Process Mining

Lesson 7 – Alpha Algoritm

Doc. Cognini Riccardo

The 3 main type of Mining

- **Discovery (here we are!)**
- Conformance
- Enhancement

Discovery and Algorithms

Definition 5.1 (General process discovery problem) Let L be an event log as defined in Definition 4.3 or as specified by the XES standard (cf. Sect. 4.3). A *process discovery algorithm* is a function that maps 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.

Definition 5.2 (Specific process discovery problem) A *process discovery algorithm* is a function γ that maps a log $L \in \mathbb{B}(\mathscr{A}^*)$ onto a marked Petri net $\gamma(L) = (N, M)$. Ideally, N is a *sound WF-net* and all traces in L correspond to possible firing sequences of (N, M).

Discovered Models

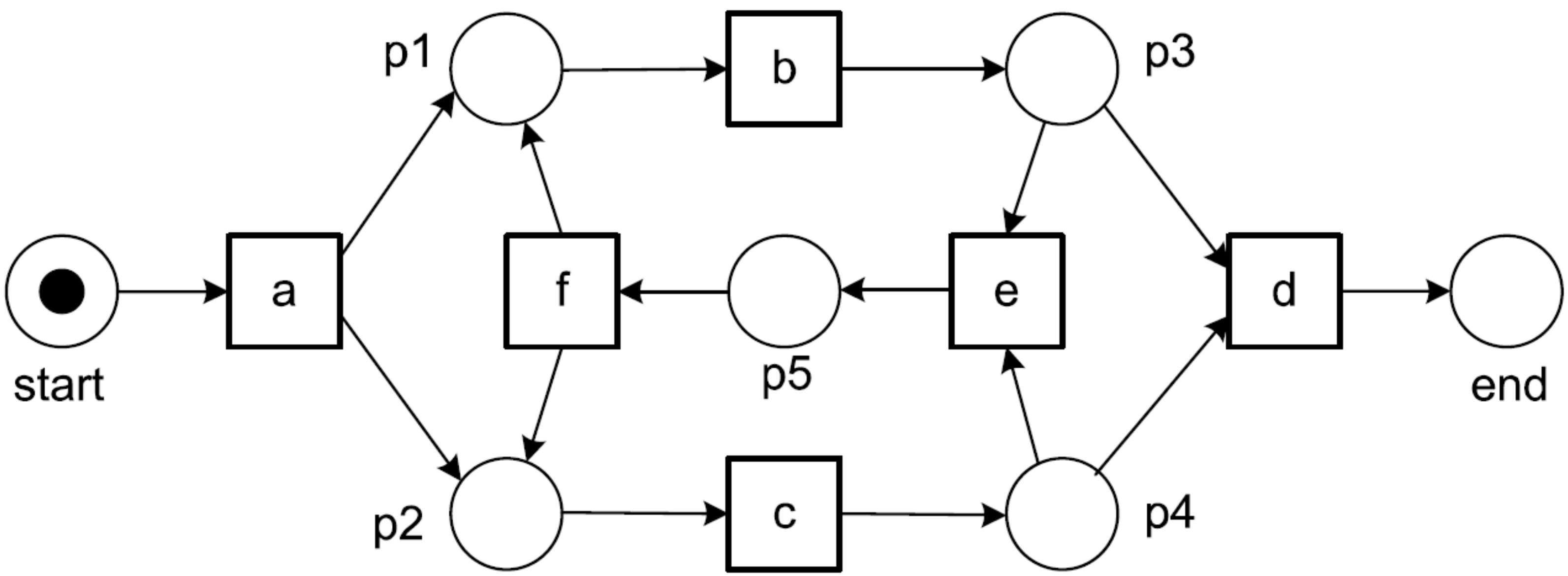
Quality Criteria of Discovered models:

- *Fitness*: the discovered model should allow for the behavior seen in the event log
- *Precision*: the discovered model should not allow for behavior completely unrelated 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

WorkFlow Nets

WorkFlow-Nets are used such as discovered models
They are a subclass Petri-NET in which:

