Discussion: Weaving temporal background knowledge into the counterfactual explanation generation process

Andrei Buliga[1,2,*], Chiara Di Francescomarino[3], Chiara Ghidini[2], Marco Montali[2] and Massimiliano Ronzani[1]

[1]Fondazione Bruno Kessler, Trento, Italy
[2]Free University of Bozen-Bolzen, Bolzano, Italy
[3]University of Trento, Trento,Italy

Abstract

Advances in Machine Learning have led to the adoption of ensemble and deep learning techniques for more accurate predictive models. However, these "black-box" models pose interpretability challenges. Explainable Artificial Intelligence (XAI) methods, like counterfactual explanations, have emerged to address this issue by proposing changes to input data that alter model predictions. Yet, current techniques often overlook temporal event data and background knowledge. To bridge this gap, we introduce a framework integrating background knowledge expressed through Linear Temporal Logic on finite traces (LTL_f) into counterfactual generation. We achieve this through three strategies within a Genetic Algorithm (GA)-based method. Empirical evaluations confirm the effectiveness of our approach, especially highlighting the potential of online approaches for generating counterfactual explanations in temporal data. This work enhances model interpretability by aligning counterfactuals with background knowledge, thereby improving the utility of "black-box" predictive models.

Keywords

counterfactuals explanations, process pattern, explainable AI

1. Introduction

State-of-the-art efforts in Machine Learning have focused on delivering accurate predictive models through the application of ensemble learning and deep learning techniques. Due to the inherent complexity of these models, commonly referred to as *black-box* models, they are often challenging for users to comprehend. For this reason, eXplainable Artificial Intelligence (XAI) techniques have been recently adopted to help interpret their predictions of these advanced predictive models [1, 2, 3, 4].

Counterfactual explanations are XAI techniques suggesting changes needed in an input instance to alter the outcome of a prediction by a black-box model. While existing methods focus on optimizing minimal changes to flip predictions, they lack consideration for dependencies within temporal data represented as event traces. These sequences capture events occurring to an object over time, common in domains like healthcare or business processes. Incorporating background knowledge, especially in the form of domain constraints, is crucial. For instance, suggesting a loan application be submitted before filling it out to flip the prediction would be impractical, violating logical event sequences.

In this work [1], we propose a novel approach to generate counterfactual explanations for temporal event traces, integrating LTL_f formulae as *temporal background knowledge* to ensure satisfaction of specified constraints. We adapt a Genetic Algorithm (GA)-based method, leveraging deterministic finite-state automata (DFA) to incorporate *temporal background knowledge*. Specifically, we modify

EKAPI 2024: 1st International Workshop On Explainable Knowledge Aware Process IntelligencE, June 20-22, 2024, Roccella Jonica (RC), Italy

[*]Corresponding author.

[†]These authors contributed equally.

abuliga@fbk.eu (A. Buliga); c.difrancescomarino@unitn.it (C. D. Francescomarino); chiara.ghidini@unibz.it (C. Ghidini); marco.montali@unibz.it (M. Montali); mronzani@fbk.eu (M. Ronzani)

0000-0002-8179-9146 (A. Buliga); 0000-0002-0264-9394 (C. D. Francescomarino); 0000-0003-1563-4965 (C. Ghidini); 0000-0002-8021-3430 (M. Montali); 0000-0001-7277-2999 (M. Ronzani)

[1]This is a discussion paper of a conference paper submitted at European Conference of Artificial Intelligence (ECAI) 2024.

the GA's fitness function, as well as crossover and mutation operators, to integrate a set of LTL$_f$ formulae represented through their conjunction. We explore three strategies for combining GA and DFA: one extracts DFA information in an *a priori* manner, while two guide the GA *online* using extracted DFA information to avoid *temporal background knowledge* violations. This approach aims to generate counterfactuals satisfying the formulae while adhering to standard counterfactual desiderata. The results suggest that the approaches that include the *temporal background knowledge* can produce better counterfactuals explanations, especially in satisfying the LTL$_f$ formulae, whilst maintaining or improving counterfactual desiderata. Thus, this work enhances model interpretability by aligning counterfactuals with *temporal background knowledge*, improving the practical utility of "black-box" models.

