Process Discovery an Introduction

Barbara Re

Process Mining

Problem Statement

Focusing on discovery the control-flow perspective

Definition (General process discovery problem)

Let $\mathcal{L}$ be an event log. A process discovery algorithm is a function that maps $\mathcal{L}$ onto a process model such that the model is representative for the behavior seen in the event log. The challenge is to find such an algorithm.

This definition does not specify what kind of process model should be generated, e.g., a BPMN, EPC, YAWL, or Petri net model

To make things more concrete:

- We define the target to be a Petri net model
- We use a simple event log as input

A simple event log $\mathcal{L}$ is a multi-set of traces over $\mathcal{A}$, i.e., $\mathcal{L} \in \mathbb{B}(\mathcal{A}^*)$
$\mathcal{L}1 = [\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$

The goal is to discover a Petri Net that can replay event log $\mathcal{L}1$
—> Ideally, the Petri Net is a **sound WF-Net**

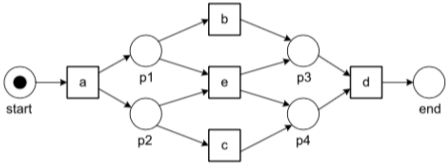
Process discovery algorithm

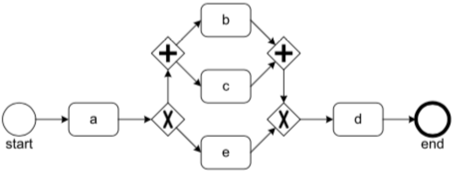
Definition (Specific process discovery problem)

A process discovery algorithm is a function γ that maps a log $\mathcal{L} \in \mathbb{B}(\mathcal{A}^*)$ onto a marked Petri net $\gamma(\mathcal{L}) = (\mathcal{N}, \mathcal{M})$. Ideally, $\mathcal{N}$ is a sound WF-Net and all traces in $\mathcal{L}$ correspond to possible firing sequences of $(\mathcal{N}, \mathcal{M})$.

Function γ defines a so-called **Play-In technique**

Based on $\mathcal{L}1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$, a process discovery algorithm γ could discover the following WF-Net





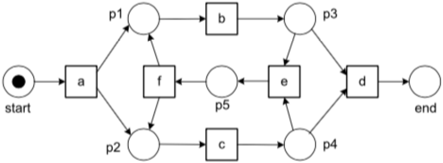
It is easy to see that the WF-Net can indeed replay all traces in the event log

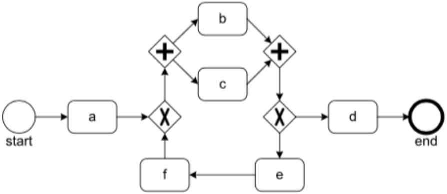
Discovery into practice

$\mathcal{L}2 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^4, \langle a, b, c, e, f, b, c, d\rangle^2,$
$\langle a, b, c, e, f, c, b, d\rangle^2, \langle a, c, b, e, f, b, c, d\rangle^2, \langle a, c, b, e, f, b, c, e, f, c, b, d\rangle]$

$\mathcal{L}2$ is a simple event log consisting of 13 cases represented by 6 different traces

Based on event log $\mathcal{L}2$, let's discover the following WF-Net!!!





This WF-Net can indeed replay all traces in the log

Not all firing sequences of $\mathcal{N}2$ correspond to traces in $\mathcal{L}2$, (e.g. the firing sequence $\langle a, c, b, e, f, c, b, d\rangle$ is a firing sequence that is not in the $\mathcal{L}2$ traces

Discovered net are sound WF-Nets

WF-Nets are a natural **subclass of Petri nets** tailored toward the modeling and analysis of **operational processes**

A process model describes the life-cycle of one case

WF-Nets explicitly model the **creation** and the **completion** of the cases:

- The **creation** is modeled by putting a token in the unique **source** place i
- The **completion** is modeled by reaching the state marking the unique **sink** place o

Given a unique source place i and a unique sink place o, the **soundness requirement** follows naturally

$\mathcal{WN}$ is sound iff:

- safeness – places cannot hold multiple tokens at the same time
- proper completion – for any marking $\mathcal{M} \in [\mathcal{WN}, [i]\rangle, o \in \mathcal{M}$ implies $\mathcal{M} = [o]$
- option to complete – for any marking $\mathcal{M} \in [\mathcal{WN}, [i]\rangle, [o] \in [\mathcal{WN}, \mathcal{M}\rangle$
- absence of dead parts – $(\mathcal{WN}, [i])$ contains no dead transitions (i.e., for any $t \in \mathcal{T}$, there is at least a firing sequence enabling t)

Quality criteria

The discovered model should be **representative** for the behavior seen in the event log

- **Fitness** - The discovered model should allow for the behavior seen in the event log
- **Precision** - The discovered model should not allow for behavior completely un-related to what was seen in the event log
- **Generalization** - The discovered model should generalize the example behavior seen in the event log
- **Simplicity** - The discovered model should be as simple as possible

The challenge is to **balance** the four **quality criteria** is needed

- **Precision** is related to the notion of underfitting –> A model having a poor precision is underfitting, i.e., it allows for behavior that is very different from what was seen in the event log
- **Generalization** is related to the notion of overfitting –> An overfitting model does not generalize enough, i.e., it is too specific and too much driven by the event log

A **trade-off** between trade-off between underfitting and overfitting is obvious

A Simple Algorithm for Process Discovery

α-algorithm

The α-algorithm focus on **control flow** such as the ordering of the activities

The α-algorithm is one of the first algorithm suitable to **discovery model including concurrency** (e.g. loops, parallel part, choice) while guarantee certain properties

The α-algorithm should not be seen as a very practical mining technique as it has problems with:

- noise
- infrequent/incomplete behavior
- complex routing constructs

INPUT: a simple event log $\mathcal{L}$ over $\mathcal{A}$
OUTPUT: a marked Petri net $\alpha(\mathcal{L}) = (\mathcal{N}, \mathcal{M})$

The α-algorithm scans the event log for particular **patterns**

We distinguish four **log-based ordering relations** to capture relevant **patterns in the log**

For any log $\mathcal{L}$ over $\mathcal{A}$ and $x, y \in \mathcal{A}$, $x >_L y$ (direct succession), $x \rightarrow_L y$ (casuality), $x \|_L y$ (parallel), $x \#_L y$ (choice) i.e., precisely one of these relations holds for any pair of activities

