On the Performance Overhead of BPMN Modeling Practices

Ana Ivanchikj, Vincenzo Ferme, and Cesare Pautasso

Software Institute, Faculty of Informatics, USI Lugano, Switzerland

Abstract. Business process models can serve different purposes, from discussion and analysis among stakeholders, to simulation and execution. While work has been done on deriving modeling guidelines to improve understandability, it remains to be determined how different modeling practices impact the execution of the models. In this paper we observe how semantically equivalent, but syntactically different, models behave in order to assess the performance impact of different modeling practices. To do so, we propose a methodology for systematically deriving semantically equivalent models by applying a set of model transformation rules and for precisely measuring their execution performance. We apply the methodology on three scenarios to systematically explore the performance variability of 16 different versions of parallel, exclusive, and inclusive control flows. Our experiments with two open-source business process management systems measure the execution duration of each model's instances. The results reveal statistically different execution performance when applying different modeling practices without total ordering of performance ranks.

Keywords: BPMN 2.0 · Execution Performance · Semantic Equivalence

1 Introduction

As customer retention becomes strongly related to service execution time, velocity requirements have gone down from days to hours, minutes and seconds. This is especially true in fully automated Business Processes (BPs), where any additional millisecond of performance boost brings companies a competitive advantage and potential cost savings on the Cloud [2]. Assuming that model's execution semantics has already been optimized, could such boost be achieved by using what La Rosa et al. [25] call the "Alternative Representation Pattern", i.e., modeling the same execution semantics with different static structures? For instance, parallelism in BPMN 2.0 can be modeled explicitly by using a parallel gateway, or implicitly through multiple outgoing flows from an activity [23]. Although the modeling practices used in two such models are different in terms of the model's graph topology (size and used constructs), their execution semantics is the same, i.e., they both depict parallelism. As modelers freely pick which modeling practice to use, their choice should result in the same execution performance. Does this expectation hold in practice? Does the answer depend on

the Business Process Management Systems (BPMSs)? These are open questions which have not received the due attention so far.

The first research question (RQ_1) we address in this work is: *Does the application of different modeling practices have significant impact on the duration of a BP instance execution?* Our null hypothesis (H_{RQ1}) is that *there is no statistically significant difference in the execution duration between instances of models which are semantically equivalent but structurally different.* We use trace equivalence [1] to define semantically equivalent models, i.e., models which produce the same traces (execution logs), regardless of the differences in their static structure (the control-flow constructs used). The data flow of the models remains unaltered. The second research question (RQ_2) we address is: *If H_{RQ1} is rejected, is there a total order between semantically equivalent but structurally different models, when ranked according to their performance?*

By answering RQ_1, BPMS vendors can decide whether there is a potential for performance improvement of their products based on alternative representations of deployed BPs. For instance, if an implicit parallel gateway executes significantly faster than an explicit one, the vendor can use the same implementation for both. The answer to RQ_2, on the other hand, indicates potential generalization opportunities of any identified optimization rules. For instance, it can show whether the execution of implicit gateways always ranks better than the execution of explicit gateways, regardless of the gateway type (e.g., parallel, inclusive, exclusive). Answering both questions is required before investing in further research towards automatic performance optimization by semantics preserving model transformations.

To this end, we propose a methodology for transforming an initial model into semantically equivalent models by using a predefined set of transformation rules. We also propose a statistical procedure to analyze the results of executing the equivalent models in order to answer the two research questions. To delimit the exploration space for this paper we have selected three scenarios which follow some of the modeling guidelines defined by Mendling et al. [21] and deal with the frequently used parallel and exclusive gateways [22], as well as with the inclusive gateway which has been found to hinder BP understandability [21]. We run the initial models as well as the derived semantically equivalent models on two open source BPMSs, Camunda and Activiti. The contribution of this paper consists of the methodology, the experimental results, and the analysis, which indicate that semantically equivalent models with different structure demonstrate statistically different execution behaviour, thus justifying further research into automated BP execution performance improvement.

The rest of this paper is organized as follows. In Sect. 2 we propose a methodology for defining the experiments needed for assessing the performance differences. We apply the methodology on three scenarios and discuss the results in Sect. 3, followed by a short survey of the related work in Sect. 4. We elaborate on the threats to the validity of our work in Sect. 5, while concluding the paper in Sect. 6.

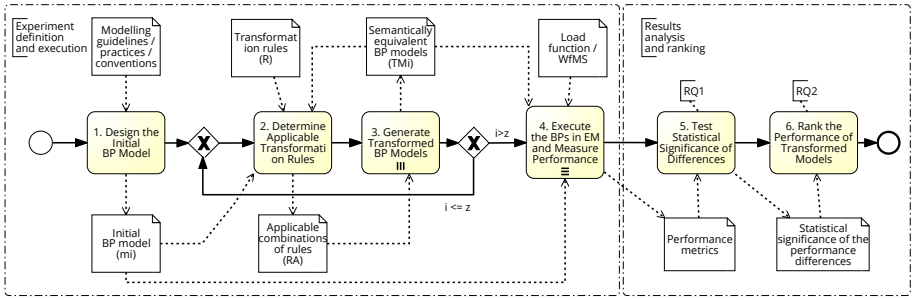

Fig. 1. Methodology for systematic exploration of the performance of structurally different but semantically equivalent models

2 Methodology

The methodology (Fig. 1) is divided into two parts: 1) define and perform experiments, which as such can be applied to obtain metrics needed for different research questions; and 2) statistically analyze the experiment results to test the H_{RQ1} hypothesis and rank the performance of the models to answer RQ_2.

1. **Initial Model**. The experiment definition process starts with the design of an initial model (m_i) (Fig. 1.1). Such initial model could follow existing best practices and conventions, or modeler's personal preferences.

2. **Applicable Rules**. In addition to the initial model, a set of transformation rules $R = \{r_1, r_2, r_3, ..., r_n\}$ are used as input to the second step of the methodology (Fig. 1.2). A transformation rule is an operation which adds/deletes elements (nodes, edges) to/from the BP model, provided that certain preconditions are met. In the context of this work, a transformation rule only affects the structure of the BP graph, leaving the execution semantics and the graph layout unchanged. The set of transformation rules can reflect different modeling practices and guidelines. Given R, all feasible combinations of the transformation rules should be iteratively applied starting from the initial model to generate semantically equivalent models until a fixed-point is reached. The output of the second step is $R_A(m_i) \subseteq R$, containing only the rules that are applicable to a given BP model (m_i in the first iteration), to produce a correctly deployable and executable, semantically-equivalent model.

