Deviations in Business Processes

The commonality between a process manual and a concrete barrier: they've been there for years and everybody walks around them.

Outline

- Introduction to deviations
- Replaying behavior
- Alignments
- Conclusions

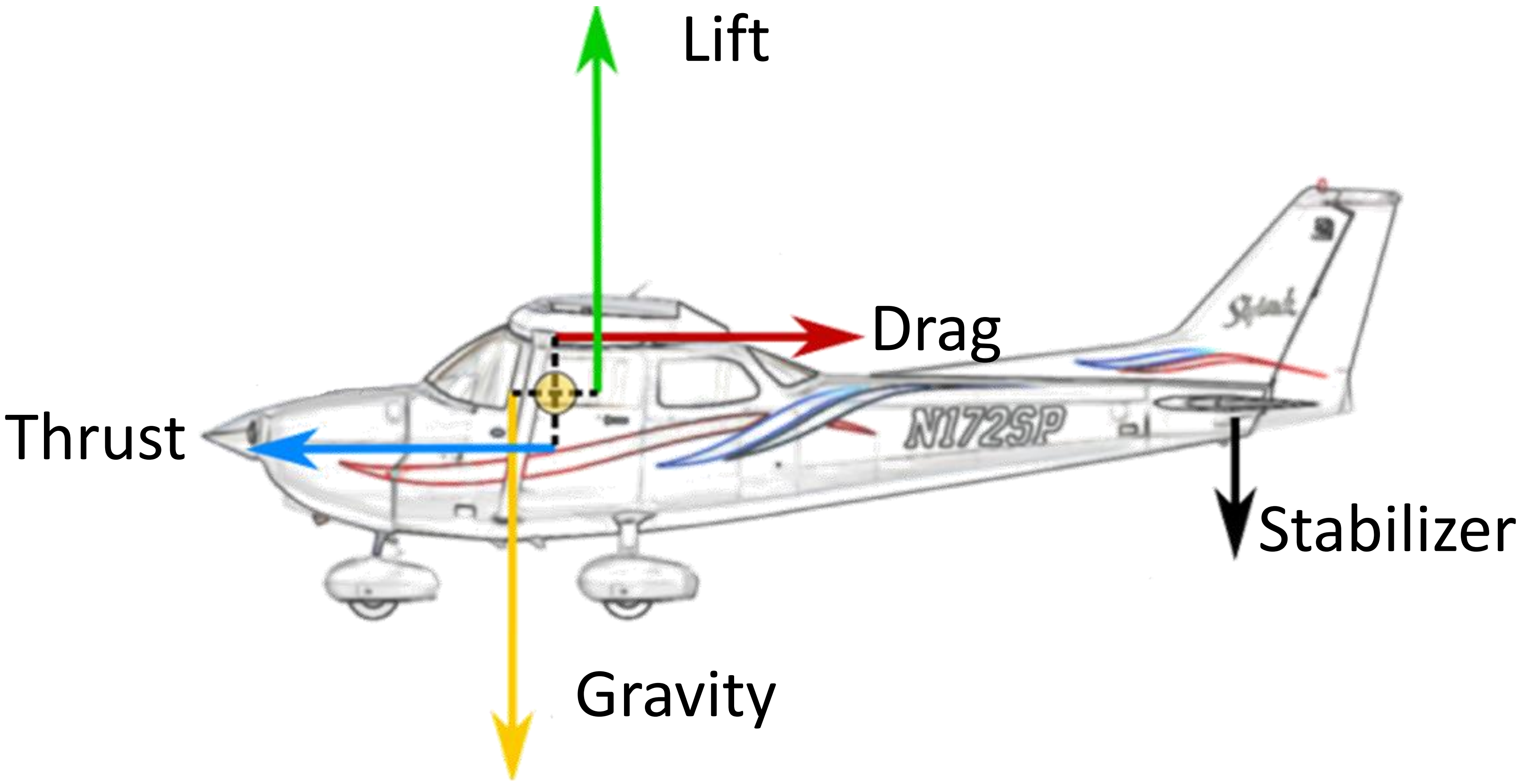
Typical Deviations

Typical Deviations

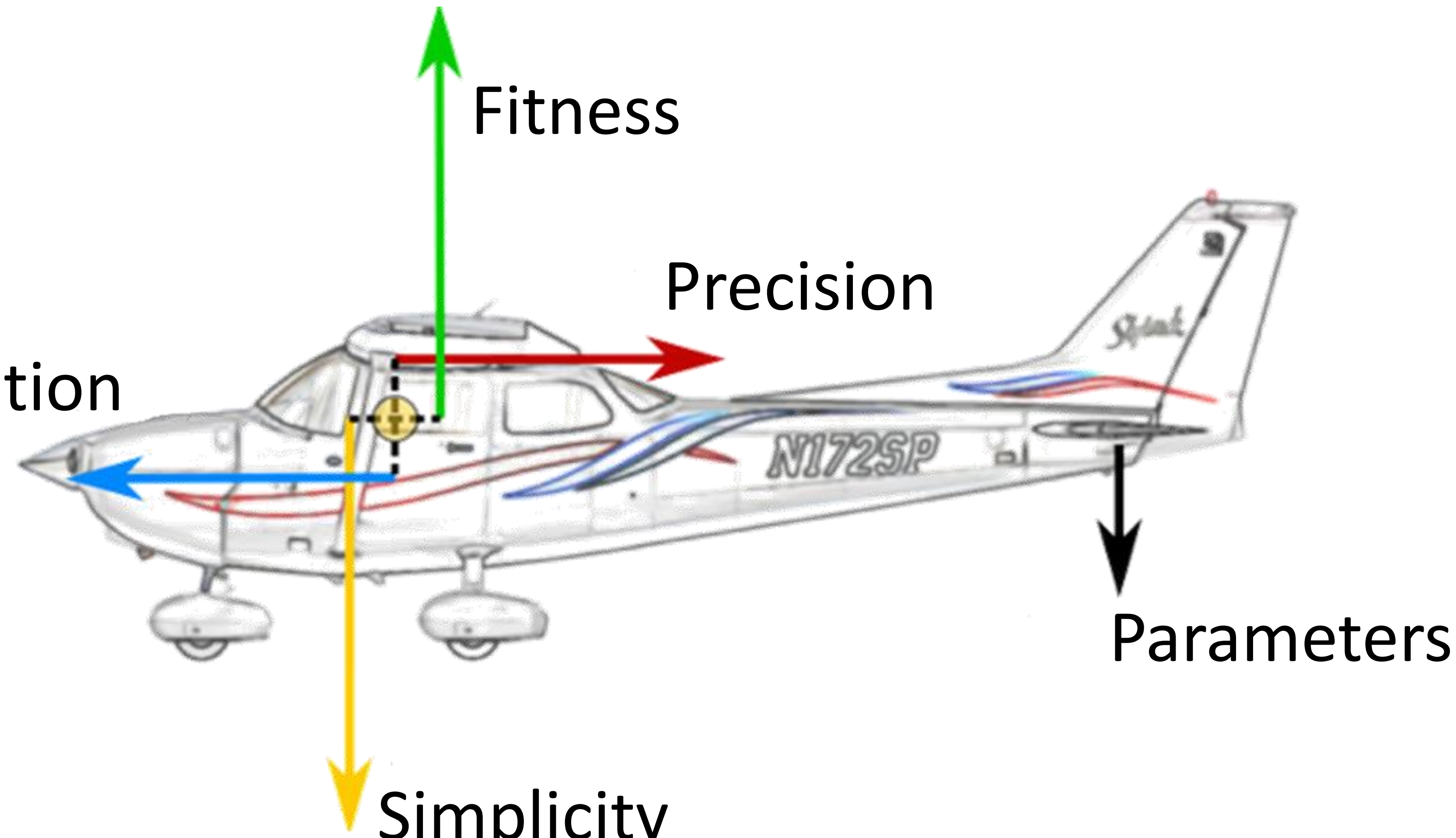
Typical Deviations

Typical Deviations

Conformance Checking

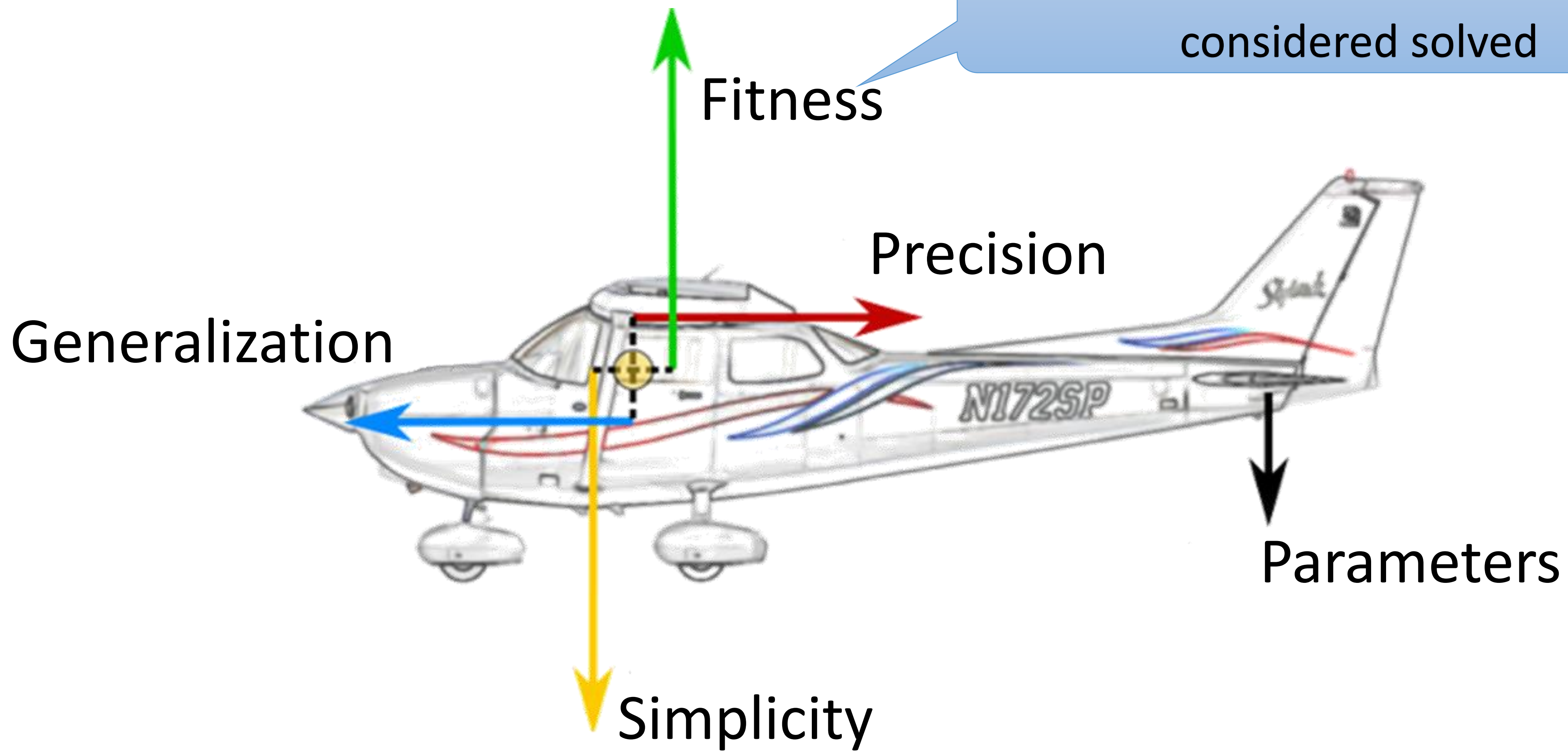


Conformance Checking

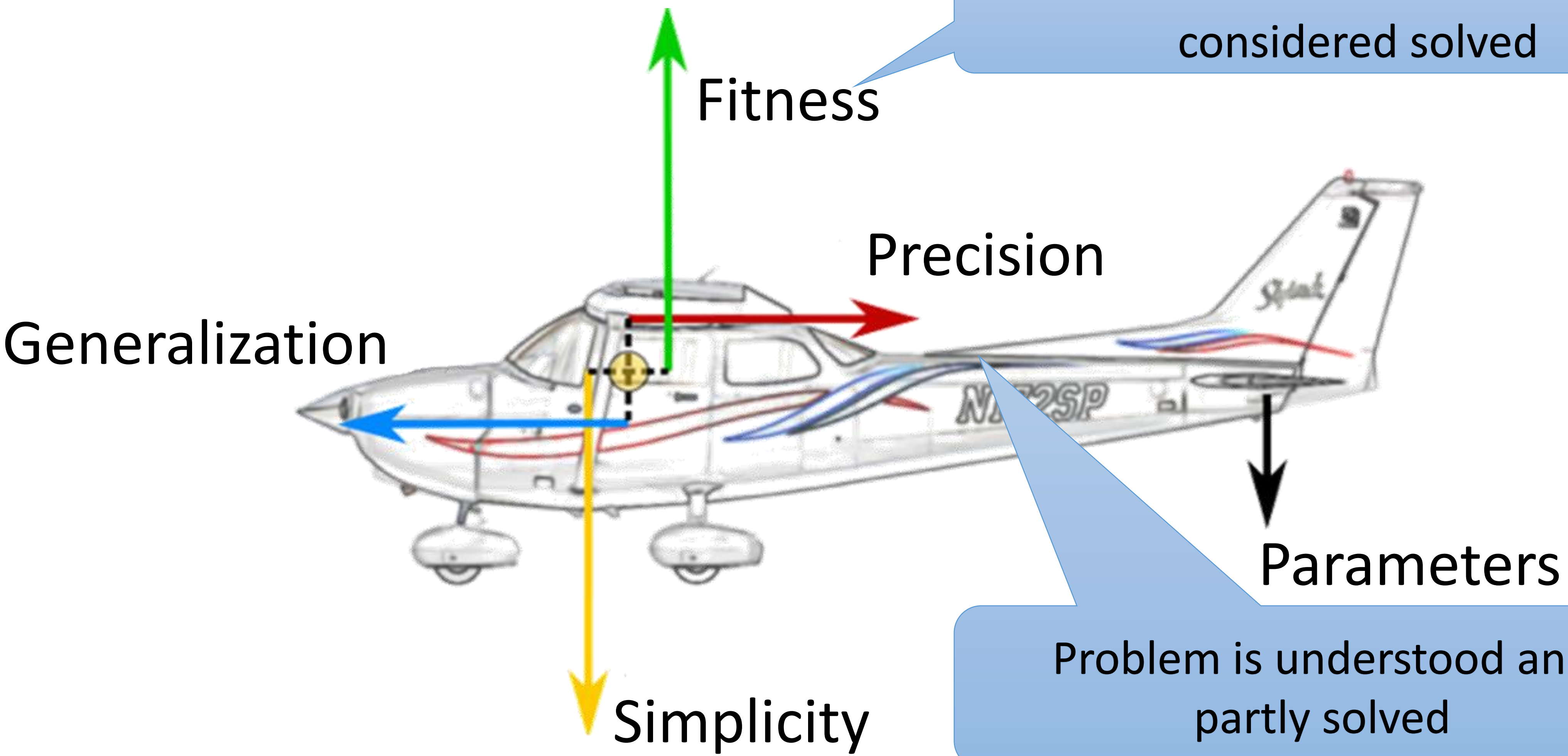


Conformance Checking

Problem is well understood and considered solved



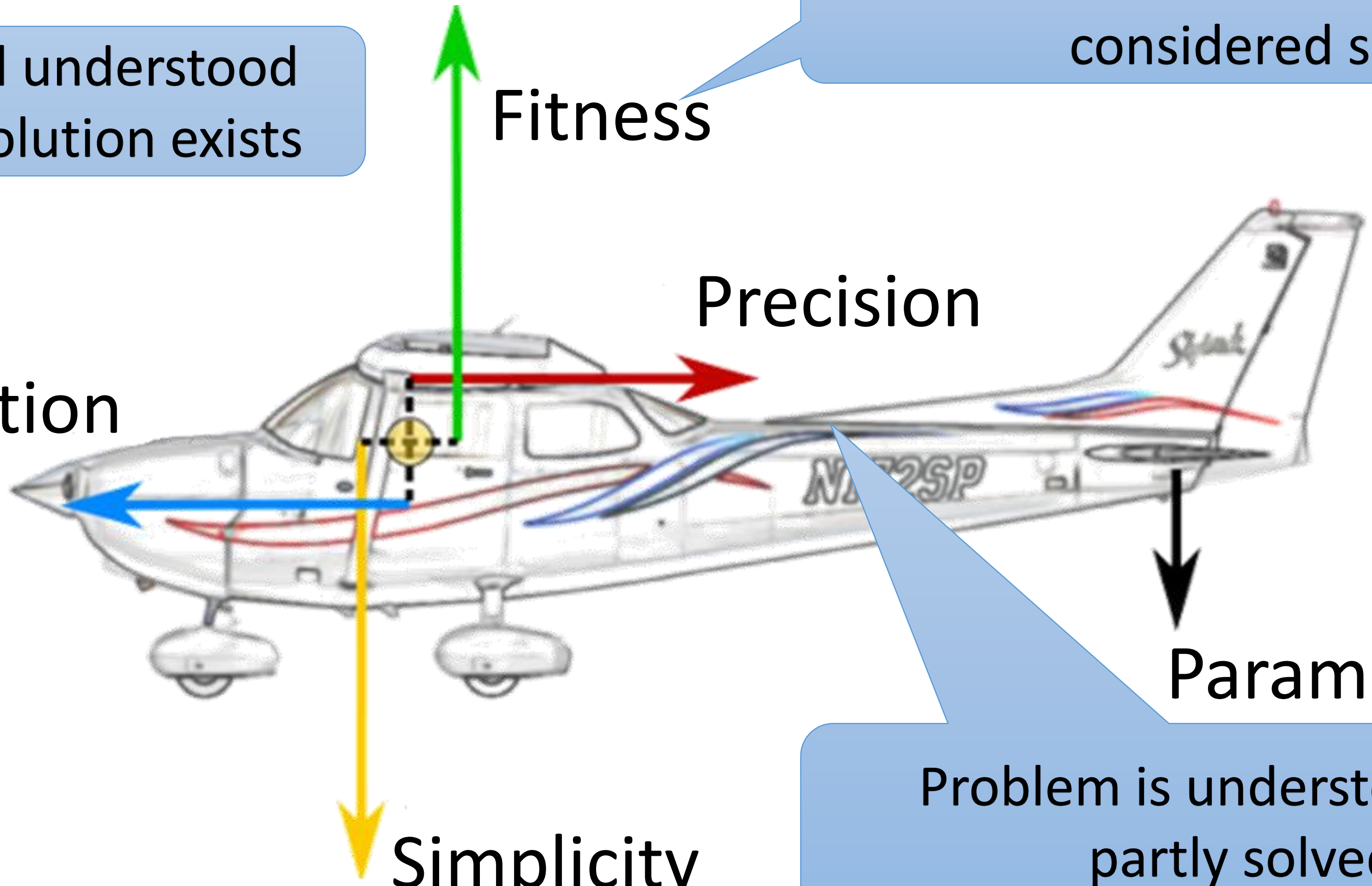
Conformance Checking



Problem is well understood and considered solved

Problem is understood and partly solved

Conformance Checking



Problem is well understood and considered solved

Problem is not well understood and no satisfying solution exists

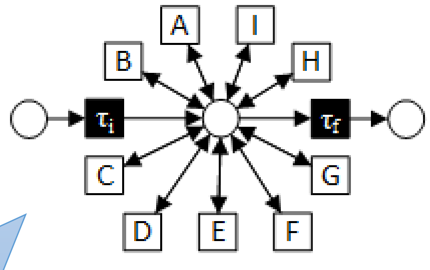
Problem is understood and partly solved

Conformance Metrics

<A,B,D,E,I>	100
<A,C,D,G,H,F,I>	50
<A,C,G,D,H,F,I>	50
<A,C,H,D,F,I>	50
<A,C,D,H,F,I>	50

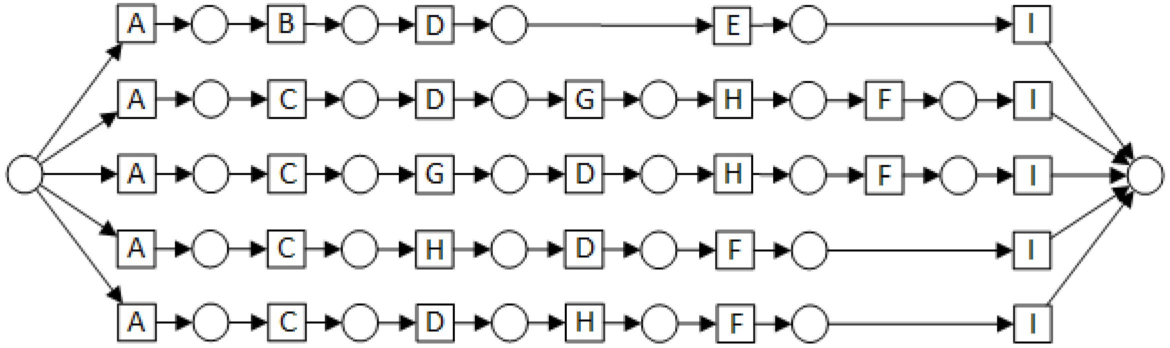
Conformance Metrics

<A,B,D,E,I>	100
<A,C,D,G,H,F,I>	50
<A,C,G,D,H,F,I>	50
<A,C,H,D,F,I>	50
<A,C,D,H,F,I>	50



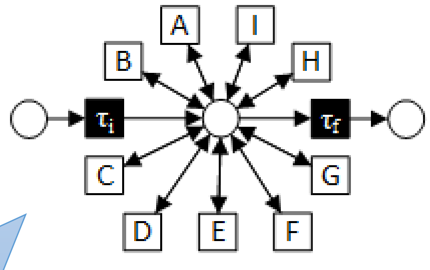
Fitting, simple, generalizing
Not precise

Conformance Metrics



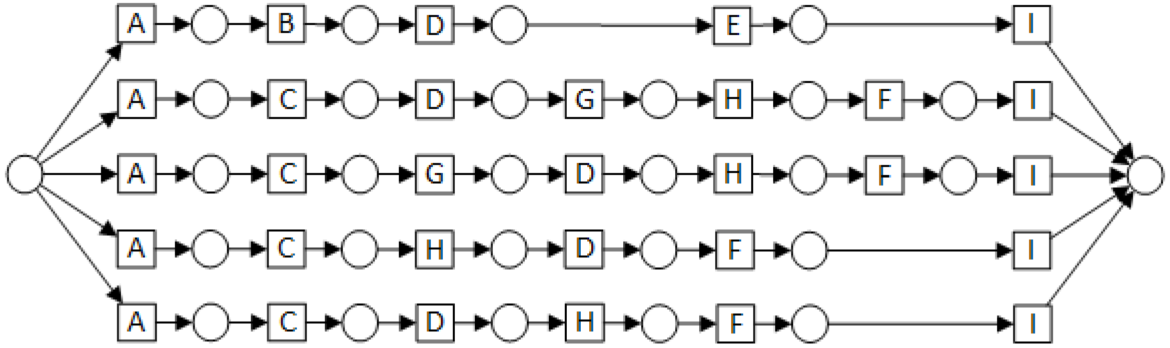
<A,B,D,E,I>	100
<A,C,D,G,H,F,I>	50
<A,C,G,D,H,F,I>	50
<A,C,H,D,F,I>	50
<A,C,D,H,F,I>	50

Fitting, complex,
Very precise



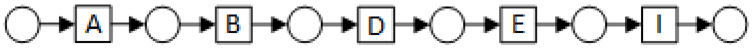
Fitting, simple, generalizing
Not precise

Conformance Metrics

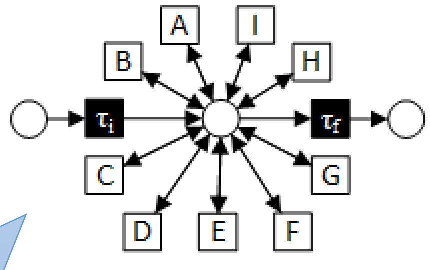


<A,B,D,E,I>	100
<A,C,D,G,H,F,I>	50
<A,C,G,D,H,F,I>	50
<A,C,H,D,F,I>	50
<A,C,D,H,F,I>	50

Fitting, complex,
Very precise

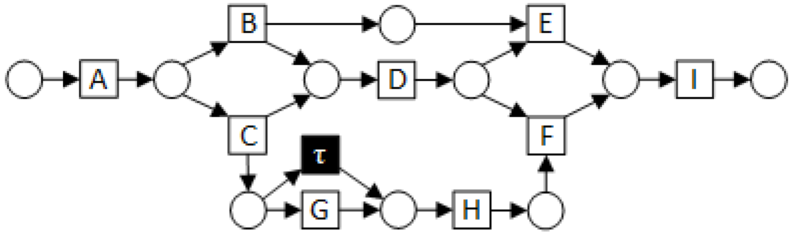


Simple, Precise
Not fitting, not generalizing

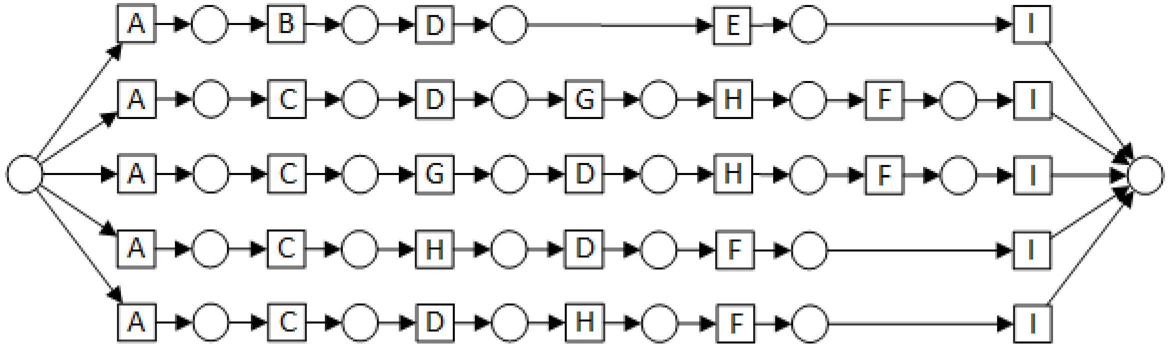


Fitting, simple, generalizing
Not precise

Conformance Metrics

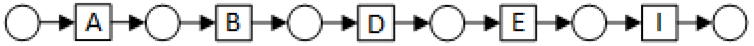


Fitting, simple,
Precise, Fairly generalizing

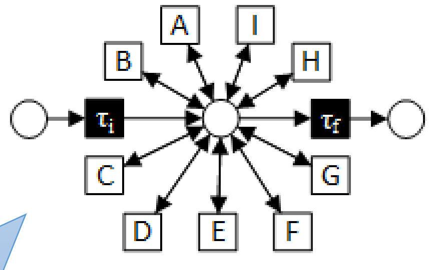


Fitting, complex,
Very precise

<A,B,D,E,I>	100
<A,C,D,G,H,F,I>	50
<A,C,G,D,H,F,I>	50
<A,C,H,D,F,I>	50
<A,C,D,H,F,I>	50



Simple, Precise
Not fitting, not generalizing



Fitting, simple, generalizing
Not precise

Conformance Metrics

- Fitness:
  - How much of the observed behavior fits the model?
  - Comparable to recall in data mining
  - Two techniques: token-replay and alignments

- Precision:
  - How much behavior does the model allow for that was not observed?
- Generalization:
  - How well does the model to explain the underlying system?
- Simplicity:
  - How simple is the model?

