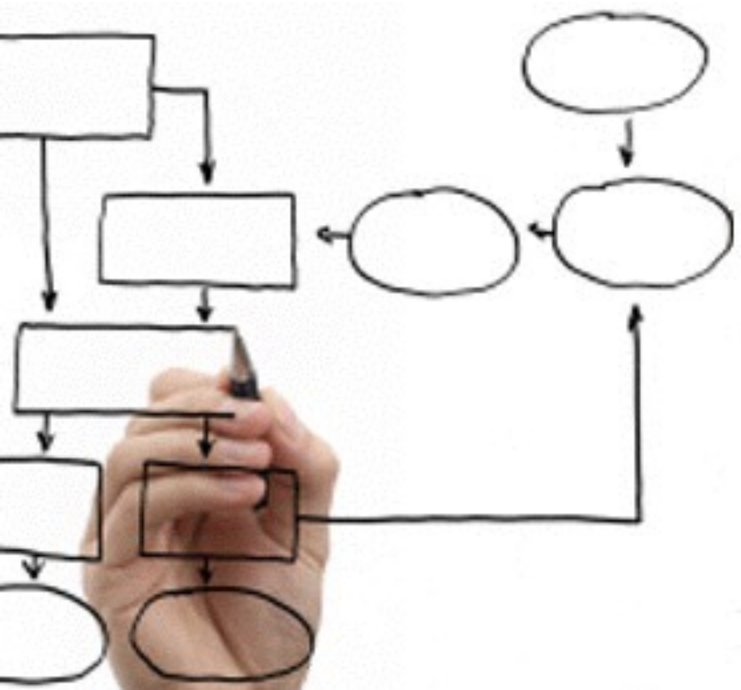
Methods for the specification and verification of business processes

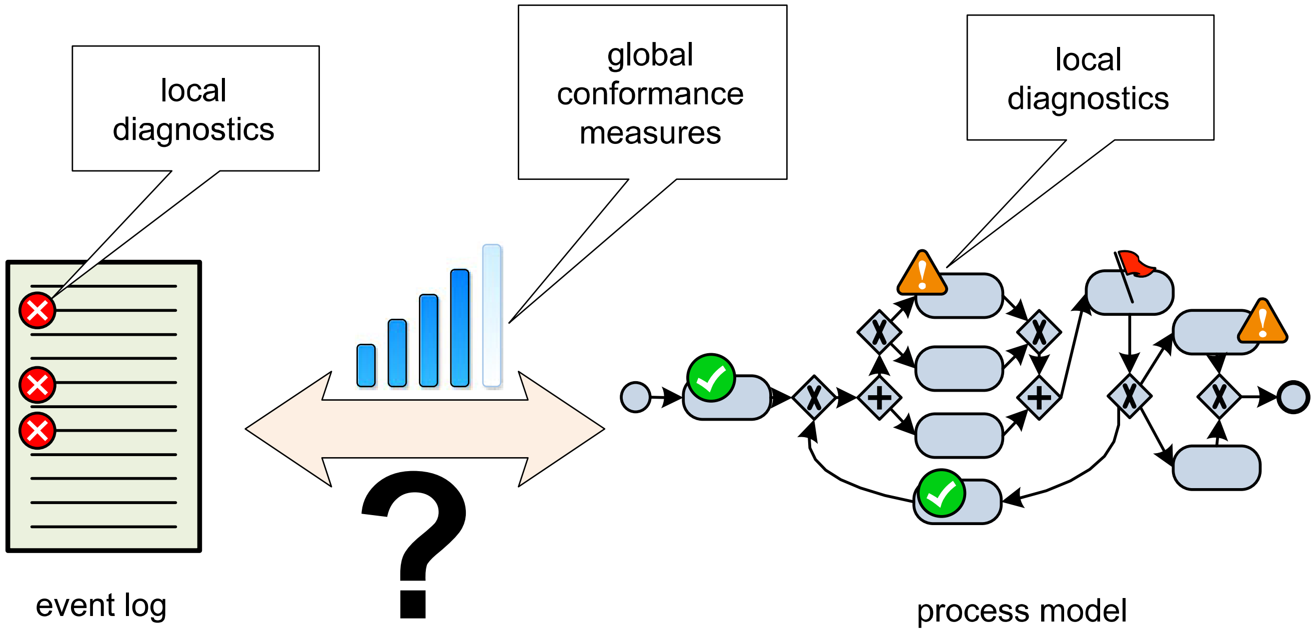
MPB (6 cfu, 295AA)

Roberto Bruni
http://www.di.unipi.it/~bruni



26 - Process Mining

Object



We overview the key principles of process mining

Chapters 1, 5, 7. Process Mining. W. van der Aalst

Process Mining

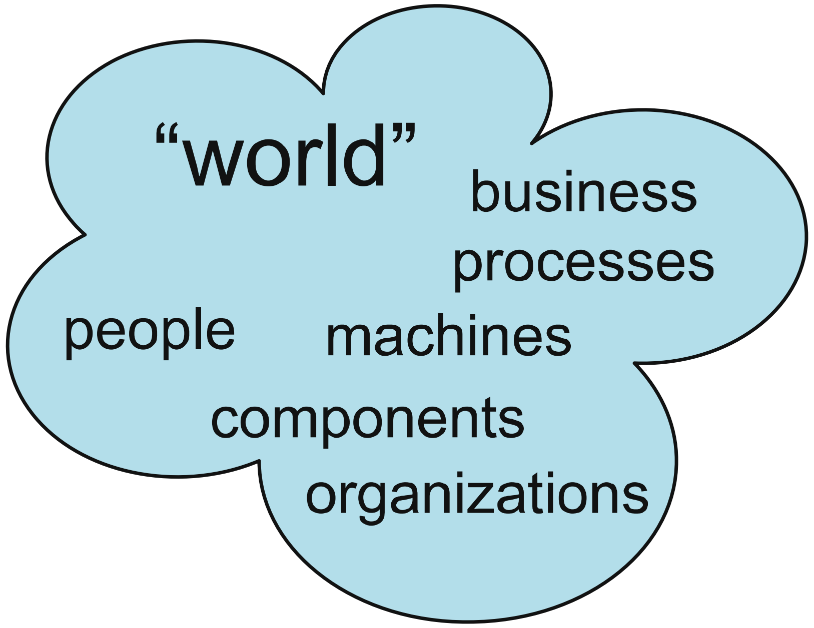
Process mining is a relative young research discipline
that sits between

machine learning and data mining on the one hand

and process modeling and analysis on the other hand.

The idea of process mining is to discover, monitor and
improve real processes (i.e., not hypothetical ones)
by **extracting knowledge from event logs**
readily available in today's systems.

Process Mining Scheme



“world”

business processes

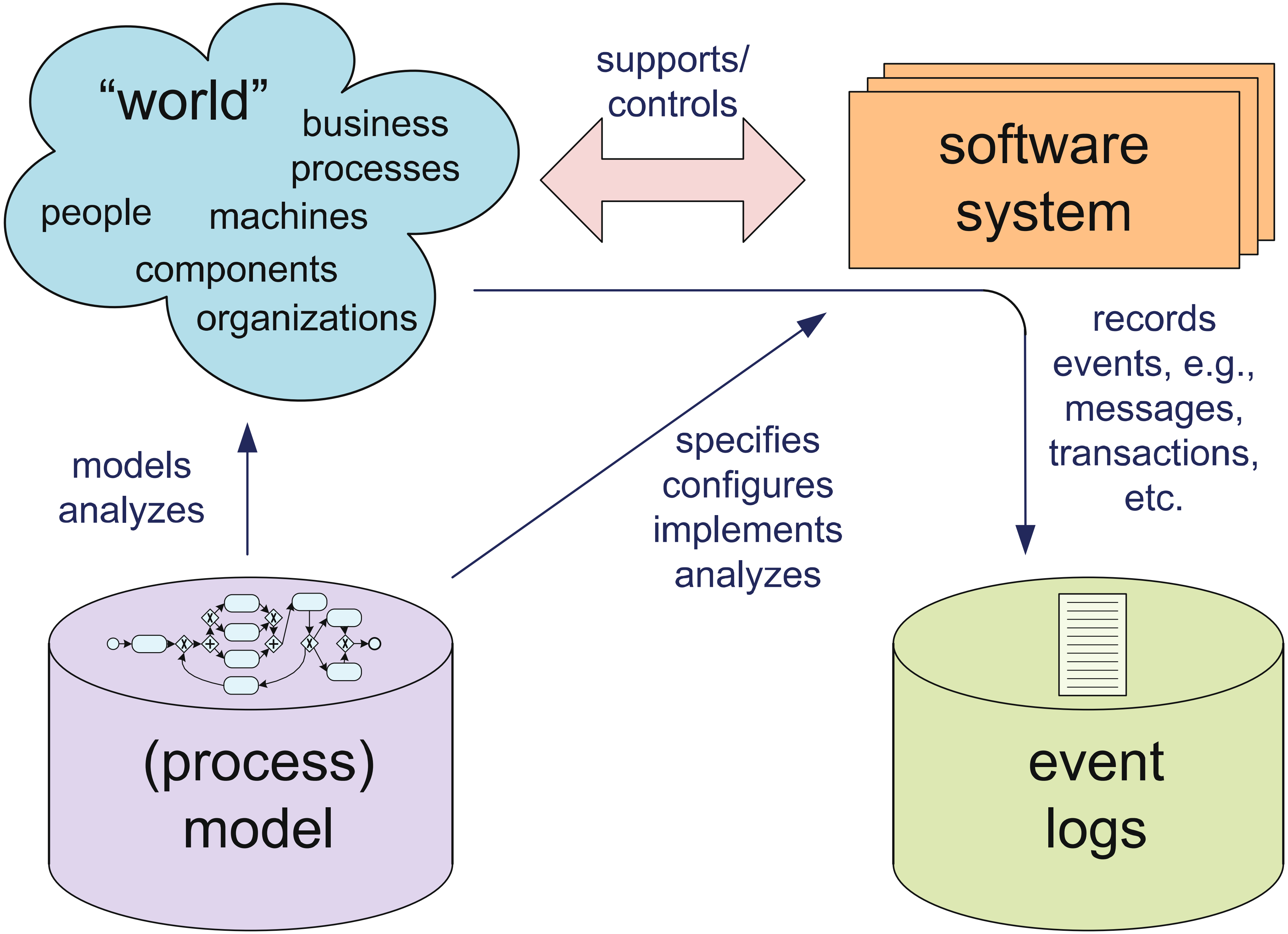
people

machines

components

organizations

Process Mining Scheme



"world"

business processes

people

machines

components

organizations

supports/ controls

software system

models analyzes

specifies configures implements analyzes

records events, e.g., messages, transactions, etc.

(process) model

event logs

Processes, Cases, Events, Attributes

A process consists of cases.

A case consists of events such that each event relates to precisely one case.

Events within a case are ordered in time.

Events can have attributes.

Examples of typical attribute names are activity, time, cost, and resource.

Event Logs

Let us assume that it is possible
to sequentially record events
such that each event:

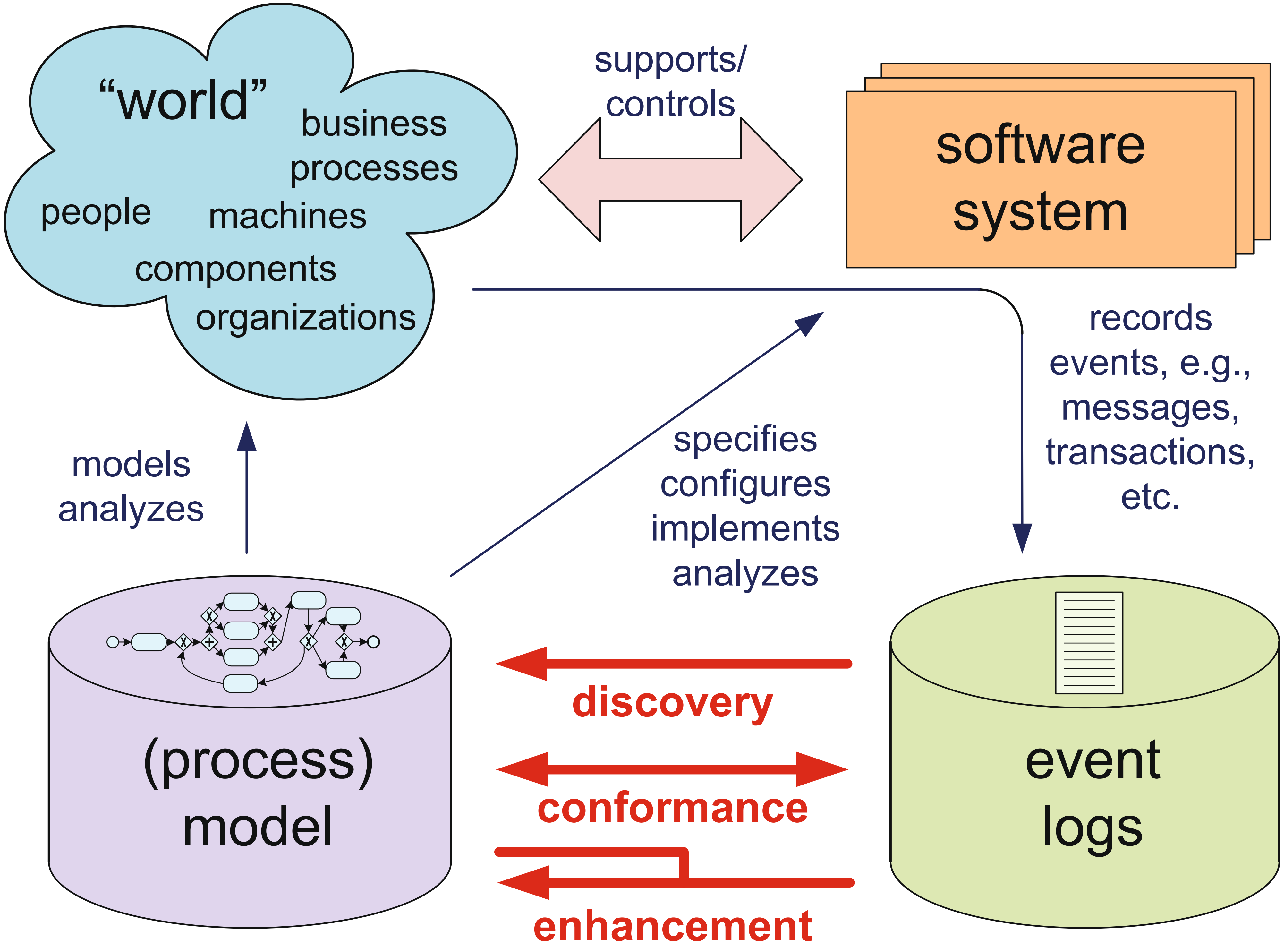
refers to an activity
(i.e., a well-defined step in the process)

and is related to a particular case
(i.e., a process instance).

Event Log Example

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	…
1	35654423	30-12-2010:11.02	Register request	Pete	50	…
	35654424	31-12-2010:10.06	Examine thoroughly	Sue	400	…
	35654425	05-01-2011:15.12	Check ticket	Mike	100	…
	35654426	06-01-2011:11.18	Decide	Sara	200	…
	35654427	07-01-2011:14.24	Reject request	Pete	200	…
2	35654483	30-12-2010:11.32	Register request	Mike	50	…
	35654485	30-12-2010:12.12	Check ticket	Mike	100	…
	35654487	30-12-2010:14.16	Examine casually	Pete	400	…
	35654488	05-01-2011:11.22	Decide	Sara	200	…
	35654489	08-01-2011:12.05	Pay compensation	Ellen	200	…

Process Mining Scheme



"world"
business processes
people
machines
components
organizations

supports/ controls

software system

records events, e.g., messages, transactions, etc.

models analyzes

specifies configures implements analyzes

(process) model

event logs

discovery

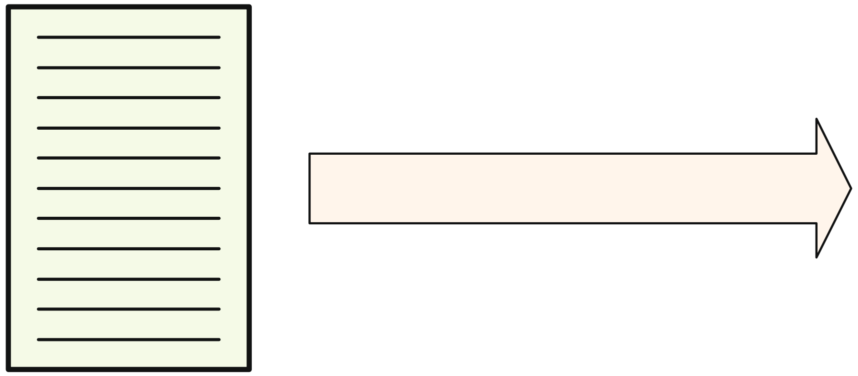
conformance

enhancement

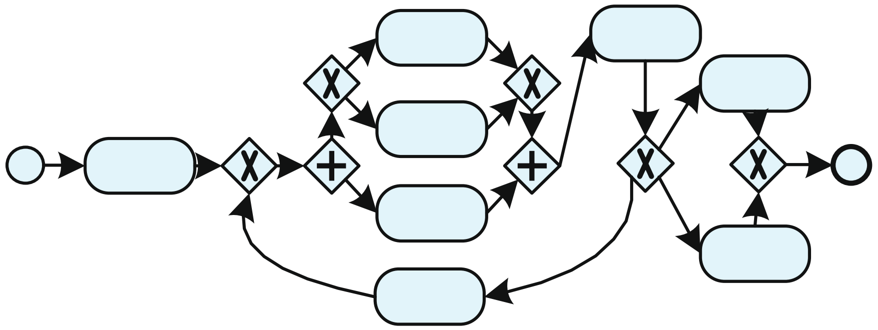
Discovery

A **discovery** technique
takes an event log and produces a model
(without using any a-priori information)

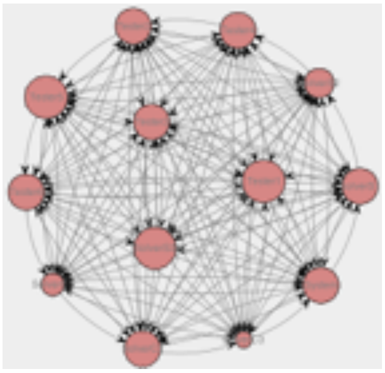
If the event log contains information about resources,
one can also discover resource-related models,
e.g., a social network
showing how people work together in an organization.



event log



process model



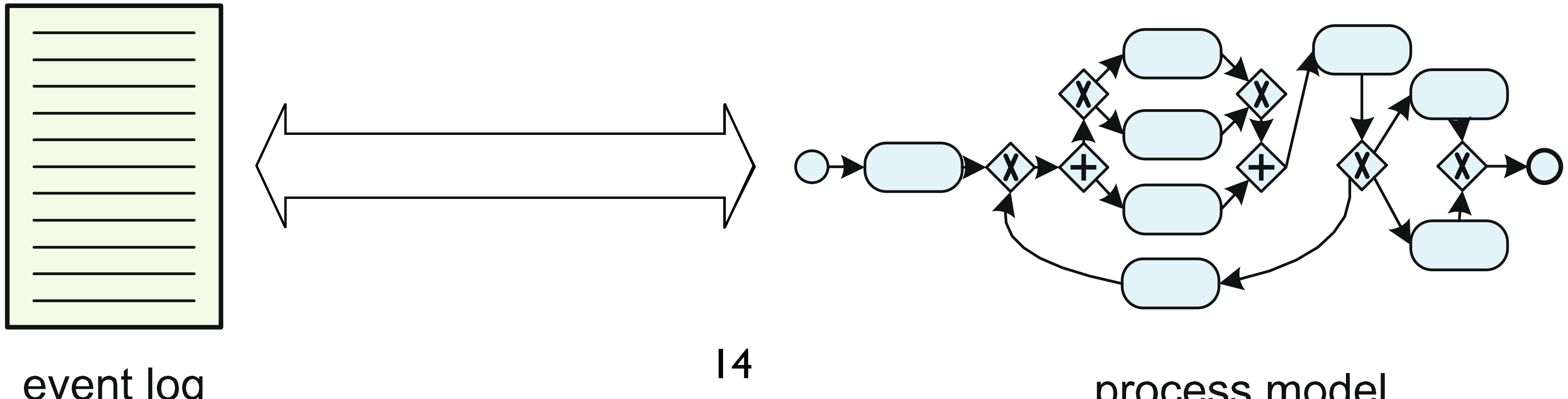
social network

Conformance

Conformance checking can be used to check
if reality, as recorded in the log, conforms to the model
and vice versa.

An existing process model is compared with an event log.

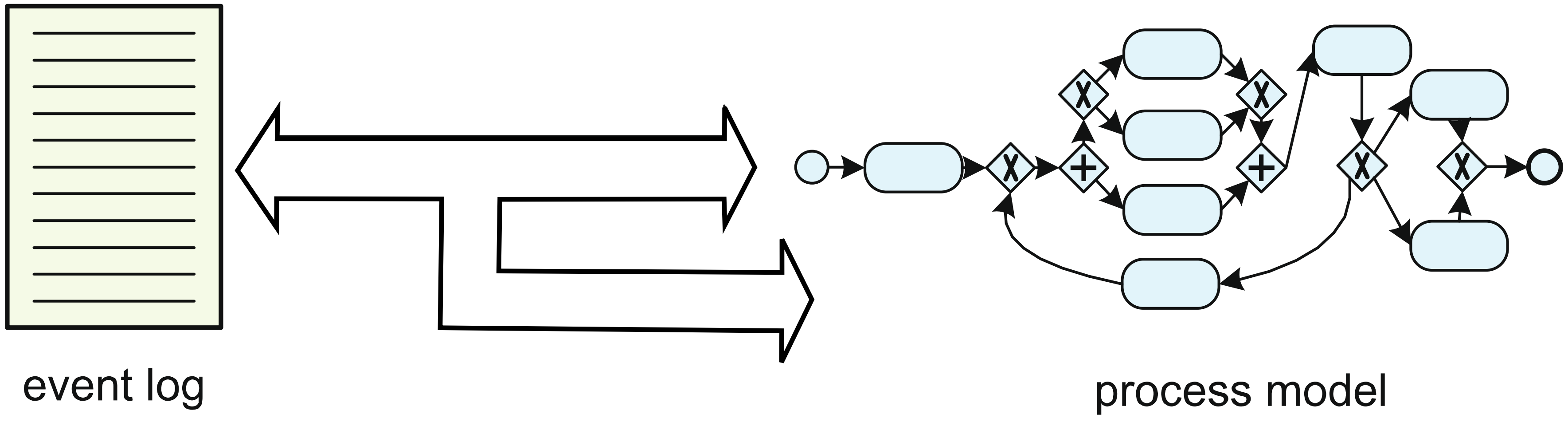
Conformance checking may be used
to detect, locate and explain deviations, and
to measure the severity of these deviations.



Enhancement

Whereas conformance checking
measures the alignment between a model and reality

enhancement aims to
extend/improve an existing process model
using information about the actual process
recorded in some event log.



Two Angles

First viewpoint (the model is supposed to be **descriptive**):
the model does not capture the real behavior
(“the model is wrong, how to improve it?”)

Second viewpoint (the model is **normative**)
reality deviates from the desired model
(“the event log is wrong, how to impose control?”).

Enhancement: Repair

One type of enhancement is **repair**,
i.e., modifying the model to better reflect reality.

For example, if two activities are modeled sequentially
but in reality can happen in any order,
then the model may be corrected to reflect this.

Business Alignment

The goal of business alignment is
to make sure that the **information systems** and
the **real business processes** are **well aligned**.
(People should be supported by the information system
rather than work behind its back to get things done)

Process mining can assist in improving the alignment of
information systems, business processes, and organization.

By analyzing the real processes and diagnosing
discrepancies, new insights can be gathered showing how to
improve the support by information systems.

Auditing

Audits are performed to ascertain
the validity and reliability of information about
organizations and processes.

This is done to check whether business processes are
executed within certain boundaries set by managers,
governments, and other stakeholders.

Rules violations may indicate
fraud, malpractice, risks, and inefficiencies.

The availability of log data and advanced
process mining techniques enables new forms of auditing

Four Perspectives

Organizational Perspective

The **organizational perspective** focuses on information about resources, i.e., which actors (e.g., people, systems, roles, and departments) are involved and how they are related.

The goal can be to structure the organization by classifying people in terms of roles and organizational units or to show the social network.

Case Perspective

The **case perspective** focuses on properties of cases.

Obviously, a case can be characterized by its path in the process or by the originators working on it. However, cases can also be characterized by the values of the corresponding data elements.

For example, it can be interesting to find out commonalities between cases that conclude with negative outcomes

Time Perspective

The **time perspective** is concerned with the timing and frequency of events (performance checking).

When events bear timestamps it is possible to discover bottlenecks, measure service levels, monitor the utilization of resources, and predict the remaining processing time of running cases.

Control-Flow Perspective

The **control-flow perspective** focuses on the control-flow, i.e., the ordering of activities.

