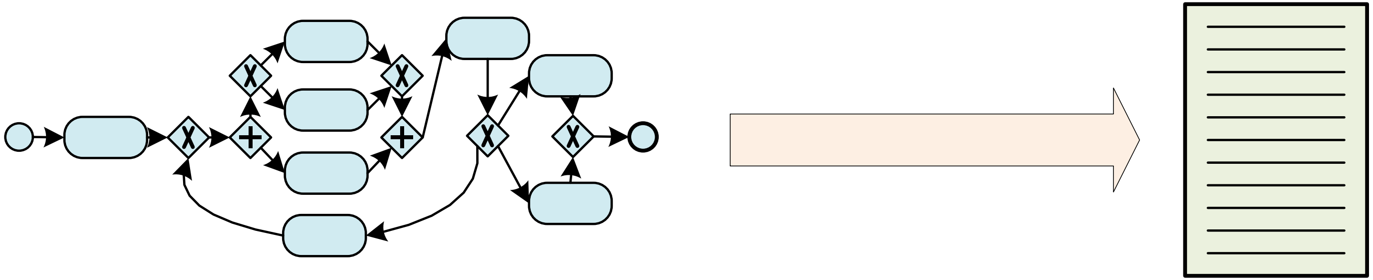
let's play

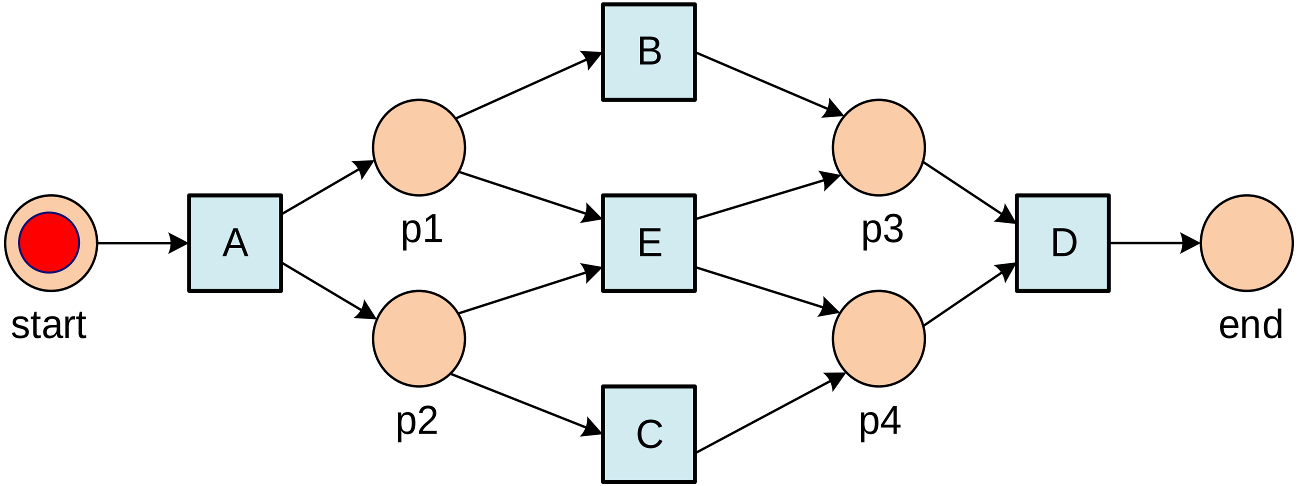
Play-Out



process model

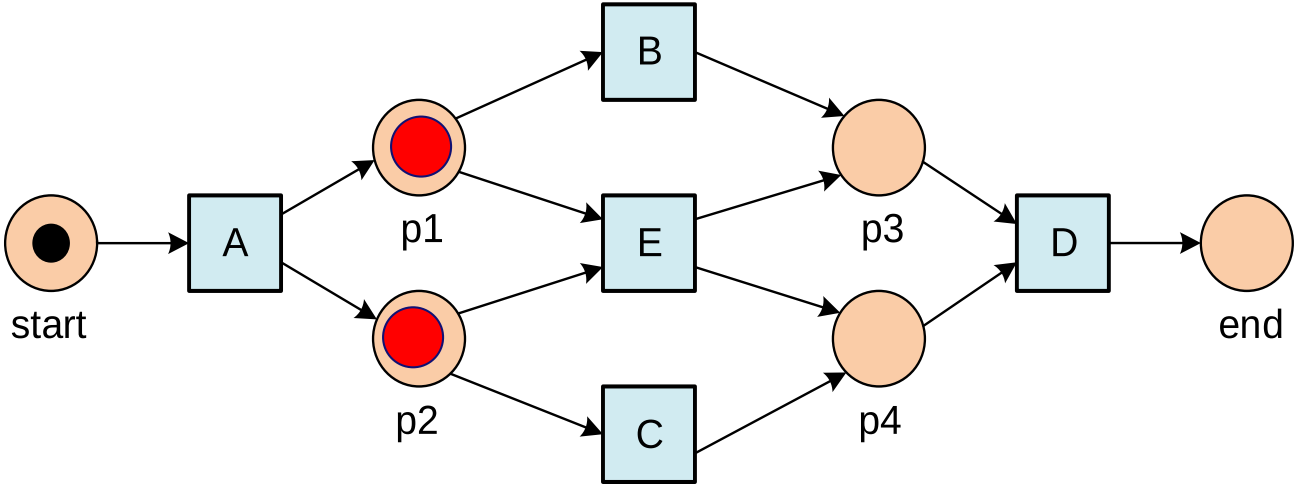
event log

Play-Out (Classical use of models)



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

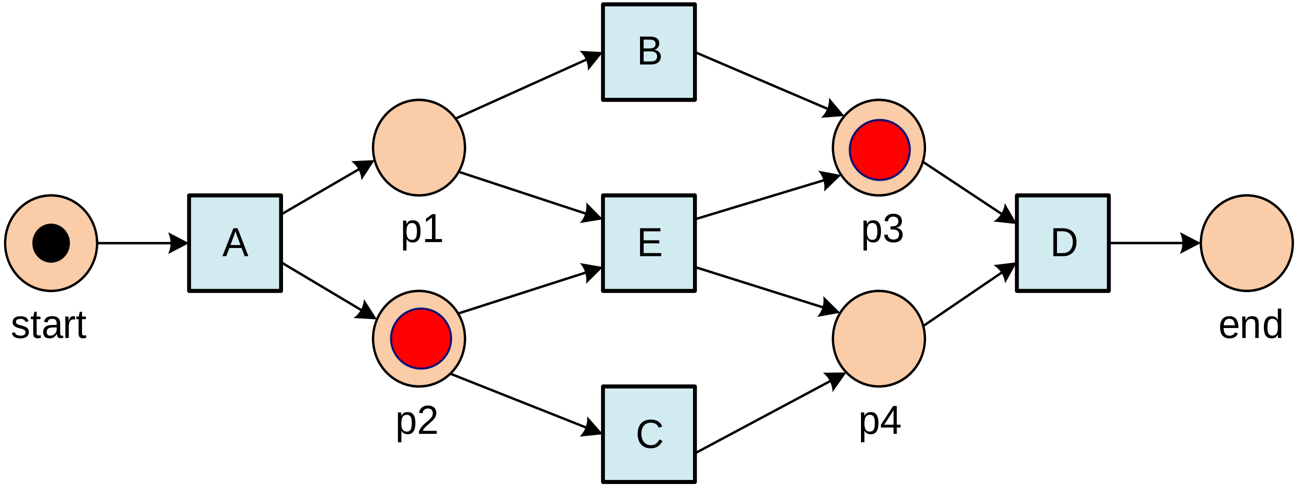
Play-Out (Classical use of models)



A

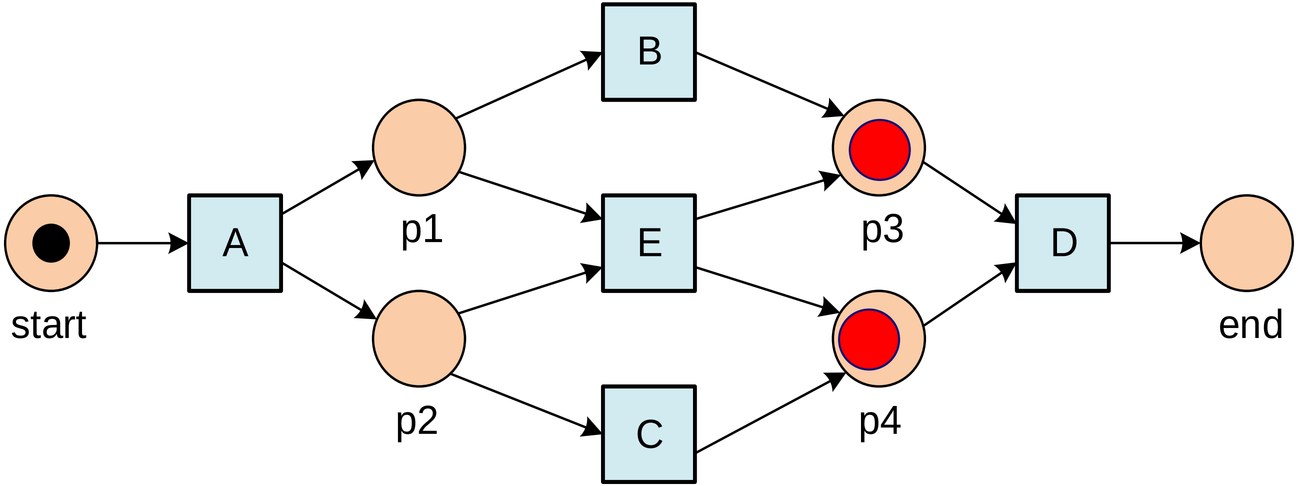
©Wil van der Aalst TU/e (use only with permission & acknowledgements)

Play-Out (Classical use of models)



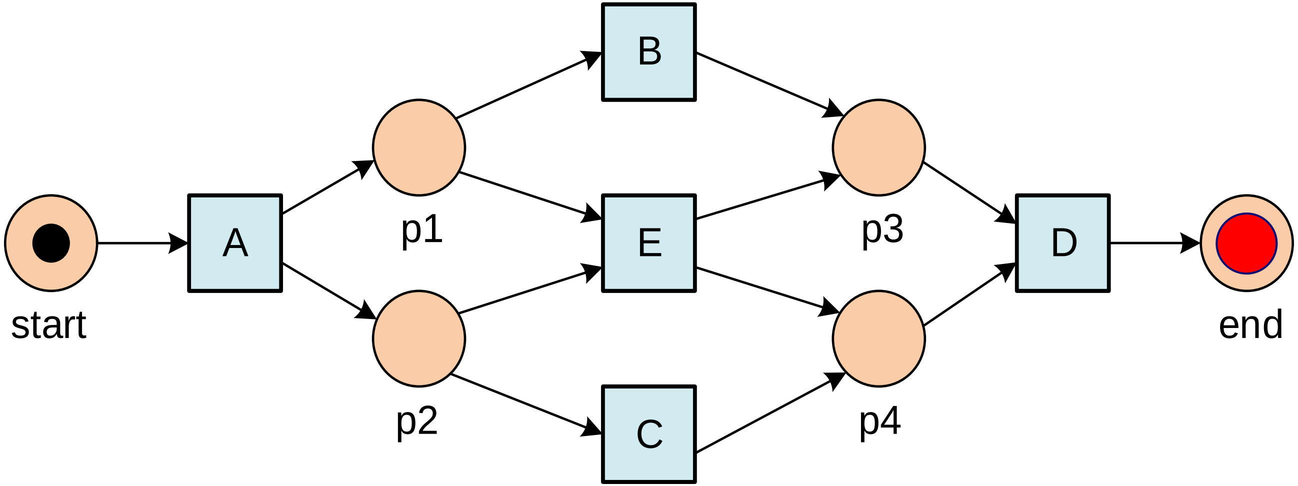
A B

Play-Out (Classical use of models)



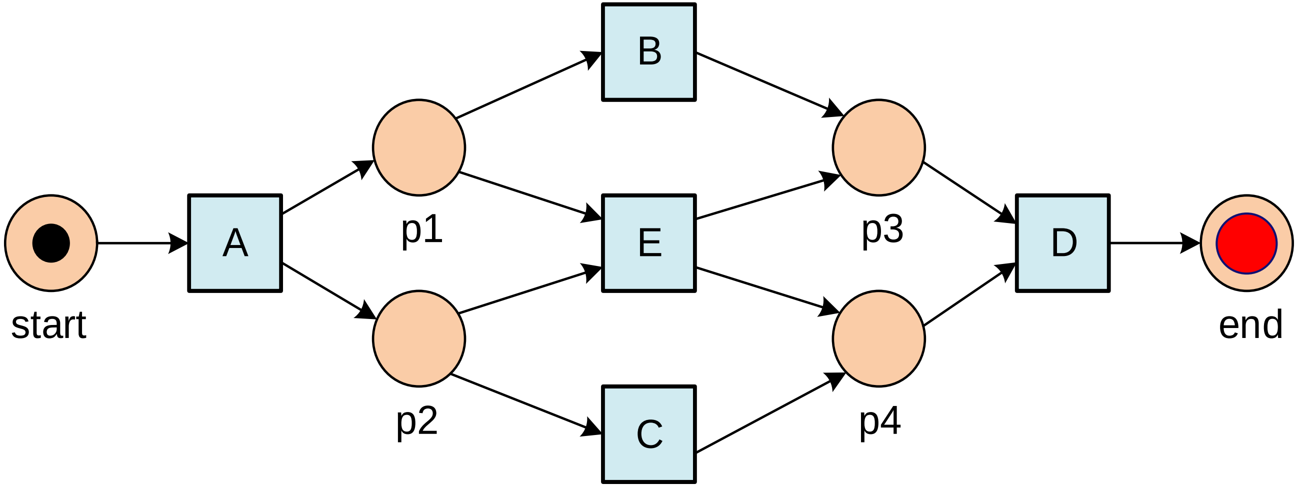
A B C

Play-Out (Classical use of models)



A B C D

Play-Out (Classical use of models)



A B C D

A E D

A E D

A C B D

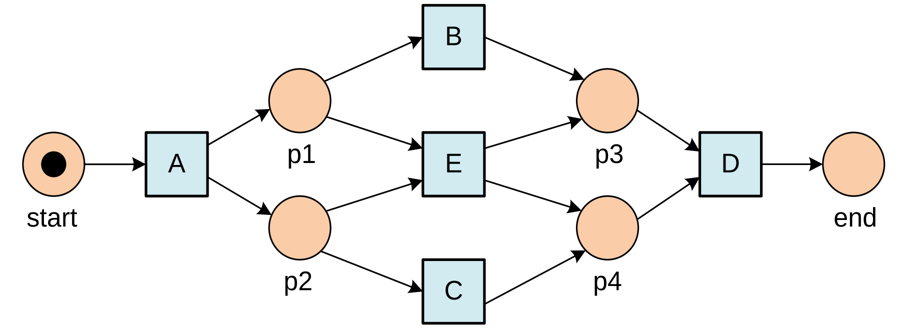
A B C D

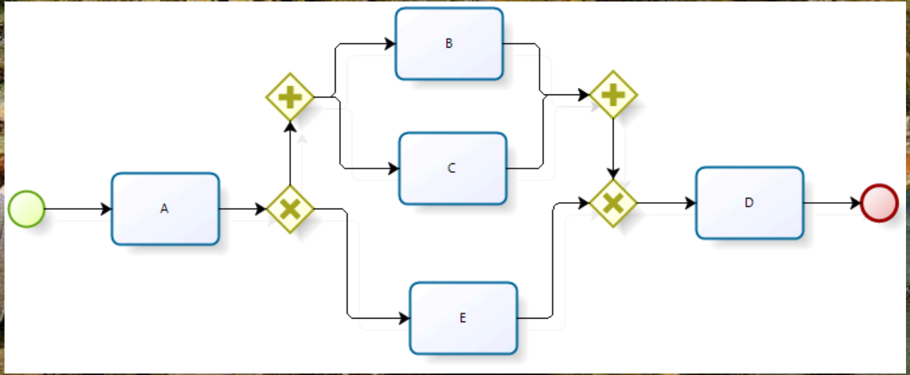
A C B D

A E D

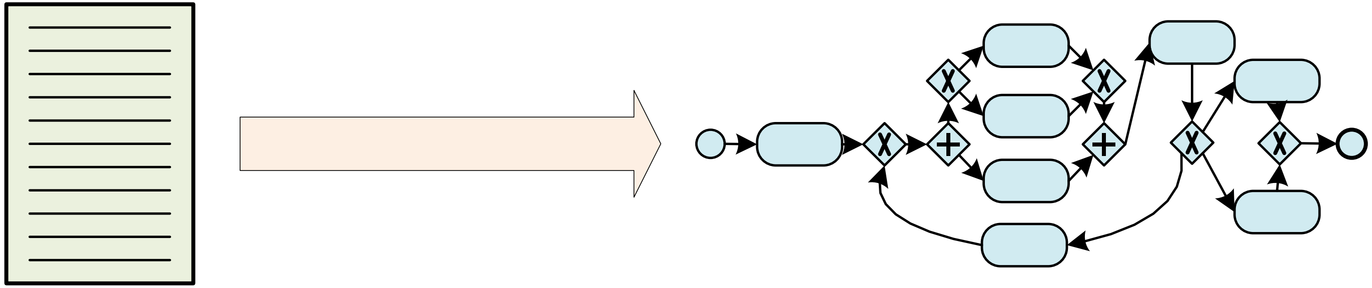
A C B D

Let’s not worry about syntax (there is difference between analysis and presentation)





Play-In



event log

process model

Play-In (Process Discovery, dude!)

A B C D

A E D

A E D

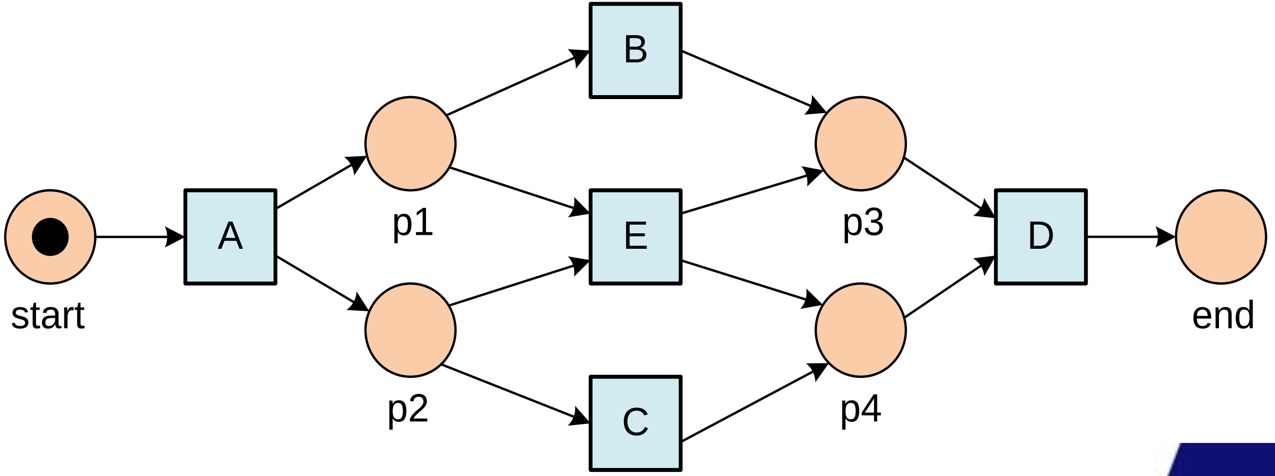
A B C D

A C B D

A C B D

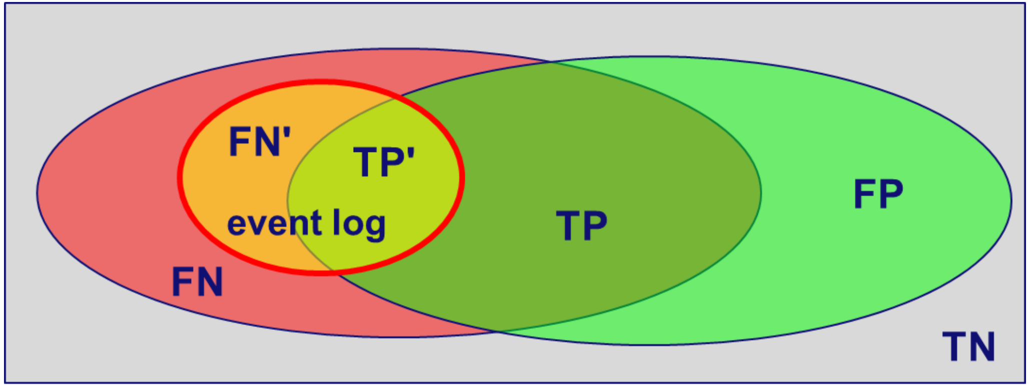
A E D

A C B D



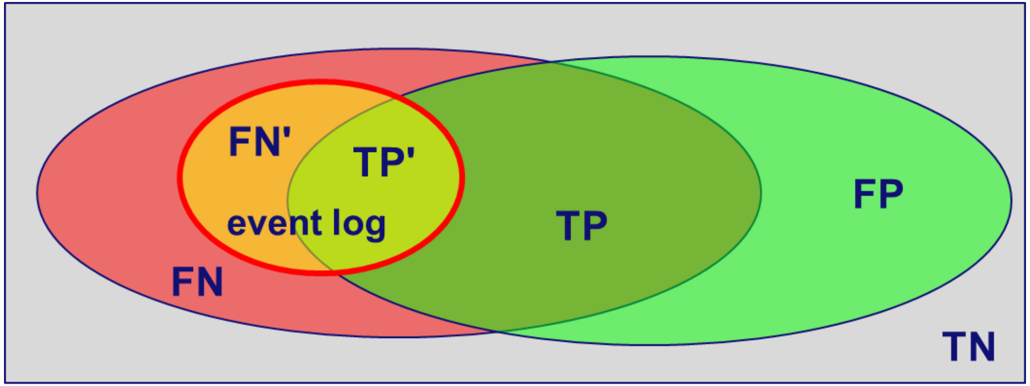
©Wil van der Aalst TU/e (use only with permission & acknowledgements)

Challenges



Challenges

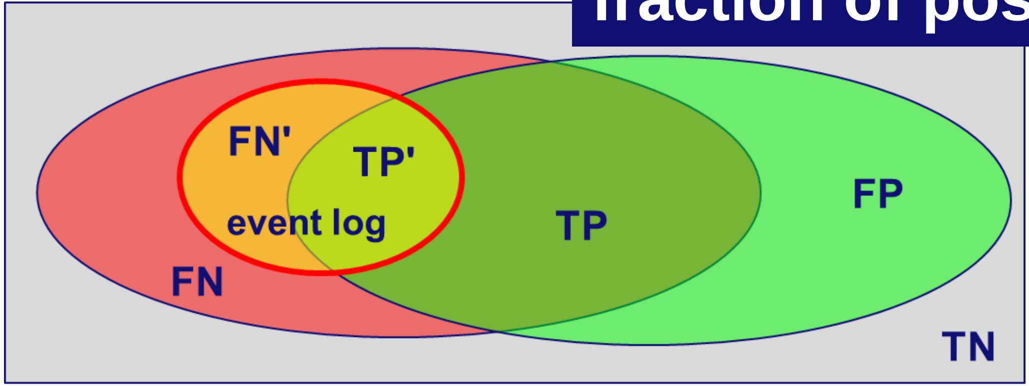
No negative examples
(cannot see what cannot happen)



Challenges

No negative examples
(cannot see what cannot happen)

Log contains only a fraction of possible traces

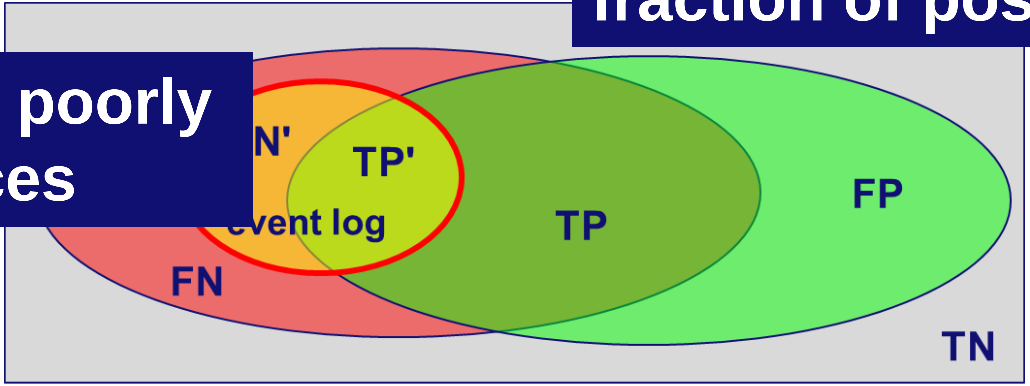


Challenges

No negative examples
(cannot see what cannot happen)

Log contains only a fraction of possible traces

Almost vs poorly fitting traces

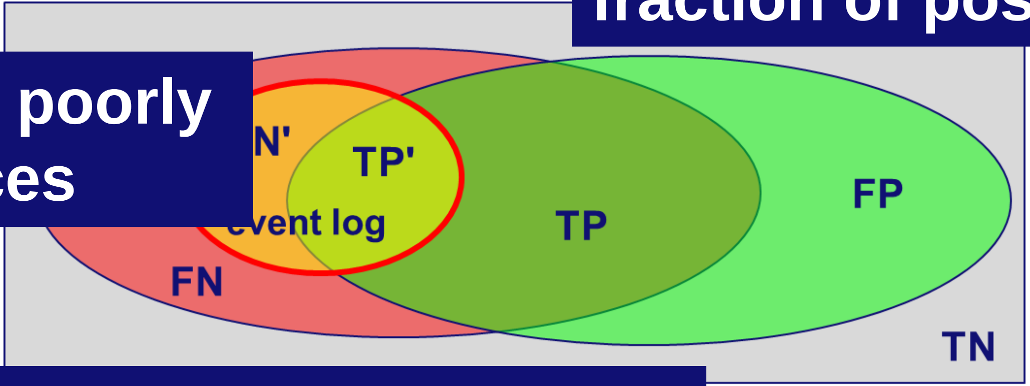


Challenges

No negative examples
(cannot see what cannot happen)

Log contains only a fraction of possible traces

Almost vs poorly fitting traces

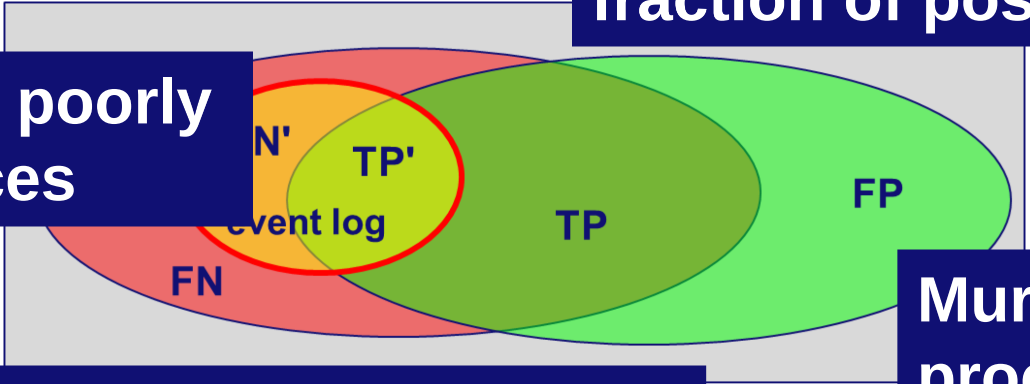


In case of loops often infinitely many possible traces

Challenges

No negative examples
(cannot see what cannot happen)

