Christian-Albrechts-Universität zu Kiel

Advanced Process Mining

Prof. Dr. Agnes Koschmider

Lecture 5: Conformance Checking II

Process Analytics

Lecture Overview

Recall: Alignments

Trace: ADAB

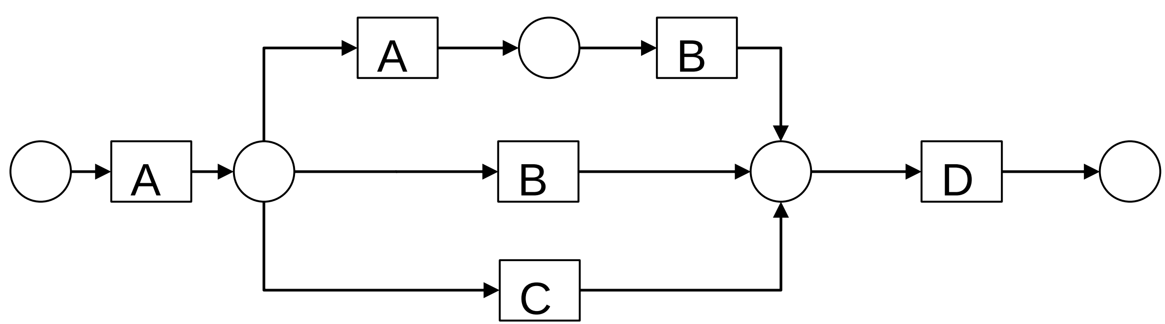
move in model: an event that should have been observed according to modeled behavior but missed in the trace

Alignment:

A	D	A	B	
A		A	B	D

synchronous move in the ideal alignment (“good move”)

move in log:
observed event not allowed by the modeled



Recall: Optimal Alignments

CAU
Christian-Albrechts-Universität zu Kiel

Trace: AD

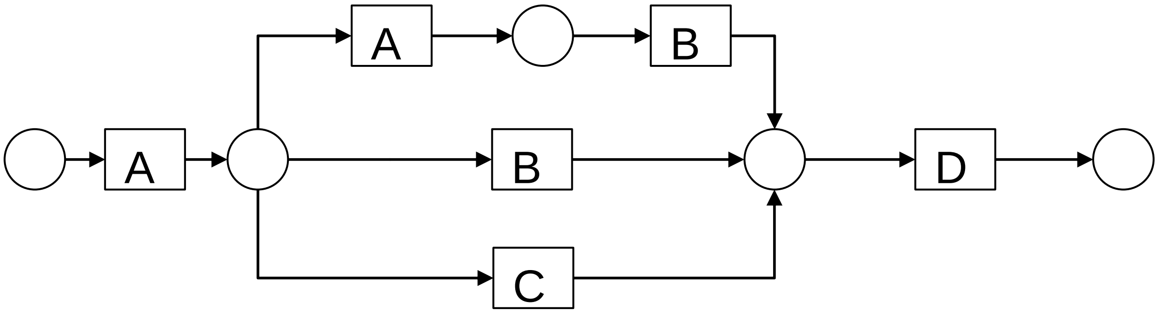
Alignment 1:

A		D
A	B	D

Alignment 2:

A		D
A	C	D

both alignments are optimal



Finding an optimal alignment

A	B	C	D	>>	E	>>	G
A	B	>>	D	C	>>	F	G

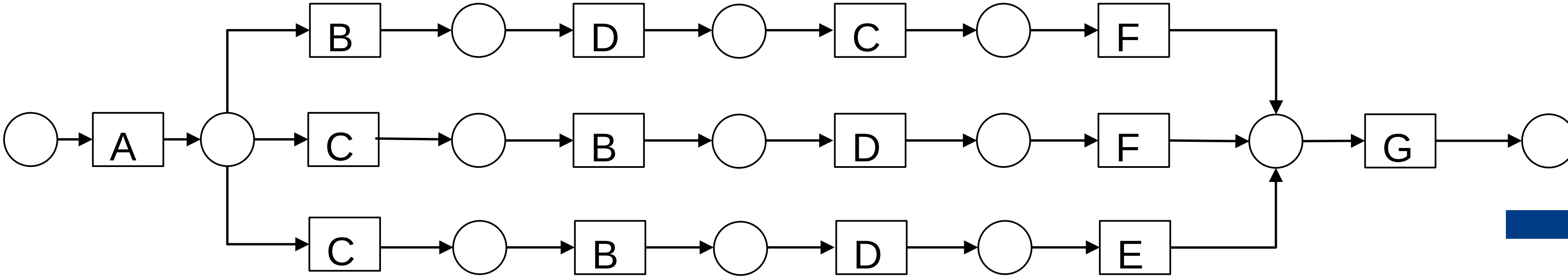
Cost = 4

A	>>	B	C	D	E	>>	G
A	C	B	>>	D	>>	F	G

Cost = 4

A	>>	B	C	D	E	G
A	C	B	>>	D	E	G

Cost = 2



Finding an optimal alignment (cont.)