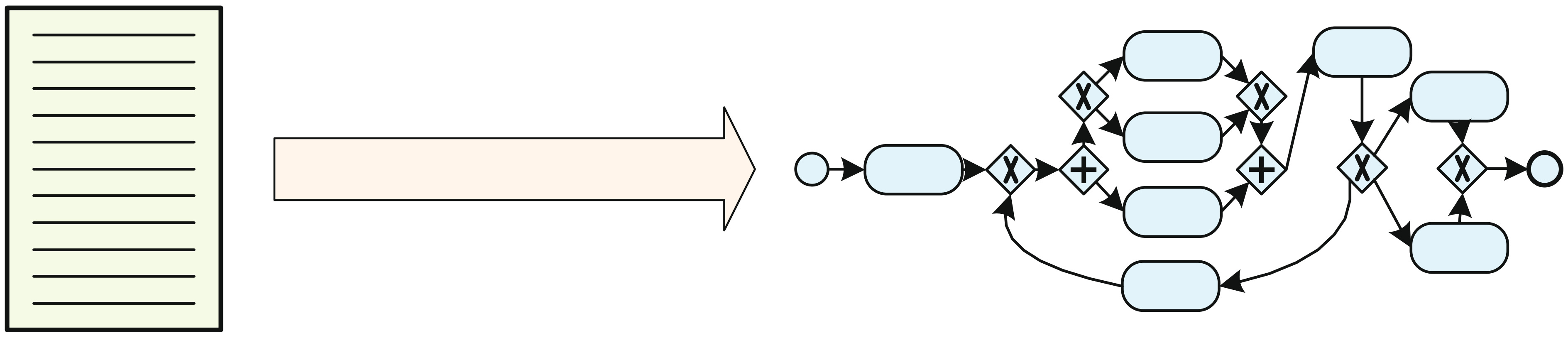
The goal of mining this perspective is to find a good characterization of all possible paths, e.g., expressed in terms of a Petri net or some other notation (e.g., EPC, BPMN, and UML AD).

We shall focus on this perspective only

Three Strategies

Play-in

Play-In

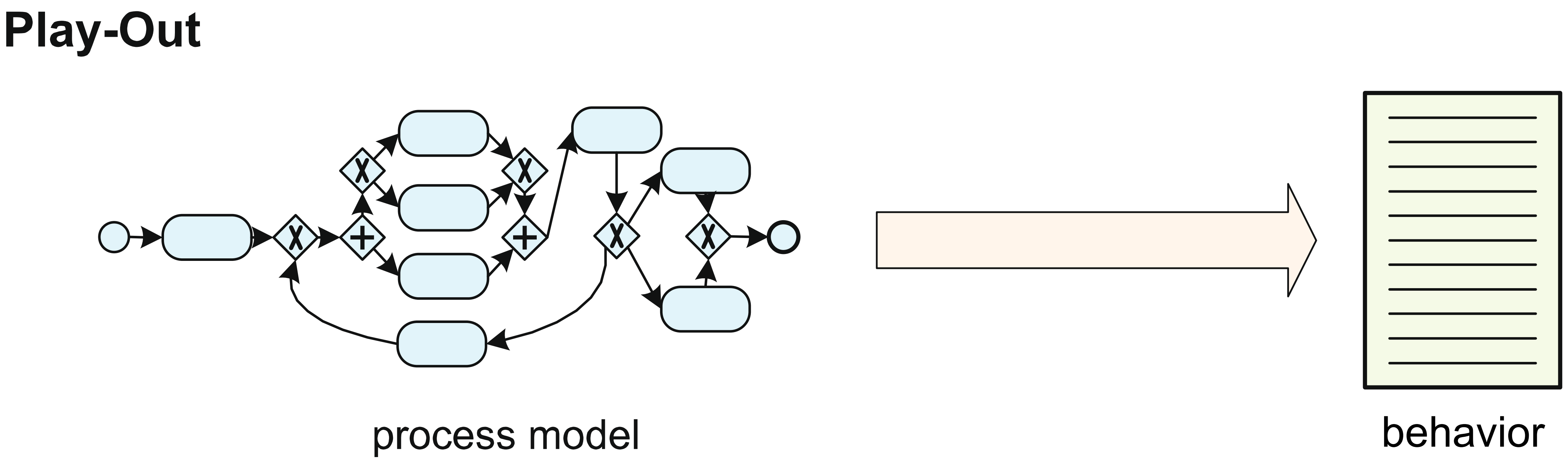


event log

process model

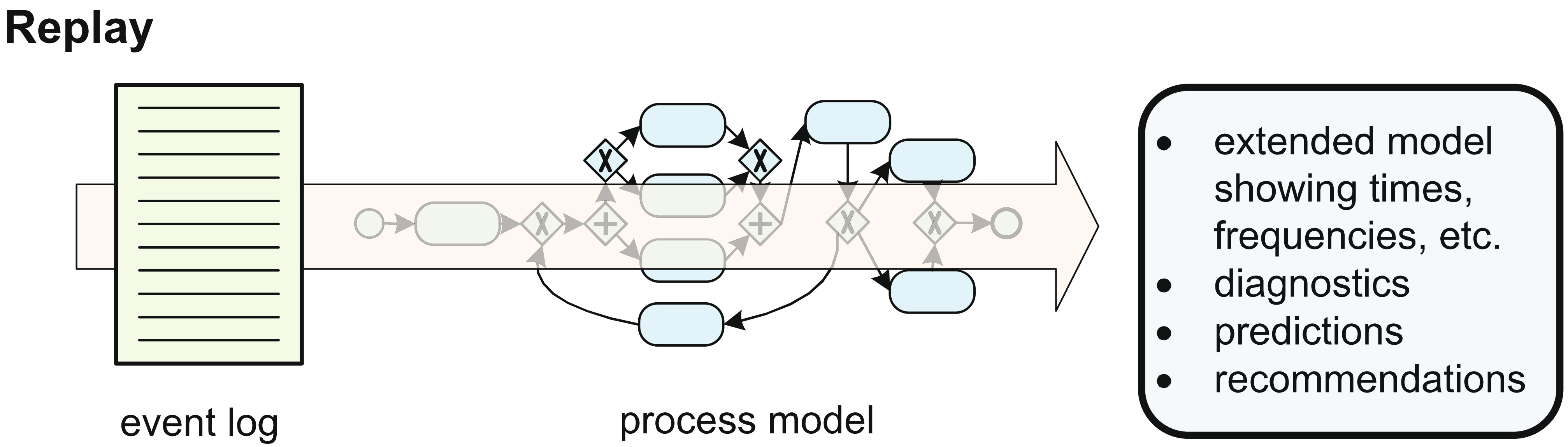
Mining
Discovery

Play-out



Workflow engine
Simulation engine
Trace generation
Model checking

Replay



- extended model showing times, frequencies, etc.
- diagnostics
- predictions
- recommendations

Conformance checking
Performance analysis
Bottlenecks detection
Predictive models (based on past)
Operational support (deviation detection)

Discovery and conformance: an example

Event Log Fragment

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	…
1	35654423	30-12-2010:11.02	Register request	Pete	50	…
	35654424	31-12-2010:10.06	Examine thoroughly	Sue	400	…
	35654425	05-01-2011:15.12	Check ticket	Mike	100	…
	35654426	06-01-2011:11.18	Decide	Sara	200	…
	35654427	07-01-2011:14.24	Reject request	Pete	200	…
2	35654483	30-12-2010:11.32	Register request	Mike	50	…
	35654485	30-12-2010:12.12	Check ticket	Mike	100	…
	35654487	30-12-2010:14.16	Examine casually	Pete	400	…
	35654488	05-01-2011:11.22	Decide	Sara	200	…
	35654489	08-01-2011:12.05	Pay compensation	Ellen	200	…

Two cases

Two traces

Ten (totally ordered) events

Event Log Fragment

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	. . .
1			Register request			
			Examine thoroughly			
			Check ticket			
			Decide			
			Reject request			
2			Register request			
			Check ticket			
			Examine casually			
			Decide			
			Pay compensation			

Event Log Fragment

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	...
1			a Register request			
			b Examine thoroughly			
			d Check ticket			
			e Decide			
			h Reject request			
2			a Register request			
			d Check ticket			
			c Examine casually			
			e Decide			
			g Pay compensation			

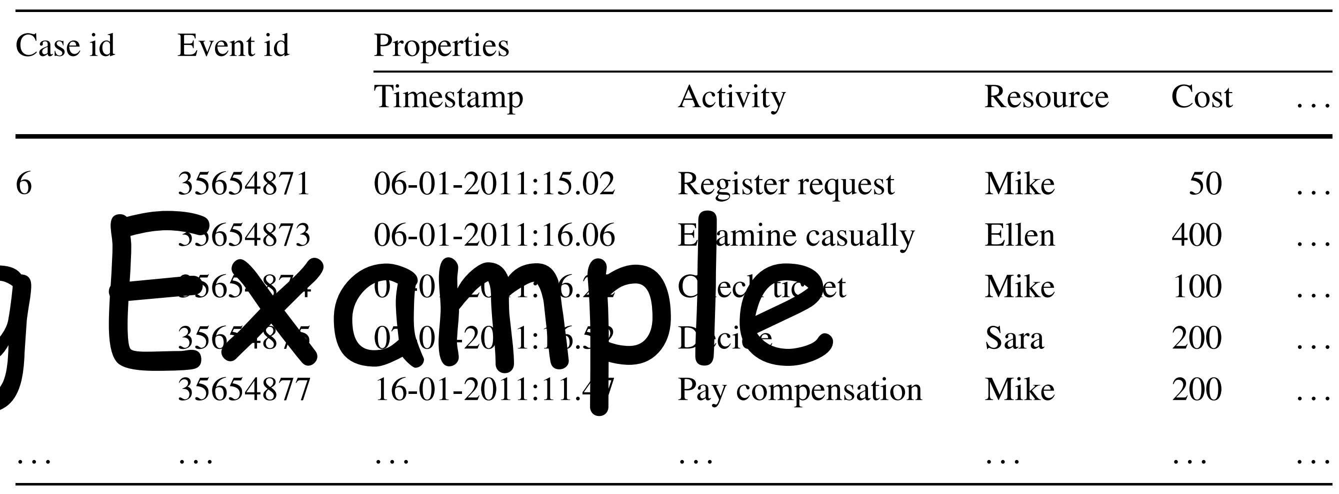
Event Log Fragment

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	...
1			a			
			b			
			d			
			e			
			h			
2			a			
			d			
			c			
			e			
			g			

Event Log Fragment

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	...
1			⟨a b d e h⟩			
2			⟨a d c e g⟩			

Event Log Example

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	…
1	35654423	30-12-2010:11.02	Register request	Pete	50	…
	35654424	31-12-2010:10.06	Examine thoroughly	Sue	400	…
	35654425	05-01-2011:1[illegible].12	Check ticket	Mike	100	…
	35654426	06-01-2011:1[illegible].18	Decide	Sara	200	…
	35654427	07-01-2011:14.24	Reject request	Pete	200	…
2	35654483	30-12-2010:11.32	Register request	Mike	50	…
	35654485	30-12-2010:12.12	Check ticket	Mike	100	…
	35654487	30-12-2010:14.16	Examine casually	Pete	400	…
	35654488	05-01-2011:11.22	Decide	Sara	200	…
	35654489	08-01-2011:12.05	Pay compensation	Ellen	200	…
3	35654521	30-12-2010:14.32	Register request	Pete	50	…
	35654522	30-12-2010:15.06	Examine casually	Mike	400	…
	35654524	30-12-2010:16.34	Check ticket	Ellen	100	…
	35654525	06-01-2011:09.18	Decide	Sara	200	…
	35654526	06-01-2011:12.18	Reinitiate request	Sara	200	…
	35654527	06-01-2011:13.06	Examine thoroughly	Sean	400	…
	35654530	08-01-2011:11.43	Check ticket	Pete	100	…
	35654531	09-01-2011:09.55	Decide	Sara	200	…
	35654533	15-01-2011:10.45	Pay compensation	Ellen	200	…
4	35654641	06-01-2011:15.02	Register request	Pete	50	…
	35654643	07-01-2011:12.06	Check ticket	Mike	100	…
	35654644	08-01-2011:14.43	Examine thoroughly	Sean	400	…
	35654645	09-01-2011:12.02	Decide	Sara	200	…
	35654647	12-01-2011:15.44	Reject request	Ellen	200	…
5	35654711	06-01-2011:09.02	Register request	Ellen	50	…
	35654712	07-01-2011:10.16	Examine casually	Mike	400	…
	35654714	08-01-2011:11.22	Check ticket	Pete	100	…
	35654715	10-01-2011:13.28	Decide	Sara	200	…
	35654716	11-01-2011:16.18	Reinitiate request	Sara	200	…
	35654718	14-01-2011:14.33	Check ticket	Ellen	100	…
	35654719	16-01-2011:15.50	Examine casually	Mike	400	…
	35654720	19-01-2011:11.18	Decide	Sara	200	…
	35654721	20-01-2011:12.48	Reinitiate request	Sara	200	…
	35654722	21-01-2011:09.06	Examine casually	Sue	400	…
	35654724	21-01-2011:11.34	Check ticket	Pete	100	…
	35654725	23-01-2011:13.12	Decide	Sara	200	…
	35654726	24-01-2011:14.56	Reject request	Mike	200	…
6	35654871	06-01-2011:15.02	Register request	Mike	50	…
	35654873	06-01-2011:16.06	Examine casually	Ellen	400	…
	35654874	07-01-2011:16.22	Check ticket	Mike	100	…
	35654875	07-01-2011:16.52	Decide	Sara	200	…
	35654877	16-01-2011:11.47	Pay compensation	Mike	200	…
…	…	…	…	…	…	…

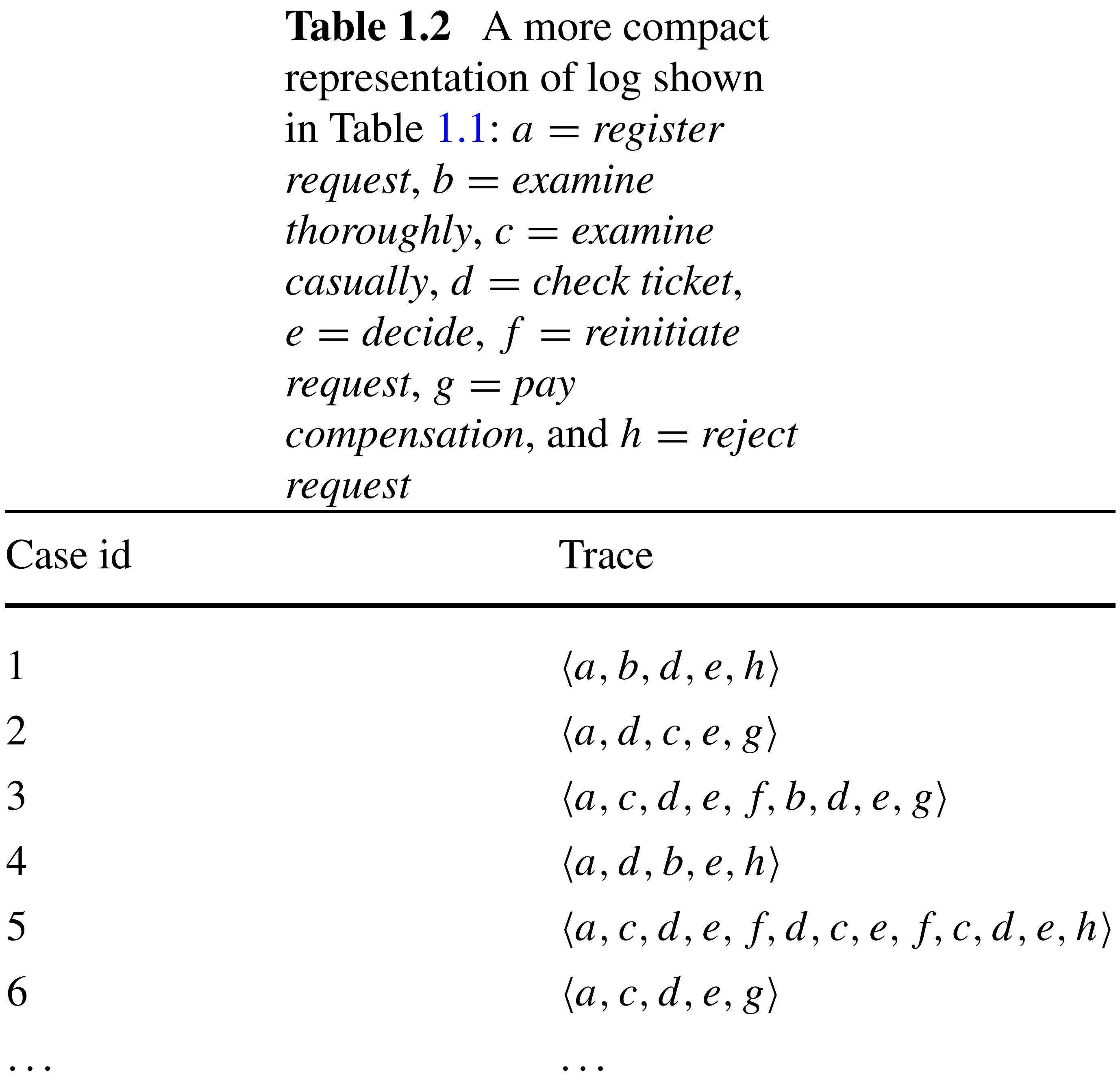


Table 1.2 A more compact representation of log shown in Table 1.1: a = *register request*, b = *examine thoroughly*, c = *examine casually*, d = *check ticket*, e = *decide*, f = *reinitiate request*, g = *pay compensation*, and h = *reject request*

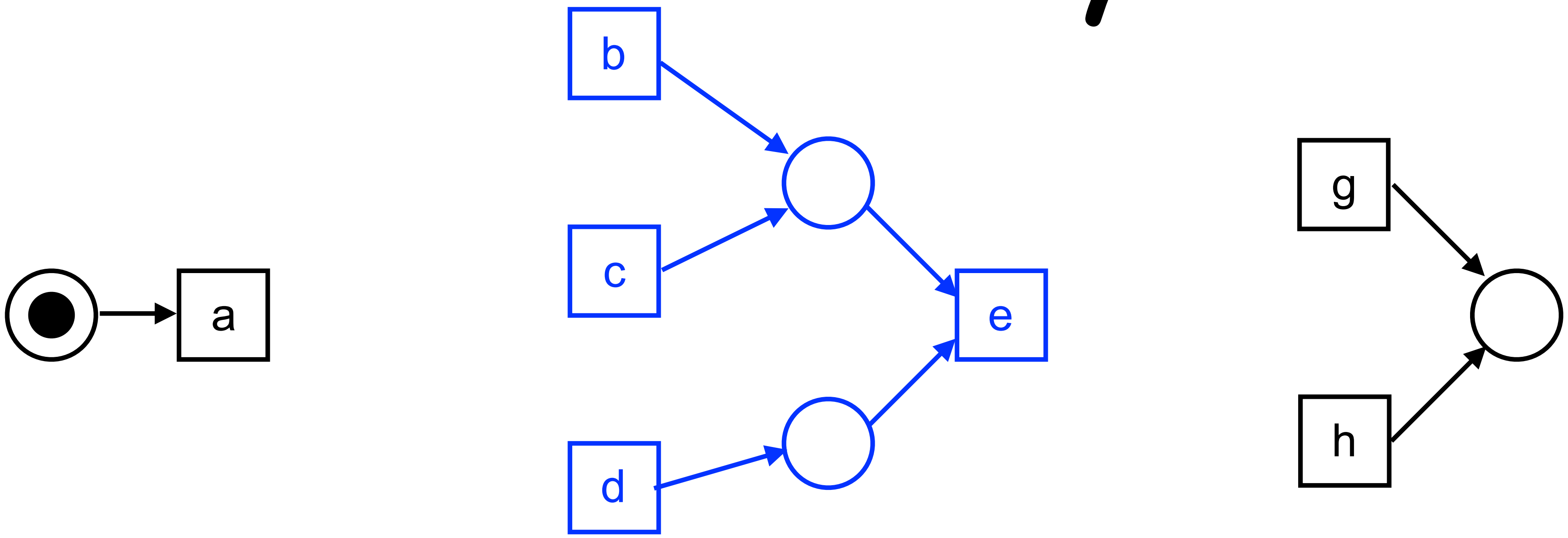
Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
…	…

Discovery Example

All cases start with a and end with either g or h.

Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
...	...

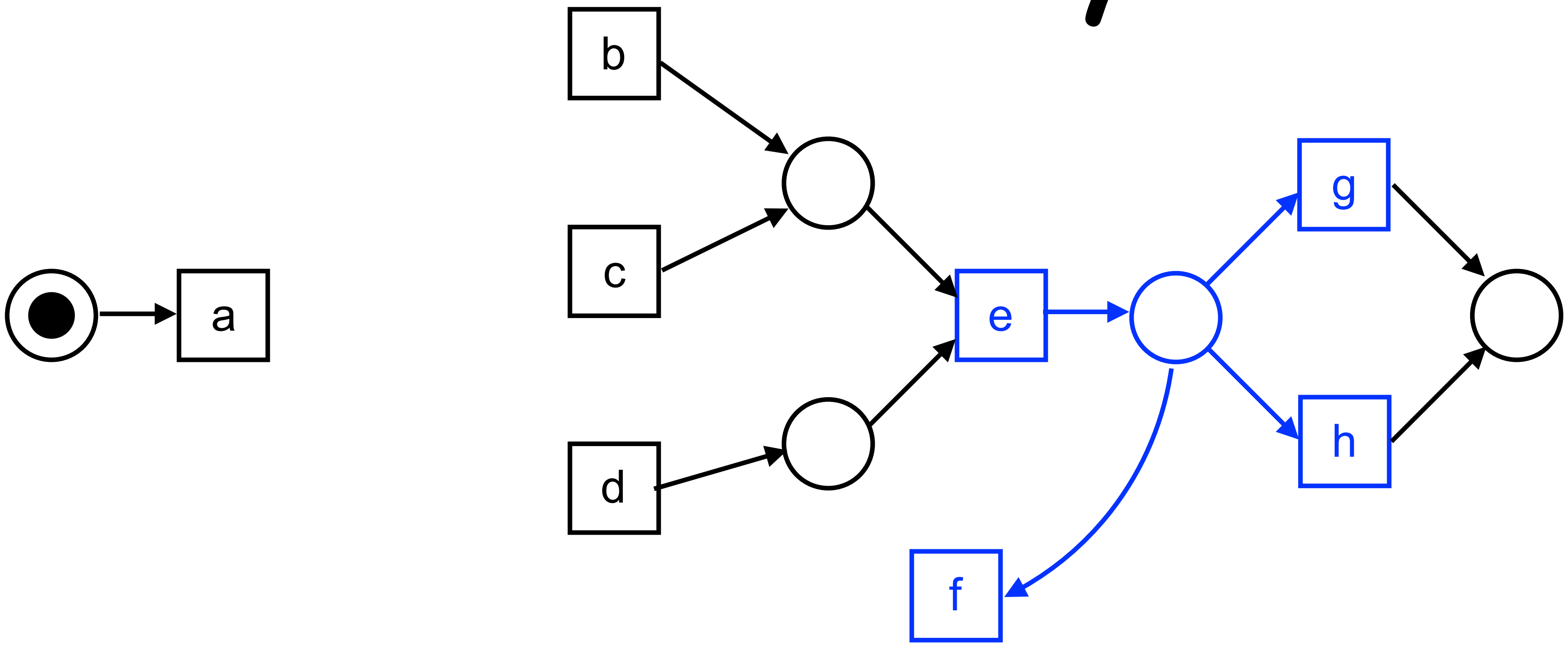
Discovery Example



Every e is preceded by d and one of the examination activities (b or c).

Case id	Trace
1	$\langle a, b, \boxed{d, e}, h\rangle$
2	$\langle a, d, \boxed{c, e}, g\rangle$
3	$\langle a, c, \boxed{d, e}, f, b, \boxed{d, e}, g\rangle$
4	$\langle a, d, \boxed{b, e}, h\rangle$
5	$\langle a, c, \boxed{d, e}, f, d, \boxed{c, e}, f, c, \boxed{d, e}, h\rangle$
6	$\langle a, c, \boxed{d, e}, g\rangle$
...	...