Log contains only a fraction of possible traces



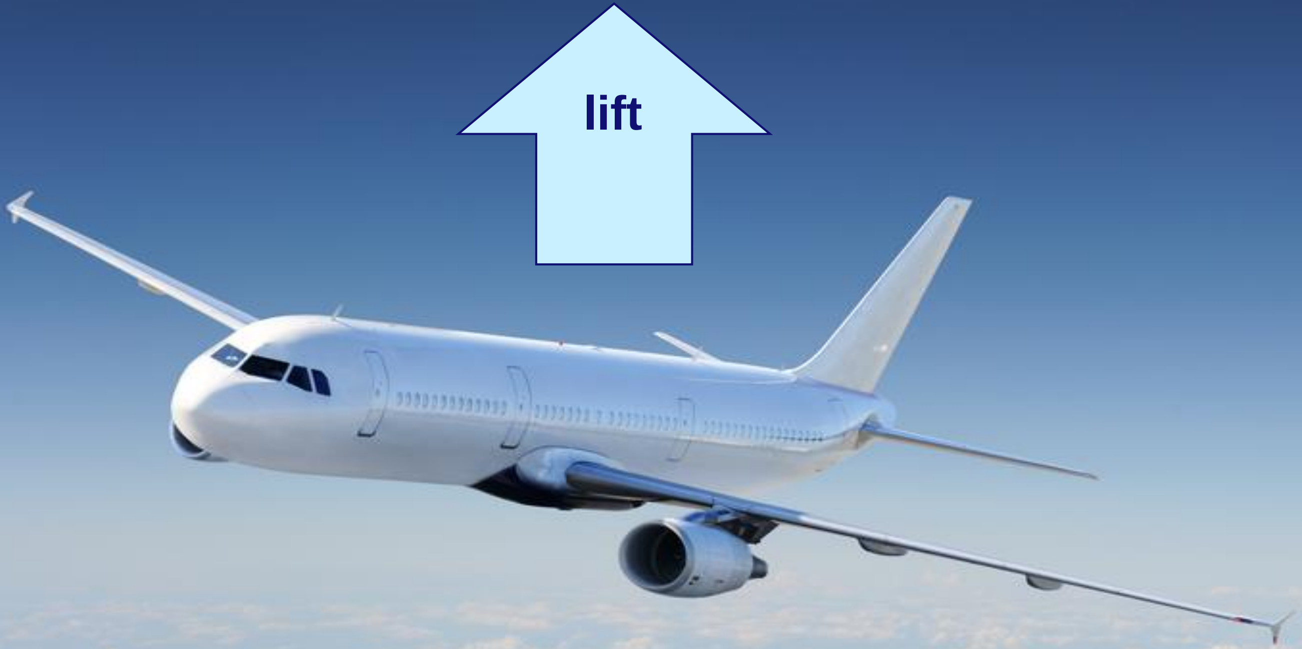
Almost vs poorly fitting traces

In case of loops often infinitely many possible traces

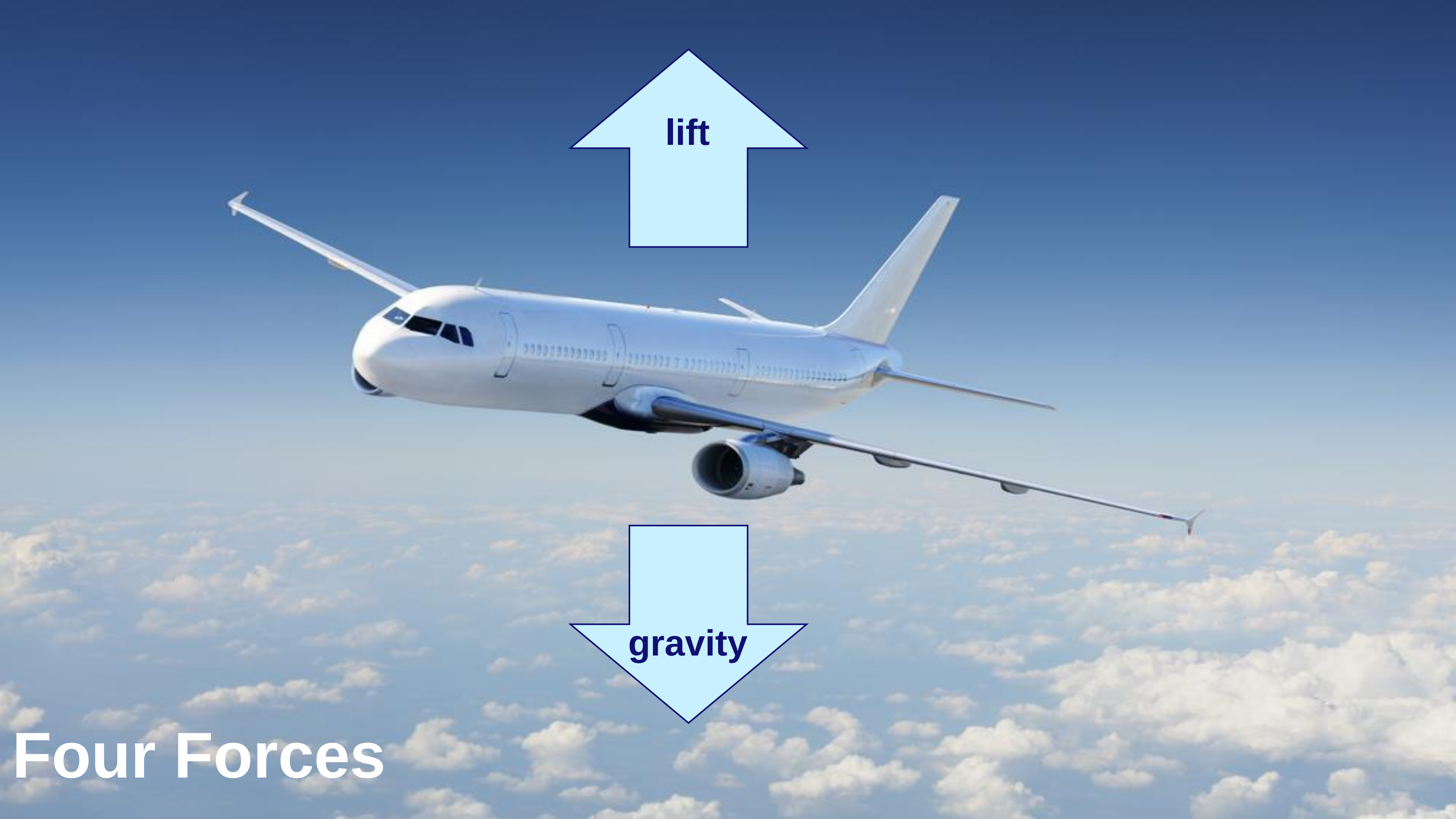
Murphy's law for process mining
(anything is possible, so probabilities matter)

Four Forces

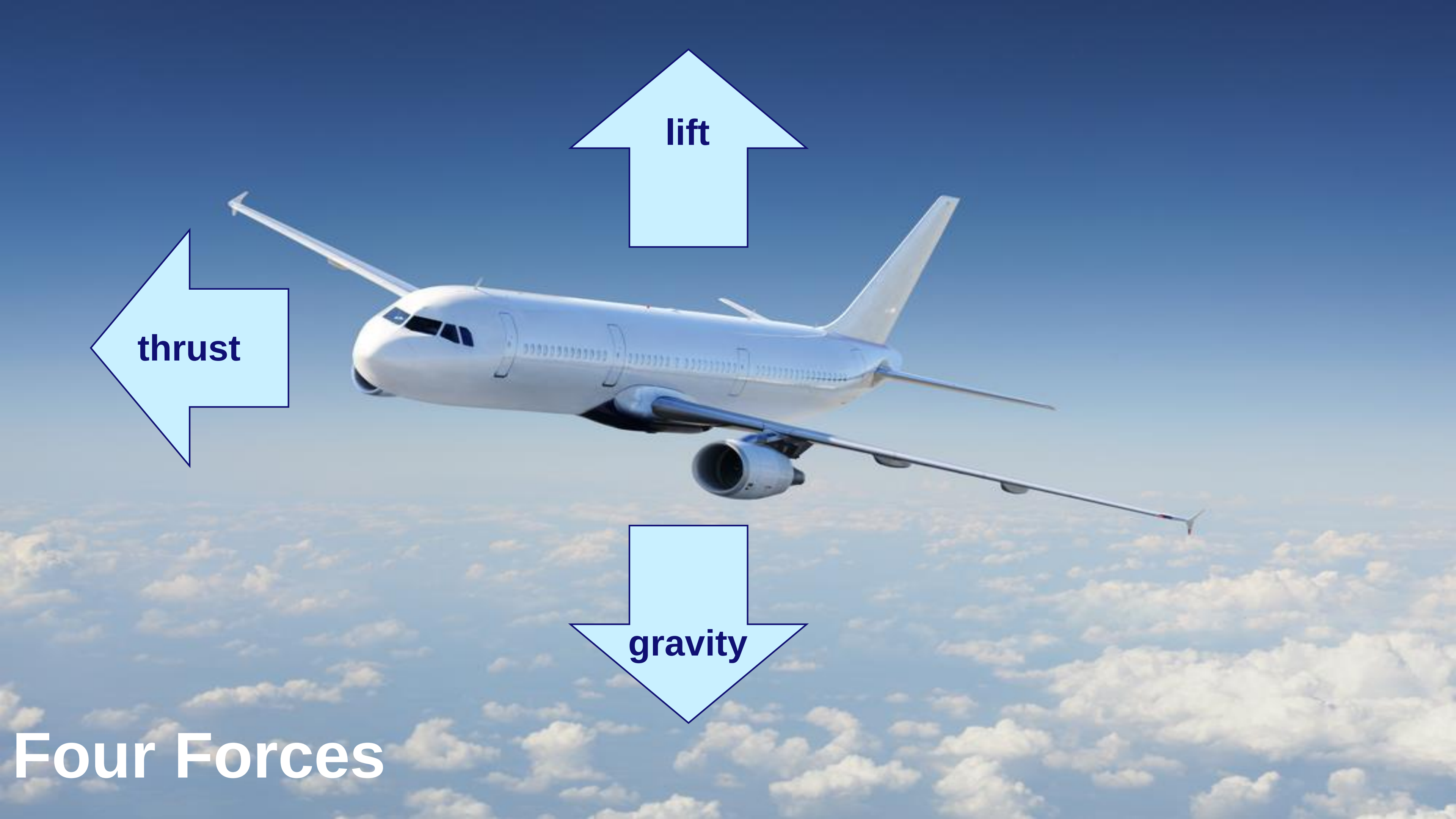
Four Forces



Four Forces



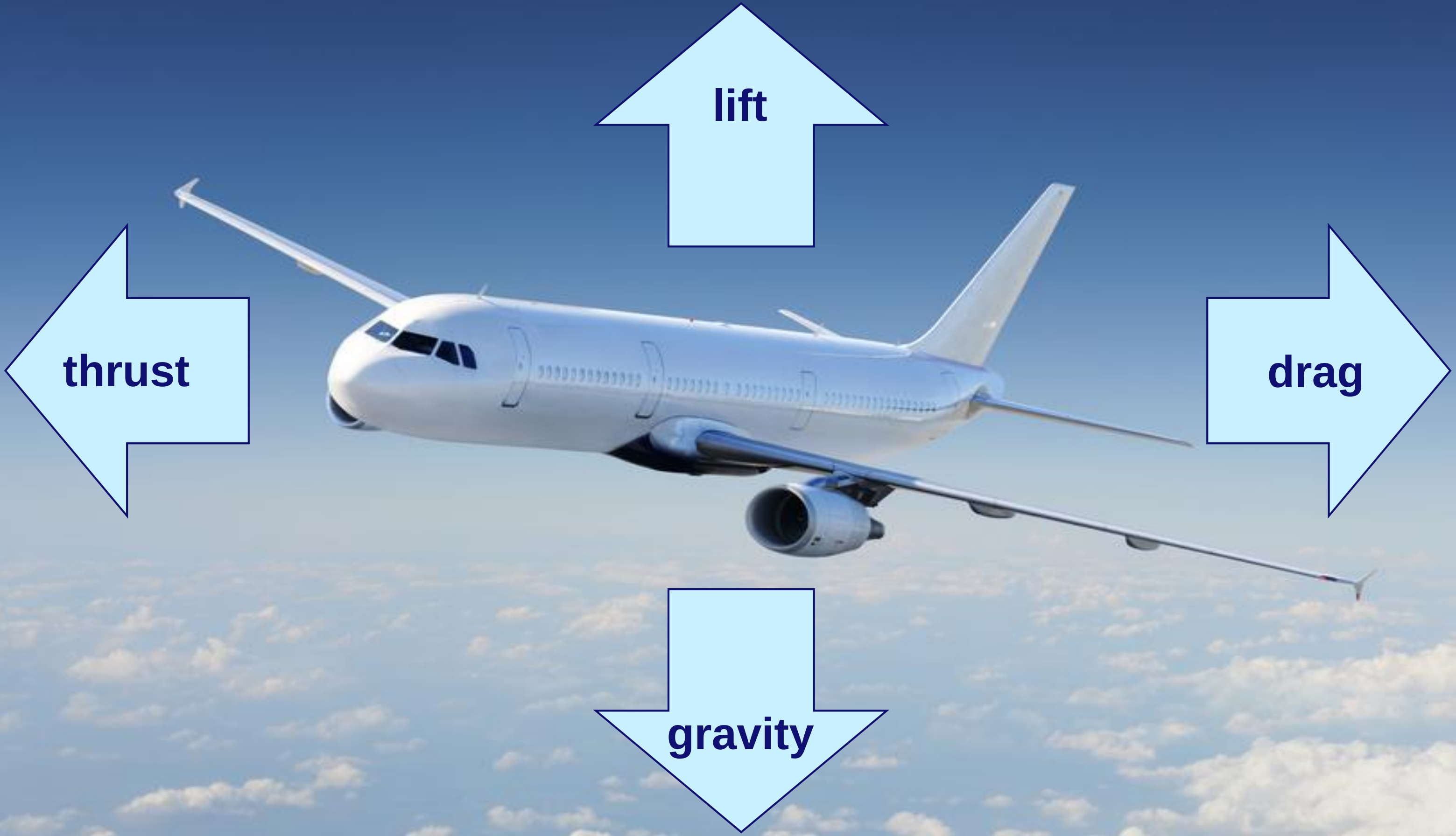
Four Forces



lift

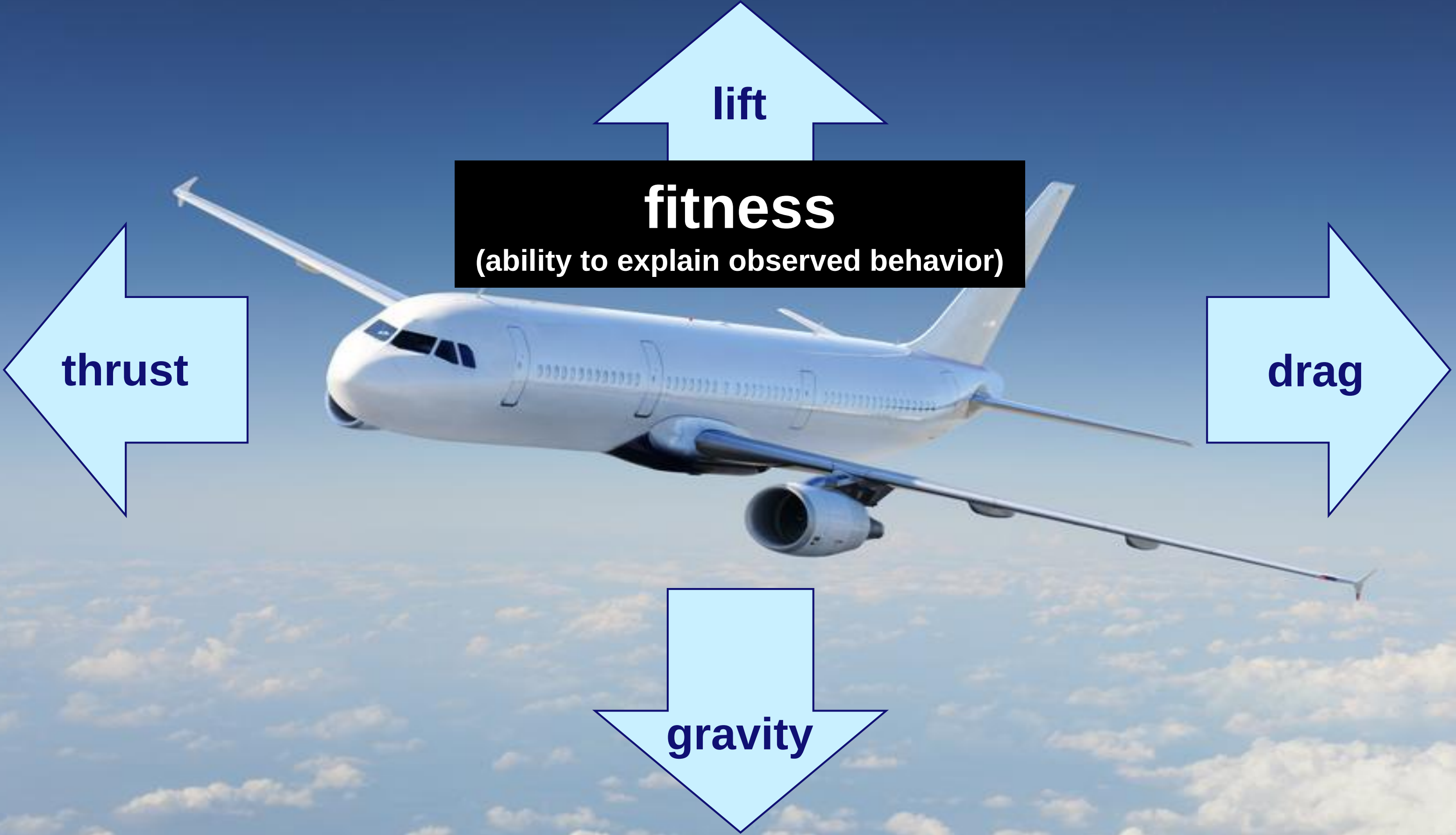
thrust

gravity



Four Forces

Four Forces



lift

fitness

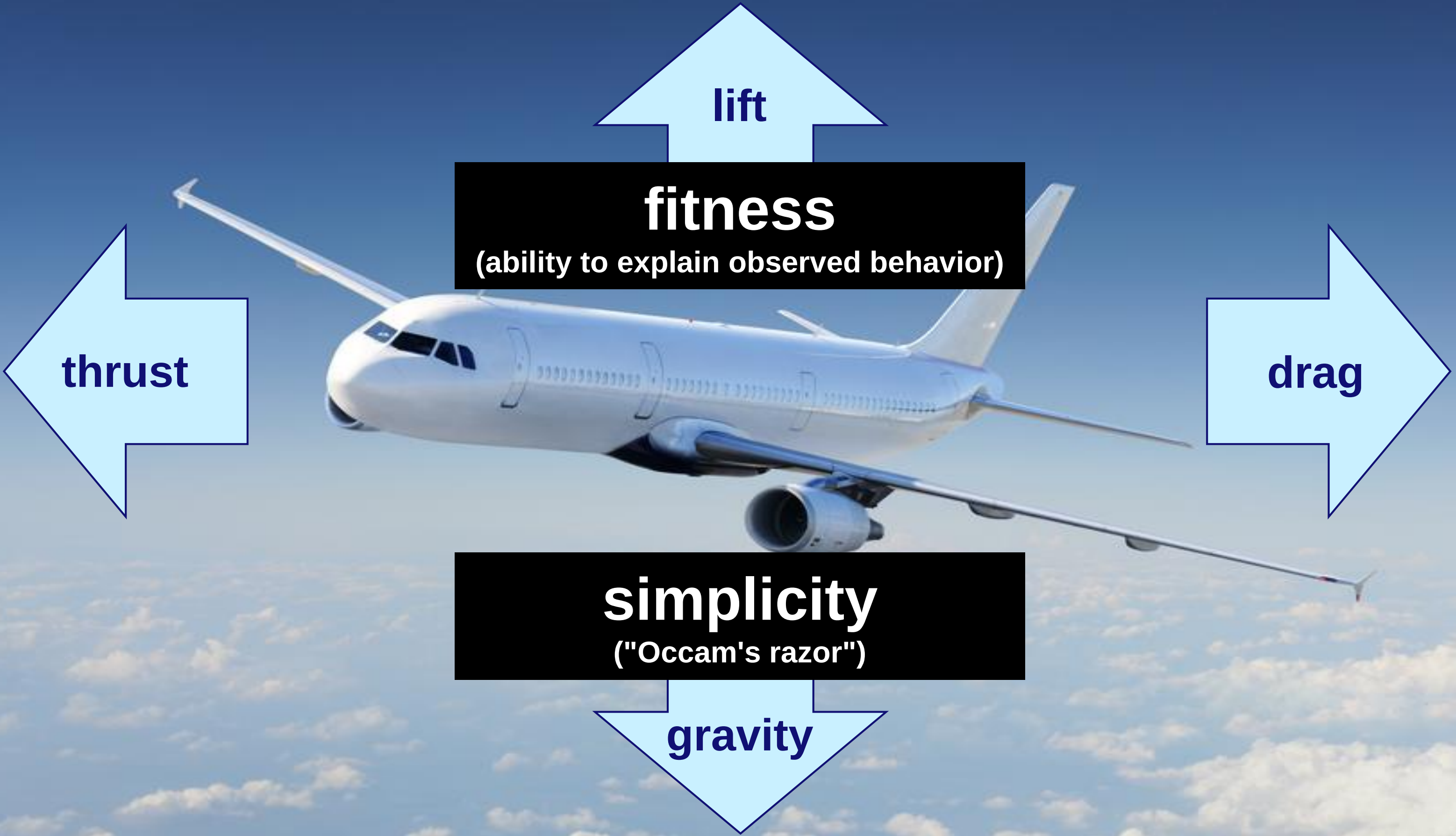
(ability to explain observed behavior)

thrust

drag

gravity

Four Forces



lift

fitness

(ability to explain observed behavior)

thrust

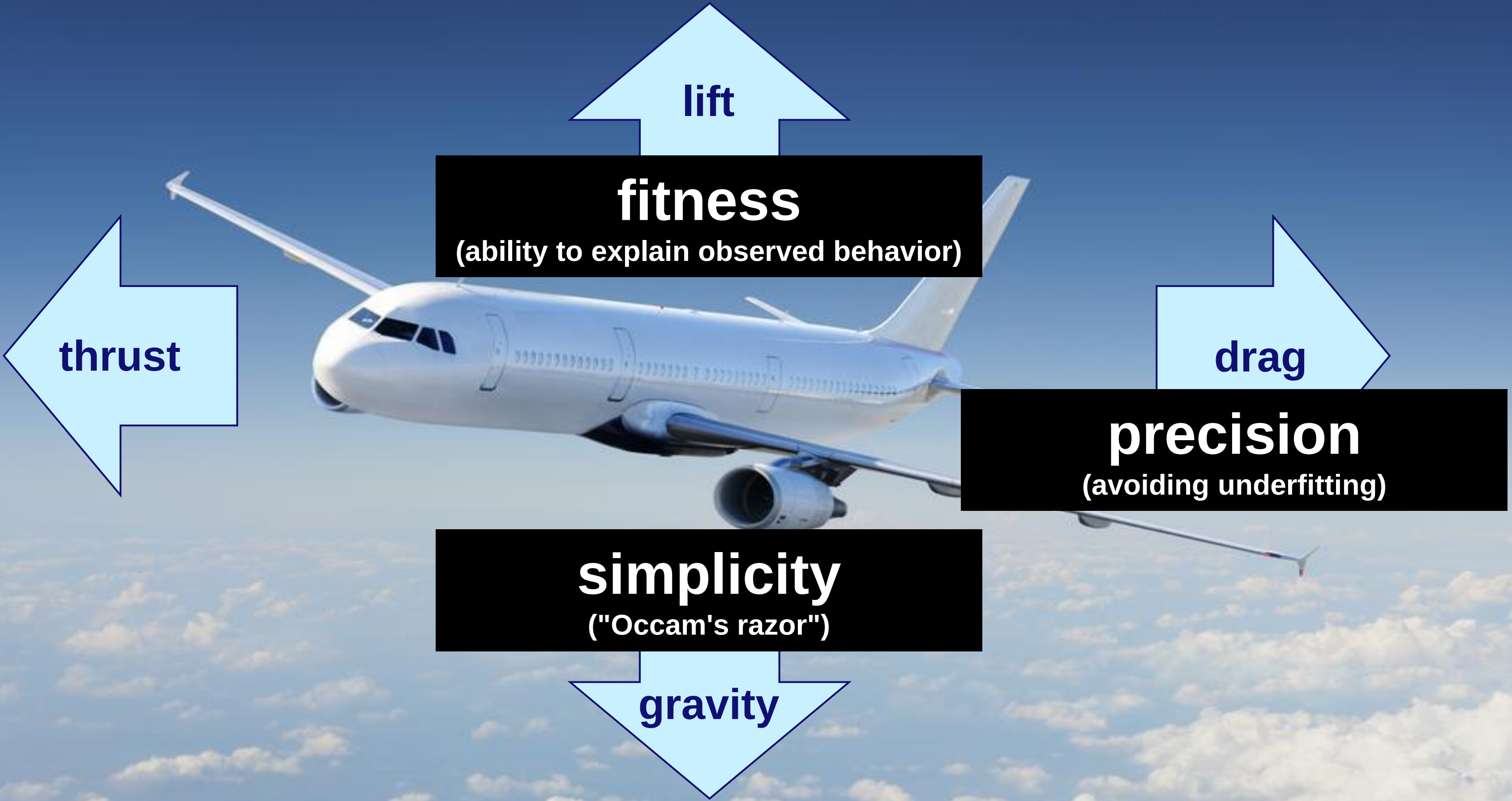
drag

simplicity

("Occam's razor")

gravity

Four Forces



lift

fitness

(ability to explain observed behavior)

thrust

drag

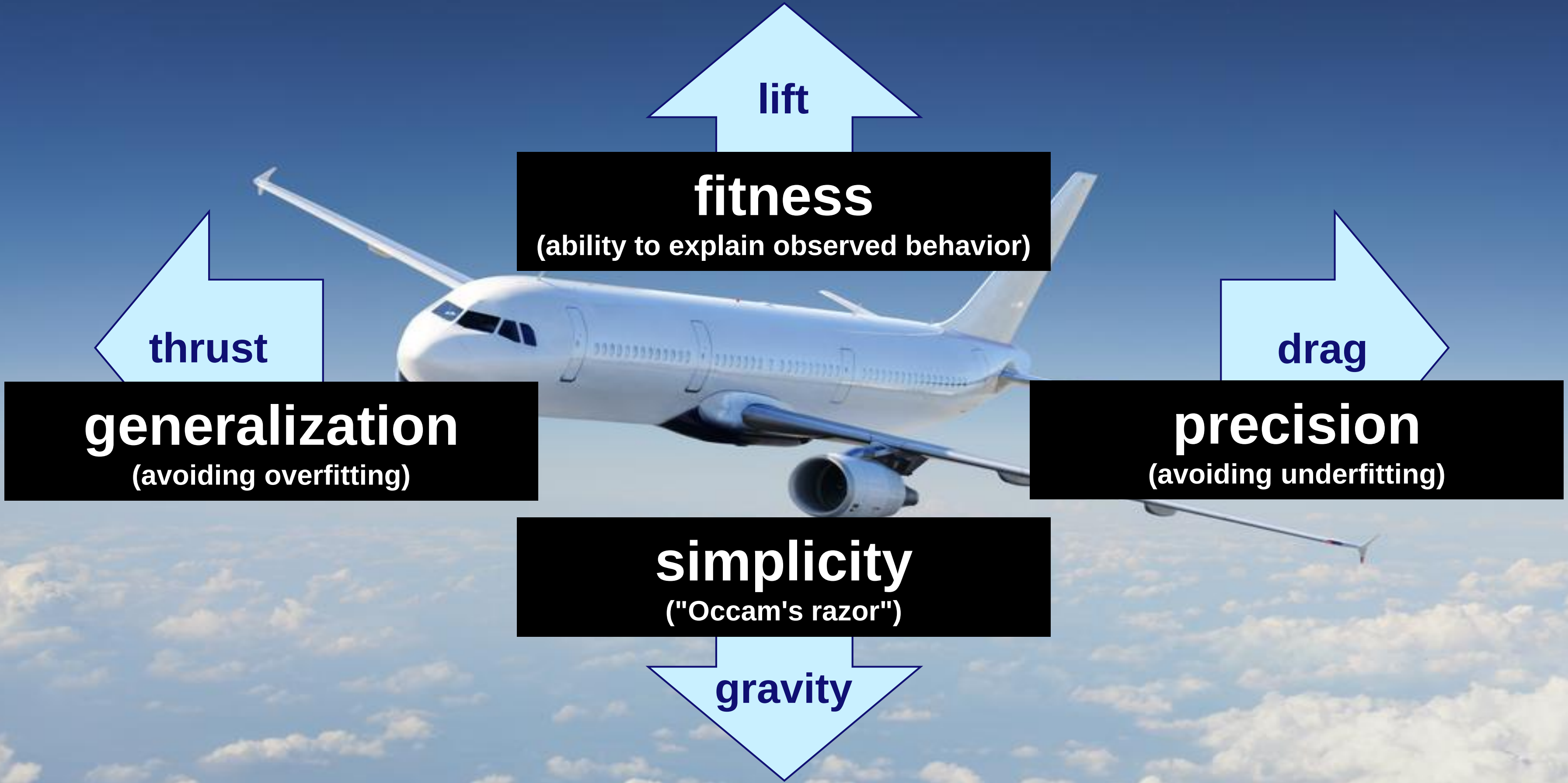
precision

(avoiding underfitting)

simplicity

("Occam's razor")

gravity



lift

fitness

(ability to explain observed behavior)

thrust

generalization

(avoiding overfitting)

drag

precision

(avoiding underfitting)

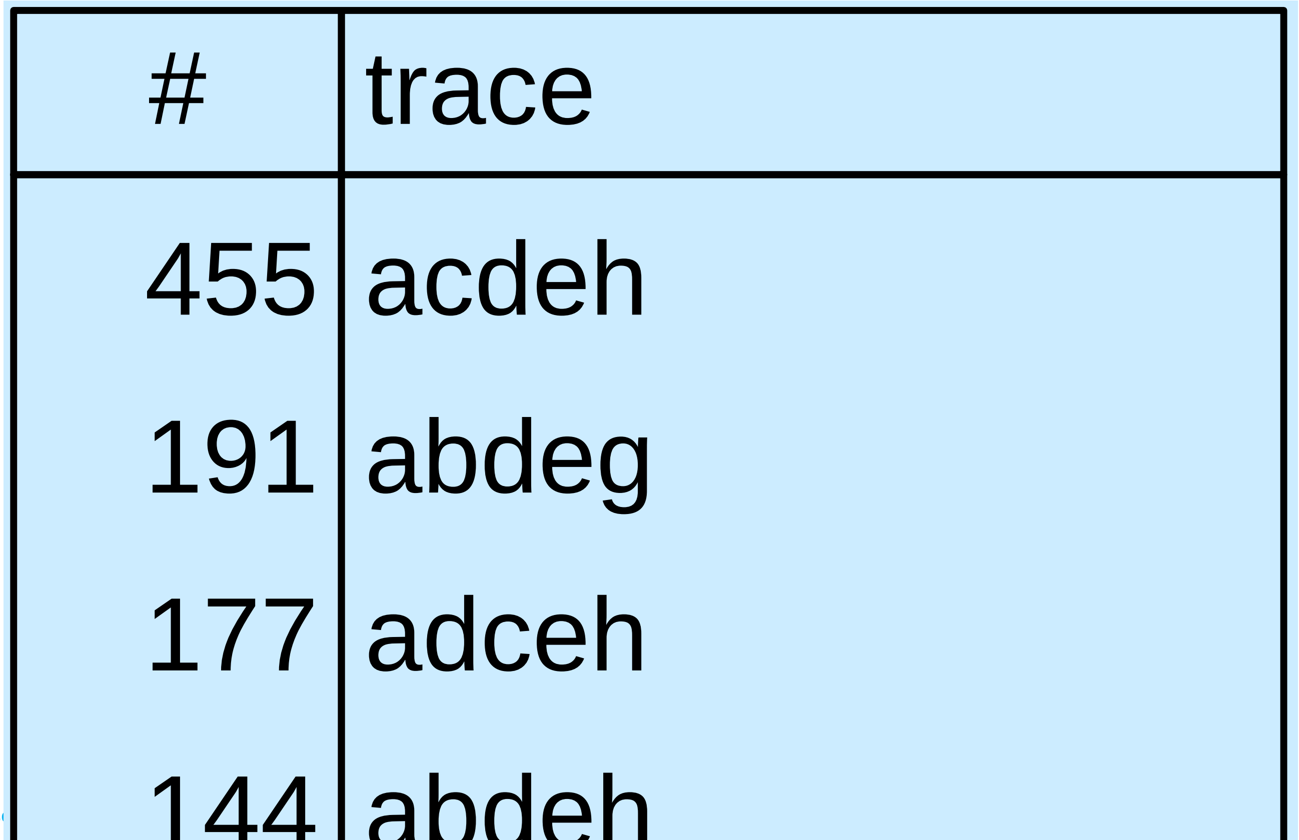
simplicity

("Occam's razor")

gravity

Four Forces

Example log

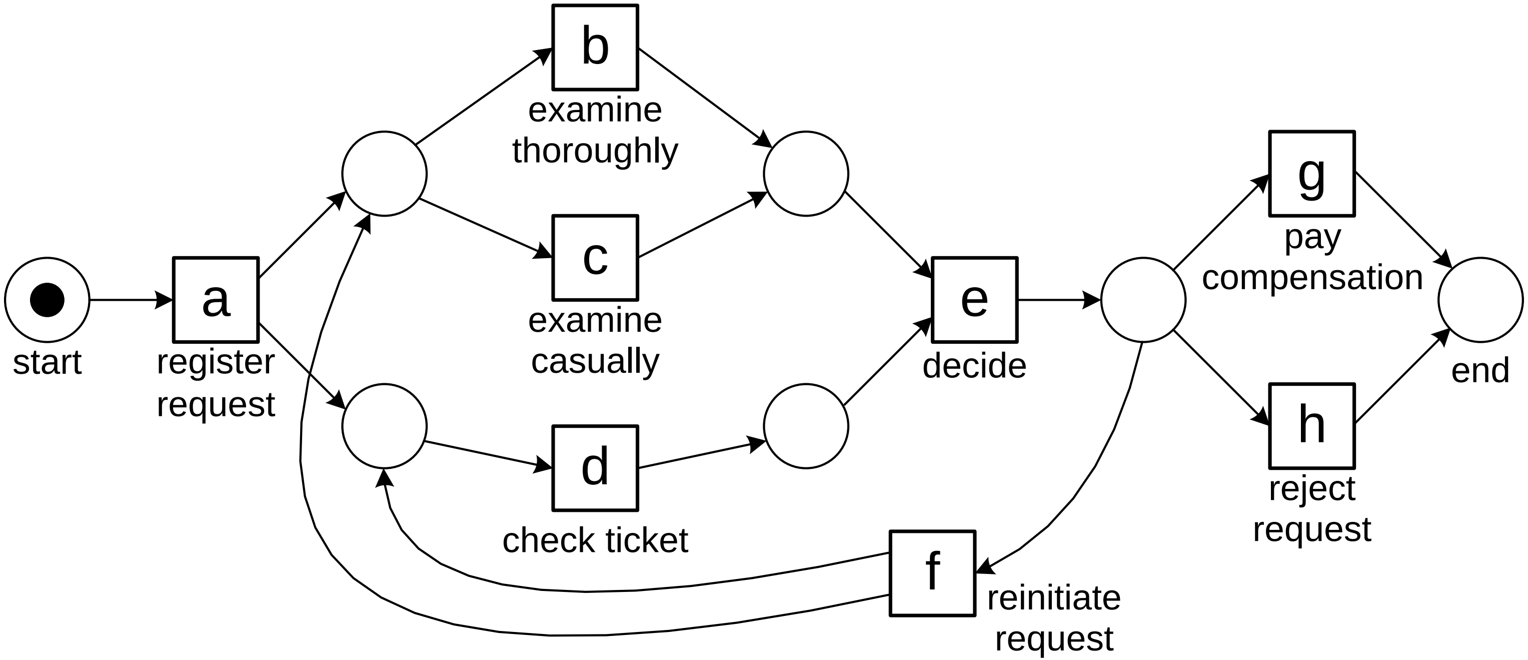


#	trace
455	acdeh
191	abdeg
177	adceh
144	abdeh

Exar

2	adcefdbeg
2	adcefbdefbdeg
1	adcefdbefbdeh
1	adbefbdefdbeg
1	adcefdbefcdefdbeg
1391	

