# Can We Find Better Process Models? Process Model Improvement using Motif-based Graph Adaptation

Alexander Seeliger ✉, Michael Stein, and Max Mühlhäuser

Technische Universität Darmstadt, Telecooperation Lab, Darmstadt, Germany
{seeliger,stein,max}@tk.tu-darmstadt.de

**Abstract.** In today's organisations efficient and reliable business processes have a high influence on success. Organisations spend high effort in analysing processes to stay in front of the competition. However, in practice it is a huge challenge to find better processes based on process mining results due to the high complexity of the underlying model. This paper presents a novel approach which provides suggestions for redesigning business processes by using discovered as-is process models from event logs and apply motif-based graph adaptation. Motifs are graph patterns of small size, building the core blocks of graphs. Our approach uses the LoMbA algorithm, which takes a desired motif frequency distribution and adjusts the model to fit that distribution under the consideration of side constraints. The paper presents the underlying concepts, discusses how the motif distribution can be selected and shows the applicability using real-life event logs. Our results show that motif-based graph adaptation adjusts process graphs towards defined improvement goals.

**Keywords:** Business Process Optimisation; Graph Adaptation; Business Process Analytics; Data Mining; Tool Support.

## 1 Introduction

The efficiency and the reliability of business processes have a high influence on organisations' success. High effort is spent in analysing processes to stay in front of the competition. With process mining, organisations get valuable insights into how their business processes are really executed, by using readily available event logs recorded by information systems. This discovered process models help to identify bottlenecks, compliance violations, and other process problems, aiming to support organisations to improve their business processes. Analysing processes using process mining helps to understand the actual use of information systems in organisations, but process mining does not necessarily provide improvement advice. It turns out that improving process models automatically is a challenging problem because the system must be able to provide automatic adjustments for existing models by providing better alternatives using extracted knowledge [1].

With respect to business process model improvement, there are basically two goals that must be fulfilled (see Fig. 1): On the one hand, any process improvement is focused on a specific goal (*improvement goal*) that should be fulfilled.

For example, organisations may want to resolve detected process problems, reduce process duration times, increase the throughput or reduce the complexity. On the other hand, provided suggestions for improvement must guarantee that the process is still executable and achieves the goal that was originally intended (*business goal*). Both the improvement goal and the business goal must be satisfied to get useful improvement suggestions for process models. However, the improvement and the business goal are often conflicting with each other. Thus, it is a challenging task to balance both goals.

This paper presents an approach that provides suggestions for process model improvement with respect to defined goals. Our approach is inspired by the systematic adaptation of communication network topologies based on motifs [9, 8, 15]. Motifs are graph patterns of small size [10]. Various studies, e.g., [10, 14], have investigated the frequency distribution of motifs, the so-called motif signature, for different kinds of graphs. These studies consistently observe that the motif signature of a graph strongly correlates with important structural metrics of the graph. Apparently, motifs are simple building blocks of complex graphs [10]. The bespoke communication network approaches [9, 8, 15] build upon this observed correlation in the following way: In a first step, a target motif signature is extracted from a network topology that performs well with respect to selected metrics. In a second step, topologies that perform badly with respect to these metrics will be adapted such that the topology approximates the discovered target motif signature.

We are the first to transfer the idea of motif-based graph adaptation to the domain of business process models. A process model can also be represented as a graph with a set of nodes and a set of edges. The motif signature of a process graph represents the core structure of the process, which also reflects various aspects, such as the complexity and standardisation in processes. The basic idea is to find an optimal motif signature that can be used to adapt non-optimised process models by removing or adding edges to the process graph. If we can approximate the process model to an optimal target signature, the resulting process model will also have desired characteristics of the optimal model. We use the LoMbA (Local Motif-based Adaptation) algorithm [15], which mod-

