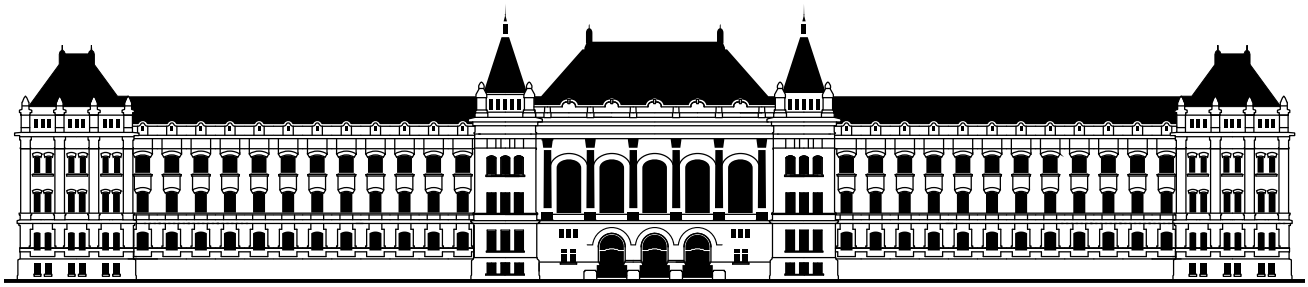
PROCEEDINGS OF THE 28TH MINISYMPOSIUM

OF THE

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

(MINISY@DMIS 2021)

FEBRUARY 1–2, 2021
BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS



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

ISBN 978-963-421-845-6

Head of the Department:
Tamás Dabóczi

General Chair:
Balázs Renczes

Scientific Chairs:
Ákos Jobbágy
András Pataricza
Tadeusz Dobrowiecki

Local Chair:
Bence Graics

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

Sponsored by:
Schnell László Foundation

FOREWORD

On behalf of the Organizing Committee, I welcome you to the 28th Minisymposium of the Department of Measurement and Information Systems at the Budapest University of Technology and Economics.

It is a pleasure to see that besides the Ph.D. students of the Department, we can also welcome our international scientific partners and our talented Bachelor and Master students. The Symposium will be a great possibility to present and discuss their scientific results and get feedback on improving their work.

The Symposium's scope covers the key research areas of the Department: from measurement theory and digital signal processing, through artificial intelligence and bioinformatics to cyber-physical systems, dependability and security.

Like the last years' practice and conforming to the international trends, the proceedings will be published only in electronic form. We have experienced that the easy accessibility of the digital version makes it insufficient to publish a printed edition. We hope that the advantages of the electronic form will dominate any emerging inconvenience.

Due to the pandemic, for the first time, the Symposium will be held online. By this means, we lose the opportunity for personal discussions during the breaks. However, the online version's benefit is that it makes it possible both for the presenters and for the audience to join the Symposium worldwide.

We wish that the forthcoming one and a half days will not only be fruitful in a sense that we will be able to gain insight into the research of one another, but it will also be a time for collecting ideas for future research, as well as for finding possible interdisciplinary cooperation areas.

Budapest, February 2021

Balázs Renczes
General Chair

PAPERS OF THE MINISYMPOSIUM

Detection of facial microexpressions

Gabor Revy
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
revy.gabor@gmail.com

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

Daniel Hadhazi
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
hadhazi@mit.bme.hu

***Abstract*—Facial microexpressions are instantaneous features signaling various details regarding the emotional and mental state of human beings. A key property of such features is that their interpretation as signals is the same or closely similar for all people. Currently, their detection requires a human expert. The automation of this task would allow a more widespread use. In this paper, we propose a hybrid solution, which is based on a framework of landmark points identified by a machine learning-based method. Upon this, we designed an expert system which utilizes image processing and signal processing algorithms such as homomorphic filtering, RANSAC parabola fitting, Hessian based shape analysis and change detection in order to identify microexpression features such as gaze detection and eyebrow raising. We evaluate these algorithms in real videos and pictures, and examine their applicability in practical scenarios. Our long-term goal is to detect complex facial expressions and emotions with the help of the detected microexpressions.**

***Index Terms*—microexpression, image processing, landmark points, expert system, facial expressions**

I. Introduction

Microexpressions are the visible features of emotions appearing on the face for a very short time, e.g. an involuntary reaction to a question. Automating the detection of facial expressions would allow a wide range of uses, e.g. to study reactions to an advertisement or to assist in the diagnosis of mental disorders. Experts usually distinguish 7 different basic emotions: anger, disgust, fear, happiness, sadness, surprise and contempt. In order to categorize reactions in videos, proper detection of microexpressions is required. In this paper, we propose algorithms to detect some of them. The first step is to detect the motion of the muscles, the so-called action units on the face. These parts of the face are described in detail in the FACS system [1]. Emotions can be determined based on the activated action units. The proposed methods estimate gaze direction, detect blinking and eyebrow raising by utilizing the output of an open-source landmark detection method [3].

This research was supported by the ÚNKP-20-5-BME-92 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund, and the János Bolyai Research Scholarship.

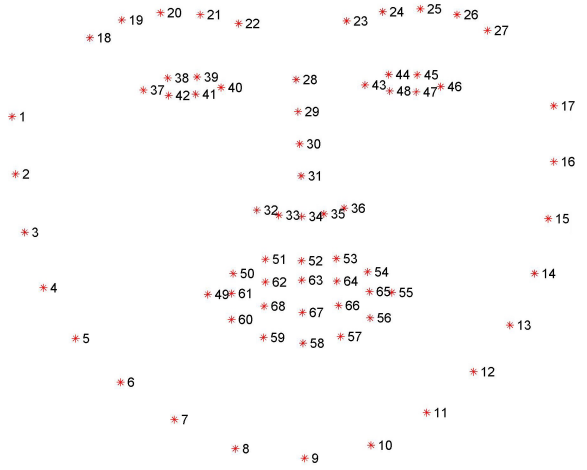


Fig. 1: Layout of the landmark points

A. Detection: machine learning vs expert system approach

There are two main approaches to detect microexpressions: machine learning based and expert image processing based systems. Machine learning based solutions can be robust if the training data is properly annotated and has adequate diversity. However, in the current scenario such a dataset is not available, and the available emotion detection models are not accurate enough. In an expert systems, several image processing algorithms are utilized to make a prediction from an image or a video. Such methods require a lot of fine tuning, most of which are arbitrary. We propose a hybrid solution: using a pre-trained, machine learning-based system, landmark points are established as shown in Figure 1. Based on the landmark points, image processing methods are applied to detect facial features and muscle movements.

II. Gaze detection

The main parts of estimating gaze detection are the localization of the eye and the iris. It also includes blink detection which can be an essential feature in the identification

of surprise, disgust or fear. The proposed solution for gaze detection was based on an open source gaze estimator called EyeTab [8] which was modified in order to increase robustness to changes in illuminations, to the variance in the visible size of the eyes.

A. Eye localization

In the first step, the regions of interest (ROIs) corresponding to eyes are determined. This was based on OpenCV Haar-like feature based cascade classifiers [7] in the EyeTab method. However, our initial analysis indicated that the ROIs defined by the bounding box of eye specific landmark points, provide more accurate results. Thus we modified EyeTab to utilize the polynom defined by the eye landmark points.

B. Pupil detection

The pupil is detected by combining two algorithms: the first is a gradient based [5] while the second is an isophote based [6] approach. In addition, the original input (the red channel of the RGB image) is replaced with the homomorphic filtered grayscale image, which resulted in more accurate pupil predictions.

1) Homomorphic filter: Homomorphic filtering can be used to compensate inadequate lighting, for example a shaded eye socket. Only the energy of low frequency components are reduced, therefore fast intensity changes (e.g. wrinkles) are preserved. It can be formulated as follows:

$$\begin{aligned}\mathbf{Im}_{homomorphic} &= T(\exp(\log(\mathbf{Im}) - \log(\mathbf{Im}) * \mathbf{G}_\sigma)) \\ T(x) &= \min(x, 1)\end{aligned}$$

Here, $\mathbf{G}_\sigma$ is the 2 dimensional, isotropic Gaussian function with σ parameter and $*$ means convolution. T cuts the values under 0 and over 1.

C. Blink detection

The pupil detection algorithm proposes a region of interest not only when the eyes are open, but also when the eyes are closed or are blinking. In order to detect blinking, a local Hessian based image analysis is applied to distinguish true and false ROIs proposed by the pupil detection method.

A typical pupil in a grayscale intensity image, after utilizing homomorphic luminance compensation method, is a round region, which is darker compared to its neighboring pixels. These regions can be highlighted by local Hessian based filtering method, which is defined by:

$$\mathbf{H}_\sigma = \alpha(\sigma) \cdot \nabla^2(\mathbf{I} * \mathbf{G}_\sigma)$$

where ∇ is the gradient operator and $*$ denotes the operator of the convolution. $\alpha(\sigma)$ is a normalization scalar compensating the multiplicative dependency of the norm of the matrix on the σ parameter. Based on the scale space theory [4] $\alpha(\sigma) = \sigma^2$. The value of σ depends on the size of the blob which is proposed as a pupil candidate, which is determined by the solution of the optimization problem:

$$\sigma(r) = \arg\max_\sigma \Big\{ (\mathbf{D}_r * (\alpha(\sigma) \cdot \mathbf{G}_\sigma))(0,0) \Big\} = r/\sqrt{2}$$

where r denotes the radius of the blob, $\mathbf{D}_r$ denotes the binary image of the $(0,0)$ centered homogeneous sphere with radius r:

$$\mathbf{D}_r(x,y) = \begin{cases} 1, & \text{if } \Big\|[x\ y]\Big\|^2 \leqslant r \\ 0, & \text{if } \Big\|[x\ y]\Big\|^2 > r \end{cases}$$

The local Hessian operator assigns a 2×2 matrix to each pixel of the examined image. Since the shape of the visible pupil is highly dependant on the direction of the gaze and the distance between the lower and the upper eyelids, its radius can not be precisely estimated by the pupil proposal algorithm, therefore a set R of possible r-s is defined. An ellipse is fitted to the limbus points using the RANSAC [2] method, the axes of which are used as radius proposals. Furthermore we found, that the eye landmark points are stable in the two corners of the eye. Based on these fixed ratios ($\frac{1}{2}, \frac{5}{12}, \frac{1}{3}$) of the half distance between the corners of the eye are added to the proposals.

The best fitting $r \in R$ is defined by its corresponding scale, in which the largest amplitude curvature of the surface defined by the intensities of the image is the minimal (which corresponds to our observation, that the pupil can be approximated by a dark blob):

$$r^* = \arg\max_r \Big\{ \lambda_{\max}\{\mathbf{H}_{\sigma(r)}\} \Big\}$$

where λ_{max} denotes the maximal amplitude eigenvalue of the Hessian matrix, and (x_0, y_0) denotes the proposed center of the pupil. Eigenvalues of the Hessian are calculated by:

$$\begin{aligned}\lambda_1\{\mathbf{H}\} &= \frac{\operatorname{trace}\{\mathbf{H}\} + \sqrt{\operatorname{trace}\{\mathbf{H}\}^2 - 4\det\{\mathbf{H}\}}}{2} \\ \lambda_2\{\mathbf{H}\} &= \operatorname{trace}\{\mathbf{H}\} - \lambda_1\{\mathbf{H}\}\end{aligned}$$

The first equality can be derived based on the following observations:

$$\begin{aligned}\operatorname{trace}\{\mathbf{H}\} &= \operatorname{trace}\{\mathbf{Q}^T\mathbf{\Lambda}\mathbf{Q}\} = \operatorname{trace}\{\mathbf{\Lambda}\mathbf{Q}\mathbf{Q}^T\} \\ &= \operatorname{trace}\{\mathbf{\Lambda}\} = \lambda_1 + \lambda_2 \\ \det\{\mathbf{H}\} &= \det\{\mathbf{Q}^T\mathbf{\Lambda}\mathbf{Q}\} = \det\{\mathbf{Q}^T\} \cdot \det\{\mathbf{\Lambda}\} \cdot \det\{\mathbf{Q}\} \\ &= \det\{\mathbf{\Lambda}\} = \lambda_1 \cdot \lambda_2\end{aligned}$$

where $\mathbf{H} = \mathbf{Q}^T\mathbf{\Lambda}\mathbf{Q}$ is the eigenvalue–eigenvector decomposition of the Hessian matrix, which always exists, because $\mathbf{H}$ is symmetric. Please note that $\mathbf{Q}$ is an orthonormal matrix, therefore $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$.

After we get the scale, the only remaining task is to examine the circularity of the examined blob, which can be measured by:

$$C_{(r)}(x_0, y_0) = \lambda_{\max}\Big\{\mathbf{H}_{\sigma(r)}(x_0, y_0)\Big\} \cdot \lambda_{\min}\Big\{\mathbf{H}_{\sigma(r)}(x_0, y_0)\Big\}$$

Since the Hessian matrix is symmetric, it can be diagonalized by an orthonormal matrix, which means that the product of the eigenvalues is equal to the determinant of the Hessian, which

can be computed faster than the value of the lower amplitude eigenvalue, therefore computed by:

$$C_{(r)}(x_0, y_0) = \det\left\{\mathbf{H}_{\sigma(r)}(x_0, y_0)\right\}$$

From there, the openness of the examined eye is calculated by:

$$\mathrm{Ind}\left(C_{(r)}(x_0, y_0) > c\right) \cdot \mathrm{Ind}\left(\lambda_{\max}\left\{\mathbf{H}_{\sigma(r)}(x_0, y_0)\right\} > 0\right)$$

Here, Ind denotes the indicator function and c is the "roundness/non-prolongation" set to 0.3. The first term is true if the portion of the eigenvalues is not too high, therefore the proposed pupil region is circular, and the second term is true, if the region is darker than its neighboring pixels. The eye is considered open if both terms are true.

III. Eyebrow-raising detection

The movement of the eyebrows is also a relevant feature in case of detecting facial expressions, as it can be used to discriminate between fear, anger and surprise. We detail our proposed eyebrow-raising detection algorithm in this section. These methods require video as input, since the change in eyebrow position is examined. Based on our empirical observations, the localization of the landmark points of the eyes and the corresponding eyebrows are accurate enough to detect eyebrow-raising based on the distances between the centroid of the eye mask and the middle of the landmark points of the eyebrow (19th and 26th points in Figure 1) respectively. The video is initially processed frame-by-frame to extract landmark points. The next step is to extract the distances of particular points.

To detect the eyebrow-raising, eyebrow-nose bottom and eyebrow-eye distances are examined. Another input for this method is the output of the blink detection.

Based solely on the eyebrow-eye distances (shown in Figure 2a) on each of the frames, the detection of corresponding microexpressions is not possible (the person in the image may squirm, lean forward, nod, etc.). Therefore, the proposed method examines the local trends of these distances by fitting a Gaussian distribution based on the previous frames. Using a 20 wide sliding window, mean and variance are calculated. In the next step, the relative likelihood is calculated for the next 3 points (one by one) using the probability density function of the normal distribution defined by the observed distances in the window. By multiplying the 3 conditional likelihoods, the joint probability is calculated assuming conditional independence. The null hypothesis is that if the records come from the same distribution, then they are independent given the window:

$$p(x_{win+3}, x_{win+2}, x_{win+1} \mid x_{win}) = \prod_{i=1}^{3} p(x_{win+i} \mid x_{win})$$

The joint probability is defined by:

$$\prod_{i=1}^{3} p(d_{win+i} \mid win) = \prod_{i=1}^{3} \frac{1}{\sigma_{win}\sqrt{2\pi}} \cdot e^{-\frac{(d_{win+i}-\mu_{win})^2}{2\sigma_{win}^2}},$$

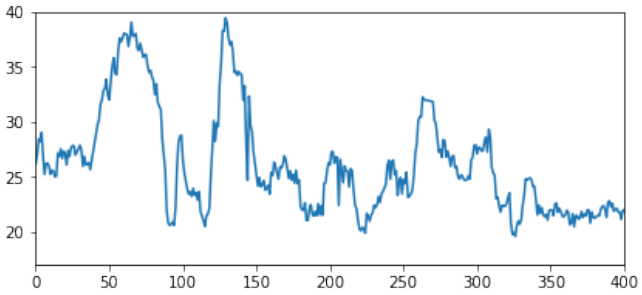
where d_{win+i} denotes the distance after the window with i frames; σ_{win} and μ_{win} denotes the standard deviation and the mean of the window respectively. The joint probability is calculated after each window in the time series. For ease of use, we work with the logarithm of the values. In the next, step eyebrow movements are detected. First, a binary threshold is applied. Using a maximum filter with a width of 5, movements close to each other are merged into one "sequence". In these sequences, local minimum search is performed to find the timestamp belonging to the most salient movement. These are points, which belong with low probability to the distribution defined by the window, thus are considered as outliers. Many of these candidates are false positives, which is detected because of the eye lowering and contraction. These proposals are eliminated by comparing the window mean and distance values belonging to that timestamp. Until this point, time series belonging to each eye are treated separately. However, while investigating the landmark points on videos, we found, that moving the eyebrow on only one side (left) affects the landmark points on the other side (right), thus "generating movement". This movement is much smaller on the unraised side (right). Thus pairing the movements of the two sides is necessary. Detections on the two sides with a distance in time less than $0.1\,\mathrm{s}$ are considered as the same movement. If the distance on one side belonging to such a movement is less than half of that on the other side, then it is a "generated movement", else it is a bilateral eyebrow-raising.

The majority of this process is also performed on the eyebrow-nose tip distance (see Figure 2b) time series. Joint probabilities are computed and are filtered using a harder threshold. Filtering for only raising movement is also applied.

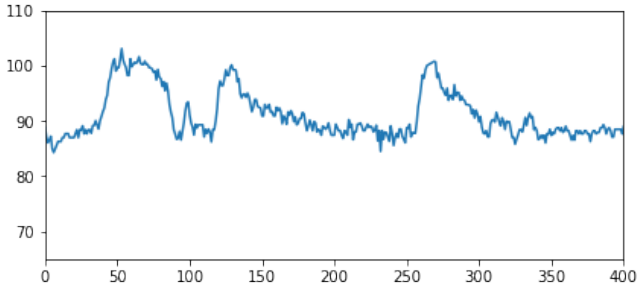
Blink timestamps resulting from the gaze detection part are used to mark possibly false positive detections in a window. If such a movement can also be found in the eyebrow-nose tip distance time series, then it is considered as a real movement, else as a false positive movement (see Figure 2d).

IV. Results

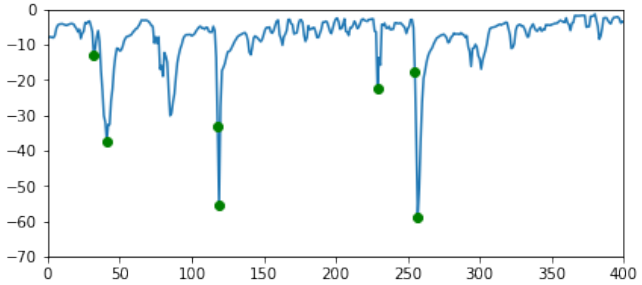
Figure 3 shows results of the eyebrow-raising detection method on adjacent video frames. Figure 3a shows an example where there is no blinking during the eyebrow raising. In case of blinking, the decision is made based on the eyebrow-nose distance. In Figure 3b a blink was detected during the eyebrow raising. However, in Figure 3c - based on the eyebrow-nose distance - it was established, that there was no eyebrow raising. The proposed algorithms for gaze and eyebrow raising detection may serve as components of an expert system aiming the detection of microexpressions. Along with other detected features, such as mouth shape and angle or wrinkles appearing on the face, these microexpressions can be utilized to identify basic emotions.



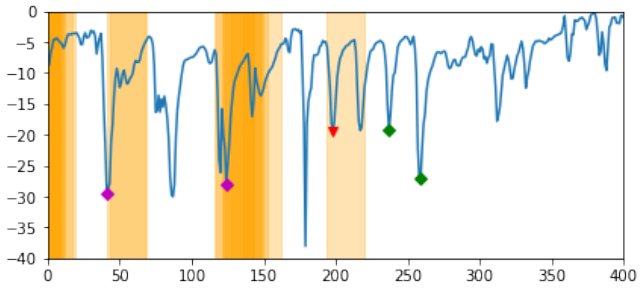
(a) Eyebrow-eye distance.



(b) Eyebrow-nose distance.



(c) Filtered detections calculated based on the eyebrow-nose distance.



(d) Final filtered detections calculated based on the eyebrow-eye distance. Orange zones are the blinking timestamps with a window around them. Green diamonds mark the detections out of the blink-windows. Magenta diamonds are the detections that fall into the orange zone but can be detected based on the eye-nose distance, thus are retained. Red triangles denote the detections that fall into the orange zone and can not be detected based on the eye-nose distance thus are permanently removed.

Fig. 2: Steps of eyebrow raising detection (right eye). Time frames are shown on the X-axis, whereas Y-axis shows pixel distances in Figues 2a and 2b and the logarithm of the probabilities in Figures 2c and 2d.

(a) Eyebrow-raising without blinking. First green diamond (at around 38) in Figure 2d.

(b) Eyebrow-raising during blinking. First purple diamond (at around 124) in Figure 2d.

(c) Blinking without eyebrow-raising. First red triangle (at around 199) in Figure 2d.

Fig. 3: Eyebrow-raising detections shown in Figure 2d.

References

[1] Paul Ekman, Joseph C. Hager, and Wallace V. Friesen. *Facial action coding system: the manual*. Research Nexus, 2002.

[2] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[3] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[4] Tony Lindeberg. Scale-space theory in computer vision, 1993.

[5] Fabian Timm and Erhardt Barth. Accurate eye centre localisation by means of gradients. *Visapp*, 11:125–130, 2011.

[6] Roberto Valenti and Theo Gevers. Accurate eye center location and tracking using isophote curvature. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[7] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001.

[8] Erroll Wood and Andreas Bulling. Eyetab: Model-based gaze estimation on unmodified tablet computers. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 207–210, 2014.

Uncertainty approximation in neural networks using parameter-space proximity regularization

Mihály Vetró
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
vetromisu@gmail.com

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

***Abstract*—A common question regarding the application of neural networks is whether the predictions of the model are reliable, in other words, what is the degree of uncertainty of our model. Generalizing a neural network into a Bayesian neural network is a frequent choice to quantify uncertainty. This means the extension of scalar weights and biases of the network to random variables. In terms of implementation, there are two main approaches: (1) the assumption of an unknown distribution for the random variables (i.e. weights) which are sampled and then utilized to infer the distribution of the output; (2) the assumption of a prior distribution for the random variables, and search for the output in the form of a similar posterior distribution. A common drawback of both approaches is that their scalability is limited, and generally require more computational resources than a simple neural network. In this paper, we introduce a new parameter sampling method, which maximizes the pairwise distance in parameter space between the models in the ensemble, therefore ensuring model diversity. Results indicate that this method generally needs less samples from the parameter space to be effective, thus it surpasses the other investigated approximation methods in terms of scalability. Finally, we apply our method to a real-life problem related to semantic segmentation of objects relevant in urban driving environments.**

***Index Terms*—uncertainty, Bayesian neural network, Bayesian approach, approximation methods**

I. Introduction

In real-world applications concerning various estimation problems, noise, observation bias and most other sources of uncertainty related to the training data, and therefore the final model, cannot be fully avoided. Thus the demand is continuously rising for more reliable models, and for measuring the output uncertainty and the error probability of a model, especially in safety-critical applications, where it is often obligatory to provide guarantees about the reliability of a given solution. Besides this, uncertainty estimation often comes with a great computational overhead compared to the construction of an ordinary model. Therefore in this paper, we introduce a new, relatively cost-effective method, and compare it to some other known approaches used to estimate uncertainty in predictive models.

This research was supported by the ÚNKP-20-5-BME-92 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund, and the János Bolyai Research Scholarship.

II. Approximating uncertainty

In general, uncertainty can be originated from the imperfections of our data (mainly incompleteness and noise), or the limits defined by the parameter space in which we are searching for the solution. Although data uncertainty can be further categorised [3] depending on its source, which can help with the improvement of our model, in this paper we are only focusing on the quantification of the total uncertainty of the model's output. In order to do this, we should consider the output (y) as a random variable (instead of a point estimate), which depends on the input (x) of the model: $p(y|x)$. The learned model is most likely not perfect, and so the distribution of the output variable will also depend on the restrictions introduced by the parameter space and the training data at our disposal. Considering these aspects, we can formulate the final distribution of the output as follows:

$$p(y|x, D) = \int p(y|x, \theta)p(\theta|D)d\theta \tag{1}$$

As we can see in equation 1, the information provided by the training data (D) is taken into consideration via the trainable parameters (θ) of the model. If we know the exact value of θ, then calculating $p(y|x, \theta)$ is basically the same as running inference on the model. The main problem arises when we consider that the training process, i.e. adjusting the parameters based on the data, is usually imperfect, due to the stochasticity of the training method, and there is also some uncertainty related to the training data. Because of this, determining the exact distribution of θ, or more precisely determining $p(\theta|D)$ is infeasible, or at least impractical in most cases. Therefore, our main goal is to provide a good approximation to this distribution, which is computable and preferably has minimal computational overhead compared to an ordinary model, which is typically a maximum a posteriori estimate of θ over $p(\theta|D)$.

III. Related work

When it comes to approximating the distribution of $p(\theta|D)$, two main approaches exist: (1) the first approach relies on taking samples from an unknown distribution over θ, (2) while the second approach assumes a known prior distribution (e.g. multivariate normal), and searches for the posterior in the

same form. The first approach is followed mostly by various Monte Carlo methods, such as *Stochastic Gradient Langevin Dynamics* (SGLD [2]) or *Anchored Ensembling* [4], while methods taking the second approach are called variational methods, like *Bayes-by-Backprop* (BBB [1]) or *Stein Variational Gradient Descent* (SVGD [7]). The methods using Monte Carlo sampling have the capability to approximate complex (but unknown) distributions, and presumably give a better estimation of the true posterior, while the variational methods have the advantage of a fully known (but more confined) posterior approximation. While both approaches and their methods have their advantages and shortcomings, it is generally true, that they are more complicated and computationally more resource-intensive than a simple neural network created for the same machine learning problem.

IV. Ensuring model diversity

There are two main criteria that an uncertainty-estimation method has to meet: (1) it has to produce a metric, which consistently correlates with the expected error of the model, and (2) this has to be achieved with minimal performance overhead compared to the original estimator on which it is based. The usual approach is to approximate the true posterior distribution of the parameters. In this case, to estimate the uncertainty of the output, such methods often require a large amount of samples from the parameter space. Considering that every sample taken from $p(\theta|D)$ is a whole new model on itself, using a large number of samples greatly increases the computational and storage requirements of the method. Therefore, instead of approximating the true posterior of θ, we are focusing on the approximation of a modified posterior, which aims to efficiently utilize every new sample, minimizing the number of samples required to reliably estimate the output uncertainty. To achieve this, our method maximizes the pairwise distance in the parameter-space between samples by sequentially training new models with the following modified loss function (for the $(m+1)$-th model):

$$L_{m+1} = \rho \left(\frac{1}{m} \sum_{i=1}^{m} ||\theta - \theta_i||_2 \right)^{-\tau} + \lambda H(y, \hat{y}) \qquad (2)$$

Where:

- ρ is the regularization multiplier
- τ is the regularization exponent
- θ_i is the parameter vector of the i-th model from the already trained models
- θ is the parameter vector of the current model
- m is the number of already trained models
- λ is the learning rate
- y and $\hat{y}$ are the true and the predicted output vectors
- H is the cross entropy function

The second half of the loss function is a simple cross-entropy error, while the first half accounts for the parameter space distance regularization. This latter part significantly increases the loss, when the average distance from the previously trained models is low, and its value diminishes as the average distance gets larger. This way, the training algorithm is forced to find a local minimum of the error function further from the previously created models, which greatly increases overall model diversity. In other words, this results in a model-ensemble, the members of which provide a different, but similarly adequate solution for the given estimation problem. This way, the variance of the members' predictions will be low when the given input is from a "well known"[1] part of the input space, and will be high otherwise. Although this property is generally true for most other known methods, we expect that this can be achieved with fewer models by ensuring model diversity. We refer to our proposed method the Distance Regularized Ensemble Approximation Method (or DREAM for short).

