Examining Case Management Demand using Event Log Complexity Metrics

Marian Benner-Wickner, Matthias Book, Tobias Brückmann, Volker Gruhn
paluno – The Ruhr Institute for Software Technology
University of Duisburg-Essen
Essen, Germany

Abstract— **One of the main goals of process mining is to automatically discover meaningful process models from event logs. Since these logs are the essential source of information for discovery algorithms, their quality is of high importance. In recent years, many studies on the quality of resulting process models have been conducted. However, the analysis of event log quality prior to the generation of models has been neglected. For example, yet there are no metrics which can measure the degree of event log quality that is needed so that discovery algorithms can be applied. Especially in the context of case management (CM) where processes are less structured, complex event logs reduce the effectiveness of the process discovery. In order to avoid mining impractical "spaghetti processes", it would be convenient to measure the event log complexity prior to discovery steps. In this paper, we provide our research results concerning the design and applicability of such metrics. First of all, they shall help to indicate whether the event log quality is sufficient for traditional process discovery. In case of very poor quality, they indicate the demand for more agile techniques (e.g. adaptive CM or agenda-driven CM).**

***Keywords*— metrics, process mining, case management**

I. INTRODUCTION

In recent years, a growing community of researchers has aimed to bring process mining techniques into practice. Experiences have been made in areas with predominantly structured processes (also referred to as lasagna processes [1]), as well as in areas with ad hoc structured (case management, CM) processes. Examples for the former functional areas are finance/accounting process in industry [2] and administrative processes in municipalities [3]. CM-like processes are found in product development [4], customer support or medical healthcare [5]. For quite a while already, the process mining community discussed the quality of resulting models, and how real-life logs can be preprocessed in order to satisfy the often challenging requirements of discovery algorithms when exposed to event logs from CM processes. Various algorithms like the Genetic Algorithm [6], the Fuzzy Miner [7] and the Cluster Miner [8] have been developed to cope with incompleteness and noise in event logs, each resulting in different models with their individual representational bias. To discuss and compare the quality of process models, a wide range of metrics have been elaborated [3, 9, 10].

However, the discussion about process model quality somewhat distracts from the more basic problem of process mining techniques, as described in the process mining manifesto [11]: the quality of event logs. Therefore, it is necessary to review the quality of event logs before proceeding in the process mining workflow (also known as the L* life-cycle model as introduced by van der Aalst in [1]). Especially when the targeted processes are complex, the event log has to be carefully analyzed before applying discovery algorithms or conformance checks. That is because the calculation of process models is expensive when the resulting model is spaghetti-like.

The main contribution of this paper is a set of event log metrics that can be used to evaluate the event log complexity. To be exact, the metrics are used to help analyzing event logs in a methodical way by calculating the so-called event and trace diversity. Together, they answer three important questions: Which event log simplicity is needed to be ready for process discovery? In case of poor quality, is there a chance that preprocessing enables proceeding with process mining? And finally, how to detect whether traditional control-flow perspectives cannot be mined? In the latter case we believe the metrics are able to indicate the need of additional CM techniques at runtime such as the agenda-driven CM [12] in order to mine more reasonable perspectives like the data perspective. Data mined by those techniques should improve the quality of future process mining results in the context of CM-like processes.

This paper is structured as follows: Section II provides information on how we define the complexity characteristics of CM processes. Section III introduces complexity metrics for event logs. Section IV shows results from experiments on both artificial and real-life event logs. Section V gives an overview of related work on metrics in process mining, and Sect. 6 outlines the next steps in our research.

II. CONNECTING CASE MANAGEMENT PROCESS CHARACTERISTICS WITH COMPLEXITY

Before introducing metrics that can indicate whether an event log originates from a process similar to those conducted in CM, we need to make some assumptions: First, we assume that there are CM process characteristics different to structured processes. Second, the main differences between both types can be expressed by what we will henceforth refer to as complexity. As a consequence, the definition of case management characteristics goes hand in hand with the definition of complexity. In this chapter, we will go in detail on how we define event log complexity in the context of CM processes.

We identified two viewpoints of event log complexity: First, we consider the complexity of each trace (i.e. sequence of events recorded during a case) individually. To put simply, a complex case is expected to contain many events, whereas a simple case should contain less events. Even though this approach may be straightforward and easy to compute, it does not reflect that long traces can have two different origins: monotonous repeating of the same events on the one side and sequences of several distinct events on the other side. Hence, when measuring complexity concerning length, the amount of reoccurring behavior has to be analyzed in more detail. This can help to identify whether the trace length has been 'artificially' protracted by routines. Since this viewpoint of complexity concentrates on the heterogeneity of events, we will henceforth refer to it as **event diversity.**

The second viewpoint of event log complexity is taken on a more comprehensive perspective. Instead of focusing on the diversity of events within a trace, it focuses on the diversity between traces. So we will refer to this viewpoint as **trace diversity**. Since CM processes tend to have many variants, these process variants differ widely and fan out early in the process, resulting in a high degree of control flow nondeterminism. This can be measured by comparing the similarity between all traces. If most traces differ – provided that a considerable number of instances have already been recorded in the past – then it is likely a CM-like process has been mined. Please note that, as one can see by the constraint, this characteristic strongly correlates with what is also known as "completeness" (see [13]): A CM process will, if at all possible, need considerably more instances to be complete than a structured processes.