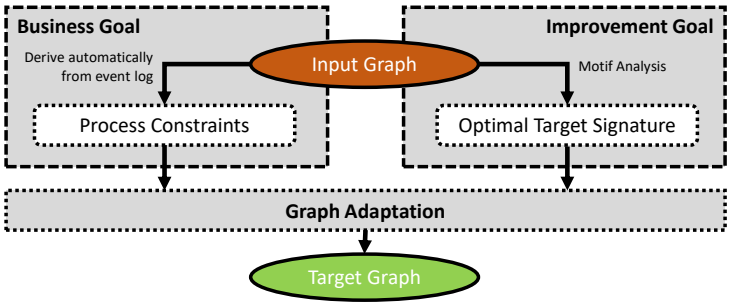


**Fig. 1.** Overview of our business process motif-based graph adaptation approach.

ifies the graph to approximate the target signature to improve process graphs. LoMbA permits the specification of constraints to retain certain aspects, e.g., graph connectivity. We use these constraints to make sure that the resulting process model is still executable and follows the original process model to achieve the desired goal. The advantage of our approach is that we can adapt process graphs with respect to different improvement goals by selecting different target signatures. These signatures can also be reused for other process graphs of different domains.

In this paper, we provide three main contributions: (1) In section 2 we show how target signatures can be derived from existing process models to optimise business processes for certain goals. (2) We present how to use motif-based graph adaptation to improve business processes in section 3.2. (3) In section 4 we conduct an experimental evaluation based on real-life event logs. The results show that motif-based graph adaptation actually improves process models. Finally, we conclude the paper by outlining open research problems.

## 2 Finding Optimal Target Motif Signature

Before any motif-based graph adaptation algorithm can be executed, an optimal target signature must be found reflecting a very good process. For better understanding how the motif signature looks like for different process models, we investigated 5 different event logs and examined the frequency of each 3 node motif (Fig. 2). For each event log we used the heuristics miner [16] in the standard settings and the "noise-free log" setting (positive observations threshold = 100, dependency threshold = 100, best to relative threshold = 0.0) to extract 10 process models (heuristics nets) in total.

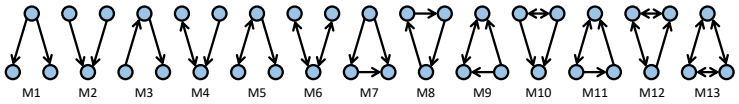


**Fig. 2.** All directed ($M1$-$M13$) motifs with 3 nodes

A motif signature is defined as a vector $M = (M_1, ..., M_l)$ which contains the relative frequency of a motif $M_i$ in the examined motif space (here: 3 node motifs). Figure 3 shows an aggregated view on the motif signature over all investigated process models using standard and "noise-free log" setting. We can see that $M1$ to $M3$ are the most dominant motifs in process models, whereas the frequencies of motifs $M4$ to $M13$ are less dominant for the standard setting. This distribution is not surprising because processes follow a specific forward path through the graph. A process has some starting and end events which are connected with some other nodes in between. $M1$ and $M2$ reflect branches in a process whereas $M3$ is a simple forward transition between events. If a process model contains events that are executed in turn, we see the occurrence of

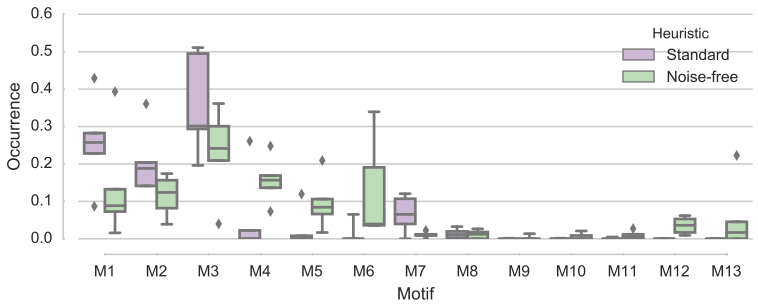


**Fig. 3.** 3 node motif frequency over the investigated process models with two different heuristic miner settings.

$M4-M6$. $M7$ and $M8$ occur if the process model contains loops and $M9-M13$ are combinations. A slightly different histogram can be seen for the "noise-free log" setting where $M1-M6$ are the most dominating motifs. With a higher threshold we also mine lower frequent behaviours which results in more edges in the model (see Fig. 3). In conclusion we can say that motifs can characterise the structure of the process as they are the core building blocks for processes.