V. Practical results

To observe the performance of our method, we have tested it on the CityScapes dataset [5], which is strongly connected to self driving, and object detection. The dataset consists of pictures taken from the windshield camera of a car driving through the urban area of various European cities. These pictures are annotated pixel-by-pixel with various class labels, from which we used the 16 most relevant ones. We achieved this by merging some of the similar classes, and dumping the remaining ones to the "Other" class. Example outputs of a small ensemble consisting of 3 models, trained with the DREAM method are shown in figure 2. Results indicate, that the variance (or uncertainty) of the ensemble's output are noticeably higher in cases when the output is incorrect, compared to the parts of the image where the output is correct. To verify this, the average output variance where the ensemble's prediction is correct should be observed an compared to the output variance of cases in which the prediction is incorrect. Figure 1 shows the variance of the DREAM method for several different ensemble sizes. This demonstrates that there is a consistently large gap between the average variance of the correct and incorrect outputs, even in case of a small sized ensemble.

A. Comparing different methods

With regards to predictive performance, the SGLD and DREAM methods both achieved an overall accuracy of 87%, while the Bayes-by-Backprop model achieved 80% with the same model architecture on the validation dataset (500 images), after being trained on the training dataset (4000 images). Arguably, the main goal of output uncertainty approximation is to estimate the likelihood of the model's prediction being incorrect. This requires that the output uncertainty is relatively high when the prediction of the model is 'wrong', and relatively low when the model provides a correct prediction. Considering the fact that the output variance may be in different ranges for different approaches, in order to enable

[1]The part of the input space that we have reliable information about, typically (but not exclusively) close to the known training data.

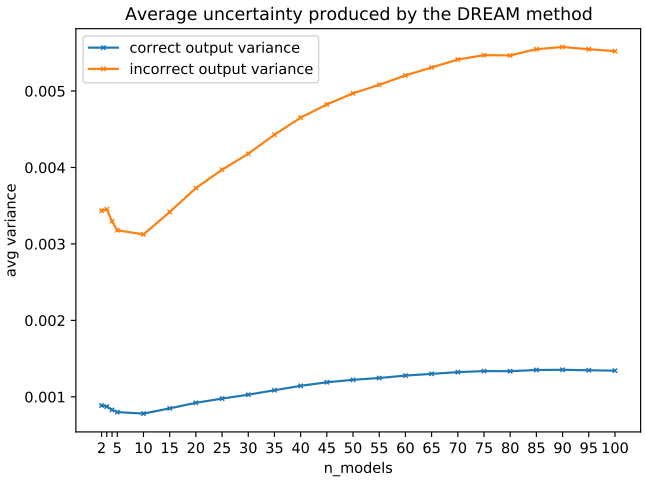


Fig. 1. Average output variance of the DREAM method on the CityScapes validation dataset for correct and incorrect predictions given various ensemble sizes.

their comparative analysis we compute the ratio between the two average variance values. This can be expressed as follows:

$$m(x_n) = \frac{U_{incorrect}(x_n)}{U_{correct}(x_n)} \tag{3}$$

Where $U_{incorrect}(x_n)$ is the average output uncertainty (variance) of an ensemble consisting of n models, produced by the x method on incorrect predictions, and $U_{correct}(x_n)$ is the average output uncertainty (variance) of an ensemble consisting of n models, produced by the x method on correct predictions. This metric indicates the amount of information that a method's output uncertainty conveys about the likelihood of the model's prediction being correct or incorrect. The comparison between three different methods (including DREAM) based on this metric is shown in figure 3. Results indicate that the $m(DREAM_n)$ value is consistently high even with smaller sized ensembles, surpassing the two other approaches on the CityScapes validation dataset. This further ensures the correlation between model error and output variance with a small ensemble size suggested by figure 2, and indicates that the DREAM method provides a more useful uncertainty estimation even with low model counts, compared to the other investigated methods (SGLD and Bayes-by-Backprop) on the CityScapes dataset.

B. Knowledge distillation

There is a fairly known approach for minimizing the size and resource requirements of estimators called *knowledge distillation* [6]. In short, this method relies on forming a large dataset by taking random samples from the output of the original, and then constructing a smaller, more efficient estimator on that dataset. Generally, this presents two challenges in our case: (1) first, we have to acquire appropriate samples from the input space, (2) and second, we must train our distilled estimator to approximate both the predictions and output variance of the original sample. The first challenge is generally considered hard, because in most cases it is infeasible to reliably model the input distribution to take samples from it. Fortunately, the CityScapes dataset contains 20.000 unlabeled images taken from various cities with the same setup, which therefore can be considered as samples taken from the very same (or at least similar) distribution from which the training and validation samples were taken. As for the second challenge, we decided to train two separate distilled models, from which one will provide the prediction and the other will estimate the output variance. We used a fully Convolutional Neural Network (CNN) architecture for this task, which consists of three main components: the encoder (3 convolutional layers with a filter count of 32, 64 and 128 respectively), the middle layers (2 convolutional layers with both having a filter count of 256 by default), and the decoder (3 convolutional layers with a filter count of 128, 64 and 32 respectively). Among these, the vast majority of the parameters are related to the middle layers, which usually also handle most of the logic contained within the model, therefore we used the filter count of these middle layers as a measure of both the models' size and complexity. We trained our distilled models on an ensemble consisting of 25 models for every method, and achieved basically the same predictive performance as the original ensemble in every one of the three tested methods (Bayes-by-Backprop, SGLD, DREAM) with a distilled model having a middle layer filter count of 256, thus this model was selected to handle predictions. As for the output uncertainty estimation, we tested multiple different filter counts, and observed their $m(x)$ value for the predictions of the previously mentioned model. This is presented in figure 4. This shows us that a relatively accurate estimation can be achieved for the output variance of the ensembles with a filter count of 256 for the SGLD and BBB methods, and 512 for the DREAM method respectively. In addition, having a higher filter count will most likely result in overfitting, and therefore worsen the overall performance on the validation data. Considering this notion, it should also be noticed, that while the model achieved an adequate distillation for the DREAM method with a middle layer filter size of 256 for the predictive model, and 512 for the uncertainty estimator model respectively, its $m(x)$ value still falls behind of a DREAM-ensemble consisting of 3 models, which all have a middle layer filter size of 256. Therefore, while knowledge distillation is a viable option for most ensemble-based methods, in case of the CityScapes dataset and the DREAM method, it was outperformed by the original ensemble with a similar overall parameter count.

References

[1] C. Blundell, J. Cornebise, K. Kavukcuoglu and D. Wierstra, "Weight Uncertainty in Neural Networks," 2015.
[2] A. Korattikara, V. Rathod, K. Murphy and M. Welling, "Bayesian Dark Knowledge," 2015.
[3] E. Hüllermeier and W. Waegeman, "Aleatoric and Epistemic Uncertainty in Machine Learning: An Introduction to Concepts and Methods," 2020.
[4] T. Pearce, F. Leibfried, A. Brintrup, M. Zaki and A. Neely, "Uncertainty in Neural Networks: Approximately Bayesian Ensembling," 2020.

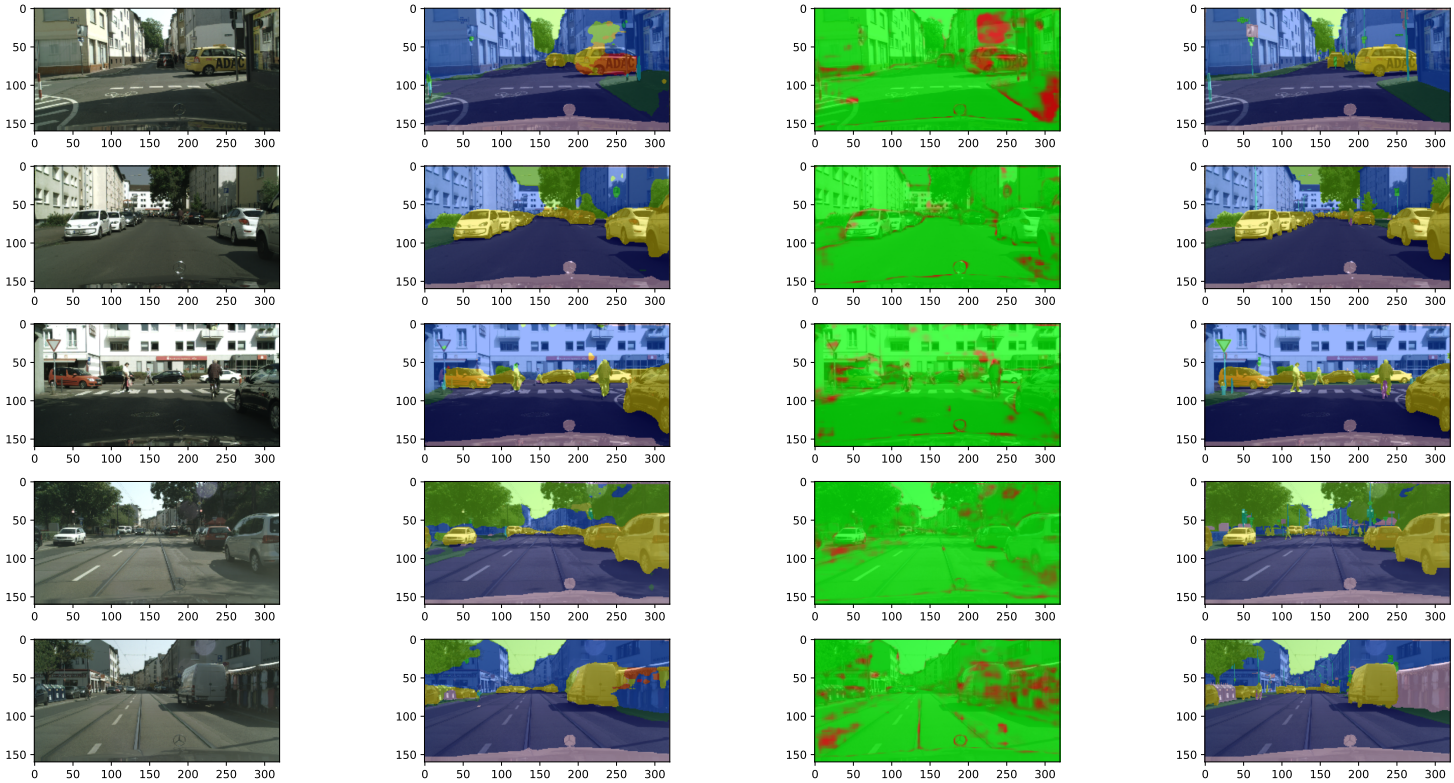


Fig. 2. Example outputs of the DREAM method applied on the CityScapes validation dataset, evaluated by an ensemble of 3 models. The columns from left to right: input, output, output variance (red: high, green: low), expected output.

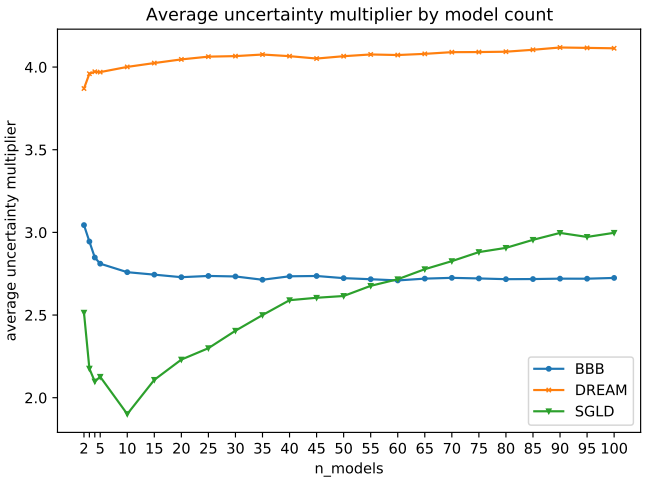


Fig. 3. The $m(x_n)$ value of three different methods by ensemble size applied on the CityScapes validation dataset.

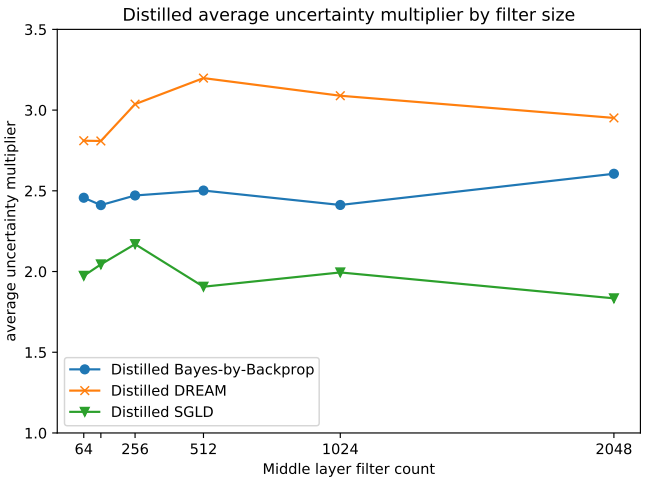


Fig. 4. The $m(x)$ value of the distilled models on three different methods by middle layer filter count on the CityScapes validation dataset, distilled from an ensemble of 25 models.

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," 2016.

[6] T. Jiaxi, S. Rakesh, Z. Zhe, L. Dong, S. Anima, H. C. Ed and J. Sagar, "Understanding and Improving Knowledge Distillation," 2020.

[7] Q. Liu and D. Wang, "Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm," 2019.

Qualitative reasoning assisted empirical system identification

András Földvári
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
fandras95@gmail.com

András Pataricza
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
pataric@mit.bme.hu

***Abstract*—The design and operation of modern IT-based systems especially cyber-physical systems (CPS), need model-based approaches due to their complexity. However, the limited faithfulness of pure analytic models with speculative layouts prohibits their use in complex systems. Complexity necessitates empirical system identification from observations.**

Exploratory data analysis (EDA) is a main approach to observation-based system identification. EDA combines visual methods and summary statistics for initial data analysis. In traditional EDA, there is a gap between the thinking of the expert and the logic of the analysis method.

In general, everyday and engineering thinking use a qualitative approach. Qualitative abstraction represents the system with a discrete model of the granularity of individual operation domains. This way, qualitative models avoid the complexity problems by identifying and focusing on the most important features observed.

Qualitative modeling is a gradual, iterative process extracting more and more abstract details of the observations. As the individual model fragments merge into the evolving system model, each step needs validation and verification (V&V). The use of a discrete formalism supports V&V by logic reasoning.

The paper presents a prototype implementation covering data profiling, continuous-qualitative abstraction, and Answer Set Programming (ASP) for logic reasoning.

***Index Terms*—empirical system identification, qualitative modeling, model validation and verification**

I. Introduction

The goal of system identification is to extract a faithful model from observations (Fig.1) to support model-based system design and management. The proposed extraction workflow includes the main characteristics of the *pure analytic models* (mathematical precision), *exploratory data analysis* (iterative process), and *qualitative reasoning* (engineering thinking for model interpretation) aspects.

The uniform representation of the models will be the embedding frame of *digital twins* (DT). DTs bridge the physical and cyber world by a model of the system. They are the core of complex CPS design and supervisory control.

The system identification workflow leading to a DT is divided into five steps, namely: 1) **engineering modeling** (merging the available design information on the system under evaluation); 2) **data preparation** (resulting in an analysis ready form of observations); 3) **data interpretation** (the core activity of EDA delivering mathematical model fragments); 4) **model construction** (composing the fragments into the evolving DT); 5) finally, **reasoning** (V&V, estimating and checking the impact of the individual steps of model building).

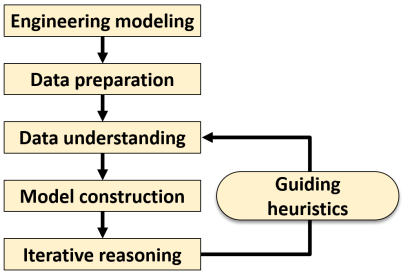


Fig. 1: System identification workflow

Understanding the data requires a *hierarchical refinement* (drill-down) to examine a sub-phenomenon or subsystem in details. Therefore, the workflow may include several *adaptive iterations*, where the results of the previous steps guide the analysis and model building progresses.

The proposed workflow supports modeling by a unified (qualitative) representation of the mathematical and engineering models. All the models originating potentially in different paradigms can be mapped to the common language of mathematical logic and this merged model defines the boundaries of the solution space.

II. System identification

A. Engineering modeling and data preparation

Strategy design requires the definition of the *engineering objectives* including *analysis* and *measurement requirements*, and the collection of *background models* about the system under evaluation. This step also includes the collection of the *observations* of measurements (benchmarking campaigns).

The purpose of the *data preparation* step is to get an analysis-ready dataset. *Data cleansing* decides upon the mitigation actions related to missing or ill-formed data. Furthermore, data validation (e.g., semantic constraints) ensures the proper *data quality* for further analysis.

The project was funded by the European Union, co-financed by the European Social Fund (EFOP-3.6.2-16-2017-00013). The results were established in the framework of the professionalcommunity of Balatonfüred Student Research Group of BME-VIK

B. Data interpretation

The main goal of data interpretation is *information (model) extraction* from the data expressing generalizable knowledge about the target system.

As observations originate in the compound of the system under evaluation, the instrumentation, and workload, EDA should cover the soundness of the entire campaign as well.

Modern data analysis frameworks facilitate the preliminary data analysis by means of *automated data profiling*, a general-purpose pre-stored workflow. Data profiling performs the most typical application-independent *statistical analysis* tasks over the target dataset. The automated data profiling guides the EDA by highlighting the *particular phenomena* and *generating a warning* on *unusual statistical characteristics* to raise the expert's attention.

Data profiling derives the overall characteristics of the dataset. The most important measures cover:

- **missing values** (the engineering interpretation of missing values is crucial in understanding the behavior from the observations),
- **quantile statistics** (allows a non-parametric analysing the ranges and distribution of the data);
- **descriptive statistics** (measures the *asymmetry of the distribution* of the variables (e.g., skewness), an important metric in benchmark analysis and a critical metric in real-time systems analysis);
- **interactions** (pairwise visual exploration of the continuous variables, allows checking the relation of the variables separately)
- **correlation analysis** (supporting *feature selection* and *causality analysis* over continuous, ordinal and categorical variables).

The *detailed analysis* focuses on several peculiarities that were highlighted by the profiling analysis.

C. Model construction

The previous steps are easy to use for domain experts to pre-process and understand the data. However, there is still a gap between the thinking of the expert and the logic of the analysis method, in general, everyday and *engineering thinking use a qualitative approach*.

Qualitative thinking is near to common sense engineering thinking (e.g., a system manager typically reasons like *if the amount of free memory is low, then the throughput is low*). Usually, the exact values of variables during analysis is irrelevant to explore the operational principle in the system. This way, our proposed method uses *qualitative models* [1]–[4]. Modern IT-based system are designed to be scalable, but their fundamental functioning depends less of the resources allocated (dimensioning).

Hybrid modeling (Fig. 2) takes the advantages of both the *qualitative* and *continuous models*. It characterizes dimensioning-independent *operational ranges* with discrete values and scales the *qualitative ranges* by choosing their boundaries. The approach allows integrating all available knowledge into a single uniform model.

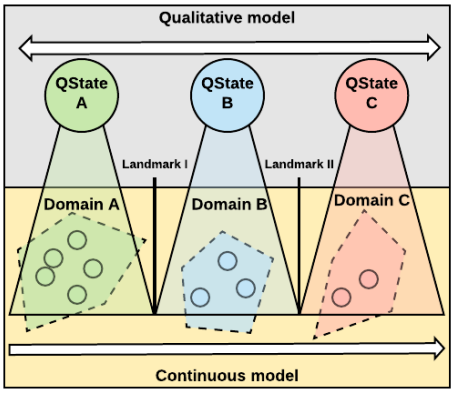


Fig. 2: Hybrid modeling

This discrete model aggregates the individual subspaces corresponding to qualitatively different modes of operation into a *single abstract qualitative state* (on observations, this corresponds to assigning a single element to an entire cluster). The engineering reason behind the hybrid modeling technique is that elements inside of a domain corresponding to a homogeneous operation regime expose *similar behavior*, which can be modeled as identical ones and map into a single state.

Creating a hybrid model as part of the system identification process necessitates the clustering of the data into domains (*operational regimes*), which show qualitatively identical behavior of the system. This task, referred to as *discretization*, can be performed by an expert manually (by visualization techniques) or by automated means (by algorithmic clustering).

D. Iterative reasoning

The reasoning step of the process allows the *exploration* and *analysis* of *relationships in the data*. The iterative *consistency checking* tests that the concluded information is still consistent with the extracted and the original models. For instance, if by correlation analysis $a \sim b$ and $b \sim c$ in the qualitative model preserving proportionality, $a_q \sim_q c_q$ should be valid for the proportionality of their qualitative counterparts. This way systematically accessing the implications of a new model fragment can enrich the model or generate criteria for V&V.

Such reasoning can reveal *hidden relations*. The *engineering interpretation* of the concluded information helps the *understanding* of the models. The iterative reasoning (Fig. 3) supports the *continuous refinement of the system model*. The initial background knowledge covers formalized requirements, constraints, and it can represent the hierarchical structure of the system, observations and other engineering models.

We propose *Answer Set Programming* (ASP) [5], [6] for universal modeling language for all the qualitative models and reasoning. As ASP provides a *knowledge representation* and *logic reasoning framework* [4], it allows the representation of both the engineering and mathematical models.

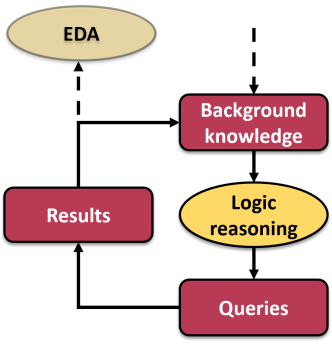


Fig. 3: Iterative reasoning process

Technically, ASP is a modularizable Prolog-like language with *stable model semantics* [7]. *Modularization* facilitates to create the different sub-models in a well-structured way. The observation, background knowledge, and queries can be stored in different ASP modules. The analyst can include these modules separately for reasoning about the model fragments.

III. Proof of concept framework

We created a proof of concept in the form of Jupyter Notebooks [8] [9]. The objective was the support of education and the conceptual validation of the qualitative approach by pilot examples. This framework presents an example for system identification process for educational and demonstrational purposes and serves as a blueprint for similar tasks.

The modern notebook data technology for modeling has important advantages in empirical systems engineering. For instance, every measurement campaign should be validated from a *metrological* point of view. This includes expert-driven processes that require a proper documentation (e.g., executed steps, tools, evaluation of the results) for *reproducibility*.

Notebook technology is the most common general-purpose data analysis tool that implements *literate programming* [10], a solution to support expert analysis activities through a self-documenting and reproducible process. In particular, it supports the *evaluation of the results*, the *preparation of the documentation*, and the traceability of the flow of adaptive model refinement process in empirical systems engineering.

The majority of open-source packages were evaluated during the notebook composition process and those were included which fit best to empirical system identification [11].

The notebooks cover the following topics:

1) **Preliminary analysis with data profiling** [12]: The first notebook executes the data profiling analysis highlighting the analysis of missing data, basic variable statistics, and correlation analysis. The profiling step triggers warning messages to highlight specific phenomena in the dataset.
2) **Detailed EDA** [13]: The second notebook focuses on the drill-down by analyzing the data using interactive visualization and checks for rare events. Special attention was paid to rare events and outlier detection fundamental in critical system design.
3) **Aggregated analysis** [14]: The third notebook uses the previously estimated operational ranges (qualitative data) for data aggregation, feature selection, and transition graph construction.
4) **Integration of ASP**: The notebook about the integration of ASP presents the discrete qualitative model and reasoning over it. This notebook represents the examples on the aggregated data.

While this series of notebooks covers the fundamental steps of qualitative model identification and analysis its open structure facilitates the inclusion of other problem specific methods as well. For instance, an other demo notebook examines xAI (*eXplainable AI*) metrics with the DALEX [15] model explanation and exploration tool, while another one the IBM AIX360 [16] model examination tool [17].

IV. Future work

The PoC implementation targeted only the evaluation and selection of the core algorithms in order to get a sense of the effectiveness of the approach. The approach is feasible as an end-to-end solution for the empirical identification of qualitative models. However, several limitations have to be resolved before applying them to large-scale real-life projects due to the nature of PoC. As embedded automation for subtask is a productivity booster in the analysis workflow, further intelligent (sub)algorithms will be integrated to increase the efficiency. Assurance of computational scalability needs proper big data ready data storage and management. The rest of the section summarizes the algorithmic challenges needed for practical usefulness.

Engineering modeling Currently, a completely unfolded ASP program represents all the models. However, efficient data management (like *ontologies* and *knowledge graphs*) is needed for knowledge representation related to complex models. The efficient handling of *many-dimensional big data* is not yet explored in full detail. However, there are some packages (DASK [18]) that promise more efficient data handling in notebook environments.

Data preparation The management of *missing data* differs in the business and engineering context. While, in business data omission or imputation provide a phenomenological solution; engineering modeling needs *root-cause analysis* to identify measurement errors or omission failures in the system.

Model extraction Identifying a state machine's highly sequential behavior from the discretized observations is a special branch in system identification. *Process mining* (PM) approaches [19]–[21] implement dedicated algorithms to extract compact models from event logs. The current notebooks illustrate the potential with a mockup-styled implementation of the functionality.

Black box models The integration of *eXplainable AI* (xAI) techniques aim at the evaluation of ML models embedded in a system. At the same time, the qualitative counterparts of the explanatory metrics can be included into a qualitative model. E.g., model explanations in xAI [15], [22], [23] allow the evaluation of *variable importance* and *sensitivity analysis*.

Temporal expressions The common ways to describe sequential behavior and requirements, ASP can be extended to solve temporal programs through an extension (Telingo [24]). Further research is required to explore how the increased expression power can facilitate the model and consistency checking for the exploration workflow which is confined currently to a few steps only.

Causal inference Exploitation of *causal relations* has a huge potential to improve the efficiency of the EDA. Causal inference is a type of *inductive reasoning*. Causal inference draws a conclusion about a causal connection based on the conditions of the occurrence of an effect. This way, inductive reasoning should be explored in the future.

Inductive learning *Inductive learning of ASP* (ILASP) [25], [26] is a machine learning paradigm for learning ASP programs from examples. This approach promises further automation of model extraction. [27] explored some aspect of the causal exploration and the inductive ASP learning environment, but this is still a promising future work.

Rough Set Theory RST [28], [29] is a mathematical paradigm to manage noisy and uncertain data. This way, it is a natural candidate for managing measurement data in a discrete form. RST provides a formal approximation of a set in the form of sets that define the lower and the upper approximation of the original set. Potentially, RST can be used for model approximation in such problems where the impacts of over and under approximations are asymmetrical like in risk and safety analysis or the validation of a requirement does not require an exact match of the model.

Back annotation The reasoning process's *total integration* into the analysis environment and the reasoning capabilities are highly promising, but their tight integration requires future work. On the one hand, the current integration leaves the evaluation of the results to the expert. On the other hand, a stepwise fine granular integration promises proper guidance for the data exploration process.

Technical difficulties Exploratory data analysis in a notebook environment uses many *external packages*. The *maintenance* of open-source packages is also a non-trivial problem. There are solutions for organizing and maintaining notebook projects (e.g., Cookiecutter [30]), but the exploration of these tools and techniques require further research.

V. Summary

The paper introduced a proof of concept framework in the form of notebooks for the proposed empirical system identification workflow and presented the potential future work in many possible directions.