Detecting Deviations

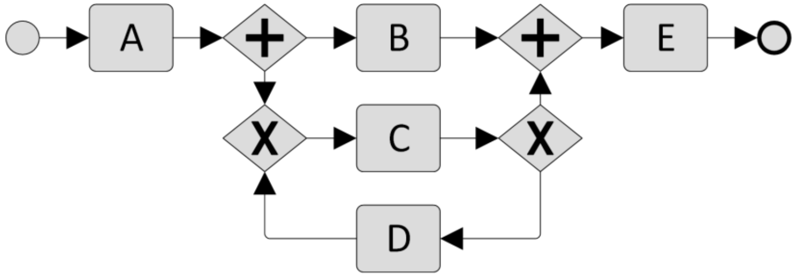
- Measuring conformance, implies
  being able to detect and explain deviations

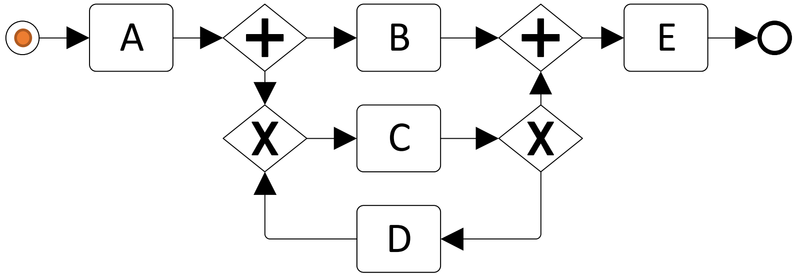
- Deviations can be detected by replaying event data on models

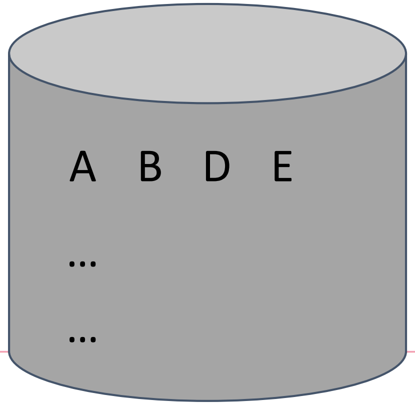
- The intuitive method: Token replay
- The sound method: Alignments

Detecting Deviations: Token replay

- Manually execute the steps in the data in the model and record produced, consumed, missing and remaining activations

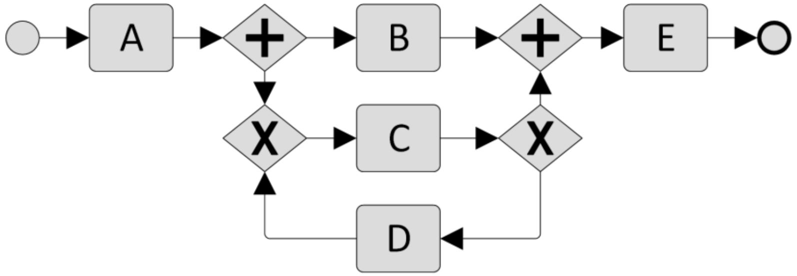


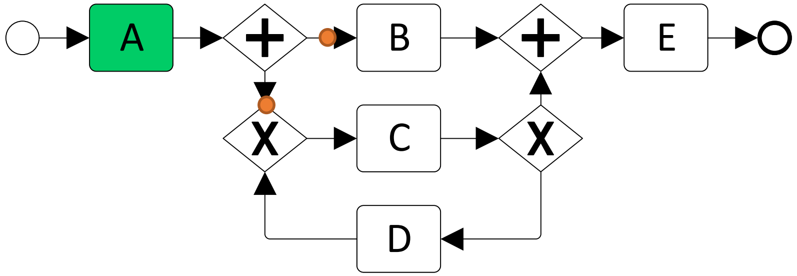


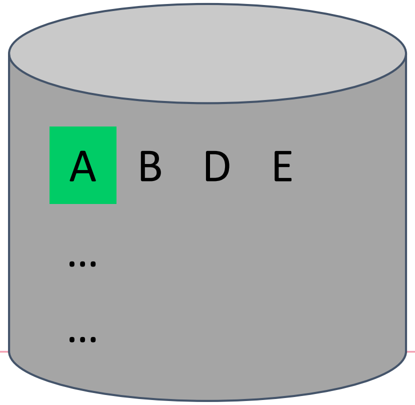


Detecting Deviations: Token replay

- Manually execute the steps in the data in the model and record produced, consumed, missing and remaining activations

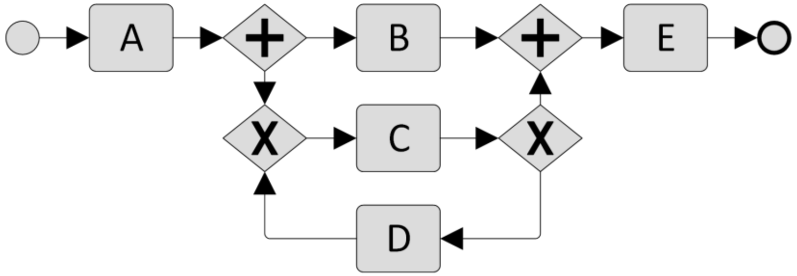


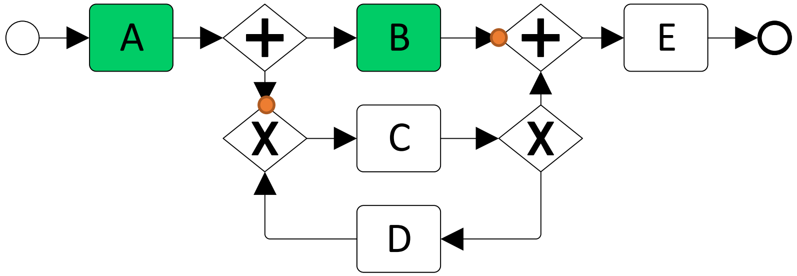


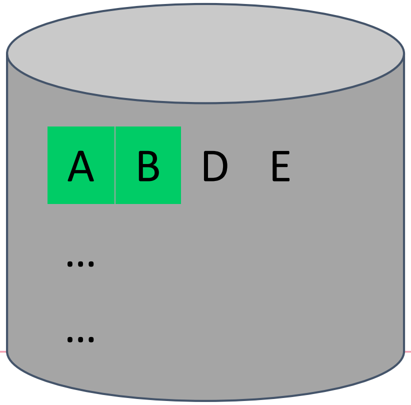


Detecting Deviations: Token replay

- Manually execute the steps in the data in the model and record produced, consumed, missing and remaining activations

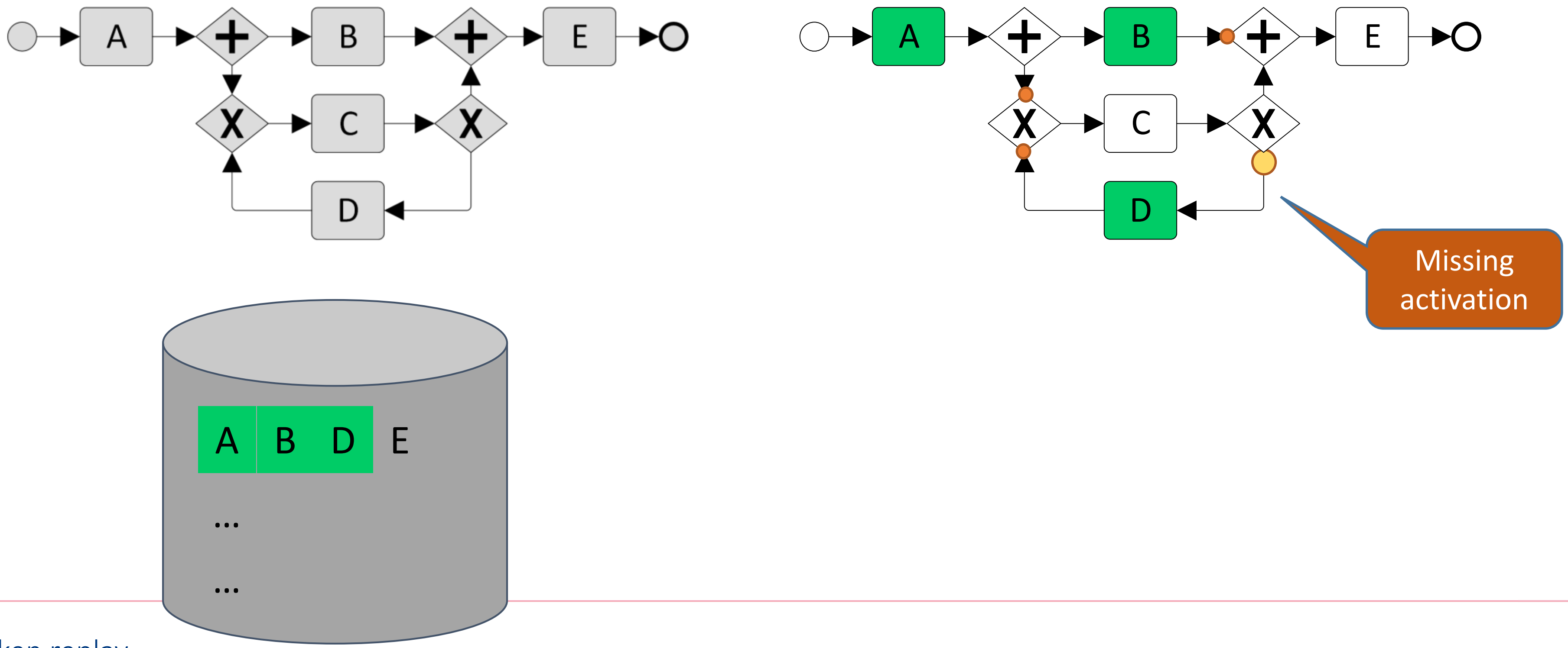






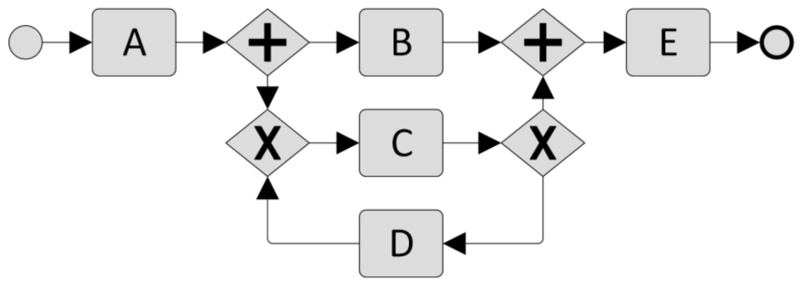
Detecting Deviations: Token replay

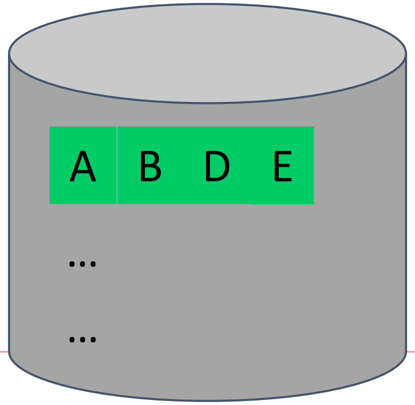
- Manually execute the steps in the data in the model and record produced, consumed, missing and remaining activations

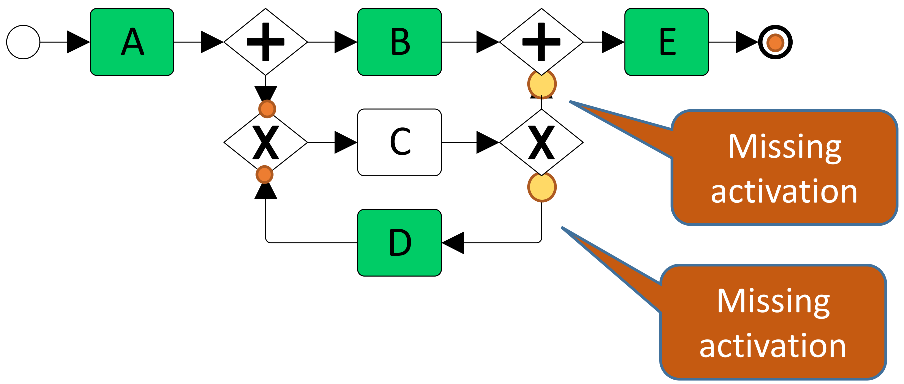


Detecting Deviations: Token replay

- Manually execute the steps in the data in the model and record produced, consumed, missing and remaining activations

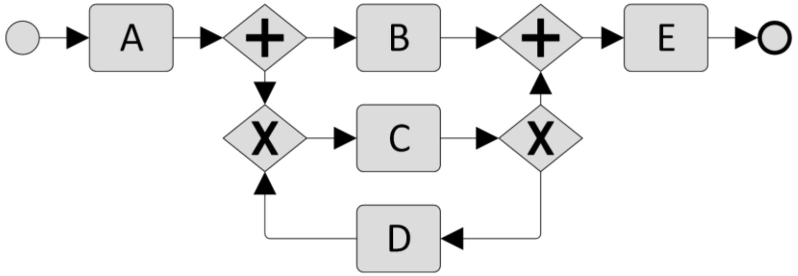


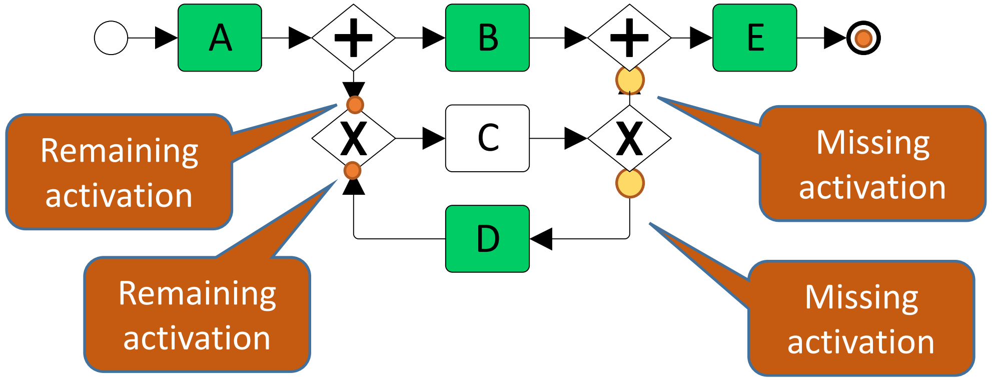


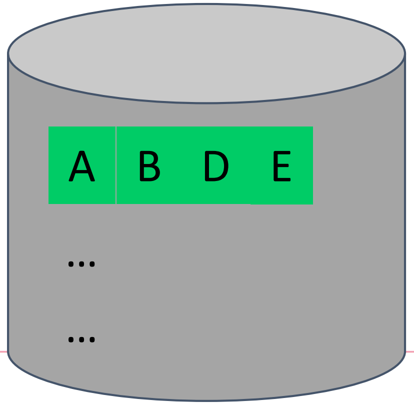


Detecting Deviations: Token replay

- Manually execute the steps in the data in the model and record produced, consumed, missing and remaining activations

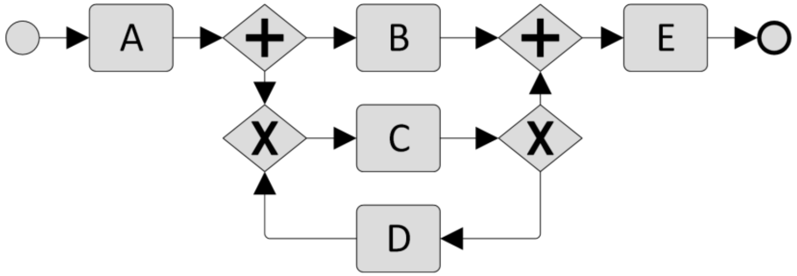


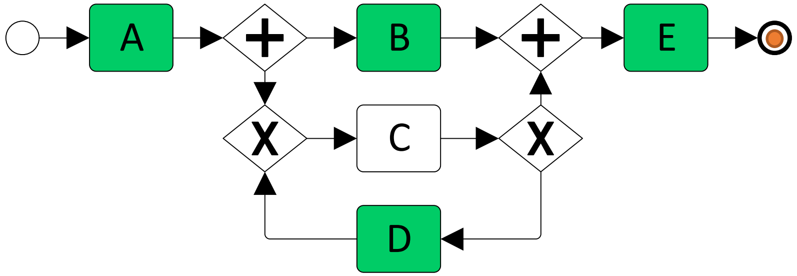


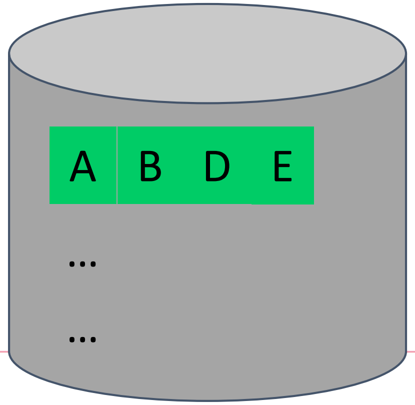


Detecting Deviations: Token replay

- Manually execute the steps in the data in the model and record produced, consumed, missing and remaining activations



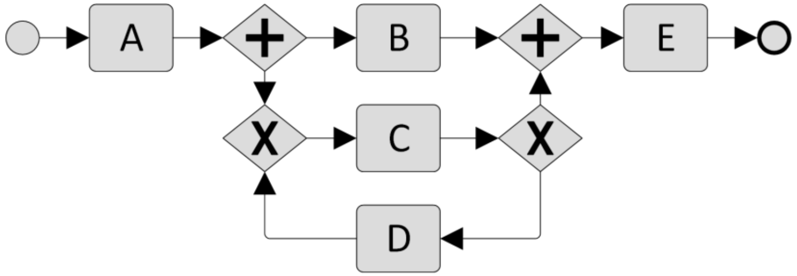


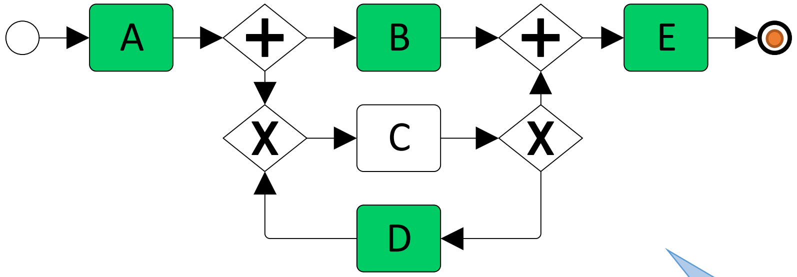


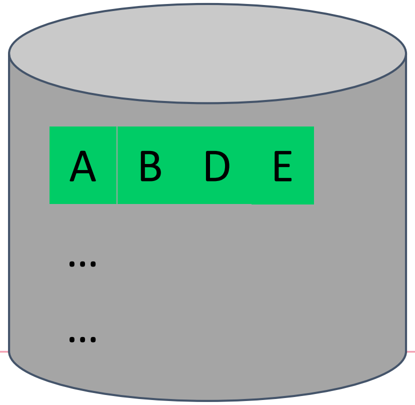
Produced activations: 6
Consumed activations: 6
Remaining activations: 2
Missing activations: 2

Detecting Deviations: Token replay

- Manually execute the steps in the data in the model and record produced, consumed, missing and remaining activations







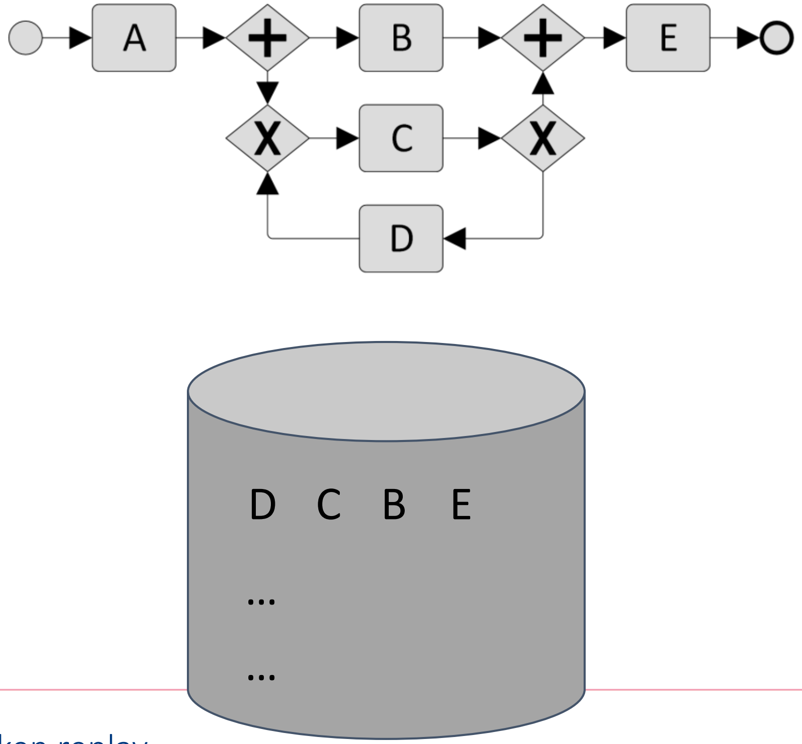
Produced activations: 6
Consumed activations: 6
Remaining activations: 2
Missing activations: 2

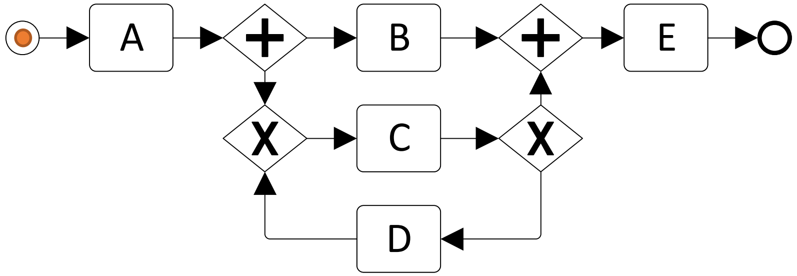
Fitness:

$$\tfrac{1}{2}\left(1 - {}^{m}/_{c}\right) + \tfrac{1}{2}\left(1 - {}^{r}/_{p}\right) = \tfrac{1}{2}\left(1 - {}^{2}/_{6}\right) + \tfrac{1}{2}\left(1 - {}^{2}/_{6}\right) = 0.67$$