Besides the relative frequency of motifs we searched for pearson correlations between the motifs and process model properties. The different motifs have different correlations to the number of edges, events, the graph density and the clustering coefficient ($\frac{3 \cdot \text{number of triangles}}{\text{number of connected triplets of vertices}}$). Although the sample of 10 models may be small, the analysis gives a quick overview how the different process properties are linked to motifs, enabling to determine a possible target signature to optimise process models. Within our analysis, we found the following correlations: $M3$ has a significant negative correlation on the number of edges, number of nodes, graph density and clustering coefficient. $M6$, $M9$, $M10$, $M12$, $M13$ have a significant positive correlation on the number of edges (cf. Fig. 4). This is not surprising because these motifs have a high connectivity between the nodes themselves which leads to a higher graph density in larger graphs.

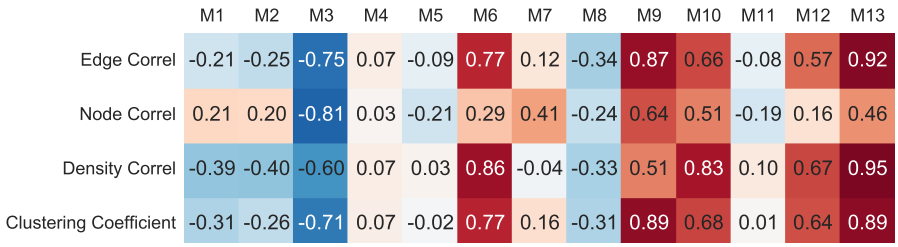


**Fig. 4.** Pearson correlations between motif and graph property. Correlations $> 0.55$ are significant with $p < 0.05$; correlations $> 0.6$ with $p < 0.01$.

The observations depicted in Figure 4 show the correlation details for each motif. If we would like to reduce the complexity of a process graph, we should decrease the occurrence of motifs which have a positive correlation with the graph density. Analogue to this observation other analysis can be performed with different optimisation goals. Another approach would be to use an optimised model and calculate the signature that can then be used as the target signature for improvement. This paper will focus on the reduction of complexity of process models. However, our approach is not limited in this respect.

## 3 Motif-based Process Graph Adaptation

In the previous section we have seen how an optimal target motif signature can be found. Our approach assumes the existence of such a target signature to adapt the given process model. In practice an analyst can use reference process models to gather an optimal target motif signature or try different target signatures and analyse their behaviour to the process graph. Next we will describe how our graph adaptation algorithm tries to improve process graphs and how optimisation constraints can be defined with DECLARE in order to retain the original business goal of the process.

### 3.1 Basic Definitions

The original local motif-based adaptation approach (LoMbA) was first introduced for the adaptation of network topologies by Stein et. al. in [15]. In this paper we will transfer this algorithm to the domain of business process model improvement, thus we will first introduce some basic definitions and formalise the graph adaptation problem in the business process domain:

**Definition 1.** ***(Process Graph)*** *Let $G = (V, E)$ be a simple graph, where nodes $v \in V$ are the event classes that can occur in a process and edges $e \in E$ between nodes represent the possible transitions between events.*

**Definition 2.** ***(Motifs)*** *A motif $M$ is a small subgraph pattern of small size, typically 3 or 4 nodes (see Fig. 2). We denote $M$ as a finite set of motifs of the same node size.*

As motifs are the building blocks of graphs we can count their relative frequency in a process graph:

**Definition 3.** ***(Motif Signature)*** *The occurrence of $M$ in $G$ is represented as a motif signature $s(G)$ which is a real-valued vector that stores the relative frequency of motif $M_i$.*

For comparing multiple graphs regarding their motif signature, we can define a function which returns the distance between two motif signatures:

**Definition 4.** ***(Motif Distance)*** *Furthermore, $distance(x, y)$ of motif signatures $x$ and $y$ is the Euclidean distance between $x$ and $y$.*

In this paper we aim to improve process graphs by modifying the process graphs' edges. One way of gathering real process graphs is to extract event logs from information systems. Event logs contain the actual use of the process involved information systems and provide the ideal input for process improvement approaches. A process mining discovery algorithm such as the heuristics miner [16] can be used to extract a process model. Due to the fact that we cannot modify the process graph arbitrarily, we need to specify a function that checks if the proposed change is allowed. In comparison to the function defined in [15] we define a function that checks various business constraints (see Section 3.3).

**Definition 5.** ***Process Constraint Functions.*** *Let $f(G) \rightarrow \{true, false\}$ be a function that receives the original graph $G$ and returns a boolean value. $f$ returns true if the graph $G$ fulfils given process constraints and false if not.*

### 3.2 Algorithm

The goal of the graph adaptation algorithm is to modify the input graph $G$ such that a target signature $t$ is approximated under the process constraint function $f$. The algorithm periodically iterates through all nodes $v \in V$ of the graph $G$ (see algorithm 1). The algorithm is divided into two steps: (1) the algorithm searches for a proper set of modification operations and (2) it selects an operation that fulfils the process constraint function and checks if the modification has approximated $G$ to the target signature $t$.

The following modifications for each node $v \in V$ are sufficient to modify a graph in our domain: *Remove-edge*, *Add-edge*, *Move-edge* operation. Due to the large amount of possible operations, the algorithm reduces the search space by calculating the *edge operation indicator* (lines 2-3). It ranks the generated modification operations by using a simple heuristic that decides which graph operations are more appropriate to approximate $G$ to the target signature:

$$EOI = \frac{\sum(t_i - s_i(G)) \cdot |E(M_i)|}{l}$$

$t_i$ is the relative frequency of motif $M_i$ in the target signature, $s_i(G)$ returns the relative frequency of motif $M_i$ in the input graph $G$, and $|E(M_i)|$ returns the number of edges in motif $M_i$. $l$ is the number of inspected motifs. $EOI$ calculates the average weighted ratio between the amount of edges in the target signature and the current graph signature. A value larger than 0 indicates that more edges need to be added to the graph, thus the algorithm will prefer the *Add-edge* operator. If $EOI$ is smaller than 0, the algorithm will prefer the *Remove-edge* operator. A value near 0 indicates that neither edges should be removed nor added, thus the algorithm will prefer the *Move-edge* operator.

In the second step of the algorithm, a graph operation is selected for $v \in V$ and a candidate graph $G'$ with the applied operation is generated. Now the algorithm checks the process constraints. If $f$ is violated then the operation is discarded. Otherwise, the motif signature distance between $G'$ and $t$ is calculated. If the motif signature of $G'$ is closer to the target signature than $G$, then $v$

**Algorithm 1:** Motif-based Process Graph Adaptation Algorithm [15]

```
1  its ← {Remove (G, v), Add (G, v), Move (G, v)};
2  EOI ← GetEdgeIndicator (G, t);
3  itOrder ← GetIteratorOrder (its, EOI, eoiThreshold);
4  for it: itOrder do
5      foundValid ← false, maxSteps ← ∞, doneSteps ← 0;
6      currentDistance ← Distance (s(G), t);
7      while op ← it.next() and doneSteps < maxSteps do
8          doneSteps ← doneSteps + 1;
9          G' ← CreateCandidate (G, op);
10         if f(G') then
11             foundValid ← true;
12             if Distance (s(G'), t) < currentDistance then
13                 G ← G';
14                 if maxSteps = ∞ then
15                     maxSteps ← doneSteps;
16                     doneSteps ← 0, currentDistance ← Distance (s(G), t);
17                 end
18             end
19         end
20     end
21     if foundValid then break;
22 end
```

modifies $G$. In the original algorithm a sampled graph is used instead of the complete graph to reduce the computational complexity of counting motifs [17]. As our graphs tend to be smaller than network topologies, we can operate on the complete graph. If the algorithm has found a valid graph operation for $v$ that fulfils $f$, it is likely that other operations are valid within the given operation iterator. Thus the algorithm will check another *doneSteps* operations from the iterator. If none of the operations could be applied to $G$ due to the violations of $f$, the algorithm will expand the search space with additional iterators.

