Estimating Completeness of Event Logs

Hedong Yang[1], B.F. van Dongen[2], Arthur HM ter Hofstede[2,3**], Moe T. Wynn[3], and Jianmin Wang[1]

[1] Tsinghua University, Beijing, China
[2] Queensland University of Technology, Brisbane, Australia
[3] Eindhoven University of Technology, Eindhoven, The Netherlands

Abstract. The field of process mining provides a collection of techniques and tools that aim to support the extraction of information out of event logs. This information may provide businesses insight into actual execution and performance of these business processes and may help identify ways of improving these processes. While the quality of the results of the application of mining algorithms depends on the degree of completeness of the event log (some even assuming that the logs are complete), it is necessary to evaluate completeness of an event log in an objective manner. Pragmatically speaking, the degree of completeness of an event log can be measured by estimating 1) the ratio of trace classes observed in the log versus the total number of trace classes, and 2) the coverage of behaviour observed (as a probability that captures how likely it is that a subsequent trace has already been observed). Based on some reasonable assumptions, we characterise the problem of estimating the degree of completeness of a log as a particular occurrence of the *species estimation problem*. Although this problem is still open in statistics, a number of methods for determining approximate solutions are available. Among these methods we provide a recommendation for those methods that are most appropriate in the context of process mining based on a number of experiments, both with real and with artificially generated event logs. In addition, a number of potential applications of (solutions to) the log completeness estimation problem are discussed.

1 Introduction

Business Process Management (BPM) is concerned, among others, with providing support for (re)design, deployment, and analysis of business processes. Process mining can be seen as a subfield of BPM providing a powerful collection of techniques for deriving information from event logs [31]. Two classical applications of process mining are *model discovery*, where the objective is to derive a process model from an event log ([16]), and *conformance checking*, where actual process behaviour, captured in an event log, is compared to expected process behaviour, captured in a process model. Over time the field of process mining has evolved, and broadly speaking, application of the techniques developed may provide valuable insight into operational performance of business processes and may

** Senior Visiting Scholar of Tsinghua University.

help identify opportunities for their improvement. Overviews of various mining algorithms can be found in some papers ([35,33,38]). An open source platform, ProM, exists which provides support for many of these algorithms and which has been used in industrial applications ([37]).

Although many problems in the field of process mining have been solved in a satisfactory manner, unsolved problems remain and these present impediments to advancement of the field ([35]). One of these problems, which has received very little attention in the literature, deals with completeness of the event logs.

Generally speaking, the quality of results mined from a given event log depends on the algorithm used and on the amount of information present in this log. Mining algorithms can be classified into several categories as argued in [38]. The first of these categories of consists of algorithms which assume the event log to be *globally complete*, i.e. all possible behaviour should have been recorded in the log. The second category contains those algorithms that make specific assumptions on completeness but do not require all possible behaviour to be captured, i.e. they require the log to be *locally complete*. The third category consists of algorithms where the quality of the result correlates with the degree of completeness of the event log, but no assumptions on completeness are made at all. Instead, for these algorithms, the more complete a log is, the better the result.

For the first category of algorithms it is necessary to be able to determine whether or not an event log indeed contains all possible behaviour. For the second category, it is important to say whether or not a log is complete with respect to the specific assumptions, while for the third category of algorithms, any information on completeness is relevant for predicting the applicability of that algorithm.

This then leads to the question: how can one determine global or local completeness of an event log *in an objective manner*?

Naturally the completeness of an event log depends on the type of information one is interested in. For example, one may wish to mine organizational structures or temporal information about task or case completions. In this paper the focus is on mining of control-flow dependencies, in other words, determining which tasks exist and what their relative execution order is. The objective of mining such dependencies is to obtain a process model of which the (repeated) execution could have led to the event log. Model discovery has been widely studied and was among the first applications of process mining.

Having established a focus on control-flow dependencies, the next issue to address is how to measure or estimate the degree of completeness of an event log with respect to control-flow information. To address this issue we need to agree on 1) the unit of information, and 2) the metrics that characterise the degree of completeness. For the unit of information a logical choice is the concept of a trace. While different definitions of this concept are possible, e.g. the various moments of choice could be preserved as part of a trace ([22]), here we simply see a trace as a sequence of actions which constitute a successful completion of a process instance. Taking the notion of trace as unit of information, the extent to which all possible trace occurrences of a workflow model (grouped in terms of trace classes) have been observed in an event log corresponds to the degree of *global completeness* of the log ([38]). Some process mining algorithms (e.g. [42,27,2,39]), require as input event logs that are globally complete. As regards metrics for characterising the degree of completeness of a log, the ratio of observed trace classes versus the total number of trace classes is an obvious candidate. This metric by itself may have a significant drawback however, as it does not take into account the occurrence probability of these classes. Clearly, we should take into account what the likelihood is that a newly observed trace corresponds to an already observed trace class when considering the degree of completeness of a log. This likelihood is referred to as the *coverage probability*, which constitutes the second metric that we will use when characterising the degree of completeness of an event log (in case the first metric's value is low).

As we do not have access to the process model that was used to generate the event log, we require a probabilistic approach to determining the degree of completeness of the log. In order to do this, we will articulate a number of underlying assumptions, which we believe to be reasonable. Based on these assumptions we will demonstrate that the problem of estimating the degree of completeness of an event log can be characterised as the problem of estimating the number of species on the basis of a finite number of observations. This *species estimation problem* has received a great deal of attention in the field of statistics ([6,19]).

The contribution of this paper is the precise characterisation of the event log completeness problem in terms of a particular occurrence of the species estimation problem and the subsequent empirical evaluation of a number of existing methods for approximating solutions to the species estimation problem in terms of their suitability for solving the event log completeness problem.

The remainder of this paper is organized as follows. Section 2 describes basic concepts needed to define the problem and to describe our approach, while Section 3 presents a number of assumptions for solving the problem. Section 4 presents the metrics for evaluating the completeness of event logs and formalizes the log completeness problem of event logs. Section 5 shows how the completeness estimation problem can be characterized as a particular occurrence of the species estimation problem, and lists some of the characteristics of the completeness estimation problem. Section 6 provides an overview to various approaches to solving this problem. In Section 7 an extensive empirical evaluation of these approaches is conducted, both using real and artificially created event logs. Section 9 discusses related work and section 10 concludes the paper and outlines potential future work.

2 Background

In this section a number of basic definitions are presented that are used to precisely characterise the problem of determining log completeness.

A *task* is an activity to be performed in the context of a business process. A *process model* provides an abstraction of a business process capturing its tasks and all possible execution orders of these tasks in a formal manner. A *process instance*, sometimes referred to as a *case* ([34]), represents the actual execution of a business process. A *trace* is the result of the successful completion of a process instance and consists of a sequence of events, where each *event* corresponds to the execution of a task ([34]).

The following definitions capture some of these concepts in a precise manner and they are almost the same as those presented in [35].

In this paper, our starting point is a set of tasks that are being executed in real life. We assume that we know all possible tasks involved in a process in advance and we assume the log ranges over these tasks. Furthermore, all events in the log should be ordered, typically using the time at which they were recorded.

Definition 1 (Log). *Let T be a finite set of tasks. We define $L = (E, C, \gamma, \tau, >)$ as an event log over T, where E is a set of events, C is a set of case identifiers, $\gamma : E \to C$ a surjective function relating events to cases, $\tau : E \to T$ a function relating events to tasks and $> \subseteq E \times E$ a total ordering on events.*

The execution of instances of various process models manifests itself through the corresponding traces whose events are recorded in process logs.

Definition 2 (Trace). *Let T be a finite set of tasks and $L = (E, C, \gamma, \tau, >)$ a log over T. For all cases $c \in C$, we define the trace $\sigma_c = \{e \in E \mid \gamma(e) = c\}$ as all events relating to case c. Since the events in E are totally ordered, the events in σ_c are also totally ordered.*

As stated before, the event log completeness problem requires the identification of the various trace classes in a log, so that traces that essentially represent the same behaviour can be grouped together.