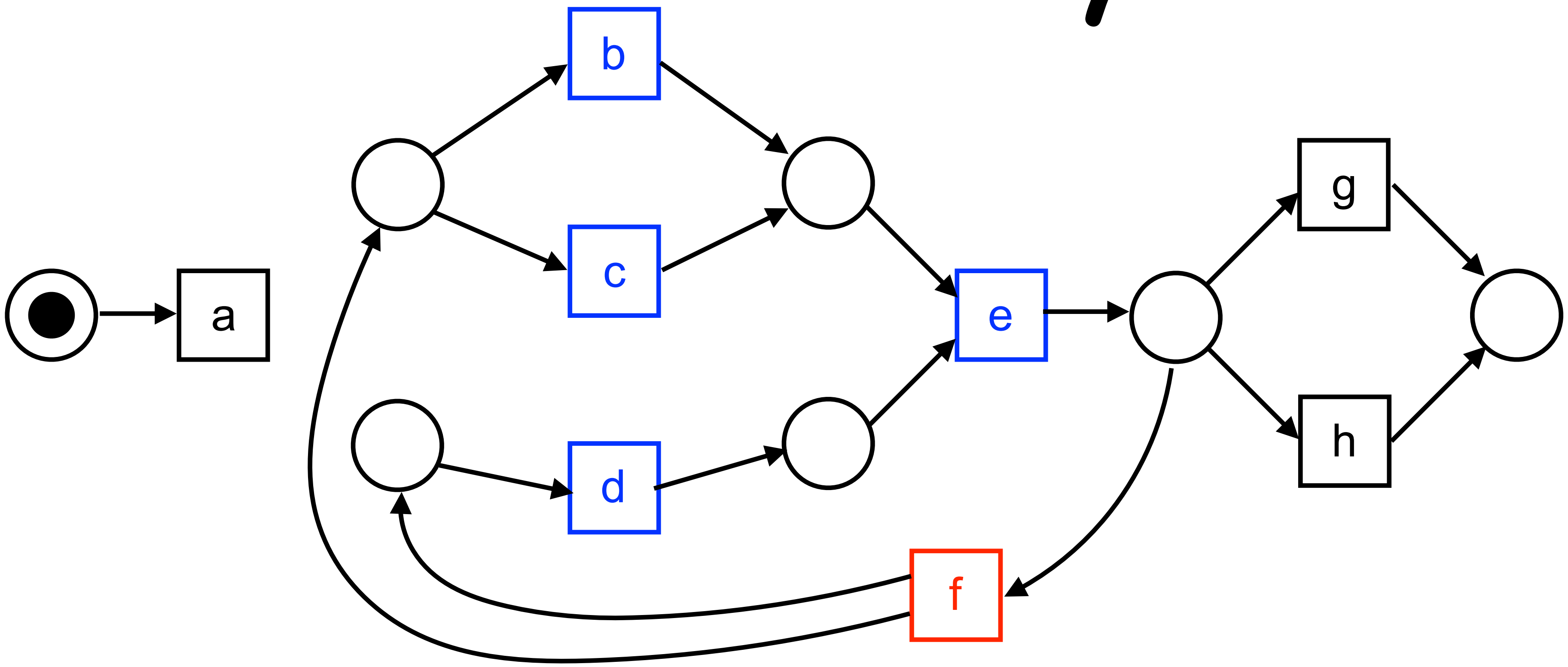
Discovery Example



Moreover, e is always followed by f, g, or h.

Case id	Trace
1	$\langle a, b, d, \boxed{e, h}\rangle$
2	$\langle a, d, c, \boxed{e, g}\rangle$
3	$\langle a, c, d, \boxed{e, f,} b, d, \boxed{e, g}\rangle$
4	$\langle a, d, b, \boxed{e, h}\rangle$
5	$\langle a, c, d, \boxed{e, f,} d, c, \boxed{e, f,} c, d, \boxed{e, h}\rangle$
6	$\langle a, c, d, \boxed{e, g}\rangle$
...	...

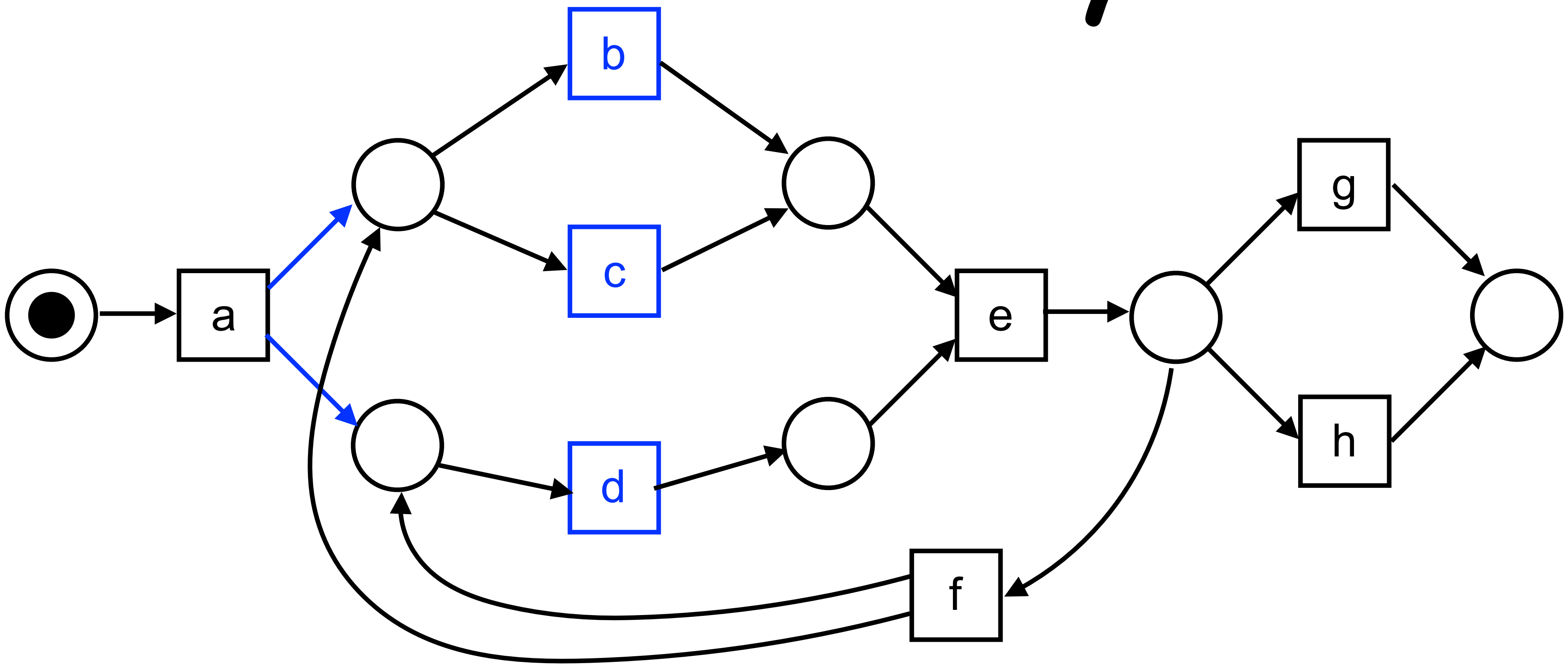
Discovery Example



The repeated execution of b/c, d, and e suggests the presence of a loop (over f).

Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
...	...

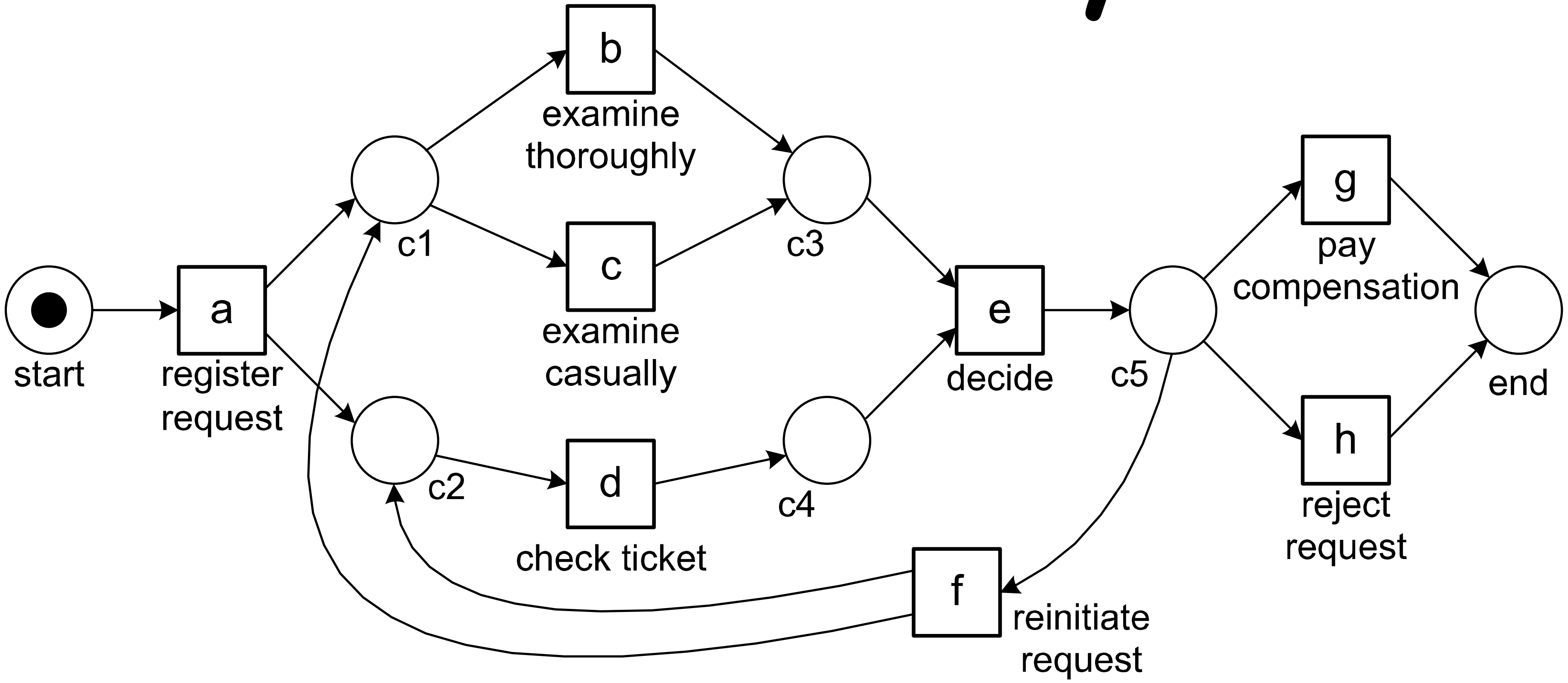
Discovery Example



b/c and d
are executed in any order
(bd,db,cd,dc)
which suggests they are
executed in parallel

Case id	Trace
1	$\langle a, b, d, e, h\rangle$
2	$\langle a, d, c, e, g\rangle$
3	$\langle a, c, d, e, f, b, d, e, g\rangle$
4	$\langle a, d, b, e, h\rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h\rangle$
6	$\langle a, c, d, e, g\rangle$
...	...

Discovery Example



Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
...	...

Log features are adequately captured by the net

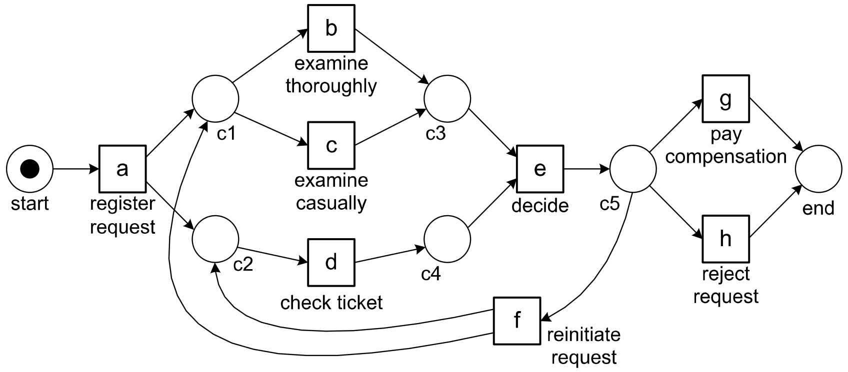
Overfitting and Underfitting

One of the challenges of process mining is
to balance between

overfitting:
the model is too specific
it only allows for the accidental behavior observed

and

underfitting:
the model is too general
it allows for behavior unrelated to the behavior observed



Discussion

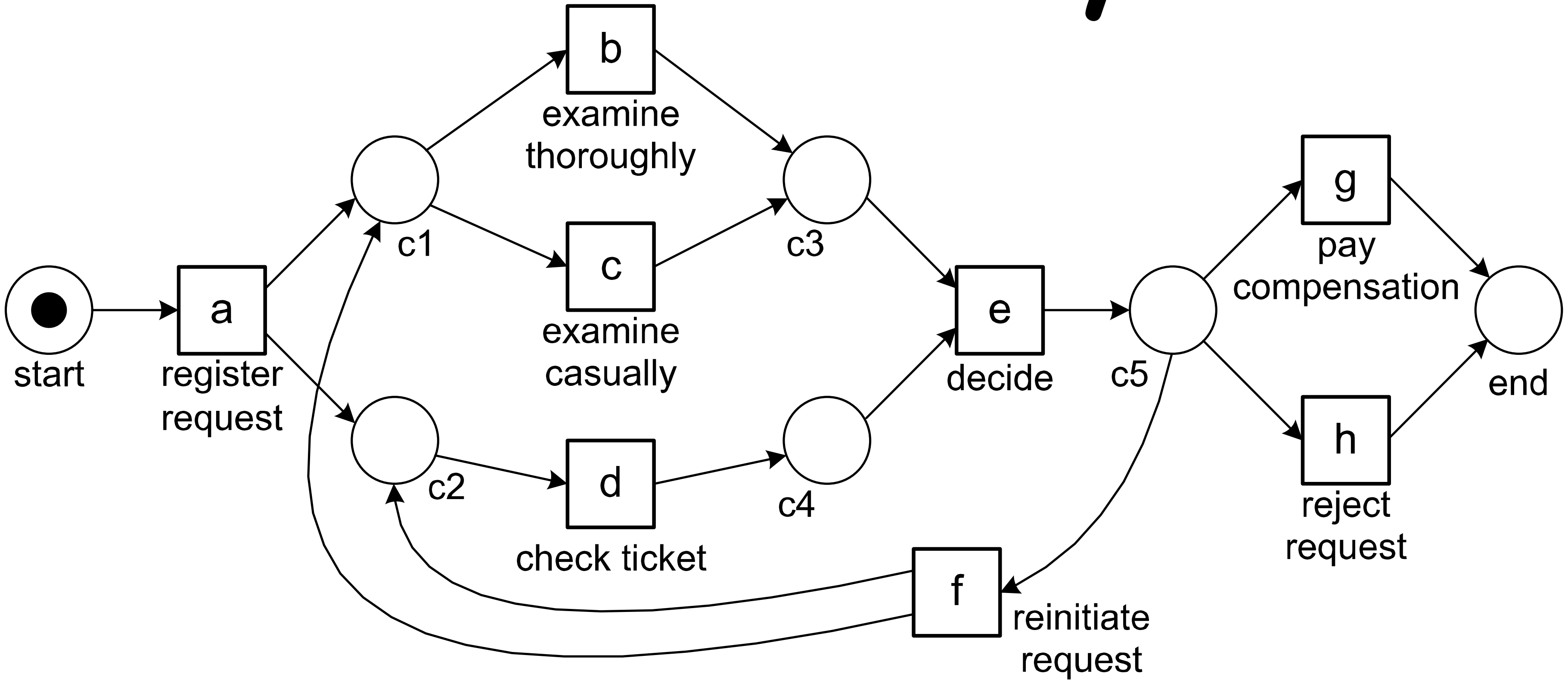
Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
...	...

The discovered net also allows for traces not in the log, e.g.
〈 a, d, c, e, f, b, d, e, g 〉
〈 a, c, d, e, f, c, d, e, f, c, d, e, f, c, d, e, f, b, d, e, g 〉

This is a desired phenomenon:
the goal of a discovery procedure is not to represent exactly the particular set of sample traces in the event log.

Process mining algorithms must generalize the behavior contained in the log to show the most likely underlying model that is not invalidated by the next set of observations

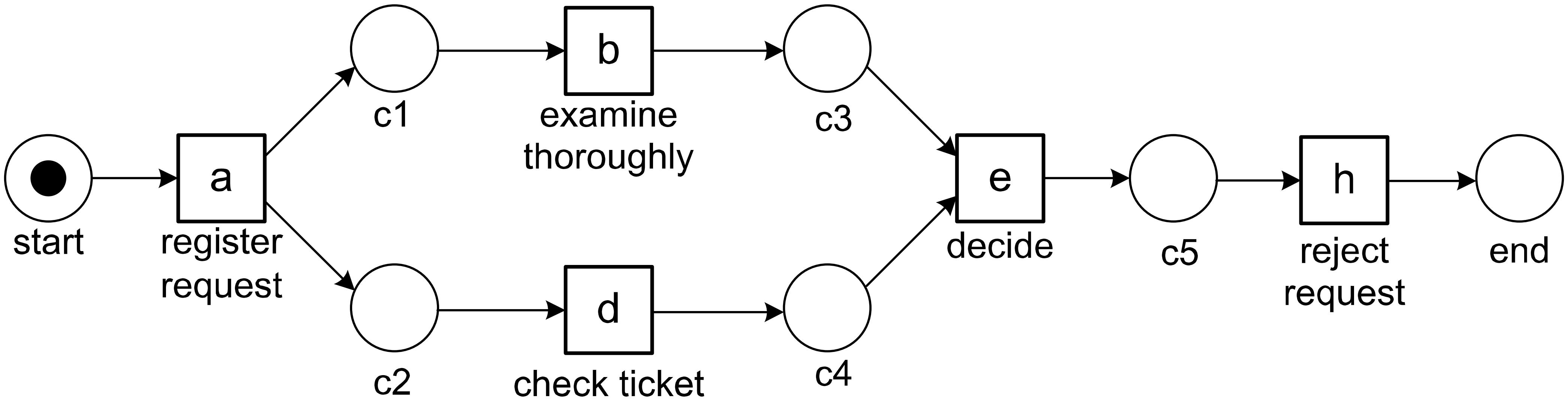
Discovery Example



When comparing the event log and the model, there seems to be a good balance between “overfitting” and “underfitting”.

Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
...	...

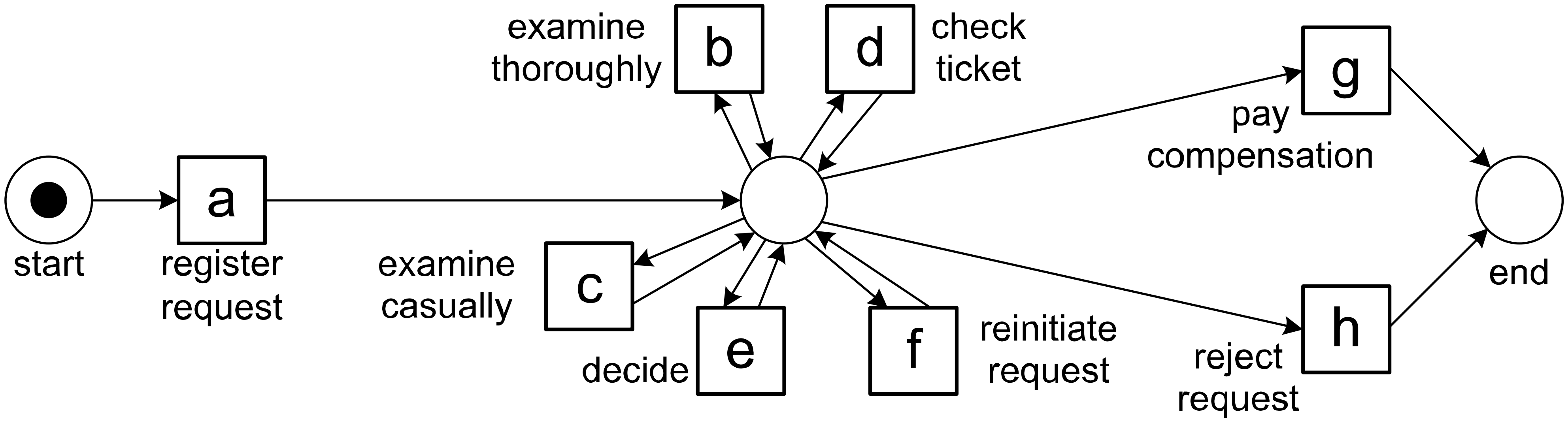
Another Discovery Example



Another net could disallow some traces

Case id	Trace
1	$\langle a, b, d, e, h \rangle$
~~2~~	~~$\langle a, d, c, e, g \rangle$~~
~~3~~	~~$\langle a, c, d, e, f, b, d, e, g \rangle$~~
4	$\langle a, d, b, e, h \rangle$
~~5~~	~~$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$~~
~~6~~	~~$\langle a, c, d, e, g \rangle$~~
...	...

Another Discovery Example

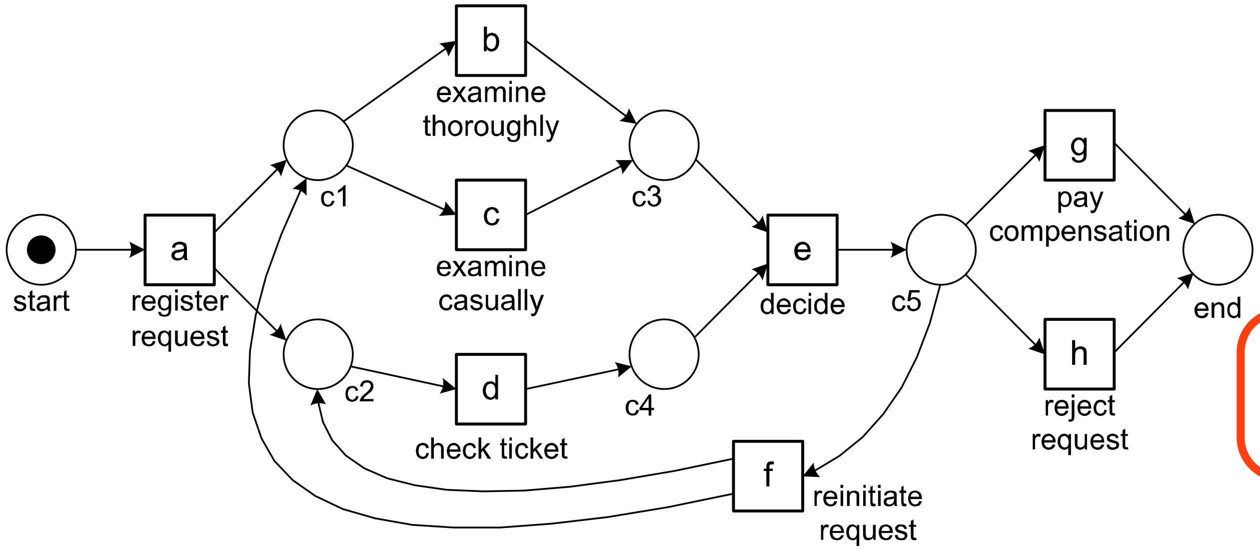


Another net could allow for too many other traces (nets of this kind are called ``flower nets'') and deliver little information about the underlying process

Case id	Trace
1	$\langle a, b, d, e, h\rangle$
2	$\langle a, d, c, e, g\rangle$
3	$\langle a, c, d, e, f, b, d, e, g\rangle$
4	$\langle a, d, b, e, h\rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h\rangle$
6	$\langle a, c, d, e, g\rangle$
...	...