3. **Model Transformation**. The model transformation function $f(r_j, m)$ is executed applying the transformation rule r_j to model m. Applying all rules found in $R_A(m_i)$ results in $TM_1 = \{t_j | (\forall r_j \in R_A(m_i))[t_j = f(r_j, m_i)]\}$, the set of transformed models which are semantically, but not syntactically, equivalent to the initial model m_i and obtained by applying just once all the applicable transformation rules. The generation process (Fig. 1.2 and Fig. 1.3) is repeated for all elements of TM_1, generating a new set of equivalent models $TM_2 = \{t_{jk} | (\forall t_j \in TM_1)(\forall r_k \in R_A(t_j))[t_{jk} = f(r_k, t_j)]\}$, by transforming all the models $t_j \in TM_1$ with the corresponding applicable rules $R_A(t_j)$. In other words, starting from the initial model m_i, only one rule is applied to generate the models in TM_1, two rules to generate TM_2, etc. This iteration stops when

no new models are generated or after a given maximum number z of model generations. The output of the iterations is $EM = \{m_i\} \cup \bigcup_{i=1}^{z} TM_i$, the set of semantically equivalent models to be executed in the experiment.

4. **Model Execution**. Finally each of the models in EM is deployed and multiple instances are started always using the same load functions and test data. During the execution of the BP instances, performance data is collected and different metrics (M) are calculated (Fig. 1.4). Depending on the optimization goal, metrics may include time (e.g., duration of the BP instance/constructs, or throughput), or resource utilization (e.g., CPU / RAM).

5. **Statistical Significance**. Determining statistical significance (Fig. 1.5) requires statistical tests, the nature of which depends on different factors [19]. The first factor to consider is the number and the nature of the samples. In our case, one sample is comprised of the instances of a given executed model $em_i \in EM$. Thus, the number of samples to be analyzed depends on the cardinality of EM, and is greater than two. The samples are unpaired, i.e., independent, since the executed instances of different $em_i \in EM$ are not related. The difference between samples can go in any direction (increase or decrease), thus a two-tailed test is required. Next we need to consider the nature and the distribution of the collected data, i.e., in our case the metrics M. We are working with quantitative continuous performance data, which based on our experience in previous work [9,27,10], is not normally distributed, and thus requires the use of non-parametric tests where the original data is recorded in ranks. This type of tests are almost as powerful as parametric tests when the sample is large enough [19]. The **appropriate sample size** can be determined with the statistical power analysis [5], which uses as input the level of significance, the power and the expected effect size. The level of significance refers to the accepted level of probability that the observed result is a false positive, i.e., it is due to chance. It is usually conventionally set to 5% or 1% [4]. The power refers to the probability of false negative, i.e., accepting the H_0 when there is actually a difference between the results. The most commonly accepted level of power is 80% or above [4]. The expected effect size refers to the expected difference in the measured variable between the different groups. It can be determined based on pilot data, previous research if available, or an educated guess. As pilot data, we use the results of one test trial.

Considering all of the above factors, the appropriate test to run is the Kruskal-Wallis which is a non-parametric one-way ANOVA [6]. The H_0 of this test is that the distribution of the variable being tested is the same across the samples. The alternative hypothesis (H_1) is that the distribution of the same variable is different across the samples. Thus, rejecting the H_0 and accepting H_1, when $P - value < 0.05 = \alpha$, means that at least one sample stochastically dominates one other sample, i.e., in our work it means that there is **statistically significant difference** in the execution performance of at least two of the tested semantically equivalent models.

Given the non-deterministic nature of BPMSs, and software systems in general, using just one trial is not sufficient for obtaining reliable results. Having

multiple trials means that each em_i is deployed multiple times in isolation, and each time multiple instances are executed. Running the Kruskal-Wallis test on multiple trials is sufficient to answer research question RQ_1.

6. **Model Ranking**. If we want to identify the pairs of models between which performance differences exist (RQ_2), an additional post-hoc test, such as Dunn's test [6], is necessary (Fig. 1.6). In this test the H_0 is that the probability of observing a randomly selected value from the first sample that is larger than a randomly selected value from the second sample equals one half, i.e., no sample dominates the other. Rejecting the H_0, implies that the first sample is dominated by the second sample. Dunn's test does not account for the number of samples, thus it needs to be adjusted by the Bonferroni correction [6]. In our methodology, we use Dunn's test to **rank the models** from the best performing one to the worst performing one in each trial, assigning the same rank to models where Dunn's H_0 cannot be rejected. To combine the results of the different trials, we calculate a base-case rank as a sum of the assigned ranks on the em_i in each performed trial using Dunn's test.

The sufficient number of trials is determined by the stability of the obtained ranking, which can be assessed using a sensitivity analysis [14], that investigates how the uncertainty in the output is related to the uncertainty in the input. In our case, the output is the order of the performance of the executed models as per the base-case rank. We test the **sensitivity** of that order on the input, i.e., the ranks from the individual trials. We use a deterministic one-at-a-time, also known as one-side, sensitivity analysis, where we vary the trials to be included in the calculation of the base-case rank. Namely, we remove one of the trials at a time in order to observe whether the aggregated order will change.

3 Use Case Scenarios

In this section we specify how we have applied the methodology to three scenarios, each comprised of sixteen semantically equivalent models generated using four transformation rules, and executed on two different BPMSs.