- There is a START place with just 1 Token inside
- There is a END place



The Alpha-Algoritm

Ordering Relation considered by the Algoritm

Definition 5.3 (Log-based ordering relations) Let L be an event log over $\mathscr{A}$, i.e., $L \in \mathbb{B}(\mathscr{A}^*)$. Let $a, b \in \mathscr{A}$:

- $a >_L b$ if and only if there is a trace $\sigma = \langle t_1, t_2, t_3, \ldots, t_n \rangle$ and $i \in \{1, \ldots, n-1\}$ such that $\sigma \in L$ and $t_i = a$ and $t_{i+1} = b$
- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \|_L b$ if and only if $a >_L b$ and $b >_L a$

Consider for instance $L_1 = [\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$ again. For this event log, the following log-based ordering relations can be found

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

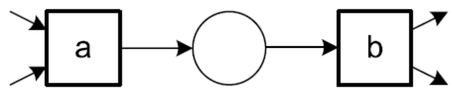
$$\rightarrow_{L_1} = \{(a, b), (a, c), (a, e), (b, d), (c, d), (e, d)\}$$

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

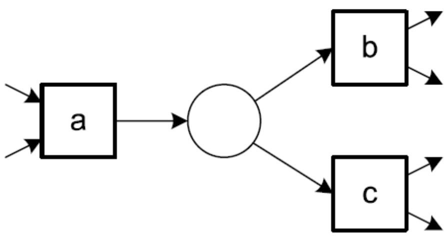
$$\|_{L_1} = \{(b, c), (c, b)\}$$

The Alpha-Algoritm

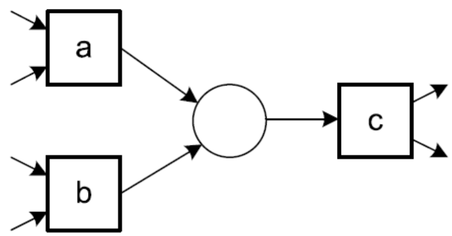
Patterns considered by the algortithm



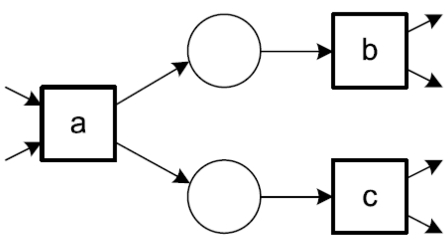
(a) sequence pattern: a→b



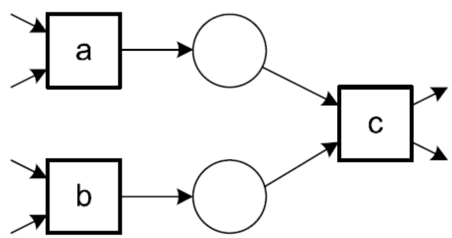
(b) XOR-split pattern:
a→b, a→c, and b#c



(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

The Alpha-Algoritm

The Sets used by the algorithm:

Definition 5.4 (α-algorithm) Let L be an event log over $T \subseteq \mathscr{A}$. $\alpha(L)$ is defined as follows.

(1) $T_L = \{t \in T \mid \exists_{\sigma \in L}\ t \in \sigma\}$
(2) $T_I = \{t \in T \mid \exists_{\sigma \in L}\ t = \mathit{first}(\sigma)\}$
(3) $T_O = \{t \in T \mid \exists_{\sigma \in L}\ t = \mathit{last}(\sigma)\}$
(4) $X_L = \{(A, B) \mid A \subseteq T_L \ \wedge \ A \neq \emptyset \ \wedge \ B \subseteq T_L \ \wedge \ B \neq \emptyset \ \wedge \ \forall_{a \in A} \forall_{b \in B}\ a \rightarrow_L b \ \wedge \ \forall_{a_1, a_2 \in A}\ a_1 \#_L a_2 \ \wedge \ \forall_{b_1, b_2 \in B}\ b_1 \#_L b_2\}$
(5) $Y_L = \{(A, B) \in X_L \mid \forall_{(A', B') \in X_L} A \subseteq A' \wedge B \subseteq B' \Longrightarrow (A, B) = (A', B')\}$
(6) $P_L = \{p_{(A,B)} \mid (A, B) \in Y_L\} \cup \{i_L, o_L\}$
(7) $F_L = \{(a, p_{(A,B)}) \mid (A, B) \in Y_L \ \wedge \ a \in A\} \cup \{(p_{(A,B)}, b) \mid (A, B) \in Y_L \ \wedge \ b \in B\} \cup \{(i_L, t) \mid t \in T_I\} \cup \{(t, o_L) \mid t \in T_O\}$
(8) $\alpha(L) = (P_L, T_L, F_L)$