α-algorithm: ordering relations

- **Direct succession**: $x > y$ iff for some case x is directly followed by y
- **Causality**: $x \rightarrow y$ iff $x > y$ and $y \not> x$
- **Parallel**: $x \| y$ iff $x > y$ and $y > x$
- **Choice**: $x \# y$ iff $x \not> y$ and $y \not> x$

$$\mathcal{L}1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$

α-algorithm: ordering relations

- **Direct succession**: $x > y$ iff for some case x is directly followed by y
- **Causality**: $x \rightarrow y$ iff $x > y$ and $y \not> x$
- **Parallel**: $x \| y$ iff $x > y$ and $y > x$
- **Choice**: $x \# y$ iff $x \not> y$ and $y \not> x$

$$\mathcal{L}1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$

$$>_{L_1} = \big\{(a,b), (a,c), (a,e), (b,c), (c,b), (b,d), (c,d), (e,d)\big\}$$

$$\rightarrow_{L_1} = \big\{(a,b), (a,c), (a,e), (b,d), (c,d), (e,d)\big\}$$

$$\#_{L_1} = \big\{(a,a), (a,d), (b,b), (b,e), (c,c), (c,e), (d,a), (d,d), (e,b), (e,c), (e,e)\big\}$$

$$\|_{L_1} = \big\{(b,c), (c,b)\big\}$$

Ordering relationship and footprint of $\mathcal{L}1$

$\mathcal{L}1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$

$$>_{L_1} = \{(a,b), (a,c), (a,e), (b,c), (c,b), (b,d), (c,d), (e,d)\}$$

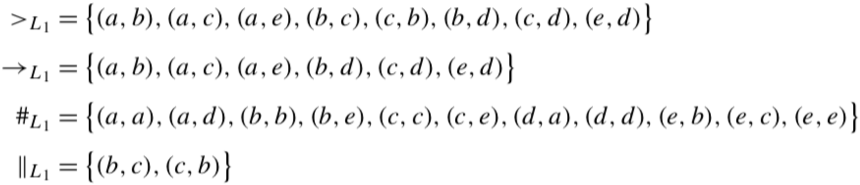
$$\rightarrow_{L_1} = \{(a,b), (a,c), (a,e), (b,d), (c,d), (e,d)\}$$

$$\#_{L_1} = \{(a,a), (a,d), (b,b), (b,e), (c,c), (c,e), (d,a), (d,d), (e,b), (e,c), (e,e)\}$$

$$\|_{L_1} = \{(b,c), (c,b)\}$$

Ordering relationship and footprint of $\mathcal{L}1$

$\mathcal{L}1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$

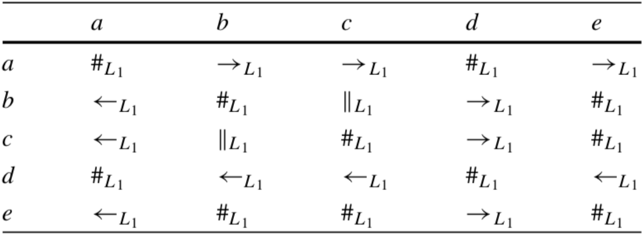


$$>_{L_1} = \{(a,b), (a,c), (a,e), (b,c), (c,b), (b,d), (c,d), (e,d)\}$$

$$\rightarrow_{L_1} = \{(a,b), (a,c), (a,e), (b,d), (c,d), (e,d)\}$$

$$\#_{L_1} = \{(a,a), (a,d), (b,b), (b,e), (c,c), (c,e), (d,a), (d,d), (e,b), (e,c), (e,e)\}$$

$$\|_{L_1} = \{(b,c), (c,b)\}$$



	a	b	c	d	e
a	$\#_{L_1}$	$\rightarrow_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$
b	$\leftarrow_{L_1}$	$\#_{L_1}$	$\|_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$
c	$\leftarrow_{L_1}$	$\|_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$
d	$\#_{L_1}$	$\leftarrow_{L_1}$	$\leftarrow_{L_1}$	$\#_{L_1}$	$\leftarrow_{L_1}$
e	$\leftarrow_{L_1}$	$\#_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$

Ordering relationship and footprint of $\mathcal{L}2$

- **Direct succession**: $x > y$ iff for some case x is directly followed by y
- **Causality**: $x \rightarrow y$ iff $x > y$ and $y \not> x$
- **Parallel**: $x \| y$ iff $x > y$ and $y > x$
- **Choice**: $x \# y$ iff $x \not> y$ and $y \not> x$

$$\mathcal{L}2 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^4, \langle a, b, c, e, f, b, c, d\rangle^2,$$
$$\langle a, b, c, e, f, c, b, d\rangle^2, \langle a, c, b, e, f, b, c, d\rangle^2, \langle a, c, b, e, f, b, c, e, f, c, b, d\rangle]$$

PLEASE DEFINE THE ORDERING RELATIONS

(direct succession) $>_L = \{(..., ...), ...\}$
(casuality) $\rightarrow_L = \{(..., ...), ...\}$
(parallel) $\|_L = \{(..., ...), ...\}$
(choice) $\#_L = \{(..., ...), ...\}$

PLEASE DEFINE THE FOOTPRINT

Ordering relationship and footprint of $\mathcal{L}3$

- **Direct succession**: $x > y$ iff for some case x is directly followed by y
- **Causality**: $x \rightarrow y$ iff $x > y$ and $y \not> x$
- **Parallel**: $x \| y$ iff $x > y$ and $y > x$
- **Choice**: $x \# y$ iff $x \not> y$ and $y \not> x$

$\mathcal{L}3 =$
$[\langle a, b, c, d, e, f, b, d, c, e, g\rangle, \langle a, b, d, c, e, g\rangle^2, \langle a, b, c, d, e, f, b, c, d, e, f, b, d, c, e, g\rangle]$

PLEASE DEFINE THE ORDERING RELATIONS

(direct succession) $>_L = \{(\ldots, \ldots), \ldots\}$
(casuality) $\rightarrow_L = \{(\ldots, \ldots), \ldots\}$
(parallel) $\|_L = \{(\ldots, \ldots), \ldots\}$
(choice) $\#_L = \{(\ldots, \ldots), \ldots\}$

PLEASE DEFINE THE FOOTPRINT

Ordering relationship and footprint of $\mathcal{L}4$

- **Direct succession**: $x > y$ iff for some case x is directly followed by y
- **Causality**: $x \rightarrow y$ iff $x > y$ and $y \not> x$
- **Parallel**: $x \| y$ iff $x > y$ and $y > x$
- **Choice**: $x \# y$ iff $x \not> y$ and $y \not> x$

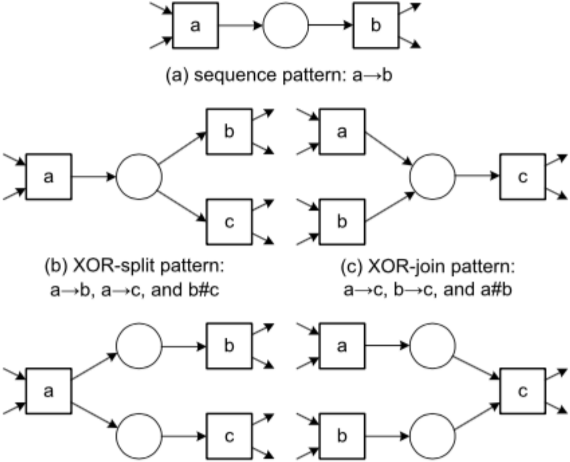
$$\mathcal{L}4 = [\langle a, c, d\rangle^{45}, \langle b, c, d\rangle^{42}, \langle a, c, e\rangle^{38}, \langle b, c, e\rangle^{22}]$$

PLEASE DEFINE THE ORDERING RELATIONS

(direct succession) $>_L = \{(..., ...), ...\}$
(casuality) $\rightarrow_L = \{(..., ...), ...\}$
(parallel) $\|_L = \{(..., ...), ...\}$
(choice) $\#_L = \{(..., ...), ...\}$