Detecting Deviations: Token replay

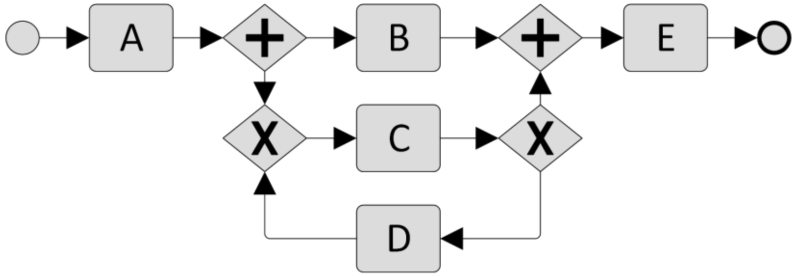
- Consider a different trace:

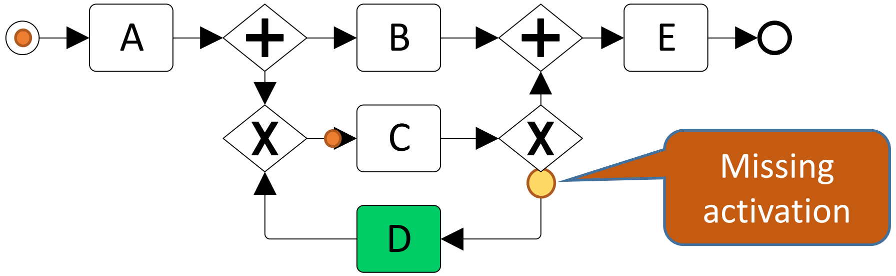


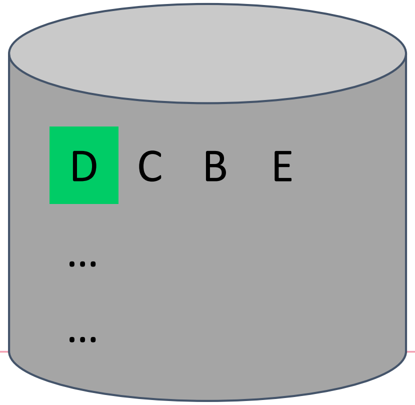


Detecting Deviations: Token replay

- Consider a different trace:

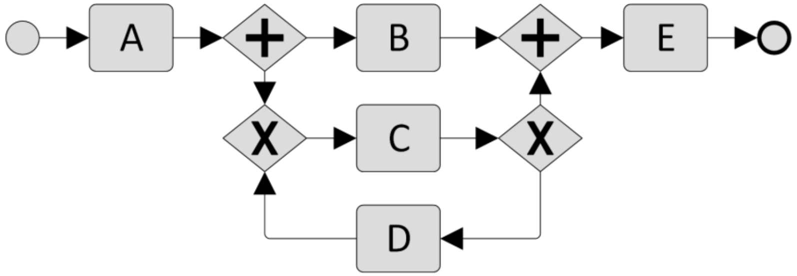


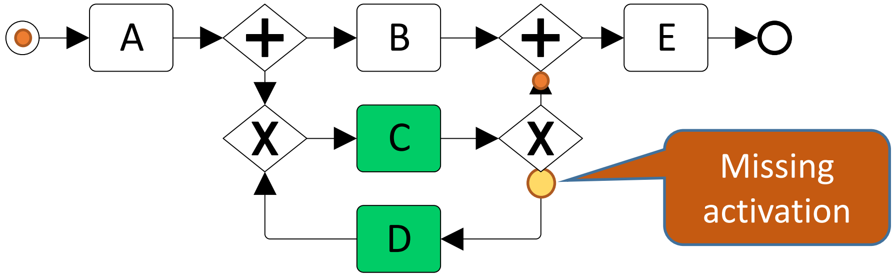


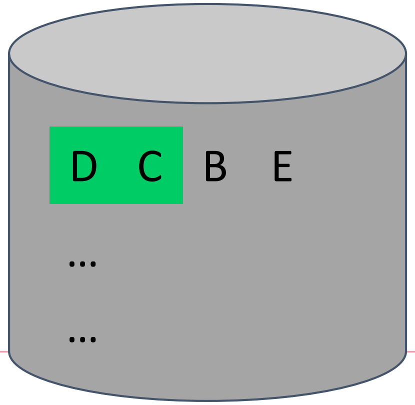


Detecting Deviations: Token replay

- Consider a different trace:

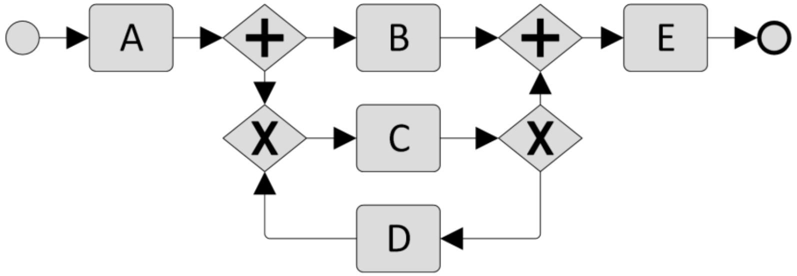


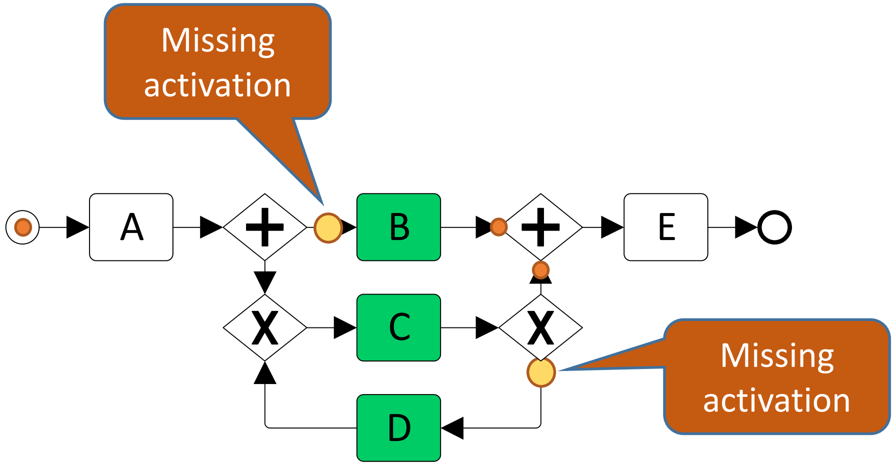


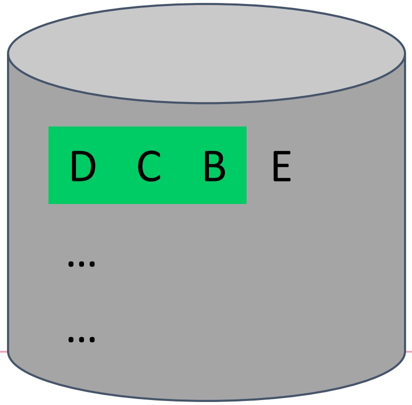


Detecting Deviations: Token replay

- Consider a different trace:

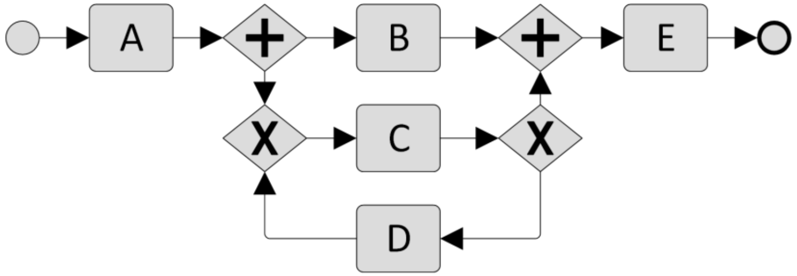


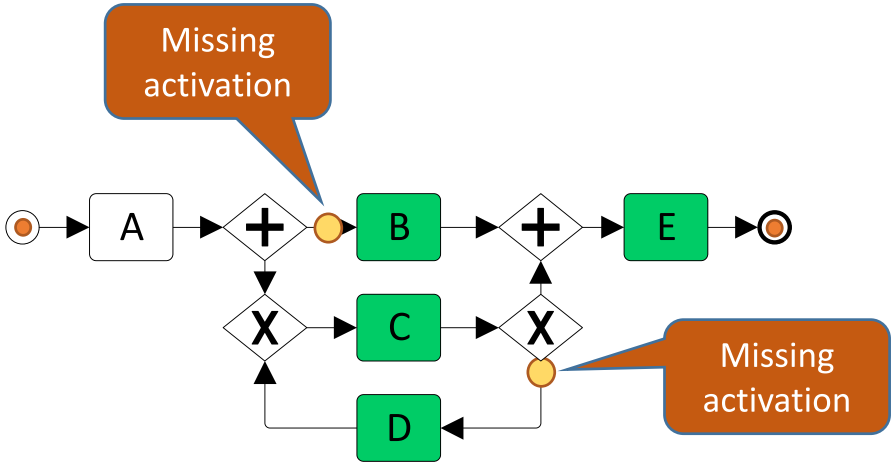


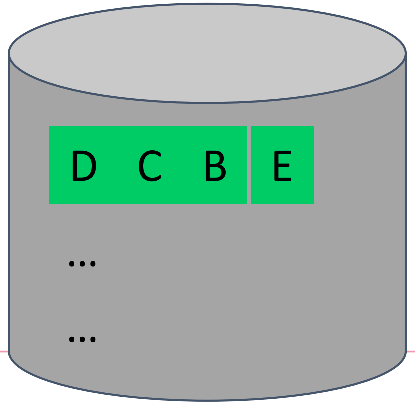


Detecting Deviations: Token replay

- Consider a different trace:

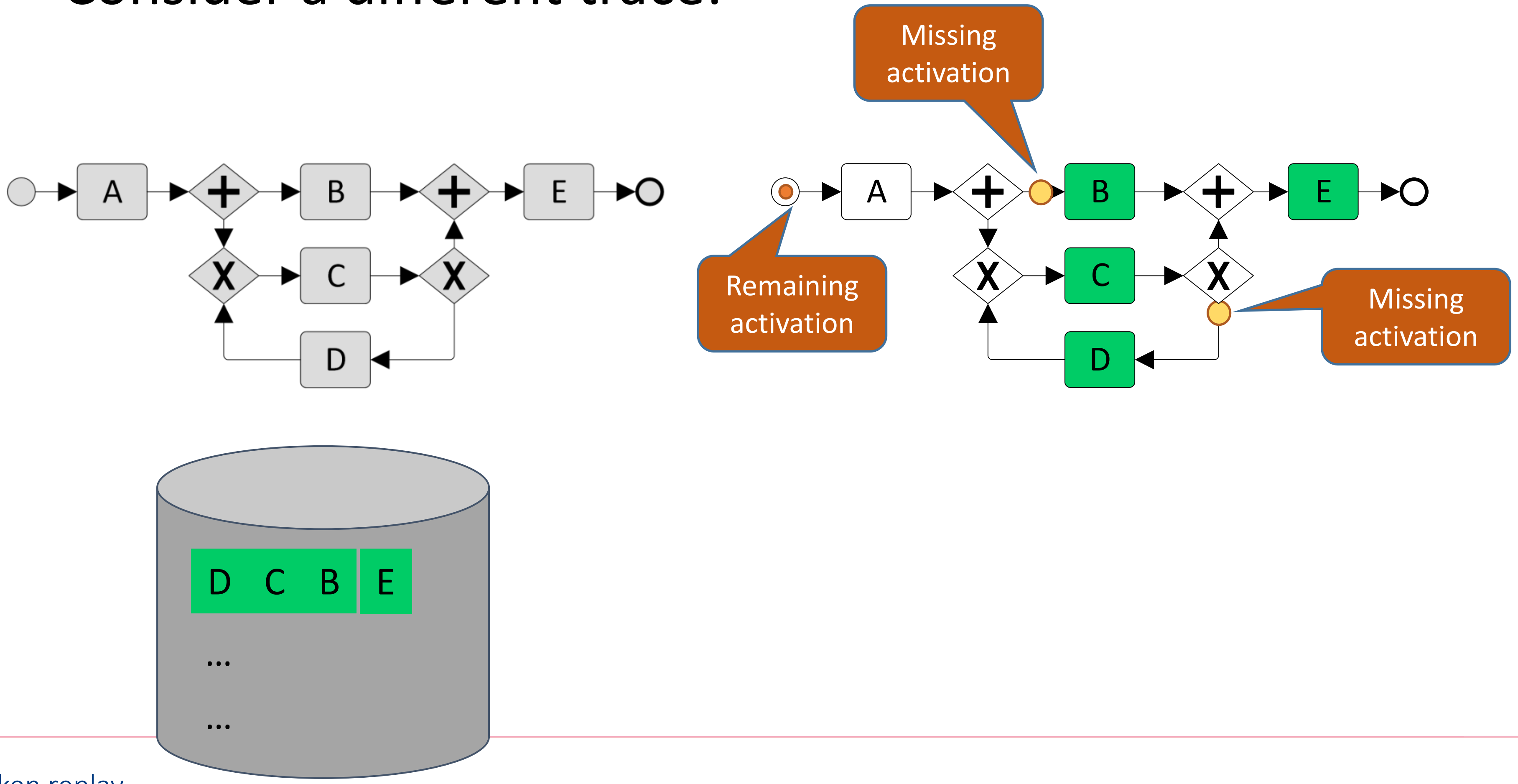






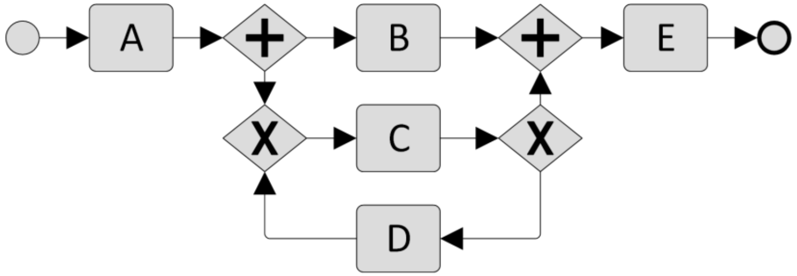
Detecting Deviations: Token replay

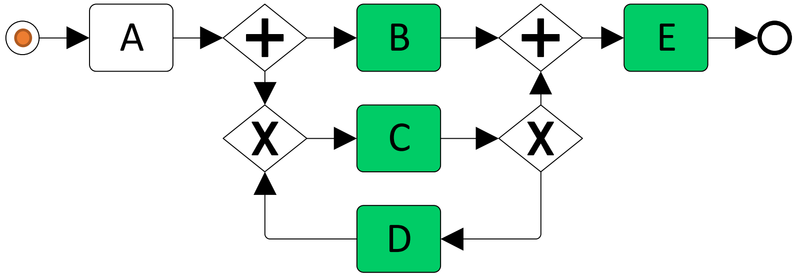
- Consider a different trace:

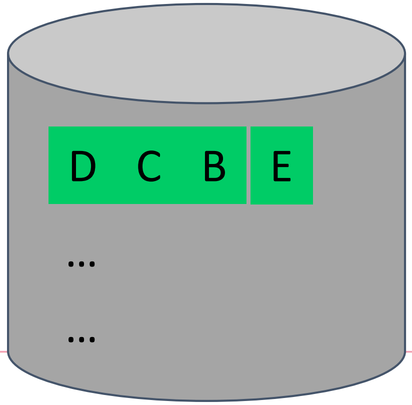


Detecting Deviations: Token replay

- Consider a different trace:



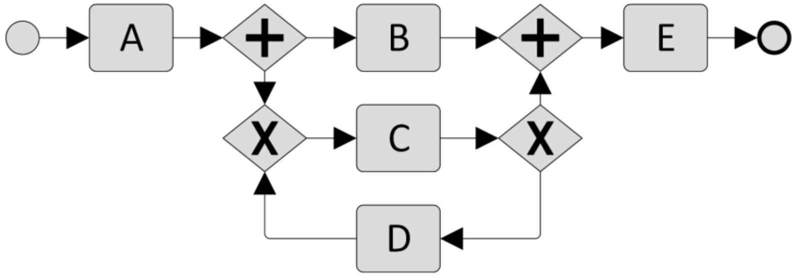


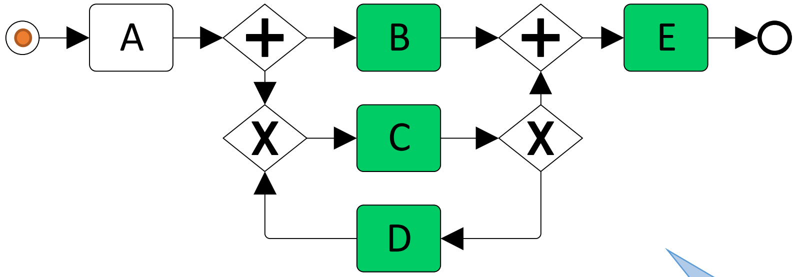


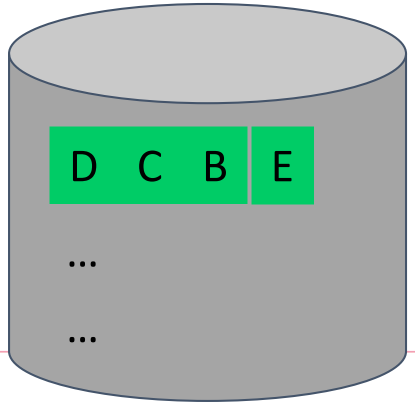
Produced activations: 5
Consumed activations: 6
Remaining activations: 1
Missing activations: 2

Detecting Deviations: Token replay

- Consider a different trace:







Produced activations: 5
Consumed activations: 6
Remaining activations: 1
Missing activations: 2

Fitness:

$$\tfrac{1}{2}\left(1 - {}^{m}/_{c}\right) + \tfrac{1}{2}\left(1 - {}^{r}/_{p}\right) = \tfrac{1}{2}\left(1 - {}^{2}/_{6}\right) + \tfrac{1}{2}\left(1 - {}^{1}/_{5}\right) = 0.73$$

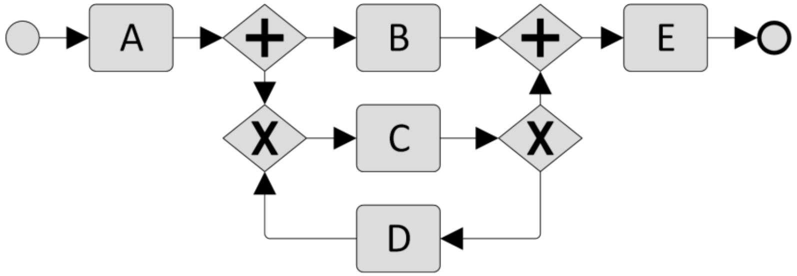
Token replay - problems

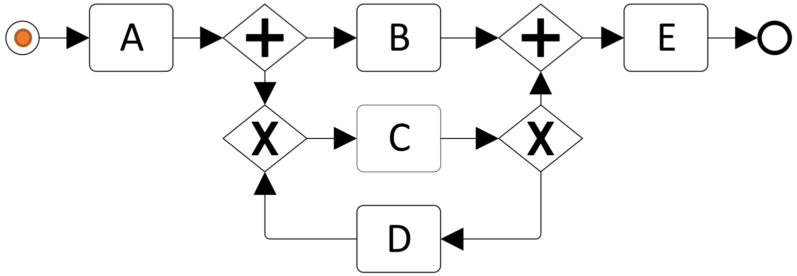
- Difficult to decide which labeled transition to execute:
  - Activities may appear on multiple transitions in the model
  - Routing of activations may not be clear
- Remaining activations from the beginning of the trace can be used in the end of the trace,
- Local diagnosis may hide global problems,

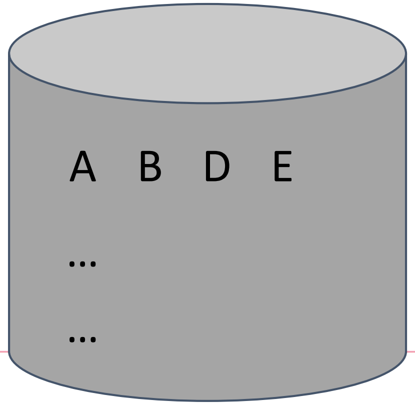
Trace	Fitness
<A,B,D,E>	0.67
<D,C,B,E>	0.73

Detecting Deviations: Alignments

- Find an execution of the model that is as close as possible to the observed trace

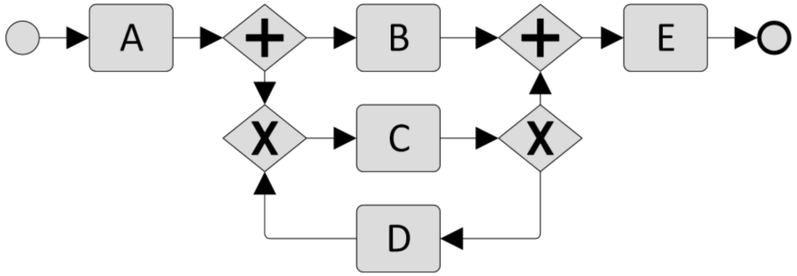


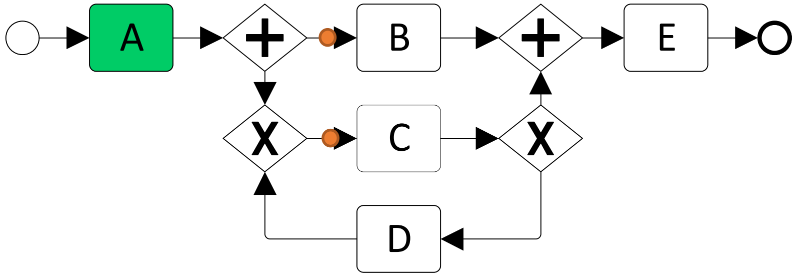


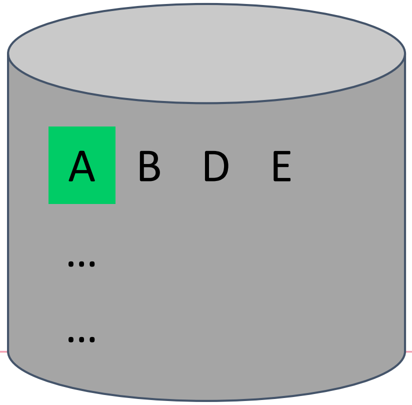


Detecting Deviations: Alignments

- Find an execution of the model that is as close as possible to the observed trace

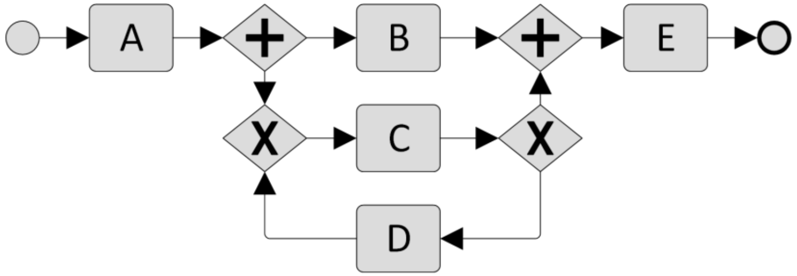


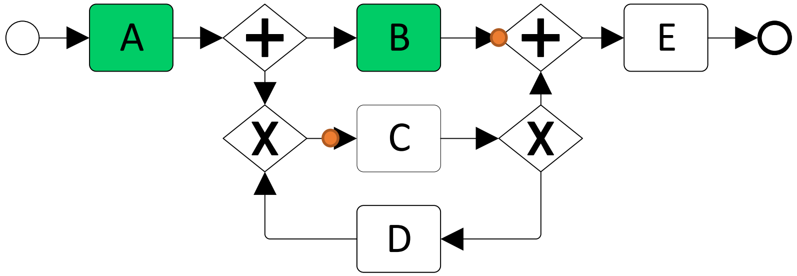


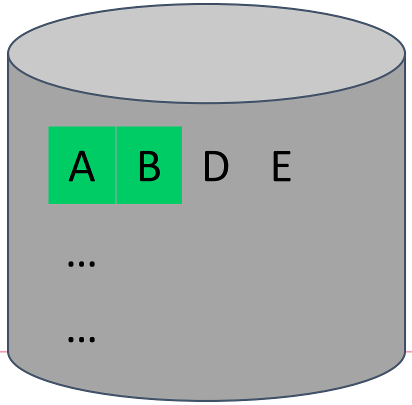


Detecting Deviations: Alignments

- Find an execution of the model that is as close as possible to the observed trace