Definition 3 (Trace equivalence). *Let T be a finite set of tasks and $L = (E, C, \gamma, \tau, >)$ a log over T. Furthermore, let $c_1, c_2 \in C$ be two case identifiers and $\sigma_{c_1}, \sigma_{c_2}$ the corresponding traces. We say that σ_{c_1} and σ_{c_2} are equivalent denoted by $\sigma_{c_1} \equiv \sigma_{c_2}$ if and only if for all $e_1 \in \sigma_{c_1}$ and $e_2 \in \sigma_{c_2}$ it holds that $(\#_{e \in \sigma_{c_1}} e < e_1 = \#_{e \in \sigma_{c_2}} e < e_2) \implies \tau(e_1) = \tau(e_2)$ and $|\sigma_{c_1}| = |\sigma_{c_2}|$.*

Definition 3 states that two traces σ_{c_1} and σ_{c_2} are equivalent if and only if their lengths are equal and every event of the first trace refers to the same task as the corresponding event of the second trace, i.e. the one having the same position if put in a sequence based on the ordering relation.

Clearly, the trace equivalence relation defined above forms an *equivalence relation* and we will refer to the corresponding equivalence classes as *trace classes*. Sometimes we will refer to two equivalent traces as being the same and we will refer to a trace σ occurring n times in an event log L if and only if the corresponding trace class has n elements.

Definition 4 (Global completeness of an event log). *Let T be a finite set of tasks, P a process model over T and L a log over T. We say that L is* globally complete *if and only if the number of trace classes in the log according to the* $\equiv$ *relation equals the number of all possible execution orders of* T *in* P.

It should be pointed out that a process model's behaviour cannot be fully characterised through its trace classes. This is due to the fact that the notion of trace used is limited and does not preserve moments of choice, but also because no currently existing process mining algorithm is capable of discovering all workflow control-flow patterns ([38]).

3 Assumptions

In this section the assumptions, based on which the log completeness problem can be precisely characterised, are made explicit. Each assumption is described in detail and it is argued that the assumption is reasonable, why it is needed, and what would go wrong if the assumption was not made.

Assumption 1 *Given a log, the total number of all possible trace classes is finite.*

On the one hand, if the number of possible trace classes is infinite, given an event log (which always has a finite size), the ratio of observed trace classes to all possible trace classes is always *zero*, which indicates that it is impossible to discover the workflow model through its event logs without additional hypotheses, e.g. the existence of loops. On the other hand, every man-made system will stop at some stage in the future, including running workflow management systems, hence always only a finite number of trace classes will be observed. It is therefore reasonable to assume that the number of possible trace classes is finite.

Assumption 2 *The event log is noise-free.*

Noise in an event log represents incorrect information, e.g. an event being missed or accidentally duplicated in the log, or the starting times of two events being mixed-up. Noise is often caused by hardware failures or by software bugs and the detection and remedy of these problems could lead one to also clean-up the event logs involved. The presence of noise in process logs still poses formidable problems for process mining. Some research in this area has been conducted and the common solution to dealing with noise is to use filtering techniques (see e.g. [29,43]). A trace or a task relationship is treated as noise if its occurrence frequency is lower than some specified threshold. The most challenging part of this solution is how to reach consensus on the right threshold.

It seems reasonable to assume that event logs are not polluted as long as there are no general and feasible ways of identifying noise in event logs, since the number of possible trace classes may be usually overestimated based on a noise-polluted log and thus to underestimate the actual degree of completeness of the log.

Assumption 3 *Traces of a process model appear randomly and independently.*

In a process-aware information system deployed in a large organisation, there are typically many instances of process models running concurrently. Instances of a process model may be initiated by a variety of stakeholders and may come about under a variety of circumstances. By observing the execution log, it cannot be determined with absolute certainty what the next trace will be, based on already observed traces. It seems reasonable to assume that traces are produced randomly and independently.

If the appearance of new traces depended on the already observed traces, we have to deal with correlated traces. Such traces should be treated as different appearances of the same trace as they can be (partially) derived from existing traces. If we cannot deduce how traces correlate with each other we may overestimate the number of existing trace classes and thus underestimate the degree of completeness of event logs.

Assumption 4 *Any trace of a process model appears with a constant but unknown probability. While constant for a particular trace class, this probability may vary across the various trace classes.*

A trace class represents a particular application scenario of a process model. When a process-aware information system has been running for years, the same scenario may appear periodically. As time goes by the *occurrence frequency* of the various traces becomes relatively stable and in the long run a trace may be perceived to appear with a constant probability, which we will refer to as its *occurrence probability.*

If this assumption is not satisfied, we cannot solve the problem of determining the degree of completeness of an event log in a mathematical way without further information about occurrence frequencies of traces.

4 Problem Definition

In this section, we first examine in more depth why global completeness is an important notion in process mining. Then, we consider again the issue of metrics that can be used for the log completeness problem, and finally, we characterise this problem precisely in terms of the species estimation problem using the assumptions articulated in the previous section.

4.1 Why global completeness?

While different control-flow mining algorithms may require different kinds of input information, their purpose is the same, to derive a process model capturing the control-flow dependencies that are reflected in event logs. The essence of control-flow information is captured by the tasks that occur in the various events in an event log and their execution order in the various process instances. A new execution order of tasks always appears in a new trace. Although a new trace may not contain any new control-flow dependencies between tasks, a complete set of traces always covers a complete set of dependency relationships. The more different traces the event log contains, the more one can derive potentially about the actual control-flow dependencies between these tasks.

In [38], an overview of mining algorithms is presented. For each algorithm, it is examined whether a log needs to be complete in order for the algorithm to correctly discover the underlying process model. Many of the powerful process mining algorithms examined in [38] require global completeness. All language-based region algorithms explicitly specify the requirement that all possible traces need to be present in an event log ([27,2,39]). The most powerful abstraction-based algorithm according to the analysis in [38] is the α^{++} algorithm ([42]). The analysis in [38] bears out that this algorithm can reconstruct all common control-flow constructs but not duplicate tasks, and also not invisible tasks (such tasks are used for routing purposes, for example in workflow models based on Petri nets). A successful outcome of the application of the α^{++} algorithm depends on whether all causal relationships between tasks, i.e. whether a certain task at runtime will ultimately always be followed by some other task, can be discovered. This implies that the event log has to be globally complete. State-based algorithms (e.g. [24]) require the construction of a state-transition diagram from an event log in order to produce an outcome. They therefore implicitly reconstruct a state space and for this state space to be accurate, the event log needs to be globally complete.

In summary, global completeness of event logs is a prerequisite for the application of a number of process mining algorithms to operate correctly. Additionally, global completeness of a log implies local completeness of the log. This means that being able to determine whether a log is globally complete may also be helpful for those algorithms that require local completeness, as it provides a sufficient though not necessary precondition.

4.2 Metrics of global completeness

When considering the question of the degree of completeness of an event log an obvious indicator is the number of trace classes observed versus the total number of trace classes. This ratio may thus be considered an intuitive metric. We will refer to this metric as the *observed trace classes* (OTC) ratio.

The OTC ratio when used by itself may have a significant drawback in case the number of trace classes has a very low occurrence probability. Consider for example the case, where three out of five possible trace classes have been observed in an event log. If the combined occurrence probability of the missing two trace classes is less than one percent, then clearly, the log can be considered quite complete in the sense that there is a high probability that the occurrence of a new trace will correspond to a trace class that has already been observed. The situation would be quite different if the two missing trace classes had a combined probability of 50 percent as in that case frequently occurring behavior has not been captured in the log and the log is thus less complete than the log in the previous scenario. This example makes it clear that using the OTC ratio by itself, in cases where the ratio is low, may not gave a sufficient enough insight into the degree of completeness of an event log.

Thus, we propose to complement the OTC metric with another metric that captures the degree to which behaviour, weighted in terms of its occurrency frequency, has already been observed. This metric can alternatively be characterised as the *coverage probability*, or CV probability for short, which is the likelihood that a new trace will already have been observed in the log. It should be pointed out that realistic workflow models usually incorporate ways of dealing with a variety of exceptions (some with a very low probability) and thus they may contain a high number of trace classes with a low occurrence probability. It is therefore expected that the OTC ratio would, as a rule, need to be complemented by the CV probability. J. Bunge et al. even believe that the CV probability is a better metric than the OTC ratio ([6] p. 370).