The complexity of the tackled graph problem is NP-hard [15]. Due to the much smaller process graphs compared with network topology graphs, we observed a feasible average runtime of 130 seconds over 5 rounds in our experiments.

### 3.3 Specification of Constraints using Declare Models

In order to retain the feasibility of the optimised process model, our algorithm allows the specification of a process constraint function that must be fulfilled after each graph modification operation. We use DECLARE [12] models to specify the process constraint function that the graph adaptation algorithm will respect. DECLARE is an LTL-based declarative process modelling language which allows the specification of process models by the instantiation of templates [2]. For instance, the *response* template specifies that a certain event must be executed if

another specific event has been executed before. Each time our algorithm generates possible modifications to the process model, it checks if all given constraints are still fulfilled after the application of proposed modifications using automata described in [5]. If so, the algorithm will apply the modification otherwise it will reject the change (see section 3.2). This allows us to restrict the algorithm to certain modifications such that the original process model is still followed in the improved one.

Constraints can be either specified by hand using the graphical representation of DECLARE models [2] or by declarative process discovery using historic process executions [6]. We decided to use declarative process discovery from historic executions gathered in event logs to automatically generate constraints for process models. By specifying a confidence level we can determine which quality the constraints should have in order to force them to be fulfilled in the resulting improved model. Still, it is possible to edit the constraints, disable specific constraint types or add custom ones. The advantage of the automatic discovery is that hidden constraints will be uncovered from the historic executions, reducing the manual constraint definition time and leading to more relevant constraints.

## 4 Experimental Evaluation

In this experimental evaluation we show the general applicability of motif-based graph adaptation in the domain of business process model improvement.

### 4.1 Setup

We tested our approach with 5 real-life event logs (see Table 1) and compared if the improvement goal was reached. For each event log we generated two process models, one with the standard setting and one with the "noise-free" setting (marked with a star *), using the heuristics miner.

**Table 1.** Characteristics of the used real-life event logs in our evaluation.

| # | Event log | Instances | Variants | Events | Events/Case | Constraints |
|---|---|---|---|---|---|---|
| I | BPI Challenge '12 | 13087 | 7179 | 36 | 20.04 | 214 |
| II | Large | 651709 | 30460 | 35 | 5.95 | 122 |
| III | Small | 873 | 101 | 45 | 7.77 | 26 |
| IV | Midsize | 90536 | 1630 | 30 | 9.06 | 56 |
| V | Environmental | 1434 | 381 | 27 | 5.98 | 49 |

The improvement goal for all process models is to reduce the complexity of the model. Based on the observations in section 2, we select three different process independent target signatures: (1) $M1, M2, M3 = 33.3\%$, (2) $M1 = 20\%, M2 = 20\%, M3 = 60\%$ and (3) $M1 = 22.8\%, M2 = 22.8\%, M3 =$

34.7%, $M8 = 19.0\%$. For retaining the original business goal, we automatically gather the constraints by generating DELARE rules using MINERful (see Section 3.3) and pick the all rules with a confidence of at least 0.95.

### 4.2 Results and Discussion

Table 2 shows the results of the process graph improvement using the three different target signatures after 5 rounds. Target signatures (1) and (2) actually decreased the number of edges, the density and the clustering coefficient, achieving our improvement goal. Only target signature (3) did not work as well as the other two, resulting in an increase of the clustering coefficient (also see Fig. 5). Most graph operations were made to the "noise-free setting" process models. Here more edges could be removed which is not surprising because these models in general have more edges. The graph constraints are the same for both models thus these additional edges were now removed by our improvement approach.