A	B	C	D	>>	E	>>	G
A	B	>>	D	C	>>	F	G

$\text{Cost} = 4 = 1 - \frac{4}{6+6} = 0.67$

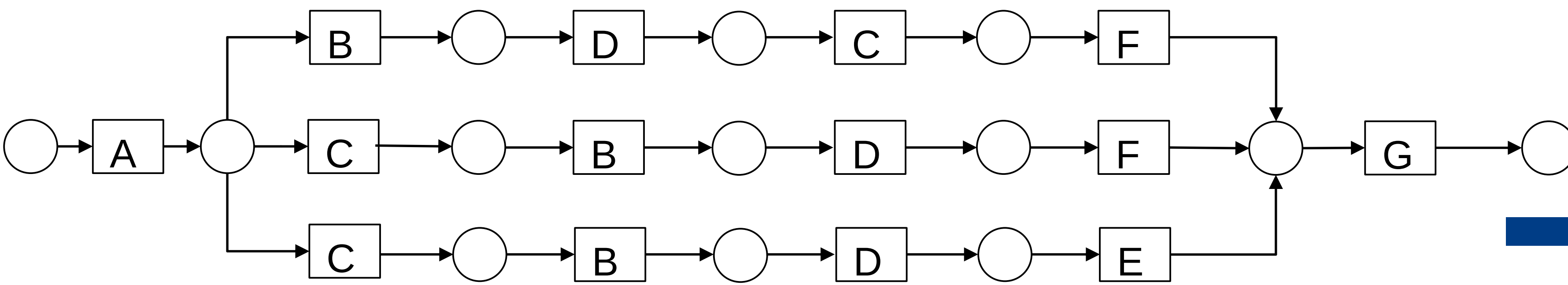
A	>>	B	C	D	E	>>	G
A	C	B	>>	D	>>	F	G

$\text{Cost} = 4 = 0.67$

A	>>	B	C	D	E	G
A	C	B	>>	D	E	G

$\text{Cost} = 2 = 1 - \frac{2}{6+6} = 0.83$

move log cost = 6



move model cost = 6

The Problem of Finding Optimal Alignments

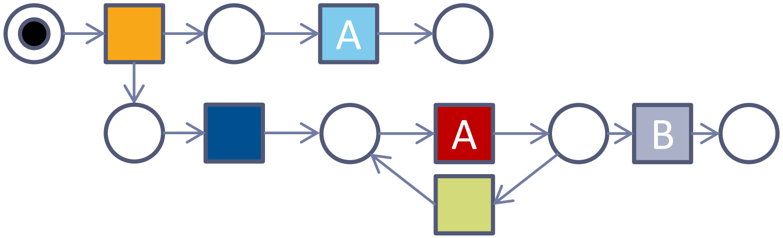
Christian-Albrechts-Universität zu Kiel

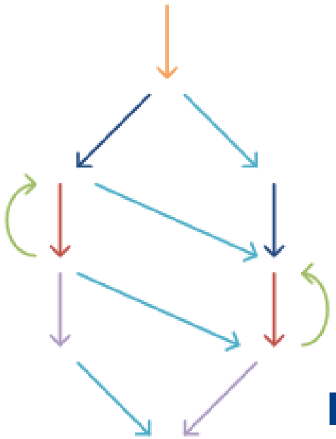
The search space is a "product" of the statespace of the model and the trace

Each node is a combination of a state in the model and the executed events in the trace

Each arc is a move in model, move in log or move in both

Example:

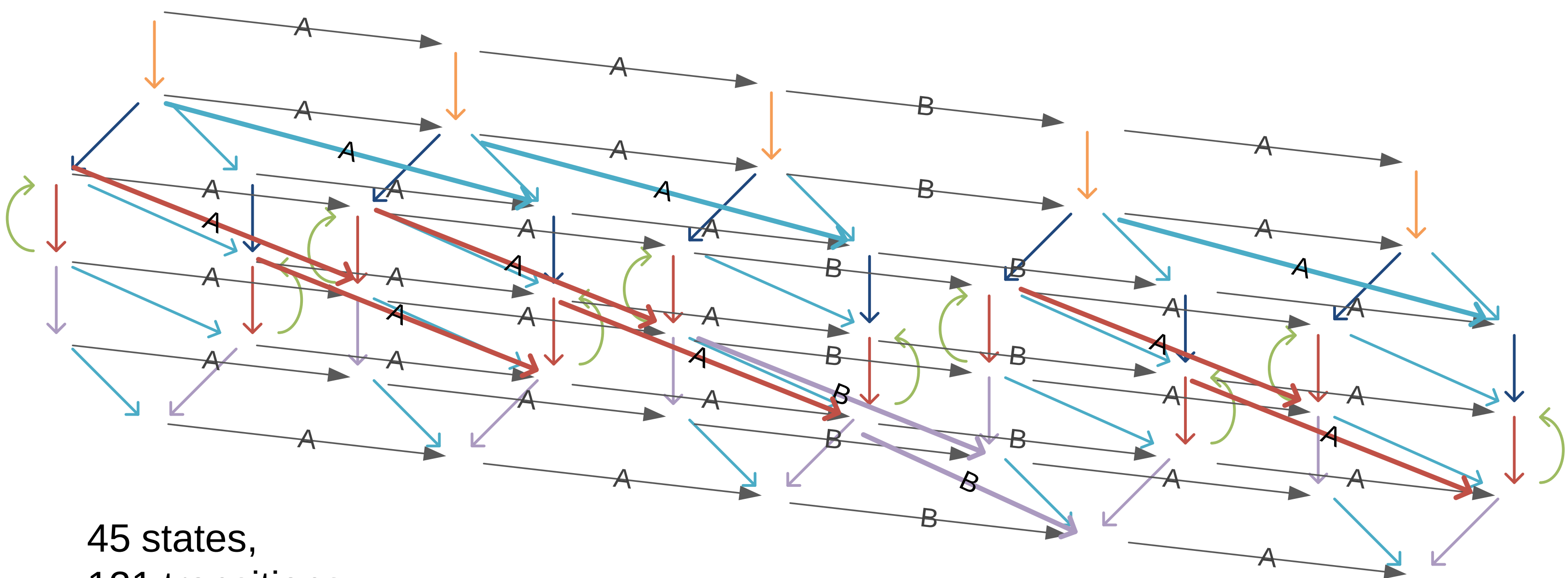




9 states,
13 transitions

Trace: < A, A, B, A >

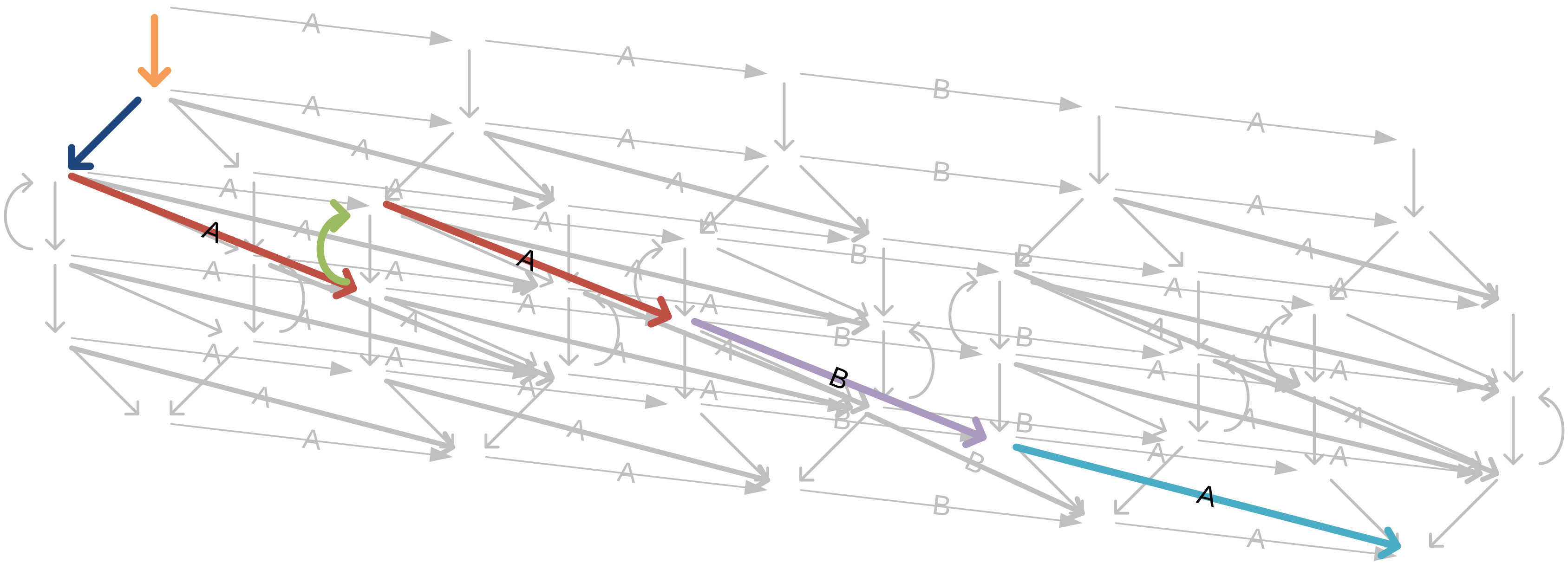
Alignment search space



45 states,
121 transitions

Find the shortest path from the top-left to the bottom right

Alignment search space

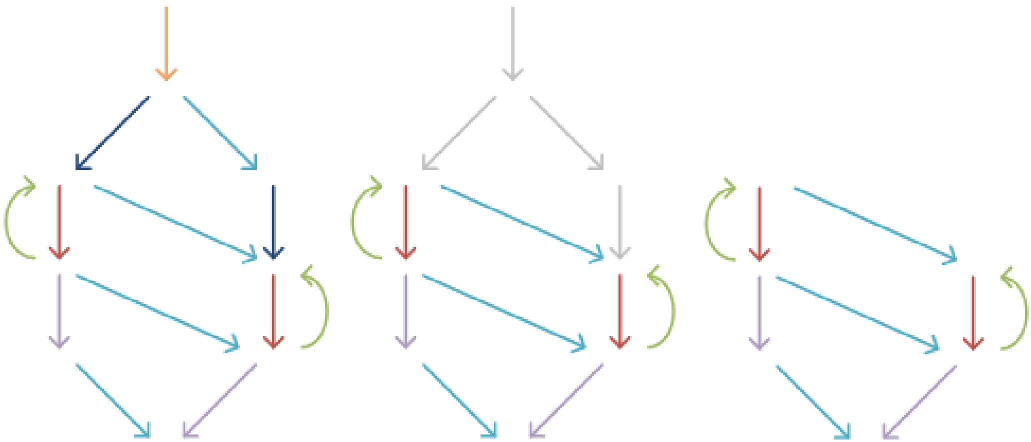


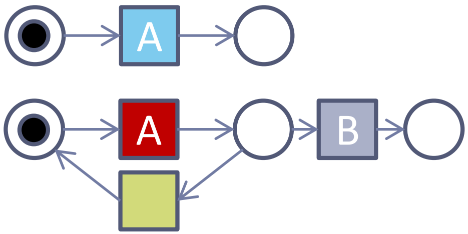
The alignment for trace <A, A, B, A>:

< (0,–), (1,–), (2, A), (3,–), (2,A), (4,B), (5,A) >

Implicit Execution

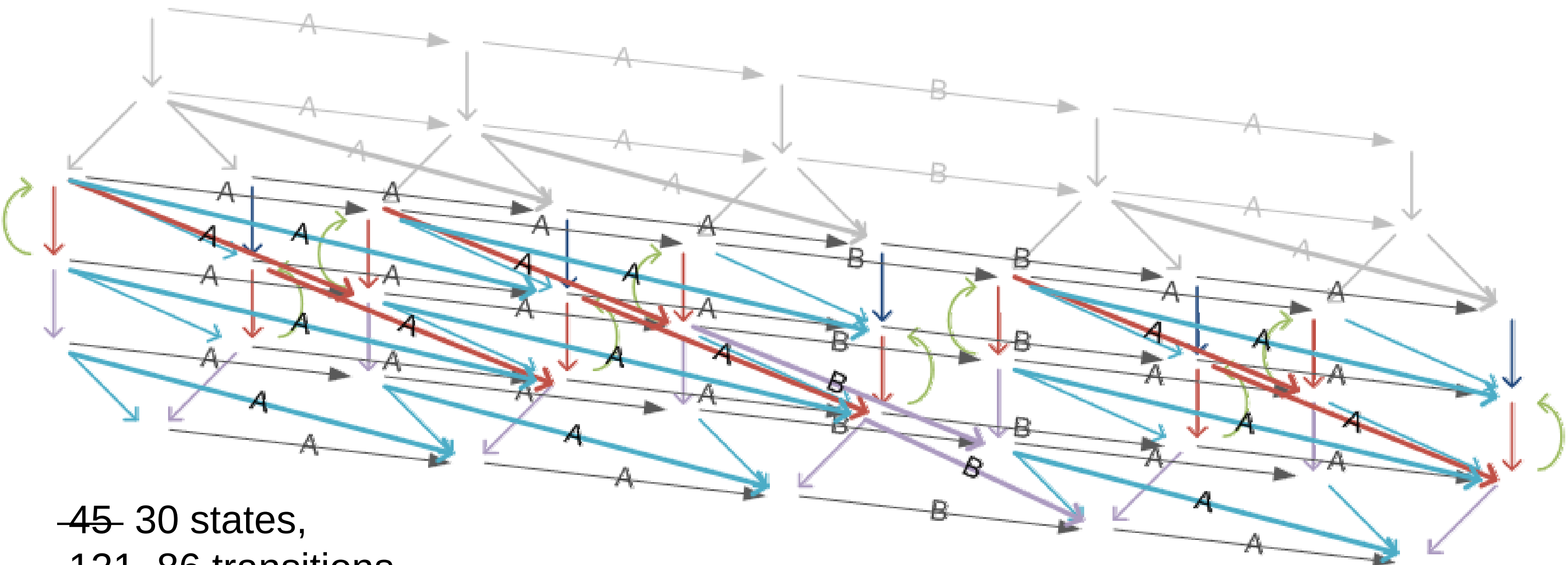
Implicit execution of silent transitions in a net system





6 states,
9 transitions

Implicit execution – Reduced Statespace



~~45~~ 30 states,
~~121~~ 86 transitions

Search Heuristics

Christian-Albrechts-Universität zu Kiel

In general: heuristic must be admissible, i.e., never overestimate the cost to reach goal and be monotonic

- The estimated cost to reach the goal is not higher than the lowest possible cost from the current node

Simple estimator for our setting:

Length of remaining trace * minimal cost for each transition

- Never overestimates: best solution needs to align all the remaining transitions in this trace
- Is monotonic: move in model yields the same estimate, move in log and move in both lower the estimate

Minimal Cost per Transition

Christian-Albrechts-Universität zu Kiel

Recall: Standard cost function, for $x \in T_\sigma$ and $y \in T_\pi$:

- $\delta(x, \perp) = 1$ (move in log)
- $\delta(\perp, y) = 1$ (move in model)
- $\delta(x, y) = 0$ if $x = y$ (equal move in both)
- $\delta(x, y) = \infty$ if $x \neq y$ (different move in both)

But: A* search requires the number of paths with zero distance between nodes in the graph to be finite

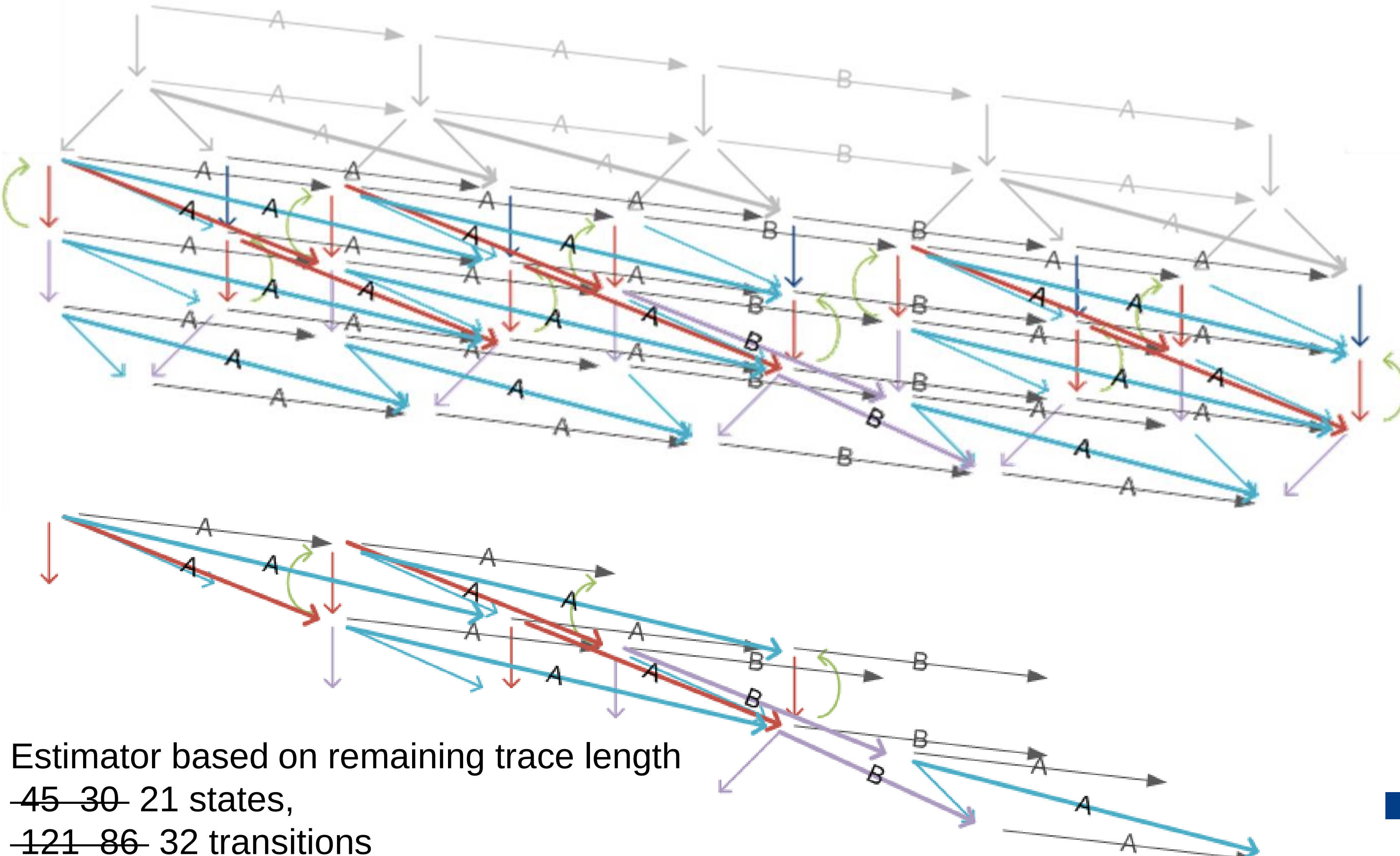
- An issue if the alignment search space contains cycles
- Such cycles may stem from cycles in the process model

