min_p}$ as the $\widehat{R}$ route estimate, see Algorithm 3 and Fig. 6.

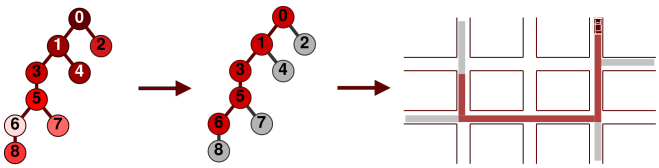


Fig. 6: An example of the *minimal probability route (minp)* algorithm. Vertices are colored following the *prior* probability (darker is more probable).

Algorithm 3: Minimal Probability Route Algorithm.

```
input: F, λ_e, p(·,·)
output: R̂

begin
    R̂ ← []
    λ ← argmin_{λ_i ∈ F} p(λ_i, δ(λ_i))
    insert(R̂, λ)

    while λ ≠ λ_e
    begin
        λ ← parent(λ)
        insert(R̂, λ)
    end

    return R̂
end
```

IV. Evaluation

To evaluate the localization algorithms, we ran microscopic traffic simulation in Eclipse SUMO [11]. During the simulations, we collected the *meeting vehicles*. As 50 m is a range in which no V2V communication methods drop performance significantly [12], we defined *meetings* as a 50 m communication range. Consequently, vehicles closer to each other than 50 m were the *meeting vehicles*.

Each car was a CAV; hence, upon meetings, every vehicle shared the most recent part, i.e., 90 s from its $\mathcal{D}$ dataset to ensure the ability to run the simulations on an average PC equipped with 16 GiB of RAM. As a constant exchange of measurement data would be unrealistic between CAVs, at least 180 s shall elapse between two data exchanges of a pair of CAVs.

For performance analysis of the localization algorithms, each vehicle ran them after data exchanges.

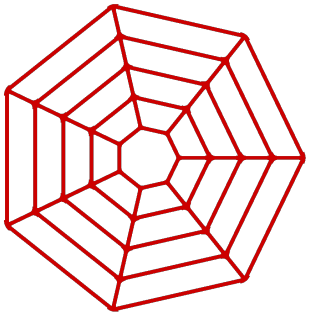


Fig. 7: The road network of the simulation scenario.

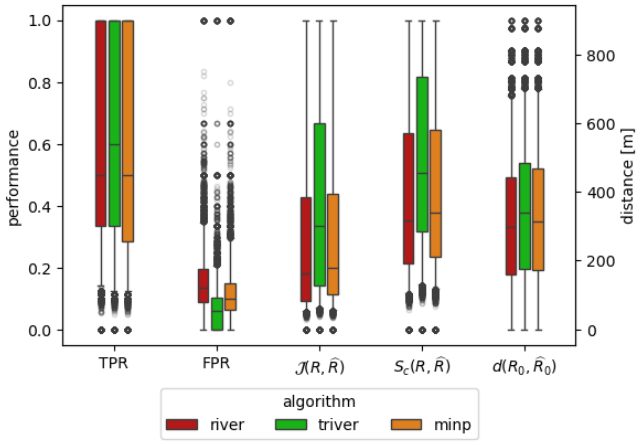


Fig. 8: Performance comparison of the localization algorithms. Whiskers indicate the 5$^{\text{th}}$ and 95$^{\text{th}}$ percentile.

A. Simulation Scenario

A concrete road network and traffic demand might influence the obtained results; hence, we created an abstract traffic scenario. The used network was spider web shaped with 7 arms and 5 rings, see Fig. 7. The longest direct driving distance in this network is slightly more than 1000 m. Therefore, one might interpret this network as an abstract model of a central business district in a city.

We generated traffic for the scenario in the following manner: 2400 CAVs depart randomly within 4 hours. The vehicles have a random origin street, which is also their destination. Each CAV has to stop for 30 min in a randomly chosen parking lot. We specified this parking lot in design time; however, this might not be free when a particular CAV arrives. In this case, Eclipse SUMO helps the CAV find a new parking lot in the neighborhood by the so-called parking lot rerouters. After 30 min of parking, the vehicles return to their origin.

The CAVs measure the occupancy of the parking lot while they are moving. We assume a vehicle can directly measure this occupancy rate when it is no more than 50 m away from the parking lot, which seems to be a realistic visual, or a reliable communication range.

B. Results

After simulations, we can analyze the obtained results. Despite P $= 1$ seeming to be a trivial case, it is not: *Ego* might have exchanged data right after it had started moving. Hence, we cannot filter out this case. Furthermore, *Ego* occasionally meets *Alter* before *Ego* has measurements from a given street (*Ego* is close to *Alter* but outside the 50 m range of the parking lots). It can result in TPR=0.0 values; see Fig. 9.

According to Fig. 9a and Fig. 9c, the *river* and *minp* algorithms have similar performance. It was expected because if $d(\lambda_e,\lambda_i) < d(\lambda_e,\lambda_j)$ then usually $p\big(\lambda_i,\delta(\lambda_i)\big) > p\big(\lambda_j,\delta(\lambda_j)\big)$. Hence, further away origins are also more unlikely, resulting in similar trajectory estimates. In terms of FPR, *minp* is slightly better than the *river* algorithm, see Fig. 8, which might be caused by the rare cases when $d(\lambda_e,\lambda_i) < d(\lambda_e,\lambda_j)$ but $p\big(\lambda_i,\delta(\lambda_i)\big) < p\big(\lambda_j,\delta(\lambda_j)\big)$; when the origin of the longest route is not as unlikely as another $\lambda_i \in \mathcal{F}$ street, similarly to the example in Fig. 3 and Fig. 6.

TABLE I: Average length of $\widehat{R}$ route estimate

algorithm	average predicted length [#streets]
river	7.99
triver	5.02
minp	6.41

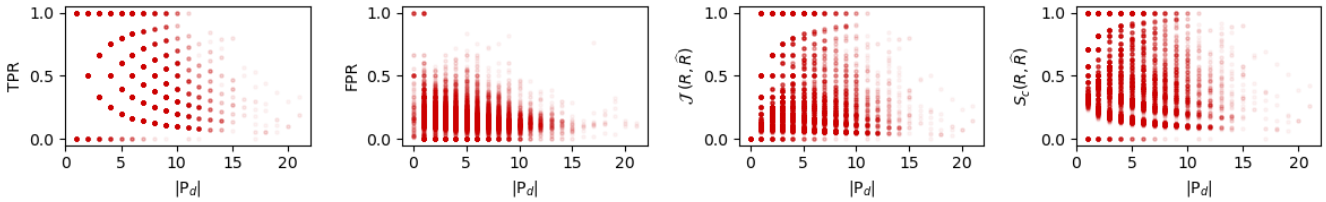
As the *triver* algorithm outperforms the *river* and *minp* in each metric, see Fig. 8, we suppose that choosing the trajectory of most recent measurements is indeed a good heuristic. The *triver* algorithm gives a significantly lower false-positive rate, see Fig. 9b, and a slightly higher true-positive rate, see Fig. 8 compared to other algorithms. As it also has significantly better performance in similarity metrics $\mathcal{J}(R,\widehat{R})$ and $S_c(R,\widehat{R})$, a slightly worse $d(R_0,\widehat{R}_0)$ distance result seems to be a paradox. Considering that *triver* tends to predict shorter routes compared to the other two methods; see Table I, we can conclude that *triver* offers a briefer but better quality $\widehat{R}$ route estimate. However, this estimate might not be long enough to be close to the true origin of *Ego*.

Finally, Fig. 8 illustrates that the raw $d(R_0,\widehat{R}_0)$ distance between the origin of the true and the predicted trajectories has a limited algorithm ranking capability: All three presented algorithms achieve approximately the same results in terms of $d(R_0,\widehat{R}_0)$ distance. In the meantime, the quality and the size of the inferred trajectories can differ significantly.

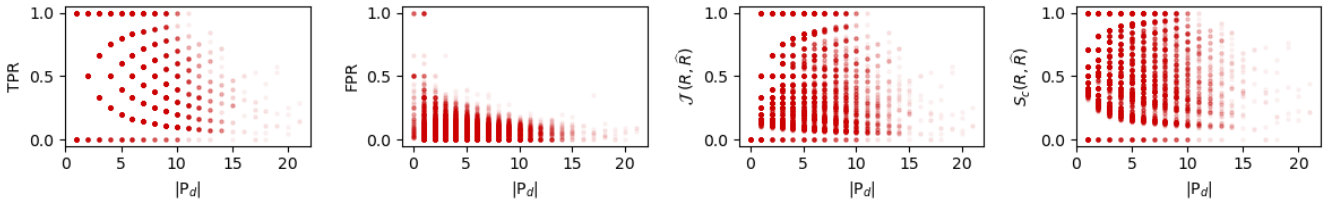
V. Conclusion

This paper *presented a general framework for assessing location privacy loss in V2V measurement data exchange*. To quantify this privacy loss, we used several metrics reflecting the quality of an adversarial trajectory reconstruction based on the exchanged data.

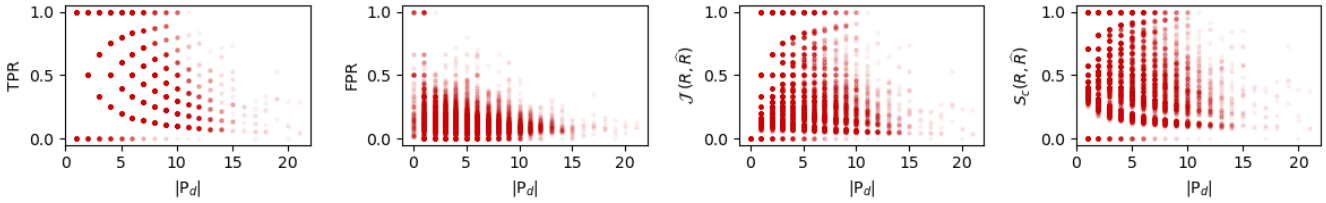
As someone's location and route are sensitive information, we shall protect it from unauthorized access. However, there exist network or data-link level privacy-preserving V2V communication techniques; an adversarial *Alter* might run



(a) Performance of the *river* algorithm.



(b) Performance of the *triver* algorithm.



(c) Performance of the *minp* algorithm.

Fig. 9: Performance of the localization algorithms in the function of the size of the P set. The *triver* algorithm provides significantly better results in the FPR metric compared to the *river* and *minp* algorithms. On the other hand, the *triver* algorithm achieves lower performance in TPR when the size of the P set is higher. Moreover, the *river* algorithm sometimes provides better $\mathcal{J}(R,\widehat{R})$ and $S_c(R,\widehat{R})$ similarities around $|P_d| = 15$ than the *minp* algorithm.

application-level inference software to track the *Ego* vehicle. In such case, as Fig. 8 shows, even *simple passive heuristic tracking algorithms can be successful in localization attacks against an unprotected sender.*

Consequently, we shall make countermeasures against such an information leakage. In our future research, we aim to develop an application-level solution that is location-secure by design and to evaluate it in more complex, realistic scenarios.

References

[1] H. A. Ameen, A. K. Mahamad, B. B. Zaidan, *et al.*, "A deep review and analysis of data exchange in vehicle-to-vehicle communications systems: Coherent taxonomy, challenges, motivations, recommendations, substantial analysis and future directions," *IEEE Access*, vol. 7, pp. 158349–158 378, 2019. DOI: 10.1109/ACCESS.2019.2949130.

[2] B. Mokhtar and M. Azab, "Survey on security issues in vehicular ad hoc networks," *Alexandria Engineering Journal*, vol. 54, no. 4, pp. 1115–1126, 2015, ISSN: 1110-0168. DOI: 10.1016/j.aej.2015.07.011.

[3] T. Limbasiya, K. Z. Teng, S. Chattopadhyay, and J. Zhou, "A systematic survey of attack detection and prevention in connected and autonomous vehicles," *Vehicular Communications*, vol. 37, p. 100 515, 2022, ISSN: 2214-2096. DOI: 10.1016/j.vehcom.2022.100515.

[4] D. Suhag and V. Jha, "A comprehensive survey on mobile crowdsensing systems," *Journal of Systems Architecture*, vol. 142, p. 102 952, 2023, ISSN: 1383-7621. DOI: 10.1016/j.sysarc.2023.102952.

[5] J. W. Kim, K. Edemacu, and B. Jang, "Privacy-preserving mechanisms for location privacy in mobile crowdsensing: A survey," *Journal of Network and Computer Applications*, vol. 200, p. 103 315, 2022, ISSN: 1084-8045. DOI: 10.1016/j.jnca.2021.103315.

[6] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux, "Quantifying location privacy," in *2011 IEEE Symposium on Security and Privacy*, 2011, pp. 247–262. DOI: 10.1109/SP.2011.18.

[7] Y. Tao, A. Both, R. I. Silveira, *et al.*, "A comparative analysis of trajectory similarity measures," *GIScience & Remote Sensing*, vol. 58, no. 5, pp. 643–669, 2021. DOI: 10.1080/15481603.2021.1908927.

[8] P. Jaccard, "The distribution of the flora in the Alpine Zone.1," *New Phytologist*, vol. 11, no. 2, pp. 37–50, 1912. DOI: 10.1111/j.1469-8137.1912.tb05611.x.

[9] L. Alekszejenkó and T. Dobrowiecki, "Privacy-aware methods for data sharing between autonomous vehicles," in *IFAC-PapersOnLine*, vol. 55, 2022, pp. 7–13. DOI: 10.1016/j.ifacol.2022.07.575.

[10] E. F. Moore, "The shortest path through a maze," in *Proceedings of the Internation Symposium on the Theory of Switching*, Harvard University Press, 1959, pp. 285–292.

[11] P. A. Lopez, M. Behrisch, L. Bieker-Walz, *et al.*, "Microscopic traffic simulation using SUMO," in *The 21st IEEE International Conference on Intelligent Transportation Systems*, IEEE, 2018. [Online]. Available: https://elib.dlr.de/127994/.

[12] E. Moradi-Pari, D. Tian, M. Bahramgiri, S. Rajab, and S. Bai, "DSRC versus LTE-V2X: Empirical performance analysis of direct vehicular communication technologies," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 5, pp. 4889–4903, 2023. DOI: 10.1109/TITS.2023.3247339.

Conditional Molecule Generation with 2D Latent Diffusion

Dániel Szarvas, Domonkos Pogány
Budapest University of Technology and Economics
Department of Measurement and Information Systems
Budapest, Hungary
Email: daniel.szarvas7@edu.bme.hu, pogany@mit.bme.hu

***Abstract*—Diffusion-based generative deep neural networks have achieved excellent results in sound and image generation, with promising applications in various fields, including drug discovery. In addition to already existing 3D approaches, we propose a new method, MoLD (Molecular Latent Diffusion) that utilizes 2D latent diffusion to achieve both unconditional and conditional molecule generation. With both sampling methods, the model is able to create numerous novel and valid molecules not present in the training dataset. Furthermore, comparing the distribution of unconditionally and conditionally sampled molecules based on the control property indicates that the diffusion model effectively influences the molecule formation process. The effectiveness of our approach demonstrates that the conditional molecule generation process can be modularly modified by substituting or retraining only the diffusion network responsible for conditioning. This modification can be achieved without restructuring the latent space, leaving the possibility for further research open.**

***Index Terms*—conditional generation, de novo molecule generation, latent diffusion, transformer, variational autoencoder**

I. Introduction

In recent years, the rapid development of machine learning fueled by technological advances has paved the way for numerous new use cases and interdisciplinary research opportunities. Great examples of these emerging techniques are diffusion-based neural networks, which have provided great results in conditional content generation [18], and their application in drug discovery awaits further research.

The task of active pharmaceutical ingredient generation has already been explored using various deep neural networks in the past. Previous solutions include various graph neural networks [2], [3] that work with the graph representation of molecules, and variational autoencoders [1] that use string representations of molecules as training data.

Diffusion-based methods have demonstrated considerable success in this task, primarily employing 3D point clouds [5]. Additionally, 3D latent diffusion [18] models has been introduced to enhance previous methods [9].

Our method utilizes the SMILES [17] representation of molecules instead of graph or 3D coordinate formats to learn the latent space using a transformer-based variational autoencoder [1], while the sampling itself is performed by diffusion. The innovation in our approach lies in the utilization of 2D latent diffusion rather than its 3D counterpart already used in multiple solutions. Our objective was to evaluate the possibility of providing guidance solely through the diffusion model, thereby exploring the potential of utilizing diffusion as a substitutable conditioning plugin module [8] for conditional molecule generation.

The project 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. The research was also supported by the National Research, Development, and Innovation Fund of Hungary under Grant TKP2021-EGA-02, and the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.

II. Background

The latent diffusion concept [18] is comprised of two main components: an autoencoder network to provide a gateway between the real and latent spaces and a diffusion model that's responsible for the generative process in the latent space.

A. Transformer-based Variational Autoencoder

To implement 2D latent diffusion, a specialized autoencoder is necessary. It has to be capable of producing spatially coherent 2D latent representations for the diffusion model.

Basic autoencoders (AE) [15] established the concept of utilizing an informational bottleneck as a means of consistently compressing real space data x into a lower dimensional form z. As a result, they have proven to be efficient in data reconstruction, however, they lack a continuous latent space.

Variational autoencoders (VAE) [12] were introduced to improve upon the latent space regularization of basic AE models. This is achieved by applying a prior Gaussian distribution to the latent representations with a unique technique, which is commonly referred to as the *reparametrization trick*. Due to their significant improvement of latent space continuity, VAE models enhance the decoding capabilities, resulting in a more robust model, thus establishing their effectiveness as autoencoders for denoising diffusion models. Their training loss using the evidence lower bound (ELBO) is formed as

$$L_{VAE} = \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x|z)\right] - \beta D_{KL} \left(q_\phi(z|x)||p(z)\right), \tag{1}$$

where the first part is the reconstruction loss ($p_\theta(x|z)$ as decoder output), and the second part is responsible for the regularization by the Kullback–Leibler divergence (D_{KL}) of the encoder output ($q_\phi(z|x)$) from the Gaussian prior ($p(z)$). To balance between the model's reconstruction and generalization capabilities, the β-VAE concept [14] has been introduced as a multiplier (β) for the regularization part to enable relative scaling of the two terms.

Upon the previously mentioned techniques, we utilized the Transformer-based VAE (TVAE) [1] architecture, which builds on the well-known Transformer [16] and attaches the *reparametrization trick* along with a bottleneck after its encoder. Following the tokenization and embedding of SMILES representations, it employs self-attention to gain deeper insight into their structure. Another valuable trait of this architecture is that the encoder's output is a list of token embeddings passed through the bottleneck, which then becomes sufficient 2D input for latent diffusion.

B. Diffusion Model

After earlier attempts to utilize the concept of nonequilibrium thermodynamics in unsupervised learning tasks [13], the breakthrough came with the introduction of DDPMs (Denoising Diffusion Probabilistic Models) [4].

The method has two distinct components. The first is a fixed noising process (q) used in training that applies Gaussian noise to the representation gradually. The other is the generative denoising process (p_θ), which starts from pure noise and removes some noise step-by-step until it reaches a clear representation. Since in both probabilistic processes, the next state only depends on the current one, they can be interpreted as Markov chains of length T, where our goal is to learn p_θ.

The underlying neural network of the DDPM sampling method is the U-Net [6], an autoencoder-like deep neural network that has achieved great results in medical imaging tasks. Its task is to predict the noise on a given representation in a given timestamp ($\epsilon_\theta(z, t)$, where $\{t \in \mathbb{Z} | 1 \leq t \leq T\}$). The loss function for the latent diffusion can be formulated as

$$L_{Diff} = \mathbb{E}_{z,\epsilon\sim\mathcal{N}(0,1),t} \left\| \epsilon - \epsilon_\theta(z_t, t) \right\|_2^2, \tag{2}$$

which is the mean squared error of the actual noise (ϵ) and the predicted noise ($\epsilon_\theta(z, t)$). The forward process is predefined. Therefore, z_t can be easily observed during training by transforming the input (x) with the VAE's encoder.

C. Conditioning Mechanism

The goal of conditioning latent diffusion is to predict the distribution of z_t based on y guidance [21], formalized as $p_\theta(z_t|y)$. Updating the weights during training requires calculating $\nabla_{z_t} \log p_\theta(z_t|y)$, therefore, we can write the Bayes' theorem for this conditional distribution in logarithmic form and calculate its partial derivative with respect to z_t to see the objective as

$$\nabla_{z_t} \log p_\theta(z_t|y) = \nabla_{z_t} \log p_\theta(y|z_t) + \nabla_{z_t} \log p_\theta(z_t). \tag{3}$$

For conditioning our latent diffusion model, we chose the classifier-free guidance (CFG) technique [7]. This approach allows the model to condition the generative process without relying on external classifiers. By utilizing conditioning dropout, the model is trained unconditionally for a certain percentage of the training time (10-20%), thereby enabling it to perform both conditional and unconditional sampling.

To achieve this, the Bayes' theorem can be rearranged for $p_\theta(y|z_t)$. After applying the logarithmic transformation and calculating the partial derivative, it can be inserted into Eq. 3, resulting in the corresponding gradient for updating the weights to be formalized as

$$\nabla_{z_t} \log p_\theta(z_t|y) = (1-\gamma)\nabla_{z_t} \log p_\theta(z_t) + \gamma\nabla_{z_t} \log p_\theta(z_t|y), \tag{4}$$

where γ is called *CFG-scale*, and is responsible for controlling the guidance strength. Setting this to a value greater than 1 can magnify the effect of the guidance and has produced exceptional results in image generation [7].

III. Method

Our latent diffusion model consists of a TVAE serving as its encoder-decoder network and DDPM as its sampling method to provide a way for conditional generation.

The TVAE has three stacked layers of encoders and decoders using the attention mechanism, transforming the tokenized SMILES representations into 2D latent representations with a size of 64×8. These encoded molecules form the latent space within which the diffusion operates.

As for the DDPM, we chose a relatively small $T = 300$ as the sampling timestep limit to avoid longer sampling times. Our model's U-Net implementation comprises four layers instead of the original five. This adjustment is due to the insufficient size of latent representations to accommodate another layer of downsampling. We ensured through testing that the deviation from the original architecture doesn't affect the model's denoising capabilities in any significant way.

To implement the CFG technique in our model, we introduced the τ_θ fully connected network to create embedded conditioning vectors matching the dimensionality of the layers' time embeddings (see Fig. 1). The guidance is provided to the U-Net by adding these conditioning vectors to the time embedding vector used inside the U-Net layers for informing the model about both the guidance and the state of sampling. The four conditioning factors in our model include *QED* (Quantitative Estimate of Druglikeness) [10], *SA Score* (Synthetic Accessibility Score) [11], *logP* and *molecular weight*.

During the training process, SMILES representations were initially tokenized, then embedded, and subsequently transformed into the latent space using the chosen autoencoder's encoder. Varying levels of Gaussian noise were then added using the fixed forward diffusion process, with target timestamp t uniformly sampled from $\{1, \ldots, 300\}$. Following that, the U-Net attempted to predict the original latent representation for all noisy representations. Lastly, the autoencoder's decoder

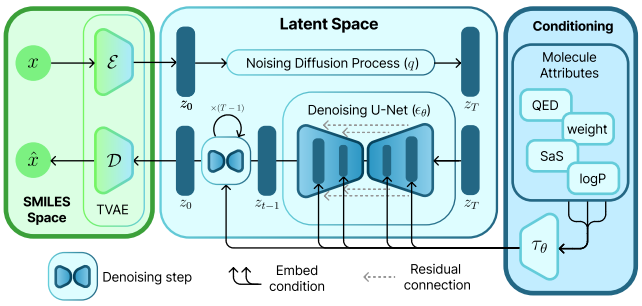


Fig. 1. Full architecture with conditioning capabilities. The TVAE serves as the gateway between SMILES and latent space, where the DDPMs generative process takes place. Conditioning is achieved by embedding conditioning vectors within the U-Net's inner representations (see Sec. III).

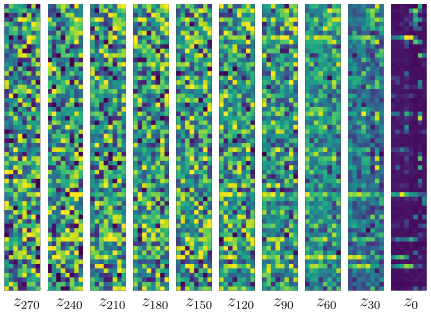


Fig. 2. Molecule sampling with generative diffusion inside the AE model's latent space, where z_0 denotes the denoised result. The DDPM utilized was trained on molecule latent representations.

transformed the result back to the original space, to then aggregate the losses and propagate them back to the corresponding weights. For scheduling the learning rate during training we used the Noam [16] technique.

We utilized the ZINC dataset [19] for training the model, which is comprised of 1.760.888 valid molecules. The model training generally lasted for 100 epochs unless the loss converged earlier or the model's output collapsed. The procedure was conducted on a 12 GB NVIDIA Titan X. The training of the final conditional model lasted two days (64 epochs).

IV. Results

The results of both unconditional and conditional sampling are presented in Tab. I, with our model referenced as *MoLD*. In each case, a total of 10.000 molecules were sampled to provide a basis for comparison with other models. The models are contrasted by the proportion of valid molecules (*Validity*), the rate of valid molecules not present in the training set (*Novelty*), and the portion of unique molecules (*Uniqueness*).

A. Unconditional Molecule Generation

Our objective was to explore whether sampling novel molecules with latent diffusion is feasible. In our initial experiments for unconditional molecule generation, we first measured only the TVAE's unconditional generative performance on the basic evaluation metrics, to compare it with the results of the complete model later (see Tab. I).

In our first experiment with diffusion, we employed a Transformer-based AE, essentially a TVAE without the *reparametrization trick*, as the latent diffusion's encoder-decoder model. While the diffusion captured certain patterns of latent representations during training (see Fig. 2), the model struggled to generate a substantial number of novel molecules.

Then, we proceeded to assemble the complete architecture with a regular TVAE (see Fig. 1), still without the conditioning components and attempted to train the model, experimenting with various training techniques. We first tried training the whole model at once with random weight initialization, but achieved little success as the latent representations faded.

The next concept we endeavored was to initialize the weights of the autoencoder to a pre-trained TVAE model's weights, which already had a regularized latent space. We then further trained it together with the diffusion model, which resulted in a stable latent space where the molecule generation was handled by diffusion.

In a separate experiment, we kept the weights of the pre-trained TVAE model fixed during training. This configuration, where gradient descent only optimized the weights of the diffusion model, yielded the best results. The rate of valid molecules surpassed the one observed when sampling with only the TVAE, as shown in Tab. I.

B. Conditional Molecule Generation

Building on the success of our most promising unconditional model, which utilized fixed, pre-trained TVAE as its autoencoder, we refined this approach for conditional molecule generation, as outlined in Sec. III. The training technique remained consistent with the most successful unconditional model, with the only difference being the inclusion of molecular properties during the training process.

To validate the accuracy of conditional molecule generation, we performed a conditional sampling of 10.000 valid molecules with various conditions, including both low and high values for each conditioning property. An equivalent number of unconditional samples were also generated. The resulting distributions were then plotted based on the selected property. We then tested whether their conditional distributions significantly differed from the distributions of the unconditionally generated molecules (see Fig. 3).

The conditional distributions on *molecular weight* and *logP* demonstrate a clear translation in the respective direction of the guidance, while the more complex *QED* and *SA Score* require evidence that their means notably differ in the relevant direction. As these distributions tend to approximate Gaussians with unequal variances, Welch's t-test can determine whether their means demonstrably differ. Regarding *QED*, the comparison between the conditional and unconditional samples demonstrated p-values significantly lower than the 0.05 threshold ($4*10^{-44}$ for low, $8*10^{-9}$ for high *QED* value). With analogous calculations, *SA Score* conditioned samples

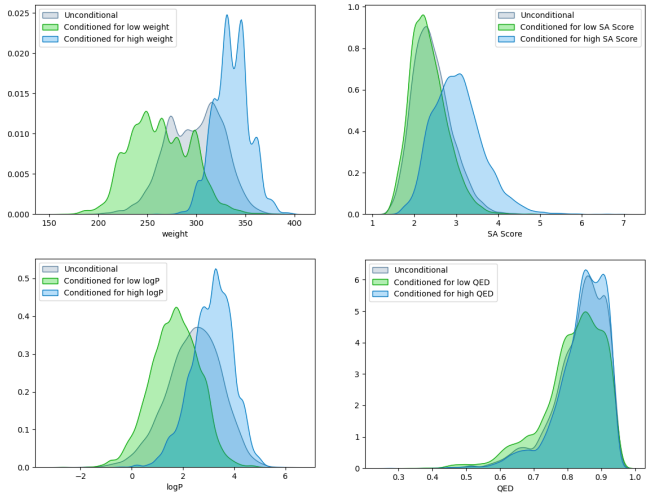


Fig. 3. Distributions of 10.000 unconditionally and conditionally sampled valid molecules plotted based on the conditioning property.

TABLE I
PERFORMANCE OF STATE-OF-THE-ART MODELS AND MOLD ON 10.000 SAMPLED MOLECULES.

	Model	Validity	Novelty	Uniqueness
Uncond.	GraphVAE [2]	0.557	0.760	0.616
	GTVAE [3]	0.746	0.225	0.942
	Transformer VAE [1]	0.567	**0.996**	-
	MoLD (TVAE)	0.773	0.768	**1.000**
	MoLD (TVAE+Diff)	**0.885**	0.543	0.634
Cond.	GraphVAE [2]	0.565	**0.598**	**0.314**
	MoLD (TVAE+Diff)	**0.863**	0.219	0.267

also produced p-values below the threshold ($7*10^{-58}$ for low, negligible for high *SA Score*).

From these results we can say with certainty that the conditional samples' means differ in a significant way from the unconditional samples in the correct direction, validating the conditioning capabilities of the diffusion model.

V. CONCLUSION

Overall, our conditional model still has room to improve in generating novel molecules, but it demonstrates strong performance in terms of their validity compared to other models (see Tab. I). As shown by the results, the model was successfully conditioned exclusively through diffusion (see Fig. 3). Consequently, there is a potential to utilize the same autoencoder while replacing the guiding diffusion network. This allows the diffusion model to function as a modular plugin for introducing various conditioning mechanisms while maintaining an unchanged latent space [8]. Moreover, diffusion-based sampling establishes the foundation for the model to handle complex multi-objective conditions in the future with minimal effort, requiring further research.

Possible ways for improvement include introducing DDIM (Denoising Diffusion Implicit Model) sampling [20] for accelerated sample times or integrating self-attention blocks within the U-Net to accomplish conditioning through them [18].

ACKNOWLEDGMENT

The authors would like to acknowledge Bence Bolgár for the initial project idea and assistance during its early phase. Computational resources for this project were provided by the Department of Measurement and Information Systems at Budapest University of Technology and Economics.

REFERENCES

[1] O. Dollar, N. Joshi, D. A. C. Beck, and J. Pfaendtner. Attention-based generative models for de novo molecular design. *Chem. Sci.*, vol. 12, no. 24, pp. 8362–8372, 2021. doi:10.1039/D1SC01050F.

[2] M. Simonovsky and N. Komodakis. GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. *CoRR*, vol. abs/1802.03480, 2018. doi:10.48550/arXiv.1802.03480.

[3] J. Mitton, H. M. Senn, K. Wynne, and R. Murray-Smith. A Graph VAE and Graph Transformer Approach to Generating Molecular Graphs. *CoRR*, vol. abs/2104.04345, 2021. doi:10.48550/arXiv.2104.04345.

[4] J. Ho, A. Jain, and P. Abbeel. Denoising Diffusion Probabilistic Models. *arXiv e-prints*, p. earXiv:2006.11239, Jun. 2020. doi:10.48550/arXiv.2006.11239.

[5] E. Hoogeboom, V. Garcia Satorras, C. Vignac, and M. Welling. Equivariant Diffusion for Molecule Generation in 3D. *arXiv e-prints*, p. earXiv:2203.17003, Mar. 2022. doi:10.48550/arXiv.2203.17003.

[6] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv e-prints*, p. earXiv:1505.04597, May 2015. doi:10.48550/arXiv.1505.04597.

[7] J. Ho and T. Salimans. Classifier-Free Diffusion Guidance. *arXiv e-prints*, p. earXiv:2207.12598, Jul. 2022. doi:10.48550/arXiv.2207.12598.

[8] Pogány, D. and Sárközy, P. Using Invertible Plugins in Autoencoders for Fast and Customizable Post-training Optimization. *Proceedings Of The 29th Minisymposium*, pp. 66-69, 2022. doi:10.3311/MINISY2022-017.

[9] M. Xu, A. S. Powers, R. O. Dror, S. Ermon, and J. Leskovec. Geometric Latent Diffusion Models for 3D Molecule Generation. *International Conference on Machine Learning*, pp. 38592–38610, 2023. doi:10.48550/arXiv.2305.01140.

[10] G. R. Bickerton, G. V. Paolini, J. Besnard, S. Muresan, and A. L. Hopkins. Quantifying the chemical beauty of drugs. *Nat Chem*, vol. 4, no. 2, pp. 90–98, Jan. 2012. doi:10.1038/nchem.1243.

[11] P. Ertl and A. Schuffenhauer. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of Cheminformatics*, vol. 1, no. 1, p. 8, Jun. 2009. doi:10.1186/1758-2946-1-8.

[12] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. *arXiv e-prints*, p. arXiv:1312.6114, Dec. 2013. doi: 10.48550/arXiv.1312.6114.

[13] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. *arXiv e-prints*, p. arXiv:1503.03585, Mar. 2015. doi:10.48550/arXiv.1503.03585.

[14] I. Higgins et al. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. *International Conference on Learning Representations*, 2017.

[15] A. NG. Sparse autoencoder. *CS294A Lecture notes*, 72(2011), pp. 1–19, 2011.

[16] A. Vaswani et al. Attention Is All You Need. *arXiv e-prints*, p. arXiv:1706.03762, Jun. 2017. doi:10.48550/arXiv.1706.03762.

[17] D. Weininger. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.*, vol. 28, pp. 31–36, 1988, doi:10.1021/ci00057a005.

[18] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. *arXiv e-prints*, p. arXiv:2112.10752, Dec. 2021. doi:10.48550/arXiv.2112.10752.

[19] J. J. Irwin and B. K. Shoichet. ZINC: A Free Database of Commercially Available Compounds for Virtual Screening. *Journal of Chemical Information and Modeling*, vol. 45, no. 1, pp. 177–182, Jan. 2005. doi:10.1021/ci049714+.

[20] J. Song, C. Meng, and S. Ermon. Denoising Diffusion Implicit Models. *arXiv e-prints*, p. arXiv:2010.02502, Oct. 2020. doi:10.48550/arXiv.2010.02502.

[21] S. Dieleman. Guidance: a cheat code for diffusion models. 2022. [Online]. Available: https://benanne.github.io/2022/05/26/guidance.html. [Accessed: Nov. 10, 2023].

From Hard-Coded to Modeled: Towards Making Semantic-Preserving Model Transformations More Flexible

Ármin Zavada, Vince Molnár
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
arminzavada@edu.bme.hu, molnarv@mit.bme.hu

***Abstract*—In the field of Model-based Systems Engineering, there is an increasing demand for the application of formal methods. However, transforming engineering models into formal, analyzable models is a complex task, often necessitating individual effort for each pair of modeling languages. While attempts have been made to simplify the N*M transformations to N+M using intermediate languages, this approach also proves challenging: modifications to the intermediate language are often necessary to support specific high-level languages, making maintenance difficult. Instead, we propose an alternative approach, inspired by the Kernel Modeling Language. The aim is to trace the semantics of the high-level engineering models back to the semantics of low-level elements with the help of a modular modeling language. This language can be derived from either an intermediate language or a low-level formal language, with a compositional transformation engine interpreting it. This paper explores, through the example of the Gamma framework, the challenges posed by existing model transformations and tools, and outlines the requirements that such a modular modeling language shall meet.**

***Index Terms*—Model-based Systems Engineering, Semantic-preserving, formal model transformation, modular modeling language**

I. Introduction

The failure of critical systems, such as those in train infrastructure, autonomous vehicles, airplanes, or nuclear power plants can result in severe economic damage or even loss of life. Thus, ensuring safety is a key priority during systems design. Various validation and verification (V&V) techniques are employed during systems design to guarantee safety. However, as systems grow in complexity, verifying them becomes increasingly challenging [14].

To simplify the engineering design work, new approaches have been adopted to help design, verify, and implement complex systems. One such methodology is Model-Based Systems Engineering (MBSE) [9], which prioritizes engineering modeling languages over document-centric solutions. The verification and implementation of designs created in such a manner can be accelerated and (at least partially) automated with the right tools, e.g., model checking [1].

This work was supported by the ÚNKP-23-3 New National Excellence Program of the Ministry for Innovation and Technology.

In MBSE, models serve as the primary artifacts of the development process [18]. These models are expressed using various modeling languages, such as UML [6], SysML v1 [5] and SysML v2 [17], and AADL, each with its specific syntax and semantics. While these languages share similar approaches, they often exhibit slight differences in execution semantics. One such difference is semantic variation: elements of languages with similar syntax often have slightly different execution semantics (e.g., top-down vs bottom-up region scheduling). Moreover, it is important to note that these languages frequently have under-specified execution semantics, leading to varied interpretations by individual systems engineers and companies [2].

The increasing complexity of systems, and the trend to develop complex systems of systems (SoS) [10] necessitates the integrated use of formal methods, such as automated model checking. Systems engineers usually have no formal methods background [12], thus such solutions must apply *hidden* formal methods, i.e., the end-to-end formal verification of engineering models without user input. However, the aforementioned semantic differences between languages make it difficult to unify the processing of modeling languages, significantly reducing the wide adoption of advanced verification tools implementing hidden formal methods across the industry.

One such verification tool is the Gamma Statechart Composition Framework [15], which introduces an intermediate language and supports the mapping from composite statecharts to detailed analysis languages using several model transformations. Thus, the framework provides a bridge between the engineering and analysis world and allows the automatic verification of engineering models using various model checker tools. However, the increasing complexity of the framework considerably hinders its extendability and maintainability due to the number of supported execution semantics variants by now. It is difficult to customize the built-in model transformations, and it is not always possible to map engineering languages without much modification to the intermediate Gamma language [8].

To mitigate the extensibility issue, we propose an alternative approach, inspired by the Kernel Modeling Language

DOI https://doi.org/10.3311/MINISY2024-008

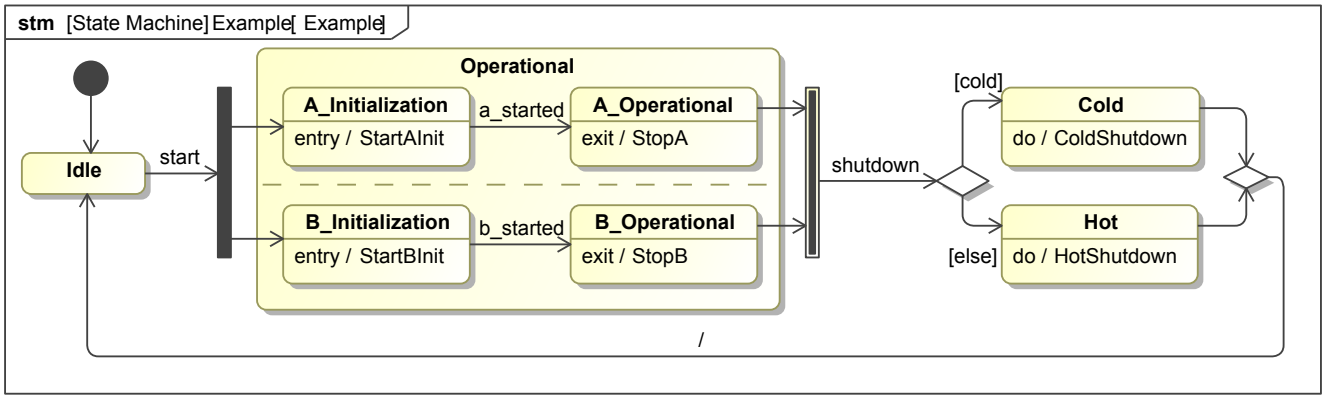


Fig. 1. A State Machine in the Systems Modeling Language. It models a two-component system initialization, operation, and shutdown.

(KerML) [16]. Instead of maintaining an intermediate language and modifying it for each custom use case, we propose the use of a new modular modeling language. This language can be derived from either an intermediate language or a low-level formal language, with a compositional transformation engine interpreting it. Given a minimal set of modeling constructs with well-defined execution semantics, the higher-level model semantics may be built using various composition techniques. Using this approach, the model transformation becomes a modeling problem rather than a software engineering problem, giving way to an increased level of customizability: the model transformation may be refined by refining the model itself, without modifying the language or the transformation engine.

In this paper, we explore the challenges faced by existing model transformation tools, through the example of the Gamma framework. As a contribution, we outline the requirements that such a modular modeling language shall meet to be useable in the proposed approach.

