Entropia: Measuring Log Representativeness and Generalization of Discovered Process Models

Anandi Karunaratne[1], Artem Polyvyanyy[1] and Alistair Moffat[1]

[1]*The University of Melbourne, Victoria 3010, Australia*

Abstract

This paper extends *Entropia*, a command-line tool for performing conformance checking between process models and corresponding event logs. The extension introduces functionalities for estimating the generalization of a process model presented as a directly-follows graph using the bootstrap generalization method and evaluating the representativeness of an event log.

Keywords

process mining, conformance checking, generalization, representativeness

Metadata description	Value
Tool name	Entropia
Current version	1.7
Legal code license	LGPL-3.0
Languages, tools and services used	JDK 1.8, Apache Maven
Supported operating environment	Any platform supporting JDK 1.8 and Apache Maven
Download/Demo URL	https://github.com/jbpt/codebase
Documentation URL	https://github.com/jbpt/codebase/blob/master/jbpt-pm/entropia/guide.pdf
Source code repository	https://github.com/jbpt/codebase
Screencast video	https://youtu.be/g6n0w5UrT8A

1. Introduction

Process mining is a field focused on analyzing and improving business processes based on event logs recorded during the executions of these processes. It involves three major areas: discovery, conformance checking, and enhancement [1]. *Conformance checking* studies ways to compare the behavior recorded in an event log with that described by a normative process model [2]. While *precision* and *recall* are widely used quality measures in process mining, *generalization* represents another important aspect of conformance assessment [3], quantifying the likelihood that the model describes future process executions. Given the need to account for unseen behavior not present in the event log and the fact that real-world systems that generate event logs are often unknown, few generalization measures have been proposed, and tool support is limited.

ICPM 2024 Tool Demonstration Track, October 14-18, 2024, Kongens Lyngby, Denmark

anandik@student.unimelb.edu.au (A. Karunaratne); artem.polyvyanyy@unimelb.edu.au (A. Polyvyanyy); ammoffat@unimelb.edu.au (A. Moffat)

0009-0008-5250-3888 (A. Karunaratne); 0000-0002-7672-1643 (A. Polyvyanyy); 0000-0002-6638-0232 (A. Moffat)

Entropia [4] is a family of command-line conformance checking tools. These tools are publicly available and can compute precision and recall between an event log and a model, two logs, or two models. In this paper, we extend the capabilities of Entropia. The extension is twofold. First, we add a tool for computing bootstrap generalization of a process model discovered from an event log [5]. Second, we introduce functionalities for calculating the representativeness of a given event log [6, 7].

2. Entropia

This section provides an overview of the Entropia tool [4], focusing on its usage and maturity.

2.1. Overview

Entropia implements functionality for quantifying precision and recall between models and event logs. The approach is based on comparing *relevant* behavior (`rel`), e.g., behavior recorded in an event log, and *retrieved* behavior (`ret`), e.g., behavior described in a process model discovered from the event log. Precision compares the magnitude of common behavior to that of `ret`, while recall compares it to `rel`. These magnitudes are quantified using techniques utilizing different notions of entropy.

2.2. Usage

Entropia tool version 1.7 (as of August 2024) can be invoked using the following command:

```
java -jar jbpt-pm-entropia-1.7.jar <options>
```

Table 1
Key CLI options of Entropia.

Option (full)	Option	Parameter	Description
`--help`	`-h`		print help message
`--silent`	`-s`		run the tool in the silent mode
`--version`	`-v`		get the version of the tool
`--relevant`	`-rel`	`<path>`	model that describes relevant traces
`--retrieved`	`-ret`	`<path>`	model that describes retrieved traces

Entropia's key command-line interface (CLI) options are listed in Table 1. Option `-h` displays a help message, while `-v` shows the tool's version. The `-s` option enables silent mode, suppressing all but the final result output. Central to the tool's purpose are the `-rel` and `-ret` options. They are used to specify the file paths for the models that describe relevant and retrieved behaviors, respectively, which are used to calculate conformance.