Solution: Avoid steps with zero costs by introducing a negligible cost ϵ, with $\epsilon < 1$:

- $\delta(x, \perp) = \epsilon + 1$ (move in log)
- $\delta(\perp, y) = \epsilon + 1$ (move in model)
- $\delta(x, y) = \epsilon$ if $x = y$ (equal move in both)
- $\delta(x, y) = \infty$ if $x \neq y$ (different move in both)

Consequence: Search heuristic based on remaining trace length always yields cost > 0

Example



Estimator based on remaining trace length

~~45 30~~ 21 states,

~~121 86~~ 32 transitions

Conformance Analysis

Alignments enable quantification of conformance of event logs w.r.t. a process model

Compared to approaches discussed earlier, two common issues are not problematic when using alignments

- Silent transitions do not need to be treated
- Duplicated transitions in the model can be integrated directly

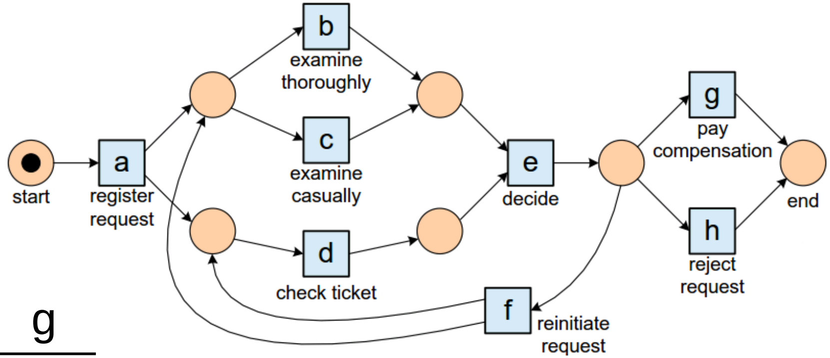
Alignments form the basis not only for conformance analysis, but a full range of analysis techniques

Local Feedback in Trace/Model

Characterise “hotspots” of non-conformance in trace or model

- Instead of considering elements of behavioural relations as entities for feedback, the violation pair (transition, move type) is used (move type is either move in log or move in model)
- Frequent occurrence of such a pair hints at hotspot

Illustration:



a	d	b	e	f	d	c	e	g
a	d	b		f	d	c		g

a	d	b	e	h
a	d	b		h

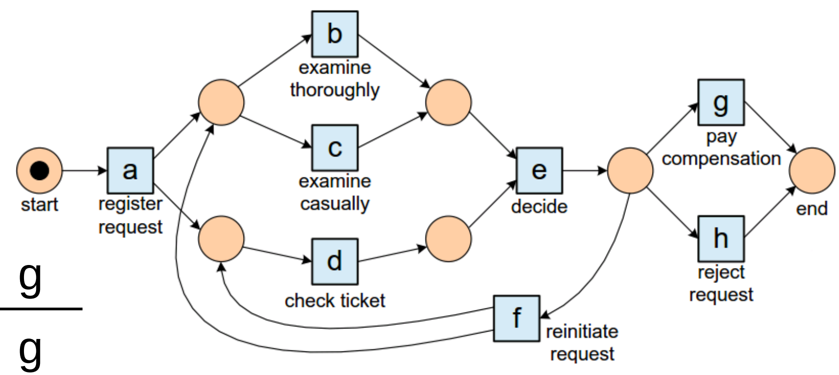
Global Feedback

Violation support: number of traces that show particular violation in optimal alignment

Confidence of violation rules: given two violations v and v', determine ratio of number of traces showing both violations and number of traces showing only violation v

