A Generic Framework for Context-Aware Process Performance Analysis

Bart F.A. Hompes[1,2(✉)], Joos C.A.M. Buijs[1], and Wil M.P. van der Aalst[1]

[1] Department of Mathematics and Computer Science,
Eindhoven University of Technology, Eindhoven, The Netherlands
{b.f.a.hompes,j.c.a.m.buijs,w.m.p.v.d.aalst}@tue.nl

[2] Philips Research, Eindhoven, The Netherlands

Abstract. Process mining combines model-based process analysis with data-driven analysis techniques. The role of process mining is to extract knowledge and gain insights from event logs. Most existing techniques focus on process discovery (the automated extraction of process models) and conformance checking (aligning observed and modeled behavior). Relatively little research has been performed on the analysis of business process performance. Cooperative business processes often exhibit a high degree of variability and depend on many factors. Finding root causes for inefficiencies such as delays and long waiting times in such flexible processes remains an interesting challenge. This paper introduces a novel approach to analyze key process performance indicators by considering the process context. A generic context-aware analysis framework is presented that analyzes performance characteristics from multiple perspectives. A statistical approach is then utilized to evaluate and find significant differences in the results. Insights obtained can be used for finding high-impact points for optimization, prediction, and monitoring. The practical relevance of the approach is shown in a case study using real-life data.

Keywords: Process mining · Performance analysis · Context-aware · Root cause analysis

1 Introduction

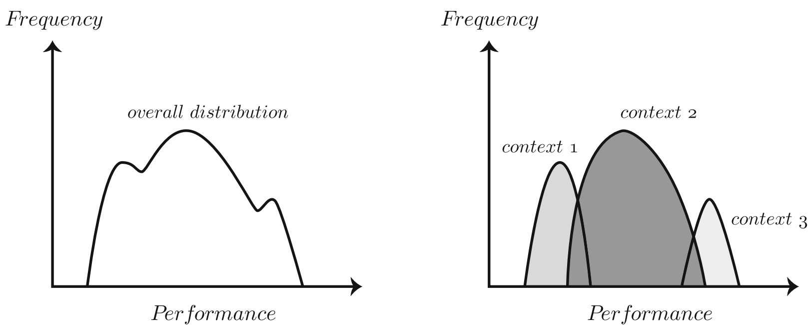
Process mining is an emerging discipline that deals with extracting knowledge and non-trivial insights from event data recorded by information systems. Such event logs capture the different steps (activities) recorded for cases (customers, patients, etc.) that follow a process. Usually, information is stored about what activity was performed by whom and at what time. Additionally, information about the involved resources or process-specific data attributes such as the customer type or the age of a patient may be recorded as well. Existing process mining techniques have focused on three main areas: process discovery, conformance checking and process enrichment. Process discovery can be defined as the automated extraction of process models from event logs. Insights can be gained

C. Debruyne et al. (Eds.): OTM 2016 Conferences, LNCS 10033, pp. 300–317, 2016.
DOI: 10.1007/978-3-319-48472-3_17

on the order of activities, parallel parts, alternative flows or iterative steps in a process. Conformance checking compares the observed and modeled behavior by aligning the cases in the event log with the process described in a model. This way, process compliance can be analyzed and skipped activities, improper execution orders or deviations from protocols can be discovered. Process enhancement deals with the extension or improvement of an existing a-priori process model with information about the actual process recorded in the log. Models can for example be extended to show performance information or conformance issues.

Typically, discovering process models for flexible processes results in models that are difficult to interpret, as most cases exhibit unique behavior. Often, no a-priori model is present, making conformance checking and process enhancement challenging tasks. As a result, gaining an understanding of the underlying process and finding points for optimization is far from trivial.

From most real-life event logs, we can gain information about different performance characteristics. Typically, we are interested in characteristics such as waiting times, throughput times, and utilization rates. Existing process performance analysis techniques, however, are limited to describing the overall behavior, such as mean waiting times and durations. By looking at the frequency distributions of these measures, we can see that these typically do not follow a single distribution curve. Rather, density plots give the impression of being composed of multiple components, as depicted in Fig. 1. The ideas presented in this paper aim to discover such underlying components. Often times, performance characteristics of a specific activity, case, or entire process highly depend on the context. For example, preceding tasks, involved resources and their workload, or even the weather can have a big effect on performance. In Business Intelligence tools and techniques, data is sliced and diced to view key performance information from different perspectives. The idea is that contextual properties of process entities such as cases and resources form the underlying components in the frequency



(a) No insights can be gained from overall frequency distributions.

(b) The performance on this KPI depends on the context.

Fig. 1. Frequency distributions are composed of multiple components. We aim to discover the underlying contextual properties that lead to these components.

distributions mentioned above, and are considered as the analysis perspectives. In this paper, we introduce a context-aware process performance analysis approach that can be used to analyze event logs in a similar way. Process entities are labeled with their context and their performance is calculated. Hypothesis testing is used to automatically discover significant differences in performance measures for different contexts.

The remainder of the paper is structured as follows. Section 2 introduces preliminary definitions. The notion of process context and performance is introduced in Sect. 3. How to automatically find significant differences in performance using hypothesis testing is explained in Sect. 4. In Sect. 5, the practical relevance of the approach is shown using a case study on a real-life dataset. The paper is positioned and related work is outlined in Sect. 6. In Sect. 7 the paper is concluded and ideas for possible future work are given.