Neither event diversity, nor trace diversity considered individually indicate CM processes. Of course, many structured processes can have a high event diversity to some extent. If it can be discovered entirely in the event log (i.e. the log is complete), this complexity is manageable by applying trace clustering techniques [14], for example. On the other hand, trace diversity taken by itself may indicate scoping issues precipitated during event log extraction; maybe events originate from another process instead of the target process. In both cases individually, preprocessing steps may be beneficial in order to adjust the event log complexity and to resume the original process mining workflow. But if both aspects of diversity coincide, we can certainly refer to CM processes.

In the following section, we will introduce metrics that can be used to extract process characteristics out of event logs, especially those metrics that can identify processes similar to those in CM.

III. Complexity Metrics for Event Logs

Following the process mining workflow, one would just apply mining first and then see whether the results are good enough to proceed in the process mining workflow. In this paper, we want to provide an alternative that measures the complexity of event data before applying expensive discovery algorithms. Thereby, we hope to avoid mining impractical spaghetti processes when dealing with CM processes. In this section, we want to contribute metrics that highlight the different viewpoints of process complexity introduced above. The basic requirement for all of the metrics is the capability to jointly gain insights about the complexity of a process based on its event logs. Please note that, as with most metrics, measurements considered individually are not convincing. So they should always be seen as a set and any result should be carefully analyzed in its specific context.

As a starting point, we will define straightforward metrics and will then refine and assemble them to more comprehensive ones. We begin with three event diversity metrics, namely the average trace length and the average trace size which are both combined into the event density. Then we introduce two metrics that focus on trace diversity.

A. Event Diversity

Consider a typical rehabilitation management process where case managers conduct patients from the occupational accident till integration. They initiate various different therapies and interact with several stakeholders (e.g. physicians, health care professionals or human resource managers). So size of the log by means of the mean number of events over all traces in a log should give insight about the process size.

Definition 1.1 (Average trace length). *Let k be the number of traces within the event log L with k > 0. For each trace i (1 <= i <=k), n_i is the number of events it contains (not to be confused with the number of distinct events, henceforth referred to as event classes). The average trace length is then defined as follows:*

$$\mathrm{atl}(L) = \frac{1}{k}\sum_{i=1}^{k} n_i\ .$$

On the contrary, this metric is very simple and does not reflect the structure of the process. For example, if many traces within the log would contain a similar sequence of activities, the process may be long, but not necessarily complex. So this metric should be complemented with other (compound) metrics discussed below in order to make more convincing statements about process complexity.

For example, one may also only consider the mean number of distinct event classes within each trace. The corresponding metric is more robust against large logs that seem to be complex at first sight, but are actually flooded with recurring events, e.g. because of loops in the actual process. We therefore propose a modified version of the average trace length which measures its "size" in terms of the average number of event classes per trace:

Definition 1.2 (Average trace size). *Let k be the number of traces within the event log L. For each trace i (1 <= i <=k), E_i is the set of event classes it contains. The average trace size is defined as follows:*

$$ats(L) = \frac{1}{k}\sum_{i=1}^{k} |E_i|$$

Please note that before comparing event logs according to this metric, logs need to be normalized on a comparable level of abstraction. For example, take into account that not all logs may contain lifecycle transition events (start, complete).

As briefly discussed in section II, when analyzing business processes in general, loops indicate complexity since they dramatically increase the solution space and may harm the possibility to terminate. However, keep in mind that they do not directly **enforce**, but only **allow** for complex behavior, and that in context of process mining we can observe the real behavior in event logs. As a consequence, the complexity resulting from loops is already covered via the average trace length. With that in mind, we can think of low event diversity as some aspect of monotonicity. Because CM processes usually comprise of various alternatives, we introduce event diversity as an indicator.

Definition 1.3 (Event density). *Recall that atl(L) is the average trace length and ats(L) is the average trace size. Then the event density is defined as follows:*

$$ed(L) = \frac{ats(L)}{atl(L)}$$

This metric relates the average number of distinct event classes over all traces to the average number of events over all traces. If the traces mostly contain unique behavior, then atl(L) is just slightly larger than ats(L). Thus the relation converges to 1 (high event density). In return, when loops cause a high value of atl(L), the *average trace size* stays the same and the metric tends to 0 (low event density). Choose this metric to find out whether the size of a log (by means of the average trace length) arises from sustained sequences or just from recurring behavior caused by loops.

B. Trace Diversity

As already explained in section II, CM processes comprise of many variants, which, at worst, never can be entirely observed. This results in constantly incomplete event logs. However, the captured variants appear in the log as dissimilar traces. So measuring trace diversity should provide a useful indication of CM processes. To achieve this, we took a closer look at other metrics. We came to the conclusion that it is valuable to consider the overall probability of whether a particular event class can be observed in any trace. This matter can be described by relating the average trace size to the total number of event classes. We introduce the metric as the *simple trace diversity* of the log.