II. Motivation

The Systems Modeling Language (SysML) [5] is a general-purpose modeling language for systems modeling that is intended to facilitate an MBSE approach during system design. State Machines [7] have been widely adopted throughout the industry for the modeling of complex safety-critical systems due to the relative simplicity of their use. Figure 1 showcases a State Machine defined in the SysML language. The system is *Idle* at the beginning, and initializes upon the *Start* signal. The two components initialize and operate in parallel, which is realized using *fork-join* pseudo-states. The *Initialization* states start the A and B components respectively. When the given component starts, the State Machine transitions to the *Operational* state. When the *shutdown* signal is received, both components must be in *Operational* state, exiting which results in the *Stop* exit behavior. Shutdowns can be either *cold* or *hot*, which is modeled using *choice-merge* pseudo-nodes. Each shutdown state executes its respective shutdown behavior, and automatically transitions to the Idle state upon the successful finish of the do-activity behavior.

In general, to describe any stateful system's behavior, the following three parts are needed.

- Storing the current state configuration of the System;
- A way to calculate the next states from the current state;
- Execute a given state transition from the current state to the next state.

In the simple State Machine case, **States** specify the possible states of the system, with a variable storing the *current* state. The following states can be calculated by checking which transitions are outgoing from the *current* state, and the state transition is executed by modifying the *active* state variable. However, the more complex semantics, such as *hierarchical states*, *parallel regions*, *entry* and *exit* actions, *pseudo* states with *composite* transitions, and *do-activities* are difficult to support. Most of the language semantics are well-defined, but several have under-defined semantics, resulting in a subjective interpretation of actual models [2].

There are existing tools supporting various modeling languages similar to SysML. The Gamma Statechart Composition Framework [15] is one such tool, capable of formally modeling composite systems using statecharts as the main atomic components, and composing them together synchronously or asynchronously. The behavior of each atomic component is captured by a statechart while assembling the system from components is driven by a composition language. Gamma supports several composition semantics, allowing the user to model systems with various (potentially mixed) execution and interaction semantics. Although Gamma provides extensive support for the composition of mixed semantics components, it is difficult to integrate with languages not exactly aligned with the Gamma semantics. Oftentimes, the integration of industrial modeling languages can not be done seamlessly; several industrial projects required Gamma developers to add additional features to the language, which increases the complexity of the framework – e.g., we had to add an entirely new language to the framework to support UML State Machines in Gamma [19]. Additionally, each new feature either must be

supported at the same time, or several Gamma forks must be maintained concurrently.

III. Gamma Statechart Composition Framework

This section provides a brief overview of the Gamma internals, to gain an understanding of their current complexity, and how it might be replaced with a new approach.

A. Gamma Internals

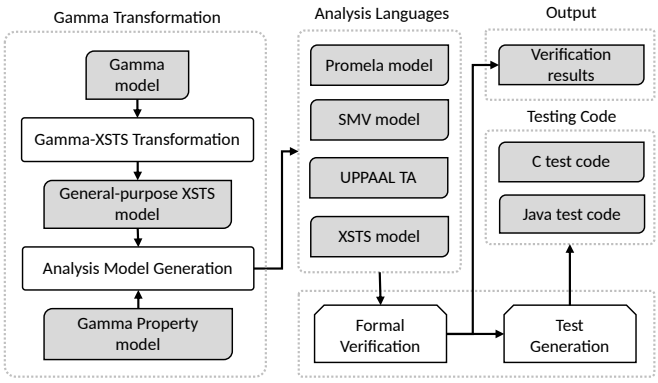


Fig. 2. The Gamma model transformation workflow.

Figure 2 shows the Gamma model transformation workflow. First, the handcrafted, or automatically derived Gamma model is transformed into a general-purpose XSTS model. XSTS, or *eXtended Symbolic Transition System* is a formal analysis language used by the Gamma framework. This XSTS model entails all execution behavior of the original Gamma model in a formal representation. The general-purpose XSTS model is transformed into the various back-end analysis languages using the defined Gamma Property model. The Property model selects the properties to be verified in the system. Using slicing and other model optimization techniques, the resulting analysis models are stripped of any detail unrelated to the verification property [4]. Using these models, Gamma can seamlessly utilize various model checkers, providing a hidden formal verification experience for the users. The framework is also able to generate concrete test cases from the verification results, which may be used as a conformance test suite for the final implementation of the system.

To model various system *behaviors*, Gamma defines several formal languages, of which the *Gamma Expression Language (GEL)* and *Gamma Action Language (GAL)* serve as the foundation. GEL and GAL together define *variables*, *types*, and *expressions* accessing and combining them using *arithmetical* and *logical* expressions. GAL builds on these constructs by providing simple *atomic actions* over variables in a reusable fashion. The most basic building blocks of Gamma components are atomic components, of which Gamma currently supports statecharts with the *Gamma Statechart Language* (GSL) [3]. Statechart formal semantics provide simple and composite states, entry and exit behaviors, orthogonal regions, and transitions with guards and effects. GSL also supports pseudo-states, such as (shallow) history and deep history states, fork-join, and decision-merge states. Using the latter, the user can model the system using complex transitions.

Components serve as types of component instances. They may be *atomic* or *composite*, *synchronous*, or *asynchronous*. A component can have zero or more ports, which serve as the only point of interaction between components. This ensures that external dependencies and interactions are explicitly modeled, leading to a fully encapsulated behavior. Table I summarizes the various components Gamma supports.

	Atomic	**Composite**
Synchronous	Statechart	Synchronous composite Cascade composite
Asynchronous	Asynchronous adapter	Scheduled asynchronous Asynchronous composite

TABLE I
THE VARIOUS COMPONENTS GAMMA SUPPORTS, GROUPED INTO ATOMIC-COMPOSITE AND SYNCHRONOUS-ASYNCHRONOUS CATEGORIES.

B. Transformation Chain

The Gamma-XSTS transformation chain produces the general-purpose XSTS models out of the original Gamma models. Figure 3 illustrates the main states the models go through during the transformation chain. The phases of the transformation are detailed below.

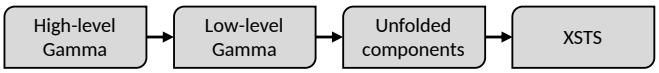


Fig. 3. An example State Machine in SysML language.

1) Syntax Desugaring: The Gamma language supports various syntax sugars, that make it easier for engineers to develop and understand models. However, it clutters the metamodel with redundant elements, that are not easy to process directly. For this reason, during the syntax desugaring phase, the high-level Gamma model is transformed into a simpler, desugared low-level representation.

2) Component Unfolding: In the next step, the now desugared low-level model is instantiated into the context of the verification case. The various components used in the model act as types and are unfolded in the context of their usage. During the unfolding step, the actual (atomic or composite) components are unfolded into concrete instances and are connected with actual channels and ports.

3) XSTS Generation: This is the code generation phase of the transformation chain. In this step, the now desugared and unfolded concrete component instances are mapped into the XSTS formalism. Atomic statechart components are directly mapped into XSTS. These statechart components are then reused during the composite component mappings, e.g., since the *cascade* component may execute the same statechart component multiple times in one cycle, the same statechart XSTS representation may be reused multiple times. In the case of the *asynchronous* components, the common ports are connected to asynchronous *channels*, which are also mapped into the XSTS formalism.

IV. Modelable Language Semantics

While Gamma provides a bridge between engineering and analysis domains, enabling automatic verification of engineering models, its complexity hampers extendability and maintainability. Customization of the hard-coded model transformations and mapping diverse engineering languages to existing Gamma variants proves difficult. To overcome these difficulties, we propose the use of a modular modeling language, with which the higher-level model semantics can be constructed using various composition techniques. With this approach, the model transformation becomes a modeling task, which in turn becomes simpler to customize, even without modifying the source code.

As the first step of designing such a modular modeling language, we synthesized a set of requirements from the current transformation chain of the Gamma framework presented in section III.

A. Pattern Support

Gamma uses syntax sugars extensively in its languages, which must be desugared during the transformation chain. Syntax sugars are necessary since they provide an encapsulation closer to the actual domain of the model. For example, *ports* are represented with simple boolean flags in the low-level model, however, in the user-facing language it makes more sense to hide this, and encapsulate it in a *Port* definition. Syntax sugars also allow the use of automatic model validation.

Thus, the modular modeling language must support the use of modeling patterns, that can encapsulate behavior and hide implementation details from the user.

B. Types and Instantiation

Implicit and explicit types are heavily used in the Gamma transformation chain. Explicit definitions range from various kinds of composite and atomic components to data structures, while implicit types appear numerously – e.g. ports, events, triggers, regions, and states – in the language. The support of extendible types and instantiation is a must in the modular language.

Types must be reusable and extendible, possibly by some kind of inheritance between types. Instantiation does not have to be dynamic, however, at least static (or compile time) instantiation is necessary to replicate the current transformation chain. The tracking of compile-time instances must be possible in the target language. Gamma currently uses similar functionality when transforming statecharts: the current state of a given *region* is modeled using a *current state variable*, whose value range is the states of the region.

C. Context Sensitivity

To realize a truly modular modeling language, the language must allow context-sensitive types. Simple context sensitivity may be achieved by passing parameters during instantiation, however, to solve complex transformation problems – e.g., choosing the transitions to be fired in the case of parallel regions – a more powerful, graph-based querying support is required. With context-sensitive types, we can use aspect-oriented modeling, significantly reducing the complexity of the model.

Pattern-based model querying could be useful in complex state machines. For example, in the State Machine presented in Figure 1, a model query would be required to collect the complex transitions. When *start* is received, both outgoing transitions must be executed at the same time, activating both *Initialization* states. Similarly, when the *shutdown* signal is received, both *Operational* states must be active, and both are deactivated at the same time. Since the transition points to a *Choice* pseudo-state, the actual target is either *Cold* or *Hot*, depending on the value of the *cold* flag. Gamma supports such fork-join and choice-merge pairs to an arbitrary complexity, which must be possible to be replicated using the modular modeling language.

D. Manipulation of the Target Model

Using patterns, context-sensitive types, and instantiation, the *structure* of the desired model can be achieved. However, the execution semantics of the model remain unaddressed. To realize the exact execution semantics, the modular language must allow the manipulation of the target language. Doing so would allow the user to manipulate, i.e. query, add, and modify the target model's execution to the exact needed semantics.

For example, the execution semantics of a *cascade* component could be executing the internal components and the internal channels one by one on a turn-based basis. Another example could be the execution of a given transition: a transition execution entails the execution of the exit behaviors, then the effect behavior, and then the enter behaviors – all in a predetermined order, all per the desired execution semantics. In a statechart with parallel regions, multiple transitions must be executed at the same time for each incoming event.

V. Related Work

Yannis Lilis and Anthony Savidis conducted an extensive survey of existing meta-programming languages in [11]. The survey classified the meta-programming languages into several categories, including the main metaprogramming model employed by the language, their phase of evaluation, and finally, the relation between the meta-language and the object language. Meta-programming languages seem to be a great candidate for the proposed modular modeling language.

KerML [16] uses a semantic library approach, in which the semantics of the language is modeled in separate layers. The core layer gives the mathematical semantics of the language, which the Kernel layer uses to define general modeling concepts. Our approach is inspired by the KerML libraries.

In [13] Marussy et. al. propose a view model transformation approach. To do so, they provide a fully compositional transformation language with graph query integration.

VI. Conclusion and Next Steps

The precise execution semantics of modeling languages are essential in the world of MBSE. For hidden formal methods to gain widespread use, advanced, and configurable formal verification tools are needed. However, current state-of-the-art tools are difficult to customize and extend with new languages, considerably prohibiting the wide use of formal verification. In this work, we proposed a new approach of using a modular language for modeling the semantics of engineering languages, instead of hard-coded model transformation. To design such a language, we studied the Gamma Statechart Composition Framework and its current model transformation chain. Using the insights gained, we outlined a set of requirements that such a modular modeling language shall meet.

As the next step, our goal is to experiment with possible languages adhering to the outlined requirements, ultimately implementing a language capable of replicating much of the Gamma transformation chain. Using this new language and a corresponding interpreter engine, we will evaluate the approach using various case studies for various high-level engineering languages.

References

[1] Edmund M Clarke, Orna Grumberg, and Doron A. Peled. *Model checking*. MIT Press, London, Cambridge, 1999.

[2] Márton Elekes, Vince Molnár, and Zoltán Micskei. Assessing the specification of modelling language semantics: a study on UML PSSM. *Software Quality Journal*, 31(2):575–617, Jun 2023.

[3] Bence Graics. Mixed-Semantics Composition of Statecharts for the Model-Driven Design of Reactive Systems. Master's thesis, BME, 2018.

[4] Bence Graics, Vince Molnár, and István Majzik. Integration test generation for state-based components in the Gamma framework. 2023.

[5] Object Management Group. Systems Modeling Language (SysML), 2012.

[6] Object Management Group. Unified Modeling Language (UML-v2.5.1), 2017.

[7] David Harel. Statecharts: a visual formalism for complex systems. *Science of Computer Programming*, 8(3):231–274, 1987.

[8] Benedek Horváth, Vince Molnár, Bence Graics, Ákos Hajdu, István Ráth, Ákos Horváth, Robert Karban, Gelys Trancho, and Zoltán Micskei. Pragmatic verification and validation of industrial executable SysML models. *Systems Engineering*, n/a(n/a), 2023.

[9] INCOSE. INCOSE Systems Engineering. https://www.incose.org/systems-engineering, 2023. Accessed: 2023-11-02.

[10] INCOSE. INCOSE Systems Engineering Vision 2035. https://www.incose.org/docs/default-source/se-vision/incose-se-vision-2035-executive-summary.pdf, 2023. Accessed: 2023-11-02.

[11] Yannis Lilis and Anthony Savidis. A Survey of Metaprogramming Languages. *ACM Comput. Surv.*, 52(6), oct 2019.

[12] Qin Ma, Monika Kaczmarek-Heß, and Sybren de Kinderen. Validation and verification in domain-specific modeling method engineering: an integrated life-cycle view. *Software and Systems Modeling*, Oct 2022.

[13] Kristóf Marussy, Oszkár Semeráth, and Dániel Varró. Incremental view model synchronization using partial models. In *Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, MODELS '18, page 323–333, New York, NY, USA, 2018. Association for Computing Machinery.

[14] Markus Maurer. *Automotive Systems Engineering: A Personal Perspective*, pages 17–35. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[15] Vince Molnár, Bence Graics, András Vörös, István Majzik, and Dániel Varró. The Gamma statechart composition framework: Design, verification and code generation for component-based reactive systems. In *Proceedings of ICSE'18: Companion Proceeedings*, pages 113–116. ACM, 2018.

[16] OMG. *Kernel Modeling Language (KerML)*, 2023. ptc/23-06-01.

[17] OMG. *OMG System Modeling Language (SysML v2)*, 2023. ptc/23-06-02.

[18] Ed Seidewitz. What models mean. *IEEE Software*, 20(5):26–32, 2003.

[19] Péter Szkupien and Ármin Zavada. Formal Methods for Better Standards: Validating the UML PSSM Standard about State Machine Semantics. Thesis for students' scientific conference, BME, 2022.

Hyperbolic Drug-Target Interaction Prediction Utilizing Differential Expression Signatures

Domonkos Pogány, Péter Antal
Budapest University of Technology and Economics
Department of Measurement and Information Systems
Budapest, Hungary
E-mail: {pogany, antal}@mit.bme.hu

***Abstract*—Efficiently predicting interactions between compounds and target proteins is pivotal in drug discovery, driving the need for machine-learning-based approaches to replace resource-intensive experiments. This study investigates the potential of hyperbolic geometry in enhancing pairwise interaction prediction models, emphasizing the identification of suitable input modalities to leverage additional information through non-Euclidean embeddings. Despite the potential benefits of hyperbolic embeddings, our study highlights their limited contribution when employing widely used structure-based pre-trained input representations. However, hyperbolic predictors outperform their Euclidean counterparts with transcriptomics-based input, underscoring the importance of an appropriate input modality, such as differentially expressed gene signatures. Besides aiding the selection of input modalities for interaction prediction, the results also confirm our prior hypothesis: differential expression signatures possess a non-Euclidean nature and thus can be better represented in a hyperbolic vector space.**

***Index Terms*—drug-target interaction prediction, hyperbolic geometry, differentially expressed gene signatures**

I. Introduction

Predicting drug-target interactions (DTIs) is a key challenge in drug discovery, necessitating the exploration of efficient computational methods to replace resource-intensive experiments. In recent years, with the increasing amount of available data, machine-learning solutions have become state-of-the-art in DTI prediction [1], [2].

Modeling DTI prediction as a binary classification problem is a standard choice in the literature [2]. Solutions commonly employ a *pairwise* neural network model with dual input, predicting binary interactions between compounds and targets. Besides raw structural inputs, embeddings pre-trained in an unsupervised way are also widely used. For instance, employing a Word2vec concept on the input sequences, we can encode drug molecules and target proteins as vectors, resulting in the *Mol2vec* [3] and *ProtVec* [4] embeddings, respectively. While utilizing the combination of these embeddings is a widespread solution in pairwise DTI prediction approaches [5], recent developments have introduced more informative input alternatives, such as the integration of gene expression data [6].

Differentially expressed gene (DEG) signatures are pivotal in computational biology, providing valuable insights into variations in gene expression between experimental conditions and offering more complex information than other static representations. The *Library of Integrated Network-Based Cellular Signatures* (LINCS) project aims to provide a universal language for biology-related tasks via an extensive repository of gene expression signatures relying on the *Connectivity Map* [7]. Through a strategic selection of 978 landmark genes based on correlated expression levels, they significantly reduced the number of expressions to be measured. Additionally, by developing the *L1000 Luminex* bead technology, a cost-effective, high-throughput microarray platform, LINCS made it possible to scale up the available data by measuring only these landmark genes [8]. With over three million available L1000 microarray profiles, machine learning benchmarks emerge, exploiting the potential inherent in gene expression signatures [9].

Building upon the hypothesis that drug-induced gene signatures correlate with the knock-down signature of the drug's target gene [10], several solutions have been proposed utilizing the unified LINCS L1000 transcriptome data as input features [6], [11]–[14]. These pairwise DTI predictors rely on Z-score-based DEG signatures to represent drugs and targets on their inputs, aiming for enhanced predictive performance given the rich information in transcriptomic data. In our current research, we hypothesize that this information can be harnessed even more effectively using non-Euclidean manifolds.

Many real-world datasets exhibit latent non-Euclidean structures, for instance, biological data with inherent hierarchies. Integrating hyperbolic vector spaces into machine-learning approaches can effectively capture these hidden structures, making it possible to learn continuous, hierarchical representations. Multiple equivalent models of hyperbolic geometry can be utilized, including the *Poincaré ball*, preferable for visualization purposes [15], and the *Lorentz* manifold, which is more frequently used in machine-learning applications due to its numerical stability [16]. Two fundamental approaches exist to incorporate hyperbolic embeddings into machine-learning

The project 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. This research was also funded by the J. Heim Student Scholarship, the OTKA-K139330, the European Union (EU) Joint Program on Neurodegenerative Disease (JPND) Grant: (SOLID JPND2021-650-233), the National Research, Development, and Innovation Fund of Hungary under Grant TKP2021-EGA-02, the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.

models, both involving *exponential* and *logarithmic maps* to transform embeddings from the Euclidean tangent space to the hyperbolic manifold and vice versa. The first option employs an Euclidean vector space for trainable model parameters, converting only the output embeddings into hyperbolic space to calculate distances. Another computationally less efficient yet more expressive approach is to represent the model parameters on a non-Euclidean manifold as well and perform the parameter updating with *Riemannian optimization* [17], facilitating the development of hyperbolic versions of known neural architectures, such as fully hyperbolic neural networks [18].

Instead of modeling the biological space with a flat geometry, hyperbolic embeddings can improve the predictive performance of pairwise DTI prediction methods as well. Previous studies investigated shallow matrix factorization models without prior input features [19], as well as fully hyperbolic neural networks applying drug-drug and protein-protein similarity-based inputs [20]. Additionally, our previous study demonstrated that applying appropriate regularization allows the embedding of drug and protein hierarchies in the shared hyperbolic latent space of pairwise DTI models, thereby increasing the interpretability of similarity-based predictors [21]. However, we also noticed that unlike with the previously mentioned inputs, when utilizing more complex, pre-trained structural representations, although enhancing explainability, hyperbolic embeddings do not necessarily contribute to improved predictive performance.

Identifying input features suitable for efficient utilization in pairwise hyperbolic DTI models is an open challenge. In our other prior work, we already showed that even the more processed differential expression signatures, a domain where hyperbolic methods have not yet been widely explored, exhibit a non-Euclidean nature [22]. According to our earlier findings, LINCS DEG signatures might be a reasonable choice for hyperbolic pairwise methods.

To the best of our knowledge, other DTI prediction approaches have not yet embedded DEG signatures into hyperbolic spaces. In addressing this research gap, we compared traditional structure-based representations with DEG signatures as inputs for pairwise DTI prediction in both hyperbolic and Euclidean settings.

II. Method

A. Data

Following the work of other previous studies [12]–[14], we utilized a dataset provided by Yang et al. [13]. It contains positive interactions from the DrugBank database [1], with entities being mapped to LINCS landmark signatures via their PubChem ID, making it possible to represent drugs and targets with drug perturbation and gene knock-down (not necessarily a landmark gene) DEG signatures, respectively. Besides the z-scored DEG signatures, we extended the data with the 300-dimensional pre-trained Mol2vec and ProtVec embeddings based on the available *Simplified Molecular Input Line Entry System (SMILES)* drug descriptors and *UniProt* gene IDs. We only kept the entities for which both the signatures and the pre-trained embeddings were available, resulting in a dataset containing seven cell lines: A375, A549, HA1E, HCC515, HEPG2, PC3, and VCAP. Table I details the main characteristics of the final dataset.

TABLE I
CHARACTERISTICS OF THE USED DATA.

Cell line	Drugs	Targets	Positive interactions	Sparsity
A375	517	360	776	0.9958
A549	522	364	787	0.9959
HA1E	530	369	797	0.9959
HCC515	467	331	675	0.9956
HEPG2	367	353	545	0.9958
PC3	639	375	940	0.9961
VCAP	518	374	788	0.9959

To avoid treating unknown drug-target interactions as negative ones, previous studies have utilized a nontarget set with no interaction record with any drug from the given cell line in DrugBank. However, we found that with this approach, pairwise models can easily learn that some targets correspond to only positive interactions while others correspond to only negative ones. Instead, we dropped the nontarget set, and for each positive pair, we sampled M times as many negative ones from the unknown interactions. Later, to deal with the imbalances and express uncertainty in the negative samples, weight ratios of 1 to M were assigned for negative to positive interactions in the objective function.

B. Model

We opted for a simple yet effective pairwise model, employing drug and target embeddings, either a Mol2vec - ProtVec or a DEG signature combination as input. These are then processed with encoders to produce the latent representations. Akin to Xie et al. [12], we used *multi-layer perceptrons* MLPs with 100 hidden and 10 output neurons and ReLU hidden activation, but instead of concatenating the input and applying one MLP, we use separate MLP encoders for the drugs and the targets, and then take the Euclidean distance between the resulting 10-dimensional latent embeddings and apply a $g(x) = e^{-x^2}$ activation to predict the binary interaction.

Besides outperforming the previously used concatenation with MLP, the distance-based prediction gives an opportunity to incorporate hyperbolic embeddings and compare them with the Euclidean counterparts. To create the hyperbolic model version, similarly to our previous work, we clipped the embeddings into the unit circle for stability reasons and subsequently applied the exponential map to transform the latent embeddings from the Euclidean tangent space to a Lorentzian manifold with a constant negative curvature of -1 [21]. Finally, we replaced the Euclidean distance with the squared Lorentzian distance, a widely-used metric in hyperbolic interaction prediction [19]–[21].

Similarly to other approaches [20], we also experimented with fully hyperbolic neural networks and a hyperbolic version of the concatenation [18] to create non-euclidean versions of the previously used benchmark models [14]. However, we found no increase in performance. More than that, the

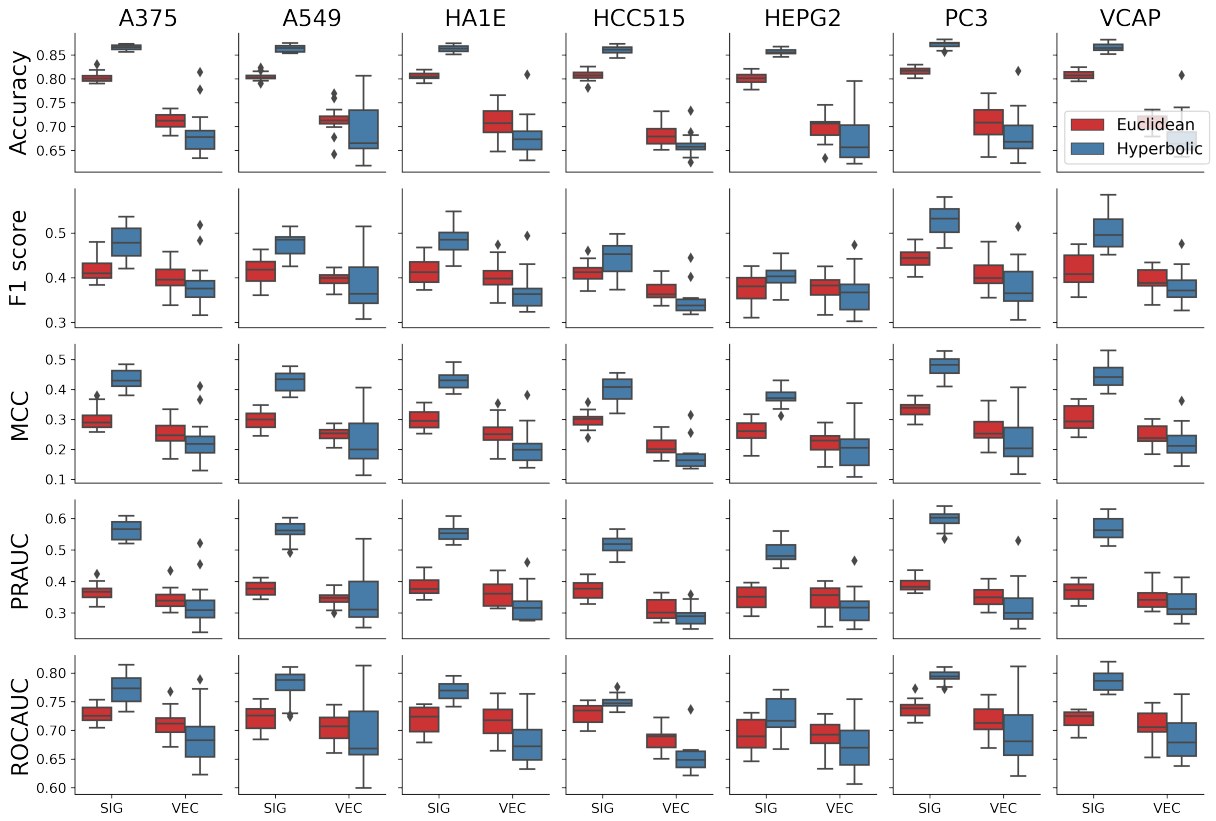


Fig. 1. Cross-validation results on different cell lines (columns) according to various evaluation metrics (rows). Four scenarios were assessed for each cell line and evaluation metric based on the combination of different input modalities and latent manifolds. Euclidean versions are denoted with red, while blue corresponds to models with latent embeddings on a hyperbolic Lorentz manifold. Input representations can be either a Mol2vec - ProtVec or a DEG signature pair denoted with VEC and SIG, respectively. For each scenario, a boxplot summarizes the results of $K * N = 20$ different runs.

Riemannian optimization and the number of exponential and logarithmic maps required in hyperbolic neural networks significantly impacted training and inference time. Therefore, we used our simple pairwise model throughout the paper.

C. Evaluation Metrics

We utilized a comprehensive set of five metrics to assess the binary interaction classification task. Besides *accuracy*, we employed the *F1 score* and the *Matthew correlation coefficient* (MCC), also known as the phi coefficient. Both are single-value metrics, summarizing the confusion matrix given by applying a fixed 0.5 threshold on the output activation. To capture model performance across varying classification thresholds, we incorporated the *area under the receiver operating characteristic curve* (ROCAUC) and the *area under the precision-recall curve* (PRAUC) scores as well. A higher value corresponds to a better model for all metrics, with the maximum achievable score being 1.

III. Results

Following other previous works, we conducted model fitting and evaluation on the seven cell lines separately. On each cell line, we performed repeated cross-validation with $K = 4$ folds and $N = 5$ repeats, applying a 1 to $M = 5$ negative sample ratio and a stratified train-test split on the interactions. After Xavier weight initialization, we trained the models for 256 epochs in a full bach manner using an *Adaptive Moment Estimation* (Adam) optimizer with a learning rate of 10^{-4} and weighted *binary cross-entropy* (BCE) as the objective function with a 5 to 1 positive sample weight. Models were implemented using the PyTorch framework and trained on a 32GB NVIDIA Tesla V100 GPU. We found that the models are robust to these hyperparameters, all reaching their saturation levels concerning the validation metrics. Figure 1 illustrates the results, comparing different manifolds and input representations. In addition to the presented simple model, we also achieved similar results with the other model architectures mentioned earlier.

The same result is obtained consistently for all cell lines and by all metrics. As we can see, utilizing signatures leads to better results, indicating that LINCS L1000 landmark DEG signatures encode relevant information for predicting interactions. However, unlike pre-trained, structure-based representations, they rely on expensive measurements, even with the L1000 platform. As for the used manifold, similarly to what we saw in our previous work, using structural inputs, traditional models slightly outperform the non-euclidean ones. However, with signatures as inputs, hyperbolic methods produce significantly better results than all other combinations.

IV. Conclusion and Future Work

Our study reveals that, when applying transcriptomics-based input, pairwise DTI models with hyperbolic embeddings exhibit superior performance compared to their Euclidean counterparts. This confirms our earlier hypothesis that DEG signatures can be better represented in a hyperbolic vector space, even when only the landmark genes are used [22]. Notably, this observation holds not only for signatures produced by the characteristic direction method as explored in our previous study, but also for the Z-score-based signatures utilized in our current work. At the same time, our work leads to other theoretical questions yet to be investigated, such as whether landmark but not differentially expressed signatures or other differentially expressed signatures consisting of not only the LINCS landmark genes have a hyperbolic nature.

It is important to note that when applying widely used structure-based pre-trained input representations, models do not benefit from hyperbolic embeddings considering predictive performance, which is consistent with our prior observations [21]. This implies that hyperbolic embeddings can only extract additional information with a suitable non-Euclidean input modality, such as the LINCS DEG signatures.

However, the number of available DTI prediction datasets incorporating expression signatures is still limited. Therefore, future research should aim at identifying appropriate structure-based inputs, potentially involving semi-supervised pre-training in a hyperbolic space or utilizing inferred gene expression signatures predicted from structural inputs [23].

Another direction worth exploring is how to deal with the lack of negative interactions. Instead of sampling and weighting, being compatible with hyperbolic distances, we can utilize the Bayesian personalized ranking-based method proposed by Ye et al. [14]. Alternatively, we can apply our previously proposed drug-target metric-learning approach, replacing negative sampling with SOTA representation learning solutions [24]. Or we can obtain known negative interactions by discriminating between inhibitory and activatory targets incorporating overexpression signatures besides knock-downs [11].

To further improve the model, multiple modalities can be used simultaneously, utilizing both Euclidean and hyperbolic embeddings to fuse the information from both transcriptomics and structure-based inputs [6]. Another option is to incorporate prior hierarchical information with regularization [21]. Besides resulting in a more interpretable latent space, it might also increase performance when using DEG signatures.

References

[1] D. S. Wishart et al., 'DrugBank 5.0: a major update to the DrugBank database for 2018', Nucleic acids research, vol. 46, no. D1, pp. D1074–D1082, 2018. doi:10.1093/nar/gkx1037

[2] M. Bagherian, E. Sabeti, K. Wang, M. A. Sartor, Z. Nikolovska-Coleska, and K. Najarian, 'Machine learning approaches and databases for prediction of drug–target interaction: a survey paper', Briefings in bioinformatics, vol. 22, no. 1, pp. 247–269, 2021. doi:10.1093/bib/bbz157

[3] S. Jaeger, S. Fulle, and S. Turk, 'Mol2vec: unsupervised machine learning approach with chemical intuition', Journal of chemical information and modeling, vol. 58, no. 1, pp. 27–35, 2018. doi:10.1021/acs.jcim.7b00616

[4] E. Asgari and M. R. K. Mofrad, 'Continuous distributed representation of biological sequences for deep proteomics and genomics', PloS one, vol. 10, no. 11, p. e0141287, 2015. doi:10.1371/journal.pone.0141287

[5] A. Chatterjee et al., 'Improving the generalizability of protein-ligand binding predictions with AI-Bind', Nature Communications, vol. 14, no. 1, p. 1989, 2023. doi:10.1038/s41467-023-37572-z

[6] X. Xia, C. Zhu, F. Zhong, and L. Liu, 'MDTips: a multimodal-data-based drug–target interaction prediction system fusing knowledge, gene expression profile, and structural data', Bioinformatics, vol. 39, no. 7, p. btad411, 2023. doi:10.1093/bioinformatics/btad411

[7] J. Lamb et al., 'The Connectivity Map: using gene-expression signatures to connect small molecules, genes, and disease', science, vol. 313, no. 5795, pp. 1929–1935, 2006. doi:10.1126/science.1132939

[8] A. Subramanian et al., 'A next generation connectivity map: L1000 platform and the first 1,000,000 profiles', Cell, vol. 171, no. 6, pp. 1437–1452, 2017. doi:10.1016/j.cell.2017.10.049

[9] M. B. A. McDermott et al., 'Deep learning benchmarks on L1000 gene expression data', IEEE/ACM transactions on computational biology and bioinformatics, vol. 17, no. 6, pp. 1846–1857, 2019. doi:10.1109/TCBB.2019.2910061

[10] N. A. Pabon et al., 'Predicting protein targets for drug-like compounds using transcriptomics', PLoS computational biology, vol. 14, no. 12, p. e1006651, 2018. doi:10.1371/journal.pcbi.1006651

[11] R. Sawada, M. Iwata, Y. Tabei, H. Yamato, and Y. Yamanishi, 'Predicting inhibitory and activatory drug targets by chemically and genetically perturbed transcriptome signatures', Scientific reports, vol. 8, no. 1, p. 156, 2018. doi:10.1038/s41598-017-18315-9

[12] L. Xie, S. He, X. Song, X. Bo, and Z. Zhang, 'Deep learning-based transcriptome data classification for drug-target interaction prediction', BMC genomics, vol. 19, pp. 93–102, 2018. doi:10.1186/s12864-018-5031-0

[13] J. Yang, S. He, Z. Zhang, and X. Bo, 'NegStacking: Drug- Target Interaction Prediction Based on Ensemble Learning and Logistic Regression', IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 18, no. 6, pp. 2624–2634, 2020. doi:10.1109/TCBB.2020.2968025

[14] Y. Ye, Y. Wen, Z. Zhang, S. He, X. Bo, and Others, 'Drug-target interaction prediction based on adversarial Bayesian personalized ranking', BioMed Research International, vol. 2021, 2021. doi:10.1155/2021/6690154

[15] M. Nickel and D. Kiela, 'Poincaré embeddings for learning hierarchical representations', Advances in neural information processing systems, vol. 30, 2017.

[16] M. Nickel and D. Kiela, 'Learning continuous hierarchies in the lorentz model of hyperbolic geometry', in International conference on machine learning, 2018, pp. 3779–3788.

[17] G. Bécigneul and O.-E. Ganea, 'Riemannian adaptive optimization methods', arXiv preprint arXiv:1810. 00760, 2018. doi:10.48550/arXiv.1810.00760

[18] O. Ganea, G. Bécigneul, and T. Hofmann, 'Hyperbolic neural networks', Advances in neural information processing systems, vol. 31, 2018.

[19] A. Poleksic, 'Hyperbolic matrix factorization improves prediction of drug-target associations', Scientific Reports, vol. 13, no. 1, p. 959, 2023. doi:10.1038/s41598-023-27995-5

[20] Y. Yue, D. McDonald, L. Hao, H. Lei, M. S. Butler, and S. He, 'FLONE: fully Lorentz network embedding for inferring novel drug targets', Bioinformatics Advances, vol. 3, no. 1, p. vbad066, 2023. doi:10.1093/bioadv/vbad066

[21] D. Pogány and P. Antal, 'Towards explainable interaction prediction: Embedding biological hierarchies into hyperbolic interaction space', BioRxiv, Preprint, 2023. doi:10.1101/2023.12.05.568518

[22] D. Pogány and P. Antal, 'Hyperbolic Manifold Learning on Differential Expression Signatures', TechRxiv, Preprint, 2023. doi:10.36227/techrxiv.24630747.v1

[23] G. Woo, M. Fernandez, M. Hsing, N. A. Lack, A. D. Cavga, and A. Cherkasov, 'DeepCOP: deep learning-based approach to predict gene regulating effects of small molecules', Bioinformatics, vol. 36, no. 3, pp. 813–818, 2020. doi:10.1093/bioinformatics/btz645

[24] D. Pogány and P. Antal, 'DT-ML: Drug-Target Metric Learning', in Proceedings of the 16th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2023) - Volume 3: BIOINFORMATICS, 2023, pp. 204–211. doi:10.5220/0011691100003414

Systematic evaluation of continuous optimization approaches for causal discovery of gene regulatory networks

Dániel Sándor, Péter Antal
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
sandor@mit.bme.hu, antal@mit.bme.hu

***Abstract*—Continuous optimization-based structure learning for Directed Acyclic Graphs (DAGs) is increasingly popular. They are used to infer the structure of graphs from high volumes of data. However, previously it has been shown that these methods are often not usable for causal discovery because of inherent algorithmic biases. The main problem stems from their sensitivity to variance in the data. In other words, they are not scale-invariant. This leads to variables with lower variance having more outgoing edges while variables with higher variance tend to have more incoming edges.**

In this paper, we test five of these methods (NOTEARS, NOTEARS-MLP, GOLEM-EV, GOLEM-NV, and DAG-NoCurl) on their performance and their robustness to variance in the data. We evaluate our findings on transcriptomic data to construct gene regulatory networks. These networks can uncover the hidden mechanisms of gene expressions. The use of scalable algorithms is well-motivated in the field. We use bootstrapping to evaluate the uncertainty of the found edges and quantify the bias of the methods. To quantify the bias, we calculate the posterior probability of a vertex being more likely to be a parent than a child and vice-versa.

***Index Terms*—Structure learning, Continuous optimization, Explainable AI, Algorithmic bias, Bayesian network, Gene expression**

I. Introduction

Gene regulatory networks (GRNs) can provide information about the inner mechanisms of a cell [1]. They consist of gene-gene connections which regulate the gene expression levels of the organism. Through the regulation of gene expression, they can maintain the biological functions of the cell. If we want to understand these mechanisms, we can model GRNs, with graphical models, for example, to understand their structure and from the structure, we can infer their operation.

As mentioned before the result of gene regulation is the expression of genes. Gene expression refers to the processing of information found in genes in the DNA to achieve some biological function [2]. This starts with the process of transcription [3]. In this process, the DNA is encoded into messenger RNA sequences, which can be observed with modern RNA sequencing (RNA-seq) technologies. With different methods, we can analyse these sequences to find out about the abundance of transcripts. The quantity of transcripts depends on several factors. By varying these factors, we can measure the expression levels of a set of genes to reconstruct the GRN.

One option to find the structure of these networks from gene expression data is to use causal discovery. Causal discovery aims to identify causal relationships between variables from data [4], [5]. While genes can have regulating effects that are not causal, if we do not allow for regulations that would induce a directed cycle in the network, then gene regulations can be characterized as causal relationships [6], thus the choice of these algorithms is correct. If we follow these assumptions, then these networks can be modelled by Directed Acyclic Graphs (DAGs), where each gene corresponds to a node, and the effect of a gene on another is indicated by the directed edge. This is how the problem of causal discovery becomes the structure learning of these DAGs.

With recent advances in DAG structure learning [7]–[10] the leading paradigm became continuous optimization-based algorithms. These allow the characterization of the DAG learning problem as a continuous function with continuous constraints, to leverage existing continuous optimization algorithms to solve these problems. These methods operate with different assumptions and biases and achieve different results on similar problems. However, to our knowledge, for gene expression datasets, they have not been systematically evaluated. Besides their performance, we are also interested in their biases, mainly their bias of not being observationally equivalent [11], which might not be detectable, when examining them on synthetic data [12], generated by Erdős-Rényi or Scale-Free graphs.

Our contributions in this paper are the following:

- We define a probabilistic measure of the bias of the algorithms to evaluate their usability to create GRNs.
- We evaluate the biases of different algorithms on gene expression data based on the DREAM4 challenge [13].
- We measure and contrast their performances to reconstruct GRNs of DREAM4.

This research was supported by the National Research, Development, and Innovation Fund of Hungary under Grant TKP2021-EGA-02, and the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.

The paper will be organized as follows: Section two gives a high-level overview of the discussed algorithms highlighting their advantages when compared to the original NOTEARS method. In section three we summarize the biases of the methods and the simulated data and propose a probabilistic measure to evaluate the bias. In section four we explain and evaluate the experiments conducted. Finally, we end with a conclusion in section five.

II. Related work

Traditionally the structures of Bayesian-networks were identified by score-based or constraint-based methods [5]. The continuous optimization-based methods represented a new paradigm. Here the first method, which we will be using as a baseline is NOTEARS [7].