a	d		b	e	h
a	d	x	b		h

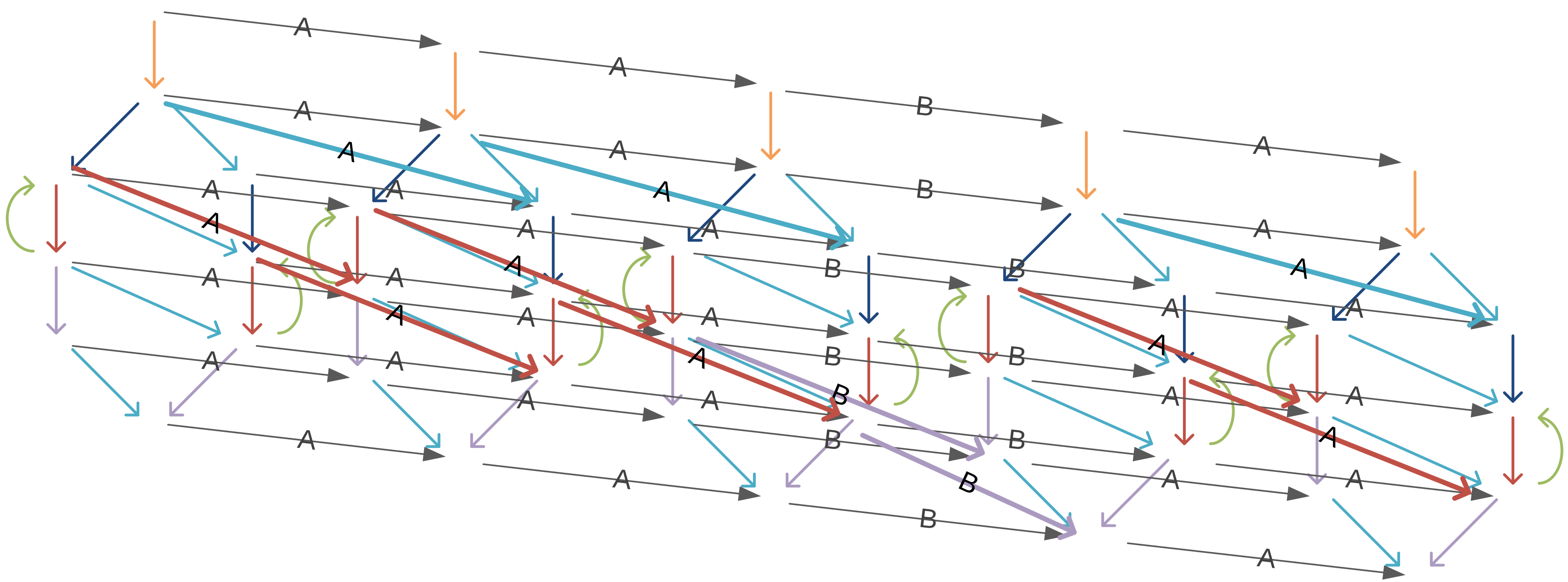
a	d	b	e	f	d	c	e	g
a	d	b	e	f	d	c	e	g



a	d		b	e	f	d	c	e	g
a	d	x	b		f	d	c		g

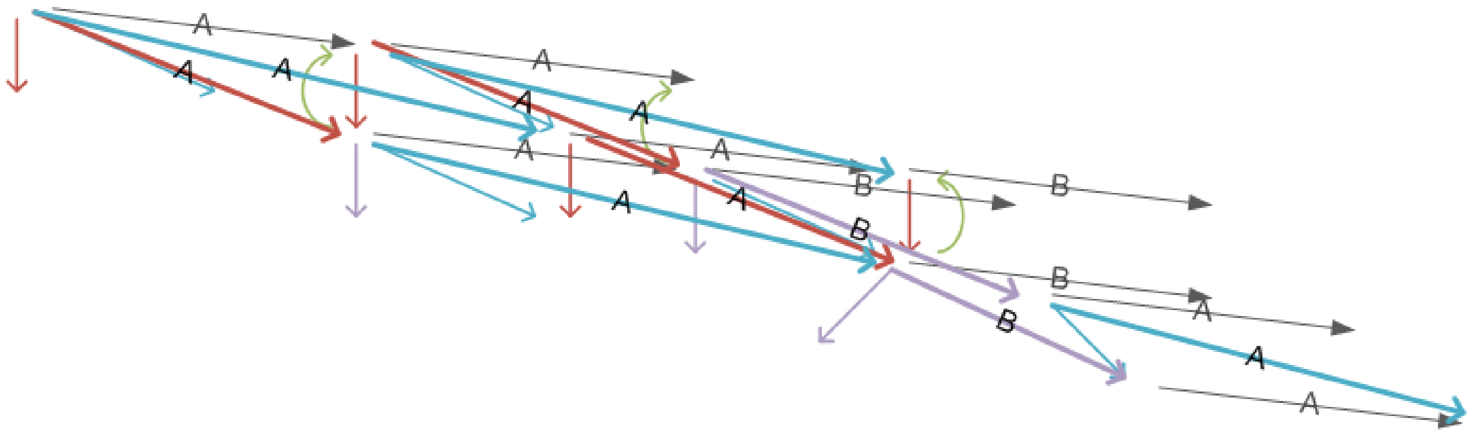
Product State Space by Example

Christian-Albrechts-Universität zu Kiel



Find the shortest path from the top-left to the bottom right

A* by Example



Simple estimator for our setting:
Length of remaining trace * minimal cost for each transition

More Efficient Alignment Construction

Search-based alignment construction faces a search space of exponential size

Simple heuristic presented so far does not work well in practice

- Many states that can be explored get the same estimate for the remaining cost
- Completely neglects the structure of the process model

To speed-up alignment construction:

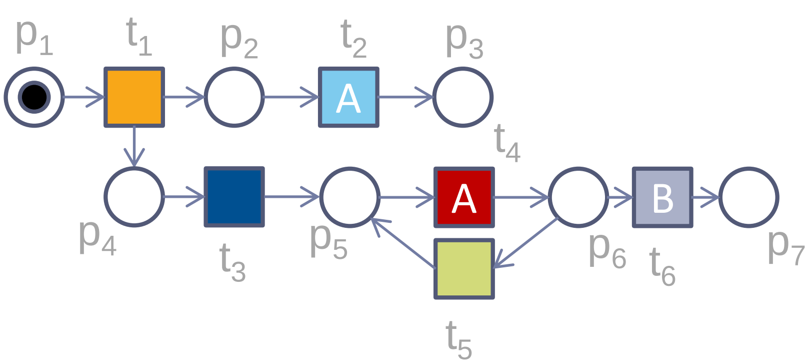
- Define a more advanced heuristic to guide the search
- Optimise the search further by pruning the search space
- Introduce second order prioritisation for candidate states in the exploration
- Reuse results in the computation of the heuristic or approximate it

Incidence Matrix

Represent the structure of a Petri net (P,T,F) in terms of the flow relation as a matrix

- Matrix of |P| rows and |T| columns
- Values of −1 and 1 indicate the presence of a flow arc from a place to a transition, or from a transition to a place, respectively