**Table 2.** Evaluation result before and after motif-based graph adaptation.

| | I | I* | II | II* | III | III* | IV | IV* | V | V* |
|---|---|---|---|---|---|---|---|---|---|---|
| Initial Edges | 39 | 54 | 106 | 244 | 29 | 37 | 73 | 61 | 40 | 47 |
| Initial Density | 0,071 | 0,098 | 0,089 | 0,205 | 0,069 | 0,088 | 0,084 | 0,070 | 0,057 | 0,067 |
| Initial Cl.Coeff. | 0,059 | 0,100 | 0,120 | 0,331 | 0,065 | 0,072 | 0,110 | 0,062 | 0,033 | 0,097 |
| **(1)** $\Delta$ Edges | -1 | -19 | -20 | -152 | 0 | -7 | 0 | -12 | -3 | -16 |
| $\Delta$ Cl.Coeff. | -0,059 | -0,100 | -0,140 | -0,325 | -0,065 | -0,056 | -0,110 | -0,056 | -0,021 | -0,097 |
| **(2)** $\Delta$ Edges | 0 | -19 | -38 | -168 | 0 | -8 | -23 | -17 | -3 | -18 |
| $\Delta$ Cl.Coeff. | -0,059 | -0,100 | -0,140 | -0,331 | -0,065 | -0,072 | -0,110 | -0,062 | -0,033 | -0,097 |
| **(3)** $\Delta$ Edges | 1 | -11 | 0 | -138 | 0 | 0 | 0 | 0 | -3 | -10 |
| $\Delta$ Cl.Coeff. | 0,133 | 0,092 | 0,023 | -0,178 | 0,122 | 0,113 | 0,076 | 0,130 | 0,151 | 0,086 |

We run the algorithm with a maximum of 5 rounds. Fig. 6 shows the distance to the target signatures over the rounds. We can see that the algorithm approximates the graph towards the target signature. For target signature (1) we were able to reach the signature for all datasets within 4 rounds, whereas we did not reach it for signature (3). The largest number of applied operations were already made in the first round, resulting in a quick converge towards the target signature. In addition, we can see that the size of the input graph influenced how fast the algorithm converges to the target.

We also investigated how many graph operations (*Add-edge*, *Remove-edge*, *Move-edge*) were performed over the number of rounds (cf. Fig. 7). After round 2 the number of added and removed edges is almost equal for all datasets indicating that only *Move-edge* operations are performed because the target signature is almost reached. Here it would make sense to stop the algorithm early until the distance to the target signature is close enough because this would retain more structures of the original process.