3.1 Transformation Rules

Based on literature review and analysis of the BPMN standard we have defined R to include the following transformation rules which preserve the execution semantics. These rules can be frequently applied on real-world models given that gateways are among BPMN's most frequently used constructs [22].

r_1 **Coalesce Joins** - *precondition*: existence of multiple nested join gateways of the same type; *rule*: collapse the multiple join gateways into a single one;

r_2 **Coalesce Splits** - *precondition*: existence of multiple nested split gateways of the same type; *rule*: collapse the multiple split gateways into a single split gateway, adjusting the predicates when necessary to maintain the execution semantics of the model. Rules r_1, r_2 have been proposed as WFT-JC1 rule in [8];

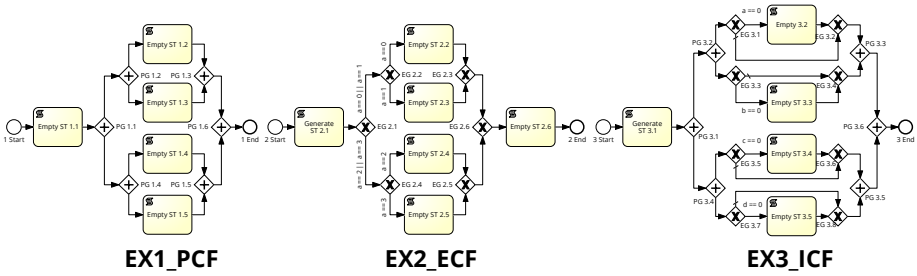

Fig. 2. Initial model m_i for each use case scenario

r_3 **Use Inclusive Gateway** - *precondition*: existence of parallel, exclusive gateway or a combination of parallel and exclusive gateway; *rule*: replace the existing gateway(s) with an inclusive gateway adjusting the predicate logic accordingly. This rule is motivated by [21];

r_4 **Use Implicit Gateway** - *precondition*: existence of a parallel split, inclusive split or an exclusive join; *rule*: replace a parallel split with multiple outgoing flows from the preceding activity, an inclusive split with multiple conditional outgoing flows from the preceding activity and an exclusive join with multiple incoming flows to the succeeding activity, as per the BPMN standard [17, pg.36-38], [23].

3.2 Executed Models

The three scenarios are aimed at testing the execution performance of semantically equivalent control flow structures expressing path parallelism (EX1_PCF), exclusive (EX2_ECF) and inclusive (EX3_ICF) path selection. The corresponding initial models (Fig. 2) are arbitrary and designed so that all four transformation rules are applicable. By adding the necessary inverse transformation rules and without loss of generality, any other of the semantically equivalent models could be used as an initial model.

If we do consider the process modeling guidelines [G] defined by Mendling et al. [21], we see that the initial models comply with: minimize the routing paths per element [G2], use one start and one end event [G3], match every split with a join of the same type [G4] and avoid OR gateways [G5]. [G4] and [G5] are also recommended by Koehler and Vanhatalo [18]. These initial models however do not use as few elements as possible [G1]. Still, some of the transformed models do follow [G1]. Use verb-object activity labels [G6] and decompose a model with more than 50 elements [G7] are out of the scope of this paper.

Starting from the three initial models, we have generated all the possible transformed models (Fig. 1.3) by applying combinations of up to four transformation rules, i.e., $z = 4$. The set of executed models $EM = \{m_i, t1, t2, ..., t1234\}$ contains 16 models, since multiple application of a single rule on a given model did not result with any structurally different models. By convention, the name of each model (e.g., $t123$) contains the index(es) of the transformation rule(s) ap-

plied to generate it (e.g., r_1, r_2, r_3). All the 48 transformed models are visualized at: `http://benchflow.inf.usi.ch/bpm2017`.

Given our goal of testing the performance impact of alternative control flow structures, all models in our experiments are fully automated, using mainly empty script tasks, and no manual or user tasks or service calls. This way we ensure that any identified performance bottleneck is due to the execution of the control flow, and not the tasks per se. Only the scripts that precede a decision gateway, inclusive or exclusive, contain code to generate random numbers, ensuring a uniform probability of executing any of the gateway's outgoing paths.

3.3 Load function

The load start function is comprised of the load time, the ramp-up period, the number of users and the think time. To avoid BPMS's saturation, based on our previous experience with running experiments on Camunda [10], we simulate 500 users who gradually become active within 30 seconds (ramp-up period), sending BP instantiation requests each second (think time) for a period of 5 min (load time). With Activiti, we reduced the number of users to 50 and increased the load time to 15 min to ensure that a sufficiently high number of instances is started. This choice allows us to obtain performance data with few outliers after removing the warm up period instances.

We conducted a pilot study for each of the scenarios to determine the effect size using each model's mean and standard deviation. The effect size was necessary to calculate the minimal sample size with a level of significance of 5% ($\alpha = 0.05$) and power level of 80% ($power = 0.80$) using GPower [1]. We have verified that the number of executed BP instances (from 22'497 to 53'754 in Camunda and from 41'912 to 44'677 in Activiti), in each trial, was sufficient to make statistical inference of the results. We run the experiments (Fig. 1.4) on two widely-used open-source BPMSs, Camunda v.7.5.0 and Activiti v.5.21.0, using the BenchFlow framework set up in the testbed environment described in [10]. We were prevented from including more BPMSs at this point due to limitations in their Management APIs [11] making the automation of the large number of experiment runs unfeasible.

3.4 Results

For each scenario (3), trial (3), executed model (16) and BPMS (2) we collected the duration of each BP instance in milliseconds (ms) and run the corresponding statistical tests as described in Sect. 2. To ease the comparison of the different models' performance, in Table 1 (Camunda) and Table 2 (Activiti) we show the 95% confidence interval (CI) of the duration of the BP instances of all the executed models (ms) in each trial for each experiment.

The results from running the experiments on **Camunda** are presented in Table 1. The initial model in the **EX1_PCF** has an average duration between 2.51

[1] `http://www.gpower.hhu.de`

Table 1. Camunda: 95% confidence intervals of the BP instance duration (ms)