2. Linear Temporal Logic on Finite Traces

Let $\mathcal{V}$ be a universe of propositional variables, and Σ a subset of $\mathcal{V}$ ($\Sigma \subseteq \mathcal{V}$). An LTL$_f$ formula φ is built over the propositional variables in Σ, by using the Boolean connectives $\neg$, $\wedge$, $\vee$, and $\Rightarrow$, together with several temporal operators [5]. In this paper, we focus on the temporal operators $\mathbf{X}$ (next), $\mathbf{X}_{\mathrm{w}}$ (weak next), $\mathbf{G}$ (always), $\mathbf{F}$ (eventually), and $\mathbf{U}$ (until). Formally, φ is built according to the grammar:

$$\varphi ::= \sigma \mid \neg\varphi \mid (\varphi_1 \vee \varphi_2) \mid \mathbf{X}(\varphi) \mid (\varphi\, \mathbf{U} \varphi), \text{ where } \sigma \in \Sigma.$$

As customary, the other boolean connectives are derived as $\varphi_1 \wedge \varphi_2 = \neg(\neg\varphi_1 \vee \neg\varphi_2)$, and $\varphi_1 \Rightarrow \varphi_2 = \neg\varphi_1 \vee \varphi_2$. We define $true = \sigma \vee \neg\sigma$ for some $\sigma \in \Sigma$. Furthermore, temporal operators $\mathbf{X}_{\mathrm{w}}$, $\mathbf{G}$, and $\mathbf{F}$ are derived as: $\mathbf{X}_{\mathrm{w}}(\varphi) = \neg\mathbf{X}(\neg\varphi)$, $\mathbf{F}(\varphi) = true\mathbf{U}\varphi$, and $\mathbf{G}(\varphi) = \neg\mathbf{F}(\neg\varphi)$.

A finite, non-empty, sequence π over the propositional variables in Σ is a sequence $\pi = (\pi_1, \pi_2, .., \pi_n)$, associating each $i \in 1, ..., n$ to a state $\pi_i \in \Sigma$, consisting of the set of all propositional variables that are assumed to hold at instant i. In the context of this paper, consistently with the literature on (business) process execution traces, we make the simplifying assumption that at each point of the sequence, one and only one propositional symbol from Σ holds, also known as the DECLARE assumption (see [6] for details). Each LTL$_f$ formula φ can be associated with a corresponding deterministic finite-state automaton (DFA), which can allow one to do reasoning in LTL$_f$ [7]. Given φ, a DFA$_\varphi$ is a tuple $M = (Q, \Sigma, \delta, q_0, F)$, where: (i) Q is a finite set of states, (ii) Σ is the input alphabet, a finite set of symbols, (iii) δ is the transition function, $\delta : Q \times \Sigma \to Q$ (iv) $q_0 \subseteq Q$ is the initial state, (v) $F \subseteq Q$ is the set of final states. A DFA can be converted into a symbolic DFA by leveraging the previously mentioned Declare assumption. We utilize this approach to simplify the propositional variables used to constrain genetic operators.

3. Counterfactual explanations and Genetic algorithms

Counterfactual explanations belong to the family of XAI methods. However, compared to other types of XAI methods, such as feature attribution methods [1], counterfactuals do not attempt to explain the inner logic of a predictive model but instead offer an alternative to the user to obtain a desired prediction [8]. When dealing with black-box models, indeed, the internal logic of a model h_θ mapping a sample $\mathbf{x}$ to a label y is unknown, or otherwise uninterpretable to humans. A counterfactual $\mathbf{c}$ of $\mathbf{x}$ is a sample for which the prediction of the black box is different from the one of $\mathbf{x}$ (i.e., $h_\theta(\mathbf{c}) \neq h_\theta(\mathbf{x})$).