The PoC implementation proved its usefulness at courses at the University of Florence and BME as teaching material and a blueprint for the students' individual work.

References

[1] K. D. Forbus, "Qualitative process theory," *Artificial intelligence*, vol. 24, no. 1-3, pp. 85–168, 1984.

[2] ——, "Qualitative modeling," *Foundations of Artificial Intelligence*, vol. 3, pp. 361–393, 2008.

[3] I. Kocsis, "Qualitative models in resilience assurance," 2019, PhD dissertation.

[4] F. Van Harmelen, V. Lifschitz, and B. Porter, *Handbook of knowledge representation*. Elsevier, 2008.

[5] G. Brewka, T. Eiter, and M. Truszczyński, "Answer set programming at a glance," *Communications of the ACM*, vol. 54, no. 12, pp. 92–103, 2011.

[6] M. Gebser, R. Kaminski, B. Kaufmann, and T. Schaub, *Answer Set Solving in Practice*, ser. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2012.

[7] M. Gelfond and V. Lifschitz, "The stable model semantics for logic programming." in *ICLP/SLP*, vol. 88, 1988, pp. 1070–1080.

[8] "Jupyter Notebook," https://jupyter.org/, accessed: 2020-11-27.

[9] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. E. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. B. Hamrick, J. Grout, S. Corlay *et al.*, "Jupyter notebooks-a publishing format for reproducible computational workflows." in *ELPUB*, 2016, pp. 87–90.

[10] D. E. Knuth, "Literate programming," *The Computer Journal*, vol. 27, no. 2, pp. 97–111, 1984.

[11] A. Földvári, "Observations based modelling of it systems," 2020, BME - MSc thesis.

[12] "Analysis Notebooks - Data Profiling," https://colab.research.google.com/drive/1zOPKkm4blwZZ7rAtWfWUq1xhQViNqQY, Accessed: 2021-01-22.

[13] "Analysis Notebooks - Detailed EDA," https://colab.research.google.com/drive/1iOqz14GJYdGcYSRJeXHCGixZw8NwWZxs, Accessed: 2021-01-22.

[14] "Analysis Notebooks - Aggregated Analysis," https://colab.research.google.com/drive/1CdB3rk0dwPFFEWsoxnv9uzO28GWRxaUM, accessed: 2021-01-22.

[15] "DALEX - moDel Agnostic Language for Exploration and eXplanation. ," https://modeloriented.github.io/DALEX/, accessed: 2020-11-27.

[16] V. Arya, R. K. Bellamy, P.-Y. Chen, A. Dhurandhar, M. Hind, S. C. Hoffman, S. Houde, Q. V. Liao, R. Luss, A. Mojsilovic *et al.*, "Ai explainability 360: An extensible toolkit for understanding data and machine learning models," *Journal of Machine Learning Research*, vol. 21, no. 130, pp. 1–6, 2020.

[17] G. Pelesz, "Modelling of explanations of ai solutions," 2020, BME - BSc thesis.

[18] M. Rocklin, "Dask: Parallel computation with blocked algorithms and task scheduling," in *Proceedings of the 14th python in science conference*, no. 130-136. Citeseer, 2015.

[19] W. Van Der Aalst, A. Adriansyah, A. K. A. De Medeiros, F. Arcieri, T. Baier, T. Blickle, J. C. Bose, P. Van Den Brand, R. Brandtjen, J. Buijs *et al.*, "Process mining manifesto," in *International Conference on Business Process Management*. Springer, 2011, pp. 169–194.

[20] C. J. Turner, A. Tiwari, R. Olaiya, and Y. Xu, "Process mining: from theory to practice," *Business Process Management Journal*, 2012.

[21] A. Berti, S. J. van Zelst, and W. van der Aalst, "Process mining for python (pm4py): bridging the gap between process-and data science," *arXiv preprint arXiv:1905.06169*, 2019.

[22] P. Biecek and T. Burzykowski, *Explanatory Model Analysis: Explore, Explain, and Examine Predictive Models*, ser. Chapman & Hall/CRC Data Science Series. Taylor & Francis Limited, 2020. [Online]. Available: https://pbiecek.github.io/ema/

[23] H. Baniecki and P. Biecek, "The grammar of interactive explanatory model analysis," *arXiv preprint arXiv:2005.00497*, 2020.

[24] "Telingo - solver for temporal programs," https://github.com/potassco/telingo/, accessed: 2020-01-22.

[25] M. Law, A. Russo, and K. Broda, "Inductive learning of answer set programs," in *European Workshop on Logics in Artificial Intelligence*. Springer, 2014, pp. 311–325.

[26] ——, "Inductive learning of answer set programs from noisy examples," *arXiv preprint arXiv:1808.08441*, 2018.

[27] A. Földvári and A. Pataricza, "Machine learning and reasoning for exploratory data analysis," 2019, BME - Scientific Students' Association Report - Available: http://tdk.bme.hu/VIK/info1/Feltaro-adatelemzes-tamogatasa-gepitanulasi.

[28] Z. Pawlak, "Rough sets," *International Journal of Computer & Information Sciences*, vol. 11, no. 5, pp. 341–356, 1982.

[29] ——, *Rough Sets: Theoretical Aspects of Reasoning about Data*. Springer Science & Business Media, 1991, vol. 9.

[30] "Cookiecutter - Jupyter Project organizer," https://github.com/cookiecutter/cookiecutter, accessed: 2020-01-22.

Compensation of the major drawback of oscillometric blood pressure measurement

Péter Nagy, Ákos Jobbágy
Budapest University of Technology and Economics,
Department of Measurement and Information Systems,
Budapest, Hungary
Email: {nagy, jobbagy}@mit.bme.hu

***Abstract*—Oscillometric blood pressure measurement is one of the most commonly used methods to estimate the state of the cardiovascular system. Despite its widespread use, accuracy of the oscillometric method is influenced by several factors and can be questionable in case of arrhythmia, jerky breathing and increased arterial stiffness. This paper analyzes the accuracy of the oscillometric method for measurements of patients affected by cardiovascular diseases. Illustrative examples are given when the conventional oscillometric method produces high errors. A method is proposed for the compensation of these errors based on the photoplethysmographic signal.**

Keywords—blood pressure measurement; oscillometric method; photoplethysmography

I. INTRODUCTION

Blood pressure is one of the most important vital signs used to assess the state of the cardiovascular system. Automated blood pressure monitors are easy to use and inexpensive devices providing a quick way to estimate blood pressure mostly applying the oscillometric method. The oscillometric method is based on the measurement of pressure oscillations in a pneumatic cuff typically wrapped around the upper left arm [1]. Most automated blood pressure monitors inflate the cuff quickly above the systolic blood pressure (SBP) and then deflate the cuff slowly. The amplitude of pressure oscillations in the cuff increases as cuff pressure falls between SBP and mean arterial pressure (MAP) and decreases below MAP. Maximal oscillation occurs close to the MAP [2]. The estimation of SBP and the diastolic blood pressure (DBP) is based on the observation that the amplitude of oscillations at SBP and DBP are nearly fixed ratios of the maximum amplitude of oscillations [3]. However, these characteristic ratios are obtained empirically, and they show large variability among individuals. The accuracy of oscillometric blood pressure measurement is influenced by several physiological factors including pulse pressure, anatomical position, elasticity and size of the measured artery and properties of the surrounding tissue [4]. Increased arterial stiffness generally impairs the accuracy of oscillometric blood pressure measurement [5]. Moreover, SBP and DBP can change 20 mmHg or more within a few heartbeats due to the intrinsic physiological variation of blood pressure [6]. Thus, momentary values provided by the oscillometric method do not characterize blood pressure appropriately.

A more comprehensive picture can be gained about the state of the cardiovascular system, if more than one physiologically relevant signal is measured. Photoplethysmography (PPG) is an optical technique for the measurement of cardiac-induced pulsatile changes in tissue blood volume [7]. PPG enables or promotes the measurement of cardiovascular parameters, which correlate with arterial blood pressure, such as heart rate, pulse wave velocity and tissue blood volume changes [8]. Nitzan reported an automatic cuff-based measuring method of SBP using an inflatable cuff and two PPG sensors at two peripheral sites [8]. The amplitude and baseline of the PPG signal, heart period and time delay between the arrival time of PPG pulses to the two sensors were used to help estimate SBP. The cuff was used for the initial calibration phase. During the calibration, disappearance and reappearance of the PPG signal was detected based on a parameter characterizing the pattern of pulses and the correlation between subsequent heart cycles in the PPG curve. Yoon et al. used both ECG and PPG signals to estimate SBP and DBP [9]. Pulse transit time, systolic upstroke time, diastolic time and the width of 2/3 pulse amplitude were used as parameters in linear regression analysis for SBP and DBP. For SBP, pulse transit time from the ECG R peak to the PPG waveform maximum derivative point, for the DBP, diastolic time allowed the best estimation when using individual calibration of regression lines. Kurylyak et al. proposed a method for continuous non-invasive blood pressure estimation from the PPG signal based on artificial neural networks [10]. Cardiac period, systolic upstroke time, diastolic time, and systolic and diastolic width at 10, 25, 33, 50, 66, 75 percent of pulse height were used as extracted parameters. Shin & Min. defined a parameter called pressure index taking into account the PPG signal and the subject's heart rate and height [11]. The proposed pressure index showed statistically significant correlation with SBP and MAP but not with DBP. Mousavi et al. reported a method for blood pressure estimation using raw values of the PPG signal without extracting special features [12]. The PPG signal between two consecutive systolic peaks was considered as a feature vector. After dimension reduction, feature vectors were used as inputs to nonlinear regression algorithms. Adaptive Boosting regression produced the

smallest mean error and was selected as the best estimator of SBP, MAP and DBP.

In this paper, we analyze the accuracy of oscillometric blood pressure measurement for recordings of patients affected by cardiovascular diseases. Illustrative examples are given when the conventional oscillometric method produces high errors. A method is proposed for the compensation of these errors making use of the extra information provided by the PPG signal.

II. Materials and Methods

A. Data Acquisition

Measurements were carried out using a home health monitoring device [13]. The device contains an inflatable cuff with two control valves and it is able to keep cuff pressure at a constant value. The device also measures ECG in Einthoven II-lead and PPG at the fingertip with a sampling frequency of 1 kHz. A transmission-type PPG sensor is used to reduce the effect of motion artifacts. The device inflates the cuff with approximately 6 mmHg/s speed to 170 mmHg, then deflates the cuff with the same speed. Deflation is stopped at 60 mmHg for 10 seconds. When 40 mmHg is reached during deflation, CP changes abruptly to 0 mmHg. During the measurements, investigated in this paper, signals were recorded in resting state of the patient, in supine position.

B. Tested persons

30 patients (13 females, 17 males) affected by cardiovascular diseases and two healthy adults (2 males) participated in the measurement series reported in this paper. Diseases of patients included aortic valve stenosis, hypertension, angina pectoris, diabetes mellitus, atrioventricular nodal reentrant tachycardia, atrial fibrillation and coronary artery disease. All tested persons gave their written consent. The research was performed in accordance with the Declaration of Helsinki and the study protocol was approved by the Scientific and Research Committee of the Hungarian Medical Research Council (SE RKEB 46/2020).

C. Estimating SBP assisted by the PPG signal

During inflation, the pulsation of the PPG signal disappears when the cuff pressure exceeds SBP. Nitzan proposed a method for the detection of the disappearance of the PPG signal based on the pattern of pulses [8]. However, the signal-to-noise ratio of some clinical recordings was very low, so we used a method that does not require the designation of fiducial points, only the envelope of the signal must be determined. The maximum difference between the upper and lower envelope was calculated in a sliding window with a length of 300 ms. Disappearance of the PPG signal was detected applying thresholds on the output of the moving window-based envelope range calculation. In this paper we focus on SBP, DBP estimation aided by the PPG signal is not discussed.

D. Comparing results to the conventional oscillometric method

Results of the PPG-based blood pressure estimation were compared to the conventional oscillometric method. Manufacturers of automatic oscillometric monitors do not publish the methods and coefficients implemented in their devices [14], therefore, we implemented the conventional oscillometric method described by Geddes et al. [15]. SBP calculated using the PPG signal was compared to SBP determined by the oscillometric method, during inflation. In the conventional oscillometric method, the cuff is inflated quickly and deflated slowly and SBP is detected during deflation. However, the device we used inflates the cuff slowly enabling the detection of SBP during inflation.

III. Results

A. Estimating SBP in case of arrhythmia

The amplitude and shape of oscillometric pulses corresponding to irregular heartbeats can be significantly different from that of normal heartbeats. As a result, the oscillometric algorithm may produce high errors. Figure 1 shows the bandpass filtered cuff pressure signal of a patient with angina pectoris (chest pain due to coronary heart disease). Heartbeats are irregular between the disappearance of the PPG signal (triangle) and the point of SBP detection by the oscillometric method (circle). Oscillometric pulses corresponding to irregular heartbeats have very small amplitude and altered shape compared to normal heartbeats. Note, that negative pressure values can occur due to bandpass filtering.

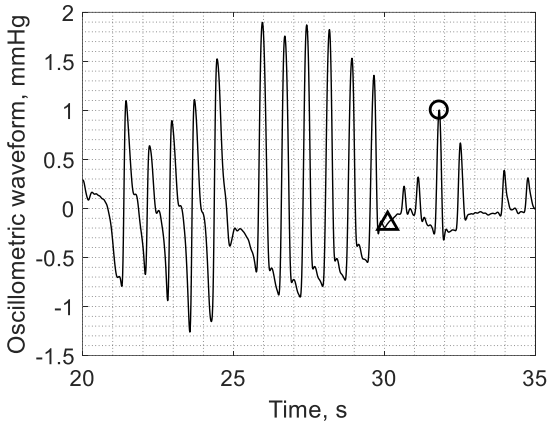


Fig. 1. Bandpass filtered cuff pressure signal (oscillometric waveform) of a patient with arrhythmia (senior male patient with angina pectoris). The oscillometric pulses corresponding to irregular heartbeats have smaller amplitude and altered shape compared to normal heartbeats. Triangle: disappearance of PPG. Circle: pulse peak corresponding to SBP detected by the conventional oscillometric method. Difference in estimated SBP = 13.1 mmHg.

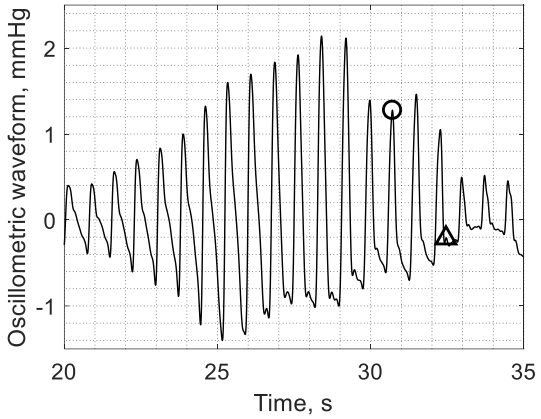


Fig. 2. Bandpass filtered cuff pressure signal (oscillometric waveform) of a patient with increased arterial stiffness (senior male patient with hypertension). There is a sudden drop in oscillometric amplitudes two heart cycles after the maximal amplitude. Triangle: disappearance of PPG. Circle: pulse peak corresponding to SBP detected by the conventional oscillometric method. Difference in estimated SBP = -9.9 mmHg.

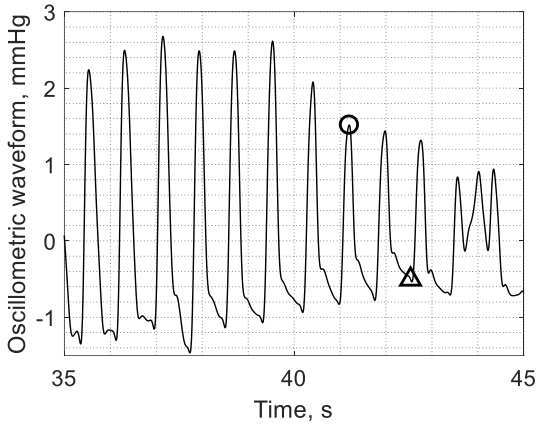


Fig. 3. Bandpass filtered cuff pressure signal (oscillometric waveform) of a healthy senior male after short physical stress. The oscillometric method (circle) detects SBP earlier than the disappearance of the PPG signal (triangle). Difference in estimated SBP = -6.8 mmHg.

B. Estimating SBP in case of increased arterial stiffness

Systolic and diastolic detection ratios are sensitive to the degree of arterial stiffness [5]. Figure 2 shows the bandpass filtered cuff pressure signal of a patient with increased arterial stiffness (senior patient with hypertension). There is a sudden drop in oscillometric amplitudes two heart cycles after the maximal amplitude (MAP).

C. Estimating SBP in case of physical stress

Physical stress affects heart rate, vascular mechanisms controlling the elasticity of arteries and the frequency of breathing. As a result, factors affecting the accuracy of the oscillometric method are more pronounced during and after physical stress. Figure 3 shows the bandpass filtered cuff pressure signal of a healthy senior adult after running one floor downstairs and one floor upstairs. The oscillometric method detects SBP earlier than the disappearance of the PPG signal.

D. Estimating SBP in case of jerky breathing

Breathing modulates heart rate and heart rate variability, pulse wave transit time and blood pressure. Irregular, jerky breathing or coughing can alter the shape and amplitude of oscillometric pulses and may lead to inaccurate measurement results. Figure 4 shows the bandpass filtered cuff pressure signal of a healthy young adult performing forced irregular breathing during the measurement. The shape of oscillometric pulses show high variability during the measurement and the change of amplitudes after exceeding MAP is not monotonic.

E. Comparing SBP estimation based on the PPG signal to the conventional oscillometric method

Figure 1-4 illustrate deviations posing difficult situation to the conventional oscillometric method. The difference between SBP values estimated based on the PPG signal and on the conventional oscillometric method was calculated for all 30 measurements of patients. The mean difference was 1.04 mmHg with a standard deviation of 6.04 mmHg. The largest positive difference found was 13.6 mmHg, the largest negative difference was -12.6 mmHg.

IV. Discussion

Detecting the disappearance of the PPG signal during cuff inflation enables the accurate detection of SBP in situations, when the oscillometric method produces high errors. However, some factors affecting the accuracy of oscillometric measurement can make the detection of SBP from the PPG signal also difficult. Irregular heartbeats, irregular breathing, or coughing occurring when cuff pressure is near SBP, can make the detection of the disappearance of the PPG signal ambiguous. The envelope-based detection of the disappearance of PPG helps mitigate this problem. However, if arrhythmia or coughing occurs in the heart cycle, which corresponds to the time point when cuff pressure exceeds SBP, the accuracy of the method decreases.

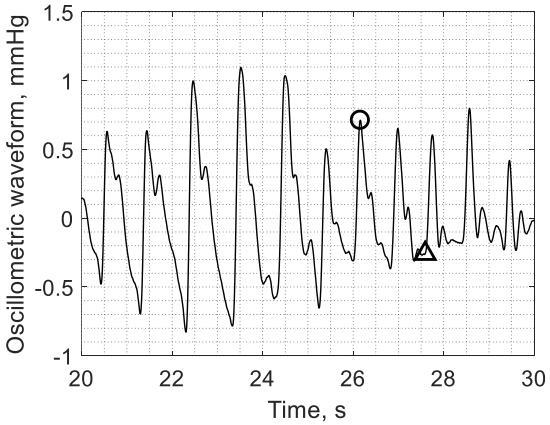


Fig. 4. Bandpass filtered cuff pressure signal (oscillometric waveform) of a healthy young male performing forced irregular breathing during the measurement. The shape of oscillometric pulses show high variability during the measurement and the change of amplitudes after exceeding MAP is not monotonic. Triangle: disappearance of PPG. Circle: pulse peak corresponding to SBP detected by the conventional oscillometric method. Difference in estimated SBP = -7.6 mmHg.

Another way to handle this problem is to limit the time interval where disappearance of the PPG signal is searched for by the algorithm. Parallel measurement of ECG and PPG enables the calculation of pulse wave transit time (PWTT). In most cases the R-peaks in the ECG signal can be detected reliably, thus an estimation can be given for characteristic points of the PPG signal in each heart cycle, based on possible limits of PWTT.

Motion artifacts in the PPG signal can also lead to erroneous results of SBP detection as they can have an impact on the calculation of threshold values. A possible way to validate thresholds is to calculate them for both inflation and deflation independently and compare the detected SBP values. It is known, that SBP values detected during inflation and deflation may differ substantially [16], however an upper limit can be defined for the difference. The comparison of SBP values can serve as a sanity check for threshold values. Motion artifact removal can also be implemented but it has to be done carefully. Occlusion of the brachial artery by the cuff and physiological phenomena mentioned earlier may produce altered PPG waveforms, similar to motion artifacts.

The difference between SBP values detected based on the PPG signal and by the conventional oscillometric algorithm may depend on the speed of inflation. In the present study, we used an inflation speed of approximately 6 mmHg/s, thus, differences shown in the figures can exceed 10 mmHg. Decreasing the speed of inflation can lead to smaller errors if time differences between detection points remain the same. For example, if only one irregular heartbeat occurs delaying SBP detection of the oscillometric method by one heart cycle, the error of the estimated SBP value is smaller if inflation speed is lower. However, if arrhythmia or jerky breathing affect more than one subsequent heart cycles, smaller inflation speed does not necessarily lead to smaller errors.

Personalization is expected to improve the accuracy of SBP estimation. Thresholds for the detection of the disappearance of PPG can be tuned to be patient specific. PWTT is also person specific. If it is used as extra information, initial measurements must be carried out for each patient to determine its normal value in resting state and during cuff inflation. The difference between SBP values during inflation and deflation shows large between-subject variance. The standard deviation of the difference for the clinical recordings was 7.9 mmHg. To use this difference effectively as a sanity check for SBP detection, personalization is needed.

V. Conclusion

Oscillometric blood pressure measurement is a widely used technique with several advantages, but it also has major limitations. In this paper we investigated the accuracy of the conventional oscillometric method using clinical recordings and compared it to SBP estimation based on the PPG signal. Our results show that the difference between SBP values estimated by the two techniques can exceed 10 mmHg and the PPG-based method enables the detection of SBP in situations, when the conventional oscillometric method produces high errors. However, the accuracy of SBP estimation from the PPG signal can also be affected my many factors. Further research work is needed to improve blood pressure measurement based on the PPG signal including both SBP and DBP estimation and personalization.

Acknowledgment

The research has been supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.2-16-2017-00013).

References

[1] C. F. Babbs “Oscillometric measurement of systolic and diastolic blood pressures validated in a physiologic mathematical model”, BioMedical Engineering OnLine, vol. 11, no. 1, 2012, p. 56.

[2] M. Ursino, C. Cristalli “A mathematical study of some biomechanical factors affecting the oscillometric blood pressure measurement”, IEEE Transactions on Biomedical Engineering, vol. 43, no. 8, 1996, pp. 761-778.

[3] G. Drzewiecki, R. Hood, H. Apple ”Theory of the oscillometric maximum and the systolic and diastolic detection ratios”, Annals of Biomedical Engineering, vol. 22, no. 1, 1994, pp. 88-96.

[4] U. Tholl, K. Forstner, M. Anlauf “Measuring blood pressure: pitfalls and recommendations”, Nephrology Dialysis Transplantation, vol. 19, no. 4, 2004, pp. 766-770.

[5] N. M. van Popele, W. J. W. Bos, N. A. M. de Beer, D. A. M. van der Kuip, A. Hofman, D. E. Grobbee, J. C. M. Witteman ”Arterial stiffness as underlying mechanism of disagreement between an oscillometric blood pressure monitor and a sphygmomanometer”, Hypertension, vol. 36, no. 4, 2000, pp. 484-488.

[6] S. Hansen, M. Staber “Oscillometric blood pressure measurement used for calibration of the arterial tonometry method contributes significantly to error”, European Journal of Anaesthesiology, vol. 23, no. 9, 2006, pp. 781-787.

[7] M. Elgendi ”On the analysis of fingertip photoplethysmogram signals”, Current cardiology reviews, vol. 8, no. 1, 2012, pp. 14-25.

[8] M. Nitzan “Measuring systolic blood pressure by photoplethysmography”, U.S. Patent 7 544 168, Jun. 9, 2009.

[9] Y. Yoon, J. H. Cho, G. Yoon ”Non-constrained blood pressure monitoring using ECG and PPG for personal healthcare”, Journal of Medical Systems, vol. 33, no. 4, 2009, pp. 261-266.

[10] Y. Kurylyak, F. Lamonaca, D. Grimaldi ”A Neural Network-based method for continuous blood pressure estimation from a PPG signal”, In: 2013 IEEE International instrumentation and measurement technology conference (I2MTC), IEEE, 2013, pp. 280-283.

[11] H. Shin, S. D. Min ”Feasibility study for the non-invasive blood pressure estimation based on ppg morphology: normotensive subject study”, BioMedical Engineering OnLine, vol. 16, no. 1, 2017, p. 10.

[12] S. S. Mousavi, M. Firouzmand, M. Charmi, M. Hemmati, M. Moghadam, Y. Moghadam “Blood pressure estimation from appropriate and inappropriate PPG signals using A whole-based method”, Biomedical Signal Processing and Control, vol. 47, 2019, pp. 196- 206.

[13] P. Nagy, Á. Jobbágy ”Personalization of the oscillometric blood-pressure measurement”, In: Lhotska, L., Sukupova, L., Lacković, I., Ibbott, G. (eds.) IFMBE Proceedings of the World Congress on Medical Physics and Biomedical Engineering 2018, IFMBE Proceedings, vol. 68/2, 2019, pp. 885-888.

[14] T. G. Pickering ”What will replace the mercury sphygmomanometer?”, Blood Pressure Monitoring, vol. 8, no. 1, 2003, pp. 23-25.

[15] L. A. Geddes, M. Voelz, C. Combs, D. Reiner, C. F. Babbs ”Characterization of the oscillometric method for measuring indirect blood pressure”, Annals of Biomedical Engineering, vol. 10, no. 6, 1982, pp. 271-280.

[16] D. Zheng, F. Pan, A. Murray ”Effect of mechanical behaviour of the brachial artery on blood pressure measurement during both cuff inflation and cuff deflation”, Blood Pressure Monitoring, vol. 18, no. 5, 2013, pp. 265-271.

Modelling a CubeSat-based Space Mission and its Operation

Carlos L. G. Batista*, Fátima Mattiello-Francisco*
* National Institute for Space Research (INPE)
Space Systems Engineering
São José dos Campos, Brazil
Email: {carlos.batista, fatima.mattiello}@inpe.br

***Abstract*—Since the early 2000' years, the CubeSats have been growing and getting more and more "space" in the Space industry. Their short development schedule, low cost equipment and piggyback launches create a new way to access the space, provide new services and enable the development of new technologies for processes and applications. That is the case of the Verification and Validation of these missions. As they are cheaper to launch than traditional space missions, CubeSats win by numbers. With more than 1000 CubeSats launched they still achieve less than 50% rate of successful missions and that is caused mainly by poor V&V processes. Model Based approaches are trying to help in these problems as they help software developers along the last years. As complex systems, space products can be helped by the introduction of models in different levels. Operational goals can be achieved by modeling behavioral scenarios and simulating operational procedures. Here, we present a possible modeling solution using a tool that integrates the functionalities of FSM and Statecharts, the ATOM SysVAP (System for Validation of Finite Automatons and Execution Plans). With this tool we are able to model the behaviour of a space mission, from its top level (i.e. system and segments) to its low level (subsystems) and simulate their interactions (operation). With the help of Lua Programming Language, it is possible to generate analysis files, specific scenarios and control internal variables.**

***Index Terms*—cubesat, model based, operations, finite state machines, verification and validation**

I. Introduction

The innovation through the micro/nano/picosatellites has been creating a new whole market and habitat for the space sector. With a short life cycle, low cost and the use of standardized platforms, these satellites enable the in-orbit qualification of new space technologies, high science and human resources [1]. The CubeSat standard , a.k.a. U-class, has predefined mechanical and electrical interfaces, including launcher interfaces [2]. They brought a new satellite development philosophy, not only for the university satellites but also for all the small-size ones. More than 1300 nanosatellites have been launched in the last fifteen years, 1200 of them are CubeSats and the foreseeable future expect more than 6 thousand in the next six years [3].