A. NOTEARS

NOTEARS describes the DAG as a Structural Equation Model (SEM), with the basic implementation assuming linear relationships between the variables. This way it describes an L2 reconstruction loss of the data:

$$\ell(W;X) := \frac{1}{2n}\|X - XW\|_F^2, \tag{1}$$

where W is the weighted adjacency matrix of the DAG. To enforce acyclicity they introduce the smooth constraint

$$h(W) := \operatorname{trace}\left(\mathrm{e}^{W\circ W}\right) - d = 0, \tag{2}$$

which minimizes the directed cycles in the graph, and it equals 0 if and only if the graph is acyclic. The objective function is given by taking the loss and adding the L1 norm of the weight matrix to induce sparsity in the matrix, which has a similar effect as the $h(W)$ constraint. The algorithm uses the augmented Lagrangian to turn the constrained problem into an unconstrained problem, and it can be solved by commonly used algorithms, like L-BFGS [14].

B. NOTEARS-MLP

NOTEARS-MLP [9] gives a more general assumption on the form of DAGs and uses a nonparametric approach, to learn the individual dependencies of the variables without the weighted adjacency matrix. To do this they use the partial derivatives of the functions (of dependencies) for every variable. Where the partial derivative is zero if and only if the dependence function is independent of the given variable. The partial derivatives are characterized as

$$[W(f)]_{kj} := ||\delta_k f_j||_{L^2}, \tag{3}$$

where f is the function of dependence, which in the linear case translates to the original NOTEARS problem. The method then uses Multi-Layer Perceptrons (MLPs) as plugin estimators for W, as their derivatives are easy to compute, and they can be easily optimized. Here we must mention NO-BEARS algorithm [15], which was specifically developed for structure learning in GRNs, however, it uses the assumption, that the f functions are polynomials of 3rd degree, which can easily be replicated by using MLPs as plugin estimators in the NOTEARS-MLP algorithm.

C. GOLEM

The GOLEM algorithm aims to introduce a soft DAG constraint instead of 2 and also apply a likelihood-based objective function instead of the regression-based one. Here the DAG constraints are not enforced, thus the search space involves graphs with directed cycles as well, but in most realistic cases the result will still be a DAG. The other change concerns the likelihood-based objective function, which is shown that combined with the soft (sparsity) constraint asymptotically results in a DAG as the result of optimization, and the optimization problem becomes simpler (GOLEM-NV):

$$\mathcal{L}_1(W;\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{d}\log\left(\sum_{k=1}^{n}\left(x_i^{(k)} - W_i^\top x^{(k)}\right)^2\right) - \log|\det(I-W)|. \tag{4}$$

In the case, where we assume that the variances are equal (GOLEM-EV), the objective function is even simpler:

$$\mathcal{L}_2(W;\mathbf{x}) = \frac{d}{2}\log\left(\sum_{i=1}^{d}\sum_{k=1}^{n}\left(x_i^{(k)} - W_i^\top x^{(k)}\right)^2\right) - \log|\det(I-W)|. \tag{5}$$

In both cases, the objective function is the sum of likelihoods, sparsity constraints, and soft DAG constraints. In the optimization phase, this results in a simpler objective function, which can be optimized with first-order methods, like Adam.

The final solution needs an iterative thresholding, to remove the smallest edges until the result is a DAG. In practice, this means that edges that have weights close to zero disappear, which would disappear NOTEARS as well if we apply thresholding (which is necessary in most cases).

D. DAG-NoCurl

DAG-NoCurl approaches the idea from a different perspective, it only allows search in the DAG space. It does this by defining the objective function to all graphs, minimizes it, and then projects the result to the DAG space, by using the gradient of a potential function as an equivalent to a graph in the DAG space.

These methods are all efficient in their time of convergence, and they are also usable in real-world scenarios, however, they do have inherent biases in their mechanisms [11] as well as in their evaluations [12].

III. Bias of the algorithms

This section gives an overview of the bias in the algorithms and a probabilistic metric to quantify this bias.

A. Non-scale invariance bias

Recently it has been shown [11] that most of these methods are not scale invariant. This means that the variance in the data can heavily influence the direction of the found edges, which can lead to wrongly identified networks.

This comes from the fact that the loss function of the methods consists of two parts: the loss for the fit and the

DAG constraint (and additional sparsity constraint). As these two work against each other, the optimization usually finds a good fit, and then constrains the graph to a DAG, when this happens the direction of the edges can be wrongly identified, as they are largely dependent on the variables' variance. This is especially noticeable in the two variable cases, where there is a linear relationship between them: $X_0 = \gamma X_1$. In this case, the loss is composed of

$$\|X_0\|^2 + \|X_1 - X_0 w_{01}\|^2 \quad \text{and} \\ \|X_1\|^2 + \|X_0 - X_1 w_{10}\|^2, \quad (6)$$

where the second term will cancel out and the loss will be dominated by the variances of the variables. The edge will be directed in the direction of $X_0 \to X_1$ in the case of $\gamma > 1$, as the variance of X_1 is larger. In general, the methods lead to variables with high variance behaving as sinks and variables with low variance behaving as sources in the network.

B. Posterior of nodes four Sourceness

A previous measure for this phenomenon has been proposed [12], called varsortability. Varsortability was used to identify why the methods can perform exceptionally well in generated datasets and gave worse performances on real data. Varsortability is calculated by dividing the number of edges leading from variables with lower variance to variables with higher variance by the number of all edges:

$$v := \frac{\sum_{k=1}^{d-1} \sum_{i \to j \in E^k} \text{ increasing } (\text{Var}(X_i), \text{Var}(X_j))}{\sum_{k=1}^{d-1} \sum_{i \to j \in E^k} 1} \in [0, 1]. \quad (7)$$

This gives us a measure for the whole graph about how well the variance can predict the topological ordering. The conclusion of the paper was that higher varsortability results in higher performance of continuous optimization methods.

Varsortability however is a simple metric, which does not consider the weights of the edges. This way we can give a more refined probabilistic metric to reason about each node, which can quantify how certain we are that a node is more source than sink.

In a probabilistic model, we can say, that the posterior probability of an edge $E(X_i, X_j)$ having the absolute edge weight $|w_{ij}|$ is

$$P(W_{ij} = |w_{ij}||data) = P(W_{ij} = |w_{ij}||W) P(W|data), \quad (8)$$

where $P(W_{ij} = |w_{ij}||W) = 1$ as it refers to a concrete model. We can use bootstrapping [16] to estimate the probability. We can assign uniform probabilities to the prior $P(W|data)$ and generate over-sampled datasets to construct models. If we assume that follows the normal distribution

$$P(W_{ij}|data) \sim \mathcal{N}(\mu_{ij}, \sigma_{ij}) \quad (9)$$

we get an approximation of the distribution by bootstrapping. To calculate the distribution of the outgoing edge weights of a node we can simply sum over the distributions of edges:

$$P(Out(N_i)|data) \sim \mathcal{N}(\sum_j \mu_{ij}, \sum_j \sigma_{ij}), \quad (10)$$

where $Out(N_j)$ equals the sum of the weights of the outgoing edges. Similarly for incoming edges:

$$P(In(N_j)|data) \sim \mathcal{N}(\sum_i \mu_{ij}, \sum_i \sigma_{ij}). \quad (11)$$

Finally, we can quantify "Sourceness" as a node having more outgoing edge weights than incoming ones. Which we can simply describe by the probability

$$P(S(N)|data) \sim \\ \mathcal{N}(\sum_j \mu_{ij} - \sum_i \mu_{ij}, \sum_j \sigma_{ij} - \sum_i \sigma_{ij}), \quad (12)$$

where

$$S(N) = Out(N_i) - In(N_i) \quad (13)$$

IV. Experiments

The algorithms were evaluated on the DREAM4 challenge datasets [13], which contain five graphs of GRNs with 10 genes and five with 100 genes. The networks are reconstructions of real GRNs of Escherichia coli, and they are measured as time series data, where the network is perturbed for a time, and then simulated for the same amount of time to see its reaction, thus simulating biological functions. The input of the algorithm is the full-time series data, without making a distinction on the timestamp the samples were generated at. The evaluation of the methods will consist of two parts, in the first part they are trained on the full datasets to measure the performance they can achieve and in the second part we try to measure the effect of biases on the performances with bootstrapping. In all cases ground truth networks are available for comparison [13].

A. Performances

In this section, we compare the performances of the methods, when they have access to the full dataset. We use the false discovery rate (FDR), true positive rate (TPR), false positive rate (FPR), structural hamming distance (SHD), and the number of predicted nonzero edges (NNZ) for comparisons of the algorithms. Following previous evaluation criteria [7], [8] we classify edges pointing in the wrong direction as misclassified (False Positive) samples. This way our evaluation stays comparable with other studies, but we advise to use the algorithms in a way where observational equivalence is taken into account. The methods only identify the Markov Equivalence classes of the graphs, which is why our the evaluation may yield more pessimistic results, than their real performance. In the case where the methods have access to the whole data, we average their performances on the five-five available DREAM4 models.

We can see the results of these experiments in Table I. As we can see none of the models can give a satisfying performance on the DREAM4 data. The best candidates are the NOTEARS and GOLEM-EV algorithms, however, almost all algorithms find too many edges, based on the SHD and NNZ metrics. Even if we lower the threshold of edge weights the first three metrics do not seem to change, and the graphs stay far away

TABLE I
PERFORMANCE OF ALGORITHMS ON THE DREAM4

		FDR	TPR	FPR	SHD	NNZ
10 genes	notears	0.693	0.633	0.492	19.200	22.000
	notears-mlp	0.806	0.511	0.659	26.600	25.200
	golem-ev	0.737	0.549	0.486	20.400	20.600
	golem-nv	0.941	0.212	0.384	20.800	12.600
	nocurl	0.790	0.508	0.645	25.200	25.200
100 genes	notears	0.978	0.096	0.099	651.400	481.000
	notears-mlp	0.976	0.084	0.083	574.000	400.600
	golem-ev	0.978	0.068	0.047	413.200	226.400
	golem-nv	0.985	0.048	0.008	238.400	40.600
	nocurl	0.979	0.099	0.109	695.600	529.000

from the ground truth. An interesting fact is the performance of GOLEM-NV, which is the only one that seems to improve when using it on larger data sizes when compared to the other algorithms.

B. Biases

In this section, we evaluate the bias described in section III for each algorithm. The algorithms are trained on 10 over-sampled datasets, to estimate the posterior of the edge weights, and the probabilities for S are calculated the previously described way.

Fig. 1. Evaluation of adjacency matrix of the first, 10 gene network with variances of the data.

If we want to qualitatively examine the bias described previously, we can examine the variances and the structures of the matrices, to see if indeed high-variance variables become sinks and low-variance ones become sources. In Figure 1 we can see that indeed higher variables with higher variances have more edges predicted as incoming and this results in more false positives in their column of the adjacency matrices.

The figure shows the NOTEARS algorithm's predictions with the first (number 1) graph of the 10-gene version of DREAM4, the variances are plotted on both sides to demonstrate the effect. This effect is present in most algorithm's predictions, we chose NOTEARS as the effect was examined here first, but later we will show that if we quantify this it applies to every listed algorithm.

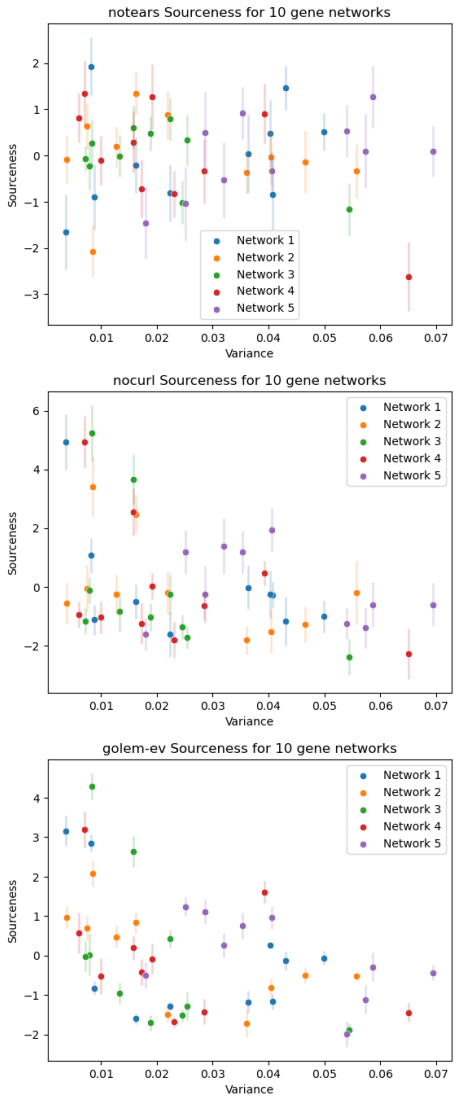


Fig. 2. Scatterplots of the Sourceness distributions (mean and standard deviation) and variances of nodes with different algorithms.

To quantify the bias, we can calculate S in the final graphs or calculate its distribution with bootstrapping, as it gives a more detailed description we chose the latter method. If we want to see if the variance influences the Sourceness of the variable, we can calculate the correlation of the expected value

of $S_i = S(N_i)$ to the variance of the node N_i to measure bias.

If we plot the Sourceness distributions of the nodes, we find that amongst the well-performing methods (NOTEARS, NOTEARS-MLP, GOLEM-EV, DAG-NoCurl) the ones with lower performance indeed have higher bias, as shown by the correlation of the Sourceness and variance. If we compute the correlation coefficients, we truly get the same results as our intuition about the plots.

TABLE II
CORRELATION OF VARIANCE AND SOURCENESS

		Average of $Corr(Var(N_i), S(N_i)$
10 genes	notears	-0.075
	notears-mlp	-0.795
	golem-ev	-0.516
	golem-nv	0.6
	nocurl	-0.455
100 genes	notears	0.031
	notears-mlp	-0.454
	golem-ev	-0.289
	golem-nv	0.032
	nocurl	-0.334

On Table II we can see that for most algorithms there is indeed a strong correlation between Sourceness and Variance, however this is not true in all cases. The Correlation is not strong in the case of NOTEARS, and varies in the case of GOLEM-NV, this means that their results should be explored further to find explanation for their performances.

V. CONCLUSION

In this paper, we have given a review of the most used structure learning algorithms in the continuous optimization domain. We gave a measure that can quantify a node's Sourceness. We have shown that in the algorithms that perform reasonably well in gene expression data, the correlation between Sourceness and the Variance of the variable is higher. The results presented in this paper should also be contrasted with an evaluation based on Markov equivalence classes. However the performances show, that these algorithms still need development, maybe even regularization for the described bias before they are usable in the domain.

REFERENCES

[1] E. Liu, L. Li, and L. Cheng, "Gene regulatory network review," in *Encyclopedia of Bioinformatics and Computational Biology*, S. Ranganathan, M. Gribskov, K. Nakai, and C. Schönbach, Eds. Academic Press, pp. 155–164. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128096338202185

[2] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, "Studying gene expression and function," in *Molecular Biology of the Cell. 4th edition*. Garland Science. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK26818/

[3] A. Brazma and J. Vilo, "Gene expression data analysis," vol. 480, no. 1, pp. 17–24. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0014579300017725

[4] J. Pearl, *Causality*. Cambridge University Press, google-Books-ID: f4nuexsNVZIC.

[5] A. Zanga, E. Ozkirimli, and F. Stella, "A survey on causal discovery: Theory and practice," vol. 151, pp. 101–129. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0888613X22001402

[6] Z. Jiang, C. Chen, Z. Xu, X. Wang, M. Zhang, and D. Zhang, "SIGNET: Transcriptome-wide causal inference for gene regulatory networks," pp. rs.3.rs–3 180 043. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10402199/

[7] X. Zheng, B. Aragam, P. K. Ravikumar, and E. P. Xing, "DAGs with NO TEARS: Continuous optimization for structure learning," in *Advances in Neural Information Processing Systems*, vol. 31. Curran Associates, Inc. [Online]. Available: https://papers.nips.cc/paper_files/paper/2018/hash/e347c51419ffb23ca3fd5050202f9c3d-Abstract.html

[8] I. Ng, A. Ghassami, and K. Zhang, "On the role of sparsity and DAG constraints for learning linear DAGs," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., pp. 17 943–17 954. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/d04d42cdf14579cd294e5079e0745411-Abstract.html

[9] X. Zheng, C. Dan, B. Aragam, P. Ravikumar, and E. Xing, "Learning sparse nonparametric DAGs," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 3414–3425, ISSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v108/zheng20a.html

[10] Y. Yu, T. Gao, N. Yin, and Q. Ji, "DAGs with no curl: An efficient DAG structure learning approach," in *Proceedings of the 38th International Conference on Machine Learning*. PMLR, pp. 12 156–12 166, ISSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v139/yu21a.html

[11] M. Kaiser and M. Sipos, "Unsuitability of NOTEARS for causal graph discovery when dealing with dimensional quantities," vol. 54, no. 3, pp. 1587–1595. [Online]. Available: https://doi.org/10.1007/s11063-021-10694-5

[12] A. Reisach, C. Seiler, and S. Weichwald, "Beware of the simulated DAG! causal discovery benchmarks may be easy to game," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., pp. 27 772–27 784. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/hash/e987eff4a7c7b7e580d659feb6f60c1a-Abstract.html

[13] A. Greenfield, A. Madar, H. Ostrer, and R. Bonneau, "DREAM4: Combining genetic and dynamic information to identify biological networks and dynamical models," vol. 5, no. 10, p. e13397. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2963605/

[14] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," vol. 45, no. 1, pp. 503–528. [Online]. Available: http://link.springer.com/10.1007/BF01589116

[15] H.-C. Lee, M. Danieletto, R. Miotto, S. T. Cherng, and J. T. Dudley, "Scaling structural learning with NO-BEARS to infer causal transcriptome networks," in *Biocomputing 2020*. WORLD SCIENTIFIC, pp. 391–402. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/9789811215636_0035

[16] S. Abney, "Bootstrapping," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, P. Isabelle, E. Charniak, and D. Lin, Eds. Association for Computational Linguistics, pp. 360–367. [Online]. Available: https://aclanthology.org/P02-1046

Modeling of Time-Dependent Behavior in Fault-Tolerant Systems

Dóra Cziborová, Richárd Szabó
Budapest University of Technology and Economics
Department of Measurement and Information Systems
Budapest, Hungary
Email: dora.cziborova@edu.bme.hu, szabor@mit.bme.hu

***Abstract*—Ensuring the correct operation of safety-critical systems often relies on model-driven engineering at design time and fault-tolerant solutions at operation time. Modeling and analyzing fault-tolerant systems is a challenging task: designing the complex control is tedious and we also need special modeling constructs to be able to represent the many aspects of operation. On the other hand, we need efficient algorithms to be able to verify the desired functionalities. In this paper, we extend our former work and develop time-dependent extensions to the modeling approach used in an automotive case study. Based on our case study, we evaluate the applicability of various model checking algorithms for the verification of systems with time-dependent behavior.**

***Index Terms*—fault-tolerant systems, timed behavior, industrial case study**

I. Introduction

Model-driven systems engineering is often employed in the design of critical systems not only to support the design process itself but the modeling artifacts can be used for verification purposes, such as the input of model-based testing or formal verification. Formal verification is a technique to mathematically represent the behavior of the system model and explore the space of the various behaviors to find erroneous situations. Formal verification is a computationally expensive task and successful verification often requires experts to design the verification models to fit the special needs of the verification engines/algorithms. Gamma is a modeling tool that supports the compositional design of engineering models and transforms them directly into the input of formal verification tools. Gamma supports the engineers in the development of models for formal verification, but it still has limited expressive power, so verification experts have to extend the systems models manually.

In our former work [1], we have designed the formal model of a subsystem of a steer-by-wire system in the Gamma framework. The subsystem contains fault-tolerance mechanisms to compensate for hardware faults and measurement uncertainties. Besides the complex logic, we have also represented the error propagation in the system, and various environmental effects were also considered and added as components constraining the possible behavior of the model. However, we faced problems when taking the environmental and physical constraints into account. Gamma does not support an expressive form for modeling time dependency. This led to the situation, that tedious work was needed to extend the model with components that are able to represent the necessary timing and scheduling conditions to gain realistic model behavior.

In this paper, we investigate how timing constructs in high-level engineering models can support the modeling of time-dependent behavior in fault-tolerant systems. We also show, how timed high-level models can be represented in the underlying formalism used for the verification. We also show the applicability of the extension by redesigning the case study to use timing to represent the physical and other environmental constraints in the systems.

The rest of the paper is structured as follows: Section II presents the modeling and verification framework, and the underlying formalism which can used to model safety-critical systems with time-dependent behaviors. We present the motivating challenges and the case-study system in Section III. A possible solution to the modeling problem is presented in Section IV and Section V presents the evaluation of the presented modeling approach. The evaluation is followed by presenting related modeling and verification tools in Section VI. Finally, Section VII provides the conclusion of the presented work and the plans for the future.

II. Background

A. The Gamma Statechart Composition Framework

There are several widely used modeling languages for defining the architecture and behavior of a system. These languages operate at both higher-level, such as UML, SysML, and AADL, as well as lower-level, such as the timed automata formalism of UPPAAL. Each of these languages has varying levels of precision in terms of 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 [2].

Various modeling tools have emerged to facilitate the creation of models with precise mathematical semantics. In our work, we chose the *Gamma Statechart Composition Framework* [3], expressly designed for modeling and verifying component-based reactive systems. The framework supports the hierarchical composition as a guiding design principle, wherein the behavior of the lowest-level components is modeled by statecharts. The hierarchical design approach enables

DOI https://doi.org/10.3311/MINISY2024-011

the engineers to break down the system into smaller, reusable subsystems, which can be composed using different composition semantics to define the system-level behavior [4], [5]: *Synchronous-reactive*, *Asynchronous-reactive*, *Cascade*, *Scheduled asynchronous-reactive*. The composition of the components introduces an additional modeling layer atop statecharts to describe the communication between components, making it well-suited for representing complex, distributed hierarchical systems.

1) Semantics of time in the Gamma Statechart Composition Framework: As presented in [4] the states of a *synchronous component* are allowed to track the values of *clock variables*, but the passage of time *may not* trigger the execution of components, thus the state of the component may only change according to the *transition function* of the component. This means that if after some passing of time a transition depending on the value of a clock variable becomes enabled, it can only be executed in the next *execution cycle*.

In the *asynchronous adapters* which are used to wrap *synchronous components* to create asynchronous components with event queues, the clocks are sources of events which emit ticks at given time units. Even though these ticks emit messages in fixed time units, the consumption of these messages can occur anytime later because of the lack of guarantees for execution time and frequency of asynchronous components.

In the Gamma Statechart Language (GSL) timed behaviors are modeled with deterministic *timeout declarations* in the statecharts and with *clock declarations* in the *asynchronous adapters*.

B. The Theta Model Checking Framework

For verifying the desired functionalities of the system, automated verification tools are used. Numerous model checking tools have been developed that support the verification of different kinds of properties.

In our case study, we focus on reachability properties, for which the *Theta Model Checking Framework* [6] provides abstraction refinement-based model checking algorithms. The analysis back-end of the framework supports various algorithms and abstract domains, in some cases aided by an SMT solver. The language front-end of the framework supports several modeling formalisms, including control flow automata, timed automata, and symbolic transition systems, the extension of the latter formalism was used in our case study as well.

C. The TXSTS Modeling Formalism

The *timed extended symbolic transition system* (TXSTS) formalism (proposed in [7] as an extension of [8] and [9]) 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. Models of the Gamma framework can be translated to TXSTS models. The TXSTS formalism is supported by the Theta framework, enabling the automated verification of models composed in the Gamma framework.

TXSTS models contain *data variables* to represent data-dependent behavior, while timed behavior is modeled by *clock variables*. Clock variables are continuous, non-negative 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 a 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 a operation set, the operation to be executed is selected from the operation set in a non-deterministic manner. The set of operations $\mathcal{O}$ contains the following types of operations:

- *Assumptions* have the form $[\varphi]$, where φ 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 φ 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* of the form $c := n$, where $c \in V_C$ and $n \in \mathbb{N}$;
- *Havoc operations* are denoted by $havoc(x)$, which is a non-deterministic 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;
- *Non-deterministic 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 non-deterministic manner;
- *Conditional operations*: $if(\varphi)\, then\, \{op_1\}\, else\, \{op_2\}$,

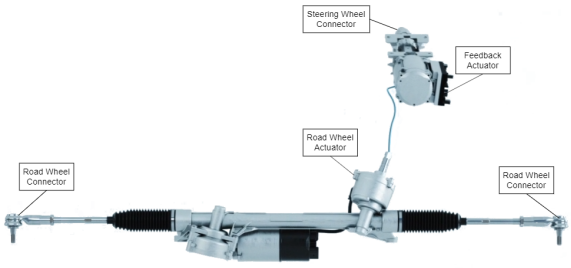


Fig. 1. Illustration of the physical structure of the Steer-by-Wire (SbW) system

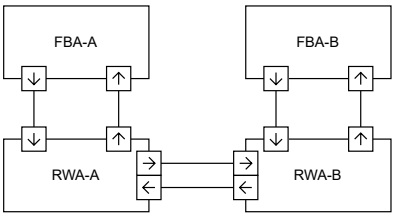


Fig. 2. Architecture of the SbW system

where φ 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, φ_a and φ_b are expressions that evaluate to integers, serving as the lower and upper bound for the loop variable i, and $op \in \mathcal{O}$.

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 τ 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, from 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 φ, otherwise $[\![[\varphi]]\!](\langle val_D, val_C\rangle) = \emptyset$;
- Havoc operations: $[\![havoc(x)]\!](\langle val_D, val_C\rangle) = \{\langle val'_D, val_C\rangle \mid val'_D(x) \in D,\ \forall x' \in V_D \backslash \{x\}\colon val'_D(x') = val_D(x')\}$, 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 $c \in V_C$;
- Non-deterministic 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. Case Study: Steer-by-Wire

In our previous work [1], we presented an approach to aid the design of fault-tolerant system architectures. The focus of this approach is to ensure that all distinct errors and failure scenarios are taken into account during the design of the fault tolerance measures of the system. We presented the applicability of our approach with a Steer-by-Wire (SbW) system from our industrial partner. In this paper, we reuse some components of the SbW case study to motivate our research. The angle of the steering wheel is used by the SbW system to provide steering functionality. The angle of the steering wheel is measured with multiple sensors in two different kinds of subsystems: the *Feedback Actuator (FBA)* and the *Road Wheel Actuator (RWA)* subsystems. The physical structure of the system can be seen in Figure 1.

The RWAs are located near the rack and turn the road wheels following the changes in the measured steering wheel angle. The FBAs contain actuators near the steering wheel to simulate the perception of mechanical steering for the driver by generating feedback torque. In the presented case study there are two FBAs and two RWAs as depicted in Figure 2.

The integration of the components provides the Absolute Steering Wheel Angle (ASWA) Measurement functionality, which is critical since its loss makes the vehicle unsteerable, and incorrect steering wheel angle measurements may cause incorrect steering and feedback.

A. Modeling and Analysis of Requirement Violation Scenarios

During the process presented in [1], we maintain four artifacts, representing different aspects of the system and under what assumptions the system must operate:

- The *(sub)system architecture* defines the structure and behavior of the (sub)system to analyze, including its error model.
- A *set of environmental assumptions* to rule out unrealistic behaviors of the system and its surroundings.
- A *set of known violation descriptions*, each of them covering a set of violation traces.
- The component-level *requirement* to analyse.

After the formalization of the artifacts we iteratively explore the different requirement violation scenarios. We analyse the traces given by the model checker and we classify them as one of the following categories:

- **Spurious**: either the formal model does not describe the real system realistically enough, or the set of environmental assumptions are too coarse and must be refined.
- **Unacceptable**: design error is found and the system architecture must be changed.
- **Acceptable**: a violation could be deemed *acceptable*, either because it has a low enough probability or because it will be taken care of by a higher-level controller.

TABLE I
IDENTIFIED TIME-DEPENDENT BEHAVIORS

Behavior	Discretization	Time-dependent Behavior
Components indefinitely waiting for response from permanently failed components	Permanently failed components explicit response signaling their failed state	Limiting the time a component waits for the response from other components
Transient failures simulating permanent failures	The number of possible transient failures is limited to a fixed finite number	The frequency of transient failures is limited by timing constraints
Starvation of components	The Environment model is constrained to send another event to a component only after all the other components have already received a message	The execution of the components is scheduled with timing constraints
No events happening in the system	The Environment Model is forced to send events in every step	The execution of the components is scheduled with timing constraints

B. Challenges

Handling spurious and unacceptable requirement violation scenarios is not an easy task, because it requires comprehensive knowledge of the system and the environment in which the system will operate.

In some cases we found that the environmental assumptions were not fair to the system, they introduced behaviors in the system where some components starved and could not execute their nominal behavior, or repeatedly happening transient failures could simulate permanent failures. In reality, these behaviors are often time-dependent, e.g. transient errors cannot occur more often than a given frequency, or the components of a distributed system are scheduled independently/concurrently of each other, hence starvation cannot happen.

In the model presented in the case study [1], we discretized all of the time-dependent behavior of the system due to shortcomings of the previously used modeling approach and the limitations of the model checking algorithms. In the following, we exemplify some aspects of the model that could be enhanced by using time in the modeling. Table I contains the identified discretized behavior which could be handled using the formalism and the related algorithms presented in Subsection II-C:

IV. MODELING TIME-DEPENDENT BEHAVIOR

In this section we overview how time-dependent behavior can be modeled using the TXSTS formalism. First we show how to give an upper bound for the time spent in states, then we show examples of how the TXSTS formalism can be used to model the behaviors indentified in Table I.

The *timeout declarations* of the GSL places a lower limit on the time elapsed before the given edge becomes fireable. However, currently no language construct can ensure an upper limit for the elapsed time.

To address the above problem of unboundedly elapsing time, we introduce *invariants* for states. The invariant of a state s in component M should always hold while s is the active state of M. Specifically, we use invariants of the form $t_s \leq n$, where $n \in \mathbb{N}$ and t_s is a variable representing time spent in state s.

We defined the following mapping between the statechart model and the TXSTS representation:

1) Timeout definitions are mapped to TXSTS clock variables and clock resets: $c := 0$, where $c \in V_C$.
2) Timeout triggers and the statements setting the value of the timeouts are mapped to TXSTS conditional operations. A transition with timeout trigger T in component M from state s to s' is translated to $if(c_i \sim k \wedge (state_M = s))\, then\, \{state_M := s'\}$, where $c_i \in V_C$ is the corresponding *clock variable*, $\sim \in \{<, \leq, =, \geq, >\}$, $state_M$ denotes the data variable in the TXSTS that represents the active state of M and $k \in \mathbb{Z}$ is the value set for T in the *set timeout statement*.
3) Invariants are translated to assumptions. An invariant I on a state s of component M gets translated to the assumption $[(state_M = s) \Rightarrow \varphi_I]$, where $state_M$ denotes the data variable in the TXSTS that represents the active state of M, and $[\varphi_I]$ is the translation of I.

A. Communication

When components communicate, we expect that the data sent by one component is eventually received by the other component. In the models, we set a timeout for these communications, i.e. we expect that sent data is received under a given time limit. This is modeled using $timeout$ transitions to states representing that the communication failed. The timeout is set to a value t_1.

In order to limit the waiting time before transitioning to the state representing failed communication, we use invariants. In each component, the invariant is introduced in the state waiting for the result of the crosscheck, and sets a fixed upper bound for the $crosscheckTimeout$ variable, which is reset to zero at entry, and represents the time spent in the state. The bound is set to a value $t_1 + \delta$, which is greater than or equal to the timeout of the communication but in the same order of magnitude. If $\delta = 0$, then the transition must fire immediately on timeout.

The timeouts and invariants to model the communication between the components are depicted in Figure 3 and the excerpt of the TXSTS model is shown in Listing 1.

B. Transient Failures

The modeled system in its working state occasionally encounters a transient failure, which is then healed. This behavior

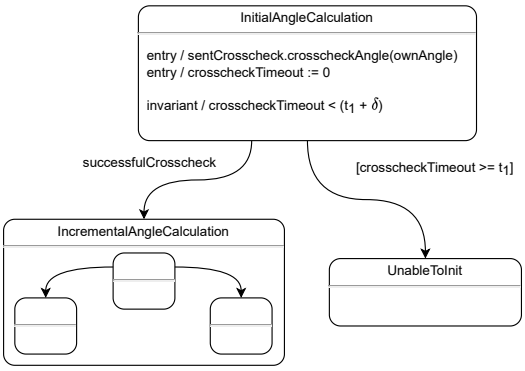


Fig. 3. Modeling communication using invariants

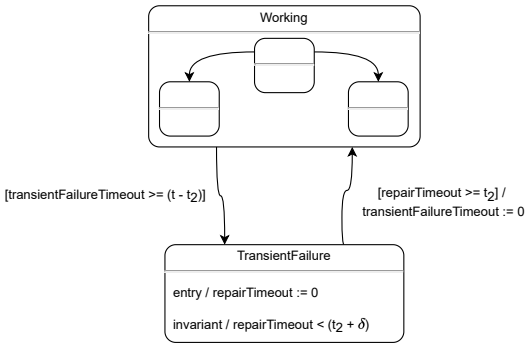


Fig. 4. Modeling transient failures using invariants

```
var crosscheckTimeout : clock
var successfulCrosscheck : boolean = false
// ...
trans {
  __delay;
  assume (state_of_M == InitialAngleCalculation) =>
      crosscheckTimeout < (t1 + delta));
  // ...
  if ((state_of_M == InitialAngleCalculation)
      && successfulCrosscheck) {
    state_of_M := IncrementalAngleCalculation;
  } else {
    if ((state_of_M == InitialAngleCalculation)
        && (t1 <= crosscheckTimeout)) {
      state_of_M := UnableToInit;
    }
  }
  // ...
}
init { successfulCrosscheck := false; /* ... */ }
env { /* ... */ }
```

Listing 1. TXSTS model of the of the communication

is represented in the model by a transition to (and from) a transient failure state.

The frequency of transient failures in the analyzed system is at most t. This is conveyed in the model by a timeout on the edge from the working state to the transient failure state.

The frequency of repairs is at most t_2, which is modeled by a timeout of t_2 on the edge from the transient failure state to the working state. The timeout on the transition from the working to the transient failure state is therefore $t - t_2$.

Transient failures of the system should be healed in a limited time. This is modeled by an invariant on the transient failure state, that limits the time spent in that state to $t_2 + \delta$, a value greater than or equal to t_2 but in the same order of magnitude.

The timeouts and invariants to model the transient failures of the components are depicted in Figure 4.

C. Handling Spurious Nondeterminism

As the model contains spurious nondeterminism, the fair execution of components is not ensured, one component may be executed infinitely many times before the other one is executed causing starvation in the system. Another example of spurious nondeterminism is when no event happening in the system. Both of these spurious behaviors could be handled by constraining the scheduling of the components with state invariants using time similar to the ones presented above.

V. Evaluation

To evaluate our approach, we choose a subset of the model presented in Section III, using two RWA components connected to each other. We modeled the communication of the components and the transient failures of the system with the modeling constructs presented in Section IV and handled the spurious nondeterministic scheduling by selecting the *synchronous-reactive* composition of the components. The model contains 24 states and 46 variables, out of which 6 were clock variables.

We evaluated our approach using 24 reachability properties, covering the reachability of each state in the model, with six different configurations of the Theta model checker.

Two configurations (TRANSF in Table II) applied a transformation of the TXSTS model to the XSTS formalism by representing time using rational variables, as described in [7], then performed the analysis using a CEGAR algorithm [10]. The two configurations differ in the abstraction used in the CEGAR algorithm (EXPL and EXPLPRED in Table II).

The other four configurations (SPLIT in Table II) employed an algorithm that combines CEGAR (with the same two abstractions as in the previous case) and lazy abstraction [11], as described in [12]. This algorithm contains an additional control flow splitting step presented in [7] to counteract the limitations of the existing algorithms when analyzing complex control flows. The control flow splitting algorithm may be parameterized with an additional option (FILTER in Table II) that filters out infeasible control flows using an SMT solver.

The measurements were conducted with tasks limited to 5 CPU cores, 10 minutes of runtime and 15 GB of memory.

The results of the analysis are shown in Table II. Out of the total 144 runs, only 10 resulted in timeout, these are the reachability of the innermost states, which require the most events to reach. However, multiple configurations could successfully verify all 24 reachability properties and return feasible traces to the states described by the properties. This shows that timed verification algorithms can handle the complexity of the subsystem of the SbW system. However, we

TABLE II
RESULTS OF THE ANALYSIS

Configuration	Success rate	Mean CPU time in seconds
TRANSF_EXPL	75%	2.385
TRANSF_EXPLPRED	100%	4.942
SPLIT_EXPL	100%	11.093
SPLIT_EXPLPRED	100%	11.997
SPLIT_FILTER_EXPL	83%	8.320
SPLIT_FILTER_EXPLPRED	100%	14.446

have to further improve the algorithms to verify the system to its full extent.

VI. Related Work

Modeling time-dependent behavior is a well-known problem in the literature. One of the most well-known tools in the model checking community is the UPPAAL [13] which uses timed automata to verify real-time systems and communication protocols. UPPAAL models can contain real-valued clock and state invariants to represent timing constraints, similar to our approach. However, timed automata are a low-level formalism, which is generally far from the models the engineers use to describe their system, creating an abstraction gap.

In [14] the authors present an extension of the MARTE UML profile. This extension defines a formal time model, which can be used to define timing constraints of structure and behavior models. However, their work focuses on finding inconsistencies between the timing constraints of the structural and behavioral models.

In [15] the input language of the nuXmv model checker [16] was extended to enable the description of symbolic synchronous timed transition systems, where initial conditions, transitions, and invariants must be described using formulas.

The authors of [17] present an iterative methodology to verify safety-critical software in the nuclear engineering domain, where the timing of the software is a critical aspect of the behavior. In their verification workflow, the authors model their system in two different model checking frameworks, UPPAAL [13] and NuSMV [18], During the modeling with NuSMV the authors faced similar challenges as we did in our previous work, thus they also had to discretize the time-dependent behaviors of their system.

VII. Conclusion and Future Work

In this paper, we extended our former work with handling time-dependent behaviors. We presented examples of the usage of the TXSTS formalism to model time-dependent behavior and timing constraints in complex industrial systems and evaluated our work on the subset of an industrial case-study system. The results of the evaluation showed us that our approach is usable for modeling these kinds of problems and it is worth investigating the possible applications of the formalism further.

The time-dependent functionalities were modeled using the syntax of the TXSTS formalism due to the limitations of the Gamma framework, thus to ease the modeling workflow, we plan to extend the Gamma framework to have a more expressive time formalism. We also plan to evaluate the algorithms behind the TXSTS formalism on the full model of the case study extended with time-dependent behaviors and on other system models from our industrial partners.

References

[1] 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*.

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

[3] 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.

[4] 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.

[5] 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.

[6] T. Tóth, Á. Hajdu, A. Vörös, Z. Micskei, and I. Majzik, "Theta: A framework for abstraction refinement-based model checking," in *FMCAD*, 2017, pp. 176–179.

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

[8] M. Mondok, "Extended symbolic transition systems: an intermediate language for the formal verification of engineering models," Scientific Students' Association Report, Budapest University of Technology and Economics, 2020.

[9] 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.

[10] E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, "Counterexample-guided abstraction refinement," in *CAV*. Springer, 2000, pp. 154–169.

[11] T. Tóth and I. Majzik, "Lazy reachability checking for timed automata using interpolants," in *FORMATS*, ser. LNCS, vol. 10419. Springer, 2017, pp. 264–280.

[12] D. Cziborová and B. Á. Vizi, "Abstraction-based model checking techniques for real-time systems," Scientific Students' Association Report, Budapest University of Technology and Economics, 2022.

[13] J. Bengtsson, K. Larsen, F. Larsson, P. Pettersson, and W. Yi, "Uppaal—a tool suite for automatic verification of real-time systems," in *International hybrid systems workshop*. Springer, 1995, pp. 232–243.

[14] S. Ni, Y. Zhuang, Z. Cao, and X. Kong, "Modeling dependability features for real-time embedded systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 2, pp. 190–203, 2015.

[15] A. Cimatti, A. Griggio, E. Magnago, M. Roveri, and S. Tonetta, "Extending nuXmv with timed transition systems and timed temporal properties," in *Computer Aided Verification*, I. Dillig and S. Tasiran, Eds. Cham: Springer International Publishing, 2019, pp. 376–386.

[16] R. Cavada, A. Cimatti, M. Dorigatti, A. Griggio, A. Mariotti, A. Micheli, S. Mover, M. Roveri, and S. Tonetta, "The nuXmv symbolic model checker," in *Computer Aided Verification*, A. Biere and R. Bloem, Eds. Cham: Springer International Publishing, 2014, pp. 334–342.

[17] J. Lahtinen, J. Valkonen, K. Björkman, J. Frits, I. Niemelä, and K. Heljanko, "Model checking of safety-critical software in the nuclear engineering domain," *Reliab. Eng. Syst. Saf.*, 2012.