2 Preliminaries

The executed *events* of multiple *cases* of a *process* are recorded in an *event log*. Event logs serve as input for any process mining technique. An event is a particular execution of an activity for a case, potentially having additional data attributes such as a timestamp or the responsible resource. A trace is a finite sequence of events, and describes one specific instance (i.e. a case) of the process at hand in terms of the executed activities. A case can also have additional (case-level) attributes such as a birthdate or customer type. Definitions for events and cases used here are based on those in [2].

Definition 1 (Event, attribute). *Let $\mathcal{E}$ be the event universe, i.e. the set of all possible event identifiers. Events may be characterized by various attributes. Let N be a set of attribute names. For any event $e \in \mathcal{E}$ and attribute name $n \in N$: $n(e)$ is the value of attribute n for event e. If event e does not have an attribute named n, then $n(e) = \perp$ (null value).*

Typically, the following attributes are present for all events: $activity(e)$ is the *activity* associated to event e, $time(e)$ is the *timestamp* of e, $resource(e)$ is the *resource* associated to e, and $trans(e)$ is the *transaction type* of e. Possible

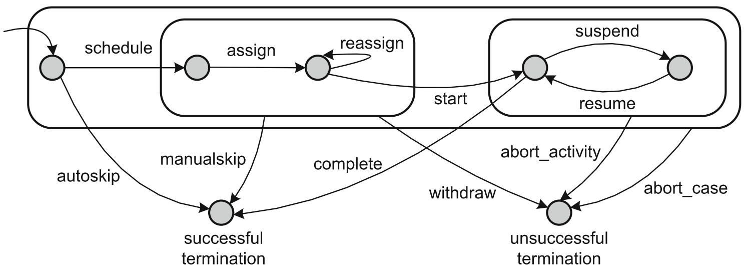


Fig. 2. Standard transactional life-cycle model.

transaction types are scheduled, assigned, suspended, etc. as depicted in the standard transactional life-cycle model in Fig. 2. Each event is associated with a state transition for a single activity instance. These states are used to calculate performance information such as duration and waiting times for the activity. An example event log (L_1) can be found in Table 1. Here, next to the standard attributes, cases have the data attributes 'Age' and 'Type', and events have attribute 'Cost'. Note that case 4 has no value for the attribute 'Age'.

Definition 2 (Case, trace, event log). *Let $\mathcal{C}$ be the case universe, i.e. the set of all possible case identifiers. Cases, like events, have attributes. For any case $c \in \mathcal{C}$ and attribute name $n \in N$: $n(c)$ is the value of attribute n for case c ($n(c) = \perp$ if c has no attribute named n). Each case has a mandatory attribute 'trace': $trace(c) \in \mathcal{E}^*$. $\hat{c} = trace(c)$ is a shorthand notation for referring to the trace of a case. A trace is a finite sequence of events $\sigma \in \mathcal{E}^*$, such that each event appears only once, i.e. for $1 \leq i < j \leq |\sigma|$: $\sigma_i \neq \sigma_j$. For any sequence $s = \langle s_1, s_2, \ldots, s_n \rangle$, $set(s) = \{s_1, s_2, \ldots, s_n\}$ converts a sequence into a set, e.g. $set(\langle a, b, c, b, c, d \rangle) = \{a, b, c, d\}$. An event log is a set of cases $L \subseteq \mathcal{C}$ such that each event appears at most once in the entire log, i.e. for any $c, c' \in L$ such that $c \neq c'$: $set(\hat{c}) \cap set(\hat{c}') = \emptyset$.*

Table 1. Example event log L_1.

Case id	Case attributes		Event id	Event attributes				
	Age	Type		Time	Activity	Transition	Resource	Cost
1	33	Premium	1	2016-1-4 8:00	A	Start	John	10
			2	2016-1-4 9:15	A	Complete	John	0
			3	2016-1-4 10:12	B	Complete	Bob	20
			4	2016-1-4 14:00	C	Schedule	Sue	0
			5	2016-1-4 14:05	C	Start	Sue	15
2	27	Basic	6	2016-1-6 10:43	A	Start	Bob	15
			7	2016-1-6 11:00	A	Complete	Bob	0
			8	2016-1-7 09:33	B	Complete	John	30
			9	2016-1-7 09:35	C	Schedule	Sue	30
3	18	Basic	10	2016-1-7 9:27	A	Start	John	15
			11	2016-1-7 10:40	A	Complete	John	0
			12	2016-1-7 15:03	B	Complete	Bob	30
4	$\perp$	Premium	13	2016-1-7 12:10	A	Start	Bob	10
			14	2016-1-7 12:24	A	Complete	Bob	0
			15	2016-1-8 08:47	B	Complete	John	30
5	41	Basic	16	2016-1-8 15:32	A	Start	Bob	15
			17	2016-1-8 15:51	A	Complete	Bob	0
...	...	...	...	...	...	...	...	...

3 Context-Aware Performance Analysis

Performance correlates with contextual information. As explained in Sect. 1, this information is lost in a single distribution. Hence, a distribution per context may be more precise, as shown in Fig. 1. Note that a composed distribution without context has limited value. For example, it could be that an activity for which one of two resources is involved, on average has a waiting time of one week. One resource might be overbooked, leading to waiting times of several weeks, while the other resource might be able to perform the task in a few hours. Clearly, which resource is assigned to perform the task for a certain case will determine the waiting time for that specific case. The average waiting time for all cases will not accurately represent the waiting time for either resource. Typically, multiple contextual properties having many possible values are in play. Our aim is to find, given the distribution of performance measures, which, if any, contextual properties compose this distribution. Such insights can lead to better predictions, help in monitoring for change, aid in optimizing scheduling, etc.

In this section, the concept of context-aware performance analysis is introduced. Subsection 3.1 describes process entities and the process context, and typical examples of contextual properties are given. Subsection 3.2 explains the concept of performance. The interrelation between context and performance is explained in Subsect. 3.3.

