Goal-oriented Process Enhancement and Discovery

Mahdi Ghasemi(✉) and Daniel Amyot

EECS, University of Ottawa, Ottawa, Canada
{mghasemi,damyot}@uottawa.ca

Abstract. Process mining practices are mainly activity-oriented and they seldom consider the (often conflicting) goals of stakeholders. Involving goal-related factors, as often done in requirements engineering, can improve the rationality and interpretability of mined models and lead to better opportunities to satisfy stakeholders. This paper proposes a new Goal-oriented Process Enhancement and Discovery (GoPED) method to align discovered models with stakeholders' goals. GoPED first adds goal-related attributes to traditional event characteristics (case identifier, activities, and timestamps), selects a subset of cases with respect to a goal-related criterion, and finally discovers a process model from that subset. We define three types of criteria that suggest desired satisfaction levels from a (i) case perspective, (ii) goal perspective, and (iii) organization perspective. For each criterion, an algorithm is proposed to enable selecting the best subset of cases were the criterion holds. The resulting process models are expected to reproduce the desired level of satisfaction. A synthetic event log is used to illustrate the proposed algorithms and to discuss their results.

Keywords: Business process management · Process mining · Goal modeling · Requirements engineering · Event logs · Performance indicators

1 Introduction

The process mining community has developed various algorithms and tools to enable the analysis of event logs to discover process models and improve their underlying processes. Process mining activities involve: (1) *Discovery*, where a model is being created from event logs; (2) *Conformance checking*, where differences between the model and reality are detected; and (3) *Enhancement*, where an existing process model is improved or extended using some additional desired data from different aspects [16].

Event logs, resulting from the execution of processes, are the main input of process discovery activities. However, process mining approaches usually do not consider specific goals that individual cases pursue and satisfaction levels that traces yielded for different stakeholders' goals [7]. This situation not only threatens the *rationality* behind the discovered models, but also often results in unstructured "*spaghetti-like*" process models. Although such models reflect reality, they cover many exceptions and many traces misaligned with goals [12]. Process mining practitioners have to deal with such problems especially in flexible environments that allow multiple alternatives within process execution.

T. Hildebrandt et al. (Eds.): BPM 2019, LNCS 11675, pp. 102–118, 2019.
https://doi.org/10.1007/978-3-030-26619-6_9

There are currently some strategies that deal with unstructured discovered processes often taking into account the *frequency* of activities and transitions. For example, keeping the activities that occur at least for 20% of cases is a way to simplify the model. In contrast to strategies that change logs, abstraction techniques such as fuzzy mining [16] are applied to the resulting process graphs. Also, current *declarative* approaches, e.g., based on linear temporal logic, exist to enforce some constraints (e.g., on sequencing) and discover complying models at the *activity level* [12].

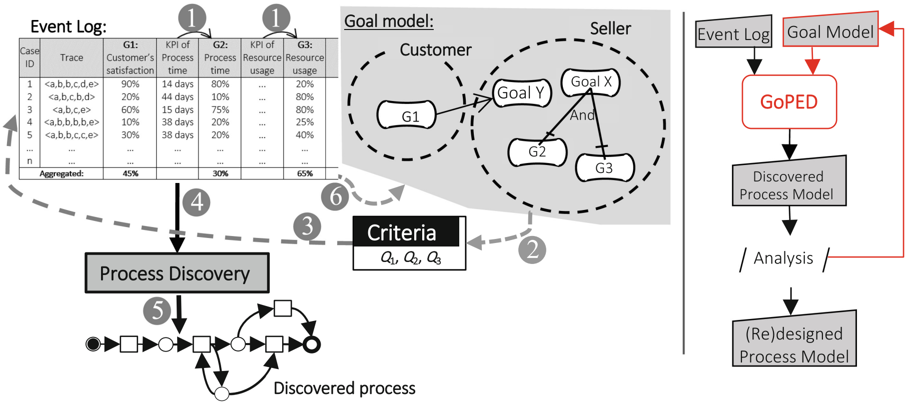
Goal modeling, a requirements engineering approach that enables the description of the interrelated (and often conflicting) goals of systems and stakeholders, can be leveraged for addressing the aforementioned problems. Goal modeling is used to support heuristic, qualitative, or formal reasoning about goals, and ultimately trade-off analysis, what-if analysis, and decision making. In contrast to process mining where "*how*", "*what*", "*where*", "*who*", and especially "*when*" questions are answered, goal modeling focuses mainly on complementary "*why*" questions [2].

We hypothesize that a goal-oriented approach combined to process mining enables leveraging goals to improve process models and their realization. Process models that are discovered with respect to different goals are aligned with such goals and hence more likely to produce high levels of satisfaction.

The objective in this paper is to offer a process mining method concerned not only with the sequencing of activities, but also with processes' goals and satisfaction indicators. To this end, we propose a *goal-oriented process enhancement and discovery* (GoPED) method that adds satisfaction levels of different goals to event logs and considers traces of activities beside their contribution to predefined goals. Goal satisfaction levels are derived from a model capturing goals, stakeholders, and their relationships. Note that the "enhancement" part of GoPED is about enhancing logs with goal information to produce higher-quality and simpler process models, and not about improving processes after their discovery.

As an example, a trace of activities in a healthcare process may take a very short time (i.e., it satisfies the goal "to decrease process time") but may end up with a wrong diagnosis (i.e., it violates the goal "to diagnose correctly"). Inversely, a trace may take a long time and impose an unaffordable cost but may end up with a correct diagnosis. GoPED takes advantage of goal models to manage such conflicting goals and to support trade-off analysis. With GoPED, good historical experiences will be found within the whole event log to be used as a basis for inferring good models and bad experiences will be found to be avoided. The *goodness* of traces and models is defined with regards to three categories of goal-related criteria: satisfaction of individual cases in terms of some goals (*case perspective*), overall satisfaction of some goals rather than individual cases (*goal perspective*), and a comprehensive satisfaction level for all goals over all cases (*organization perspective*). GoPED is expected to guide process discovery approaches towards specific goal-related properties of interest.

The paper is structured as follows. Section 2 explains the fundamentals of GoPED and highlights its contribution to current process mining approaches. Section 3 explicitly describes three algorithms for selecting traces according to the three categories of criteria discussed in the previous paragraph. Then, in Sect. 4, the GoPED method is applied to an illustrative example of a healthcare process, with a discussion of the results. Related work at the intersection of process mining and goal modeling is briefly discussed in Sect. 5. Finally, a summary and conclusions are provided in Sect. 6.