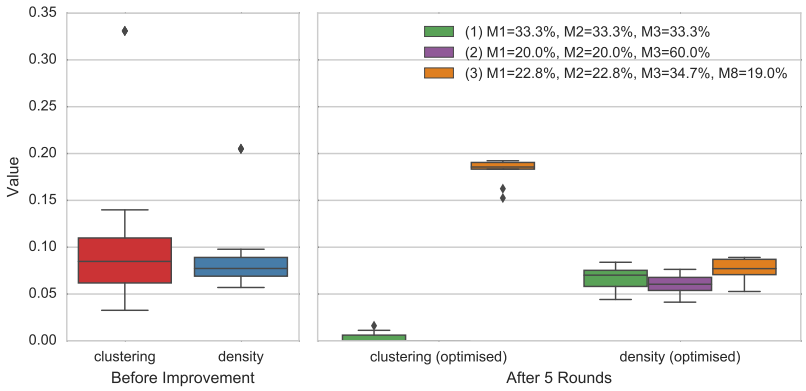


**Fig. 5.** Normed number of edge operations performed for target signature (1) and (3).

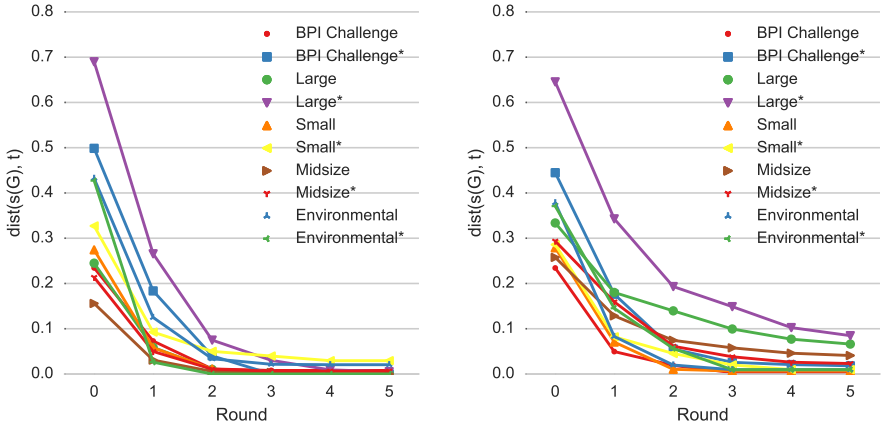


**Fig. 6.** Distance to target signatures (1) and (3) over the number of rounds.

## 5 Related Work

According to the BPM survey [1] from 2013 the improvement of existing process models using process mining is not much researched. Currently there exists no tool, neither from research nor from industry, that provides automatic guidance for redesigning processes, although there is the need for proper tools [3].

In [13] best practices for redesigning business processes are presented. The authors evaluate different approaches and provide an overview of the most common redesign operations. Niedermann et al. [11] propose a semi-automatic process optimisation platform which matches new processes to existing processes using similarity metrics (e.g. syntactic, linguistic and context). Best practice op-

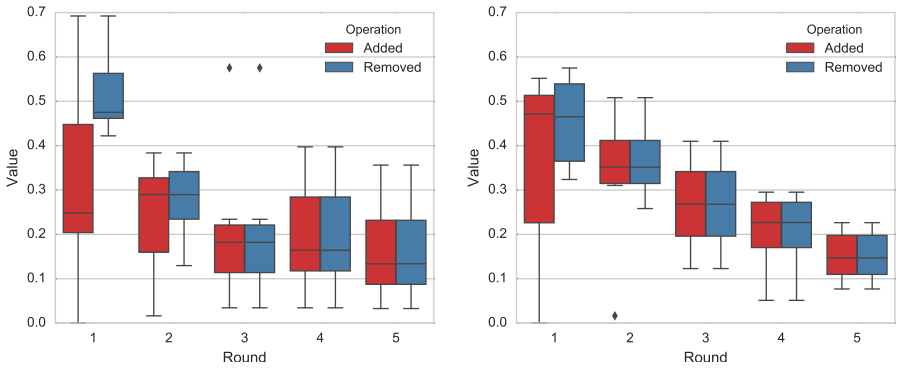


**Fig. 7.** Normed number of edge operations performed for target signature (1) and (3).

timisation patterns that consist of a detection and an application component (based on [13]) are matched and applied to the original process.

Process model improvement can also be indirectly achieved using conformance checking methods such as presented in [4]. [7] propose the use of reference models (ITIL) that usually best practices to improve existing process models. The authors present an approach which overcomes the problems of different levels of detail, partial views and overemphasis of the order. Another approach was presented in [18] which calculate various process performance indicators. By matching and clustering PPIs together with other process models from different organisations, the presented system can make suggestions and recommendations for performance improvement.

## 6 Conclusion

This paper first showed that motif analysis can also be applied to business processes to find relations between the core structure and other process properties. We have found correlations between specific motifs and process improvement goals. We further adapted the motif-based graph adaptation algorithm to modify process graphs in order to improve them using an optimised motif target signature based on the observed motif histogram. We also showed how to automatically define required process constraints to retain the feasibility. Finally, in our experimental evaluation we presented the applicability of our approach for real event logs. The results show that process models were improved based on a given target signature to achieve the improvement goal.

In future work, we will investigate how target signatures can be defined for specific business goals and how they can be derived from existing process models or reference models. Another aspect that we will address is the heuristic that ranks the possible graph operations. Currently, the heuristic is just target signature focused but it should also be targeting the improvement goal. Lastly, we will expand our approach to use larger motifs with more complex graph patterns.