Example:



$$
\begin{array}{c|cccccc}
 & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 \\
\hline
p_1 & -1 & & & & & \\
p_2 & 1 & -1 & & & & \\
p_3 & & 1 & & & & \\
p_4 & 1 & & -1 & & & \\
p_5 & & & 1 & -1 & 1 & \\
p_6 & & & & 1 & -1 & -1 \\
p_7 & & & & & & 1
\end{array}
$$

Marking Equation

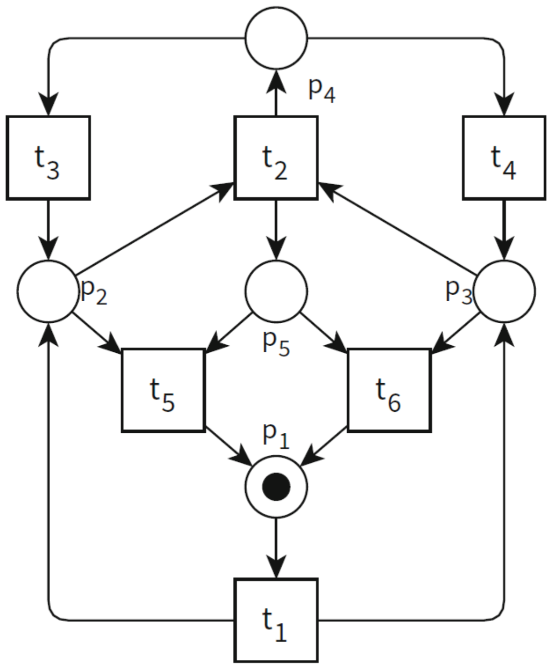
Christian-Albrechts-Universität zu Kiel

The Marking Equation of a Petri net system

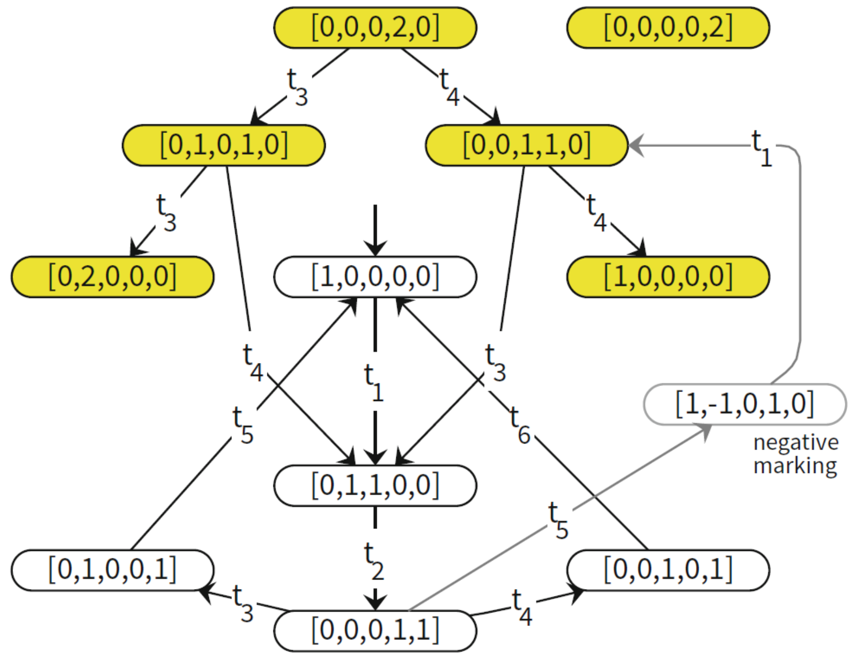
- Establishes link between the net structure and reachability of markings
- Provides a necessary, yet not sufficient condition for reachability of markings

Marking Equation Example

Christian-Albrechts-Universität zu Kiel



Potentially reachable markings
(highlighted are not reachable):



Consider a firing sequence:

$\langle t_1, t_2, t_5, t_1, t_3 \rangle$

$$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 & 0 & 0 & 1 & 1 \\ 1 & -1 & 1 & 0 & -1 & 0 \\ 1 & -1 & 0 & 1 & 0 & -1 \\ 0 & 1 & -1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & -1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Another Heuristic for A*

Christian-Albrechts-Universität zu Kiel

Idea:

- Sequence of transition firings in the product state space yields an integer solution to the marking equation
- Cost function over this sequence is the alignment cost
- If we simply solve the linear equation system of the marking equation, while minimizing cost, we get a solution that cannot have higher costs than the actual path

The heuristic is defined as being the aggregated cost of the non-zero transitions in

Optimisation: Step Sorting

Christian-Albrechts-Universität zu Kiel

In the product state space: Model moves and log moves are independent

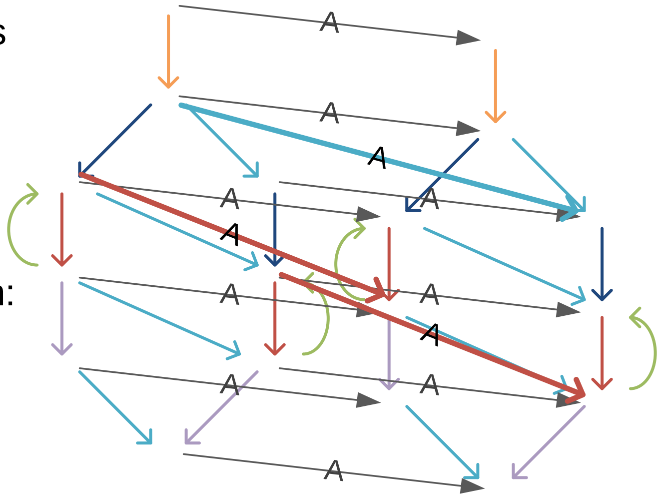
- If an optimal alignment contains a sequence of model moves and log moves without any synchronous moves in between, these can be interleaved arbitrarily