Conformance Example

We would like to measure the ``conformance'' between a net and en event log (how well they pair together)

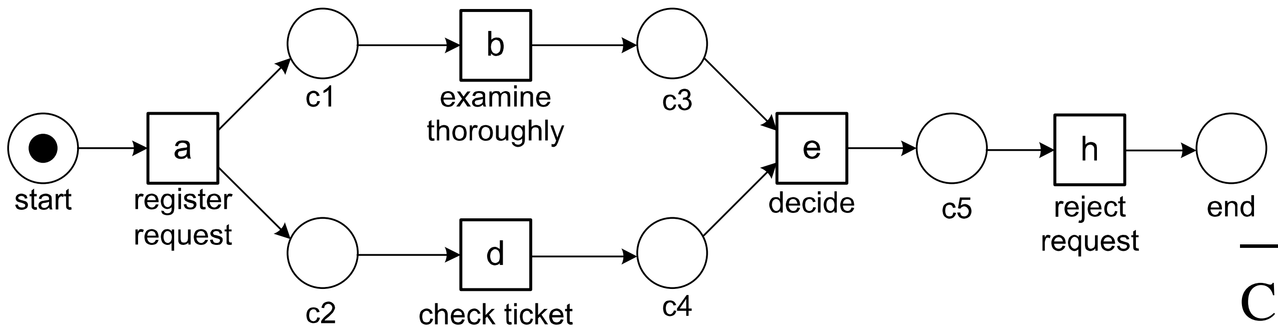


7 ok out of 10

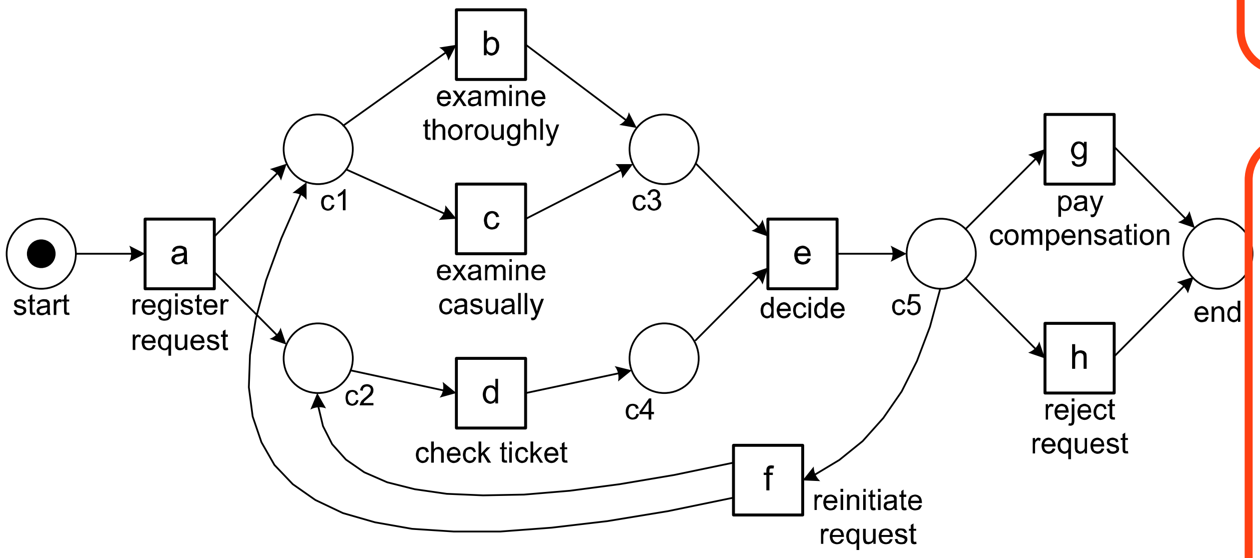
Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
7	$\langle \mathbf{a}, \mathbf{b}, \mathbf{e}, \mathbf{g} \rangle$
8	$\langle \mathbf{a}, \mathbf{b}, \mathbf{d}, \mathbf{e} \rangle$
9	$\langle a, d, c, e, f, d, c, e, f, b, d, e, h \rangle$
10	$\langle \mathbf{a}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}, \mathbf{b}, \mathbf{d}, \mathbf{g} \rangle$

Conformance Example

We would like to measure the ``conformance'' between a net and en event log (how well they pair together)



2 ok out of 10



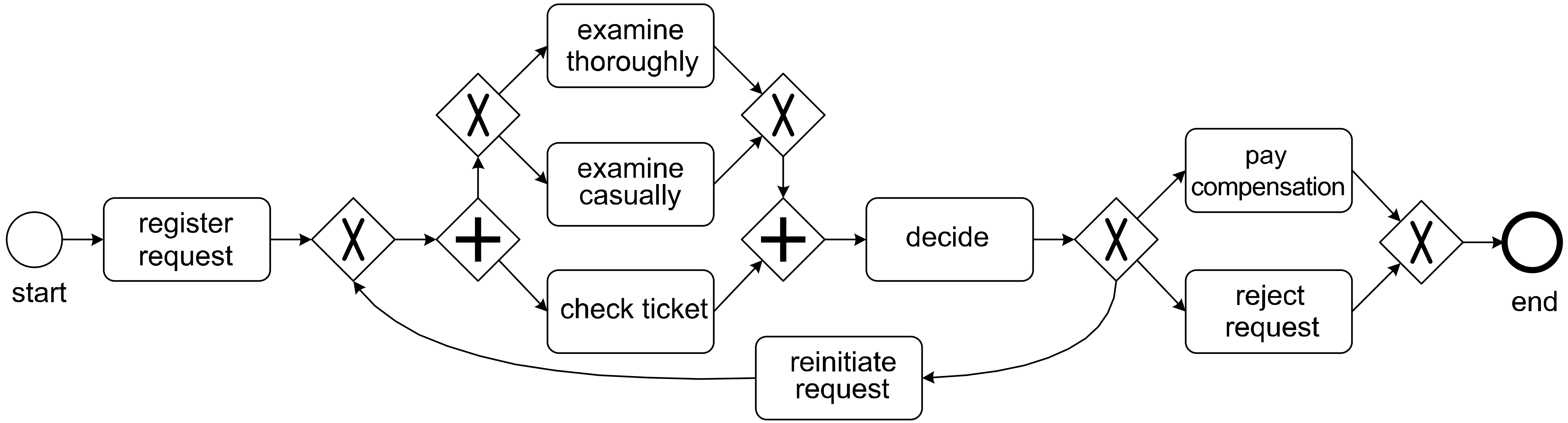
7 ok out of 10

Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
7	$\langle \mathbf{a}, \mathbf{b}, \mathbf{e}, \mathbf{g} \rangle$
8	$\langle \mathbf{a}, \mathbf{b}, \mathbf{d}, \mathbf{e} \rangle$
9	$\langle a, d, c, e, f, d, c, e, f, b, d, e, h \rangle$
10	$\langle \mathbf{a}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}, \mathbf{b}, \mathbf{d}, \mathbf{g} \rangle$

Mining Other Models

We used Petri nets to represent the discovered process models, because Petri nets are a succinct way of representing processes and have unambiguous but intuitive semantics.

However, some mining techniques can apply to other representations as well.



Process Discovery: α-Algorithm

Process Discovery

Process discovery is the activity that combines
Discovery with the Control-flow Perspective.

The general problem:
A **process discovery algorithm** is a function
that maps an event log L onto a process model M
such that the model M is “representative”
for the behavior seen in the event log L.

We focus on *simple event logs* and Petri net models
(possibly sound workflow nets).

Simple Event Log

Let A be a set of activities.

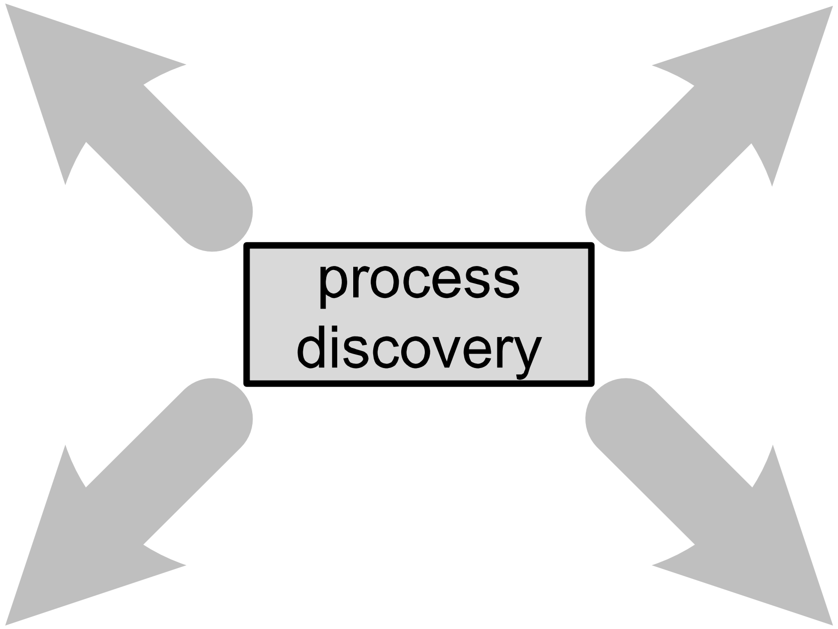
A **simple trace** over A is a finite sequence of activities.

A **simple event log** over A is a multiset of traces.

$$L_1 = \left[\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle\right]$$

$$L_2 = \left[\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^4, \langle a, b, c, e, f, b, c, d\rangle^2, \langle a, b, c, e, f, c, b, d\rangle, \right.$$
$$\left. \langle a, c, b, e, f, b, c, d\rangle^2, \langle a, c, b, e, f, b, c, e, f, c, b, d\rangle\right]$$

Quality Criteria



“able to replay event log”

fitness

Simple structure

“Occam’s razor”

simplicity

process discovery

generalization

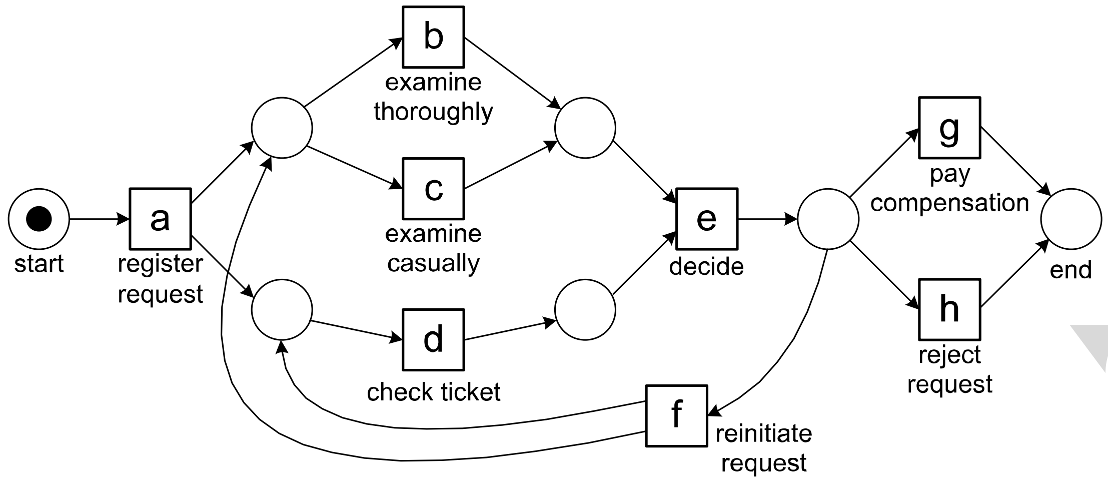
“not overfitting the log”

Other behaviors allowed

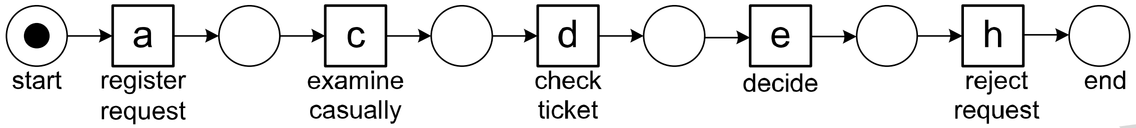
precision

“not underfitting the log”

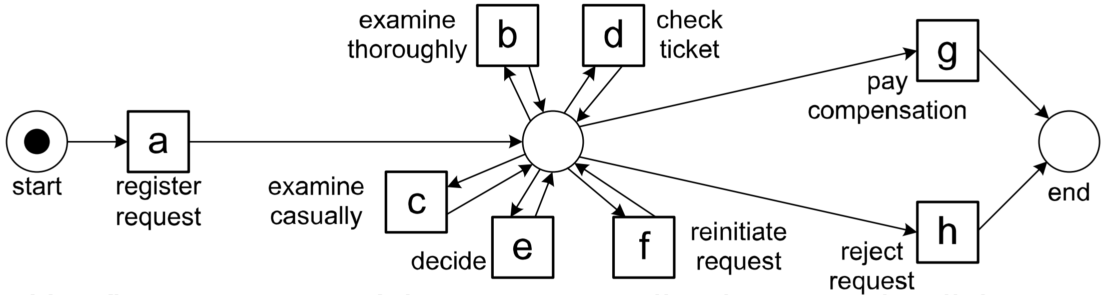
No completely unrelated behavior



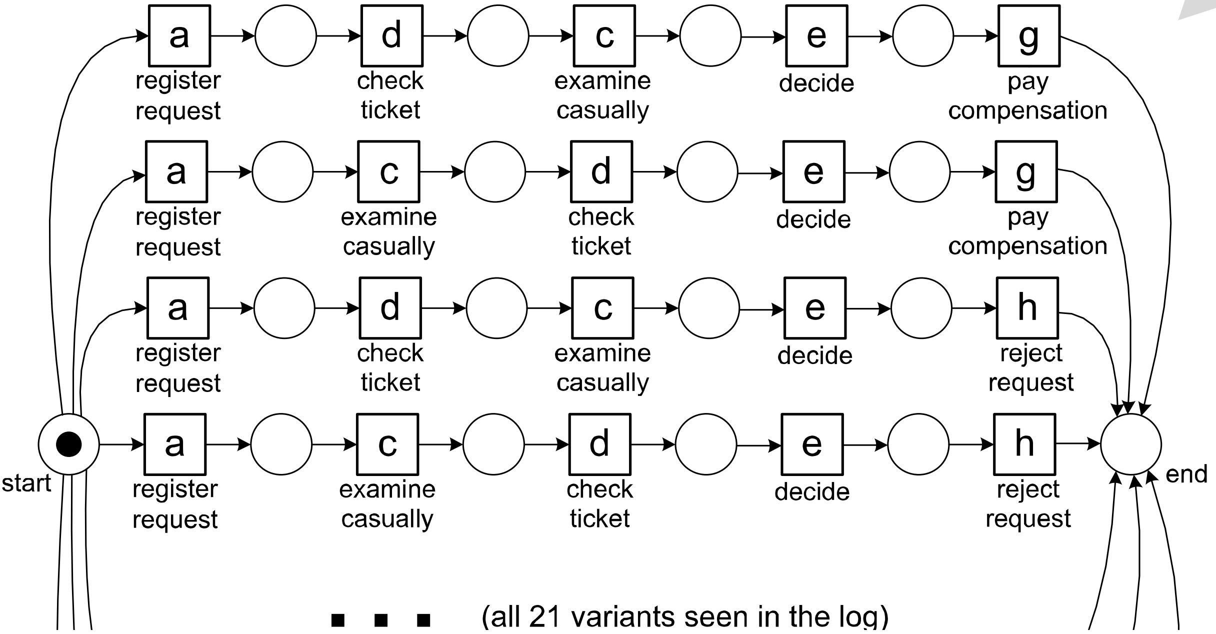
N_1 : fitness = +, precision = +, generalization = +, simplicity = +



N_2 : fitness = -, precision = +, generalization = -, simplicity = +

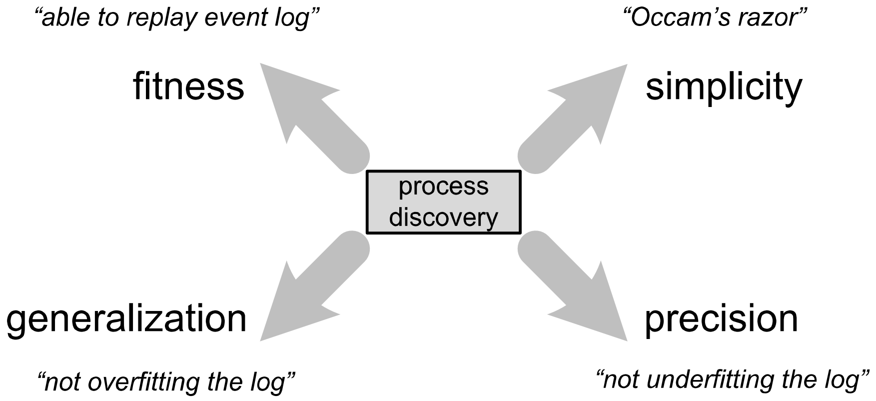


N_3 : fitness = +, precision = -, generalization = +, simplicity = +



N_4 : fitness = +, precision = +, generalization = -, simplicity = -

#	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	



"able to replay event log"

fitness

"Occam's razor"

simplicity

process discovery

generalization

"not overfitting the log"

precision

"not underfitting the log"

Quality Measures

We have seen four quality criteria:
fitness, precision, generalization, and simplicity.

In the example, for each of these models, a subjective judgment is given with respect to the four quality criteria. As the models are rather extreme, the scores +/- for the various quality criteria are evident.

However, in a more realistic setting it would be much more difficult to judge the quality of a model.

We will discuss how the notion of fitness can be quantified.

Question time

Suppose you are given a log with:

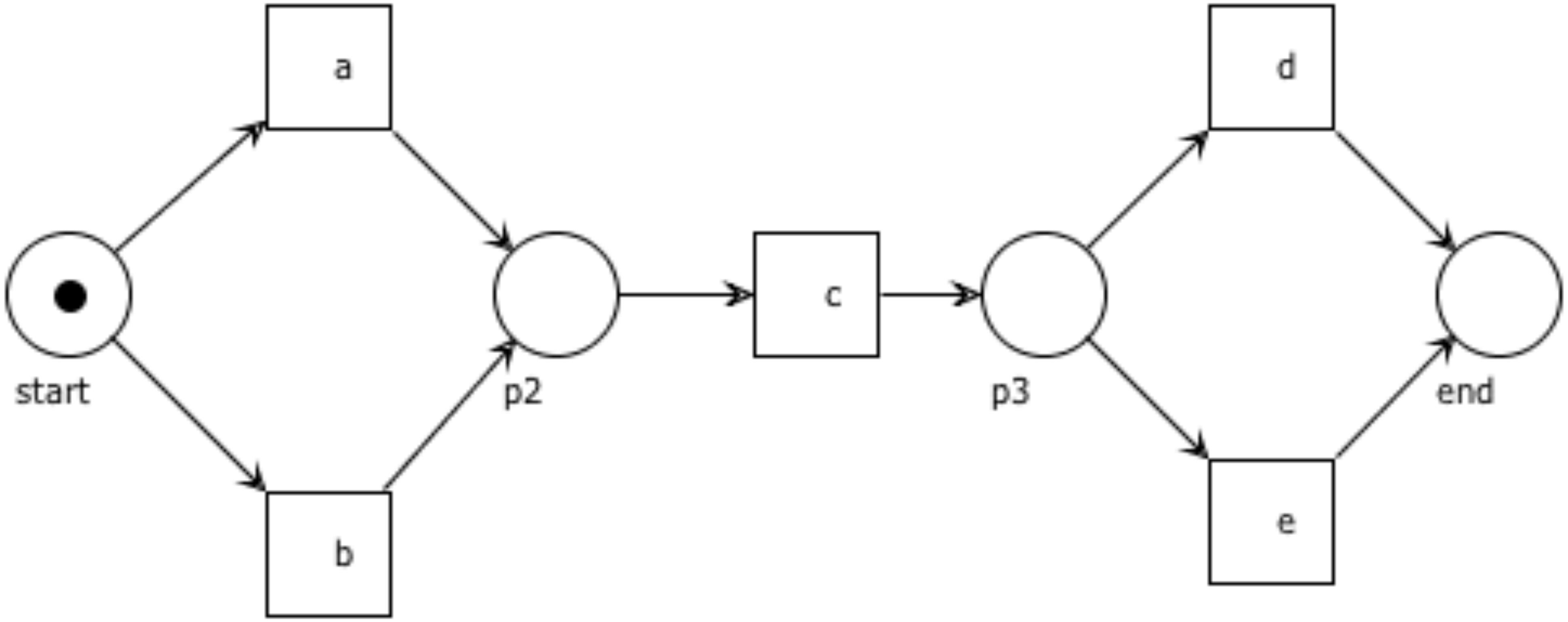
#6 traces of the form 〈 a , c , d 〉

#3 traces of the form 〈 b , c , e 〉

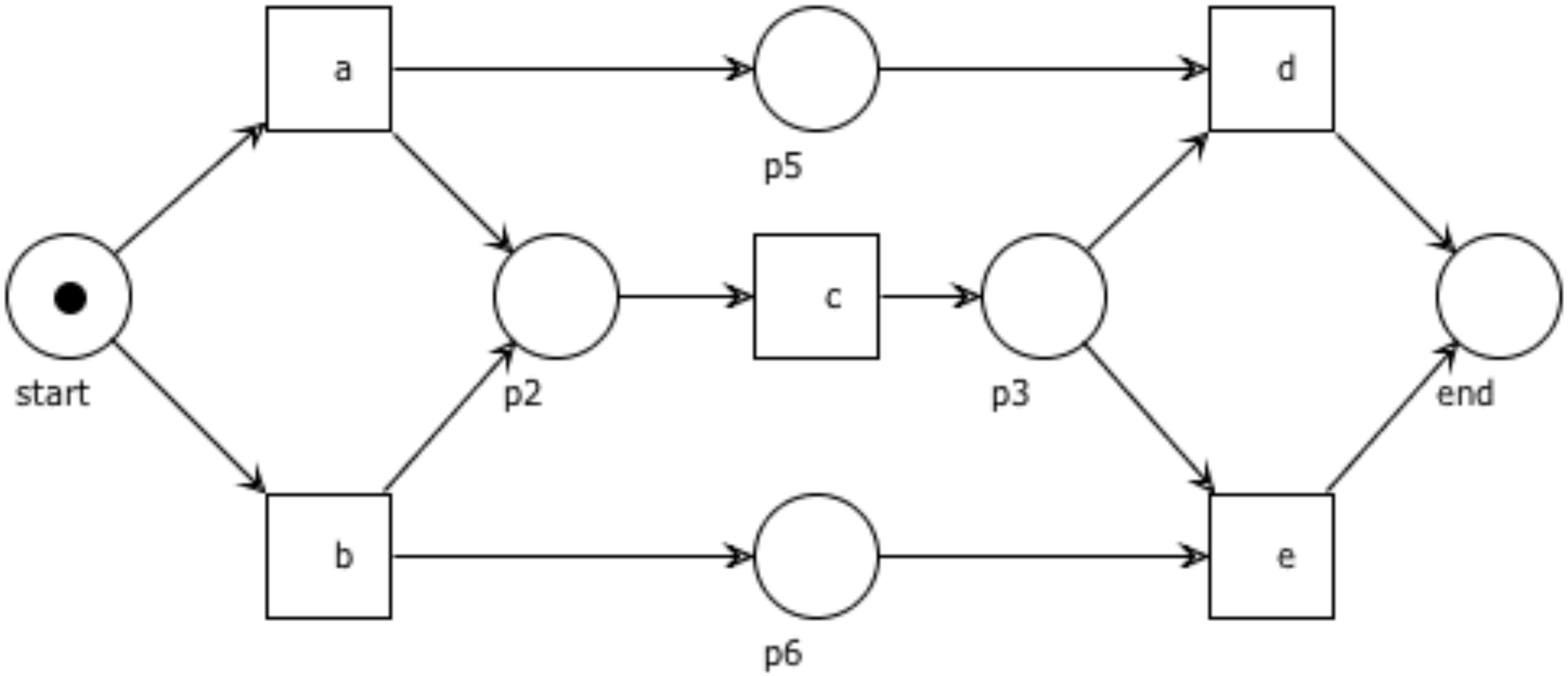
Which model (i.e., Petri net) would you infer?

The Petri net you derive must have exactly
five transitions named a, b, c, d, e
(and the places / arcs you like)

Question time



Question time



Question time

Suppose you are given a log with:

#3 traces of the form ⟨ a , b , c , d ⟩

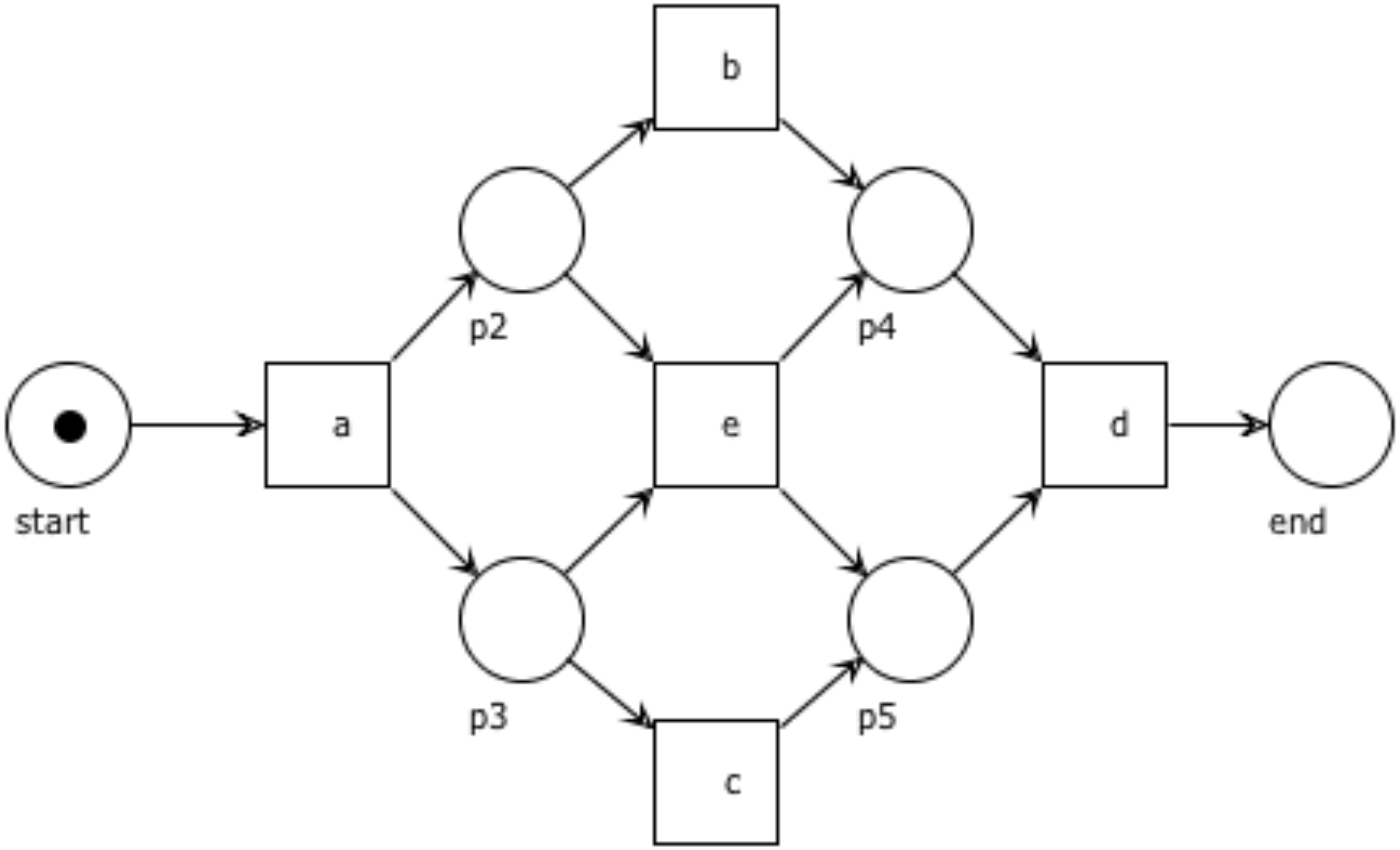
#1 traces of the form ⟨ a , e , d ⟩

#2 traces of the form ⟨ a , c , b , d ⟩

Which model (i.e., Petri net) would you infer?

