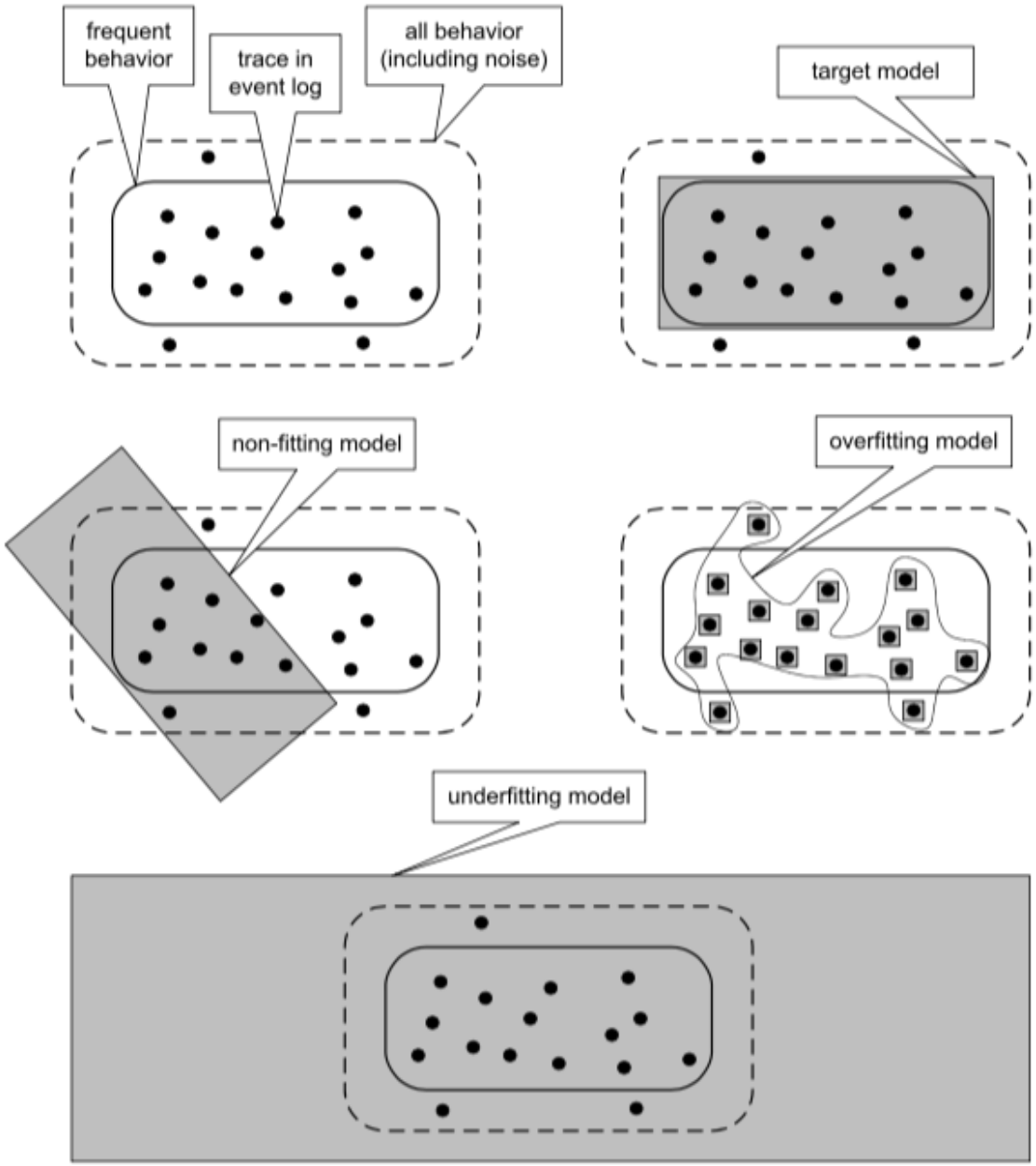
Advanced Process Discovery Techniques

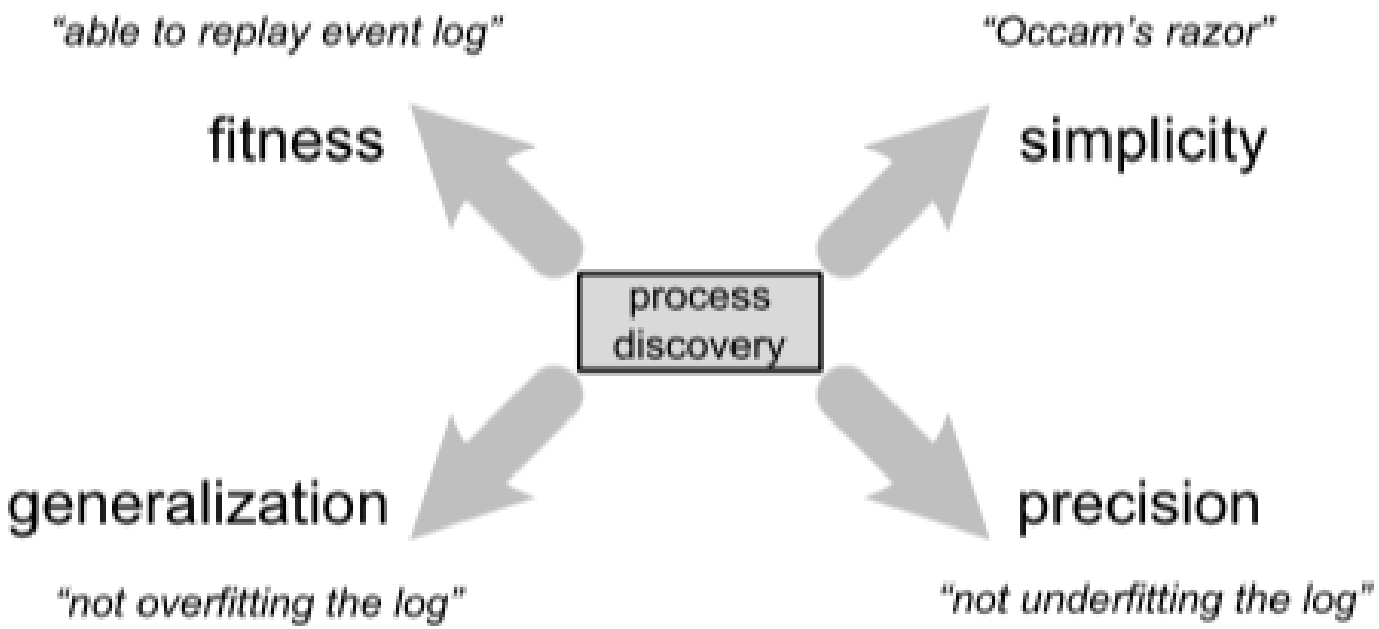
Barbara Re

Process Mining

Four Competing Quality Criteria



Quality Dinensions



Fitness

A model has a perfect fitness if all traces in the log can be replayed by the model from beginning to end

There are various ways of defining fitness.

- It can be defined at the **case level**, e.g., the fraction of traces in the log that can be fully replayed.