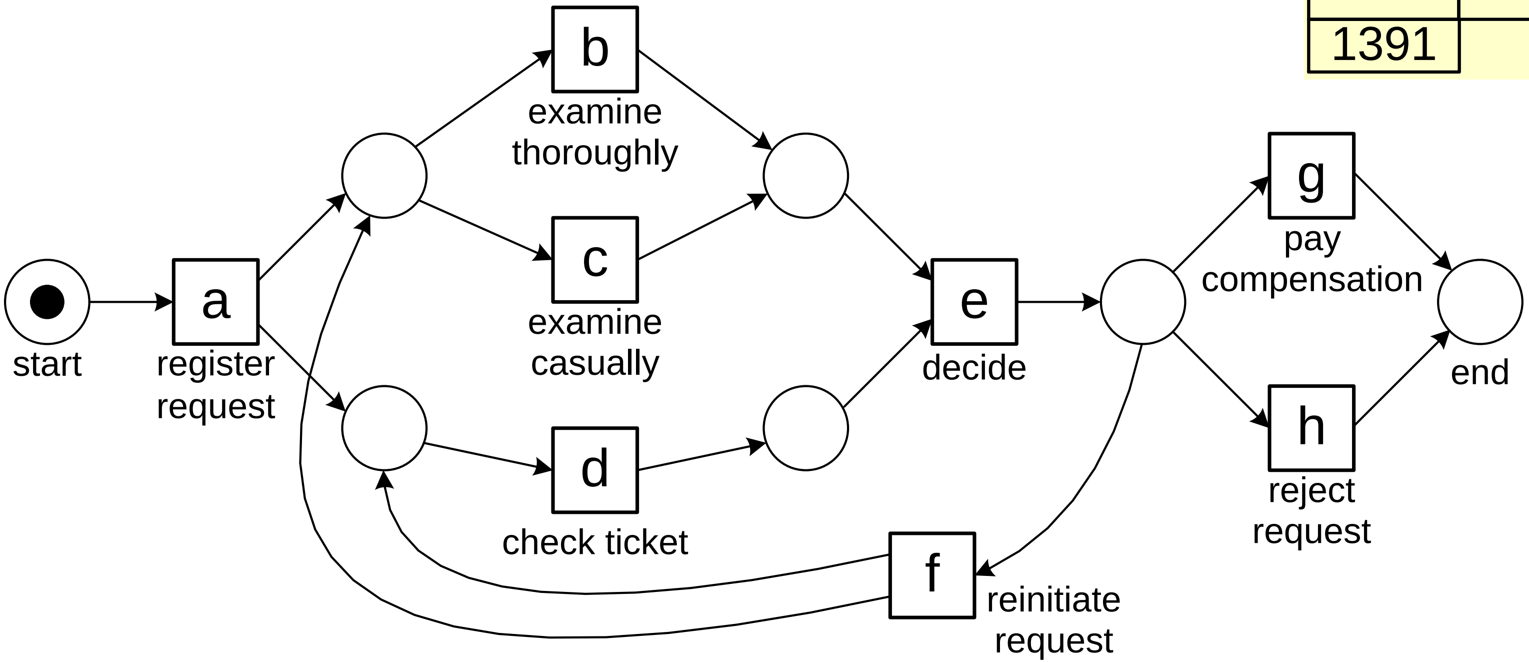
Model that seems to be OK …



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

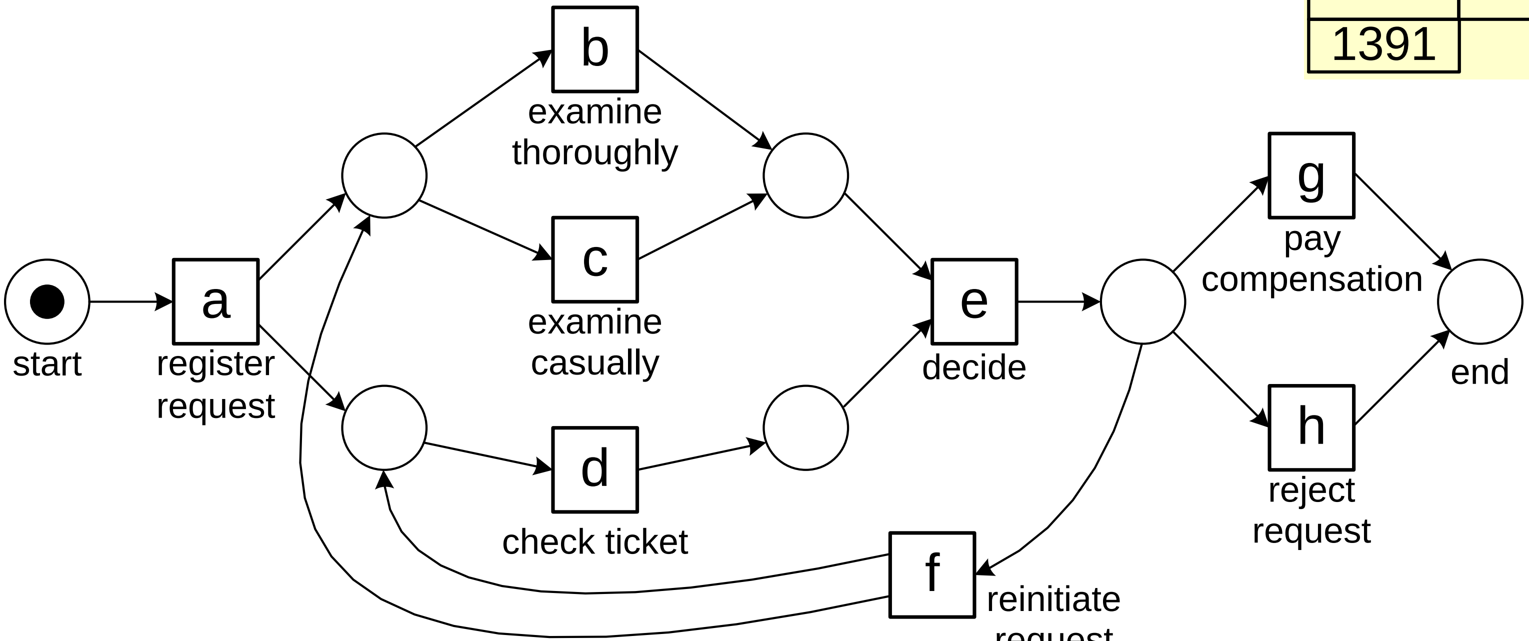
Model that seems to be OK …

#	trace
455	acdeh
191	abdeg
…	…
1	adcefdbefcdefdbeg
1391	



Model that seems to be OK …

#	trace
455	acdeh
191	abdeg
…	…
1	adcefdbefcdefdbeg
1391	

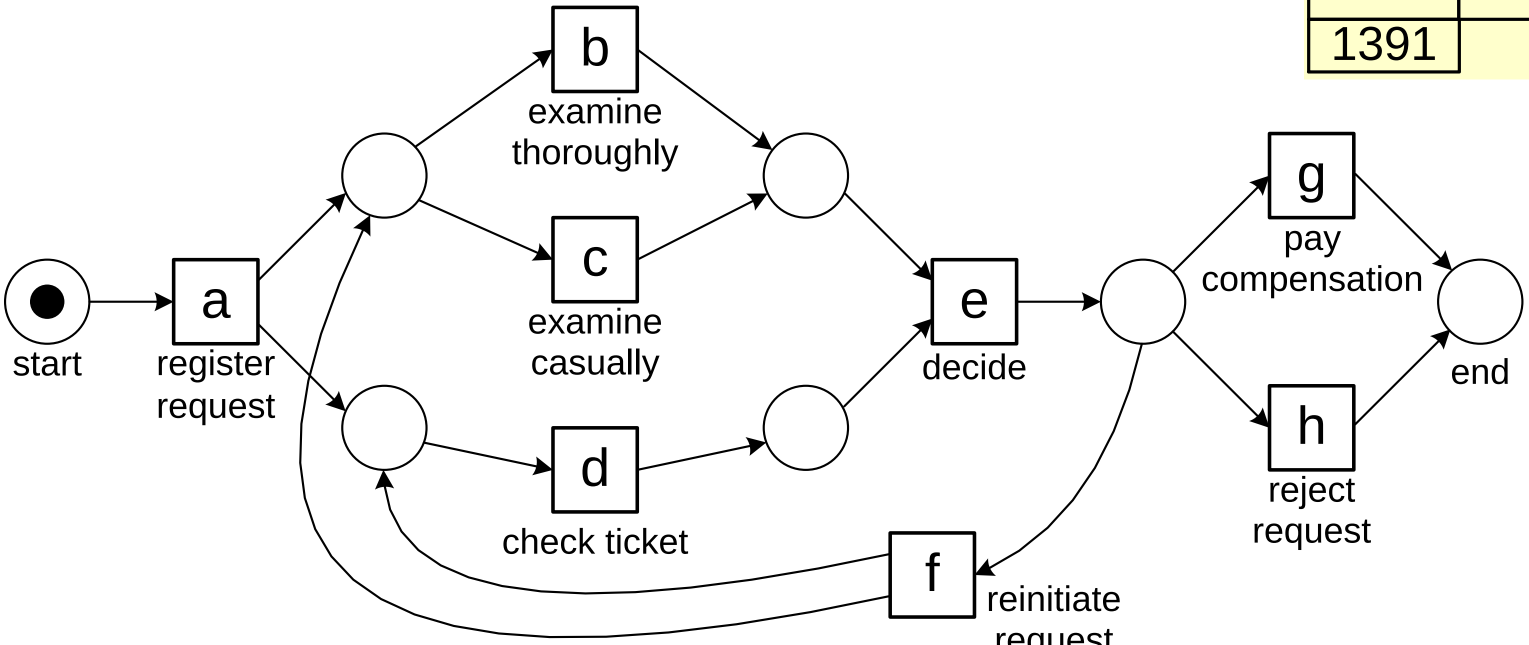


fitness
(observed behavior fits)

ermission & acknowledgements)

Model that seems to be OK …

#	trace
455	acdeh
191	abdeg
…	…
1	adcefdbefcdefdbeg
1391	

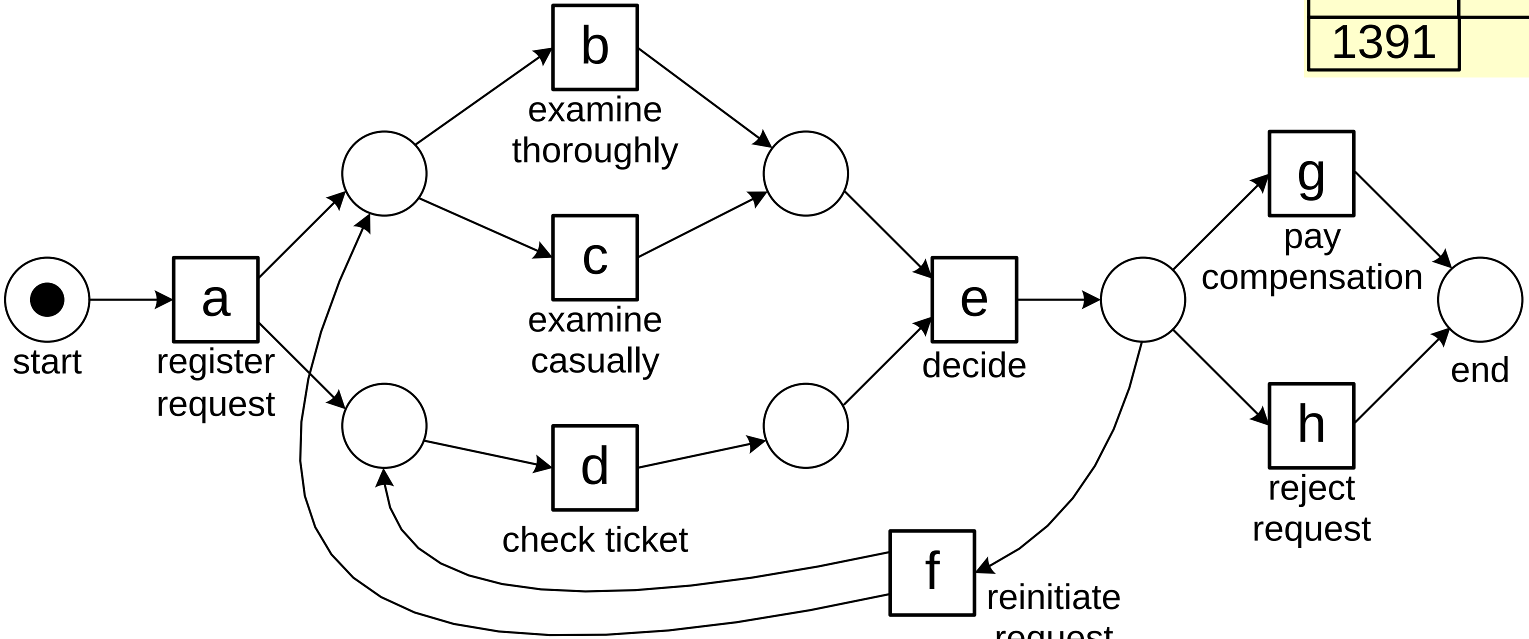


fitness
(observed behavior fits)

simplicity
("Occam's razor")

Model that seems to be OK …

#	trace
455	acdeh
191	abdeg
…	…
1	adcefdbefcdefdbeg
1391	



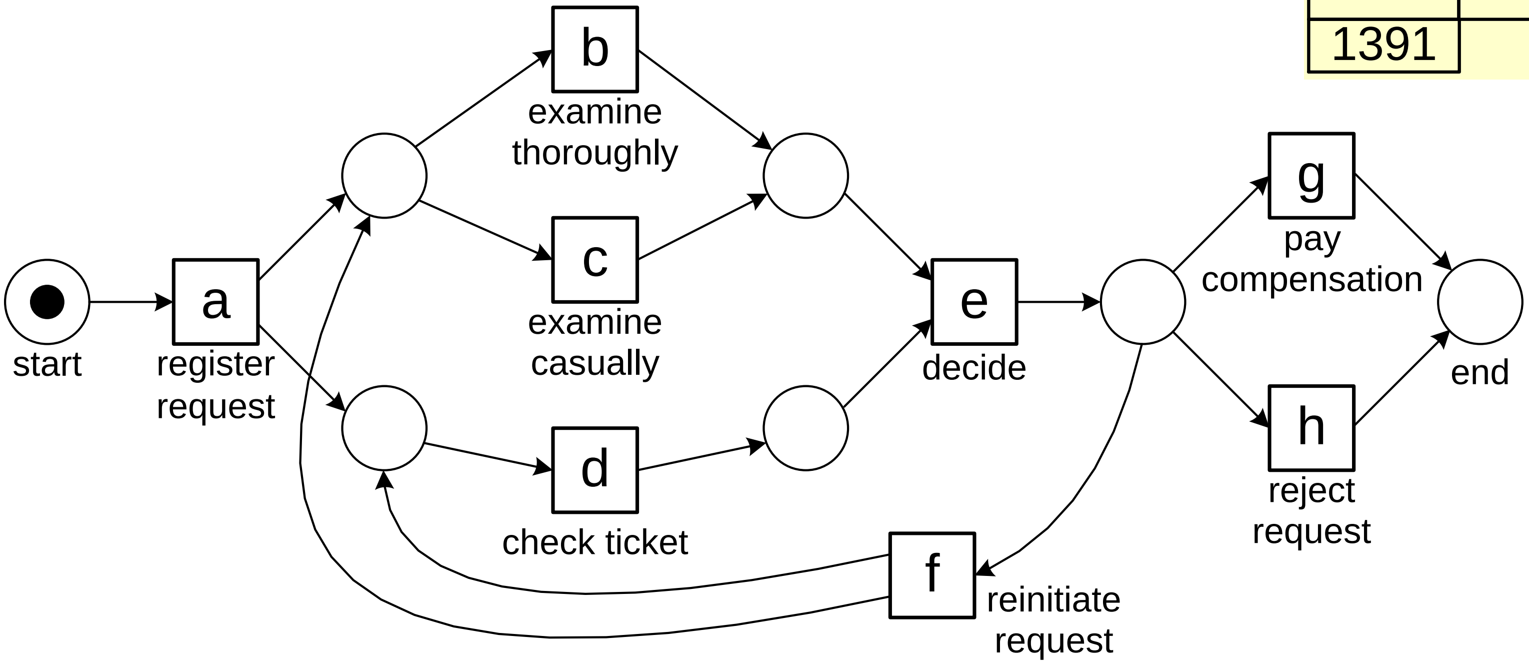
fitness
(observed behavior fits)

simplicity
("Occam's razor")

precision
(avoiding underfitting)

Model that seems to be OK …

#	trace
455	acdeh
191	abdeg
…	…
1	adcefdbefcdefdbeg
1391	



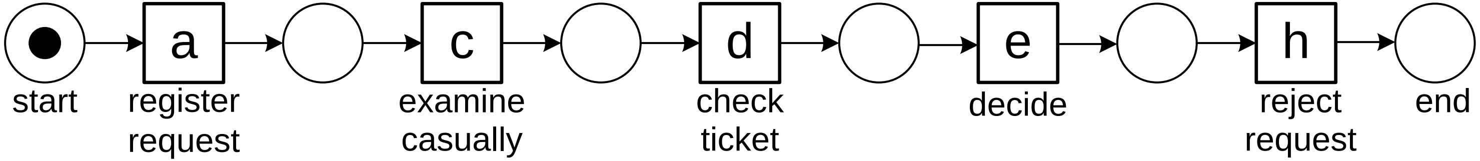
fitness
(observed behavior fits)

simplicity
("Occam's razor")

precision
(avoiding underfitting)

generalization
(avoiding overfitting)

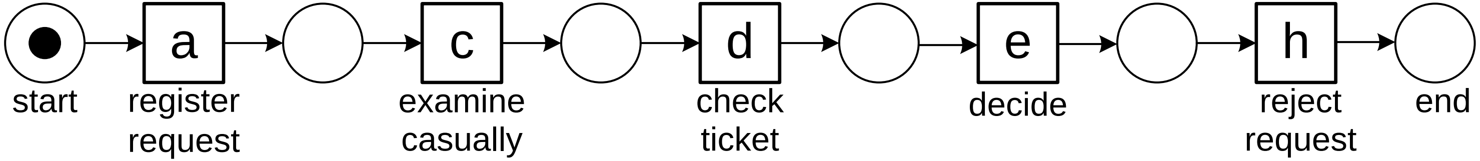
Non-fitting model



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

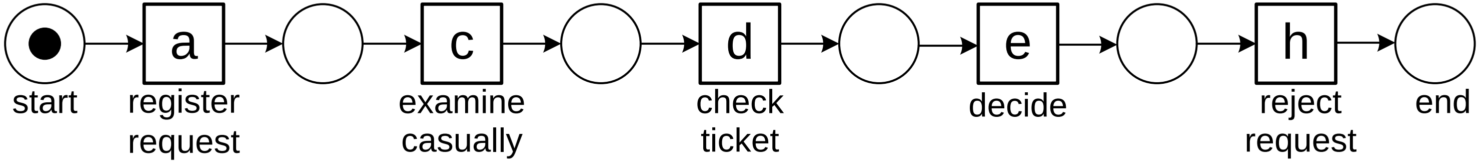
Non-fitting model

#	trace
455	acdeh
191	abdeg
...	...
1	adcefdbefcdefdbeg
1391	



Non-fitting model

#	trace
455	acdeh
191	abdeg
...	...
1	adcefdbefcdefdbeg
1391	



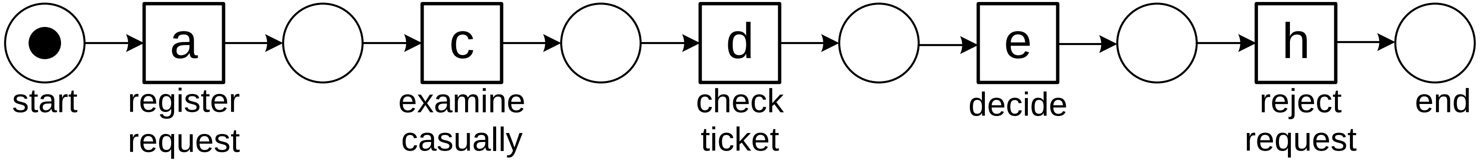
fitness

(observed behavior fits)

ermission & acknowledgements)

Non-fitting model

#	trace
455	acdeh
191	abdeg
...	...
1	adcefdbefcdefdbeg
1391	

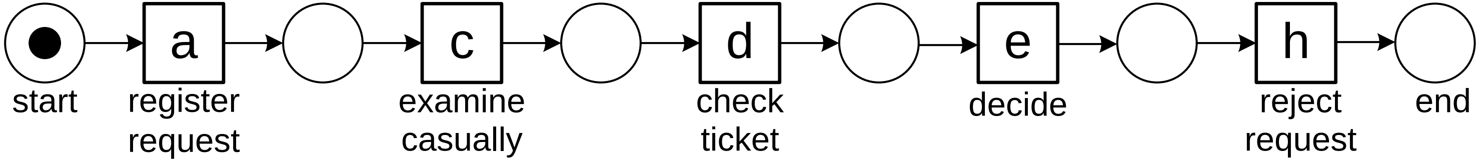


fitness
(observed behavior fits)

simplicity
("Occam's razor")

Non-fitting model

#	trace
455	acdeh
191	abdeg
...	...
1	adcefdbefcdefdbeg
1391	



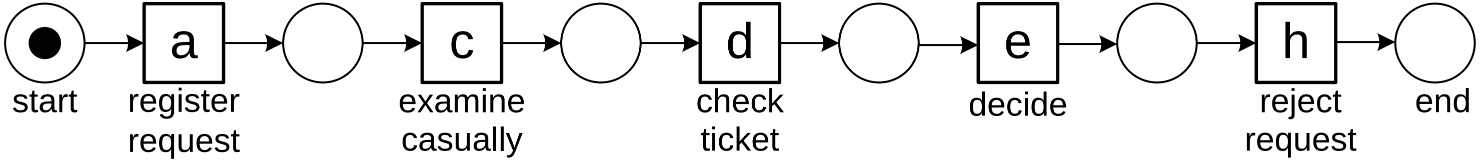
fitness
(observed behavior fits)

simplicity
("Occam's razor")

precision
(avoiding underfitting)

Non-fitting model

#	trace
455	acdeh
191	abdeg
...	...
1	adcefdbefcdefdbeg
1391	



fitness (observed behavior fits)

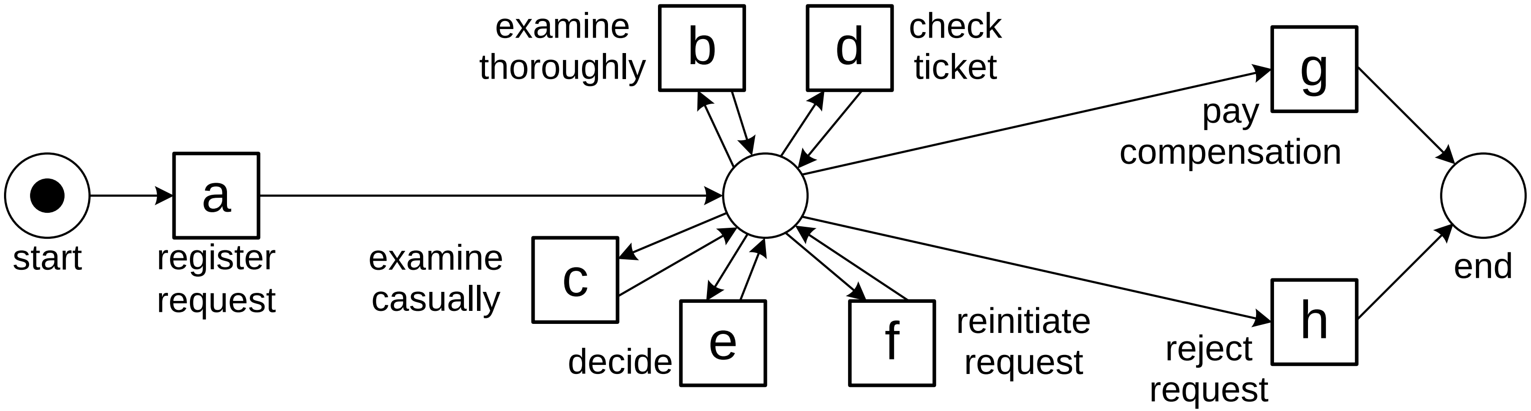
simplicity ("Occam's razor")

precision (avoiding underfitting)

generalization (avoiding overfitting)

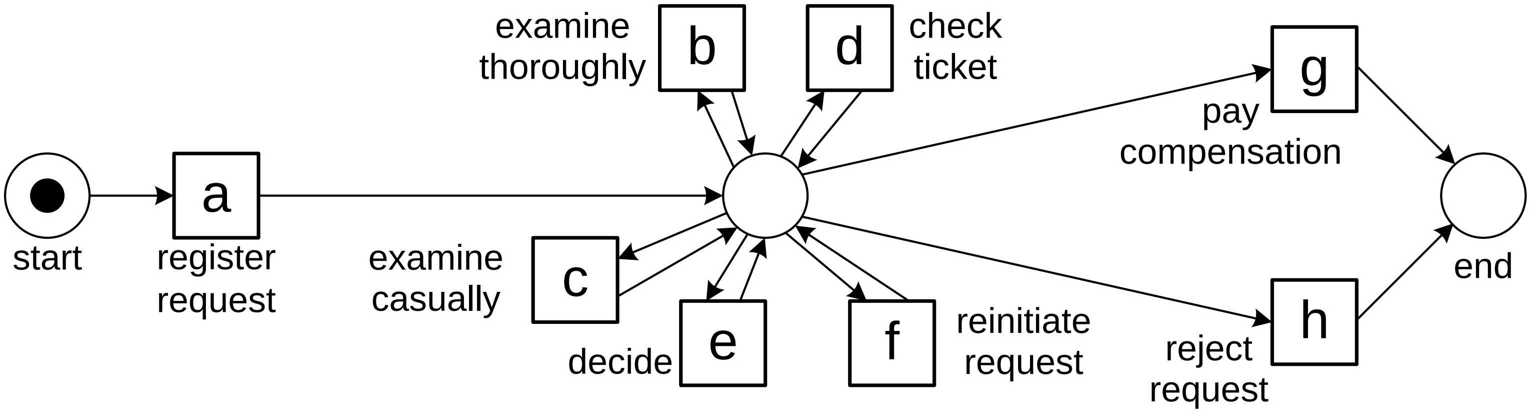
TU/e

Underfitting model



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

Underfitting model

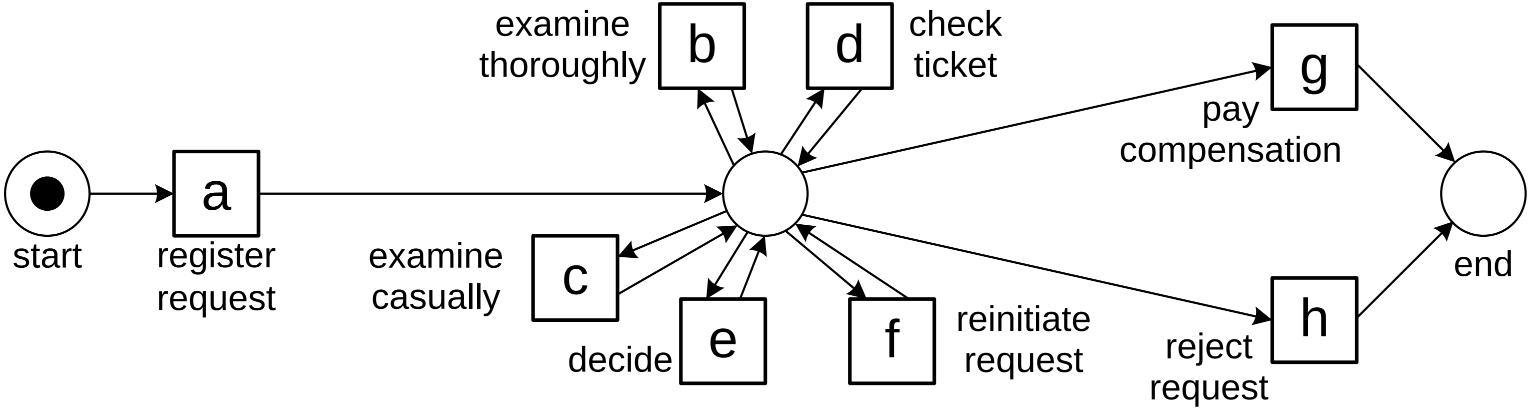


#	trace
455	acdeh
191	abdeg
177	adceh
144	abdeh
111	acdeg
82	adceg
56	adbeh
47	acdefdbeh
38	adbeg
33	acdefbdeh
14	acdefbdeg
11	acdefdbeg
9	adcefcdeh
8	adcefdbeh
5	adcefbdeg
3	acdefbdefdbeg
2	adcefdbeg
2	adcefbdefbdeg
1	adcefdbefbdeh
1	adbefbdefdbeg
1	adcefdbefcdefdbeg
1391	

fitness
(observed behavior fits)

ermission & acknowledgements)