The Petri net you derive must have exactly
five transitions named a, b, c, d, e
(and the places / arcs you like)

Question time



α-Algorithm

The α-algorithm was one of the first process discovery algorithms that could adequately deal with concurrency.

It has several limitations,
but it provides a good introduction into the topic:
The α-algorithm is simple and many of its ideas have been embedded in more complex and robust techniques.

The α-algorithm scans the event log for particular patterns,
called **log-based ordering relations,**
to create a **footprint** of the log.

Log-based Ordering Relations

a is sometimes followed by b

$\boxed{a >_L b}$ if and only if there is a trace $\sigma = \langle t_1, t_2, t_3, \ldots, t_n \rangle$ and $i \in \{1, \ldots, n-1\}$ such that $\sigma \in L$ and $t_i = a$ and $t_{i+1} = b$

- $\boxed{a \rightarrow_L b}$ if and only if $a >_L b$ and $b \not>_L a$ (causality)
- $\boxed{a \#_L b}$ if and only if $a \not>_L b$ and $b \not>_L a$ (mutual exclusion)
- $\boxed{a \parallel_L b}$ if and only if $a >_L b$ and $b >_L a$ (concurrency)

$$x \rightarrow_L y, \; y \rightarrow_L x, \; x \#_L y, \text{ or } x \parallel_L y$$

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \parallel_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle \boxed{a, b}, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$

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

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

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

$$\parallel_{L_1} = \{(b,c), (c,b)\}$$

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \parallel_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, \boxed{b, c}, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$

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

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

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

$$\parallel_{L_1} = \{(b,c), (c,b)\}$$

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \|_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, \boxed{c, d} \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$$

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

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

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

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

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \parallel_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d\rangle^3, \langle \boxed{a, c}, b, d\rangle^2, \langle a, e, d\rangle]$$

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

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

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

$$\parallel_{L_1} = \{(b,c), (c,b)\}$$

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \parallel_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d\rangle^3, \langle a, \boxed{c, b}, d\rangle^2, \langle a, e, d\rangle]$$

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

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

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

$$\parallel_{L_1} = \{(b,c), (c,b)\}$$

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \parallel_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d\rangle^3, \langle a, c, \boxed{b, d}\rangle^2, \langle a, e, d\rangle]$$

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

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

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

$$\parallel_{L_1} = \{(b, c), (c, b)\}$$

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \|_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle \boxed{a, e}, d\rangle]$$

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

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

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

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

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \parallel_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, \boxed{e, d}\rangle]$$

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

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

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

$$\parallel_{L_1} = \{(b, c), (c, b)\}$$

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \|_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$$

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

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

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

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

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \|_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$

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

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

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

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

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \|_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$$

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

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

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

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

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \parallel_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$$

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

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

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

$$\parallel_{L_1} = \{\boxed{(b,c), (c,b)}\}$$

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \parallel_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$

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

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

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

$$\parallel_{L_1} = \left\{\boxed{(b,c), (c,b)}\right\}$$

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \|_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$$

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

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

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

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

Log-based Ordering Relations: Example

- $a \rightarrow_L b$ if and only if $a >_L b$ and $b \not>_L a$
- $a \#_L b$ if and only if $a \not>_L b$ and $b \not>_L a$
- $a \parallel_L b$ if and only if $a >_L b$ and $b >_L a$

$$L_1 = [\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$$

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

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

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

$$\parallel_{L_1} = \{\boxed{(b,c), (c,b)}\}$$

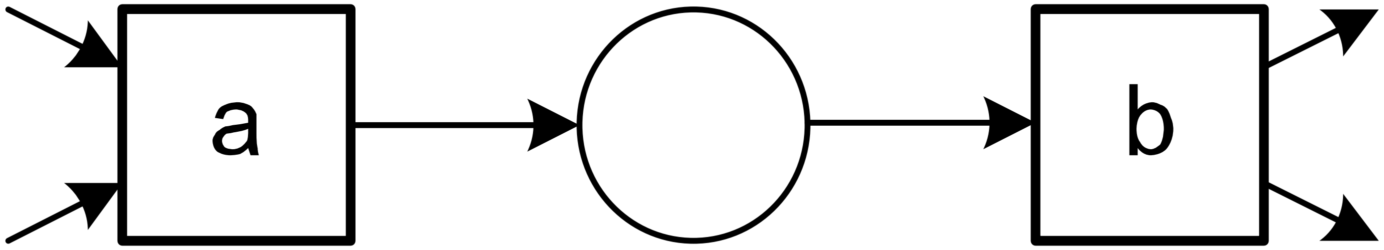
Footprint Matrix: Example

$$L_1 = [\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$$

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

Patterns

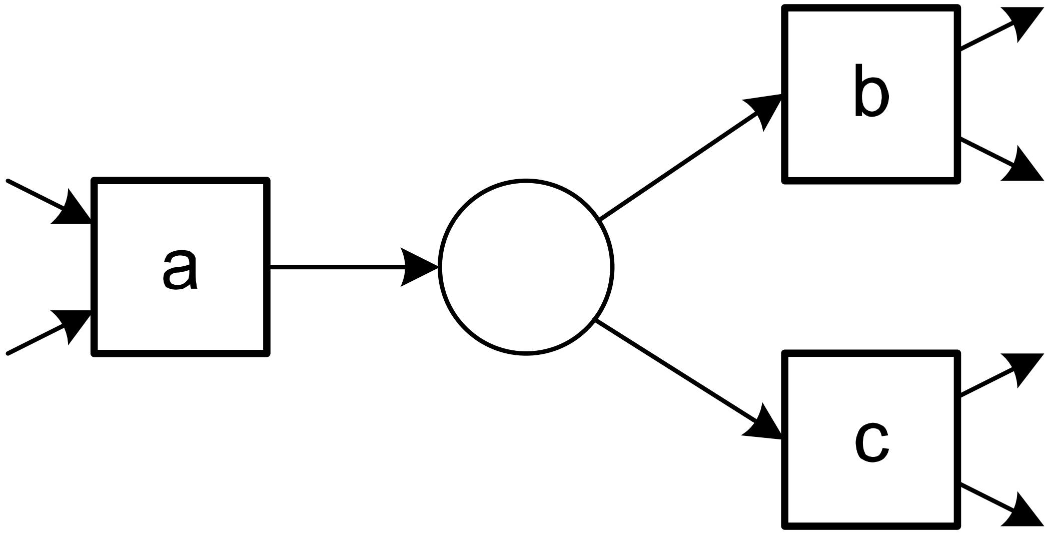
Footprints are useful to discover typical patterns of activities in the corresponding process model



(a) sequence pattern: a→b

Patterns

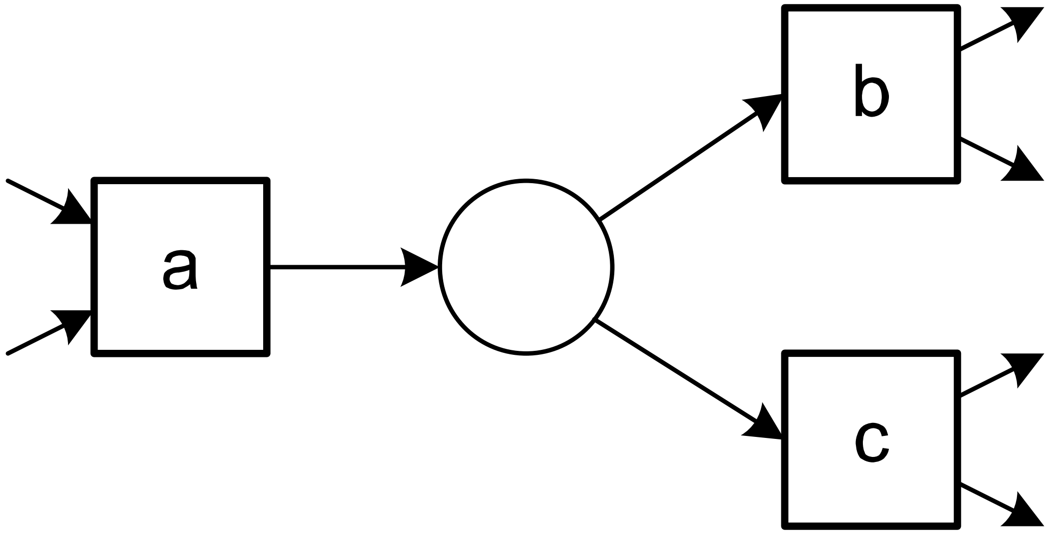
Footprints are useful to discover typical patterns of activities in the corresponding process model



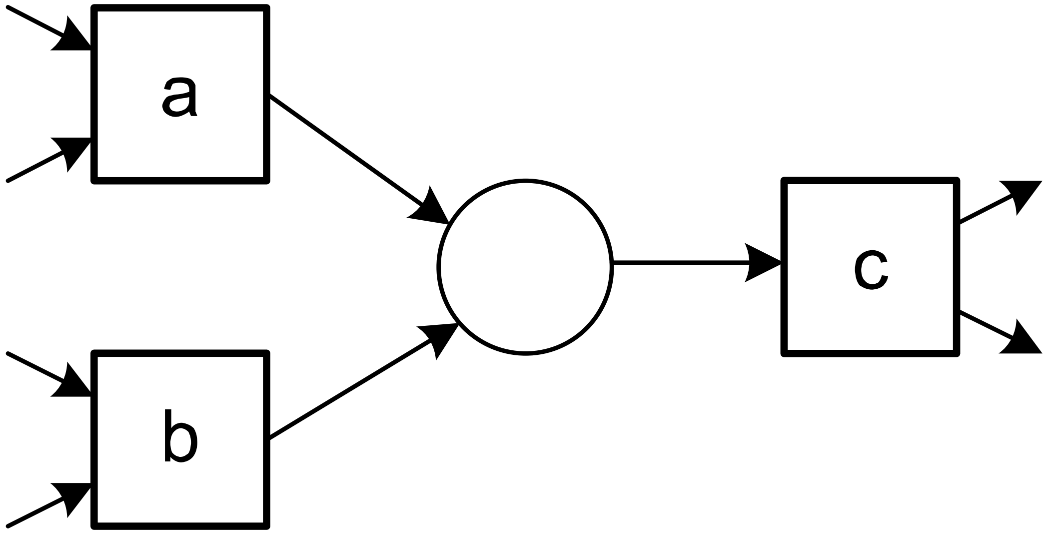
(b) XOR-split pattern:
a→b, a→c, and b#c

Patterns

Footprints are useful to discover typical patterns of activities in the corresponding process model



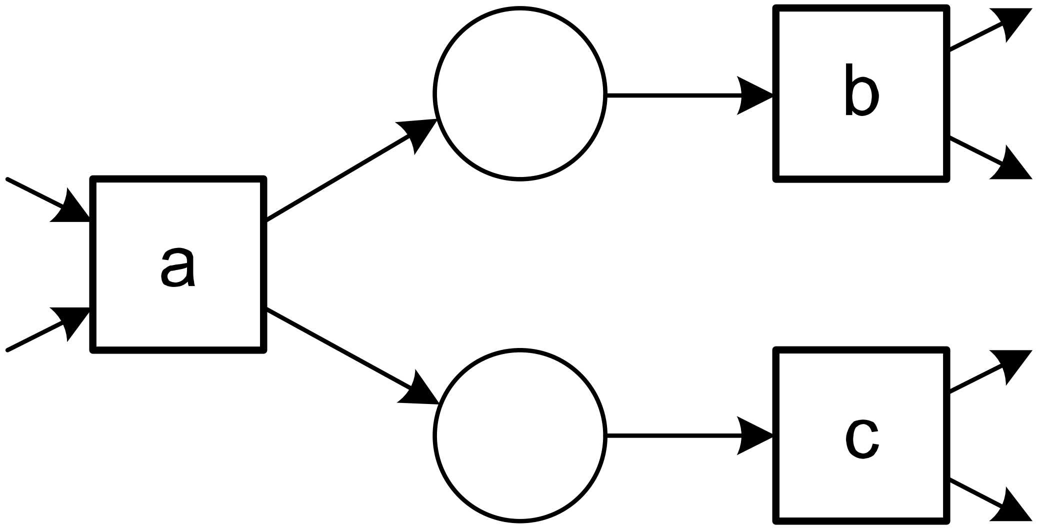
(b) XOR-split pattern:
a→b, a→c, and b#c



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

Patterns

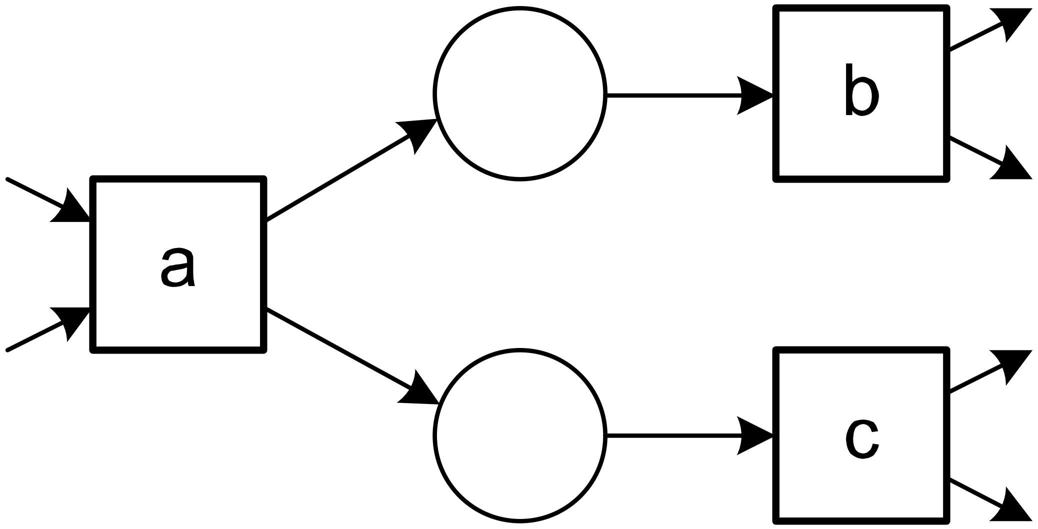
Footprints are useful to discover typical patterns of activities in the corresponding process model



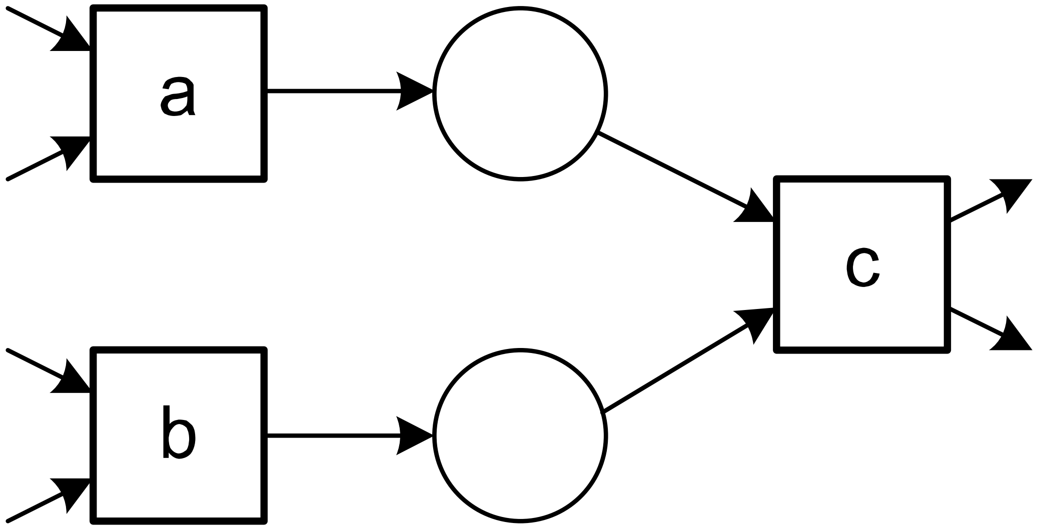
(d) AND-split pattern:
a→b, a→c, and b||c

Patterns

Footprints are useful to discover typical patterns of activities in the corresponding process model



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



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

The alpha-Algorithm

1. $T_L = \{\, t \in T \mid \exists_{\sigma \in L}\ t \in \sigma \,\}$ transitions

2. $T_I = \{\, t \in T \mid \exists_{\sigma \in L}\ t = \mathit{first}(\sigma) \,\}$ start events

3. $T_O = \{\, t \in T \mid \exists_{\sigma \in L}\ t = \mathit{last}(\sigma) \,\}$ end events

4. $X_L = \left\{ (A,B) \;\middle|\; \begin{array}{lll} A,B \subseteq T_L & \wedge\ \ A,B \neq \emptyset & \wedge \\ \forall_{a \in A} \forall_{b \in B} & a \rightarrow_L b & \wedge \\ \forall_{a_1,a_2 \in A} & a_1 \#_L a_2 & \wedge \\ \forall_{b_1,b_2 \in B} & b_1 \#_L b_2 & \end{array} \right\}$ decision points

5. $Y_L = \left\{ (A,B) \in X_L \;\middle|\; \forall_{(A',B') \in X_L} \begin{array}{c} A \subseteq A' \ \wedge\ B \subseteq B' \\ \Rightarrow \\ (A',B') = (A,B) \end{array} \right\}$ max. dec. points

6. $P_L = \{\, p_{(A,B)} \mid (A,B) \in Y_L \,\} \cup \{\, i_L, o_L \,\}$ places

7. $F_L = \{\, (a, p_{(A,B)}) \mid (A,B) \in Y_L \ \wedge\ a \in A \,\} \cup$
   $\{\, (p_{(A,B)}, b) \mid (A,B) \in Y_L \ \wedge\ b \in B \,\} \cup$
   $\{\, (i_L, t) \mid t \in T_I \,\} \cup$
   $\{\, (t, o_L) \mid t \in T_O \,\}$ arcs

8. $\alpha(L) = (P_L, T_L, F_L, i_L)$ net

The alpha-Algorithm

1. $T_L = \{\ t \in T \ \mid\ \exists_{\sigma \in L}\, t \in \sigma\ \}$ transitions
2. $T_I = \{\ t \in T \ \mid\ \exists_{\sigma \in L}\, t = \mathit{first}(\sigma)\ \}$ start events
3. $T_O = \{\ t \in T \ \mid\ \exists_{\sigma \in L}\, t = \mathit{last}(\sigma)\ \}$ end events

The alpha-Algorithm

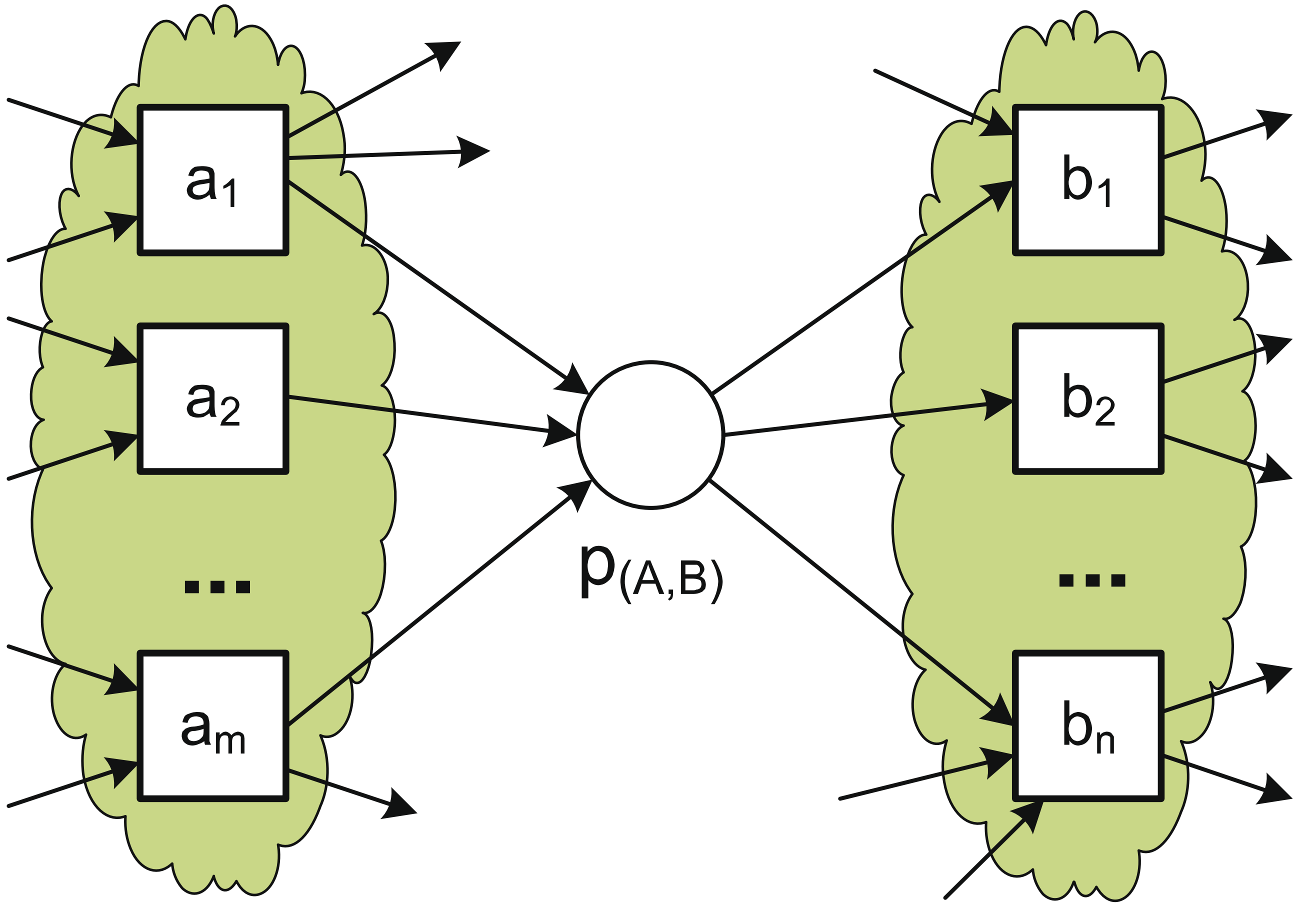
4. $$X_L = \left\{ (A,B) \;\middle|\; \begin{array}{lll} A,B \subseteq T_L & \wedge \quad A,B \neq \emptyset & \wedge \\ \forall_{a \in A} \forall_{b \in B} & a \rightarrow_L b & \wedge \\ \forall_{a_1,a_2 \in A} & a_1 \#_L a_2 & \wedge \\ \forall_{b_1,b_2 \in B} & b_1 \#_L b_2 & \end{array} \right\}$$ decision points

The Core of the algorithm: Steps 4, 5

$$\begin{array}{ll} \forall_{a \in A} \forall_{b \in B} & a \rightarrow_L b \\ \forall_{a_1, a_2 \in A} & a_1 \#_L a_2 \\ \forall_{b_1, b_2 \in B} & b_1 \#_L b_2 \end{array}$$

	a_1	a_2	$\ldots$	a_m	b_1	b_2	$\ldots$	b_n
a_1	#	#	$\ldots$	#	→	→	$\ldots$	→
a_2	#	#	$\ldots$	#	→	→	$\ldots$	→
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
a_m	#	#	$\ldots$	#	→	→	$\ldots$	→
b_1	←	←	$\ldots$	←	#	#	$\ldots$	#
b_2	←	←	$\ldots$	←	#	#	$\ldots$	#
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
b_n	←	←	$\ldots$	←	#	#	$\ldots$	#

The Core of the Algorithm: Step 4, 5



$A=\{a_1,a_2, \ldots a_m\}$

$B=\{b_1,b_2, \ldots b_n\}$

The alpha-Algorithm

5. $Y_L = \left\{ (A,B) \in X_L \;\mid \forall_{(A',B') \in X_L} \begin{array}{c} A \subseteq A' \;\wedge\; B \subseteq B' \\ \Rightarrow \\ (A',B') = (A,B) \end{array} \right\}$ max. dec. points

The Algorithm: Example

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

$$X_{L_1} = \Big\{\boxed{\big(\{a\},\{b\}\big)}, \big(\{a\},\{c\}\big), \boxed{\big(\{a\},\{e\}\big)}, \boxed{\big(\{a\},\{b,e\}\big)}, \big(\{a\},\{c,e\}\big), \big(\{b\},\{d\}\big), \big(\{c\},\{d\}\big), \big(\{e\},\{d\}\big), \big(\{b,e\},\{d\}\big), \big(\{c,e\},\{d\}\big)\Big\}$$

$$Y_{L_1} = \Big\{\boxed{\big(\{a\},\{b,e\}\big)}, \big(\{a\},\{c,e\}\big), \big(\{b,e\},\{d\}\big), \big(\{c,e\},\{d\}\big)\Big\}$$

The Algorithm: Example

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

$$X_{L_1} = \Big\{ \big(\{a\},\{b\}\big), \boxed{\big(\{a\},\{c\}\big)}, \boxed{\big(\{a\},\{e\}\big)}, \big(\{a\},\{b,e\}\big), \boxed{\big(\{a\},\{c,e\}\big)},$$
$$\big(\{b\},\{d\}\big), \big(\{c\},\{d\}\big), \big(\{e\},\{d\}\big), \big(\{b,e\},\{d\}\big), \big(\{c,e\},\{d\}\big) \Big\}$$

$$Y_{L_1} = \Big\{ \big(\{a\},\{b,e\}\big), \boxed{\big(\{a\},\{c,e\}\big)}, \big(\{b,e\},\{d\}\big), \big(\{c,e\},\{d\}\big) \Big\}$$