	Parallel			Exclusive			Inclusive		
	Trial 1	**Trial 2**	**Trial 3**	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 1**	**Trial 2**	**Trial 3**
m_i	2.59±0.05	2.55±0.05	2.51±0.05	1.53±0.02	1.54±0.02	1.55±0.02	2.55±0.03	2.76±0.08	2.73±0.06
$t1$	2.36±0.06	2.44±0.05	2.42±0.04	1.49±0.02	1.46±0.02	1.42±0.02	2.63±0.06	2.43±0.03	2.44±0.03
$t2$	2.39±0.05	2.45±0.05	2.39±0.04	1.46±0.02	1.49±0.02	1.46±0.02	2.67±0.08	2.53±0.04	2.64±0.07
$t3$	2.57±0.05	2.51±0.05	2.58±0.05	1.57±0.02	1.53±0.02	1.53±0.02	2.02±0.03	2.03±0.03	2.09±0.04
$t4$	2.49±0.06	2.45±0.04	2.42±0.04	1.46±0.02	1.49±0.02	1.47±0.02	2.57±0.04	2.66±0.05	2.68±0.08
$t12$	2.21±0.04	2.24±0.04	2.25±0.04	1.42±0.02	1.41±0.02	1.40±0.02	2.43±0.04	2.51±0.05	2.45±0.07
$t13$	2.49±0.06	2.44±0.03	2.45±0.04	1.47±0.02	1.48±0.02	1.42±0.02	1.90±0.03	1.98±0.03	2.03±0.05
$t14$	2.31±0.05	2.28±0.04	2.28±0.03	1.44±0.02	1.41±0.02	1.36±0.02	2.43±0.03	2.50±0.05	2.68±0.08
$t23$	2.37±0.04	2.48±0.05	2.41±0.05	1.48±0.02	1.47±0.02	1.46±0.02	1.89±0.03	1.97±0.03	2.05±0.04
$t24$	2.29±0.04	2.35±0.05	2.30±0.04	1.43±0.02	1.39±0.02	1.40±0.02	2.38±0.03	2.57±0.04	2.44±0.03
$t34$	2.48±0.06	2.49±0.04	2.58±0.06	1.47±0.02	1.43±0.02	1.46±0.02	1.99±0.03	2.05±0.03	2.03±0.03
$t123$	2.23±0.04	2.29±0.05	2.28±0.05	1.41±0.02	1.41±0.02	1.39±0.02	1.90±0.03	1.87±0.03	1.83±0.02
$t124$	2.16±0.03	2.21±0.04	2.23±0.04	1.38±0.02	1.33±0.02	1.32±0.02	2.25±0.03	2.34±0.03	2.48±0.06
$t134$	2.35±0.04	2.38±0.05	2.34±0.05	1.45±0.02	1.40±0.02	1.41±0.02	1.92±0.03	1.90±0.03	1.91±0.03
$t234$	2.33±0.03	2.39±0.04	2.34±0.04	1.40±0.02	1.43±0.02	1.42±0.02	1.92±0.03	1.90±0.03	1.98±0.04
$t1234$	2.20±0.03	2.21±0.04	2.28±0.04	1.36±0.02	1.35±0.02	1.37±0.02	1.78±0.02	1.81±0.02	1.87±0.03

and 2.59 ms with CI range of ±0.05. Overlapping CIs with the initial models are noticed for $t3$, $t4$, $t13$ and $t34$, while for the rest of the transformed models the CI goes down to 2.16 ± 0.03 for $t124$ in trial 1. The average duration of the initial model in **EX2_ECF** is between 1.53 and 1.55 ms with CI range of ±0.02. In this experiment only the model $t3$ has a significantly overlapping interval with the initial model, which means that their performance is very similar. The best performing model with CI of 1.32 ± 0.02 is in trial 3 for $t124$, as is the case in EX1_PCF. The average duration of the initial model in **EX3_ICF** is between 2.55 and 2.76 ms with CI range of ±0.03 to ±0.08. When the inclusive gateway is not used, i.e., for models not applying r_3, the magnitude of the variation in performance compared to the initial model is similar as in the other two experiments, with the best performing model remaining $t124$ with CI of 2.25 ± 0.03 ms in trial 1. However, CI values get much lower when r_3 is used, with CI of 1.78 ± 0.02 ms in trial 1 for the model that applies all rules ($t1234$).

The results from running the experiments on **Activiti** are presented in Table 2. The average values of the duration of the initial model in **EX1_PCF** are between 25.22 and 26.04 ms with CI range of ±0.21. None of the transformed models has an overlapping interval with the initial parallel flow model, and the CI of the duration goes down to 15.83 ± 0.12 ms for $t1234$ in trial 1. Exclusive control flow executes faster with average duration of the initial model in **EX2_ECF** between 2.01 and 2.14 ms with CI range between ±0.06 to ±0.07. In this experiment, although many of the transformed models have overlapping intervals with the initial model, still the best performing model $t124$ in trial 1 has a rather lower duration CI of 1.39 ± 0.02 ms. In **EX3_ICF** the average duration of the initial model returns closer to the one in EX1_PCF with values between 27.93 and 29.87 ms with CI range between ±0.24 and ±0.30. Overlapping intervals are only noticed for $t4$, and while the CI of the duration for models without inclusive gateway goes only down to 22.42 ± 0.21 ms in trial 2 for $t124$, for model $t1234$ in trial 2 it goes all the way down to 9.93 ± 0.14 ms .

Table 2. Activiti: 95% confidence intervals of the BP instance duration (ms)

	Parallel			Exclusive			Inclusive		
	Trial 1	**Trial 2**	**Trial 3**	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 1**	**Trial 2**	**Trial 3**
m_i	26.04±0.21	25.33±0.21	25.22±0.21	2.12±0.07	2.01±0.06	2.14±0.07	29.87±0.30	27.93±0.24	29.57±0.27
$t1$	21.62±0.18	21.47±0.16	20.90±0.17	2.09±0.07	1.93±0.05	1.91±0.06	24.04±0.21	27.14±0.26	24.75±0.22
$t2$	21.34±0.17	20.38±0.14	20.55±0.15	2.02±0.06	1.98±0.06	1.95±0.05	25.43±0.24	27.29±0.26	26.75±0.26
$t3$	19.57±0.18	20.02±0.19	20.00±0.19	2.10±0.06	2.06±0.06	2.02±0.06	11.82±0.16	11.66±0.16	11.74±0.17
$t4$	25.32±0.21	24.55±0.20	24.66±0.20	2.06±0.06	1.92±0.05	1.99±0.06	28.92±0.26	28.24±0.26	29.64±0.27
$t12$	18.83±0.14	18.38±0.14	18.79±0.15	1.84±0.05	1.91±0.06	1.89±0.06	23.68±0.22	23.00±0.21	24.53±0.23
$t13$	18.26±0.15	17.39±0.14	17.42±0.15	1.92±0.06	1.93±0.06	2.02±0.06	11.38±0.18	11.19±0.18	11.11±0.15
$t14$	21.25±0.16	21.39±0.17	20.88±0.18	1.91±0.06	1.85±0.05	1.77±0.05	24.78±0.23	23.07±0.19	23.61±0.20
$t23$	16.52±0.14	16.37±0.14	15.62±0.12	2.11±0.06	1.93±0.05	2.12±0.07	10.56±0.14	10.90±0.16	11.06±0.16
$t24$	21.67±0.17	21.80±0.17	21.72±0.18	1.91±0.05	1.86±0.06	1.86±0.05	23.66±0.21	25.08±0.24	23.45±0.21
$t34$	19.06±0.17	18.06±0.15	19.13±0.17	2.11±0.07	1.89±0.05	1.93±0.05	11.14±0.15	11.02±0.15	11.06±0.15
$t123$	17.35±0.16	17.39±0.21	16.47±0.13	1.92±0.06	2.01±0.07	2.05±0.07	10.19±0.14	10.27±0.14	10.01±0.15
$t124$	19.09±0.14	17.86±0.12	18.64±0.14	1.39±0.02	1.45±0.03	1.59±0.04	22.46±0.21	22.42±0.21	23.31±0.24
$t134$	18.03±0.16	18.61±0.17	18.53±0.16	1.92±0.06	2.03±0.09	1.92±0.05	10.51±0.14	11.46±0.20	10.83±0.15
$t234$	18.04±0.16	17.36±0.16	17.08±0.15	1.89±0.05	1.98±0.07	2.01±0.07	10.53±0.15	10.51±0.15	10.71±0.16
$t1234$	15.83±0.12	15.75±0.13	15.86±0.13	1.72±0.04	1.67±0.04	1.62±0.04	10.03±0.13	9.93±0.14	9.99±0.13