The real deal is that in order to develop these missions in universities, reliability requirements, usually specified at traditional space missions, were reduced. Therefore, these satellites have demonstrated high failure rate. Some recent studies [3], [4] pointed out that 50% of the CubeSat-based missions fail, and 70% of them are developed by inexperienced teams. The big difference between the traditional developers and these "hobbyists" is the lack of good practices in design, assembly, and tests. We still have a lack of studies showing us the main problems that cause these missions to fail, especially during the operational phase. For most of them the absence of processes and procedures documentation is the main problem to investigate this kind of operation failure [5].

In this context we can point out the lack of testing methods and operational procedures validated in ground, which are techniques used in the satellite development cycle to support the verification of the expected behavior of the system in the operational environment, increasing the system reliability [6].

The Verification and Validation – V&V – processes are crucial to all space missions to be successful. But for most traditional space missions they are usually onerous in terms of time and money costs, which is incompatible with the U-class development philosophy, "faster and cheaper" [7].

The fundamental for any V&V processes is the system requirements specification. Most CubeSat-based projects have given less importance to the mission requirements definition and analysis phase because the use of the CubeSat standard is considered the architectural solution to the satellite platform. Although CubeSat provides COTS (Commercial Off The-Shelf) subsystems with standardized interfaces and communication protocols, they are not enough to define the mission goals. The mission requirements and constraints are the major elements to drive the use of the COTS in right way, as result of Phase 0 analysis, i.e. mission conception analysis and feasibility.

The mission concept of operation – ConOps – is a discipline in space system engineering that plays a very important role in the mission requirements analysis. ConOps addresses the mission analysis under the user's perspective. It traverses both ground and space segments, aiming to highlight dependencies among mission elements and the space environment to meet the mission operation requirements [8]. The mission ConOps analysis deals with multiple operation scenarios that reveal requirements and needed design functions.

V&V and ConOps are intimately related because the system requirements and specifications derived from the operational needs shall be verified and the intended use of the system shall be validated considering the stakeholders expectations [9], [10].

Modelling the expected behavior of a system for requirements analysis purpose has been helping different engineering areas across the years. Model Based approaches are getting more and more of this piece of the market, including Cube/NanoSats due to its speed, reliability, and reproducibility. Modelling allows to represent the system in different level of abstraction, under different perspective. Depending on the purpose, appropriated modelling formalism is used to explore particular aspect of the system.

Focusing on the concept of operation aspects of a Cubesat-based mission, we present an experience on the use of a modelling tool developed at INPE, by space system engineering student. The tool aids the mission design analysis allowing anticipate requirements verification by means of operational scenarios validation. The purpose is to increase confidence in the mission requirements, reducing the cost of design rework in terms of time and resources consuming.

II. Related Works

New initiatives have been trying to increase the chance of success of CubeSat-based missions using different approaches, mainly Model-Based and Model-Driven approaches. Modelling tools have helped many projects and missions to achieve the confidence on their projects design and development before going to production and operations since the beginning of space racing.

[11] presents the CubeSat Reference Model – CRM – a reference example that is being developed by the INCOSE Space Systems Working Group – SSWG. The goal of the CRM is to facilitate the design, verification and validation of a CubeSat project using SysML. The CRM is being developed flexible enough to be used in different missions and different teams.

Other studies, [12], [13], focus on Model Driven Engineering, aiming to automate the V&V of on-board embedded software. The description of a MDV&V, or Model Driven Verification and Validation, uses SysML to anticipate the platform V&V, suggesting a systemic approach to simulate all the real spacecraft tests. And the usage of a new models philosophy (activity of space product V&V where different models are specified for the space mission, like MockUps, Engineering Model, Qualification Model, Flight Model, etc) with intermediate models, e.g. FlatSats and DevSats [14].

Two highlighted works on this field are: (i) the development of a process and metamodel, in Arcadia Method, using Capella, for the flux of information and functionalities of an spacecraft from its ConOps in order to automatically fulfil the parametric information of a configuration file in a orbiter/operational simulator [15] and; (ii) the proposed Scalable Architecture Test System – SATS –, in order to support the V&V, combine MDE and MBT to anticipate the possible problems of interoperability and robustness [16].

III. Modelling a Space Mission

Aiming at representing the behavior of Cubesat-based mission under the perspective of operation for studies purpose concerning operation scenarios telecommanded from ground stations, the ATOM SysVAP modelling tool was used. Four main reasons support the choice: (i) the tool was developed in-house. ATOM SysVAP is a result from from an INPE's master student work; open source and available online in a git repository; (ii) this tool has been used in graduation class, at INPE, to aid students on the specification of Operational Procedures, visualizing their execution on an operational simulator (iii) the use of Lua Programming scripts creates a new level of modeling where it is possible to associate full operational scripts to different states and events and; (iv) the use of FSM is more simple and easy to deal and interfacing it with other modelling and model checking tools, if necessary.

A. ATOM SysVAP

The ATOM SysVAP – Atom SYStem for Validation of finite Automatons and execution Plans – is a tool developed to validate an architecture proposed to execute execution operational plans [17].

The ATOM enables the modeler to: (i) represent the Transition and States Domain as Mealy, Moore Machines and Statecharts; (ii) represent sub-FSM as part of other FSM; (iii) execute and validate operational plans in debug/real time mode and; (iv) access to environmental variables in order to evaluate the results and determine if the operational plan can or cannot be accepted.

Also, the ATOM uses the Lua Programming language for the development of the Domain Language. This capability supports the model execution offering new facilities for the data description.

B. Elements of the Model

The element of the modelled space systems follows the hierarchical sequence presented at the Figure 1.

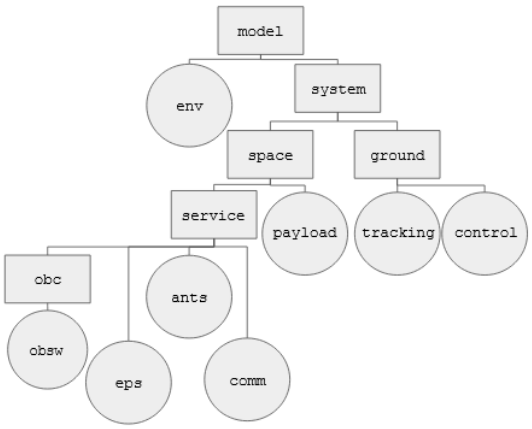


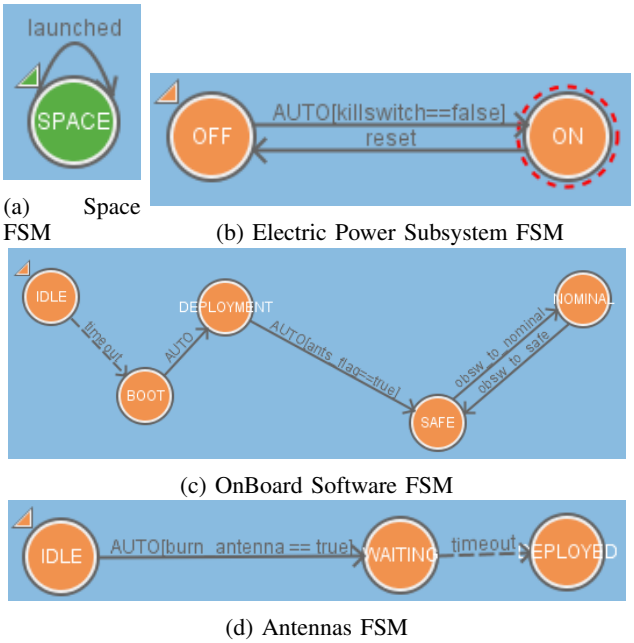
Fig. 1: Model breakdown hierarchy

At Figure 1, the rounded shapes represent FSM without sub-FSM (bottom level of abstraction) as the square shapes represent FSM that can be grained in order to lower the level of abstraction.

From the top level, the FSM "model" represents the model as a whole, their sub-FSM are representations of what is inside the system to be modeled, "system", and what is outside but still interferes with the system, the "environment". As a space system, we are able to divide into two other sub-FSM, "space segment" and "ground segment". This separation is common at space missions to more easily organize the attributions, needs and capabilities of each one. At this point the ground segment can be divided into "tracking", a model to deal with the pointing and passover constraints of ground stations and satellite, and "control", the model responsible to deal with the operation of the satellite, this model can represent specific operational plans as multiple FSM. Similarly, the space segment can be divided into the "service" and "payload" buses. As a service bus, the platform is divided again into the different subsystems. As an example of granularity, the onboard computer model can be splitted into a sub-FSM to represent only the logical functioning of the embedded software, "obsw".

Considering the launcher requirements for CubeSats, (a) the satellite must be in "off" state while inside the dispenser; (b) the satellite must have a device to ensure the batteries are disconnected to the power system, a killswitch, while inside the dispenser and; (c) the satellite must not enter any operational state or boot mode for at least 30 minutes after the launch from the dispenser.

So, the Figure 2 represents the models used for this situation, once the space segment receive the event "launched" (fig. 2a), the variable "killswitch" turns false, activating the event at the EPS – Electric Power Subsystem – model (fig 2b). The powering on of the EPS activates two events in two different FSM: (i) the OBSw – On Board Software – FSM starts a countdown of 30 minutes to start the booting process, BOOT state, (fig 2c) and; (ii) at the Antennas Subsystem, the deployment process of the antennas is started also (fig.2d). Only after the 30 minutes, the OBSw is able to go to the DEPLOYMENT state, where it awaits for the state of the antennas (deployed or not, i.e. ants_flag variable) changing its state to SAFE. DEPLOYMENT, SAFE and NOMINAL are states considered as operational for satellite already.



(a) Space FSM

(b) Electric Power Subsystem FSM

(c) OnBoard Software FSM

(d) Antennas FSM

Fig. 2: Examples of modelled FSM

C. Analysis Files

Besides the modelling capabilities of the ATOM SysVAP tool, one great advantage and feature of the IDE is the integration with the Lua Programming Language. With that we are capable of reading and writing "comma separated values" files, a.k.a. csv files.

Once we have modelled the behaviour of the intended space system, some routines and functions in Lua can be used to generate analysis files, as csv files. These analysis files represent the internal variables that we want to monitorate, such as telemetries and more.

For example, Figure 3 shows an analysis file, read by a python notebook, with information about the spacecraft battery voltage level, the subsystems FSM actual states, and even data about a reliability model implemented and the number of faults in the system, also implemented due to the Lua capabilities.

	time(s)	battery	orbit	obc	obsw	eps	antennas	communication	total_drop	reliability	fault_count
4031	4032	19.201067	ECL	ON	SAFE	ON	DEPLOYED	ON	0.000533	0.151998	3001
4032	4033	19.200533	ECL	ON	SAFE	ON	DEPLOYED	ON	0.000533	0.151947	3002
4033	4034	19.200000	ECL	ON	SAFE	ON	DEPLOYED	ON	0.000533	0.151896	3003
4034	4035	19.199467	ECL	ON	SAFE	ON	DEPLOYED	ON	0.000533	0.151844	3004
4035	4036	19.198933	ECL	ON	SAFE	ON	DEPLOYED	ON	0.000533	0.151793	3005

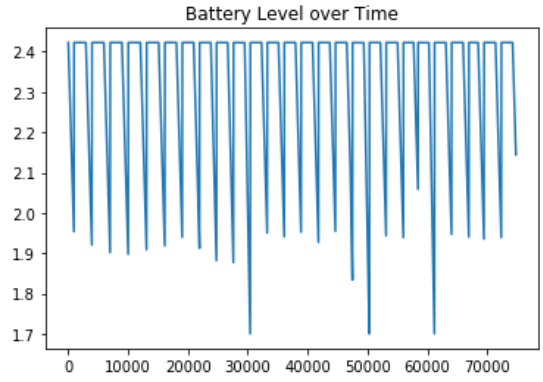
4036 rows × 11 columns

Fig. 3: Example of analysis file opened in Jupyter Notebook

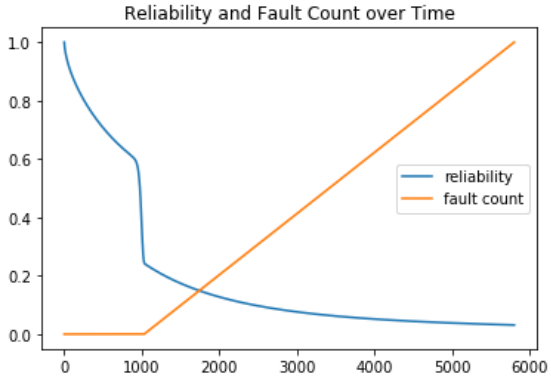
The generation of these analysis files can help the development team to check the behaviour of specific variables of the spacecraft and how to control them and also help the operational team to understand them even during the operational phase.

Figure 4 helps us to understand some graphs that can be plotted in order to evaluate the behaviour of some variables. At Figure 4a we can access information about the battery level as each state of the environment changes (i.e. sun or eclipse) and if a certain number of subsystems are powered or not. As this, we can monitor the depth of discharge of the battery and simulate different conditions of operation taking into account the power budget. Figure 4b represents a derived comparison, from a fault model minimally implemented from a reliability model (a Weibull distribution). In this case, the chance of a fault occurrence, at the system level, is calculated as a normal distribution from 0 to 1. Once the chance of a fault is less than the reliability level at the same point in time, a fault occurs.

That is a simple fault model implemented without any fault behaviour associated with.



(a) Battery Level



(b) Reliability and Fault Count

Fig. 4: Examples of graphs derived from the analysis file

IV. Conclusions and Future Work

As we evolve the technologies for the access of space, the processes, procedures, activities and methods should evolve together.

We presented an experience on the use of a modelling tool, ATOM SysVAP, to represent the behavior of Cubesat-based mission, under operational perspective, for the purpose of the verification and validation as a point of view from ConOps. The purpose is to highlight the dependencies among the major elements of the mission and space environment under perspective of the mission operation, which contributes to increase confidence on the definition of system requirements, reducing the project development cycle and the time spend during V&V processes.

The ATOM SysVAP has demonstrated, so far, its capability to, not only, model the operational phase of a space system, but also help the conception, development, and V&V (validation of specific FSM paths). With the use of Lua as scripting language, the ATOM exploits its usage being capable of generating telemetry files, payload data, mathematical models, evaluating behaviour in real time (debug) and, modeling faults (using fault/error/failure behaviours and scenarios). This capability enables the FSM to be more reduced and the models simpler.

Moreover, the method used at this point is not completely tool independent. Based on the capabilities of the ATOM SysVAP, the models were evolved and implemented in order to represent the space mission elements and behaviour. Compared to other approaches, this paper does not goes too much far away. FSM is a well known modeling method and scripting languages are being used for a long time in the context of repetitive activities. The use of both methods together in the same tool enables the conception of a single modeling environment that can be shard during the space product life cycle (from concept to operations).

The idea for the future of these models is to evolve them in order to model the behaviour for different operational plans/situations and being capable of modeling faults at the subsystem level. Some studies are being developed to use the ATOM to simulate the operational procedures of the NanoSatC-BR2, 2U CubeSat developed by INPE to be launched in March/2021. These models will help the Operational Team, at the Ground Stations, to be trained to deal with possible situations that can happen at the in-orbit phases and to analyze the autonomous operation of the payloads on board of the nanosatellite.

References

[1] G. Martin, "Newspace: The emerging commercial space industry," 2015.

[2] H. Heidt, J. Puig-Suari, A. Moore, S. Nakasuka, and R. Twiggs, "Cubesat: A new generation of picosatellite for education and industry low-cost space experimentation," 2000.

[3] T. Villela, C. A. Costa, A. M. Brandão, F. T. Bueno, and R. Leonardi, "Towards the Thousandth CubeSat: A Statistical Overview," *International Journal of Aerospace Engineering*, vol. 2019, pp. 1–13, jan 2019.

[4] M. Swartwout, "Secondary spacecraft in 2016: Why some succeed (And too many do not)," in *2016 IEEE Aerospace Conference*. IEEE, mar 2016, pp. 1–13. [Online]. Available: http://ieeexplore.ieee.org/document/7500791/

[5] C. L. G. Batista, T. Basso, F. Mattiello-Francisco, and R. Moraes, "Impacts of the space technology evolution in the V&V of embedded software-intensive systems," *arXiv preprint arXiv:2011.14914*, 2020.

[6] L. Berthoud, M. Swartwout, L. Blvd, S. Louis, J. Cutler, and D. Klumpar, "University CubeSat Project Management for Success," 2019.

[7] K. Forsberg and H. Mooz, "System engineering for faster, cheaper, better," in *INCOSE International Symposium*, vol. 9. Wiley Online Library, 1999, pp. 924–932.

[8] W. J. Larson and J. R. Wertz, *Space mission analysis and design*, 1999, vol. 29, no. 09.

[9] NASA, *NASA System Engineering Handbook Revision 2*, 2016.

[10] ECSS, "ECSS-E-HB-10-02A Space engineering: Verification guidelines," *ECSS Standard*, no. December, 2010.

[11] D. Kaslow and A. M. Madni, "Validation and Verification of MBSE-Compliant CubeSat Reference Model," in *Disciplinary Convergence in Systems Engineering Research*, no. January, 2018, pp. 1–1201.

[12] S. A. Jacklin, "Survey of Verification and Validation Techniques for Small Satellite Software Development," pp. 138–147, 2017.

[13] M. Omair Khan, M. Sievers, and S. Standley, "Model-based verification and validation of spacecraft avionics," *AIAA Infotech at Aerospace Conference and Exhibit 2012*, pp. 1–11, 2012.

[14] J. Eickhoff, R. Hendricks, and J. Flemmig, "Model based development and verification environment," in *54th IAC of the IAF*, vol. 3. AIAA, dec 2003, pp. 1275–1285.

[15] D. P. d. Almeida, F. Mattiello-Francisco, and F. L. d. Sousa, "Towards modeling the concept of operations for cubesat-based missions," X WETE. São José dos Campos: INPE, 2019.

[16] C. A. C. Conceicao, F. Mattiello-Francisco, and C. C. L. Batista, "Dependability Verification of Nanosatellite Embedded Software Supported by a Reusable Test System," in *LADC*. IEEE, oct 2016, pp. 157–163.

[17] A. A. S. Ivo, "A tool for evaluating satellite flight plans using state models," 2013.

Model-Driven Development of Heterogeneous Cyber-Physical Systems

János Csanád Csuvarszki, Bence Graics, András Vörös
Budapest University of Technology and Economics,
Department of Measurement and Information Systems
Budapest, Hungary
Email: csanad.csuvarszki@outlook.com, {graics, vori}@mit.bme.hu

***Abstract*—Cyber-physical systems (CPS) are becoming more prevalent for the reliable execution of critical tasks, e.g., in the aerospace and medical fields. The development of such systems is challenging, due to the heterogeneity of the components ranging from sensors and embedded controllers to cloud solutions. Model-driven approaches can provide a high-level abstraction to combat the challenges. This paper proposes a model-driven development approach supporting the precise modeling of CPS and the automatic derivation of implementation from the resulting models. The approach introduces a composition semantics tailored to heterogeneous architectures for the precise description of component interactions. Automated code generators allow the execution of standalone software components on real-time controllers and support the synthesis of standalone hardware components, enabling both the derivation of embedded software and the description of logical circuit behavior. Component interactions are supported by multiple communication solutions generally used in critical, real-time embedded systems. The approach is implemented in the Gamma framework and its applicability is demonstrated in two case studies.**

I. Introduction

Cyber-physical systems (CPS) [1] are present in numerous industrial fields, e.g., control systems in automotive, aerospace and healthcare applications. These complex systems not only gather data from their surroundings, but they actively alter the physical world to achieve certain goals. CPS can consist of many heterogeneous components, such as microcontrollers with sensors and actuators, and can connect to processes running on other embedded devices. This heterogeneity makes the development and analysis of CPS a challenge.

This complexity can be handled by introducing a model-driven development approach, which focuses on the high-level description of the system. This allows developers to focus on the structure, behaviour and communication of the components instead of low-level implementation details. Model-driven approaches can support verification and validation (V&V) techniques, and based on a sufficiently detailed model, automatic code generation is also feasible.

However, most modeling tools do not provide support for the design of heterogeneous systems, which would require the systematic integration of models describing the behavior of standalone components. Furthermore, the automatic code generation for different hardware components is rarely supported. Therefore, approaches aiming to support the model-driven design of cyber-physical systems have to support *1)* the integration of components with well-defined execution and interaction semantics tailored to heterogeneous systems and *2)* automatic code generators for different software and hardware platforms, and communication modes. To solve this, we propose a solution in the context of the Gamma framework.

The Gamma Statechart Composition Framework [2] is an integrated modeling tool for the semantically well-founded composition and analysis of statechart [3] components. Gamma offers a UML-influenced [4] statechart language to define atomic component behaviour with complex constructs, e.g., orthogonal regions and history states. At its core, it provides a composition language supporting the hierarchical integration of components with precise semantics [5]. Gamma supports system-level formal V&V based on formal models of various model checkers and back-annotating the results. The framework provides test generation functionalities and automated code generators for both statechart and composite models.

In this work, we extend Gamma to support *1)* the modeling of data-centric communication, *2)* the generation of implementation in C and SystemVerilog and *3)* communication between component implementations via DDS or shared memory.

II. Model-Driven Development of CPS

We integrated our model-driven development approach into the Gamma framework in the form of three contribution groups (depicted in Fig. 1), that is, the modeling of composite CPS (Sec. II-A), the derivation of standalone reactive component implementations (Sec. II-B) and supporting the communication of the derived implementation (Sec. II-C). The applicability of our contributions is demonstrated using a running example (Sec. II-D).

A. Modeling Composite CPS

In Gamma, statechart components can be hierarchically composed to create composite components. Components inside such compositions receive inputs and produce outputs through *ports*. These input and output ports can serve as composite system ports, receiving from and producing to outside

This work was supported by the ÚNKP-20-3 and 4-II New National Excellence Program of the Ministry for Innovation and Technology and the EU ECSEL JU under the H2020 Framework Programme, JU grant nr. 826452 (Arrowhead Tools project) and from the partners' national funding authorities.

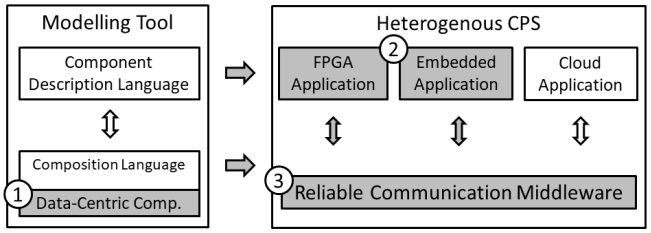


Fig. 1. The steps of our CPS-development approach introduced to Gamma

sources. Components inside a composition use *channels* to connect their ports and communicate with each other.

Contribution 1 introduces a new asynchronous component type, the *data-centric component* to the composition language of Gamma, which uses *data* as a means of communication, a common characteristic of components in heterogeneous CPS. As this paper focuses on the overall model-driven development approach, we give only an overview of data-centric component, and do not include its formal definition.

Data-centric components share data using shared variables, which are modeled by *event declarations* having only *one parameter*: sending and receiving an event with a parameter is equivalent to writing and reading a variable in this case. Currently, each shared variable can have one well-defined *reader* (as a specific event declaration is owned by a single component) and one or more *writers* (other components that can send the specific event via a channel). During execution, if there are multiple writes to the same variable (instances of the same event can be sent multiple times to the same component), always the latest will prevail. This semantic variant does not restrict the running frequency or runtime of components, they run independently of each other.

B. Supporting New Architectures and Platforms

Contribution 2 (detailed in Sec. III) supports the *generation of implementation for embedded controllers and FPGAs* by introducing new code generators to Gamma that rely on the framework's statechart and composition languages to create implementation from state-based models. Currently, Gamma supports the derivation of Java implementation, however, in the case of heterogeneous CPS, there is a need to support code generation for embedded devices. To achieve this, the newly introduced code generators derive implementation in C and SystemVerilog languages. C is one of the most popular languages when developing embedded systems. SystemVerilog is often used in FPGA development, and has high-level language elements that facilitate code generation.

C. Establishing Communication

Contribution 3 (detailed in Sec. IV) introduces RTI DDS, an implementation of the Data Distribution Service standard [6] to Gamma to *establish reliable communication between distributed components*. DDS uses a publish-subscribe model to share data on different topics and operates with a global data space, acting as a local memory that can be accessed via an API. Applications read from and write to these local memories, and the DDS sends messages to update the memories of remote nodes. These local stores supports the applications to the access a global data space. DDS also provides QoS options to increase the reliability of the communication.

To implement the communication between Gamma components, the previously presented C code generator as well as the code generator of RTI DDS is used.

D. Running Example: Crossroads Traffic Controller

We present our approach in the context of a running example involving the design and development of a distributed *crossroads traffic controller* system, presented in the official Gamma tutorial.[1] The crossroads traffic controller system is modeled as a composite component consisting of three atomic statechart components, a *central controller* and two *traffic lights* controlled by the central controller.

The first case study presents *shared memory communication* using a Digilent ZedBoard[2], which consists of a dual-core CPU and a Xilinx FPGA (found inside a Zynq-7000 SoC). The controller runs on the CPU as software, while both traffic lights are synthesised on the FPGA as hardware.

The second case study presents the *distributed communication* of components using DDS. The controller and the traffic lights run in different processes and communicate via RTI DDS. This case study presents our approach both in terms of the operation of standalone HW and SW components and the distributed communication of such heterogeneous components.

III. Modeling and Code Generation for Standalone Hardware

We integrated our modeling and code generation approach into the Gamma transformation chain depicted in Fig. 2.

As an optional step, statechart models created in integrated modeling tools (such as MagicDraw[3] and Yakindu[4]) can be imported by executing model transformations that map these models into the Gamma Statechart Language. This language provides full support to describe self-contained control-oriented, reactive behavior with high-level statechart constructs (e.g., variables, orthogonal regions, history states and complex transitions) and simple action language constructs (e.g., assignments, branches and fixed-iteration loops) and supports data-intensive behavior with manually fillable stubs.

Gamma statecharts can be hierarchically composed according to multiple precise execution and interaction semantics, e.g., asynchronous-reactive, synchronous-reactive and cascade [5]. Both atomic statecharts and composite components can be mapped to models of a low-level transition systems formalism, called EXtended Symbolic Transition Systems (XSTS), serving as input to our introduced code generators.

An XSTS model describes state-based behavior based on *variables* of type boolean, integer or enum, describing system states, and *transitions* specifying possible changes in the state.

[1]The tutorial can be found at http://gamma.inf.mit.bme.hu.
[2]http://zedboard.org/product/zedboard
[3]https://www.nomagic.com/products/magicdraw/
[4]https://www.itemis.com/en/yakindu/state-machine/

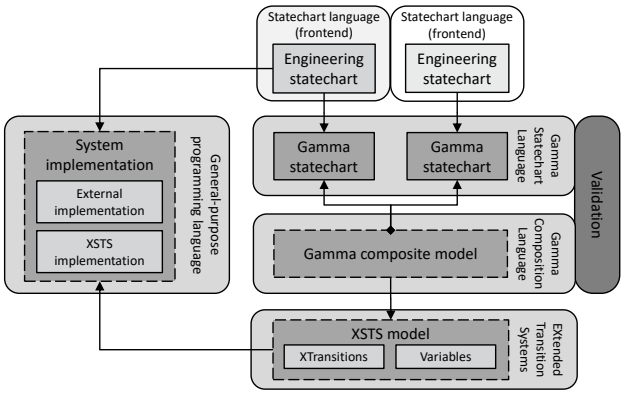


Fig. 2. The model transformation chain for code generation in Gamma

The high-level constructs of (composite) statechart models are mapped to one or more variables. For example, an *event declaration* is transformed into a boolean variable denoting the occurrence of the event and optionally more variables according to its parameters; a *region* is mapped to an enum variable with literals denoting the contained states (which can also be composite). Transitions can use atomic actions, such as assignment actions or assume actions (assumptions that must hold in order to execute specific action branches), which can be reused to construct composite actions, such as sequential actions (i.e., blocks) or deterministic choices.