[18] A. Cimatti, E. Clarke, F. Giunchiglia, and M. Roveri, "NuSMV: a new Symbolic Model Verifier," in *Proceedings Eleventh Conference on Computer-Aided Verification (CAV'99)*, ser. Lecture Notes in Computer Science, N. Halbwachs and D. Peled, Eds., no. 1633. Trento, Italy: Springer, July 1999, pp. 495–499.

Efficient Manipulation of Logical Formulas as Decision Diagrams

Milán Mondok, Vince Molnár
Budapest University of Technology and Economics
Department of Measurement and Information Systems
Email: mondok@mit.bme.hu, molnarv@mit.bme.hu

***Abstract*—Constraint solving and the manipulation of Satisfiability Modulo Theories (SMT) formulas is a fundamental task in symbolic model checking. SMT solvers have proven to be efficient tools in exploiting the high expressive power and flexibility offered by SMT formulas. Decision diagram based approaches have also gained popularity for their capability to represent all solutions in a compact way and are used in numerous efficient algorithms. However, there is a gap between these two approaches.**

In this paper, we present a novel data structure that can combine the flexibility of SMT formulas and the power of SMT solvers with the efficient representation of the solutions. This data structure is a blend of decision diagrams and SMT formulas: it allows us to handle logical formulas as decision diagrams, leveraging both the power of SMT solvers and the advantages of diagram representation. The compatibility with decision diagrams allows the integration of efficient algorithms working on the two different representations. When discussing the benefits of this approach, we also emphasize how the intersection operation - a common problem in constraint solving - can be carried out more efficiently using lazy evaluation. We can also build on the same advantage in transitive closure calculations.

***Index Terms*—SMT, decision diagram, symbolic, model checking**

I. Introduction

Constraint solving and the manipulation of logical formulas is a fundamental task in symbolic model checking. First order logic [2] is an expressive language widely used in computer science, but its satisfiablity is undecidable in the general case. Satisfiability Modulo Theories (SMT) [1] solvers offer efficient algorithms for practical subsets (theories) [2] of first order logic and have proven to be efficient tools in exploiting the high expressive power and flexibility offered by first order logic formulas. Decision diagram based approaches have also gained popularity for their capability to represent all solutions in a compact way and efficient algorithms [3] are known that manipulate the sets directly in the encoded form. However, there is a gap between these two approaches.

In this paper, we present a novel data structure called *substitution diagram*, that can combine the flexibility of first order logic formulas and the power of SMT solvers with the efficient representation of the solutions. This data structure is a blend of decision diagrams and logical formulas: it allows us to handle logical formulas as decision diagrams, leveraging both the power of SMT solvers and the advantages of diagram representation. The compatibility with decision diagrams allows the integration of efficient algorithms working on the two different representations. When discussing the benefits of this approach, we also emphasize how the intersection operation - a common problem in constraint solving - can be carried out more efficiently using lazy evaluation. We can also build on the same advantage in transitive closure calculations.

Supported by the **ÚNKP-23-3-I-BME-8** New National Excellence Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund

II. Background

In this section, we briefly discuss the fundamentals of two different symbolic approaches, namely *satisifiability modulo theories* and *decision diagrams*, that we build on in the later sections of this paper.

A. Satisifiability Modulo Theories

First order logic (FOL) [2] is a rich language that is commonly used in the context of formal verification to capture constraints and reason about system designs. It extends propositional logic with predicates, functions and quantifiers. Consequently, FOL formulas can not only evaluate to truth values, but to any abstract concept, e.g., integers, animals, names. The satisfiability of first order logic formulas is undecidable in the general case.

First order theories [2] formalize the structures of a domain to enable reasoning about them formally. These theories can for example capture the concepts of equality ($v_1 = v_2$), linear arithmetic ($2x \leq (y + 5)$), bit vectors ($(a >> b)$ & c) or arrays ($a[i] = b[0]$). While satisfiability in FOL is undecidable in general, it is decidable in many practical first order theories (or in their fragments).

The satisfiability modulo theories (SMT) [1] problem is to decide if a given formula is satisfiable in a given theory (or combination of theories). Since Boolean satisfiablity is already NP-complete, the SMT problem is typically NP-hard even in decidable theories. Despite this, more and more SMT solvers are known that can efficiently solve the SMT problem for a practical set of inputs. Solvers like Z3 [4] and cvc5 [5] are used as foundational building blocks across a wide range of applications, ranging from theorem proving and model checking to software testing.

In this work, we consider SMT formulas that can contain Boolean literals ($\top, \bot$), Boolean connectives ($\neg, \wedge, \vee, \rightarrow, \leftrightarrow$),

DOI https://doi.org/10.3311/MINISY2024-012

integer variables and literals, linear arithmetic ($+, -, *, <, >, =$), but the presented approach can work with any first order theory given a suitable SMT solver. Let the syntax of quantifier-free first order formulas be the following:

- the Boolean literals $\top$ and $\bot$ are formulas;
- the natural numbers $\mathbb{N}$ are formulas;
- variables $x \in V$ are formulas;
- if φ, ψ are formulas, then $\varphi \wedge \psi, \varphi \vee \psi, \neg\varphi, \varphi \rightarrow \psi, \varphi \leftrightarrow \psi$ are also formulas;
- if φ, ψ are formulas, then $\varphi + \psi, \varphi - \psi, \varphi * \psi, \varphi < \psi, \varphi > \psi, \varphi \leq \psi, \varphi \geq \psi, \varphi = \psi$ are also formulas;

Let $\mathcal{L}$ denote the set of first order logic formulas that can be generated with the rules above. Let $\varphi[v/\psi]$ denote the operation of substituting all appearances of the variable $v \in V$ in the formula $\varphi \in \mathcal{L}$ with the formula $\psi \in \mathcal{L}$. For example, $(x < 2 \wedge y = x + 1)[x/0]$ is $(0 < 2 \wedge y = 0 + 1)$.

B. Decision Diagrams

Decision diagrams offer a compressed representation for sets and relations. Binary decision diagrams (BDD) [3], which are essentially binary decision trees with all the identical subtrees merged are commonly used to represent Boolean functions. Multi-valued decision diagrams [7] generalize this idea and can be used to reason about variables with larger discrete domains, e.g. integers.

An ordered quasi-reduced multi-valued decision diagram (MDD) [8] over a set of variables V ($|V| = K$), a variable ordering, and domains D is a tuple $\mathit{MDD} = (\mathcal{V}, \mathit{lvl}, \mathit{children})$ where:

- $\mathcal{V} = \bigcup_{k=0}^{K} \mathcal{V}_k$ is the set of *nodes*, where items of $\mathcal{V}_0$ are the *terminal* nodes $\mathbf{1}$ and $\mathbf{0}$, the rest ($\mathcal{V}_{>0} = \mathcal{V} \setminus \mathcal{V}_0$) are *internal* nodes ($\mathcal{V}_i \cap \mathcal{V}_j = \emptyset$ if $i \neq j$);
- $\mathit{lvl} : \mathcal{V} \rightarrow \{0, 1, \ldots, K\}$ assigns non-negative *level numbers* to each node, associating them with variables according to the variable ordering (nodes in $\mathcal{V}_k = \{n \in \mathcal{V} \mid \mathit{lvl}(n) = k\}$ belong to variable x_k for $1 \leq k \leq K$ and are terminal nodes for $k = 0$);
- $\mathit{children} : \mathcal{V}_{>0} \times \mathbb{N} \rightarrow \mathcal{V}$ defines edges between nodes labeled with elements of $\mathbb{N}$ (denoted by $n[i] = \mathit{children}(n, i)$, $n[i]$ is left-associative), such that for each node $n \in \mathcal{V}_k$ ($k > 0$) and value $i \in D(x_k) : \mathit{lvl}(n[i]) = \mathit{lvl}(n) - 1$ or $n[i] = \mathbf{0}$; as well as $n[i] = \mathbf{0}$ if $i \notin D(x_k)$;
- for every pair of nodes $n_1, n_2 \in \mathcal{V}_{>0}$, if for all $i \in \mathbb{N} : n_1[i] = n_2[i]$, then $n_1 = n_2$, meaning the equivalence of two nodes is decided recursively through their outgoing edges and the equivalence of their children.

An MDD node $n \in \mathcal{V}_k$ encodes a set of vectors $S(n)$ over variables $V_{\leq k}$ such that for each $\vec{s} \in S(n)$ the value of $n[\vec{s}[k]] \cdots [\vec{s}[k]]$ (recursively indexing n with components of $\vec{s}$) is $\mathbf{1}$ and for all $\vec{s} \notin S(n)$ it is $\mathbf{0}$. When speaking about an MDD encoding the set $S(n)$, we mean the pair (MDD, n) and we refer to n as the root node of the MDD. The size of such an MDD is defined as the number of nodes reachable from the root node and is denoted by $|n|$.

Figure 1 shows an example for the graphical representation of an MDD. The circles represent internal nodes, while the squares represent terminal nodes. The edges in the figure correspond to the *children* edges. The terminal $\mathbf{0}$ node and all edges leading to it are omitted for readability. The MDD on the left encodes 3 vectors: $(x = 0, y = 0)$, $(x = 1, y = 1)$ and $(x = 2, y = 2)$.

An interesting property of decisions diagrams is that the number of the nodes does not grow proportionally with the size of the encoded set. The size of the MDD can even decrease when a new state is added because of the exploited regularities. This is illustrated in Fig. 1, where the MDD on the left encodes 3 vectors with 4 nodes, while the MDD on the right encodes 9 vectors with only 2 nodes.

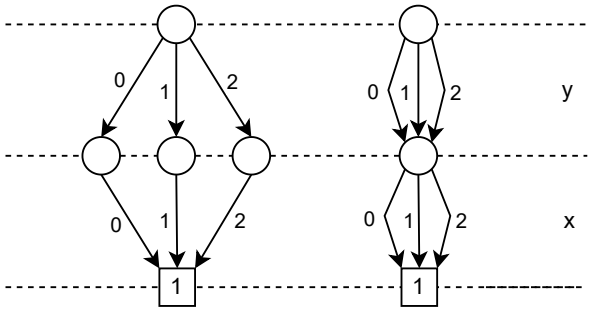


Fig. 1. An MDD with 4 nodes encoding 3 vectors (left) and and MDD with 2 nodes encoding 9 vectors (right).

III. Substitution Diagrams

We present a novel data structure called *substitution diagram*, which allows for lazy evaluation of SMT formulas in a decision diagram like representation. The idea behind substitution diagrams comes from the observation that first order logic formulas and the variable substitution operation ($\varphi[x/\psi]$) span a structure that is very similar to decision diagrams. A substitution diagram (SMDD) over a set of variables V ($|V| = K$), a variable ordering, and domains D is a tuple $\mathit{SMDD} = (\mathcal{V}, \mathit{lvl}, \mathit{children}, \mathit{semantics})$, where:

- $\mathcal{V} \subseteq \bigcup_{k=0}^{K} \mathcal{V}_k \subseteq (\mathcal{L} \cup (\mathcal{L} \times V))$ is the set of *nodes*, where $\mathcal{L}$ denotes the set of quantifier-free first order logic formulas over V', where $V \subseteq V'$ (this means that the formulas are allowed to contain additional variables compared to V). Items of $\mathcal{V}_0 \subseteq \mathcal{L}$ are the *terminal* nodes, which are first order logic formulas. *Internal* nodes $(\mathcal{V}_{>0} = \mathcal{V} \setminus \mathcal{V}_0) \subseteq \mathcal{L} \times V$ are formula-variable pairs;
- $\mathit{lvl} : \mathcal{V} \rightarrow \{0, 1, \ldots, K\}$ assigns non-negative *level numbers* to each node, associating them with variables according to the variable ordering. The nodes of level k are denoted with $\mathcal{V}_k = \{n \in \mathcal{V} \mid \mathit{lvl}(n) = k\}$. For all $1 \leq k \leq K$ and $n = (\varphi, x) \in \mathcal{V}_k$, $x = x_k$, meaning that the nodes of level k can only contain the variable associated with their level;
- $\mathit{semantics} : \mathcal{L} \rightarrow \mathcal{L}$ assigns semantically equivalent first order logic formulas to first order logic formulas. This can be thought of as an operation that brings the formulas to some normal form;

- *children* : $\mathcal{V}_{>0} \times \mathbb{N} \to \mathcal{V}$, define edges between nodes labeled with elements of $\mathbb{N}$ (denoted by $n[i] = \textit{children}(n, i)$, $n[i]$ is left-associative). The presence of the edges is defined as $n_1[i] = n_2$, iff $\textit{semantics}(\varphi_1[x_k/i]) = \varphi_2$, where $n_1 = (\varphi_1, x_k) \in \mathcal{V}_k$, $n_2 = (\varphi_2, x_{k-1}) \in \mathcal{V}_{k-1}$, $k > 1$, $i \in D(x_k)$, meaning an edge points from the node n_1 to the node n_2 with the label i if and only if substituting all appearances of the variable x_k in the expression of n_1 with the literal i and bringing this expression to a normal form results in the expression of n_2. In the case of $k = 1$, $n_1[i] = n_2$ iff $\textit{semantics}(\varphi_1[x_1/i]) = n_2$, where $n_1 = (\varphi_1, x_1) \in \mathcal{V}_1$, $n_2 \in \mathcal{V}_0$, $i \in D(x_1)$, the edge points to a terminal node in this case;
- for every terminal node $\varphi \in \mathcal{V}_0$, $\textit{semantics}(\varphi) = \varphi$, and for every internal node $(\varphi, x) \in \mathcal{V}_{>0}$, $\textit{semantics}(\varphi) = \varphi$, meaning the formulas of the nodes must be in the normal form;
- for every pair of nodes of the same level $n_1 = (\varphi_1, x_k) \in \mathcal{V}_k$, $n_2 = (\varphi_2, x_k) \in \mathcal{V}_k$, if $\textit{semantics}(\varphi_1) = \textit{semantics}(\varphi_2)$, then $\varphi_1 = \varphi_2$, meaning two nodes with different formulas that evaluate to the same normal form are not allowed on the same level. This means that contrary to decision diagrams, the equivalence of nodes in substitution diagrams is decided through the *semantics* function.

A node $n = (\varphi, x_k) \in \mathcal{V}_k$ encodes the set of satisfying partial assignments to its formula φ, such that for an assignment $(i_k, i_{k-1}, \ldots, i_0) \in D(x_k) \times D(x_{k-1}) \times \ldots \times D(x_0)$, $\varphi[x_k/i_k] \cdots [x_0/i_0] = \varphi_\top$ (i.e. substituting the values of the assignment into φ results in the formula $\varphi_\top$) iff the value of $n[i_k] \cdots [i_0]$ (recursively indexing n with the values of the assignment) is $\varphi_\top$.

Consider the substitution diagram in Fig. 2 as an example. We start the construction of the diagram top-down with the formula $(y = 0 \wedge x = y) \vee (y = 1 \wedge x \geq 0 \wedge x < y)$ and the top-level variable y. This formula can only be satisfied with two values for the variable y, 0 and 1 (which can be enumerated using an SMT solver), so the node is going to have two children. If we substitute the appearances of y with the literal 0, we get the formula $(0 = 0 \wedge x = 0) \vee (0 = 1 \wedge x \geq 0 \wedge x < 0)$. We can replace $0 = 0$ with $\top$ and remove it from the conjunction. Similarly, on the right side of the disjunction, we can replace $0 = 1$ with $\bot$ and replace the conjunction with $\bot$. After removing the right operand $\bot$ from the disjunction, we arrive at the formula $x = 0$ and create a new node for it. This formula can only be satisfied with the assignment $x = 0$, which results in the formula $\top$, so the node only has one outgoing edge that leads to the terminal $\top$ node.

Semantic or Syntactic Equivalence

Deciding whether two nodes are equivalent and can be merged is not straightforward in a substitution diagram. Consider the two nodes $x = 0$ and $x \geq 0 \wedge x < 1$ in Fig. 2. The two formulas are syntactically different, but both can only be satisfied with the model (assignment) $x = 0$. In case of an MDD, equivalence is decided semantically, i.e., through the outgoing edges and children of the two nodes. We can see that in the equivalent MDD, the two nodes are merged.

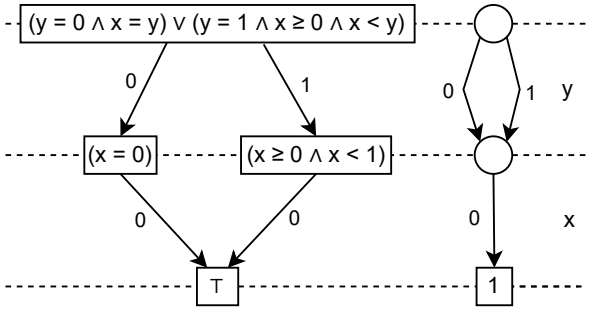


Fig. 2. Substitution diagram (left) and the equivalent MDD (right).

Semantic equivalence in the case of two SMT formulas means that exactly the same models satisfy them. However, exact semantic uniqueness among the nodes of the substitution would be too costly to maintain because it would require a solver call for each existing node on a level of the diagram whenever a new node is added. A good practical compromise could be to rely on syntactic equivalence and a low-cost transformation that brings the formulas to some sort of normal form before comparing them. We denote this transformation with the name *semantics*. The size of a substitution diagram is drastically affected by the precision of this transformation, however, in most cases the more precise the transformation is, the costlier it is to calculate it. In the extreme case where the *semantics* function is maximally precise (meaning it maps all semantically equivalent formulas to the same formula), the constructed substitution diagrams have the same number of nodes as the equivalent decision diagrams.

Lazy Evaluation

The most promising feature of substitution diagrams is that they allow for lazy evaluation of paths (solutions). When the diagram is initialized, only the root node (the node on the highest level) is stored, further nodes can be obtained by either enumerating children using an SMT solver or by querying the presence of either singular edges or paths consisting of multiple edges. Whenever the presence of an edge is confirmed, it can be cached, so that querying an SMT solver will not be required if the edge is needed again.

Operations on Substitution Diagrams

Substitution diagrams are intentionally compatible with decision diagrams in the sense that a substitution diagram $\textit{SMDD} = (\mathcal{V}, \textit{lvl}, \textit{children}, \textit{semantics})$ can be interpreted as a relaxed decision diagram $\textit{MDD} = (\mathcal{V}', \textit{lvl}', \textit{children}')$, where:

- $\mathcal{V}'_{>0} = \mathcal{V}_{>0}$, $\textit{lvl}' = \textit{lvl}$, $\textit{children}' = \textit{children}$, i.e. the internal nodes, the level numbering and the edges are analogous in the two representations;
- the terminal node $\mathbf{0} = \bot$, and all satisfiable terminal nodes $\varphi_\top \in \mathcal{V}_0$, $\varphi_\top \nrightarrow \bot$ can be interpreted as the terminal node $\mathbf{1}$;

- the semantic uniqueness of the nodes is not guaranteed, this depends on the precision (i.e. if it maps all semantically equivalent formulas to the same formula) of the *semantics* function.

This means that substitution diagrams can be used in operations that are defined on MDDs. This compatibility also allows us to mix the two representation modes in these operations such that one of the operands is an MDD, and the other operand is a substitution diagram. We can exploit the advantages offered by the lazy evaluation of substitution diagrams in these operations. A fundamental operation of constraint solving, the intersection operation returns the intersection of the sets encoded by two diagrams (i.e. the vectors that are contained by both diagrams). Typically, this is done by iterating over the edges of one of the diagrams and checking if the edge is present in the other diagram as well. When calculating the intersection of an SMDD and an MDD, by consciously choosing the MDD as the diagram that we iterate through and only checking the presence of those edges in the SMDD that are present in the MDD, we can avoid iterating over the potentially infinite solutions of the SMDD.

IV. Evaluation

We implemented a normal form transformation (*semantics* function) that (1) is carried out together with the variable substitution ($\varphi[x/\psi]$) operation such that it has extra information about the substituted variable, (2) uses simple rewriting rules to remove unnecessary operands from the expressions (for example replaces conjunctions like $\bot \wedge \varphi$ with $\bot$, or disjunctions like $\bot \vee \varphi$ with φ), (3) removes operators that can be expressed with other operators (for example replaces $\leq$ with a negation and $>$, or replaces implications of the form $\varphi \rightarrow \psi$ with $\varphi \vee \psi$), (4) is carried out purely syntactically without the use of an SMT solver.

To evaluate the precision of our transformation, we assembled a benchmark set of 10000 SMT formulas created from the transition relations of randomly generated symbolic transition system [6] models, which describe the relationship between the values of the variables before and after executing transitions, describing operations including but not limited to assignments, guards and various control structures (e.g. if-else, nondeterministic choice, sequence). The expressions can refer to 10 variables and the values of the variables are bound between 0 and 5 (which means that a formula can have at most $6^{10} = 60466176$ satisfying assignments). The expressions are generated such that the maximum depth of expression embedding is 13.

Out of the 10000 generated formulas, 3789 were satisfiable. For each satisfiable formula, we generated a random variable ordering and constructed a substitution diagram (SMDD) and a decision diagram (MDD) using the same variable ordering. We plotted the size difference between the two diagram representations on the scatterplot in Fig. 3. Each dot of the diagram corresponds to a satisfiable formula, with the number of nodes in the MDD representation plotted on the x axis and the number of nodes in the SMDD representation plotted on the y axis. Both axes are plotted using a logarithmic scale for better readability. The color of the dots corresponds to the number of vectors encoded by the diagrams (i.e. the number of possible satisfying assignments to the formula). The first, second and third quartiles of the number of vectors encoded by the formulas are, respectively, $q_1 = 1714$, $q_2 = 5184$ and $q_3 = 17140$, meaning more than half of the diagrams encoded more than 1700 and less than 17200 vectors. The largest number of vectors encoded by a diagram was 4.5 million. We can see that while the MDD is representation is always smaller or equal in size to the SMDD representation, the latter is not significantly larger in most cases. This indicates that our normal form transformation can provide a good approximation of semantic equality and should be studied further. Note that the transformation to an MDD representation is only possible in case of a finite number of solutions, but an SMDD can encode infinite solutions.

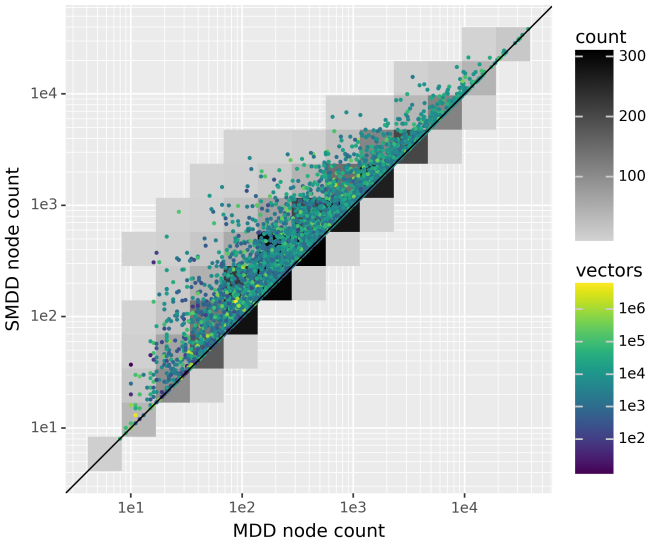


Fig. 3. SMDD and MDD node count for 3789 randomly generated SMT formulas. The color of the dots corresponds to the number of vectors encoded by the diagrams.

V. Conclusion

In this paper, after introducing the fundamentals of satisfiability modulo theories (SMT) and decision diagrams (MDD), we presented a novel data structure called substitution diagram (SMDD). This data structure combines the advantages of decision diagram representations with the power offered by SMT solvers to allow for efficient manipulation of logical formulas. We evaluated to compactness of substitution diagram representation on randomly generated SMT formulas and found that our approach can approximate decision diagrams promisingly.

As future work, we plan on exploring the possibilities of applying substitution diagrams in formal verification, more specifically in decision diagram based symbolic model check-

ing algorithms, as these could benefit from the flexibility offered by SMT formulas.

VI. Acknowledgements

We would like to thank Nóra Almási for her contributions to the development of substitution diagrams and Dániel Szekeres for the help he provided with the practical evaluation of our approach.

References

[1] Clark Barrett and Cesare Tinelli. *Satisfiability Modulo Theories*, pages 305–343. Springer International Publishing, Cham, 2018.

[2] Aaron R Bradley and Zohar Manna. *The calculus of computation: decision procedures with applications to verification*. Springer Science & Business Media, 2007.

[3] Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, 1986.

[4] Leonardo de Moura and Nikolaj Bjørner. Z3: an efficient smt solver. In *2008 Tools and Algorithms for Construction and Analysis of Systems*, pages 337–340. Springer, Berlin, Heidelberg, March 2008.

[5] Haniel Barbosa et al. cvc5: A versatile and industrial-strength SMT solver. In Dana Fisman and Grigore Rosu, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 28th International Conference, TACAS 2022*, volume 13243 of *Lecture Notes in Computer Science*, pages 415–442. Springer, 2022.

[6] Ákos Hajdu, Tamás Tóth, András Vörös, and István Majzik. A configurable CEGAR framework with interpolation-based refinements. In Elvira Albert and Ivan Lanese, editors, *Formal Techniques for Distributed Objects, Components and Systems*, volume 9688 of *Lecture Notes in Computer Science*, pages 158–174. Springer, 2016.

[7] D.M. Miller and R. Drechsler. Implementing a multiple-valued decision diagram package. In *Proceedings. 1998 28th IEEE International Symposium on Multiple- Valued Logic (Cat. No.98CB36138)*, pages 52–57, 1998.

[8] Vince Molnár. *Extensions and Generalization of the Saturation Algorithm in Model Checking*. PhD thesis, Budapest University of Technology and Economics, February 2020.

Dominant failure analysis using importance measures in an automotive case-study

Simon József Nagy
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
simon.jozsef.nagy@edu.bme.hu

Anfrás Vörös
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
vori@mit.bme.hu

***Abstract*—The dependable operation of electronic components and subsystems is a primary concern in the automotive industry since erroneous behavior may cause serious harm to passengers. As a result, customers, regulators, and industry-specific standards define numerous extra-functional, availability, reliability, and safety requirements. Thus, electronic automotive systems may include advanced redundancy patterns, fault detection, and control mechanisms to provide fault-tolerant behavior. The complexity of automotive solutions may impose a challenge for engineers since weaknesses and inefficiencies in the system design may remain hidden during development. In reliability engineering importance measures can facilitate the identification of dominant fault contributors. This paper uses a working example inspired by real-world industrial solutions to examine the practical applicability of importance measures in modern automotive systems.**

***Index Terms*—verification, reliability, analysis, SMC, automotive systems, error propagation, fault-tolerance, importance measures**

I. Introduction

In automotive systems, safety is a primary concern [1] since the malfunction of automotive systems can cause serious harm to people. As a result, customers and industrial standards require engineers to reduce the risk to an acceptable level by implementing safety measures and executing safety analyses. Safety measures of modern automotive systems might include robust architecture with redundant components, smart electronic components, sensor fusion, and adaptive reconfiguration algorithms to ensure safe and fault-tolerant operation [2]. Safety measures are applied at both hardware and software levels, and some safety mechanics utilize multiple hardware and software components. The increasing complexity of safety measures might impose a challenge for engineers since dominant failure contributors might remain hidden in the case of advanced fault-tolerance mechanisms. Traditional safety analysis techniques, such as fault-trees and FMEDA-s, cannot evaluate reconfiguration and sensor fusion algorithms efficiently [3].On the other hand, simulation-based approaches can evaluate complex fault-tolerance mechanisms, yet simulation-based approaches may hide the internal failure propagation of the system from the user. Importance measures are widely used in safety-critical applications to estimate the contribution of component faults to system-level failures. Importance measures may indicate how substantial the causation is between component- and system-level malfunctions. Moreover, some important measures can indicate how a combination of component-level faults can contribute to system-level failures. Importance measures can be calculated using both traditional and simulation-based safety analysis methods.

In this paper, we present the application of importance measures using an automotive case study inspired by the technical solutions of thyssenkrupp Components Technology Hungary Ltd. Because of the high complexity of modern automotive systems, we estimate the importance measures using simulation-based dependability analysis techniques, presented in [2]. Additionally, we extend an existing importance measure to analyze how a given sequence of component faults can contribute to system failures. Finally, we demonstrate how importance measures may help engineers understand the failure propagation within advanced fault-tolerant and help engineers find hidden weaknesses in the system architecture.

II. Background

A. Importance Measures

In reliability engineering, importance measures are widely used to evaluate the cause-causation relation between component- and system-level failures [4]. Such a measure is the "*Bayesian Reliability Importance*", which can be defined using the following formula:

$$I_{Bayes}^{fault_i} = P\left(fault_i | system\_failure\right)$$

The analysis of *Bayesian Reliability Importance* estimates the posterior probability of the component-level failure modes assuming a given system-level failure mode occurred.

B. Probabilistic programming

The probabilistic programming paradigm [5] has been developed to model and analyze complex stochastic behavior reusing programming language elements, such as while and if statements and arithmetic expressions. Probabilistic programming has been applied for statistical modeling of natural environmental systems [6], yet to the best of our knowledge, probabilistic programming has been applied for empirical data analysis and has not been used for dependability modeling of CPSs.

This paradigm uses a new type of stochastic modeling formalism, namely, the probabilistic programming language

DOI https://doi.org/10.3311/MINISY2024-013

(PPL). PPLs support arbitrary stochastic distribution, even multinomial and time-dependent ones. Probabilistic programs include dedicated modeling elements called *primitives* such as *sample* and *observe* (*observe* is also known as *condition*) statements to define the prior and the posterior models, respectively. The *sample* statement defines that a sample has been drawn from a stochastic distribution. The *observe* statement specifies that we assume that a random sample equals a given constant value.

There are numerous PPLs have been developed, such as Pyro [7], PRISM [8] and Anglican [9], which extend the syntax of an existing programming language. For instance, Pyro extends Python, PRISM extends Prolog, and Anglican extends Clojure. Numerous analysis algorithms have been implemented for PPL. Pyro and Anglican can be evaluated by only MCMC methods. In contrast, PRISM can be evaluated by analytical solvers.

C. Stochastic Gamma Composition Language

Gamma Composition Language (GCL) uses engineering modeling concepts such as hierarchical (de)composition, and mixed semantics to facilitate the modeling of complex systems, yet GCL can model only functional behavior [10]. SGCL is based on GCL and extends mixed semantics modeling with stochastic components for dependability and performability modeling [11]. As a result, engineers can develop the extra-functional model of the embedded systems by extending the functional GCL behavior with extra-functional, stochastic modeling elements.Using stochastic components, engineers can directly model unpredictable phenomena within the system or in its environment.

Using high-level, engineering modeling concepts is beneficial for dependability and performability modeling, since this way, the structure of the dependability models can be similar to the system and software models. Moreover, software and system models can be directly reused without modification during dependability and performability modeling. This structural similarity increases the tractability and interoperability between the system and dependability models. SGCL supports the use of the high-level modeling elements of Gamma, such as statecharts, composite components, ports, and event-based interfaces for dependability and performability modeling. Using statecharts, one can concisely model complex software components, such as safety mechanisms, load-balance algorithms, and sensor diagnostics.

III. Working example

The working example is a simplified failure propagation model of an electronic power assist system (EPAS), which facilitates the steering in automobiles and trucks [2]. The working example is inspired by real-world steering solutions of thyssenkrupp Components Technology Hungary Ltd. As shown in Figure 1, the system contains two redundant microcontrollers and six redundant sensors.

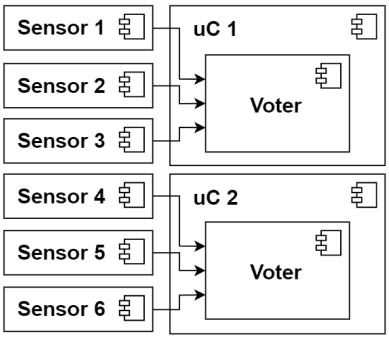


Fig. 1. Structure of the working example

A. System-level failure modes

The system has two system-level failure modes:

- ***lost steering***: When the power assistance is gone, the driver has to steer the car without any support.
- ***uncontrolled steering***: When the assist torque is completely erroneous and might cause an accident.

The two failure modes have significantly different safety criticality. As a result, we cannot analyze only the failure of the system; we have to analyze the two failure modes separately. Moreover, it is hard to give a precise upper estimate for the overall operation time of road vehicles. As a result, the probability of the two failure modes is essential for the safety of road vehicles.

B. Failure model of the sensors

The sensors provide essential measurements for steering assistance, and at least one available sensor is required for the correct operation of the EPAS. As shown in Figure 2, we modeled the error behavior of the sensors using a statechart. The sensors have the following component-level failure modes:

- ***invalid faulty***: The sensor provides incorrect and invalid data output. This failure mode can be detected easily using a simple data validation.
- ***valid faulty***: The sensor provides incorrect data output, but the validity check cannot detect this kind of failure mode. This type of fault also can be called *latent* fault since this failure mode might remain unnoticed by the diagnostic mechanisms.

C. Failure model of the microcontroller

The microcontroller processes the sensor data and calculates controls for the power assist. As shown in Figure 3, we modeled the error behavior of the microcontroller using a statechart. The microcontroller has only one failure mode, namely ***stop faulty*** (uC): The microcontroller suddenly stops and cannot execute software further.

D. Voter mechanisms in the microcontroller

In the microcontroller, a voting algorithm may detect invalid measurements if more correct sensors are connected to the microcontroller than valid faulty sensors. As shown in Figure 4,

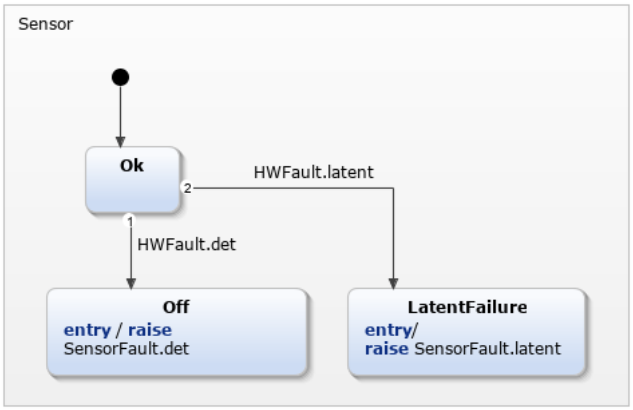


Fig. 2. Statechart failure propagation model of the sensors

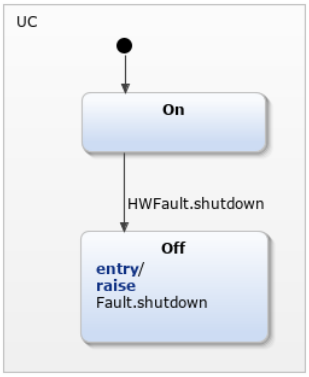


Fig. 3. Statechart failure propagation model of the microcontrollers

we modeled the error propagation within the voting algorithms using a statechart. If in a microcontroller there is more or an equal number of valid faulty sensors than correct sensors, then the state of the system becomes "*uncontrolled steering*". If there is no operational sensor connected to the operational microcontrollers, then the state of the system becomes "*lost steering*".

E. Parameters of the system

Parameters of the EPAS system Parameters are the failure rates of the failure modes of the system:

$$\Phi = (\lambda_{\mu C}, \lambda_{sensor\_det}, \lambda_{sensor\_latent})$$

The failure rates are calculated using the hardware FMEDA analysis. The default values of the parameters are the following, where FIT is the unit of failure rates and $FIT = \frac{1}{10^9 h}$:

$$\Phi = (100.0\ FIT, 10.0\ FIT, 10.0\ FIT)$$

IV. Overview of the Approach

Our approach can identify the dominant failure contributors using system-level dependability simulation and importance

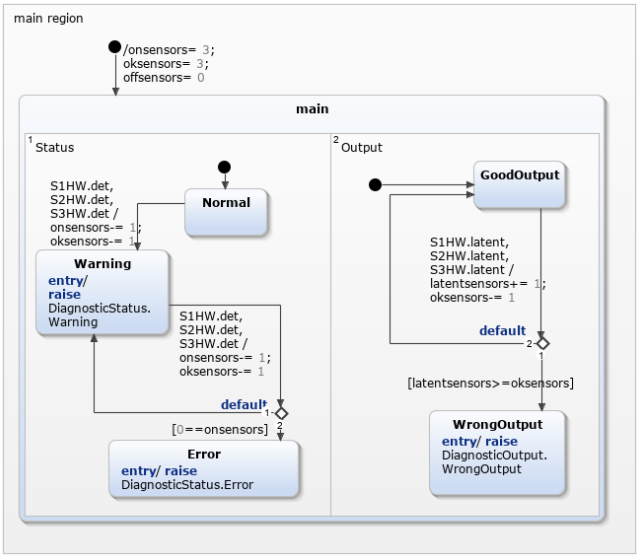


Fig. 4. Statechart failure propagation model of the voter mechanisms

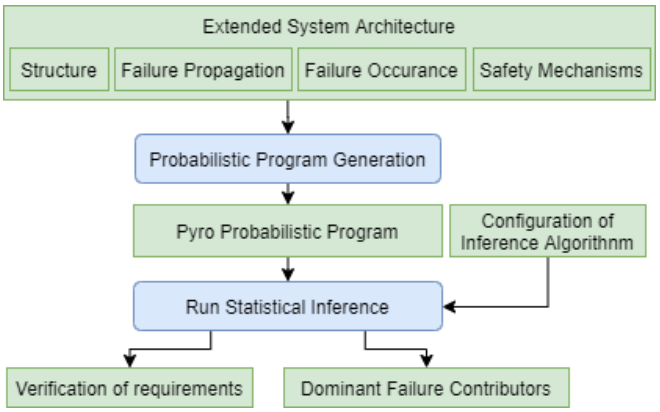


Fig. 5. Overview of the approach

measures. Our approach requires the system architecture models with additional stochastic component failure model extensions as inputs. The extended system architecture specifies the failure modes of the components and the failure propagation between the components. The extended system architecture can specify the behavior of fault-tolerance mechanisms. The system architecture and the dependability extensions are defined using Stochastic Gamma Composition Language, which uses statecharts to define behavior models and component composition models using event-based interfaces to define the structure of the system. The proposed dependability analysis approach first generates a Pyro deep probabilistic program from the system architecture using the Stochastic Gamma Composition Framework. The,n we use probabilistic inference to verify dependability requirements and identify dominant failure contributors.

A. Supported failure propagation behavior

Our approach supports a wide area of deterministic system behavior. Our analysis approach uses a black-box deterministic error propagation simulation. The stochastic event generator injects faults into the model, and the inference algorithm observes the high-level system failures. One can define the error propagation using a statechart. We support all typical modeling elements of statecharts, such as pseudo-states, timed transition, orthogonal states, transition effects, guards, arithmetic expressions, and entry/exit actions. The effects and entry/exit actions can be defined using pseudo code. Our approach does not restrict the deterministic modeling. In the Gamma Statecharts, one can use arbitrary deterministic code in translation effects if the runtime of the code is finite.

B. Supported stochastic behavior

The simulation might be faster in some cases if all distribution is exponential (see Markovian simulation), yet any simulate-able distribution can be used:

- the variance and expected value of the distribution shall exist, and
- the *pdf* and inverse *cdf* shall be calculate-able in finite time.

In all of our simulations, we used fixed distributions or fixed distributions, which have parameters. Theoretically, our approach can support distributions, which depend on the state of the system.

C. Identifying dominant failure contributors

We model the component faults as external interventions from the viewpoint of the system, and our purpose is to calculate the system-level effect of the component faults. Our objective is to analyze the causation besides the correlation between component faults and system failures. We analyze the $P(fault_i|system\_failure)$ distribution instead of $P(system\_failure|fault_i)$, since it can be calculated faster, and is widely used in the literature for causality analysis. We run the inference algorithm $N_{system\_failure}$ times, where $N_{system\_failure}$ is the number of system-level failure modes. Calculating $P(system\_failure|fault_i)$ directly may require running the inference algorithm N_{fault} times, where N_{fault} is the number of component-level failure modes and in most systems $N_{fault} << N_{system\_failure}$. Moreover $P(system\_failure|fault_i)$ can be calculated from $P(fault_i|system\_failure)$ using the Bayes theorem:

$$P(system\_failure|fault_i) = \dots$$

$$P(fault_i|system\_failure) \cdot \frac{P(system\_failure)}{P(fault_i)}$$

where $P(fault_i)$ can be calculated analytically and $P(system\_failure)$ can be estimated accurately by running $N_{system\_failure}$ number of inference.