The XAI literature has detailed the desiderata that a counterfactual explanation should possess [1, 2]: (i) *Validity:* Counterfactuals should flip the prediction of the original input to align with the desired class. (ii) *Input Closeness:* Counterfactuals should minimize changes for better understanding. (iii) *Sparsity:* Counterfactuals should change as few attributes as possible for conciseness. (iv) *Plausibility:* Counterfactuals should adhere to observed correlations among features for feasibility. (v) *Causality:* Counterfactuals should respect known causal relations between features for realism and actionability. (vi) *Diversity:* Counterfactuals in a set should maximize differences to offer diverse alternatives.

Genetic Algorithms (GAs) represent a powerful class of optimisation techniques, drawing inspiration from the natural processes of evolution, used across diverse domains due to their effectiveness in addressing intricate optimisation problems [9]. GAs operate within a population of potential solutions, evaluating their quality through a *fitness function.*

In the case of counterfactual generation, the initial population is a starting set of counterfactual candidates [10, 3]. The *fitness function* helps select the most promising counterfactuals to move into the mating phase, based on some desired properties.The problem can be framed as either a single-objective or multi-objective optimisation, depending on the desired properties of the generated counterfactuals and potential trade-offs. The fittest counterfactual solutions will undergo the crossover and mutations operators to produce better offsprings, i.e., more suitable counterfactual explanations given the input sample of interest [3]. Both operators help maintain the diversity of the generated counterfactuals in C.

4. Approach

Our novel framework merges genetic algorithms (GAs) with finite-state automata to generate counterfactual explanations adhering to LTL$_f$ formulae. By converting LTL$_f$ formulae into deterministic finite-state automata (DFA), we ensure reliability and adherence to specified background knowledge. We propose three strategies within the modified GA: one extracts DFA information in advance, while the other two dynamically navigate the DFA online to meet temporal constraints. All methods guarantee satisfaction of LTL$_f$ formulae while preserving diversity in generated counterfactual explanations

4.1. Optimisation problem formulations for counterfactual explanations

We introduce the formulation of the optimisation problem for the generation of general counterfactual explanations. To this end, we make use of the objective function introduced in Buliga et al. [11], specifically the single-objective baseline and adapted fitness functions.

The baseline fitness function, is composed of the following fitness function, which we call $\mathbf{f}_{\text{baseline}}$:

$$\mathbf{f}_{\text{baseline}} := o_1(h_\theta(\mathbf{c}), y) + \alpha \cdot o_2(\mathbf{x}, \mathbf{c}) + \beta \cdot o_3(\mathbf{x}, \mathbf{c}) + \gamma \cdot o_4(\mathbf{c}, \mathcal{X}) \tag{1}$$

where α, β, γ, as factors used to control the influence of each term on the fitness. The objectives o_1, o_2, o_3, and o_4 gauge the (i) validity, (ii) input closeness, (iii) sparsity, and (iv) plausibility of a counterfactual.

To extend the baseline fitness function, a fifth objective o_5 is introduced, aiming to address the *causality* (desideratum (v)) directly in the fitness function, that is the adherence to the *temporal background knowledge* that we assume to be available in the form of a LTL$_f$ formula. To measure this aspect, we transform the LTL$_f$ formula into a Deterministic Finite Automaton (DFA) and check whether by traversing the automaton we end up in a final state. Formally, we say that o_5 measures if $\mathbf{c} \models \varphi$, that is if $\mathbf{c}$ is compliant to φ: $o_5(\mathbf{c}, \varphi) = \mathbf{c} \models \varphi$. The adapted fitness function, called $\mathbf{f}_{\text{adapted}}$, is formatted as:

$$\mathbf{f}_{\text{adapted}} = o_1(h_\theta(\mathbf{c}), y) + \alpha \cdot o_2(\mathbf{x}, \mathbf{c}) + \beta \cdot o_3(\mathbf{x}, \mathbf{c}) + \gamma \cdot o_4(\mathbf{c}, \mathcal{X}) + \delta \cdot o_5(\mathbf{c}, \varphi) \tag{2}$$

where $\alpha, \beta, \gamma, \delta$ are weighting factors for each term, controlling their influence on the fitness.

