Multi-Objective Optimization for Automated Business Process Discovery

Mohamed A. Ghazal[a], Samy Ghoniemy[b] and Mostafa A. Salama[c]
Department of Computer Science, The British University in Egypt, Cairo, Egypt

Keywords: Multi-Objective Optimization, Process Mining, Multi-Objective Evolutionary Algorithms, Process Model Discovery, Non-dominated Sorting Genetic Algorithm II.

Abstract: Process Mining is a research field that aims to develop new techniques to discover, monitor and improve real processes by extracting knowledge from event logs. This relatively young research discipline has evidenced efficacy in various applications, especially in application domains where a dynamic behavior needs to be related to process models. Process Model Discovery is presumably the most important task in Process Mining since the discovered models can be used as an objective starting points for any further process analysis to be conducted. There are various quality dimensions the model should consider during discovery such as Replay-Fitness, Precision, Generalization, and Simplicity. It becomes evident that Process Model Discovery, with its current given settings, is a Multi-Objective Optimization Problem. However, most existing techniques does not approach the problem as a Multi-Objective Optimization Problem. Therefore, in this work we propose the use of one of the most robust and widely used Multi-Objective Optimizers in Process Model Discovery, the NSGA-II algorithm. Experimental results on a real life event log shows that the proposed technique outperforms existing techniques in various aspects. Also this work tries to establish a benchmarking system for comparing results of Multi-Objective Optimization based Process Model Discovery techniques.

1 INTRODUCTION

In the last decade Process Mining has proved its effectiveness in every industrial application and is gaining popularity among the research community (van der Aalst et al., 2007). Process mining can be defined as this emerging discipline providing comprehensive sets of tools to provide fact-based insights and to support operational processes. Process model discovery is one of the three main types of process mining (van der Aalst, 2011) and it is also sometimes referred to as Workflow Mining (van der Aalst et al., 2004). In this work (ABPD) will be used which stands for Automated Business Process Discovery. ABPD aims to automatically infer process models that accurately describe any process under analysis by considering only available records of this process. In other words there is no prior process model, the model is discovered based on event logs only.

ABPD is a very challenging task. Naturally event logs are often noisy, and far from being complete. (van der Aalst et al., 2004; Van der Aalst et al., 2005). Adding to this, the quality of the automatically discovered process model should be assessed on several quality dimensions which are actually competing with each other (van der Aalst, 2011). Generally speaking, four quality dimensions are often used to measure the results of ABPD namely Replay Fitness, Precision, Simplicity and Generalization. Replay Fitness quantifies the extent to which the discovered model can accurately reproduce the recorded behaviour in the log. Replay Fitness by itself was widely used as the main measure for the performance of a process discovery algorithm because it only makes sense to consider other dimensions if the replay fitness is acceptable. Precision ensures that the model does not underfit the event log. In other words a Precision metric quantifies the fraction of the behavior allowed by the model which is not seen in the event log. Generalization assesses the extent to which the model generalizes the behavior in the log to avoid overfitting the data at hand (i.e. the available traces in the event log). Simplicity measures the complexity of a model and more often irrespective of the event log.

Recently, several ABPD techniques have been developed. Discovering process models which can be graphically represented in different process modelling

[a] https://orcid.org/0000-0001-5438-9320
[b] https://orcid.org/0000-0001-7327-4983
[c] https://orcid.org/0000-0003-2559-8056

Ghazal, M., Ghoniemy, S. and Salama, M.
Multi-Objective Optimization for Automated Business Process Discovery.
DOI: 10.5220/0008072400890104
In *Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2019)*, pages 89-104
ISBN: 978-989-758-382-7

notation. The most prominent discovery techniques can be roughly categorized into two groups based on the search strategy and the nature of the algorithm used. The first group uses a local search strategy and here will be referred to as the conventional techniques. It includes ABPD techniques that uses general algorithmic approach or a frequency based heuristics approach. These techniques have some known drawbacks, especially their inability to focus on more than one or at maximum two quality dimensions at the same time (Buijs et al., 2012b). Also conventional ABPD typically generates a single process model that may not describe the recorded behavior effectively (Buijs et al., 2013). The second group includes ABPD techniques adopting general search strategies, mainly a meta-heuristic evolutionary approach. The majority of these techniques, except for a single one proposed in (Buijs et al., 2013) uses a single-objective meta-heuristic approach. Consequently, not only it has the same problem as for the conventional techniques in producing a single solution each run. Moreover, it mainly depends on the weighted sum method(WSM) (Marler and Arora, 2010). Section 3 of this paper discusses the shortcomings expected with the use of the WSM in more details. What is important to note here is that even those techniques used a multi-objective meta-heuristic approach, none of which used any of the well recognized Multi-Objective Optimization Evolutionary Algorithms (MOEAs), such as NSGA-II (Kalyanmoy et al., 2002), SPEA2 (Zitzler et al., 2000), or the PAES (Knowles and Corne, 2000). In other words, the ABPD techniques which adopted a multi-objective meta-heuristic approaches were focusing on incorporating some ideas from some notable Pareto-Front(PF) Based multi-objective optimizers such as the crowding distance selection (Kalyanmoy et al., 2002). But never truly considered using one of the recognized MOEAs. Not only this makes the performance of such discovery techniques questionable, but also raises many other questions such as how to evaluate and compare the results of such techniques.

The remainder of the paper is structured as follows. Next, in Section 2 a literature review on ABPD is presented. In Section 3 a brief review on Multi-Objective Optimization(MOO) and a discussion on what benefits MOEAs can bring to ABPD. Section 4 explain the proposed Multi-Objective Evolutionary Tree Miner(MOETM) and how it has been built to extend the Evolutionary Tree Miner(ETM) (Buijs et al., 2012a) by incorporating a new NSGA-II based evolutionary engine. In Section 5, the proposed technique was experimented with a real-life event log and the findings are discussed. Section 6 concludes the paper.

2 PROCESS MODEL DISCOVERY

Since the process model is the starting point for most of the process mining activities ABPD is one of the most important research topics in process mining. As mentioned earlier ABPD aims to discover a process model that reflects the causal dependencies of activities observed in an event log. Then presenting the model in one of the process modeling notations. Accordingly, ABPD can be conceived as a search problem (i.e. a search for the most appropriate process model of the search space of candidate process models). In the last two decades, various techniques have been developed (e.g. Alpha (van der Aalst et al., 2004), Heuristic (Weijters et al., 2006), Fuzzy (Günther and Van Der Aalst, 2007), Genetic Miners (Van der Aalst et al., 2005)) producing process models in various forms (e.g. Petri nets, BPMN models, EPCs, YAWL-models). As in (Van der Aalst et al., 2005) ABPD techniques can be categorized into two groups based on the search strategy adopted a local or global search strategy. Adding to this and from a chronological point of view these two categories can be referred to as conventional ABPD techniques and the more recent meta-heuristics based ABPD techniques.

2.1 Conventional ABPD Techniques

Table 1: Control-flow discovery algorithms based on their output model type.

Discovery Algorithm	Output model type
-Alpha, Alpha++, and Alpha# algorithms -Parikh Language-based Region miner -Region miner -Tsinghua-alpha algorithm -Petrify mining	Petri net
-Duplicate Tasks GA -Genetic algorithm -Heuristics miner	Heuristic net
-Frequency abstraction miner -Fuzzy Miner	Fuzzy model
-FSM miner -k-RI Miner	Finite state machine / Transition system
-Multi-phase Macro Plugin	EPC Event-driven Process Chain
-DWS mining plug-in -Workflow patterns miner	Other

ABPD techniques adopting a local search strategy by means of using a general algorithmic or a

frequency based heuristics approach here will be referred to as conventional techniques. Table 1 list the most notable and widely used of these techniques based on their output model type. Lack of space allows us only to summarize the literature review regarding the conventional techniques. If more background on this topic is required, there are several good references that the reader is invited to consult such as the literature reviews and the comparative studies in (Gupta, 2014; Tiwari et al., 2008). What could be said briefly here is that these conventional techniques are known for the following problems:

- These techniques typically returns a single process model each run, which sometimes may not be able to effectively describe the recorded behavior (Buijs et al., 2013).
- They suffer the inability to focus on more than one or two quality dimensions at the same time (Buijs et al., 2012c).
- A major drawback is that they can not mine all the common constructs of a process model (Van der Aalst et al., 2005). Such Problematic constructs are discussed in details in (de Medeiros, 2006).