On the raw data for the duration of each BP instance of all of the models belonging to the same experiment, we have run the Kruskal-Wallis Test (Fig. 1.5) for the significance of the differences between the models using IBM SPSS Statistics Version 24. Summary of the test's results is provided in Table 3. The *Total #* shows the total number of BP instances compared in the test, where larger *test statistic* values indicate larger differences between the compared models. The H_{RQ1} gets rejected if the values of the *Asymptotic Sig.* are smaller than 0.05.

The post-hoc Dunn's test (Fig. 1.6) indicates the pairs of models with statistically significant differences in the duration of their instances. We used its results to rank each model, assigning the same rank to models where the performance differences were not significant (Table 4). The sensitivity analysis confirmed that running 3 trials in each experiment is sufficient for the desired rank stability.

3.5 Discussion

3.5.1 Statistically Significant Performance Differences (RQ_1) The values of the asymptotic significance very close to 0 in the results of the statistical tests (Table 3) shows that there are statistically significant differences in the execution duration between instances of models which are semantically equivalent but structurally different. Thus, we can reject H_{RQ1}. The extent of performance differences between the models varies between experiments (EX1_PCF, EX2_ECF and EX3_ICF), as well as between BPMSs (Camunda and Activiti).

Table 3. Kruskal-Wallis Test Summary Results

		EX1_PCF			EX2_ECF			EX3_ICF		
		Trial 1	**Trial 2**	**Trial 3**	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 1**	**Trial 2**	**Trial 3**
Camunda	Total #	739'845	721'265	775'940	765'393	817'415	850'040	487'213	494'020	544'059
	Test Statistic	6'036	5'686	3'989	2'852	3'290	3'695	27'512	29'646	29'530
	Asymptotic Sig.	0	0	0	0	0	0	0	0	0
Activiti	Total N	697'492	698'976	699'478	708'138	710'987	710'467	687'930	687'011	687'746
	Test Statistic	135'513	141'611	139'567	1'268	1'268	1'882	351'816	354'735	354'047
	Asymptotic Sig.	0	0	0	0	0	0	0	0	0

Table 4. Models ranked over their performance in three trials using Dunns' test

Camunda						**Activiti**					
EX1_PCF		**EX2_ECF**		**EX3_ICF**		**EX1_PCF**		**EX2_ECF**		**EX3_ICF**	
Model	Rank	Model	Rank	Model	Rank	Model	Rank	Model	Rank	Model	Rank
$t124$	***1***	$t124$	***1***	$t1234$	***1***	$t1234$	***1***	$t124$	***1***	$t1234$	***1***
$t1234$	2	$t1234$	***1***	$t123$	2	$t23$	2	$t1234$	2	$t123$	***1***
$t12$	3	$t12$	2	$t134$	3	$t123$	3	$t24$	2	$t234$	2
$t123$	3	$t123$	2	$t234$	3	$t234$	4	$t12$	3	$t23$	2
$t14$	4	$t24$	2	$t13$	4	$t13$	5	$t14$	3	$t13$	3
$t24$	5	$t234$	2	$t23$	4	$t134$	6	$t123$	4	$t134$	3
$t234$	6	$t14$	3	$t34$	5	$t34$	7	$t134$	4	$t34$	3
$t134$	6	$t134$	3	$t3$	6	$t3$	8	$t234$	4	$t3$	4
$t2$	7	$t1$	4	$t124$	7	$t12$	9	$t1$	5	$t124$	5
$t1$	7	$t13$	4	$t12$	8	$t124$	9	$t2$	5	$t12$	6
$t23$	8	$t2$	5	$t1$	9	$t14$	10	$t4$	5	$t14$	7
$t4$	8	$t23$	5	$t24$	9	$t1$	11	$t13$	5	$t24$	8
$t13$	9	$t34$	5	$t14$	9	$t2$	12	$t23$	5	$t1$	9
$t34$	10	$t4$	6	$t2$	10	$t24$	12	$t34$	5	$t2$	10
m_i	11	$t3$	7	$t4$	11	$t4$	13	m_i	6	m_i	11
$t3$	12	m_i	7	m_i	12	m_i	14	$t3$	6	$t4$	11

3.5.2 Total Order (RQ_2) Given the cardinality of the executed models set, $|EM| = 16$, the theoretical maximal rank is 16 and would imply significantly different performance among all models. As can be seen in Table 4, execution of the models on Camunda results with 12 ranks for the parallel, 6 for the exclusive and 12 for the inclusive control flow experiment. Execution on Activiti results with 14 ranks for the parallel, 6 for the exclusive and 11 for the inclusive control flow experiment. Therefore, it is not possible to induce a total ordering (16 ranks) between all semantically equivalent models in a given experiment based on their execution duration. The actual number of ranks depends both on the BPMSs and on the experiment. In our use case the exclusive control flow models seem to have the most performance similarities, thus resulting with the smallest number of ranks (6 ranks in both BPMSs).