3.1 Process Entities and Context

Process entities have a type and represent a collection of events. The default process entity types are *case*, *activity instance*, *event*, and *resource*. Resources are process entities as an event log can be seen as a collection of resources, each of which is performing a set of events. This list can be extended with additional entity types depending on the information that is available in the event log at hand. Formally, process entities are defined as follows.

Definition 3 (Process entity). *Let $\mathcal{T}$ be the universe of process entity types, and $\mathcal{D}$ the universe of entity identifiers. Let $\mathcal{I} = \mathcal{D} \times \mathcal{P}(\mathcal{E})$[1] denote the universe of process entities. $\mathcal{I}_t$ denotes the set of process entities of type $t \in \mathcal{T}$. Function $\Phi : \mathcal{P}(\mathcal{C}) \times \mathcal{T} \to \mathcal{P}(\mathcal{I})$ maps an event log to a set of entities of a given entity type.*

For example, applying Φ to event log L_1 and the resource, activity, and case entity types we obtain:

$$\begin{aligned}
\Phi(L_1, resource) =&\{\ (Bob, \{3,6,7,12,13,14\}),\ (Sue, \{4,5,9\}),\\
&(John, \{1,2,8,10,11,15\})\ \}\\
\Phi(L_1, activity) =&\{\ (A, \{1,2,6,7,10,11,13,14,16,17\}),\ (B, \{3,8,12,15\}),\\
&(C, \{4,5,9\})\ \}\\
\Phi(L_1, case) =&\{\ (1, \{1,2,3,4,5\}),\ (2, \{6,7,8,9\}),\ (3, \{10,11,12\}),\\
&(4, \{13,14,15\}),\ (5, \{16,17\})\ \}
\end{aligned}$$

[1] $\mathcal{P}(\mathcal{E})$ denotes the powerset over $\mathcal{E}$, i.e. all possible subsets of $\mathcal{E}$.

Table 2. Example context functions.

Context function	Applicable to entity type
Number of cases	Process
Number of previous events	Case
Prefix	Activity instance, Event
Suffix	Activity instance, Event
Number of occurrences	Activity instance
Utilization rate of resource	Event
Number of concurrent events	Event
Resource ID	Event
...	...

Typically, entities are related to other entities. For example, activity instances are made of one or more events (representing a certain life-cycle transition, mentioned in Sect. 2) and, at the same time, are part of a trace of a case.

The contextual properties of a process entity are obtained by applying a context function to that entity. A context function maps a process entity to a context label (descriptive value) that aims to describe the entity's context. For example, taking activity instances as entities and the executing resource as the context, we can see how different resources affect activity KPIs such as waiting time, duration, etc. Context functions are defined only for certain types of process entities. For example, cases typically do not have one resource associated with them, but events do. Context can be explicit (i.e. attribute values such as the involved resource or customer type) or implicit (i.e. calculated values such as the number of days spent in the hospital or the cost-profit ratio of a customer). Formally, context functions are defined as follows.

Definition 4 (Context function). *Let $\mathcal{I}$ be the universe of process entities and $\mathcal{T}$ the universe of entity types. Let $\mathcal{V}$ be the universe of contexts. A context function $\Upsilon : \mathcal{I}_t \rightarrow \mathcal{V}$ maps a process entity of type $t \in \mathcal{T}$ to a context.*

For example, we can define the case-type context function of case entities as $type : \mathcal{I}_{case} \rightarrow \mathcal{V}$. Consequently, $\{type(i) | i \in \Phi(L_1, case)\} = \{Basic, Premium\}$. Other examples of context functions can be found in Table 2. Note that additional context functions can be defined based on the data available in the event log at hand. After mapping each process entity of a certain type to its context with one or more applicable context functions, performance characteristics are calculated.

3.2 Performance

Process performance analysis aims to improve processes with respect to time, cost, and/or quality [2]. In traditional Business Intelligence methods, key performance indicators are typically used to discover and monitor bottlenecks, deviations from protocol, violations of regulations, service level agreements, etc. In process mining, measures such as throughput times of activity instances or cases, sojourn times, waiting times, frequencies, time between activities, etc. are calculated based on the data stored in event logs. Metrics such as utilization rates and case load or the cost to gain new customers are derived from these measures. Both measures and metrics can be KPIs, depending on their business value. Performance can be seen as the *result* or *score* of a certain entity on a certain performance function. Performance functions are formally defined as follows.

Definition 5 (Performance function). *Let $\mathcal{I}$ be the universe of process entities and $\mathcal{T}$ the universe of entity types. A performance function $\Lambda : \mathcal{I}_t \rightarrow \mathbb{R}$ maps a process entity of type $t \in \mathcal{T}$ to a KPI result.*

We let the result of performance functions be numeric (continuous) values. For example, we can define the duration of activity instance entities as $duration : \mathcal{I}_{activity} \rightarrow \mathbb{R}$. Consequently, $\{duration(i) | i \in \Phi(L_1, activity)\} = \{75, 0, 0, 17, 0, 0, 73, 0, 14, 0, 19\}$ (in minutes). Note that only activities that have start and complete events have a duration. Otherwise a duration of 0 is recorded.

In order to calculate these KPIs, information from the events related to the entity is necessary. Typically, information about when events (activity transitions) were performed is required. Sometimes information from events related to related process entities is necessary as well. For example, the duration of an activity is calculated as the time difference between the start and completion of the activity. These time values are stored in two separate events representing the respective life-cycle transitions of the activity. After calculating KPIs, performance characteristics are related to contextual properties of the process entities.