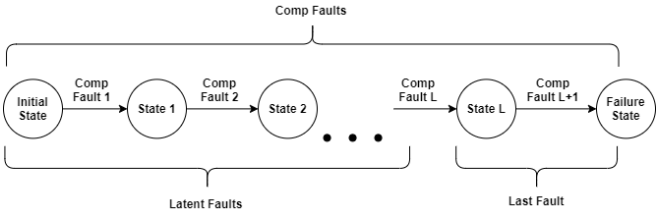


Fig. 6. Categorization of component faults

V. Failure contributions of component faults

A. Categorisation of fault occurrences

As it can be seen in Figure 6, we distinguish three different categories of component faults:

- ***Component fault***: the time of the component fault is not greater than the time of the system failure
- ***Latent fault***: the time of the component fault is smaller than the time of the system failure
- ***Latent fault***: the time of the component fault is equal to the time of the system failure

B. Bayesian Importance Measure of Component Faults

We generated probabilistic programs from the extended system architecture and used the inference algorithms to calculate the posterior probabilities of the component faults, assuming a given system-level failure mode occurred. The comparison of the Bayesian importance measures of the component-level faults is denoted in Figure 7. We denoted the importance measures of component faults belonging to the lost steering and uncontrolled steering system-level failure modes with red and blue colors, respectively. One can see that the importance measures of sensor faults, especially the latent sensor faults, are significantly larger for uncontrolled steering than lost steering. On the other hand, the importance measures of the microcontroller faults are higher for lost steering than uncontrolled steering. The results are in accordance with our preliminary expectations since the sensor faults may cause erroneous operation within the voter, which is the main cause of uncontrolled steering. Moreover, the microcontroller faults hide all the sensor faults if the faulty sensors are connected to the microcontroller.

C. Bayesian Importance Measure of Latent Faults

The comparison of the Bayesian importance measures of the latent component-level faults is denoted in Figure 8. We denoted the importance measures of component faults belonging to the lost and uncontrolled steering system-level failure modes with red and blue colors, respectively. One can see that the results of the latent faults are similar to the component faults. However, the difference between the lost and uncontrolled steering failure modes is significantly weaker.

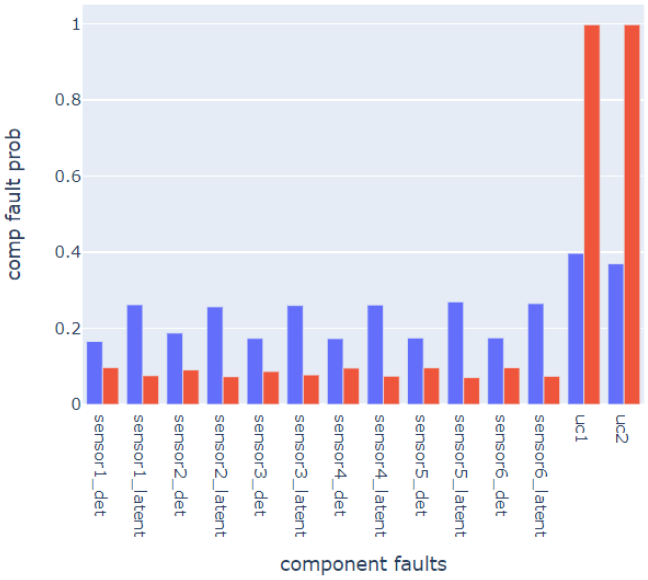


Fig. 7. Bayesian Importance Measure of Component Faults

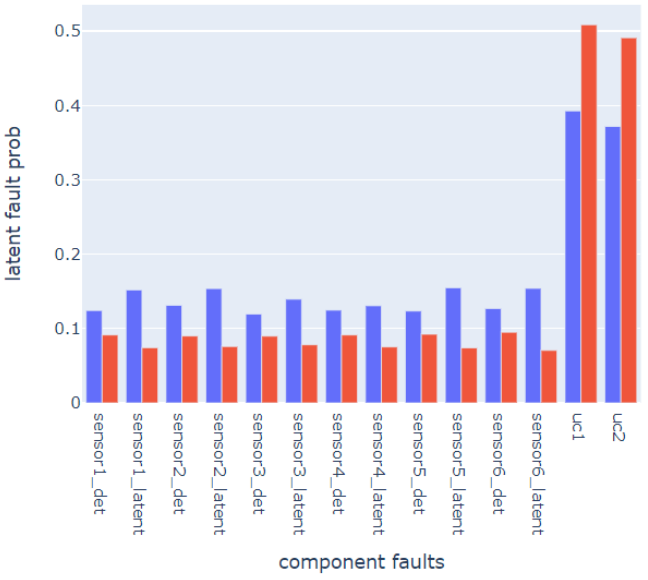


Fig. 8. Bayesian Importance Measure of Latent Faults

D. Bayesian Importance Measure of Last Faults

The comparison of the Bayesian importance measures of the latent component-level faults is denoted in Figure 8. We denoted the importance measures of component faults belonging to the lost and uncontrolled steering system-level failure modes with red and blue colors, respectively. One can see that the results of the latent faults are similar to the component faults. However the difference between the lost and uncontrolled steering failure mode is significantly stronger since the last faults are directly related to the system-level failure. Moreover, we can see that the importance measures of

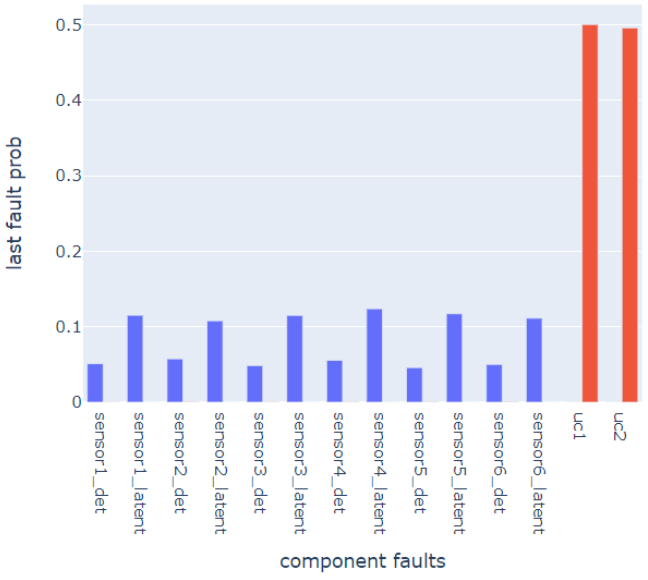


Fig. 9. Bayesian Importance Measure of Last Faults

the component faults are the superposition of the importance measure of the last and latent faults.

VI. Failure Contributions of Component Fault Pairs

The Bayesian Reliability Importance cannot analyze the interaction of component faults. Therefore we extended the Bayesian Reliability Importance for dual points of failure with respect to the order of the faults. We defined a virtual component fault $fault_1$ **After** $fault_2$, which occurs if two component faults occur in a given order. For this virtual fault, we can calculate the importance measure of $fault_1$ **After** $fault_2$ using the following formula:

$$P(fault_1 \textbf{ After } fault_2 \mid lost\_steering)$$

We note that our approach can be used for arbitrary temporal logic operations between the component faults and system states in addition to $fault_1$ **After** $fault_2$. The extension of importance measures for temporal logic expressions is future work.

A. Joint Bayesian Reliability Importance Measure for Lost Steering

In Figure 10, we depicted the *Joint Bayesian Reliability Importance Measures* of the two long component fault sequences of *Lost Steering* using a heat map. The brightness of each cell in the heat map is proportional to the importance of a sequence, where the first and second faults are specified by the row and column of the cell, respectively. We can see that the joint importance measures of the microcontroller faults are high, as expected. Moreover, the row of the microcontroller faults is darker than expected, since if sensor failures occur before the microcontroller fault, then the probability that a second sensor fault causes uncontrolled steering is higher.

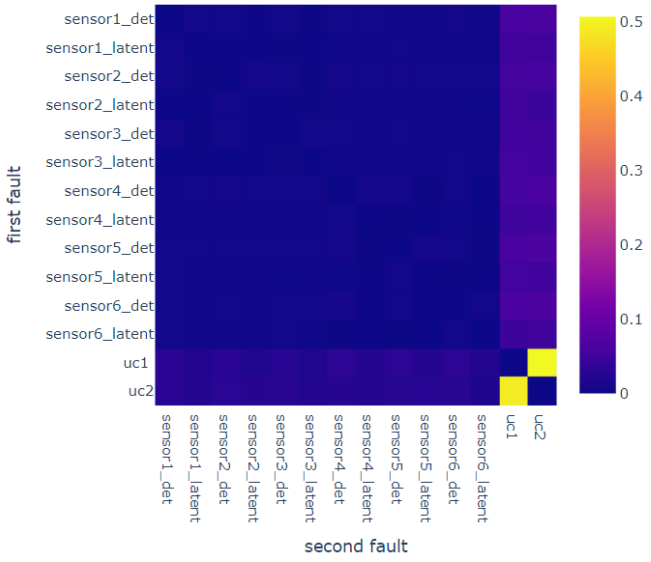


Fig. 10. Enter Caption

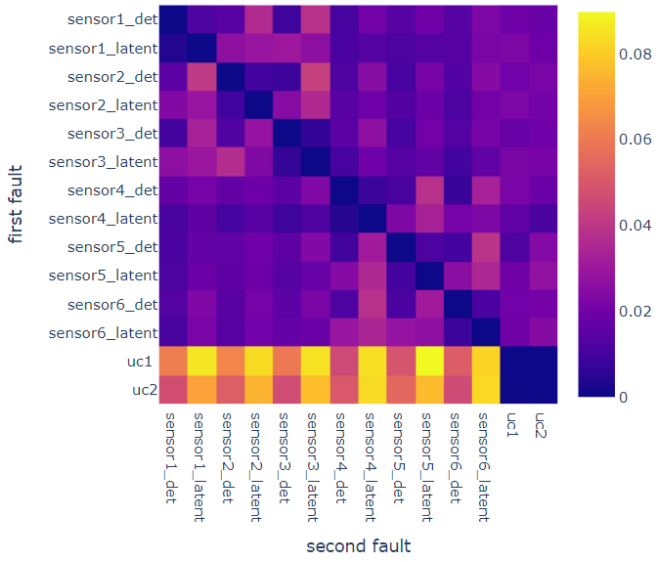


Fig. 11. Enter Caption

B. Joint Bayesian Reliability Importance Measure for Uncontrolled Steering

In Figure 11, we depicted the *Joint Bayesian Reliability Importance Measures* of the two long component fault sequences of *Lost Steering* using a heat-map. We can see that the column of the microcontroller faults (when the microcontroller fault is the second fault), since the microcontroller faults not only can hide the effect of sensor faults. Moreover, there are two brighter clusters in the rows and columns of the sensors since the sensors of the two microcontrollers are separated, and their faults cannot interact.

VII. Conclusion

The increasing complexity of safety mechanisms imposes a challenge for engineers in the automotive industry, since systematic weaknesses in the system design and dominant failure contributors may remain hidden during the design phase of the development. Importance measures are widely used in reliability engineering to evaluate the causation between component and system-level malfunctions. In this paper, we presented the applicability of state-of-the-art Bayesian reliability importance measures in an automotive case study. We extended the Bayesian importance measure for fault sequences and used the results to identify dominant contributors of the system-level failure modes.

References

[1] R. Sobti and P. Kaur, "Model-based architecture of software-intensive intelligent automotive systems," in *ICCS 2018*. IEEE, 2018, pp. 132–136.

[2] S. J. Nagy, B. Graics, K. Marussy, and A. Vörös, "Simulation-based safety assessment of high-level reliability models," *arXiv preprint arXiv:2004.13290*, 2020.

[3] S. Sharvia, S. Kabir, M. Walker, and Y. Papadopoulos, "Model-based dependability analysis: state-of-the-art, challenges, and future outlook," *Software Quality Assurance*, pp. 251–278, 2016.

[4] K. P. Amrutkar and K. K. Kamalja, "An overview of various importance measures of reliability system," *International Journal of Mathematical, Engineering and Management Sciences*, vol. 2, no. 3, pp. 150–171, 2017.

[5] A. D. Gordon, T. A. Henzinger, A. V. Nori, and S. K. Rajamani, "Probabilistic programming," in *Future of Software Engineering Proceedings*, 2014, pp. 167–181.

[6] C. Krapu and M. Borsuk, "Probabilistic programming: a review for environmental modellers," *Environmental Modelling & Software*, vol. 114, pp. 40–48, 2019.

[7] E. Bingham, J. P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P. Szerlip, P. Horsfall, and N. D. Goodman, "Pyro: Deep universal probabilistic programming," *The Journal of Machine Learning Research*, vol. 20, no. 1, pp. 973–978, 2019.

[8] M. Kwiatkowska, G. Norman, and D. Parker, "Prism: Probabilistic symbolic model checker," in *International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*. Springer, 2002, pp. 200–204.

[9] D. Tolpin, J.-W. v. d. Meent, and F. Wood, "Probabilistic programming in anglican," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2015, pp. 308–311.

[10] 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.

[11] S. J. Nagy, "Component-based stochastic modeling and analysis," Master's thesis, Budapest University of Technology and Economics, 2022.

Towards the Requirement-Driven Generation and Evaluation of Hyperledger Fabric Network Designs

Noor Al-Gburi and Imre Kocsis
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
Email: noor.algburi@edu.bme.hu, kocsis.imre@vik.bme.hu

***Abstract*—Hyperledger Fabric (HLF) is an adaptable blockchain platform that enables the requirement-driven construction of cross-organizational distributed ledger networks. While this flexibility offers many advantages, it also introduces challenges in ensuring Fabric networks' fit-for-purpose nature and extra-functional requirement adherence, e.g., with respect to fault tolerance. This challenge calls for the use of mathematically precise formal analysis techniques, tooling and models. In this paper, we propose a combination of model-driven engineering principles and diverse graph generation to validate Fabric network designs and to facilitate their design for dependability. Specifically, we present a Fabric network meta-model and a set of well-formedness requirements in the Graph-solver-as-a-Service tool Refinery. Uniquely, Refinery can check conformance on partial models, enabling analysis already during early design.**

***Index Terms*—blockchain, DLT, Hyperledger Fabric, Refinery, graph queries, architecture Analysis**

I. Introduction

Hyperledger Fabric (HLF) [1] is a versatile blockchain platform widely adopted by enterprises for building secure and scalable distributed applications. Its adaptability allows the creation of diverse network architectures, offering flexibility in meeting specific requirements and adhering to extra-functional criteria. However, this flexibility simultaneously poses challenges in ensuring that Fabric networks align with desired parameters and meet the required functional and non-functional requirements.

Adapting HLF networks to the specific requirements of cross-organizational cooperation introduces complexities in assessing the alignment of the initial architecture with expectations. In this paper, we propose using HLF network meta-modeling and subsequent diverse graph generation to facilitate a) design space exploration [2], b) hidden formal analysis of architecture options [3], and c) choosing an architecture in a requirement-driven manner. As an initial contribution, we describe the HLF meta-model in diverse graph generation, utilizing the Graph-Solver-as-a-Service tool *Refinery* [4]. We demonstrate how simple dependability requirements can be already enforced in graph generation [5], adhering to resource constraints without involving external tools. Additionally, we outline a workflow model to perform dependability analysis in external tooling.

II. Overview of Hyperledger Fabric Network Architecture

Hyperledger Fabric is well-suited for developing blockchain-distributed ledgers for cross-organizational use cases. The framework allows adaptation of its architecture to meet specific trust models, diverse use case scenarios, and performance requirements. It has been shown in the practice and literature that Fabric networks can be created to meet specific performance and capacity criteria [6]. Fabric supports various ordering services, including a Byzantine fault tolerant (BFT) ordering service [7].

Moreover, Hyperledger Fabric is a modular blockchain framework that provides a flexible and scalable architecture for developing distributed ledger applications. The architecture supports standard programming languages and industry-standard identity management [1]. The design flexibility of HLF enables tailoring blockchain networks to specific use cases and trust models.

HLF network architecture comprises various components, including network nodes known as peers. Peers are categorized as endorsing (simulating and endorsing transactions) and committing (validating and updating the ledger). The network is organized into entities called organizations, each owning one or more peers and governed by its own Certificate Authority (CA) for participant authentication. Channels enable private communication and transactions among specific participants, ensuring confidentiality. Consortiums are critical in connecting organizations, establishing network rules, and defining parameters. An ordering service orchestrates transaction sequencing and maintains consistency across peers. Smart contracts, known in Fabric as *chaincode*, encapsulate business logic, executed during transaction simulation, while the ledger serves as the repository for transaction history.

III. Proposed Approach

The increasing adoption of Hyperledger Fabric within enterprises to develop secure and scalable distributed applications reflects a contemporary trend. At the same time, designing and analyzing Hyperledger Fabric networks that meet the specific requirements of a given application can be challenging —- for instance, requirements on fault tolerance.

To address these challenges, we propose in Figure 1 a workflow for specifying network architecture, conducting a formal

DOI https://doi.org/10.3311/MINISY2024-014

analysis to validate these architectures against dependability requirements, and using the Refinery tool to facilitate the translation between generated models and formal analysis. This workflow is intended to support an Answer Set Programming (ASP)-based Error Propagation Analysis (EPA) approach for Hyperledger Fabric in the future.

- **Partial model and extra-functional requirements identification:** Starting with the definition of the partial model and extra-functional requirements of an HLF network (e.g., specifying which organization should participate in which channel), this first phase sets up a framework that guides the next steps and provides the information needed for the Refinery tool. The workflow input includes requirements and a partial model, which already reflects some design choices made by the engineer.
- **Formal Specification for Architecture Requirements:** Based on the requirements, we construct a formal specification of the architecture candidates. For this task, we are using the partial modeling language [8] of the Refinery framework [4]. This consists of a meta-model describing the main concepts and relations of HLF networks, as well as constraints about the networks.
- **Instantiate:** for this step we are using the graph solver algorithm [5] in the Refinery framework to automatically generate valid architecture candidates.
- **Architecture options:** After graph generation, we interpret the generated graphs as architecture candidates, forming the basis for subsequent analysis. Since Refinery provides a consistent graph generator, all generated candidates will satisfy the constraints provided in the previous step.
- **Architecture Analysis:** Following the generation of architectural options, we use Refinery to complement the architectural model with model elements, facilitating the translation of the model to the input languages of analysis tools. Subsequently, analysis tools will be employed to test and analyze the system to ensure alignment with specified requirements and adherence to quality constraints. In instances of identified errors, the workflow accommodates iterative refinement, allowing a return to the initial stages to modify the partial model or requirements.
- **Rank-Ordered candidates:** At the end of the process, we will obtain a set of rank-ordered architectural candidate options. Through analysis, we can identify requirement-compliant architectural candidates, and these candidates undergo evaluation and ranking based on criteria such as scalability and performance. This systematic approach facilitates the identification and prioritization of configurations that best align with the requirements of the HLF network.

IV. Graph-Based Modeling

Our research builds on previous work of researchers recognizing the need for specialized tools to manage and deploy complex network structures in Hyperledger Fabric as

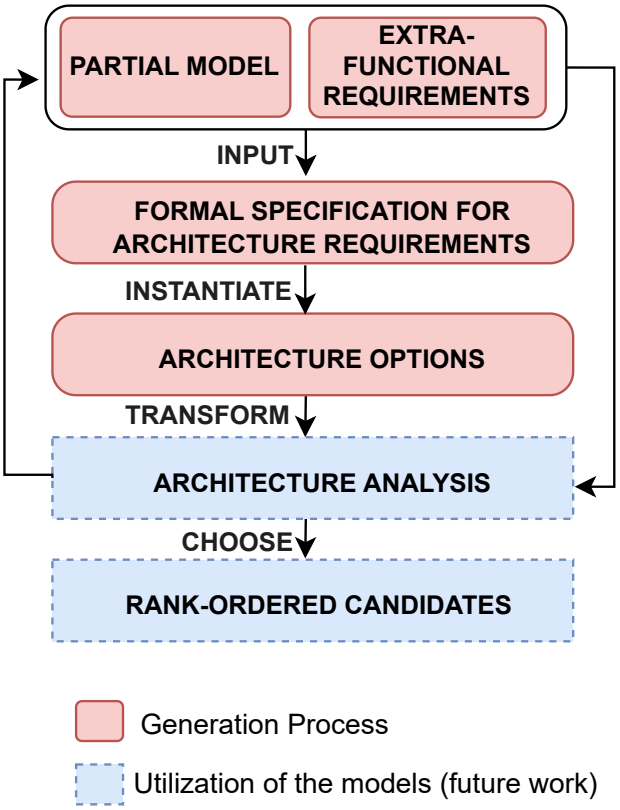


Fig. 1: Workflow of the proposed approach

highlighted by [9]. Fowler [10] proposed domain-specific languages (DSLs) to provide a more intuitive description of network structures. Nguyen [11] implemented DSLs on a general-purpose infrastructure to enhance performance and flexibility. In recent years, researchers have focused on enhancing the scalability and efficiency of graph solver algorithms. Zhang [12] introduces GraphRex, optimizing declarative graph queries.

Several works are trying to improve Refinery, Ahmad [13] enhances graph solver scalability. Semerath [14] proposes a method for detecting issues in DSL specifications using EMF-IncQuery framework meta-models and first-order logic analyzed by a Satisfiability Modulo Theories solver (SMT-solver). Grüner [15] demonstrated the practical applicability of rule-based systems using declarative graph database queries in engineering use cases. Semerath [5] presents a graph solver framework for automatically generating consistent domain-specific instance models. Our approach facilitates the introduction of an analysis framework to address the challenges of designing and validating Hyperledger Fabric networks while leveraging the strengths of previous research in this area.

Graph-based models are crucial in software engineering, representing compositions used in test environments and system designs for model-based engineering. Our research applies the direct and automated generation of consistent graph models to support design space exploration. We use Refinery to gener-

ate well-formed, consistent, varied, and realistic Hyperledger Fabric network models. The concept of Refinery as a tool for producing consistent models has been thoroughly examined in the literature.

Our approach uses Refinery's specification language based on partial models, first presented by Marussy [8] and further expanded by [16] to incorporate multiplicity reasoning. Consistently generated graph models are created using these languages. Semerath [17] and Varro [18] share a common interest in automating the generation of realistic and diversified graph models, with Semerath [17] focusing on the requirement for structurally realistic models.

Refinery provides a simple yet powerful specification language for generating graphs. We use this language to define Hyperledger Fabric network models inside Refinery to accurately reflect the component types and connections in the network architecture. The language was designed to be concise and intuitive, enabling designers to express complex network configurations without delving into complex formal notation.

V. Modeling Hyperledger Fabric Networks

The meta-modeling language of Refinery offers full support for a concise and expressive representation for describing a wide range of network components, including peers, organizations, channels, and more. The meta-modeling language facilitates the creation of diverse Hyperledger Fabric networks, allowing for the easy construction of complex network architectures through linking elements. We have created a meta-model in Refinery for HLF network architecture. The example model presented here, and other artifacts are available at the GitHub repository: https://github.com/noorsabr/Refinery-HLF. To demonstrate the modeling style of Refinery in the code example below, we present a snippet from our meta-model, introducing classes such as EndorsingNode, OrderingNode, and OrdererOrganization.

```
class Host{
    contains Node[1..*] deployedComponents
}

abstract class Node { }
class EndorsingNode extends Node { }
class OrderingNode extends Node { }

abstract class AbstractOrganization { }
class ParticipantOrganization extends
    AbstractOrganization {
    contains Host[1..*] hosts
}
class OrdererOrganization extends
    AbstractOrganization {
    contains OrderingNode[1..*] orderingNodes
}
```

Listing 1: Snippet from the meta-model

A. Modeling Approach

In this research, we employed a systematic approach to design HLF networks, modeling communication dependencies, organizational structures, and hierarchies. The design process involves identifying specification statement requirements using natural language. Subsequently, these specifications are transposed into a meta-model within Refinery tool, effectively capturing the fundamental concepts and relationships inherent in Hyperledger Fabric networks. This modeling approach delivers representation by bridging the gap between the complexities of controlled natural language and the descriptive model. The intermediate stage of using a clear, controlled natural language, such as intermediary specification, helps to understand concepts and relationships within the Fabric network before formally defining them in the meta-model. The complete textual intermediary meta-model specification is in the GitHub repository.

Our modeling approach is tailored to represent operational Hyperledger Fabric networks [19]. We prioritize capturing the current state and ongoing operations of deployed networks rather than focusing on transient configurations or the initial setup phase. We currently simplify our model by not allowing for peerless organizations, which may be lifted in future iterations.

Examples of statements utilized in the meta-model include:

- "A Fabric network consists of organizations and channels," defining the fundamental building blocks of a Fabric network: organizations and channels.
- "A peer is either an endorsing peer or an ordering peer," identifying two types of peers in a Fabric network: endorsing peers and ordering peers.
- "An organization owns at least one peer," ensuring that organizations have at least one peer to participate in the network.
- "For Kafka-based ordering, there's exactly one orderer organization," specifying that there should be only one orderer organization per Kafka-based network.
- "For Raft-based ordering, there's no orderer organization," differentiating between Kafka-based and Raft-based ordering services. Raft-based ordering uses a different consensus mechanism that does not require dedicated orderer organizations.

B. Meta-model Description

Nodes, hosts, organizations, channels, and ledgers are the fundamental entities defined by the meta-model, each with specific associations and relationships. The meta-model captures the organizational hierarchy and distinguishes between orderer and participant organizations. Two general network types, RaftFabricNetwork and KafkaFabricNetwork, are modeled along with different channel types, such as SystemChannel and ApplicationChannel. Optionally, specifications can include attributes and errors/predicates to capture communication relationships, making identifying configuration errors or violations possible. Additionally, we can formulate complex logic predicates and constraints to control the range of valid graphs precisely. The predicate language of Refinery allows us to develop more complex predicates using node equivalence, negation, conjunction and disjunction, transitive closure, or

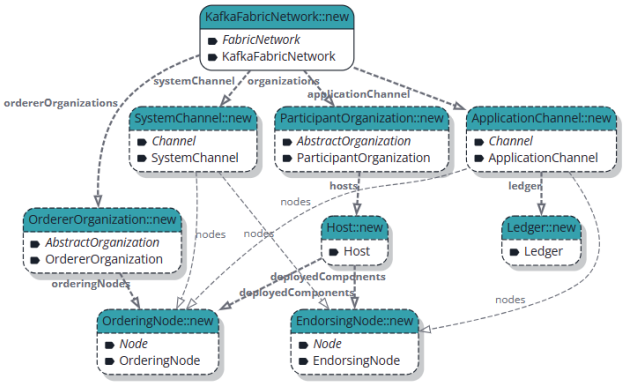


Fig. 2: HLF network partial model (without accompanying predicates)

reference to another predicate. Figure 2 shows the initial model of HLF based on the meta-model description.

The description of the meta-model for Hyperledger Fabric network from our approach is outlined as follows:

1) Key Classes:

- **FabricNetwork**: HLF network that contains different types of organizations (Participant and Orderer)
  - **ParticipantOrganizations**: actively participate by running the endorsing nodes and committing peers. They contribute to transaction processing and consensus.
  - **OrdererOrganizations**: ensuring transaction order and contributing to consensus in some capacity. While focusing on ordering
- **DeployedComponents**: software applications operating on HLF network hosts. Endorsing and ordering nodes are typical types of deployed components.
- **Host**: physical machines that host software components (DeployedComponent). Connected to DeployedComponent to illustrate where software components operate.
- **Channel**: a virtualized environment facilitating secure communication and transaction processing between participating organizations. This class is further specialized into SystemChannel and ApplicationChannel.
  - **SystemChannel**: a type of Channel specifically designed for system-level operations and communication. It contains one or more ApplicationChannels.
  - **ApplicationChannel**: a type of Channel tailored for application-level transactions and interactions. It contains exactly one Ledger.
- **EndorsingNode**: a node responsible for validating transactions before submitting them to the ordering service.
- **OrderingNode**: a node responsible for ordering and persisting transactions to the ledger.
- **Ledger**: the distributed database that stores the history of transactions within a channel. Connected to Node to depict its relationship within the channel.
- **KafkaFabricNetwork** an HLF network that utilizes the Kafka-based ordering service.
- **RaftFabricNetwork** an HLF network that utilizes the Raft-based ordering service.

2) Relationships:

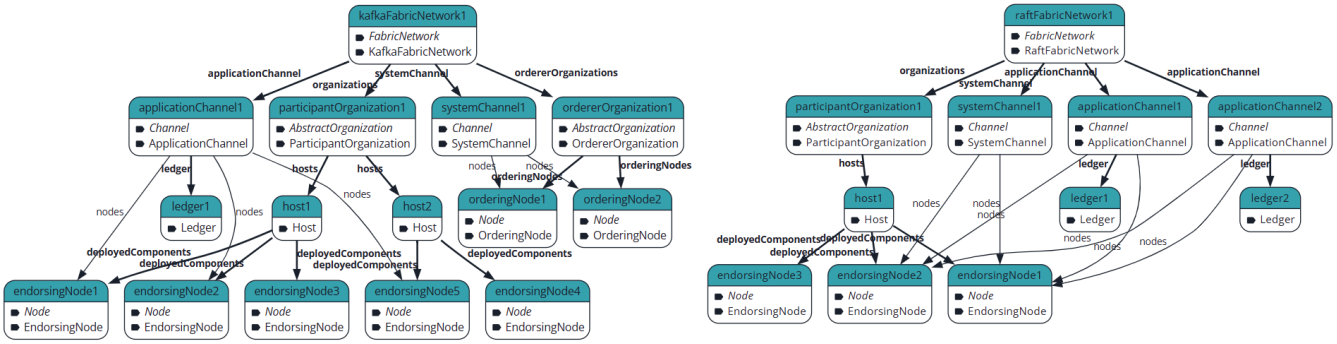
- **Organization contains Node**: Organizations play a crucial role in managing and owning nodes within the network. This relationship helps establish the hierarchical structure where organizations are responsible for specific nodes.
- **Organization contains Host**: Organizations are directly associated with the physical infrastructure (hosts) necessary to deploy and operate their network nodes.
- **Channel has Organization**: Channels serve as isolated communication environments, and this relationship defines the set of organizations involved in a particular channel.
- **Channel has Node**: This relationship establishes the connectivity between nodes and channels, illustrating which nodes are engaged in the transaction processes of a given channel.
- **Channel has Ledger**: Each channel has its ledger, and this relationship signifies the ledger's role in recording and maintaining the transaction history specific to that channel.

3) Predicate Definitions:

- **ledgerInNodes**: a Ledger is associated with a particular Node within a Channel.
- **organizationCommunicatingWithChannel**: OrdererOrganization should have an OrderingNode deployed in the channel. ParticipantOrganization should have a Host, and the Host should have a Node deployed in the channel `deployedComponents(h, n)`.

4) Error Definitions:

- **ordererInOtherChannel**: specifies an error condition where an OrdererOrganization is associated with nodes in an ApplicationChannel.
- **ordererNodeInNotKafkaNetwork**: identifies a specific condition where an OrderingNode, typically associated with a ParticipantOrganization in a context that is not a Kafka-based network.
- **participantInSystemChannelForKafka**: checks whether a ParticipantOrganization communicates with a SystemChannel within a KafkaFabricNetwork. `orgCommunicatingWithChannel(o, s)`: The organization should communicates with the specified SystemChannel.
- **participantInSystemChannelForRaft**: checks whether a ParticipantOrganization is not communicating with a SystemChannel within a RaftFabricNetwork. `!orgCommunicatingWithChannel(o, s)`: The organization should not communicates with the specified SystemChannel.
- **lonelyOrganization**: checks if an AbstractOrganization is not communicating with any channel.



(a) An example of a generated HLF network using Kafka-based ordering

(b) An example of a generated HLF network using Raft-based ordering

Fig. 3: Generated models examples for HLF network

C. Model Generation

Given a meta-model and possibly a partial model, Refinery will generate instances of the meta-model or complement partial models to full ones. The transition from the meta-model to concrete models is governed by specifications, ensuring the generated models adhere to predefined constraints and logic predicates. The models presented in Figure 3 were generated using specific settings within the Refinery tool. We utilized a *scope* definition, with parameters such as node count, organization count, and node types to constrain the size and characteristics of the generated models. For example, in the code below, we set the scope as "node = 20..25" and specify that each generated model should include a network with 20 to 25 nodes. These settings were chosen to reflect realistic scenarios and constraints within HLF networks.

```
scope node = 20..25,
    Node = 3 .. 10,
    AbstractOrganization = 2,
    FabricNetwork = 1.
```

Listing 2: Snippet shows a scoping example of the meta-model of HLF

These generated models showcase various models generated for Hyperledger Fabric networks. This model represents elements and connections within an HLF network, including various ordering services such as Raft-based and Kafka-based ordering. Notably, Kafka-based ordering requires a specialized orderer organization, distinct from Raft-based ordering. The generation time for the models varied based on the complexity of the specified settings. On average, the Refinery tool took approximately [0.02 ms] to produce a single instance of the HLF network meta-model. This duration may vary depending on the computational resources and the specific characteristics defined in the scope.

EXAMPLE: Regarding dependability requirements, Figure 4 is a generated example model that complies with the requirements. These requirements state that every Organization participating in a channel must have at least two nodes in each channel they belong to, and that no organization should have only one host in any channel in which it participates. These requirement specifications aim to enhance the overall dependability of the network by preventing a single organization from being the cause of a channel going down if one of its nodes fails. Using key classes such as Node, Host, Organization, and Channel, along with the `notEnoughPeersInChannel` predicate, our modeling in Refinery captures essential requirements for Hyperledger Fabric networks. The `notEnoughPeersInChannel` predicate indicates whether two distinct nodes (nodeA and nodeB) owned by the same ParticipantOrganization (org) are present in a given SystemChannel (ch). This predicate enforces that the organization deploys these nodes on different hosts (hostA and hostB) within the channel. The deployment specifics include nodeA on hostA and nodeB on hostB. Furthermore, the predicate ensures the distinctiveness of both nodes (nodeA != nodeB) and both hosts (hostA != hostB). Additionally, an error predicate, `notEnough(org, ch)`, is defined to check the negation of the `notEnoughPeersInChannel` condition. If `notEnoughPeersInChannel(org, ch)` is false, indicating that there are enough peers in the channel, then `notEnough(org, ch)` becomes true, signalling a violation of the dependability requirement. This comprehensive modeling approach in Refinery ensures the explicit representation

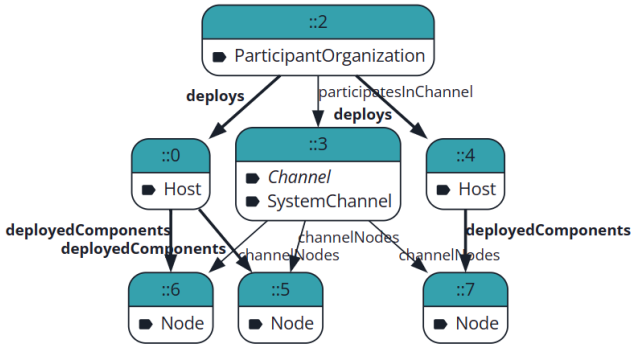


Fig. 4: Generated model of the example

of network structures and dependencies while facilitating the identification of potential issues related to the number of peers in a given channel.

VI. Summary

Our paper introduces an innovative approach to design and evaluate Hyperledger Fabric (HLF) network architectures, employing formal analysis and graph generation techniques. The method ensures clear network descriptions, well-formedness, and dependability. Leveraging Refinery, the proposed approach addresses challenges in creating fit-for-purpose HLF networks with diverse architectures. The approach specifies dependability requirements, generates architecture candidates, and enables the representation of complex network structures. The Refinery engine automates the creation of fully specified HLF network instances, offering a comprehensive framework for requirement-driven design and evaluation.

References

[1] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yellick, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, ser. EuroSys '18. New York, NY, USA: Association for Computing Machinery, 2018.

[2] M. Famelis, R. Salay, and M. Chechik, "Partial models: Towards modeling and reasoning with uncertainty," in *2012 34th International Conference on Software Engineering (ICSE)*. IEEE, 2012, pp. 573–583.

[3] A. Pataricza, I. Kocsis, F. Brancati, L. Vinerbi, and A. Bondavalli, "Lightweight formal analysis of requirements," in *Certifications of Critical Systems–The CECRIS Experience*. River Publishers, 2022, pp. 143–166.

[4] "graphs4value/refinery: Refinery: an efficient graph solver for generating well-formed models." [Online]. Available: https://github.com/graphs4value/refinery

[5] O. Semeráth, A. S. Nagy, and D. Varró, "A graph solver for the automated generation of consistent domain-specific models," in *Proceedings of the 40th International Conference on Software Engineering*, ser. ICSE '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 969–980.

[6] A. Baliga, N. Solanki, S. Verekar, A. Pednekar, P. Kamat, and S. Chatterjee, "Performance characterization of hyperledger fabric," in *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, 2018, pp. 65–74.

[7] A. Bessani, J. a. Sousa, and M. Vukolić, "A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform," in *Proceedings of the 1st Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers*, ser. SERIAL '17. New York, NY, USA: Association for Computing Machinery, 2017.

[8] K. Marussy, O. Semeráth, A. A. Babikian, and D. Varró, "A specification language for consistent model generation based on partial models," *J. Object Technol.*, vol. 19, pp. 3:1–22, 2020.

[9] I. Zikos, A. Sendros, G. Drosatos, and P. S. Efraimidis, "Hfabd+m: A web-based platform for automated hyperledger fabric deployment and management," in *2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain & Beyond (iGETblockchain)*, 2022, pp. 1–6.

[10] M. Fowler, "A pedagogical framework for domain-specific languages," *IEEE Software*, vol. 26, no. 4, pp. 13–14, 2009.

[11] D. Nguyen, A. Lenharth, and K. Pingali, "A lightweight infrastructure for graph analytics," in *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, ser. SOSP '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 456–471.

[12] Q. Zhang, A. Acharya, H. Chen, S. Arora, A. Chen, V. Liu, and B. T. Loo, "Optimizing declarative graph queries at large scale," in *Proceedings of the 2019 International Conference on Management of Data*, ser. SIGMOD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 1411–1428.

[13] F. Ahmad, "Graph solver as a service," in *2023 IEEE/ACM 45th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, 2023, pp. 291–293.

[14] O. Semeráth, Á. Horváth, and D. Varró, "Validation of derived features and well-formedness constraints in dsls," in *Model-Driven Engineering Languages and Systems*, A. Moreira, B. Schätz, J. Gray, A. Vallecillo, and P. Clarke, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 538–554.

[15] S. Grüner, P. Weber, and U. Epple, "Rule-based engineering using declarative graph database queries," in *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*, 2014, pp. 274–279.

[16] K. Marussy, O. Semerath, and D. Varró, "Automated generation of consistent graph models with multiplicity reasoning," *IEEE Transactions on Software Engineering*, vol. 48, no. 5, pp. 1610–1629, 2022.

[17] B. Semeráth O., C. A.A., and B. et al, "Automated generation of consistent, diverse and structurally realistic graph models. softw syst model," *IEEE Transactions on Software Engineering*, vol. 20, pp. 1713–1734, 2021.

[18] D. Varró, O. Semeráth, G. Szárnyas, and Á. Horváth, *Towards the Automated Generation of Consistent, Diverse, Scalable and Realistic Graph Models*. Cham: Springer International Publishing, 2018, pp. 285–312.

[19] "Hyperledger Fabric Model — Hyperledger Fabric Docs main documentation." [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.2/fabric{_}model.html

Requirement-based, structural design for confidentiality in Hyperledger Fabric

Damaris J. Kangogo, Imre Kocsis
Budapest University of Technology and Economics
Department of Measurement and Information Systems
Budapest, Hungary
Email: {dkangogo, ikocsis}@mit.bme.hu

***Abstract*—Hyperledger Fabric is a permissioned blockchain-based distributed ledger framework that enables organizations to collaborate securely in a network only shared between them. Its architecture facilitates ledger data partitioning via so-called channels, ensuring that only the proper subsets of the organizations in the network consortium replicate specific data segments. However, the requirement-based design of the partitioning of a consortium-wide logical data model into channel-based data management has not been investigated yet. In this paper, we explore the application of Formal Concept Analysis (FCA) to compute the channel structures that emerge from organizational data manageability requirements. We establish the "need-to-know" data sub-models as formal concepts over organizations and use their lattice to determine data to channel partitionings.**

***Index Terms*—blockchain, Distributed Ledger Technology, Hyperledger Fabric, confidentiality, Formal Concept Analysis**

I. Introduction

After being first introduced in 2008 as a result of the Bitcoin digital currency [2], blockchain has become a disruptive force in several industries, including finance, supply chains, healthcare, and education, to enhance accountability, efficiency, and trust [16]. Blockchain uses cryptographic methods to store information and provides better security than centralized systems [9]. The fundamental idea behind blockchain technology, which has contributed to its widespread popularity, is the notion of a distributed ledger kept up-to-date by a peer-to-peer network in which every node, or peer, has an identical copy of the ledger. No single node owns the ledger, meaning there's no single party to trust concerning its content. Some kind of Byzantine fault- and error-tolerant consensus mechanism guarantees the integrity of the ledger content. The validated transactions are ordered and grouped into blocks, and a hash chain is built over the blocks, forming a chain, hence the name blockchain.

Hyperledger Fabric [1] is a leading open-source, permissioned blockchain platform that facilitates the creation of bespoke blockchain networks for consortia of collaborating organizations. Among its most important characteristics is the ability to establish *channels*, distributed ledgers managed by a subset of organizations in the Fabric network. Channels enable data partitioning, ensuring that only a proper subset of the organizations in the consortium can replicate data segments relevant to them.

However, designing and configuring channels in Hyperledger Fabric is not a trivial task. It requires careful analysis of the data model and the business requirements of the network participants, such as who can see what data and who can perform or validate what transactions. Additionally, the optimal number and channel composition may vary depending on the use case and the network dynamics. Therefore, to guarantee the data confidentiality requirements of complex use cases, there is a need for a systematic and formal approach to properly partition the logical data model and the organizations into multiple Fabric channels.