Case ID	Trace	**G1:** Customer's satisfaction	KPI of Process time	**G2:** Process time	KPI of Resource usage	**G3:** Resource usage
1	<a,b,b,c,d,e>	90%	14 days	80%	...	20%
2	<a,b,c,b,d>	20%	44 days	10%	...	80%
3	<a,b,c,e>	60%	15 days	75%	...	80%
4	<a,b,b,b,b,e>	10%	38 days	20%	...	25%
5	<a,b,b,c,c,e>	30%	38 days	20%	...	40%
...	...	...	...	...	...	...
n	...	...	...	...	...	...
Aggregated:		**45%**		**30%**		**65%**

Fig. 1. Overview of goal-oriented process enhancement and discovery (GoPED)

2 GoPED Method

Figure 1 gives an overview of the proposed method through an example that exploits the Goal-oriented Requirement Language (GRL) standard [2]. Let us assume that there are three leaf goals (G1, G2 and G3), with G1 contributing to Goal Y and Goal X being AND-decomposed by G2 and G3. Each goal may be fed by its Key Performance Indicators (KPIs), allowing to quantify its satisfaction level. Let us also assume that the event logs store the value of each goal-related KPI, e.g., "process time" associated to the goal G2 "to take a short process time". Such KPIs and how they contribute to goal satisfaction (e.g., by providing a function that converts a current, observable value to an abstract satisfaction value between 0/violated and 100/satisfied) are also defined in the goal model (see [1] for details). Based on this scheme, the satisfaction level of the leaf goals in Fig. 1 are computed from corresponding KPIs (arrows ❶). We define the satisfaction level of Goal *i* as *Sat* (G_i). The satisfaction of actors (dashed circles in Fig. 1) and of the whole models can be computed in a similar fashion [1].

After finding the current satisfaction of considered goals, GoPED defines some criteria related to the goals (arrow ❷). The main objective is to design a process model that fulfills one or many such criteria. The goal-related criteria are defined from three perspectives as follows:

- The resulting process model achieves (based on current evidence) a minimum satisfaction level for every single case in terms of one or multiple goals (*row* or *case perspective*). For example, in Fig. 1, *Sat*(G_2) should be more than 60 for all cases.
- The resulting process model achieves a threshold for the *aggregated* satisfaction level of one or multiple goals rather than the level for individual cases (overall *column* or *goal perspective*). For example, in Fig. 1, the aggregated satisfaction level of G2 (where the aggregation function is defined as the *average* here) should be higher than 70 (but is currently 45).
- The resulting process model achieves a threshold for the comprehensive satisfaction level of many goals over all cases (*table* or *organization perspective,* arrow ❻).

This may be computed through the goal model (e.g., in GRL) or through a function derived from that model. For example, according to the structure of the model in Fig. 1, the satisfaction of the stakeholder "Seller" is the average of Goal Y (computed from G1) and Goal X (the minimum of G1 and G2).

The basis of process mining is generally to use historical event logs and infer valuable insights. Following this general approach, GoPED selects a *subset* of the input traces that have already fulfilled the given criteria and uses them to find process models of interest (arrow ❹). For example, if the objective is to secure at least 50 as a satisfaction level for G2 for all the customers, the cases #1 and #3 will be selected because they have a satisfaction level over 50 for goal G2. After such a selection, a process model is mined through the selected traces using a process discovery algorithm (arrows ❹ and ❺). The discovered model does not represent all existing behaviours, but rather represents the desired behaviours towards the goals. Different model mined through different goal-related criteria can shed some light on different aspects and alternatives involved in the real or the discovered model. Such criteria are purposely defined by a domain expert in collaboration with a modeller. Moreover, an analyst can compare the model discovered from the whole log with the model discovered by GoPED. Such a comparison can also reveal some valuable insights from potential discrepancies. Goal-oriented conformance checking approaches [5, 6] can also suggest some way of reconsidering the goal model with respect to misalignments between the process and goal perspectives, as shown in the right graph of Fig. 1.

The process model resulting from GoPED is inferred from cases selected based on their goals. Therefore, irrelevant cases (that likely pursue goals different from the expected goals) are filtered out. The discovered model will be more likely well-structured as GoPED intentionally decreases the number of variations of traces and, in turn, decreases the chance of producing a spaghetti-like process model.

Another benefit of GoPED relates to the quality dimensions considered in usual process mining activities. In addition to the *fitness*, *precision*, *generalization*, and *simplicity* dimensions [16], GoPED brings into consideration a new *intention* dimension, formalized by the goal model.

3 Algorithms to Select Cases

In process mining, the three attributes (columns) that must minimally exist in an event table are *case identifier*, *activity* and *timestamp*. There might be some other event attributes stored in such a table that can be used for the analysis of discovered models (e.g., *resource*). Similarly, there might be some attributes about the case (e.g., age) or about a case's trace (e.g., total process time). In GoPED, we add some new case attributes related to goals, which are usually absent in process mining practice. Table 1 shows the architecture of event logs enhanced with goal-related attributes, used as input of GoPED.

Table 1. Event log enhanced with n goal-related attributes (*EnhancedLog*)

Case	Trace	Goal 1	Goal 2	…	Goal n	Overall
c_1	t_1	$s_{1,1}$	$s_{1,2}$	…	$s_{1,n}$	$s_{1.Ove}$
c_2	…	$s_{2,1}$	$s_{2,2}$	…	$s_{2,n}$	$s_{2.Ove}$
…	…					…
c_m	t_m	$s_{m,1}$	$s_{m,2}$	…	$s_{m,n}$	$s_{m.Ove}$
Aggregated satisfaction:		$s_{Agg.1}$	$s_{Agg.2}$	…	$s_{Agg.n}$	s_{Comp}

3.1 Preliminaries

The notations that are used through this paper are defined as follows:

Definition 1. Basic concepts (*activity*, *trace*, *case, event log*).

- A is the set of all experienced *activities* labelled a_i.
- A *trace* is a finite sequence of activities $t = \langle a_1, \cdots, a_k \rangle$, where $k \in \mathbb{N}^+$ is the trace length. T is the set of all observed traces.
- A *case* $c = \langle id, t \rangle$ has a case identifier $id \in \mathbb{N}^+$ and contains a trace $t \in T$.
- $trace(c) = t$ is a shorthand to indicate that the trace of the case c is $t \in T$.
- $L = \langle c_1, \cdots, c_m \rangle$ is an *event log* consisting of a finite sequence of cases of size m.
- C is the set of all possible cases (with traces) represented in the log L.