3.3 Context-Aware Performance

By splitting performance measurements for a specific entity over the different context labels assigned to it, we obtain context-aware performance results. Relating the performance characteristics of process entities to their contextual properties shows whether and where correlations exist. From these results, we can see if (and how) contextual properties influence the performance of the process. For example, we can analyze the duration of activity A from L_1. The resource involved in executing the activity is taken as the context. Formally, we take all entities from $\Phi(L_1, activity)$ that are related to activity A and compute their duration. These durations are then linked to the context by grouping them based on the context label assigned to the specific activity, i.e. $resource(\Phi(L_1, activity))$. As activity A can only be performed by either John or Bob, this leads to two groups of measurements. The specific values are: John {75, 73} averaging 74 min, and Bob {17, 14, 19}, averaging 16.7 min. Clearly, using the overall average of 39.6 min as an estimation for the duration of activity A in calculations, prediction and planning is imprecise, while using the context-aware averages gives a much better estimation.

Manually analyzing every possible combination of performance and context function is a tedious and error-prone task. Therefore, we propose an automated

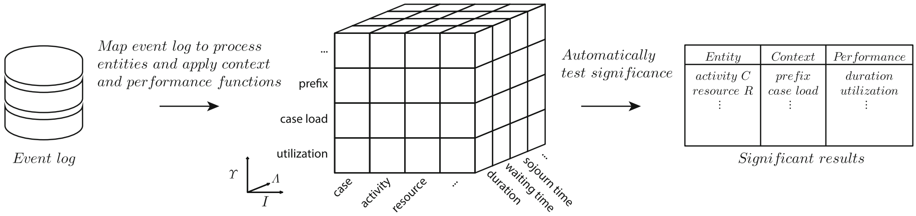


Fig. 3. A graphical overview of the approach. Performance measurements for process entities are related to their context. Hypothesis testing is used to automatically find statistically significant differences.

approach to test for significant differences between contexts for all entities and applicable context and performance functions. This analysis is done by means of statistical hypothesis testing, and will be explained in Sect. 4.

4 Statistical Hypothesis Testing

As explained in Subsect. 3.3, context-aware performance results are used to find correlations between contextual properties of process entities and process performance. However, the number of combinations and therefore possible correlations is often quite high, complicating manual analysis. In order to automate the analysis of significance of differences between context results, we follow a statistical approach. A graphical overview can be seen in Fig. 3.

Performance results for different contexts are seen as different samples. For these samples, the variance is analyzed. The null hypothesis is that no significant differences exists between the samples, and therefore, that the chosen context function does not have a significant effect on the performance results. Hence, if the null hypothesis is rejected, a possible cause for performance differences is found. By automating these analyses, many combinations of context and performance functions can be tested, and possible points for performance improvement can be rapidly discovered. Nonetheless, this automated technique has some drawbacks, which are discussed in Subsect. 4.4.

In the automated approach, the samples are first tested for normality, i.e. it is tested whether the values are sampled from a normally distributed population. If so, the one-way analysis of variance test is used. If not, a power transformation is applied in an attempt to make the data follow a normal distribution. When the data cannot be transformed, a non-parametric analysis of variance test is used (Subsect. 4.1). In case the null hypothesis is rejected, post-hoc analysis is performed using a multiple comparison procedure (Subsect. 4.2), in order to identify which samples are significantly different from each other, as the analysis of variance only indicates the existence of such a difference in a set of samples, but not the location. In other words, by applying our approach, those contexts that lead to significantly different performance results are discovered (Subsect. 4.3).

Besides relating individual contextual properties to performance results, contexts can be grouped to find combined correlations. For example, the involved resource or the previous activities might not influence an activity's waiting time individually, but the combination might. The analysis of combined contexts is analogous to that of a single context. However, it should be noted that this analysis significantly increases the search space.

4.1 Hypothesis Testing

Statistical models can be used to analyze the difference (variance) between groups. Analysis of variance (ANOVA) provides a statistical test of whether or not the means of several groups have the same standard deviation. The one-way (single-factor) ANOVA tests whether a single factor leads to a significant difference in the groups [16]. Hence, it generalizes the t-test to more than two groups. In our case, it is used to test whether different contexts lead to significant differences in KPI results for a given process entity and context function.

As explained, ANOVA is based on a hypothesis test where the null hypothesis is that the means of all groups are equal. A critical value (or p-value) is used as a number that the test statistic must exceed to reject the null hypothesis. Typically, a value of 0.05 is used. Multiple variants of the ANOVA have been proposed in literature and are widely used [16,17,21]. The basic assumptions are independence of observations, homogeneous variances, and population normality. The former two will be discussed in Subsect. 4.4.

In order to be able to perform an analysis of variance test on the performance results, first, the samples are tested for normality by a goodness-of-fit test. The Shapiro-Wilk test [25] was found to have the best power for a given significance in several studies [23]. In case the null hypothesis is accepted, i.e. the data come from a normally distributed population, we can proceed with the analysis of variance. In case the null hypothesis is rejected, i.e. the data do not follow a normal distribution, we attempt to transform the data to make it follow a normal distribution by applying the Box-Cox power transformation [8]. If the data does not come from a normally distributed population and cannot be transformed, we apply a non-parametric analysis of variance test. These types of tests, known as ANOVA on ranks, are less powerful but are designed for situations where normality cannot be assumed. We use the Kruskal-Wallis test [17] in case normality cannot be assumed for the KPI values for the different contexts.

In order to evaluate the effect of a combination of contexts, we can either create combined context functions or use a multiple-factor analysis of variance procedure.