In this paper, we propose using Formal Concept Analysis (FCA) [14] as a formal approach to reason about the possible channel structures that emerge from various organisations' data access requirements. We use the formal concepts formulated over the organizations to establish data sub-models and determine the appropriate channel partitioning of the consortium-wide logical data model.

The rest of the paper is organized as follows: Section II presents the basic concepts of Hyperledger Fabric and various approaches to compartmentalizing data access in Fabric. Section III describes the conceptual definitions behind FCA and relevant use cases. We propose our approach in Section IV and discuss the results in light of Fabric channels in Section V, followed by a conclusion in Section VI.

II. Compartmentalizing data in Fabric

Hyperledger Fabric (HLF) [1], developed under the umbrella of the Hyperledger Foundation, is a modular, general-purpose blockchain framework that offers unique identity management and access control features, making it suitable for a variety of industry applications such as supply chain management, trade finance, healthcare, and government. Its modular and adaptable architecture facilitates the customization of various components, including consensus, membership, and smart contracts.

HLF network is composed of nodes provided by the organizations participating in a *consortium*, and access to the network is typically restricted to those organizations. It employs an Execute-Order-Validate consensus approach, enabling concurrent transaction execution, thereby enhancing the network's throughput [6]. Transactions are initiated by network clients (with organizational credentials)

DOI https://doi.org/10.3311/MINISY2024-015

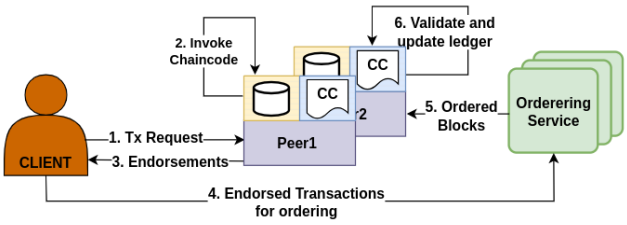


Fig. 1. Hyperledger Fabric transaction flow

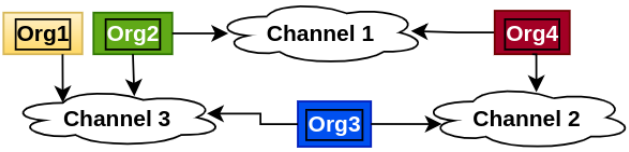


Fig. 2. Hyperledger Fabric Channels

and pre-computed by endorsing peers. The pre-processed transactions and their execution results are returned to the client, who then forwards them to the ordering service. The ordering service generates blocks without re-executing the transactions or verifying their correctness. The generated blocks are distributed to all the participating peers, who validate them and append them to their respective ledgers. Figure 1 shows a simplified Hyperledger Fabric transaction process described above.

One of the main approaches offered by HLF for data compartmentalization is that this transaction processing mechanism is scoped to so-called ***channels*** illustrated in Figure 2. A channel is a distributed ledger operated (and synchronized) by a subset of the organizations in the network. Channels share network-level services, which are important for transaction ordering, authentication, and authorization, but they remain logically isolated from each other. Smart contracts, in Fabric terminology, *chaincodes* (CC on Figure 1) are also deployed on channels; a chaincode on a channel can be able to call a chaincode on a different channel during its execution if the channels share an appropriate number of peers, but the cross-channel call has to be side-effect-free (i.e., only queries are allowed, not writes).

Other approaches include the use of private data collections, Fabric private chaincodes, and access control lists.

A ***Private Data Collection*** (PDC) is a subset of data stored in a separate, private database called a sideDB on authorized peer nodes only instead of being recorded on the ledger. PDCs enable a select group of channel members to endorse, commit, and query private data while keeping it hidden from the rest of the network [4]. Only the hashes of the private data objects are subject to full channel consensus. A collection configuration file defines PDCs by specifying each collection's permitted members, endorsement policy, and data retention policy.

Fabric Private Chaincode (FPC), introduced in [3], permits the execution of chaincode in a trusted execution environment (TEE), such as Intel SGX. FPC seeks to ensure the secrecy and integrity of chaincode execution and chaincode state. It also ensures the privacy of chaincode arguments as well as the transaction read/write sets.

Last but not least, it is also an option to create Access Control Lists (ACLs), to be observed by chaincode essentially at the application level. However, a malicious organization can easily access the ledger states of its own peers through off-chain means. This means that such ACLs are not a solution for organizational-level confidentiality requirements.

III. Formal Concept Analysis

Formal Concept Analysis (FCA) [5, 14] is a mathematical framework that enables the formalization of concepts and concept hierarchies based on a *formal context* $C = (X, Y, R)$, where $X = \{x_1 \dots x_n\}$ is a set of objects, $Y = \{y_1 \dots y_j\}$ is a set of attributes, and R is a binary relation between them (such that $R \subseteq (X \times Y)$). Formal contexts are usually represented as a cross table with the objects as rows, and the attributes as columns, and each table entry is a boolean, indicating whether the object x_i has an attribute y_j. From a given context, FCA extracts formal concepts, which consist of an ordered pair (A, B) where A is a subset of all objects (X) also known as the "extent" and B is a subset of all attributes (Y) also called the "intent".

The operators $\uparrow: 2^X \rightarrow 2^Y$ and $\downarrow: 2^Y \rightarrow 2^X$ are defined for every extent $(A \subseteq X)$ and intent $(B \subseteq Y)$ using the operations $A^{\uparrow} = \{y \in Y \mid \forall x \in A : (x, y) \in R\}$, and $B^{\downarrow} = \{x \in X \mid \forall y \in B : (x, y) \in R\}$ such that $A^{\uparrow}$ is the set of all attributes shared by all objects from A and $B^{\downarrow}$ is the set of all objects that share all of the attributes from B (more details in [5, 14]). Therefore, every object in the object set shares every attribute in the attribute set, and every attribute in the attribute set must be shared by all the objects in the object set.

The hierarchical ordering among these concepts represents a partial order relation where each concept is connected to its more general (*super-concept*) and more specific (*sub-concept*) concepts, reflecting the inclusion of extents or intents. For instance, a concept (A_1, B_1) is a sub-concept of (A_2, B_2) if $(A_1, B_1) \leq (A_2, B_2)$ and iff $A_1 \subseteq A_2$ and $B_2 \subseteq B_1$. Under the closure operation, the collection of all pairs of concepts in a given context forms a complete **lattice** [5] (also called concept lattice), which can be visualized as a line diagram. In addition to the concept lattice, FCA algorithms can derive attribute implications and association rules from the formal context.

A. Relevant Use Cases

Formal concept analysis has been applied in numerous domains to discover hidden knowledge, for knowledge representation, promoting reasoning, as well as decision-making [13]. Researchers have also shown the application of FCA in the security domain, specifically for modelling access control. For instance, [10, 11] and [15] used FCA to model role-based access control (RBAC). [10] proposed a method for constructing a dyadic formal context (or simply a formal

context) from a triadic security context, which is a three-dimensional access control matrix representing the access permissions of users, roles, and objects. They also performed attribute exploration on the formal context to generate a set of implications that define the roles and permissions. From their analysis, the authors claim that the proposed solution adheres to RBAC constraints, such as static separation of duties and role hierarchy. [11] proposed a way for modelling RBAC permissions with triadic concept analysis (TCA), an extension of FCA that can model complex relations among three sets of entities without converting the triadic access control context into dyadic formal contexts. They showed how TCA can suitably handle role hierarchy and constraints in RBAC without transforming the triadic context into dyadic formal contexts. [15] proposed a methodology for modelling RBAC using three-way formal concept analysis (3WCA). 3WCA provides two types of three-way concepts and associated lattices to allow three-way decisions from a binary information table. The authors conclude that, unlike the existing two-way decisions in classical and triadic FCA, 3WCA can better represent RBAC policy that follows role hierarchy and RBAC constraints.

[12] proposed a method for modelling Chinese wall access control (CWAC) using FCA. To represent the hierarchical structure of access permissions and conflicts, the authors created a concept lattice from the formal context, which included consultants and companies and a binary relation between them (access rights). They also defined a set of rules to check the validity of the lattice for the CWAC properties. The authors report that, based on their analysis, the lattice structure created is able to implement the Chinese wall security policy's access rights and regulations, such as the read-and-write rule.

To the best of our knowledge, no application scenarios exist for FCA in Hyperledger Fabric specifically for modelling data isolation and security aspects. In the next section, we demonstrate how FCA can support the planning of a channel structure.

IV. Applying FCA to logical data models

Our main objective is to use FCA to compute possible channel structures that emerge from the data access requirements of various organizations in a Hyperledger Fabric network. For this, we used a simple supply chain case study. The case study presented here draws inspiration from the TPC-C benchmark framework [18], a widely recognized standard for evaluating OLTP performance. Authors in [8] developed a blockchain-based implementation of TPC-C, offering a structured approach to transforming the original database schema into a smart contract data model.

We use the following simplified system model: We model ledger content as a set of relations, mapping each TPC-C table into a relation. Omitting the more technical details means that a row r in a TPC-C source table R with primary key value pk is represented, e.g., as $(R.pk, r)$ in a Fabric channel key-value store. We model organizational access in a binary way: whether an organization shall be privy to the contents of a table (and thus have a replica of it in a Fabric-based implementation) or not. In a broader sense, an organization has read access to the data if at least one of its network peers possesses a copy of the data model. Otherwise, if an organization is not privy to the data, it should not hold a replica of such data on any of its network peers. Note that we can safely omit modeling write permissions for our purposes without the loss of generality; write permission checking can be handled in chaincode. For simple CRUD operations, note that starting from a relational model and under this simple mapping, the cross-channel dependencies we must consider are row deletions ("disappearing" primary key) and foreign key value updates (a new value must exist as a primary). Conceptually, integrity in both cases can be easily handled with cross-channel reads as smart contracts. More general transaction atomicity is a more involved question; the specifics for CRUD operation consistency and complex transaction atomicity will be worked out in future research. (Note that there are existing techniques for atomic multi-channel writes; e.g. even classic two-phase commit provides a viable approach.)

We followed the following steps in applying FCA to our problem domain:

1) **Identifying objects**: The first step was to identify the stakeholders in our supply chain example. The stakeholders represent the various organizations participating in the Fabric network and thus serve as objects in the formal context. We identified six (6) stakeholder organizations: Manufacturer, Distributor, Retailer, Regulatory agency, Courier, and Customer. These stakeholders and their roles in the supply chain example are listed in Table I.
2) **Identifying attributes**: We use the TPC-C database tables as the data that various organizations (objects) are allowed to access (or have a replica of) and are, therefore, the attributes in the formal context. There are nine (9) TPC-C database tables (Warehouse, District, Customer, Order, Order_Line, Stock, Item, New_Order, Customer_History). We have used the following abbreviations to shorten the attribute lengths: O_Line for Order_Line, N_Order for New_Order and C_History for Customer_History.
3) **Choosing a tool**: Several tools and software have been developed to handle FCA tasks [13]. Our experiment uses the ConExp tool[1], specifically version 1.3.
4) **Constructing the formal context**: Using the Concept Explorer (ConExp) tool, we created a formal context (shown in Table II) of the organizations (objects) as the rows and the data (attributes) as columns, putting a mark against the attributes that an object is allowed to access. The mark ***X*** is put against an organization and a TPC-C table (attribute) that an organization is allowed to access and, therefore, can store a replica of it on at least one of its participating peers. A cell in the table is left empty if an organization is not allowed access to the respective TPC-C table, meaning

[1]https://conexp.sourceforge.net/

TABLE I
SUPPLY CHAIN STAKEHOLDERS

Organization	Role description
Manufacturer	Transform raw materials into finished products that meet customer's needs.
Distributor	Purchase products from the manufacturer and sell to the retailer.
Retailer	Final point of contact in the supply chain. Sells product directly to the customer.
Regulatory agency	Regulate and oversee the supply chain to ensure compliance with the laws, taxes, quality, and safety requirements.
Courier	Perform point-to-point delivery to deliver products to the customer from the retailer.
Customer	Purchase products from the retailer.

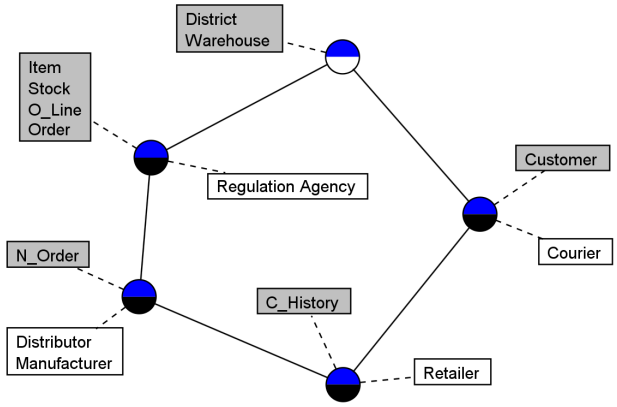


Fig. 3. Concept Lattice

it should not store a replica of the specific table in any of its network peers. The *customer* is an important stakeholder in a supply chain application; however, in our case study, the customer does not participate in the blockchain network and thus should not access any data stored on the blockchain network. For this reason, we do not include the customer in the formal context.

5) **Building a Concept Lattice**: Using the ConExp tool, we generate formal concepts from the context shown in Table II. The result is the concept lattice shown in Figure 3.
6) **Interpretation of the results**: The last step was to extract and interpret the generated concepts with respect to Hyperledger Fabric channels. Section V discusses how the concept lattice in Figure 3 relates to various channel structures.

V. FORMAL CONCEPTS TO CHANNEL SETS

Figure 3 shows the concept lattice resulting from the formal context in Table II. There are 5 nodes in the concept lattice. Each node represents a concept, and edges represent relationships between concepts. The nodes are typically labeled with sets of objects (extent) representing the various organizations and attributes (intent) representing the different TPC-C tables. The concepts are organized into layers based on their generality or specificity. To identify the formal concepts from the concept lattice, we observe the hierarchical relationships between concepts, moving from more general concepts at the top to more specific concepts at the bottom. (More details on how to explore the concept lattices can be found in [5]). The identified formal concepts from the concept lattice are listed in Table III.

Using a *greedy algorithm* like approach, we initially select the largest set of organizations along with the smallest set of associated attributes. Progressing layer by layer through the lattice, we systematically create new channels, incrementally adding the largest set of attributes possible to the new channel without breaching the confidentiality requirements we specified in Table II. We can interpret the lattice in Figure 3 and the resulting formal concepts in Table III in relation to Fabric channels as follows:

Concept Number 1: The *Warehouse* and *District* attributes owned by the first concept should be seen or accessed by all the organizations participating in the blockchain network. Thus, a design decision would be to put these data objects in a common channel maintained by all the participating parties.

Concept Number 2: To access specifically the *Order*, *O_Line*, *Stock* and *Item* data, aside from the common data objects (*Warehouse, District*), a new channel only maintained by the *Manufacturer*, *Distributor*, *Retailer* and the *Regulation Agencies* is needed.

Concept Number 3: The *New_Order* data should only be accessed by the *Manufacturer*, *Distributor*, and the *Retailer*. This implies that a channel maintained only by these three organizations should be created specifically for accessing the New_Order data.

Concept Number 4: The *Customer* data can only be accessed by the *Retailer* and the *Courier*. Therefore, a design approach would be to create a separate channel only maintained by the *Courier* and the *Retailer* specifically for sharing customer data.

Concept Number 5: Only the *Retailer* has access to the *Customer_History* data. By design, this data should not be stored on the blockchain. This means establishing a channel for maintaining such data is unnecessary if only one organization maintains it. Instead, such data can be stored on-site in a private database owned and managed by the retailer alone.

Therefore, we can perceive four (4) channels, as shown in Figure 4. The figure shows the hierarchical structure of the proposed channels resulting from the concepts listed in Table III. The nodes are connected by a partially ordered sub-

TABLE II
EXAMPLE CONTEXT

	Warehouse	District	Customer	Order	O_Line	Stock	Item	N_Order	C_history
Manufacturer	X	X		X	X	X	X	X	
Distributor	X	X		X	X	X	X	X	
Retailer	X	X	X	X	X	X	X	X	X
Courier	X	X	X						
Regulation Agencies	X	X		X	X	X	X		

TABLE III
LIST OF CONCEPTS RESULTING FROM THE EXAMPLE CONTEXT IN TABLE II

Concept Number	Concepts
1	{ [Manufacturer, Distributor, Retailer, Courier, Regulation Agencies] [Warehouse, District] }
2	{[Manufacturer, Distributor, Retailer, Regulation Agencies] [Warehouse, District, Order, O_Line, Stock, Item] }
3	{[Manufacturer, Distributor, Retailer] [Warehouse, District, Order, O_Line, Stock, Item, N_Order]}
4	{[Retailer, Courier] [Warehouse, District, Customer]}
5	{[Retailer] [Warehouse, District, Customer, Order, O_Line, Stock, Item, N_Order, C_History]}

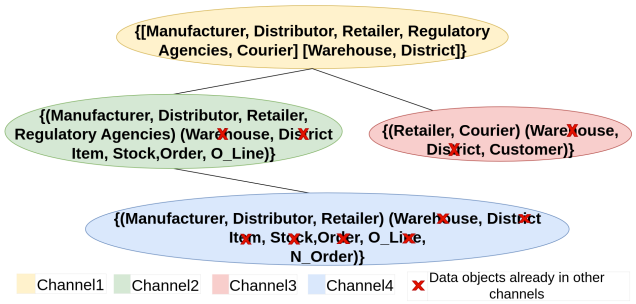


Fig. 4. Proposed hierarchical channel structure based on concepts in table III

concept super-concept hierarchy. This hierarchical structure enables us to visualize the channel-building logic by showing the hierarchy of organizations and the data objects they are allowed to access. The data objects (intents) marked with a red cross represent data objects already available to the respective organizations (extents) through other channels in which they participate and hence do not need to be replicated in every other channel.

Although channels are widely used in production in Hyperledger Fabric, introducing multiple channels can impact performance and introduce technical overhead [17, 7]. This is because channels consume CPU, memory, and storage resources on the participating nodes. Channels also compete for resources like orderers and endorsing peers, which can increase the overall latency for transaction processing, especially if channels have high transaction volumes.

Multiple ways to minimize overhead and optimize performance have been proposed, including employing sophisticated features like PDCs and ledger caching to improve transaction processing within specific channels [17, 7].

VI. CONCLUSION

In this paper, we have proposed an approach for partitioning a consortium-wide logical data model into channel-based data models based on the data access requirements of the participating organizations in a Hyperledger Fabric network using FCA. Our goal is to reason about possible channel structures with the member organizations and the data they will manage.

Using a supply chain case study, we demonstrate that FCA can generate appropriate channel partitionings based on the concepts resulting from the formal context. We used a greedy algorithm approach to reason about the possible channel structures that can result from the contexts without breaching the data confidentiality requirements.

The results prove that FCA is a promising approach for modeling data confidentiality in Hyperledger Fabric, establishing a need-to-know principle, and enhancing data security. It is important to note that, apart from channels, the proposed approach is also applicable to other data partitioning methods like PDCs, and we intend to explore this further in the future.

REFERENCES

[1] Elli Androulaki et al. “Hyperledger Fabric: A distributed operating system for permissioned blockchains”. In: *Proceedings of the thirteenth EuroSys conference*. 2018, pp. 1–15.

[2] Nakamoto S Bitcoin. *Bitcoin: A peer-to-peer electronic cash system*. 2008.

[3] Marcus Brandenburger et al. “Blockchain and trusted computing: Problems, pitfalls, and a solution for Hyperledger Fabric”. In: *arXiv preprint arXiv:1805.08541* (2018).

[4] Hyperledger Fabric. *Private data*. https://hyperledger-fabric.readthedocs.io/en/release-2.5/private-data/private-data.html.

[5] Bernhard Ganter and Rudolf Wille. *Formal concept analysis: mathematical foundations*. Springer Science & Business Media, 2012.

[6] Christian Gorenflo et al. "FastFabric: Scaling Hyperledger Fabric to 20 000 transactions per second". In: *International Journal of Network Management* 30.5 (2020), e2099.

[7] Houshyar Honar Pajooh et al. "Experimental performance analysis of a scalable distributed Hyperledger Fabric for a large-scale IoT testbed". In: *Sensors* 22.13 (2022), p. 4868.

[8] Attila Klenik and Imre Kocsis. "Porting a benchmark with a classic workload to blockchain: TPC-C on Hyperledger Fabric". In: *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*. 2022, pp. 290–298.

[9] C Komalavalli, Deepika Saxena, and Chetna Laroiya. "Overview of blockchain technology concepts". In: *Handbook of Research on Blockchain Technology*. Elsevier, 2020, pp. 349–371.

[10] Ch Aswani Kumar. "Designing role-based access control using formal concept analysis". In: *Security and communication networks* 6.3 (2013), pp. 373–383.

[11] Ch Aswani Kumar et al. "Role based access control design using triadic concept analysis". In: *Journal of Central South University* 23 (2016), pp. 3183–3191.

[12] S Chandra Mouliswaran, Ch Aswani Kumar, and C Chandrasekar. "Modeling Chinese wall access control using formal concept analysis". In: *2014 International Conference on Contemporary Computing and Informatics (IC3I)*. IEEE. 2014, pp. 811–816.

[13] Prem Kumar Singh, Cherukuri Aswani Kumar, and Abdullah Gani. "A comprehensive survey on formal concept analysis, its research trends and applications". In: *International Journal of Applied Mathematics and Computer Science* 26.2 (2016), pp. 495–516.

[14] Frano Škopljanac-Mačina and Bruno Blašković. "Formal concept analysis–overview and applications". In: *Procedia Engineering* 69 (2014), pp. 1258–1267.

[15] Chandra Mouliswaran Subramanian, Aswani Kumar Cherukuri, and Chandrasekar Chelliah. "Role based access control design using three-way formal concept analysis". In: *International Journal of Machine Learning and Cybernetics* 9 (2018), pp. 1807–1837.

[16] Bayu Adhi Tama et al. "A critical review of blockchain and its current applications". In: *2017 International Conference on Electrical Engineering and Computer Science (ICECOS)*. IEEE. 2017, pp. 109–113.

[17] Parth Thakkar, Senthil Nathan, and Balaji Viswanathan. "Performance benchmarking and optimizing Hyperledger Fabric blockchain platform". In: *2018 IEEE 26th international symposium on modeling, analysis, and simulation of computer and telecommunication systems (MASCOTS)*. IEEE. 2018, pp. 264–276.

[18] TPC. *Overview of the TPC-C Benchmark*. https://www.tpc.org/tpcc/detail5.asp.

Using Fault Tolerant Design Patterns to Assure Data Veracity

Nada Akel, László Gönczy
Department of Measurement and Information Systems
Faculty of Electrical Engineering and Informatics
Budapest University of Technology and Economics
Budapest, Hungary
nada.akel@edu.bme.hu , gonczy.laszlo@vik.bme.hu

***Abstract*—Data forms the vital asset of many organizations, as the quality of their decisions depends on the quality of their data. Trust in data is, therefore, critical. This paper aims to evaluate different aspects of data quality, examine the existing data veracity characteristics, and propose a methodology to assess the impact of fault-tolerant design patterns on data veracity. Data generated by IoT devices often reveals characteristics such as noise, incompleteness, and imprecision [1], which make it a prime example for data quality assessment. This paper investigates how we can effectively address the attributes and characteristics associated with data veracity by applying fault-tolerant design patterns within the data processing workflow.**

***Index Terms*—Data quality, data veracity,CPS , IOT Data, ISO/IEC 25012, fault tolerance, data processing**

I. Introduction

The swift incorporation of cyber-physical systems (CPS), such as the Internet of Things (IoT), within today's technological framework in diverse fields highlights the importance of addressing dimensions and challenges related to data quality. IoT data often comes with various quality issues during data generation, collection, transmission, and parsing [2]. It is essential to address and correct potential quality issues to benefit from this data. Some applications require a decision before the actual event occurs. For a long time, the AI industry has focused on improving models, libraries, and frameworks to deal with the data. However, improving data quality often has a more considerable input on the overall trustworthiness of data-driven applications [3]. In this context, we aim to assess different aspects of data veracity by exploring taxonomies, data quality standards, guidelines, and tools. We analyze how to effectively address specific data quality issues using fault-tolerant design patterns, such as N-Version Programming and Recovery Block. We aim to design a more resilient data processing workflow, focusing on enhancing data quality.

II. Data veracity

This section introduces the definition of data veracity and discusses data quality standards, dimensions, and taxonomies.

A. Definition

Data scientists have identified a series of characteristics that represent Big Data, commonly known as the V words. Veracity, the fourth V of Big Data, is used in data analytics research to cover data quality, accuracy, and truthfulness. In the context of IoT applications, veracity refers to the problem associated with data usability and quality [1]; one of the most important aspects of data veracity, Data Quality which refers to the degree to which a set of inherent data characteristics fulfills requirements [4], how suitable the gathered data are for providing ubiquitous services for IoT users [5]. Although there is no standard definition for data veracity in the literature, this term is used to express the level of trust in the collected data, its accuracy, and the data source's reliability.

Dig deeper into studies about the dimension of Veracity and enrich our understanding. Additionally, exploring the established data quality standards provides a structured approach to assess and improve the reliability and accuracy of data.

B. Data quality standards

In our paper, we will concentrate on the ISO standards among the multiple standards related to data quality. ISO/IEC introduced the ISO/IEC 25000:2005 document [6], also known as the ISO 25000 family, to standardize and unify software product quality standards. Within this family, ISO/IEC 25012, titled "Data Quality Model," and ISO/IEC 25024, focused on data quality measurement, are particularly noteworthy. ISO/IEC 25012 establishes a general quality model for structured data within computer systems, classifying quality attributes into fifteen characteristics analyzed from inherent and system-dependent perspectives. These characteristics are assigned varying importance and priority based on individual evaluators' specific needs.

- **Inherent data quality** refers to the degree to which data quality characteristics have the intrinsic potential to satisfy stated and implied needs when data is used under specified conditions.
- **System-dependent data quality** refers to the degree to which data quality is reached and preserved within a computer system when data is used under specified conditions.

Another standard related to data quality is ISO 8000 [4]; the ISO 8000 series provides frameworks for improving data quality for specific kinds of data. It defines data characteristics relevant to data quality and provides guidelines for enhancing it.

DOI https://doi.org/10.3311/MINISY2024-016

C. Related work

A wide range of problems can affect data quality. As a result, the outcome derived from these data will be affected. So, it is essential to fully understand the wide variety of dirty data that may exist in data sources captured by different taxonomies. [7] proposed a taxonomy developed from real-world databases covering thirty-five dirty data types. They follow a bottom-up approach, from the lowest-level problems (the ones that occur in a single attribute value of a single tuple) to the highest-level problems (those that involve multiple data sources). [8] divides data quality issues into two categories: single-source and multi-source. However, unlike [7], they do not differentiate the problems at a single source into those in a single relation and those in multi relations. On the other hand, [9] suggested a dirty data taxonomy based on data quality rules. Thirty-eight distinctive dirty data types were defined, creating the most comprehensive taxonomy.

III. Fault tolerant design patterns

In this section, we partially introduce the definition of fault tolerance and discuss two of its patterns and their implementation.

As a service represents a series of external states of the system, a service **failure** means that one or more external states of the system deviate from the correct service state. The deviation is called an **error**. An error adjudged or hypothesized cause is called a **fault**. Faults can be internal or external to the system; the fault is active when it causes an error dormant in other cases [10].

Fault tolerance means the ability to avoid service failures in the presence of faults [10]. It is the system or components' ability to function normally, even with the existence of hardware faults or software faults [11]. It aims to avoid service failures in the presence of active faults, which is crucial for designing dependable and secure systems. Redundancy is the foundation of fault tolerance. Over time, numerous fault-tolerant design patterns have been developed and proven effective in enhancing dependability and security.

A. Voting design pattern

It is one of the fundamental design patterns where replicated components execute identical operations concurrently. A majority-voting system (a voter) compares the N results generated to confirm the correct outcome. The system operates appropriately as long as most components are free from faults. They are choosing an odd value for N to facilitate majority voting. In the typical scenario, N is set to three, enabling the masking of a fault in one component [12].

N-Version Programming, one of the software implementations of fault-tolerant design patterns, utilizes the voting pattern and relies on design diversity. The concept involves independently developing N functionally identical software versions based on the exact specification [12] as shown in Figure 2.

B. Failover design pattern

Another design pattern is known as failover. It involves activating a spare component when an in-built error detection unit identifies a fault in an active component. Only one component operates in this design, while the remaining N - 1 components act as spares. Standby sparing has three types: hot, warm, and cold. In the hot case, the spare components are actively involved in computations, prepared to take over at any moment. For warm standby, the spare components are initialized and remain idle. In cold standby, the spare components are powered up only when their usage is needed [12]. **The Recovery Block** represents a software application of the cold failover design pattern that utilizes the checkpoint and restart technique across multiple software versions to address faults in software [12]. Initially, the main version is active, and its results are checked in the acceptance test (AT). If the AT does not pass, an alternative version takes over, as shown in Figure 2.

IV. Assuring data veracity through fault-tolerant design patterns

This section explores how IoT data quality characteristics can be addressed using previously discussed patterns.

The fault originating in a data source can be permanent or transient. It can be a natural fault caused by natural phenomena without human participation or a human-made fault. The human fault could be a malicious or non-malicious fault introduced without malicious objectives [10].

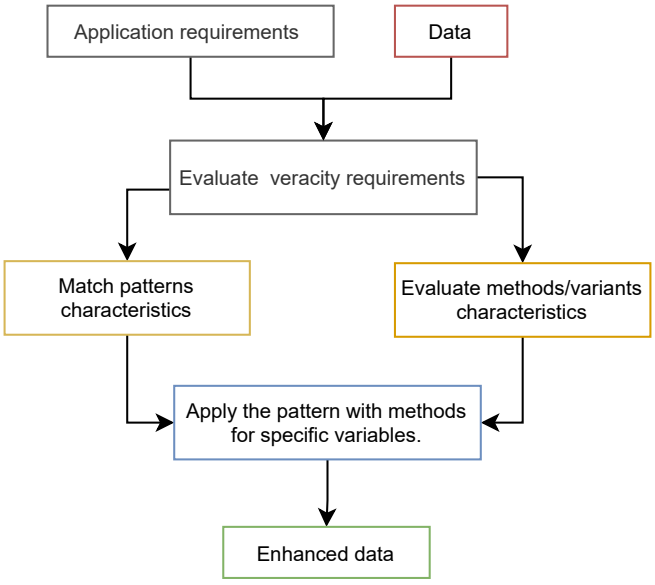


Fig. 1. Enhancing veracity using Fault tolerance patterns.

The broader version of our proposed approach is illustrated in Figure 1. After getting the application requirements and the raw data as input, we evaluate the data veracity requirements and identify the veracity dimension related to our use case. In this step, a data profiling tool can be used to identify

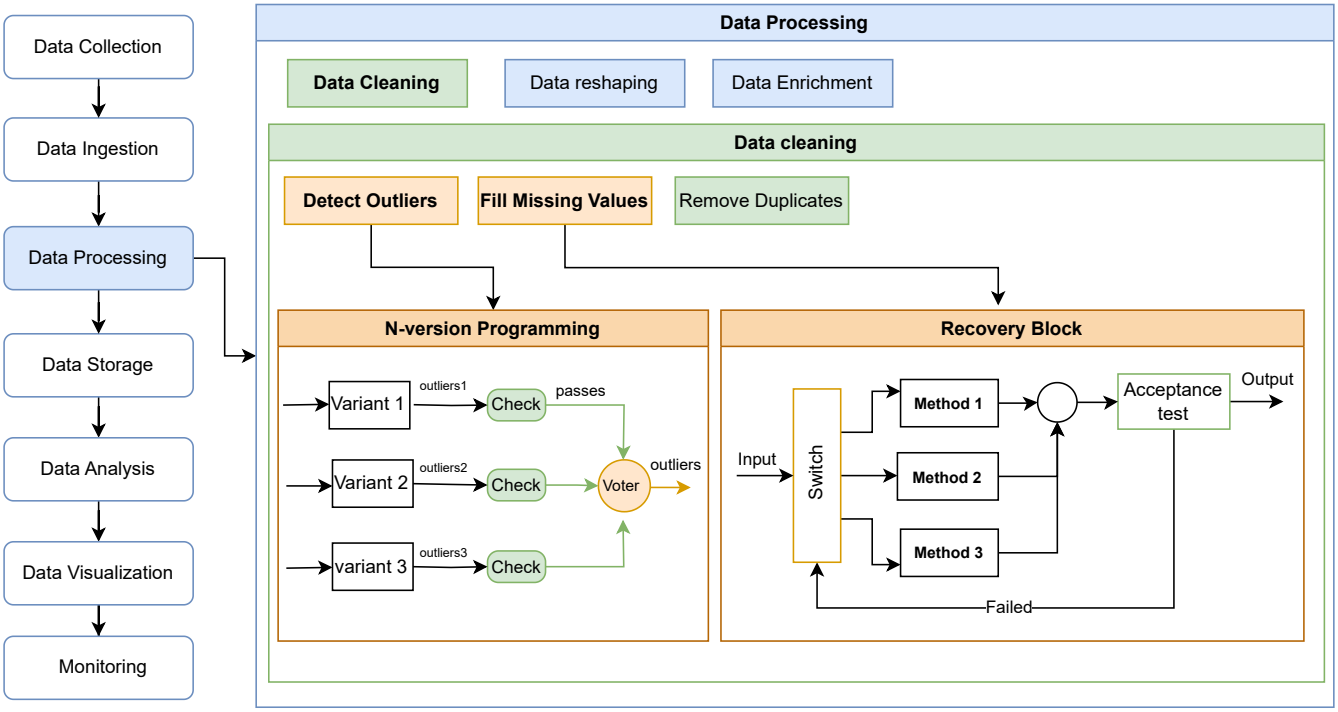


Fig. 2. Data Processing Workflow

the characteristics of the data. Following this, we assess existing methods known for their efficiency in fulfilling these veracity requirements. The next step involves the analysis of various fault tolerance patterns, aligning their characteristics with our needs to select the most fitting one. Finally, we apply the chosen pattern to the methods, incorporating it with specific variables and considerations derived from our detailed analysis of the application requirements. The output is the data enhanced to meet the application's requirements.

A simple IoT architecture adopted consists of three layers: the physical perception layer, The network layer, and the Application [13]. Faults could happen in any of the IoT layers. Sensors in the physical perception layer may be imperfect and often have a margin of error due to various unexpected factors, such as electromagnetic interference, packet loss, or issues during signal processing. This error margin becomes significant when dealing with many devices. It can lead to an accumulated chance of error occurrence, effectively reducing the generated data accuracy and quality [5]. Similarly, the fault could occur in the network layer due to unstable connectivity and intermittent loss of connection. The data processing approaches should be resilient to faults and failures to provide a reliable service.

Our emphasis is on the application layer, where the execution of the control and data plane application logic occurs. The main challenge is the development of efficient data processing and analytics methods for different levels of complexity, from simple to rich analytics based on the requirements of use cases where timely information generation is critical for some IoT systems, and the data should be processed before the data becomes outdated [1]. On the other hand, data cleaning approaches, which generally interact with the physical perception layer, should be resilient to faults and node failures to provide a stable service [5].

A **Data processing workflow** is a model representing a sequential execution of tasks to accomplish a specified objective with the data, as defined by, e.g., CRISP-DM [14]. We used a general data workflow, which aligns with the third stage of CRISP-DM, the Data preparation stage, where we prepare the data for building the predefined model. The data workflow tasks involve diverse operations, including data collection, ingestion, processing, storage, analysis, visualization, and monitoring. We evaluate how certain operations within this workflow may be enhanced using fault-tolerant design patterns, mainly focusing on improving data quality. We present a methodology that utilizes N-Version Programming and Recovery Block design patterns for addressing the **processing phase**, which involves multiple tasks that enhance the data accuracy, consistency, and completeness, such as cleaning, transforming, and enriching. Figure 2 illustrates our proposed strategy that employs multiple methods for tasks such as outlier detection and missing value handling.

A. N-Version programming for outliers detection

Outliers are readings that fall outside what is considered a normal state [15]. An outlier could represent an error, an event, a point anomaly, a contextual anomaly, or a collective anomaly [5]. The main objective is to manage outliers that align with our specific analysis purposes: clustering,

classification, forecasting, or monitoring critical systems. In critical systems, outliers frequently signify important events, warranting emphasis; conversely, outliers may be irrelevant to the analysis during classification tasks and could be excluded. However, these outliers might serve as a basis for further study in different scenarios.

Various factors can lead to outliers in data, and each outlier detection algorithm uses a unique approach to identify the outliers, resulting in its own set of outliers. Some of these outliers overlap with those detected by other methods. Using a consensus or voting approach among different methods can enhance the accuracy of outlier detection compared to relying on a single method alone.

In our proposed method, we employ the concept of **N-Version Programming** in the context of outlier detection using different variants to detect outliers in a specific data set. Each variant identifies outliers using a method known for effectiveness in outlier detection, where every variant independently detects outliers within the input data using its specific approach. Subsequently, we propose conducting an additional **invariant check** on the output of each variant to determine its contribution to the voting step. The invariant check is domain case specific and highly related to the case of the study. A **voting mechanism** is then employed to assess the consensus among the detected outliers. We can choose between two **voting type**: **majority voting**, where the data point is considered as an outlier if it is detected by the majority of variant, or **consensus of all participating variants**, where the data point is considered as an outlier if all variant detect it. Only data points consistently labeled as outliers according to the chosen voting type are considered outliers. Addressing the specific requirements of a business case presents a challenge, particularly in determining the invariant check thresholds that should be done before starting the voting process to ensure accuracy and consistency in the results. Figure 2 visually represents the **extended N-Version Programming** applied to outlier detection.

The proposed approach is independent of the data processing schedule. It can be applied to each data point or the entire data set, depending on the schedule and operating mode of the system.

After the detection process, we should analyze the result, examining whether there is a spatial or temporal relation between the detected outlier. At the same time, we should decide how we will treat these outliers according to our purpose. For example, we should not remove the outliers in the critical systems case. On the other hand, in some business cases, like when we build a prediction model, removing the outliers will enhance the model prediction.

Various methods could be employed to identify outliers, such as statistical methodologies, clustering-based approaches, supervised or unsupervised learning algorithms, and domain-specific knowledge. Each technique comes with its own set of advantages and limitations. Consequently, selecting the most suitable strategies for a particular data set is crucial.

B. Recovery Block for missing values

Missing values indicate the absence of data regarding a specific entity. Since data-driven knowledge extraction processes utilize these data, such gaps may result in partial knowledge or wrong decisions. Consequently, missing values can contribute to a decline in data quality. Missing values in a data set can be detected easily. We can use diverse methods to interpolate the missing value. A **Recovery Block** can be employed using different methods to interpolate **missing values** based on other values. Each one estimates the missing value using an approach known for effectiveness in interpolation. It uses an **acceptance test** as a threshold to assess whether the estimated value is acceptable, focusing on the fact that the estimated value does not change the characteristics of the data. If the outcome fails to surpass the initial method's threshold, the data is transmitted to the subsequent method. Tuning the threshold of the Recovery Block detector may necessitate different tuning for optimal performance.

For example, we can start using the Univariate Nearest Neighbor method, considered the most straightforward method for filling in missing values [16]. Next, we assess the characteristics of our data distribution. If the acceptance check finds unacceptable characteristics such as frequent values or unusual skewness, we switch to the second method, like Linear Interpolation. After applying this method, we check if the values meet our acceptance test. If the previous methods do not produce an acceptable value, we can use a dedicated simulation, which may need more design time effort and computational resources runtime.

Implementing the proposed method can help us improve part of the **inherent data quality** characteristics. A robust outlier detection methodology using N-Version programming enhances the **consistency** and **accuracy** of our data. Jointly, employing a recovery block for interpolation assists in addressing data **completeness** and **currentness** and **accuracy** issues.

We built a Jupyter Notebook to analyze the hourly energy consumption of 22 houses from the HUE dataset [13]. With the object of discovering trends and patterns within this data. The presence of outliers could lead to misleading interpretations, so our requirement in this case is to remove the outlier from the data set. Our process begins with an assessment of various outlier detection methods to identify the one most relevant to our scenario. We detect the outlier using three distinct statistical methods. Subsequently, we select an appropriate fault tolerance design pattern, which is the N-version programming that aligns with our case. Finally, we applied all-participating voting to detect the outliers. This approach enables us to accurately identify and eliminate the most precise outliers from the data, ensuring a more reliable analysis.

V. Conclusion and Future Work

In conclusion, our paper has dived into utilizing fault tolerance design patterns to enhance data veracity within a data workflow. We built two of the data workflow cleaning steps using the logic of the fault tolerance design pattern.

We started with addressing the data processing phase due to its significant impact on data quality. We chose two basic patterns. Through our exploration, we have identified that the issues of accuracy, completeness, currentness, and consistency can be addressed and enhanced by employing N-version programming and recovery block strategies. These findings underline the potential of these fault tolerance techniques in strengthening data integrity and reliability in complex data workflows.