In our C and SystemVerilog code generators, we map the variables and low-level transition constructs of the XSTS model to constructs of the respective languages (an overview of the whole code generation process is depicted in Fig. 3).

A. C Code Generation

When generating code from a single statechart, multiple source code components are generated. At its core, the derived implementation contains a structure that encompasses all the variables associated with the model: states, non-timeout and timeout events. Multiple functions are generated as well, which perform operations on the structure that contains the model variables. A wrapper structure is also introduced, which is used to implement timing mechanisms and callback functions. This wrapper contains an instance of the statechart implementation.

Generating implementation for composite components produces files the same way as generating for standalone components does, and functions identically to them. The difference is that structures contain variables from all statechart components, and functions execute operations on the whole system.

B. SystemVerilog Code Generation

In SystemVerilog, Gamma components are described as modules. Modules have input and output signals that can be connected to external sources such as GPIOs or other modules when instantiated; the input and output events of components are realised using such input and output signals. The input signals are processed inside the module, where the behaviour is described, and output signals are produced when the corresponding events are raised. Inside the module, the

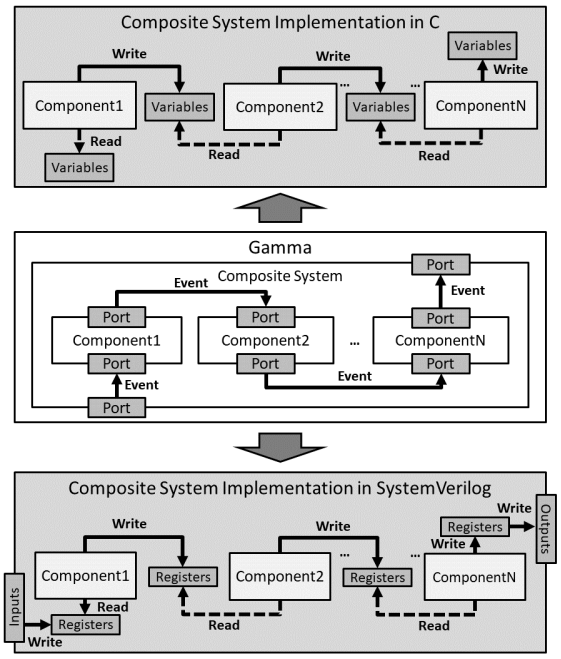


Fig. 3. An overview of the code generation in C and SystemVerilog

enumerations representing the states are generated as well. Unlike the C source code, the SystemVerilog implementation does not utilise functions to operate the model. Instead, it does every action in a clock-driven loop, changing the state of the model, and incrementing timeout variables.

C. Example: Digilent ZedBoard

In this example, the implementations of the controller and the traffic lights of the crossroads traffic controller were generated using the C and SystemVerilog code generators.

The traffic light implementations are generated in SystemVerilog from the Gamma statechart model. Then, using Vivado Design Suite, the implementation is packaged as an AXI4 slave peripheral. The traffic light IPs are then connected to the Zynq-7000 SoC and the design is exported as hardware, serving as a target platform for the next step where the controller is implemented. The controller implementation is generated in C from its Gamma model. Then, the controller is instantiated, gets reset, and is placed in a loop, where it is executed. These steps happen in the context of the previously exported target hardware, using Xilinx Vitis. Finally, the system is loaded on the Zynq-7000 SoC, where the controller uses memory mapping to write to the addresses of the traffic lights, signaling them to switch their states.

IV. Code Generation for Distributed CPS

We introduce an RTI DDS code generator for Gamma asynchronous composite components realising the data-centric, distributed communication of composite model implementations.

A. Sharing Data with DDS

Generating the implementation of DDS communication is currently supported in C for asynchronous composite compo-

nents containing data-centric components. In addition to the component implementation, RTI DDS is also required to implement DDS-based communication. First, an IDL (Interface Description Language) file is generated from the composite system, containing a data type for each channel (port-port connection), system input and system output in the composition. Publishers and subscribers are generated for each component in the composite system. Finally, using the code generator of RTI DDS, stubs are generated from the IDL file, creating the files to implement the generated publishers and subscribers.

For each channel, a data type is generated, which is used by the corresponding readers and writers. Components have one publisher and one subscriber. Publishers serve as the output of the component, while subscribers handle inputs (Fig. 4 presents an overview of this).

B. Example: RTI Connext DDS

To realize DDS communication , the statechart implementation wrapper of the component is placed in a loop, where it is repeatedly executed. When its subscriber receives data, it passes it to the statechart implementation. The statechart processes the input, executes a step, and if possible, produces an output. The output is passed to the corresponding publisher, which publishes the data to other component subscribers.

The implementation of the controller and traffic light components is generated via the introduced C code generator. Then, from the composition, the necessary files are generated to establish communication via DDS. From the generated IDL, the necessary stubs are created using the code generator of RTI DDS. Finally, the project is built, and the processes of components are executed. As a result, the components communicate via RTI DDS, behaving according to their statechart model.

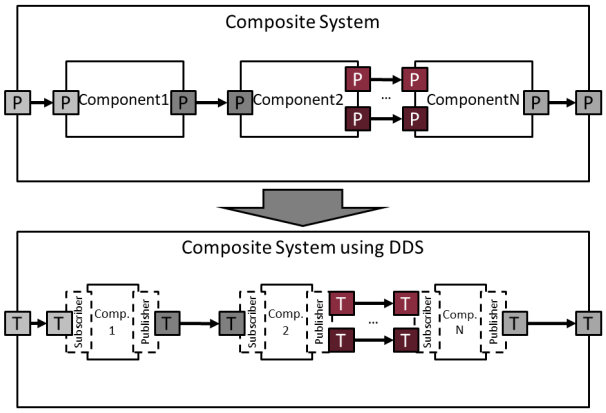


Fig. 4. An overview of the implementation of DDS communication (P stands for port, T stands for topic)

V. Related Work

This section presents solutions addressing the model-driven development of heterogeneous CPS that provide similar approaches in modeling and generating implementation.

ThingML In [7], a tool supporting a model-driven software engineering approach is presented, focusing on the heterogeneity and distribution challenges of cyber-physical systems. The ThingML language uses composite state machines to model component behaviour and supports asynchronous communication while supporting the generation of implementation and communication. Contrary to Gamma, ThingML does not support the semantically sound mix-and-match composition of components according to various execution and interaction semantics (synchronous and asynchronous), code generation for hardware description languages and formal verification.

xMAS In [8], a set of microarchitectural primitives is defined, allowing for the description of complete systems by composition alone. It focuses on the efficient system-level connection of different IP blocks, which is an integral part of SoC design. The article proposes an approach based on creating executable and analyzable high-level models, called xMAS (eXecutable Microarchitectural Specification) models. For formal verification purposes, it generates Verilog out of the models and uses inhouse and academic tools. This approach focuses on defining models on a microarchitectural level, rather than defining high-level behaviour models, like Gamma.

VI. Conclusion and Future Work

In this paper, we presented extensions to the Gamma framework that enable the model-driven development of heterogeneous CPS. We introduced a *data-centric communication semantics* for the precise description of communication between modeled components, *code generators* to support the development of multiple embedded platforms as well as a *middleware-based communication* solution to enable the distributed functioning of components.

In the future, we plan to extend the architectural modeling capabilities of Gamma. By defining the target platforms, the framework could allocate components to the specified targets, automatically creating corresponding implementation.

References

[1] N. I. of Standards and Technology, "Framework for cyber-physical systems: Volume 1, Overview," 2017, http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-201.pdf.

[2] V. Molnár, B. Graics, A. Vörös, I. Majzik, and D. Varró, "The Gamma Statechart Composition Framework: design, verification and code generation for component-based reactive systems," in *Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings*. ACM, 2018, pp. 113–116.

[3] D. Harel, "Statecharts: A visual formalism for complex systems," *Sci. Comput. Program.*, vol. 8, no. 3, pp. 231–274, Jun. 1987.

[4] "OMG Unified Modeling Language (OMG UML), Superstructure," pp. 525–582, 2009, https://www.omg.org/spec/UML/2.2/Superstructure/PDF.

[5] 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, pp. 1483 – 1517, 2020.

[6] "What is DDS?" https://www.dds-foundation.org/what-is-dds-3/.

[7] F. Fleurey and B. Morin, "ThingML: A generative approach to engineer heterogeneous and distributed systems," in *2017 IEEE International Conference on Software Architecture Workshops (ICSAW)*, pp. 185–188.

[8] S. Chatterjee, M. Kishinevsky, and U. Y. Ogras, "xMAS: Quick formal modeling of communication fabrics to enable verification," *IEEE Design Test of Computers*, vol. 29, no. 3, pp. 80–88, 2012.

Towards Interactive Learning for Model-based Software Engineering

Áron Barcsa-Szabó, Balázs Várady, Rebeka Farkas, Vince Molnár, András Vörös
Budapest University of Technology and Economics
Department of Measurement and Information Systems
Budapest, Hungary
Email: $\{abarcsa, balazs.varady\}@edu.bme.hu, \{farkasr, molnarv, vori\}@mit.bme.hu$

***Abstract*—Model-based technologies improve the efficiency of the design and development of IT systems by making it possible to automate verification, code generation and system analysis based on a formal model. A simple way of describing the behavior of systems is state-based modeling, which - due to the advancements of formal analysis techniques in recent years - can be widely and effectively utilized when analyzing systems. A possible way of synthesizing such models is to apply active automata learning algorithms. Acquiring a correct formal model of a system can be challenging because of the different theoretical and practical obstacles of both manual and automated approaches. We propose a semi-automated solution, that applies automata learning to provide an interactive environment for model development.**

***Index Terms*—Automata Learning, Formal Methods, Model-based Software Engineering, System Design**

I. Introduction

Model-based engineering is the formalized application of modeling during system design and development. It improves the efficiency of designing and developing IT systems by formalizing verification, code generation and system analysis, and in certain cases enables their automation as well. Such models can be designed both manually and in automated ways, both of which have their disadvantages. On one hand, it is difficult for the designing engineer to keep every property of the envisioned system in mind at a given time, partly because of the complexity of the system, and because of possible hidden implications and contradictions. Additionally, to conveniently specify requirements, different scopes, abstractions and formalisms may be applied, which may be difficult to reconcile. On the other hand, there are fully automated solutions – usually stricter in these aspects –, for instance, active automata learning, where the model construction is characterized by a teacher component - which is familiar with the extensive behavior of the system under learning - and a learner component – which synthesizes the model via queries to the teacher component. However, such solutions have practical boundaries when validating the inferred behavior of the system. The objective of our work is to support the design of systems and components from the ground up through a semi-automated solution – *InterActive* automata learning – which utilizes both the frequent input of the designing engineers and automated techniques. As a result of this approach, the users themselves are regarded as the teacher component of the algorithm, resolving the infeasibility of automated equivalence validation. This results in a semi-automated solution driven by declarative behavioral requirement specification, which allows the designing engineers to focus on the expected behavior of the system and on evaluating the model proposed by the algorithm. This paper presents an adaptive state-based modeling framework combining the advantages of manual and automated solutions, into which we designed and integrated the interactive algorithm. We created a proof-of-concept implementation of the approach, allowing system design through different formalisms, and the analysis and development of interactive automata learning algorithms to support model-driven development with an extended scope.

This work was partially supported by the ÚNKP-20-4-II New National Excellence Program of the Ministry for Innovation and Technology and by the National Research, Development and Innovation Fund of Hungary, financed under the 2019-2.1.1-EUREKA-2019-00001 funding scheme.

II. Background

A. Model-Based Engineering

Due to the application of the modeling concept in several completely different domains, first of all, we need to define the meaning of *model*.

Model A model is the simplified image of an element of the real or a hypothetical world (the system), that replaces the the system in certain considerations. Two main types of models exist in the literature: structural and behavioral.

Model-Based Systems Engineering (MBSE) is the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases [4]. This concept can also be applied to software engineering. Note, that the models may be the primary artifact of the development process, in which case precisely defined formal models are required. When the models are the primary artifacts, the process is called *Model-Driven Engineering*.

In this paper, our goal is to assist *MBSE* through defining a new metodology for requirement-driven **component** design.

B. Specifying Requirements

During our work, the concept of requirements is used widely, therefore, it is essential to define it precisely.

Requirement [1]

1) A condition or capability needed by the user to solve a problem or achieve an objective.
2) A condition or capability that must be met or possessed by a system component to satisfy a contract, standard, specification or other formally imposed documents.
3) A documented representation of a condition or capability as in (1) or (2).

Requirements are important, as both system design and verification depend on them. They can be specified in many different ways, the most common being textual requirements in traditional feature lists – an informal way of requirement specification.

The rationale behind the precise formalization of requirements is the wide range of automated applications, especially in *formal methods* – such as validation, formal verification or test generation. One possible requirement formalism is Linear-Time Temporal Logic (LTL) expressions over paths of automata models, resembling propositional logic extended with temporal connectives – e.g. X (true in the ne*X*t state), F (at some point in the *F*uture), G (*G*lobally) or U (*U*ntil a certain condition is met). One possible application of LTL is the transformation to Büchi-automata (or in general ω-automata), which then can be utilized in *formal verification* [3].

C. Automata Learning

Due to our solution utilizing automata learning to synthesise formal models, here we define its essential concepts.

Mealy machine A Mealy machine or Mealy automata is a Tuple of $M = (S, s_0, \Sigma, \Omega, \delta, \lambda)$, where:

- S is a finite, non-empty set containing the states of the automata,
- $s_0 \in S$ is the initial state,
- Σ is the input alphabet of the automata,
- Ω is the output alphabet of the automata,
- $\delta : Q \times \Sigma \to Q$ is the transition function and
- $\lambda : Q \times \Sigma \to \Omega$ is the output function.

A commonly used semantic helper can be defined over Mealy machines as $\lambda^* : S \times \Sigma^* \to \Omega$, defined by $\lambda^*(s, \epsilon) = \varnothing$ and $\lambda^*(s, w\alpha) = \lambda(\delta^*(s, w), \alpha)$, essentially providing the output of the automata after running an input sequence from a specified state.

Automata Learning is a way of modeling a system as an automata through observations without having specific knowledge of its internal behavior.

Formally: Automata learning is concerned with the problem of inferring an automata model for an unknown formal language L over some alphabet Σ [5].

Two types of automata learning approaches exist in the literature.

*a) **Passive Automata Learning**:* In case of passive automata learning, the gathering of information is not part of the learning process, but rather a prerequisite. The learning is performed on a pre-gathered positive an/or negative example set of the systems behavior. In passive automata learning, the success of the process is affected greatly by the methodology and time used to gather the data.

*b) **Active Automata Learning**:* In case of active automata learning, the behavioral information is gathered by the learning algorithm via queries. Active automata learning follows the Minimally Adequate Teacher (MAT) model proposed by Dana Angluin [2], which defines the separation of the algorithm to a teacher and a learner component in a way, where the teacher can only answer the minimally adequate queries needed to learn the system. These two queries are described in the following.

Membership query Given a $w \in \Sigma^*$ word, the query returns the $o \in \Omega$ output o corresponding to it, treating the word as a string of inputs. We write $mq(w) = o$ to denote that executing the query w on the system under learning (SUL) leads to the output o: $[\![\text{SUL}]\!](w) = o$ or $\lambda^*(s_0, w) = o$.

Equivalence query Given a hypothesis automata M, the query attempts to determine if the hypothesis is behaviorally equivalent to the SUL, and if not, finding the diverging behavior, and return with an example. We write $eq(H) = c$, where $c \in \Sigma^*$, to denote an equivalence query on hypothesis H, returning a counterexample c. The counter example provided is the sequence of inputs for which the output of system under learning and the output of the hypothesis differ: $[\![H]\!](c) \neq mq(c)$.

The learner component uses membership queries to construct a hypothesis automata, then refines this hypothesis by the counterexamples provided by equivalence queries. Once counterexamples can not be found this way, the learners hypothesis is behaviorally equivalent to the SUL. The learning can terminate and the output of the learning is the current hypothesis.

D. Simple Case Study

In order to illustrate the application of the proposed learning algorithm, we iterate through a running example of an engineer utilizing it to model a composite system of an intersection with three distinct components: a traffic light, a pedestrian light and a controller component. For the sake of conciseness, we only showcase the capabilities of the framework and omit certain steps of the modeling process.

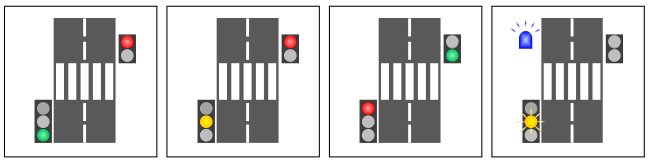


Fig. 2. Possible states of the system: normal operation and the interrupted state.

The functionality of the modeled system is simple: the traffic light loops through the red-green-yellow cycle, while the pedestrian light has a red-green cycle, as visualized on Figure 2. A controller component is responsible for their safe synchronization. Also, the police can interrupt the lights to be

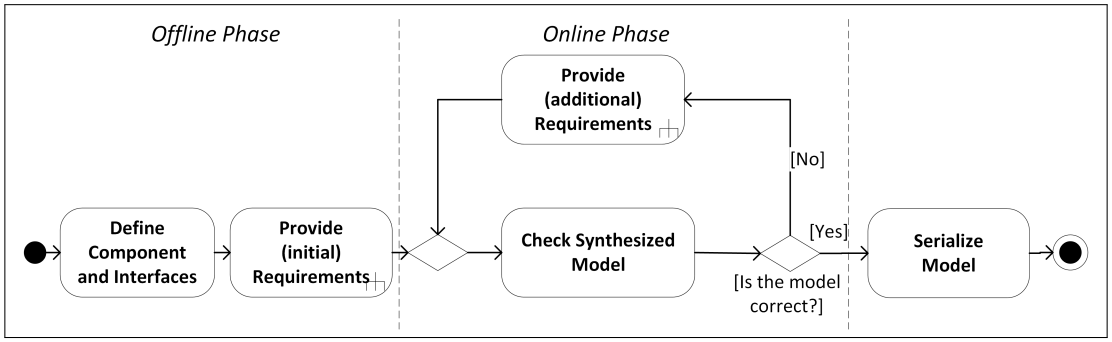


Fig. 1. The Proposed Workflow.

able to pass swiftly, in which case the traffic light switches to blinking yellow, and the pedestrian light switches off. After the police leaves the intersection, the lights return operation through a safe (red) state.

III. Overview of the Approach

Our methodology is heavily based on the interactions of the user with the proposed ***Interactive Learning Entity*** or **ILE** – characterized by the interactive automata learning framework. As Figure 3 illustrates, the two types of queries asked by the ILE are equivalent to that of active automata learning. In contrast to automated equivalence queries – where a non-approximate guarantee of correctness is infeasible – when delegated to a designing engineer, the only obstacles in correctness are the design skills of the engineer. Since no active automata learning algorithm exists in the literature which can learn interactively, while adapting to different types and the changing of reqirements, the ILE encompasses a new interactive algorithm to learn, paired with an oracle to adapt and keep the requirements consistent. The interactions with the ILE take place in a predefined order – the *proposed workflow*, illustrated on Figure 1. The workflow consists of two phases: first, an *offline* one, then an *online* one, and ends with the serialization of the models. During the offline phase, the ILE offers little assistance, the designing engineer must determine the required details by other means. The interactive system design happens during the online phase.

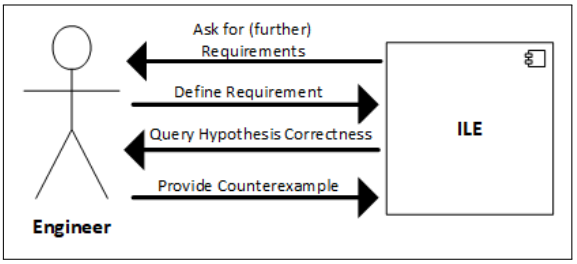


Fig. 3. Queries of the *ILE*.

a) ***Interface and Component Definitions****:* This step happens in the offline phase, as the determination of the system components, their exact boundaries and interfaces are part of the architecture, not the behavior. The engineer must provide the names of the system components, along with their interfaces – in other words their input- and output alphabets – before the workflow can proceed to the next step. During the online phase, the components are handled separately, and are connected later based on their defined interfaces.

In our running example, the engineer first provides the component *TrafficLight* with the input interface(s) containing $\{toggle,\ interrupt\}$ – corresponding to the light-switching sequence and the police interrupt respectively – and output interface(s) containing $\{red,\ green,\ yellow,\ blinkingYellow\}$ – corresponding to the states of a possibly connected basic traffic light display. Then the *PedestrianLight* and the *Controller* components follow, with interfaces defined through the behavior seen in Figure 2 and specified analogously to the previous example.

b) ***Providing Requirements****:* During the workflow, the engineers can provide requirements in both phases. These requirements can vary greatly in their scope – from being specific to individual runs to being generally valid for the whole component – in addition to the differences in the formalism the user defines them through.

In the offline phase, this means that the users add requirements they have formulated in advance. This is useful for more general requirements, with the scope of the whole component, easily formulated as program logic expressions, or long and complex traces.

In the online phase, adding requirements means answering the questions formulated by the automata learning algorithm about a yet unspecified behavior at a specific place in the trace currently being examined. This too can be answered through program logic – e.g. when the engineer realizes a general property during the model construction – but also through traces and through giving the corresponding output directly.

The ILE currently supports the following requirement formalisms:

1) Corresponding Output
2) Valid Trace
3) Invalid Trace
4) Sequence Diagram
5) LTL Expression

Conflicting requirements are expected during the system design, especially when abstraction refinement is applied. However, conflict handling is a difficult and resource intensive task for algorithmic reasons. Consequently, the ILE only guarantees to handle the conflict, when it also interferes with the model synthesis, in which case, the user is asked to remove one of the conflicting models.

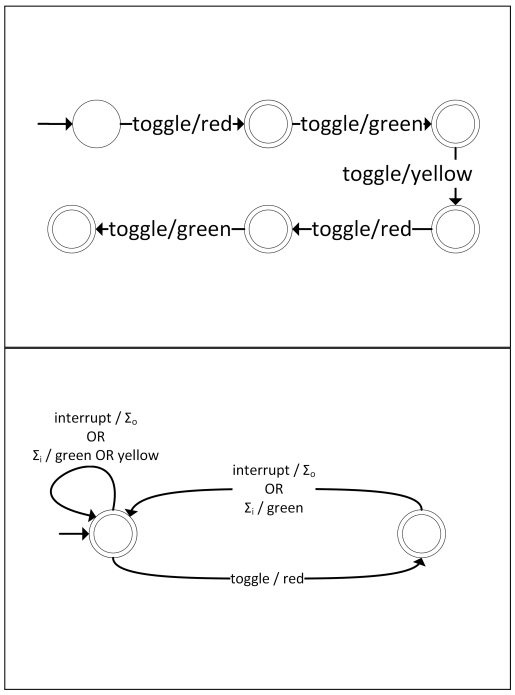


Fig. 4. Examples for attempting to model the requirement 'after toggle/red, for toggle the output is green' through a valid trace automaton (*upper*) and a Büchi automaton (*lower*).

For the traffic light component of our running example, the engineer may provide *toggle/red toggle/green toggle/yellow toggle/red toggle/green* as an initial, Valid Trace-type requirement to illustrate the toggle sequence of the traffic light. A similar, less redundant and more expressive LTL requirement can be formulated for a subset of this behavior, as 'after toggle/red, for the next toggle the output is green': $G(toggle \wedge red \rightarrow X(toggle \rightarrow green))$. The corresponding automata of the above two requirements are illustrated on Figure 4, note the the contrast in expressiveness and information capacity.

c) ***Checking the Correctness of the Synthesized Model:*** During the online phase, whenever the ILE assumes that it has gathered enough information to construct a model for the given component, the engineer is offered a model representing the current state of the model synthesis – the equivalent of an equivalence query in automata learning algorithms. The user can either approve this model – in which case the automata learning and therefore the designing of the behaviour is complete – or provide a counterexample where the model does not meet the – not yet specified – requirements.

The proposed equivalence model is a Mealy machine, which, based on the information provided by the user, can be incomplete in multiple ways. The behavior of the desired model can differ from that of the learned system because of lacking information, in which case the user needs to provide the separating behavior. Another reason for incompleteness can be newly discovered states, whose behavior is unknown based on their input signatures. This case prompts the user to evaluate the validity of state separation and to provide the lacking information. If, for some reason the hypothesized behavior is contradicting that of the desired system (by the users oversight in providing requirements), the actual, conflicting requirement can be provided to guide the learning algorithm through the process described in the previous subsection.

If the model is accepted, the design phase – for the currently learned component – is complete. If a counterexample is provided, the online phase resumes and the system design continues until the next possible model is reached.

In the case of our running example, the engineer might receive a model for equivalence validation, in which after a police interrupt, a consecutive second police interrupt does not end the blinking yellow state. Here a possible counterexample would be the input sequence $\{interrupt\ interrupt\}$, signaling a diverging behavior.

d) ***Creating and Serializing a Composite Model:*** When each of the component models declared during the first step of the workflow are completed, the resulting components are connected into a composite system model, which then can be serialized and handed over to the engineer for further (manual) extensions or usage, e.g. for code generation.

IV. Conclusion

In summary, we designed a new, semi-automated methodology to support component-based system design powered by adaptive automata learning. Additionally, we created a proof-of-concept implementation in order to validate our approach and demonstrated the capabilities and limitations of the implementation and the approach through a case study.

This paper is only the first step of designing the interactive learning methodology, we present our future goals in the following. We plan to thoroughly evaluate the complexity and applicability of our methodology, as they also depend on the skills of designing engineers. Since the by-design correctness of the components does not guarantee the correctness of the system, the inclusion of model-checking capabilities could further improve the methodology. Furthermore, as currently only Mealy machines are supported, we plan to also synthesize MBSE artifacts such as statecharts.

References

[1] IEEE Standard Glossary of Software Engineering Terminology. *IEEE Std 610.12-1990*, pages 1–84, 1990.

[2] Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87 – 106, 1987.

[3] Bartha Tamás, Majzik István. *Biztonságra tervezés és biztonságigazolás formális módszerei*. Akadémiai Kiadó, 2019.

[4] Sanford Friedenthal, Regina Griego, and Mark Sampson. Incose model based systems engineering (mbse) initiative. 01 2009.

[5] Falk Howar and Bernhard Steffen. *Active Automata Learning in Practice*, pages 123–148. Springer International Publishing, Cham, 2018.

Using Auction Mechanism for Assigning Parking Lots to Autonomous Vehicles

1st Levente Alekszejenkó
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
ale.levente@gmail.com

2nd Tadeusz Dobrowiecki
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
dobrowiecki@mit.bme.hu

***Abstract*—Considering that the Connected Autonomous Vehicles (CAVs) of the future can choose arbitrary parking lots when being idle, parking lot assignment for CAVs will be a challenging task. Auctions are computationally efficient and distributable assignment mechanisms, hence they may offer a suitable solution. In this paper, a single-unit demand, simultaneous independent auction is presented to solve the CAV parking lot assignment problem.**

The proposed method takes into account both the financial interests of the CAV operators and ecological or social interests of the community to favor closer alternatives instead of more distant ones.