Definition 2.1 (Simple trace diversity). *Let n be the total number of event classes (with n > 0), and recall that ats(L) is the average trace size of log L. The simple trace diversity is defined as follows:*

$$std(L) = 1 - \frac{ats(L)}{\mathrm{n}}.$$

Imagine that each event class has been observed in almost every trace (ats(L) ≈ n). In this case, there are no variants (except for the order of occurrence) and the metric indicates low trace diversity (std(L) ≈ 0). Instead, if just a fraction of events has been observed (ats(L) << n), then the trace diversity is high (std(L) ≈ 1).

As stated above, the event coverage does not consider the order of occurrence. Imagine an event log in which each trace covers all event types. Of course, such an event log has high event coverage, suggesting it originates from a structured process. But what if every trace is executed in a different order of occurrence? This should point out a high variability that is specific to CM processes. So there is a need for a metric that can describe in which degree the order of events differs. Fortunately, trace clustering techniques use metrics like the Hamming or Levenshtein distance to group similar trace parts. We adopt this approach and apply it on measuring mutual dissimilarity between all traces.

Definition 2.2 (Advanced trace diversity). *Let k be the total number of traces and LD(T_i,T_j) be the levenshtein distance between two traces and T_i, T_j. Recall that atl(L) is the average trace length of L. The advanced trace diversity is defined as follows:*

$$atd(L) = \frac{2}{k \times (k-1) \times atl(L)} \sum_{i=1}^{k} \sum_{j=i+1}^{k} LD(T_i, T_j)$$

We preferred Levenshtein to Hamming distance, because the latter can only measure distance by counting replacements between strings. It is not able to compare two strings with different length. Details about both distance measures can be found in literature [15, 16].

First, the metric sums up all LD values mutually among all traces. To avoid that logs with long event names have a higher distance compared to logs with short names, we have to abstract from event names length. In this definition we argue to assign a Unicode character to each event class and replace the event names using the Unicode characters. This works well as long as the amount of event classes does not exceed 65.535 (2^{16}-1). By way of comparison, the most complex real-life event log introduced in section IV.C comprises 624 event classes, so we believe that this is an appropriate limit.

However, in order to get an average LD value over all traces, we also have to divide the result by the number of summands, which is $\frac{\mathrm{k \times (k-1)}}{2}$. This results in the mean LD between any two traces of the log, which in fact should already be a quite useful metric to express log complexity. Nevertheless, remember that the range of the LD scales between 0 and the maximum length of the compared character sequences. Hence, to avoid that the metric is biased when comparing logs with different average trace lengths, it also reflects the metric atl(L).

The final metric value can be interpreted as the mean extent of modifications necessary to transform an arbitrary trace to any other trace within the log. For example, consider a log containing five traces with each five different event classes {<a, b, c, d, e>, <f, g, h, i, j>, <k, l, m, n, o>, <p, q, r, s, t>, <u, v, w, x, y>}. Since every event is different, the mean LD value is 5. To abstract from trace length, we relate this value to the average trace length, which is also 5. This results in an advanced trace diversity value of 5/5 = 1, i.e. 100% of the events in average have to manipulated between any two traces.

Unfortunately this metric does not scale between 0 and 1. For example, consider a slightly different log containing the traces <a, b, c, d, e>, <f, g, h, i, j, 1>, <k, l, m, n, o, 2, 3>, <p, q, r, s, t, 4, 5, 6> and <u, v, w, x, y, 7, 8, 9, 0>. Each trace of the previous log has been extended using different numbers, resulting in different trace lengths. In this particular example, the mean LD value is 8, resulting in an advanced trace diversity of about 1.14. So any atd value above 1.0 also indicates that the traces are not only very different by means of the events used, but also very different in length. We refer to the next section for detailed experiments with this metric.

Besides the trace diversity, modularization is also one important metric in business processes [17]. To get an impression of this in our domain, one could apply the analysis of fan-in and fan-out (as introduced by Henry and Kafura [18]) to business processes as well. However, since modularization affects trace diversity as well, we refrain from introducing a dedicated metric for this specific kind of diversity.

IV. Experimental Evaluation

To evaluate the validity of the metrics described above, i.e. to check whether they measure the aspects of complexity presented in section II, we conducted several controlled experiments with different artificial and real-life data sets. In subsection IV.A, we describe the experiment design, and present our results in subsection IV.B. Based upon these, in subsection IV.C we outline the findings from a case study with three publicly available real-life event logs. Therein we will discuss whether the metrics can indicate poor event log quality that is insufficient for traditional process discovery – which in turn would indicate demand for case management techniques like adaptive CM.

A. Metrics Experiment Design

First, we generated four artificial processes. We carefully paid attention to use only those processes for evaluation that highlight one distinct aspect of each metric. Then we simulated these processes and recorded almost 100 MB of event log data containing 6000 traces with 51,242 events. The event data has been generated using the PLG Framework [19] and CPN Tools [20] (see Table I for the log details).

Please note that, since our metrics do not measure noise caused by undesired infrequent behavior, the generators did not include such noise into the event logs in order to avoid skewing the results of this first step. The real-life log we examined in the second step (subsection IV.C) naturally contains noise.