2.2 Meta-heuristic ABPD

The first time a meta-heuristic method was used in ABPD was when the authors in (de Medeiros, 2006) used a Genetic Algorithm (GA) to discover process models represented in *petri-nets*. In (de Medeiros, 2006) the author used a two dimensional fitness measure. There was a problem with the measurement of the preciseness dimension and it was actually claimed that it is not practical to accurately measure this dimension. Moreover the genetic process model discovery algorithm suffered from the long execution time. In (Bratosin et al., 2010a), it was argued that the fitness calculation phase of the algorithm was the main reason for the long running time and a sampling technique to address the issue was proposed. A sample of the event log in the fitness calculation phase was used instead of the whole log. In (Bratosin et al., 2010b) a distributed architecture for the genetic process discovery algorithm was proposed to further improve the performance regarding the execution time. The work in (Tsai et al., 2010) extended the work in (de Medeiros, 2006) by adding a time interval analysis between the events. In all of the studies mentioned so far petri-nets were used as the internal representation. The use of petri-nets as an internal representation was a fundamental reason behind the genetic ABPD limited performance. One of the main requirements for an ABPD technique is to produce error-free models, also known as sound models. A definition for the notion of soundness for petri-nets can be found in (van der Aalst et al., 2011), likewise a deep analysis for all important requirements which should be considered when choosing a suitable process model notation for a process discovery algorithm can be found in the PhD dissertation in (Buijs, 2014).

The work in (Buijs et al., 2012b) proposed the use of a tree representation to ensure the soundness of the model. *Process trees* (Buijs, 2014) ensured soundness since block-structured process models are inherently sound because they require that each control-flow split has a corresponding join of the same type. An example illustrating the notion of process discovery using process trees can be found in the first section (Introduction) in (Buijs et al., 2012b). The use of *process trees* as an internal representation in a genetic ABPD not only ensures a sound model in the final output. Moreover, it improves the performance of the genetic algorithm due to the considerable reduction in the size of the search space. When using Petri-nets to describe process models, the search space consists of all possible Petri-nets of both correct and incorrect models. However, process trees regardless of how they are created, are always sound process models. This means that many of the unwanted unsound models are not going to be created in the first place resulting a reduction in the size of the search space. In addition to introducing *process trees* the work in (Buijs et al., 2012b) also proposed a new fitness measure reflecting the quality metrics for process models described in (van der Aalst, 2011).

All previous studies mentioned so far use a classical optimization method by converting a multi-objective optimization problem (MOP) into a single-objective optimization problem (SOP) and emphasize one particular optimal solution as the final result. Generating a single process is not the only problem these techniques suffers. Most importantly these techniques use the weighted sum method to reformulate the MOP into an SOP. The WSM has some known drawbacks when used for solving MOPs (Marler and Arora, 2010). The next section in this paper discusses this in a bit more details and it will become apparent that especially in the case of genetic ABPD the WSM may not be a good choice. To the best of our knowledge only one case study adopts a multi-objective optimization approach in ABPD. The researchers in their work in (Buijs et al., 2012b) concluded that often is not one single process model that describes the observed behavior best in all quality dimensions. Motivated by the findings in their previous work they further extended it in (Buijs et al., 2013) to obtain a collection of mutually non-dominating process models. While the proposed algorithm is following a multi-objective optimization methodology and even uses

a fitness function inspired by the *crowding distance* used in NSGA-II. However, there are some fundamental differences between the solution proposed in (Buijs et al., 2013) and most of the recognized Multi-Objective Evolutionary Algorithms (MOEAs). The size of the Pareto-Front which is unbounded by any limits such as the size of the population, the overall fitness calculation, and the selection strategy are some of these notable differences. Hence, from a MOO point of view the performance of this solution remains questionable. In addition, how to assess the performance of MOO techniques in the context of ABPD and how to compare its outcomes represent a far more interesting research questions.

3 MULTI-OBJECTIVE OPTIMIZATION

Optimization refers to find the best possible solution to a problem given a set of limitations (or constraints). When building an optimizer for a SOP, the aim is to find the best possible solution available (i.e. global optimum) or at least a good approximation of it. However, in most real world problems there is not one but several objectives to be optimized simultaneously. And in fact it is normally the case that these objectives are in conflict with each other. These problems with two or more objective functions are called MOPs and require different mathematical and algorithmic tools than those adopted to solve SOPs (Tamaki et al., 1996). Moreover, even the notion of *optimality* changes when dealing with MOPs. In MOP the optimization result is not a single solution like in SOP rather its a set of solutions. And it is often unclear which one constitutes an optimal solution. A solution may be optimal for one objective function but sub-optimal for another. Thus, it is required to find a number of solutions in order to provide the decision maker with insight into the characteristics of the problem before a final solution is chosen (Marler and Arora, 2004). The set of solutions, which are the optimization result of an MOP are often called Pareto-Front (*PF*). The basic idea is to obtain a set of solutions which are mutually non-dominating. A solution dominates another if for all objective functions it is at least equal or better, and is strictly better in at least one objective.

3.1 The Weighted Sum Method (WSM)

The main goal in solving a MOP is to obtain a Pareto-optimal set (or to sample solutions from the set as uniformly as possible). Classical optimization methods suggest converting the MOP into a SOP by emphasizing one particular Pareto-optimal solution at a time (Kalyanmoy et al., 2002). When there is a need to find multiple solutions using this method it has to be applied many times hopefully finding a different solution at each simulation run. Several methods were proposed to convert a MOP into an appropriately formulated SOP. Despite deficiencies with respect to depicting the Pareto-optimal set the WSM continues to be used extensively in MOO (Marler and Arora, 2010). And in the case of ABPD there is no much difference as it is observable that all discovery techniques which employed a meta-heuristic approach (de Medeiros, 2006; Bratosin et al., 2010a; Bratosin et al., 2010b; Tsai et al., 2010; Buijs et al., 2012a) except for (Buijs et al., 2013), used the WSM. However, this method is known for its weaknesses in solving MOPs. According to (Buijs et al., 2013) the following drawbacks can be listed:

- Determining the correct weights upfront is difficult. A small change in in weights may results in big changes in the objective vectors.
- Since only one solution is returned if the solution is not acceptable due to inappropriate setting of the weights, new runs of the optimizer is required

More importantly the following should also be emphasize. Generally speaking, the WSM is having a widely known problem of being very sensitive to the shape of the Pareto frontier (convex, or concave) or to discontinuous Pareto fronts. In (Marler and Arora, 2010) it is stated that many case studies demonstrated the method's inability to capture Pareto optimal points that lie on non-convex portions of the Pareto optimal curve. It is also acknowledged the method does not provide an even distribution of points in the Pareto optimal set. This will have a crucial impact in the case of ABPD. ABPD is currently being solved as a maximization MOP and with its current settings a non-convex if not a concave shape of the *PF* is often expected. Figure 1 illustrates the expected shape of the *PF* in both minimization and maximization problems along with ideal and nadir points in both cases. More resources and information on the WSM and the notion of convexity (concavity) is in Appendix B.

3.2 Multi-Objective Evolutionary Algorithms (MOEAs)

One of the most successful methods applied to obtain a *PF* in MOO is evolutionary algorithms (EAs). The main reason for this is their ability to find multiple Pareto optimal solutions in one single simulation run (Kalyanmoy et al., 2002). In the last two

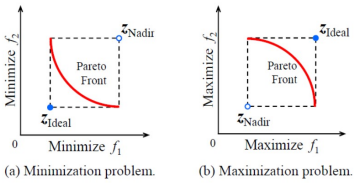


Figure 1: Pareto front with ideal and nadir points for minimization and maximization problems, from (Ishibuchi et al., 2017).

decades a number of MOEAs have been suggested. The most prominent are NSGA-II (Kalyanmoy et al., 2002), PAES (Knowles and Corne, 2000), and SPEA-II) (Zitzler et al., 2000). Several studies had extensively compared them and other MOEAs regarding their performance, but no clear overall winner can be announced (Gadhvi et al., 2016). MOEAs performance assessment in and of itself is a very active research area. In fact the amount of effort in this area of research is immense and several innovations in various studies can be found (e.g. (Knowles et al., 2006; Azarm and Wu, 2001; Li et al., 2014; Schott, 1995; Van Veldhuizen, 1999; Riquelme-Granada et al., 2015)). However, the lack of a unified MOEA Evaluation Framework to systematically compare algorithms which can be used and extended by researchers to benchmark the different algorithms makes comparing the different algorithms a bit problematic. As described in (Riquelme-Granada et al., 2015), a MOEA main goal it to both converge close to the real, yet unknown, PF and at the same time maintain a good diversity among the solutions on the current PF. Therefore, the research efforts focuses on developing various Quality Indicators (QIs) to evaluate the MOEA Convergence and Diversity. Convergence metrics are concerned with ensuring whether the non-dominated solutions in the obtained PF is close to the true optimal Pareto Front(PF_t) and whether it covers the whole extension of the PF_t. Notice that when the PF_t is unknown a reference set is considered instead. Diversity metrics are to indicate whether the obtained solutions are well spread and spaced among each others. In other words Diversity evaluates both the uniformity and spread. Uniformity and spread are two very closely related facets, yet they are not completely the same (Riquelme-Granada et al., 2015). Accordingly, the Quality Indicators can be classified based on the aspect that a QI measures as follows:

- Convergence metrics: Indicates how distant an approximation set from the true Pareto optimal front (e.g. Generational distance (GD) (Van Veldhuizen, 1999), Dominance Ranking (Knowles et al., 2006)).
- Cardinality metrics: The number of solutions that exists in the obtained Pareto Front. Intuitively, a larger number of solutions is preferred (e.g. Generational Nondominated Vector Generation (GNVG) (Van Veldhuizen, 1999)).
- Uniformity metrics: The distribution, refers to the relative distance among solutions in Pareto Front(e.g. Spacing (Schott, 1995)).
- Spread metrics: Also known as the extent, refers to the range of values covered by the solutions (e.g. Overall Pareto Spread (Azarm and Wu, 2001)).

In this study a single QI was chosen from each of the previously illustrated categories based on a criterion presented in section 5 of this paper. Also the criteria used for choosing which MOEA to be implement in the proposed solution is presented in the next section.

4 MULTI-OBJECTIVE EVOLUTIONARY TREE MINER

ProM framework (Verbeek et al., 2010) is the *de facto* standard process mining platform in the academic world. In fact most of the previously mentioned ABPD techniques were implemented as plug-ins for the ProM framework. The work in (Buijs et al., 2012a) has been implemented as a plug-in for the ProM framework, namely the Evolutionary Tree Miner (ETM). Also, the work in (Buijs et al., 2013) has been implemented as the ETM with the Pareto-Front extension (ETM-Preto used in the following). Since the ETM is an extensible evolutionary process discovery algorithm it was decided to extend the ETM in the solution proposed in this paper. Not only this will save a considerable amount of time and resources, but also it enables conducting a more objective comparison when ones considers comparing the performance of the solution proposed in this paper to the previously proposed solutions such as the one in (Buijs et al., 2013). Initially, it was decided to choose the MOEA to be implemented based on the simple criteria:

- The algorithm must support elitism.
- The algorithm should be a parameter-less GA, i.e. there is no need to specify any parameters (e.g. niching operators).
- The algorithm is simple and straightforward.
- Relatively fast, with an overall complexity of $O(MN^2)$ or better.
- Converges steadily and fast towards Pareto front.

However, since NSGA-II meets all of the requirements above, and since the solution proposed in (Buijs et al., 2013) is using a fitness calculation inspired by the crowding distance used in the NSGA-II. Again, in order to include greater objectivity in

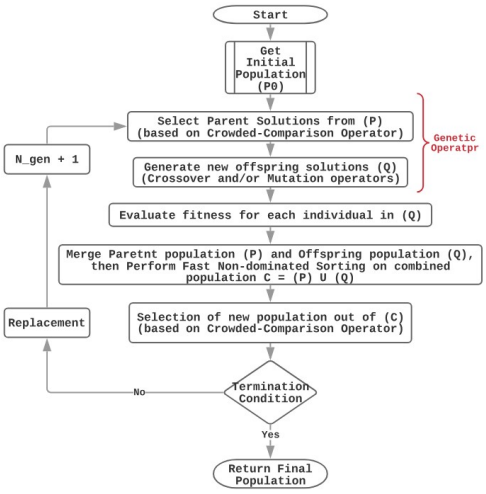


Figure 2: MOETM / NSGA-II based ABPD.

the correlation, the decision was made to implement NSGA-II.

The ABPD technique proposed in this paper has been implemented as a ProM plug-in namely Multi-Objective Evolutionary Tree Miner (MOETM). Figures 2 and 3 depicts how both algorithms MOETM, and ETM-Pareto works respectively. For a detailed description of how the ETM-Pareto works, please review (Buijs et al., 2013) (section 3). The MOETM implements the NSGA-II originally proposed in (Kalyanmoy et al., 2002) with a slight difference in the first step of the algorithm *Generating Initial Population*. Evaluating MOETM initial results, it was observed that duplicates are negatively affecting the performance of the algorithm in a way that cannot be overlooked. In order to reduce the impact of duplicates from the beginning and to improve diversity in the initial population the procedure *Get Initial Population*, depicted in Figure 4, was introduced. In NSGA-II there are two kinds of populations P_t and Q_t. Assuming that population size is referred to as N and Generation number as N_{Gen}. Originally in NSGA-II creating the initial population is as follows: A random parent population P_t of size N is created. The population is sorted based on the non-domination then each solution is assigned a fitness (or rank) equal to its non-domination level (1 is the best level, 2 is the next-best level, and so on). Thus, minimization of fitness is assumed. At first, the usual binary tournament selection, recombination, and mutation operators are used to create an offspring population Q_t size N. While Generating Initial Population in MOETM is as follows: When $N_Gen = 0$, initialize both initial P and initial Q randomly using a duplicate eliminator procedure. Combine both P and P in C and calculate the different quality dimensions for each candidate in C. Perform Fast-non-dominated sorting, and if necessary crowding distance selection on C to get P_t of size N. Rest of the Procedures in MOETM are identical to the original NSGA-II. Detailed description for rest of procedures in NSGA-II can be found in (Kalyanmoy et al., 2002) (section 3).

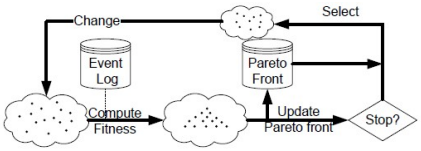


Figure 3: The different phases of the ETM-Pareto genetic algorithm (Buijs et al., 2013).

Both ETM-Pareto and MOETM envision the same approach to solve the problem of process model discovery, both algorithms deal with ABPD as a MOP and are trying to obtain an approximation set as close as possible to the (PF_t). However, there are some fundamental differences in how each algorithm works. Especially, in the way the PF is constructed and maintained internally in each algorithm. In ETM-Pareto, the PF represents an autonomous construct somehow independent from the current population in every generation. It was expected that this implementation with the unlimited size PF, which is unbounded by any limits such as the size of the population, will have its impact on the performance of the ETM-Pareto. While this unlimited size Pareto-Front allows for the opportunity to guarantee inclusion of optimal solutions in each generation. However, the simulation results and analysis in the next section will demonstrate that this approach has a negative impact on the overall performance of the ETM-Pareto algorithm as expected.

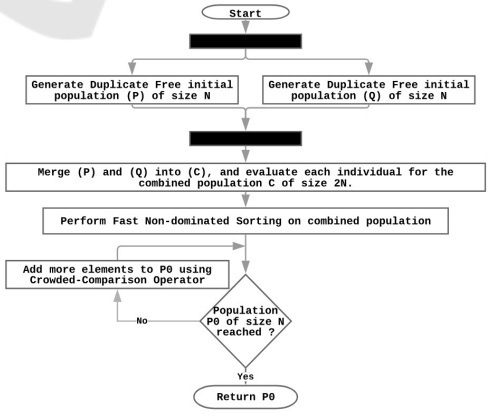


Figure 4: MOETM Get-Initial-Population Procedure.

5 SIMULATION AND RESULTS

In the following subsections, we first describe the environment in which the experiment was conducted, the used data set, the experiment parameters values, and the platform. The quality indicators used to assess the performance of the two algorithms are provided in the next subsection, followed by a detailed description of the comparative results.

5.1 Testing Environment

The data set is a real-life event log from Volvo IT Belgium which was available for the Third International Business Process Intelligence Challenge (BPIC'13) (Verbeek, 2016). The log contains events from an incident and problem management system called VINST. More information about the data set as well as documents detailing the data set is available in (Verbeek, 2016). The third log file (The problem management log-closed problems) was chosen as a start to test the proposed technique. The event log contains 1487 traces and 6660 events in total.Since the study focuses on the benefits of using NSGA-II in ABPD, and since this is the first time a well tested and proven MOEA is applied in ABPD therefore ETM-Pareto results will be used as the control group. All parameters in both groups were kept the same such as the style and the rate of crossover and mutation. The probability of performing crossover and mutation was 0.1 and 0.5 respectively. Both algorithms ran for 50 times. A maximum generations number of 300 and a total population size of 100 were used. Any differences in the settings were mainly due to the different nature of the two algorithms. For instance, ETM-Pareto was assigned an elite count of size 20. On the other hand the MOETM does not require any parameters such as elite size since elitism in NSGA-II works differently. Finally, for the selection strategy the MOETM used the tournament selection while ETM-Pareto was left with its default settings using Sigma Scaling. The experiment was performed on a machine running Windows 10 64-bit with 4 cores Intel Core i7-4710HQ Processor running at 2.50 Ghz and 8 GB memory, of which maximum of 5 GB was used by the algorithms.