Definition 2. *EnhancedLog* structure.

To select the best subset of cases in a log through GoPED's algorithms, an event log enhanced with additional goal-related attributes is needed. The structure of such log, shown in Table 1, and the elements of that structure are defined as follows:

- *EnhancedLog* is the event log L enhanced with goal-related attributes. This log is a table of all cases $c \in C$ beside their traces $t \in T$. The satisfaction levels of all considered goals (including KPIs and actors) are stored in the next columns. $\mathbb{G}$ is the set of all considered goals, i.e., $\mathbb{G} = \{G_1, G_2, \cdots, G_n\}$. We assume that *EnhancedLog* consists of m cases and n considered goals.
- $s_{i,j}$ in *EnhancedLog* shows the level of satisfaction of $case_i$ in terms of $Goal_j$.
- $s_{i.Ove}$, found in the last column of the table *EnhancedLog*, is the overall satisfaction level for $case_i$. This represents the satisfaction level of the whole goal model. The satisfaction level of the goal model is evaluated through bottom-up analysis as elaborated in the goal-oriented modeling literature [1]. This evaluation is based on AND/OR refinements, contribution links, the importance level of a goal to its actor, and the actor importance in the whole model.
- Function g is derived from the goal model to compute the overall satisfaction level based on satisfaction levels of all sub-goals in $\mathbb{G}$ [4]. Therefore, as $s_{i,j}$ is the satisfaction level of $Goal_j$ for $case_i$, we have $s_{i.Ove} = g(s_{i.1}, s_{i.2}, \cdots, s_{i.n})$.
- $s_{Agg.i}$, in the last row of *EnhancedLog*, show the aggregated satisfaction level of each goal based on the satisfaction level of all cases in terms of that goal. $s_{Agg.i}$ is a function (e.g., average, median, etc.) of satisfaction levels of all m cases for $Goal_j$.

- Function f_j is the aggregation function of $Goal_j$. For each goal $G_j \in \mathbb{G}$ we have $s_{Agg.j} = f_j(s_{1.j}, s_{2.j}, \cdots, s_{m.j})$. F is a tuple of functions $(f_1, f_2, \ldots, f_n)$ that keeps aggregation functions of all goals.
- s_{Comp} is the comprehensive satisfaction level that the process has yielded. This factor can be defined either by composing the aggregated satisfactions (last row) or by aggregating the overall satisfaction levels (last column) using some function.

Definition 3. GoPED offers three types of goal-related criteria, discussed in Sect. 3.1:

- $\boldsymbol{Q}_{\mathbf{case}}$ is a set of criteria q_j. Each q_j is a tuple composed of one goal $G_j \in \mathbb{G}$ and a threshold, sl_j, for the satisfaction level of that goal, $Q_{\text{case}} = \{q_j = (G_j, \overline{sl_j}) | G_j \in \mathbb{G} \wedge 0 \leq \overline{sl_j} \leq 100\}$. A confidence level, $0 \leq conf \leq 1$, together with Q_{case}, constitute the whole criteria. Such criteria represent that (with a confidence *conf)* the satisfaction level of every single case in terms of the considered goals G_j will be at least $\overline{sl_j}$. It is noteworthy that all goals in $\mathbb{G}$ are not necessarily considered by Q_{case}. For example, when $\mathbb{G} = \{\mathrm{G}_1, \mathrm{G}_2, \mathrm{G}_3\}$ and $Q_{case} = \{(\mathrm{G}_2, 90), (\mathrm{G}_3, 75)\}$ and *conf* = 0.8, GoPED is looking for a process model that will yield minimum satisfaction levels of 90 for G2 and 75 for G3, for at least 80% of the cases (i.e., confidence of 0.8).
- $\boldsymbol{Q}_{\mathbf{goal}}$ refers to the second type of goal-related criteria. Q_{goal} consists of a set of criteria q_j composed of one goal $G_j \in \mathbb{G}$ and a satisfaction level for that goal. $Q_{goal} = \{q_j = (G_j, \overline{sl_j}) | G_j \in \mathbb{G} \wedge 0 \leq \overline{sl_j} \leq 100\}$. Q_{goal} is looking for a process model that can deliver an *aggregated* satisfaction level for the considered $G_j \in \mathbb{G}$ of at least $\overline{sl_j}$. Again, all goals in $\mathbb{G}$ are not necessarily considered by Q_{goal}. For example, when $\mathbb{G} = \{\mathrm{G}_1, \mathrm{G}_2, \mathrm{G}_3\}$ and $Q_{\text{goal}} = \{(\mathrm{G}_2, 90), (\mathrm{G}_3, 75)\}$, Q_{goal} is looking for a process model that will yield minimum *aggregated* satisfaction levels of 90 for G2 and 75 for G3.
- $\boldsymbol{Q}_{\mathbf{Comp}}$ consists of one value between 0 and 100 called $\overline{sl}_{Comp}$. This criterion looks for a process model that can yield a *comprehensive* satisfaction of at least $\overline{sl}_{Comp}$.

Definition 4. *SelectedCases* $\subseteq C$ is the main output of GoPED algorithms and the set of selected cases that satisfy one of the aforementioned criteria.

3.2 GoPED Algorithms

As the goal-related criteria are based on three different viewpoints of *EnhancedLog*, three different algorithms for trace selection are required. The main idea in all three algorithms is to select the largest subset of cases that satisfy the selected criterion.

Searching for the *largest* subset of cases is needed because if one simply selects very few cases that meet the desired criteria, the discovered model will be based on an event log suffering from potential *incompleteness* problems. When the event log consists of too few events, the discovered model is less realistic and risks becoming overfitted.

Another feature of our search approach is that we look over the *cases* $\in C$ rather than the *traces* $\in T$. This is because many cases might have a same trace but different levels of satisfaction for the goals. Moreover, the frequency of each trace contains very important knowledge about real-world behaviors. Therefore, we need to end up with a subset that consists of variations of traces together with their frequencies.