Some Examples!

Alpha Algorithm Limitations

Short Loops (i.e. loops of one or two activities)

Fig. 5.10 Incorrect WF-net N_7 derived from $L_7 = [\langle a, c\rangle^2, \langle a, b, c\rangle^3, \langle a, b, b, c\rangle^2, \langle a, b, b, b, b, c\rangle^1]$

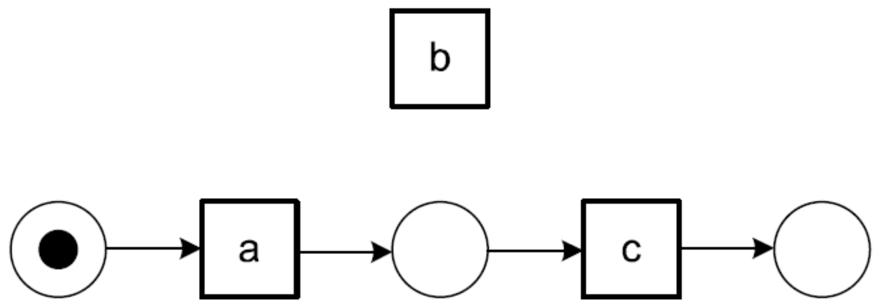
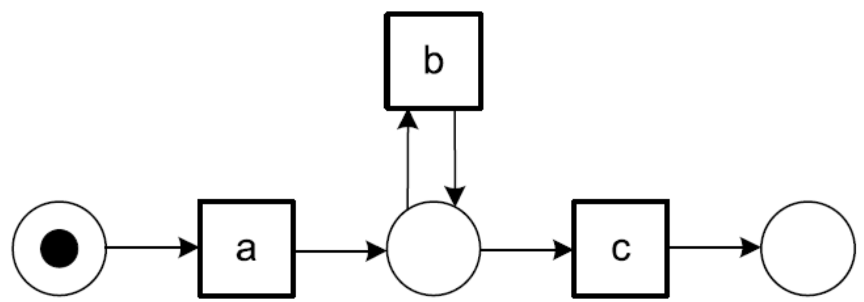


Fig. 5.11 WF-net N_7' having a so-called "short-loop" of length one



Alpha Algorithm Limitations

Short Loops (i.e. loops of one or two activities)

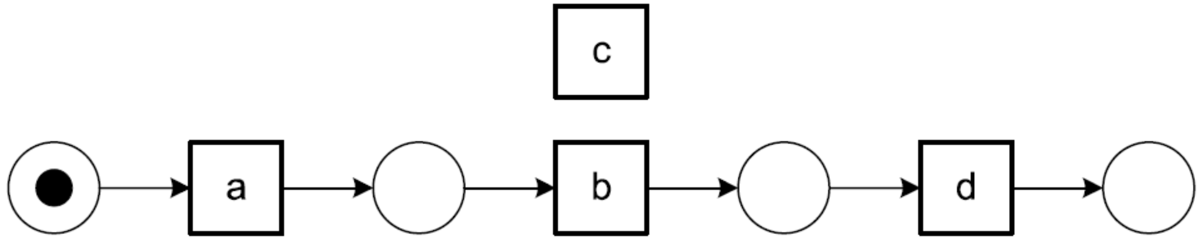


Fig. 5.12 Incorrect WF-net N_8 derived from $L_8 = [\langle a,b,d\rangle^3, \langle a,b,c,b,d\rangle^2, \langle a,b,c,b,c,b,d\rangle]$