5.2 Quality Indicators

As discussed earlier, in MOO two aspects should be considered and both should be measured simultaneously. First, to what extent the obtained solutions converge to the PF_t. Secondly, to what extent these solutions are distributed. In Section 3 of this paper it was discussed how the most prominent QIs can be categorized where a grouping as in (Riquelme-Granada et al., 2015; Li et al., 2014) was followed. Due to the large number of QIs available, also these metrics ranges from straightforward and easy to compute to the not so easy and computationally intensive. The following criteria were developed to choose a single QI from each category:

- Reliability, the QI has been used frequently in recent case studies, and its usage reflects an unfaltering quality.
- The QI should be straightforward and computationally inexpensive.
- The QI does not require prior knowledge of the PF_t.
- The QI does not use reference sets or reference points.

Based on the criteria defined above, the following QIs were being chosen to evaluate the performance.

5.2.1 QI for Evaluating Convergence / Outperformance

Generally, there are various types of accuracy QIs depending on whether the PF_t is known or not. When the PF_t for a given problem is known accuracy QIs usually focuses on quantifying the rate of how many real PF_t solutions exist among all non-dominated solutions [1] returned by the MOEA, or the distance how far the non-dominated solutions are from the PF_t. When the PF_t is unknown accuracy QIs usually uses a reference point or a reference set instead. Since the PF_t for a process model discovery problem is unknown and since it was decided not to use any QI that depend on a reference point the *Dominance Ranking* introduced in (Knowles et al., 2006) is used. Dominance Ranking compares the quality of PFs generated by two or more MOEAs. It is a binary or even arbitrary QI (i.e., it takes as an input two or more PF results of two or more MOEAs). Dominance Ranking has the following definition. Suppose $q \geq 2$ is the number of MOEAs to be compared. For each MOEA $i \in \{1,\ldots,q\}$, a number of runs $r_i \geq 1$ are performed, generating approximation sets $A_1^1, A_2^1, \ldots, A_{r^1}^1, \ldots, A_1^q, \ldots, A_{r_q}^q$, and C is the combined collection of all approximation sets. Each approximation set in C is assigned a rank, on the basis of dominance relations listed in Table 2 in (Knowles et al., 2006), by counting the number of sets by which a specific approximation set is dominated.

$$rank(C_i) = 1 + |\{C_j \in \mathbf{C} : C_j \lhd C_i\}|. \quad (1)$$

The lower the rank, the better the corresponding approximation set with respect to the entire collection.

[1] Solutions returned by a stochastic multiobjective optimizer are known as the PF, approximation set, or non-dominated set (NDS).

5.2.2 Cardinality QI

For an approximation set A as a result of multi-objective optimizer, the cardinality of A refers to the number of solutions that exists in A. Intuitively, a larger number of solutions is preferred.

Generational Nondominated Vector Generation (GNVG) is a simple metric that tracks the number of non-dominated vectors produced each MOEA generation. GNVG was introduced in (Van Veldhuizen, 1999) and is defined in equation 2.

$$GNVG \triangleq |PF_{current}(t)| \tag{2}$$

5.2.3 Spread QI

The primary goal for a multi-objective optimizer is to provide the decision maker with a large enough but limited number of solutions. Also it is highly desirable that this limited number of solutions are uniformly spread over the whole PF and are as diverse as possible. The QIs under Pareto spread are concerned with the range of objective function values. An approximation set that spreads over a wider range of the objective function values provides the designer with broader optimized design choices. In (Azarm and Wu, 2001) the researchers introduced a spread metric that quantifies how widely the obtained approximation set spreads over the objective space when the objective functions are considered altogether. The Overall Pareto Spread (OS) is defined as the volume ratio of two hyper-rectangles. One of these rectangles is HR_{gb} that is defined by the good and bad points with respect to each design objective. Similarly, the extreme points for an observed Pareto solution set defines the other hyper-rectangle that is denoted by HR_{ex}. The overall Pareto spread is defined as the ratio of the area or volume of HR_{ex} to that of HR_{gb}:

$$OS(P) = \frac{HR_{ex}(P)}{HR_{gb}} \tag{3}$$

where P refers to an observed approximation set. By using the objective values to interpret $HR_{ex}(P)$ and HR_{gb}, equation 3 can be expressed as:

$$\begin{aligned} OS(P) &= \frac{\prod_{i=1}^{m} |max_{k=1}^{\overline{np}}(P_k)_i - min_{k=1}^{\overline{np}}(P_k)_i}{\prod_{i=1}^{m} |(P_b)_i - (P_g)_i|} \\ &= \prod_{i=1}^{m} |max_{k=1}^{\overline{np}}[\overline{f_i}(x_k)] - min_{k=1}^{\overline{np}}[\overline{f_i}(x_k)]| \end{aligned} \tag{4}$$

5.2.4 Uniformity QI

A Spread QI alone will not be able to fully characterize the diversity of solutions in a given approximation set. If the solutions in a given approximation set are all very similar, these solutions will not be able to reflect the trade-offs between the different objectives. Uniformity QI measure the evenness of distribution of solutions across the PF. A measure of uniform distribution *(UD)* to measure the distribution of non-dominated individuals was proposed in (Tan et al., 2002) . Mathematically, $UD(X')$ for a given set of non-dominated individuals X' in a population X, where $X' \subseteq X$, is defined as

$$UD(X') = \frac{1}{1+S_{nc}} \tag{5}$$

where S_{nc} is the standard deviation of niche count of the overall set of non-dominated individuals formulated as,

$$S_{nc} = \sqrt{\frac{\sum_{i}^{N_{x'}} \left(nc(x'_i) - \overline{nc}(X') \right)}{N_{x'} - 1}} \tag{6}$$

where $N_{x'}$ is the size of the set X', $nc(x'_i)$ is the niche count of i^{th} individual x'_i where $x'_i \in X'$, and $\overline{nc}(X')$ is the mean value of $nc(x'_i)$, $\forall\ i = 1, 2, \ldots, N_{x'}$ as shown in the following equations:

$$nc(x'_i) = \sum_{j, j \neq i}^{N_{x'}} f(i, j), \tag{7a}$$

$$where\ f(i,j) = \begin{cases} 1, & dis(i,j) < \sigma_{share} \\ 0, & \text{else} \end{cases} \tag{7b}$$

$$\overline{nc}(X') = \frac{\sum_{i}^{N_{x'}} nc(x'_i)}{N_{x'}} \tag{8}$$

where $dis(i, j)$ the distance between individual i and j in the objective domain. In the paper, σ was set to *0.01*. Generally speaking, the larger the spread the better it is, as it means a better distribution.

5.3 Result Analysis

A real-life event log was used to test and verify the validity of the newly proposed MOEA based ABPD technique. The ETM-Pareto acts as the control group and all settings are the same. Four Quality Indicators of four different types are used to evaluate the validity of the proposed technique more comprehensively.

Table 2 demonstrates a sample of the obtained result in a randomly chosen five runs of both engines (10, 20, 30, 40, and 50). In Table 2 the size of the final approximation set *PF-size* and the corresponding UD, OS values obtained by MOETM and ETM-Pareto are provided. Along with the first process model (represented by the scores that model obtains on each quality dimension) to exist in that final approximation set. Interestingly, in all the cases, the MOETM attains a considerable better UD and OS. Even with its *PF-size* far less than of the control group. Which indicates that all approximation sets obtained by the MOETM

Table 2: A sample of results in a randomly chosen five runs (Run-ID) 10, 20, 30, 40, and 50.