3.5.3 Experiments Performance Variability To facilitate the visualization (Fig. 3) of the differences between the duration interval of the initial model and that of the transformed models, we have decided to use the acceptability index [26]. The **acceptability index** is calculated as $I(A,B) = \frac{m(B)-m(A)}{w(B)+w(A)}$ such that $A = [a_l, a_r]$ and $B = [b_l, b_r]$ are interval values, where a_l, b_l and a_r, b_r stand for the left and right limits of the interval. $m(A)$ and $m(B)$, in our case, are the average duration of all the instances of the respective model, $w(A)$ and $w(B)$ are the half-width of the corresponding confidence interval. In this paper A always refers to the duration interval of the initial model m_i, and B refers to the duration interval of the transformed model $t1, ..., t1234$. Thus, negative values of the index show that on average the initial model's instances have longer execution than the instances of the respective transformed model, and vice-versa with positive index values. Index values between -1 and 1 indicate an overlap between the compared intervals A and B.

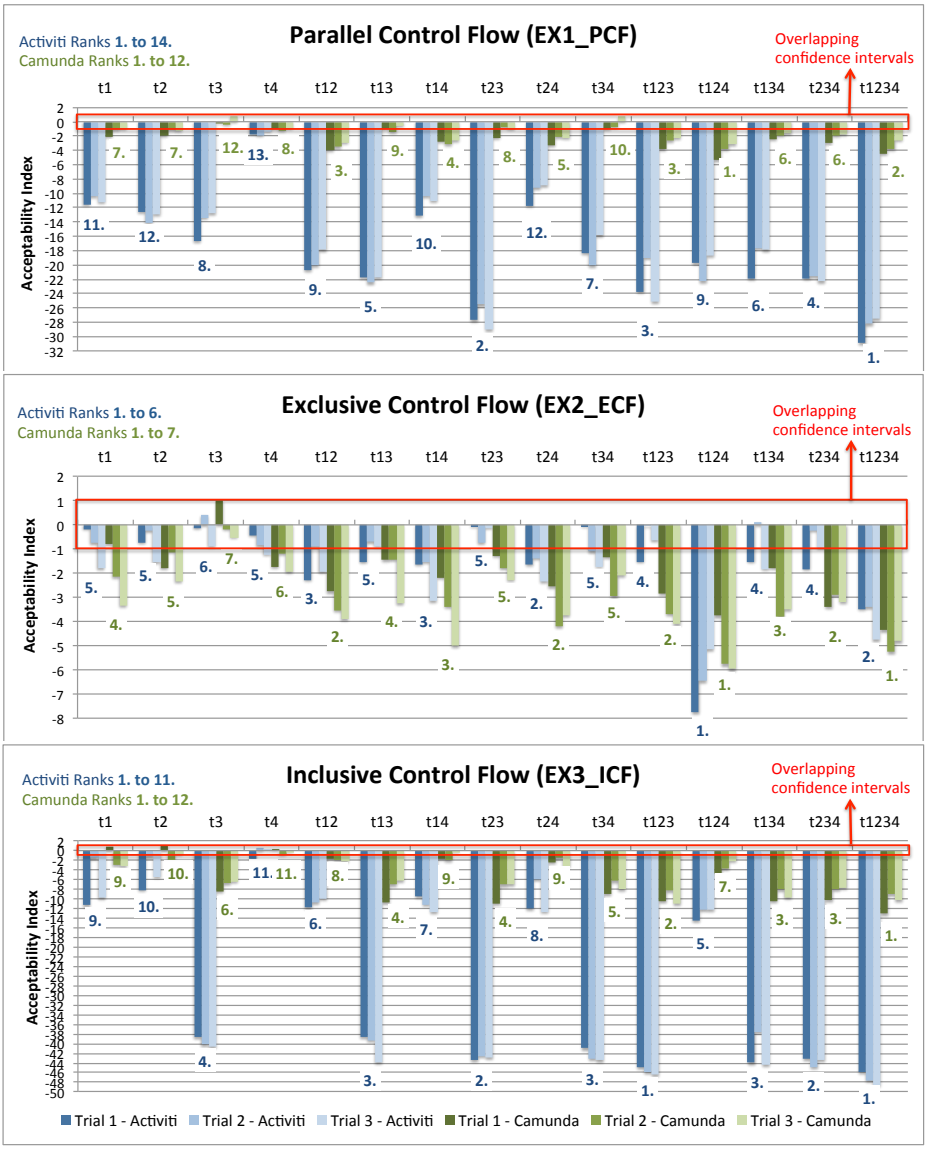

Fig. 3. Performance differences of transformed models relative to the three initial models with Activiti and Camunda (ranks and acceptability index of three trials)

As visualized in Fig. 3, the differences between BPMSs are particularly evident in the parallel (EX1_PCF) and the inclusive (EX3_ICF) control flow experiments. In **EX1_PCF** all transformed models perform better than the initial model when executed on Activiti (c.f. Table 2). When EX1_PCF is run on Camunda, the differences are lower in magnitude (c.f. Table 1), but still statistically significant as shown by Dunns' test (c.f. Table 4). In Camunda, the only trans-

formed model that performs worse than the initial model is $t3$ which uses the inclusive gateway with parallel gateway logic, i.e., with conditional statements which are always true. In Activiti on the other hand, $t3$ performs better than half of the models. As evident from Fig. 3, although applying the combination of all the transformation rules ($t1234$) seems to work well for both BPMSs, as mentioned earlier, in Camunda it is better not to use the inclusive gateway to implement parallelism since $t124$ ranks as the most performant one. Table 4 shows that, in Camunda all of the top performant models coalesce the split and join, i.e., contain the combination of r_1 and r_2, while in Activiti they coalesce the split while using the inclusive gateway to implement parallelism, i.e., they result from the combination of r_2 and r_3.

In **EX2_ECF**, both the difference between BPMSs and the difference between the transformation models (c.f. Fig. 3), is much smaller than in the other two experiments, with the confidence intervals, when applying just one transformation rule in Activiti, overlapping with initial model's confidence interval. The best performance is obtained by combining all rules, using either exclusive ($t124$) or inclusive gateway ($t1234$). These two models rank the same in Camunda, while in Activiti $t124$ ranks first, and $t1234$ second. The noticed trend in the first experiment for Camunda, to combine r_1 and r_2 in all top performing models, is also evident in this experiment.