**Acknowledgement.** This project (HA project no. 522/17-04) is funded in the framework of Hessen ModellProjekte, financed with funds of LOEWE – Landes-Offensive zur Entwicklung Wissenschaftlich-ökonomischer Exzellenz, Förderlinie 3: KMU-Verbundvorhaben, by the LOEWE initiative (Hessen, Germany) within the NICER project [III L 5-518/81.004] and by the German Research Foundation (DFG) as part of project A1 within the Collaborative Research Center (CRC) 1053 – MAKI.

## References

1. van der Aalst, W.M.P.: Business Process Management : A Comprehensive Survey. ISRN Software Engineering 2013, 1–37 (2013)
2. Burattin, A., Maggi, F.M., van der Aalst, W.M.P., Sperduti, A.: Techniques for a posteriori analysis of declarative processes. EDOC pp. 41–50 (2012)
3. Cater-Steel, A., Tan, W.G., Toleman, M.: Challenge of Adopting Multiple Process Improvement Frameworks. ECIS pp. 1–12 (2006)
4. De Leoni, M., van der Aalst, W.M.P.: Aligning event logs and process models for multi-perspective conformance checking: An approach based on integer linear programming. LNCS 8094, 113–129 (2013)
5. Di Ciccio, C., Maggi, F.M., Montali, M., Mendling, J.: Resolving inconsistencies and redundancies in declarative process models. Information Systems 64, 425–446 (2017)
6. Di Ciccio, C., Schouten, M.H.M., de Leoni, M., Mendling, J.: Declarative process discovery with MINERful in ProM. CEUR Workshop Proc. 1418, 60–64 (2015)
7. Gerke, K., Tamm, G.: Continuous quality improvement of IT processes based on reference models and process mining. AMCIS (2009)
8. Krumov, L., Fretter, C., Müller-Hannemann, M., Weihe, K., Hütt, M.T.: Motifs in co-authorship networks and their relation to the impact of scientific publications. European Physical Journal B 84(4), 535–540 (2011)
9. Krumov, L., Schweizer, I., Bradler, D., Strufe, T.: Leveraging network motifs for the adaptation of structured peer-to-peer-networks. GLOBECOM pp. 0–4 (2010)
10. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N.: Network Motif: Simple Building Blocks of Complex Networks. Science 298(5594), 298. (2002)
11. Niedermann, F., Radeschütz, S., Mitschang, B.: Design-time process optimization through optimization patterns and process model matching. CEC pp. 48–55 (2010)
12. Pesic, M., Schonenberg, H., Van Der Aalst, W.M.P.: DECLARE: Full support for loosely-structured processes. EDOC pp. 287–298 (2007)
13. Reijers, H.A., Liman Mansar, S.: Best practices in business process redesign: An overview and qualitative evaluation of successful redesign heuristics. Omega 33(4), 283–306 (2005)
14. Schreiber, F., Schwöbbermeyer, H.: Motifs in biological networks. Statistical and Evolutionary Analysis of Biological Networks pp. 45–64 (2010)
15. Stein, M., Weihe, K., Wilberg, A., Kluge, R., Klomp, J.M., Schnee, M., Wang, L., Mühlhäuser, M.: Distributed Graph-based Topology Adaptation using Motif Signatures. ACM-SIAM Meeting on Algorithm Engineering & Experiments (2017)
16. Weijters, a.J.M.M., van der Aalst, W.M.P., Medeiros, a.K.A.D.: Process Mining with the HeuristicsMiner Algorithm. BETA Working Paper Series 166, 1–34 (2006)
17. Wernicke, S.: Efficient detection of network motifs. In: IEEE/ACM Transactions on Computational Biology and Bioinformatics. vol. 3, pp. 347–359 (2006)
18. Yilmaz, O., Karagoz, P.: Generating Performance Improvement Suggestions by using Cross-Organizational Process Mining. SIMPA 6, 3–17 (2015)