TABLE I. Details of the example logs used for evaluation

ID	Description	Traces	Events	Event classes
L_1	Long sequence, no loops	1000	13 000	14
L_2	Long, because of loops	1000	13 034	9
L_3	Short, many variants	1000	6 166	38
L_4	Long and complex	1000	11 042	20

Before outlining our results, we will explain each artificial process in more detail. First, consider the event log $L_{1.}$ It has been generated in order to test how the metrics behave on a long but very simple event log with hardly any process variants (see top process on Fig. 1). They should indicate that the event diversity is very high due to the process length and event density. However, they shouldn´t indicate trace diversity due to the lack of process variants.

Event log L_2 has been generated to check if our metrics can indicate whether or not the length of a process is caused by recurring events. The process contains a short sequence with an arc from the last to the first activity. The metrics should reflect this feature through a low degree of event density.

The last remaining metric that has to be dealt with is the trace diversity. Since it shall reveal a high amount of process variants within a given log, we need an artificial process that spans many alternative paths straight from the beginning. We generated log L_3 from such a process. Metrics of such a log should not indicate any loops through a low event density value. In addition, let us point out two further interesting aspects of this process: (1) although this process seems to be extensive at a first glance; its traces will be surprisingly short. (2) The process contains a lot of different events; however, event density will be very high since the event density only takes into account the diversity of events within each trace individually.

As opposed to the logs described before, event log L_4 has been generated to test both aspects of diversity: It is designed to check whether (1) the existence of loops and (2) many process variants can be indicated by the metrics, without interference.

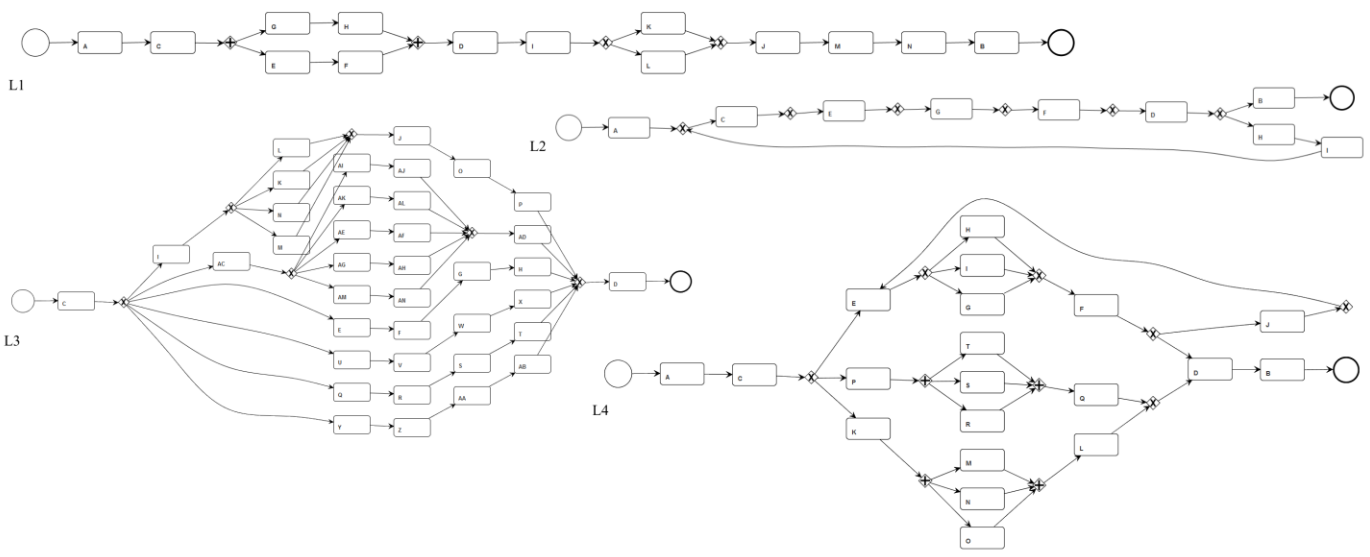

Fig. 1. Artificial processes that produced event logs L1 (top) to L4 (bottom)

B. Artificial Log Results

In the following, we will present and discuss the experiment's results after applying all metrics to the artificial logs stemming from the processes introduced above. We will check whether the metrics can reveal the process characteristics needed to decide how to continue in the process mining workflow. Table II provides a condensed view of the metric values calculated for each log listed above.

TABLE II. COMPLEXITY METRICS FOR THE EVENT LOGS IN FIG. 1

Log	atl	ats	ed (%)	std (%)	atd (%)
L_1	13	13	100	7	4
L_2	13	7	54	22	15
L_3	6	6	100	84	15
L_4	11	8	73	60	18

At first, we will discuss the results for log L_1: The metrics indicate a maximum *event density*, since there is no recurring behavior. On the other hand, both trace diversity metrics show that all traces are very similar. More than any other artificial process in this paper, this is a structured process and it is clear that the discovery of control flow perspectives is possible without any process mining preprocessing steps.

The findings from log L_2 show a very low event diversity which confirms that the relatively high *average trace length* (atl = 13) is caused by recurring behavior due to the loop, not via sequences of distinct activities. As expected, they also show that the loop to some extent causes different traces, which is indicated by trace diversity values of 22 % and 15 %.