The performance of the initial model in **EX3_ICF**, in both BPMSs, is similar to the one in EX1_PCF, with an even greater magnitude of differences between the initial and the transformed models in EX3_ICF (c.f. Fig. 3). The top two models ($t1234$ and $t123$) are the same in both BPMSs, with no statistically significant performance differences between them in Activiti as evident by their equal rank. The trend mentioned in EX1_PCF, i.e., to combine r_2 and r_3 in Activiti in the top performing models, reemerges in this experiment as well. In general, in EX1_PCF and EX3_ICF, the results indicate greater potential for performance improvement in Activiti than in Camunda.

3.5.4 Impact of Modeling Practices Dumas et al. [7] show that there is no absolute truth about the impact of structuredness on understandability and that it depends on the number of gateways in the model. Minimizing the overall model size [G1] and minimizing the number of routing paths per element [G2] cannot be both fulfilled, as acknowledged by [21]. Another controversial guideline refers to the implicit gateway which according to Koehler and Vanhatalo [18] simplifies BP model's visualization, while Recker's empirial study [23] shows that explicit gateways improve the interpretational fidelity, i.e., understandability.

Having this in mind, systematically applying the transformation rules (R) described in this paper generates a variety of models which give priority to different modeling practices. The graph layout guidelines and practices are not taken into consideration.

The initial model in EX3_ICF follows the guidance of avoiding inclusive gateways [G5], by using a combination of parallel and exclusive gateways. However, the results in both BPMSs, show that using r_3, i.e., an inclusive gateway, on

EX3_ICF models significantly improves the performance, as all the transformations applying r_3 outrank the ones that do not. As evident from Table 4, the performance of the model combining all rules together ($t1234$) is always ranked as first or second in all experiments and in both BPMSs. When it is ranked second, $t124$ is ranked as first. While $t1234$ is indeed the smallest executed model in size [G1], not all the best performing models are among the smallest ones. For instance, in Activiti in EX1_PCF model $t23$ is second ranked, but it has 24 elements, as opposed to the smallest models in this experiment ($t1234$ and $t124$) which have 18 elements. Furthermore, both $t1234$ and $t124$ do not comply with other modeling guidelines, since their maximum number of routing paths per element is 5, as opposed to 3 in the initial models [G2], they do not use explicit gateways thus they are not structured [G4], and $t1234$ also uses inclusive gateways [G5]. Thus, in most of the cases when modeling for deployment, if performance is important, priority should be given to minimizing the overall model size [G1] with respect to [G2] or [G4].

However, as previously stated, modeling should be a discretional decision of the modeler and we do not aim at changing modeling practices (or guidelines) to improve the execution performance. Our goal is to elicit BPMS performance improvements by enabling different BPMS vendors to test their products by using the proposed methodology, with the same or different set of transformation rules and models. If they notice that the acceptability indexes they calculate are significantly lower than -1, dedicating time to implementing performance optimization can bring them competitive advantage. For instance, although further work with greater number of initial models and with different BPMSs is required to make generalization about the effect of the transformation rules included in this paper, the initial results already provide some useful hints. Clearly, BPMS vendors can boost the performance of their products by coalescing multiple splits and joins of the same type. Activiti could also take inspiration of their implementation of the inclusive gateway, when implementing parallelism as r_3 provides a non-negligible performance improvements.

4 Related Work

Previous studies have analyzed the impact of BP model structure on its understandability [24,23] or error-proneness [20]. For example, Mendling et al. [21] provide seven modeling guidelines towards more comprehensive and syntactically correct models, synthesized from empirical work linking model's structural characteristics with its understandability, error probability and label ambiguity.

We are not aware of any existing work studying the connection between the BPMN static control flow structure and its execution performance. On the other hand, there is extensive work on programming language compiler optimization based on transformations that reduce the number of instructions or maximise parallelism. Bacon et al. [3] provide a comprehensive overview of compiler transformations, while Hoste et al. [15] discuss optimization space exploration strategies to provide for inevitable optimization trade-offs. Furthermore,

in Database Management Systems (DBMSs), queries are optimized using transformation rules which preserve their execution semantics. Jarke and Koch [16] propose a framework for evaluation of query optimization, comprised of four steps: 1) find an internal query representation, 2) apply logical transformations, 3) define alternative sequences of elementary operations, and 4) find the cheapest alternative among the ones proposed in step 3 and execute it. Taking inspiration from this existing work, we focus on transformation optimization strategies in BPMSs, and our initial goal is to assess whether different representations of the same BP significantly impact its execution performance. Gournaris [13] already points to DBs and data-centric flows as automated performance optimization opportunity in BPMN process models' execution. BPMN elements are mapped to annotated directed acyclic graphs used for optimal task ordering and task assignment based on statistical metadata, such as task duration, gathered from execution logs. While [13] targets optimal task execution, in this paper we focus on the control flow.

Work on BP models' equivalence [1] and modeling best practices [21] is related to what we do, since we use semantically equivalent models to study the effect that their structure has on their execution by a given BPMS. Eder et al. [8] propose a set of basic operations (e.g., moving or confluence of gateways) to transform a given BP model represented as a structured graph to a semantically equivalent model. Gert et al. [12] propose a language independent algorithm for determining semantical equivalence of fragments of structured or unstructured BP models, motivated by the industry need of BP model change management. They use a non-exhaustive set of rules for rewriting BPs into a normal form, later used for fragment comparison. While we use the operations and rules mentioned in existing work [8,12], we also take into consideration BPMN-specific transformations, such as replacing explicit gateways with implicit ones. Our approach also differs in the goal of the use case which requires such transformations.

5 Threats to Validity

Construct Validity - We conduct our experiments on a single version of two BPMSs in a standalone deployment and only in their default configuration, since it is the configuration usually utilized by potential users when evaluating system's performance. Although each model is executed in isolation from the other models, all the instances of the same model are executed together.

Internal Validity - The experiments we perform are inherently subject to variability in the obtained metrics value, due to the many factors impacting the runtime of a software system. We mitigate this variability by defining load functions that do not overload the BPMSs [27], performing multiple trials for each of the models, and verifying the variance among trials in order to provide reliable measures validated by significance testing.