Naturally, the OTC ratio and the CV probability are not independent. When the OTC ratio increases, so does the CV probability. On the other hand, in the case of unbalanced logs, i.e. logs resulting from workflows with many trace classes with low occurrence probabilities, the OTC ratio may be low, but the CV probability will be high.

4.3 Problem Formulation

Having established two metrics in the previous section, we can now formally capture the log completeness estimation problem.

Problem 1 (Completeness problem). Given a set of tasks T, an event log L over T of an unknown workflow P (which was used to produce the traces recorded in L), 1) what is the ratio between the number of observed trace classes in L versus the total number of trace classes in P, and 2) what is the coverage probability, i.e. the probability that a new trace of the workflow has already been observed?

As we require the use of a process model, which is unknown, in the formulation of the problem, it is clear that we are dealing with an estimation problem.

5 Problem Characterisation

Based on the assumptions in Section 3, the problem of determining the degree of completeness of an event log can be recast as the problem of guessing for an urn with an unknown but finite number of marbles, how many different colours these marbles may have and how many marbles there are of each colour. The only input to the guessing problem is a number of selections of marbles where it was recorded what the colour of the marble was before it was put back in the urn ([26]). Here a marble in the urn is equivalent to a trace of a workflow model. Marbles with the same colour are equivalent to traces belonging to the same trace classes. Although the exact numbers of marbles in different colours are unknown, the numbers are static and so are their ratios. Similarly, although the occurrence probabilities of trace classes are unknown, the probabilities are constant according to Assumption 4. The most important part of the guessing problem is the sampling method, i.e. marbles are chosen randomly and independently and they are put back in the urn afterwards. This corresponds to the generation of traces of a workflow model according to Assumption 3. Summarising, the problem of guessing the numbers of marbles of different colours corresponds to being able to determine the degree of completeness of an event log.

This marble-guessing problem constitutes a particular occurrence of the so-called *species estimation problem*, wherein the number of species of a population, based on a finite sample, needs to be estimated. This is a well-known problem in the field of statistics ([6,19]). Hence, by characterising the log completeness problem as a particular manifestation of the species estimation problem, we may benefit from the vast body of knowledge, notably the multitude of approximation methods, that exists for solving this well-established statistical problem.

There are a number of different dimensions that relate to the species estimation problem. Specific choices for each of these dimensions create particular instances of the problem (these dimensions, articulated in the form of questions, are discussed in [6]). The dimensions in [6] are the size of the population (e.g. whether the population is finite or infinite), the sampling methods (e.g. Bernouilli, multinomial or Poisson), whether there is prior knowledge about the distribution of the population (e.g. whether it is known that all instances occur with an equal probability or whether their occurrences are governed by a particular curve), and the objective (e.g. estimating the total number of species or the probability that a new species is encountered in the next draw).

The log completeness estimation problem, as it is defined in section 4, can be seen as a particular manifestation of an instance of the species estimation problem with a finite population, Bernoulli sampling, and where estimating both the exact number of species and the probability of seeing a previously unobserved species are objectives.

Approaches to solving the species estimation problem can be divided into two categories ([6]). One category is based on data analysis, which covers extrapolation of curves (e.g. species accumulation curve) and fitting a truncated distribution to the observed species abundance (e.g. lognormal fit). These approaches are referred as non-sampling-based approaches in ([11]). Approaches using extrapolation based on previously observed samples suffer from the fact that there is no commonly agreed upon theoretical basis ([15] p. 107), while approaches using distribution fit assume that sample occurrences are governed by specific distributions ([6] p. 369, [15] p. 108). The other category is based on sampling theory and probabilistic theory. This category of approaches can be further classified into four groups according to [19] depending on the modelling method (the variables used and the way they are used in the estimation formula): 1) approaches which aim to provide a formal model capturing the current sample, 2) approaches which aim to characterise future drawings by (implicitly) assuming that these will be closely aligned with the sample, 3) approaches which model potential distributions of the occurrence probabilities of various species (this is the starting point of Bayesian methods), and 4) approaches that combine the latter two approaches.

However, it has been proven in [21] that no unbiased estimator exists if it is unknown whether the sample size is larger than the population size of the species with the highest occurrence. This implies that an approach may work well for a certain type of distribution but not for another type of distribution ([17] p. 198). According to [6] (p. 370), Good even believes it is usually impossible to estimate the total number of species. We therefore choose an experimental approach through the conduct of an in-depth comparison of a number of available estimators applied to a collection of log files, both real and artificially created ones.

The characteristics of the log completeness estimation problem exclude some approaches used for the species estimation problem as suitable candidates for solution methods:

– Since no a priori occurrence distribution of trace classes is assumed for the problem of event log completeness, approaches depending heavily on the presence of specific occurrence distributions are not considered to be suitable. For example, approaches requiring that occurrence distributions of trace classes are known (e.g. [23,18]), and approaches based on parametric models which assume that the occurrence distributions belong to a specific distribution family. Other examples are approaches proposed to estimating the vocabulary size of an author ([3]) because the frequency of words in literary works complies with Zipf's distribution ([1]), while the occurrence distribution of trace classes is unknown.
– Approaches assuming sampling without replacement are ignored. Sampling on a finite population without replacement means that a marble sampled later has a different occurrence probability from that of a marble of the same class previously sampled. Marbles are therefore not sampled in an independent manner which does not comply with Assumption 3.
– Approaches based on data analysis are ignored as these approaches are not compatible with the uncertain nature inherent to estimation problems. As argued before, the problem of determining the degree of log completeness is an estimation problem and the result is correct with a certain probability.

6 Approaches

Based on the analysis of characteristics of the completeness problem, we found a number of estimators as candidate approaches for solving the problem. Our experiments will be carried out using these approaches.

6.1 Conventions

We assume an event log with n traces and o trace classes. The number of traces classes which occur i times in the log is given by n_i (where $0 \leq i \leq n$). The estimations of the number of all possible trace classes, the occurrence probability of observed trace classes, and the probability of a new trace class appearing are denoted by $\hat{N}, \hat{C}$, and $\hat{U}$ respectively. Obviously $\hat{C} + \hat{U} = 1$ and thus if $\hat{C}$ is known then $\hat{U}$ is known and vice versa. The OTC ratio, which we denote as r, is given by $r = o/\hat{N}$. As $\hat{N}$ is the key factor in calculating this ratio, we focus on $\hat{N}$ rather than r in this section.

6.2 Boender and Rinnooy Kan's Estimator

Boender and Rinnooy Kan [5] proposed a Bayesian approach, in which the number of species and their occurrence probabilities are assumed to be random variables. The occurrence distribution is initially set to the uniform distribution. The a posteriori distribution is then calculated using Bayes' Theorem given a sample. Based on the a posteriori distribution, the estimators for the total number of classes and the probability of the next sample being a new species are computed as follows:

$$\hat{U}_{BRK} = \frac{o(o+1)}{n(n-1)} \qquad (n \geq o+2) \tag{1}$$

$$\hat{N}_{BRK} = o\frac{n-1}{n-o-2} \qquad (n \geq o+3) \tag{2}$$

Although Bayes' Theorem guarantees that the initial distribution, however it is chosen, will converge to the correct distribution when the sample is large enough, the accuracy of these estimators may still be questionable as we do not know whether any given sample is large enough or not.

6.3 Turing and Good's Estimator

On assumption H stating that there is a occurrence distribution which governs the sampling, Good [20] proved that

$$E(q_r|H) = \frac{r+1}{n+1}\frac{E_{n+1}(n_{r+1}|H)}{E_n(n_r|H)}$$

where $E(q_r|H)$ is the expectation of the occurrence probability of a species occurring r times in the sample, and $E_n(n_r|H)$ is the expectation of the number of species occurring r times in the log with size n. By approximating $E_{N+1}(n_{r+1}|H)$ and $E_N(n_r|H)$ with the values n_{r+1} and n_r respectively or with these values smoothed (in case of division by zero), Good proposed an estimator $\hat{U}_{TG}$ with smoothing, inspired by A.M. Turing, hence the name Turing and Good's Estimator.

$$\hat{U}_{TG} = \frac{n_1}{n+1} \tag{3}$$

On the assumption that all trace classes are equally likely, an estimator for $\hat{N}$ can be produced.

$$\hat{N}_{TG} = \frac{o}{\hat{C}_{TG}} = \frac{o}{1 - \frac{n_1}{n+1}} \tag{4}$$

From $\hat{U}_{TG}$, it is clear that the number of estimated unobserved species approximates the number of species that occur once only. It can be observed that this estimator is a bit coarse as only n_1 is used.