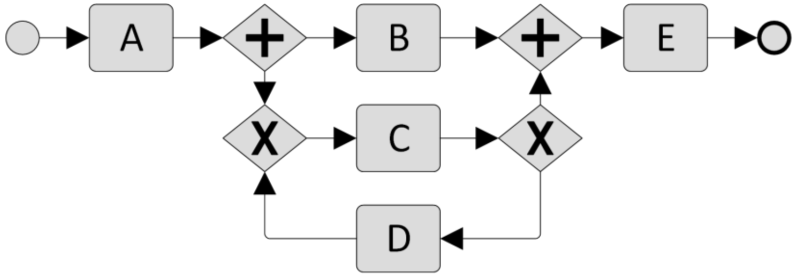


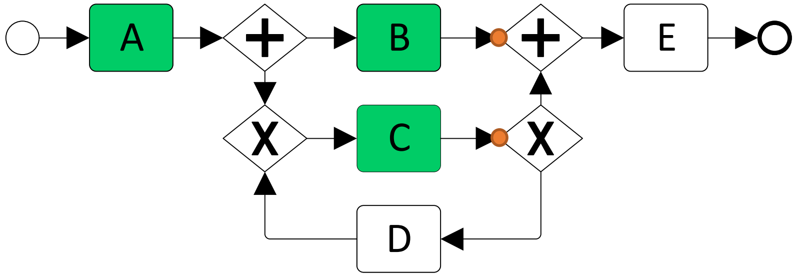


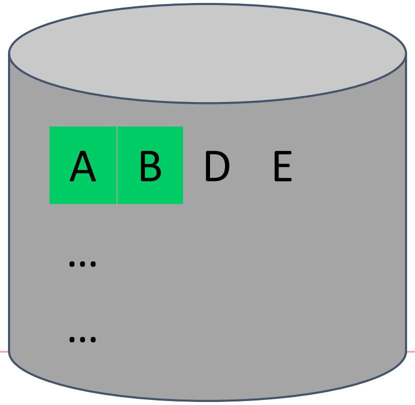


Detecting Deviations: Alignments

- Find an execution of the model that is as close as possible to the observed trace

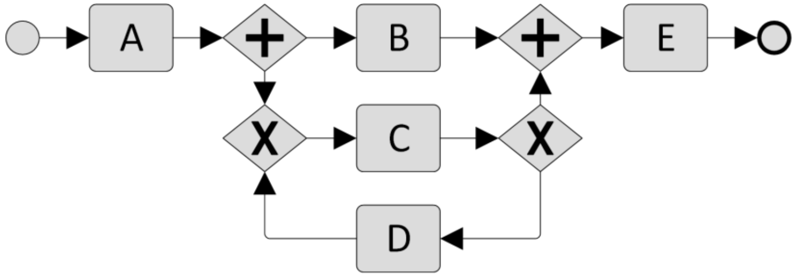


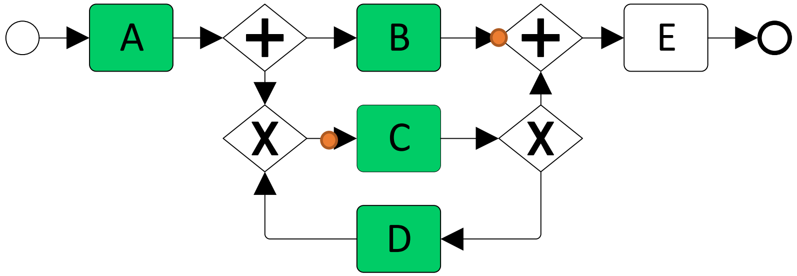


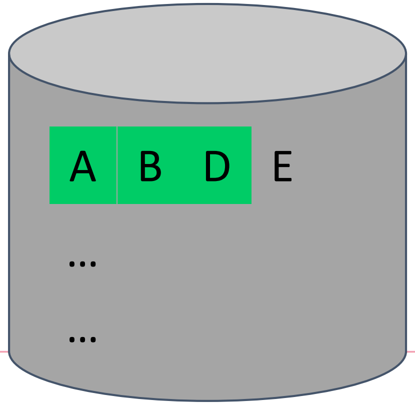


Detecting Deviations: Alignments

- Find an execution of the model that is as close as possible to the observed trace

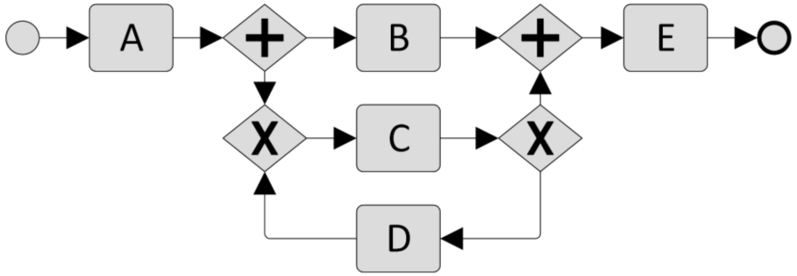


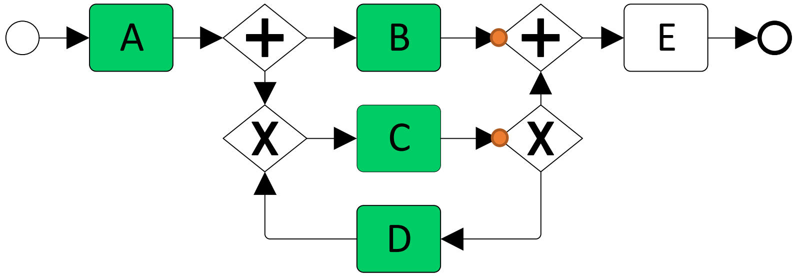


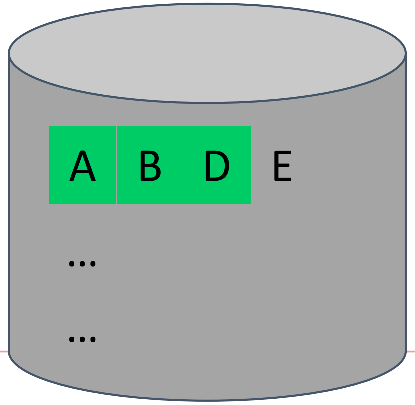


Detecting Deviations: Alignments

- Find an execution of the model that is as close as possible to the observed trace

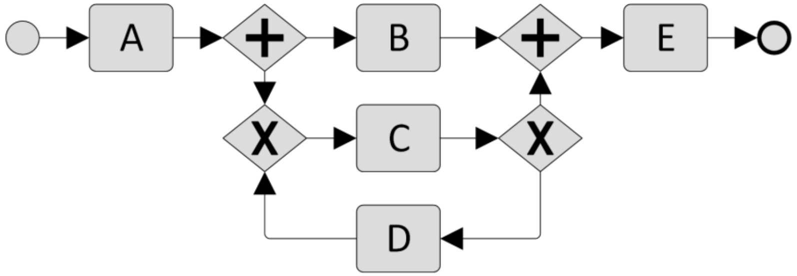


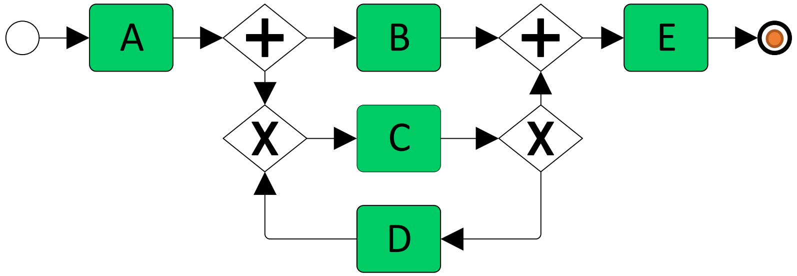


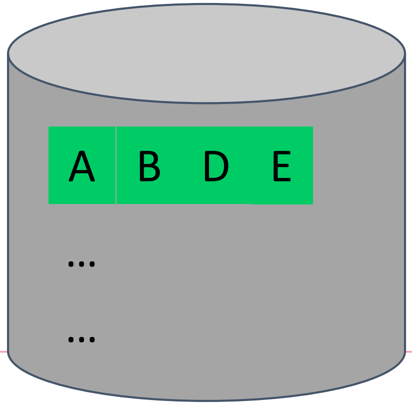


Detecting Deviations: Alignments

- Find an execution of the model that is as close as possible to the observed trace

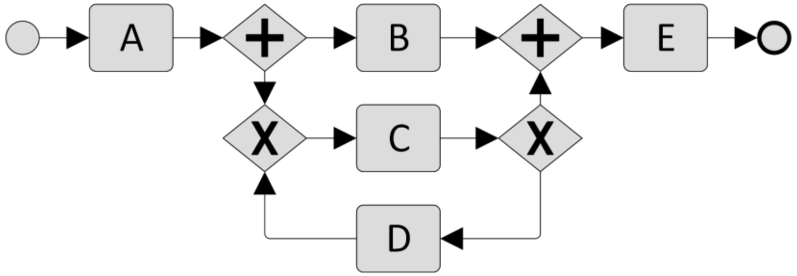


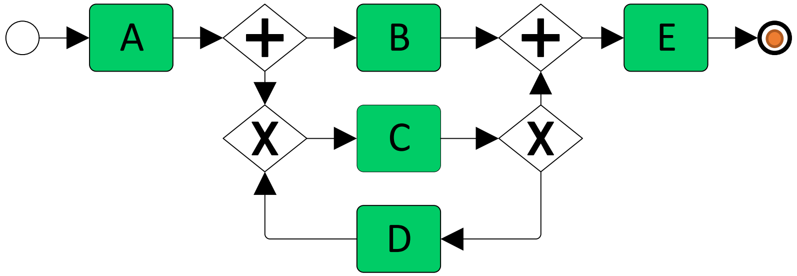


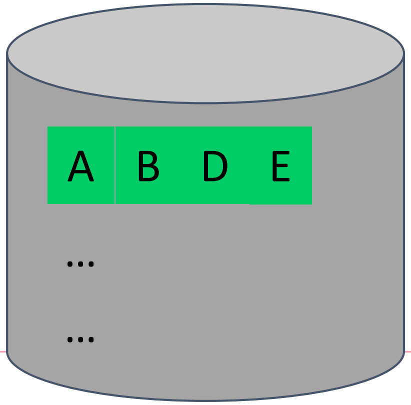


Detecting Deviations: Alignments

- Find an execution of the model that is as close as possible to the observed trace



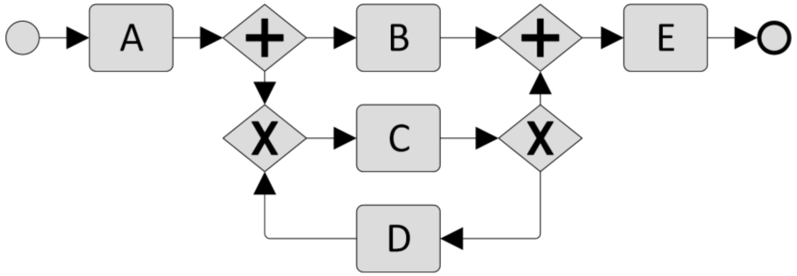


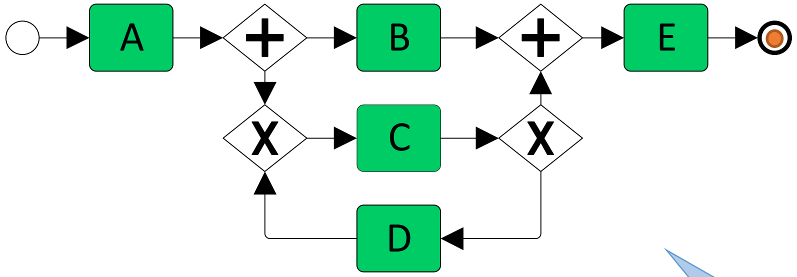


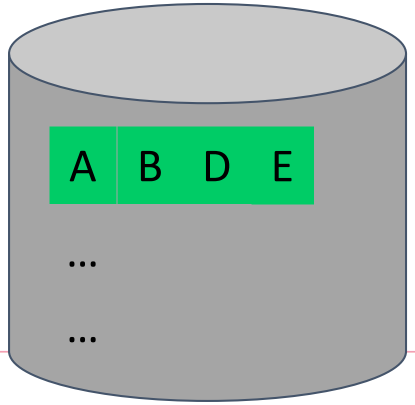
Synchronous activities: 4
Model only: 2
Log only: 0

Detecting Deviations: Alignments

- Find an execution of the model that is as close as possible to the observed trace







Synchronous activities: 4
Model only: 2
Log only: 0

Fitness:

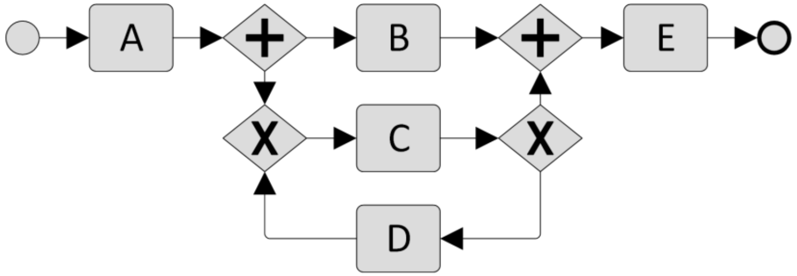
$$1 - {}^{(l+m)}/_{(\text{length trace + min model})} =$$

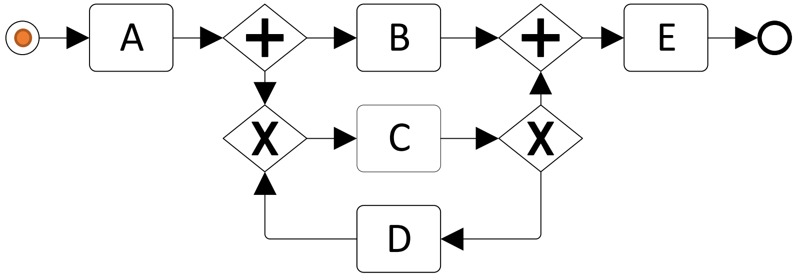
$$1 - {}^{2}/_{(4+4)} =$$

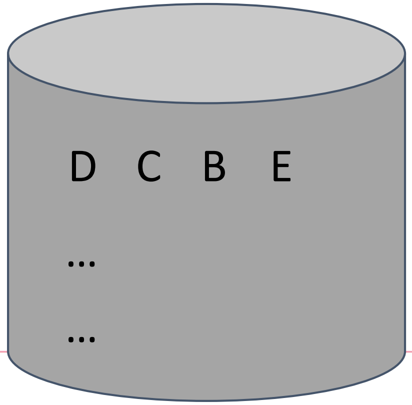
$$0.75$$

Detecting Deviations : Alignments

- Find an execution of the model that is as close as possible to the observed trace

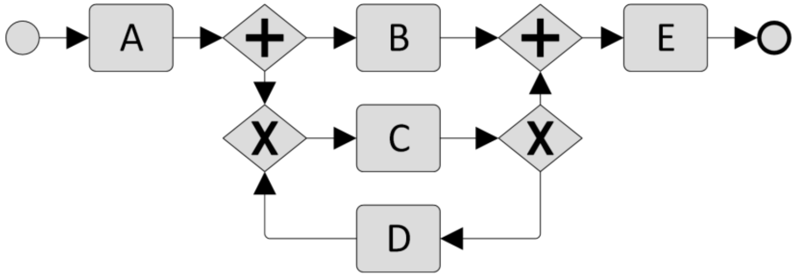


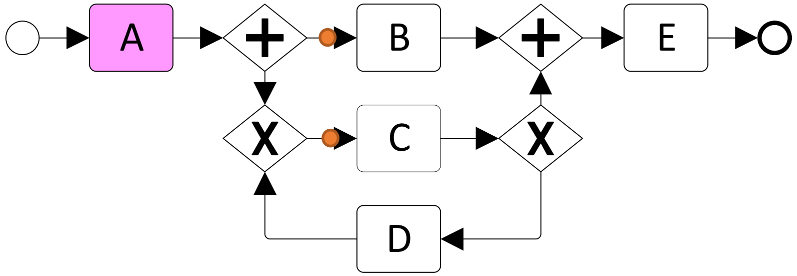


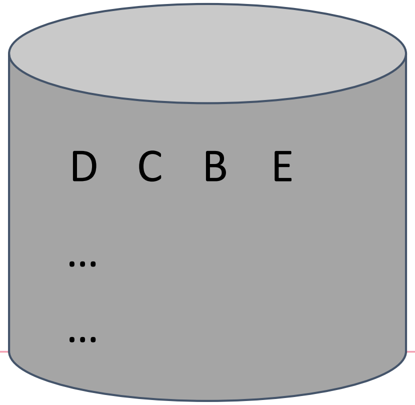


Detecting Deviations : Alignments

- Find an execution of the model that is as close as possible to the observed trace

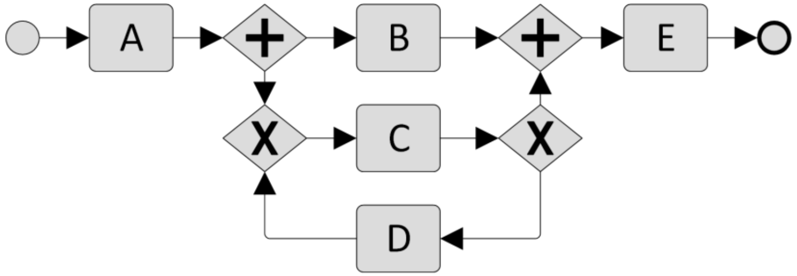


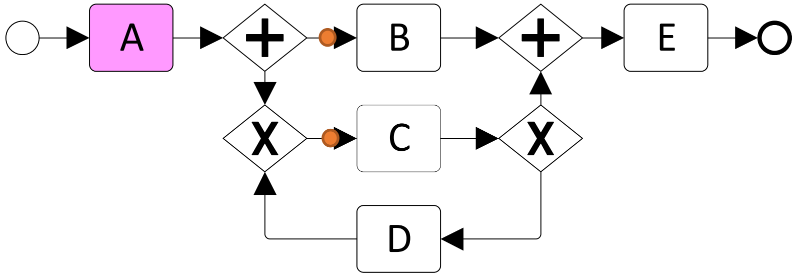


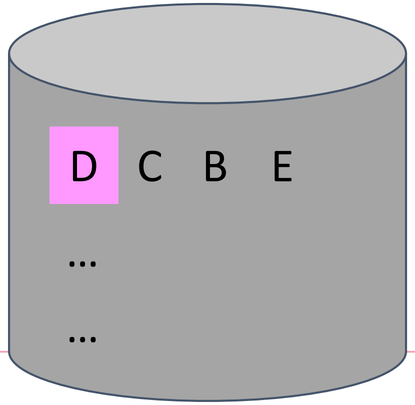


Detecting Deviations : Alignments

- Find an execution of the model that is as close as possible to the observed trace

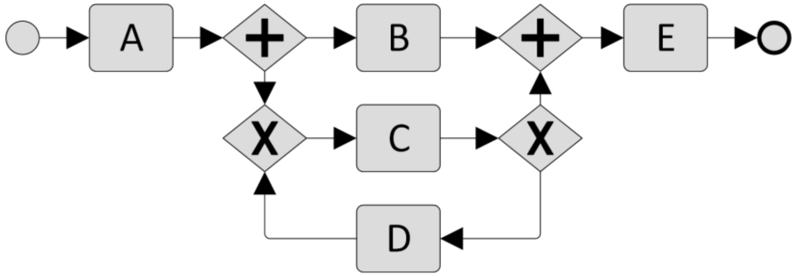


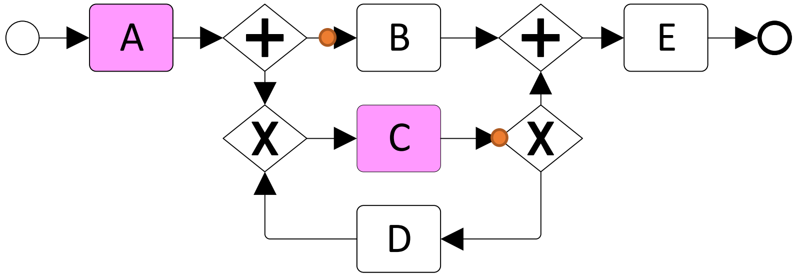


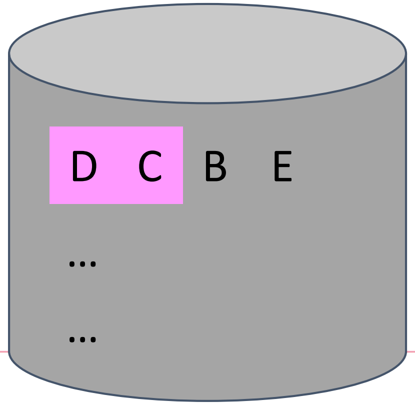


Detecting Deviations : Alignments

- Find an execution of the model that is as close as possible to the observed trace

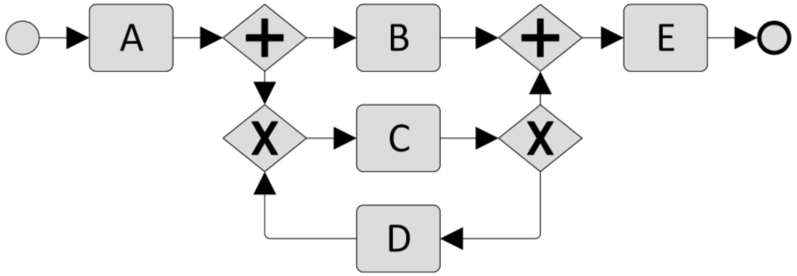


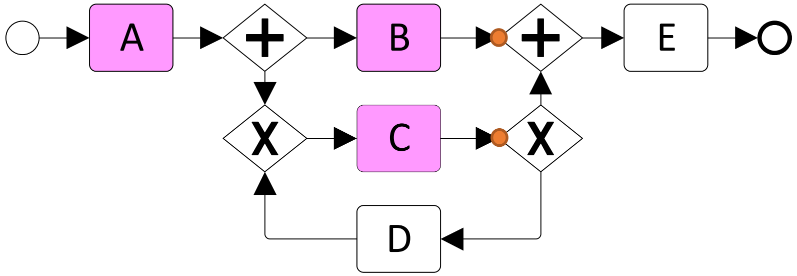


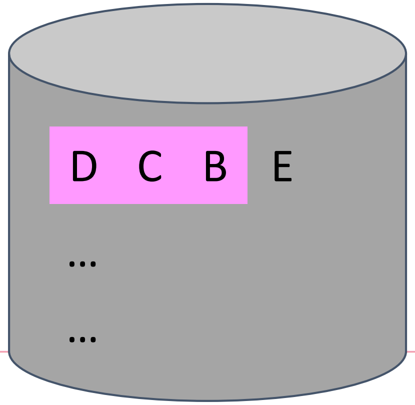


Detecting Deviations : Alignments

- Find an execution of the model that is as close as possible to the observed trace

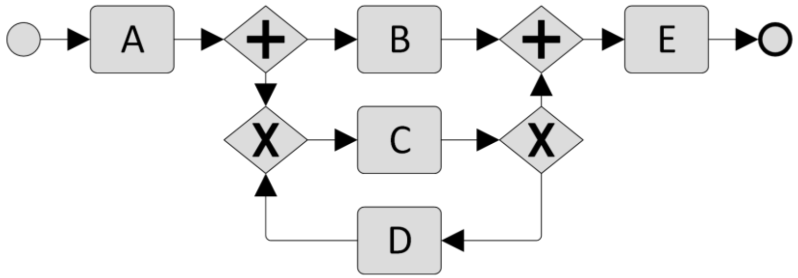


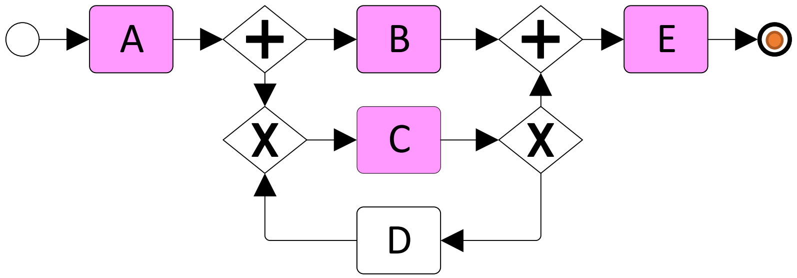


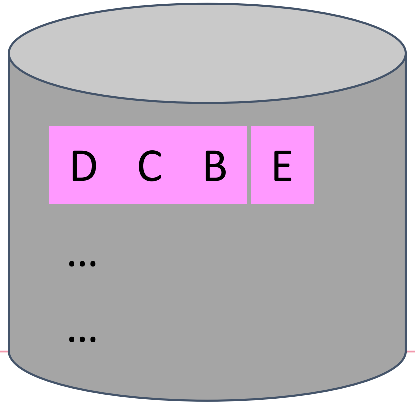


Detecting Deviations : Alignments

- Find an execution of the model that is as close as possible to the observed trace