We also applied the metrics on log L_3 as an example for a log containing traces that are very different, but of equal length. The results are promising, since each desired characteristic has been disclosed by the metrics: Firstly, the traces indeed are very short (atl = 6), and the *event density* confirms that there is no recurring behavior present in the log. Secondly, this log seems to contain the highest *simple trace diversity* amongst the other artificial logs (std = 84 %). However, it is very interesting that the *advanced trace diversity* (atd = 15 %) is equal to the previously examined log. This is a perfect example to show how the atd metric behaves in certain conditions: When applied to logs containing few events (resulting in a low *event density*) but highly variable trace lengths (e.g. due to loops), the atd metric provides the same results as if applied to logs containing many different events (resulting in a high *event density*) but constant trace lengths.

To conclude the results for log L_3, in conjunction with the short length, the metrics indicate that it would be hard to mine for a common process model. Whether process mining preprocessing steps (e. g. scoping) can help will heavily depend on whether the log is complete or not. If not, these values indicate the application of CM techniques.

Finally, we will discuss the results of the experiment on the artificial event log L_4. It aims to test the metrics regarding more than one aspect of complexity. The metrics correctly show that the process is relatively long and that there is some recurring behavior causing less *event density*. The process variants, both from the loop as well as the XOR splits, have caused a *simple trace diversity* of 60 % and an *advanced trace diversity* of 18 %. These results are satisfying, because they properly reflect the increased complexity compared to the previous logs in all three aspects: many different events, different trace lengths and many alternatives.

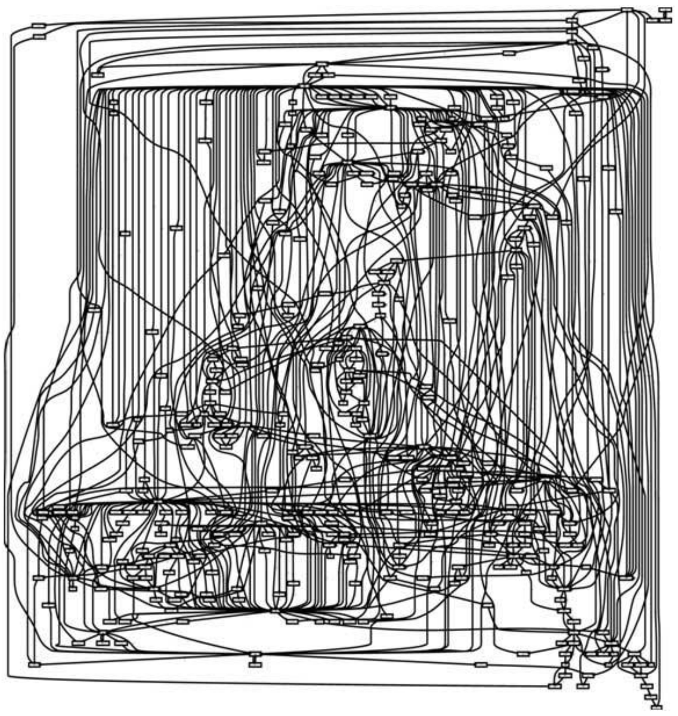
Fig. 2. Process model for event log L_5, as depicted in [1]

C. Real-Life Log Results

Of course, applying the complexity metrics on event logs from artificial processes can only be first step towards evaluation. So we also conducted experiments with real-life event logs that are publicly available. In this paper, we draw upon datasets provided by the IEEE task force on process mining (see http://data.3tu.nl/repository/collection:event_logs).

The first real-life event log (henceforth referred to as L_5) originates from a case study conducted at a Dutch hospital by Mans et al. [5]. It has already been used in several studies as an example for a CM process. Due to the log complexity, it is not possible to mine a meaningful process model without extensive preprocessing steps (see Fig. 2). We refer to the publication cited above for more details (e.g. efforts in mining different perspectives) and continue with the measurement results (see table III).

As the *average trace length* (atl) shows, the traces in this log contain many events. However, when relating this metric to the *average trace size* (ats) using the *event diversity* metric (ed = 25%), we can also see that 75% of the trace length is caused by recurring activities. The *simple trace diversity* (std) indicates that – in average – about 95% of the events are not observed in every trace. In addition, the *advanced trace diversity* (atd = 37%) calculated using the Levenshtein distance also exposes the highest trace dissimilarity measured in all experiments. Comparing the results with the process model depicted in Fig. 2, we can conclude that the metrics reflect the complexity pretty well. Although the length is caused by recurring behaviour to a high extent, it should be considered whether the process would benefit from the application of CM techniques at runtime due to its extraordinary absolute length and trace diversity values.