The Algorithm: Example

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

$$
\begin{aligned}
X_{L_1} = \Big\{ & \big(\{a\},\{b\}\big), \big(\{a\},\{c\}\big), \big(\{a\},\{e\}\big), \big(\{a\},\{b,e\}\big), \big(\{a\},\{c,e\}\big), \\
& \boxed{\big(\{b\},\{d\}\big)}, \big(\{c\},\{d\}\big), \boxed{\big(\{e\},\{d\}\big)}, \boxed{\big(\{b,e\},\{d\}\big)}, \big(\{c,e\},\{d\}\big) \Big\}
\end{aligned}
$$

$$
Y_{L_1} = \Big\{ \big(\{a\},\{b,e\}\big), \big(\{a\},\{c,e\}\big), \boxed{\big(\{b,e\},\{d\}\big)}, \big(\{c,e\},\{d\}\big) \Big\}
$$

The Algorithm: Example

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

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

$$(\{b\},\{d\}), \boxed{(\{c\},\{d\})}, \boxed{(\{e\},\{d\})}, (\{b,e\},\{d\}), \boxed{(\{c,e\},\{d\})}\}$$

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

The alpha-Algorithm

6. $P_L = \{\ p_{(A,B)}\ \mid\ (A,B) \in Y_L\ \} \cup \{\ i_L, o_L\ \}$ places

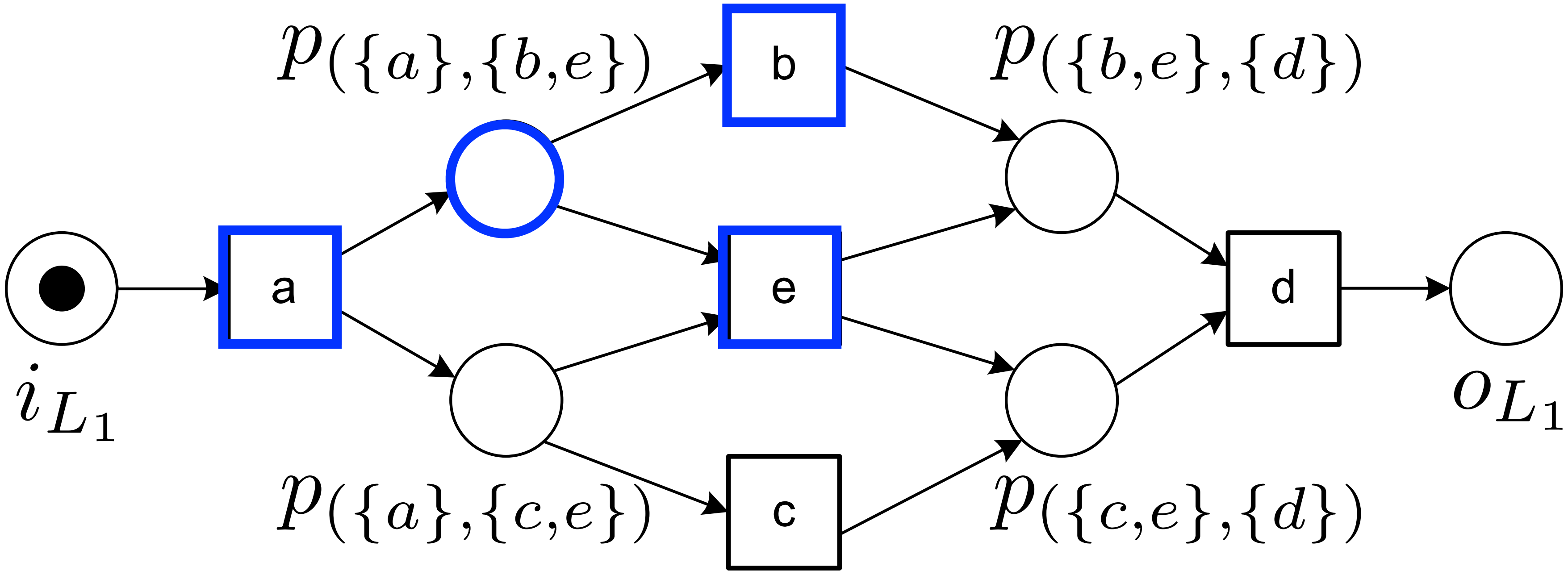
7. $F_L = \{\ (a, p_{(A,B)})\ \mid\ (A,B) \in Y_L\ \wedge\ a \in A\ \} \cup$
$\{\ (p_{(A,B)}, b)\ \mid\ (A,B) \in Y_L\ \wedge\ b \in B\ \} \cup$
$\{\ (i_L, t)\ \mid\ t \in T_I\ \} \cup$
$\{\ (t, o_L)\ \mid\ t \in T_O\ \}$ arcs

8. $\alpha(L) = (P_L, T_L, F_L, i_L)$ net

The Algorithm: Example

$$L_1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$

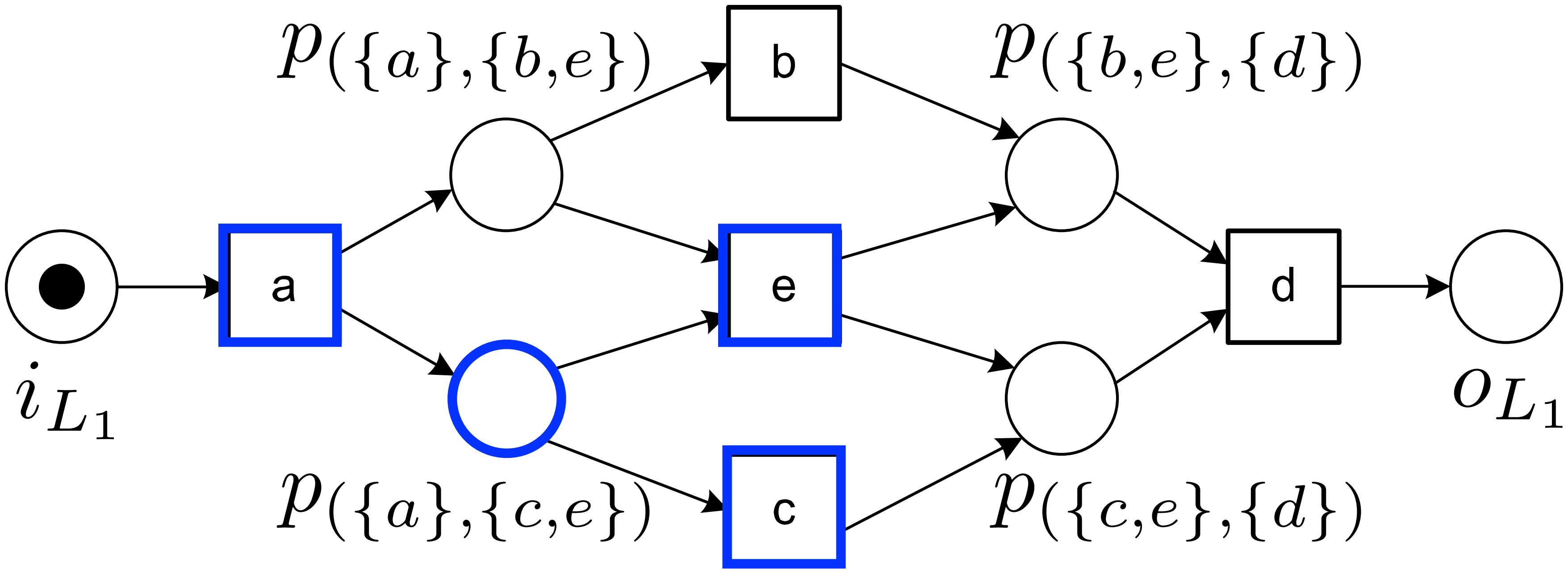
$$Y_{L_1} = \{\boxed{(\{a\}, \{b, e\})}, (\{a\}, \{c, e\}), (\{b, e\}, \{d\}), (\{c, e\}, \{d\})\}$$



The Algorithm: Example

$$L_1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$

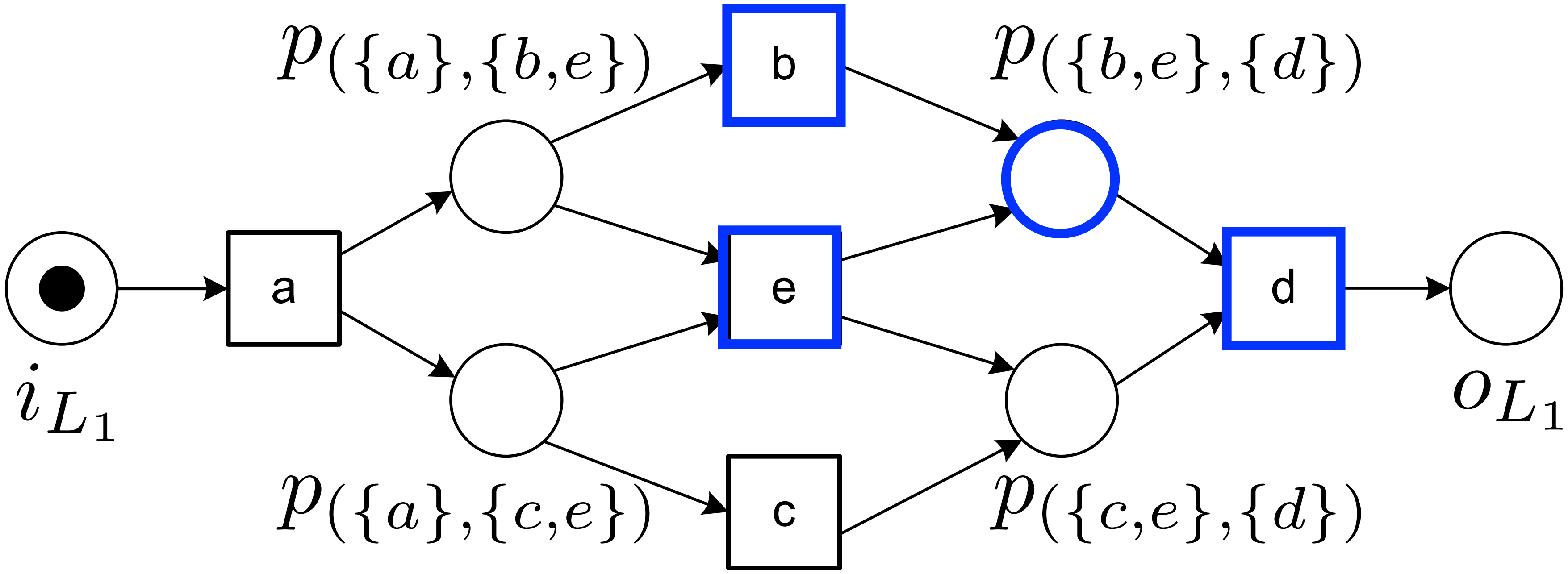
$$Y_{L_1} = \{(\{a\},\{b,e\}), \boxed{(\{a\},\{c,e\})}, (\{b,e\},\{d\}), (\{c,e\},\{d\})\}$$



The Algorithm: Example

$$L_1 = [\langle a,b,c,d\rangle^3, \langle a,c,b,d\rangle^2, \langle a,e,d\rangle]$$

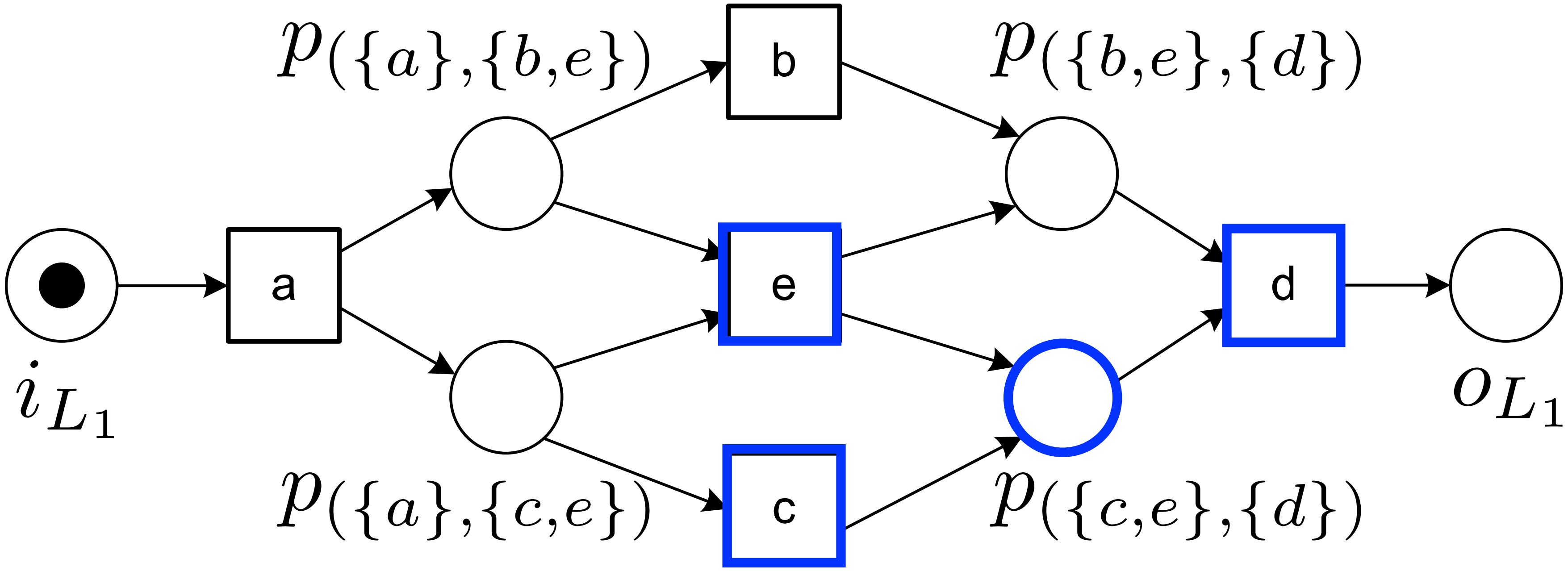
$$Y_{L_1} = \{(\{a\},\{b,e\}), (\{a\},\{c,e\}), \boxed{(\{b,e\},\{d\})}, (\{c,e\},\{d\})\}$$



The Algorithm: Example

$$L_1 = [\langle a, b, c, d\rangle^3, \langle a, c, b, d\rangle^2, \langle a, e, d\rangle]$$

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



Another Example

$$L_5 = \big[\langle a, b, e, f\rangle^2, \langle a, b, e, c, d, b, f\rangle^3, \langle a, b, c, e, d, b, f\rangle^2,$$
$$\langle a, b, c, d, e, b, f\rangle^4, \langle a, e, b, c, d, b, f\rangle^3\big]$$

	a	b	c	d	e	f
a	$\#$	$\rightarrow$	$\#$	$\#$	$\rightarrow$	$\#$
b	$\leftarrow$	$\#$	$\rightarrow$	$\leftarrow$	$\parallel$	$\rightarrow$
c	$\#$	$\leftarrow$	$\#$	$\rightarrow$	$\parallel$	$\#$
d	$\#$	$\rightarrow$	$\leftarrow$	$\#$	$\parallel$	$\#$
e	$\leftarrow$	$\parallel$	$\parallel$	$\parallel$	$\#$	$\rightarrow$
f	$\#$	$\leftarrow$	$\#$	$\#$	$\leftarrow$	$\#$

$$T_L = \{a, b, c, d, e, f\}$$

$$T_I = \{a\}$$

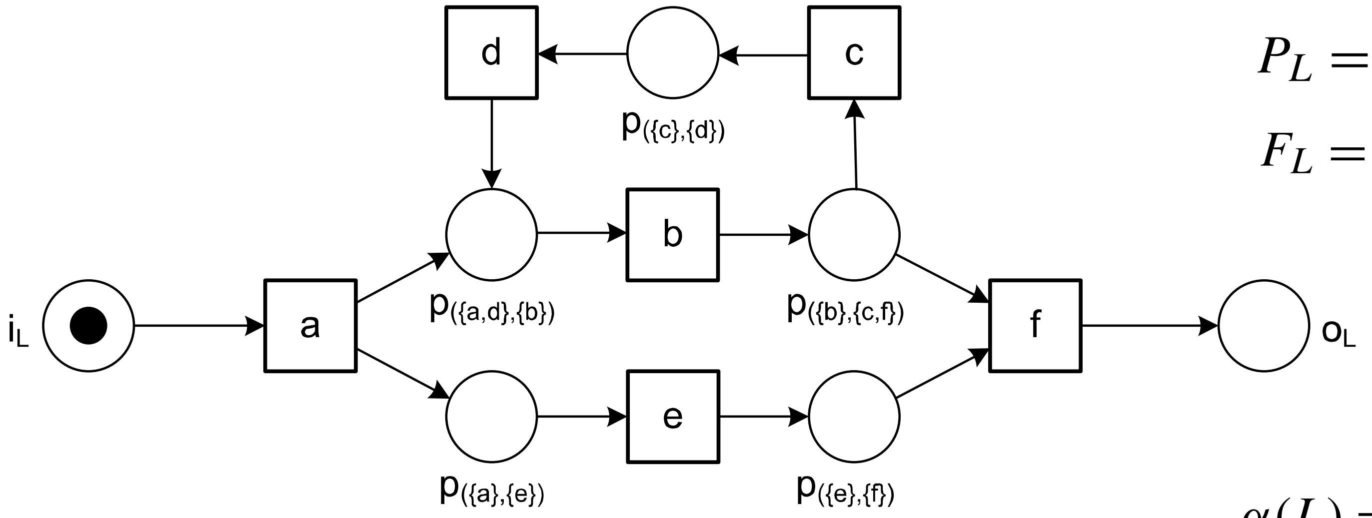
$$T_I = \{f\}$$

$$X_L = \big\{\big(\{a\},\{b\}\big), \big(\{a\},\{e\}\big), \big(\{b\},\{c\}\big), \big(\{b\},\{f\}\big), \big(\{c\},\{d\}\big),$$
$$\big(\{d\},\{b\}\big), \big(\{e\},\{f\}\big), \big(\{a,d\},\{b\}\big), \big(\{b\},\{c,f\}\big)\big\}$$

$$Y_L = \big\{\big(\{a\},\{e\}\big), \big(\{c\},\{d\}\big), \big(\{e\},\{f\}\big), \big(\{a,d\},\{b\}\big), \big(\{b\},\{c,f\}\big)\big\}$$

$$P_L = \big\{p_{(\{a\},\{e\})}, p_{(\{c\},\{d\})}, p_{(\{e\},\{f\})}, p_{(\{a,d\},\{b\})}, p_{(\{b\},\{c,f\})}, i_L, o_L\big\}$$

$$F_L = \big\{(a, p_{(\{a\},\{e\})}), (p_{(\{a\},\{e\})}, e), (c, p_{(\{c\},\{d\})}), (p_{(\{c\},\{d\})}, d),$$
$$(e, p_{(\{e\},\{f\})}), (p_{(\{e\},\{f\})}, f), (a, p_{(\{a,d\},\{b\})}), (d, p_{(\{a,d\},\{b\})}),$$
$$(p_{(\{a,d\},\{b\})}, b), (b, p_{(\{b\},\{c,f\})}), (p_{(\{b\},\{c,f\})}, c), (p_{(\{b\},\{c,f\})}, f),$$
$$(i_L, a), (f, o_L)\big\}$$



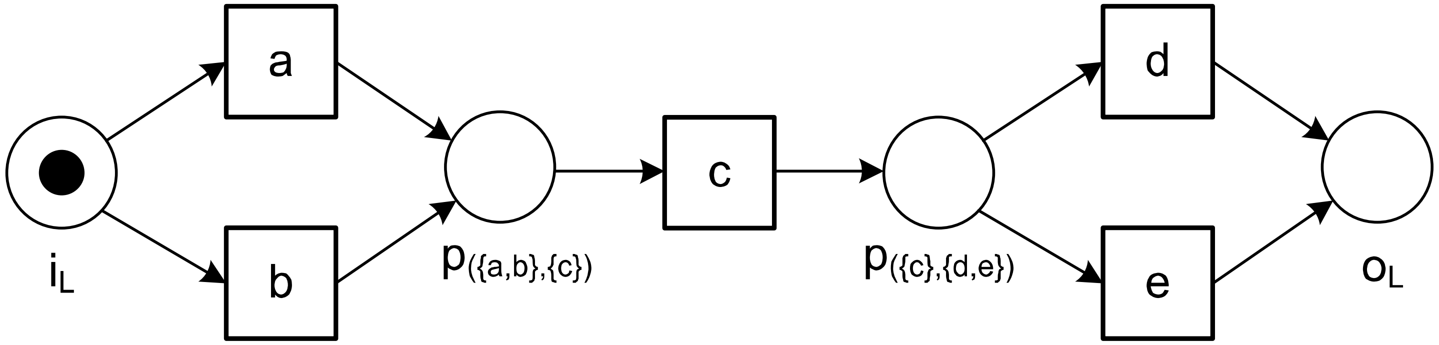
$$\alpha(L) = (P_L, T_L, F_L)$$

Exercises

$$L_4 = \left[\langle a, c, d\rangle^{45}, \langle b, c, d\rangle^{42}, \langle a, c, e\rangle^{38}, \langle b, c, e\rangle^{22}\right]$$

	a	b	c	d	e
a	$\#$	$\#$	$\rightarrow$	$\#$	$\#$
b	$\#$	$\#$	$\rightarrow$	$\#$	$\#$
c	$\leftarrow$	$\leftarrow$	$\#$	$\rightarrow$	$\rightarrow$
d	$\#$	$\#$	$\leftarrow$	$\#$	$\#$
e	$\#$	$\#$	$\leftarrow$	$\#$	$\#$

Check that the footprint matrix corresponds to the log and that the net below is the one discovered by the alpha-algorithm

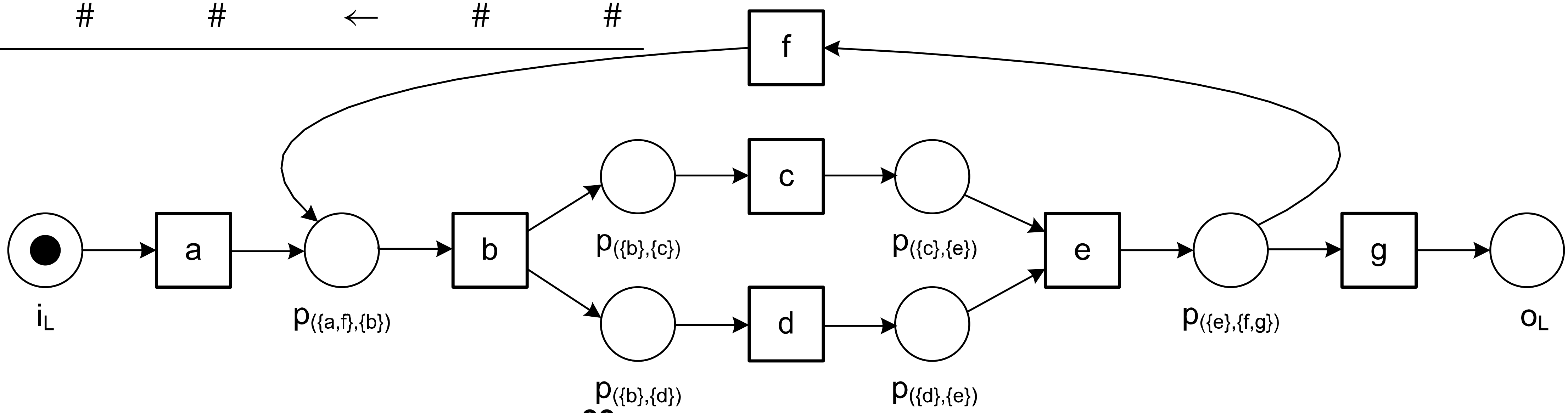


Exercises

$$L_3 = \left[\langle a, b, c, d, e, f, b, d, c, e, g\rangle, \langle a, b, d, c, e, g\rangle^2, \langle a, b, c, d, e, f, b, c, d, e, f, b, d, c, e, g\rangle\right]$$

	a	*b*	*c*	*d*	*e*	*f*	*g*
a	#	→	#	#	#	#	#
b	←	#	→	→	#	←	#
c	#	←	#	∥	→	#	#
d	#	←	∥	#	→	#	#
e	#	#	←	←	#	→	→
f	#	→	#	#	←	#	#
g	#	#	#	#	←	#	#

Check that the footprint matrix corresponds to the log and that the net below is the one discovered by the alpha-algorithm