Beyond these basic options, the tool includes options for selecting specific conformance measures to apply to the input data, as well as parameters for fine-tuning the configuration of these measures. The measures, their capabilities, and example usage are presented in [4].

Entropia supports multiple input formats, including standard formats like eXtensible Event Stream (XES) [8] and Petri Net Markup Language (PNML)[9], and specialized formats specific to Entropia for capturing stochastic Petri Nets (sPNMLs), Directly-Follows Graphs (DFGs), and Stochastic Deterministic Finite Automata (SDFAs).

2.3. Maturity

The development of Entropia began in August 2017 with an entropy-based method for assessing precision and recall [10]. This tool is integrated into the jBPT library [11], a collection of open-source business process technologies that was initiated in January 2009.

The approach for evaluating precision and recall presented in [10] treats compared models and event logs as sets of traces, where shared behavior is identified only if the models describe identical traces. The quantification of behavior captured by each model and their shared behavior is determined using topological entropy [12] of the trace collections. Expanding on this work, a partial match approach [13] has been proposed, considering all subtraces within the traces for precision and recall calculations. Shared behavior is defined as all sequences of actions that are subtraces in both collections, measuring all common subsequences of actions in the models being compared. Kalenkova and Polyvyanyy [14] introduced a new approach, which allows users to specify the maximum number of actions that can be skipped within a trace when identifying shared subtraces, making it more flexible than the two previous approaches [10, 13] which represent two extremes of excluding subtraces entirely and considering all possible subtraces.

Extending the usage of entropy to stochastic process mining, which considers both control flow and behavior frequency, Leemans and Polyvyanyy [15] offer measures of stochastic recall and precision, by converting the log and model into stochastic deterministic finite automata and construct their conjunction. Additionally, the entropic relevance measure [16] calculates a stochastic conformance measure based on the average number of bits needed to compress a trace from the log using the model's encoded likelihood of traces.

3. Bootstrap Generalization

Bootstrap generalization [5] presents a framework for estimating the generalization of a process model discovered from an event log generated by an unknown system. This approach uses bootstrapping techniques from computational statistics. Building upon this theoretical foundation, we introduce a tool designed to compute this generalization measure. The following sections describe its usage and examples.

3.1. Usage

To calculate the bootstrap generalization, `-bgen` (long option `--bootstrap-gen`) should be invoked with two primary inputs: the process model for which generalization is to be estimated and an event log that serves as a sample of the system behavior. In this framework, the event log represents the relevant behavior, while the model embodies the behavior "retrieved" from this log.

The event log should be provided in the XES format [8] using the `-rel` option of the tool. The model file should be specified using the `-ret` option. Due to the characteristics of the log sampling with breeding method used to estimate system behavior [5], the current implementation restricts the model representation to DFGs, which must adhere to a specific JSON[1]

[1]https://www.json.org/json-en.html

file format [16], as detailed in Listing 1. This structured format comprises two main elements: *nodes* and *arcs*. Each node in the DFG is characterized by a string label representing the activity name, a numerical frequency indicating execution frequencies of the activities, and a unique numerical identifier. Arcs, which represent the connections between nodes, are defined by their source and target nodes (both specified as numbers) and the frequency of the arc's occurrence. To represent the process boundaries, the start and end nodes should be explicitly labeled as "INPUT" and "OUTPUT" respectively.

3.2. Configuration

New options are introduced to support bootstrap generalization estimation, including sample size (n); number of samples (m); number of log generations (g), crossover subtrace length (k), and the breeding probability (p). These parameters, detailed in [5], are optional for tool usage, allowing to compute bootstrap generalization without in-depth knowledge of bootstrapping or its parameters. To allow this, we have defined default values, derived from an experiment[2] consumed 2.5 CPU years, specified in the table 2.