Run-ID	Engine-Name	UD	OS	PF-size	Sample candidate			
					Replay-Fitness	Precision	Simplicity	Generalization
	MOETM	0.741	0.547	100	1	0.801	0.739	0.630
10	ETM-Pareto	0.313	0.339	194	0.844	0.992	0.916	0.881
	MOETM	0.730	0.8991	100	1	0.850	0.75	0.683
20	ETM-Pareto	0.367	0.0718	116	0.827	0.998	0.9	0.869
	MOETM	0.713	0.515	100	1	0.754	0.972	0.906
30	ETM-Pareto	0.426	0.264	196	0.733	0.999	0.666	0.647
	MOETM	0.753	0.481	100	1	0.728	0.928	0.817
40	ETM-Pareto	0.455	0.174	174	0.988	0.886	0.933	0.900
	MOETM	0.709	0.367	100	1	0.822	0.937	0.853
50	ETM-Pareto	0.383	0.079	356	0.800	0.998	0.888	0.847

has a much better distribution and spread over the *PF*. Furthermore, and regarding the quality of the process models obtained by both engines, as figure 5 illustrates it is observable that the models obtained by the MOETM scores best on the Replay-Fitness(FR) followed by Precision(PE) then and at the same time has an acceptable and a more diverse scores on the other two quality dimensions Simplicity(SU) and Generalization(GV). More analysis results are available in Appendix A.

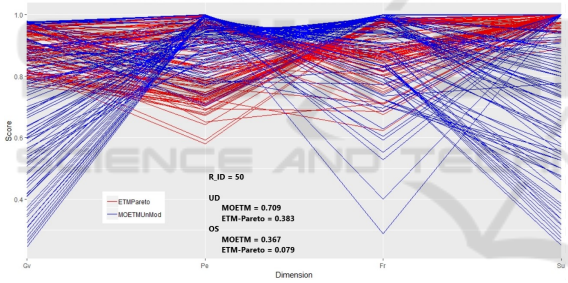


Figure 5: Process models scores on Four quality Dimensions FR, SU, GV, and PE for models obtained by both engines MOETM(Blue) and ETM-Pareto(Red) in the last run (R-ID = 50) along with UD and OS scores of each engine.

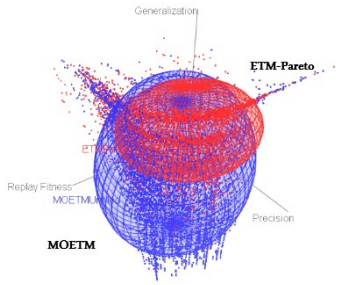


Figure 6: All Approximation Sets obtained by both MOETM(Blue) and ETM-Pareto(Red) on three dimensions Replay Fitness, Precision, and Generalization.

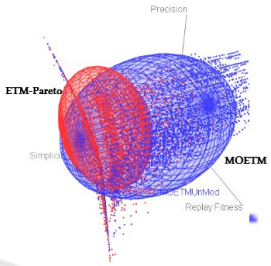


Figure 7: All Approximation Sets obtained by both MOETM(Blue) and ETM-Pareto(Red) on three dimensions Replay-Fitness, Precision, and Simplicity.

In (Knowles et al., 2006) it is recommended that the Dominance Ranking is to be used as the first QI. Since if a significance difference can be demonstrated using the ranking of approximation sets alone, there will be no need to use other QIs to conclude which of the MOEAs generates the better sets. In our case the Dominance Ranking QI did not reveal much difference in the ranking of approximation sets probably due to the large size of the final approximation sets (100 in case of MOETM and more than 100 in for ETM-Pareto) along with a relatively low number of maximum generations (300). A maximum generation number of 300 may be considered to be low for such kind of experiment especially with the given settings. Originally, it was decided that the experiment will have a maximum generation number of 2000. However, the number of generations had to be reduced due to the inability of ETM-Pareto to finish execution successfully. Even with a maximum generation number of 500, ETM-Pareto was unable to successfully finish execution for three times. This problem is an inherent problem for the ETM-Pareto, and its main reason is the unlimited size of the *PF* the ETM-Pareto maintains internally. While this implementation guarantees that the *PF* keeps all *elite* non-dominated candidates found in all previous generations. However, Always the moment will come when the size of the maintained *PF* exceeds the upper lim-

its of the available computational resources. When that moment comes, the engine will be forced to stop running. This premature end of execution is depriving the algorithm the chance of exploring more candidates which could be more *optimal* and can replace some or maybe many of the existing *elite* candidates existing in the *PF*. No matter how much computational resource are available (e.g., RAM size available and allocated), with this implementation it will never be guaranteed to avoid this problem. And due to the stochastic nature of MOEAs its could happen in early or late generations. MOETM does not suffer such problem since the *PF* size is always trimmed and is guaranteed to be less than or equal to the size of the population.

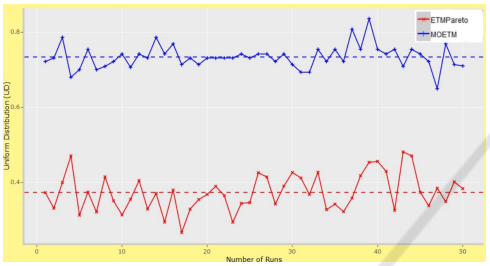


Figure 8: UD in 100 Approximation Sets for both MOETM and ETM-Pareto (50 Approximation sets each).

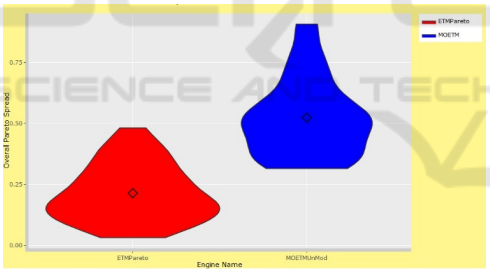


Figure 9: OS in 100 Approximation Sets for both MOETM and ETM-Pareto (50 Approximation sets each).

Since high-dimensional visualization and including more than three-dimensions at once is very difficult and sometimes incomprehensible. Two and three dimensional plotting will be used to visualize the *PF* and the improvements achieved in the quality of the obtained *PF* represented in a better spread and uniformity. Figure 7, and figure 6 shows a distribution of Approximation Sets (NDSs) obtained by both MOETM, and ETM-Pareto on three different process model quality dimensions. It is visually observable that in both figures the MOETM has a larger and an even spread of solutions. Figure 8 shows the UD of every *PF* obtained in each of the runs for both algorithms. The figure shows that in all of the 50 runs the MOETM has a better UD. The MOETM has a minimum UD of 0.65 while the maximum UD for ETM-Pareto was 0.48. Overall, the MOETM has an average UD of 0.733 while the control group has an average UD of 0.372. Similarly, figure 9 shows that the MOETM has minimum and maximum *OS* of 0.32 and 0.91 respectively while ETM-Pareto has a maximum *OS* of 0.48 and a minimum *OS* of 0.03. Overall, the MOETM has an average *OS* of 0.523 while the control group has an average *OS* of 0.214. Although the ETM-Pareto usually has a larger *PF* size than it of MOETM. However, the MOETM maintained a much better diversity in the obtained solutions in all runs.

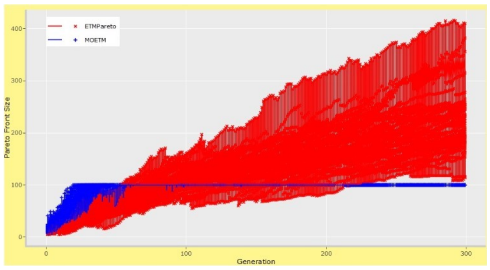


Figure 10: GNVG in 100 Approximation Sets for both MOETM and ETM-Pareto in all 300 Generations.

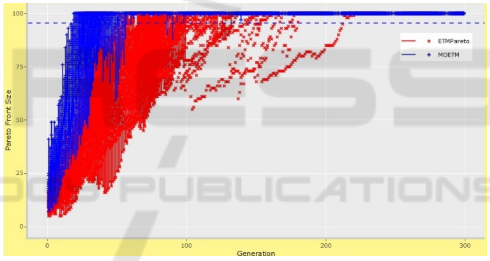


Figure 11: GNVG in 100 Approximation Sets for both MOETM and ETM-Pareto in all 300 Generations, imited to maximum Approximation Set size of 100.

From a point of view considering cardinality QIs, it is true that an optimizer which can obtain a larger *PF* size is preferred. And it is also true that the ETM-Pareto always returned a *PF* of a larger size than MOETM. However, we believe this measure is misleading in our case. First, early it was explained that the untrimmed *PF* of ETM-Pareto is the reason behind its larger *PF* size, also it was demonstrated earlier what side effects this has on the overall performance of the algorithm. Second, the GNVG QI indicates that the MOETM has a faster convergence and always obtained a larger Approximation set earlier than the control group. Figure 10 shows the GNVG for both algorithms in all 300 generations (all Approximation sets included). Figure 11 is basically a zoom-in version of figure 10 where the Y axis (the axis representing the Approximation set size) is limited to 100 (the maximum size of population). As

figure 11 shows it was observed that in all runs the MOETM was faster in getting a non-dominated set of solutions with larger size (e.g., the ETM-Pareto has never reached a non-dominated set of solutions with size of 100 before the 50th generation, while MOETM had this size in some runs before the 25th generation).