4.2. Temporal knowledge-aware crossover and mutation operators

Adapted Crossover operator: The Adapted Crossover Operator, presented in Algorithm **??**, generates offspring individuals from two parent individuals ($\mathbf{P_1}$ and $\mathbf{P_2}$), as the classical crossover operator, while however guaranteeing that the LTL$_f$ formula satisfied in the original query instance $\mathbf{x}$ is still satisfied for the offspring $\mathbf{O}$. It takes as input the original query instance $\mathbf{x}$, two parent individuals, $\mathbf{P_1}$ and $\mathbf{P_2}$, the crossover probability p_c and the alphabet Σ of the activities defining the LTL$_f$ formula φ.

The Adapted Crossover Operator starts by initiating an offspring individual ($\mathbf{O}$) by retaining from the original query instance $\mathbf{x}$ the phenotype that enables the satisfaction of the LTL$_f$ formula φ, that is, any propositional variables that appear in Σ. This ensures that if φ is satisfied in $\mathbf{x}$, it is satisfied also in $\mathbf{O}$. The empty features in the offspring individual phenotype are then filled with one of the two parents' genetic material, but only if the corresponding parent's feature is not in Σ. In detail, a random probability p is sampled for every empty feature. The genetic material is then chosen, as in classical crossover operators, from either parent $\mathbf{P_1}$ or $\mathbf{P_2}$, according to a given crossover probability p_c, if the parent's feature does not belong to Σ. Otherwise, if both parents' features belong to Σ, the crossover operator uses the corresponding segment from the original query instance $\mathbf{x}$. This ensures that no new constraint activations are introduced in the offspring.

Adapted Mutation operator: As we are constraining the mutation operator, which is designed to increase the overall diversity of the population, we devise a series of strategies to maintain the diversity of the generated offsprings without violating φ. To do so, given an LTL$_f$ formula φ, we transform it into a Deterministic Finite Automaton (DFA, see Section 2) $M = (Q, \Sigma, \delta, q_0, F)$ and devise three strategies. The first strategy is done before the Genetic algorithm (GA) iteration cycle (*a priori*), while the last two strategies are done while performing the mating operations (*online*). These strategies affect which features can be mutated and which values they can assume to remain compliant with φ.

Consider an offspring of the crossover operation $\mathbf{O} = (\mathbf{O}_i \mid i \in \{1, \dots, |\mathbf{O}|\})$ that we want to mutate, and the set of all possible values that the i-th feature $\mathbf{O}_i$ can take ($\mathcal{D}_i$). The a priori strategy defines a subset $\mathcal{D}_{\text{safe},i} \subset \mathcal{D}_i$ of possible mutations for each feature i, which always guarantee the compliance with φ. **`Outback`:** It mutates only features $\mathbf{O}_i$ such that $\mathbf{O}_i \notin \Sigma$. The domain of the mutation is limited to the values that do not appear in Σ, i.e., $\mathcal{D}^{h1}_{\text{safe},i} = \mathcal{D}_i \setminus \Sigma$.

The second strategy defines the domain for the mutation $\mathcal{D}_{\text{safe},i}$ in an online way, based on the state in the DFA of the current prefix. **`Pathfinder`:** It can mutate any features $\mathbf{O}_i, i = 1, \dots, |\mathbf{O}|$ The domain of the mutation is defined as follows. Let q_i be the state of the DFA associated with the partial trace $\mathbf{O}_{:i} = (\mathbf{O}_1, \dots, \mathbf{O}_i)$. The strategy defines the subset of the domain $\mathcal{D}_i$ of safe mutations as $\mathcal{D}^{h2}_{\text{safe},i} = \mathcal{D}_i \cap A^{\text{safe}}$, where $A^{\text{safe}} = \{\sigma \in \mathcal{V} \mid \delta(q_{i-1}, \sigma) = q_i\}$. In other words, this strategy allows replacing $\mathbf{O}_i$ with any other propositional variable that causes the same transition in the DFA.