Underfitting model

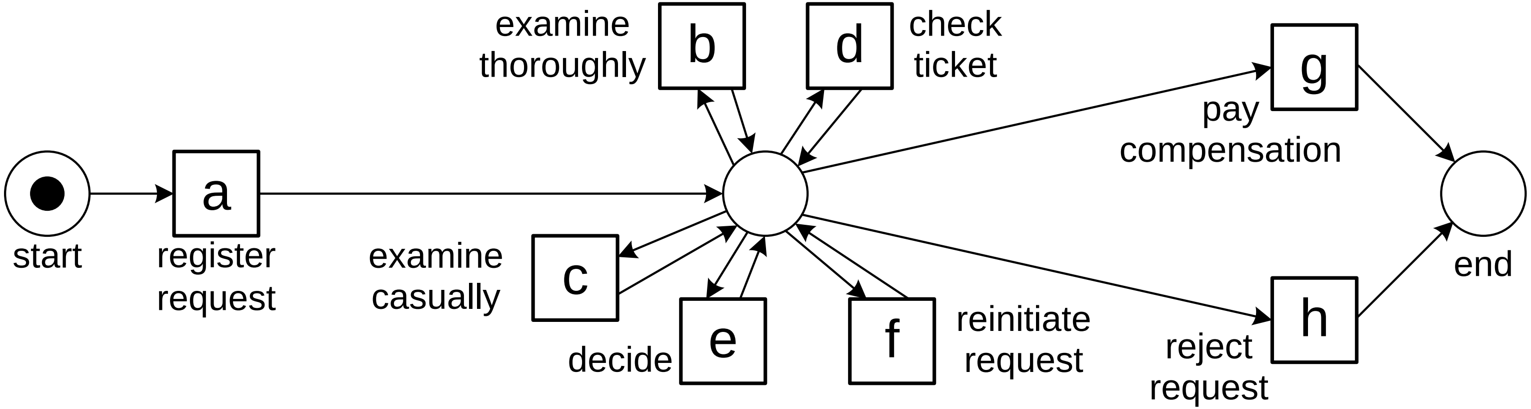


#	trace
455	acdeh
191	abdeg
177	adceh
144	abdeh
111	acdeg
82	adceg
56	adbeh
47	acdefdbeh
38	adbeg
33	acdefbdeh
14	acdefbdeg
11	acdefdbeg
9	adcefcdeh
8	adcefdbeh
5	adcefbdeg
3	acdefbdefdbeg
2	adcefdbeg
2	adcefbdefbdeg
1	adcefdbefbdeh
1	adbefbdefdbeg
1	adcefdbefcdefdbeg
1391	

fitness
(observed behavior fits)

simplicity
("Occam's razor")

Underfitting model



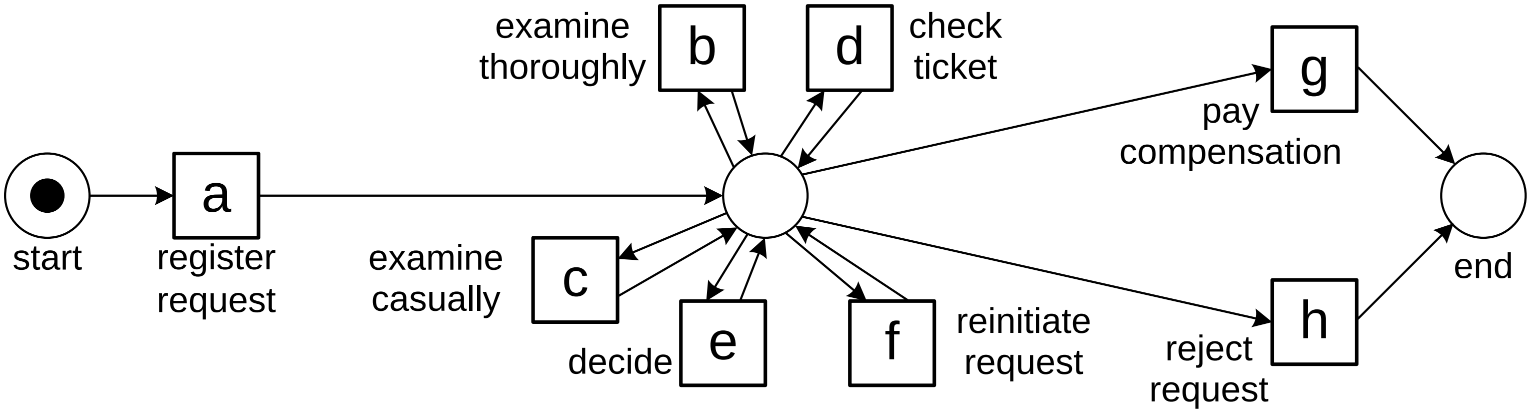
#	trace
455	acdeh
191	abdeg
177	adceh
144	abdeh
111	acdeg
82	adceg
56	adbeh
47	acdefdbeh
38	adbeg
33	acdefbdeh
14	acdefbdeg
11	acdefdbeg
9	adcefcdeh
8	adcefdbeh
5	adcefbdeg
3	acdefbdefdbeg
2	adcefdbeg
2	adcefbdefbdeg
1	adcefdbefbdeh
1	adbefbdefdbeg
1	adcefdbefcdefdbeg
1391	

fitness
(observed behavior fits)

simplicity
("Occam's razor")

precision
(avoiding underfitting)

Underfitting model



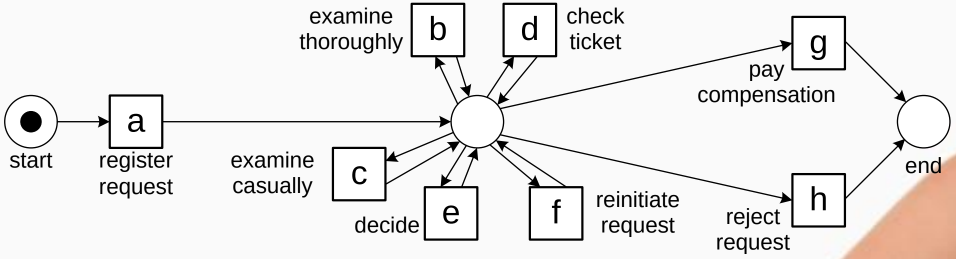
#	trace
455	acdeh
191	abdeg
177	adceh
144	abdeh
111	acdeg
82	adceg
56	adbeh
47	acdefdbeh
38	adbeg
33	acdefbdeh
14	acdefbdeg
11	acdefdbeg
9	adcefcdeh
8	adcefdbeh
5	adcefbdeg
3	acdefbdefdbeg
2	adcefdbeg
2	adcefbdefbdeg
1	adcefdbefbdeh
1	adbefbdefdbeg
1	adcefdbefcdefdbeg
1391	

fitness
(observed behavior fits)

simplicity
("Occam's razor")

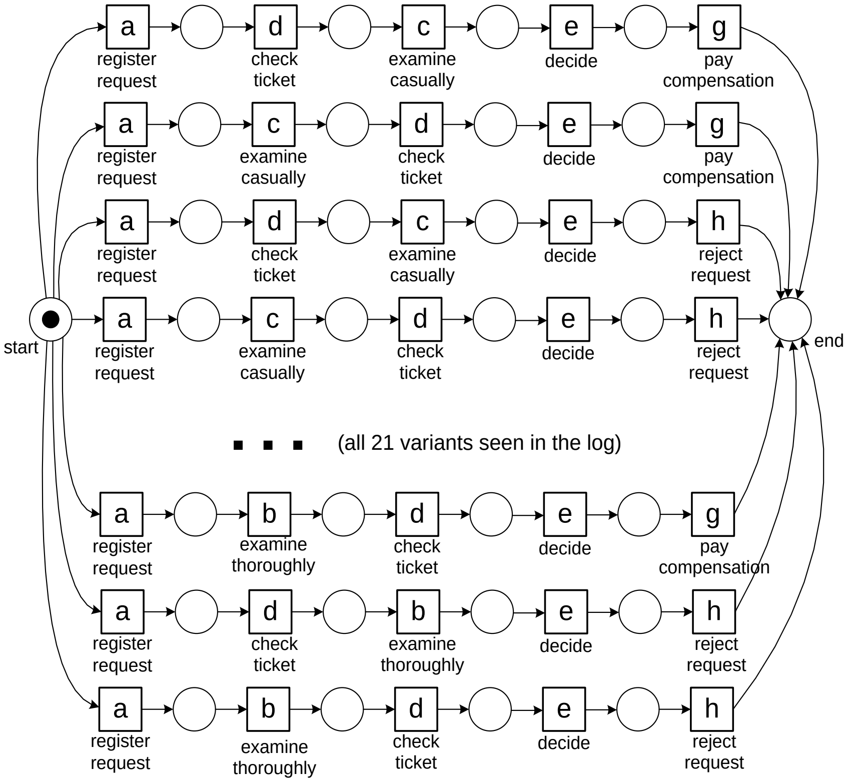
precision
(avoiding underfitting)

generalization
(avoiding overfitting)



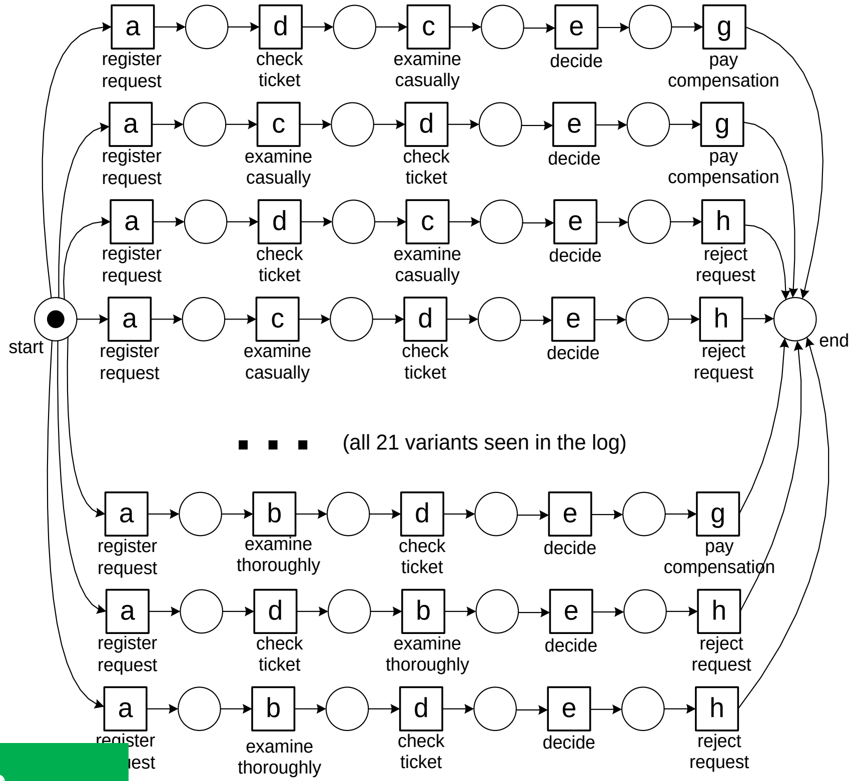
underfitting

Overfitting model



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

Overfitting model



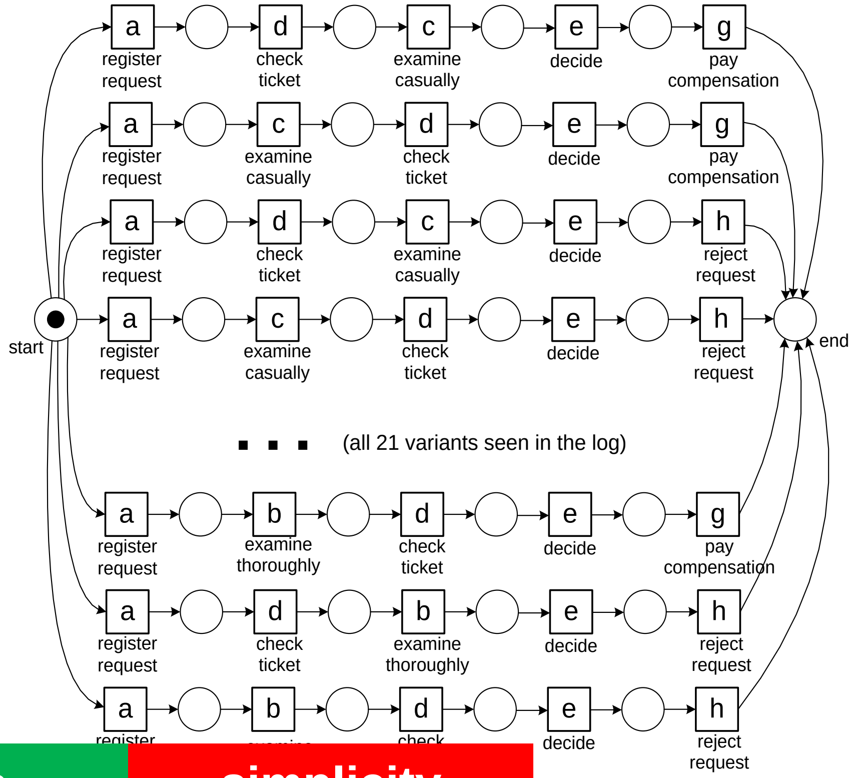
#	trace
455	acdeh
191	abdeg
177	adceh
144	abdeh
111	acdeg
82	adceg
56	adbeh
47	acdefdbeh
38	adbeg
33	acdefbdeh
14	acdefbdeg
11	acdefdbeg
9	adcefcdeh
8	adcefdbeh
5	adcefbdeg
3	acdefbdefdbeg
2	adcefdbeg
2	adcefbdefbdeg
1	adcefdbefbdeh
1	adbefbdefdbeg
1	adcefdbefcdefdbeg
1391	

fitness

(observed behavior fits)

ermission & acknowledgements)

Overfitting model



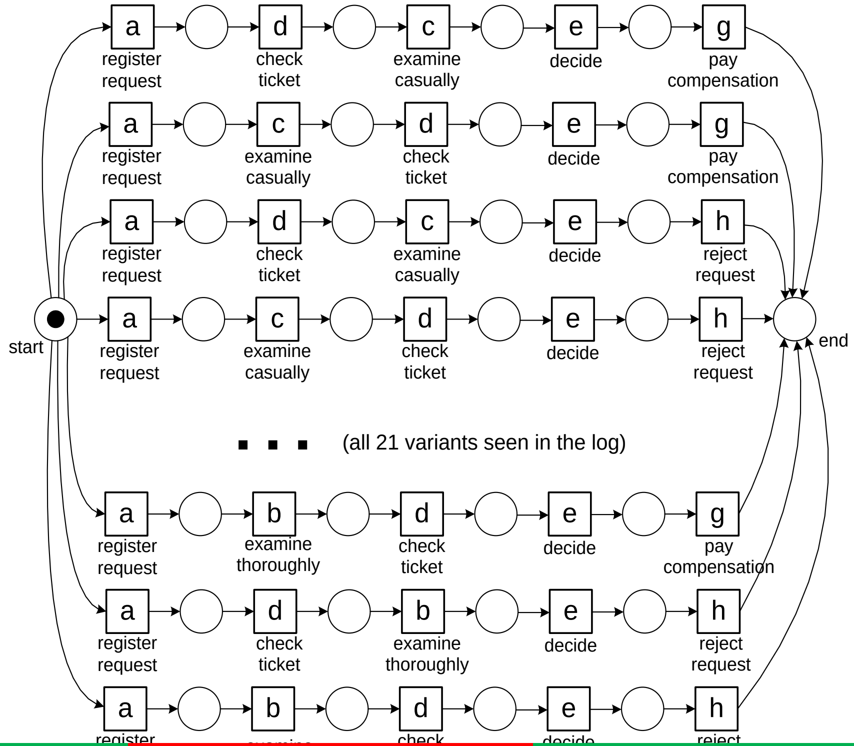
(all 21 variants seen in the log)

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

fitness
(observed behavior fits)

simplicity
("Occam's razor")

Overfitting model


a register request → d check ticket → c examine casually → e decide → g pay compensation
a register request → c examine casually → d check ticket → e decide → g pay compensation
a register request → d check ticket → c examine casually → e decide → h reject request
a register request → c examine casually → d check ticket → e decide → h reject request
start
end
. . . (all 21 variants seen in the log)
a register request → b examine thoroughly → d check ticket → e decide → g pay compensation
a register request → d check ticket → b examine thoroughly → e decide → h reject request
a register → b examine → d check → e decide → h reject

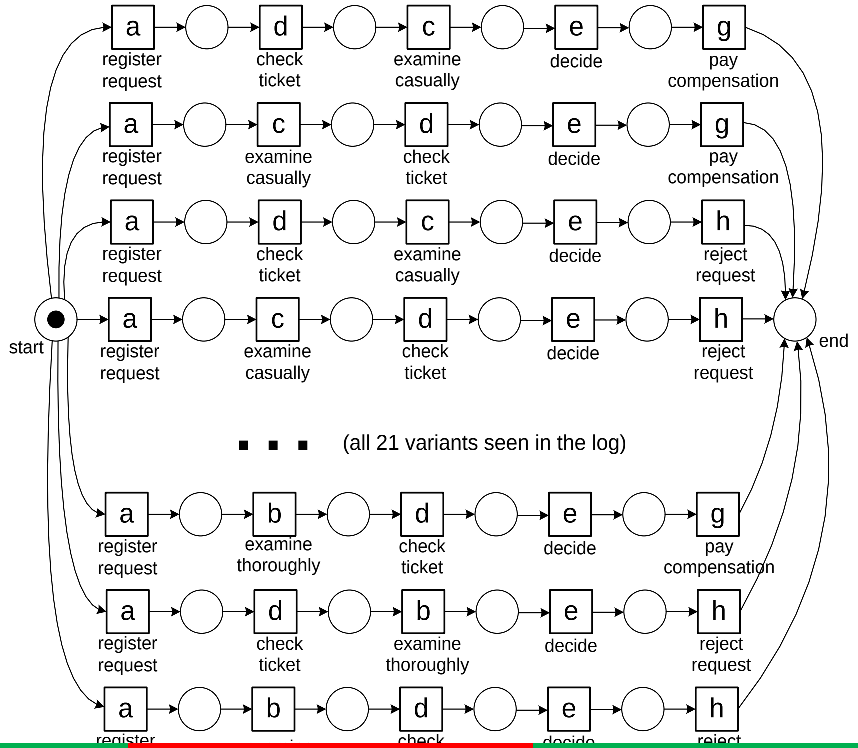
#	trace
455	acdeh
191	abdeg
177	adceh
144	abdeh
111	acdeg
82	adceg
56	adbeh
47	acdefdbeh
38	adbeg
33	acdefbdeh
14	acdefbdeg
11	acdefdbeg
9	adcefcdeh
8	adcefdbeh
5	adcefbdeg
3	acdefbdefdbeg
2	adcefdbeg
2	adcefbdefbdeg
1	adcefdbefbdeh
1	adbefbdefdbeg
1	adcefdbefcdefdbeg
1391	

fitness
(observed behavior fits)

simplicity
("Occam's razor")

precision
(avoiding underfitting)

Overfitting model



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

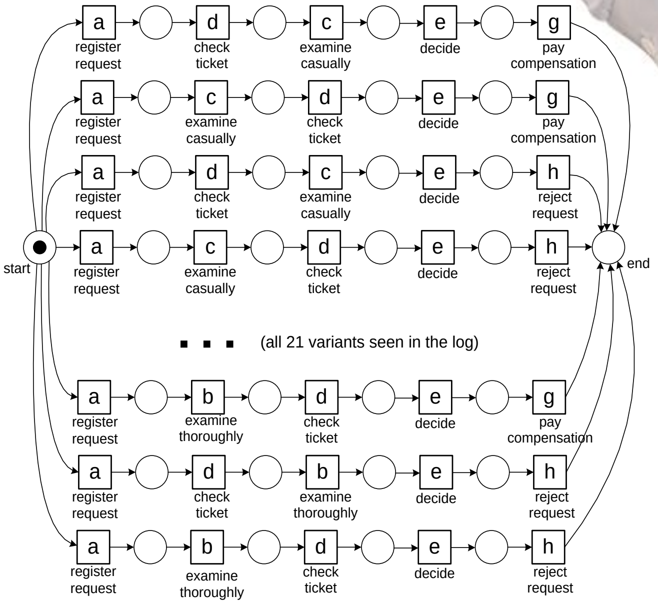
fitness
(observed behavior fits)

simplicity
("Occam's razor")

precision
(avoiding underfitting)

generalization
(avoiding overfitting)

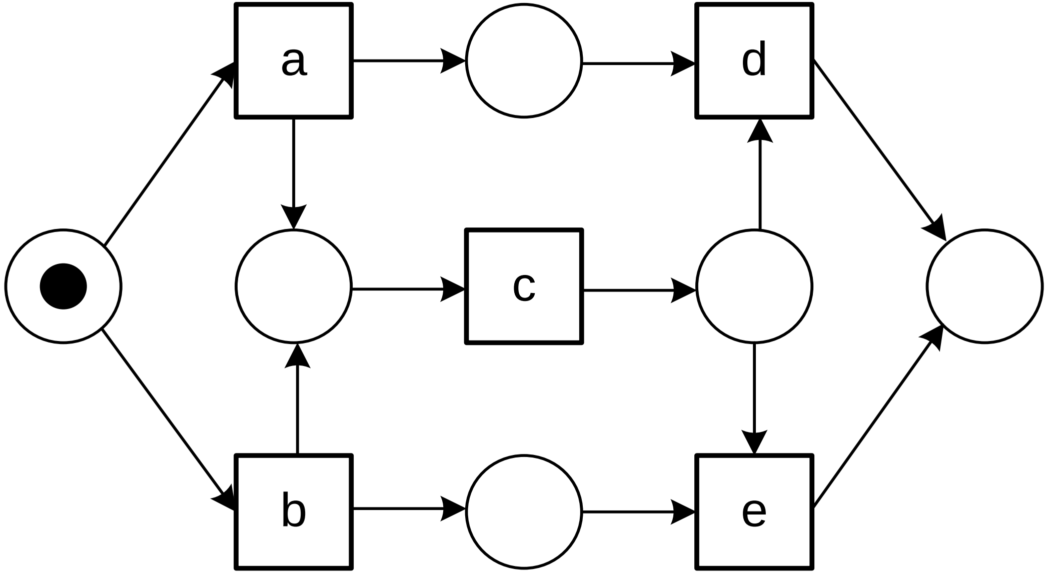
overfitting


a register request → d check ticket → c examine casually → e decide → g pay compensation
a register request → c examine casually → d check ticket → e decide → g pay compensation
a register request → d check ticket → c examine casually → e decide → h reject request
a register request → c examine casually → d check ticket → e decide → h reject request
start
end
(all 21 variants seen in the log)
a register request → b examine thoroughly → d check ticket → e decide → g pay compensation
a register request → d check ticket → b examine thoroughly → e decide → h reject request
a register request → b examine thoroughly → d check ticket → e decide → h reject request

Fitness: good or bad?

$\langle a,c,e \rangle^{77}$

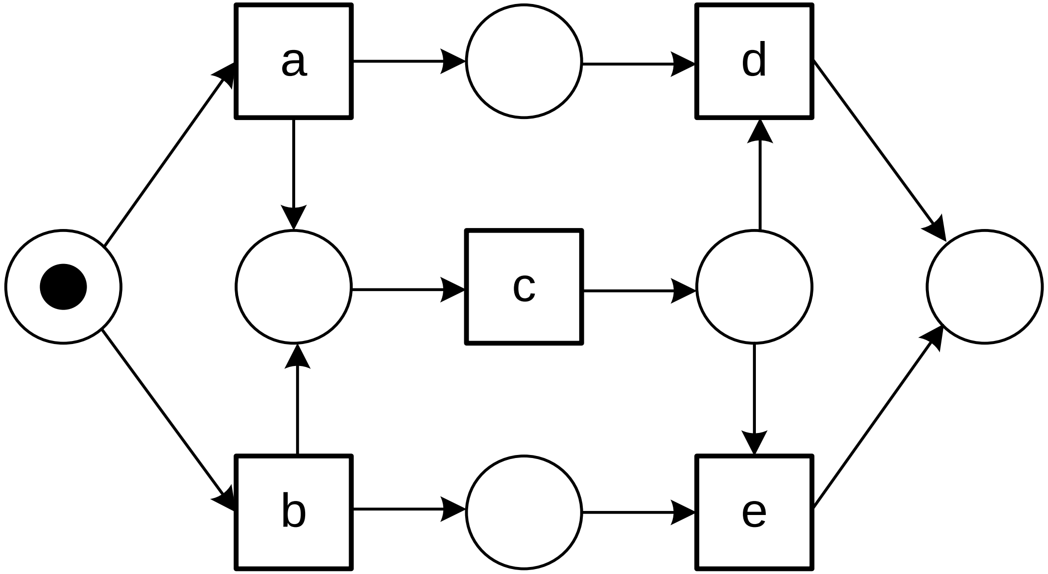
$\langle b,c,d \rangle^{92}$



Fitness: bad!

$\langle a,c,e\rangle^{77}$

$\langle b,c,d\rangle^{92}$

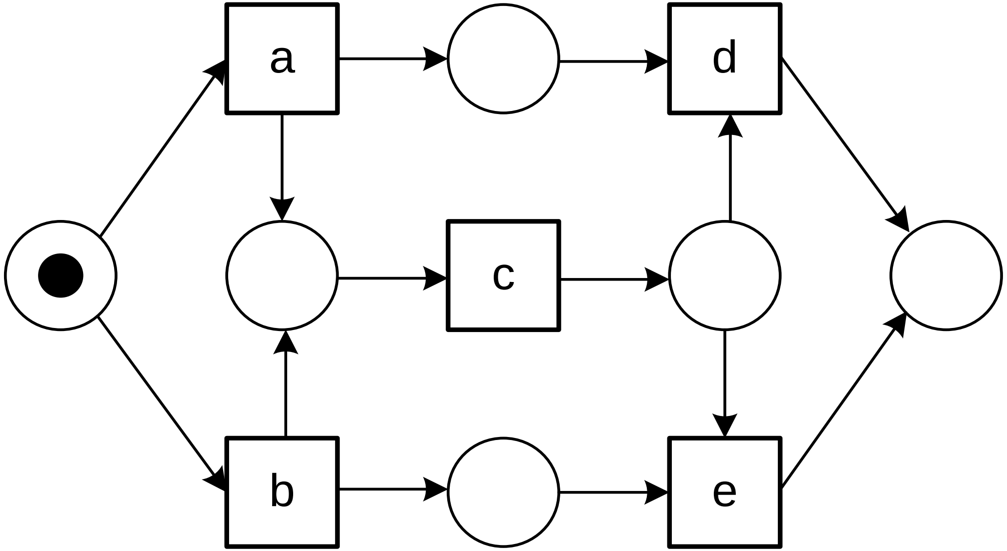


both traces do not fit …

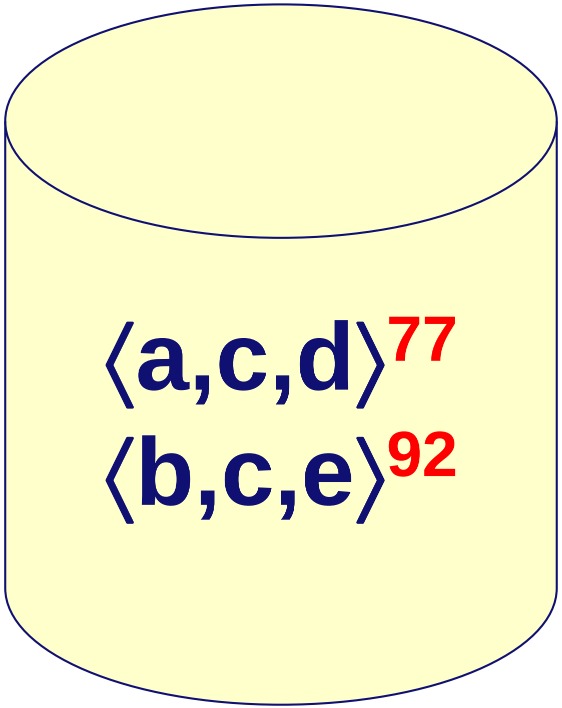
Precision: good or bad?

$\langle a,c,d\rangle^{77}$

$\langle b,c,e\rangle^{92}$

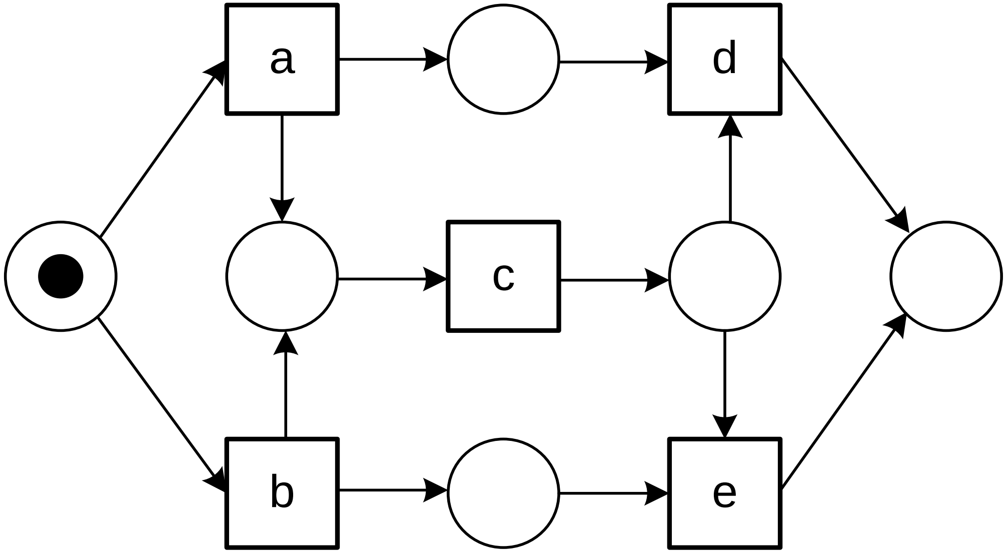


Precision: good!