Idea: Exploit this in A* search

- Consider only sequences of model moves followed by log moves or vice versa between synchronous moves
- Sorting model and log moves can speed up the search: Fewer edges are explored

Realisation: Sort moves in the exploration

- Consider type of move when selecting a successor
- Take selection based on last step of the shortest path to reach the current state
- Generally: Sort model moves first or log moves first
- Biggest effect if an optimal alignment contains long sequences of log and model moves

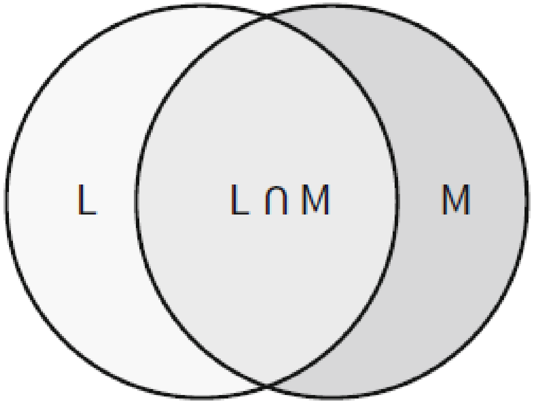


Alignment-based Measures

Alignments enable the quantification of the relation between an event log and a process model

In the context of conformance checking:
Fitness between a log and a model is most relevant

- Assume that the model is normative
- Quantify the extent of deviations of a trace from a model



However, alignments further enable the definition of measures to assess the precision-generalisation trade-off

Fitness Measures

Absolute fitness of a trace σ regarding a given model

- Defined by the cost δ of the optimal alignment of σ with the model
- Using the standard cost function, this corresponds to the number of moves in log and moves in model

Absolute fitness of a multiset $S = [\sigma_1^{n_1}, \dots, \sigma_m^{n_m}]$ of traces regarding model M with complete execution sequences Π

- Sum of costs of optimal alignments of traces

$$\mathrm{fcost}(S, \Pi) = \sum_{1 \le i \le m} \mathrm{n_i}\, \delta(\gamma^*(\sigma_i, \Pi))$$

Assess Precision

Quantify the behaviour of the model not seen in the log

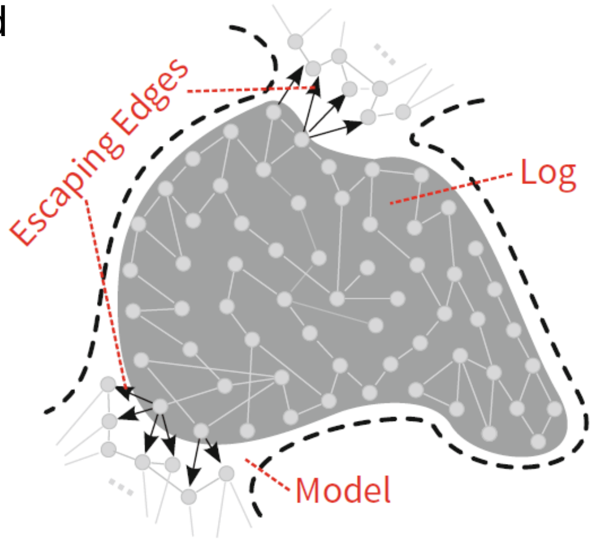
Assumption: All traces fit the model

Assumption: Model is deterministic

- In any state, there cannot be two transitions with the same label
- Non-deterministic models can be determinised (before computing an alignment)

General idea

- Based on traces of the log, characterise states in the model with "escaping edges"
- Those are transitions representing behaviour that immediately follows behaviour of the log but is not contained in the log



Precision Measure

Determine the language of "escaping edges", denoted L_{esc}

- It contains all sequences of activity labels $\sigma \cdot \langle a \rangle$, such that
  - σ is a prefix of a trace in the log and a prefix of an execution sequence of the model
  - $\sigma \cdot \langle a \rangle$ is not a prefix of a trace in the log, but still a prefix of an execution sequence of the model
- Note: Since both the log and the set of activity labels are finite, so is the language of escaping edges

Precision of a log as a set of traces $\mathrm{L} = [\sigma_1, \ldots, \sigma_m]$ and a model M with execution sequences Π:

$$\mathrm{precision} = \frac{|\,\mathrm{Prefix}(\mathrm{L} \cap \Pi)|}{|\,\mathrm{Prefix}(\mathrm{L} \cap \Pi)| + |\mathrm{L}_{esc}|}$$

where Prefix(X) is the set of all prefixes of language X.

Alignment-based Precision Measure

Alignments enable us to drop the assumption of fitting traces in the computation of precision

- Intuitively, the alignment gives an execution sequence best resembling an unfitting trace
- This execution sequence is used when building the joint language
- Instead of relying on $\mathrm{L} \cap \Pi$, we rely on $\mathrm{L}_{\mathrm{align}} = \left\{ \gamma^*(\sigma, \Pi)_{|2} \mid \sigma \in \mathrm{L} \right\}$ where $\gamma^*{}_{|2}$ is the projection of the alignment on the second component (i.e., the execution sequence)

Then, redefine the language of “escaping edges”, denoted $\mathrm{L}_{\mathrm{esc}}$ based on alignments:

- It contains all sequences of activity labels $\sigma \cdot \langle a \rangle$, such that
  - σ is a prefix of an element of $\mathrm{L}_{\mathrm{align}}$
  - $\sigma \cdot \langle a \rangle$ is not a prefix of a trace in the log, but still an element of $\mathrm{L}_{\mathrm{align}}$

Precision is then measured as :

$$\text{precision} = \frac{|\,\mathrm{Prefix}(\mathrm{L}_{\mathrm{align}})|}{|\,\mathrm{Prefix}(\mathrm{L}_{\mathrm{align}})| + |\mathrm{L}_{\mathrm{esc}}|}$$

Chapter 5