Finally, the third strategy, differently from the previous ones, does not constrain the domain of the mutation operation, but performs the mutation and checks, a posteriori, if the mutated offspring is compliant with the formula $\mathbf{O} \models \varphi$. **`Freelander`:** It tries to mutate the feature $\mathbf{O}_i$ of the offspring $\mathbf{O}$, using the full domain $\mathcal{D}_i$ of that feature. If the mutated offspring satisfies the formula φ the mutation is kept, otherwise a new mutation attempt is tried.

The Adapted Mutation Operator focuses on mutating an offspring individual ($\mathbf{O}$) while preserving the satisfaction of the formula φ. The operator takes as input the offspring $\mathbf{O}$, the mutation probability p_m, the set of the domain of the features $D = \{\mathcal{D}_i \mid i \in \{1, \dots, |\mathbf{O}|\}\}$, and the mutation strategy S, that is one of the three strategies defined above, returning the mutated offspring individual as output.

5. Datasets and Evaluation settings

Claim Management [12] The synthetic dataset of Claim Management consists of 4800 traces with an average trace length of 11 events. The dataset contains 52 935 events and 16 activities. For this dataset, the length of the alphabet is $|\mathcal{V}| = 16$.

BPIC2012. [13] This dataset, based on the Business Process Intelligence Challenge (BPIC) of 2012, includes the loan application process execution of a bank. The BPIC2012 event logs consists of 4685 traces with an average length of 35 events. The dataset contains 186 693 events and 36 different activities. Thus, for this dataset, we have the length of the alphabet $\mathcal{V}$, $|\mathcal{V}| = 36$.

To evaluate the performance of the selected counterfactual approaches, we utilize the evaluation protocol presented in [14], with its afferent seven evaluation metrics. The evaluation metrics include: *distance*, *sparsity*, *implausibility*, *satisfaction rate*, *diversity*, *hit rate*, and *runtime*.

Experiments were done to evaluate proposed strategies across various datasets, trace lengths, and requested counterfactuals. We also explored different LTL$_f$ formulae with varying alphabet sizes Σ. We assessed the impact of requested counterfactuals on results, execution time, and satisfiability. For φ, we tested LTL$_f$ formulae with Σ representing coverage percentages of possible activities $\mathcal{V}$, ranging from 10% to 50%. Each Σ had tailored LTL$_f$ formulae for specific datasets and coverage percentages.

6. Results

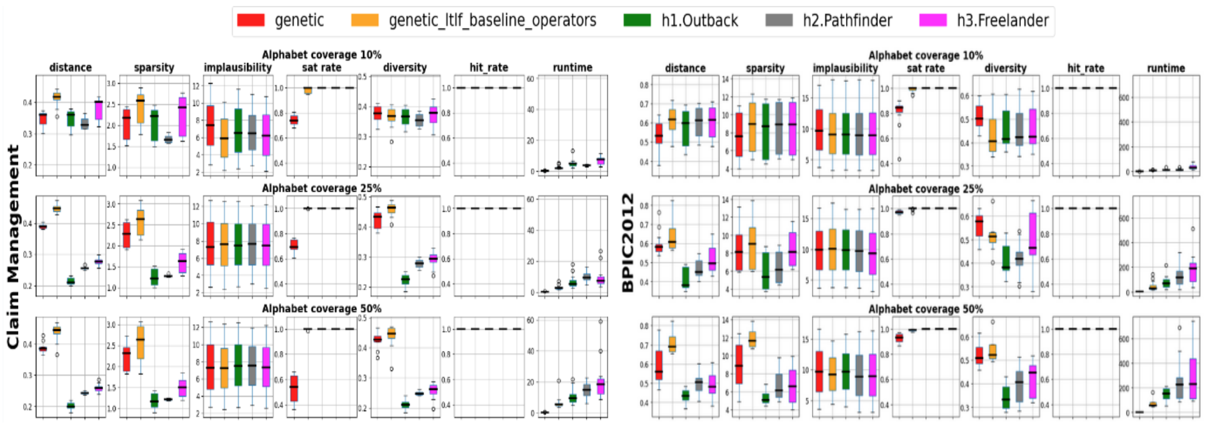

Figure 1: Overall evaluation results for the Claim Management and BPIC2012 datasets across all method, strategies, prefix lengths