$\langle a,c,d \rangle^{77}$

$\langle b,c,e \rangle^{92}$

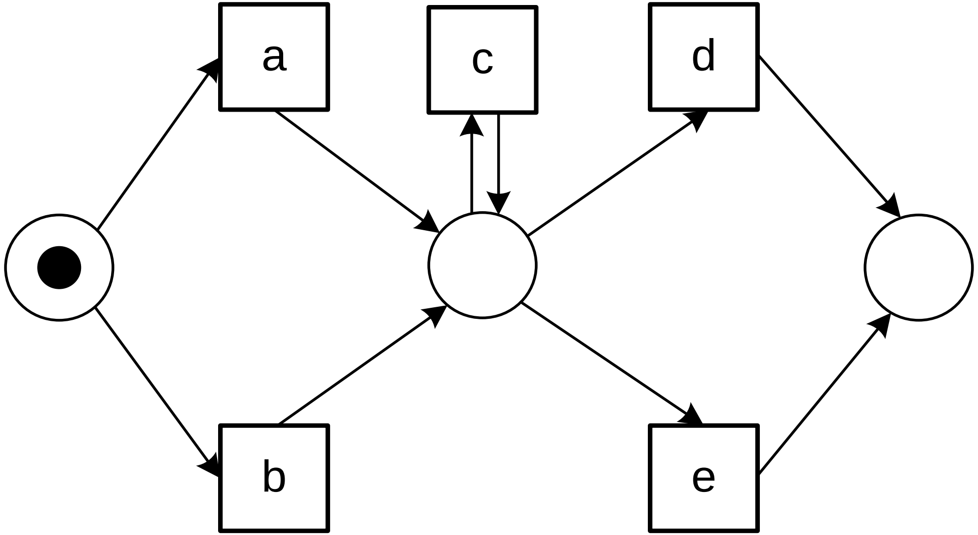


not underfitting…

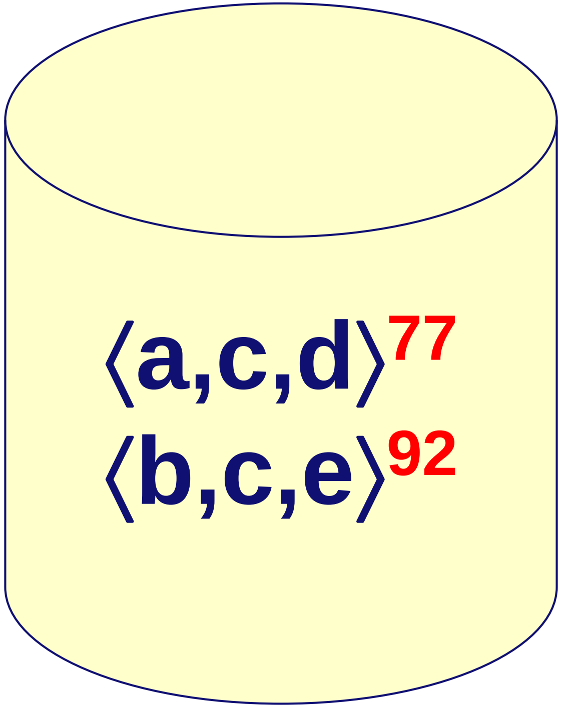
Precision: good or bad?

$\langle a,c,d \rangle^{77}$

$\langle b,c,e \rangle^{92}$

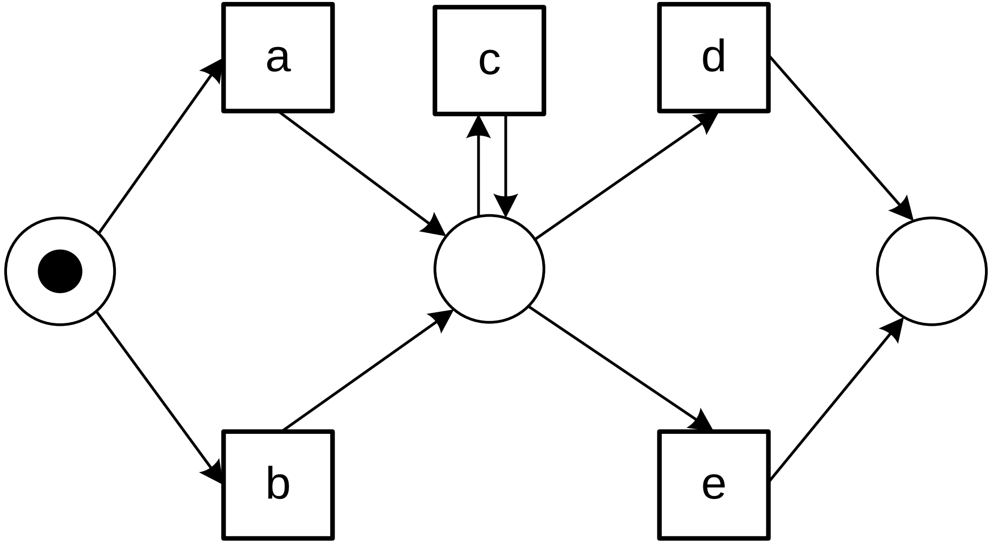


Precision: bad!



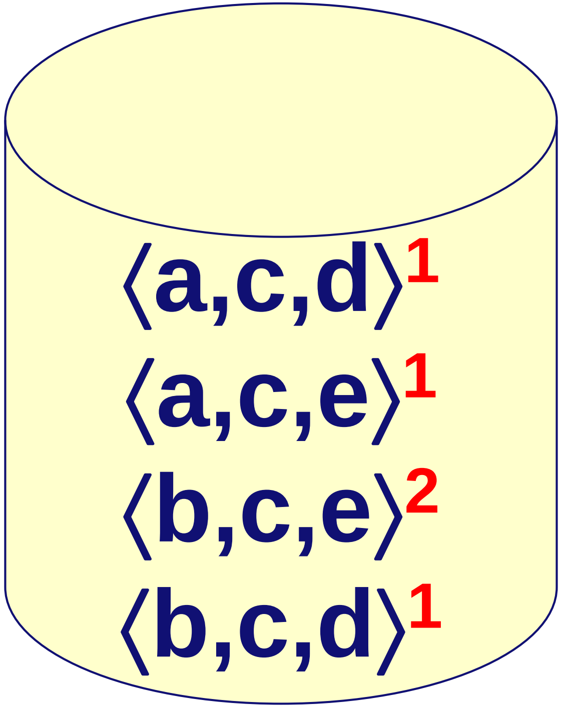
$\langle a,c,d\rangle^{77}$

$\langle b,c,e\rangle^{92}$



underfitting (allows for highly unlikely behavior) …

Generalization: good or bad?

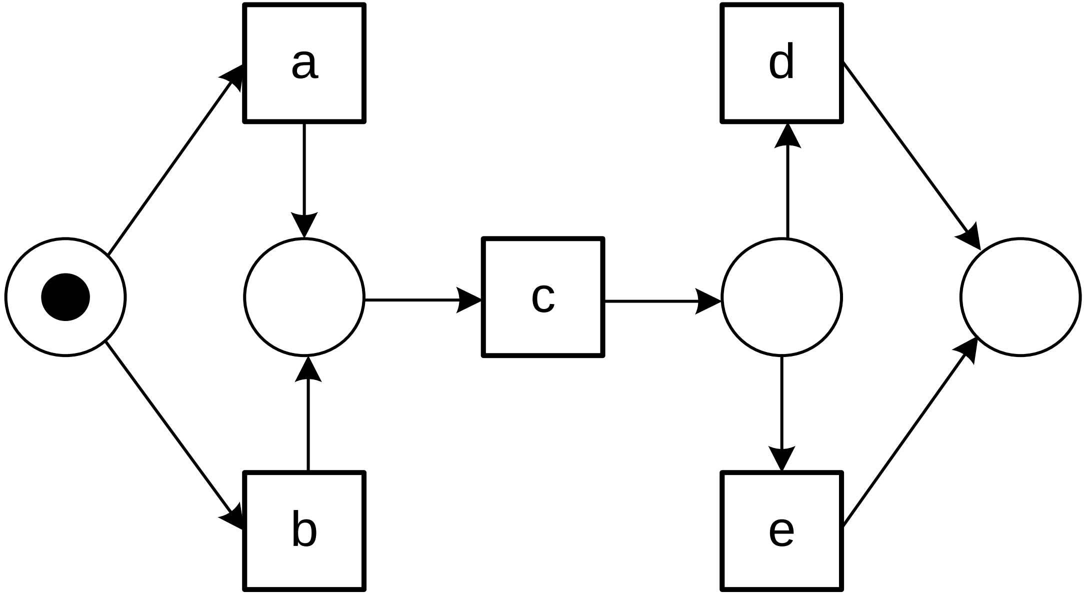


$\langle a,c,d\rangle^{1}$

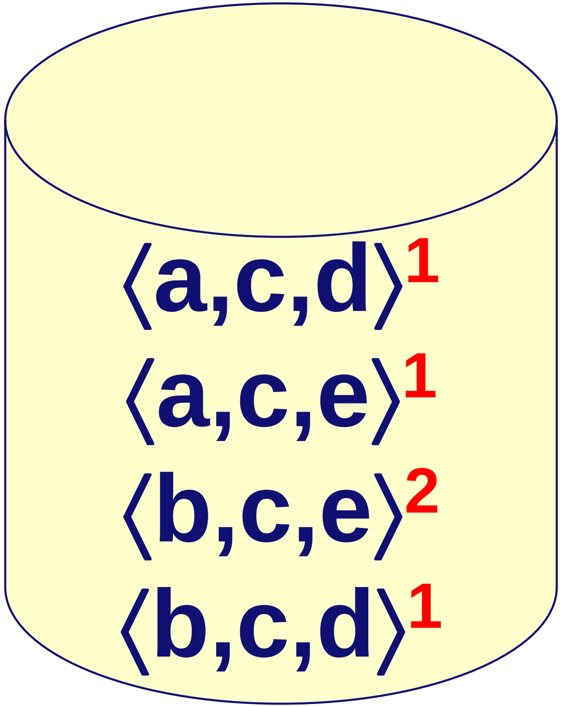
$\langle a,c,e\rangle^{1}$

$\langle b,c,e\rangle^{2}$

$\langle b,c,d\rangle^{1}$



Generalization: bad!

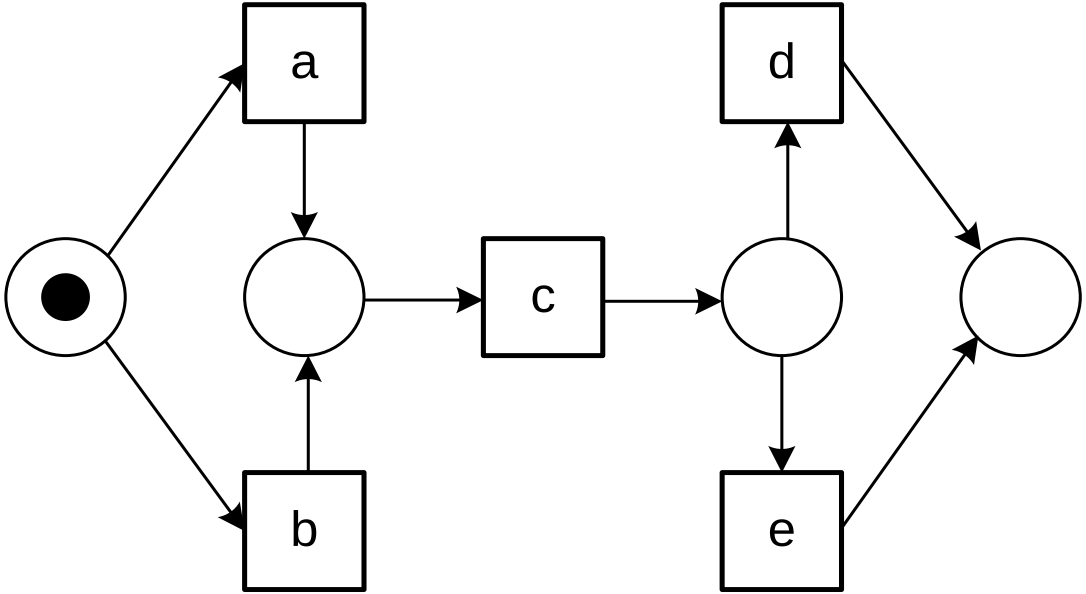


$\langle a,c,d\rangle^{1}$

$\langle a,c,e\rangle^{1}$

$\langle b,c,e\rangle^{2}$

$\langle b,c,d\rangle^{1}$



risk of overfitting on 5 example traces …

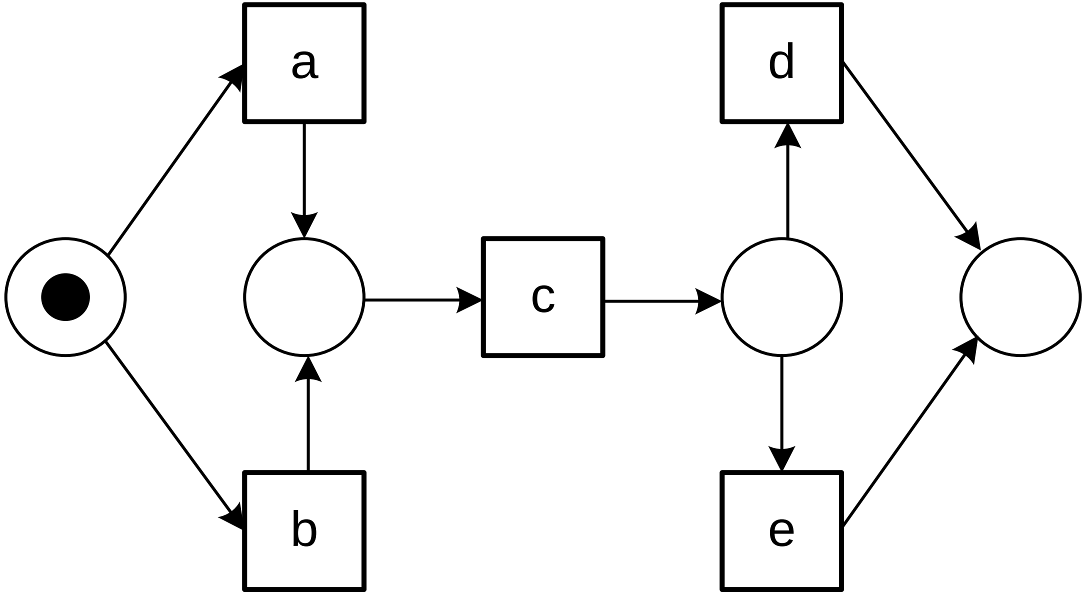
Generalization: good or bad?

$\langle a,c,d\rangle^{99}$

$\langle a,c,e\rangle^{50}$

$\langle b,c,e\rangle^{85}$

$\langle b,c,d\rangle^{48}$



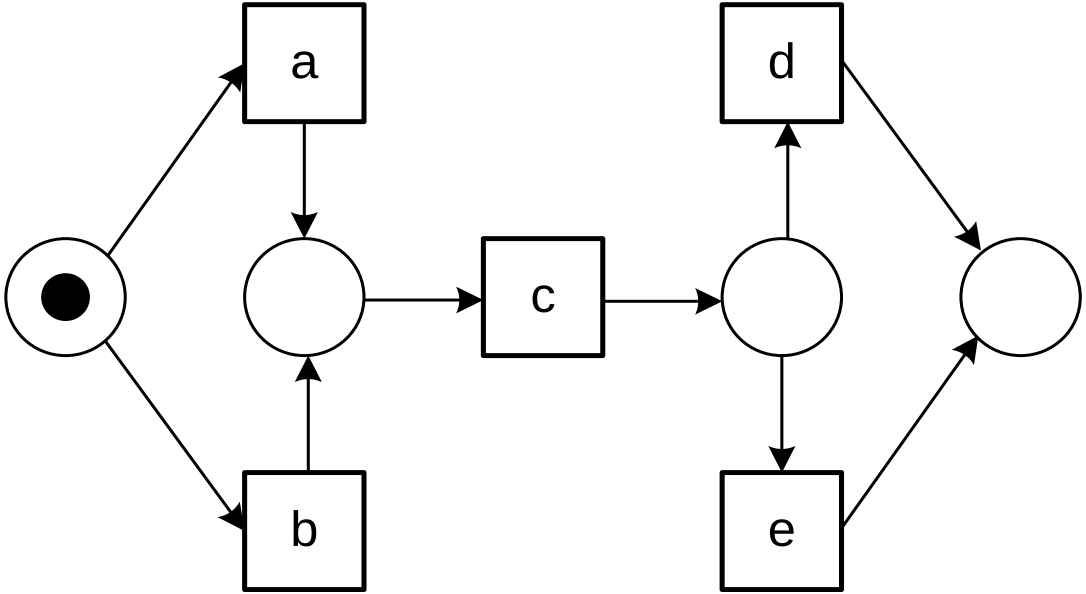
Generalization: good!

$\langle a,c,d \rangle^{99}$

$\langle a,c,e \rangle^{50}$

$\langle b,c,e \rangle^{85}$

$\langle b,c,d \rangle^{48}$



not overfitting…

Simplicity: good or bad?

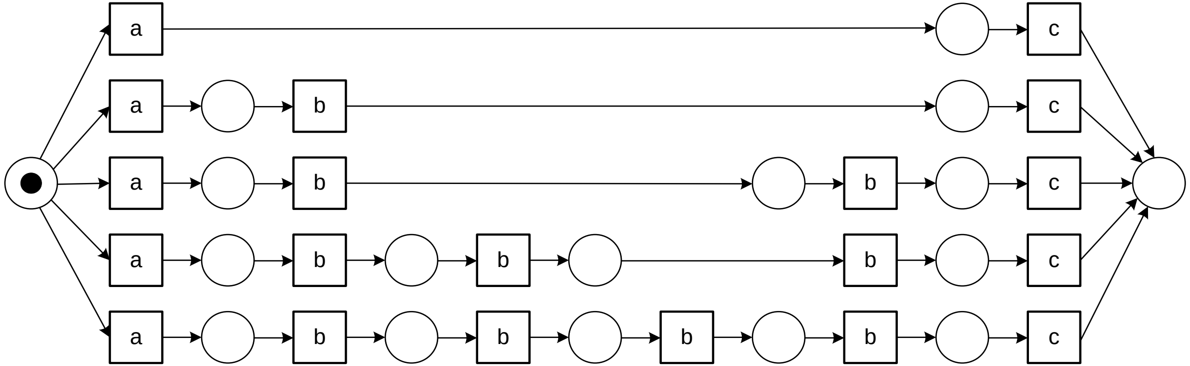
$\langle a,c\rangle^{16}$

$\langle a,b,c\rangle^{8}$

$\langle a,b,b,c\rangle^{4}$

$\langle a,b,b,b,c\rangle^{2}$

$\langle a,b,b,b,b,c\rangle^{1}$



Simplicity: bad!

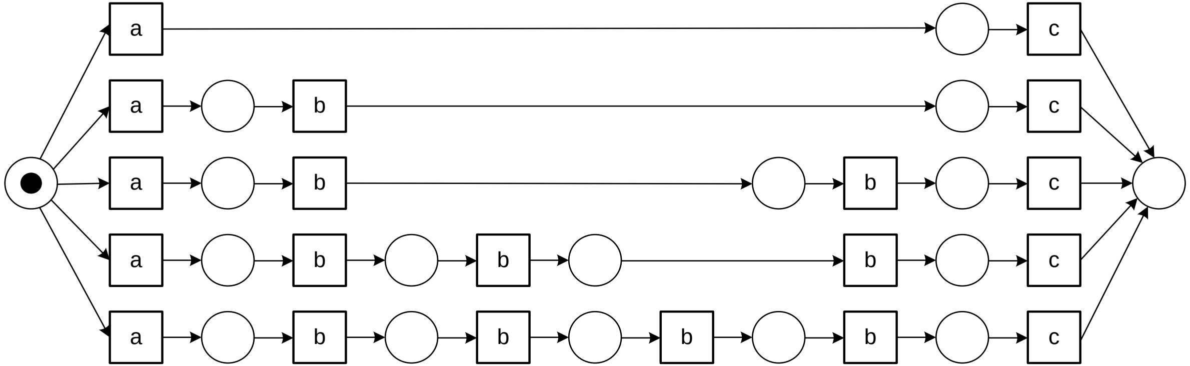
$\langle a,c\rangle^{16}$

$\langle a,b,c\rangle^{8}$

$\langle a,b,b,c\rangle^{4}$

$\langle a,b,b,b,c\rangle^{2}$

$\langle a,b,b,b,b,c\rangle^{1}$



too complex/specific…

Simplicity: good or bad?

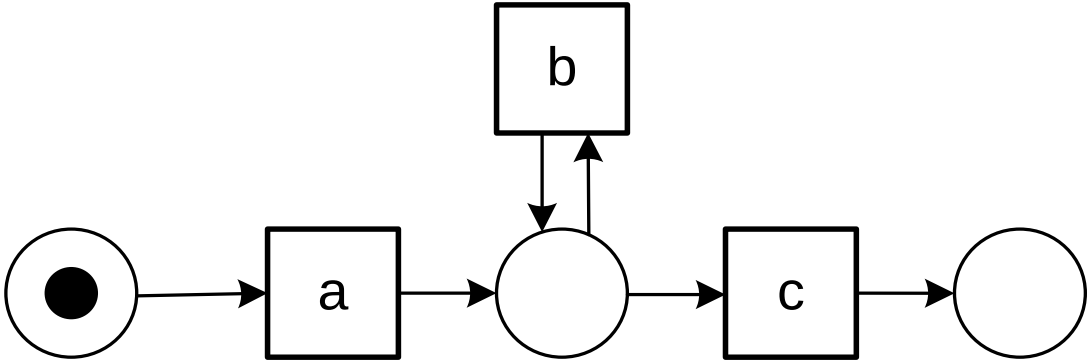
$\langle a,c\rangle^{16}$

$\langle a,b,c\rangle^{8}$

$\langle a,b,b,c\rangle^{4}$

$\langle a,b,b,b,c\rangle^{2}$

$\langle a,b,b,b,b,c\rangle^{1}$



Simplicity: good!

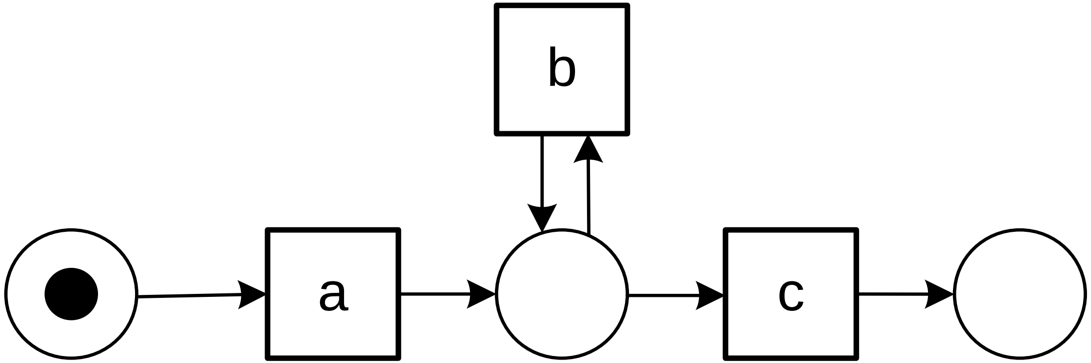
$\langle a,c\rangle^{16}$

$\langle a,b,c\rangle^{8}$

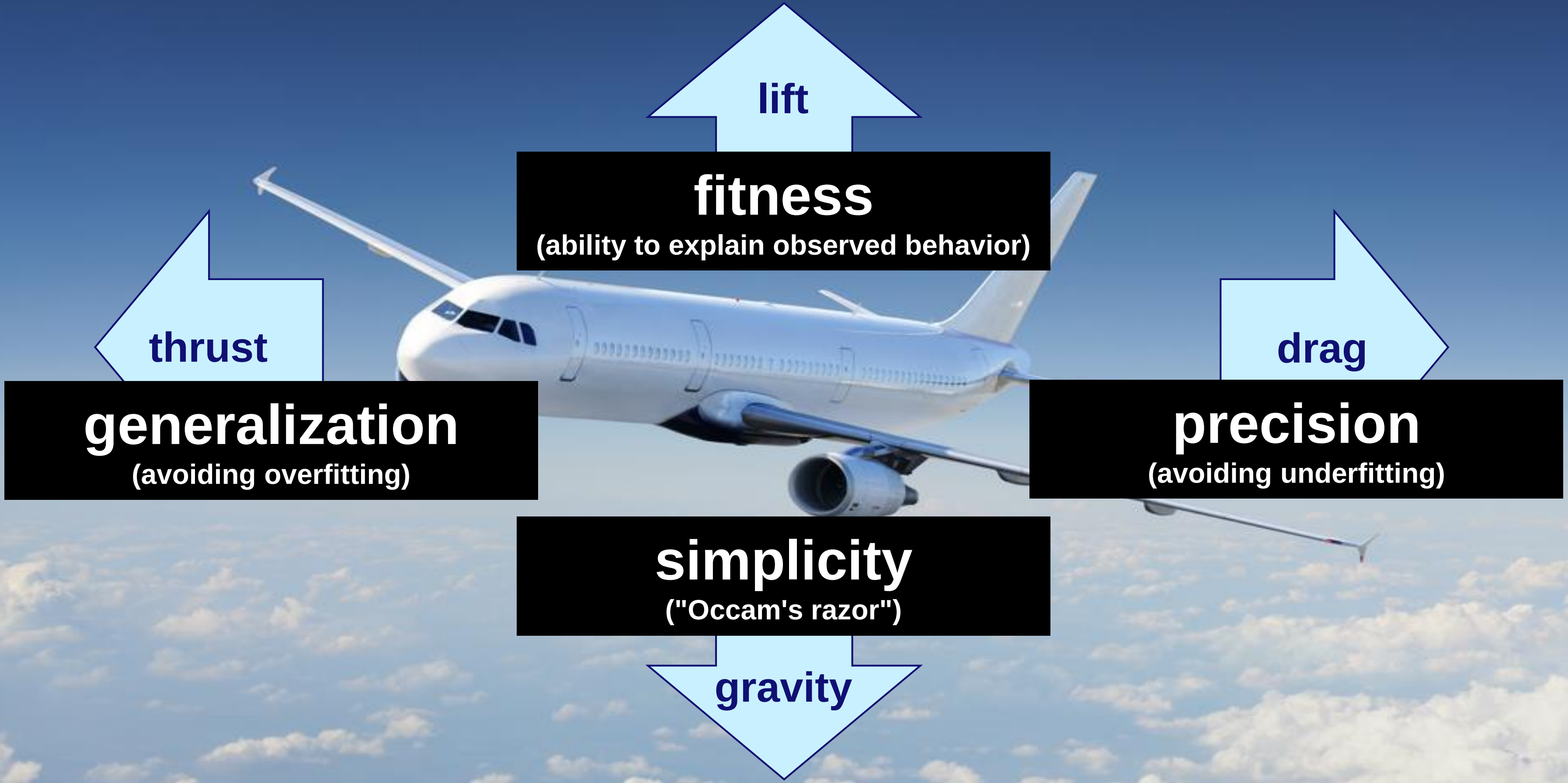
$\langle a,b,b,c\rangle^{4}$

$\langle a,b,b,b,c\rangle^{2}$

$\langle a,b,b,b,b,c\rangle^{1}$



Four Forces



lift

fitness

(ability to explain observed behavior)

thrust

generalization

(avoiding overfitting)

drag

precision

(avoiding underfitting)

simplicity

("Occam's razor")

gravity

THIS WAY
THAT WAY
ANOTHER WAY

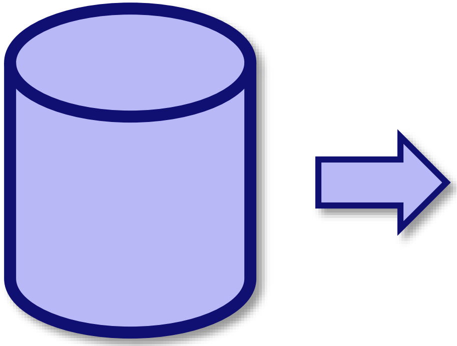
THIS WAY
THAT WAY
ANOTHER WAY

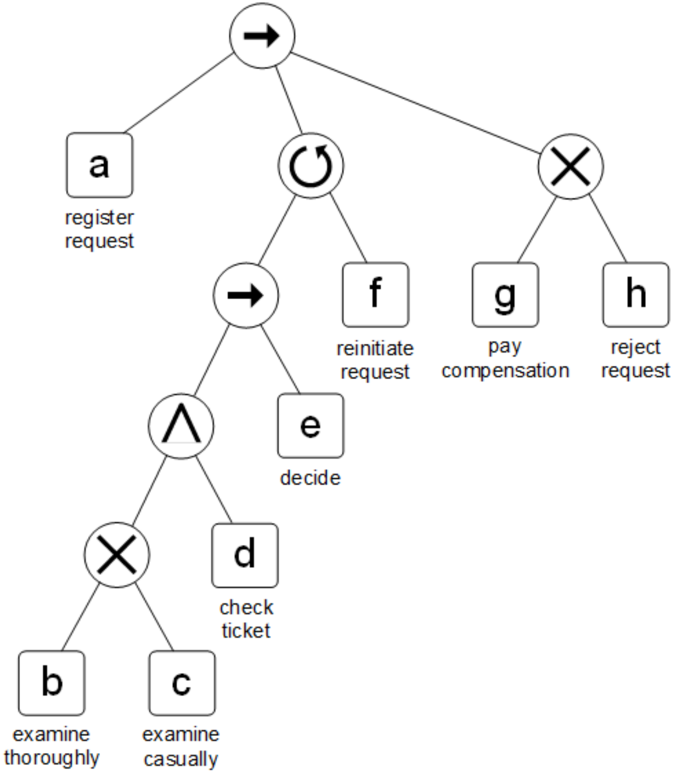
Characteristics

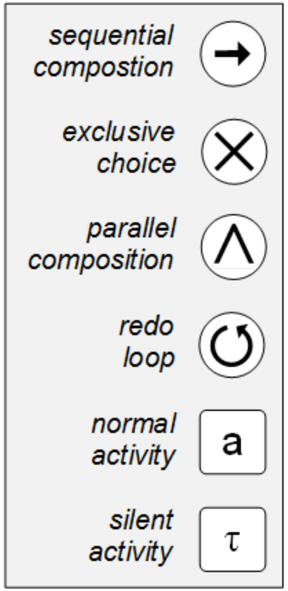
1. Representational bias (class of target models)
2. Ability to deal with noise/infrequent/incomplete behavior
3. Formal guarantees (in the limit, rediscoverability)
4. Scalability
5. Approach used:
   - Direct algorithmic (alpha-family, heuristic/fuzzy miner)
   - Region-based (language/state-based)
   - Generic/evolutionary
   - Inductive