6.4 Maximum Likelihood Estimation(MLE)

When all trace classes occur with equal probability, Lewontin and Prout [25] proposed a Maximum likelihood estimator $\hat{N}_{MLE}$, which can be approximated by solving equation (5).

$$o = \hat{N}_{MLE} \left(1 - e^{-n/\hat{N}_{MLE}}\right) \tag{5}$$

Once $\hat{N}_{MLE}$ has been estimated, it is straightforward to estimate the coverage probability, $\hat{C}_{MLE}$, since all trace classes are governed by a uniform distribution.

$$\hat{C}_{MLE} = \frac{o}{\hat{N}_{MLE}} \tag{6}$$

The estimator $\hat{N}_{MLE}$ will underestimate the total number of all possible trace classes when the occurrence of trace classes are not uniformly distributed. When the size of the log is much smaller than the number of all possible trace classes, it is highly possible that each observed trace class occurs only once in the log. This estimator can work for such kinds of logs and this is the reason we have chosen this estimator as a candidate.

6.5 Jackknife Estimation

By extending the trivial estimator of $\hat{N}_{trivial} = o$ and on the assumption that $\hat{o} - N = \sum_{i=1}^{\infty} \frac{c_i}{n}$ where c_i is a(n) (unknown) constant, Burnham et al. [7] proposed an estimator $\hat{N}_{Jk} = \sum_{i=1}^{o} a_{ik} n_i$ by means of the jackknife approach, which is a linear combination of the n_i. Given different values for the order k, we can get a series of estimators.

$$\hat{N}_{Jk} = \frac{1}{k!} \sum_{i=0}^{k} (-1)^i \binom{k}{i} (n-i)^k \hat{N}_{n-i} \tag{7}$$

$$\hat{N}_{n-i} = o - \binom{n}{i} \sum_{r=1}^{i} \binom{n-r}{i-r} n_r \tag{8}$$

The larger k is, the smaller the estimation bias and the bigger the variance. Burnham et al. in [8] proposed a method to select an appropriate k for a given sample. The weakness of these estimators is that the assumption is very rigid for some distributions ([6] p. 369).

6.6 Gandolfi and Sastri's Estimator

Gandolfi and Sastri [19] reviewed a number of available estimators and proposed a Bayesian-based estimator $\hat{N}_{GS}$.

$$\hat{N}_{GS} = \frac{o * n}{o - n_1} + \frac{o * n_1}{o - n_1}\gamma^2, \tag{9}$$

where $0 \leq \gamma^2 = \frac{-o-n+n_1+\sqrt{5o^2+2o(n-3n_1)+(n-n_1)^2}}{2n} \leq 1$. Four types of estimators were listed in Section 5 and this estimator is of the fourth kind. Experiments on some real data sets with nonuniform populations show that the estimator works best compared to the other estimators examined, since it benefits from both Bayes' theorem and sampling theory [19].

6.7 First Estimator of Chao

Chao [9] proposed an estimator $\hat{N}_{Chao1}$ for the case where (o, n_1, n_2) carries most information of the sample, i.e. the total number of species which occur more than 2 times in the given log is much smaller than o.

$$\hat{N}_{Chao1} = o + \frac{n_1^2}{2n_2}. \tag{10}$$

The above estimator would not work when $n_2 = 0$. Thus Chao proposed a bias-corrected version as follows.

$$\hat{N}_{Chao1} = o + \frac{n_1(n_1 - 1)}{2(n_2 + 1)}. \tag{11}$$

The approach demonstrated encouraging results for real data [9]. Colwell et al. [15] believed that the estimator deserves consideration since it is usually the case that the low frequency species, i.e. those occurring 1 or 2 times, provide most of the information in a sample.

6.8 Second Estimator of Chao

Chao [10] proposed a variation of the $\hat{N}_{Chao1}$ estimator, which yet requires o to be large and p_i (the probabilities for the various species occurring) to be small. Colwell [14] proposed another bias-corrected estimator, $\hat{N}_{Chao2}$, for the $\hat{N}_{Chao1}$ estimator as follows.

$$\hat{N}_{Chao2} = o + \frac{n_1^2}{2(n_2 + 1)} - \frac{n_1 n_2}{2(n_2 + 1)^2}. \tag{12}$$

6.9 Chao and Lee's Estimators

Chao and Lee [12] proved that $N \approx \frac{E(o)}{E(C)} + \frac{E(n_1)}{E(C)}\gamma^2$, where $E(.)$ is the expectation of a random variable and γ is the coefficient of the variance of the potential occurrence distribution of trace classes [12].

$E(o), E(C)$, and $E(n_1)$ in the approximation formula, depend on the unknown occurrence distribution of trace classes and are approximated by $o, 1 - n_1/o$, and n_1 respectively. There are two approaches in [12] to estimating γ, each leading to a different estimator, $\hat{N}_{CL1}$ and $\hat{N}_{CL2}$.

$$\hat{N}_{CL1} = \frac{o * n}{o - n_1} - \frac{o * n_1}{o - n_1}\hat{\gamma}_1^2, \tag{13}$$

$$\hat{N}_{CL2} = \frac{o * n}{o - n_1} - \frac{o * n_1}{o - n_1}\hat{\gamma}_2^2, \tag{14}$$

where

$$\hat{\gamma}_1^2 = max\left(\frac{o * n}{o - n_1}\sum \frac{j(j-1)n_j}{o(o-1)} - 1, 0\right), and \tag{15}$$

$$\hat{\gamma}_2^2 = max\left(\hat{\gamma}_1^2(1 + n_1 \sum \frac{j(j-1)n_j}{(o-n_1)(o-1)}), 0\right). \tag{16}$$

These two estimators are considered to be the preferred choice in the absence of information about the sampling method and the population structure [6].

6.10 The Abundance-based Coverage Estimator (ACE)

In [13], Chao extended the work of [12], and proposed an Abundance-based Coverage Estimator (ACE). The most interesting part of the estimator is that it seperates the species into two groups according to the given log: rare species whose occurrence frequencies are less than 11 and abundant species whose occurrence frequencies are at least 11.

$$\hat{C}_{ACE} = 1 - \frac{n_1}{n_{rare}} = 1 - \frac{n_1}{\sum_{j=1}^{10} j * n_j}, \tag{17}$$

$$\hat{N}_{ACE} = \sum_{j>10} n_j + \frac{n_{rare}N_{rare}}{n_{rare} - n_1} + \frac{n_{rare}n_1}{n_{rare} - n_1}\hat{\gamma}_{rare}^2, \tag{18}$$

where

$$\hat{\gamma}_{rare}^2 = max\left(\frac{n_{rare}N_{rare}}{n_{rare} - n_1}\sum_{j=1}^{10}\frac{j(j-1)n_j}{n_{rare}^2} - 1, 0\right).$$

This estimator emphasises low frequency species which are believed to be important for estimating the number of unobserved species. Not separating the long-tailed data (i.e. with a large number of rare species) will cause positively biased estimates ([11] p. 8).

6.11 Tools

There are a number of tools available for the species estimation problem, such as Estimats[4], SPADE[5] and ws2m[6]. The former two tools are still under active development and cover most estimators available, while ws2m has not been updated since 2004. We implemented the selected estimators mentioned in this section ourselves since neither a data format for event logs nor the OTC ratio is directly supported by these tools. In addition, some Bayesian estimators are not supported by these tools either.

There is a package, SPECIES (Species Richness Estimation), for the open source statistical software program R[7], which needs some programming work to use the estimators supported. Hence, we also chose not to use this package.