6 DISCUSSION, CONCLUSIONS AND FUTURE WORK

In this paper a Multi-Objective Optimization based Process Model Discovery technique is presented. The proposed solution has been implemented as a plug-in for the ProM framework named the Multi-Objective Evolutionary Tree Miner (MOETM). The MOETM is an extension to the infamous Evolutionary Tree Miner (ETM) with a new evolutionary engine based on the NSGAII. The proposed ABPD technique is able to obtain a Pareto Front of mutually non-dominating process models. The MOETM was tested on a real-life event log and the results were compared to a control group of the ETM-Pareto. In order to systematically and accurately compare the results of the two algorithms, four different Quality Indicators were implemented to assess the quality of both convergence and diversity in the final approximation set.

The results shows that the MOETM had a faster convergence toward the Pareto front. Results also demonstrated that the MOETM had a vastly improved distribution characteristic evident in the much better spread and uniformity of its obtained results. In comparison to the control group, the MOETM average Uniform Distribution was better by 97.04% , and it had a 144.39% increase in the average Overall Pareto Spread. This paper also points out the potential problem the original ETM may suffer. The ETM will more likely return sub-optimal or extremal solutions (Emmerich and Deutz, 2018) due to the use of the weighted sum method. The experiment shows that ETM-Pareto sometimes had improper termination due to its Pareto Front of unlimited size. It is clear that truncation of the Pareto Front and keeping its size under a certain limit is necessary in stochastic multi-objective optimizers to avoid running out of computational resources in late generations and to ensure proper termination. Moreover, and also from a stand point considering MOO, there exists plenty of MOEAs Frameworks available (e.g. Opt4J, MOEA, ECJ, and JMetal (Durillo and Nebro, 2011)) but there is a need for a MOO Framework with a non-invasive API like the Watchmaker Framework (Dyer, 2010) which currently supports only SOO. Such a framework with a non invasive API will allow researchers to put to the test different MOEAs to solve various problems in many application domains with more ease.

Future work will focus on investigating the use of other MOEAs in ABPD and the inclusion of other QIs to assess the results of MOEAs in the context of ABPD. Especially, convergence QIs as there is a need for a more accurate but at the same time less computationally expensive metrics to assess the convergence in MOEAs. Furthermore, MOEAs returns a Pareto Front as its final output. This Pareto Front is a set of mutually non-dominated optimal solutions. Since, this set is usually very large and the decision maker faces the problem of reducing the size of this set to a manageable number of solutions to analyze. There is a need for investigating an approach which can objectively reduces the non-dominated set of solutions obtained by a MOEA, or even better, guides the decision maker in his choice for the best solution according to the trade offs.

REFERENCES

Azarm, S. and Wu, J. (2001). Metrics for quality assessment of a multiobjective design optimization solution set. *ASME J. Mech. Des*, 123(1):18–25.

Bratosin, C., Sidorova, N., and van Der Aalst, W. M. (2010a). Discovering process models with genetic algorithms using sampling. *international conference on knowledge based and intelligent information and engineering systems*, 6276:41–50.

Bratosin, C., Sidorova, N., and van Der Aalst, W. M. (2010b). Distributed genetic process mining. In *IEEE Congress on Evolutionary Computation*, pages 1–8.

Buijs, J. (2014). *Flexible evolutionary algorithms for mining structured process models*. PhD thesis, Eindhoven university of technology.

Buijs, J. C., La Rosa, M., Reijers, H. A., van Dongen, B. F., and van der Aalst, W. M. (2012a). Improving business process models using observed behavior. In *International Symposium on Data-Driven Process Discovery and Analysis*, pages 44–59. Springer.

Buijs, J. C., van Dongen, B. F., and van der Aalst, W. M. (2012b). A genetic algorithm for discovering process trees. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE.

Buijs, J. C., Van Dongen, B. F., and van Der Aalst, W. M. (2012c). On the role of fitness, precision, generalization and simplicity in process discovery. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 305–322.

Buijs, J. C., van Dongen, B. F., and van der Aalst, W. M. (2013). Discovering and navigating a collection of process models using multiple quality dimensions. In

International Conference on Business Process Management, pages 3–14. Springer.

de Medeiros, A. K. A. (2006). *Genetic process mining*. PhD thesis, Eindhoven university of technology.

Durillo, J. J. and Nebro, A. J. (2011). jmetal: A java framework for multi-objective optimization. *Advances in Engineering Software*, 42(10):760–771.

Dyer, D. W. (2010). *Watchmaker framework for evolutionary computation version 0.7.1*.

Emmerich, M. T. and Deutz, A. H. (2018). A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural computing*, 17(3):585–609.

Gadhvi, B., Savsani, V., and Patel, V. (2016). Multi-objective optimization of vehicle passive suspension system using nsga-ii, spea2 and pesa-ii. *Procedia Technology*, 23:361–368.

Günther, C. W. and Van Der Aalst, W. M. (2007). Fuzzy mining–adaptive process simplification based on multi-perspective metrics. In *International conference on business process management*, pages 328–343. Springer.

Gupta, E. (2014). Process mining a comparative study. *International Journal of Advanced Research in Computer and Communications Engineering*, 3(11):5.

Ishibuchi, H., Imada, R., Setoguchi, Y., and Nojima, Y. (2017). Hypervolume subset selection for triangular and inverted triangular pareto fronts of three-objective problems. In *Proceedings of the 14th ACM/SIGEVO Conference on Foundations of Genetic Algorithms*, pages 95–110. ACM.

Jin, Y. (2012). *Advanced fuzzy systems design and applications*, volume 112. Physica.

Kalyanmoy, D., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on evolutionary computation*, 6(2):182–197.

Knowles, J. D. and Corne, D. W. (2000). Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary computation*, 8(2):149–172.

Knowles, J. D., Thiele, L., and Zitzler, E. (2006). A tutorial on the performance assessment of stochastic multiobjective optimizers. *TIK-Report*, 214.

Li, M., Yang, S., and Liu, X. (2014). Diversity comparison of pareto front approximations in many-objective optimization. *IEEE Transactions on Cybernetics*, 44(12):2568–2584.

Marler, R. T. and Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395.

Marler, R. T. and Arora, J. S. (2010). The weighted sum method for multi-objective optimization: new insights. *Structural and multidisciplinary optimization*, 41(6):853–862.

Riquelme-Granada, N., Von Lücken, C., and Baran, B. (2015). Performance metrics in multi-objective optimization. In *2015 Latin American Computing Conference (CLEI)*, pages 1–11. IEEE.

Schott, J. R. (1995). *Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization.* PhD thesis, AIR FORCE INST OF TECH WPAFB AFB OH.

Tamaki, H., Kita, H., and Kobayashi, S. (1996). Multi-objective optimization by genetic algorithms: A review. In *Proceedings of IEEE international conference on evolutionary computation*, pages 517–522.

Tan, K. C., Lee, T. H., and Khor, E. F. (2002). Evolutionary algorithms for multi-objective optimization: Performance assessments and comparisons. *Artificial intelligence review*, 17(4):251–290.

Tiwari, A., Turner, C. J., and Majeed, B. (2008). A review of business process mining: state-of-the-art and future trends. *Business Process Management Journal*, 14(1):5–22.

Tsai, C.-Y., Jen, H.-Y., and Chen, I.-C. (2010). Time-interval process model discovery and validation—a genetic process mining approach. *Applied Intelligence*, 33:54–66.

Van der Aalst, W. M., De Medeiros, A. A., and Weijters, A. (2005). Genetic process mining. In *International conference on application and theory of petri nets*, pages 48–69. Springer.

van der Aalst, W. M., Reijers, H. A., Weijters, A. J., van Dongen, B. F., De Medeiros, A. A., Song, M., and Verbeek, H. (2007). Business process mining: An industrial application. *Information Systems*, 32(5):713–732.

van der Aalst, W. M. P. (2011). *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer, 1st edition.

van der Aalst, W. M. P., Hee, K., Ter, A., Sidorova, N., M. W. Verbeek, H., Voorhoeve, M., and Wynn, M. (2011). Soundness of workflow nets: Classification, decidability, and analysis. *Formal Aspects of Computing*, 23:333–363.

van der Aalst, W. M. P., Weijters, A., and Măruşter, L. (2004). Workflow mining: Discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142.

Van Veldhuizen, D. A. (1999). *Multiobjective evolutionary algorithms: classifications, analyses, and new innovations*. PhD thesis, AIR FORCE INST OF TECH WPAFB OH SCHOOL OF ENGINEERING.