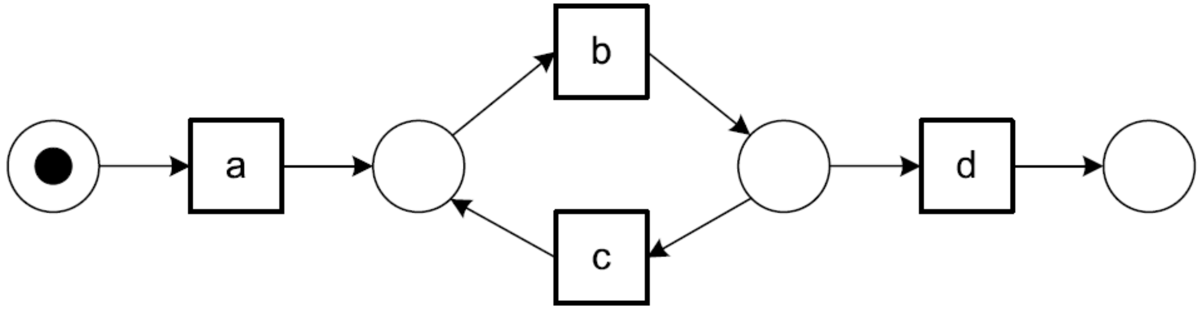


Fig. 5.13 Corrected WF-net N_8' having a so-called "short-loop" of length two

Alpha Algorithm Limitations

Non-Local Dependences

$$L_9 = [\langle a,c,d\rangle^{45}, \langle b,c,e\rangle^{42}]$$

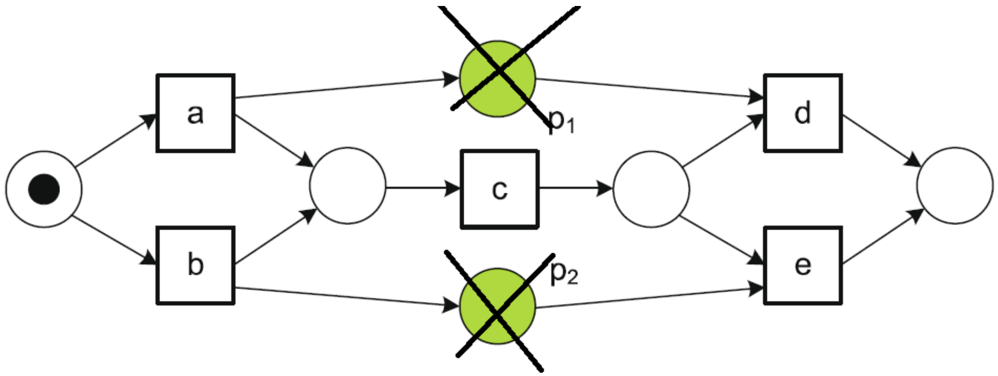
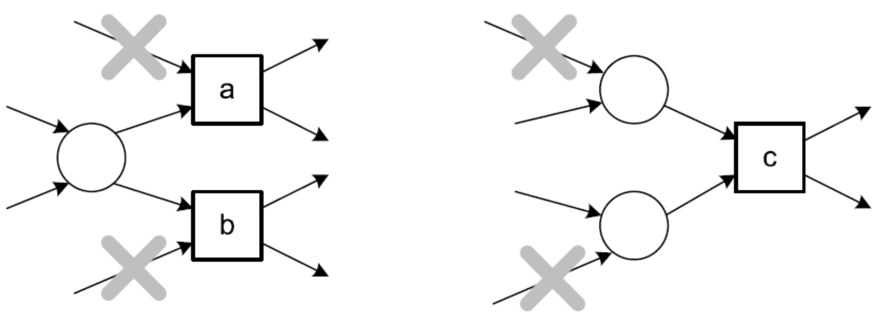


Fig. 5.14 WF-net N_9 having a non-local dependency

Fig. 5.15 Two constructs that may jeopardize the correctness of the discovered WF-net



QUESTIONS?