Algorithm 1 Selecting a subset of an event log to infer a process model that guarantees a minimum satisfaction level for one or multiple goals in each selected case

```
Input: EnhancedLog: An enhanced structured event log ▷ explained in Definition 2
       $Q_{\text{case}}$: A set of criteria; ▷ explained in Definition 3
       conf: a confidence level ▷ explained in Definition 3
Output: SelectedCases ▷ a subset of cases selected according to the criteria
                        and the all-or-none rule

1   sort_by_trace(EnhancedLog) ▷ sort the cases based on their traces
2   $trace(case_0) \leftarrow \langle\rangle$ ▷ $\langle\rangle$ is an empty trace, which cannot happen in reality
3   $trace(cases_{\text{NumberOfCases}+1}) \leftarrow \langle\rangle$ ▷ also flag the end of the log
4   SelectedCases ← ∅
5   index ← 1
6   while index ≤ NumberOfCases ▷ NumberOfCases is $m$ in Table 1
7       SameTraceCases ← ∅ ▷ a set of cases whose traces are the same
8       NumberOfSatisfiedCasesOfTrace ← 0 ▷ counts the satisfied cases of a trace
9       do
10          SameTraceCases ← SameTraceCases ∪ $\{case_{index}\}$
11          if $case_{index}$ meets all criteria of $Q_{\text{case}}$ then
12              NumberOfSatisfiedCasesOfTrace + +
13          end if
14          index + +
15      while $trace(case_{index}) = trace(case_{index-1})$
16      if NumberOfSatisfiedCasesOfTrace / size(SameTraceCases) ≥ conf then
17          SelectedCases = SelectedCases ∪ SameTraceCases
18      end if
19  end while
20  return SelectedCases ▷ the resulting subset of cases
```

One consequence of searching within cases is that there might be some cases with trace t_k that are eligible to be selected and, simultaneously, some cases with the same trace that are not. Although including the former cases and excluding the latter ones appears to be a simple solution, it would not be correct. The reason is that a discovered model either allows a trace (and all its cases) or avoids it. We respect an "*all-or-none*" rule, i.e., the *SelectedCases* should have either all cases of a same trace or none of them. Based on the above explanation, we define the three algorithms for selecting the best subset of cases regarding the three types of goal criteria and the *all-or-none* rule.

Algorithm 1: Guaranteeing One or Multiple Goals for All Cases. This type of goal-related criterion is looking for a model that guarantees (with a given confidence level) a predefined satisfaction level for one or multiple goals for all cases. This criterion considers every single case in a row viewpoint, therefore each case will be assessed against all goals considered in the criterion. There might be cases with trace t_k that meets all $q_i \in Q_{\text{case}}$ and some cases with the same trace that do not. Algorithm 1 checks all cases of one trace against Q_{case} (line 11). If the proportion of complying cases is not

inferior to the given confidence level *conf*, all the cases with that trace will be selected, otherwise all of them will be filtered out (lines 16–17). For example, assume *conf* is 0.8 and the event log has 100 cases with trace $\langle a, b, c, g \rangle$, including 83 cases that meet Q_{case} and 17 cases that do not. As 83% of these cases comply with the criterion, which is above the confidence level of 80%, all 100 cases are selected. Algorithm 1 first sorts all cases according to their trace (line 1). Searching within all cases of a trace and checking them against the criteria is hence efficient (lines 9–15).

Algorithm 2 Selecting a subset of an event log to infer a process model that guarantees the overall satisfaction level(s) of one or multiple goals

```
Input: EnhancedLog: An enhanced structured event log ▷ explained in Definition 2
$Q_{\text{goal}}$: A set of criteria (some goals and thresholds for their satisfaction level) ▷ Definition 3
$g$: A function computing the satisfaction of the whole goal model ▷ Definition 3
Output: SelectedCases ▷ a subset of all cases selected regarding the criteria and
                        the all-or-none rule, $SelectedCases \subseteq C$, Definition 4

SelectedCases ← ∅
Solve the binary optimization below: ($x_i$ is a flag for either selecting case$_i$ or not)
    $Max\ z = \sum_{i=1}^{m} x_i$ ▷ this is to find the largest subset
    s.t.
    $\forall\, r, t \;\; 1 \leq r < t \leq m$ : if $trace(c_r) = trace(c_t) \;\; x_r = x_t$ ▷ all-or-none rule
    $\forall\, j$ where $G_j \in \mathbb{G}$ : $\frac{\sum_{i=1}^{m} x_i . s_{i,j}}{\sum_{i=1}^{m} x_i} \geq \overline{sl_j}$ ▷ $|Q_{\text{goal}}|$ constraints
    $x_i = 0, 1$ ▷ if $x_i = 1$, case i should be selected
end of binary optimization
    for $i = 1$ to NumberOfCases do ▷ NumberOfCases is m in Table 1
        if $x_i = 1$ then
            SelectedCases ← SelectedCases ∪ {$c_i$}
        end
    end
return SelectedCases ▷ the resulting subset of cases that meets the criteria
```

Algorithm 2: Guaranteeing the Aggregated Satisfaction Levels of Goals. Here, in a column perspective, the focus is on the *aggregated* satisfaction level of one or multiple goals. Logically, the largest subset that simultaneously meets all criteria is the intersection of the largest subsets that separately meet all criteria. Therefore, one can focus on all considered goals individually and find the largest subsets for each G_j regarding $\overline{sl_j}$, then use their intersection as *SelectedCases*. However, finding the largest subset is not trivial because the largest subset that satisfies the criterion of one goal might not be unique and different subsets with similar (and largest) sizes may satisfy the condition. This might be the case even when the aggregation function is simple (e.g., *average*). In this situation, a subset that makes the largest intersection of all subsets (generated by all considered goals) should be selected. If this situation happens for several goals, we have to deal with the challenge of selecting one combination that finally makes the largest set.

Addressing such a difficulty, Algorithm 2 generates a binary optimization whose number of variables (x_i) equals the number of cases (m). The binary variable x_i is a flag variable associated to case c_i. If $x_i = 1$, then the case c_i will be selected and, if not, it will be excluded. As we are looking for the largest subset, $\sum x_i$ should be as large as possible. There are two categories of constraints for the optimization problem. The first aims to preserve the *all-or-none* rule, i.e., the selected subset should have either all cases of a same trace or none of them. The second category of constraints takes care of the threshold for the aggregated satisfaction level of each goal, i.e., $\overline{sl_j}$. This category of constraints is based on f_j, i.e., the aggregation function. In Algorithm 2 we assumed that all f_j are the *average* function (but others could be defined).