PLEASE DEFINE THE FOOTPRINT

Typical process patterns



(a) sequence pattern: a→b

(b) XOR-split pattern:
a→b, a→c, and b#c

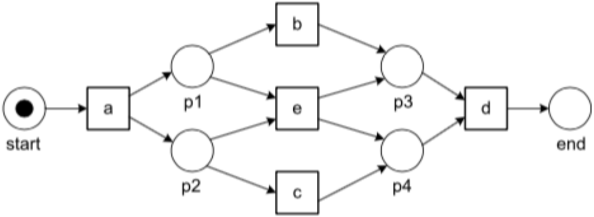
(c) XOR-join pattern:
a→c, b→c, and a#b

(d) AND-split pattern:
a→b, a→c, and b||c

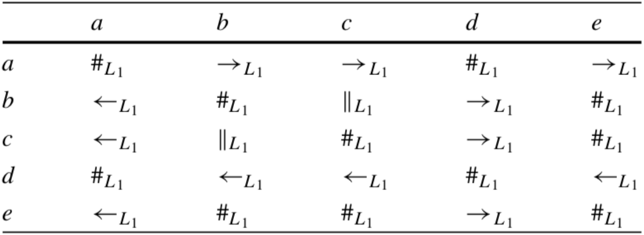
(e) AND-join pattern:
a→c, b→c, and a||b

α-algorithm: footprint of $\mathcal{L}1$

$\mathcal{L}1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$



Model and event log have the same footprint!



	a	b	c	d	e
a	$\#_{L_1}$	$\rightarrow_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$
b	$\leftarrow_{L_1}$	$\#_{L_1}$	$\|_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$
c	$\leftarrow_{L_1}$	$\|_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$
d	$\#_{L_1}$	$\leftarrow_{L_1}$	$\leftarrow_{L_1}$	$\#_{L_1}$	$\leftarrow_{L_1}$
e	$\leftarrow_{L_1}$	$\#_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$

α-algorithm

Let $\mathcal{L}$ be an event log over $\mathcal{T} \subseteq \mathcal{T}$, than $\alpha(\mathcal{L})$ is defined as follows:

1. $\mathcal{T}_{\mathcal{L}} = \{t \in \mathcal{T} | \exists_{\sigma \in \mathcal{L}} t \in \sigma\}$
2. $\mathcal{T}_{\mathcal{I}} = \{t \in \mathcal{T} | \exists_{\sigma \in \mathcal{L}} t = \mathit{first}(\sigma)\}$
3. $\mathcal{T}_{\mathcal{O}} = \{t \in \mathcal{T} | \exists_{\sigma \in \mathcal{L}} t = \mathit{last}(\sigma)\}$
4. $\mathcal{X}_{\mathcal{L}} = \{(\mathcal{A}, \mathcal{B}) | \mathcal{A} \subseteq \mathcal{T}_{\mathcal{L}} \wedge \mathcal{A} \neq \emptyset \wedge \mathcal{B} \subseteq \mathcal{T}_{\mathcal{L}} \wedge \mathcal{B} \neq \emptyset$ $\wedge \forall_{a \in \mathcal{A}} \forall_{b \in \mathcal{B}} a \rightarrow_L b \wedge \forall_{a_1, a_2 \in \mathcal{A}} a_1 \#_L a_2 \wedge \forall_{b_1, b_2 \in \mathcal{B}} b_1 \#_L b_2\}$
5. $\mathcal{Y}_{\mathcal{L}} = \{(\mathcal{A}, \mathcal{B}) \in \mathcal{X}_{\mathcal{L}} | \forall_{(\mathcal{A}', \mathcal{B}') \in \mathcal{X}_{\mathcal{L}}} \mathcal{A} \subseteq \mathcal{A}' \wedge \mathcal{B} \subseteq \mathcal{B}' \Longrightarrow (\mathcal{A}, \mathcal{B}) = (\mathcal{A}', \mathcal{B}')\}$
6. $\mathcal{P}_{\mathcal{L}} = \{p_{(\mathcal{A}, \mathcal{B})} | (\mathcal{A}, \mathcal{B}) \in \mathcal{Y}_{\mathcal{L}}\} \cup \{i_L, o_L\}$
7. $\mathcal{F}_{\mathcal{L}} = \{(a, p_{(\mathcal{A}, \mathcal{B})}) | (\mathcal{A}, \mathcal{B}) \in \mathcal{Y}_{\mathcal{L}} \wedge a \in \mathcal{A}\} \cup \{(p_{(\mathcal{A}, \mathcal{B})}, b) | (\mathcal{A}, \mathcal{B}) \in \mathcal{Y}_{\mathcal{L}} \wedge b \in \mathcal{B}\} \cup \{(i_L, t) | t \in \mathcal{T}_{\mathcal{I}}\} \cup \{(t, o_L) | t \in \mathcal{T}_{\mathcal{O}}\}$
8. $\alpha(\mathcal{L}) = (\mathcal{P}_{\mathcal{L}}, \mathcal{T}_{\mathcal{L}}, \mathcal{F}_{\mathcal{L}})$

Do not be scared! :)

α-algorithm

Let $\mathcal{L}$ be an event log over $\mathcal{T} \subseteq \mathcal{T}$, than $\alpha(\mathcal{L})$ is defined as follows:

1. $\mathcal{T}_{\mathcal{L}} = \{t \in \mathcal{T} | \exists_{\sigma \in \mathcal{L}} t \in \sigma\}$

$\mathcal{T}_{\mathcal{L}}$ is the set of activities do appear in the log, these will correspond to the transitions of the generated WF-Net

α-algorithm

Let $\mathcal{L}$ be an event log over $\mathcal{T} \subseteq \mathcal{T}$, than $\alpha(\mathcal{L})$ is defined as follows:

1. $\mathcal{T}_{\mathcal{L}} = \{t \in \mathcal{T} | \exists_{\sigma \in \mathcal{L}} t \in \sigma\}$

$\mathcal{T}_{\mathcal{L}}$ is the set of activities do appear in the log, these will correspond to the transitions of the generated WF-Net

2. $\mathcal{T}_{\mathcal{I}} = \{t \in \mathcal{T} | \exists_{\sigma \in \mathcal{L}} t = \mathit{first}(\sigma)\}$

$\mathcal{T}_{\mathcal{I}}$ is the set of start activities, i.e., all activities that appear first in some trace such as $\langle t_1, ..., t_n \rangle, ... \langle t'_1, ..., t'_m \rangle$

α-algorithm

Let $\mathcal{L}$ be an event log over $\mathcal{T} \subseteq \mathcal{T}$, than $\alpha(\mathcal{L})$ is defined as follows:

1. $\mathcal{T}_{\mathcal{L}} = \{t \in \mathcal{T} | \exists_{\sigma \in \mathcal{L}} t \in \sigma\}$

$\mathcal{T}_{\mathcal{L}}$ is the set of activities do appear in the log, these will correspond to the transitions of the generated WF-Net

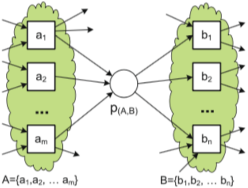
2. $\mathcal{T}_{\mathcal{I}} = \{t \in \mathcal{T} | \exists_{\sigma \in \mathcal{L}} t = \mathit{first}(\sigma)\}$

$\mathcal{T}_{\mathcal{I}}$ is the set of start activities, i.e., all activities that appear first in some trace such as $\langle t_1, ..., t_n \rangle, ... \langle t'_1, ..., t'_m \rangle$

3. $\mathcal{T}_{\mathcal{O}} = \{t \in \mathcal{T} | \exists_{\sigma \in \mathcal{L}} t = \mathit{last}(\sigma)\}$