- It can also be defined at the **event level**, e.g., the fraction of events in the log that are indeed possible according to the model.

Simplicity

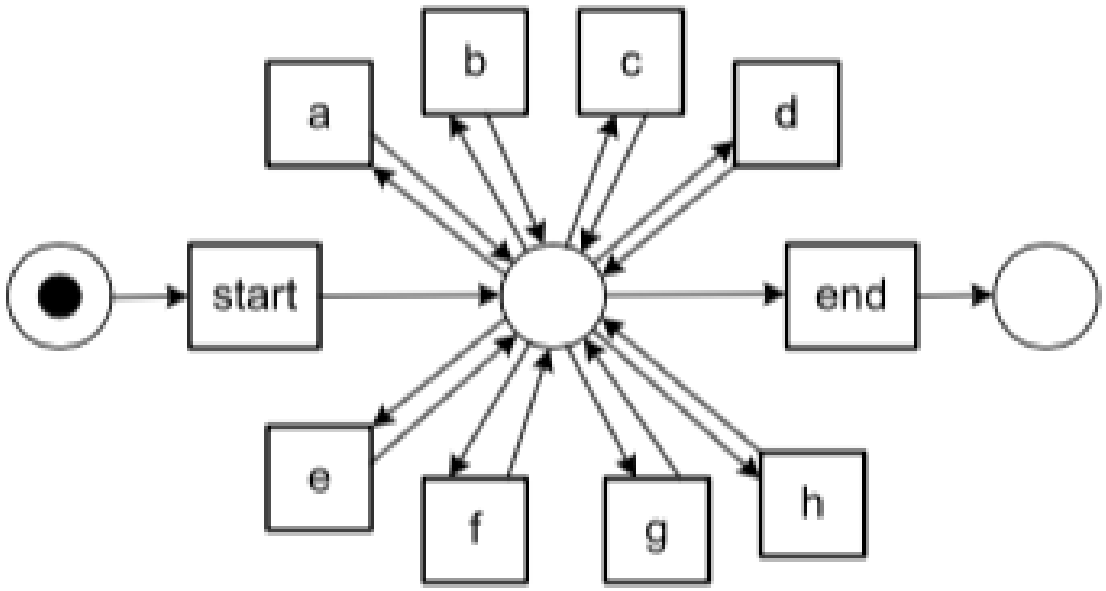
The simplicity dimension refers to Occam's Razor. In the context of process discovery this means that the simplest model that can explain the behavior seen in the log, is the best model.