4.2 Post Hoc Analysis

In case the analysis of variance's null hypothesis is rejected for a given context and performance function, we know for which entity the context function can explain differences in performance. However, we do not know yet how these differences can be explained, as the analysis of variance tests indicate the existence of differences, but not their location. For example, the ANOVA can indicate that the resource responsible for executing an activity has an effect on its duration, but not which resources stand out. Post hoc analyses are typically used to perform multiple comparisons with the aim of finding which samples can be considered as distinct.

We use Tukey's range test [26], which compares the means of every sample to the means of every other sample. It identifies any difference between two means that is greater than the expected standard error. Like the single-factor ANOVA described in Subsect. 4.1, Tukey's range test assumes normality. As a non-parametric alternative (i.e. after applying the Kruskall-Wallis test) Nemenyi's distribution-free multiple comparison test (also known as Nemenyi-Damico-Wolfe-Dunn test) is used as a post hoc test [17]. Both tests do not require sample sizes to be balanced and correct for the multiple comparison problem [17].

Note that it is possible that the null hypothesis for the analysis of variance is rejected, but the post hoc test fails to reject all pairwise tests for a given critical value (e.g. because of the multiple comparison correction). In other words, across all samples there can be a significant difference in means, while between every pair of samples there is not. In this case, further automated or manual inspection is necessary. However, when this is the case, the discovered differences in performance are generally of low impact to the business process being analyzed.

4.3 Analysis of Results

After the automated context-aware process performance analysis has been performed, the results need to be interpreted. By showing the entities and context functions for which the most significant differences in performance have been found, we obtain a list of possible optimization points. However, statistical significance does not equal real-world impact. It might be the case that even though the means of two or more contexts are very different, their absolute values differ little. For example, consider a process with two activities: make scan (machine activity) and read scan (human activity). Some scanners are faster than others, and some people read faster than others. It is possible that the significance in differences between scanners is much higher than that between readers, even though usually, bulk of the time will be spent in reading a scan rather than making it. Thus, the importance of a performance difference is not determined only by its significance. It is important to determine the *impact* of a discovered difference. This can be done by implementing an impact formula that for example multiplies the significance of the difference with the absolute variance.

Once significant and impactful differences are discovered they can be transformed into performance insights in natural language or by visual representations such as those in Sect. 5. For example, sentences such as "activity X takes three times longer when resource R is involved in the preceding activity" or "cases often violate the maximum throughput SLA if the caseload is higher than 80 % at the time of activity X" can be constructed. Since the impact provides a natural ordering it is possible to, for example, only show the top 10 most impactful context-dependent performance differences.

4.4 Assumptions and Drawbacks

Automating the analysis of variance between different samples has some drawbacks. The existence of outliers can affect the test for normality. Filtering out outliers first can restore normality. Depending on the desired results of the context-aware performance analysis, outliers may need to be removed before testing for significant differences in performance for different contexts.

Besides outliers, sample sizes are most critical in determining the value of the automated test. This is related to the homogeneity of variance assumption of the analysis of variance tests. Since normality tests have little power with small data sets and can be too sensitive with large data sets, an automated approach might give false confidence of normality, and consequently the assumptions of the chosen analysis of variance test might be violated [16]. For example, this assumption is violated when sample sizes are very unbalanced. In this case, the null hypothesis is at risk of being falsely rejected. In other words, in case a certain context is very infrequent and/or there are big differences in the frequencies of different contexts, the results might falsely indicate (in)significant differences in performance. As such, the quality of the results depend on the size of the samples that are tested. The F-statistic used by this test is considered robust to the homogeneity of variance assumption when sample sizes are balanced [16].

Analysis of variance also assumes independence of observations. However, there might exist some relation between performance characteristics of different process entities. For example, it might be the case that a machine takes an extra 10 min every 100th task, or the duration of the task alternates between two values every 50 times. This however has no effect on the results of our hypothesis tests. In fact, the analysis of variance tests whether the samples are drawn from the same distribution. If the context function does not explain any difference in results, the test statistic will not be significant. In other words, taking the two examples mentioned before, each 100th task performed by the machine or the different durations will randomly reside in any of the samples, in case the context function does not describe those problems.

In conclusion, results of the automated approach need to be carefully interpreted before using them as basis for process performance statements. This can for example be done by visual comparison of the performance results. Nonetheless, the results provide a powerful basis for context-aware process performance analysis, and can provide important insights into root causes for performance problems such as bottlenecks or deviations from protocol and can be used for better scheduling of resources.

5 Case Study

The context-aware performance analysis approach described above has been implemented as an extensible analysis framework in the process mining tool ProM[2]. New process entities or context and performance functions can be easily

[2] See http://promtools.org.

added in order to analyze their effect on performance. We evaluate our technique on a publicly available, real-life event log. This event log was used so that results can be reproduced and compared. The aim is to demonstrate our approach rather than to exhaustively analyze the process recorded in this log. The dataset stems from a loan application process from a Dutch bank, and was originally used in the Business Processing Intelligence Challenge (BPIC) in 2012 [11]. The log contains 13,087 cases of a loan application process, for which in total 262,200 events have been recorded. There are 36 distinct activities and 69 different resources are involved in this process. There are 4,366 different control-flow variants.

In order to test our approach, we analyze the duration of activities with respect to their context. This is done by measuring the time between the events that represent the start and complete transitions for an activity. We look at the measurements from two different perspectives: the resource involved in the activity and the activities preceding the activity (prefix of the trace). In other words, one performance function and two context functions are used. Of course, other functions potentially leading to additional insights can be applied as well.