Creating a model that employs the fault tolerance design pattern in data processing workflow operations is a future work after developing a more comprehensive understanding of how we can improve other data processing workflow tasks using fault tolerance patterns, how we can employ the other's fault tolerance design pattern in the data processing workflow, evaluating the applicability of the proposed model in data processing tools.

Regarding detecting outliers, a good area for future work involves enhancing our semantics understanding of outliers, including exploring potential spatial or temporal connections among the identified outliers. Additionally, there is a prospect to investigate the feasibility of applying a fault correction to the detected outliers.

VI. 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] X. Liu, S. Tamminen, X. Su, P. Siirtola, J. Röning, J. Riekki, J. Kiljander, and J.-P. Soininen, "Enhancing Veracity of IoT Generated Big Data in Decision Making," in *2018 IEEE Intl. Conf. on Pervasive Computing and Communications Workshops*. IEEE, 2018, pp. 149–154.

[2] T. Mansouri, M. R. Sadeghi Moghadam, F. Monshizadeh, and A. Zareravasan, "IoT Data Quality Issues and Potential Solutions: A Literature Review," *The Computer Journal*, vol. 66, no. 3, pp. 615–625, 11 2021. [Online]. Available: https://doi.org/10.1093/comjnl/bxab183

[3] N. Sambasivan, S. Kapania, H. Highfill, D. Akrong, P. Paritosh, and L. M. Aroyo, "Everyone Wants to Do the Model Work, Not the Data Work: Data Cascades in High-Stakes AI," in *Proc. of the 2021 CHI Conf. on Human Factors in Computing Systems*, ser. CHI '21. New York, NY, USA: ACM, 2021. [Online]. Available: https://doi.org/10.1145/3411764.3445518

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

[5] A. Karkouch, H. Mousannif, H. Al Moatassime, and T. Noel, "Data Quality in Internet of Things: A State-of-The-Art Survey," *Journal of Network and Computer Applications*, vol. 73, pp. 57–81, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804516301564

[6] ISO, *ISO/IEC 25024: 2015: Systems and Software Engineering-Systems and Software Quality Requirements and Evaluation (SQuaRE)-Measurement of Data Quality*. ISO/IEC, 2015.

[7] P. Oliveira, F. Rodrigues, P. Rangel Henriques, and H. Galhardas, "A Taxonomy of Data Quality Problems," *Journal of Data and Information Quality - JDIQ*, 01 2005.

[8] E. Rahm and H. Do, "Data Cleaning: Problems and Current Approaches," *IEEE Data Eng. Bull.*, vol. 23, pp. 3–13, 01 2000.

[9] L. Li, T. Peng, and J. Kennedy, "A Rule Based Taxonomy of Dirty Data," *GSTF INTERNATIONAL JOURNAL ON COMPUTING*, vol. 1, 01 2011.

[10] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE Trans. on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, 2004.

[11] M. Al-Kuwaiti, N. Kyriakopoulos, and S. Hussein, "A Comparative Analysis of Network Dependability, Fault-Tolerance, Reliability, Security, and Survivability," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 2, pp. 106–124, 2009.

[12] K. Ding, A. Morozov, and K. Janschek, "Classification of Hierarchical Fault-Tolerant Design Patterns," in *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, 2017, pp. 612–619.

[13] A. Goknil, P. Nguyen, S. Sen, D. Politaki, H. Niavis, K. J. Pedersen, A. Suyuthi, A. Anand, and A. Ziegenbein, "A Systematic Review of Data Quality in CPS and IoT for Industry 4.0," *ACM Comput. Surv.*, vol. 55, no. 14s, jul 2023. [Online]. Available: https://doi.org/10.1145/3593043

[14] C. Schröer, F. Kruse, and J. M. Gómez, "A Systematic Literature Review on Applying CRISP-DM Process Model," *Procedia Computer Science*, vol. 181, pp. 526–534, 2021.

[15] A. Karkouch, H. Mousannif, H. Al Moatassime, and T. Noel, "Data Quality in Internet of Things: A State-of-The-Art Survey," *Journal of Network and Computer Applications*, vol. 73, pp. 57–81, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804516301564

[16] H. Junninen, H. Niska, K. Tuppurainen, J. Ruuskanen, and M. Kolehmainen, "Methods for Imputation of Missing Values in Air Quality Data Sets," *Atmospheric Environment*, vol. 38, no. 18, pp. 2895–2907, 2004.

Landmark Estimation for Qualitative Analysis over Distributed Traces

Bertalan Zoltán Péter, Imre Kocsis
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
bpeter@edu.bme.hu, ikocsis@mit.bme.hu

***Abstract*—The proliferation of cloud-based, distributed microservices architectures poses novel challenges in system-level diagnosis for performance as well as hard errors. Industry has responded with the widescale adoption and enablement of distributed, standardized transaction tracing. However, distributed tracing largely lacks proper diagnostic inference support. This paper proposes a qualitative error propagation model for distributed tracing-based monitoring and probing. An approach to estimate model landmarks – quantization thresholds – from observations in accordance with possible transaction structure is also proposed. The presented results enable the application of the results and approaches of classic Error Propagation Analysis to the cloud-based microservice domain.**

***Index Terms*—distributed tracing, qualitative reasoning, error propagation analysis, landmark estimation, microservices, cloud**

I. Introduction

Ever since production software has existed, it has been necessary to collect telemetry data to understand the root cause of errors and observe system behaviour. Decades ago, collecting logs and metrics from the server hosting the application was sufficient. However, since production applications have replaced their monolithic architectures with distributed microservices, obtaining telemetry data has become much more challenging, as now there are potentially hundreds of components involved in serving each request.

The telemetry data collection from such architectures is called Distributed Tracing (DT) [1]: 'a type of correlated logging that helps one gain visibility into the operation of a distributed software system for use cases such as performance profiling, debugging in production, and root cause analysis of failures or other incidents.'

From a system diagnostics perspective, DT can be used to validate task execution models or knowledge and to drive diagnostic models derived from such system models [2]. Its strength lies in its ability to provide delay and error traces correlated on the transaction level, which 'test' several components of distributed systems. Consequently, DT enables the application of existing logical diagnostic approaches [3] to analyze issues such as whether some inter-component resource-dependence assumptions are correct. Further, it may be an effective non-invasive tool for online diagnostics [4].

Klenik has already presented a general Allen-interval-algebra-based model for logical reasoning over the execution times of hierarchically mixed sequential and parallel task executions in [2]. However, his methodology has not yet been applied to the synthesis and parameterization of activity-model-based qualitative diagnostic models.

As a concrete example, consider the microservices-based architecture with representative complexity from [5] in Figure 1. Having traces for a given execution path already provides valuable information regarding what may have gone wrong during serving requests. However, combining several concurrent traces may provide even richer diagnostic options. For the sake of a simple example, in the figure, two disjunct traces have been marked with thickened arrows (and highlighted executors). If we compare several traces on the two paths and notice a time-based correlation between increased execution times, we can make logical deductions that possibly contradict prior co-deployment assumptions based on the system. For instance, if the *travel/2* and *seat* services were allocated to the same physical host, this host's failure or degraded service would cause detectable effects on both trace paths.

Our research goal is to investigate the applicability of qualitative Error Propagation Analysis (EPA) [3, Fig. 3.2], [6] approaches to the diagnostics of web-scale distributed systems, where trace-based diagnostics are possible. This is motivated by system-level diagnostics being strikingly underdeveloped in such environments as of date.

We do not yet consider the qualitative error propagation models' automated synthesis – that will be a natural future step of our research (on the basis of [2]). In this work, we analyze how landmarks necessary for the parameterization of such qualitative models can be selected. As an initial novel contribution, we present an approach to select good landmarks for such analyses and a prototype implementation of an estimator program that gives optimal landmarks.

The rest of this paper is organized as follows. In the next section, we summarize the technical background of DT and the related work. Then, in section III, we present our landmark estimation approach and, in section IV, our prototype implementation. Finally, we conclude our results and outline further work in section V.

The project 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.

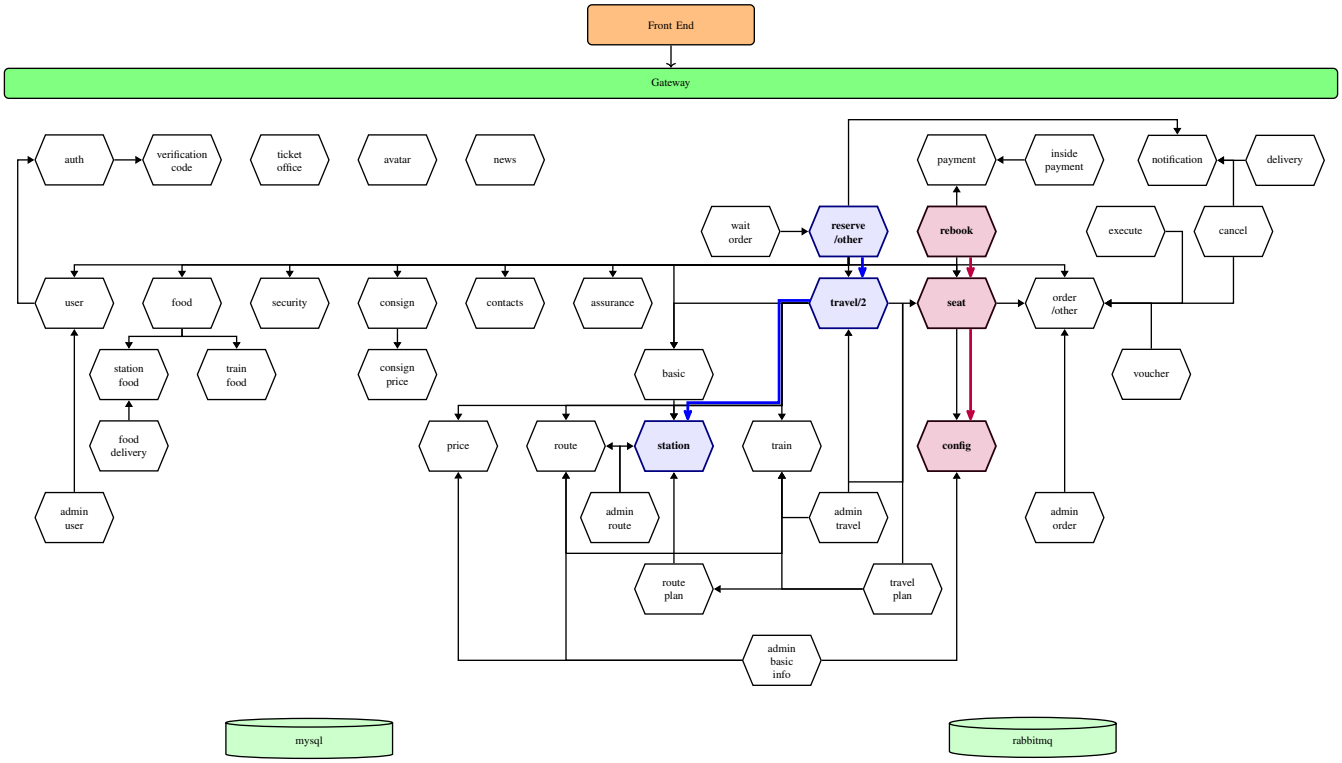


Fig. 1. Architecture of the *TrainTicket* benchmark [5]

II. EPA over Distributed Traces

Distributed microservices architectures are replacing monolithic setups. Unfortunately, such architectures can be much more complex, especially from the perspective of error diagnosis. Typically, one has a limited and partial understanding of how the services affect each other or how faults may propagate throughout the system. The number and interconnections of services may be dynamic, further encumbering such analyses.

A. Background

Historically, several tools and standards have been developed to realize DT, such as OpenTracing [7], Dapper [8], and Apache HTrace [9]. By today, most of these have been retired or deprecated – OpenTracing has been obsoleted by OpenTelemetry [10] and the two state-of-the-art tools became [11] Jaeger [12] and Zipkin [13].

OpenTelemetry [10], which is under the stewardship of the Cloud Native Computing Foundation (CNCF), is a set of Application Programming Interfaces (APIs), Software Development Kits (SDKs), and tools for instrumenting, collecting, and exporting telemetry data. It also maintains a specification that – among other concepts – defines data models for traces and spans.

A distributed *trace* is defined as a set of events triggered by a single logical operation, e.g., serving a top-level user request. Traces comprise *spans,* representing a single transaction operation. Spans always include information such as unique identifiers, operation names, timestamps, and relationships with other spans. Encapsulating additional knowledge in span data such as HyperText Transfer Protocol (HTTP) status codes is also possible via *attributes*. We note that such additional information carried within spans can also be the subject of qualitative diagnostics; thus, multivariate, vector-based fault propagation can also be supported by DTs. Traces and spans are typically represented as JavaScript Object Notation (JSON) files. Figure 2 show a simple visualization of a trace and its spans.

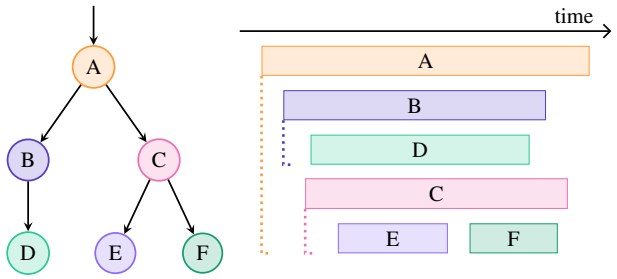


Fig. 2. Visualization of a Distributed Trace [10]

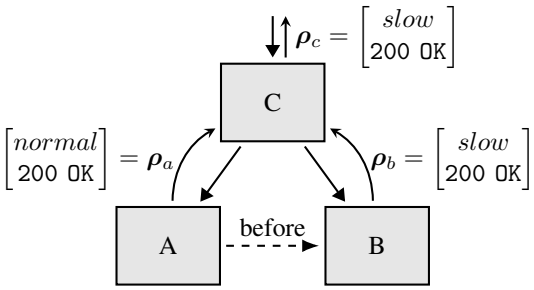


Fig. 3. Simplified Architecture of Three Services A, B, and C

B. Related Work

The qualitative analysis of microservices-based systems is mainly unexplored. However, we are aware of some initial work about logical reasoning and fault diagnosis in these environments. Namely, [14] presents a logical reasoning approach to manage uncertainties in such systems. The foundations of this work also lie in analyzing DTs using logical programming techniques. Furthermore, [15] proposes a fault diagnosis method based on DTs.

While these works establish the foundations of qualitative analysis over DTs, they do not consider landmark selection methods. In this paper, we focus on how to select optimal landmark values for discretization in such environments.

III. Landmark Estimation Approach

This section presents our approach to estimating suitable landmarks for DT data discretization. For simplicity, we will only consider a small fragment of a distributed system where three services interact. As visualized in Figure 3, service C is at the 'top-level'. It calls down to two additional services named A and B. We also assume that A and B are invoked sequentially in this order – we leave the analysis of interleaving or parallelly executed services for future work.

As explained in section II, traces may contain arbitrary information beyond timing data. For simplicity, let us only consider the delays (durations) of the spans for now. Then, vectors $\boldsymbol{\rho}_a$, $\boldsymbol{\rho}_b$, and $\boldsymbol{\rho}_c$ may be represented as scalars (more precisely, ordinals) with values from the set $\{normal, slow\}$. We should note that at least one more categorical duration value ($fast$) is most likely necessary to perform a sound analysis since services responding 'too quickly' is also a sign of erroneous system behaviour. At this stage of the work, however, we shall focus only on services responding in nominal ($normal$) or increased time ($slow$).

We model the propagation of errors as slow services slowing down those services that depend on them. This means that if $\boldsymbol{\rho}_a$ or $\boldsymbol{\rho}_b$ is $slow$, $\boldsymbol{\rho}_c$ will also be $slow$. Otherwise, if both services are operating as $normal$, $\boldsymbol{\rho}_c$ will also be $normal$. Table I summarizes these rules for clarity. Such rules are used in a specific type of EPA based on syndromes and the abstraction of the time domain and dynamics [3]. As we do not consider the direction of error propagation (which allows for nondeterministic models), the rules describe *relations,* which are applicable in both root cause analysis and impact analysis, as well as possibly hybrid diagnostic problems [16].

TABLE I
Error Propagation Rules

$\boldsymbol{\rho}_c$	$\boldsymbol{\rho}_a$	$\boldsymbol{\rho}_b$
$normal$	$normal$	$normal$
$slow$	$normal$	$slow$
$slow$	$slow$	$normal$
$slow$	$slow$	$slow$

As service C is considered top-level, we assume that its landmark value is predetermined – an expert with domain knowledge or a Quality of Service (QoS) guarantee can decide at what point a service is slow or inadequate. The landmark estimation task, in essence, comprises of finding landmark values λ_a and λ_b (for services A and B, respectively) such that when the set of observed service response times (i.e., span durations for a number of observed traces) are discretized using these landmarks, the resulting duration triples adhere to the error propagation rules. In addition, we optimize the selection of landmark values by the heuristic that the landmarks should be possible to shift as much as possible in either direction without breaking the propagation rules. A visualization of landmark selection can be seen in Figure 4.

A. Formal Optimization Problem

We define the optimization problem in general terms, i.e., for arbitrary span trees. Given $k \in \mathbb{N}$ observations and $s \in \mathbb{N}$ services (excluding the top-level service), let us define the following notations:

- Let $\boldsymbol{t}$ denote the observed response times for a service as a vector in $\mathbb{N}^k$.
- Let $\boldsymbol{T}$ be a $k \times s$ matrix that encodes the observation vectors of all services; i.e., $\boldsymbol{T} = \begin{bmatrix} \boldsymbol{t}_1 & \cdots & \boldsymbol{t}_s \end{bmatrix}$.
- Let λ denote the landmark value for a service and $\boldsymbol{\lambda}$ the set of landmark values for all s services in a vector; i.e., $\boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 & \cdots & \lambda_s \end{bmatrix}^\intercal$.

Furthermore, let us denote with $\mathbf{1}_k$ the all-ones vector (i.e., $\mathbf{1}_k = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}^\intercal \in \mathbb{N}^k$) and with $\operatorname{grsum} \boldsymbol{M}$ the grand sum of the matrix $\boldsymbol{M}$ (i.e., the sum of its elements). Finally, let $\boldsymbol{M}^{|\circ|}$ denote the element-wise absolute value of the matrix $\boldsymbol{M}$.

Then, the optimization problem is the following:

$$\begin{aligned} &\textbf{Given } \mathcal{D} : (\mathbb{N}^{k\times s}, \mathbb{N}^s) \to \mathbb{N} \\ &\qquad \text{with } (\boldsymbol{T}, \boldsymbol{\lambda}) \mapsto \operatorname{grsum}\left((\boldsymbol{T} - \mathbf{1}_k \boldsymbol{\lambda}^\intercal)^{|\circ|}\right) \\ &\textbf{We seek } \boldsymbol{\lambda}_0 \text{ such that } \forall \boldsymbol{\lambda} \in \mathbb{N}^s : \mathcal{D}(\boldsymbol{T}, \boldsymbol{\lambda}) \leq \mathcal{D}(\boldsymbol{T}, \boldsymbol{\lambda}_0) \\ &\qquad \text{and service delays discretized by } \boldsymbol{\lambda}_0 \\ &\qquad \text{satisfy the rules in Table I} \end{aligned} \tag{1}$$

B. Alternative Formulations

Here, we have shown a formulation of landmark selection as an optimization problem and will demonstrate a possible

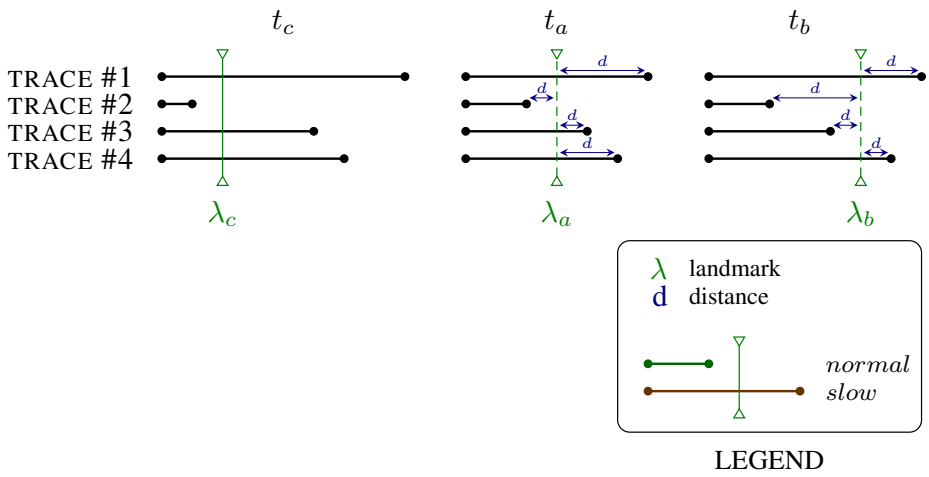


Fig. 4. Overview of Landmark Estimation

solution using Answer Set Programming (ASP) in the next section. Another valid and more direct approach would be to perform clustering on the observations and use the resulting decision boundaries as landmarks. Since overly long service execution times are likely the results of rare events, it would be advisable to rely on density rather than neighbourhood when choosing a metric for clustering.

C. Further Considerations

Duration values may vary by magnitudes in a real system due to data dependencies. We note that these extreme values need special handling in a practical application of the approach. Generally, looking at all available observations does not provide a homogenous dataset suitable for clustering or landmark optimization without extending the proposed algorithm.

Moreover, we should point out that discretizing numerical values this way implies a loss of information. Our approach preserves variable order, but statistical analyses may yield significantly different results on the discretized data.

IV. ASP-Based Prototype Landmark Estimator

We implemented a prototype that solves the landmark optimization problem (1) in ASP, using the toolkit developed at the University of Potsdam (the Potsdam Answer Set Solving Collection – Potassco [17]). Declarative programming is highly advantageous for the optimization problem at hand as it relieves the need to implement concrete algorithms and produces very readable, simple code. On the other hand, we should note that ASP does not scale well and, given more than a few observations as inputs or a domain of possible (integer) time values in the order of magnitude greater than 10^4, finding an optimal solution may take unacceptably long. For these cases, we consider approaches based on Linear Programming (LP) or Mixed-Integer LP (MILP) instead, but this is left for future work – the capabilities of ASP are sufficient for the simple proof-of-concept demonstration in this paper.

Although the prototype is agnostic of the source of observation data and what services A, B, and C represent, we should mention that we used real observation data. *TrainTicket* [5] is a benchmark microservice system developed by the Software Engineering Laboratory of Fudan University with a publically available dataset of detected anomalies in various system configurations created by Monika Steidl [18]. We have tested our prototype implementation in an ad-hoc manner on observed duration values from three arbitrarily selected services with the configuration of Figure 3. We then compared the optimal landmarks suggested by the ASP model to see how they correspond to the decision boundary when the $normal$ and $slow$ execution times are quantized by other methods (i.e., K-means clustering) and found that for the minimal example, the selected landmarks are adequate.

First, several observations from the dataset are manually encoded in facts. Traces are defined as `trace(tr_<identifier>)`; then, the execution time (i.e., duration) observations of each service in each trace are represented as trace-service-time triples in facts named `time_obs`.

```
trace(tr_5203cb8b61b215; tr_6707c0ba4b2265; [...]).
time_obs(tr_5203cb8b61b215, c, 846). [...]
```

Then, the following facts describe the set of discrete response time values, the set of existing services (in unary facts named `time` and `service` respectively), as well as the *before* relationship between service A and B:

```
time(normal; slow).
service(c; a; b).  before(a, b).
```

The contents of Table I are encoded in the following ASP rules:

```
time(Tr, c, slow) :- time(Tr, a, slow), trace(Tr).
time(Tr, c, slow) :- time(Tr, b, slow), trace(Tr).
time(Tr, c, normal) :-
    time(Tr, a, normal), time(Tr, b, normal),
    trace(Tr).
```

We establish that every service must have a single landmark value generated:

```
1 { landmark(S, L) : number(L) } 1 :- service(S).
```

This means a single `landmark(<service>, <value>)` fact will be generated for each service. Then, observed (numerical) execution times are discretized according to the landmarks:

```
time(Tr, S, slow) :-
    time_obs(Tr, S, T), landmark(S, L), T >= L.
time(Tr, S, normal) :-
    time_obs(Tr, S, T), landmark(S, L), T < L.
```

If the observed execution time in a trace exceeds the corresponding landmark value for the service, we discretize it as `slow`; otherwise, we consider it `normal`. Finally, to perform optimization, we calculate the *distances* of the landmarks from the observed duration values and ask the solver to maximize:

```
1 { landmark_dist(Tr, S, D)
    : time_obs(Tr, S, T),
      landmark(S, L), D = |L - T| } 1 :- service(S),
                                         trace(Tr).
#maximize{ D : landmark_dist(Tr, S, D) }.
```

A. Consequences of Rounding to Integers

Being a logic language, ASP is not optimally suited for processing numerical values. We circumvent this by defining a large domain of natural numbers as facts (by adding `number(1..<max>)` to the program). Unfortunately, this approach does not scale well, but we already touched on scalability issues at the start of this section.

However, working with integers also introduces the problem of having to truncate service execution durations, which are usually decimal values. It is theoretically possible to scale the observed values up so that everything is an integer, but the technically feasible solution is to allow the loss of precision and round the values to, e.g. only thousands. One should keep in mind that such rounding can significantly affect the results of the analysis.

B. Unsatisfiable Models

In the current model, such observation data may be given that there is simply no choice of landmarks to satisfy all the rules. The ASP program cannot provide a solution in these cases.

In future work, we plan to tackle this problem by changing some of the rules into soft constraints, essentially allowing some rules not to be satisfied but adding a 'penalty' to models that violate the constraints. In other words, this means the extension of the optimization problem.

V. Conclusion and Future Work

In summary, we postulate that applying classic fault-tolerant and dependable computing methodologies would enable system-level inference capabilities in microservices. More precisely, qualitative Error Propagation Analysis over distributed traces could be used for multiple types of logical reasoning (both inductive and deductive).

As quantization is an essential step in qualitative analysis with significant effects on its effectiveness, we have presented an initial approach to estimate optimal landmarks for discretizing service response time durations and an ASP-based prototype implementation.

In future work, we want to investigate other possible heuristics for landmark optimization, for example, choosing landmarks that minimize the non-determinism of the resulting system models. Furthermore, the classical diagnostic problems solvable by EPA [16] have been built on three parameters: inputs, fault activations, and outputs. Essentially, the third one can be inferred from the knowledge of any two. We believe that in the context of distributed microservices-based systems, these should be extended by at least two additional variables: hierarchical composition behaviour (formalized in Allen's interval algebra [19]) and deployment. This work is a step towards encoding tracing-based microservice diagnostic problems in configurable ASP.

References

[1] A. Parker, D. Spoonhower, J. Mace, B. Sigelman, and R. Isaacs, *Distributed Tracing in Practice: Instrumenting, Analyzing, and Debugging Microservices*. O'Reilly Media, Inc., Apr. 2020, ISBN: 978-1-4920-5660-7.

[2] A. Klenik, "Measurement-based performance evaluation of distributed ledger technologies," Ph.D. dissertation, Budapest University of Technology and Economics, 2022.

[3] A. Pataricza, *Model-based dependability analysis*, DSc Thesis, Hungarian Academy of Sciences, 2008.

[4] I. Rish, M. Brodie, S. Ma, *et al.*, "Adaptive diagnosis in distributed systems," *IEEE Transactions on Neural Networks*, vol. 16, no. 5, pp. 1088–1109, 2005. DOI: 10.1109/TNN.2005.853423.

[5] S. E. L. of Fudan University, *TrainTicket*, version 1.0.0, 2018. [Online]. Available: https://github.com/FudanSELab/train-ticket.

[6] A. Földvári, F. Brancati, and A. Pataricza, "Preliminary risk and mitigation assessment in cyber-physical systems," in *2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, 2023, pp. 267–274. DOI: 10.1109/DSN-W58399.2023.00067.

[7] O. S. Council, *The opentracing data model specification*, 2019. [Online]. Available: https://opentracing.io/specification/.

[8] B. H. Sigelman, L. A. Barroso, M. Burrows, *et al.*, "Dapper, a large-scale distributed systems tracing infrastructure," Google, Inc., Tech. Rep., 2010. [Online]. Available: https://research.google.com/archive/papers/dapper-2010-1.pdf.

[9] A. S. Foundation, *Apache htrace*, 2014. [Online]. Available: http://htrace.org/.

[10] T. O. Project, *Opentelemetry specification*, version 1.27.0, 2023. [Online]. Available: https://opentelemetry.io/docs/specs/otel/.

[11] A. Bento, J. Correia, R. Filipe, F. Araujo, and J. Cardoso, "Automated analysis of distributed tracing: Challenges and research directions," *Journal of Grid Computing*, vol. 19, no. 1, p. 9, Mar. 2021, ISSN: 1570-7873, 1572-9184. DOI: 10.1007/s10723-021-09551-5.

[12] U. Technologies, *Jaeger*, version 1.52, 2023. [Online]. Available: https://www.jaegertracing.io/.

[13] OpenZipkin, *Zipkin*, version 2.25.1, 2023. [Online]. Available: https://zipkin.io/.

[14] M. Tarnay, *Managing operational uncertainties in deployed systems using logic reasoning*, Scientific Students' Association Report, 2023. [Online]. Available: https://tdk.bme.hu/VIK/distsys/Mukodesi-Bizonytalansagok-Kezelese-Elosztott.

[15] R. Bodó, *Distributed tracing-based adaptive fault diagnosis in distributed systems*, Scientific Students' Association Report, 2023. [Online]. Available: https://tdk.bme.hu/VIK/distsys/Elosztott-Nyomkovetes-Alapu-Adaptiv.

[16] A. Pataricza and P. Urbán, "A combination of petri-nets and linear programming in design for dependability," Budapest University of Technology and Economics, Tech. Rep., 1997.

[17] M. Gebser, B. Kaufmann, R. Kaminski, M. Ostrowski, T. Schaub, and M. Schneider, "Potassco: The potsdam answer set solving collection," *AI Commun.*, vol. 24, pp. 107–124, Jan. 2011. DOI: 10.3233/AIC-2011-0491.

[18] M. Steidl, *Anomalies in microservice architecture (train-ticket) based on version configurations*, Zenodo, Aug. 2022. DOI: 10.5281/zenodo.6979726.

[19] J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, no. 11, pp. 832–843, Nov. 1983, ISSN: 0001-0782. DOI: 10.1145/182.358434.

Investigating the natural product subspace within the Transformer-VAE foundation model's drug-like molecule space.

Márk Marosi, Péter Sárközy
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
marosi.mark@edu.bme.hu, psarkozy@mit.bme.hu

***Abstract*—We explore the intricate world of natural product chemistry through the lens of computational modelling. Utilising a Transformer-VAE foundation model originally pre-trained on the GuacaMol dataset, known for its comprehensive collection of small drug-like molecules. We explore the COCONUT dataset's natural products within this model's latent space. This approach allows us to investigate these complex natural compounds' structural organisation and relationships in a latent space tailored to smaller, drug-like molecules. Our findings provide insightful revelations about the similarities and divergences between these two distinct molecular realms. We uncover new perspectives on molecular similarity and potential bioactivity by examining how natural products, with their diverse and often complex structures, are represented and structured in a latent space initially trained on more simplistic molecules. This research sheds light on the capabilities and adaptability of pre-trained models in chemical informatics. It could help pave the way for innovative approaches in discovering and analysing natural products for pharmaceutical applications.**

***Index Terms*—Drug discovery, Machine Learning, Transformer-VAE, Natural Products**

I. Introduction

In our study, we engage with the complex domain of natural product chemistry through computational modelling, focusing on a Transformer-VAE model pre-trained on the GuacaMol dataset. This dataset is noted for collecting small, drug-like molecules, providing a valuable foundation for our computational analysis. We aim to understand the COCONUT dataset's natural product representation within this model's latent space, tailored initially to drug-like molecules.

Our approach involves a careful examination of the structural details and relationships within this molecular space. We look at how natural products, characterised by their structural diversity and complexity, are represented in a latent space initially shaped by simpler molecular forms. This process allows us to explore the intersections and distinctions between these two molecular worlds.

The analysis offers insights into molecular similarities and potential bioactivities, shedding light on the representation of complex natural products in a latent space trained on simpler molecules. It also provides a perspective on pre-trained models' adaptability and representational capabilities in chemical informatics, particularly in their application to diverse and complex molecular structures.

This research contributes to the broader field of cheminformatics, exploring how pre-trained models might be used to understand and analyse the vast array of structures present in natural products. It suggests the possibility of using these models to aid in discovering and exploring new compounds for pharmaceutical use.

While this study provides valuable insights, we recognise the need for cautious interpretation and further research. The findings hint at the potential of computational models to enhance drug discovery processes, especially by leveraging the structural variety inherent in natural products. Our work aims to contribute to the ongoing development of cheminformatics, hoping to facilitate faster and more efficient discovery of therapeutically relevant compounds.

II. Data Description

Our research employs two pivotal datasets, GuacaMol [1] and COCONUT [2], each playing a distinct yet interconnected role in model training and evaluation.

A. GuacaMol Dataset as a Training Foundation

The GuacaMol dataset forms the foundational basis for our model's training. It is a comprehensive collection of bioactive molecular entities specifically designed for benchmarking in *de novo* drug design. This dataset has played a significant role in developing and evaluating generative models within the pharmaceutical sector. The key characteristics of the GuacaMol dataset that contribute to its significance in model training include:

- An extensive array of bioactive molecular structures crucial for training models to identify and generate potential drug candidates.
- A broad base for the model to learn from and predict chemical structures, integral for generalisation purposes, particularly in creating compounds with potential therapeutic applications.

The GuacaMol dataset thus equips the model with a comprehensive understanding of the bioactive molecular landscape.

DOI https://doi.org/10.3311/MINISY2024-018

B. COCONUT Dataset in Zero-Shot Learning

The COCONUT (COlleCtion of Open Natural ProdUcTs) dataset is an essential testbed in our zero-shot learning framework. This dataset enables us to assess the model's generalisation capabilities despite training on a distinct chemical structure set. The COCONUT dataset's notable features that make it particularly relevant for our study are:

- A diverse collection of natural product structures thoroughly examines the model's ability to extend its learned patterns to new and structurally varied compounds.
- The dataset includes complex and unique molecular architectures often found in natural products, posing a considerable challenge to the model's ability to modify its pre-learned representations to handle such diversity.

This approach underscores the model's potential in applying its learned knowledge to unfamiliar datasets, underlining its applicability in situations where predictive models are expected to interpret data beyond their initial training parameters.

III. Methods

We are utilising our state-of-the-art pre-trained Transformer-Variational Autoencoder. This model combines the self-attention mechanism inherent in transformers [3] with the generative capabilities of VAEs [4], effectively capturing the complex relationships and variations present in molecular data. The foundation model pre-trained on the GuacaMol dataset provides a nuanced and detailed representation of small molecular structures.

Our approach to analysing the latent space of the Transformer-VAE involves a qualitative assessment, leveraging Uniform Manifold Approximation and Projection (UMAP) [5]. UMAP is renowned for its ability to maintain both local and global structures within data during dimensionality reduction. This technique is instrumental in enabling us to visualise and interpret the organisation of molecular structures within the latent space in a setting devoid of additional training—essentially, a zero-shot learning scenario.

We performed a zero-shot analysis by projecting the COCONUT dataset's natural product structures into the Transformer-VAE's latent space. This method sheds light on how the model, which was not explicitly trained on these compounds, organises and relates to them. We mapped these complex molecules to assess their distribution and proximity to the drug-like molecules of the GuacaMol dataset within the latent space.

Applying UMAP, we aim to offer qualitative insights into the latent space organisation. These insights will illuminate how natural products, with their inherent structural complexity, are interpreted and situated within a space initially optimised for simpler molecular forms. This analysis is expected to underscore the versatility of the pre-trained model in accommodating a diverse range of molecular structures, highlighting its potential role in the discovery and analysis of natural products. Therefore, this study extends our comprehension of how pre-trained models can be adapted in cheminformatics, especially in scenarios that involve the exploration of vast and complex molecular spaces, like those found in natural product chemistry.

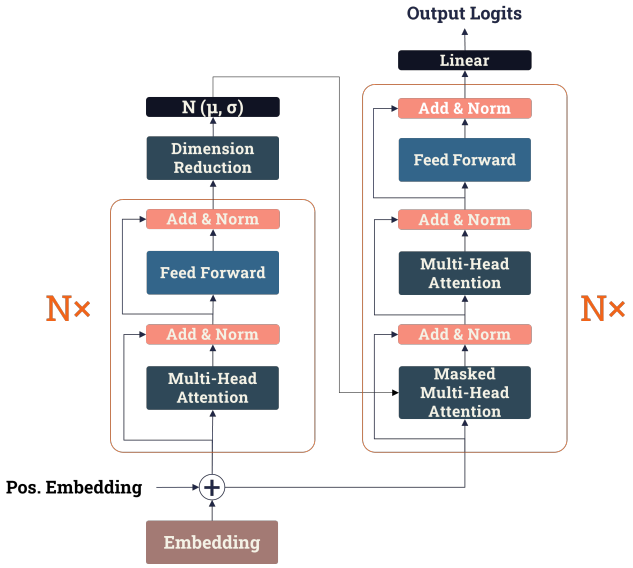


Fig. 1: **Transformer-VAE:** Schematic representation of the architecture. The encoder stack on the left consists of multiple layers, each with multi-head self-attention and position-wise feedforward networks, followed by an add & norm step. The middle section depicts the dimension reduction process to derive the mean μ and standard deviation σ for the latent space representation. The decoder stack, mirrored on the right, includes masked multi-head attention for auto-regressive prediction.

IV. Results

Our qualitative analysis provides a deeper understanding of the Transformer-VAE (T-VAE) model's generalisation capabilities, particularly in a zero-shot learning context. This analysis allows us to delve into how the T-VAE, pre-trained on drug-like molecules from the GuacaMol dataset, interprets and integrates the COCONUT dataset's structurally diverse and complex natural products within its latent space.

To provide context and a baseline for our observations, we include a performance comparison of the T-VAE with other generative models (SMILES LSTM and VAE) on the GuacaMol dataset using standard benchmark metrics [1]. This comparison helps to situate the T-VAE's capabilities within the broader landscape of generative models. Notably, while the T-VAE exhibits a higher score in the Fréchet ChemNet Distance (FCD) metric [6], this indicates its ability to traverse and explore new areas of the chemical space, potentially leading to the discovery of novel molecular structures.

This table and our qualitative analysis contribute to a nuanced understanding of the T-VAE's potential in navigating

Benchmark	SMILES LSTM	VAE	T-VAE
Validity	0.959	0.870	**0.992**
Uniqueness	**1.000**	0.999	**1.000**
Novelty	0.912	0.974	**0.994**
KL divergence	0.991	0.982	**0.997**
FCD	0.913	**0.863**	2.482

TABLE 1 **Benchmarking Generative Models:** This table delineates the performance metrics of SMILES LSTM, VAE, and T-VAE models on the GuacaMol dataset. Key metrics—Validity, Uniqueness, Novelty, KL Divergence, and FCD—are presented, with the most notable scores emphasised in bold. The T-VAE model exhibits exceptional performance in nearly all metrics except FCD, indicating its proficient exploration within novel chemical spaces and affirming its potential for innovative compound discovery.

and representing complex molecular spaces, an essential aspect of advancing cheminformatics and drug discovery.

A. Qualitative Latent Space Analysis

The visualisations presented in Figures 2, 3, and 4 offer critical insights into the generalisation capabilities of the Transformer-VAE model. These figures collectively illustrate the model's proficiency in adapting to the GuacaMol dataset's structures and encompassing the diverse molecular array found in the COCONUT dataset.

In Figure 2, we observe a complex pattern of molecular signatures intertwined within the latent space between the GuacaMol and COCONUT datasets. The discernible overlap where these datasets intersect highlights the model's capacity to distinguish and integrate the diverse characteristics of synthetic and natural molecules. This blending in the latent representation indicates the model's nuanced approach to capturing the essence of molecular structures, which is paramount for identifying compounds with potential therapeutic benefits.

Delving deeper into Figure 3, the visualisation of natural product-likeness [7] across the latent space is striking. Molecules from the GuacaMol dataset that share a high degree of this natural product-likeness appear to cluster near the space occupied by the COCONUT dataset, forming a gradient that spans from synthetic to natural. This gradation underlines the model's refined ability to sort molecules not just by their origin (synthetic or natural) but by their inherent structural and biochemical properties, which indicate their likeliness to be classified as natural products.

Figure 4 further complements our understanding of the model's perception of molecular complexity. It demonstrates the model's application of synthetic accessibility scores [8], revealing a gradation from simpler to more complex synthetic challenges. While the synthetic accessibility (SA) score is not the definitive measure of a molecule's synthesisability, its use in this context is valuable. The score provides an approximation rather than a precise prediction of synthetic feasibility, reflecting practical constraints in drug synthesis. Its integration into the model offers a pragmatic perspective on the feasibility of synthesising new compounds, an essential consideration in predictive cheminformatics.