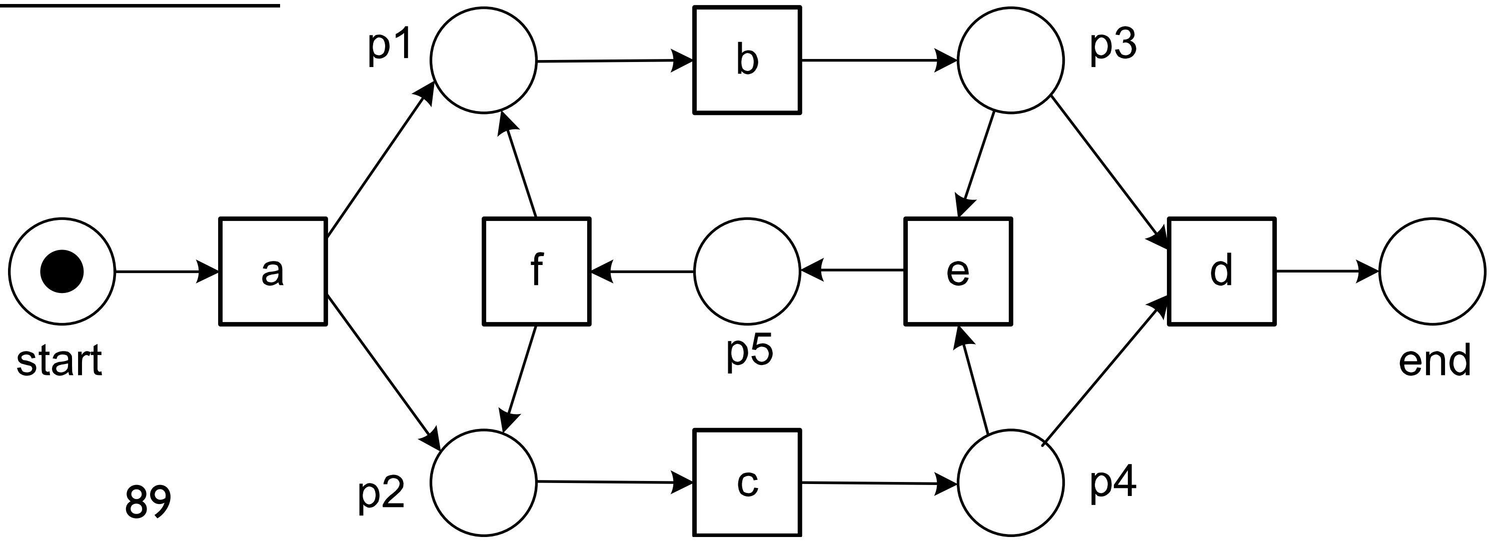


Exercises

$$L_2 = [\langle a,b,c,d\rangle^3, \langle a,c,b,d\rangle^4, \langle a,b,c,e,f,b,c,d\rangle^2, \langle a,b,c,e,f,c,b,d\rangle, \langle a,c,b,e,f,b,c,d\rangle^2, \langle a,c,b,e,f,b,c,e,f,c,b,d\rangle]$$

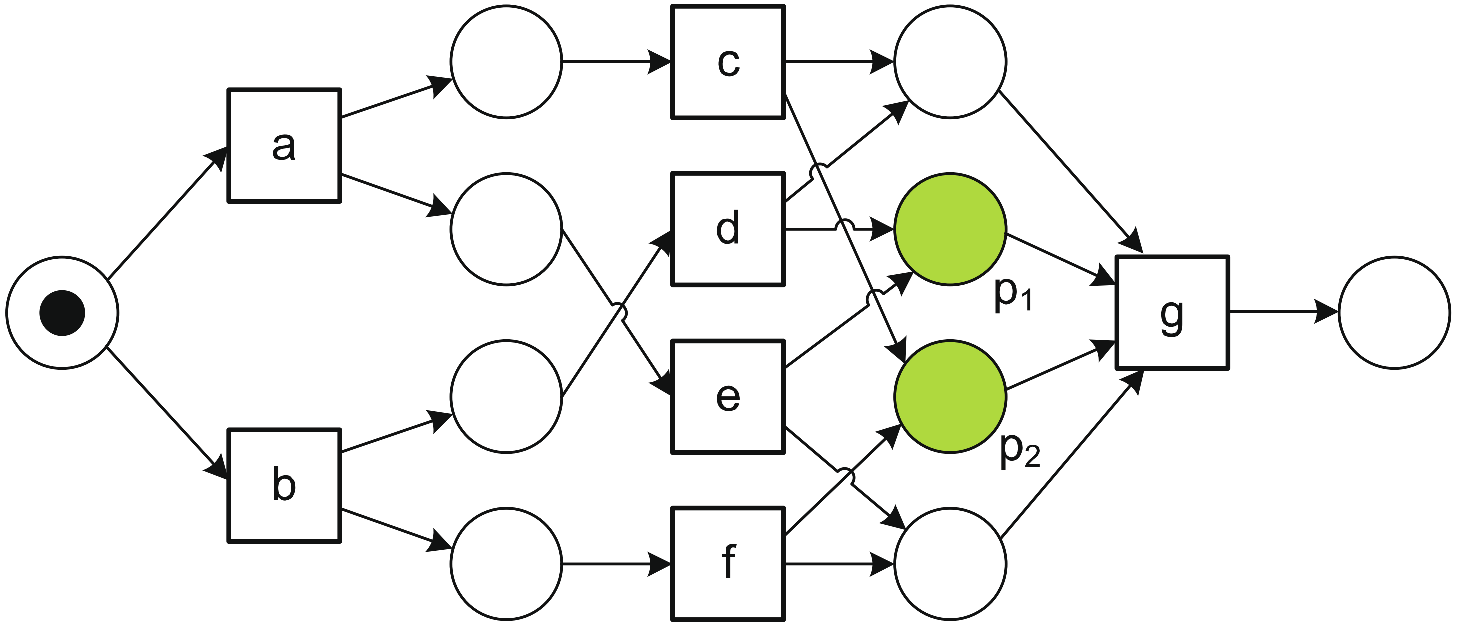
	a	b	c	d	e	f
a	#	$\rightarrow$	$\rightarrow$	#	#	#
b	$\leftarrow$	#	$\parallel$	$\rightarrow$	$\rightarrow$	$\leftarrow$
c	$\leftarrow$	$\parallel$	#	$\rightarrow$	$\rightarrow$	$\leftarrow$
d	#	$\leftarrow$	$\leftarrow$	#	#	#
e	#	$\leftarrow$	$\leftarrow$	#	#	$\rightarrow$
f	#	$\rightarrow$	$\rightarrow$	#	$\leftarrow$	#

Check that the footprint matrix corresponds to the log and that the net below is the one discovered by the alpha-algorithm



Limitation: Implicit Places

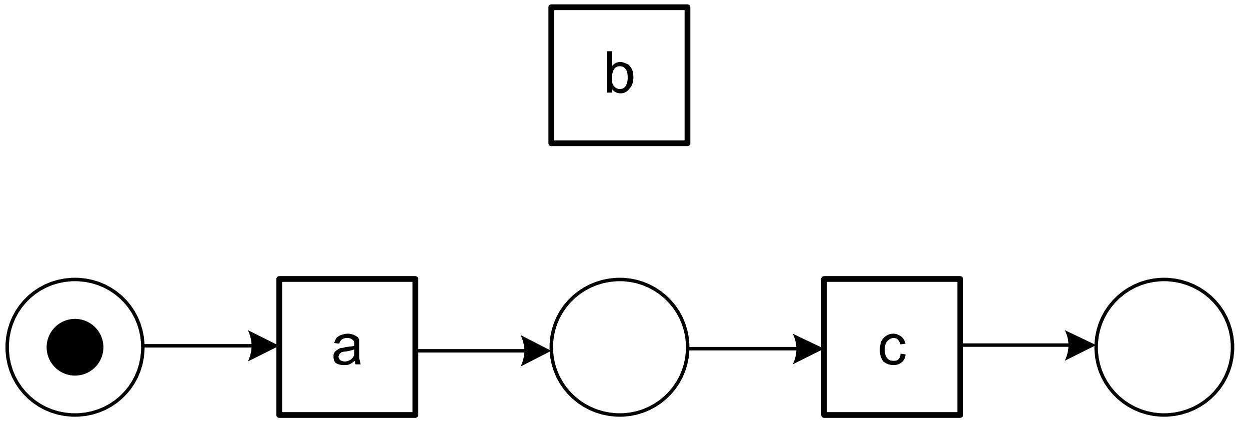
$$L_6 = [\langle a, c, e, g\rangle^2, \langle a, e, c, g\rangle^3, \langle b, d, f, g\rangle^2, \langle b, f, d, g\rangle^4]$$



p_1 and p_2 are redundant

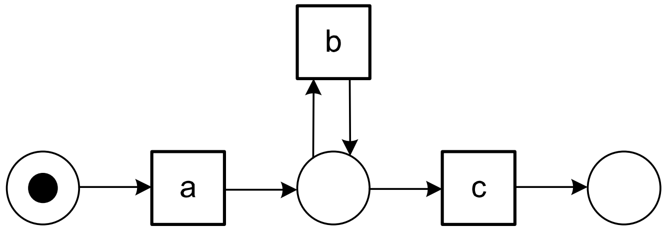
Limitation: Short Loop

$$L_7 = \left[\langle a, c\rangle^2, \langle a, b, c\rangle^3, \langle a, b, b, c\rangle^2, \langle a, b, b, b, b, c\rangle^1\right]$$



b is disconnected from the model

Expected net:



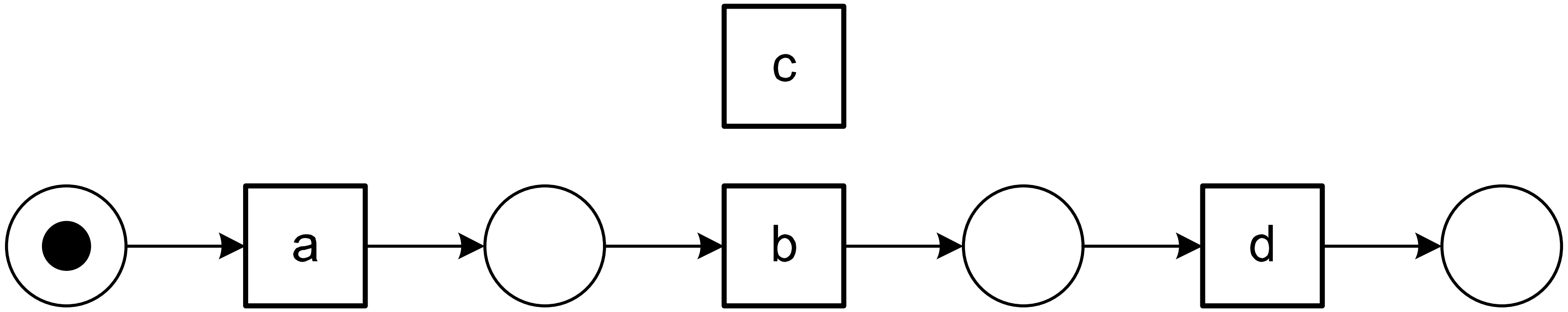
Limitation: Short Loop

$$L_8 = [\langle a,b,d\rangle^3, \langle a,b,c,b,d\rangle^2, \langle a,b,c,b,c,b,d\rangle]$$

$a \rightarrow_{L_8} b,$

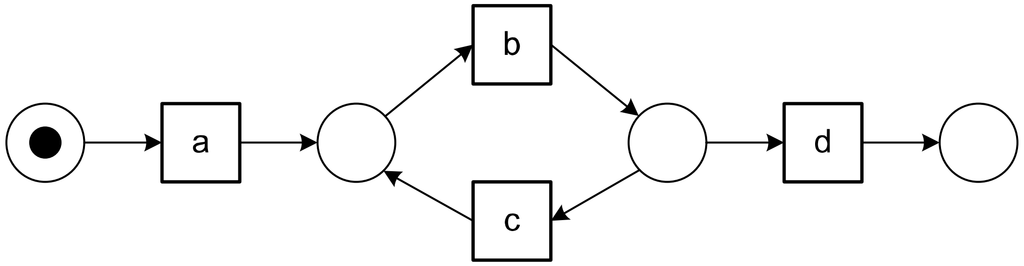
$b \rightarrow_{L_8} d,$

$b \parallel_{L_8} c.$



c is disconnected from the model

Expected net:

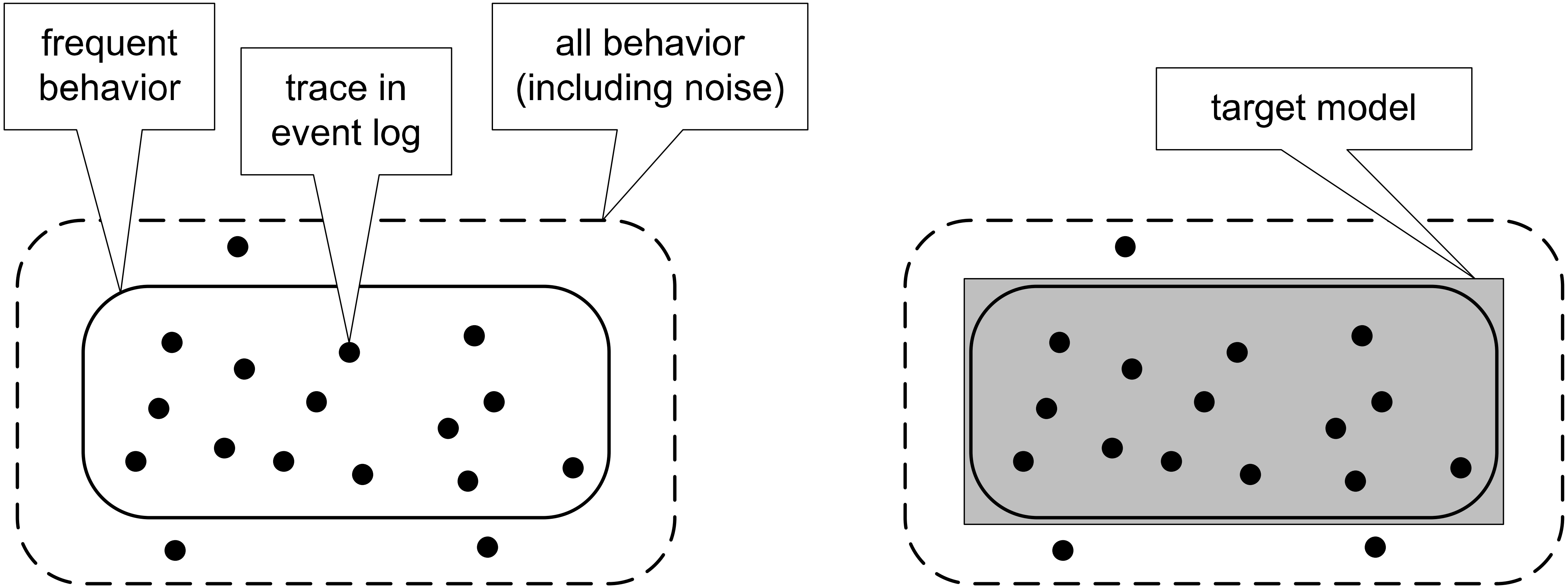


Limitation: Noise

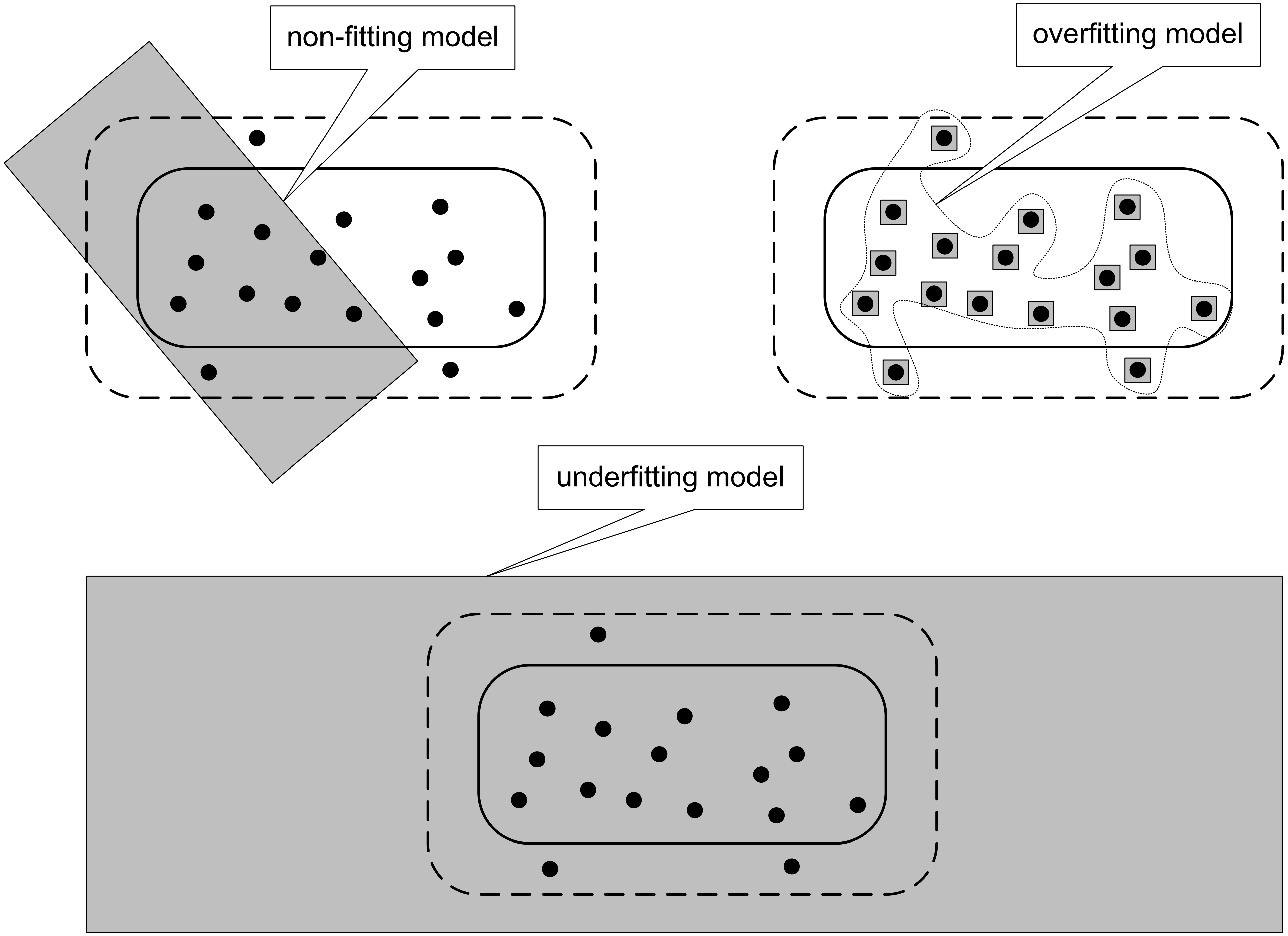
We use the term “noise” to refer to rare and infrequent behavior rather than errors related to event logging.

For example, frequencies are not taken into account by the α-algorithm (discard less frequent traces?).

Limitation: Noise



Limitation: Noise



Limitation: Incompleteness

Noise refers to the problem of having *“too many data”* (including rare behavior), **(in)completeness** refers to the problem of having *“too little data”*.

Process models typically allow for an exponential or even infinite number of different traces (in case of loops).

Moreover, some traces may have a much lower probability than others. Thus, it is unrealistic to assume that every possible trace is present in the event log.

Limitation: Incompleteness

The α-algorithm uses a **local completeness notion**:

if there are two activities a and b,
and a can be directly followed by b,
then this should be observed at least once in the log.

Conformance Checking: fitness measures

Measures and Diagnostic

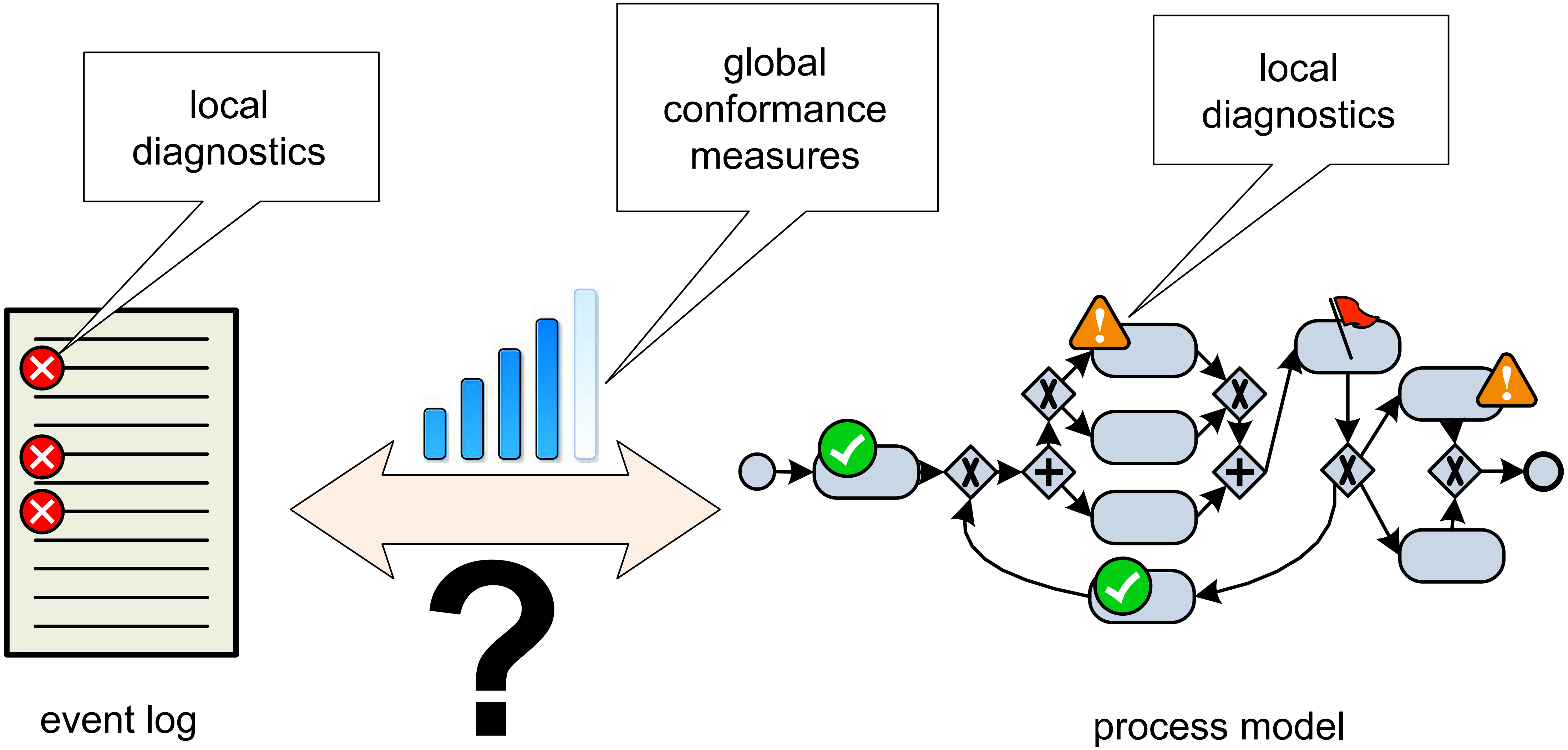


Fig. 7.1 Conformance checking: comparing observed behavior with modeled behavior. Global conformance measures quantify the overall conformance of the model and log. Local diagnostics are given by highlighting the nodes in the model where model and log disagree. Cases that do not fit are highlighted in the visualization of the log

Measuring Fitness

Fitness measures “the proportion of behavior in the event log possible according to the model”.

Of the four quality criteria,
fitness is the closest to conformance.

A naïve approach toward conformance checking would be to count the fraction of cases that can be “parsed completely” (i.e., the proportion of cases corresponding to firing sequences leading from [start] to [end]).