Applying the technique as described in Sect. 3 for both context functions and all 36 activities, results in 72 sets of measurements. Here, each measurement set represents the duration of a specific activity when analyzed for a specific context function, i.e. each set contains multiple samples of duration values representing a specific context (label). For each set, we analyze the variance between the set of samples using the statistical approach described in Sect. 4. The post hoc tests are used to discover exactly which context stands out.

For the prefix context function, the length-2 prefix of activities in their trace is taken. The result is abstracted to a set. In this way, we look at whether the last two activities (in any order) have an affect on the duration. Significant differences are found for two activities: “w afhandelen leads” and “w beoordelen fraude”. These results are shown in Figs. 4 and 5 respectively. For “w afhandelen leads”, it can be seen that if the activity is performed three times in a row, the duration is significantly higher than when it is preceded by “w beoordelen fraude” or “w completeren aanvraag”. Similarly, in Fig. 5, we can see that when “w beoordelen fraude” is preceded by “w completeren aanvraag” and itself, it takes significantly more time. From these results we might conclude that rework leads to an increase in duration. However, note that even though the differences in duration are significant, the impact in this case is limited to maximally several minutes difference, and as a result the differences might be negligible for the process owner.

Using the resource context function, we check whether the resource involved in the execution of an activity has an effect on its duration. A significant difference in duration is found for the activity “w completeren aanvraag”, as can be seen in Fig. 6. We can see that two resources (11079 and 11254) take considerably more time to perform the activity. In Fig. 7, we can see that the other resources generally take up to half an hour. In this case, the differences span several hours. This big difference might be due to the fact that the two resources handle difficult cases or that the activities they perform span multiple days.

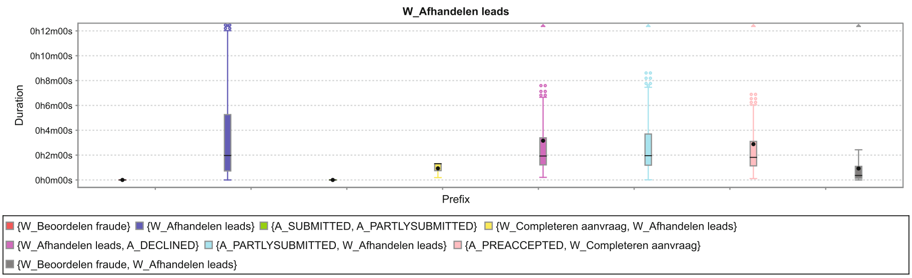


Fig. 4. Duration for the activity "W afhandelen leads" for different prefix-sets of length 2. The third consecutive execution of the activity is found to take significantly more time. The impact is in the order of several minutes.

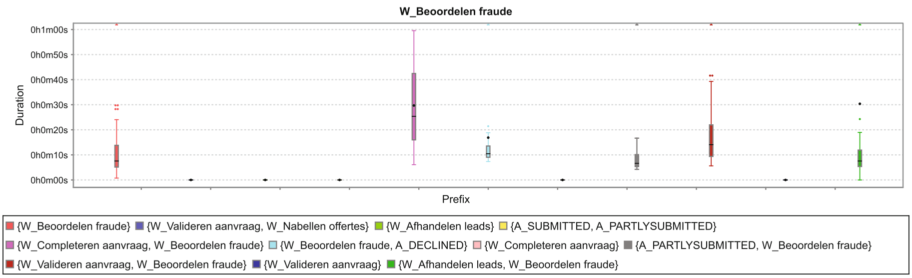


Fig. 5. Duration for the activity "W beoordelen fraude" for different prefix-sets of length 2. When the activity is preceded by "w completeren aanvraag" and itself, it takes significantly longer. The impact of the discovered differences is limited to several seconds.

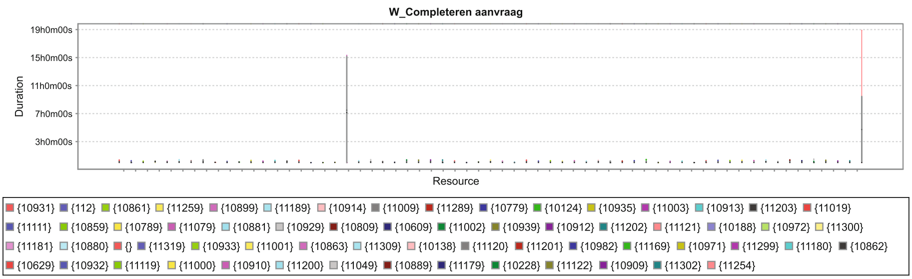


Fig. 6. Duration for the activity "W completeren aanvraag" for different resources. Two resources are found to take significantly more time. An impact of several hours can be seen.

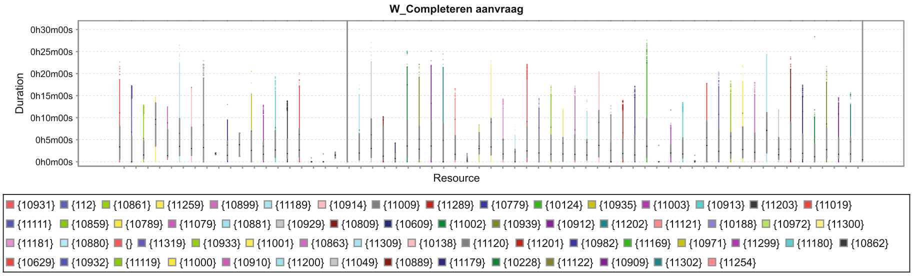


Fig. 7. Figure 6 zoomed in to show differences between resources. Most other resources take a comparable time to execute "W completeren aanvraag".