The principle states that one should not increase, beyond what is necessary, the number of entities required to explain anything, i.e., one should look for the simplest model that can explain what is observed in the data set. This principle is related to finding a natural balance between overfitting and underfitting.

- The complexity of the model could be defined by the number of nodes and arcs in the underlying graph
- Also more sophisticated metrics can be used, e.g., metrics that take the **structuredness** or **entropy** of the model into account.

Simplicity

Fitness and simplicity alone are not adequate!



The flower Petri net allows for any sequence starting with start and ending with end and containing any ordering of activities in between

This model allows for all event logs used to introduce the α-algorithm., the added start and end activities are just a technicality to turn the **flower model** into a WF-net

From flower model to enumerating model

The flower model does not contain any knowledge other than the activities in the log

The flower model can be constructed based on the occurrences of activities only, and the resulting model is simple and has a perfect fitness

Considering fitness and simplicity the flower model is acceptable, this shows that they are necessary, but not sufficient

If the flower model is on one end of the spectrum, then the enumerating model is on the other end of the spectrum

The enumerating model of a log simply lists all the sequences possible, i.e., there is a separate sequential process fragment for each trace in the model

- At the start there is one big XOR split selecting one of the sequences and at the end these sequences are joined using one big XOR join
- If such a model is represented by a Petri net and all traces are unique, then the number of transitions is equal to the number of events in the log.

The enumerating model is simply an encoding of the log. Such a model is complex but, like the flower model, has a perfect fitness.

From flower model to enumerating model

Extreme models:

- the flower model (anything is possible)
- the enumerating model (only the log is possible)

show the need for two additional dimensions!!

Precision and Generalization

A model is precise if it does not allow for too much behavior

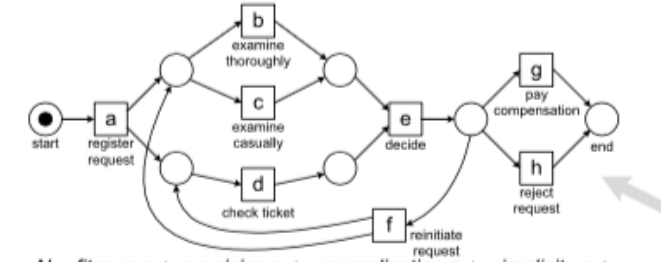
- The flower model lacks precision
- A model that is not precise is underfitting
- Underfitting is the problem that the model over-generalizes the example behavior in the log, i.e., the model allows for behaviors very different from what was seen in the log.

A model should generalize and not restrict behavior

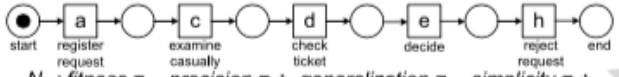
- The enumerating model lacks generalization
- A model that does not generalize is overfitting
- Overfitting is the problem that a very specific model is generated whereas it is obvious that the log only holds example behavior, i.e., the model explains the particular sample log, but a next sample log of the same process may produce a completely different process model.