Verbeek, H. (2016). 9th international workshop on business process intelligence. Last accessed 25 January 2017.

Verbeek, H., Buijs, J. C., Van Dongen, B. F., and Van Der Aalst, W. M. (2010). Xes, xesame, and prom 6. In *International Conference on Advanced Information Systems Engineering*, pages 60–75. Springer.

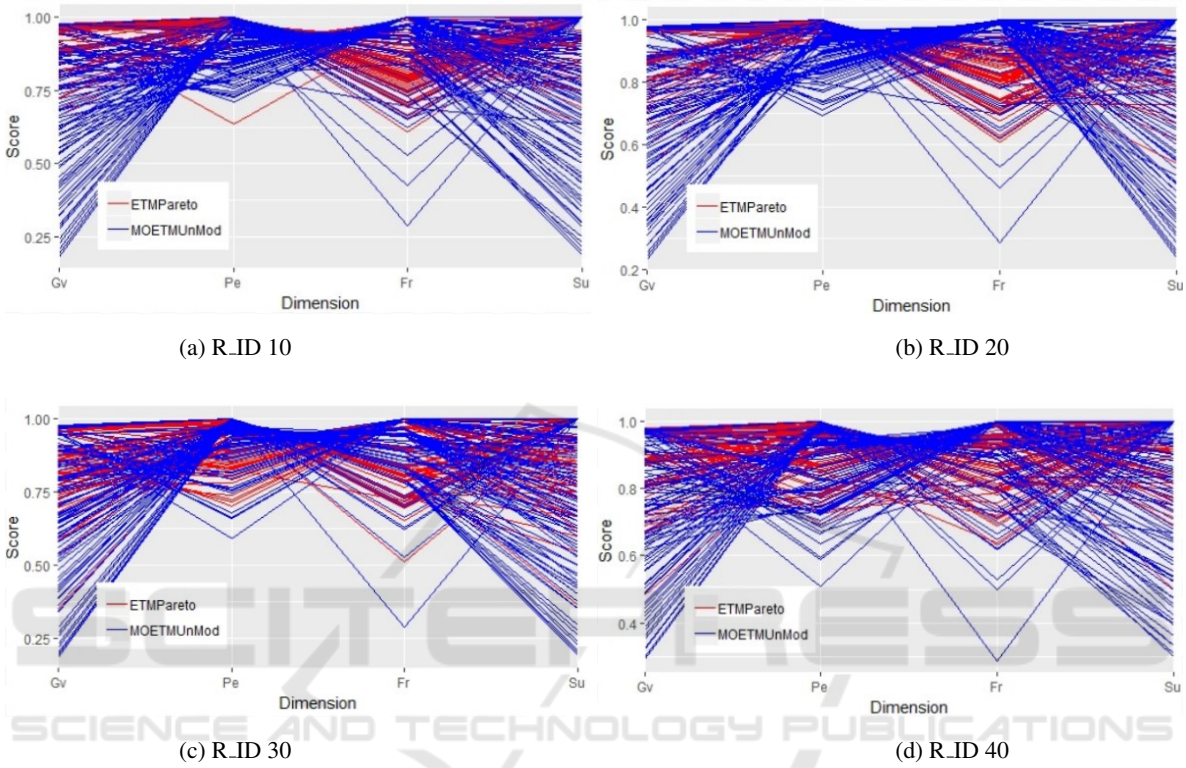
Weijters, A., van Der Aalst, W. M., and De Medeiros, A. A. (2006). Process mining with the heuristics miner-algorithm. *Technische Universiteit Eindhoven, Tech. Rep. WP*, 166:1–34.

Zitzler, E., Laumanns, M., and Thiele, L. (2000). Improving the strength pareto evolutionary algorithm. *EUROGEN 2001, Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pages 95–100.

APPENDIX

A: More Analysis Results

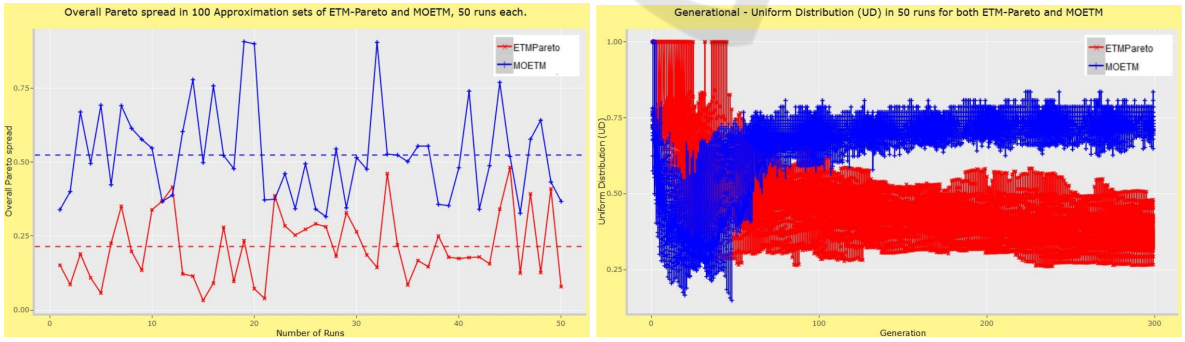
More on Quality Indicators



(a) R_ID 10 (b) R_ID 20

(c) R_ID 30 (d) R_ID 40

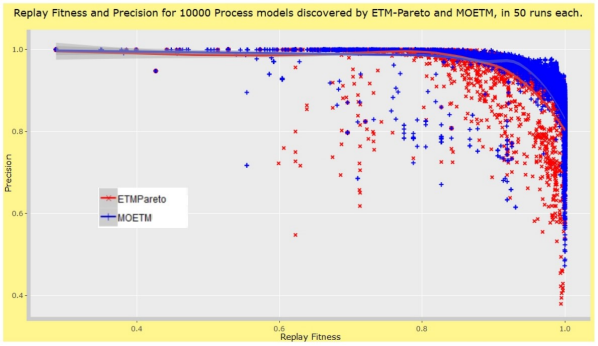
Figure 12: Process models scores on Four quality Dimensions Replay-Fitness(Fr), Simplicity(SU), Generalization(Gv), and Precision(Pe). The models are obtained by both engines MOETM(Blue), and ETM-Pareto(Red) in runs (R_ID) 10, 20, 30, 40.



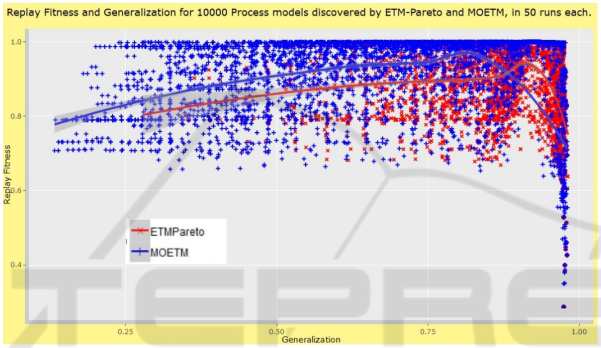
(a) OS. (b) UD.

Figure 13: (a) is a self-explanatory figure, (b) Generational-Uniform Distribution (UD) (i.e. UD is measured for approximation sets in every generation not just in the final obtained approximation set) for both engines MOETM(Blue) and ETM-Pareto(Red) in 100 runs.

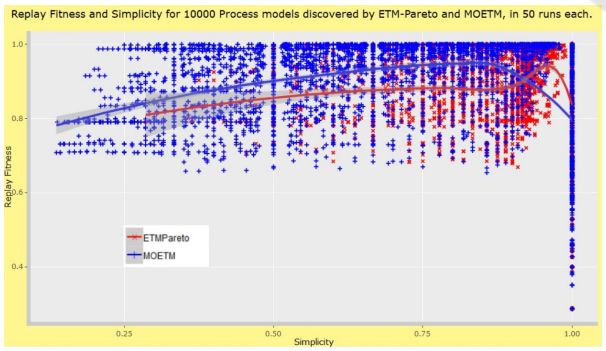
Process Models Scores on Different Quality Dimensions and the Pareto Front Shape



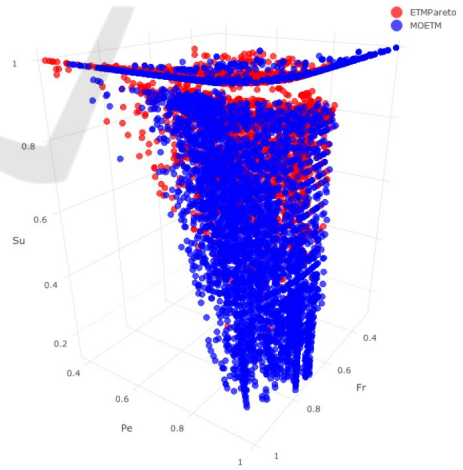
(a) Replay-Fitness and Precision.