7 Experiments

In this section, we compare the estimators discussed in Section 6 in a qualitative way. We apply all estimators on a number of event logs, both artificial and real-life, to get insights into their performance, both in situations where traces occurrences are uniformly distributed as well as when they are not. Our work focuses on the accuracy rather than the precision, i.e. we present the mean square errors of comparing the estimated values to the true values for the simulated data.

7.1 Estimating the Event Classes on Artificial Logs

For our experiments, we use an example model describing the process within a travel agency. In Figure **??**, the model is depicted. When a customer books a trip, he first gets registered in the system. Then, one, two or three hotels are booked and in parallel either a plane or a bus ticket is reserved. Afterwards, the trip costs are calculated and then two types of insurances may be purchased, namely trip insurance and cancellation insurance. Finally, the insurance costs are added to the total sum and the payment is made by the customer. This leads to 4 decision points in the process, i.e. the choice between a bus and a plane, the choice between one, two or three hotel bookings, the option to take travel insurance and the option to take cancellation insurance.

In order to generate log files, we simulated three variants of this model and for each variant, we produced 50 log files containing 1000 cases (i.e. in total we produced 50,000 cases per variant). The three variants differed in the probabilities used to make the choices. In Table 1, we show the probabilities used in the three variants. Theoretically, ignoring the chosen time distributions, this model allows for 72 possible traces. However, timing constraints are such that the last

[4] `http://viceroy.eeb.uconn.edu/estimates`
[5] `http://chao.stat.nthu.edu.tw`
[6] `http://eebweb.arizona.edu/diversity/`
[7] `www.r-project.org`

of three hotel bookings will only be completed after a bus or a plane ticket have been booked and hence only 64 trace classes can be seen in practice.

Table 1. The parameters for three variants of our model

Variant	Bus/Plane	1/2/3 hotels	yes/no travel insurance	yes/no cancellation insurance
Balanced	.5 / .5	.5 / .25 / .25	.5 / .5	.5 / .5
Unbalanced	.6 / .4	.3 / .21 / .49	.7 / .3	.4 / .6
Extremely unbalanced	.9 / .1	.05 / .0475 / .9025	.9 / .1	.05 / .95

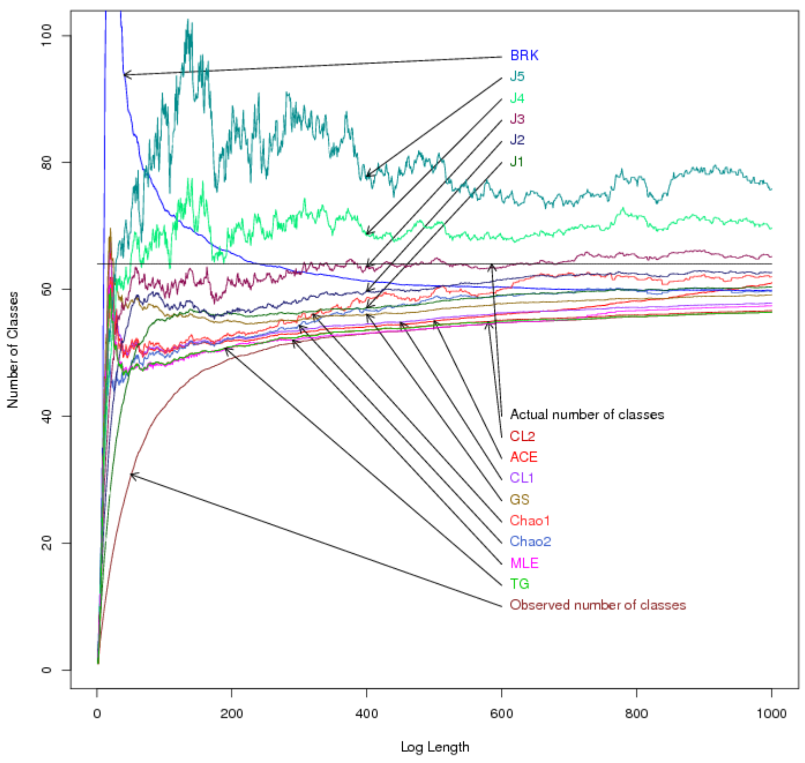

Fig. 1. Estimators on variant with balanced choices.

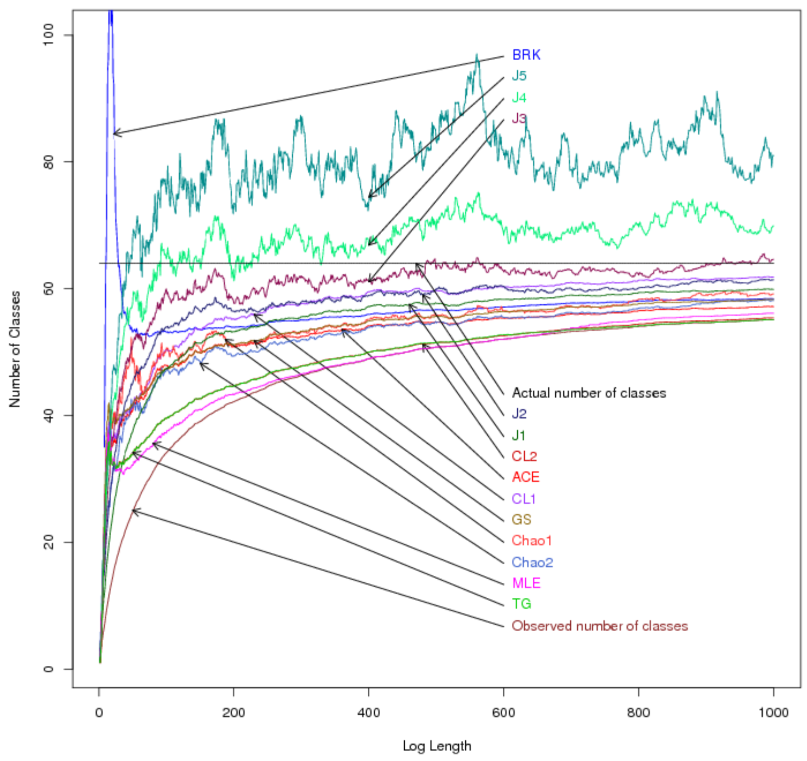

Fig. 2. Estimators on variant with unbalanced choices.

We conducted several experiments in the following way. For each log, we considered the first n traces (with n ranging from 1 to 1000). Then, we estimated the total number of trace classes and the coverage. We performed these experiments for all of the 50 logs per variant and we obtained both the average of the estimated values, as well as the mean square error. The full results are presented in Figure 1 for the balanced model, Figure 2 for the unbalanced model and Figure 3 for the extremely unbalanced model. In each of the figures 1 to 3, we depicted the actual number of observed classes, the number of possible classes and the two parametric estimators TG and MLE as dotted lines.

In figure 1, no estimator gives a significantly better result than any other, i.e. all estimators except for J4 and J5 eventually come close to estimating the actual value of 64 trace classes. However, the Chao1 estimator (Section 6.7) and the J2 and J3 estimators (Section 6.5) perform the best.