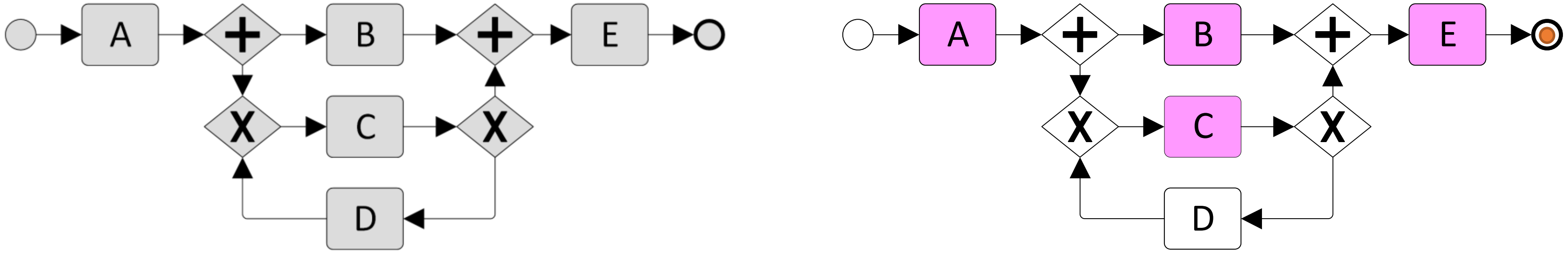


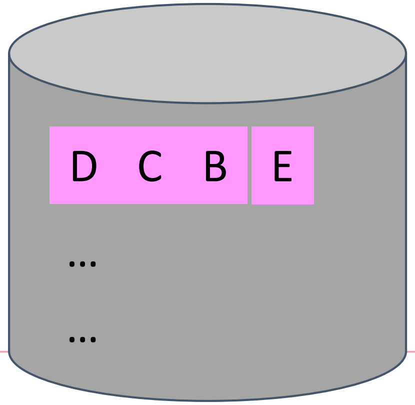


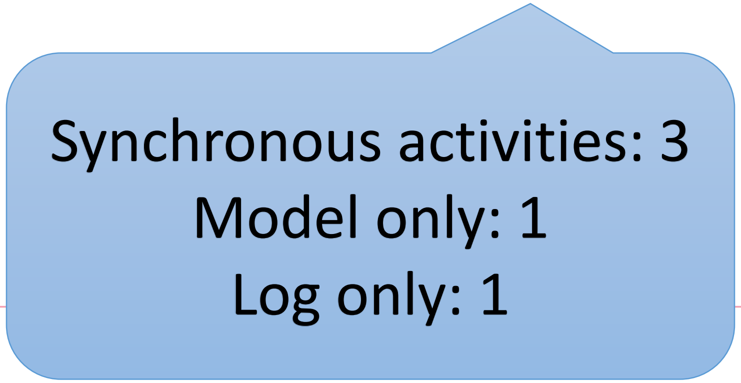


Detecting Deviations : Alignments

- Find an execution of the model that is as close as possible to the observed trace

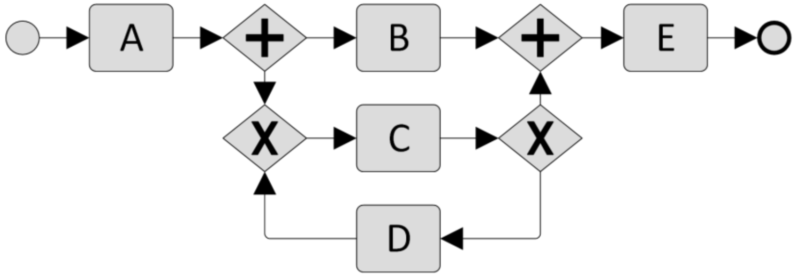


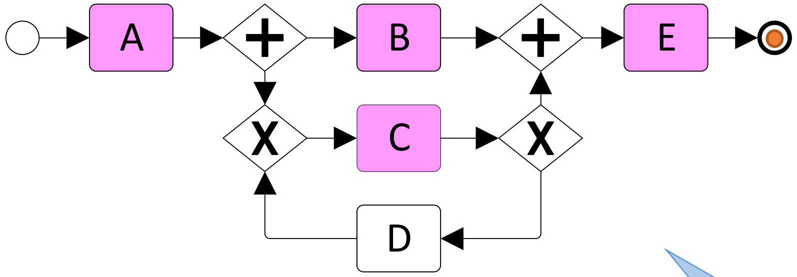


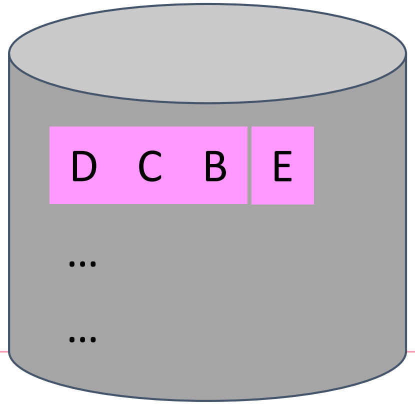


Detecting Deviations : Alignments

- Find an execution of the model that is as close as possible to the observed trace







Synchronous activities: 3
Model only: 1
Log only: 1

Fitness:

$1 - {}^{(l+m)}/_{(\text{length trace + min model})} =$

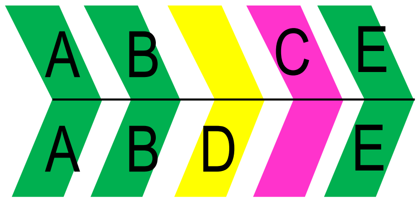
$1 - {}^{2}/_{(4+4)} =$

0.75

Alignments

An alignment contains the most likely execution of the model, corresponding to an observed execution

- Alignments explain where deviations occur and which deviations occur

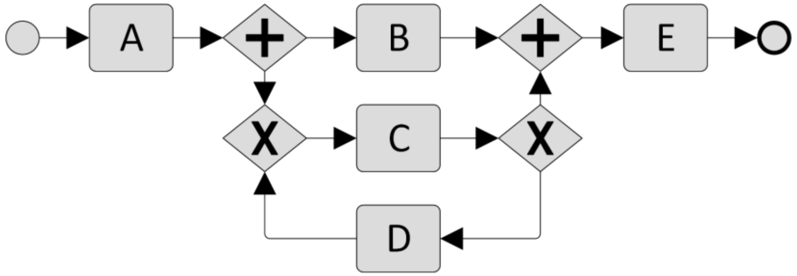


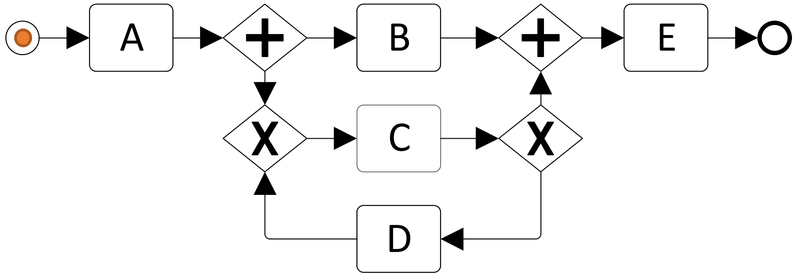
- Alignments:
  a) Are globally optimal
  b) Are robust to label duplication
  c) Are robust to routing transitions
  d) Provide a true execution of the model
  e) Can handle partially ordered traces

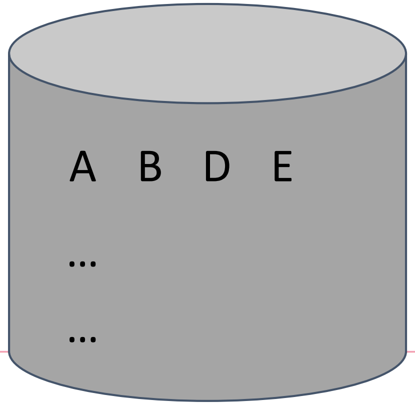
An alignment shows where deviations occurred and why these deviation are considered as such

Alignments are not unique

- Find an execution of the model that is as close as possible to the observed trace

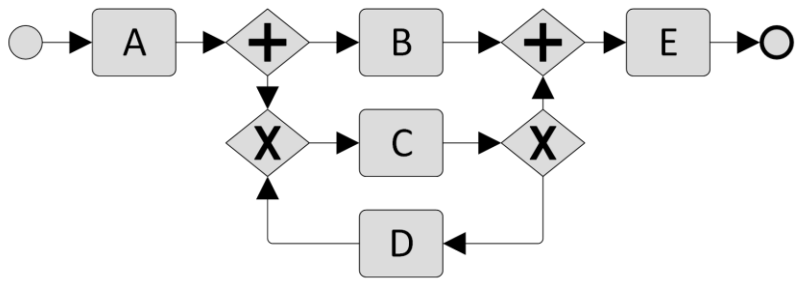


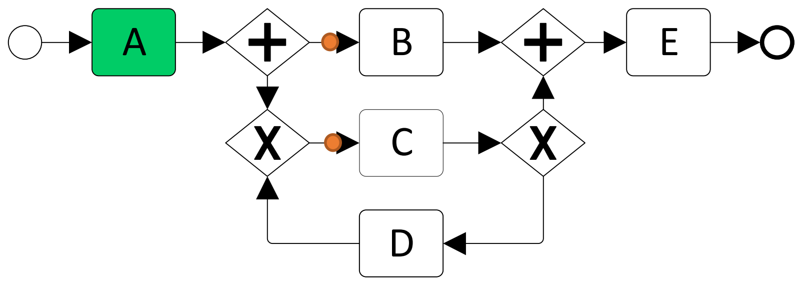


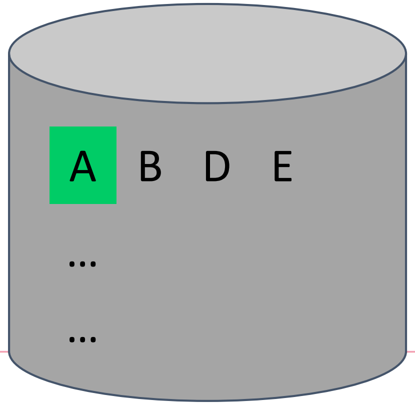


Alignments are not unique

- Find an execution of the model that is as close as possible to the observed trace

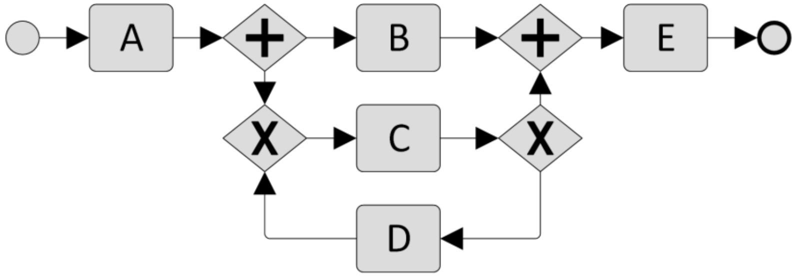


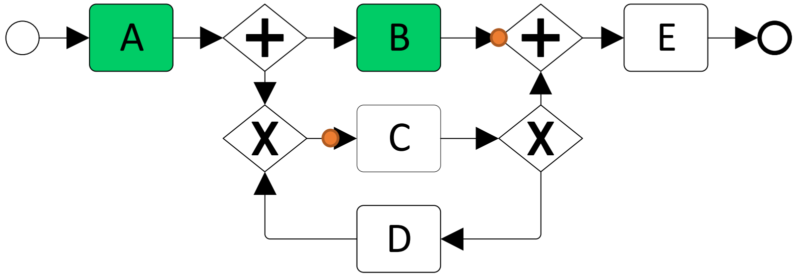


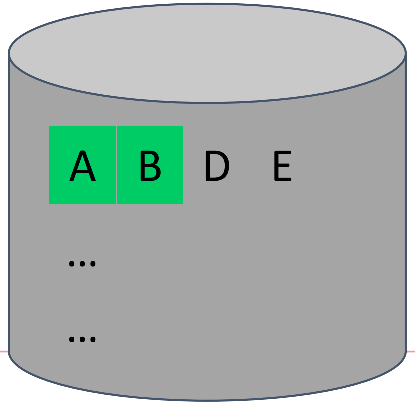


Alignments are not unique

- Find an execution of the model that is as close as possible to the observed trace

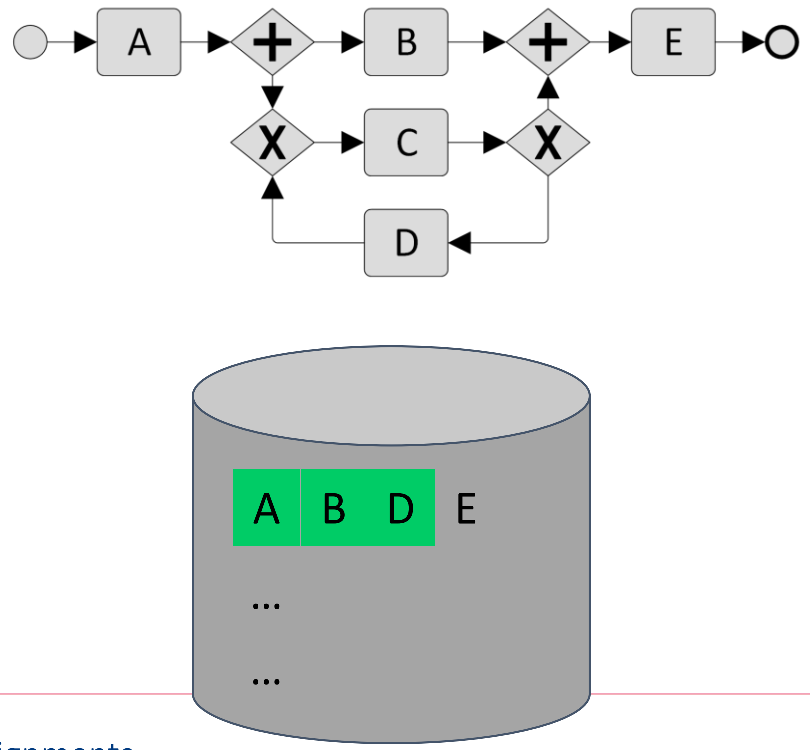


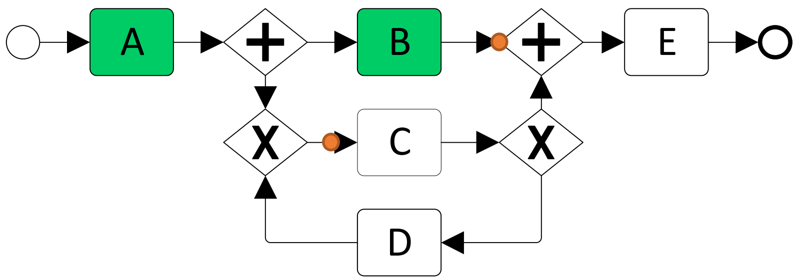




Alignments are not unique

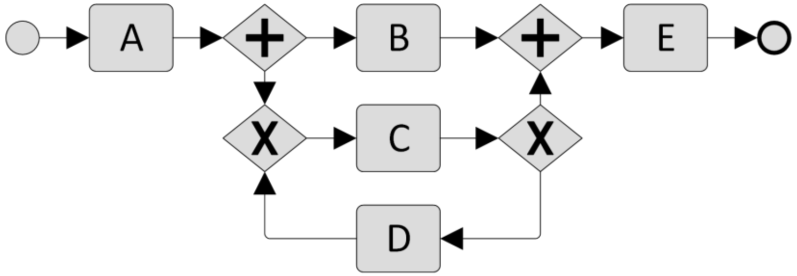
- Find an execution of the model that is as close as possible to the observed trace

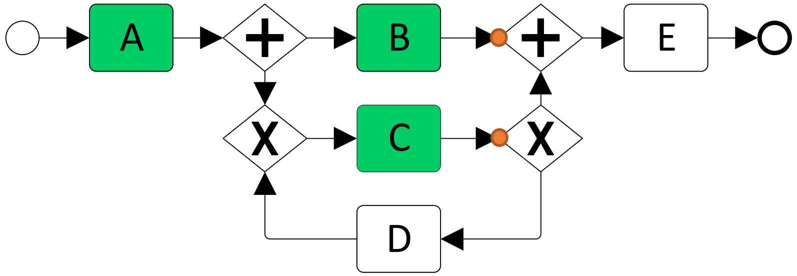


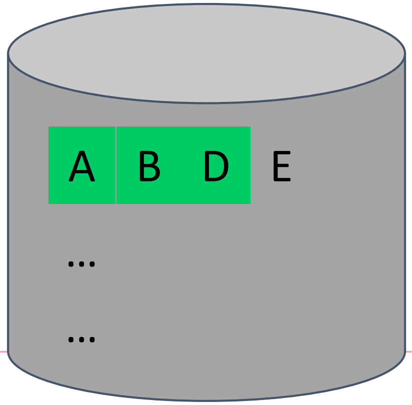


Alignments are not unique

- Find an execution of the model that is as close as possible to the observed trace

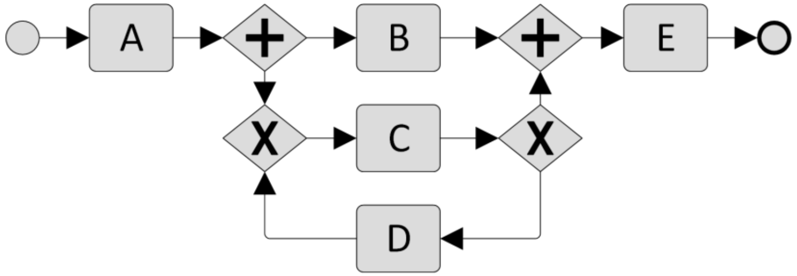


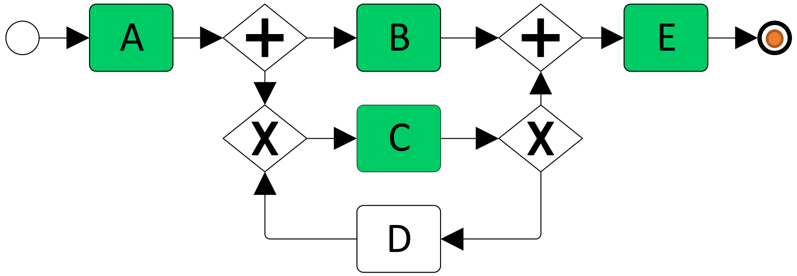


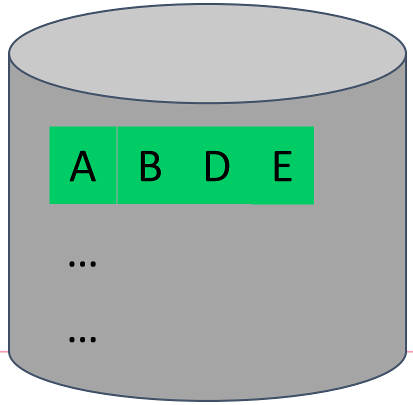


Alignments are not unique

- Find an execution of the model that is as close as possible to the observed trace

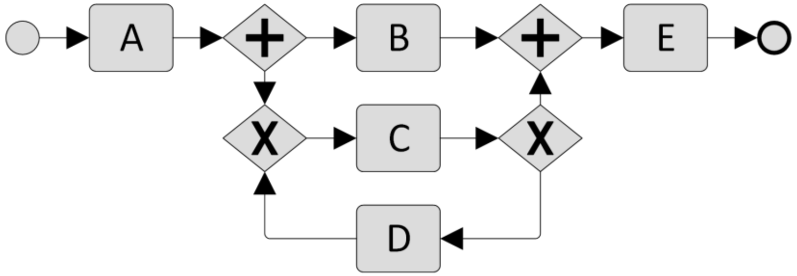


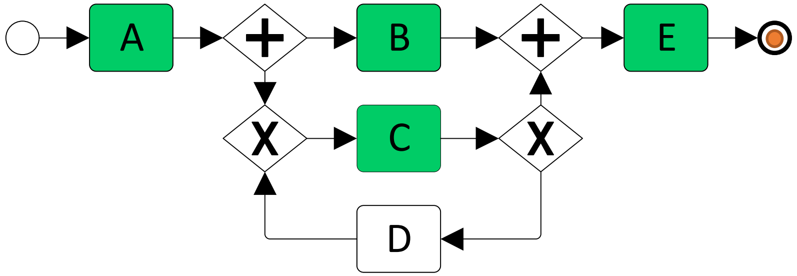


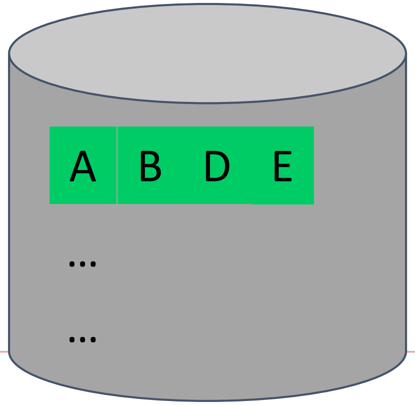


Alignments are not unique

- Find an execution of the model that is as close as possible to the observed trace



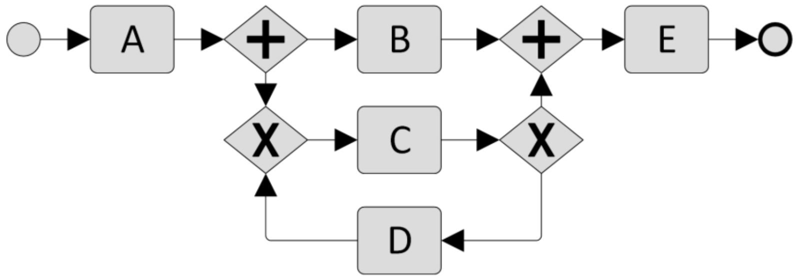


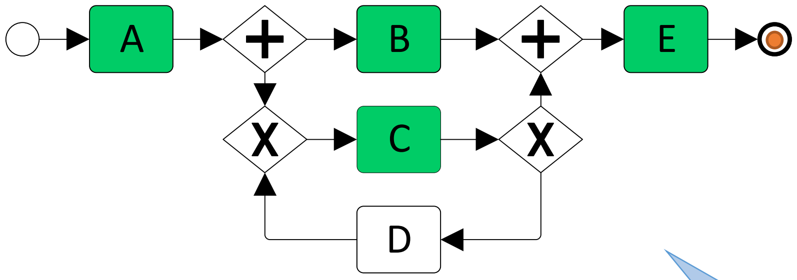


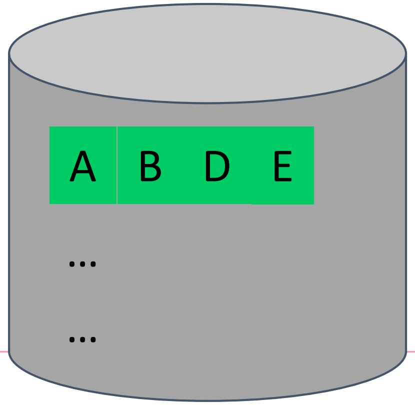
Synchronous activities: 3
Model only: 1
Log only: 1

Alignments are not unique

- Find an execution of the model that is as close as possible to the observed trace







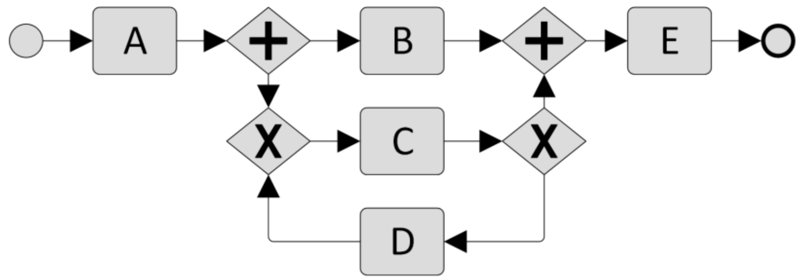
Synchronous activities: 3
Model only: 1
Log only: 1

Fitness:

$$1 - {}^{(l+m)}/_{(\text{length trace + min model})} =$$
$$1 - {}^{2}/_{(4+4)} =$$
$$0.75$$