(b) Generalization and Replay-Fitness.



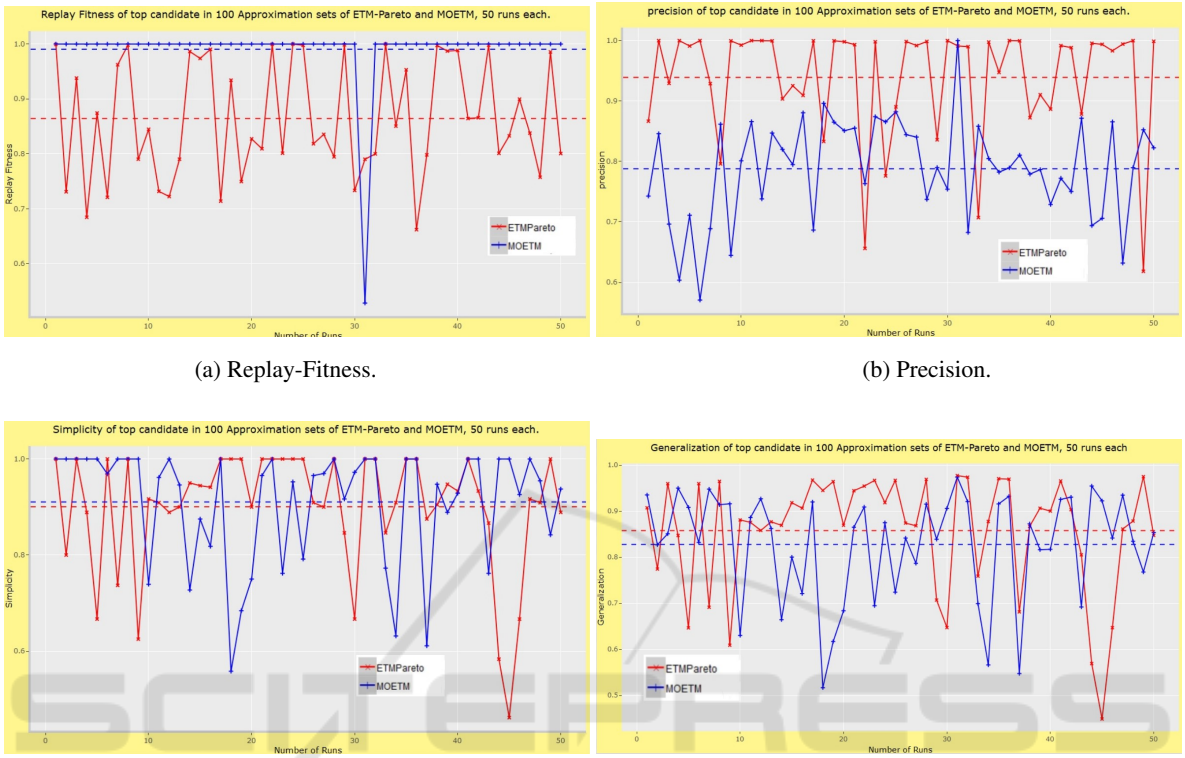
(c) Simplicity and Replay-Fitness.



(d) Replay-Fitness, Precision, and Simplicity.

Figure 14: Process models scores on different quality Dimensions Replay-Fitness, Simplicity, Generalization, and Precision for models obtained by both engines MOETM(Blue) and ETM-Pareto(Red) in 100 runs along with the pareto front shape.

Process Models Scores on Four Quality Dimensions in All 100 Approximation Sets Obtained



(a) Replay-Fitness.

(b) Precision.

(c) Simplicity.

(d) Generalization.

Figure 15: Process models scores on Four quality Dimensions Replay-Fitness(Fr), Simplicity(SU), Generalization(Gv), and Precision(Pe) of the first candidate (only the first process model happens to exist in the final result) in the 100 Approximation sets obtained by MOETM and ETM-Pareto.

B: The Weighted Sum Method for MOO

A general Multi-Objective Optimization Problem can be formulated mathematically as follows:

$$\begin{aligned} \underset{x}{\text{Minimize}}\ & \mathbf{F}(x) = [f_1(x), f_2(x), \ldots, f_m(x)]^T \\ \text{s.t}\quad & x \in S \end{aligned} \tag{9}$$

where m [2]is the number of scalar objective functions and x is the decision vector with a domain of definition $S \subseteq \mathbb{R}^n$, where n is the number of independent variables x_i, while $Z \subseteq \mathbb{R}^m$ refers to the objective space and is the forward image of S under the mapping $\mathbf{F} : S \to \mathbb{R}^m$.

Scalarization techniques are some classical methods often used in solving MOPs. Scalarizing a MOP is an a priori method[3]. Briefly, scalarization means that the objective functions are aggregated or reformulated as constraints then a SOP is solved. By using different parameters of the constraints and aggregation function it is possible to obtain different points on the Pareto front. Various scalarization approaches exists most notably the weighted sum method. Despite its well-known drawbacks with respect to depicting the Pareto optimal set the weighted sum method continues to be used extensively not only to provide multiple solution points by varying the weights consistently, but also to provide a single solution point that reflects preferences presumably incorporated in the selection of a single set of weights.

[2]When the number of objectives, m, is more than 3 then the problem defined by 9 is sometimes also referred to as many-objective among the evolutionary multi-objective optimization community.

[3]A priori here stand for "a priori articulation of preferences" which refers to a set of methods for solving a MOP. More information on different multi-objective optimization methods, and other scalarization techniques can be found in (Emmerich and Deutz, 2018; Marler and Arora, 2004).

The Weighted Sum Method

One of the simplest methods for solving a MOP is to scalarize the problem using a scalarization function. Many implementations of scalarization functions exist such as the weighted sum, the Chebyshev scalarization functions, and ε-constraint method. The weighted sum method is the most common general scalarization method. Since it is easy to linear weighting by simply attaching non-negative weights to each objective function and then optimize a weighted sum of the objective functions using any method for SOO. In such case the MOP is reformulated to:

$$\text{Minimize} \sum_{i=1}^{m} w_i f_i(x), \quad x \in S. \tag{10}$$

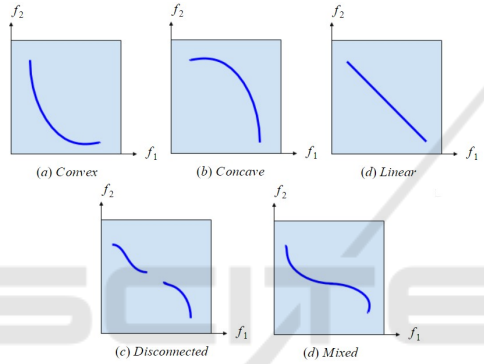


Figure 16: Pareto fronts with different shapes.

In MOO the shape of the Pareto front is an important property that affects the optimization process. Since the shape of the Pareto front influences the effectiveness of multi-objective optimizers which rely on linear scalarization functions. Generally speaking, figure 16 illustrates five types of Pareto fronts that can be distinguished depending on their shape: Convex, Concave, Linear, Disconnected shape, and a Pareto fronts which contain combinations of the former shapes. Formal definitions for the notion of convexity and concavity of Pareto fronts can be found in (Jin, 2012; Emmerich and Deutz, 2018).

In case of a convex Pareto front, possibly all points on the Pareto front can be obtained by the weighted sum method. However, if the Pareto front is non-convex there are points on the Pareto front which the weighted sum method can not generate. Generally, in the case of concave Pareto fronts the weighted sum method will tend to give only extremal solutions, that is solutions that are optimal in one of the objectives. Figure 17 is an illustration of the non-convex Pareto front case, the weighted sum method is unable to obtain solutions in the middle part of the Pareto front. This well-known drawback of the weighted sum method along with other drawbacks are discussed in a number of studies (Marler and Arora, 2004; Emmerich and Deutz, 2018).

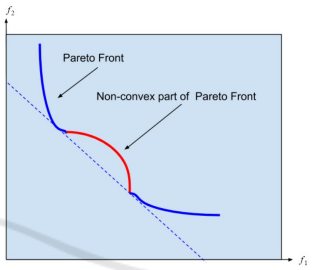


Figure 17: Weighted Sum Method unable to find non-dominated points in non-convex regions of the Pareto front.