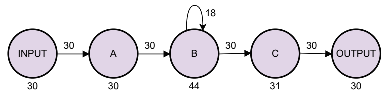

Figure 1: Example DFG.

```
{ "nodes": [
    { "id": 1, "label": "INPUT",  "freq": 30 },
    { "id": 2, "label": "A",      "freq": 30 },
    { "id": 3, "label": "B",      "freq": 44 },
    { "id": 4, "label": "C",      "freq": 31 },
    { "id": 5, "label": "OUTPUT", "freq": 30 }
  ],
  "arcs": [
    { "from": 1, "to": 2, "freq": 30 },
    { "from": 2, "to": 3, "freq": 30 },
    { "from": 3, "to": 3, "freq": 18 },
    { "from": 3, "to": 4, "freq": 30 },
    { "from": 4, "to": 5, "freq": 30 }
  ]
}
```

Listing 1: Corresponding JSON representation.

In addition to the parameters presented in [5], we introduce a new threshold parameter for confidence interval of bootstrap sample values (ep), which serves as a termination criterion for bootstrapping. The bootstrapping process calculates the 95% confidence interval for precision and recall values of the bootstrap samples and the model. The computation terminates when this confidence interval value falls below the specified threshold, with a default value set at 0.01.

Termination of the bootstrapping process depends on the parameters provided. With both m and ep, the process terminates when either the sample count reaches m or the confidence interval of the samples falls below ep. If only ep is provided, it runs until the confidence interval is lower than the specified ep. If neither m nor ep is provided, the default ep of 0.01 is used.

3.3. Examples

This section provides example usages of the -bgen command.

To calculate the generalization of a DFG with default configurations, use:

```
-bgen -rel=/path/to/log.xes -ret=/path/to/model.json
```

If customization of the process is required, the configuration can be adjusted, for example:

```
-bgen -rel=/path/to/log.xes -ret=/path/to/model.json -m=1000 -ep=0.005
```

Here, bootstrapping continues until 1,000 samples are reached, unless the confidence interval for both precision and recall of the bootstrap samples falls below 0.005, triggering early termination.

Another configuration might be as follows:

```
-bgen -rel=/path/to/log.xes -ret=/path/to/model.json -n=1000 -p=0.5 -m=100
```

This configuration generates 100 bootstrap samples, each containing 1,000 traces, with new traces being generated 50% of the time, and existing traces used the remaining 50%.

[2]Refer to https://doi.org/10.26188/26410486 for the data.

Table 2
CLI options for bootstrap generalization.

Option (full)	Option	Parameter	Default value
`--bootstrap-gen`	`-bgen`		
`--sample-size`	`-n`	`<sample-size>`	8 × log size
`--num-of-samples`	`-m`	`<number-of-samples>`	
`--num-of-generations`	`-g`	`<number-of-generations>`	0.5 × log size
`--subtrace-length`	`-k`	`<subtrace-length>`	2
`--breeding-probability`	`-p`	`<breeding-probability>`	1.0
`--epsilon`	`-ep`	`<threshold>`	0.01

4. Event Log Representativeness

The other addition to Entropia is event log representativeness measures. These measures allow users to compute representativeness in terms of completeness, coverage, [7] and Log Representativeness Approximation (LRA) [6]. This new feature enables users to assess the representativeness of a log with respect to activities, directly-follows relations, and traces present in the log.

4.1. Usage

The log representativeness analysis is invoked via the `-l` option (or its long form `–log`) followed by the path to the log file in XES format [8]. Users can specify particular representativeness aspects such as completeness (`-com`), coverage (`-cov`), or LRA (`-lra`). Furthermore, the analysis can be focused on specific event data aspects: activities (`-act`), directly-follows relations (`-dfr`), or traces (`-tr`). A summary of these options is presented in table 3.

Table 3
CLI options for log representativeness.