6 Related Work

As mentioned, relatively little work has been done in automatically analyzing process performance characteristics. In general, related work can be divided in the following groups: research on performance characteristic calculation, which aims at proposing new performance measurements and metrics, analysis, which aims to find root causes for performance issues, and prediction, which tries to predict remaining waiting times, cycle times, etc.

In [15], the authors approximate the cycle time of processes based on queuing theory, using expected times and process structure. However, no context is used in the prediction process. Techniques as [5,13,14] focus on context-aware performance predictions by applying a clustering approach, where different context-related scenarios relate to separate prediction models. Different clusters of behavior can be discovered for which different prediction models are created. New cases can then be compared to all clusters, and predictions can be obtained from the most closely related cluster. However, in these techniques, the context is limited to a representation of the features (attributes) of cases or events. Also, no rules or descriptions are given as to what differences exist between the clusters. In [14], the prediction is restricted to predicates that can be evaluated over completed cases.

Different techniques have been proposed to predict the remaining running time of processes [4,6,20,22,24]. For example, [24] uses stochastic Petri nets with arbitrary firing delays as a basis for prediction. In [22], a technique to learn a prediction model is proposed that can predict performance characteristics such as remaining waiting time, but also the following activity or resource can be predicted. Though having their merits and specific use cases, these approaches heavily rely on a process model to be present. In flexible process environments, these models are difficult to obtain and change continuously, restricting the use of these techniques.

Simulation can be used to analyze process performance characteristics. In [1], the authors highlight possibilities. The downsides of simulation is that it is often difficult to mimic the steady-state behavior in flexible processes. As a result, analyzing the effects of changes becomes infeasible. Also, contextual properties of process characteristics underlying performance differences are not discovered.

In [19], the authors aim to predict behavior based on classification of event- and case attributes using decision trees. Though contextual properties can be used, the user needs to specify the dependent and independent variables in order to perform a single analysis. The interpretation of the significance of the result is left to the user as well. As a result, multiple analyses need to be performed and interpreted to find root causes for performance issues.

Most related to the technique used in this paper is the approach proposed in [12], where non-parametric regression is used to predict remaining cycle time, activity durations or attribute values. Here, however, contextual properties of process entities are not considered yet. As such, no definite description of what is causing performance differences can be given.

In our technique, performance differences are explained by the contextual information underlying those differences. Furthermore, our approach does not depend on a process model, and thus can be applied to both structured as well as flexible processes where no process description is present. We purposely utilize broad definitions for context and performance functions in order to generalize our analysis approach. However, more detailed formalizations can be found in literature and can be used to further clarify, formally define, and represent context and performance functions. For example, in [9], the authors provide an approach to characterize the context of a process in a given domain through conceptual models structured in layers. Here, both internal and external context is included and the relationship between entities is formalized. In [10] a meta-model is proposed that can be used to unambiguously define process performance indicators that are amenable to automated analysis. Techniques such as these can be used in conjunction with our approach to further automate the analysis of process performance.

7 Conclusion

Most existing process mining techniques focus on process discovery and conformance checking. Relatively little research has been performed on the analysis of business process performance. Performance characteristics of process entities such as events, activities, cases, resources, or entire processes typically highly depend on their context. This paper has introduced a novel approach to analyze key process performance indicators by considering the *process context*. We have introduced a generic framework that aids in discovering significant differences in performance results and their causes. Process entities are assigned descriptive context labels by applying context functions to them. Statistical hypothesis testing is used to verify whether a context label explains a significant difference in performance. In other words, using this technique, the effect of any context on KPIs can be automatically analyzed. Insights can be gained on which contextual properties of process entities have an effect on the key performance indicators of business processes. As such, root causes for delays, bottlenecks, deviations to protocol and violations of service level agreements, etc. can be discovered.

Even though a generic framework was introduced, often specific context functions need to be created to analyze real-life processes. To this end, several context functions have been mentioned. Sometimes, however, the contexts underlying performance differences are domain-dependent and therefore have to be defined by the process analyst. To this end, we have implemented the approach as a generic and extensible framework in the process mining tool ProM. Also, as described in Subsect. 4.3, careful interpretation of the results of the automated analysis technique is essential. In order to better assist the analyst in providing performance optimizations, more work is needed to analyze the impact of discovered differences in performance. Information on significant performance differences can be used as input for better prediction techniques or can be used in a streaming data monitoring setting, where states of alert are reached once differences become significant.

Besides using only event data, information obtained from process models can aid in providing more accurate measurements. In case of parallelism for example, process models can help to identify simultaneous activities. Furthermore, process models can be used when timing information stored in the event log is imprecise or (partially) missing. Using alignments, process models and event logs can be combined to show conformance and performance information. However, how to use process models to better calculate performance is outside the scope of this paper. More information on this topic can be found in [2,3,7,18]. In the future we would also like to look into how to visualize the results in different ways. For example, the most impactful differences in performance can be translated in natural language. Process models can also be extended with information on performance issues and their root causes.