TABLE III. DETAILS AND METRICS OF THE REAL-LIFE LOGS

Log	Description	Traces	Events	Event classes
L_5	Dutch hospital ad-hoc structured	1143	150 291	624
L_6	Loan application explicitly structured	13 087	164 506	23
L_7	Volvo incidents ad-hoc structured	7554	65 533	13

Log	atl	ats	ed (%)	std (%)	atd (%)
L_5	131	33	25	95	37
L_6	20	12	60	67	17
L_7	9	4	44	69	19

The second real-life event log (L_6) has been identified by the authors as "explicitly structured" (see Fig. 4) and originates from a workflow system supporting loan applications. Most of the traces are short (atl = 20) and contain a moderate amount of reoccurrences (ed = 60%). Also the *advanced trace diversity* (atd = 17%) is as low as expected for a structured process. However, as the *simple trace diversity* shows, 67% of the events are not observed in every trace. To sum up, the metrics show that there is a fair chance to mine a process model that fits to most cases, though the process may contains many activities.

The last real-life log (L_7) comes from an incident management system of the private company Volvo IT and has been identified as "ad-hoc structured". According to the average trace length and the event density, the traces are very short and moderately filled with re-occurring events. However, both trace diversity metrics are slightly higher than the less complex log discussed before. At this point, the metrics are not able to indicate a CM process, since neither the event diversity nor the trace diversity values are prominent. As a result, it should be possible to mine a meaningful process model from this log. We followed this advice and applied the heuristics miner algorithm [21] to the event log (see Fig. 3). A detailed look on the process shows that it is similar to what van der Aalst defined as "flower model" [1]. On a first glance, such a model seems to be simple, but in fact it allows for many different logs by frequently switching between the leaves and the center node (here: "Accepted, In Progress"). To conclude, the metrics fail to detect a process that has been identified as ad-hoc structured. However, it was easy to mine a model, so that the main goal of the metrics (avoid effort in mining impractical models) can still be

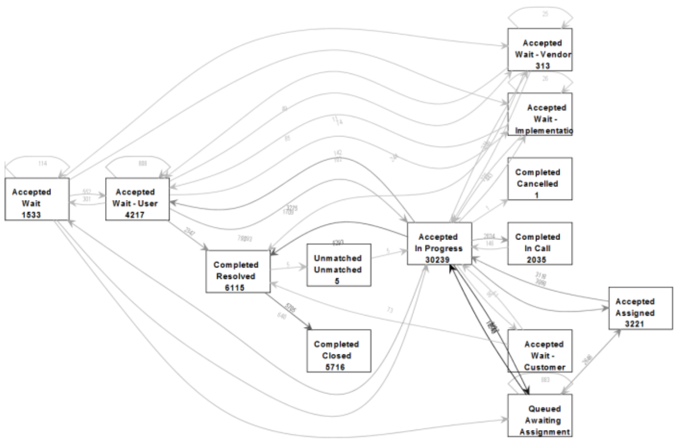

Fig. 3. Process model for event log L_7

upheld. The definition of ad-hoc structured processes as defined in [22] (*"At any time the process model for a case is structured, but the model can be changed while the case is being handling."*) indeed also fits to the case management process characteristics definition used in this paper. However, we argue that an event log containing very few (here: 4) distinct event classes per trace in average may be ad-hoc, but certainly not complex.

V. RELATED WORK

Metrics play an important role in the analysis of business processes. Both metrics for process models and event logs have been developed in recent years. For example, Guenther and van der Aalst [7] defined three types of log-based process metrics: unary and binary significance and binary correlation. However, besides the unary significance, these metrics are calculated using information about the relationship between pairs of events in a graph. So in fact, they are not directly based on event logs but on complex models derived from the logs.

Van der Aalst et al. propose to use aggregation and abstraction concepts to cope with complexity [23]. But abstraction by means of removing less significant events is difficult when there is high event diversity in the logs. That is because the event significance is spread almost evenly, leading to tricky clustering decisions. In most real-life processes, we see no clearly recognizable cut between significant and less significant events. Instead, we believe that the aggregation of events is more beneficial. Aggregated process parts can be supported specifically by agenda-driven CM without having to define control flow in detail.

In addition to the metrics used in this paper, there are more metrics that can be applied on event logs. First, Yang et al. [13] describe a metric and estimators for completeness of event logs. They monitor the occurrences of new classes during the analysis of possible trace classes with similar behavior. Based on this saturation curve, they estimate the total amount of clas-

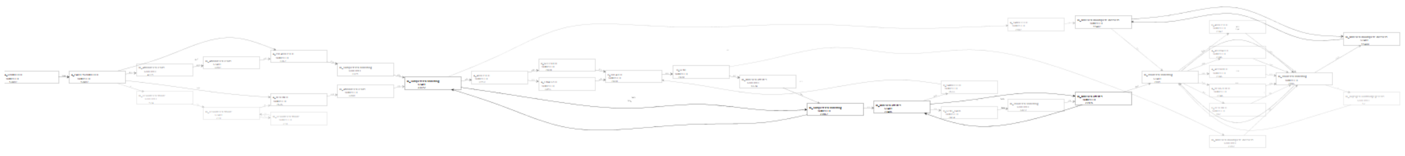

Fig. 4. Process model for event log L_6

ses. In CM, one of the assumptions needed for those estimators cannot withstand process complexity: The number of trace classes may not be finite, because there are too many possible variants e.g. in a patient's road to recovery. We propose to follow agenda-driven CM techniques to cope with this problem domain.