Algorithm 3 Selecting the largest subset of an event log to infer a process model that guarantees a comprehensive satisfaction level

```
Input:  EnhancedLog: An enhanced structured event log; ▷ as explained by Definition 2
        $Q_{Comp}$ : $\overline{sl}_{Comp}$ ▷ a minimum threshold for comprehensive satisfaction level
        F: a tuple of functions $(f_1, f_2, ..., f_n) \mid s_{Agg.j} = f_j(s_{1,j}, s_{2,j}, ..., s_{m,j}), j = 1,...,n$
        g: a function derived from the goal model. $s_{i.ove} = g(s_{i,1}, s_{i,2}, \dots, s_{i,n})$
Output: SelectedCases ▷ a subset of all cases selected regarding the criteria and the
        all-or-none rule, $SelectedCases \subseteq C$

SelectedCases ← ∅
if $s_{Comp} = f_{n+1}(s_{1.ove}, s_{2.ove}, \dots, s_{m.ove})$ then
    use Algorithm 2 and exit.
else
Solve the binary optimization below:
    $Max\ z = \sum_{i=1}^{m} x_i$  ▷ this is to find the largest subset.
    s.t.
    $\forall r, t\ \ 1 \le r < t \le m$ : if $trace(c_r) = trace(c_t)$  $x_r = x_t$ ▷ all-or-none rule
    $g\left(\frac{\sum_{i=1}^{m} x_i.s_{i1}}{\sum_{i=1}^{m} x_i}, \frac{\sum_{i=1}^{m} x_i.s_{i2}}{\sum_{i=1}^{m} x_i}, \dots, \frac{\sum_{i=1}^{m} x_i.s_{in}}{\sum_{i=1}^{m} x_i}\right) \ge \overline{sl}_{Comp}$
    $x_i = 0, 1$
end
for $i = 1$ to NumberOfCases do ▷ NumberOfCases is m in Table 1
    if $x_i = 1$ then
        SelectedCases ← SelectedCases ∪ {$c_i$}
    end
end
return SelectedCases ▷ the resulting subset of cases that meets the criterion
```

Algorithm 3: Guaranteeing Comprehensive Satisfaction Levels. The two above types of criteria considered the goals from a row perspective and a column perspective. The third type of criteria, however, considers the goals from a *table* perspective.

Here, the overall satisfaction level of all columns is aggregated and represented by one number as a *comprehensive satisfaction level*. Finding the largest subset of cases that guarantees a minimum threshold for comprehensive satisfaction level ($\overline{sl}_{comp}$) is,

Table 2. *EnhancedLog*, event log and satisfaction level of goals for the DGD process

Case	Trace	G_1: To decrease process time	G_2: To decrease cost	G_3: To do a smooth process	G_4: To screen accurately	Overall (To satisfy the patient)
Patient#1	$\langle a, b, c, g\rangle$	100	100	88	100	97
Patient#2	$\langle a, b, c, g\rangle$	94	100	88	100	95
Patient#3	$\langle a, b, c, g\rangle$	94	100	88	0	0
Patient#4	$\langle a, b, c, d, e, c, g\rangle$	61	59	75	100	64
Patient#5	$\langle a, b, c, d, e, c, g\rangle$	72	59	63	100	65
Patient#6	$\langle a, b, c, d, e, c, g\rangle$	67	59	75	100	66
Patient#7	$\langle a, b, c, f, b, c, g\rangle$	78	82	63	100	76
Patient#8	$\langle a, b, c, f, b, c, d, e, c, g\rangle$	41	20	50	100	36
Patient#9	$\langle a, b, c, f, b, c, d, e, c, g\rangle$	43	20	40	100	34
Patient#10	$\langle a, b, c, d, b, c, d, e, c, g\rangle$	9	10	30	100	15
Aggregated satisfaction:		65.9	60.9	66	90	**64.1**

also, not trivial. This is because adding a trace to the subset might increase the aggregated level of one goal and, at the same time, decrease the level of another goal.

As explained in Definition 3, the *comprehensive* level might be calculated in two different ways. It can be the overall satisfaction level of the aggregated levels of all goals, or the aggregated level of the overall satisfaction level of each case. The latter one will work only with the column of the overall satisfaction level (the right column of Table 1), therefore, the problem will be solved in a way similar to that of the second type of criteria (Algorithm 2). Accordingly, Algorithm 3. first checks the definition of *comprehensive*. If it is not like Algorithm 2, then Algorithm 3 generates a new binary optimization problem. Here, the first category of constrains aims to preserve the *all-or-none* rule, whereas the last constraint makes sure that the comprehensive level of the selected subset is not less than the given threshold $\overline{sl}_{comp}$.

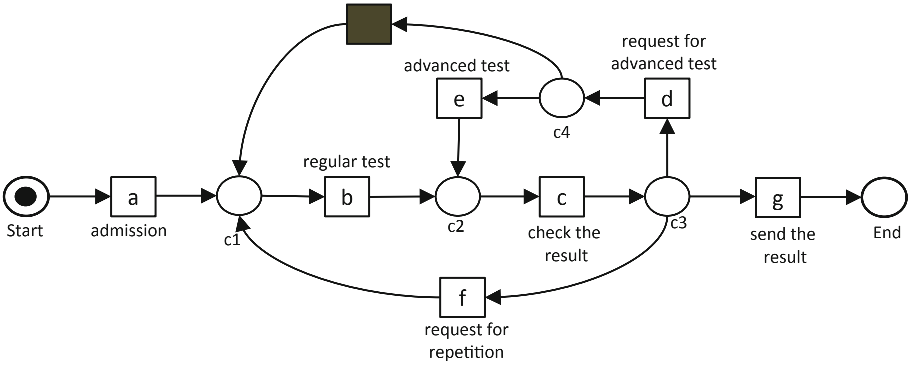
4 Illustrative Example

The process of *diagnosis of gestational diabetes* (DGD) will be used to illustrate the proposed methods. To this end, three types of goal-related criteria discussed in Sect. 2 will be taken into consideration. The main assumption here is that the log is realistic but *not real* and is used only to study the GoPED method and its algorithms.