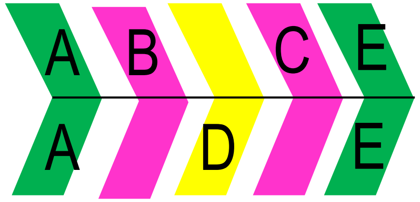
Alignments are not unique

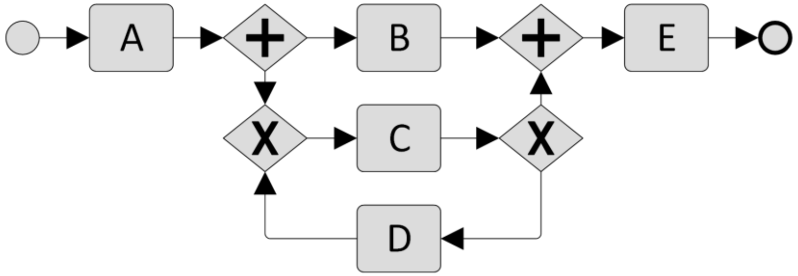
Consider the trace <A,D,E>



Alignments are not unique

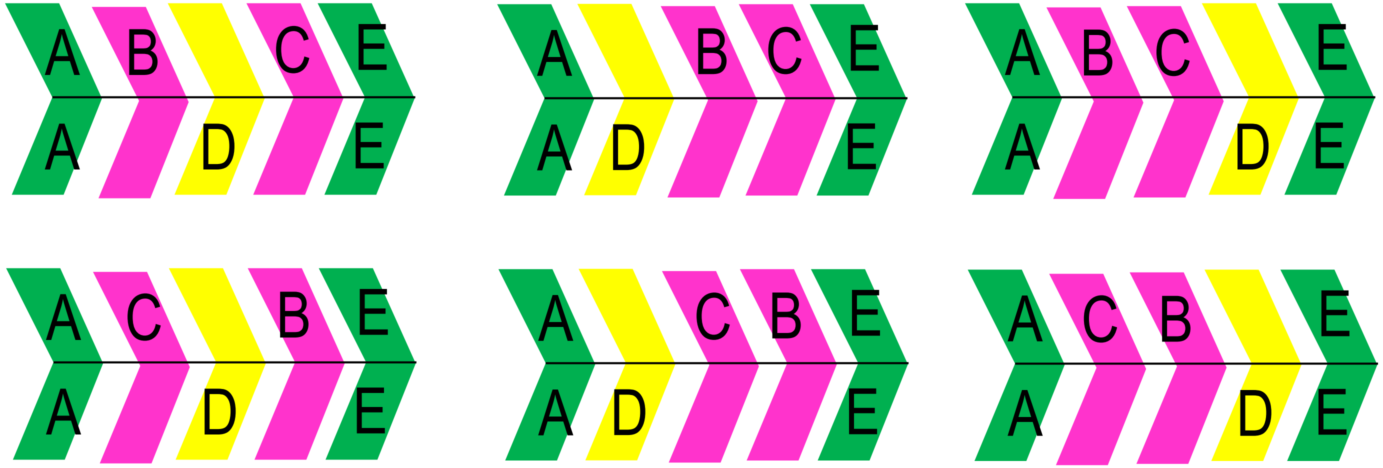
Consider the trace <A,D,E>

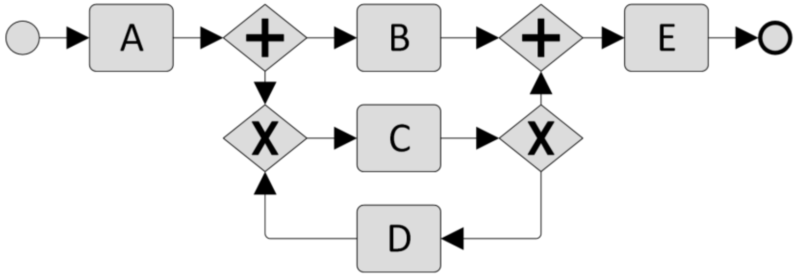




Alignments are not unique

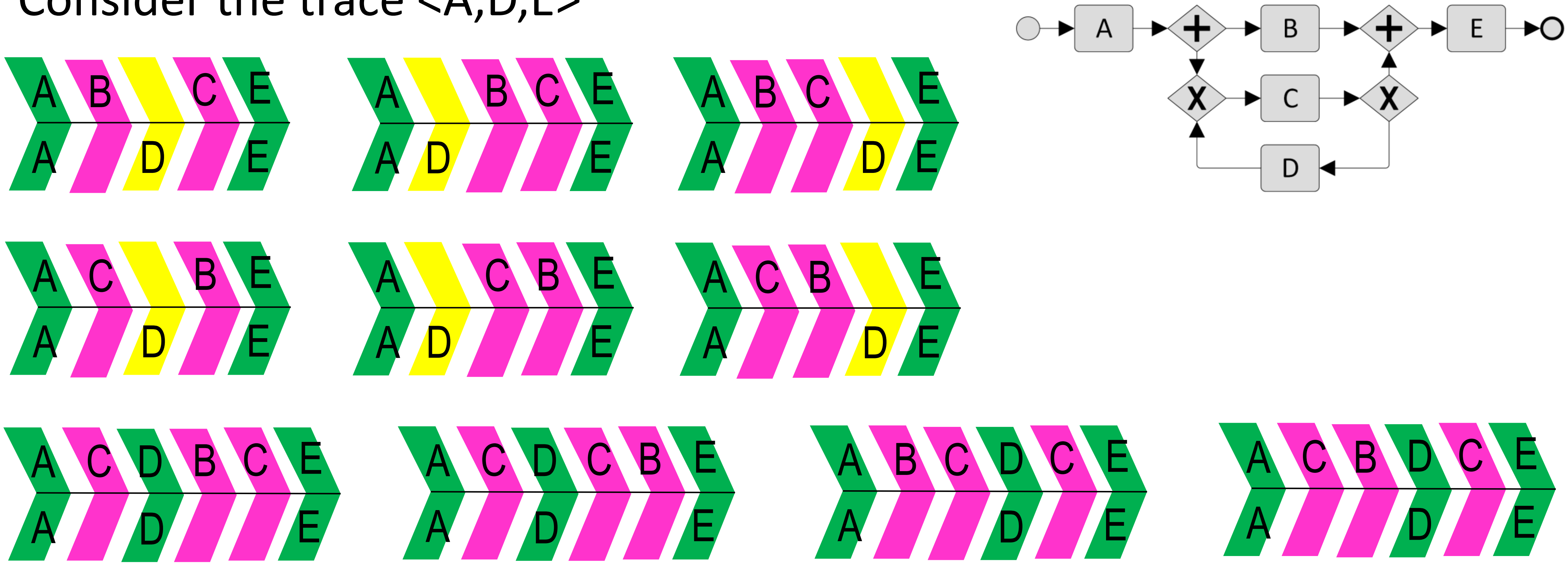
Consider the trace <A,D,E>





Alignments are not unique

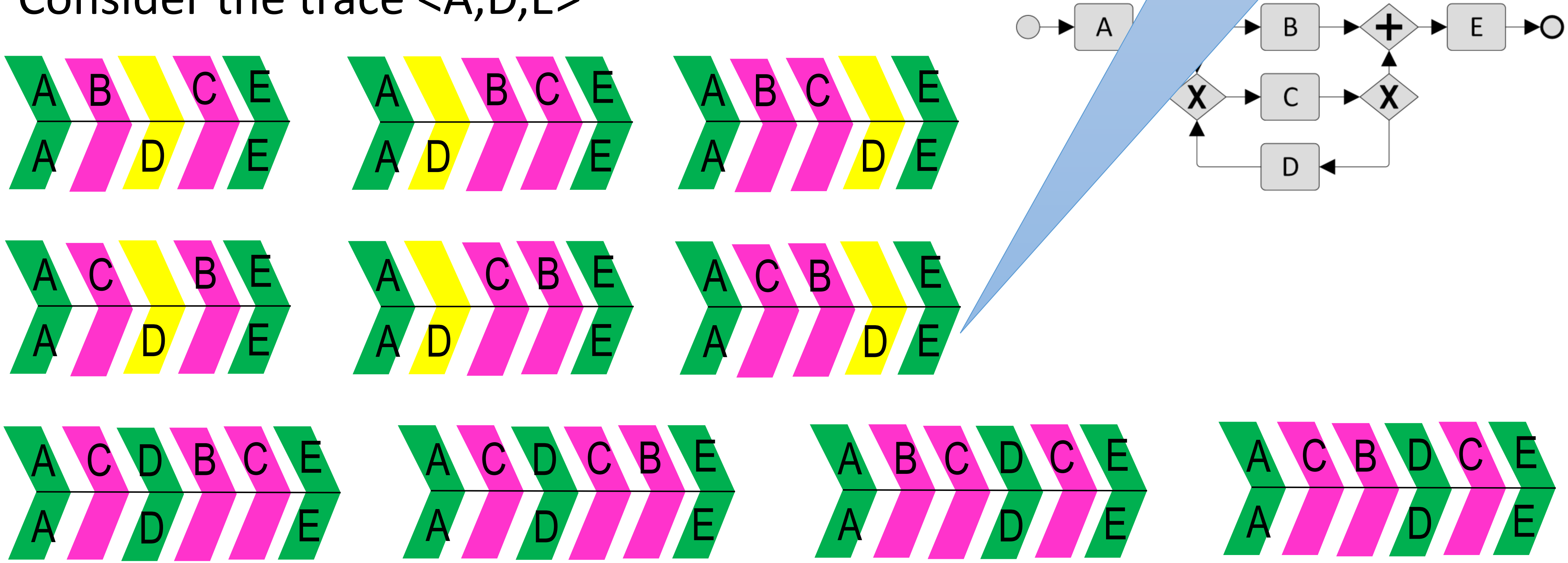
Consider the trace <A,D,E>

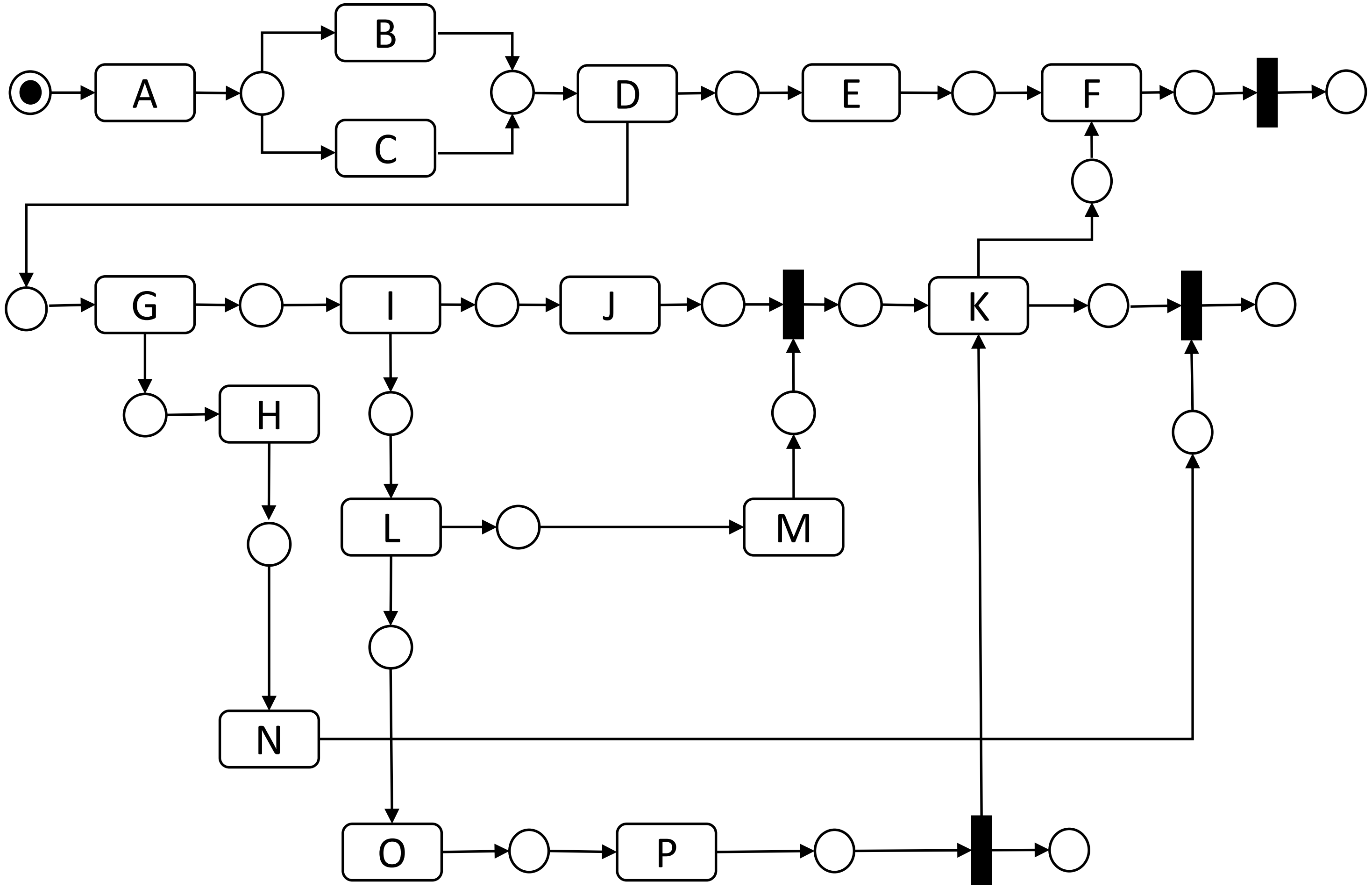


Alignments are not unique

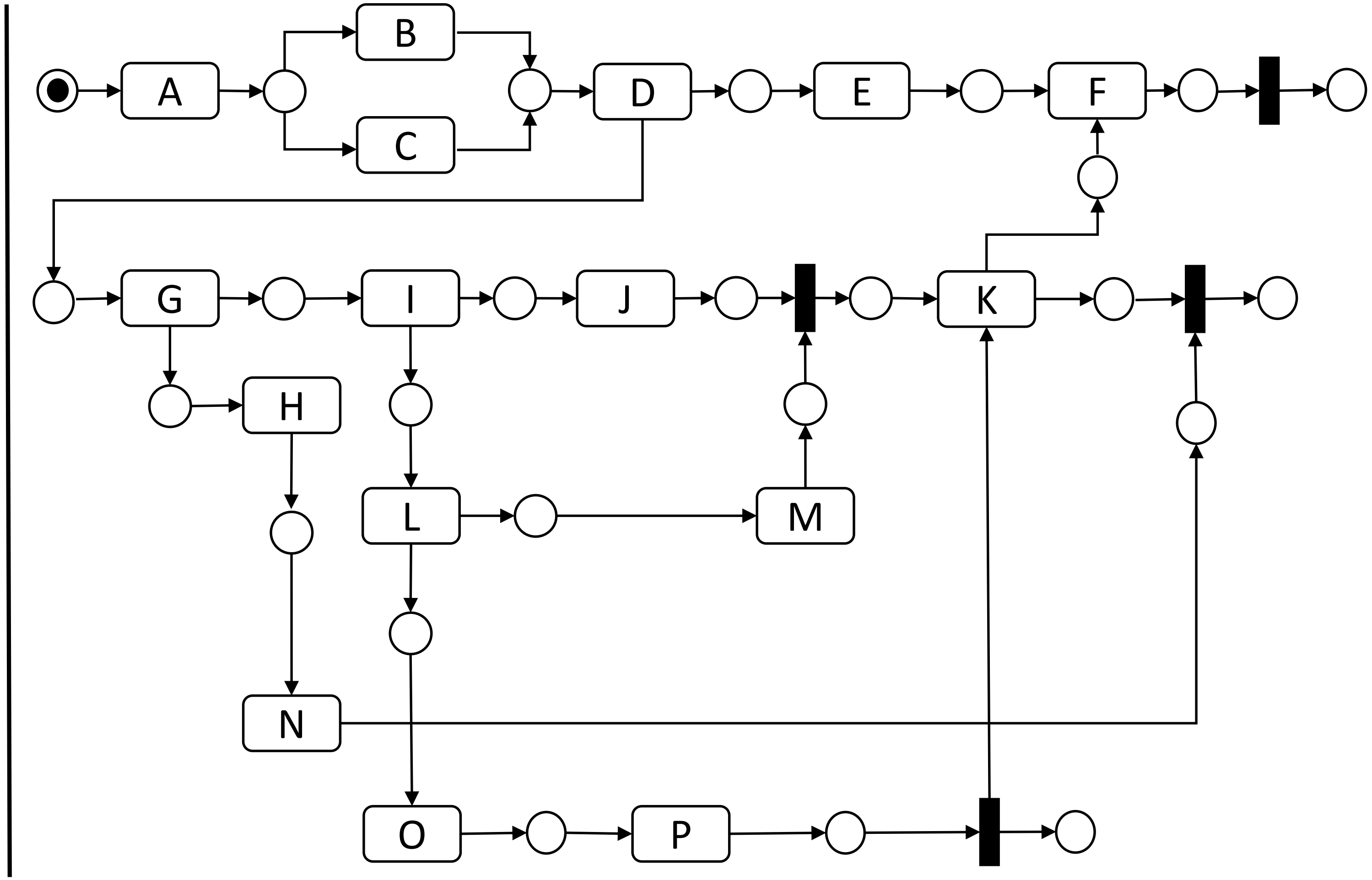
These alignments are all optimal, yet they explain deviations differently

Consider the trace <A,D,E>

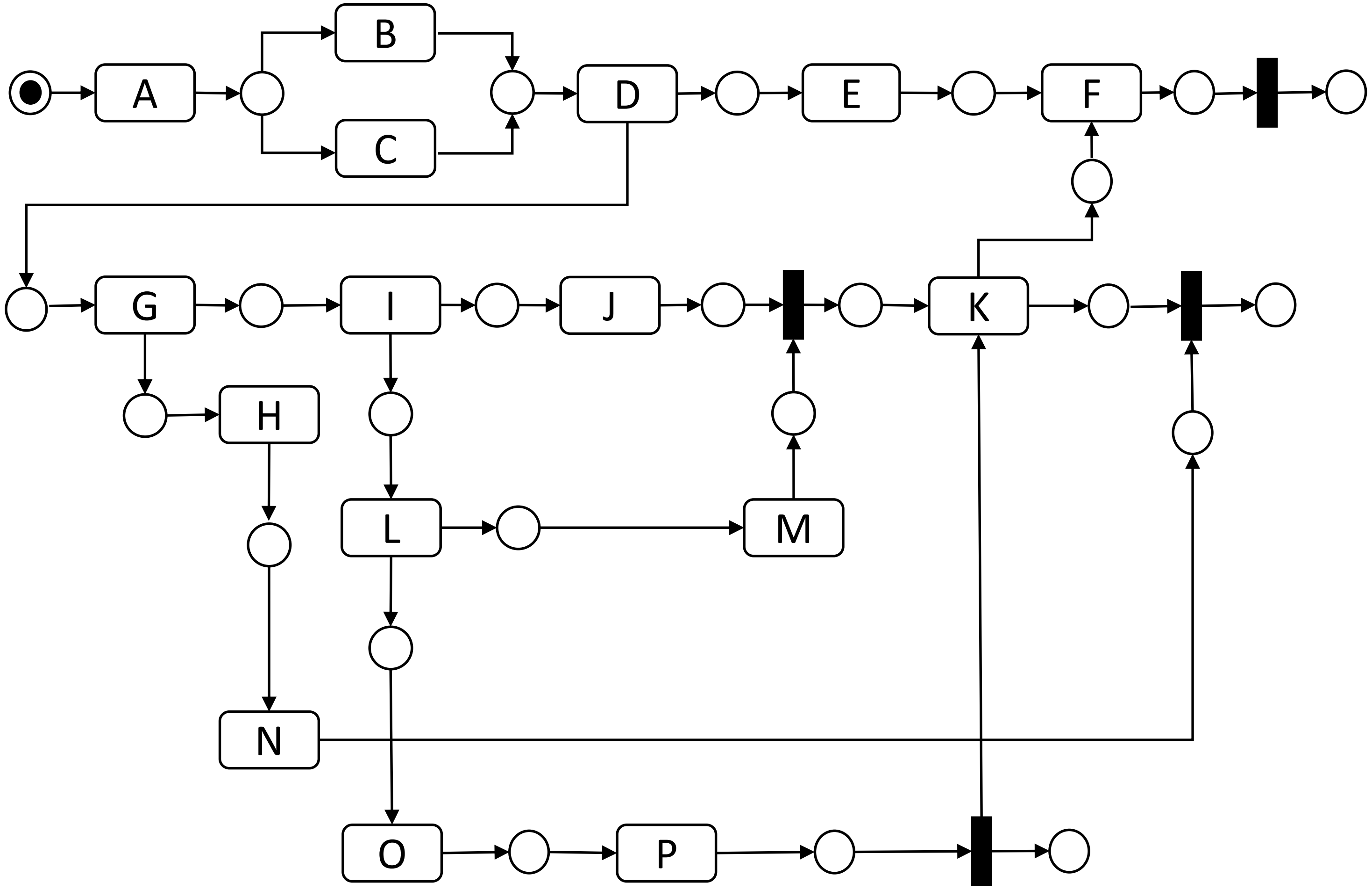



A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

Model for the Reality



Model for the Reality

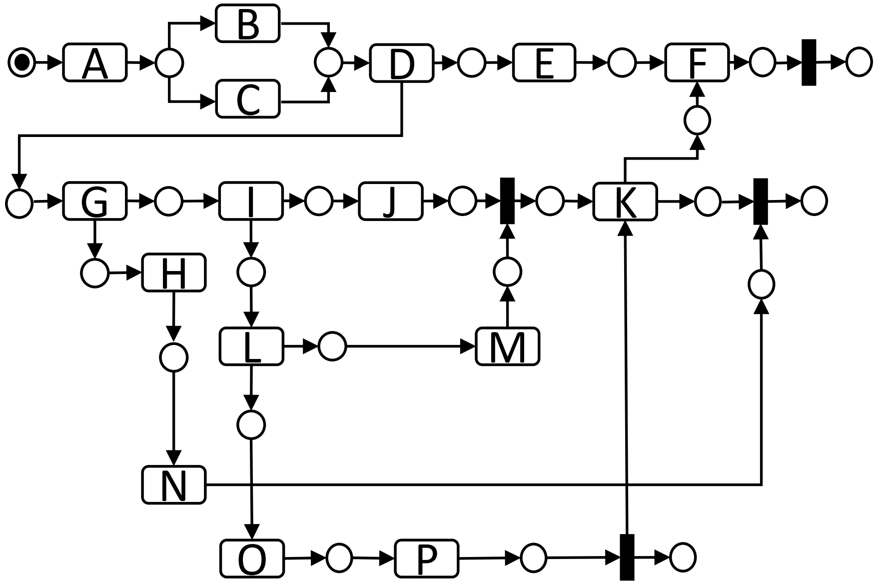


The Reality

<A,D,B,G,E,N,H,I,L,J,M,P,O,K>

THE ALIGNMENT BETWEEN MODELS AND TRACES

- GIVEN:



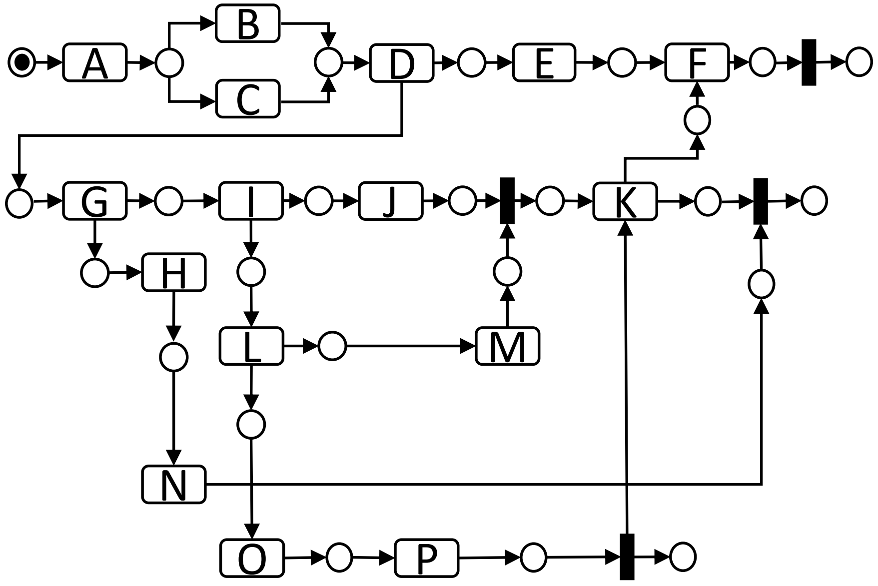
<A,D,B,G,E,N,H,I,L,J,M,P,O,K>

- COMPUTE:

Trace	A	>	D	B	G	E	>	N	H	I	L	J	M	>	>	P	O	>	K	>	>	>
Model	A	B	D	>	G	E	H	N	>	I	L	J	M	τ	O	P	>	τ	K	F	τ	τ

THE ALIGNMENT BETWEEN MODELS AND TRACES

- GIVEN:



<A,D,B,G,E,N,H,I,L,J,M,P,O,K>

- COMPUTE:

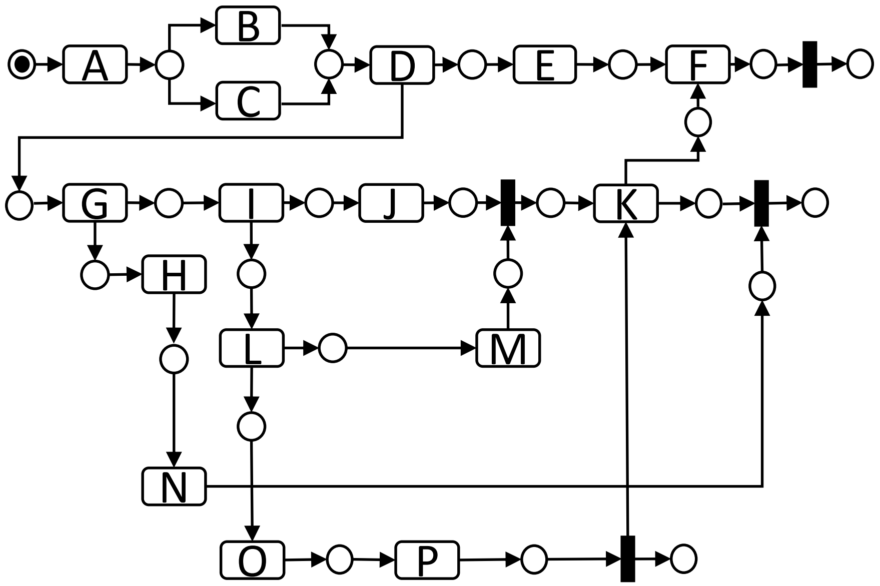
Synchronous Moves

Trace	A	>	D	B	G	E	>	N	H	I	L	J	M	>	>	P	O	>	K	>	>	>
Model	A	B	D	>	G	E	H	N	>	I	L	J	M	τ	O	P	>	τ	K	F	τ	τ

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Campus de Excelencia Internacional

THE ALIGNMENT BETWEEN MODELS AND TRACES

- GIVEN:



<A,D,B,G,E,N,H,I,L,J,M,P,O,K>

- COMPUTE:

Model Moves

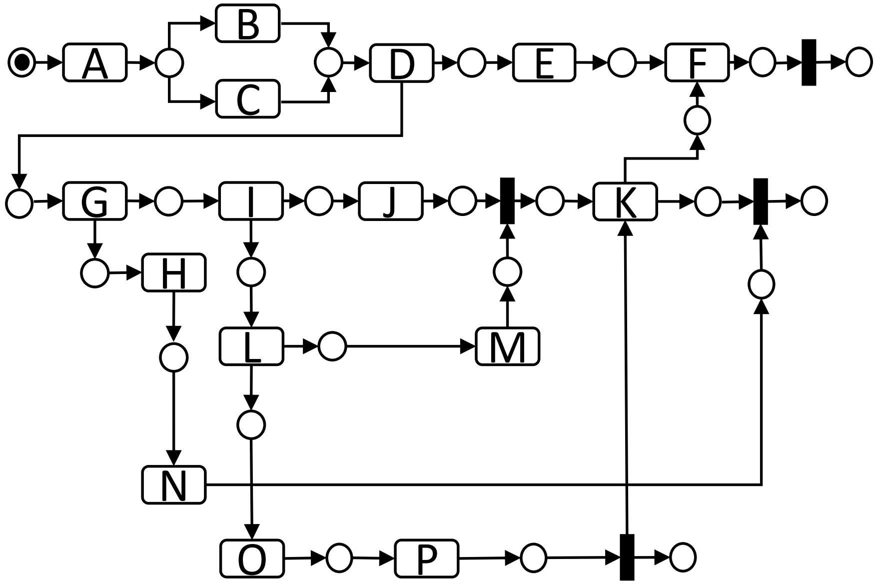
Trace	A	>	D	B	G	E	>	N	H	I	L	J	M	>	>	P	O	>	K	>	>	>
Model	A	B	D	>	G	E	H	N	>	I	L	J	M	τ	O	P	>	τ	K	F	τ	τ

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Campus de Excelencia Internacional

THE ALIGNMENT BETWEEN MODELS AND TRACES

- GIVEN:



<A,D,B,G,E,N,H,I,L,J,M,P,O,K>

- COMPUTE:

Log Moves

Trace	A	>	D	B	G	E	>	N	H	I	L	J	M	>	>	P	O	>	K	>	>	>
Model	A	B	D	>	G	E	H	N	>	I	L	J	M	τ	O	P	>	τ	K	F	τ	τ

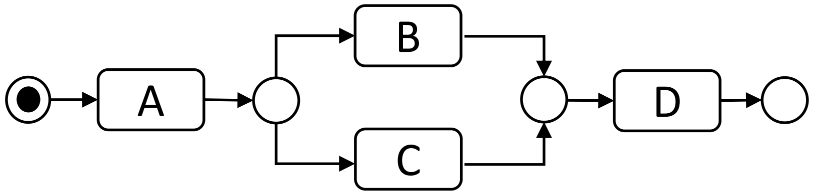
ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

- ARISING FROM **A. ADRIANSYAH PhD. (TU/e)**
- COSTS CAN BE ASSIGNED TO SYNCHRONOUS/ASYNCHRONOUS MOVES
- CURRENT IMPLEMENTATION BASED ON THE NOTION OF **SYNCHRONOUS PRODUCT NET**:

ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

- ARISING FROM **A. ADRIANSYAH PhD. (TU/e)**
- COSTS CAN BE ASSIGNED TO SYNCHRONOUS/ASYNCHRONOUS MOVES
- CURRENT IMPLEMENTATION BASED ON THE NOTION OF **SYNCHRONOUS PRODUCT NET**:

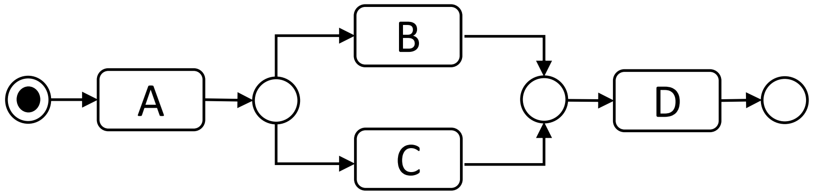
Process Model



ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

- ARISING FROM **A. ADRIANSYAH PhD. (TU/e)**
- COSTS CAN BE ASSIGNED TO SYNCHRONOUS/ASYNCHRONOUS MOVES
- CURRENT IMPLEMENTATION BASED ON THE NOTION OF **SYNCHRONOUS PRODUCT NET**:

Process Model

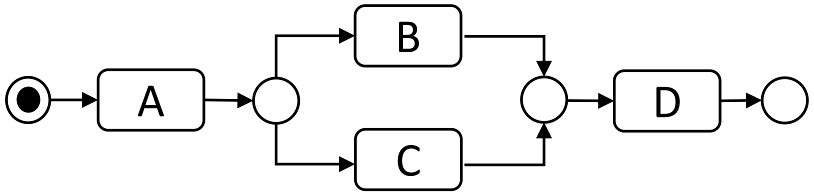


`<A,D,B>`

ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

- ARISING FROM **A. ADRIANSYAH PhD. (TU/e)**
- COSTS CAN BE ASSIGNED TO SYNCHRONOUS/ASYNCHRONOUS MOVES
- CURRENT IMPLEMENTATION BASED ON THE NOTION OF **SYNCHRONOUS PRODUCT NET**:

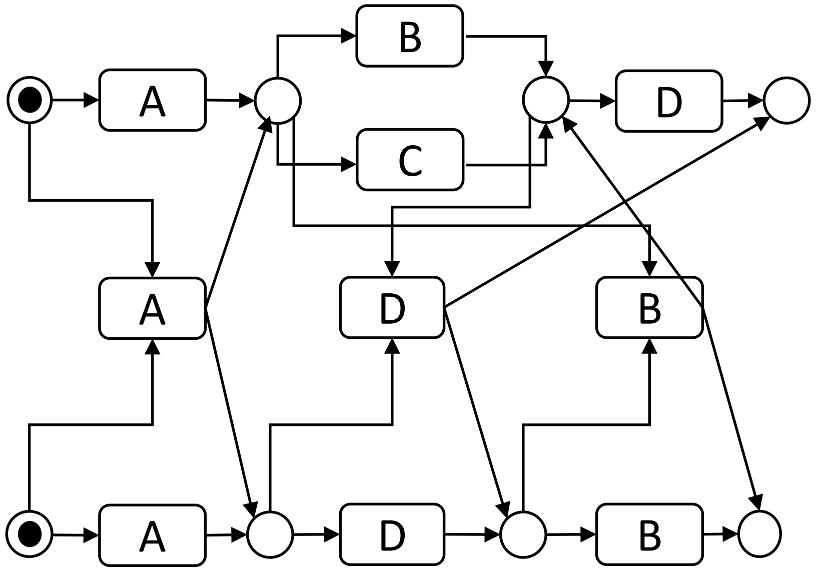
Process Model



Trace Net

ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

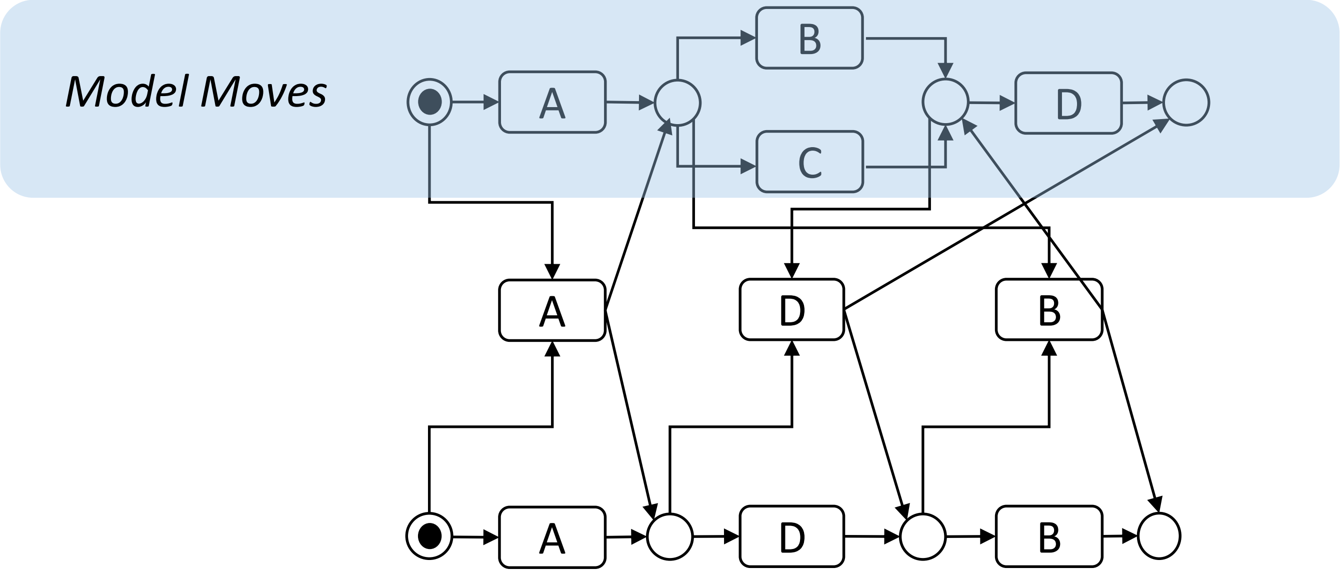
- ARISING FROM **A. ADRIANSYAH PhD. (TU/e)**
- COSTS CAN BE ASSIGNED TO SYNCHRONOUS/ASYNCHRONOUS MOVES
- CURRENT IMPLEMENTATION BASED ON THE NOTION OF **SYNCHRONOUS PRODUCT NET**:



Synchronous Product Net

ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

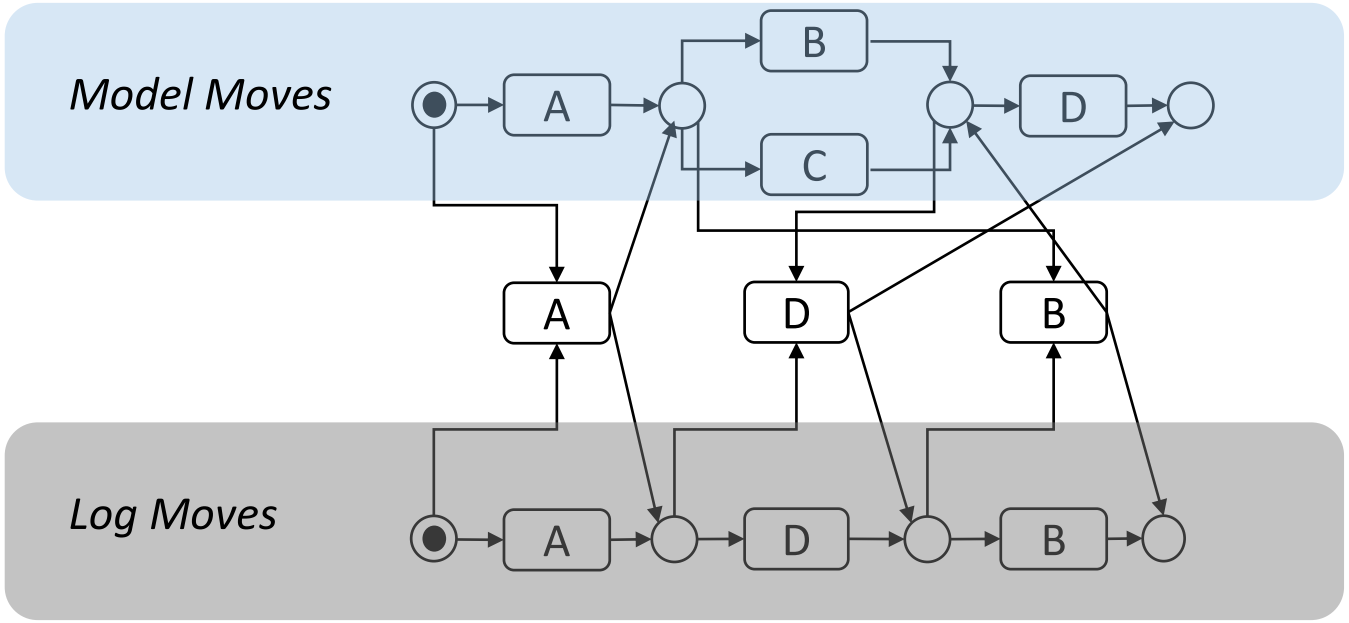
- ARISING FROM **A. ADRIANSYAH PhD. (TU/e)**
- COSTS CAN BE ASSIGNED TO SYNCHRONOUS/ASYNCHRONOUS MOVES
- CURRENT IMPLEMENTATION BASED ON THE NOTION OF **SYNCHRONOUS PRODUCT NET**:



Synchronous
Product
Net

ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

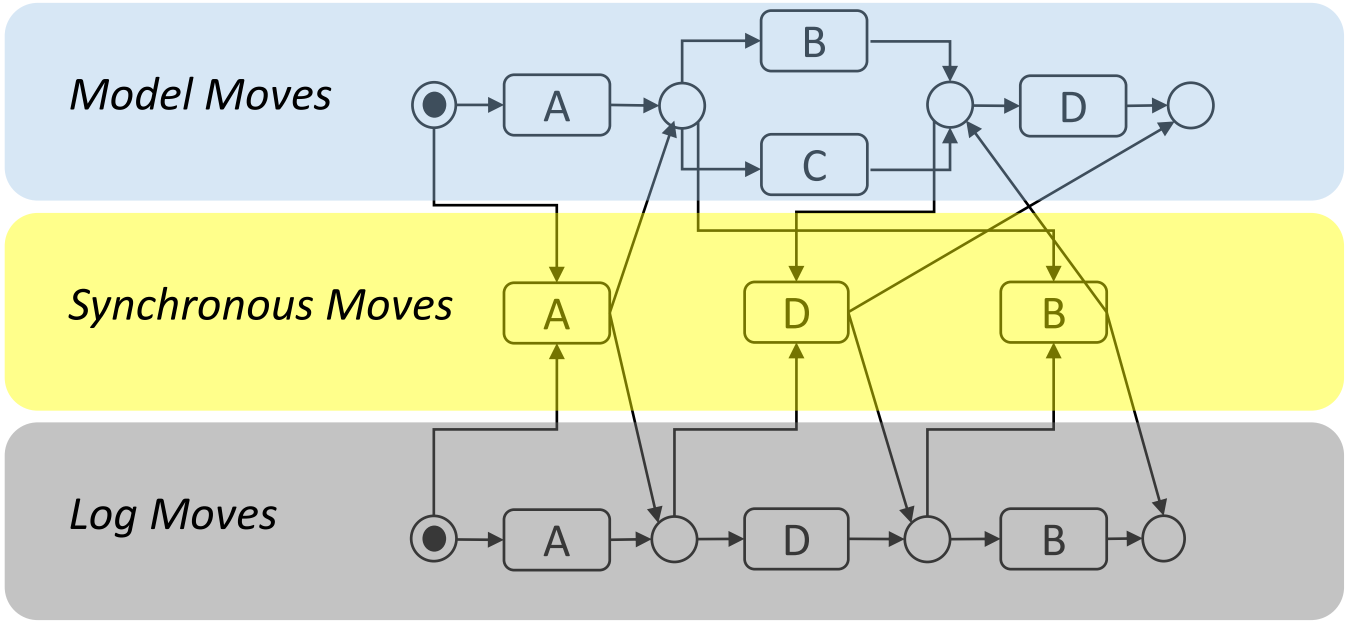
- ARISING FROM **A. ADRIANSYAH PhD. (TU/e)**
- COSTS CAN BE ASSIGNED TO SYNCHRONOUS/ASYNCHRONOUS MOVES
- CURRENT IMPLEMENTATION BASED ON THE NOTION OF **SYNCHRONOUS PRODUCT NET**:



Synchronous Product Net

ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

- ARISING FROM **A. ADRIANSYAH PhD. (TU/e)**
- COSTS CAN BE ASSIGNED TO SYNCHRONOUS/ASYNCHRONOUS MOVES
- CURRENT IMPLEMENTATION BASED ON THE NOTION OF **SYNCHRONOUS PRODUCT NET**:



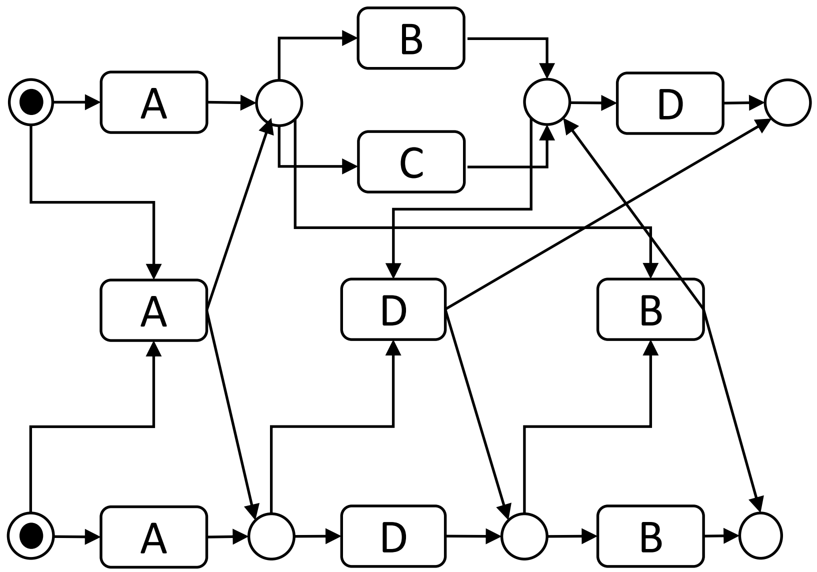
Synchronous Product Net

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Campus de Excelencia Internacional

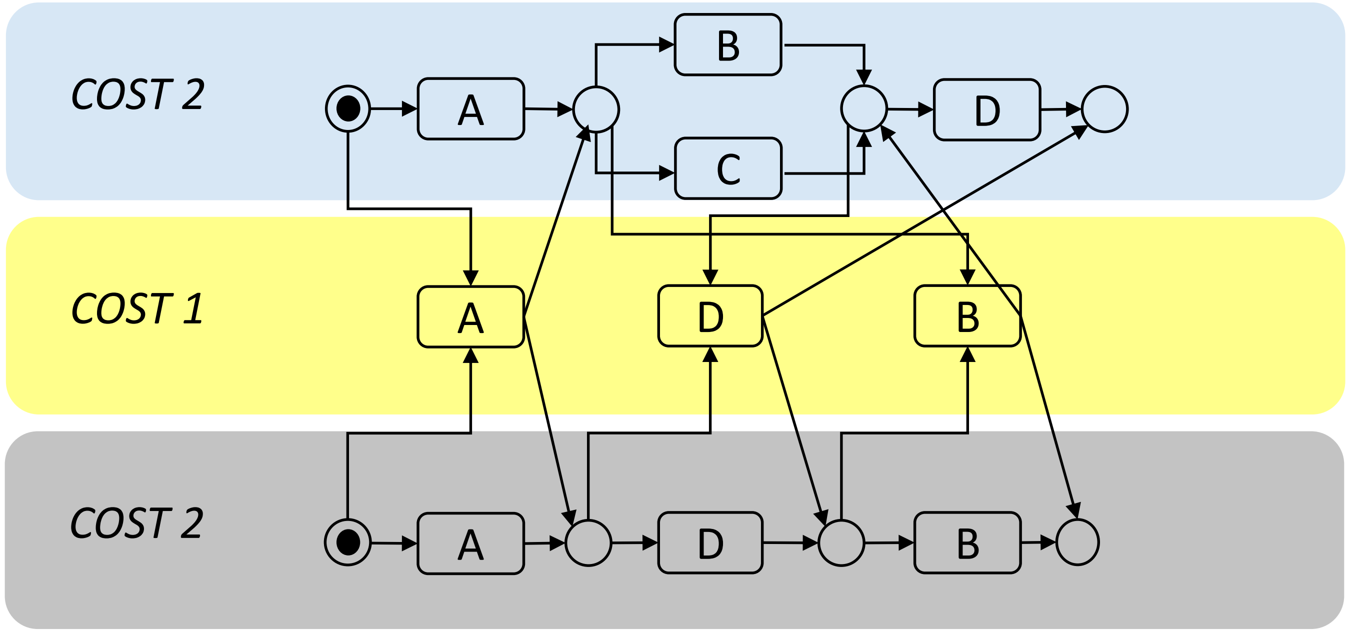
ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

- PROBLEM CASTED AS **REACHABILITY WITH COSTS**!



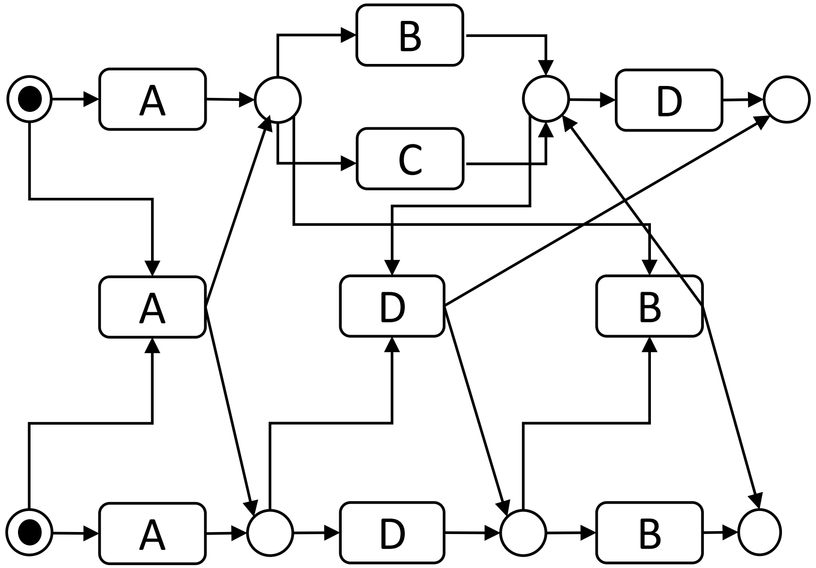
ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

- PROBLEM CASTED AS **REACHABILITY WITH COSTS**!



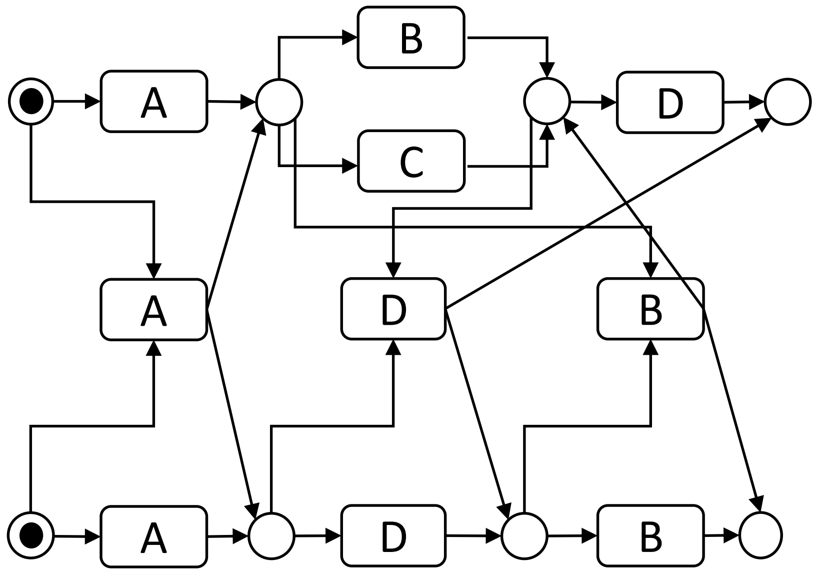
ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

- PROBLEM CASTED AS **REACHABILITY WITH COSTS**!