In his process mining book [1], van der Aalst terms two metrics to measure the diversity of data in general: entropy and the Gini index. Since entropy increases with problem space, one has to use a normalized definition of entropy. So another approach is the Gini index, which has proven to scale well in both artificial and real event logs during our experiments. We also found out that both metrics can be used to detect noise (less frequent behavior). But since they turned out to be too coarse for indicating agenda-driven CM needs, we omitted these diversity metrics.

In the process mining manifesto [11], criteria with quality aspects are proposed using an event log maturity model. One of these is trustworthiness – that is, inhowfar one can rely on the validity of event log data. Unfortunately, this requirement cannot be measured by any metric. Secondly, most of the process mining techniques act on the assumption the event logs are complete at a certain level of granularity. Other quality requirements are that event logs provide semantics and that the extraction of event data conforms to law and organizational privacy and security guidelines. Of course, the latter can hardly be measured by event log metrics. But since current log file standards support semantical annotations, a metric could easily count the annotations within the event log and thus provide some kind of semantic coverage.

Finally, we think that it is possible to see the contribution of this paper as a fragment of the process mining algorithm evaluation framework proposed by Rozinat et al. [23]. To be exact, the metrics introduced here could come in useful for both their verification and modification tool elements.

VI. Conclusion

In this paper we defined characteristics of CM processes in order to develop metrics that can identify such processes in event logs. The metrics as a set can help to decide whether the event log quality is sufficient to continue with the discovery of control-flow perspectives. If it is not, we argue to apply CM-specific techniques at runtime such as the adaptive CM or agenda-driven CM [25] in order to be able to mine additional perspectives that are more suitable to CM processes.

The metrics have been designed to expose complexity using two different viewpoints: event and trace diversity. Starting with straightforward metrics like the average trace length, which can already be examined using tools like ProM, we refined and assembled them to more comprehensive metrics in order to identify different aspects of event log complexity.

Subsequently we developed four artificial processes for evaluation purpose, each designed to embody one single aspect of CM process complexity introduced beforehand. Starting with these processes, we generated event logs using the PLG framework.

The evaluation results from the artificial log experiments show that the metrics introduced in section III are valid. Each aspect of event and trace complexity could be discovered. But the experiments also show that some metrics like event density and the average trace length should always be considered mutually.

Of course, such artificial logs only represent unique perspectives that usually don´t occur solely in practice. So we also applied the metrics on three real-life event logs from the IEEE task force on process mining. As with the artificial logs, they cover different types of complexity, too: An ad-hoc hospital process similar to our definition of CM processes, a structured process extracted out of a workflow system and finally a simple, but ad-hoc structured process from incident management.

When applying the metrics to the real-life logs, it emerged that the metrics can indeed measure complexity in the field and thus can identify demand for alternatives to traditional process discovery (or at least preprocessing steps). On the contrary, an event log evaluated as not complex does not necessarily has to be a structured process. This accounts for both the definition of ad-hoc processes and the level of abstraction of the events. Future research could incorporate an event name complexity measure that weights every single event, e.g. based on frequency or path depth in ontologies like WordNet or domain ontologies like MeSH.

However, despite the formal definition of our metrics, we believe that it would not be helpful to aggregate them into one overall complexity value. As with other metrics, measurements have to be analyzed carefully within their context. Therefore, we want to conclude this paper with several guidelines we found valuable during the analysis of the metrics:

- Recurring behavior correlates with trace diversity. So, be sure to check the event density before continuing in the process mining workflow.
- Do not analyze the event diversity apart from trace diversity, since both can provide an illusion of complexity, e. g. due to scoping issues, noise or incompleteness.
- In case preprocessing, noise-filtering and realigning the scope do not help to improve the diversity metrics, or the logs seem incomplete no matter how many traces have been recorded, it is very likely that the analyzed process should not be supported by workflow systems, but rather by CM techniques such as agenda-driven CM [12, 23].

In the end, there are still some issues left that we will focus on in our future research. At first, since arbitrary sequences cannot be discovered by the metrics described in this paper, we plan to find a straightforward metric to measure concurrency in event logs. Second, we plan to gain more experience in how the metrics scale by conducting more comprehensive case studies. Additionally, we aim to develop and contribute a plug-in for the ProM framework that calculates and displays all metrics discussed in this paper from any given event log.

Acknowledgment

We want to thank all researches from Eindhoven and Ghent University involved in extracting and sharing real life event logs on http://data.3tu.nl/, especially B.F. van Dongen and Ward Steeman. We also want to thank all contributors to the process mining tool ProM, which enabled us to retrieve important information about the logs.

References

[1] van der Aalst, W., ed. (2011), Process Mining: Discovery, Conformance and Enhancement of Business Processes, Springer.

[2] van der Aalst, W. M. P.; Reijers, H. A.; Weijters, A. J. M. M.; van Dongen, B. F.; Alves de Medeiros, A. K.; Song, M. & Verbeek, H. M. W. (2007), 'Business process mining: An industrial application', *Inf. Syst.* **32**(5), 713-732.

[3] Rozinat, A. & van der Aalst, W. M. P. (2008), 'Conformance checking of processes based on monitoring real behavior', *Inf. Syst.* **33**(1), 64-95.