$\mathcal{T}_{\mathcal{O}}$ is the set of end activities, i.e., all activities that appear last in some trace, such as $\langle t_1, ... t_n \rangle, ... \langle t'_1, ..., t'_m \rangle$

Place p(A,B) connects the transitions in set A to the transitions in set B



4. Calculate pairs (A, B)

$$\begin{aligned}\mathcal{X}_{\mathcal{L}} = \{(\mathcal{A},\mathcal{B}) \mid & \mathcal{A} \subseteq \mathcal{T}_{\mathcal{L}} \wedge \mathcal{A} \neq \emptyset \wedge \mathcal{B} \subseteq \mathcal{T}_{\mathcal{L}} \wedge \mathcal{B} \neq \emptyset \\ & \wedge \forall_{a \in \mathcal{A}} \forall_{b \in \mathcal{B}}\, a \rightarrow_L b \\ & \wedge \forall_{a_1, a_2 \in \mathcal{A}}\, a_1 \#_L a_2 \\ & \wedge \forall_{b_1, b_2 \in \mathcal{B}}\, b_1 \#_L b_2\}\end{aligned}$$

5. Delete non maximal pairs (A, B)

$$\mathcal{Y}_{\mathcal{L}} = \{(\mathcal{A},\mathcal{B}) \in \mathcal{X}_{\mathcal{L}} \mid \forall_{(\mathcal{A}',\mathcal{B}') \in \mathcal{X}_{\mathcal{L}}} \mathcal{A} \subseteq \mathcal{A}' \wedge \mathcal{B} \subseteq \mathcal{B}' \implies (\mathcal{A},\mathcal{B}) = (\mathcal{A}',\mathcal{B}')\}$$

6. Determine place $p_{(\mathcal{A},\mathcal{B})}$ from pairs (A, B)

$$\mathcal{P}_{\mathcal{L}} = \{p_{(\mathcal{A},\mathcal{B})} \mid (\mathcal{A},\mathcal{B}) \in \mathcal{Y}_{\mathcal{L}}\} \cup \{i_L, o_L\}$$

4. Calculate pairs (A, B)

4. Calculate pairs (A, B)

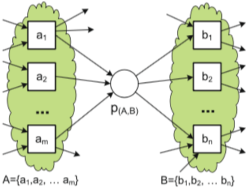
$$
\begin{aligned}
\mathcal{X}_{\mathcal{L}} = \{(\mathcal{A},\mathcal{B}) \mid \mathcal{A} \subseteq \mathcal{T}_{\mathcal{L}} \wedge \mathcal{A} \neq \varnothing \wedge \mathcal{B} \subseteq \mathcal{T}_{\mathcal{L}} \wedge \mathcal{B} \neq \varnothing \\
\wedge \forall_{a \in \mathcal{A}} \forall_{b \in \mathcal{B}}\, a \rightarrow_L b \\
\wedge \forall_{a_1, a_2 \in \mathcal{A}}\, a_1 \#_L a_2 \\
\wedge \forall_{b_1, b_2 \in \mathcal{B}}\, b_1 \#_L b_2 \}
\end{aligned}
$$

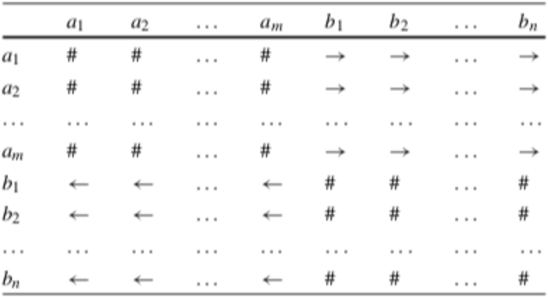
We have to find two sets of activities, A and B, and these activities should have the following properties.

- If we take any activity in the set A and we take any activity in the set B, there should always be a direct succession between these two activities. So there should be at least one position in the log where the element of A is followed by the element of B and that should hold for all combinations.
- If I take two activities in the set A, they should never follow one another. If I take two activities in the set B, they should also never follow one another. Even if we take the same activity, it should never follow itself.

How to identify $(\mathcal{A}, \mathcal{B}) \in \mathcal{X}_{\mathcal{L}}$?

Loking at the footprint matrix we can recognize this structure because we are looking for a set a and b where things never follow one another. And we are looking for these other connections where any element of a is directly followed by any element of b, but never the other way around.





	a_1	a_2	...	a_m	b_1	b_2	...	b_n
a_1	#	#	...	#	→	→	...	→
a_2	#	#	...	#	→	→	...	→
...	...	...	...	...	...	...	...	...
a_m	#	#	...	#	→	→	...	→
b_1	←	←	...	←	#	#	...	#
b_2	←	←	...	←	#	#	...	#
...	...	...	...	...	...	...	...	...
b_n	←	←	...	←	#	#	...	#

5. Delete non maximal pairs (A, B)

5. Delete non maximal pairs (A, B)

$\mathcal{Y}_{\mathcal{L}} = \{(\mathcal{A}, \mathcal{B}) \in \mathcal{X}_{\mathcal{L}} | \forall_{(\mathcal{A}', \mathcal{B}') \in \mathcal{X}_{\mathcal{L}}} \mathcal{A} \subseteq \mathcal{A}' \wedge \mathcal{B} \subseteq \mathcal{B}' \implies (\mathcal{A}, \mathcal{B}) = (\mathcal{A}', \mathcal{B}')\}$

Delete the element that are contained in others

6. Determine place $p_{(\mathcal{A},\mathcal{B})}$ from pairs (A, B)

6. Determine place $p_{(\mathcal{A},\mathcal{B})}$ from pairs (A, B)
$\mathcal{P}_{\mathcal{L}} = \{p_{(\mathcal{A},\mathcal{B})} | (\mathcal{A},\mathcal{B}) \in \mathcal{Y}_{\mathcal{L}}\} \cup \{i_L, o_L\}$

All the maximal pairs that we have just discovered in step 5. are places and we add an initial place i_L and a final place o_L

Final Steps

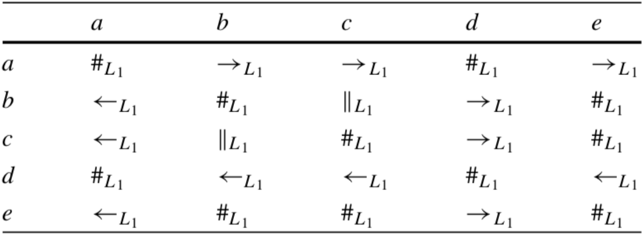
7. $$\begin{aligned}\mathcal{F}_{\mathcal{L}} = &\{(a, p_{(\mathcal{A},\mathcal{B})}) | (\mathcal{A},\mathcal{B}) \in \mathcal{Y}_{\mathcal{L}} \wedge a \in \mathcal{A}\} \cup \\ &\{(p_{(\mathcal{A},\mathcal{B})}, b) | (\mathcal{A},\mathcal{B}) \in \mathcal{Y}_{\mathcal{L}} \wedge b \in \mathcal{B}\} \cup \\ &\{(i_L, t) | t \in \mathcal{T}_{\mathcal{I}}\} \cup \\ &\{(t, o_L) | t \in \mathcal{T}_{\mathcal{O}}\}\end{aligned}$$

We already have the transitions and the places. Here you see the arcs. So here, you can see all connections from the initial place, I, to all the initial transitions in $\mathcal{T}_{\mathcal{I}}$. From all the transitions in the set $\mathcal{T}_{\mathcal{O}}$. So the transitions corresponding to the activities that happen at the end. And all internal places, and internal places are represented by sets $\mathcal{A}$ and $\mathcal{B}$ and the connections are made accordingly.