***Index Terms*—CAV, simultaneous independent auction, single-unit demand auction, parking lot assignment**

I. Introduction

Replacing traditional parking lot searching methods might be possible as the mass usage of connected autonomous vehicles (CAVs) is emerging. Instead of the closest parking lots demanded by the drivers leaving their cars in the direct proximity of the destinations (to minimize the walking effort), CAVs can choose cheaper alternatives or a parking lots better positioned (e.g. not under direct sunlight). As [8] points out, cruising of (C)AVs, instead of parking, potentially doubles the traveled distances. Hence, it is beneficial to find parking places for privately owned CAVs. To make this possible, development of new parking lot choice strategies is inevitable.

Those methods naturally require distributed and efficient calculation. Moreover, they shall respect privacy as well, e.g. the origin and destination of the journey will be kept secret.

Besides, different CAVs can value the same parking lot differently. For example, for an expectedly short time parking, a closer parking place is more valuable than a distant one. Or, if returning to the garage is not a rational option, CAVs may value every parking lot more.

For these reasons, auctions might be a beneficial way for parking lot assignments as well. In this paper, we present a single-unit-demand, simultaneous-independent auction implementation, and its application to a parking lot assignment. The choice of this particular mechanism is dictated by the following assumptions: (1) the CAVs are (will be) privately owned, (2) the owner costs of using CAVs, and the community costs of providing infrastructure, are based on travel distance, energy costs, and community taxes, (3) the cooperation between an idle CAV and a parking lot is based on an auction, (4) a single CAV enters negotiations with multiple parking lots, rated according to its optimal interests.

The research has been supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.2-16-2017-00013, Thematic Fundamental Research Collaborations Grounding Innovation in Informatics and Infocommunications).

II. Related Literature

Literature on auction mechanisms is abundant [1]. Now, we focus on multiple auctions, running simultaneously and in which we only have at most a single demand. It is trivially not a multi auction as defined in [2]. Weber [3] has mentioned simultaneous independent auctions, in which bidders must act simultaneously in multiple auctions, but the single-unit demand was not considered in that paper.

Menezes et al. [4] presents an overview of single-unit demand auctions, but simultaneous, independent auctions were not discussed yet. A general method for simultaneous independent auctions with n-unit demand was discussed in [5]. A concrete implementation of their method for a single-unit demand case is shown in this paper.

Polak et al. [6] summarizes the most common parking lot searching methods of human drivers. Unfortunately, none of these algorithms is guaranteed to secure a parking place for vehicles. Since parking lots are considered to be a limited resource also in the future, efficient transportation systems might ensure parking lot reservation for autonomous vehicles. As auctions provide the possibility of efficient and distributed calculation, no central agent is needed (and we can dispense with its negotiation complexity problems) [7]. Therefore, privacy of travel might be guaranteed as well, as no entity will know exactly the preferences or the destination of the parking lot seeking CAV.

Our solution assigns parking lots to CAVs using an auction method. Auction methods have been proposed already to solve some problems of CAVs, such as electric vehicle charging and discharging in microgrids [9], or for efficient distribution of parking lots and resources in a vehicular fog computing system [10]. The auction proposed in the latter paper considers the interest of both CAVs and fog node controllers. We assume, however, that monetary infrastructural interests are hidden

from the CAVs. We believe that CAVs shall prefer closer parking places as it is more beneficial in ecological and social terms.

III. An Implementation of a Single-Unit-Demand, Simultaneous Independent Auction

A. Definitions and Assumptions

Considering that a CAV require at most one parking lot at a time, we focus now on a single-unit demand auction. Every free parking place can be bid at discrete points in time. Hence, the proposed algorithm is a single-unit demand simultaneous independent auction (SSIA).

Every single parking lot work as an auctioneer, trying to sell its free capacity. For efficient implementation, auctioneers ask every bidder whether it is willing to pay the current price or not (a cyclic auction).

Auctions are scheduled regularly, in our implementation in every three minutes. Every CAV which will arrive at its destination within this time frame shall participate in these scheduled auctions. Three minutes is roughly equivalent to traveling 1 kilometer in a city, therefore CAVs can estimate their arrival time accurately.

We assume that CAVs know (through V2I – Vehicle-to-Infrastructure communication) all parking lots in the area and can and will participate in auctions as bidders for a free parking place.

For convenience, let us call a bidder active in an auction if and only if it is currently bidding in that auction. A CAV can be active at most one auction at a time [1], otherwise, it might win more than one parking place, which is avoidable for an efficient parking lot assignment.

B. Greedy bidding

Every CAV has a monotonic preference list regarding the offered parking lots. Thus, a utility function must be defined to provide a monotonic ordering. This function naturally contains parking and traveling prices, moreover, a component which is proportional to the distance of the parking lot. Therefore, CAVs can opt for cheaper alternatives, meanwhile, the overload of the road network (and increased energy consumption, pollution or wearing out) is also avoidable.

As a b CAV sees it, the $u_{i,b}$ utility function of the ith parking lot is given by equation (1), where p_i stands for the sum of the parking price at the ith parking lot and the travel price to this parking facility. d_i is the distance to the parking lot (in meters) and c is a constant to scale distance values to the parking prices.[2] The $0.0 \leq \alpha \leq 1.0$ coefficient reflects how a CAV values parking prices over travel distances.[3] As a CAV has to travel $d_{i,b}$ distance two times, we shall represent this knowledge as a coefficient of 2 in the distance component.

$$u_{i,b} = \alpha \cdot p_i + 2 \cdot (1 - \alpha) \cdot c \cdot d_{i,b} \tag{1}$$

[1]Regardless the number of auctions at which the CAV is a participant.

[2]For Hungarian Forint (HUF), we simply use c=1.0

[3]In current paper, in favour of simplicity, we use an $\alpha = 0.5$ value.

As mentioned before, p_i has two components as well. One part of p_i is the c_p parking cost, and the other component is denoted as a $c_f(d_i)$ function. c_f represents a distance related fuel price (or some virtual form of fuel price as electronic vehicles might be more common in the future, and it seems unlikely that governments will waive fuel taxes) and additional amortization. Altogether, equation (2) shows the formula for calculating p_i.

$$p_i = c_p + 2 \cdot c_f(d_i) \tag{2}$$

By calculating u_i for all possible parking lots, CAVs will have a preference list covering them all. Let us assume, that auctions can run in an instant, therefore the only variable part of this formulation is the c_p parking price component. Thus, every CAV will have a currently favored parking lot to bid for. If possible, then a CAV will bid in the auction of this particular parking lot actively. To guarantee that a CAV will not win more than one parking lot, it shall be active in at most one auction. In the following, a detailed description of the auction mechanism is given.

C. Multiagent system and its negotiation algorithms

From a multiagent system point of view, auctioneers (the parking lots) and bidders (the CAVs) are two kinds of agents.

Auctioneers sell, in our particular case, free parking places. Given the $\mathcal{B}$ list of bidders and the starting price s_p, Algorithm 1 describes the task of an $a \in \mathcal{A}$ auctioneer agent. Auctioneers ask every $b \in \mathcal{B}$ participant cyclically whether they are willing to give the c_p current bid or not. When a bidder sends in the current bid, the auctioneer will raise it by a small $epsilon$ amount. An auction terminates when only one bidder entered the current bid ($n_w = 1$), or every participant left the auction ($|\mathcal{B}| = 0$). (The latter is possible if every bidder had already won in another auction, e.g. when the supply is higher than the demand for parking lots.) The only one bidder, who is willing to give the current bid, will be the b_w winner of the auction.

Algorithm 1 Algorithm of an auctioneer

$c_p \leftarrow s_p$
repeat
  $n_w \leftarrow 0$ ▷ No one has bid in the current cycle yet
  for $b \in \mathcal{B}$ **do**
    answer $\leftarrow$ ASK(b, a, c_p)
    if answer = "bid" **then**
      $c_p \leftarrow c_p + \epsilon$ ▷ Increasing the current price
      $n_w \leftarrow n_w + 1$ ▷ One more bidder gives c_p
      $b_w \leftarrow b$ ▷ Current winner can be b
    end if
  end for
until $|\mathcal{B}| > 0 \wedge n_w > 1$ ▷ There are bidders, no one won
if $n_w \neq 1$ **then** $b_w \leftarrow \varnothing$ ▷ No one had bid, no one won
end if
return b_w

For a b bidder, the most important function is *ASK* and is presented as Algorithm 2. This function, given an auctioneer a and a current price c_p determines the *answer* whether to enter or not with the current bid. A bidder leaves an auction, if its current price exceeds the m_p maximal price of a bidder. If the auctioneer (the caller of the *ASK* function) is the preferred one ($a = a_p$), and the b bidder has already overbid ($c_p > c_g$, where c_g is the last bid of b), then a bidder might choose another preferred auction (a_p) to bid in. If this preference is unchanged, then the bidder will enter the current price, otherwise it will not, moreover it will note that it cannot win any auctions currently, as it did not place a winning bid.

Algorithm 2 Algorithm of a bidder

```
function ASK(b, a, c_p)
    answer ← "do not bid"
    if c_p > m_p then
        LEAVE_AUCTION(a, b)
    end if
    if a = a_p then                    ▷ Caller is the preferred auction
        if c_p > c_g then              ▷ Have we been overbid?
            a_p ← arg min_i u_{i,b}, where i ∈ [0..|A| − 1]
        end if
        if a = a_p then                ▷ Has our preference changed?
            answer ← "bid"
            c_g ← c_p                  ▷ Last bid is the current bid
        else c_g ← 0                   ▷ No bid has been entered
        end if
    end if
    return answer
end function
```

D. Computational efficiency of SSIA

When every bidder is willing to give the m_p, maximal price for an object in a specific auction, then it requires $\lceil \frac{m_p - s_p}{\epsilon} \rceil$ steps for the auctioneers. Supposing $n = |\mathcal{B}|$, n further steps are also necessary to ask every bidder once again (when none of them will accept the current price). Altogether, this case requires at most $k = \lceil \frac{m_p - s_p}{\epsilon} \rceil + n$ steps. The auctioneer has to maintain a list of its bidders, besides some technical information (eg. current price, bid step). Hence, the algorithm of an auctioneer runs in $\mathcal{O}(n)$ time with $\mathcal{O}(n)$ memory demand in the n number of its bidders.

For the bidders, the most steps are required when the bidders have exactly the same preference lists in every auction cycles. Supposing that a bidder participates in m auctions, the auctioneers make at most $k - n$ steps without any of the bidders winning in an auction. In the last cycle, however, the bidders might win the m auctions one-by-one. Therefore, at most $l = m(k - n) + m$ steps are required. As actual bids or some additional technical data of the auctions shall be maintained, arrays of m-length might be required. Thus, the algorithm of a bidder also runs in $\mathcal{O}(m)$ time and with $\mathcal{O}(m)$ memory demand, in the m number of auctions.

IV. SSIA Simulation Experiments

Considering that we cannot anticipate the ways future cities are expected to adapt to the future needs and behavior of their citizens yet, detailed simulations do not necessarily yield assessable results of the future traffic. Therefore, we shall use a more abstract simulation framework, like the one described in details in [11].

Now, we simulate the activity-chains of 3000 people in a city with an area of 10000 m $\times$ 10000 m with residential surrounding of 20000 m $\times$ 20000 m. These human activities might be going to work or going to shopping and every movement involves using a CAV which ultimately will be left to itself awaiting to be recalled at the end of the activity. At least one and at most three specific activities make every activity chain, which start with a departure from a home position and ends with an arrival at this home position. Home positions are distributed evenly ($X_h, Y_h \sim \mathcal{U}(-10000, 10000)$) in the city.

Between every human activity in a chain, the CAVs shall find a suitable parking place. We model two types of parking lots: curb-side parkings and parking houses. 350 curb-side parking lots are concentrated in the city center according to the Gaussian distribution ($X_c, Y_c \sim \mathcal{N}(0, 5000)$) with capacities $N_c \sim \mathcal{U}(1, 10)$, i.e. from a single place up to 10 parking places, while 5 parking houses are distributed in an annulus around the city (with their distance from the city center, distributed circularly with Gaussian radius and uniform angle: $R_p \sim \mathcal{N}(0, 3000) + 10000$, $\Phi \sim \mathcal{N}(0, 2\pi)$). They have a capacity of $N_p = 300$.

To realistically represent the d driving distance between two points of the city, the $d = E + s(M - E)$ formula is used, where E stands for the Euclidean, and M stands for the Manhattan (or taxi-cab) distance of the two points. The s is a probabilistic coefficient ($s \sim \max(0, \mathcal{N}(1.3, 1.8))$), estimated from measurement of real-world driving distances in Budapest.

A. Parking lot seeking methods

In the described simulation framework, we compared a traditional parking lot searching strategy to an SSIA based one. The traditional strategy was to *spiral away* from the last destination (similar to Strategy VI of [6]). In the simulation, this method requires listing parking lots based on their distance from the place of the human activities. To check whether these parking lots are free or not, a CAV shall travel from one parking lot to another following the order of the distance list, which ensures spiraling away and around the destination, instead of a greedy depth first search.

The SSIA method was used to licit for a parking fee calculated on a per second basis. The starting prices are the same as they were when the parking lots were generated, see [11]. We used $\epsilon = \frac{5}{60 \cdot 60}$ $[\frac{HUF}{s}]$ increment and $m_p = \frac{5000}{8 \cdot 60 \cdot 06}$ $[\frac{HUF}{s}]$ maximal price, representing an increment of 5 HUF/hour and a maximum of 5000 HUF for a parking of eight hours.

As travelling further than the cheapest alternative or paying more than at the closest alternative are irrational parking lot choices, CAVs will not participate in such irrational auctions. In our simulation, going home is always an option, hence if

a CAV cannot win any auction, it shall return to its home position. (There is a possibility, that CAVs will find the first free parking lot at their own home position in the traditional method as well.)

B. Compared values

The simulations of both parking lot searching methods were run for 10 times, each time with a newly generated city model. To compare the possible benefits of SIAA over the traditional method, the following values were measured.

Unloaded distances: The commulated distance the CAVs travel without carrying passengers or cargo. (In this particular case, it is the distance traveled to and from parking places to the destination of the passengers of the CAVs.) *Total parking prices:* The total amount of money spent on parking. *Ratio of occupied parking places:* The ratio between the occupied and the total parking places in the simulation by hours of a day.

C. Measurement results

The SSIA method is capable of directly assigning parking places to CAVs, hence no unnecessary traveling is needed to find a free parking place. As it can be seen in Fig. 1, SSIA can reduce unloaded distances even by 90% compared to the traditional method.

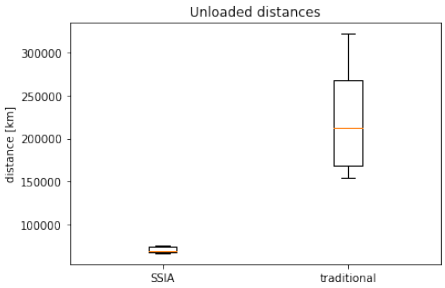


Fig. 1. Cumulated unloaded distances traveled by 3000 CAVs to and from parking places.

According to Fig. 2, parking prices are also reduced by SSIA, however, we shall check Fig. 3 as well for parking lot occupancy. As approximately two times more parking lots are occupied with the traditional method, we can conclude that SSIA slightly increases the paid parking costs. We would like to add, that as SSIA results more free parking lots (with significantly lower parking lot searching traffic demand), hence these unoccupied parking lots can be recultivated as e.g. sidewalks, terraces of cafés or bike lanes, etc.

V. Conclusion

In this paper, a single-unit demand, simultaneous independent auction (SSIA) method is proposed for assigning parking lots to connected autonomous vehicles (CAVs). Simulations with an abstract, qualitative city model were carried out, which prove that, however, the SSIA increases the parking costs, it efficiently reduces the distance traveled unloadedly while looking for free parking lots, and it may also reduce the number of required parking lots as well.

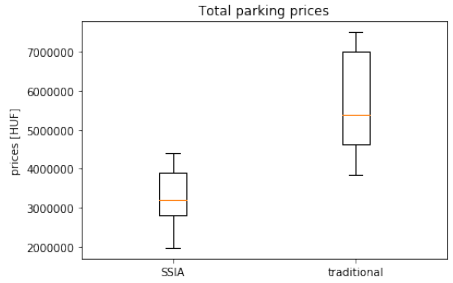


Fig. 2. Cumulated parking fees payed by 3000 CAVs.

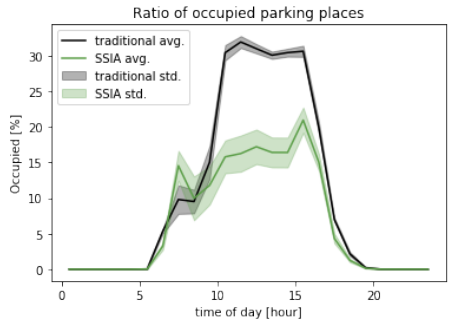


Fig. 3. Occupation of parking lots during a simulated day.

References

[1] J. Cassady R., *Auctions and Auctioneering*. Berkeley: University of California Press, 1967.

[2] W. Vickrey, "Counterspeculation, Auctions, and Competitive Sealed Tenders," *The Journal of Finance*, vol. 16, no. 1, pp. 8–37, Mar. 1961.

[3] R. J. Weber, "Multiple-object Auctions." J.L. Kellogg Graduate School of Management Northwestern University, Evanston, Illinois, Aug. 1981. [Online]. Available: https://www.kellogg.northwestern.edu/research/math/papers/496.pdf

[4] F. Menezes, "Auctions of Identical Objects With Singleunit Demands: a Survey," *Brazilian Review of Econometrics*, vol. 18, no. 2, pp. 309–340, Nov. 1998. [Online]. Available: http://bibliotecadigital.fgv.br/ojs/index.php/bre/article/view/2839

[5] V. Bansal and R. Garg, "Simultaneous Independent Online Auctions with Discrete Bid Increments," *Electronic Commerce Research*, vol. 5, no. 2, pp. 181–201, Apr. 2005. [Online]. Available: https://doi.org/10.1007/s10660-005-6156-1

[6] J. Polak and K. Axhausen, "Parking Search Behaviour: A Review of Current Research and Future Prospects," *University of Oxford, Transport Studies Unit (1. Jan. 1990)*.

[7] S.-Y. Chou, S.-W. Lin, and C.-C. Li, "Dynamic parking negotiation and guidance using an agent-based platform," *Expert Systems with Applications*, no. 35, pp. 805–817, 2008.

[8] J. Bischoff, M. Maciejewski, T. Schlenther, and K. Nagel, "Autonomous Vehicles and their Impact on Parking Search," *IEEE Intelligent Transportation Systems Magazine*, vol. 11, no. 4, pp. 19–27, 2019.

[9] D. Li, Q. Yang, D. An, W. Yu, X. Yang, and X. Fu, "On Location Privacy-Preserving Online Double Auction for Electric Vehicles in Microgrids," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 5902–5915, Aug. 2019.

[10] Y. Zhang, C. Wang, and H. Wei, "Parking Reservation Auction for Parked Vehicle Assistance in Vehicular Fog Computing," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3126–3139, Apr. 2019.

[11] L. Alekszejenkó, "Connected autonomous vehicle (cav) and the urban parking problems," Master's thesis, Budapest, Hungary, 2021, in Hungarian. [Online]. Available: https://diplomaterv.vik.bme.hu/en/Theses/Kommunikalo-onvezeto-auto-CAV-es-a/Attachment/50899

Investigating Static Analyzers Detection Capabilities on Ethereum Smart Contracts

Mirko Staderini, Andrea Bondavalli
Department of Mathematics and Informatics
University of Florence
Florence, Italy
{mirko.staderini, andrea.bondavalli}@unifi.it

***Abstract*—Ethereum smart contracts had ever-increasing development in recent years. Hidden vulnerabilities can not be patched once a smart contract is deployed on a blockchain because of the code immutability. The use of static analyzers reduces the number of vulnerabilities in smart contracts. The paper focuses on the outcomes of some static analyzers for Solidity smart contracts. Based on a language-independent systematization of vulnerabilities, the work performs an analysis of positives detection of some static analyzers on a smart contracts dataset. Such analysis permits (i) identifying a positive correlation among some smart contracts' metrics and tools' outcomes, and (ii) investigating where tools detect specific classes of the systematization.**

Keywords—smart contracts, tools, classification, metrics, correlation, location detection.

I. INTRODUCTION

Smart contracts are one of the most important innovations of the second generation of the Blockchain. The basic idea is to execute computerized transactions automatically. Their diffusion has allowed the development of applications in different areas (e.g., financial, medical, insurance, gaming, betting). Nevertheless, design and coding faults can cause *weaknesses* in implementing smart contracts. Weaknesses could lead to exploitable *vulnerabilities*. The problem is even more remarkable, considering that developers can not patch smart contracts once deployed on the Blockchain. Ethereum is one of the most used platforms, and it offers Solidity as the main (and Turing complete) programming language. Thus, the analysis of vulnerabilities for Solidity smart contracts is extremely important for the platform security.

The main methods to analyze code are the following [1]:

- *Static analysis* inspects the code without running it.
- *Dynamic analysis* executes the program and acts as an attacker looking for vulnerabilities.
- *Formal analysis* uses theorem provers or formal methods to analyze specific patterns.

Static analysis is widely used to discover vulnerabilities in the early stages of the software life cycle. It can cover 100% of the code at a low cost, despite the incomplete fault coverage [2]. The work focuses on selected static analysis tools (SATs) that analyze Solidity smart contracts. Each line of a contract under test can have two different outcomes: *a negative result* or a *positive result*. A positive result can be a *true positive* (*TP* - correct detection of an existing vulnerability) or a *false positive* (*FP* - wrong detection of a non-existing vulnerability). For our purpose, the paper investigates the positive results, referring to them as *positives* (P).

There are several vulnerability databases, e.g., *BugTraq*, and *Common Vulnerabilities and Exposures* (CVE). The *National Vulnerable Database* (NVD), the U.S: Government repository for vulnerabilities, uses the Common Weakness Enumerator (CWE) to classify CVE entries. CWE is a wide-spread used list of software weaknesses based on a hierarchical structure. Proceeding top-down, root is the most generic abstraction, and it represents the view. P*illars* (or *categories*) and *classes* are independent of any specific language. *Bases* and *variants* describe the weakness at a lower level of detail. The entire CWE list has three hierarchical representations that focus on a specific aspect. The *research concept* is a language-independent representation based on the behavior of weaknesses [3].

Some studies show the link between the software's complexity and the outcomes of static analyzers (e.g. [4]). More recent work analyzes software metrics distribution in a set of Ethereum smart contracts extracted from Etherscan [5]. Our paper analyzes positives of SATs in checking Solidity smart contracts, focusing on software metrics and positives locations.

- As the first step, the work identifies a set of Solidity vulnerabilities, mapping them to a CWE taxonomy developed in [6]. Then, the work selects the SATs.
- Through a random extraction from Etherscan, a subset of smart contracts defines the *dataset*. The work identifies metrics for the analysis.
- Checking the dataset by all static analyzers permits to determine the correlation between some software metrics and positives.
- Finally, the work investigates locations in smart contracts where tools detect specific classes of vulnerability.

The organization of the paper is as follows. Section II provides the taxonomy and the SATs selection. Section III focuses on the dataset and software metrics. Section IV focuses on experiments, and Section V concludes the work.

II. TAXONOMY AND STATIC ANALYZERS SELECTION

To provide a list of vulnerabilities, we examine several papers, among them [7], [8], [9], [10], [11]. Moreover, we investigate some online documentation (e.g., the *Smart Contract Weakness Classification and Test Case* (SWC) registry) and Github repositories. For our purpose, we consider only vulnerabilities that can be exploited in the Solidity release >= 0.5. At the same time, we group vulnerabilities with a similar or overlapped definition. At the end of the selection process, we end up with 34 vulnerabilities. Using the methodology detailed in [6], the paper maps the vulnerabilities in ten categories that TABLE I

describes[1]. The complete taxonomy is available

To define a set of SATs, the paper identifies a set of candidates of interest, starting with the survey of Di Angelo et al. [12]. Then, we look for Github repositories and research papers that treat static analyzers. Identifying some publicly available tools, we refine our research with the following criteria. At first, the standard release of the tool supports the Solidity release 0.5. Then, SATs can perform the analysis without assertions or user-defined properties. Finally, tools can detect vulnerabilities.

The process ends up with six tools. TABLE II summarizes the tool name and the tool release under analysis. Moreover, for each tool, it highlights the input mode (BC - *bytecode* or SC - *source code*) and the internal representation (CFG – *control flow graph* or AST – *abstract syntax tree*).

III. Dataset and Metrics

We build a *dataset* in two steps. At first, a Java crawler extracts randomly smart contracts from the public repository Etherscan. Next, we select smart contracts with Solidity release 0.5, ending up the process with 400 smart contracts.

Software metrics measure software complexity. Complexity is often related to the number of software faults. TABLE III summarizes software metrics that the work uses, grouped in four categories: *length* metrics, *contract-oriented* metrics, the *cyclomatic complexity*, and *Halstead* metrics. Note that the *.sol represents the smart contract file. Moreover, the table highlights the average and the standard deviation of each metric in the whole dataset.

Length metrics represent the line of code of the program, and they are strongly dependent on the programming style. Contracts oriented metrics are related to the number of logical contracts, functions, libraries, and parameters that a smart contract file contains. A Perl script parses the code of smart contracts and then calculates these metrics. The cyclomatic complexity is a quantitative measure computed using the CFG of the program. The paper uses a variant of the original metric, calculated explicitly for the Solidity language. It results in the sum of each function's cyclomatic complexity (SCC [13]). Halstead metrics interprets the software as a sequence of tokens, in turn, classified into *operators* (OP) and *operands* (OD). Different combinations between OP and OD highlight distinct software characteristics. The paper calculates these metrics using a Solidity grammar for Another Tool for Language Recognition (ANTLR).

The work uses *location metrics*. The *location of detection* (LoD) is the line of a smart contract where a tool detects a positive. The *relative location of detection* (RLoD) is the LoD compared with the number of lines of the smart contract under analysis.

IV. Experiments

A. Methodology

The work performs an analysis of positives detection of SATs on a dataset. The basic idea consists of processing dataset by each SAT. Tools use several checking rules to detect vulnerabilities; each positive is related at least to one checking rule. Moreover, each tool delivers results in a different format; then, results are harmonized, mapped into the CWE category, and refined. Finally, each positive is represented by the tuple (*tool, smart contract, location, CWE category*). We use software metrics to correlate positives and contracts' complexity. Finally, location metrics permits to highlight where tools find the different category of vulnerabilities.

[1] The complete taxonomy is available at this link.

TABLE I CATEGORIES OF TAXONOMY

CWE Systematization	CWE description
CWE-20	Missing proper input validation
CWE-284	Improper access control
CWE-330	Use of insufficiently random values
CWE-345	An insufficient verification of data authenticity
CWE-400	Uncontrolled resource consumption
CWE-668	Exposing resources to wrong sphere
CWE-669	Incorrect resource transfer between spheres
CWE-682	Incorrect calculation
CWE-691	Insufficient control flow management
CWE-703	Improper check or handling of exceptional conditions

TABLE II TOOLS SELECTION

Tool		Characteristics	
Name	**Release**	***Input***	***Internal representation***
Securify2 (Sfy2)	30th June 2020	BC	CFG
Securify (Sfy)	5th June 2020	BC	CFG
Slither (Sli)	0.6.2	SC	AST, CFG
SmartCheck (SmC)	2.1	SC	AST
Remix IDE (Rmx)	30th June 2020	SC	AST
Mythril (Myt)	0.22.6	BC	CFG

TABLE III SOFTWARE METRICS

Metrics			The complexity of the dataset	
Group	**Acronym**	***Description***	***Average***	***Standard Deviation***
Length	LOC	Number of lines of code	474	753
	LLOC	Logic lines of code	282	448
Contract oriented	Fun	Functions in *.sol	33	53
	Par	Functions' parameters in *.sol	55	91
	Stat	Statements in *.sol	110	161
Cyclomatic complexity	SCC	Sum of McCabe complexity of each function	47	71
Halstead	(U)OP	(Unique)Operators of smart contract	(45) 1309	(13) 1845
	(U)OD	(Unique)Operands of smart contract	(177) 768	(203) 1112
	H_LEN	Smart contract's length: OP+OD	2078	2952
	H_VOC	Smart contract's vocabulary: UOD+UOP	222	214

B. Results

Fig. 1 and Fig. 2 show a preliminary data overview. Fig. 1 highlights the frequency distribution of positives. More than 65% of smart contracts have a number of positives between 0 and 49. Fig. 2 shows the distribution of positives for each tool, highlighting that one tool (Sfy2) detects the relative majority of positives compared with the other ones.