External Validity - The results we obtained present limited generalizability since: they depend on the behaviour of different BPMSs, or the same BPMS under a different load function; the size and the number of the initial models are

rather small; and all models are realized by script tasks. We plan further experiments to improve and delimit the generalizability of our results w.r.t. different BPMS, load functions, initial model sizes and used BPMN 2.0 elements.

6 Conclusion and Future Work

In this work we study and compare the execution performance of semantically equivalent BP models with different control flow structures. To do so, we propose a methodology for deriving such models based on an initial model and a set of semantics-preserving transformation rules. The models are executed on different BPMSs measuring the corresponding process instance duration, which is statistically analyzed to identify and characterize significant performance differences. By applying the methodology on three scenarios (parallel, exclusive and inclusive control flows), we identify significantly different performance among the models in both BPMSs (RQ_1). However, in all experiments, it was not possible to establish a total order among all 16 semantically equivalent models (RQ_2).

The observed performance variability is more evident in Activiti (acceptability index up to -38.53) than in Camunda (up to -13.07). We discover that following certain modeling guidelines, e.g., avoiding the use of inclusive gateways when implementing inclusive control flow execution semantics, has a negative performance impact on the model's execution duration. These are only initial but promising results, measured with load functions designed to avoid system saturation: 500 users for Camunda and 50 for Activiti. Further experiments are necessary to investigate the impact of the load function on the observed performance differences. However, these results are already sufficient to demonstrate the existence of statistically significant differences in the execution of semantically equivalent models designed following different modeling practices.

Our research efforts will further explore the execution performance improvement opportunities by using larger initial models, larger sets of transformation rules and more BPMSs. We are currently comparing each of the transformation rules individually to draw conclusions on which of them are good candidates for optimization rules. These initial results pave the way towards automatic BP model performance optimization by means of semantics-preserving transformation rules that can be applied when a BP model is deployed on a specific BPMS.

Acknowledgements This work is partially funded by the "BenchFlow" project (DACH Grant Nr. 200021E-145062/1).

References

1. van der Aalst, W.M.P., et al.: Process equivalence: Comparing two process models based on observed behavior. In: Proc. of BPM. pp. 129–144. Springer (2006)
2. Abbott, M.L., Fisher, M.T.: The art of scalability. Pearson (2009)
3. Bacon, D.F., Graham, S.L., Sharp, O.J.: Compiler transformations for high-performance computing. ACM Computing Surveys (CSUR) 26(4), 345–420 (1994)

4. Cohen, J.: A power primer. Psychological bulletin 112(1), 155 (1992)
5. Dattalo, P.: Determining sample size: Balancing power, precision, and practicality. Oxford University Press (2008)
6. Dinno, A.: Nonparametric pairwise multiple comparisons in independent groups using dunns test. Stata Journal (2015)
7. Dumas, M., La Rosa, M., et al.: Understanding business process models: the costs and benefits of structuredness. In: Proc. of CAiSE '12. pp. 31–46. Springer (2012)
8. Eder, J., Gruber, W., Pichler, H.: Transforming workflow graphs. In: Interoperability of Enterprise Software and Applications, pp. 203–214. Springer (2006)
9. Ferme, V., Ivanchikj, A., Pautasso, C.: A framework for benchmarking BPMN 2.0 workflow management systems. In: Proc. BPM '15. Springer (2015)
10. Ferme, V., Ivanchikj, A., Pautasso, C.: Estimating the cost for executing business processes in the cloud. In: Proc. of BPM Forum 2016. pp. 72–88. Springer (2016)
11. Ferme, V., et al.: Workflow management systems benchmarking: Unfulfilled expectations and lessons learned. In: In Proc. of ICSE 2017 (May 2017)
12. Gerth, C., et al.: Detection of semantically equivalent fragments for business process model change management. In: Proc. of SCC. pp. 57–64. IEEE (2010)
13. Gounaris, A.: Towards automated performance optimization of BPMN business processes. In: Proc. of ADBIS. pp. 19–28. Springer (2016)
14. Hamby, D.: A review of techniques for parameter sensitivity analysis of environmental models. Environmental monitoring and assessment 32(2), 135–154 (1994)
15. Hoste, K., Eeckhout, L.: Cole: compiler optimization level exploration. In: Proc. of CGO. pp. 165–174. ACM (2008)
16. Jarke, M., Koch, J.: Query optimization in database systems. ACM Computing surveys (CsUR) 16(2), 111–152 (1984)
17. Jordan, D., Evdemon, J.: Business Process Model And Notation (BPMN) Version 2.0. OMG (January 2011), `http://www.omg.org/spec/BPMN/2.0/`
18. Koehler, J., Vanhatalo, J.: Process anti-patterns: How to avoid the common traps of business process modeling. IBM WebSphere Devel. Tech. Journal 10(2), 4 (2007)
19. Marusteri, M., Bacarea, V.: Comparing groups for statistical differences: how to choose the right statistical test? Biochemia medica 20(1), 15–32 (2010)
20. Mendling, J.: Metrics for process models: empirical foundations of verification, error prediction, and guidelines for correctness, vol. 6. Springer (2008)
21. Mendling, J., Reijers, H.A., van der Aalst, W.M.: Seven process modeling guidelines (7pmg). Information and Software Technology 52(2), 127–136 (2010)
22. Muehlen, M., Recker, J.: How much language is enough? theoretical and practical use of the business process modeling notation. In: Proc. of CAiSE. pp. 465–479. Springer (2008)
23. Recker, J.: Empirical investigation of the usefulness of gateway constructs in process models. European Journal of Information Systems 22(6), 673–689 (2013)
24. Reijers, H.A., Mendling, J.: A study into the factors that influence the understandability of business process models. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 41(3), 449–462 (2011)
25. Rosa, M.L., et al.: Managing process model complexity via concrete syntax modifications. IEEE Transactions on Industrial Informatics 7(2), 255–265 (May 2011)
26. Sengupta, A., Pal, T.K.: On comparing interval numbers. European Journal of Operational Research 127(1), 28–43 (2000)
27. Skouradaki, M., et al.: Micro-benchmarking BPMN 2.0 workflow management systems with workflow patterns. In: Proc. of CAiSE'16. pp. 67–82. Springer (2016)