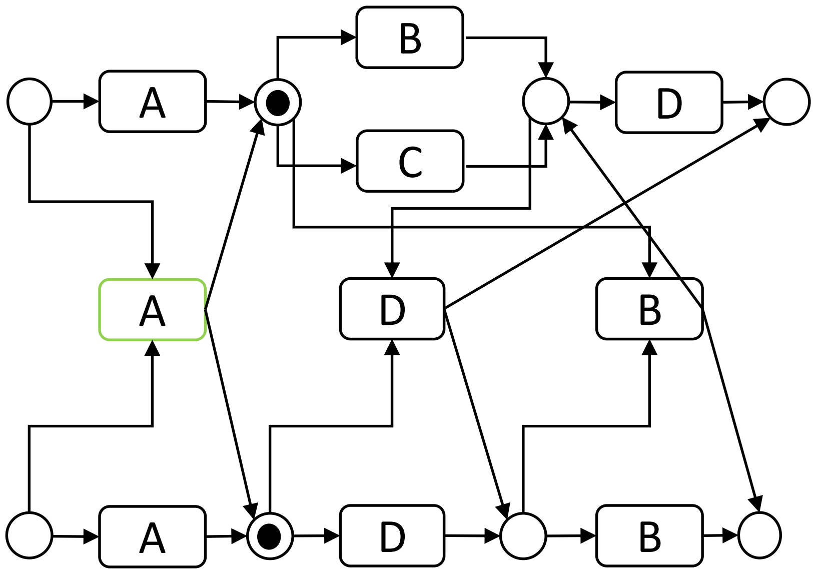
ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

- PROBLEM CASTED AS **REACHABILITY WITH COSTS**!



ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

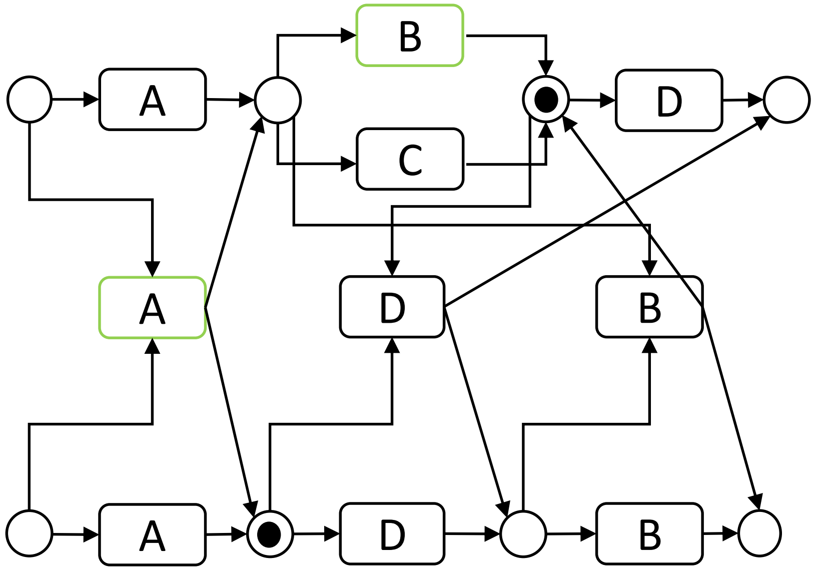
- PROBLEM CASTED AS **REACHABILITY WITH COSTS**!



A	
A	

ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

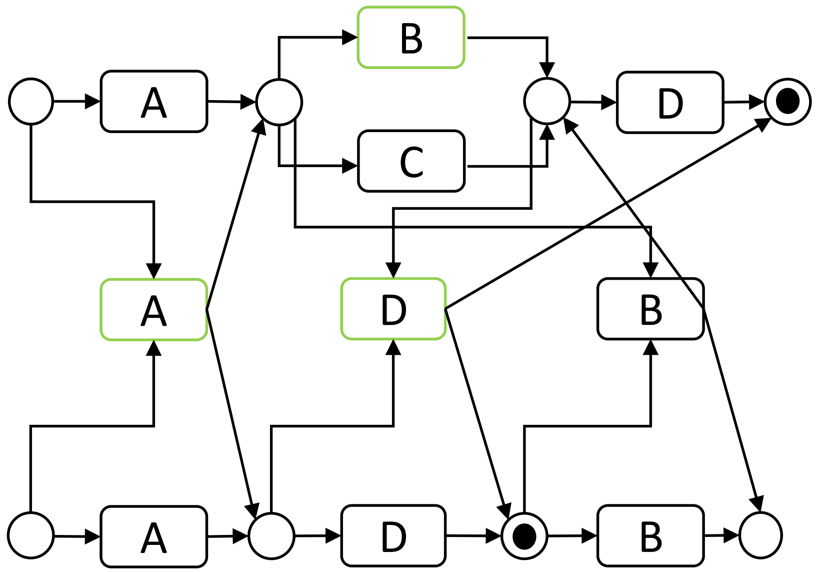
- PROBLEM CASTED AS **REACHABILITY WITH COSTS**!



A	B	
A	>	

ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

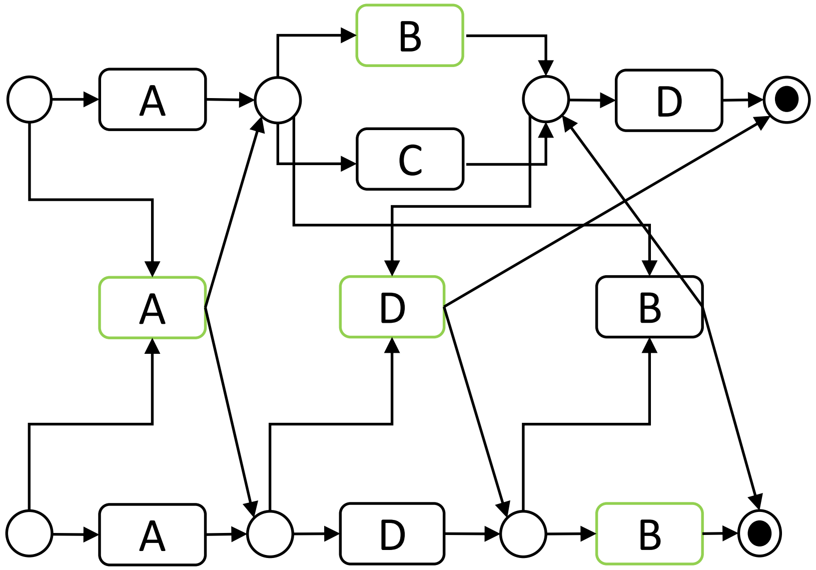
- PROBLEM CASTED AS **REACHABILITY WITH COSTS**!



A	B	D
A	>	D

ALIGNMENTS BETWEEN EVENT LOGS AND PROCESS MODELS

- PROBLEM CASTED AS **REACHABILITY WITH COSTS**!



A	B	D	>
A	>	D	B

Alignments: The Theory II

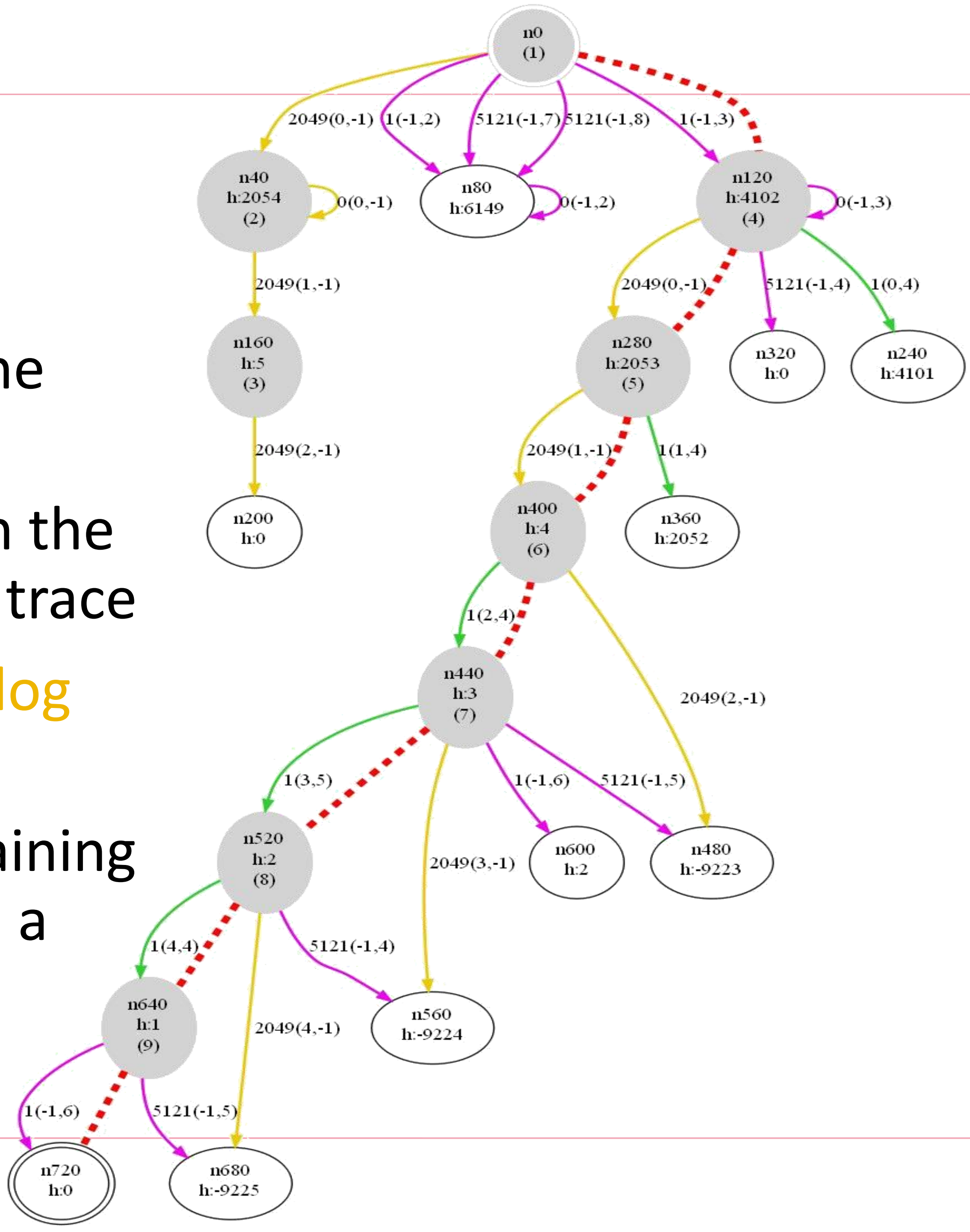
Finding an alignment for a given model $\mathtt{M}$ and trace $\mathtt{t}$ for cost function c:

Identify the cheapest firing sequence from m_i to m_f in a synchronous product model S, with cost function c

- For Reset/Inhibitor nets this problem is *undecidable*,
- For Petri nets this problem is EXPSPACE hard,
- For 1-safe nets, this problem is PSPACE hard,
- For Free-choice Petri nets this problem is NP hard,
- For marked graphs, this problem is polynomial…

Computing Alignments

- The search space is the statespace of the synchronous product model
- Each node is a combination of a state in the model and the remaining events in the trace
- Each arc is a move on model, move on log or a synchronous move
- A heuristic function estimates the remaining distance to the final node (i.e. model in a final state and all events executed)
- We find a shortest path!



Computing Alignments (A* with fast lowerbound)

```
Initialize HashBackedPriorityQueue q
While (peek(q) is not target t)
  VisitedNode n = head(q)
  If the estimate for node(n) is not exact
    Compute the exact estimate for the remaining distance to t and
    add n to the priority queue
    continue
  add n to the considered nodes
  For each edge in the graph from node(n) to m
    If m was considered before, continue
    If m is in the queue with lower cost, continue
    If m is in the queue with higher cost, update and reposition it
    If m is new,
      compute a fast lowerbound for the remaining distance to t
      v = new VisitedNode(m)
      set n a predecessor for v
      add v to the priority queue
Return head(q)
```

Computing Alignments (A* with fast lowerbound)

O(log(size q)+α)

```
Initialize HashBackedPriorityQueue q
While (peek(q) is not target t)
  VisitedNode n = head(q)
  If the estimate for node(n) is not exact
    Compute the exact estimate for the remaining distance to t and
    add n to the priority queue
    continue
  add n to the considered nodes
  For each edge in the graph from node(n) to m
    If m was considered before, continue
    If m is in the queue with lower cost, continue
    If m is in the queue with higher cost, update and reposition it
    If m is new,
      compute a fast lowerbound for the remaining distance to t
      v = new VisitedNode(m)
      set n a predecessor for v
      add v to the priority queue
Return head(q)
```

Computing Alignments (A* with fast lowerbound)

O(log(size q)+α)

?

```
Initialize HashBackedPriorityQueue q
While (peek(q) is not target t)
  VisitedNode n = head(q)
  If the estimate for node(n) is not exact
    Compute the exact estimate for the remaining distance to t and
    add n to the priority queue
    continue
  add n to the considered nodes
  For each edge in the graph from node(n) to m
    If m was considered before, continue
    If m is in the queue with lower cost, continue
    If m is in the queue with higher cost, update and reposition it
    If m is new,
      compute a fast lowerbound for the remaining distance to t
      v = new VisitedNode(m)
      set n a predecessor for v
      add v to the priority queue
Return head(q)
```

Computing Alignments (A* with fast lowerbound)

```
Initialize HashBackedPriorityQueue q
While (peek(q) is not target t)
  VisitedNode n = head(q)
  If the estimate for node(n) is not exact
    Compute the exact estimate for the remaining distance to t and
    add n to the priority queue
    continue
  add n to the considered nodes
  For each edge in the graph from node(n) to m
    If m was considered before, continue
    If m is in the queue with lower cost, continue
    If m is in the queue with higher cost, update and reposition it
    If m is new,
      compute a fast lowerbound for the remaining distance to t
      v = new VisitedNode(m)
      set n a predecessor for v
      add v to the priority queue
Return head(q)
```

O(log(size q)+α)

?

O(log(size q)+ α)

Computing Alignments (A* with fast lowerbound)

```
Initialize HashBackedPriorityQueue q
While (peek(q) is not target t)
  VisitedNode n = head(q)
  If the estimate for node(n) is not exact
    Compute the exact estimate for the remaining distance to t and
    add n to the priority queue
    continue
  add n to the considered nodes
  For each edge in the graph from node(n) to m
    If m was considered before, continue
    If m is in the queue with lower cost, continue
    If m is in the queue with higher cost, update and reposition it
    If m is new,
      compute a fast lowerbound for the remaining distance to t
      v = new VisitedNode(m)
      set n a predecessor for v
      add v to the priority queue
Return head(q)
```

O(log(size q)+α)

?

O(log(size q)+ α)

O(log(size q)+ α)

Computing Alignments (A* with fast lowerbound)

```
Initialize HashBackedPriorityQueue q
While (peek(q) is not target t)
  VisitedNode n = head(q)
  If the estimate for node(n) is not exact
    Compute the exact estimate for the remaining distance to t and
    add n to the priority queue
    continue
  add n to the considered nodes
  For each edge in the graph from node(n) to m
    If m was considered before, continue
    If m is in the queue with lower cost, continue
    If m is in the queue with higher cost, update and reposition it
    If m is new,
      compute a fast lowerbound for the remaining distance to t
      v = new VisitedNode(m)
      set n a predecessor for v
      add v to the priority queue
Return head(q)
```

O(log(size q)+α)

?

O(log(size q)+ α)

O(log(size q)+ α)

O(1)

Computing Alignments (A* with fast lowerbound)

```
Initialize HashBackedPriorityQueue q
While (peek(q) is not target t)
  VisitedNode n = head(q)
  If the estimate for node(n) is not exact
    Compute the exact estimate for the remaining distance to t and
    add n to the priority queue
    continue
  add n to the considered nodes
  For each edge in the graph from node(n) to m
    If m was considered before, continue
    If m is in the queue with lower cost, continue
    If m is in the queue with higher cost, update and reposition it
    If m is new,
      compute a fast lowerbound for the remaining distance to t
      v = new VisitedNode(m)
      set n a predecessor for v
      add v to the priority queue
Return head(q)
```

O(log(size q)+α)

?

O(log(size q)+ α)

O(log(size q)+ α)

O(1)

O(log(size q)+ α)

Estimating remaining distance I

- Naïve estimation: 0 — Trivial
- LP-based estimation: — "Polynomial"
  - `Minimize  c.x`
    `Where     A.x = r`
    `          c.x ≥ c.x′`
- ILP-based estimation: — Exponential
- Hybrid ILP (1 sec for "I" part) — Exponential
- Caching LP basis solutions — Exponential

Little to no difference in practice (LP-based, ILP-based, Hybrid ILP, Caching LP basis solutions)

Implementations: Estimator Versions

Petri nets:

- Dijkstra (estimator 0)
- Naive (estimator parikh)
- ILP (estimator using ILP)
- Basis caching
- Fast lowerbounds

Process Trees:

- Naive (estimator parikh)
- LP (estimator using LP)
- Hybrid ILP (ILP & LP)
- Special constraints for OR
- Fast lowerbounds
- Basis caching within LP
- Statespace reduction (stubborn sets)

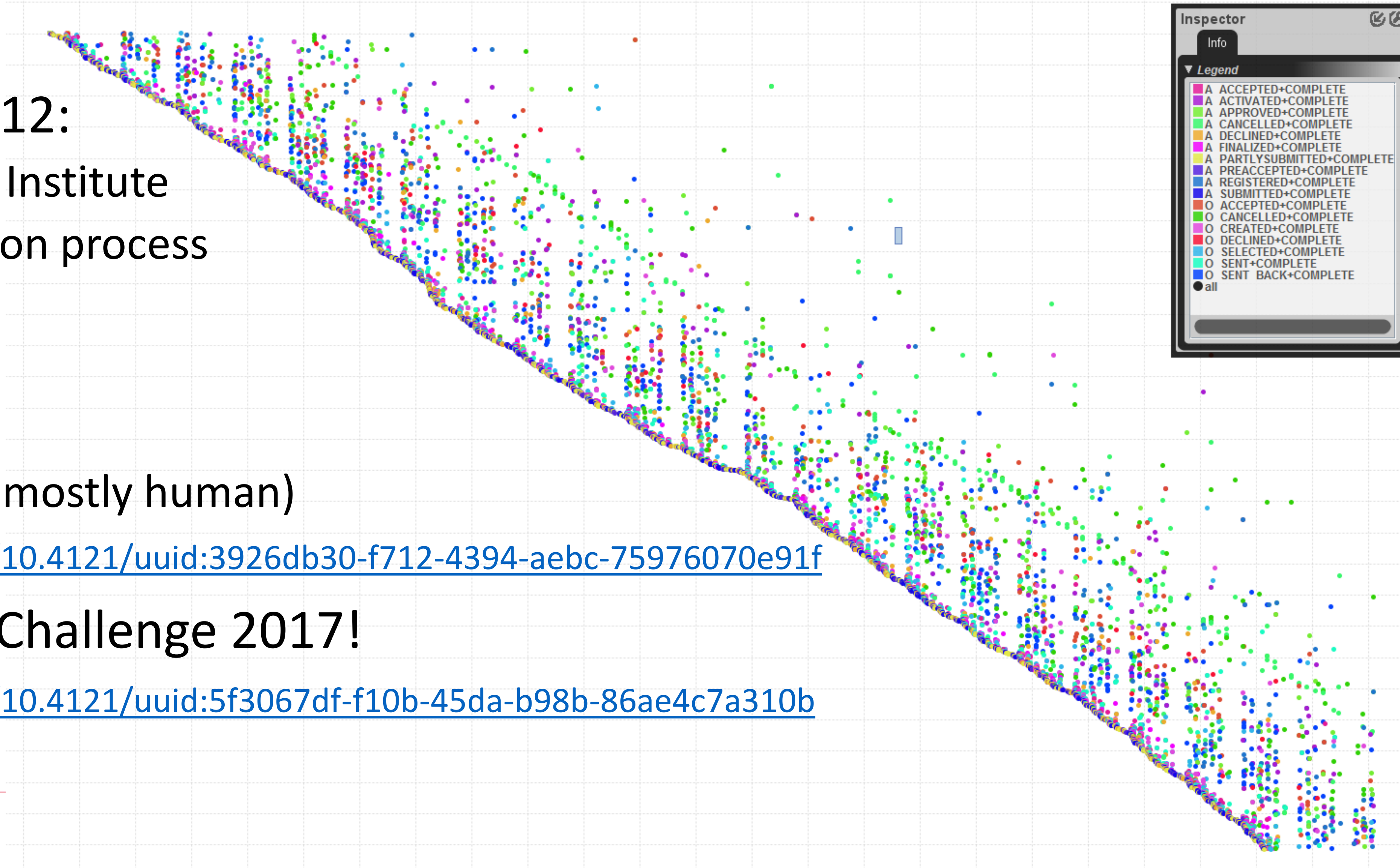
ProM (Lite) Demo

BPI Challenge 2012:

- Real Financial Institute
- Loan application process
- 13087 cases
- 92093 events
- 17 activities
- 61 resources (mostly human)
- http://dx.doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f

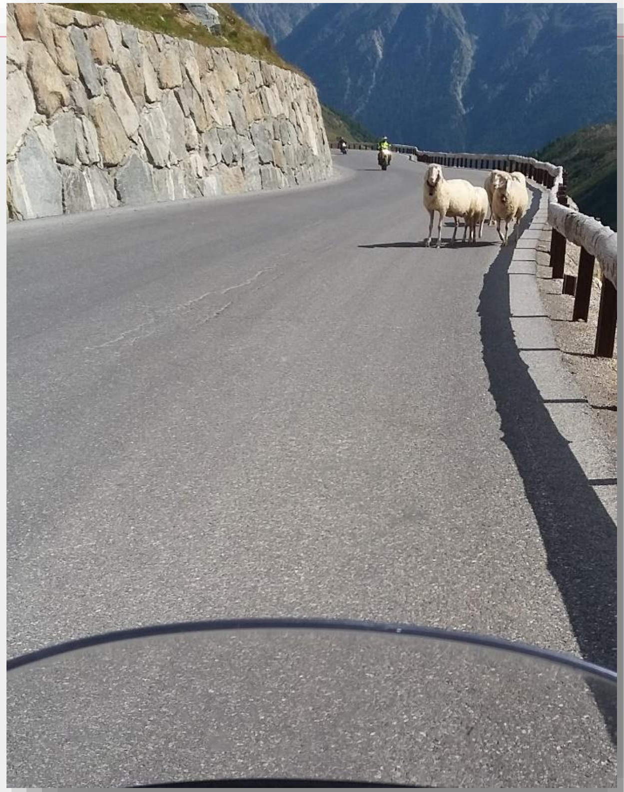
Updated for BPI Challenge 2017!

- http://dx.doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b



Applications of Alignments

- Conformance Metrics
  - Fitness
  - Precision
  - Generalization
- Enhancement
- Process discovery
  - Genetic algorithms
  - Model Repair
- Process model animation
  - Animate models based on alignments (Sander Leemans)
- Automated compliance checking
  - Uses anti-patterns and *n*-to-*m* mappings for activities (Elham Rhamezani)



Active research on alignments

- Data/Resource aware alignments
  - Dealing with infinity and inverse function theory
  - Multi perspective
- Online Alignments
  - In constant time and memory
- Incremental Alignments
  - Using the ILP to the maximum to decompose the problem
- Alignments for Different model classes
  - Mixed paradigm models (mix of declarative constraints and Petri nets)
  - DCR Graphs
- Approximating alignments

Conclusions

- Conformance Checking deals with the relation between event logs and process models
- Alignments are foundational to process mining
  - Form the basis for fitness/precision (and generalization)
  - They explain exactly where deviations occur
- Computing alignments is computationally hard
- Many variants exist, but all guarantee:
  - The projection to the log provides the observed trace
  - The projection to the model provides a valid run thereof

Future challenges

- Computational complexity
  - Especially with data, resources
  - Find tight bounds for online alignments
  - Fast "approximate" alignments
- Determinism of alignments
  - Same choices across alignments
- Obtaining the right models,
  - Translate informal text to formal models
  - Alignments on BPMN
- Industry adaptation