1) Correlation Analysis

This section shows the correlation between positives and software metrics. The use of the Pearson coefficient permits identify which metrics of TABLE III have the correlation coefficient greater than 0.80: (i) LOC (0.85) and LLOC (0.84) for the lines of code; (ii) functions (0.83), parameters (0.82), and statements (0.82) for the metrics smart contract oriented; (iii) SCC (0.82) for McCabe's complexity; (iv) H_LEN (0.86), H_VOC (0.85), H_VOL (0.85) for Halstead metrics.

Figure 3 depicts scatterplots showing, for each group, the metric that has the highest correlation with positives (functions, LOC, SCC, H_LEN). Observing evidence, simple contracts (low values on the x-axis) have a higher correlation with positives than complex contracts. On the other side, the high complexity of contracts (e.g., the number of variables) makes it hard to find vulnerabilities.

There is also a strong correlation among metrics and positives of a tool (except Mythril and Securify). Mythril focuses on detecting specific vulnerabilities depending on language features more than contracts' complexity. The low correlation of Securify indicates its lower effectiveness. Positives grouped by CWE classes have a low correlation with each metric (coefficient between 0.2 and 0.4). These results suggest that smart contracts' complexity impacts the number of positives more than the class they belong to.

From these results, we can argue that contract developers should pay attention in the complexity of contracts. Complexity leads to error-prone contracts. Conversely, tools' developers should focus on more complex contracts.

2) Analysis of classes distribution

This section highlights the distribution of CWE categories and the relation between classes and location metrics. Fig.4 highlights the category distribution: CWE-20 and CWE-284 compose 65% of the positives in the dataset. Referring to a specific class, the vulnerability detection capability of SAT affects the CWE distribution. Fig. 5 shows that every tool detects the class CWE 284; conversely, only two tools identify positives in CWE20, CWE-345, and CWE-400. As a consequence, subsequent tool updates should focus on finding bugs in less investigated classes.

Fig. 6 shows the frequency distribution of positives detection. The absolute location distribution reflects the distribution of LOC (the average value is 247). More than 60% of positives are located in the first 250 lines of contracts. The related location shows that positives are distributed throughout each contract.

Combing evidence of Fig. 7 permits investigating the location of contracts where tools can find a specific class of vulnerability. On the left of Fig. 7, a boxplot compares the location distribution for each class. For seven CWE classes, except CWE-400 and CWE-703, the lower hinge has a maximum value of 50. For the same CWE classes, half of the positives have location among lines 50 and 250 of contracts. Class CWE-703 (error in handling exceptions) has a slightly different distribution. CWE-400 (exhausting resources) is the class that shows the highest 50th percentile and upper hinge. Moreover, each distribution has many outliers caused by big smart contracts (in terms of LOC).

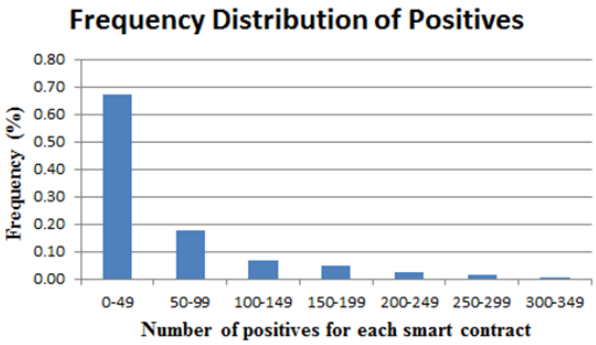


Fig. 1. Frequency distribution of positives. Each label of axis x identifies the number of positive outcomes for each smart contracts of the whole set of tools. Axis y represents the frequency.

Tool	Count	Frequency (%)
securify2	4793	37.4%
smartcheck	2003	15.6%
slither	1952	15.2%
securify	1695	13.2%
remix	1250	9.8%
mythril	1109	8.7%

Fig. 2. Positives distribution for each tool.

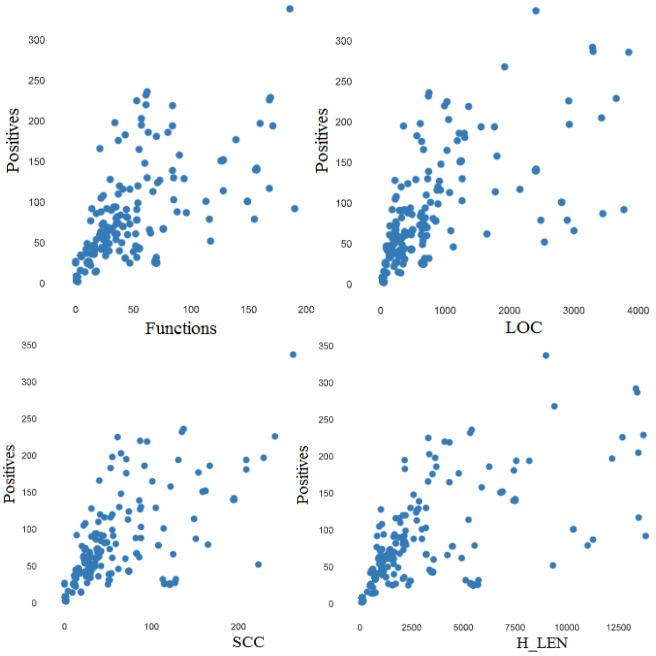


Fig. 3. Relations among positive outcomes of the dataset and some metrics. Functions of smart contracts (left) and lines of code (right) at top. Sum of McCabe's complexity (left) and Halstead length (right) at the bottom.

On the right of Fig. 7, a boxplot highlights the distributions of the relative location of positives for each class. Each distribution has a lower hinge greater than 25%. In the first part of each contract, there are preliminary definitions, libraries and interface declarations, and comments: they are not error-prone. CWE-330 has the lowest 25th percentile: this class refers to contracts that misuse random generation. Definition of random generators is typically located in the first part of a contract; moreover, the vulnerability affects the entire contract.

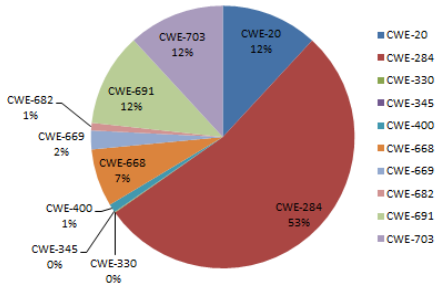


Fig. 4. Distribution of positives detection grouped by CWE classes.

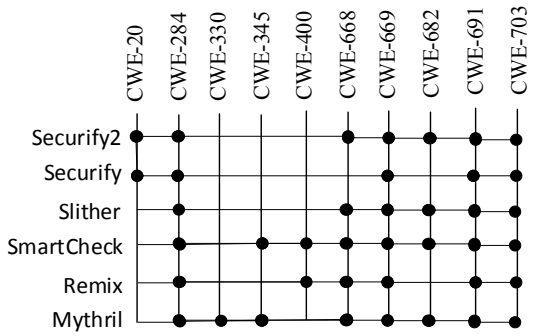


Fig. 5. Relations among tool detections and CWE classes. Each dot indicates that a tool has outcomes in the related class.

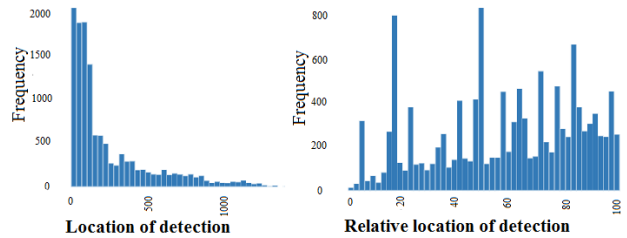


Fig. 6. Frequency distribution of the location of detection (on the left) and the relative location of detection (in percentage, on the right).

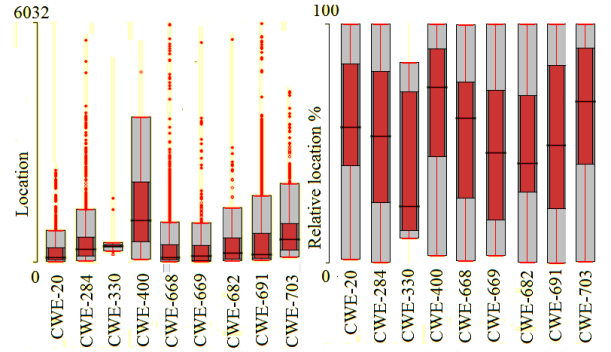


Fig. 7 Boxplots that highlight relation among the location of detection (on the left) and relative location of detection (in percentage, on the right) with the CWE classes. The lower hinge and upper hinge represent the 25th and the 75th percentile of the distribution, respectively.

Listing classes based on the increasing value of the upper hinge, CWE-682, CWE-669, CWE-668, CWE-691, and CWE-284 have the lower hinge among the 20 and the 35%. CWE-20, CWE-400, and CWE-703 have a lower hinge around 45%. CWE-20 refers to the missing input validation. Because of the Solidity smart contracts construct, there is no main function that validates inputs. Generally, functions that need input validation are complex, and they are typically located in the second part of a contract. CWE-400 and CWE-703 have the highest value of the lower and the upper hinge. CWE-703 refers to the error propagation. Solidity constructs differ among them on the handling of error propagation: missing checks can cause vulnerability. The analysis of 15 sample contracts shows that functions with this vulnerability are located in the second part of contracts, although they have low complexity. CWE-400 refers to resources exhaustion. Both boxplots highlight that we find this class in big contracts and, in particular, it is located in their second part where functions are complex.

V. Conclusion

This work focused on investigating static analyzers detection on Ethereum smart contract dataset. At first, it highlights a strong positives correlation between some software metrics and positives outcomes. Then, the paper investigates where tools can find some classes of vulnerabilities. Results of the work are affected by the dataset composition (randomly made). Future works can extend results by analyzing the true and false positives. For this purpose, it is planned to perform a manual inspection of a subset of smart contracts to create a ground truth to refer to. Moreover, we plan to analyze the overlap of positives to identify common detected faults.

References

[1] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, and Carl Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE Trans. Dependable Secur. Comput.*, vol. 01, no. 1, pp. 11–33, 2004.

[2] V. Okun, W. F. Guthrie, R. Gaucher, and P. E. Black, "Effect of static analysis tools on software security: Preliminary investigation," in *Proceedings of the ACM Conference on Computer and Communications Security*, 2007.

[3] Mitre, "CWE Common Weakness Enumeration." [Online]. Available: https://cwe.mitre.org/. [Accessed: 30-Oct-2020].

[4] P. Nunes, I. Medeiros, J. C. Fonseca, N. Neves, M. Correia, and M. Vieira, "Benchmarking Static Analysis Tools for Web Security," *IEEE Trans. Reliab.*, vol. 67, no. 3, pp. 1159–1175, 2018.

[5] A. Pinna, S. Ibba, G. Baralla, R. Tonelli, and M. Marchesi, "A Massive Analysis of Ethereum Smart Contracts Empirical Study and Code Metrics," *IEEE Access*, vol. 7, pp. 78194–78213, 2019.

[6] M. Staderini, C. Palli, and A. Bondavalli, "Classification of Ethereum Vulnerabilities and their Propagations," in *2020 Second International Conference on Blockchain Computing and Applications (BCCA)*, 2020, pp. 44–51.

[7] L. Luu, D. H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proceedings of the ACM Conference on Computer and Communications Security*, 2016, vol. 24-28-Octo, pp. 254–269.

[8] N. Atzei, M. Bartoletti, and T. Cimoli, "A Survey of Attacks on Ethereum Smart Contracts (SoK)," in *Lecture Notes in Computer Science (2017)*, vol. 10204 LNCS, Springer Verlag, 2017, pp. 164–186.

[9] A. Dika and M. Nowostawski, "Security Vulnerabilities in Ethereum Smart Contracts," in *2018 IEEE International Conference on Internet of Things (iThings)*, 2018, pp. 955–962.

[10] H. Chen, M. Pendleton, L. Njilla, and S. Xu, "A Survey on Ethereum Systems Security: Vulnerabilities, Attacks, and Defenses," *ACM Comput. Surv.*, vol. 53, no. 3, pp. 1–43, Jul. 2020.

[11] P. Praitheeshan, L. Pan, J. Yu, J. Liu, and R. Doss, "Security analysis methods on ethereum smart contract vulnerabilities - a survey," *arXiv*, Aug. 2019.

[12] M. Di Angelo and G. Salzer, "A survey of tools for analyzing ethereum smart contracts," *Proc. - 2019 IEEE Int. Conf. Decentralized Appl. Infrastructures, DAPPCON 2019*, pp. 69–78, 2019.

[13] P. Hegedűs, "Towards Analyzing the Complexity Landscape of Solidity Based Ethereum Smart Contracts," *Technologies*, vol. 7, no. 1, p. 6, Jan. 2019.

[14] Consensys, "Solidity Metrics." [Online]. Available: https://marketplace.visualstudio.com/items?itemName=tintinweb.solidity-metrics. [Accessed: 10-Dec-2020].

Local and global causal discovery methods for observational data with discrete variables

Mihály Vetró
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
vetromisu@gmail.com

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

***Abstract*—Identifying causal relationships based on observational data is challenging, because in the absence of interventions (i.e. experiments), causal discovery algorithms have to tackle the problem of observational equivalence. The latter means that causal relationships cannot be inferred from dependency relationships unambiguously, except for certain cases. Thus, the causal interpretation of dependency patterns between variables is limited, and in most cases it is only possible with adequate background knowledge, such as temporal information. To address the problem, a number of methods have emerged that follow either a global or a local approach. The former seek to reconstruct the entire dependency structure based on the available data and apply a causal interpretation if possible, while the latter examine small local units of the whole dependency structure and infer local causal relationships. In this paper, we investigate selected local and global causal discovery algorithms and compare their performance on artificial datasets. Results indicate that the accuracy of local causal discovery methods is lower than that of global methods, which is inline with expectations as not all types of causal relationships can be identified locally. On the other hand, the causal direction of discovered relationships appear to be reliable.**

***Index Terms*—causal relationships, local causal discovery, dependency structures, structure learning, Bayesian networks**

I. Introduction

The approximation of large multivariate distributions, which can be characterized by multivariate dependency relationships, from observational data is considered challenging for multiple reasons, especially when causal interpretation is involved. First, large variable sets with high sample counts tend to indicate false correlations, due to observation noise, and second, there are several causal structures (including some really simple ones) that show observational equivalence, meaning that they cannot be distinguished from each other by observing the value of their variables. Because of this, determining the direction of a causal relationship between variable pairs proves to be one of the weakest aspects of known causal discovery algorithms. Furthermore, one other shortcoming of most known methods is that they tend not to scale well to larger variable sets and in some cases even to larger sample size, with regards to computational cost and memory requirement. To attempt to mitigate this problem, we investigate a modified version of the *Local Causal Discovery* (LCD) algorithm [1], and compare it to the *Peter-Clark* algorithm [2] and the Bayesian Multi-Level Analysis method (BMLA) [3] with regards to both predictive performance and practical scalability.

This research was supported by the ÚNKP-20-5-BME-92 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund, and the János Bolyai Research Scholarship.

II. Methods

There are two main approaches when it comes to structure learning and causal discovery: (1) the local approach, where we analyze only a subset of the variables at a time, (2) and the global approach, where we take into account the whole variable set at once. In general, global approaches tend to produce better predictive performance, while local approaches usually scale better to larger variable sets and observation counts. With regards to local methods, we introduce our own version of the Local Causal Discovery (LCD) method [1], and we also investigate the performance and scalability of the Peter-Clark (PC) algorithm [2]. As for the global approach, we utilize the Bayesian Multi-Level Analysis (BMLA) method, which applies a Markov chain Monte Carlo technique, and performs a random walk in the space of directed acyclic graphs, and thus estimates possible Bayesian network structures representing the dependency relationships of variables [3].

A. The Peter-Clark algorithm

The Peter-Clark algorithm is a local method, developed by Peter Spirtes and Clark Glymour in 1990, and it was considered one of the fastest and most reliable causal discovery methods at the time [2]. The algorithm relies on examining variable pairs to determine if they are (conditionally) dependent, and variable triplets - or more precisely, chain structures containing exactly 3 variables - to determine the direction of causality between the dependent variables. Based on this, there are two main conditions that the algorithm checks for every variable pair and variable triplet of a dependency graph G with variable set $\boldsymbol{V}$:

1) Variables X and Y are dependent if and only if $X \not\perp Y|\boldsymbol{S}$ is true for every variable set $\boldsymbol{S} \subseteq \boldsymbol{V} \backslash \{X, Y\}$. In other words, X and Y are directly dependent if and only if they are dependent, and there is no variable set within $\boldsymbol{V}$ that D-separates them [4].

2) The non-oriented dependency chain $X - Z - Y$ is a V-structure ($X \rightarrow Z \leftarrow Y$) if and only if $X \not\perp Y|\boldsymbol{S}$ is true for every variable set $\boldsymbol{S}$ where $\boldsymbol{S} \subseteq \boldsymbol{V} \backslash \{X, Y\}$ and $Z \subseteq \boldsymbol{S}$. This means that the non-oriented dependency chain $X - Z - Y$ forms a V-structure if and only if there is no subset of $\boldsymbol{V}$ which contains Z and D-separates X and Y.

In theory, the PC algorithm is capable of reconstructing almost the entirety of the original dependency structure, although it has two significant shortcomings: (1) typically in the original dependency graph there are some causal relationships that are not part of a V-structure, so the PC algorithm cannot determine their direction reliably, and also, if we want to run the full PC algorithm on a variable set, (2) then the conditional independence (CI) test has to be performed on every possible $\boldsymbol{S}$ subset for every (X, Y) variable pair and ($X - Z - Y$) dependency chain, which means that the number of CI tests that have to be performed increases exponentially with the maximum size of $\boldsymbol{S}$. We can mitigate the latter scalability issue by limiting the maximum size of $\boldsymbol{S}$, that will presumably decrease the predictive performance of the algorithm, and only reduces the original exponential complexity to a polinomial, which consists of an exhaustive search.

B. Local Causal Discovery

In the previous section, we suggested that the scalability of the Peter-Clark algorithm to large variable sets is still limited, even after confining the maximum size of the investigated ($\boldsymbol{S}$) separation sets. Therefore, we have implemented a more scalable method, which is partly based on the premises of the PC algorithm and also on the original LCD algorithm [1]. Before describing the method, we must consider two significant rules that apply to every (X, Z, Y) variable triplet:

1) If X and Y are independent by default ($X \perp Y$), and the other two possible pairs are dependent (so $X \not\perp Z$ and $Z \not\perp Y$), furthermore X and Y are not independent given Z ($X \not\perp Y|Z$), then the only possible structure that the variable triplet can form is the V-structure ($X \rightarrow Z \leftarrow Y$) [4].
2) If X and Y are dependent by default ($X \not\perp Y$), and the other two possible pairs are also dependent ($X \not\perp Z$ and $Z \not\perp Y$), and Z D-separates X and Y ($X \perp Y|Z$), then there are three possible structures that the triplet can form: the common parent structure ($X \leftarrow Z \rightarrow Y$) and two chains ($X \leftarrow Z \leftarrow Y$ and $X \rightarrow Z \rightarrow Y$). These three structures are observationally equivalent, therefore cannot be distinguished from each other based on the observational data alone.

Using these axioms, our algorithm performs the following steps:

1) Create an initial "skeleton" model, by performing a pairwise independence test on the whole variable set, and connecting the pairs that are not independent with a non-directed edge.
2) Orient the edges of all the non-oriented variable chains which fulfill the V-structure criteria, so if ($X - Z - Y$), ($X \perp Y$) and ($X \not\perp Y|Z$) then ($X \rightarrow Z \leftarrow Y$).
3) Orient the second edge of every half-directed variable chain ($X \rightarrow Z - Y$), which complies with the ($X \perp Y|Z$) rule, so it can only be a directed chain ($X \rightarrow Z \rightarrow Y$) on an exclusionary basis.
4) Repeat the previous step while it produces new directed edges.
5) Orient the remaining half-oriented variable chains ($X - Z \rightarrow Y$) as a "common parent" structure ($X \leftarrow Z \rightarrow Y$) on an exclusionary basis.
6) For every triplet (X, Z, Y), if there is a direct edge between X and Y, but ($X \perp Y|Z$) then delete the edge that connects X and Y.
7) Search for directed circles, and delete the edge from each that represents the weakest causal dependence (therefore the end result will be a DAG).

We refer to this method as extended Local Causal Discovery, or LCD in short. As it can be seen from the algorithm's description, the method only performs CI tests with a single Z variable instead of a larger variable set, and only uses exhaustive search on variable pairs instead of larger variable sets. Therefore, we expect that this method scales better computationally in case of higher dimensionality (large variable sets) and larger observation sets than the PC algorithm.

C. Bayesian Multi-Level Analysis

Among the global causal structure learning algorithms, we used a Bayesian model averaging based method [5] that performs a random walk in the space of possible directed acyclic graph structures (DAG) using a Markov chain Monte Carlo (MCMC) technique [6], [7]. The transition between the states of DAG space is implemented by edge operators: add edge, remove edge, and reverse edge. At each step only one of the operators is applied by random selection. The operator performs the transformation of the current structure thus generating a 'proposed' structure. Then the proposed structure is assessed by a metric which estimates the likelihood of the data given the structure and also takes a given prior into consideration. In our case, a special form of the Bayesiam Dirichlet score was utilized (BDeu with a virtual sample size of 1) for this purpose [8]. If the score of the proposed structure is better than the current one, then the transition is made (with a probability of 1), otherwise the ratio of the scores of the proposed and the current structures (and additional factors) determines the probability of accepting such a step. The notion behind this is that after an adequate number of steps the MCMC process should reach a convergent state, i.e. the transition between DAG structures occurs between a limited number of possible structures. The result of this process is a set of DAG structures (i.e. samples obtained by MCMC sampling) that can be used to evaluate the presence of various simple structural properties, such as directed edges, or more complex properties, such as Markov blankets. These collected samples can be used to estimate the posterior probabilities of

each property [8], [9]. The initial steps of the MCMC process (called burn-in), is a phase in which convergence is typically not achieved, and therefore the corresponding samples are disregarded during the estimation of the posterior probability. In our case, $3,000,000$ steps were carried out, out of which $1,000,000$ steps were considered as burn-in.

Based on these principles, a systems-based methodology was developed by Antal et al. at the Department of Computational Biomedicine and Bioinformatics at the Department of Measurement and Information Systems, BUTE, which is called Bayesian Multi-Level Analysis of relevance (BMLA) [3]. This method has been used in several studies to investigate the relevance and causal patterns of genetic variants, environmental factors, and lifestyle descriptors influencing multifactorial diseases such as depression [10].

In addition, it is important that although both local and global methods give acceptable results in terms of dependency structure in most cases, their causal interpretation is conditional. These are based on, among others, the causal Markov condition, the faithfulness condition, the sufficiency condition, the stability condition, the no selection bias condition, and the sufficient sample condition [11]. When these conditions are satisfied, we can state that each dependency relation read from such a dependency graph represents a real dependence corresponding to the joint probability distribution and does not include anything else.

III. Results

The recorded average execution time of the three investigated methods using 5 randomly generated models with 60 variables is shown in Table I. These results indicate that the extended LCD method is significantly faster than the PC algorithm (where the size of $\boldsymbol{S}$ was maximized at 3), and an order of magnitude faster than the BMLA method. Although execution time is not an absolute measure for scalability (not like step count), it provides an overview in terms of resource-effectiveness of the three methods.

TABLE I
Execution time of the three methods[a]

	Time (in minutes)		
Sample count	**LCD**	**PC**	**BMLA**
10000	1	17	486
20000	2	55	828
50000	5	200	2077
100000	10	520	3501
150000	14	921	-

For the extended LCD and PC methods we both used a Chi-squared (χ^2) dependence test [12] with an alpha threshold of $\alpha = 0.001$ and Bonferroni correction [13] to determine independence and conditional independence between the variable pairs. In the case of BMLA, an edge probability threshold of 0.5 was used as a criterion for the presence of an edge in the predicted model. We tested the performance and efficiency of the investigated methods on 25 artificial datasets, each sampled from a randomly generated Bayesian Network consisting of 60 variables. These datasets can be split into 5 categories based on their sample count: they either have 10.000, 20.000, 50.000, 100.000 or 150.000 samples. We evaluate the output of the different methods based on 6 distinct metrics:

1) The proportion of the original causal relations that were accurately reproduced. (*Recall*)
2) The proportion of the original relationships that were reproduced with a reversed causal direction. (*Reverse*)
3) The proportion of the original relationships that where missed by the model. (*Absent*)
4) The proportion of the predicted model's causal relationships that were correct. (*Precision*)
5) The proportion of the predicted model's relationships that were entirely incorrect. (*Unnecessary*)
6) The proportion of the predicted model's relationships that were correct but with a reversed causal direction. (*Reverse*)

As it can be seen in Table II, the extended LCD method finds $54.63\% - 61.55\%$ of the causal relationships of the original graph, which can be attributed to its local approach.Among the predicted edges $58.88\% - 86.12\%$ of them are correctly directed, whereas $9.26\% - 11.17\%$ of them has an incorrect direction. Another remarkable observation is that the number of unnecessary edges increased significantly from 3.02% to 29.01% as sample size increased. This clearly indicates, that the scoring metric responsible for assessing dependence between variable pairs is sensitive to spurious correlations amplified by the increased sample size. On the other hand, the direction of edges that were present in the original model is reliably predicted in the majority of the cases, which is indicated by the ratio of reversed edges: $9.26\% - 11.17\%$.

In addition, Table III shows results of the Peter-Clark algorithm, on the same 25 datasets. These results indicate, that it has an acceptable performance on predicting the presence of relationships, that is the combined percentage of correct and reversed edges with respect to the original model. However, it misses the direction of causality in between $23.25\% - 43.23\%$ of the original relationships. Although the number of absent edges decreased with increasing sample size from 47.29% to 6.83%, the newly discovered edges were often reversed.

Finally, the predictive performance of the BMLA method is shown in Table IV. BMLA correctly identified the majority of causal relationships $70.32\% - 84\%$. The proportion of reversed edges compared to the original model remained relatively low $3.03\% - 5.29\%$. Based on these results, we can clearly draw the conclusion, that the BMLA method is capable of predicting the presence and direction of causal relationships with high precision, considering that over 90% of its predicted relationships are correct, it missed the direction of causality only $\sim 5\%$ of the time, and the ratio of unnecessary edges is low $1.90\% - 3.78\%$. Note that although the aggregated results do not indicate it directly, but the BMLA produced one outlier (divergent) result for the 20K, 50K and 100K datasets, which considerably worsened the aggregated performance of these three categories.