Figure 1 displays evaluation results for both the Claim Management and BPIC2012 datasets. We compare the three proposed strategies against the two baseline methods, `genetic`, and `ltlf_baseline`. `genetic`, uses the fitness function (1), and `ltlf_baseline`, incorporating the adapted fitness function (2), but using the standard crossover and mating operators. This lets us compare using only the adapted fitness function, versus combining it with the proposed crossover and mutation operators and associated strategies. We then compare results among the three proposed strategies. Finally, we observe how the percentage of the alphabet $|\Sigma|/|\mathcal{V}|$ used to build φ influences the outcomes of generated counterfactuals.

In Figure 1, the proposed strategies consistently yield counterfactuals closer to the input instance, with lower *distance* and *sparsity* compared to the baselines. They make fewer changes, especially with increased coverage of Σ over $\mathcal{V}$. All proposed strategies consistently satisfy the LTL$_f$ formula, unlike the baselines, which violate the given background knowledge. For *diversity*, the proposed strategies yield comparable results, contingent on φ. In terms of *runtime*, the proposed strategies require more time, though achieving comparable results in some cases. Overall, proposed methods generate counterfactuals with lower *distance* and *sparsity*, maximizing *satisfaction rate*, with a trade-off in *diversity* compared to baselines. Counterfactuals from baseline methods risk violating φ, especially for `genetic`.

In Fig. 1, differences among the proposed strategies are evident. The `Outback` consistently yields the lowest *distance* scores compared to the online strategies (`Pathfinder` and `Freelander`), which tend to have higher *distance* scores. Across various settings, no significant differences are observed in *sparsity* and *implausibility*. All strategies achieve 100% in *satisfaction rate* and *hit rate*. Regarding *diversity*, the online strategies, particularly `Freelander`, produce the most diverse counterfactuals, followed by `Pathfinder`. As for *runtime*, `Outback` generally requires less time, while `Pathfinder` and `Freelander` have higher runtimes. Overall, `Freelander` shows the highest variability in *runtime*, while consistently generating counterfactuals with the highest *diversity* that satisfy φ. However, it also exhibits the most variability in *distance*, *sparsity*, and *runtime*. Conversely, `Pathfinder` provides the best trade-off across all metrics. In summary, the proposed methods offer different trade-offs: `Freelander` is optimal for diverse counterfactuals satisfying φ, `Pathfinder` is preferable for closer and sparser ones, and `Outback` offers lower runtime, *distance*, and *sparsity*, albeit with lower *diversity*.

As alphabet coverage increases, differences between the methods also grow, particularly in *distance*, *sparsity*, *diversity*, and *satisfaction rate*. At 10% coverage, little variation is observed in these metrics for proposed methods. Increasing coverage decreases *diversity* for proposed strategies but also decreases *distance* and *sparsity* compared to baselines. Moreover, with increasing coverage, baseline methods, especially `genetic`, exhibit more violations of φ. The enhancement of `ltlf_baseline` with larger coverage can be attributed to its use of o_5, enabling faster convergence and higher *satisfaction rate*.

7. Conclusions

In this paper, we introduced a novel framework for generating counterfactuals for Linear Temporal Logic on finite traces by combining Deterministic Finite Automata (DFAs) and Genetic Algorithms (GAs), through different strategies. The results of the evaluation show that the proposed strategies ensure the satisfaction of the φ while maintaining or improving general counterfactual explanation desiderata when compared to state-of-the-art counterfactual generation methods.