Option (full)	Option	Parameter	Description
`--log`	`-l`	`<path>`	invoke representativeness analysis of a log
`--completeness`	`-com`		calculate completeness of the log
`--coverage`	`-cov`		calculate coverage of the log
`--log-rep-approx`	`-lra`		calculate LRA of the log
`--activity`	`-act`		conduct activity-based analysis
`--df-relation`	`-dfr`		conduct DF-relation-based analysis
`--trace`	`-tr`		conduct trace-based analysis

4.2. Examples

To compute the completeness and coverage of activities in an event log:

```
-l=/path/to/log.xes -com -cov -act
```

To compute the LRA specifically for traces in an event log:

```
-l=/path/to/log.xes -lra -tr
```

With no options, completeness, coverage, and LRA for all event data aspects are calculated:

```
-l=/path/to/log.xes
```

5. Conclusion

The extension of *Entropia* introduces new functionalities for estimating process model generalization and evaluating event log representativeness. These additions complement the tool's existing entropy-based conformance checking measures, enhancing its analytical capabilities.

References

[1] W. M. P. van der Aalst, et al., Process mining manifesto, in: BPM Workshops, Springer, 2012, pp. 169–194.

[2] J. Carmona, B. van Dongen, A. Solti, M. Weidlich, Conformance Checking: Relating Processes and Models, Springer, 2018.

[3] W. M. P. van der Aalst, Relating process models and event logs—21 conformance propositions, in: ATAED, CEUR-WS.org, 2018, pp. 56–74.

[4] A. Polyvyanyy, et al., Entropia: A family of entropy-based conformance checking measures for process mining, in: ICPM Doc. cons. & tool demo. trck., CEUR-WS.org, 2020, pp. 39–42.

[5] A. Polyvyanyy, A. Moffat, L. García-Bañuelos, Bootstrapping generalization of process models discovered from event data, in: CAiSE, 2022, pp. 36–54.

[6] A. Karunaratne, A. Polyvyanyy, A. Moffat, The role of log representativeness in estimating generalization in process mining, in: ICPM, IEEE, 2024. to appear.

[7] M. Kabierski, M. Richter, M. Weidlich, Addressing the log representativeness problem using species discovery, in: ICPM, 2023, pp. 65–72.

[8] IEEE Standard for eXtensible Event Stream (XES) for achieving interoperability in event logs and event streams, IEEE Std. 1849-2016 (2016) 1–50.

[9] J. Billington, S. Christensen, K. M. van Hee, E. Kindler, O. Kummer, L. Petrucci, R. Post, C. Stehno, M. Weber, The petri net markup language: Concepts, technology, and tools, in: ICATPN, Springer, 2003, pp. 483–505.

[10] A. Polyvyanyy, A. Solti, M. Weidlich, C. Di Ciccio, J. Mendling, Monotone precision and recall measures for comparing executions and specifications of dynamic systems, ACM Trans. Soft. Eng. and Meth. 29 (2020) 17:1–17:41.

[11] A. Polyvyanyy, M. Weidlich, Towards a compendium of process technologies - the jbpt library for process model analysis, in: CAiSE Forum, CEUR-WS.org, 2013, pp. 106–113.

[12] T. Ceccherini-Silberstein, A. Machi, F. Scarabotti, On the entropy of regular languages, Theor. Comput. Sci. 307 (2003) 93–102.

[13] A. Polyvyanyy, A. A. Kalenkova, Monotone conformance checking for partially matching designed and observed processes, in: ICPM, IEEE, 2019, pp. 81–88.

[14] A. A. Kalenkova, A. Polyvyanyy, A spectrum of entropy-based precision and recall measurements between partially matching designed and observed processes, in: ICSOC, Springer, 2020, pp. 337–354.

[15] S. J. J. Leemans, A. Polyvyanyy, Stochastic-aware conformance checking: An entropy-based approach, in: CAiSE, Springer, 2020, pp. 217–233.

[16] A. Polyvyanyy, A. Moffat, L. García-Bañuelos, An entropic relevance measure for stochastic conformance checking in process mining, in: ICPM, IEEE, 2020, pp. 97–104.