In Figure 2, J2 and J3 again perform very well. However, now the CL1 estimator discussed in Section 6.9 outperforms the Chao1 estimator. This is

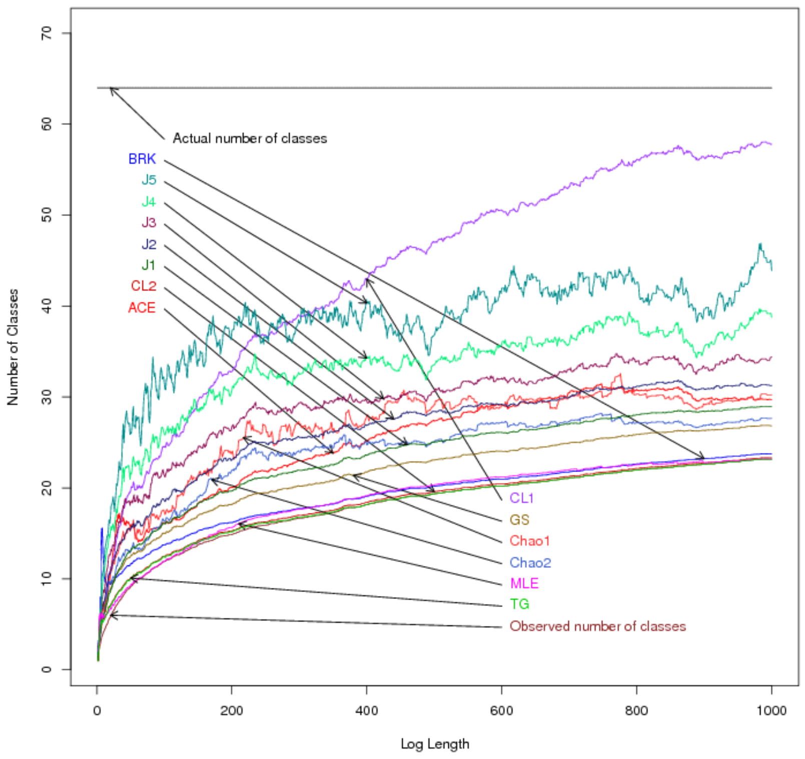

Fig. 3. Estimators on variant with extremely unbalances choices.

not surprising, as the Chao1 estimator gives an estimated lower bound for the number of classes, while the CL1 estimator improves on that by correcting for the downward bias introduced by other estimators in the case that classes appear with non-equal probabilities [6].

In Figure 3, it is clear that the CL1 estimator outperforms all others. This validates the claim in [12] that: "the proposed estimators for non-equiprobable cases are generally biased downward due to the underestimation of [γ]".

Overall, our experiments on artificial logs suggest that both the Jackknife estimators J2 and J3 are generally good estimators for logs with balanced behavior, but once behavior becomes more unbalanced, the CL1 estimator becomes better. However, it should be noted that determining the parameter k for the Jackknife estimation is not trivial and all Jk estimators behave rather erradically, i.e. adding a single trace to the log can yield very different estimations. Therefore, we favor the CL1 estimator.

7.2 Estimating the Coverage on Artificial Logs

Besides estimating the total number of trace classes, some estimators also provide estimates for the coverage, i.e. the fraction of classes observed. For the parametric estimators TG and MLE, this number is derived directly from the estimation of the number of classes, or vice versa. In figures 4 to 6, we depicted the four coverage estimators we used on our artificial logs. In each of the figures, we also indicated the actual coverage.

From figures 4 to 6 it is clear that none of the estimators actually give a good estimate of the coverage. Most estimators overestimate the coverage. Only the ACE estimator provides a better estimate, mainly because this estimator takes into account the sample variance of the number of observations in each class if there are less than 10 observations in that class.

It is important to realize however that if the log size grows to the situation where all observed trace classes appear more than 10 times, the addition of a single new class will change the ACE estimate from 100% to 0%. Therefore, some care should be taken when using the ACE estimator for coverage and in Section 7.3, we use both the ACE and BRK for coverage, as these are the only two non-parametric estimators.

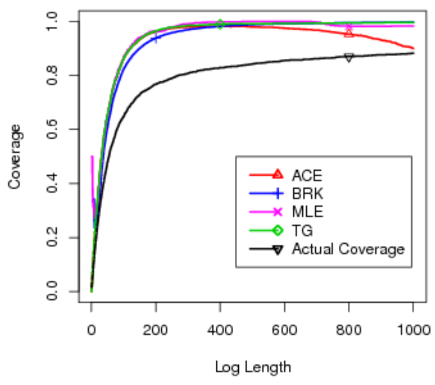

Fig. 4. Estimators on variant with balanced choices.

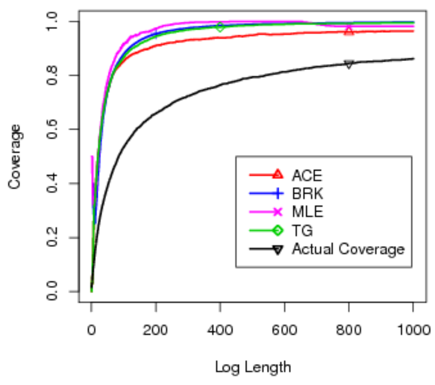

Fig. 5. Estimators on variant with unbalanced choices.

7.3 Application to Real life Logs

We applied the best estimators to a collection of real-life logs with different degrees of variability. Some details of the logs are given in Table 2. These logs are taken from a Dutch municipality and they contain events from four different

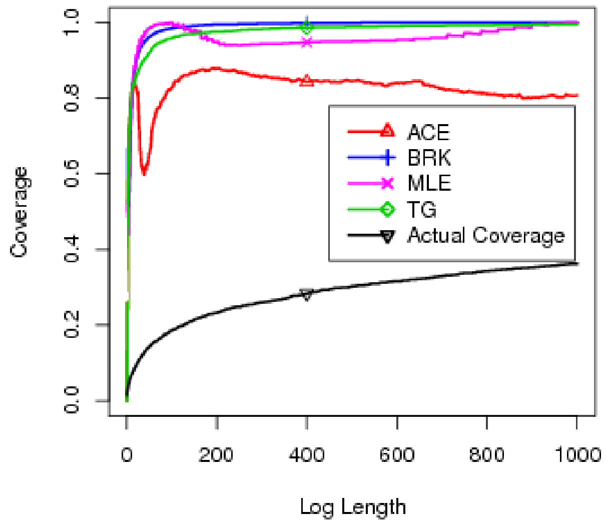

Fig. 6. Estimators on variant with extremely unbalanced choices.

administrative processes. Table 2 also shows the estimated number of classes by the CL1 estimator as well as the estimated coverage by the ACE estimator and the BRK estimator.

Table 2. Four real-life logs

Log	Number of traces	Number of events	Distinct activities	Observed trace classes	Estimated classes (CL1)	Estimated coverage (ACE)	Estimated coverage (BRK)
Statements	374	1,116	7	8	18	88.24 %	99.95 %
Objections	127	863	15	65	642	46.39 %	73.19 %
Housing Tax	1,982	12,726	17	201	988	73.46 %	98.97 %
Building per.	2,058	11,105	31	424	85,959	44.96 %	95.74 %

The first log contains events relating to the municipality giving out statements and this is a simple administrative process. The CL1 estimater estimates the total number of trace classes to be 18 (the result is rounded to the nearest integer), while the ACE estimates the coverage to be 88.24%. In other words, these estimates indicate that there are still 11 unobserved trace classes, but that these only make up 11.76% of the total behavior, while the BRK estimator even estimates them to only make up 0.05% of the total behavior.

The second log contains events about a general objection handling process within the municipality, with the exception of housing tax objections. This process is more involved than the statement process, i.e. there are more activities, but it is about 3 times less frequent. The estimated number of trace classes by the CL1 estimator is 642, while the coverage is estimated to be 46.39%, hence we have observed less than half of the possible behavior.

The third log deals with objections to housing tax. This process is similar in nature to the other objection process, but much more frequent. The CL1 estimates that there are 988 trace classes, of which 201 have been observed so far. The coverage of these trace classes is estimated to be 73.46%.

Finally, the building permit log is a very complex process, with 424 trace classes observed in only 2058 cases. The CL1 estimator therefore estimates that there is a total of $85,959$ trace classes, but the observed classes are estimated to cover 44.09% of the behavior.

The results of our estimators on the real life logs suggest that the log on statements and housing tax provide sufficient information to start a process mining experiment on, i.e. the log covers more than 70% of the possible behavior, even though there are still many classes of traces that were not observed. Therefore, one should be careful using process mining techniques that assume the event log to be complete. The other two logs (objections and building permits) seem to contain too little information to do process mining, or at least, reservations should be made regarding the fact that the logged behavior is estimated to represent less than half of the possible behavior, and that the total number of trace classes is at least 10 times more than the observed number.

8 Implementation

The work presented in this paper is implemented as a plug-in of ProM, a generic open-source framework for implementing process mining tools in a standard environment [37,40]. Figure 7 shows a screenshot of the implementation, showing the coverage and trace classes estimates for the housing tax objection process of Table 2.