Acknowledgments

This work was partially supported by the Italian (MUR) under PRIN project PINPOINT Prot. 2020FNEB27, CUP H23C22000280006 and H45E21000210001 and PNRR project FAIR-Future AI Research (PE00000013), under the NRRP MUR program funded by NextGenerationEU.

References

[1] S. Verma, J. P. Dickerson, K. Hines, Counterfactual explanations for machine learning: A review, CoRR abs/2010.10596 (2020).

[2] R. Guidotti, Counterfactual explanations and how to find them: literature review and benchmarking, Data Mining and Knowledge Discovery (2022). URL: https://doi.org/10.1007/s10618-022-00831-6. doi:10.1007/s10618-022-00831-6.

[3] S. Dandl, C. Molnar, M. Binder, B. Bischl, Multi-objective counterfactual explanations, in: T. Bäck, M. Preuss, A. Deutz, H. Wang, C. Doerr, M. Emmerich, H. Trautmann (Eds.), Parallel Problem Solving from Nature – PPSN XVI, Springer International Publishing, Cham, 2020, pp. 448–469.

[4] K. Beckh, S. Müller, M. Jakobs, V. Toborek, H. Tan, R. Fischer, P. Welke, S. Houben, L. von Rueden, Harnessing prior knowledge for explainable machine learning: An overview, in: 2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML), 2023, pp. 450–463. doi:10.1109/SaTML54575.2023.00038.

[5] G. De Giacomo, M. Y. Vardi, Linear temporal logic and linear dynamic logic on finite traces, in: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, IJCAI '13, AAAI Press, 2013, p. 854–860.

[6] G. De Giacomo, R. De Masellis, M. Montali, Reasoning on ltl on finite traces: Insensitivity to infiniteness, Proceedings of the AAAI Conference on Artificial Intelligence 28 (2014). URL: https://ojs.aaai.org/index.php/AAAI/article/view/8872. doi:10.1609/aaai.v28i1.8872.

[7] G. De Giacomo, M. Favorito, Compositional approach to translate ltlf/ldlf into deterministic finite automata, Proceedings of the International Conference on Automated Planning and Scheduling 31 (2021) 122–130. URL: https://ojs.aaai.org/index.php/ICAPS/article/view/15954. doi:10.1609/icaps.v31i1.15954.

[8] S. Wachter, B. D. Mittelstadt, C. Russell, Counterfactual explanations without opening the black box: Automated decisions and the GDPR, CoRR abs/1711.00399 (2017).

[9] M. Mitchell, An Introduction to Genetic Algorithms, MIT Press, Cambridge, MA, USA, 1998.

[10] M. Schleich, Z. Geng, Y. Zhang, D. Suciu, Geco: Quality counterfactual explanations in real time, Proc. VLDB Endow. 14 (2021) 1681–1693. URL: https://doi.org/10.14778/3461535.3461555. doi:10.14778/3461535.3461555.

[11] A. Buliga, C. D. Francescomarino, C. Ghidini, I. Donadello, F. M. Maggi, Guiding the generation of counterfactual explanations through temporal background knowledge for predictive process monitoring, CoRR abs/2403.11642 (2024). URL: https://doi.org/10.48550/arXiv.2403.11642. doi:10.48550/ARXIV.2403.11642. arXiv:2403.11642.

[12] W. Rizzi, C. Di Francescomarino, F. M. Maggi, Explainability in predictive process monitoring: When understanding helps improving, in: International Conference on Business Process Management, Springer, 2020, pp. 141–158.

[13] B. van Dongen, Bpi challenge 2012, 2012. URL: https://data.4tu.nl/articles/dataset/BPI_Challenge_2012/12689204/1. doi:10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f.

[14] A. Buliga, C. Di Francescomarino, C. Ghidini, F. M. Maggi, Counterfactuals and ways to build them: Evaluating approaches in predictive process monitoring, in: M. Indulska, I. Reinhartz-Berger, C. Cetina, O. Pastor (Eds.), Advanced Information Systems Engineering, Springer Nature Switzerland, Cham, 2023, pp. 558–574.