8. $\alpha(\mathcal{L}) = (\mathcal{P}_{\mathcal{L}}, \mathcal{T}_{\mathcal{L}}, \mathcal{F}_{\mathcal{L}})$

α-algorithm application considering $\mathcal{L}1$

$$\mathcal{L}1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$



	a	b	c	d	e
a	$\#_{L_1}$	$\rightarrow_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$
b	$\leftarrow_{L_1}$	$\#_{L_1}$	$\|_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$
c	$\leftarrow_{L_1}$	$\|_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$
d	$\#_{L_1}$	$\leftarrow_{L_1}$	$\leftarrow_{L_1}$	$\#_{L_1}$	$\leftarrow_{L_1}$
e	$\leftarrow_{L_1}$	$\#_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$

$\mathcal{T}_{\mathcal{L}} =$

$\mathcal{T}_{\mathcal{I}} =$

$\mathcal{T}_{\mathcal{O}} =$

$\mathcal{X}_{\mathcal{L}} =$

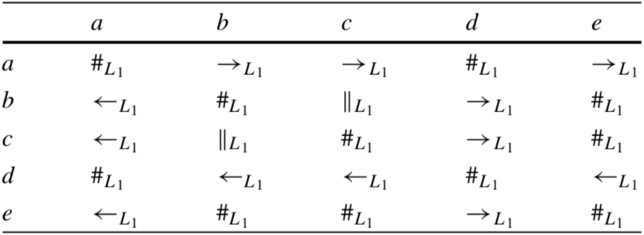
$\mathcal{Y}_{\mathcal{L}} =$

$\mathcal{P}_{\mathcal{L}} =$

$\mathcal{F}_{\mathcal{L}} =$

α-algorithm application considering $\mathcal{L}1$

$$\mathcal{L}1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$



	a	b	c	d	e
a	$\#_{L_1}$	$\rightarrow_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$
b	$\leftarrow_{L_1}$	$\#_{L_1}$	$\|_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$
c	$\leftarrow_{L_1}$	$\|_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$
d	$\#_{L_1}$	$\leftarrow_{L_1}$	$\leftarrow_{L_1}$	$\#_{L_1}$	$\leftarrow_{L_1}$
e	$\leftarrow_{L_1}$	$\#_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$

$$\begin{aligned}
\mathcal{T}_{\mathcal{L}} &= \{a, b, c, d, e\} \\
\mathcal{T}_{\mathcal{I}} &= \{a\} \\
\mathcal{T}_{\mathcal{O}} &= \{d\} \\
\mathcal{X}_{\mathcal{L}} &= \{(\{a\}, \{b\}), (\{a\}, \{c\}), (\{a\}, \{e\}), (\{a\}, \{b, e\}), (\{a\}, \{c, e\}), \\
&\quad\ (\{b\}, \{d\}), (\{c\}, \{d\}), (\{e\}, \{d\}), (\{b, e\}, \{d\}), (\{c, e\}, \{d\})\} \\
\mathcal{Y}_{\mathcal{L}} &= \{(\{a\}, \{b, e\}), (\{a\}, \{c, e\})(\{b, e\}, \{d\}), (\{c, e\}, \{d\})\} \\
\mathcal{P}_{\mathcal{L}} &= \{p_{(\{a\},\{b,e\})}, p_{(\{a\},\{c,e\})}, p_{(\{b,e\},\{d\})}, p_{(\{c,e\},\{d\})}, i_L, o_L\} \\
\mathcal{F}_{\mathcal{L}} &= \{(i_L, a), (a, p_{(\{a\},\{b,e\})}), (p_{(\{a\},\{b,e\})}, b), (p_{(\{a\},\{b,e\})}, e), \ldots, (d, o_L)\}
\end{aligned}$$

α-algorithm application considering $\mathcal{L}2$

$$\mathcal{L}2 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^4, \langle a, b, c, e, f, b, c, d\rangle^2,$$
$$\langle a, b, c, e, f, c, b, d\rangle^2, \langle a, c, b, e, f, b, c, d\rangle^2, \langle a, c, b, e, f, b, c, e, f, c, b, d\rangle]$$

$\mathcal{T}_{\mathcal{L}} =$
$\mathcal{T}_{\mathcal{I}} =$
$\mathcal{T}_{\mathcal{O}} =$
$\mathcal{X}_{\mathcal{L}} =$
$\mathcal{Y}_{\mathcal{L}} =$
$\mathcal{P}_{\mathcal{L}} =$
$\mathcal{F}_{\mathcal{L}} =$

α-algorithm application considering $\mathcal{L}3$

$\mathcal{L}3 =$
$[\langle a, b, c, d, e, f, b, d, c, e, g\rangle, \langle a, b, d, c, e, g\rangle^2, \langle a, b, c, d, e, f, b, c, d, e, f, b, d, c, e, g\rangle]$

$\mathcal{T}_{\mathcal{L}} =$
$\mathcal{T}_{\mathcal{I}} =$
$\mathcal{T}_{\mathcal{O}} =$
$\mathcal{X}_{\mathcal{L}} =$
$\mathcal{Y}_{\mathcal{L}} =$
$\mathcal{P}_{\mathcal{L}} =$
$\mathcal{F}_{\mathcal{L}} =$

α-algorithm application considering $\mathcal{L}4$

$$\mathcal{L}4 = [\langle a, c, d\rangle^{45}, \langle b, c, d\rangle^{42}, \langle a, c, e\rangle^{38}, \langle b, c, e\rangle^{22}]$$

$$\begin{aligned}
\mathcal{T}_{\mathcal{L}} &= \\
\mathcal{T}_{\mathcal{I}} &= \\
\mathcal{T}_{\mathcal{O}} &= \\
\mathcal{X}_{\mathcal{L}} &= \\
\mathcal{Y}_{\mathcal{L}} &= \\
\mathcal{P}_{\mathcal{L}} &= \\
\mathcal{F}_{\mathcal{L}} &=
\end{aligned}$$

α-algorithm application considering $\mathcal{L}5$

$\mathcal{L}5 =$
$[\langle a, b, e, f\rangle^2, \langle a, b, e, c, d, b, f\rangle^3, \langle a, b, c, e, d, b, f\rangle^2, \langle a, b, e, f\rangle^2, \langle a, b, e, c, d, b, f\rangle^3,$
$\langle a, b, c, e, d, b, f\rangle^2, \langle a, b, c, d, e, b, f\rangle^4, \langle a, e, b, c, d, b, f\rangle^3]$

$\mathcal{T}_{\mathcal{L}} =$
$\mathcal{T}_{\mathcal{I}} =$
$\mathcal{T}_{\mathcal{O}} =$
$\mathcal{X}_{\mathcal{L}} =$
$\mathcal{Y}_{\mathcal{L}} =$
$\mathcal{P}_{\mathcal{L}} =$
$\mathcal{F}_{\mathcal{L}} =$

Limitation of α-algorithm

In what situations doesn't the α-algorithm produce the result that you expect?