Inductive mining

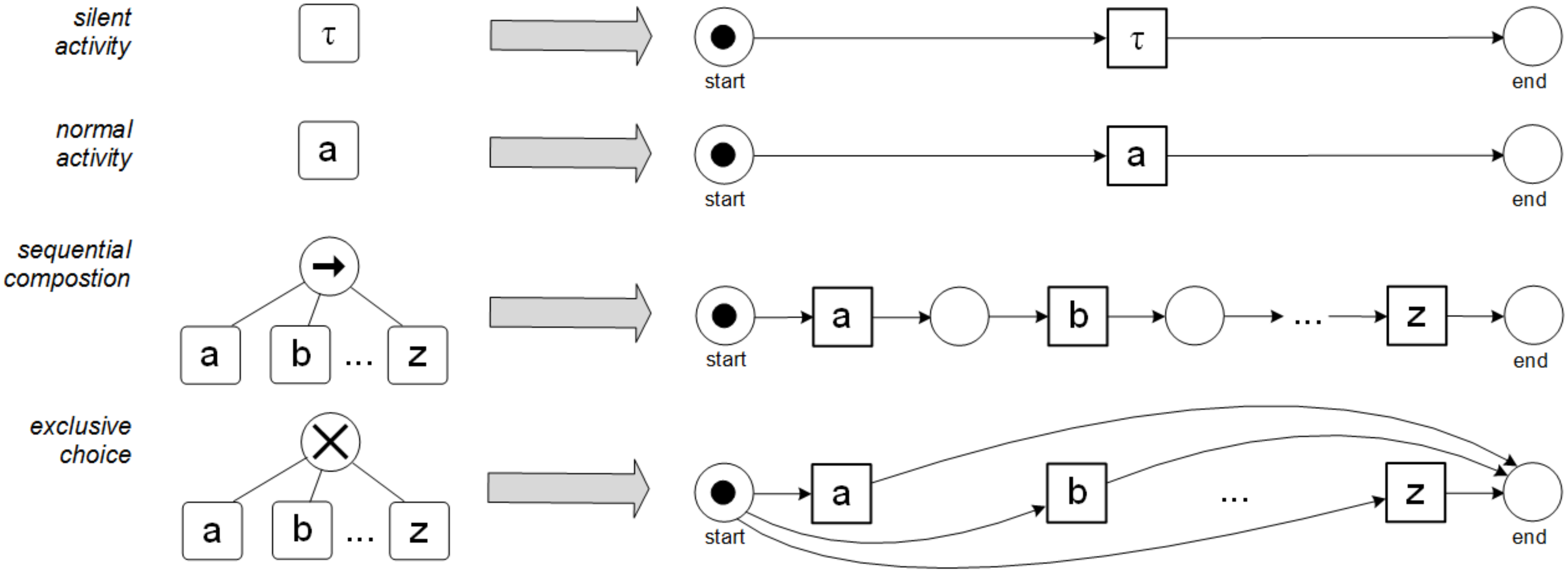
Process trees (to ensure soundness)





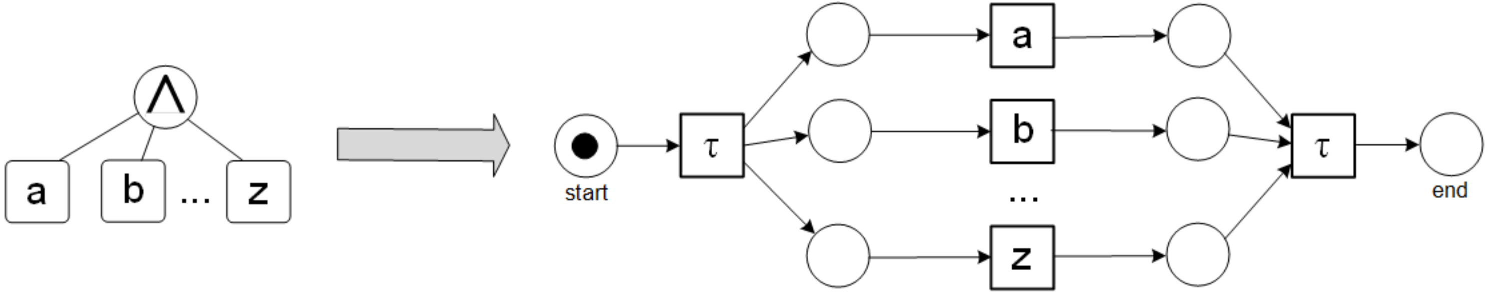


Process trees (semantics)

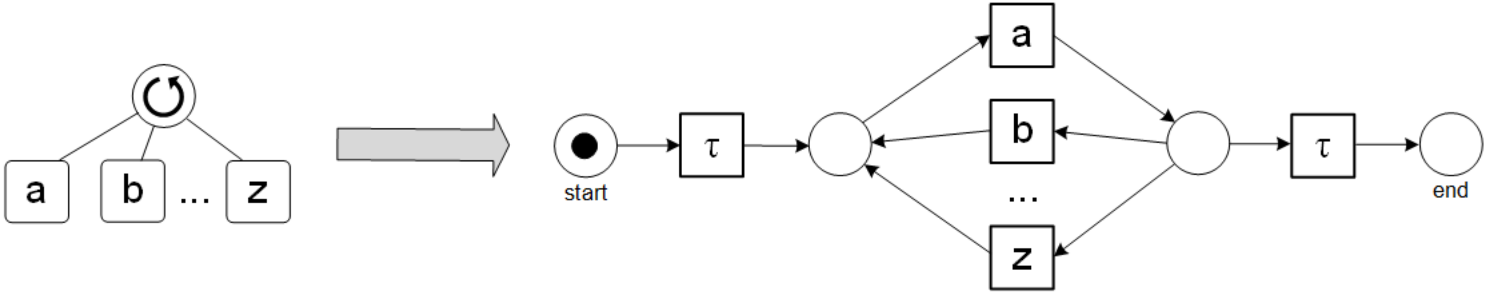


Process trees (semantics)

parallel composition



redo loop



Split event logs based on activity labels

abdef
acdef
adbef
adcef
abdeg
acdeg
adbeg
adceg

Split {a,b,c,d,e,f,g,h} into {a,b,c,d} and {e,f,g} using sequence decomposition

abdef

acdef

adbef

adcef

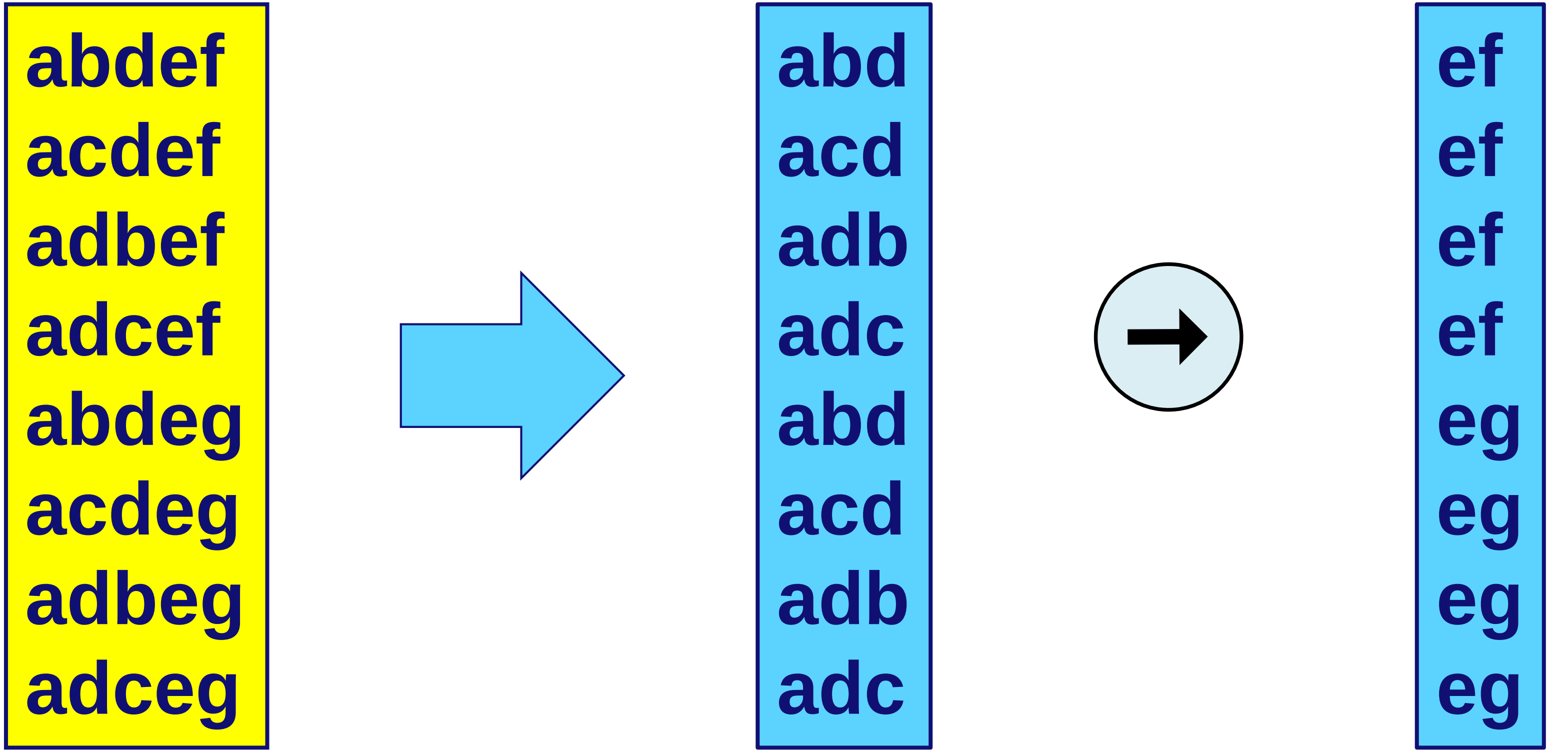
abdeg

acdeg

adbeg

adceg

Result



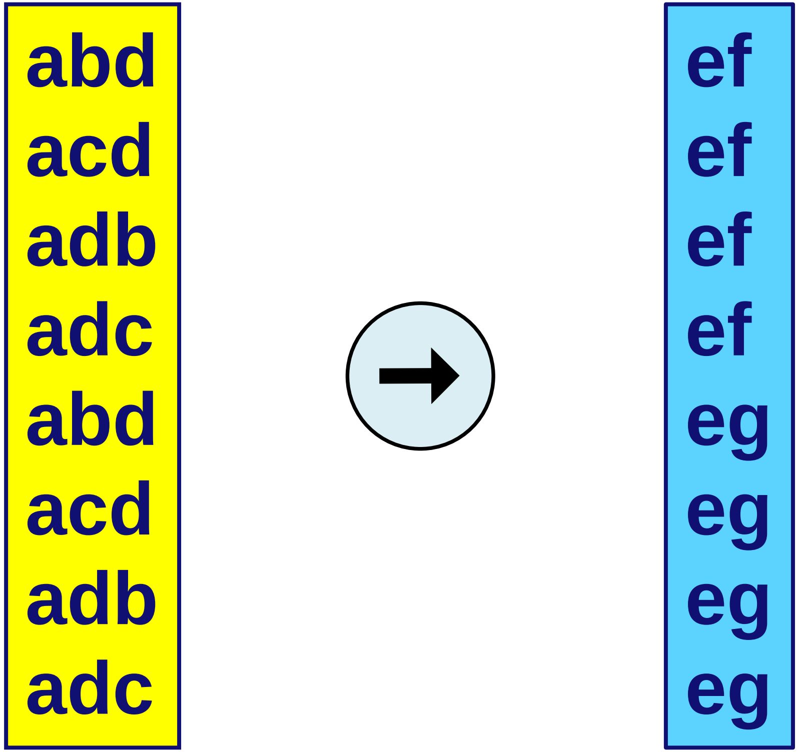
abdef
acdef
adbef
adcef
abdeg
acdeg
adbeg
adceg

abd
acd
adb
adc
abd
acd
adb
adc

→

ef
ef
ef
ef
eg
eg
eg
eg

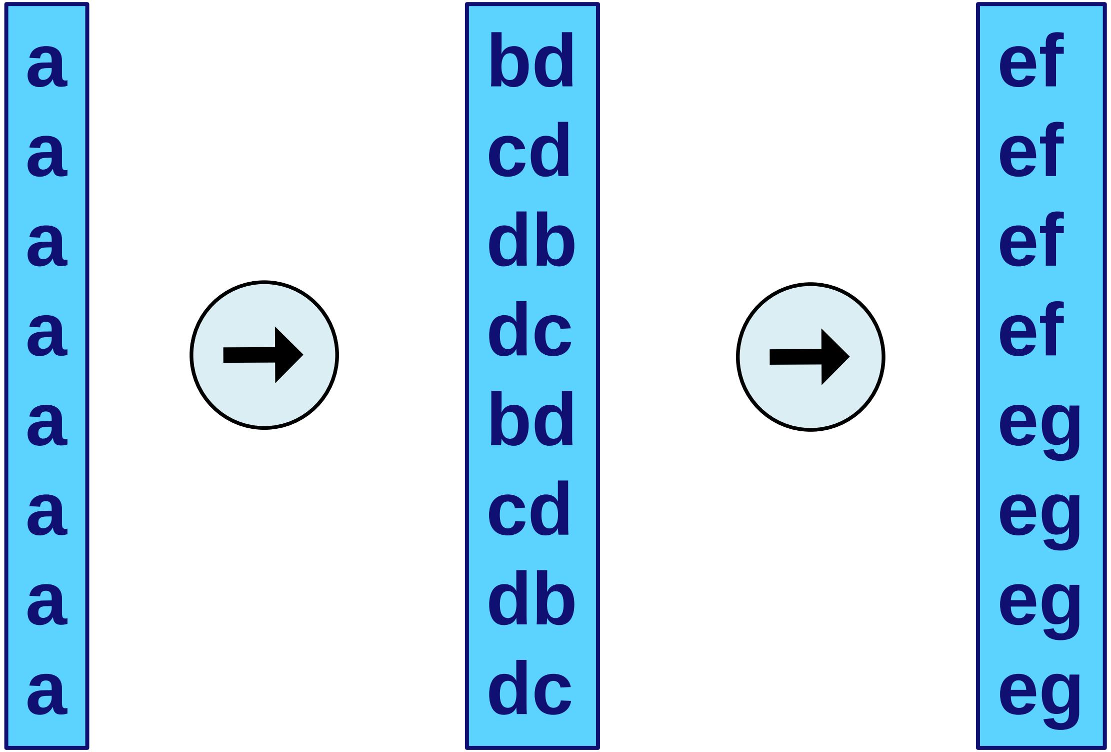
Split {a,b,c,d} into {a} and {b,c,d} using sequence decomposition



