Testing Representational Biases

Prof.dr.ir. Wil van der Aalst
www.vdaalst.com

Process discovery is a challenging task. There are dozens, if not hundreds, of process discovery algorithms. These algorithms may use very different representations during the discovery process. The so-called *representational bias* enables or disables the discovery of certain process constructs. When using something as simple as Petri nets there are already several different representational biases possible.

- Is it possible to discover process models where multiple transitions refer to the same activity?
- Is it possible to discover process models with invisible transitions (e.g. to model skips)?
- Can the discovery technique handle non-block structured processes?
- Etc.

These choices may seem innocent, but they have a huge impact on what can be discovered. For example, try to mine the log [<a,b,c>,<a,c>] not allowing for duplicate and silent transitions/activities. Note that the representational bias refers to the target representation used *during* discovery. After discovery one may convert one notation to another, but this irrelevant for the bias during discovery. For example, subclasses of discovered Petri nets generated by some algorithm can be mapped onto BPMN, but this does not imply that the discovery technique could discovery all possible BPMN models.

Process discovery often needs to deal with *erratic and infrequent behavior*. The discovery technique may want to expose such behavior or just visualize the mainstream behavior. However, the erratic and infrequent behavior should never conceal the more structured parts of the process. This is often challenging for discovery techniques. Some are no longer able to discover the main flow properly after inserting one randomly executed activity.

To test the effect of the representational bias used and the effect of erratic and infrequent behavior added to highly regular behavior, we have created a set of 120 event logs. For each log we know the underlying model. Therefore, we have a “gold standard model” that serves as a reference. For real-life event logs we do not have such a reference model. Moreover, quality criteria like precision and generalization are still subject to discussion. Therefore, we created these synthetic event logs.

The collection has the following characteristics:

- Each event log is available in the form of a **CSV** file and a **XES** file.
- For each example process we generate **five log files**. This is for cross-validation purposes. The last letter of the file name (a-e) refers to the instance of the event log.
- The number in the file name refers to the number of instances (in this collection we always generated 1000 cases, but it is easy to create larger, smaller, or more event logs).
- We consider four process structures that may be difficult for some techniques:
    - **Parallel** (five activities that can be executed in any order).
    - **Skip** (three activities where the middle one can be skipped).
    - **Duplicates** (an activity appears at two positions in the process).
    - **Non-free-choice** (a choice in the later part of the process depends on an earlier choice).
- The processes described above are small and simple. However, many discovery techniques have difficulties with one or two of these processes because of their representational bias.
- To complicate matters we also added **erratic and infrequent behavior** by adding an infrequent activity **x**. We consider six settings:
    - **No noise** (activity x does not appear in the event log)
    - **Local very infrequent noise** (activity x very seldom appears at a particular place in the process)
    - **Local infrequent noise** (activity x sometimes appears at a particular place in the process)
    - **Global very infrequent noise** (activity x very seldom appears at some random place in the process)
    - **Global infrequent noise** (activity x sometimes appears at some random place in the process)
    - **Semi-local noise** (activity x sometimes appears at one of two places in the process)

The naming of files is self-explanatory. For example:

- **duplicates-log-little-local-noise-1000e.csv** refers to a log file with 1000 cases stored in a CSV file. The representational bias tested is duplication of activities (an activity appears at two positions in the process). Local very infrequent noise has been added (activity x very seldom appears at a particular place in the process). The letter e indicates that this is the fifth log for this process.
- **nfc-log-global-noise-1000a.xes** refers to a log file stored in XES format. The representational bias tested is non-free-choice behavior (a choice in the later part of the process depends on an earlier choice). Global infrequent noise has been added (activity x very seldom appears at some random place in the process). The letter a indicates that this is the first log for this process.

CPN Files

The event logs included are rather small. If you need even bigger datasets, you can generate these yourself using the CPN Tools sources files included (*.cpn).

I hope that these event logs will contribute to better process discovery tools that can deal with different representational biases and different types of noisy behavior.

Best regards,

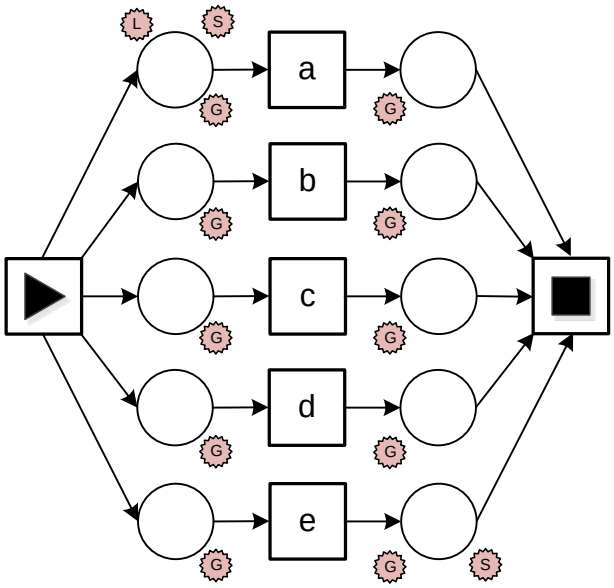
Trento October 2017

Prof.dr.ir. Wil van der Aalst
www.vdaalst.com

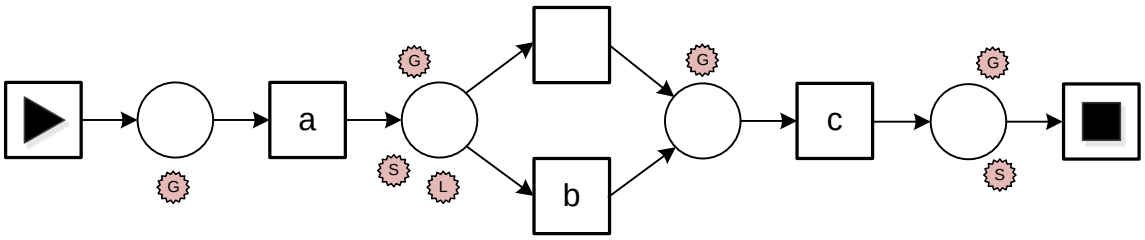
Appendix

In the remainder the process models are depicted in terms of labeled Petri nets. Using the letters L=local, G=Global, and S=Semi-local we indicate where activity x may occur.

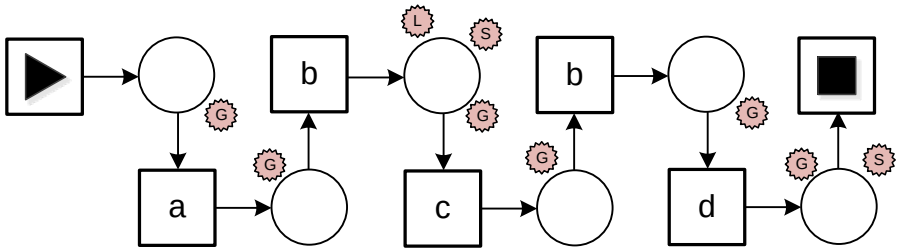
Parallel (five activities that can be executed in any order)



Skip (three activities where the middle one can be skipped)



Duplicates (an activity appears at two positions in the process)



Non-free-choice (a choice in the later part of the process depends on an earlier choice)