In ProM, all estimators mentions in this paper have been implemented in the "LogMetrics" package. After opening logs, the estimates can be made by invoking the "estimate completeness" task on a number of logs. The result is an object which can be visualized as a chart as shown in Figure 7, but the estimates can also be exported to the csv format for further processing.

9 Related Work

In the area of process mining, the notions of event, event log, and trace have been well defined ([36]). A typical definition of an event log covers the name of a

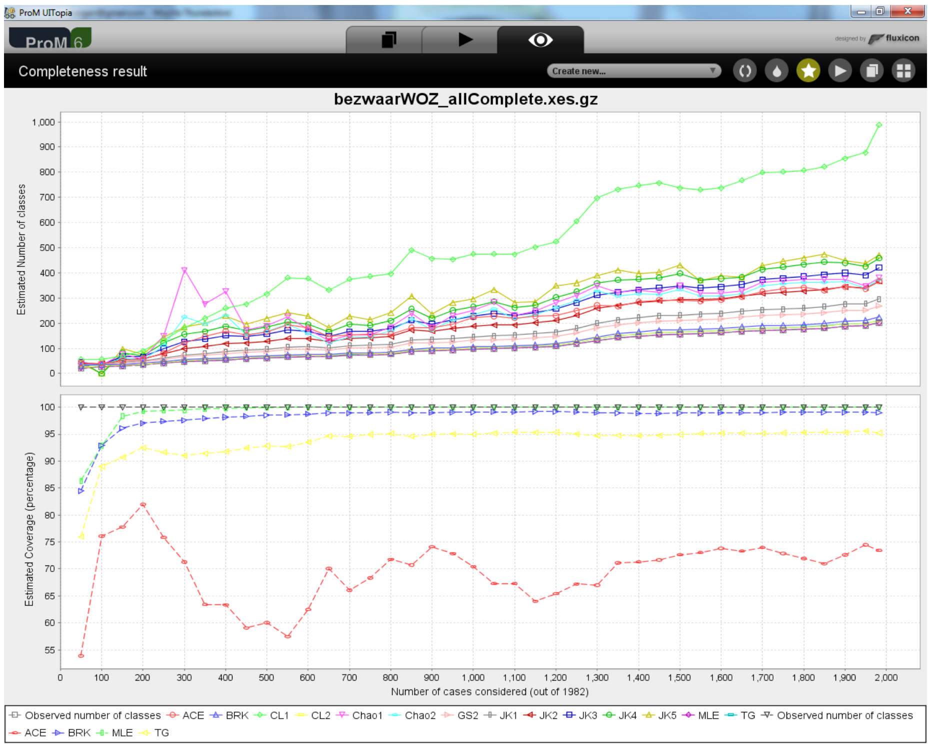

Fig. 7. Screenshot of ProM, showing the housing tax objection estimates.

task, a process instance and the execution order between tasks (e.g. [41]) because these are the most important aspects for control-flow mining. This is referred to as the required *minimal information* in [33]. While the completeness problem has been mentioned in several papers (e.g. [36,28,32]), it has not been discussed in detail. The issue of log completeness was first raised in [35]. The incompleteness of logs is often seen as a kind of noise ([33]). Although many process mining algorithms require the log to be complete, there are other algorithms such as the genetic mining algorithm ([32]) that can work with incomplete logs. In [38], completeness is one of the metrics used to evaluate process mining algorithms. As the quality of mining results relies heavily on the degree of completeness of event logs, an in-depth exploration of this notion, the topic of this paper, is worthwhile.

The definition of completeness given in the literature varies depending on the objective of a process mining algorithm ([38]). If an algorithm requires information about relationships between tasks, the log is considered to be incomplete if not all such relationships are present in the log. A typical definition is given in [36] where the α-algorithm uses the $>_W$ relationship defined between tasks and requires all possible pairs to appear in the log at least once. In this paper a log is considered to be complete if at least one trace of all possible trace classes is present in the log. The advantage of the definition is that it is general with respect to the notion of global completeness and is therefore not tailored to any specific algorithm that requires such completeness. The disadvantage of this definition is that process models with loops have an infinite number of traces theoretically. Fortunately in practice an algorithm that can recognise the presencee of loops would reduce the number of trace classes to a reasonable level during preprocessing stage of process mining.

The problem of completeness is similar to, but not the same as, the Coupon Collecting Problem (refer to page 322 of [30])[8] in the field of probability theory. Both problems require one to determine the number of different elements in an unknown collection through sampling with replacement. The difference between these problems is that both the total number of coupon types and the appearance probability of each coupon type are known in the Coupon Collecting Problem, while neither the number of trace classes nor the occurrence probability of each class is known in the completeness problem. Thus none of the solutions (e.g. [18]) to the Coupon Collecting Problem can be applied to directly solve the completeness problem.

10 Conclusions and Future Work

In this paper, we compared several estimators for estimating how much of the potential behavior of an information system is captured by an event log. We compared several estimators for the number of trace classes (i.e. the possible number of different execution sequences) as well as several estimators for the coverage (i.e. the fraction of the behavior that is covered by the log). In both cases, we looked at parametric and non-parametric estimators.

To evaluate the estimators, we benchmarked them on three sets of simulated logs, each with different characteristics, for which the actual coverage and the actual number of trace classes is known. Then, we applied the best estimators to some real-life datasets to asses the completeness thereof.

Our experiments show that even the best estimators generally over-estimate the coverage while at the same time, on real-life datasets, the coverage is estimated to be less than 50 %. This leads us to believe that it is generally a bad decision to assume event logs to be even near complete.

The work presented in this paper can be extended in several directions. For logs that result from the execution of models with loops the global completeness estimation may not be very accurate as the estimators tend to underestimate the actual degree of completeness. This is due to the fact that different iterations of the body of a loop will lead to different trace classes (while such repeated executions add less and less information). In order to overcome this weakness, one could consider alternative definitions of trace classes. For example, one could define trace classes based on task precedence relations as this would always yield

[8] This problem is also referred to as Coupon Collector Problem in [23,18] or as Coupon Collection Problem in [4].

a finite number of trace classes (given that the number of tasks is assumed to be finite). Such a definition is particularly meaningful in case of the well-known α-algorithm (as introduced in [36]). In fact, one could examine various definitions of trace classes in relation to the log length and information completeness problems, and link these definitions to process mining algorithms for which they are particularly suited. In that case the focus can be increased to include local completeness. Another extension to this work could assume the existence of noise in logs.

Acknowledgements

The work reported in this paper is supported by the National Basic Research Program (973 Plan) of China (No. 2009CB320700), the National High-Tech Development Program (863 Plan) of China (No. 2008AA042301) and an NSF Project of China (No. 61003099). This work is also supported under Australian Research Council's Discovery Projects funding scheme (project number DP110100091).

References

1. R. Harald Baayen. *Word Frequency Distributions.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.
2. Robin Bergenthum, Jörg Desel, Robert Lorenz, and Sebastian Mauser. Process Mining Based on Regions of Languages. In Gustavo Alonso, Peter Dadam, and Michael Rosemann, editors, *BPM'07: Proceedings of the 5th International Conference on Business Process Management*, volume 4714 of *Lecture Notes in Computer Science*, pages 375–383, Berlin, Heidelberg, 2007. Springer-Verlag.
3. Suma Bhat and Richard Sproat. Knowing the Unseen: Estimating Vocabulary Size over Unseen Samples. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1*, ACL '09, pages 109–117, Morristown, NJ, USA, 2009. Association for Computational Linguistics.
4. Rabi N. Bhattacharya and Edward C. Waymire. *Stochastic Processes with Applications.* John Wiley & Sons, Inc., New York, NY, USA, 1990.
5. C. G. E. Boender and A. H. G. Rinnooy Kan. A Bayesian Analysis of the Number of Cells of a Multinomial Distribution. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 32(1/2):240–248, 1983.
6. John A. Bunge and M. Fitzpatrick. Estimating the Number of Species: A Review. *Journal of the American Statistical Association*, 88(421):pp. 364–373, 1993.
7. Kenneth P. Burnham and Walter Scott Overton. Estimation of the Size of a Closed Population when Capture Probabilities vary Among Animals. *Biometrika*, 65(3):pp. 625–633, 1978.
8. Kenneth P. Burnham and Walter Scott Overton. Robust Estimation of Population Size When Capture Probabilities Vary Among Animals. *Ecology*, 60(5):pp. 927–936, 1979.
9. Anne Chao. Nonparametric Estimation of the Number of Classes in a Population. *Scandinavian Journal of Statistics*, 11(4):pp. 265–270, 1984.