abd
acd
adb
adc
abd
acd
adb
adc

ef
ef
ef
ef
eg
eg
eg
eg

Result



a
a
a
a
a
a
a
a

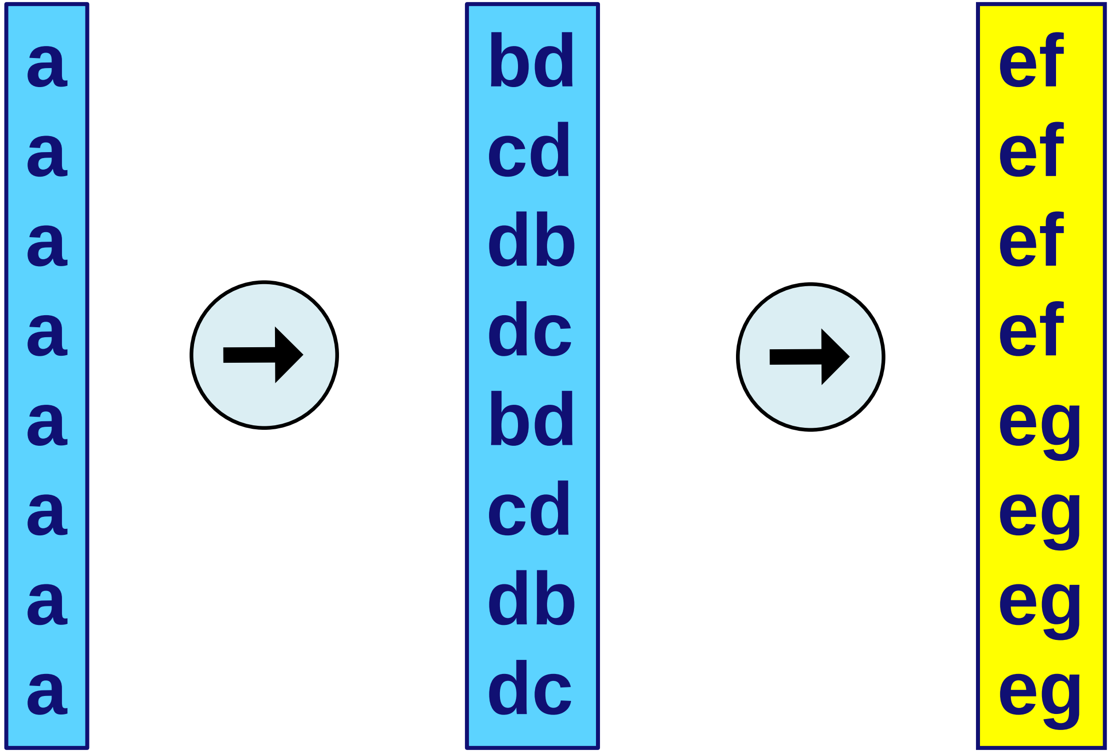
→

bd
cd
db
dc
bd
cd
db
dc

→

ef
ef
ef
ef
eg
eg
eg
eg

Split {e,f,g} into {e} and {f,g} using sequence decomposition

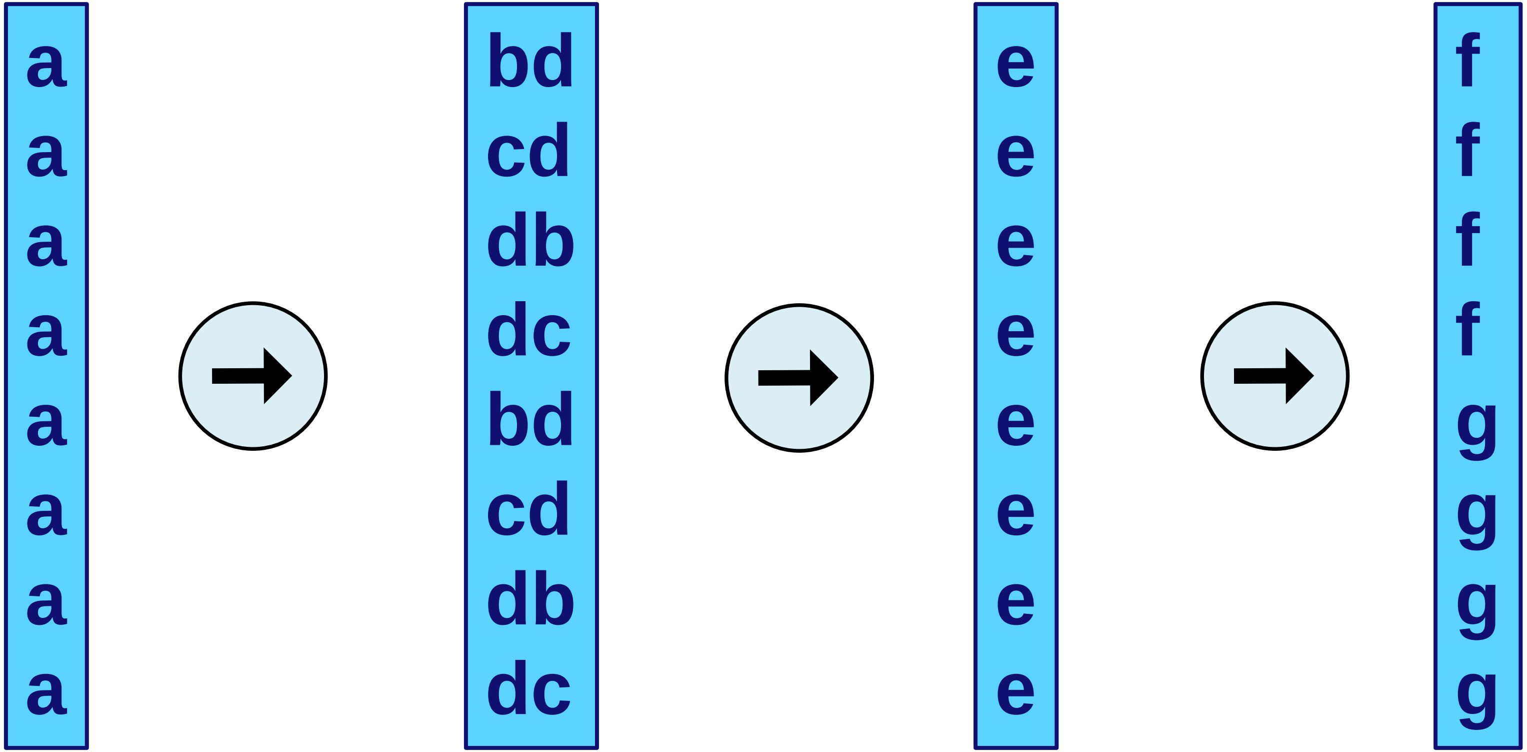


a
a
a
a
a
a
a
a

bd
cd
db
dc
bd
cd
db
dc

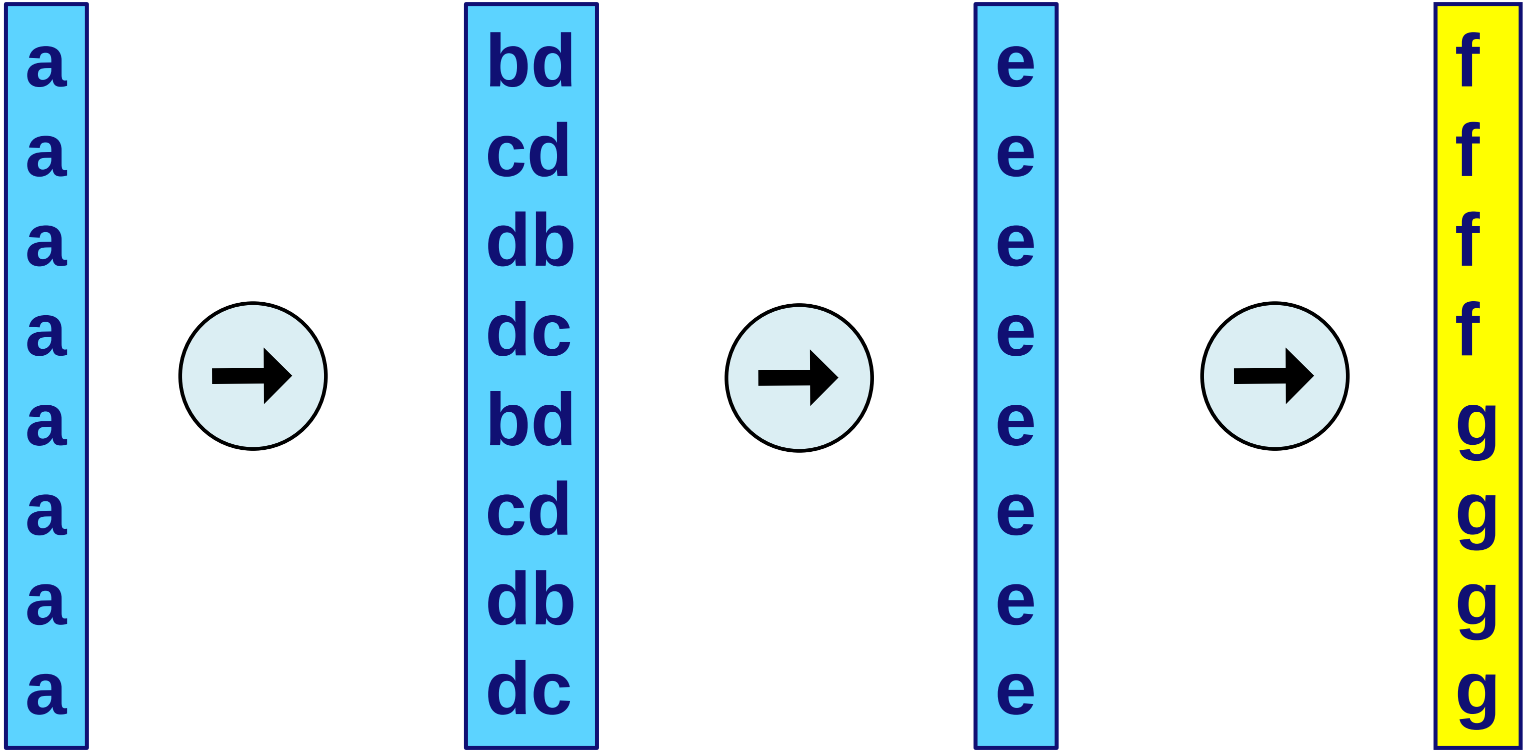
ef
ef
ef
ef
eg
eg
eg
eg

Result



a	bd	e	f
a	cd	e	f
a	db	e	f
a	dc	e	f
a	bd	e	g
a	cd	e	g
a	db	e	g
a	dc	e	g

Split {f,g} into {f} and {g} using XOR decomposition



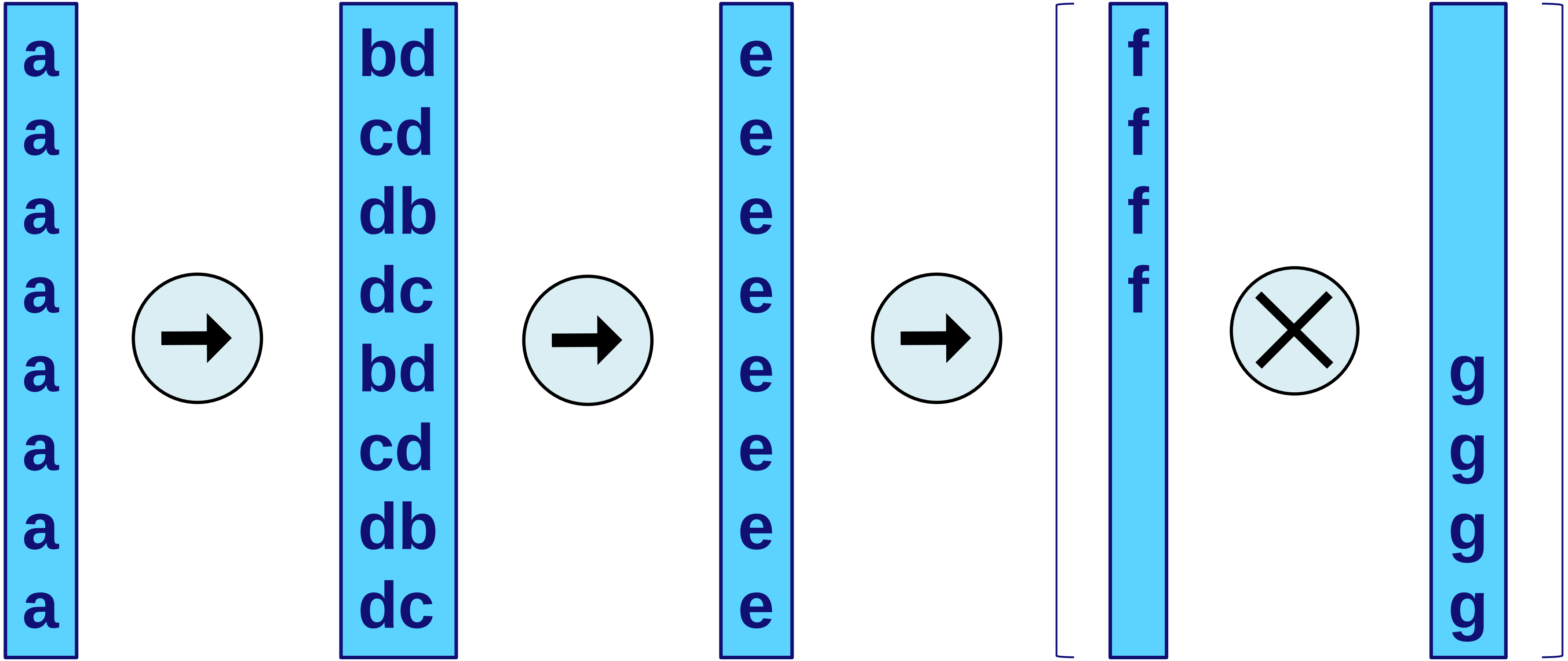
a
a
a
a
a
a
a
a

bd
cd
db
dc
bd
cd
db
dc

e
e
e
e
e
e
e
e

f
f
f
f
g
g
g
g

Result



a
a
a
a
a
a
a
a

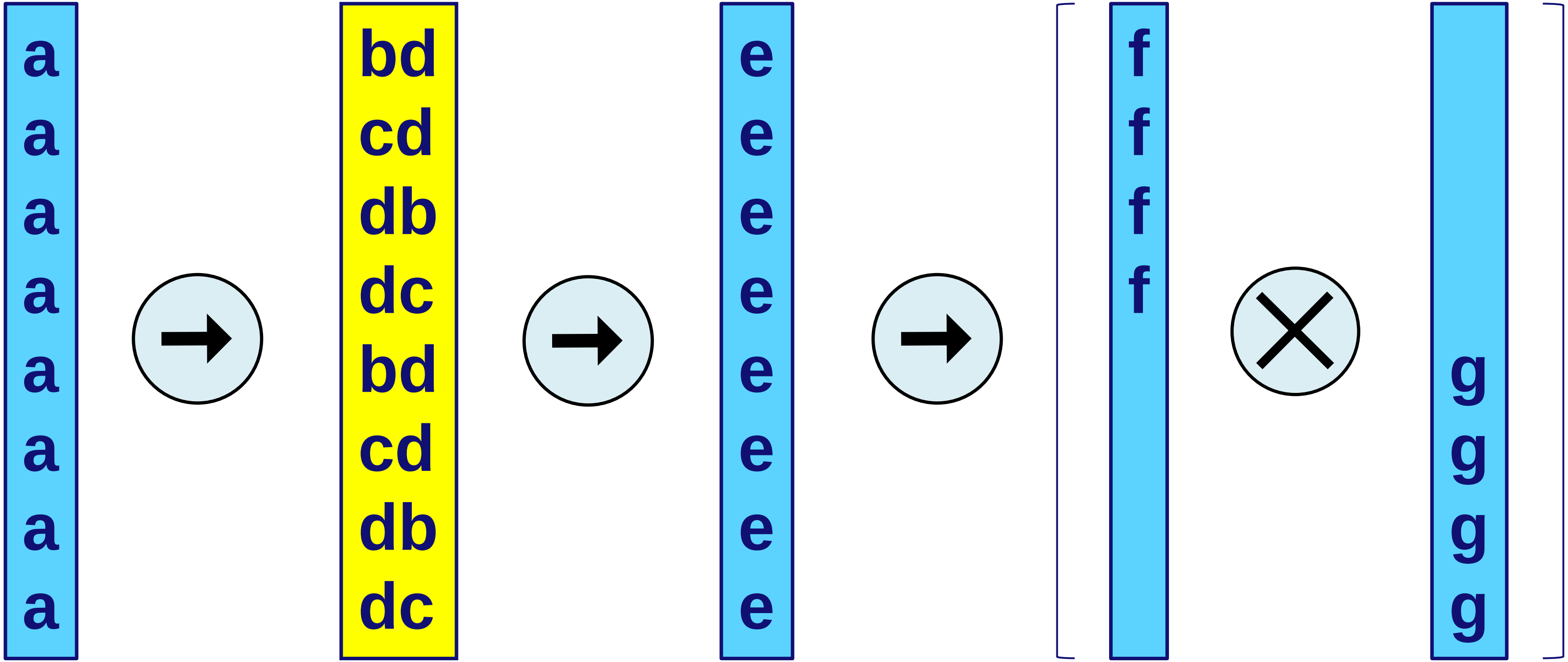
bd
cd
db
dc
bd
cd
db
dc

e
e
e
e
e
e
e
e

f
f
f
f

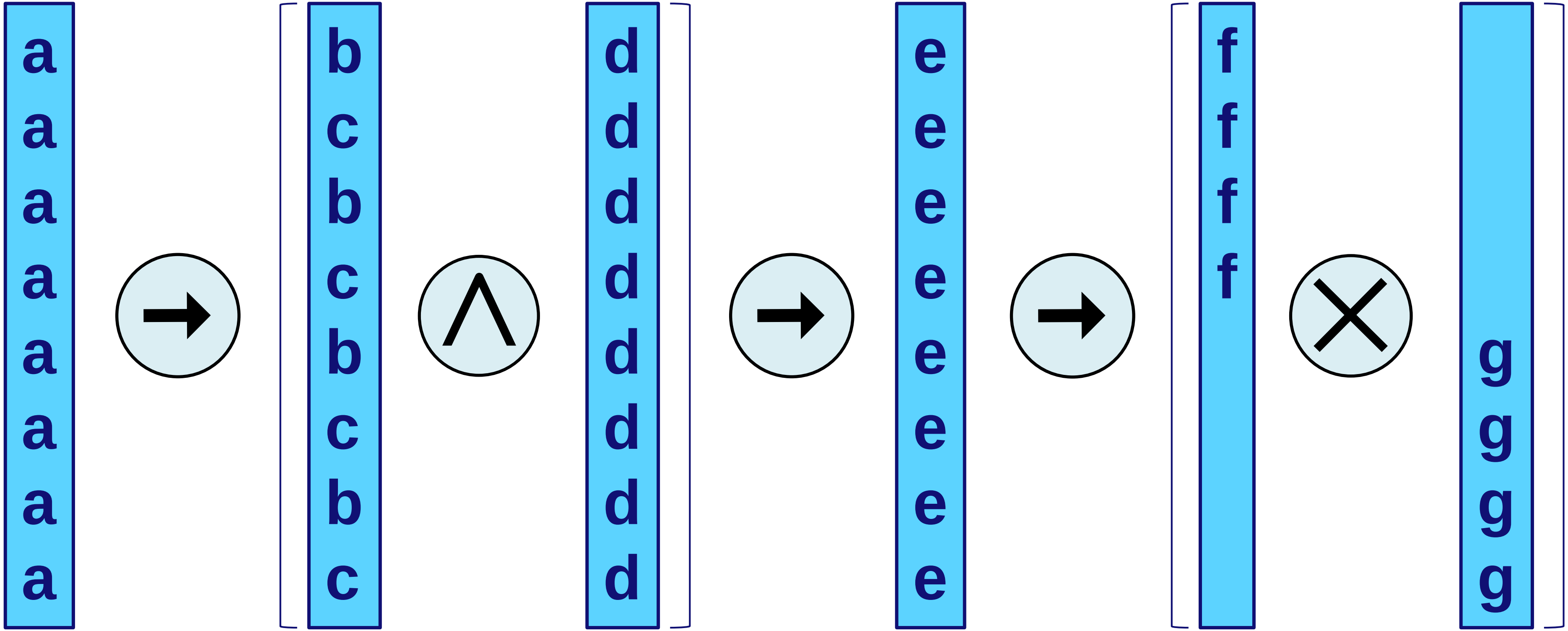
g
g
g
g

Split {b,c,d} into {b,c} and {d} using AND decomposition

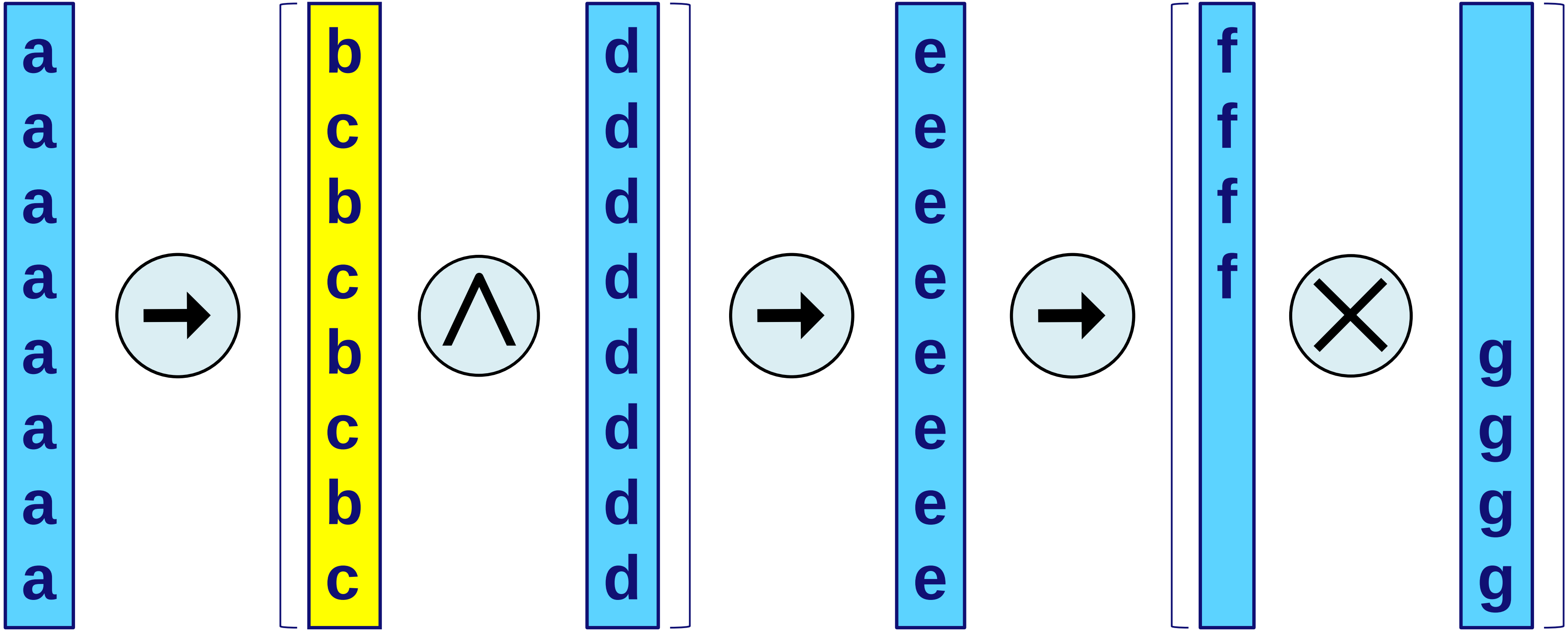


a	→	bd	→	e	→	f	×	
a		cd		e		f		
a		db		e		f		
a		dc		e		f		
a		bd		e				g
a		cd		e				g
a		db		e				g
a		dc		e				g

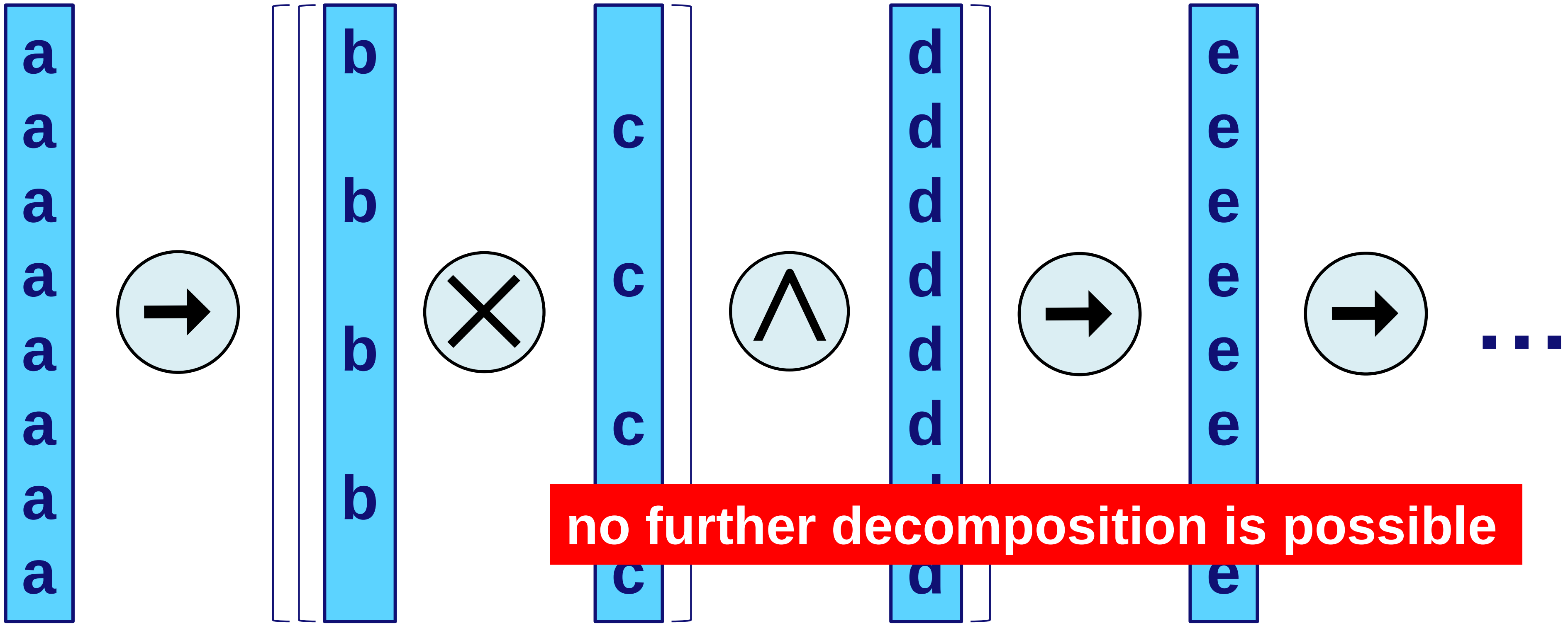
Result



Split {b,c} into {b} and {c} using XOR decomposition

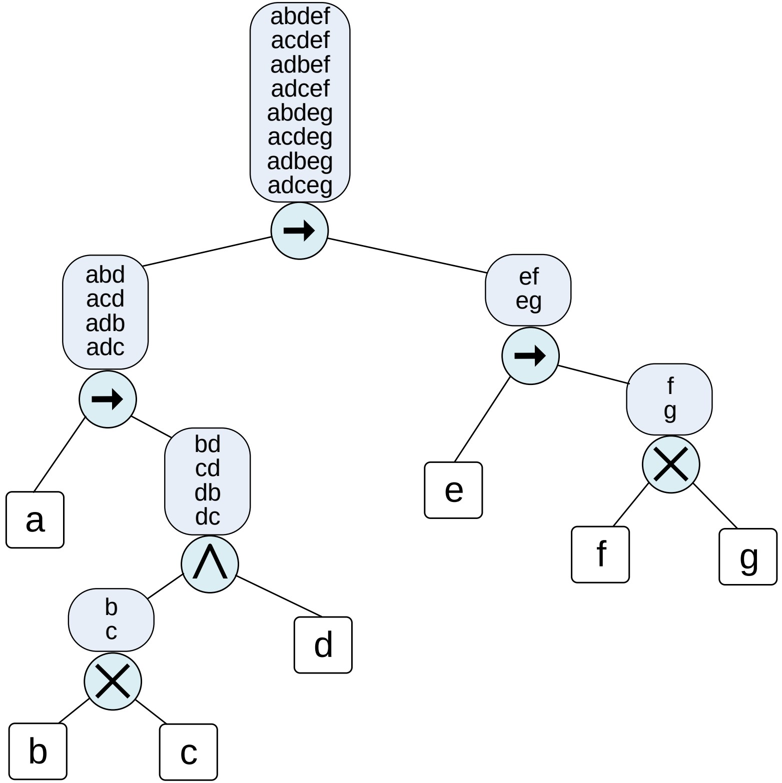


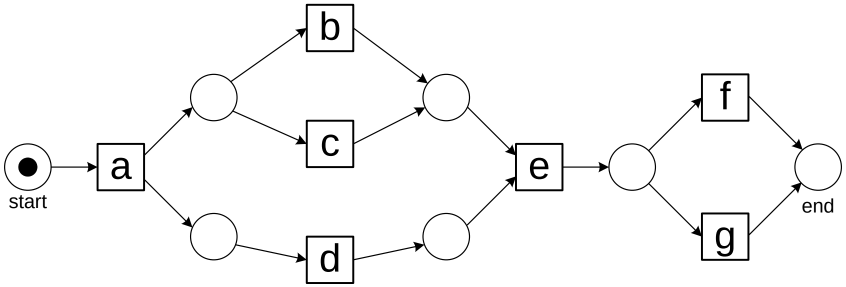
Result

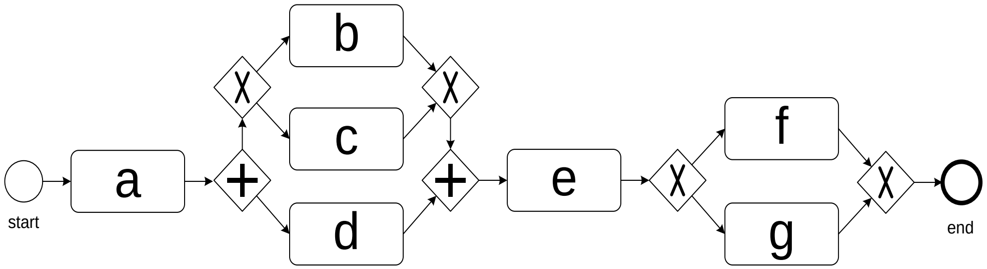


no further decomposition is possible

Process tree



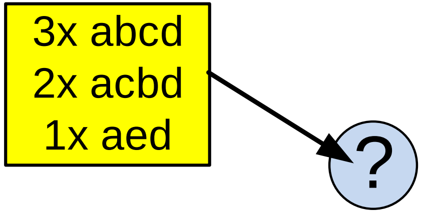




An example log (6 traces, 23 events)

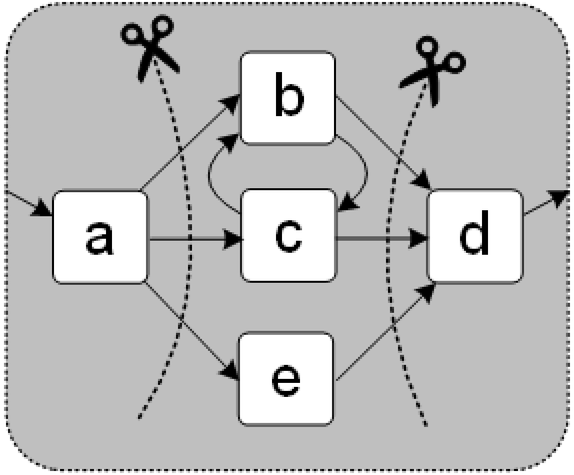
3x abcd
2x acbd
1x aed

How to split this event log?

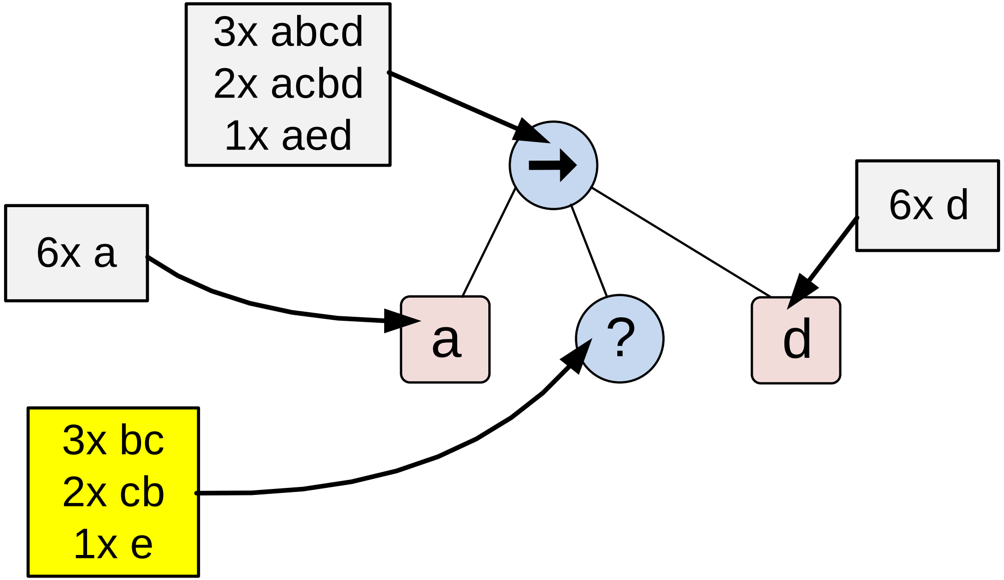


3x abcd
2x acbd
1x aed

?



How to split this event log?

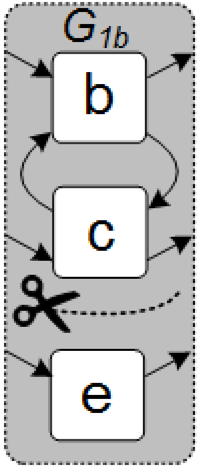


3x abcd
2x acbd
1x aed

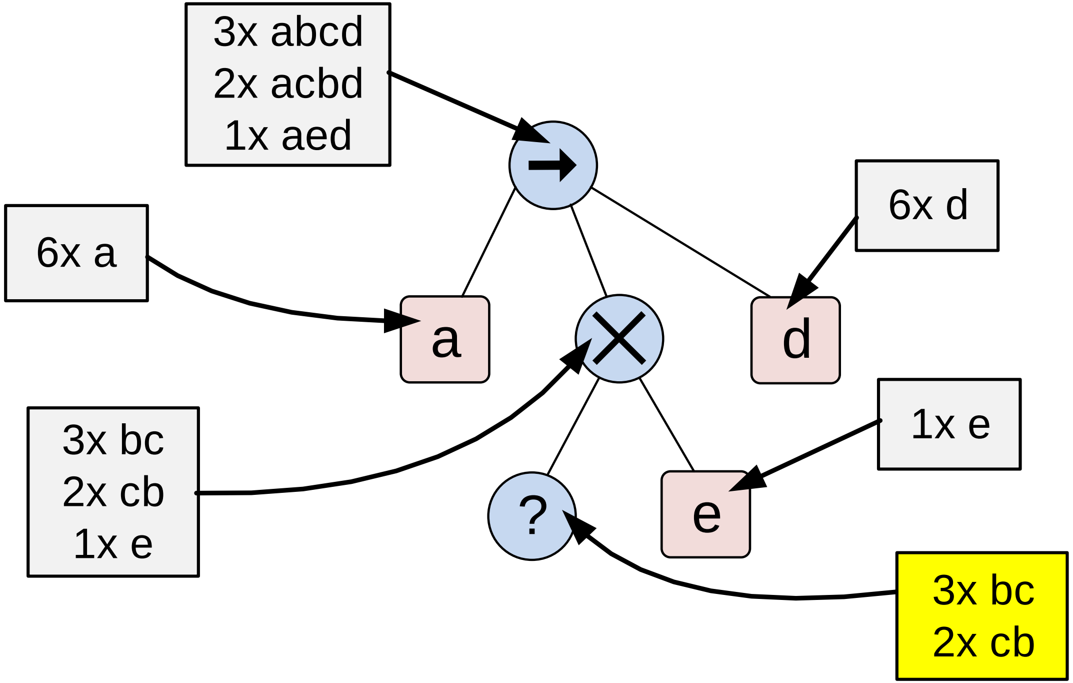
6x a

6x d

3x bc
2x cb
1x e



How to split this event log?



3x abcd
2x acbd
1x aed

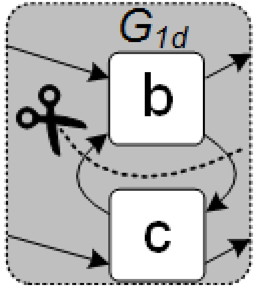
6x a

6x d

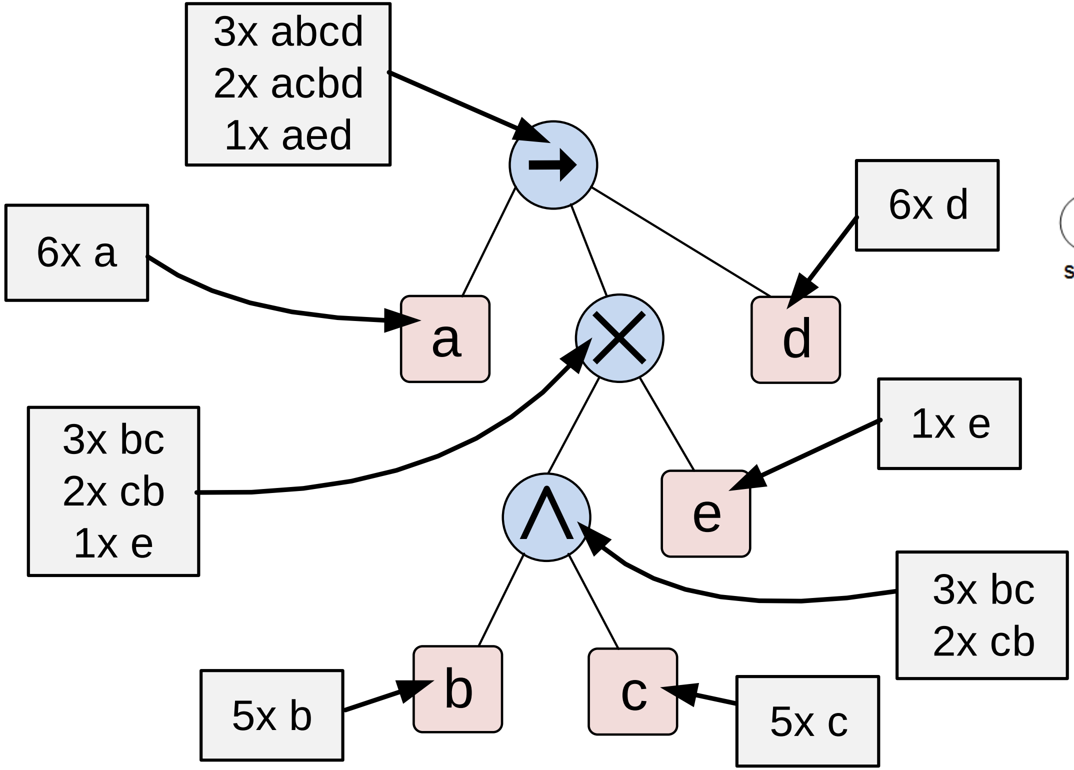
3x bc
2x cb
1x e

1x e

3x bc
2x cb



Final result



3x abcd
2x acbd
1x aed

6x a

6x d

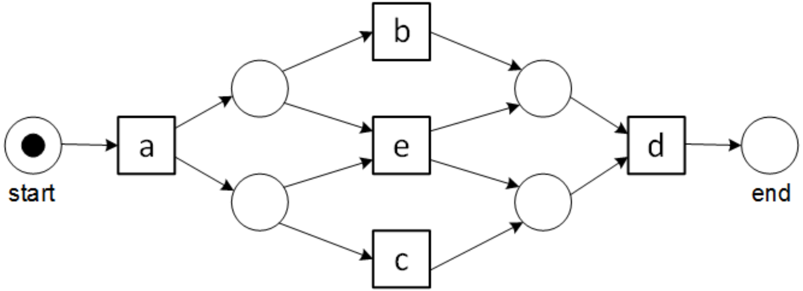
3x bc
2x cb
1x e

1x e

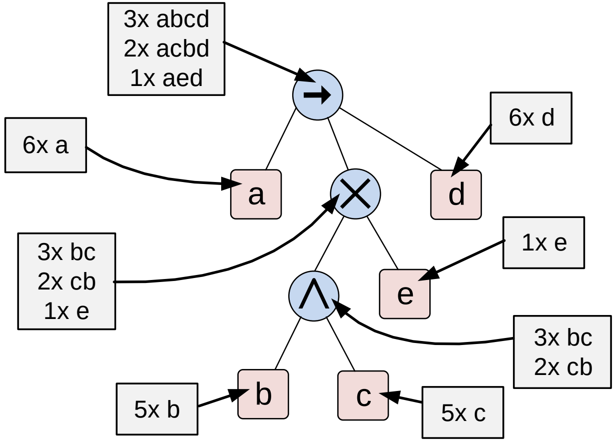
3x bc
2x cb

5x b

5x c



In ProM



3x abcd
2x acbd
1x aed

6x a

6x d

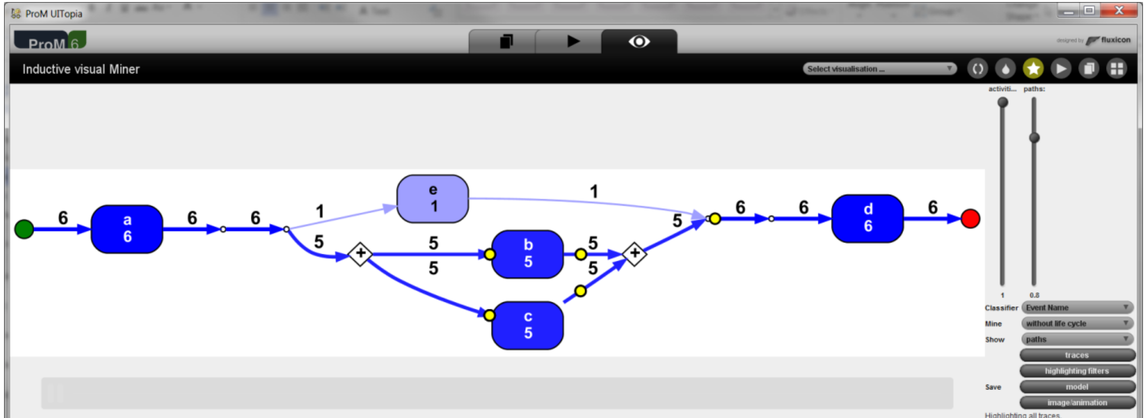
3x bc
2x cb
1x e

1x e

3x bc
2x cb

5x b

5x c



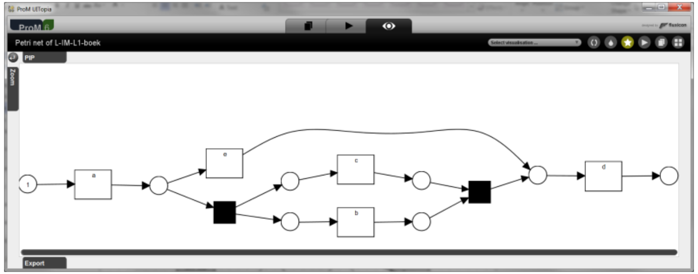
ProM UITopia

Inductive visual Miner

Classifier: Event Name

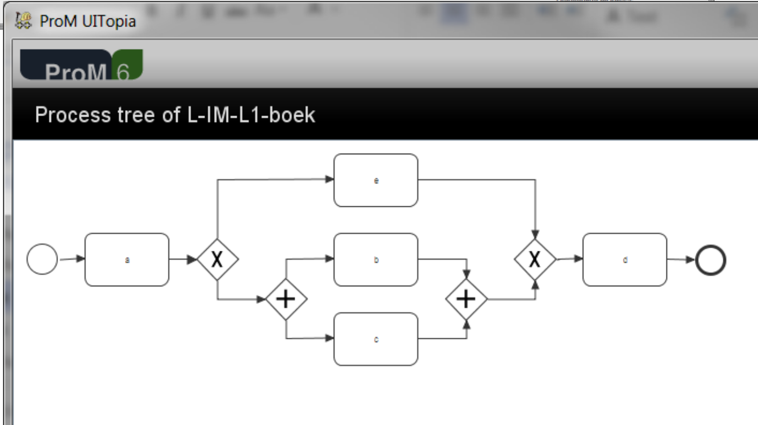
Mine: without life cycle

Show: paths



ProM UITopia

Petri net of L-IM-L1-boek



ProM UITopia

Process tree of L-IM-L1-boek

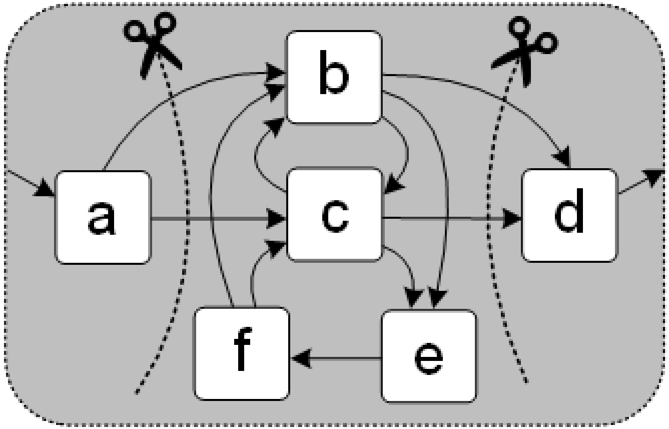
Example log with loops (13 traces, 80 events)

3x abcd
4x acbd
2x abcefbcd
2x acbefbcd
1x abcefcbd
1x acbefbcefcbd

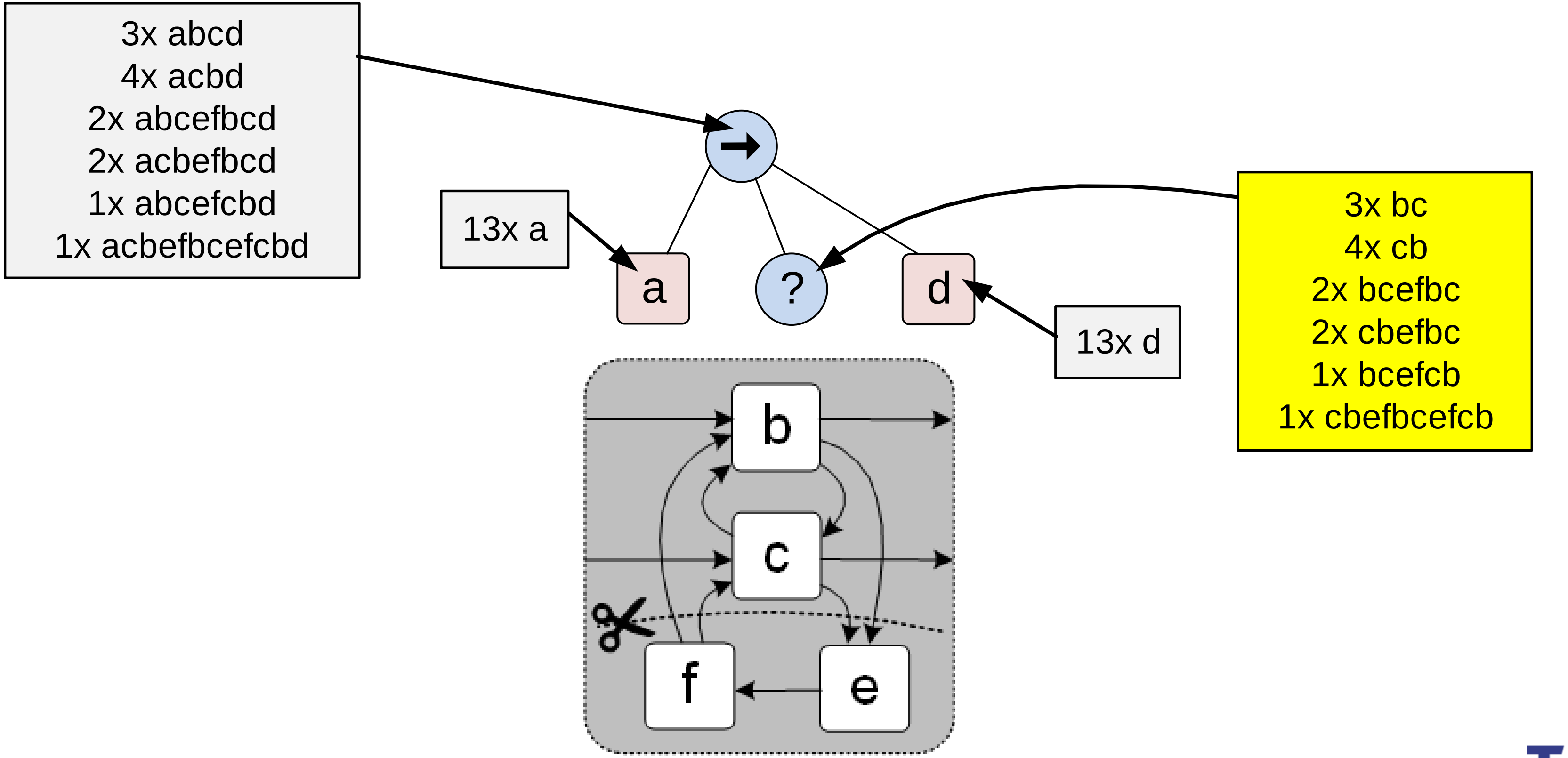
How to split this event log?