The α-algorithm can discover a large class of WF-nets assuming that **the log is complete** with respect to the log-based ordering relation $>_L$

This assumption implies that, for any complete event log L, $a >_L b$ if a can be directly followed by b

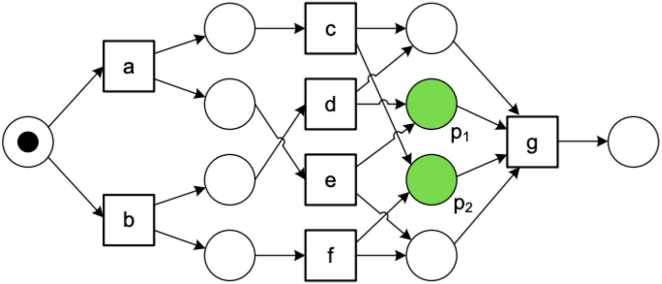
Even if we assume that the log is complete, the α-algorithm has some problems

There are many different WF-nets that have the same possible behavior, i.e., two models can be structurally different but trace equivalent

Let's take a look at some logs that show limitations of the α-algorithm.

Limitation of α-algorithm (implicit places)

$\mathcal{L}6 = [\langle a, c, e, g\rangle^2, \langle a, e, c, g\rangle^3, \langle b, d, f, g\rangle^2, \langle b, f, d, g\rangle^4]$

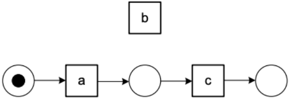


The places denoted as p_1 and p_2 are so-called implicit places and can be removed without problem, they only complicate matters and don't add anything

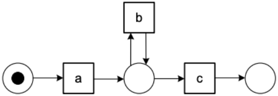
Limitation of α-algorithm (loops of length 1)

$$\mathcal{L}7 = [\langle a, c\rangle^2, \langle a, b, c\rangle^3, \langle a, b, b, c\rangle^2, \langle a, b, b, b, b, c\rangle^1]$$

$a > b, a > c, b > b, b > c$
$a \rightarrow b, a \rightarrow c, b \rightarrow c$
$b \| b$
$a \# a, a \# c$



Discovered Model



Desidered model

The resulting model is not a WF-net as transition b is disconnected from the rest of the model. The models allows for the execution of b before a and after c. This is not consistent with the event log.

This problem can be addressed using an improved version of α-algorithm.

Limitation of α-algorithm (loops of length 2)

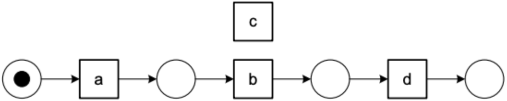
$$\mathcal{L}8 = [\langle a, b, d\rangle^3, \langle a, b, c, b, d\rangle^2, \langle a, b, c, b, c, b, d\rangle]$$

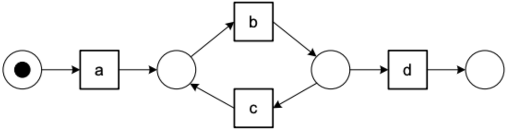
$$a > b, b > c, b > d, c > b$$

$$a \rightarrow b, b \rightarrow d$$

$$b \| c$$

Discovered Model





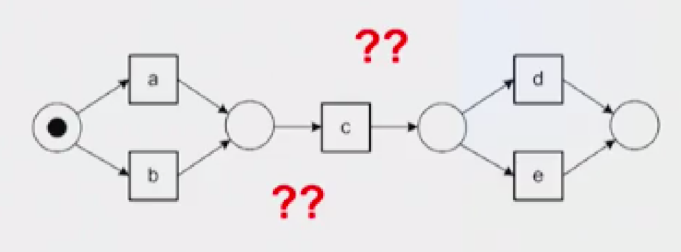
Desidered model

The basic algorithm has no problems mining loops of length three or more!!!

Limitation of α-algorithm (non-local dependencies)

$\mathcal{L}9 = [\langle a, c, d\rangle^{45}, \langle b, c, e\rangle^{42}]$

Discovered Model



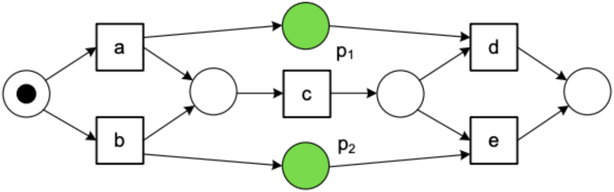
The two traces that we see in the log are indeed possible

But we also allow for a trace where we first do b, then c, and then d - Which was not observed in the log!!!!

Limitation of α-algorithm (non-local dependencies)

$\mathcal{L}9 = [\langle a, c, d\rangle^{45}, \langle b, c, e\rangle^{42}]$

What we would like to discover is this process model



But p1 and p2 are not discovered because a and d and b and e never follow one another directly, only indirectly.

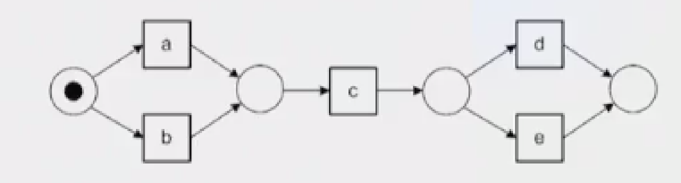
Such problems can be (partially) resolved using refined versions of the α-algorithm

Limitation of α-algorithm (non-local dependencies)

$\mathcal{L}9 = [\langle a, c, d\rangle^{45}, \langle b, c, e\rangle^{42}]$
$\mathcal{L}4 = [\langle a, c, d\rangle^{45}, \langle b, c, d\rangle^{42}, \langle a, c, e\rangle^{38}, \langle b, c, e\rangle^{22}]$

The problem that we see here is that we have these two event logs are very different but if we look at the corresponding footprints they are the same

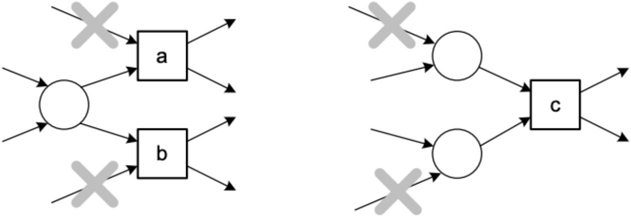
In both cases we produce this model



The resulting model is under fitting, if we look at the first log $\mathcal{L}9$

Difficult constructs for α-algorithm

The non-local dependencies, correspond to so called non-free choice constructs, situations where there is a mixture of choice and synchronization



If we have process models where these things happen, the α-algorithm is likely to produce an incorrect result.

Challenges

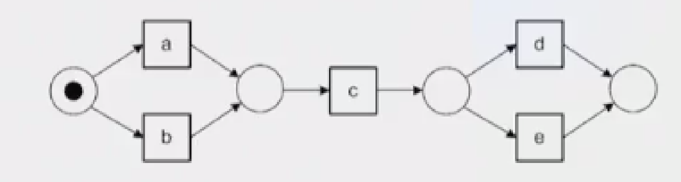
Question

$$\mathcal{L} = [\langle a, c, d \rangle^{45}, \langle b, c, e \rangle^{42}, \langle a, c, e \rangle^{20}]$$

- What model will the α-algorithm generate?
- What is the model that would actually generate the behavior that you see in the log and nothing more?