10. Anne Chao. Estimating the Population Size for Capture-Recapture Data with Unequal Catchability. *Biometrics*, 43(4):pp. 783–791, 1987.
11. Anne Chao. Species Richness Estimation. New York, 2004. http://chao.stat.nthu.edu.tw/paper/SpeciesRichnessEstimation.pdf.
12. Anne Chao and Shen-Ming Lee. Estimating the Number of Classes via Sample Coverage. *Journal of the American Statistical Association*, 87(417):pp. 210–217, 1992.
13. Anne Chao and Mark C. K. Yang. Stopping Rules and Estimation for Recapture Debugging with Unequal Failure Rates. *Biometrika*, 80(1):pp. 193–201, 1993.
14. Robert K. Colwell. EstimateS: Statistical estimation of species richness and shared species from samples. Version 8.2. User's Guide and application published at: http://purl.oclc.org/estimates, 2009.
15. Robert K. Colwell and Jonathan A. Coddington. Estimating Terrestrial Biodiversity through Extrapolation. *Philosophical Transactions: Biological Sciences*, 345(1311):pp. 101–118, 1994.
16. Marlon Dumas, Wil M. P. van der Aalst, and Arthur H. M. ter Hofstede, editors. *Process-Aware Information Systems : Bridging People and Software through Process Technology*. Wiley-Interscience, Hoboken, NJ, USA, 2005.
17. Warren W. Esty. Estimation of the Size of a Coinage: A Survey and Comparison of Methods. *Numismatic Chronicle*, 146:185–215, 1986.
18. Philippe Flajolet, Danièle Gardy, and Loÿs Thimonier. Birthday Paradox, Coupon Collectors, Caching Algorithms and Self-Organizing Search. *Discrete Appl. Math.*, 39(3):207–229, 1992.
19. Alberto Gandolfi and Chelluri C. A. Sastri. Nonparametric Estimations about Species Not Observed in a Random Sample. *Milan Journal of Mathematics*, 72:81–105, 2004. 10.1007/s00032-004-0031-8.
20. Irving John Good. The Population Frequencies of Species and the Estimation of Population Parameters. *Biometrika*, 40(3-4):237–264, 1953.
21. Leo A. Goodman. On the Estimation of the Number of Classes in a Population. *The Annals of Mathematical Statistics*, 20(4):pp. 572–579, 1949.
22. Jan Hidders, Marlon Dumas, Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, and Jan Verelst. When are Two Workflows the Same? In M. Atkinson and F. Dehne, editors, *Proceedings of Computing: The 11th Australasian Theory Symposium (CATS'2005)*, volume 41 of *Conferences in Research and Practice in Information Technology*, pages 3–11, Newcastle, Australia, February 2005. Australian Computer Society.
23. Edward P. C. Kao. *An Introduction to Stochastic Processes*. Duxbury Press, North Scituate, MA, USA, 1997.
24. Ekkart Kindler, Vladimir Rubin, and Wilhelm Schäfer. Process Mining and Petri Net Synthesis. In Johann Eder and Schahram Dustdar, editors, *Business Process Management Workshops, BPM 2006 International Workshops, BPD, BPI, ENEI, GPWW, DPM, semantics4ws, Vienna, Austria, September 4-7, 2006, Proceedings*, volume 4103 of *Lecture Notes in Computer Science*, pages 105–116. Springer, 2006.
25. Richard C. Lewontin and Timothy Prout. Estimation of the Number of Different Classes in a Population. *Biometrics*, 12(2):pp. 211–223, 1956.
26. Manuel E. Lladser. Prediction of Unseen Proportions in Urn Models with Restricted Sampling. In *Proceedings of the Sixth Workshop on Analytic Algorithmics and Combinatorics (ANALCO'09)*, pages 85–91, New York, NY, USA, January 2009. SIAM.

27. Robert Lorenz, Sebastian Mauser, and Gabriel Juhás. How to Synthesize Nets from Languages - A Survey. In *WSC '07: Proceedings of the 39th Conference on Winter Simulation*, pages 637–647, Piscataway, NJ, USA, 2007. IEEE Press.
28. Laura Maruster. *A Machine Learning Approach to Understand Business Processes*. PhD thesis, Eindhoven University of Technology, Eindhoven, The Netherlands, 2003.
29. Laura Maruster, A. J. M. M. Weijters, Wil M. P. van der Aalst, and Antal Bosch. A Rule-Based Approach for Process Discovery: Dealing with Noise and Imbalance in Process Logs. *Data Min. Knowl. Discov.*, 13(1):67–87, 2006.
30. Sheldon M. Ross. *Introduction to Probability Models, Tenth Edition*. Academic Press, Inc., Orlando, FL, USA, 2009.
31. Wil M. P. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer Verlag, 2011.
32. Wil M. P. van der Aalst, Ana Karla Alves de Medeiros, and A. J. M. M. Weijters. Genetic Process Mining. In Gianfranco Ciardo and Philippe Darondeau, editors, *Applications and Theory of Petri Nets 2005, 26th International Conference, ICATPN 2005, Miami, FL, USA, June 20-25, 2005, Proceedings*, volume 3536 of *Lecture Notes in Computer Science*, pages 48–69. Springer, 2005.
33. Wil M. P. van der Aalst, Boudewijn F. van Dongen, J. Herbst, Laura Maruster, G. Schimm, and A. J. M. M. Weijters. Workflow Mining: A Survey of Issues and Approaches. *Data and Knowledge Engineering*, 47(2):237–267, 2003.
34. Wil M. P. van der Aalst and K. M. van Hee. *Workflow Management: Models, Methods, and Systems*. MIT Press, Cambridge, MA, USA, 2004.
35. Wil M. P. van der Aalst and A. J. M. M. Weijters. Process Mining: A Research Agenda. *Computers in Industry*, 53(3):231–244, 2004.
36. Wil M. P. van der Aalst, A. J. M. M. Weijters, and Laura Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.
37. Boudewijn F. van Dongen, Ana Karla Alves de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters, and Wil M. P. van der Aalst. The ProM Framework: A New Era in Process Mining Tool Support. In G. Ciardo and P. Darondeau, editors, *Application and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 444–454. Springer, 2005.
38. Boudewijn F. van Dongen, Ana Karla Alves de Medeiros, and Lijie Wen. Process Mining: Overview and Outlook of Petri Net Discovery Algorithms. *T. Petri Nets and Other Models of Concurrency*, 2:225–242, 2009.
39. Boudewijn F. van Dongen, N. Busi, G. M. Pinna, and Wil M. P. van der Aalst. An Iterative Algorithm for Applying the Theory of Regions in Process Mining. BETA Working Paper Series, WP 195, Eindhoven University of Technology, Eindhoven, the Netherlands, 2007.
40. H.M.W. Verbeek, Joos C.A.M. Buijs, Boudewijn F. van Dongen, and Wil M. P. van der Aalst. ProM: The Process Mining Toolkit. In *BPM Demos 2010*, volume 615 of *CEUR-WS*, 2010.
41. A. J. M. M. Weijters and Wil M. P. van der Aalst. Workflow Mining: Discovering Workflow Models from Event-Based Data. In C. Dousson, F. Höppner, and R. Quiniou, editors, *Proceedings of the ECAI Workshop on Knowledge Discovery from Temporal and Spatial Data*, pages 78–84, Lyon, France, 2002.
42. Lijie Wen, Wil M. P. van der Aalst, Jianmin Wang, and Jiaguang Sun. Mining Process Models with Non-free-choice Constructs. *Data Mining and Knowledge Discovery*, 15(2):145–180, 2007.

43. Hui Xiong, Gaurav Pandey, Michael Steinbach, and Vipin Kumar. Enhancing Data Analysis with Noise Removal. *IEEE Transactions on Knowledge and Data Engineering*, 18(3):304–319, 2006.