4.1 Event Log of an Illustrative DGD Process

The event log of 10 patients who have used the DGD process is shown in Table 2. We use short names to encode the activities: a = admission, b = regular test, c = check the result, d = request for advanced test, e = advanced test, f = request for repetition, and g = send the result. According to Table 2 and the definitions described in Sect. 3.1, the event log (L) includes five different variants of traces:

$$L = [\langle a,b,c,g\rangle^3, \langle a,b,c,d,e,c,g\rangle^3, \langle a,b,c,f,b,c,g\rangle^1,$$
$$\langle a,b,c,f,b,c,d,e,c,g\rangle^2, \langle a,b,c,d,b,c,d,e,c,d,e,c,g\rangle^1]$$



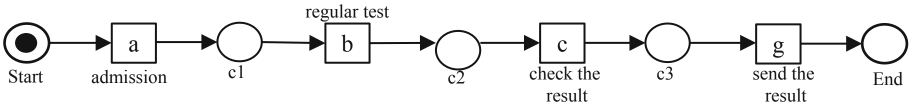
Model 1. Process model discovered from the original event log by the α-algorithm

As shown in Table 2, there are four additional goal-oriented fields related to the DGD process. Due to space limitation, we directly show the satisfaction levels of the goals in the table (without their indicators or actors), which are values in the range [0–100].

An advanced version of the α-algorithm [17] generates Model 1 using the whole event log. The main story of this DGD process is as follows: after admission of a patient, a regular blood test is done. Then, based on the result of the test the patient may need to do an advanced test, the patient may need to repeat the regular test, or the result will be sent to the related department, and then the process ends. A *silent* transition is shown in Model 1 with black color. That is a particular transition not observable in the event log, but needed to make a *sound* Petri-net [16]. Considering the traces, the source of this need is that for Patient#10, after the activity *d*, request for advanced test, the activity *b*, regular test, has executed, while the activity *e*, advanced test, was supposed to execute. Model 1 will be used as a basis for considering the resulting models from GoPED respecting three types of goal-related criteria.

4.2 Example Models Resulting from GoPED

Guaranteeing Satisfaction of One or Multiple Goals for All Cases. This goal-related criterion is looking for a model that guarantees a predefined satisfaction level for all cases in terms of one or multiple goals, with a given confidence level. For example, the condition is as follows:



Model 2. To satisfy goal criteria of Q_{case}

- *Case perspective:* generate a model that guarantees (with a confidence of 90%) that the satisfaction level for all patients in terms of goal "To decrease process time" will be at least 75 and that in terms of goal "To do a smooth process" will be at least 80.

In this case, we have Q_{case} = {(G$_1$, 75), (G$_3$, 80)} and *conf* = 0.9. Using Algorithm 1, only all the cases of trace $\langle a, b, c, g\rangle$ are returned, i.e., Patients #1, #2 and #3. All cases of this trace meet $Q_{case.,}$ so the fraction of eligible cases of this trace is 100%, which is more than the required 90% confidence level. Such a parameter for the four remaining traces is zero, which is less than 90% by far. Therefore, we have *SelectedCases* = {Patient#1, Patient#2, Patient#3}, resulting in the log $\{\langle a, b, c, g\rangle^3\}$. The α-algorithm [17] produces Model 2 from this log. This is the process to be encouraged in the organization to meet these goals.

Guaranteeing the Aggregated Satisfaction Levels of Goals. Here, from a column perspective, the focus is on the aggregated satisfaction level of one or multiple goals rather than on the satisfaction of every single case.

- *Goal perspective:* Generate a model that results in an *aggregated* satisfaction levels of the goal "To decrease time process" higher than 80 and of the goal "To do a smooth process" higher than 78, simultaneously.

In this case, we have Q_{goal} = {(G$_1$, 80), (G$_3$, 78)}. Here the functions *f* showing how to calculate the aggregation of each column is required. Let us assume that for all goals in the DGD process, the function is the *average*. Therefore, the optimization problem of Algorithm 2 can be formalized as follows:

$$Max\ z = \sum_{i=1}^{10} x_i$$

s.t.

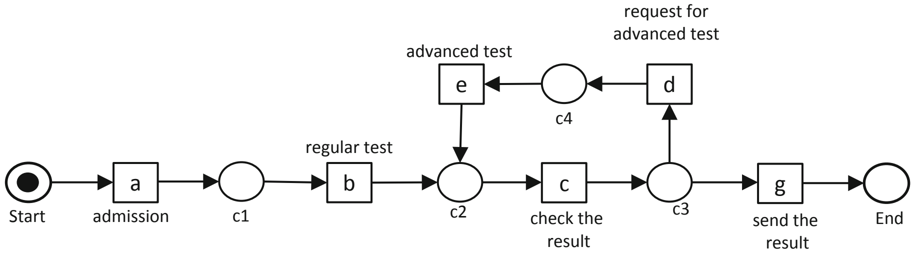
$$x_1 = x_2 = x_{3,} \quad x_4 = x_5 = x_{6,} \quad x_8 = x_9 \quad \leftarrow (\textit{all-or-none}\ \text{rule})$$

$$\frac{100x_1+94x_2+94x_3+61x_4+72x_5+67x_6+78x_7+41x_8+43x_9+9x_{10}}{\sum_{i=1}^{m} x_i} \geq 80$$

$$\frac{88x_1+88x_2+88x_3+75x_4+63x_5+75x_6+63x_7+50x_8+40x_9+30x_{10}}{\sum_{i=1}^{m} x_i} \geq 78$$

$$x_i = 0\ ,\ 1$$

The answer of the above problem is unique: $x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = 1$ and $x_7 = x_8 = x_9 = x_{10} = 0$. Therefore, *SelectedCases* = {Patient#1, Patient#2, Patient#3, Patient#4, Patient#5, Patient#6}, leading to the log $\{\langle a, b, c, g\rangle^3, \langle a, b, c, d, e, c, g\rangle^3\}$. For this subset of cases, the aggregation satisfaction level for G1 and G3 will be 81.3 and 79.5, respectively. The α-algorithm produces Model 3 from this log.