Answer 1 - Model Generated by α-algorithm

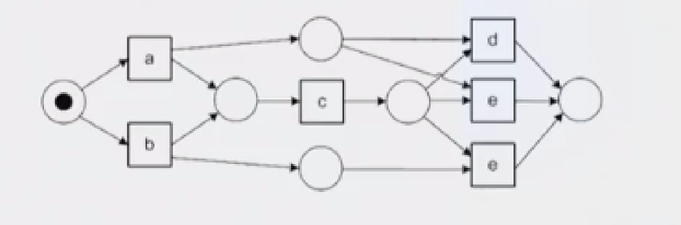
$$\mathcal{L} = [\langle a, c, d\rangle^{45}, \langle b, c, e\rangle^{42}, \langle a, c, e\rangle^{20}]$$



We see the same problem as before that we don't see these non-local dependencies. So, this allows for a trace that was never observed.

Answer 2 - Model that can produce the observed behaviour and nothing more

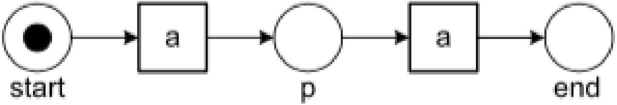
$$\mathcal{L} = [\langle a, c, d\rangle^{45}, \langle b, c, e\rangle^{42}, \langle a, c, e\rangle^{20}]$$



In the model only the three different types of traces that we see in the log can be generated by this model. But in the model there are multiple transitions having the same label. Using the α-algorithm, you could never discover this model.

Limitation: representational bias

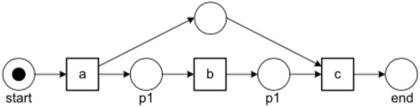
$\mathcal{L}10 = [\langle a, a\rangle^{55}]$



We can never discover it because our representation doesn't allow for the discovery of a model with multiple transitions having the same label

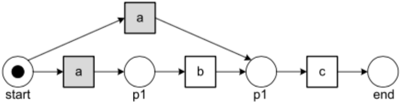
Another Example

$$\mathcal{L}11 = [\langle a, b, c\rangle^{20}, \langle a, c\rangle^{30}]$$

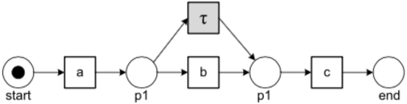


it is incorrect because in this model we cannot skip b

Duplicated Transitions

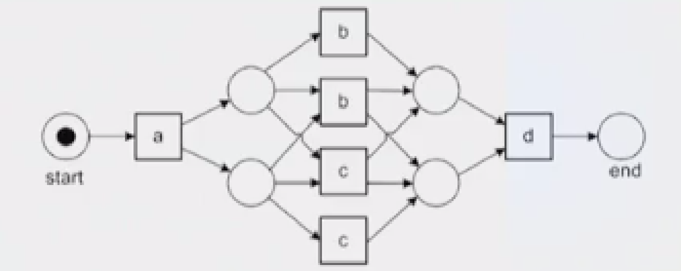


Silent Step via tau transition



Both are possible, but they are not within the representational bias of the α-algorithm

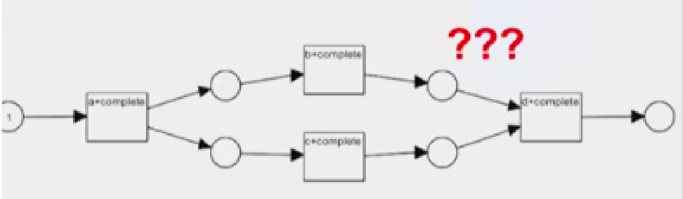
OR join/split pattern



- Let us take an event log containing all possible full firing sequence and apply the α-algorithm
- What will happen?

OR join/split pattern

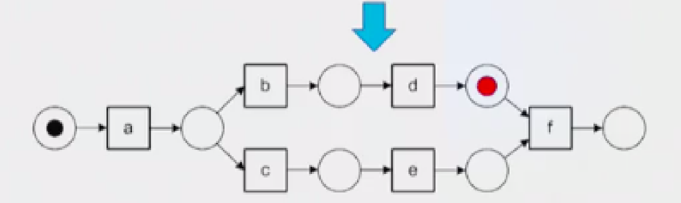
$$\mathcal{L} = [\langle a, b, d\rangle, \langle a, c, d\rangle, \langle a, b, c, d\rangle, \langle a, c, b, d\rangle]$$



It is incorrect because b and c are always executed, rather than that they are optional

Limitation: resulting model does not need to be a sound WF-net

$$\mathcal{L}11 = [\langle a, b, d, e, f\rangle^{10}, \langle a, c, e, d, f\rangle^{10}]$$



The dicovered model is not sound!

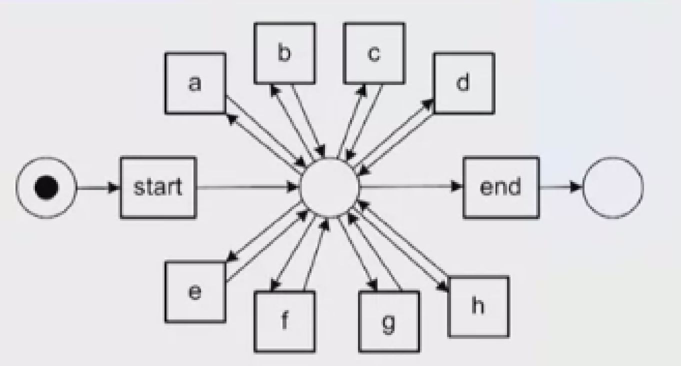
This is an assumption in the application of α-algorithm

Challenge: Noice and Incompleteness

To discover a suitable process model it is assumed that the event log contains a representative sample of behavior

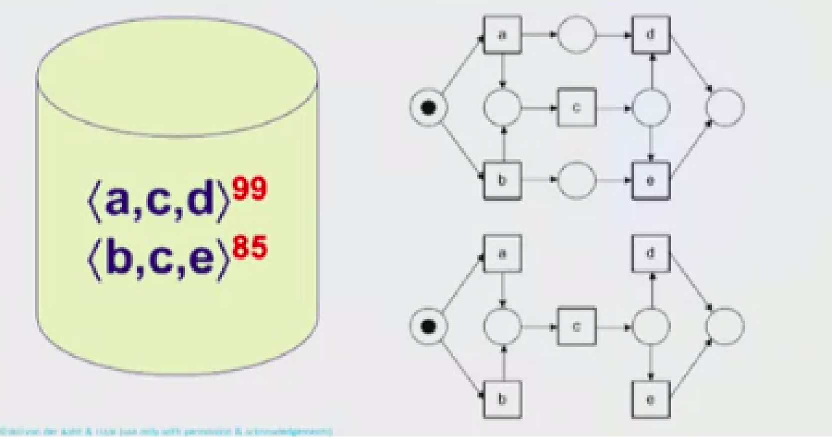
- **Noise**: the event log contains rare and infrequent behaviour not representative for the typical behaviour of the process
- **Incompleteness**: the event log contain too few events to be able to discover some of the underlying control-flow structures

Flower Model



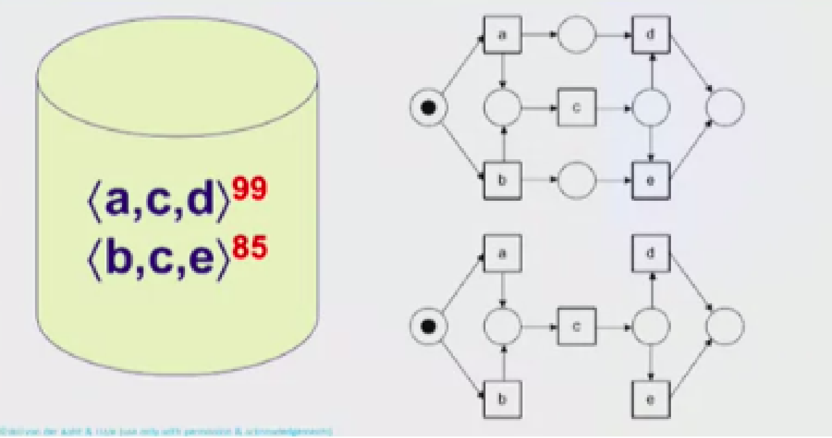
It allow for any behaviour, this is underfitting!