[4] Rozinat, A.; De Jong, I. S. M.; Günther, C. W. & van der Aalst, W. M. P. (2009), 'Process mining applied to the test process of wafer scanners in ASML', *Trans. Sys. Man Cyber Part C* **39**(4), 474-479.

[5] Mans, R.; Schonenberg, M.; Song, M.; van der Aalst, W. & Bakker, P. (2009), Application of Process Mining in Healthcare: A Case Study in a Dutch Hospital, *in* 'Biomedical Engineering Systems and Technologies', Springer, pp. 425-438.

[6] Medeiros, A. K.; Weijters, A. J. & van der Aalst, W. M. (2007), 'Genetic process mining: an experimental evaluation', *Data Min. Knowl. Discov.* **14**(2), 245-304.

[7] Günther, C. W. & van der Aalst, W. M. P. (2007), Fuzzy Mining - Adaptive Process Simplification Based on Multi-perspective Metrics, *in* 'BPM', pp. 328-343.

[8] Medeiros, A.; Guzzo, A.; Greco, G.; van der Aalst, W.; Weijters, A.; Dongen, B. & Sacca, D. (2008), Process Mining Based on Clustering: A Quest for Precision, *in* 'Business Process Management Workshops', Springer, pp. 17-29.

[9] Greco, G.; Guzzo, A.; Ponieri, L. & Sacca, D. (2006), 'Discovering expressive process models by clustering log traces', *Knowledge and Data Engineering, IEEE Transactions on* **18**(8), 1010-1027.

[10] van der Aalst, W.; Medeiros, A. & Weijters, A. (2006), Process Equivalence: Comparing Two Process Models Based on Observed Behavior, *in* Schahram Dustdar; José Luiz Fiadeiro & Amit P. Sheth, ed., 'Business Process Management', Springer, pp. 129-144.

[11] van der Aalst, W. et al. (2012), Process Mining Manifesto, *in* 'Business Process Management Workshops', Springer, pp. 169-194.

[12] Benner, M.; Book, M.; Brückmann, T.; Gruhn, V.; Richter, T. & Seyhan, S. (2012), Managing and tracing the traversal of process clouds with templates, agendas and artifacts, *in* 'Proceedings of the 10th international conference on Business Process Management', Springer, pp. 188-193.

[13] Yang, H.; van Dongen, B.; ter Hofstede, A.; Wynn, M. & Wang, J. (2012), 'Estimating Completeness of Event Logs', Technical report, BPM Center.

[14] Song, M.; Günther, C. W. & van der Aalst, W. M. P. (2008), Trace Clustering in Process Mining, *in* 'Business Process Management Workshops', pp. 109-120.

[15] Levenshtein, V. I. (1966): Binary Codes Capable of Correcting Deletions, Insertions and Reversals. In: *Soviet Physics Doklady* 10, S. 707

[16] Hamming, Richard W. (1950): Error detecting and error correcting codes. In: Bell Systems Tech Journal, Bd. 29, S. 147–160.

[17] Laue, R. & Gruhn, V. (2006), Complexity Metrics for Business Process Models, *in* 'BIS', pp. 1-12.

[18] Henry, S. M. & Kafura, D. G. (1981), 'Software Structure Metrics Based on Information Flow', *IEEE Trans. Software Eng.* **7**(5), 510-518.

[19] Burattin, A. & Sperduti, A. (2011), PLG: A Framework for the Generation of Business Process Models and Their Execution Logs, *in* Michael Muehlen & Jianwen Su, ed., 'Business Process Management Workshops', Springer, pp. 214-219.

[20] Ratzer, A.; Wells, L.; Lassen, H.; Laursen, M.; Qvortrup, J.; Stissing, M.; Westergaard, M.; Christensen, S. & Jensen (2003), CPN Tools for Editing, Simulating, and Analysing Coloured Petri Nets, *in* 'Applications and Theory of Petri Nets 2003', Springer, 450-462.

[21] Weijters, A.J.M.M. & Ribeiro, J.T.S. (2011). Flexible heuristics miner (FHM). Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2011, Paris, France, April 11-15, 2011). (pp. 310-317). Piscataway, USA: IEEE.

[22] Reijers, Hajo; Rigter, Jaap; Wil Van Der Aalst (2003): The Case Handling Case. In: *International Journal of Cooperative Information Systems* 12, S. 365–391.

[23] van der Aalst, W. M. P. (2009), Using Process Mining to Generate Accurate and Interactive Business Process Maps, *in* 'BIS (Workshops)', pp. 1-14.

[24] Rozinat, A.; Medeiros, A.; Günther, C.; Weijters, A. & van der Aalst, W. (2008), The Need for a Process Mining Evaluation Framework in Research and Practice, *in* 'Business Process Management Workshops', Springer, pp. 84-89.

[25] Benner-Wickner, Marian; Book, Matthias; Brückmann, Tobias; Gruhn, Volker (2014): Execution Support for Agenda-Driven Case Management. In: SAC ’14: Proceedings of the 29th Annual ACM Symposium on Applied Computing, forthcoming.