TABLE II
RESULTS OF THE LCD METHOD[a]

	Compared to the original model			Edges of the predicted model		
Sample count	***Recall***	***Reverse***	***Absent***	***Precision***	***Unnecessary***	***Reverse***
10000	54.63%	6.88%	38.49%	86.12%	3.02%	10.85%
20000	61.55%	6.81%	31.64%	83.67%	7.06%	9.26%
50000	60.23%	9.30%	30.47%	72.35%	16.48%	11.17%
100000	60.65%	10.30%	29.05%	65.75%	23.09%	11.17%
150000	57.13%	10.60%	32.27%	59.88%	29.01%	11.11%

[a]Averaged on 5 randomly generated models with 60 variables.

TABLE III
RESULTS OF THE PC ALGORITHM[a]

	Compared to the original model			Edges of the predicted model		
Sample count	***Recall***	***Reverse***	***Absent***	***Precision***	***Unnecessary***	***Reverse***
10000	29.46%	23.25%	47.29%	52.62%	5.85%	41.53%
20000	33.37%	29.68%	36.95%	50.00%	5.54%	44.46%
50000	44.53%	39.53%	15.93%	50.59%	4.49%	44.91%
100000	47.57%	40.51%	11.92%	51.25%	5.11%	43.64%
150000	49.94%	43.23%	6.83%	50.54%	5.72%	43.74%

[a]Averaged on 5 randomly generated models with 60 variables.

TABLE IV
RESULTS OF THE BMLA METHOD[a]

	Compared to the original model			Edges of the predicted model		
Sample count	***Recall***	***Reverse***	***Absent***	***Precision***	***Unnecessary***	***Reverse***
10000	84.00%	5.29%	10.71%	92.02%	2.19%	5.79%
20000	70.44%	3.35%	26.21%	92.99%	2.59%	4.42%
50000	74.77%	3.26%	21.98%	94.01%	1.90%	4.09%
100000	70.32%	3.03%	26.66%	92.25%	3.78%	3.97%

[a]Averaged on 5 randomly generated models with 60 variables.

IV. DISCUSSION

There are three important aspects to consider during the evaluation. (1) Due to observational equivalence, some causal relationships cannot be clearly identified, so their directionality is ambiguous. However, these directed edges can be searched efficiently when multiple models are considered [14], which could make Bayesian model averaging based global methods more accurate. (2) Local methods do not take into account higher-order dependencies, so in the case of densely connected causal-dependence structures they are not able to identify some of the relationships. (3) The presence of an edge in a Bayesian model is represented by a posterior probability for which there is no fixed acceptance threshold.

Our experiment demonstrates that in case of casual models related to moderate sized real world problems – that are more complex than the generally investigated toy examples with at most 10 variables – both local and global methods face considerable challenges. Global methods are restricted due to infeasible execution times, whereas local methods lack the precision, and in some cases even the scalability. Our extended LCD method has shown adequate results in terms of scalability, and promising results in terms of precision, outperforming the Peter-Clark algorithm in certain aspects.

REFERENCES

[1] S. Mani and G. F. Cooper, A Study in Causal Discovery from Population-Based Infant Birth and Death Records, Pittsburgh, 1999.

[2] P. Spirtes, C. Glymour, and R. Scheines, Causation, Prediction, and Search, 2nd ed. Cambridge, MA: MIT Press, 2000

[3] P. Antal, A. Millinghoffer, G. Hullám, C. Szalai and A. Falus, A bayesian view of challenges in feature selection: feature aggregation, multiple targets, redundancy and interaction, FSDM, pp. 74-89, 2008.

[4] J. Pearl, Causality: Models, Reasoning and Inference, C. U. P., 2000.

[5] J. A Hoeting and D. Madigan and A. E. Raftery and C. T. Volinsky, Bayesian Model Averaging: A Tutorial, Statistical Science, vol.4, 4, pp. 382-417, 1999.

[6] N. Friedman and D. Koller, Being Bayesian about network structure, in Mach. Learn., 2003, p. 95–125.

[7] D. Madigan, S.A. Andersson, M. Perlman and C. T. Volinsky, Bayesian model averaging and model selection for Markov equivalence classes of acyclic digraphs, Comm.Stat.Theory Meth., vol.25, pp.2493-2520, 1996.

[8] G. Cooper, and E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, Machine learning, vol.9, 4, pp.309-347, 1992.

[9] W. L. Buntine, Theory Refinement of Bayesian Networks, Proc. of the 7th Conf. on Uncertainty in AI (UAI-1991), pp.52-60, 1991.

[10] G. Hullam et al., The UKB envirome of depression: from interactions to synergistic effects, Sci Rep, 2019.

[11] J. Pearl, Causal inference in statistics: An overview, Statistics Surveys, pp. 96-146, 2009.

[12] A. Agresti, An Introduction to Categorical Data Analysis, Wiley, 2007.

[13] Bonferroni Correction, [Online]. Available: https://mathworld.wolfram.com/BonferroniCorrection.html.

[14] D. M. Chickering, C. Meek, Selective Greedy Equivalence Search: Finding Optimal Bayesian Networks Using a Polynomial Number of Score Evaluations, 2015.

Bayesian Network Multimorbidity Models in COVID-19 Mortality

Tamás Nagy, Bence Bruncsics, Péter Antal
Budapest University of Technology and Economics
Department of Measurement and Information Systems
Budapest, Hungary
Email: nagytam606@gmail.com, bruncsics@mit.bme.hu, antal@mit.bme.hu

***Abstract*—Clusters of multimorbidities indicate common molecular and physiological causal mechanisms, which can help to identify promising targets for novel drugs and treatments. We analyzed the comorbidities for COVID-19 death cases in 2020 in Hungary (9537 cases). The daily, public reports of anonymous cases contained age, sex and free-text description of comorbidities for each officially COVID-19 deceased. We cleaned, clinically encoded and aggregated the reported morbidities resulting in 22 medical categories, which are also used in a reference national database with normal Hungarian population. Using Bayesian networks, we analyzed the dependency models of reported comorbidities in different population cohorts, such as in the first wave (2020 March-August) versus in the second wave (September-December) or above versus below the expected life expectancy (76 years). To explore latent morbidities, not yet manifested or not reported, but potentially already influencing COVID-19 mortality risk, we predicted excess multimorbidity for deceased. Our results suggest significant excess multimorbidity, which also calls for an improved and more transparent data dissemination policy. Multimorbidity models, especially models also based on multimorbidity data from normal population could provide vital risk prediction for an improved health care and vaccination programs.**

***Index Terms*—multimorbidity, Bayesian network, COVID-19, text-mining, disease coding**

I. Introduction

Today the COVID-19 pandemic (caused by the SARS-CoV-2 virus) has a major impact on everyone's life. One of the most striking feature of the disease is its highly varying risk ranging from asymptomatic infection to very high mortality rates for older people with comorbidities. Indeed, comorbidities and particularly complex multimorbidities seems to be the most important risk factors in COVID-19 disease and even in COVID-19 vaccination [4], [7], [8], [11], [13], [14], [16]–[18], [21], [24]. Given the rapidly rising prevalence of multimorbidity in modern societies [10], [15], [19], [23], the investigation of this risk factor is of primary importance, especially in countries with high multimorbidity rates, such as in Hungary, wich has the highest multimorbidity rate in the European Union [1], [22].

Firstly, we describe the derivation of morbidities from free-text, public reports about COVID-19 deceased in 2020 in Hungary and discuss the quality and completeness of this information source. Secondly, we describe the application of Bayesian networks for predicting expected excess morbidity. Finally, we present statistics about multimorbidities and expected excess multimorbidities.

II. Data and Methods

We obtained the data set of the COVID-19 deceased from the official pandemic-information site of the Hungarian Government[1]. The retrieved database contains the following items:

- ID ("*Sorszám*")
- Gender ("*Nem*")
- Age ("*Kor*")
- Comorbidities ("*Alapbetegségek*") - In the format of free text

The data set contains the data of 9537 deceased people. There are slightly more males (4883, approx. 51.2%) than females, the mean age is 76 years, from this 73 years is the average age for males and 78.5 for females. More details about the age and gender distribution can be seen on Fig. 1.

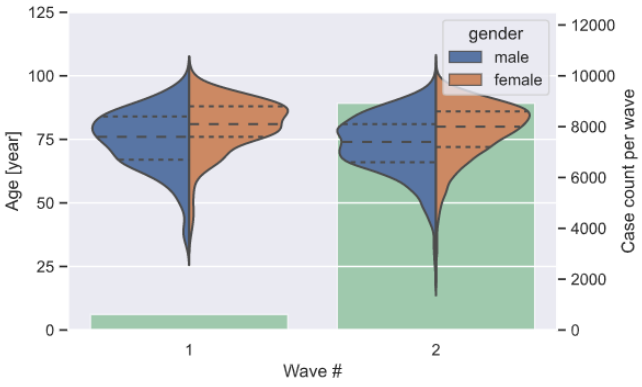


Fig. 1. The distribution of age within gender and waves. The individual violinplots are scaled by population with respect to gender, but the comparable population of each wave is represented with a green bar in the background, the dotted lines represent the quartiles. The age-gender distribution was similar during both waves, males are slightly over represented with 51.2%, the average female age is 5.5 years higher than male.

A. Separating Waves

It was a world-wide trend that during the summer the number of new COVID-19 cases started to drop, but in August

[1]https://koronavirus.gov.hu/elhunytak

the trend got reversed, the number of new cases started to increase again. This trend is valid in Hungary as well, on the first day of September, the official Hungarian source reported[2] 118 new cases of COVID-19, yielding 6257 as the number of total diagnosed cases, and 1 death (increasing the total death toll to 616). Thus, we introduced a binary variable indicating cases before and after the end of August in 2020 (Fig. 1 shows the age distribution in the first and in the second wave with 615 and 8922 cases).

B. Query for Most Common Disease Names

The retrieved data set contains the disease names as free-text, which were normalized and the free-text was split into individual terms. The normalization was done by Python's *casefold* and *lower* function, trimming the white spaces and converting UNICODE characters into their closest ASCII form with the help of the *unidecode* library.

The medical categorization was challenging and required manual processing by medical experts. It seems like the text field allowed any kind of input, which resulted in many typos in disease names. However, there are terms that were used much more frequently than others, which suggests a free-text input with suggestions.

From the most common terms, we selected every term that occurred five times or more. Then, we manually processed this list, categorizing each term and constructing regular expression queries (e.g. the most common term was "high blood-pressure" and the third was "high blood-pressure disease", these are two terms that correspond to the same disease). The regular expressions were as broad as possible to catch most of the typos and variants, but restrictive enough to make the manual exclusion of false-positives possible.

Another aspect of this issue was the generality of some terms (e.g. "renal disease") and the number of cases to fall under a given category. Thus, we made categories as specific as possible, while keeping the number of records in the category above 50. In the end, this method yielded 22 categories that we used for the analysis. We encoded every disease category by a binary variable.

We suspected that the reported data could have missing comorbidities at cases with 2-3 comorbidities, i.e. the fact that a disease is not present in the report does not mean that it was not present in the deceased. Furthermore, the apparent clustering of reported comorbidities suggested that the reported status of a given disease depends on the presence of other diseases. Figure 6 confirms this suspicion, which means that the data set can be interpreted as an incomplete, not-missing-at-random data containing only reported diseases, where the status of the not reported diseases is unknown.

C. Bayesian Network multimorbidity models

Bayesian networks were successfully applied in multimorbidity modeling [19] and already applied in COVID-19 modeling [6], [20]. Previous studies utilized causal Bayesian network analysis to investigate the bias in COVID-19 data, successfully pointing out that the seemingly protective effect of smoking is due to the sampling, and not the actual effects of smoking [5], [9].

[2]https://koronavirus.gov.hu/cikkek/118-fovel-emelkedett-beazonositott-fertozottek-szama-es-elhunyt-egy-idos-beteg

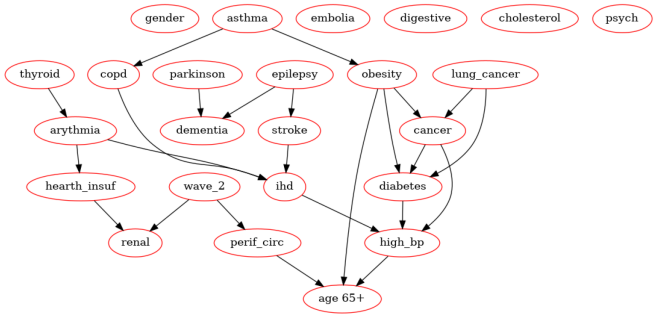


Fig. 2. Structure of the Bayesian network multimorbidity model based on the deceased in 2020 in Hungary with age below 76.

We used a Bayesian systems-based approach to filter mediated dependencies between the diseases and to estimate the *a posteriori* probabilities of direct relevances [2], [19].

We also used Bayesian networks to cope with the not reported, uncertain data. After the investigation of various missing data management schemes, we adopted a transparent and computationally effective strategy: we trained Bayesian network models using the complete data set assuming the absence of not reported diseases and we predicted the expected conditional probabilities of the not reported diseases given the reported diseases. This strategy feeds the list of known diseases into a probabilistic generative model, which yields probabilistic guesses about the missing diseases. The augmented data set with predicted probabilities, i.e., with the predicted expected values for the binary variables, was used in later analysis. With simple terms: the model made probabilistic guesses about the presence of not reported diseases based on the relationship between the reported diseases.

We used the *pomegranate* Python package to learn Bayesian Network multimorbidity models from the data set. The models were trained on different age groups, the limit for the separation was 76 years, because this is the life expectancy in Hungary (and also, this is the mean age in the database). During this step we had to dispose some categories, because they were nearly constant, i.e. only a few cases were present in the given group (namely: alcoholism and TIA). The structure of the resulting Bayesian network models can be seen in Fig. 2 and Fig. 3.

III. Results

A. Exploration of dependencies

Firstly, we calculated pairwise Pearson correlation between the variables (Fig. 4). The greatest scores among inter-disease correlations was attributed to:

- IHD (ischemic heart disease) - stroke (corr. 0.24; p-value < 0.001)

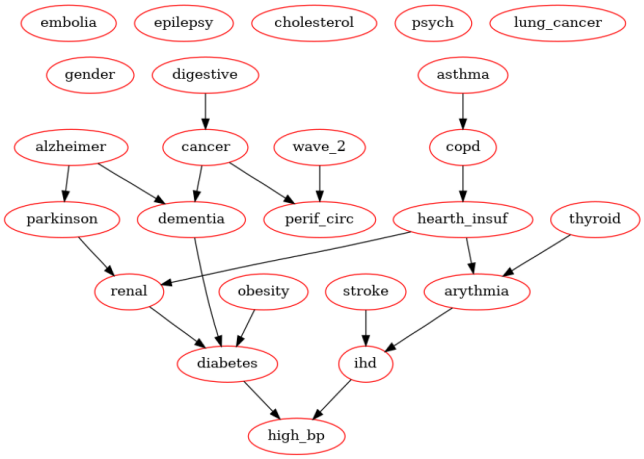


Fig. 3. Structure of the Bayesian network multimorbidity model based on the deceased in 2020 in Hungary with age above 76.

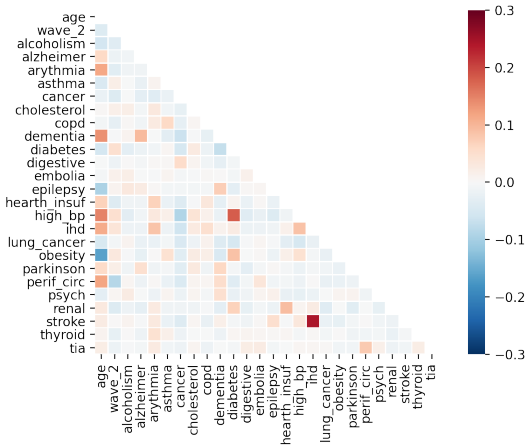


Fig. 4. Pearson correlation between variables. This analysis contains two categories, which were later excluded because of low sample count (below 50: TIA=38, alcoholism=48).

- diabetes - high blood pressure (BP) (corr. 0.18; p-value < 0.001)

Besides that age correlated with many diseases, but this findings are expected; IHD is related to stroke and diabetes is associated with high blood pressure, and these are significant predictors of mortality [3], [12].

To explore the direct, unmediated dependencies, we applied a Bayesian systems-based approach [2], [19] (see 5).

B. Observed and predicted multimorbidities

In addition to the detailed dependency maps of multimorbidities, we also calculated to distribution of multimorbidities. Fig. 6 shows the distribution of aggregated multimorbidities, including the case of complex multimorbidities, which denotes 3 or more chronic conditions [25].

The aggregated multimorbidity curves for the populations with age below and above 76 years follow the same slope and they do not separate, which is not compatible with already

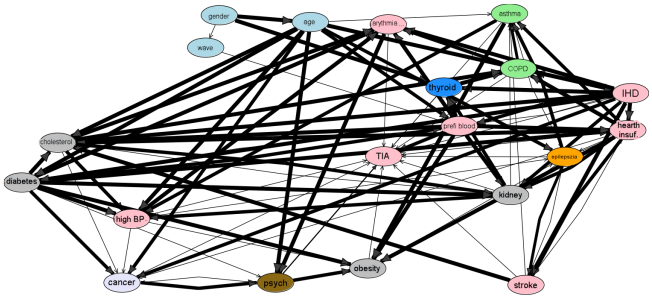


Fig. 5. Direct, unmediated dependencies between COVID-19 comorbidities. The colors denote the type of the variable and group the diseases into broader categories.

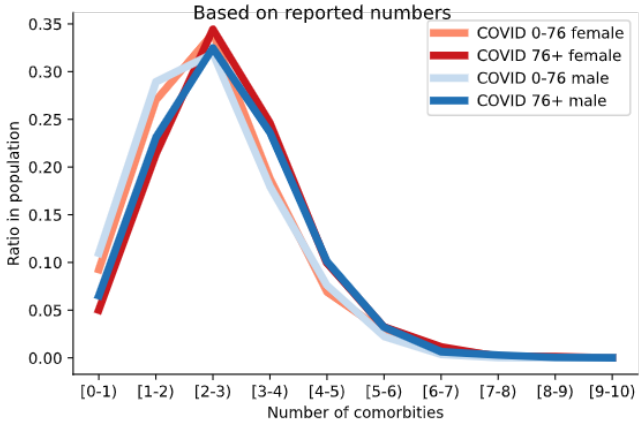


Fig. 6. Reported multimorbidity for age and gender groups from the officially reported data.

established multimorbidity trends [8], [15], especially with the uniquely high rate of multimorbidity in Hungary [1], [22]. This suggests that only a few comorbidities are reported and the rest remains undocumented. Using our trained Bayesian network multimorbidity models, we predicted this excess, latent multimorbidity as described in Section II-C. Fig. 6 shows the distribution of aggregated multimorbidities using the sum of the observed and expected value of excess multimorbidities.

Notably, the prediction and incorporation of the expected excess multimorbidity shifts the multimorbidity curves for the older population towards more realistic, higher values, i.e., the age and gender differences became more emphasized, even though the source of the extrapolation is the same data.

We have also looked for differences between multimorbidity dependencies with respect to chronological time, especially by comparing the death cases of the first and second waves, but we did not found any discrepancy in this regard.

IV. Conclusion

Our findings suggest that the reporting policy of comorbidities for COVID-19 deceased is highly incomplete, which hinders the analysis of risk factors based on multimorbidities. Because multimorbidities, especially complex multimorbidities are critical factors in COVID-19 risk prediction, this limits

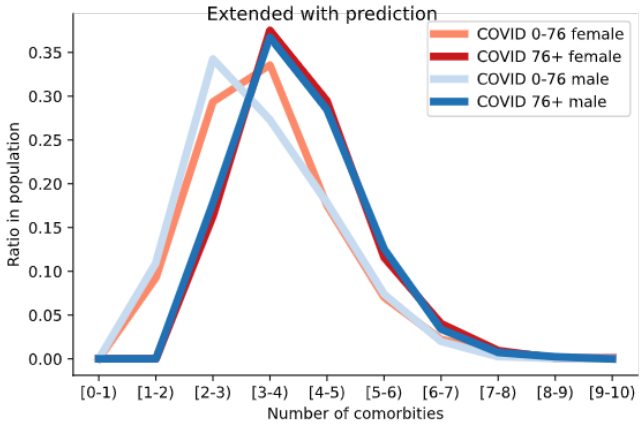


Fig. 7. Complemented multimorbidity for age and gender groups with added expected value of excess multimorbidities.

the usability of comorbidity data of COVID-19 deceased to construct preventive diagnostics systems and effective vaccination policies. However, the described statistical approach could be utilized to predict the latent, excess multimorbidity, if a representative national multimorbidity data set could be used as a reference.

References

[1] Sara Afshar, Paul J Roderick, Paul Kowal, Borislav D Dimitrov, and Allan G Hill. Multimorbidity and the inequalities of global ageing: a cross-sectional study of 28 countries using the world health surveys. *BMC Public Health*, 15(1):776, 2015.

[2] Péter Antal, András Millinghoffer, Gábor Hullám, Csaba Szalai, and András Falus. A bayesian view of challenges in feature selection: feature aggregation, multiple targets, redundancy and interaction. In *New Challenges for Feature Selection in Data Mining and Knowledge Discovery*, pages 74–89, 2008.

[3] Michael J. Cryer, Tariq Horani, and Donald J. DiPette. Diabetes and Hypertension: A Comparative Review of Current Guidelines. *Journal of Clinical Hypertension*, 18(2):95–100, 2016.

[4] Xiaoyu Fang, Shen Li, Hao Yu, Penghao Wang, Yao Zhang, Zheng Chen, Yang Li, Liqing Cheng, Wenbin Li, Hong Jia, et al. Epidemiological, comorbidity factors with severity and prognosis of covid-19: a systematic review and meta-analysis. *Aging (Albany NY)*, 12(13):12493, 2020.

[5] Norman Fenton. A note on'collider bias undermines our understanding of covid-19 disease risk and severity'and how causal bayesian networks both expose and resolve the problem. *arXiv preprint arXiv:2005.08608*, 2020.

[6] Norman Fenton, Scott McLachlan, Peter Lucas, Kudakwashe Dube, Graham Hitman, Magda Osman, Evangelia Kyrimi, and Martin Neil. A privacy-preserving bayesian network model for personalised covid19 risk assessment and contact tracing. *medRxiv*, 2020.

[7] Julian Alfredo Fernandez-Nino, Jhon A Guerra-Gomez, and Alvaro Javier Idrovo-Velandia. Multimorbidity patterns among covid-19 deaths: considerations for a better medical practice. *medRxiv*, 2020.

[8] Miguel T Froes, Bernardo Duque Neves, Bruno Martins, and Mario J Silva. Comparison of multimorbidity in covid-19 infected and general population in portugal. *medRxiv*, 2020.

[9] Gareth Griffith, Tim T Morris, Matt Tudball, Annie Herbert, Giulia Mancano, Lindsey Pike, Gemma C Sharp, Tom M Palmer, George Davey Smith, Kate Tilling, et al. Collider bias undermines our understanding of covid-19 disease risk and severity. *medRxiv*, 2020.

[10] Bruce Guthrie, Boikanyo Makubate, Virginia Hernandez-Santiago, and Tobias Dreischulte. The rising tide of polypharmacy and drug-drug interactions: population database analysis 1995–2010. *BMC medicine*, 13(1):74, 2015.

[11] Stephanie L Harrison, Elnara Fazio-Eynullayeva, Deirdre A Lane, Paula Underhill, and Gregory YH Lip. Comorbidities associated with mortality in 31,461 adults with covid-19 in the united states: A federated electronic medical record analysis. *PLoS medicine*, 17(9):e1003321, 2020.

[12] Akhtar Hussain, Bishwajit Bhowmik, and Nayla Cristina do Vale Moreira. Covid-19 and diabetes: Knowledge in progress. *Diabetes research and clinical practice*, page 108142, 2020.

[13] Guido Iaccarino, Guido Grassi, Claudio Borghi, Claudio Ferri, Massimo Salvetti, and Massimo Volpe. Age and multimorbidity predict death among covid-19 patients: results of the sars-ras study of the italian society of hypertension. *Hypertension*, 76(2):366–372, 2020.

[14] Shweta Jakhmola, Omkar Indari, Budhadev Baral, Dharmendra Kashyap, Nidhi Varshney, Ayan Das, Sayantani Chatterjee, and Hem Chandra Jha. Comorbidity assessment is essential during covid-19 treatment. *Frontiers in physiology*, 11, 2020.

[15] Andrew Kingston, Louise Robinson, Heather Booth, Martin Knapp, Carol Jagger, and MODEM project. Projections of multi-morbidity in the older population in england to 2035: estimates from the population ageing and care simulation (pacsim) model. *Age and ageing*, 47(3):374–380, 2018.

[16] Ernesto Maddaloni, Luca D'Onofrio, Francesco Alessandri, Carmen Mignogna, Gaetano Leto, Giuseppe Pascarella, Ivano Mezzaroma, Miriam Lichtner, Paolo Pozzilli, Felice Eugenio Agrò, et al. Cardiometabolic multimorbidity is associated with a worse covid-19 prognosis than individual cardiometabolic risk factors: a multicentre retrospective study (covidiab ii). *Cardiovascular diabetology*, 19(1):1–11, 2020.

[17] Frances S Mair, Hamish ME Foster, and Barbara I Nicholl. Multimorbidity and the covid-19 pandemic–an urgent call to action, 2020.

[18] Alessandra Marengoni, Alberto Zucchelli, Davide Liborio Vetrano, Andrea Armellini, Emanuele Botteri, Franco Nicosia, Giuseppe Romanelli, Eva Andrea Beindorf, Paola Giansiracusa, Emirena Garrafa, et al. Beyond chronological age: Frailty and multimorbidity predict in-hospital mortality in patients with coronavirus disease 2019. *The Journals of Gerontology: Series A*, 2020.

[19] Peter Marx, Peter Antal, Bence Bolgar, Gyorgy Bagdy, Bill Deakin, and Gabriella Juhasz. Comorbidities in the diseasome are more apparent than real: What bayesian filtering reveals about the comorbidities of depression. *PLoS computational biology*, 13(6):e1005487, 2017.

[20] Scott McLachlan, Peter Lucas, Kudakwashe Dube, GS McLachlan, GA Hitman, M Osman, and NE Fenton. The fundamental limitations of covid-19 contact tracing methods and how to resolve them with a bayesian network approach, 2020.

[21] Ross McQueenie, Hamish Foster, Bhautesh D Jani, Srinivasa Vittal Katikireddi, Naveed Sattar, Jill P Pell, Frederick K Ho, Claire L Niedzwiedz, Claire E Hastie, Jana Anderson, et al. Multimorbidity, polypharmacy, and covid-19 infection within the uk biobank cohort. *medRxiv*, 2020.

[22] Raffaele Palladino, John Tayu Lee, Mark Ashworth, Maria Triassi, and Christopher Millett. Associations between multimorbidity, healthcare utilisation and health status: evidence from 16 european countries. *Age and ageing*, 45(3):431–435, 2016.

[23] Anna J Koné Pefoyo, Susan E Bronskill, Andrea Gruneir, Andrew Calzavara, Kednapa Thavorn, Yelena Petrosyan, Colleen J Maxwell, YuQing Bai, and Walter P Wodchis. The increasing burden and complexity of multimorbidity. *BMC public health*, 15(1):415, 2015.

[24] Peishan Qiu, Yunjiao Zhou, Fan Wang, Haizhou Wang, Meng Zhang, Xingfei Pan, Qiu Zhao, and Jing Liu. Clinical characteristics, laboratory outcome characteristics, comorbidities, and complications of related covid-19 deceased: a systematic review and meta-analysis. *Aging clinical and experimental research*, pages 1–10, 2020.

[25] Siri H Storeng, Kristin H Vinjerui, Erik R Sund, and Steinar Krokstad. Associations between complex multimorbidity, activities of daily living and mortality among older norwegians. a prospective cohort study: the hunt study, norway. *BMC geriatrics*, 20(1):21, 2020.