Model 3. To satisfy goal criteria of Q_{goal}

Guaranteeing Comprehensive Satisfaction Levels. Here, from a table perspective, the focus is on the *comprehensive* satisfaction level, which we assume to be the *overall* satisfaction level of the *aggregated* levels of all goals. Therefore, the goal model should be used to evaluate s_{Comp} based on the satisfaction levels of all sub goals. Figure 2 shows the goals model related to the DGD process using the GRL language. In the graph, the root is the main goal and the sub goals are the leaves. Based on the goal model and its AND/OR refinements and the weight of contributions, the s_{Comp} is defined as follows:

$$s_{Comp} = Sat(G_6) = \text{Minimum}(s_{Agg.4,}\, 0.4 \times s_{Agg.1} + 0.35 \times s_{Agg.2} + 0.25 \times s_{Agg.3})$$

This kind of evaluation is known as *forward propagation* in GRL. The jUCMNav tool is an Eclipse-based graphical editor that can be used for evaluating GRL models [9].

- *Organization perspective*: Generate a model where the comprehensive satisfaction level is no less than 75.

The above criterion leads to $Q_{\text{Comp}} = 75$. Recall that $s_{Agg.j}$ is the aggregated satisfaction of goal j, in our case the *average* of column of Goal$_j$ in Table 2. According to the function derived from the goal model of Fig. 2, Algorithm 3 generates the optimization problem as follows:

$$Max\ z = \sum_{i=1}^{10} x_i$$

s.t.

$$x_1 = x_2 = x_3, \quad x_4 = x_5 = x_6, \quad x_8 = x_9 \quad \leftarrow (\textit{all-or-none}\ \text{rule})$$

$$Minimum\left(\frac{\sum_{i=1}^{m} x_i . s_{i,4}}{\sum_{i=1}^{m} x_i}\ ,\ 0.4 \times \frac{\sum_{i=1}^{m} x_i . s_{i,1}}{\sum_{i=1}^{m} x_i} + 0.35 \times \frac{\sum_{i=1}^{m} x_i . s_{i,2}}{\sum_{i=1}^{m} x_i} + 0.25 \times \frac{\sum_{i=1}^{m} x_i . s_{i,3}}{\sum_{i=1}^{m} x_i}\right) \geq 75$$

$$x_i = 0,\ 1$$

($s_{i,j}$ refers to the cells of Table 2, e.g., $s_{2,1} = 94$).

The answer of the above problem is, also, unique: $x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = x_7 = 1$ and $x_8 = x_9 = x_{10} = 0$. Therefore, *SelectedCases* = {Patient#1, Patient#2, Patient#3, Patient#4, Patient#5, Patient#6, Patient#7}, resulting in the log $\{\langle a, b, c, g\rangle^3, \langle a, b, c, d, e, c, g\rangle^3, \langle a, b, c, f, b, c, g\rangle\}$. For this subset, the *comprehensive satisfaction level* is 79.5. The α-algorithm produces Model 4 from this log.

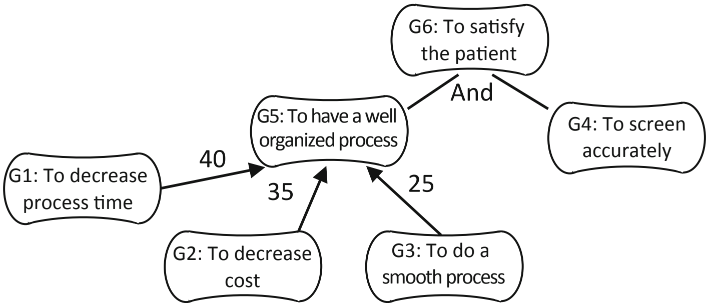
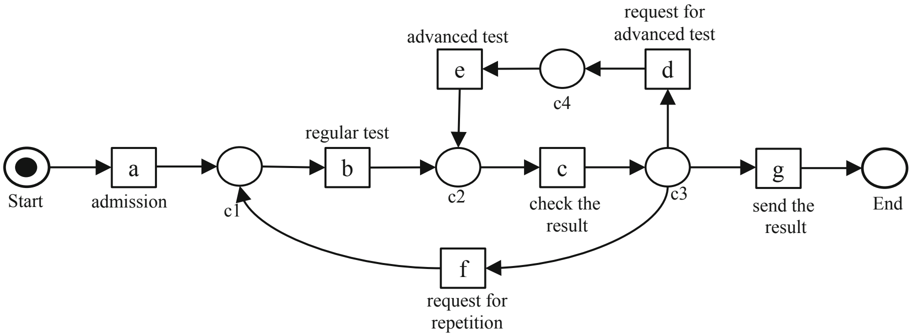

G6: To satisfy the patient
And
G5: To have a well organized process
G4: To screen accurately
G1: To decrease process time
40
35
25
G2: To decrease cost
G3: To do a smooth process

Fig. 2. Goal model showing the relations between the goals pursued by the DGD process



Model 4. To satisfy goal criteria of $\mathrm{Q_{Comp}}$

4.3 Discussion

We generated three models using three types of goal-related criteria. Comparing the models, we find that the main difference between Model 2 and the other models relates to the loops. Model 2 does not allow repeating the regular test or to do an advanced test. Here, the decision point "check the result" is spurious as it will not actually make any decision. According to the goals considered in the generation of such a model, i.e., G1 and G3, one can hypothesize that doing advanced blood test or repeating the regular test are not aligned with the goals of having short process time and of providing all patients with a smooth process. Considering the trace $\langle a, b, c, g \rangle$ that Model 2 can generate, we find that there are three patients who experienced this trace. One of these patients has been diagnosed wrongly ($s_{3,4} = 0$). The goal model shown in Fig. 2 implies that the goal "To screen accurately" participates to an AND refinement; therefore, when this goal is denied, regardless of the other goals in the refinement, the main goal of the process gets denied. Using the goal model, the overall satisfaction level of those three patients are 97, 95 and 0, respectively. This suggests that although Model 2 highly satisfies all cases in terms of process time and smoothness of the process, it will end up with a third of the patients who will be dissatisfied.

The above analyses are simple examples of knowledge that GoPED can provide. Such knowledge, together with the discovered models, can help domain experts (re)design goal-aligned process models, encourage good behaviors, and discourage bad ones.

5 Related Work

Our systematic literature review of goal-oriented process mining showed that although process mining and goal modeling are growing research topics, there are only a few rare studies conducted at their intersection [7]. Therefore, this suggests that goal-oriented processes discovery can still be considered a gap to be filled between the process mining and requirements engineering communities.

From an *agent viewpoint*, the goals behind activities of agents who contribute in a process (e.g., employees) are considered by Yan et al. [18]. Their proposed approach adopts a decision tree algorithm to learn goals of agents by classifying their activities in different situations. In this viewpoint, Outmazgin and Soffer [11] used process discovery techniques to analyze different types of intentional incompliances, where employees intentionally deviate from prescribed models, to find their causes.