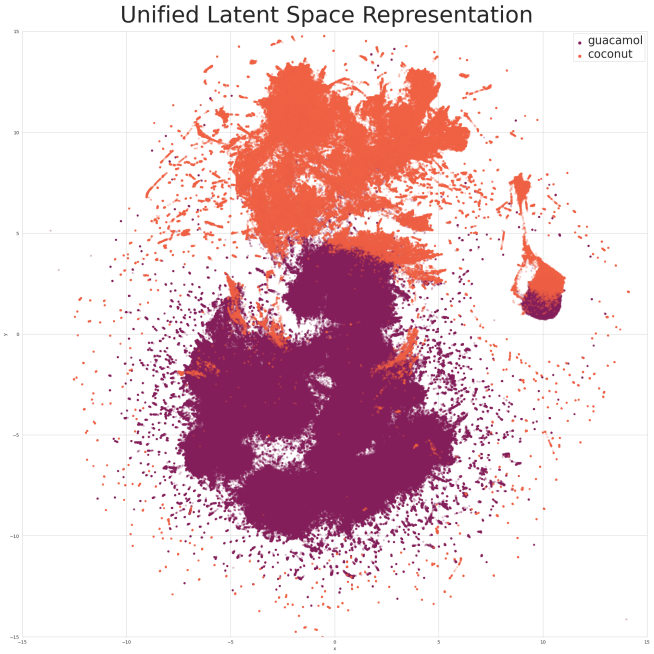


Fig. 2: **UMAP Visualization of Molecular Latent Space:** This plot presents the integrated chemical space of the GuacaMol (in purple) and COCONUT (in orange) datasets. It illustrates a significant interspersion of the two datasets, revealing shared regions where drug-like and natural molecules coexist. The visualisation serves as a comparative analysis tool, highlighting the extent of overlap and the distinct clusters that emerge from the multidimensional scaling, emphasising the model's ability to generalise across diverse molecular spaces.

Taken collectively, the detailed analysis of these visualisations suggests a model adept at encoding and interpreting complex molecular patterns. The Transformer-VAE model is highly competent at navigating the nuanced continuum of molecular diversity, offering promising avenues for discovering novel compounds. Its capacity to simultaneously maintain the distinctive features of synthetic and natural molecules while bridging the gap between these domains showcases its potential as an invaluable tool in advancing drug discovery and the broader field of cheminformatics.

V. Conclusion

Throughout this investigation into the Transformer-VAE (T-VAE) model, our inquiry has been guided by the model's ability to generalise between different molecular datasets. The visual analyses, as provided by Figures 2, 3, and 4, highlight the nuanced capability of the T-VAE to navigate and assimilate the complex structural characteristics of both synthetic molecules and natural products.

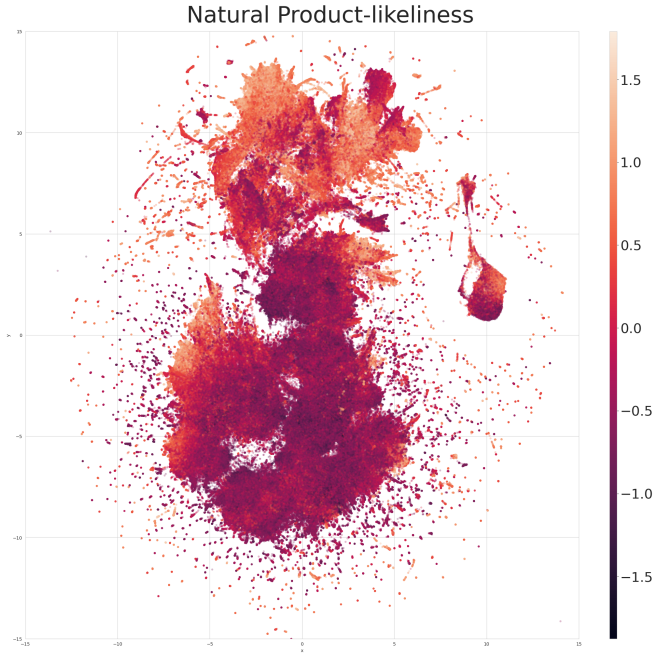


Fig. 3: **Natural Product-Likeness Gradient:** This plot renders the natural product-likeness of compounds on a continuum, as indicated by the colour gradient. It reveals a transition from GuacaMol molecules, typically characterised as more synthetic, to regions densely populated by COCONUT dataset molecules, indicative of compounds with higher natural product-likeness. The gradation in colour from one dataset to the other suggests the model's nuanced understanding of the underlying chemical properties that govern natural product likeness.

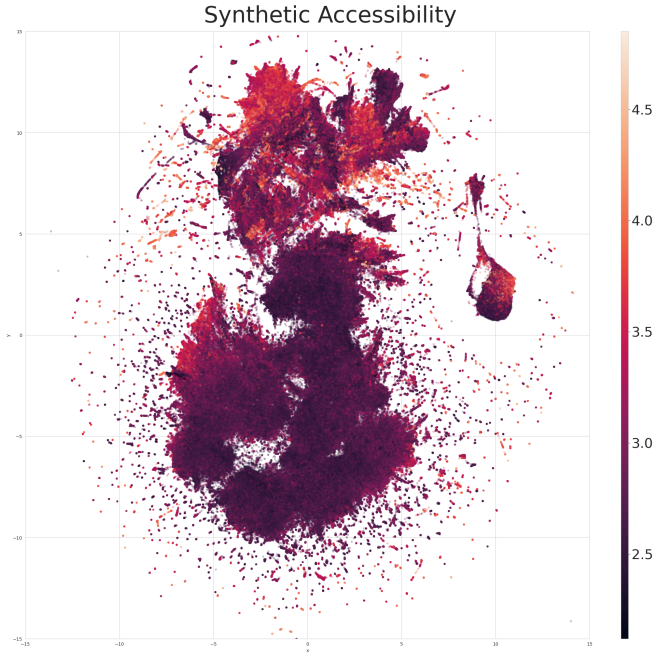


Fig. 4: **Synthetic Accessibility Mapping in Latent Space:** In this UMAP visualisation, the spectrum of synthetic accessibility scores highlights the chemical space's complexity. The colour gradient from GuacaMol to COCONUT datasets correlates with the perceived ease of synthesis, with lighter regions signifying molecules with greater synthetic challenges. This plot provides insight into the model's capability to contextualise molecular structures within the practical constraints of synthetic feasibility.

The observed overlaps and gradients within the latent space suggest a degree of generalisation that transcends simple dataset replication. Instead, the T-VAE appears to identify shared structural motifs and properties across different chemical spaces. While these findings are promising, they are initial observations pointing to generative models' potential in drug discovery.

The subtle transition between molecular datasets regarding natural product-likeness and synthetic accessibility, as depicted in the visualisations, indicates an understanding by the T-VAE of molecular complexity that could be valuable to the pharmaceutical industry. Nevertheless, we approach these results with a measured perspective, acknowledging the need for further validation and exploration to fully ascertain the model's predictive and generative powers.

Our work contributes a preliminary yet insightful piece to the broader cheminformatics puzzle, suggesting how pre-trained models might be leveraged to navigate vast chemical spaces. The implications of such a model's performance in natural product analysis and drug discovery are cautiously optimistic, inviting further research and collaborative efforts to harness the full potential of these computational tools.

In summary, while our findings offer a glimpse into the capabilities of the T-VAE model, we recognise the importance of continued scrutiny and iterative improvement. As computational chemistry advances, we hope that studies such as ours will inspire and inform the development of more sophisticated models, ultimately contributing to the discovery of novel therapeutic agents.

Acknowledgements

Supported by the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory. This research was funded by the National Research, Development, and Innovation Fund of Hungary under Grant TKP2021-EGA-02.

References

[1] N. Brown, M. Fiscato, M. H. Segler, and A. C. Vaucher, "Guacamol: Benchmarking models for de novo molecular design," *Journal of Chemical Information and Modeling*, vol. 59, no. 3, pp. 1096–1108, 2019.

[2] M. Sorokina and C. Steinbeck, "Coconut online: Collection of open natural products database," *Journal of Cheminformatics*, vol. 12, no. 1, p. 2, 2020.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[4] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2014.
[5] L. McInnes, J. Healy, and J. Melville, “Umap: Uniform manifold approximation and projection for dimension reduction,” *arXiv preprint arXiv:1802.03426*, 2018.
[6] K. Preuer, P. Renz, T. Unterthiner, S. Hochreiter, and G. Klambauer, “Frechet chemnet distance: A metric for generative models for molecules in drug discovery,” *Journal of Chemical Information and Modeling*, vol. 58, no. 9, pp. 1736–1741, 2018.
[7] P. Ertl, S. Roggo, and A. Schuffenhauer, “Natural product-likeness score and its applications in the drug discovery process,” *Chemistry Central Journal*, vol. 2, no. Suppl 1, p. S2, 2008. [Online]. Available: https://doi.org/10.1186/1752-153X-2-S1-S2
[8] P. Ertl and A. Schuffenhauer, “Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions,” *Journal of Cheminformatics*, vol. 1, no. 1, p. 8, 2009. [Online]. Available: https://doi.org/10.1186/1758-2946-1-8

Independent Component Analysis based Microphone Array Source Separation

Máté Tóth, Péter Fiala
Budapest University of Technology and Economics
Department of Networked Systems and Services
Budapest, Hungary
Email: toth.mate@edu.bme.hu, fiala@hit.bme.hu

***Abstract*—Microphone Array Source Separation is a well-known topic in the field of Acoustic Signal Processing. In the literature, there are two main approaches to this problem: Blind Source Separation (BSS), and Acoustic Beamforming. Beamforming methods are specifically designed for array signal processing, they achieve source separation by altering the directionality of the microphone array. In contrast to this, BSS methods such as Independent Component Analysis (ICA) only use the statistical independence of the sources without considering the properties of the microphone array (significantly higher number microphones than sources and known geometric structure). This paper investigates the possibilities of optimizing ICA based source separation for microphone array usage. We propose novel solutions to improve the separation performance of ICA based BSS using the array properties, and evaluate the performance of the proposed method on real microphone array recordings.**

***Index Terms*—Blind Source Separation, ICA, Frequency Domain ICA, Beamforming, Array Signal Processing**

I. Introduction

In the field of acoustic signal processing, it is fairly common that we need to separate the different sources of the acoustic environment for further processing, but we cannot measure them separately. In this case, most often microphone arrays are used to record multiple, slightly different mixtures of the sources, then some source source separation method is used to extract the separated signals from the mixtures.

The ”classical” array signal processing approach to microphone array source separation is called beamforming. Beamforming assumes that the geometric properties of the array are known, and it processes the output signals of the different microphones in a way that signals reaching the microphone array at given angles experience constructive interference, while other angles experience destructive interference. This way beamformers can focus on signals that reach the array in a given direction, basically separating the signal of the source that is in a given direction relative to the microphone array. Several beamformer algorithms were proposed in literature, for example the widely used Delay and Sum (DAS) [1] and FROST [2] algorithms. While these ”classical” algorithms can have really good separation performance in certain situations, they have some inherent weaknesses due to their ”directional” nature. The most notable drawbacks are generally poor performance for sources that are at a similar angle and the inability to achieve good separation at higher frequencies that are above the spatial Nyquist frequency, that is given by the spacing of the microphones.

Another source separation approach is Blind Source Separation (BSS). The most widely studied and used BSS method is called Independent Component Analysis (ICA) [3]. ICA is a statistical method for BSS that aims to decompose a mixture of signals into its statistically independent components. This implies that ICA assumes the original sources to be independent (this is almost always the case in practice). While ICA algorithms such as FastICA [4] achieve remarkably convincing separation results when used on signals that are mixed using the linear, instantaneous (delay-free) mixing model, they fail to separate convolutive mixtures, that are naturally created in real acoustic environments. Frequency Domain ICA (FD-ICA) [5] is an extension of a the ICA framework, that works by transforming the time domain mixtures into frequency domain, where the convolutive mixture can be written as a frequency dependent instantaneous mixture for every frequency bin. It has been shown that FD-ICA can be used to effectively separate mixtures recorded by multiple microphones [5].

While the FD-ICA framework proposed by [5] can be used for microphone array source separation without any major modifications, it can only be used for cases where the number of microphones equals the number of sources, therefore we cannot fully take advantage of the high microphone count that is common in microphone array systems, and the geometric properties of the array that are assumed to be known.

This paper proposes a novel source separation method called Beamformer FD-ICA (BF-FD-ICA), loosely based on the FD-ICA framework proposed by [5] that that is optimized for microphone array usage by taking advantage of the aforementioned properties of microphone arrays.

The separation performance of the proposed is solution is evaluated on real-world microphone array recordings, and it is then compared to the performance of a naïve FD-ICA implementation, and the classical DAS beamformer algorithm.

II. Preliminaries

A. Independent Component Analysis

Let $\boldsymbol{\sigma} = [\sigma_1, \sigma_2, \ldots, \sigma_N]^{\mathrm{T}}$ be a random vector variable, where $\sigma_1, \sigma_2, \ldots, \sigma_N$ are independent and nongaussian, and their observations represent the discrete samples of the latent sources. Let $\boldsymbol{\xi} = [\xi_1, \xi_2, \ldots, \xi_N]^{\mathrm{T}}$ be another random variable

DOI https://doi.org/10.3311/MINISY2024-019

that represents the measured mixed signals, and $\mathbf{A}$ be an unknown mixing matrix. The linear, instantaneous mixing model is then given by:

$$\boldsymbol{\xi} = \mathbf{A}\boldsymbol{\sigma} \tag{1}$$

The task of BSS is approximating a $\mathbf{W}$ demixing matrix, such that:

$$\boldsymbol{\sigma} \approx \boldsymbol{\upsilon} = \mathbf{W}\boldsymbol{\xi} = \mathbf{W}\mathbf{A}\boldsymbol{\sigma} \tag{2}$$

Since $\mathbf{A}$ and $\boldsymbol{\sigma}$ are both unknown, directly approximating $\mathbf{W}$ is not possible. The main idea of ICA is to take advantage of the independence of the components of $\boldsymbol{\sigma}$. It can be shown that finding a $\mathbf{W}$ that maximizes the independence of the separated components guarantees that the separated signals approximate the original signals up to a scaling and permutation [6]. There are several different approaches to maximize independence, for example nongaussianity, mutual information, maximum-likelihood estimate or tensorial methods [6]. In this paper, we will consider nongaussianity based ICA.

The Lindeberg-Lévy form of Central Limit Theorem (CLT) states that the distribution of the sum of independent nongaussian random variables approaches to the normal distribution as we increase the number of terms [7]. This means that independent components can be extracted form the $\boldsymbol{\xi}$ mixture by finding $\mathbf{w_i}$ demix vectors, such that:

$$\mathbf{w_i} = \operatorname{argmax} G(\mathbf{w}^{\mathrm{T}}\boldsymbol{\xi}) \tag{3}$$

Where G is a contrast function that measures nongaussianity. It is important to note, that we have to make sure that the $\mathbf{w_i}$ demix vectors do correspond to different independent components. This can be done by using a whitening transform (eg. PCA) as a preprocessing step and orthogonalizing the $\mathbf{w_i}$ vectors.

There are several ways to measure nongaussianity, most notably kurtosis and entropy based methods. The differential entropy, that basically measures the uncertainty of a random continuous variable ξ is defined by the following:

$$H(\xi) = \mathbb{E}[-\log(p_\xi(\xi))] \tag{4}$$

It has been shown that the normal distribution has the highest differential entropy for a given variance [8]. Define negentropy in the following form:

$$J(\xi) = H(\xi_{\mathrm{gauss}}) - H(\xi) \tag{5}$$

Where $J(\xi)$ is the negentropy, and ξ_{gauss} is a Gaussian random variable, that has the same variance as ξ. The nongaussianity is then maximized by maximizing $J(\xi)$.

In practical applications, the PDF of ξ is not known exactly, therefore G can be defined as the sample mean of the x_i observations of ξ, transformed by a predefined f nonlinearity that approximates $-\log(p_\xi(x))$:

$$G(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} f(x_i) \tag{6}$$

Several nongaussianity based ICA algorithms have been proposed that differ by the optimization method, and the f contrast function, most notably FastICA [9] that uses an approximate Newton optimization method and $f(x) = \log\cosh(x)$ as a contrast function.

Since ICA estimates the demix matrix and the original sources together, it has some inherent ambiguities. Let $\mathbf{D}$ be a diagonal matrix whose diagonal elements are nonzero, and let $\mathbf{P}$ be a permutation matrix. Then the following equation holds:

$$\tilde{\boldsymbol{\xi}} = \mathbf{A}\tilde{\boldsymbol{\sigma}} = \mathbf{APD}\boldsymbol{\sigma} = \tilde{\mathbf{A}}\boldsymbol{\sigma} \tag{7}$$

Where $\tilde{\boldsymbol{\sigma}} = \mathbf{PD}\boldsymbol{\sigma}$ and $\tilde{\mathbf{A}} = \mathbf{APD}$. Notice, that if the components of $\boldsymbol{\sigma}$ were independent, then the components of $\tilde{\boldsymbol{\sigma}}$ will be independent too. This means that ICA methods cannot recover the original permutation, and the original scale of the sources.

B. Frequency Domain ICA

In real acoustic environments the speed of sound is finite, so the mixtures recorded by real microphones are very rarely instantaneous, since there might be time delays (because the distance of a given source is different for the different microphones), different reflections, and frequency dependent effects between the signal of the same source recorded by the different microphones. The mixtures created by this wave propagation can be modeled as convolutive mixtures. Convolutive mixtures can be written in the following form, using FIR matrix algebra [10]:

$$\mathbf{x}[n] = \underline{\mathbf{H}} * \mathbf{s}[n] \tag{8}$$

Where $\underline{\mathbf{H}}$ is the matrix of the discrete impulse responses between the source and the microphone positions, $\mathbf{s}[n]$, $\mathbf{x}[n]$ are the original and the mixed signal vectors at the nth dicrete timestep, and $*$ is the discrete convolution operator.

The main idea of Frequency Domain ICA is using the convolution theorem of the Discrete Fourier Transform (DFT) to transform the convolutive mixture into multiple the frequency dependent instantaneous mixtures that can be separated using "traditional" ICA. It can be shown that:

$$\mathbf{x}[n] = \underline{\mathbf{H}} \circledast_F \mathbf{s}[n] \Longleftrightarrow \mathbf{x_f} = \mathbf{A_f}\mathbf{s_f} \qquad \forall \quad 0 \leq f < F \tag{9}$$

Where $\circledast_F$ is the F length circular convolution, $\mathbf{s_f}$ and $\mathbf{x_f}$ are vectors of the Discrete Fourier Transform coefficients corresponding to the fth frequency bin of the STFT of the $\mathbf{s}$ original and $\mathbf{x}$ mixed signals, $\mathbf{A_f}$ is the frequency dependent mixing matrix, and F is the STFT window length. It has been shown that using sufficiently large F, $\circledast_F$ approximates $*$ well enough.

Taking advantage of this, a naïve FD-ICA algorithm can be implemented the following way:

1) Transform the time domain microphone input signals to time-frequency domain using Short Time Fourier Transform (STFT).
2) Use ICA for every STFT frequency bin separately to extract the independent components.
3) Transform the separated signals back to time domain using inverse STFT.

III. Challenges of FD-ICA

While FD-ICA provides a theoretical framework for convolutive BSS, there are three main challenges of implementing a practically usable algorithm based on it:

1) As we transform real valued signals to frequency domain using DFT (or STFT), they become complex valued, meanwhile traditional ICA algorithms operate on real values.
2) The scaling ambiguity of ICA algorithms mean that separating the signals in each bin independently can lead to severe spectral distortion of the separated signals (basically we are "randomly EQ-ing" the signals).
3) The permutation ambiguity of ICA means, that although we can separate the signals at every frequency bin, we do not know which separated signal components belong to the same components across the frequency bins.

A. Complex ICA

Several methods have been proposed for complex valued ICA, eg. the tensorial JADE [11] algorithm, complex kutrosis based RobustICA [12] algorithm or the extension of FastICA, called Complex FastICA [4]. For our FD-ICA implementation, we have chosen the Adaptable Complex Maximization of Non-gaussianity (A-CMN) algorithm [13] that to our knowledge have not been used for FD-ICA to date. A-CMN is derived from the family of CMN algorithms [14]. These algorithms use an approximate Newton method for optimizing a complex contrast function, that approximates the negentropy of its input. CMN algorithms differ by the choice of this contrast function. A-CMN is considered the the most advanced CMN method, since it uses an adaptive Exponential Power Family [15] contrast function, whose parameters are optimized iteratively using a Maximum-likelihood estimate. The reason for choosing A-CMN is simply because our experiments showed that it outperforms the aforementioned other methods most of the time.

B. Scale Ambiguity

To solve the scaling problem, we opted to to choose a trivial multichannel extension of a method called "Mapping to the observation space", originally proposed in [5]. This method works by calculating the estimated mixing coefficients for each source in each frequency bin, and using them to cancel out the unknown scaling factor. The main advantage of this method is that it is proven that it "perfectly" cancel out the arbitrary scaling factors. (However it has a side effect, that it transforms the separated signals the same way as the microphones would record it independently.)

C. Permutation Ambiguity

The most difficult challenge of implementing a robust FD-ICA algorithm is the permutation ambiguity. Although several methods were proposed to solve permutation, they only achieved limited success without using any additional information. One of the most promising methods is the so-called Reduced likelihood ratio jump (RLRJ) proposed in [16]. This is an iterative a source modeling based approach, that uses a time-frequency domain source representation. The core idea of the method is to use that the time dependent energy envelope of "real" acoustic signals at a given frequency is usually not too dissimilar from the overall energy envelope of the signal.

IV. Optimizing FD-ICA for Microphone Arrays

A. Using Additional Microphones

"Classical" FD-ICA (and ICA) based source separation assumes that the number of microphones (M) equals the number of sources (S). In microphone array signal processing practice however, it is common that M is significantly larger than S. A trivial solution to this is simply using the signal of S out of the M microphones. Although this method technically works, we are not using the additional information provided by the other microphones. Moreover this method needs the number of sources to be known in advance which in not always the case. The main idea of our proposed solution is to decompose the observation space (the space of the signals recorded by all microphones) to a signal and a noise subspace (similarly, how the MUSIC algorithm works [17]) and then project the observations to the signal subspace. To achieve this, we are using Principal Component Analysis (PCA) as a preprocessing step for ICA in every frequency bin. Let $\mathbf{X_f} \in \mathbb{C}^{M \times T}$ be the matrix whose rows are the vectors corresponding to the fth frequency bin in the STFT transforms of the microphone signals (T is the number of the time segments of the STFT). The complex covariance matrix $\mathbf{\Sigma}$ is given by:

$$\mathbf{\Sigma} = \mathbf{X_f} \mathbf{X_f}^H \tag{10}$$

Let λ_i be the ith largest eigenvalue of $\mathbf{\Sigma}$ and let $\mathbf{v_i}$ be the corresponding eigenvector. It is obvious that the first S eigenvalues will be significantly larger than the other eigenvalues, since the "real" sources are considered to be louder than the noise sources, so the variances will be significantly bigger in the directions of the real sources in the observation space. Define $\mathbf{Q} = [\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_S}]$ and $\mathbf{\Lambda} = \mathrm{Diag}\,(\lambda_1 \lambda_2 \ldots, \lambda_S)$. The dimensionally reduction PCA (D-PCA) transform that projects to the signal subspace, and conveniently whitens the signals is then given by:

$$\mathbf{V} = \mathbf{\Lambda}^{-\frac{1}{2}} \mathbf{Q} \tag{11}$$

B. Using known Array Geometry for Permutation Alignment

The geometric properties of a microphone array such as microphone distances are usually known in advance. A few methods have been proposed to take advantage of the known microphone positions in the FD-ICA framework [5] [18], most of these aim to solve the permutation ambiguity by calculating the directionality of the array. The main idea of these methods is that it has been shown that the demix matrices estimated by the complex ICA algorithm together with the known microphone spacing (d) can be used to calculate the sensitivity of the array as a function of the Direction Of Arrival (DOA) at every f frequency bin. For this, the "traditional" one

dimensional equidistant linear sensor array sensitivity equation [19] can be used, that uses complex sinusoidal signals, that are phase shifted corresponding to the phase difference of the microphones for a plane wave reaching the array at a θ angle. This states the following:

$$F\left(f, \mathbf{w^f}, \theta\right) = \sum_{m=1}^{M} \tilde{w}_m^f \exp\left(j2\pi f(m-1)d\frac{\sin(\theta)}{c}\right) \quad (12)$$

Where F is the sensitivity, $\mathbf{w^f}$ is a row vector of the $\mathbf{W^f}$ demix matrix θ is the DOA, c is the speed of sound and $\tilde{w}_m^f$ is the normalized complex weight of the mth microphone at the fth frequency bin:

$$\tilde{w}_m^f = \frac{w_m^f}{|w_m^f|} \quad (13)$$

The aforementioned methods use these directivity patterns for permutation alignment: since the ICA separates exactly one source from the mixture, the sensitivity of the array should be consistently near zero at DOAs of the other sources at every frequency bin (it is basically a null beamformer), and these null positions should be the same for every bin for a given source. Our experiments showed that for our microphone array case, where it is assumed that $M \gg S$ and we use D-PCA preprocessing, the absolute value of the directivity patterns tend to be high near the DOA of the separated source and zero everywhere else (until we reach the spatial Nyquist frequency of the array and encounter aliasing effects). Figure 1 shows example directivity patterns as a function of θ (x axis) and F (y axis) Our approach uses this to solve permutation alignment:

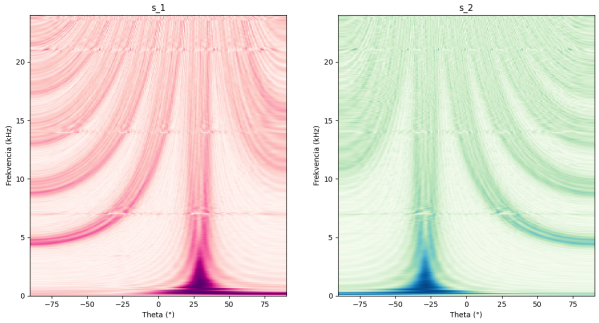


Fig. 1. Directivity patterns of FD-ICA

1) We firstly use the MUSIC algorithm [17] to determine the DOAs of the sources, let the DOA vector be $\boldsymbol{\theta} = [\theta_1, \theta_2, \dots \theta_S]$
2) We approximate the integral of the absolute value of the directivity for each θ_s DOA at every f frequency bin:

$$F_{k\theta_s}^f = \int_{\theta min}^{\theta max} F\left(f, \mathbf{w_k^f}, \theta\right)\, d\theta \quad (14)$$

$$\theta_{min} = \max\left(-\frac{\pi}{2}, \theta_s - \min\left(\frac{\Delta\theta}{2}, \Delta\theta_0\right)\right) \quad (15)$$

$$\theta_{max} = \min\left(\frac{\pi}{2}, \theta_s + \min\left(\frac{\Delta\theta}{2}, \Delta\theta_0\right)\right), \quad (16)$$

Where θ_0 is a given maximal DOA "radius".
3) We construct the $\mathbf{F^f}$ adjacency matrix for every frequency bin, that basically represents how sensitive the microphone array is at each θ_i DOA for each s source:

$$\mathbf{F^f} = \begin{bmatrix} F_{1\theta_1}^f & F_{1\theta_2}^f & \cdots & F_{1\theta_S}^f \\ F_{2\theta_1}^f & F_{2\theta_2}^f & \cdots & F_{2\theta_S}^f \\ \vdots & \vdots & \ddots & \vdots \\ F_{S\theta_1}^f & F_{S\theta_2}^f & \cdots & F_{S\theta_S}^f \end{bmatrix} \quad (17)$$

4) Finding the optimal permutation is then equivalent to finding a maximal weight perfect matching in a graph represented by the $\mathbf{F^f}$ adjacency matrix. This is called a linear sum assignment problem, that can be solved using the Hungarian method [20].
5) The solution of the linear sum assignment problem is given by a subset of the graph edges called the optimal matching. Let $v_i = [s_k, \theta_l]$ $(1 \leq i, k, l \leq S)$ be an edge of the optimal matching. The optimal permutation in the given frequency bin is given by moving the kth source to the l index for all i edges.

Notice that this 1D formulation only considers the azimuth angles of the sources, therefore it is only adequate for "planar" configurations, although our solution can be easily extended to arbitrary 3D configurations using a planar microphone array.

We use this permutation alignment method to create an initial permutation that is further refined with RLRJ. We do this because this method can fail at higher frequencies (due to aliasing), but those permutation errors are mostly corrected by the RLRJ pass applied after it, and RLRJ itself tends to converge to significantly better solutions using this initialization compared to "random" initialization.

V. BF-FD-ICA Architecture

The high level architecture of our proposed final solution, BF-FD-ICA can be seen on figure 2. The components highlighted in red are our specific modifications for microphone array usage. D-PCA blocks are described in subsection IV-A, C-ICA blocks represent the used complex ICA algorithm described in subsection III-A, Scale blocks are described in subsection III-B, the BF-perm block represents our permutation alignment method described in subsection IV-B and RLRJ represents the RLRJ algorithm described in subsection III-C.

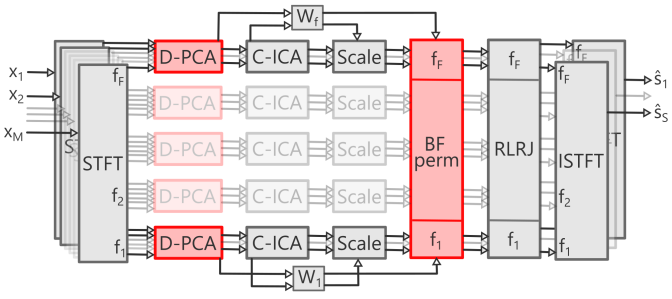


Fig. 2. The BF-FD-ICA architecture

VI. Evaluation Methodology

For the evaluation and comparison of our BF-FD-ICA method, we used real microphone array measurements. We decided to use real measurements instead of simulated mixtures, because the simulation framework we have access to currently only supports simulating infinite open space sound propagation with "perfect" sources and microphones, that does not really correspond to real acoustic environments.

A. Measurement Arrangement

For the sources we used 5 *Genelec 1030A* studio monitors in a linear alignment using regular $D = 0.7$ m source distance, and for the microphone array we used a *Gfai mcdRec* data recorder with 24 omnidirectional microphones in a linear configuration, parallel to the speaker array, using $d = 0.05$ m microphone distance (the length of the array is then $L = 1.15$ m). We placed the microphones and the speakers on the ground in an arrangement where microphones and the speakers lie on the same plane, and the center of the microphone array and the edge speakers form an equilateral triangle. We conducted the measurements in a near anechoic room with minimal reflections. For the measurements signals, we used spectrally complex electronic music excerpts. We also did measurements in a controlled reflective environment. For this, we placed a 2×2 m sized wooden plate behind the microphones, parallel to the microphone array, to around $R = 2$ m distance from it.

B. Evaluation Metrics

For the quantitative evaluation of the separation performance, we recorded the the sources independently (by playing one signal at a time on the same speaker that is used to play that signal for the measurements), and we calculated the frequency dependent error of the separated sources compared to the independently recorded reference signals. We calculated the separation error (denoted as SNR) using the following methodology:

1) We normalized and aligned the separated signals to the reference signals in time domain.
2) We calculated the spectograms of the normalized, aligned signals ($\mathbf{\hat{sn}_i}$) and the reference signals ($\mathbf{sn_i^{ref}}$):

$$\mathbf{S_i} = |\mathrm{STFT}(\mathbf{\hat{sn}_i})| \qquad \mathbf{S_i^{ref}} = \left|\mathrm{STFT}(\mathbf{sn_i^{ref}})\right| \quad (18)$$

3) We calculated the time-frequency domain signal and noise components of the separated signals:

$$\mathbf{S_i^{err}} = \left|\mathbf{S_i^{ref}} - \mathbf{S_i}\right| \qquad \mathbf{S_i^{sig}} = \mathbf{S_i} - \mathbf{S_i^{err}} \quad (19)$$

4) We calculated the Root Mean Square (RMS) values of the signal and noise components.

$$err_i[f] = \left[\sum_{t=1}^{T} S_i^{err}[f,t]^2\right]^{1/2} \quad (20)$$

$$sig_i[f] = \left[\sum_{t=1}^{T} S_i^{sig}[f,t]^2\right]^{1/2} \quad (21)$$

5) Finally we calculated the frequency dependent RMS errors for every separated source:

$$SNR_{i_{dB}}[f] = 20\log_{10}\left(\frac{sig_i[f]}{err_i[f]}\right) \quad (22)$$

VII. Results

For evaluation we calculated the frequency dependent relative error of the separation (using VI-B) for each of the 5 sources (faint lines on the figures), and averaged them ("normal" lines on the figures). Since we used RMS normalized measurement signals the "baseline" SNR, without any separation was $\approx 1/4 = -6$ dB (the signal of the other 4 sources is considered as noise for a given source), thus we refer to the separation performance as the improvement from this value. Where it is not specifically mentioned, we used the whole 24 channel input.

A. BF-FD-ICA vs FD-ICA

For this measurement, we used our BF-FD-ICA and a "classical" FD-ICA implementation that we obtained by removing our array specific improvements. For FD-ICA, we used 5 microphones out of the 24. Figure 3 shows the results of this measurement. While both methods achieved good separation performance, especially at higher frequencies, our optimized method outperformed FD-ICA by $6-8$ dB, and it achieved a 20 dB separation in a very wide 1000 Hz - 15000 Hz frequency range. At lower frequencies the performance started to decline, but we still achieved a respectable 12 dB separation at 500 Hz. We suspect that the main limiting factor of the low frequency separation is the array length, since these higher wavelength signals reach the microphones in almost the same phase.

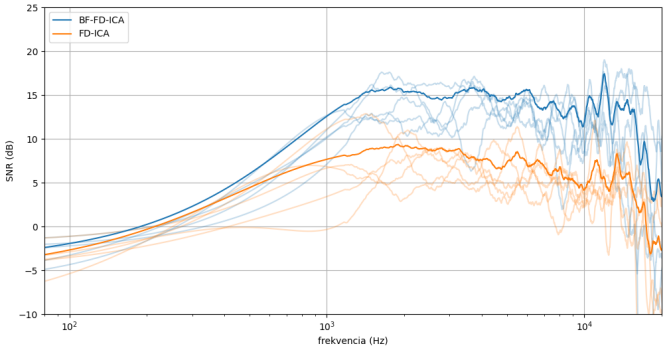


Fig. 3. Separation performance of BF-FD-ICA compared to FD-ICA

B. BF-FD-ICA vs Traditional Beamformers

We used the official Matlab implementation of the DAS beamformer for this measurement with ground truth DOA information, Figure 3 shows the results. Our method significantly outperformed the DAS beamformer especially above the spatial Nyquist frequency of the array (≈ 3400 Hz), where traditional beamformers always struggle. At lower frequencies DAS is more competitive, but our method still outperforms it by around $8-10$ dB.

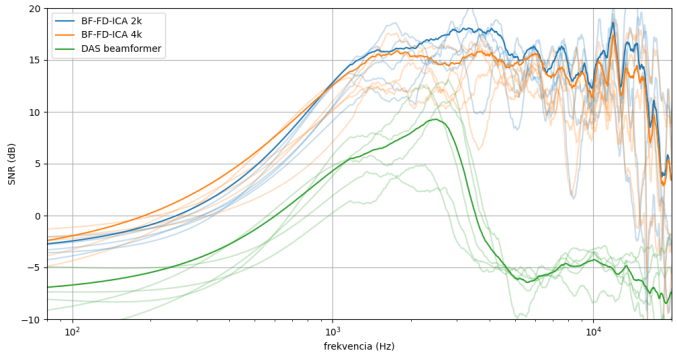


Fig. 4. Separation performance of BF-FD-ICA compared to the DAS beamformer

C. BF-FD-ICA vs Traditional Beamformers in Reflective Environment

For this experiment, we used the reflective configuration. We tried 2048 and 4096 STFT length for BF-FD-ICA, since it can have a significant effect with longer reverbation times. Figure 5 shows the results. In an environment with a strong reflection, the separation performance of BF-FD-ICA decreased by around $5-10$ dB, mainly at higher frequencies, but we still achieved more than 10 dB separation above 1000 Hz. The reason for this might be the more complex transfer characteristics between the sources and the microphones, that the algorithm has to estimate. Compared to the DAS beamformer the advantage of our method decreased to around 3 dB at lower frequencies, but overall our method still performs favorably, especially at higher frequencies.

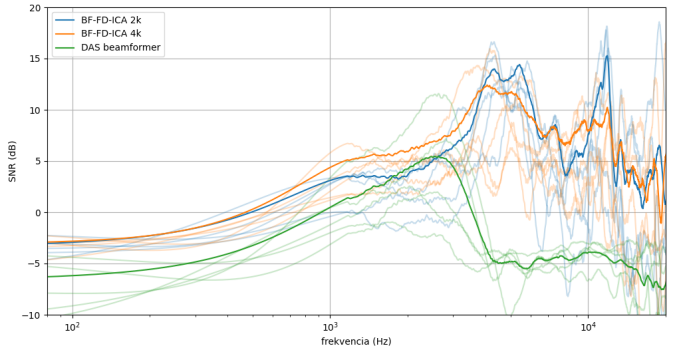


Fig. 5. Separation performance of BF-FD-ICA compared DAS beamformer in reflective environment

VIII. Conclusion

In this paper we proposed a novel solution for microphone array beamforming by improving on the traditional FD-ICA methods. We proposed a novel dimension reduction based solution to take advantage of the redundancy caused by the additional microphones, and a beamforming based solution for the permutation problem, that uses the known geometry of the microphone array. Our experiments showed that the separation performance of the proposed BF-FD-ICA architecture significantly surpasses the performance of the traditional frequency domain ICA. Our method also compares favorably to classical beamforming, since it reliably works at frequencies higher than the spatial Nyquist frequency of the array, where those methods fail.

Acknowledgements

This work has been supported by the Hungarian National Research, Development and Innovation Office under contract No. K–143436.

References

[1] R. Mucci, "A comparison of efficient beamforming algorithms," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 3, pp. 548–558, 1984.

[2] O. L. Frost, "An algorithm for linearly constrained adaptive array processing," *Proceedings of the IEEE*, vol. 60, no. 8, pp. 926–935, 1972.

[3] P. Comon, "Independent component analysis, a new concept?" *Signal Processing*, vol. 36, no. 3, pp. 287–314, 1994, higher Order Statistics. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0165168494900299

[4] E. Bingham and A. Hyvärinen, "A fast fixed-point algorithm for independent component analysis of complex valued signals," *International journal of neural systems*, vol. 10, pp. 1–8, 03 2000.

[5] N. Mitianoudis, "Audio source separation using independent component analysis," Ph.D. dissertation, University of London, 2004.

[6] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*, 06 2001, vol. 26.

[7] W. Kramer, "Probability & Measure : Patrick Billingsley (1995): (3rd ed.). New York : Wiley, ISBN 0-471-0071-02, pp 593, [pound sign] 49.95," *Computational Statistics & Data Analysis*, vol. 20, no. 6, pp. 702–703, December 1995. [Online]. Available: https://ideas.repec.org/a/eee/csdana/v20y1995i6p703-702.html

[8] T. M. Cover, *Elements of information theory*. John Wiley & Sons, 1999.

[9] A. Hyvarinen, "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Transactions on Neural Networks*, vol. 10, no. 3, pp. 626–634, 1999.

[10] R. H. Lambert, *Multichannel blind deconvolution: FIR matrix algebra and separation of multipath mixtures*. University of Southern California, 1996.

[11] D. N. Rutledge and D. J.-R. Bouveresse, "Independent components analysis with the jade algorithm," *TrAC Trends in Analytical Chemistry*, vol. 50, pp. 22–32, 2013.

[12] V. Zarzoso and P. Comon, "Robust independent component analysis by iterative maximization of the kurtosis contrast with algebraic optimal step size," *IEEE Transactions on neural networks*, vol. 21, no. 2, pp. 248–261, 2009.

[13] M. Novey and T. Adali, "Adaptable nonlinearity for complex maximization of nongaussianity and a fixed-point algorithm," in *2006 16th IEEE Signal Processing Society Workshop on Machine Learning for Signal Processing*. IEEE, 2006, pp. 79–84.

[14] ——, "Complex ica by negentropy maximization," *IEEE Transactions on Neural Networks*, vol. 19, no. 4, pp. 596–609, 2008.

[15] G. Lunetta, "Di una generalizzazione dello schema della curva normale," *Annali della Facolta di Economia e Commercio di Palermo*, vol. 17, no. 2, pp. 237–244, 1963.

[16] D. Mallis, T. Sgouros, and N. Mitianoudis, "Convolutive audio source separation using robust ica and an intelligent evolving permutation ambiguity solution," *Evolving Systems*, vol. 9, pp. 315–329, 2018.

[17] M. H. Hayes, *Statistical digital signal processing and modeling*. John Wiley & Sons, 1996.

[18] H. Saruwatari, S. Kurita, K. Takeda, F. Itakura, T. Nishikawa, and K. Shikano, "Blind source separation combining independent component analysis and beamforming," *EURASIP Journal on Advances in Signal Processing*, vol. 2003, pp. 1–12, 2003.

[19] D. H. Johnson, "Array signal processing," *concepts and techniques*, 1993.

[20] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.