What is the best model?



©Wil van der Aalst & TU/e (use only with permission & acknowledgements)

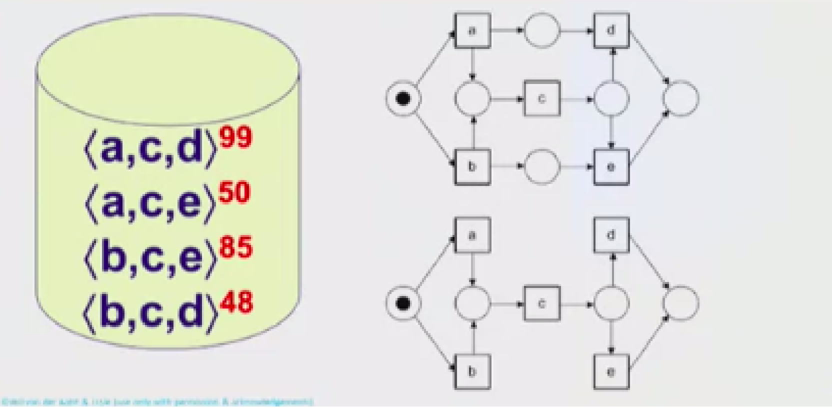
What is the best model?



©Wil van der Aalst & TU/e (use only with permission & acknowledgements)

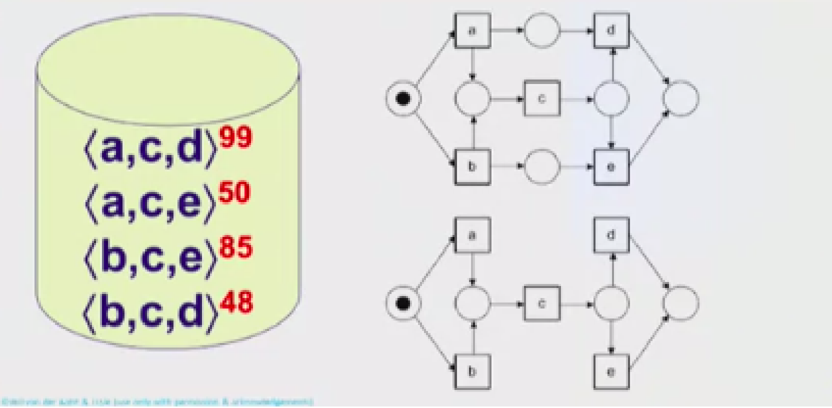
The first model!

What is the best model?



©Wil van der Aalst & TU/e (use only with permission & acknowledgements)

What is the best model?

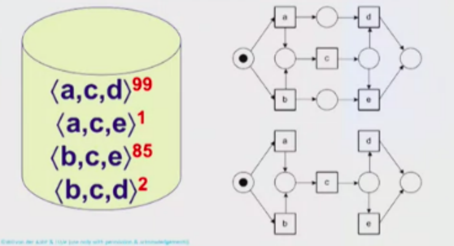


©Wil van der Aalst & TU/e (use only with permission & acknowledgements)

The second model!

What is the best model?

A more mixed situation, where the traces (a,c,e) and (b,c,d) are infrequent

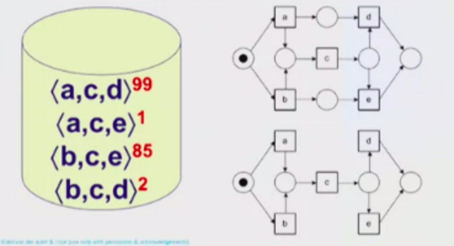


$\langle a,c,d\rangle^{99}$
$\langle a,c,e\rangle^{1}$
$\langle b,c,e\rangle^{85}$
$\langle b,c,d\rangle^{2}$
a
d
c
b
e
a
d
c
b
e

©Wil van der Aalst & TU/e (use only with permission & acknowledgements)

What is the best model?

A more mixed situation, where the traces (a,c,e) and (b,c,d) are infrequent



It is unclear which of the two models we prefer

- The top model describes the dominant behavior, but it does not allow for some of the infrequent traces that we have seen in the log
- The bottom model capture all behavior, but we would not be able to distinguish between the highway and the traces that are less frequent

Noice and incompleteness

The α-algorithm cannot deal with noise and incompleteness.

Process models are like maps:
we may not want to see all paths
and only see the highways

This is also a challenge for many of the other algorithms.

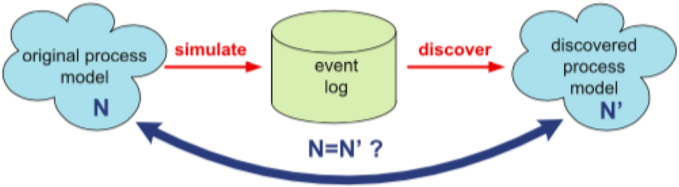
Limitation summing up

- Implicit places (places that are redundant): harmless and be solved through preprocessing
- Loops of length 1: can be solved in multiple ways (change of algorithm or pre/post-processing)
- Loops of length 2: idem
- Non local dependencies: foundational problem, not specific for α-algorithm
- Representation bias (cannot discover transition with duplicate or invisible labels): other algorithms may have a different bias
- Discovered model does not need to be sound: some algorithm ensure this
- Noise: foundational problem, not specific for α-algorithm
- Completeness: also foundational problem

Rediscovering Process Models

How to measure the quality of a discovered model?

For the moment, we only mention the rediscovering process model

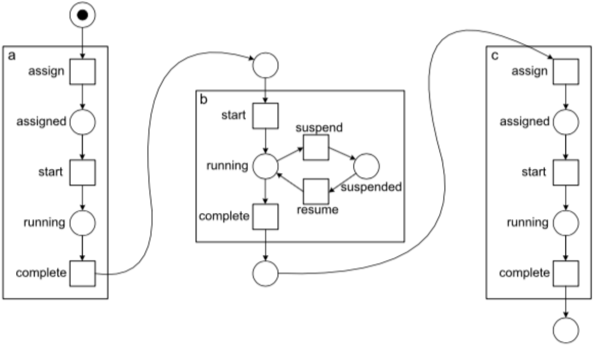


The rediscovery problem: is the discovered model N' equivalent to the original model N?

Taking the Transactional Life-Cycle into Account

Taking the Transactional Life-Cycle into Account

Mining event logs with transactional information; the life-cycle of each activity is represented as a subprocess



The α-algorithm can be easily adapted to take this information into account.