In addition, in a *process viewpoint* (or case/customer viewpoint) all activities constituting a trace are considered. Ponnalagu et al. [13] proposed an approach for analyzing and validating a family of variants of a single process based on a goal model. From the same view point, Horita et al. [8] proposed a method to detect and analyze the effects of disagreements between real logs and prescribed models using a goal-oriented conformance checking approach. Bernard and Andritsos [3] used process discovery in conjunction with customers' journeys and developed a tool that facilitates navigation through many different journeys in a goal-oriented fashion.

An *organization viewpoint* considers the overall goals that should be achieved by performing business processes. Santiputri et al. [14] considered the sequence of events in multi-layered event logs and proposed an approach to discover goal refinement patterns of the goal models.

Trace clustering is a solution proposed by the process mining community to improve interpretability of discovered process models by splitting different behaviors on different process perspectives into multiple sub-logs. Similar to our approach, the main idea in the existing clustering approaches is to discover models from subsets of logs. However, the clustering approach yet considers the log at an activity-level and does not bring the satisfaction of competing goals of stakeholders into account [10, 15].

6 Conclusion

This paper proposed a new method that exploits the capabilities of goal modeling and performs process discovery in a goal-oriented fashion. This method first enhances an event log by adding new goal-related information to all traces. Then, it quantifies the satisfaction level of goals using a goal model. Such a goal model shows correlations between (often conflicting) goals of different stakeholders and allows what-if analysis and balancing trade-offs between confliction goals. Three types of goal-related criteria were introduced as the basis for generating goal-oriented models promising to achieve predefined goals. The real behaviors that are aligned with the goals and achieve desired satisfaction levels are selected. Three algorithms for such a selection were explicitly explained. The selected subset becomes the basis for conventional process discovery. The resulting model can be compared to a model discovered from the original event log to reveal new insights about the ability of different forms of process models to satisfy the goals. Learning from *good* behaviors that satisfy goals and detecting *bad* behaviors that hurt them is an opportunity to redesign models so they are better aligned with goals.

Acknowledgment. This work was supported by Natural Sciences and Engineering Research Council of Canada (NSERC, Discovery and CGS-D).

References

1. Amyot, D., Ghanavati, S., Horkoff, J., Mussbacher, G., Peyton, L., Yu, E.: Evaluating goal models within the goal-oriented requirement language. Int. J. Intell. Syst. **25**(8), 841–877 (2010). https://doi.org/10.1002/int.20433
2. Amyot, D., Mussbacher, G.: User requirements notation: the first ten years, the next ten years. J. Softw. **6**(5), 747–768 (2011)
3. Bernard, B., Andritsos, P.: CJM-ex: goal-oriented exploration of customer journey maps using event logs and data analytics. In: BPM Demo Track and BPM Dissertation Award (BPMD&DA), vol. 1920. EUR-WS (2017)
4. Fan, Y., Anda, A.A., Amyot, D.: An arithmetic semantics for GRL goal models with function generation. In: Khendek, F., Gotzhein, R. (eds.) SAM 2018. LNCS, vol. 11150, pp. 144–162. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01042-3_9
5. Ghasemi, M.: Towards goal-oriented process mining. In: 2018 IEEE 26th International Requirements Engineering Conference (RE), pp. 484–489. IEEE CS (2018). https://doi.org/10.1109/re.2018.00066
6. Ghasemi, M.: What requirements engineering can learn from process mining. In: 2018 1st International Workshop on Learning from other Disciplines for Requirements Engineering (D4RE), pp. 8–11. IEEE (2018). https://doi.org/10.1109/d4re.2018.00008
7. Ghasemi, M., Amyot, D.: From event logs to goals: a systematic literature review of goal-oriented process mining. Requir. Eng. 1–27 (2019). https://doi.org/10.1007/s00766-018-00308-3
8. Horita, H., Hirayama, H., Tahara, Y., Ohsuga, A.: Towards goal-oriented conformance checking. In: Proceedings of the International Conference on Software Engineering and Knowledge Engineering SEKE, pp. 722–724 (2015)
9. jUCMNav (2016). http://softwareengineering.ca/jucmnav
10. Mannhardt, F., de Leoni, M., Reijers, H., van der Aalst, W., Toussaint, P.: Guided process discovery – a pattern-based approach. Inf. Syst. **76**, 1–18 (2018). https://doi.org/10.1016/j.is.2018.01.009
11. Outmazgin, N., Soffer, P.: A process mining-based analysis of business process workarounds. Softw. Syst. Model. **15**(2), 309–323 (2016)

12. Pesic, M., van der Aalst, W.M.P.: A declarative approach for flexible business processes management. In: Eder, J., Dustdar, S. (eds.) BPM 2006. LNCS, vol. 4103, pp. 169–180. Springer, Heidelberg (2006). https://doi.org/10.1007/11837862_18
13. Ponnalagu, K., Ghose, A., Narendra, Nanjangud C., Dam, H.K.: Goal-aligned categorization of instance variants in knowledge-intensive processes. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) BPM 2015. LNCS, vol. 9253, pp. 350–364. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23063-4_24
14. Santiputri, M., Deb, N., Khan, M.A., Ghose, A., Dam, H., Chaki, N.: Mining goal refinement patterns: distilling know-how from data. In: Mayr, H.C., Guizzardi, G., Ma, H., Pastor, O. (eds.) ER 2017. LNCS, vol. 10650, pp. 69–76. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69904-2_6
15. Seeliger, A., Nolle, T., Mühlhäuser, M.: Finding structure in the unstructured: hybrid feature set clustering for process discovery. In: Weske, M., Montali, M., Weber, I., vom Brocke, J. (eds.) BPM 2018. LNCS, vol. 11080, pp. 288–304. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98648-7_17
16. van der Aalst, W.: Process Mining Data Science in Action, 2nd edn. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
17. Wen, L., van der Aalst, W., Wang, J., Sun, J.: Mining process models with non-free-choice constructs. Data Min. Knowl. Disc. **15**(2), 145–180 (2007)
18. Yan, J., Hu, D., Liao, S., Wang, H.: Mining agents' goals in agent-oriented business processes. ACM Trans. Manag. Inf. Syst. **5**(4), 1–22 (2014)