References

1. van der Aalst, W.M.P.: Business process simulation revisited. In: Barjis, J. (ed.) Enterprise and Organizational Modeling and Simulation. LNBIP, vol. 63, pp. 1–14. Springer, Heidelberg (2010)
2. van der Aalst, W.M.P.: Process Mining - Data Science in Action. Springer, Heidelberg (2016)
3. van der Aalst, W.M.P., Adriansyah, A., van Dongen, B.F.: Replaying history on process models for conformance checking and performance analysis. Wiley Interdisc. Rev. Data Mining Knowl. Disc. **2**(2), 182–192 (2012)
4. van der Aalst, W.M.P., Schonenberg, M.H., Song, M.: Time prediction based on process mining. Inf. Syst. **36**(2), 450–475 (2011)
5. Bevacqua, A., Carnuccio, M., Folino, F., Guarascio, M., Pontieri, L.: A data-driven prediction framework for analyzing and monitoring business process performances. In: Hammoudi, S., Cordeiro, J., Maciaszek, L.A., Filipe, J. (eds.) Enterprise Information Systems. LNBIP, vol. 190, pp. 100–117. Springer, Heidelberg (2014)
6. Bolt, A., Sepúlveda, M.: Process remaining time prediction using query catalogs. In: Lohmann, N., Song, M., Wohed, P. (eds.) BPM 2013. LNBIP, vol. 171, pp. 54–65. Springer, Heidelberg (2014). doi:10.1007/978-3-319-06257-0_5

7. Jagadeesh Chandra Bose, R.P., Aalst, W.: Trace alignment in process mining: opportunities for process diagnostics. In: Hull, R., Mendling, J., Tai, S. (eds.) BPM 2010. LNCS, vol. 6336, pp. 227–242. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15618-2_17
8. Box, G.E., Cox, D.R.: An analysis of transformations. J. R. Stat. Soc. Ser. B (Methodological) **26**(2), 211–252 (1964)
9. da Cunha Mattos, T., Santoro, F.M., Revoredo, K., Nunes, V.T.: A formal representation for context-aware business processes. Comput. Ind. **65**(8), 1193–1214 (2014)
10. del-Río-Ortega, A., Resinas, M., Cabanillas, C., Cortés, A.R.: On the definition and design-time analysis of process performance indicators. Inf. Syst. **38**(4), 470–490 (2013)
11. van Dongen, B.F.: BPI challenge 2012 (2012). http://dx.doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f
12. Dongen, B.F., Crooy, R.A., Aalst, W.M.P.: Cycle time prediction: when will this case finally be finished? In: Meersman, R., Tari, Z. (eds.) OTM 2008. LNCS, vol. 5331, pp. 319–336. Springer, Heidelberg (2008). doi:10.1007/978-3-540-88871-0_22
13. Folino, F., Guarascio, M., Pontieri, L.: Context-aware predictions on business processes: an ensemble-based solution. In: Appice, A., Ceci, M., Loglisci, C., Manco, G., Masciari, E., Ras, Z.W. (eds.) NFMCP 2012. LNCS (LNAI), vol. 7765, pp. 215–229. Springer, Heidelberg (2013). doi:10.1007/978-3-642-37382-4_15
14. Francescomarino, C.D., Dumas, M., Maggi, F.M., Teinemaa, I.: Clustering-based predictive process monitoring. CoRR abs/1506.01428 (2015)
15. Ha, B.-H., Reijers, H.A., Bae, J., Bae, H.: An approximate analysis of expected cycle time in business process execution. In: Eder, J., Dustdar, S. (eds.) BPM 2006. LNCS, vol. 4103, pp. 65–74. Springer, Heidelberg (2006). doi:10.1007/11837862_8
16. Hill, T., Lewicki, P.: Statistics: Methods and Applications: A Comprehensive Reference for Science, Industry, and Data Mining. StatSoft Inc., Tulsa (2006)
17. Hollander, M., Wolfe, D.A., Chicken, E.: Nonparametric Statistical Methods. Wiley, New York (2014)
18. de Leoni, M., van der Aalst, W.M.P.: Data-aware process mining: discovering decisions in processes using alignments. In: Shin, S.Y., Maldonado, J.C. (eds.) Proceedings of the 28th Annual ACM Symposium on Applied Computing (SAC 2013), Coimbra, Portugal, 18–22 March 2013, pp. 1454–1461. ACM (2013)
19. de Leoni, M., van der Aalst, W.M.P., Dees, M.: A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. Inf. Syst. **56**, 235–257 (2016)
20. Maggi, F.M., Francescomarino, C., Dumas, M., Ghidini, C.: Predictive monitoring of business processes. In: Jarke, M., Mylopoulos, J., Quix, C., Rolland, C., Manolopoulos, Y., Mouratidis, H., Horkoff, J. (eds.) CAiSE 2014. LNCS, vol. 8484, pp. 457–472. Springer, Heidelberg (2014). doi:10.1007/978-3-319-07881-6_31
21. Marques de Sá, J.P.: Applied Statistics Using SPSS, STATISTICA, MATLAB and R. Springer, Heidelberg (2007)
22. Pravilovic, S., Appice, A., Malerba, D.: Process mining to forecast the future of running cases. In: Appice, A., Ceci, M., Loglisci, C., Manco, G., Masciari, E., Ras, Z.W. (eds.) NFMCP 2013. LNCS (LNAI), vol. 8399, pp. 67–81. Springer, Heidelberg (2014). doi:10.1007/978-3-319-08407-7_5
23. Razali, N.M., Wah, Y.B.: Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests. J. Stat. Model. Analytics **2**(1), 21–33 (2011)

24. Rogge-Solti, A., Weske, M.: Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays. In: Basu, S., Pautasso, C., Zhang, L., Fu, X. (eds.) ICSOC 2013. LNCS, vol. 8274, pp. 389–403. Springer, Heidelberg (2013). doi:10.1007/978-3-642-45005-1_27
25. Shapiro, S.S., Wilk, M.B.: An analysis of variance test for normality (complete samples). Biometrika **52**(3/4), 591–611 (1965)
26. Tukey, J.W.: Comparing individual means in the analysis of variance. Biometrics **5**(2), 99–114 (1949)