3x abcd
4x acbd
2x abcefbcd
2x acbefbcd
1x abcefcbd
1x acbefbcefcbd

?



How to split this event log?



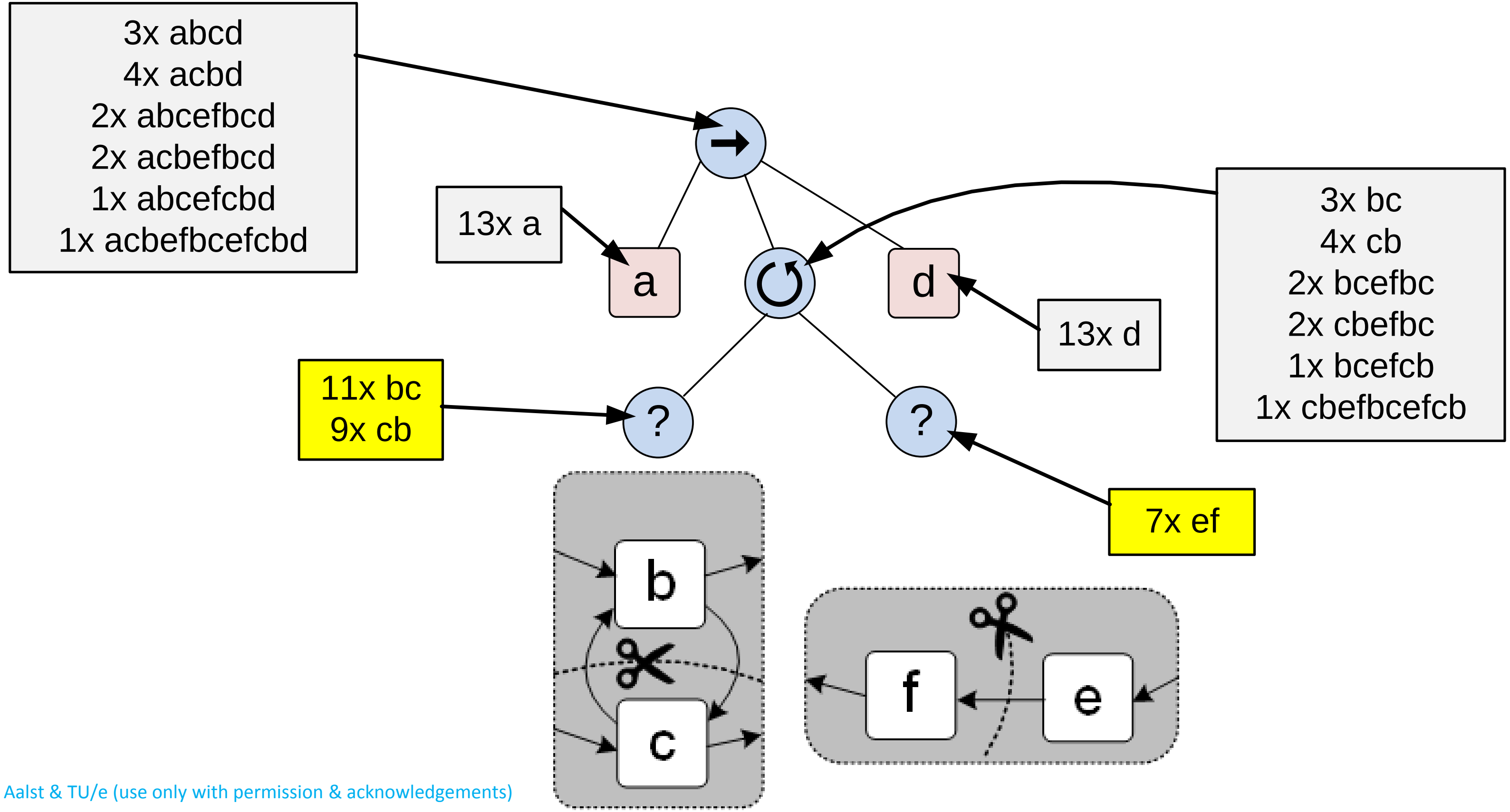
3x abcd
4x acbd
2x abcefbcd
2x acbefbcd
1x abcefcbd
1x acbefbcefcbd

13x a

13x d

3x bc
4x cb
2x bcefbc
2x cbefbc
1x bcefcb
1x cbefbcefcb

How to split this event log?



3x abcd
4x acbd
2x abcefbcd
2x acbefbcd
1x abcefcbd
1x acbefbcefcbd

13x a

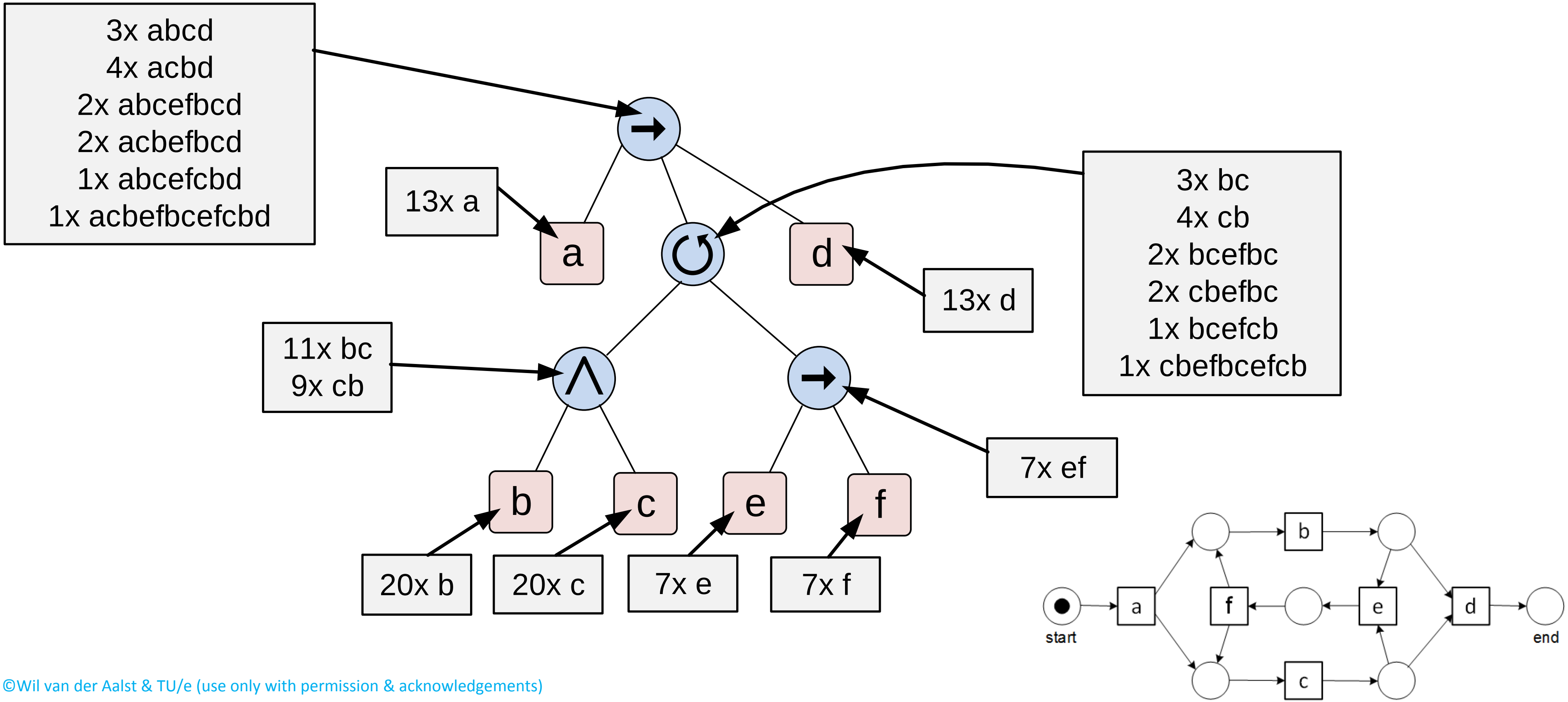
13x d

3x bc
4x cb
2x bcefbc
2x cbefbc
1x bcefcb
1x cbefbcefcb

11x bc
9x cb

7x ef

Result



3x abcd
4x acbd
2x abcefbcd
2x acbefbcd
1x abcefcbd
1x acbefbcefcbd

13x a

3x bc
4x cb
2x bcefbc
2x cbefbc
1x bcefcb
1x cbefbcefcb

13x d

11x bc
9x cb

7x ef

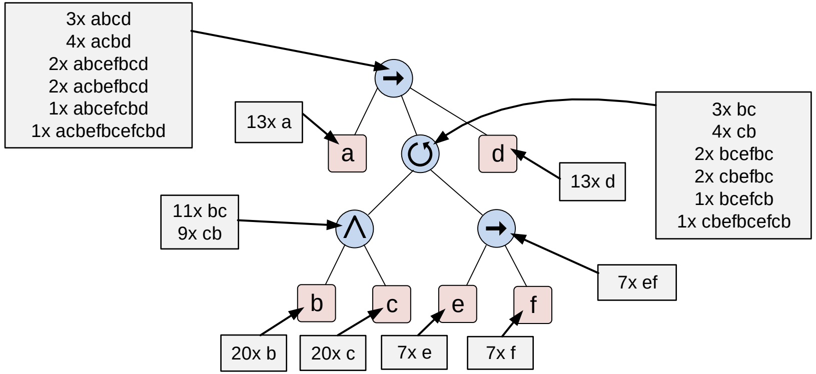
20x b

20x c

7x e

7x f

In ProM



3x abcd
4x acbd
2x abcefbcd
2x acbefbcd
1x abcefcbd
1x acbefbcefcbd

13x a

13x d

3x bc
4x cb
2x bcefbc
2x cbefbc
1x bcefcb
1x cbefbcefcb

11x bc
9x cb

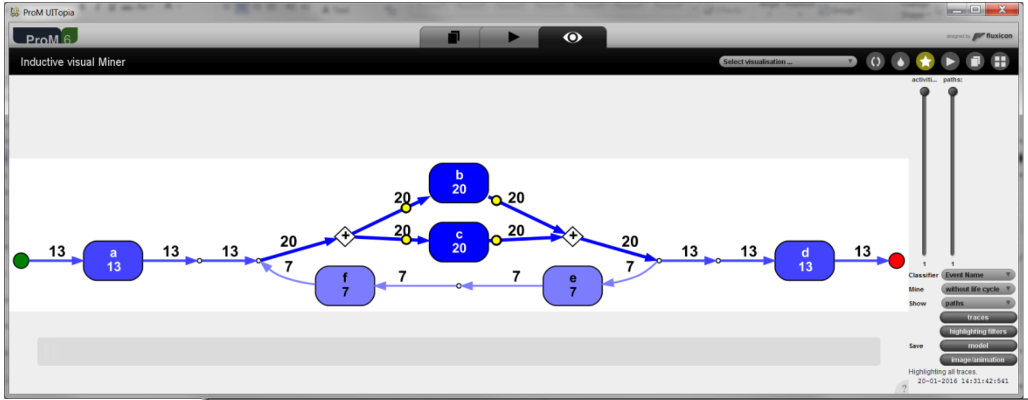
7x ef

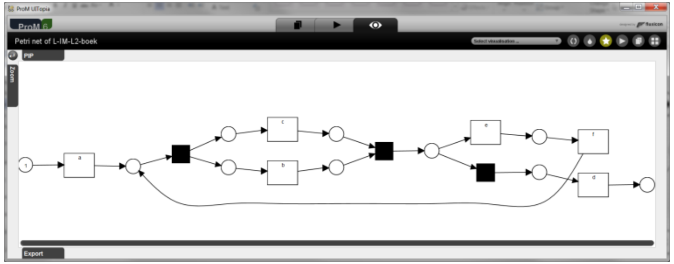
20x b

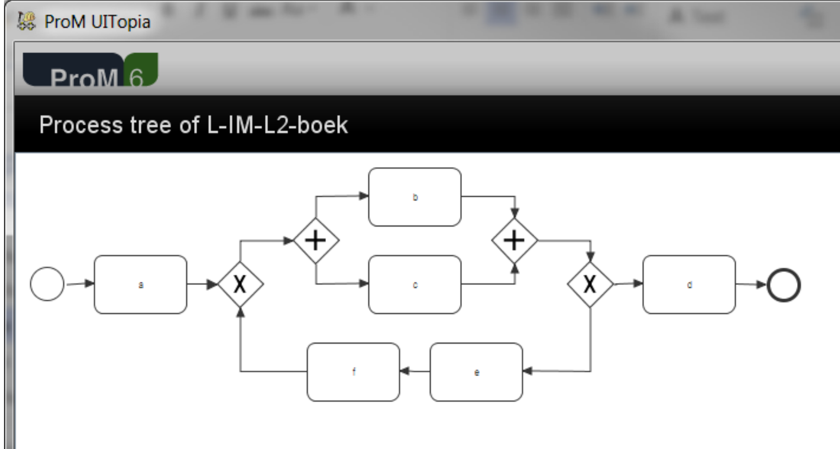
20x c

7x e

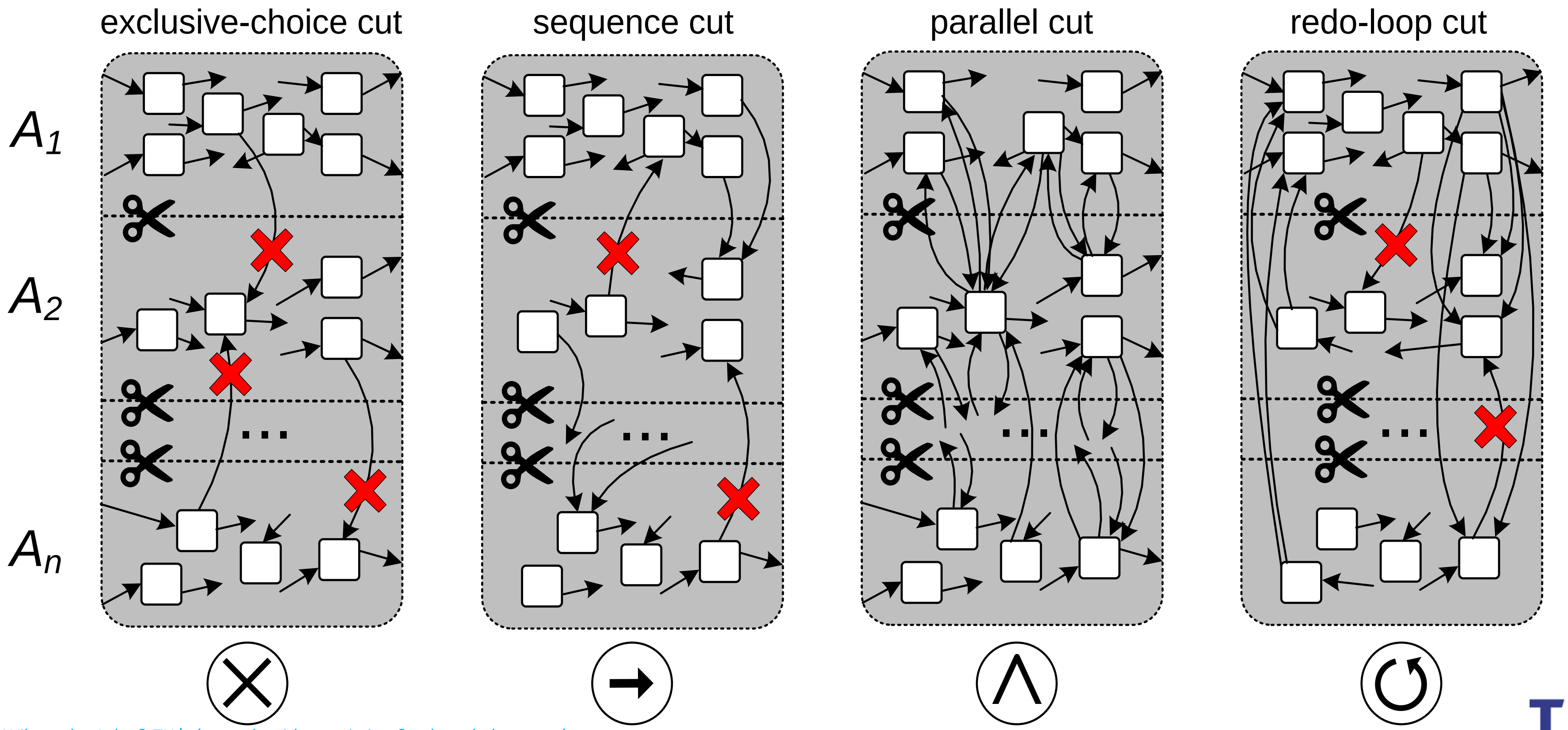
7x f





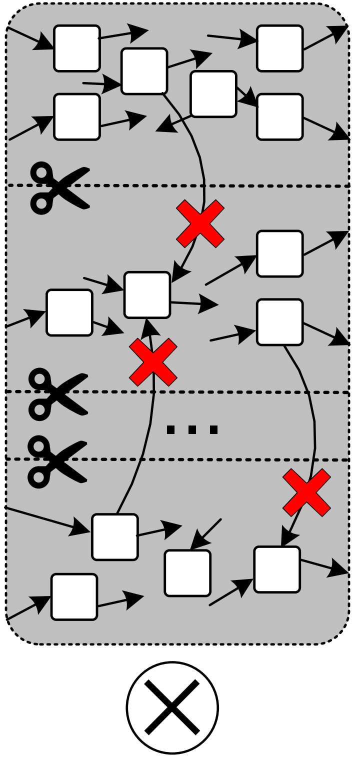


How to cut the event log?



How to cut the event log?

exclusive-choice cut

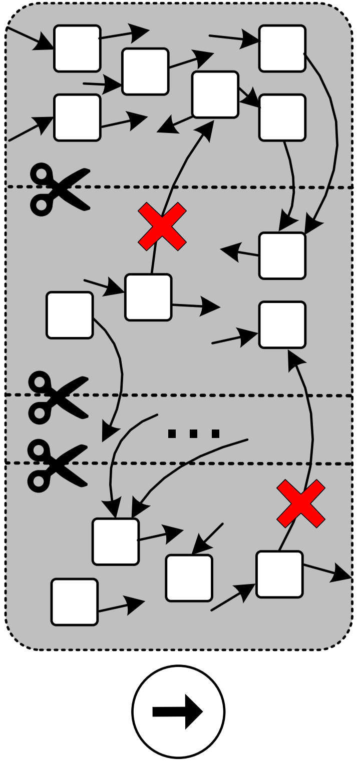


An *exclusive-choice cut* of $G(L)$ is a cut $(\times, A_1, A_2, \ldots, A_n)$ such that

- $\forall_{i,j \in \{1,\ldots,n\}} \forall_{a \in A_i} \forall_{b \in A_j}\ i \neq j \ \Rightarrow\ a \not\mapsto_L b.$

How to cut the event log?

sequence cut



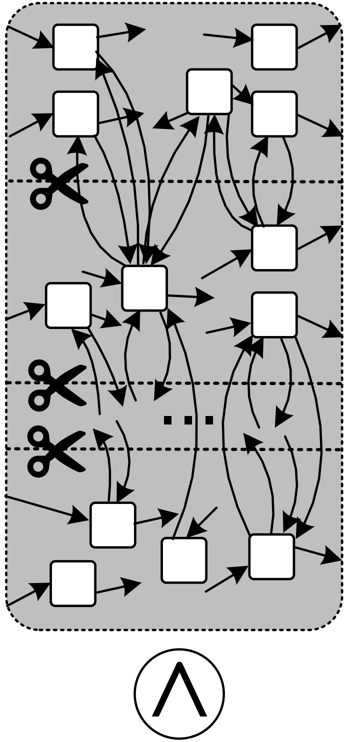
A *sequence cut* of $G(L)$ is a cut $(\rightarrow, A_1, A_2, \ldots, A_n)$ such that

– $\forall_{i,j \in \{1,\ldots,n\}} \forall_{a \in A_i} \forall_{b \in A_j}\ i < j \ \Rightarrow\ (a \mapsto^{+}_{L} b \ \wedge\ b \not\mapsto^{+}_{L} a)$.

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

How to cut the event log?

parallel cut

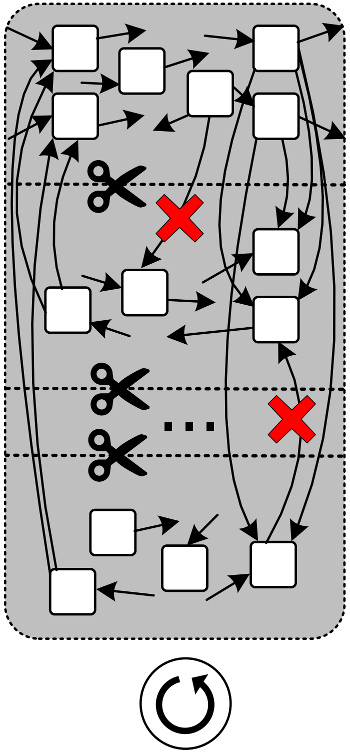


A *parallel cut* of $G(L)$ is a cut $(\wedge, A_1, A_2, \ldots, A_n)$ such that

- $\forall_{i \in \{1,\ldots,n\}}\ A_i \cap A_L^{start} \neq \emptyset\ \wedge\ A_i \cap A_L^{end} \neq \emptyset$ and
- $\forall_{i,j \in \{1,\ldots,n\}} \forall_{a \in A_i} \forall_{b \in A_j}\ i \neq j\ \Rightarrow\ a \mapsto_L b$.

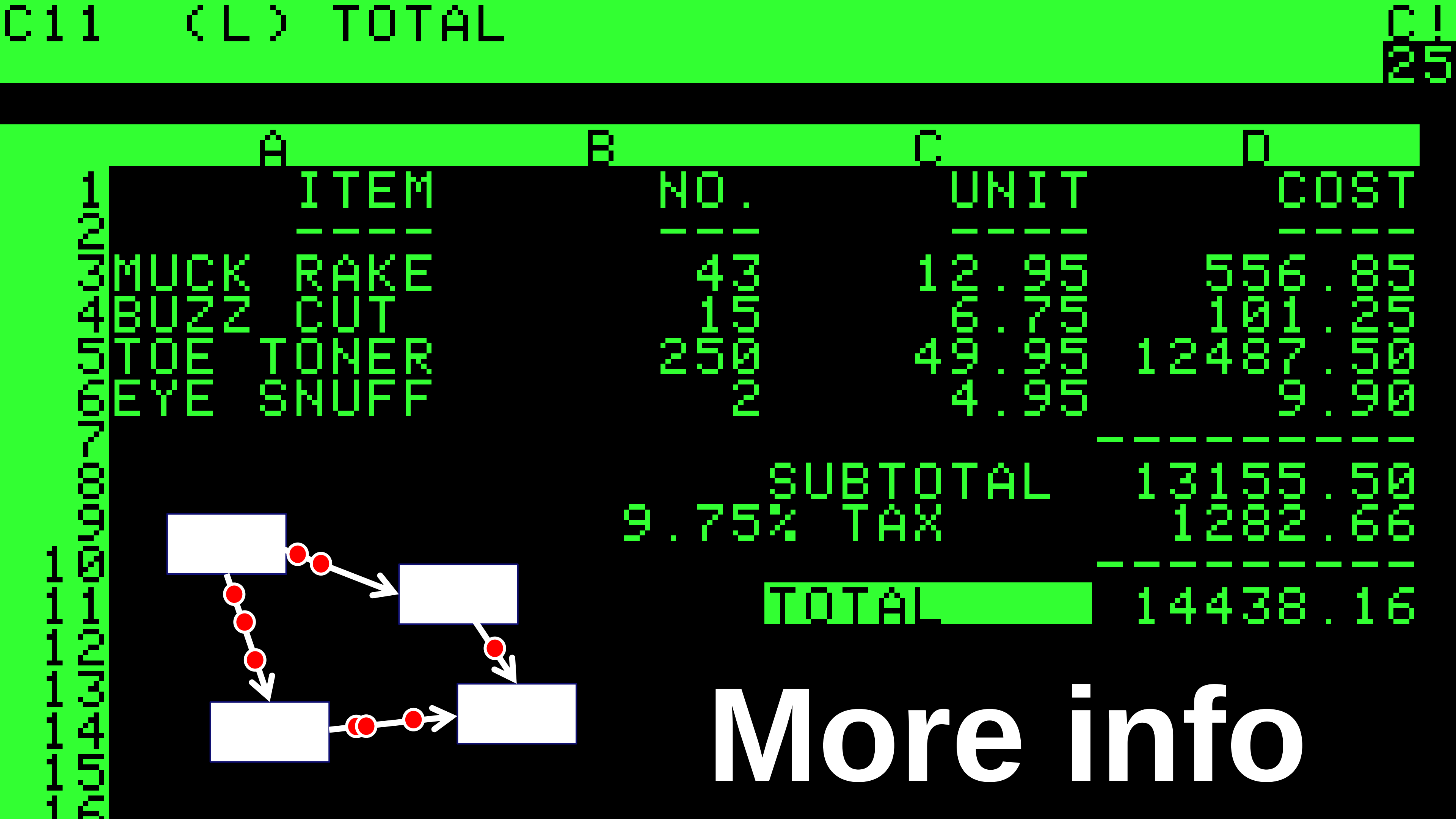
How to cut the event log?

redo-loop cut



A *redo-loop cut* of $G(L)$ is a cut $(\circlearrowleft, A_1, A_2, \ldots, A_n)$ such that

- $n \geq 2$,
- $A_L^{start} \cup A_L^{end} \subseteq A_1$,
- $\{a \in A_1 \mid \exists_{i \in \{2,\ldots,n\}} \exists_{b \in A_i}\ a \mapsto_L b\} \subseteq A_L^{end}$,
- $\{a \in A_1 \mid \exists_{i \in \{2,\ldots,n\}} \exists_{b \in A_i}\ b \mapsto_L a\} \subseteq A_L^{start}$,
- $\forall_{i,j \in \{2,\ldots,n\}} \forall_{a \in A_i} \forall_{b \in A_j}\ i \neq j \Rightarrow a \not\mapsto_L b$,
- $\forall_{i \in \{2,\ldots,n\}} \forall_{b \in A_i} \exists_{a \in A_L^{end}}\ a \mapsto_L b \Rightarrow \forall_{a' \in A_L^{end}}\ a' \mapsto_L b$, and
- $\forall_{i \in \{2,\ldots,n\}} \forall_{b \in A_i} \exists_{a \in A_L^{start}}\ b \mapsto_L a \Rightarrow \forall_{a' \in A_L^{start}}\ b \mapsto_L a'$.



```
C11  (L) TOTAL                                C!
                                              25

             A              B             C             D
 1         ITEM            NO.          UNIT          COST
 2         ----            ---          ----          ----
 3MUCK RAKE                 43         12.95        556.85
 4BUZZ CUT                  15          6.75        101.25
 5TOE TONER                250         49.95      12487.50
 6EYE SNUFF                  2          4.95          9.90
 7                                            ------------
 8                           SUBTOTAL             13155.50
 9                     9.75% TAX                   1282.66
10                                            ------------
11                           TOTAL                14438.16
12
13
14
15
16
```

More info