Process mining algorithms need to strike a balance between overfitting and underfitting

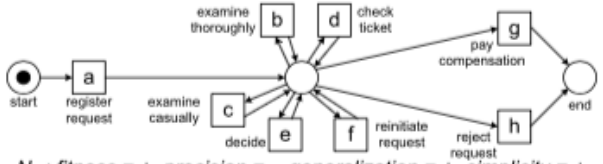
Four alternative models for the same log



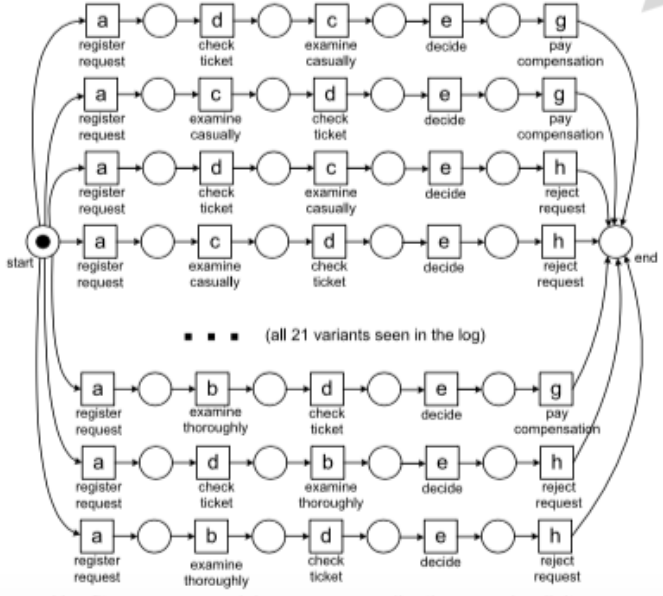
N_1 : *fitness* = +, *precision* = +, *generalization* = +, *simplicity* = +



N_2 : *fitness* = -, *precision* = +, *generalization* = -, *simplicity* = +



N_3 : *fitness* = +, *precision* = -, *generalization* = +, *simplicity* = +



N_4 : *fitness* = +, *precision* = +, *generalization* = -, *simplicity* = -

#	trace
455	acdeh
191	abdeg
177	adceh
144	abdeh
111	acdeg
82	adceg
56	adbeh
47	acdefdbeh
38	adbeg
33	acdefbdeh
14	acdefbdeg
11	acdefdbeg
9	adcefcdeh
8	adcefdbeh
5	adcefbdeg
3	acdefbdefdbeg
2	adcefdbeg
2	adcefbdefbdeg
1	adcefdbefbdeh
1	adbefbdefdbeg
1	adcefdbefcdefdbeg
1391	

Characteristics of Process Discovery Alghoritms

- Representational Bias
  - Inability to represent concurrency
  - Inability to deal with (arbitrary) loops
  - Inability to represent silent actions
  - Inability to represent duplicate actions
  - Inability to model OR-splits/joins
  - Inability to represent non-free-choice behavior
  - Inability to represent hierarchy
- Ability to Deal With Noise
- Completeness Notion Assumed
- Approach Used
  - Direct Algorithmic Approaches
  - Two-Phase Approaches
  - Divide-and-Conquer Approaches
  - Computational Intelligence Approaches
  - Partial Approaches

Heuristic Mining

Learning the Dependency Graph

Definition 7.1 (Dependency measure) Let L be an event log[1] over $\mathscr{A}$ and $a, b \in \mathscr{A}$. $|a >_L b|$ is the number of times a is directly followed by b in L, i.e.,

$$|a >_L b| = \sum_{\sigma \in L} L(\sigma) \times \left|\left\{1 \leq i < |\sigma| \;\middle|\; \sigma(i) = a \wedge \sigma(i+1) = b\right\}\right|$$

$|a \Rightarrow_L b|$ is the value of the dependency relation between a and b:

$$|a \Rightarrow_L b| = \begin{cases} \frac{|a>_L b| - |b>_L a|}{|a>_L b| + |b>_L a| + 1} & \text{if } a \neq b \\ \frac{|a>_L a|}{|a>_L a| + 1} & \text{if } a = b \end{cases}$$

Frequency of the directly follows

$\lvert >_L \rvert$	a	b	c	d	e
a	0	11	11	13	5
b	0	0	10	0	11
c	0	10	0	0	11
d	0	0	0	4	13
e	0	0	0	0	0

Genetic Process Mining

Region Based Mining

Inductive Mining

Historical Perspective