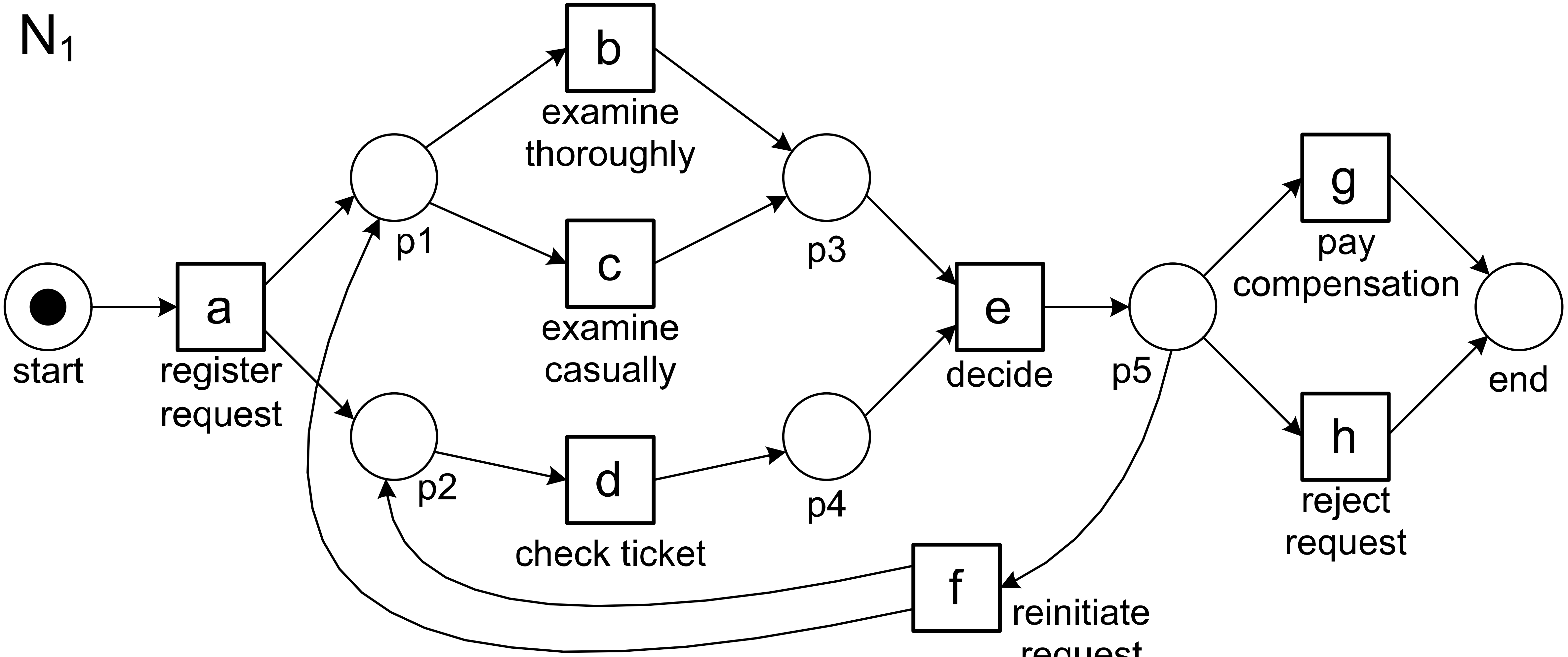
Table 7.1 Event log L_{full}: a = *register request*, b = *examine thoroughly*, c = *examine casually*, d = *check ticket*, e = *decide*, f = *reinitiate request*, g = *pay compensation*, and h = *reject request*

Example

1391 cases

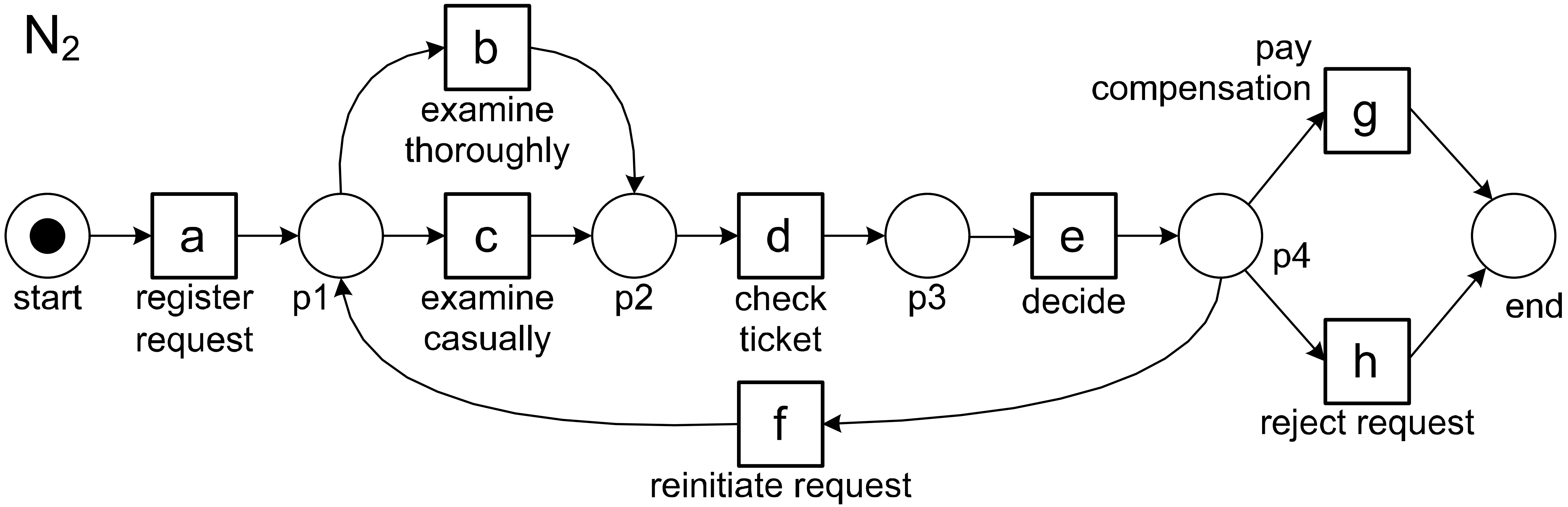
Frequency	Reference	Trace
455	σ_1	$\langle a, c, d, e, h \rangle$
191	σ_2	$\langle a, b, d, e, g \rangle$
177	σ_3	$\langle a, d, c, e, h \rangle$
144	σ_4	$\langle a, b, d, e, h \rangle$
111	σ_5	$\langle a, c, d, e, g \rangle$
82	σ_6	$\langle a, d, c, e, g \rangle$
56	σ_7	$\langle a, d, b, e, h \rangle$
47	σ_8	$\langle a, c, d, e, f, d, b, e, h \rangle$
38	σ_9	$\langle a, d, b, e, g \rangle$
33	σ_{10}	$\langle a, c, d, e, f, b, d, e, h \rangle$
14	σ_{11}	$\langle a, c, d, e, f, b, d, e, g \rangle$
11	σ_{12}	$\langle a, c, d, e, f, d, b, e, g \rangle$
9	σ_{13}	$\langle a, d, c, e, f, c, d, e, h \rangle$
8	σ_{14}	$\langle a, d, c, e, f, d, b, e, h \rangle$
5	σ_{15}	$\langle a, d, c, e, f, b, d, e, g \rangle$
3	σ_{16}	$\langle a, c, d, e, f, b, d, e, f, d, b, e, g \rangle$
2	σ_{17}	$\langle a, d, c, e, f, d, b, e, g \rangle$
2	σ_{18}	$\langle a, d, c, e, f, b, d, e, f, b, d, e, g \rangle$
1	σ_{19}	$\langle a, d, c, e, f, d, b, e, f, b, d, e, h \rangle$
1	σ_{20}	$\langle a, d, b, e, f, b, d, e, f, d, b, e, g \rangle$
1	σ_{21}	$\langle a, d, c, e, f, d, b, e, f, c, d, e, f, d, b, e, g \rangle$

Example N1



naïve fitness $\frac{1391}{1391} = 1$ The net can ``replay” any trace

Example N2

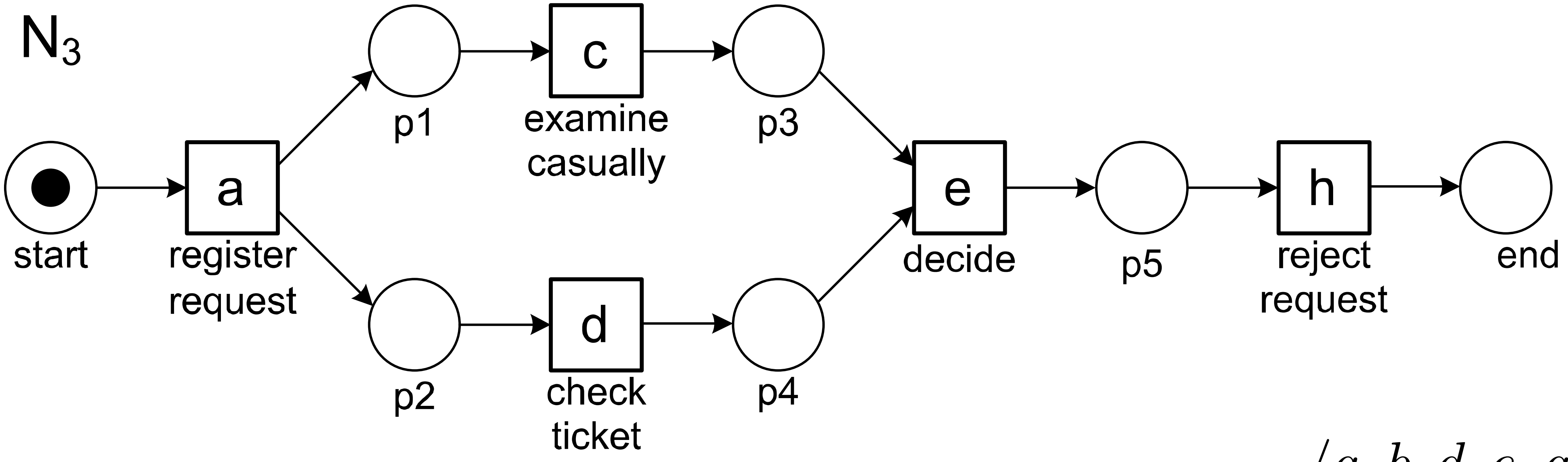


443 cases do not correspond to a firing sequence

$\langle a, d, c, e, h \rangle^{177}$
$\langle a, d, c, e, g \rangle^{82}$
$\langle a, d, b, e, h \rangle^{56}$
$\cdots$

naïve fitness $\frac{948}{1391} = 0.6815$

Example N3

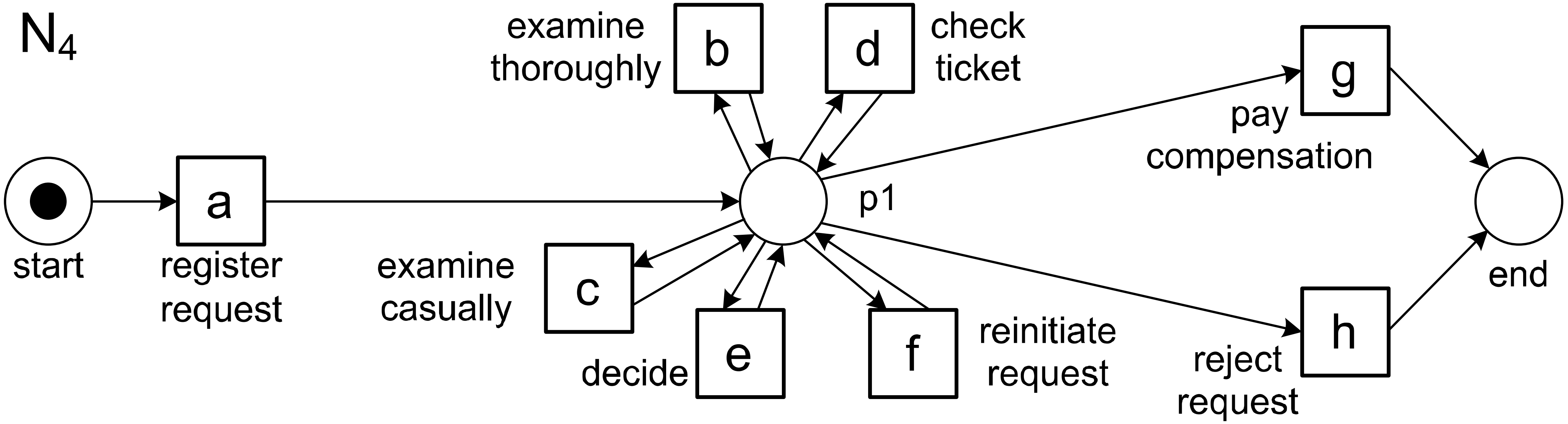


759 cases do not correspond to a firing sequence

$\langle a, b, d, e, g\rangle^{191}$
$\langle a, b, d, e, h\rangle^{144}$
$\langle a, c, d, e, g\rangle^{111}$
$\ldots$

naïve fitness $\frac{632}{1391} = 0.4543$

Example N4



"flower model" (poorly structured)

naïve fitness $\frac{1391}{1391} = 1$ The net can ``replay" any trace

Almost Fitting Traces

This naïve fitness notion seems to be too strict as traces can differ only slightly and not be counted at all.

$$\sigma = \langle a_1, a_2, \ldots, a_{100} \rangle$$

Now consider a model M1 that cannot replay σ ,
but that can replay 99 of the 100 events in σ.
Then, consider another model M2 that can only replay
only 10 of the 100 events in σ.

Using the naïve fitness metric, the trace would simply be classified as nonfitting for both models without acknowledging that σ was almost fitting
in M1 and in complete disagreement with M2.

Missing and Remaining Tokens

We next introduce a more accurate fitness notion.

When computing the naïve fitness,
we stop replaying a trace as soon as we find a problem
(and tag that trace as nonfitting).

Let us instead just continue replaying the trace on the model
but record all situations where a transition is
forced to fire without being enabled,
i.e., we count all **missing** tokens.
Moreover, we record the tokens that **remain** at the end.

Four Counters

p (produced tokens)

r (remaining tokens)

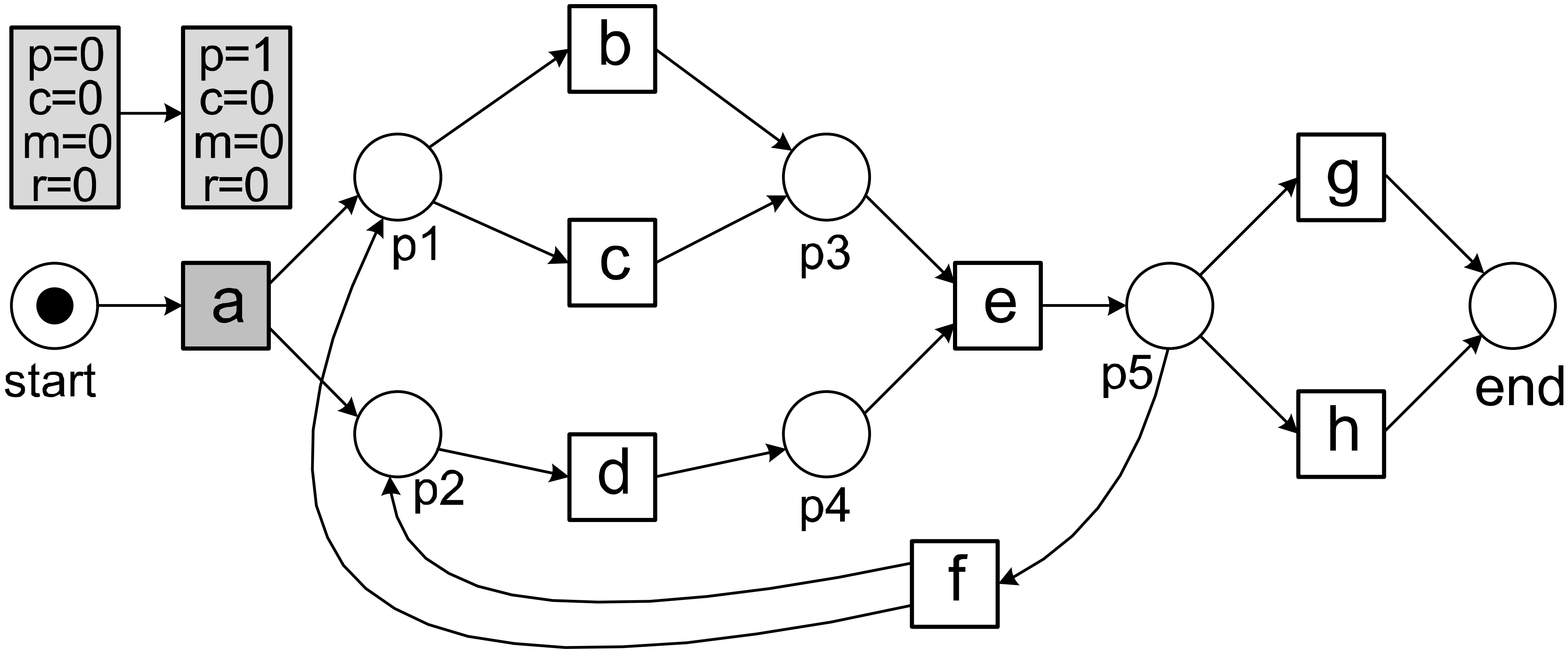
c (consumed tokens)

m (missing tokens)

$$fitness(\sigma, N) = \frac{1}{2}\left(1 - \frac{m}{c}\right) + \frac{1}{2}\left(1 - \frac{r}{p}\right)$$

Example

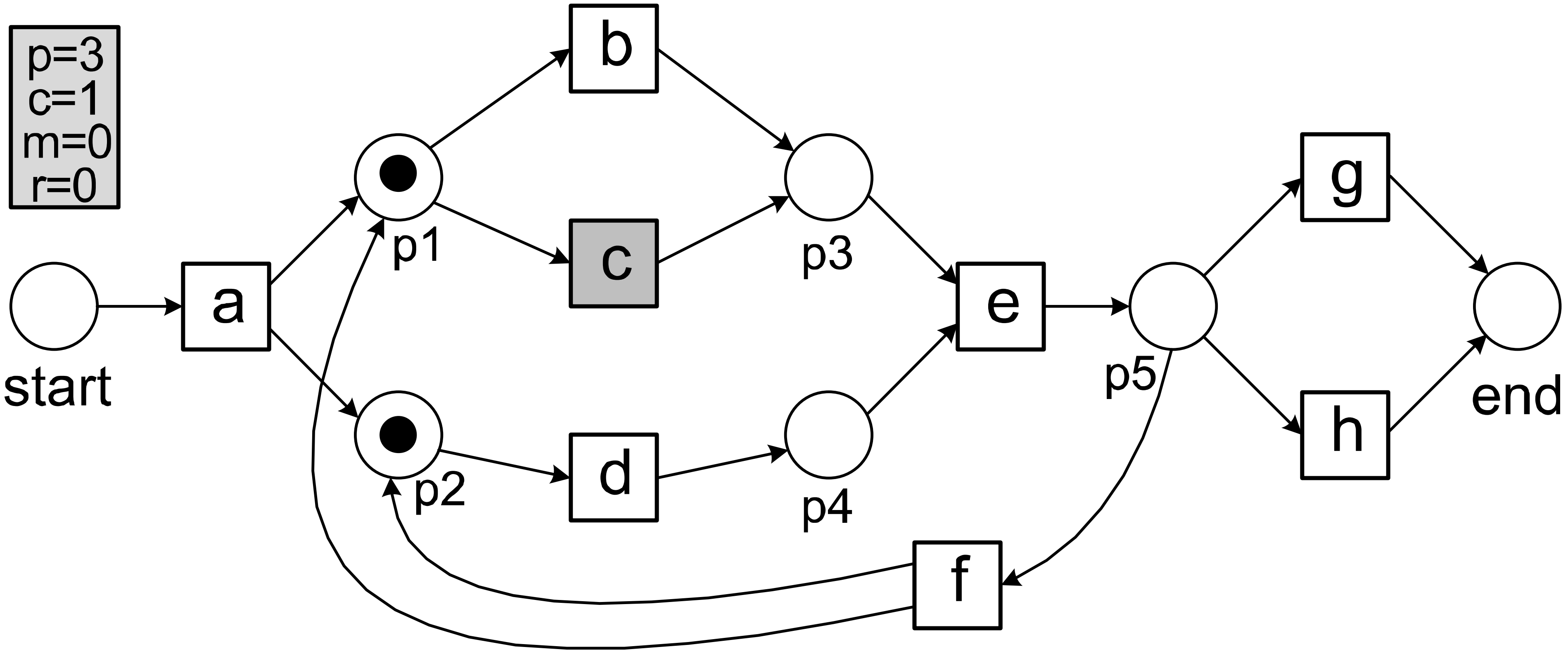
the environment produces a token for place start



$$\sigma_1 = \langle a, c, d, e, h \rangle$$

Example

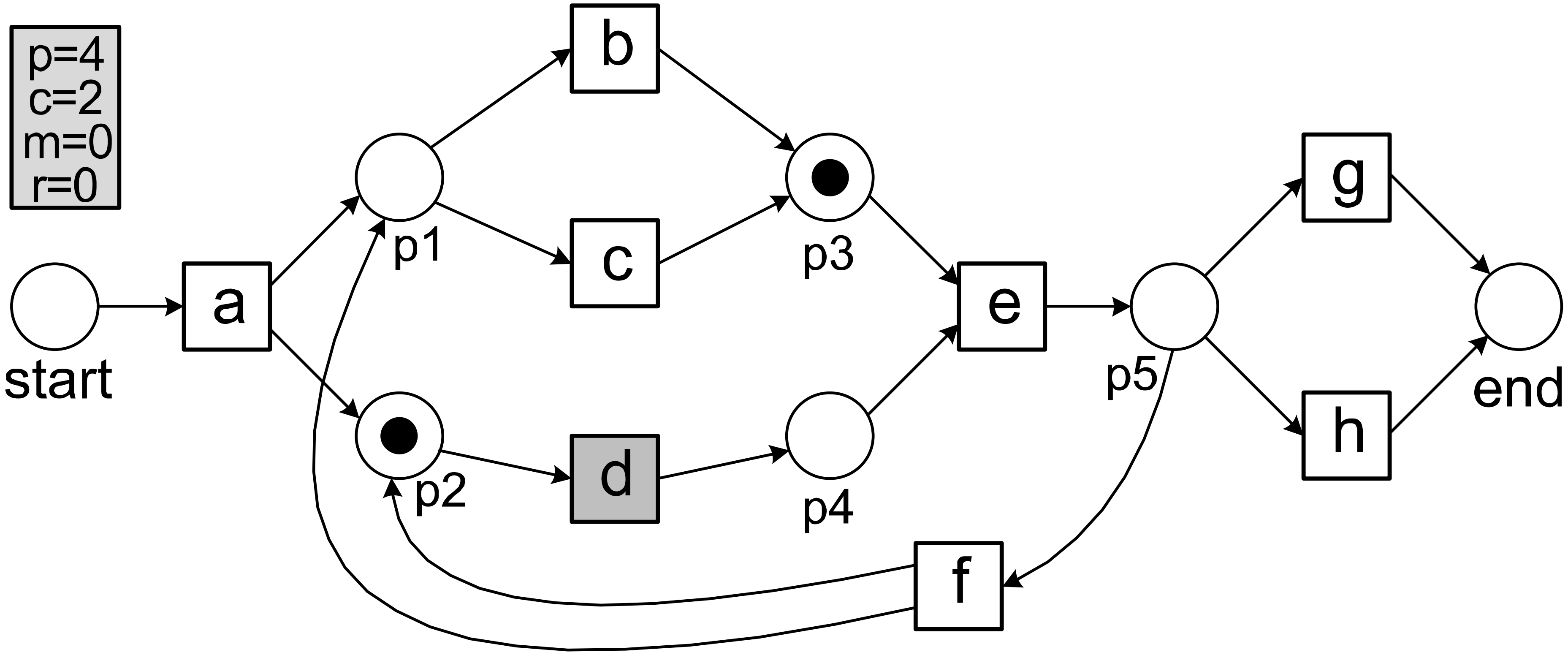
replaying a is possible
one token is consumed, two produced



$$\sigma_1 = \langle a, c, d, e, h \rangle$$

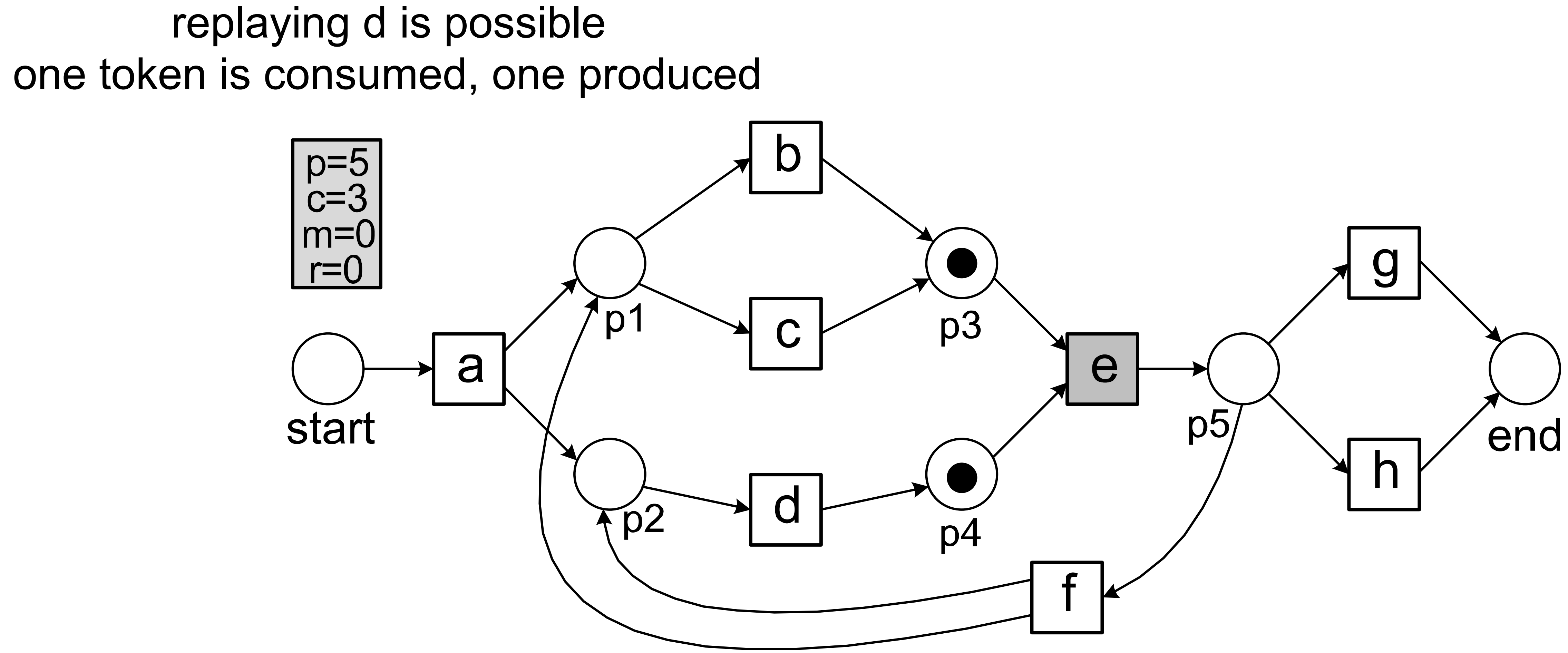
Example

replaying c is possible
one token is consumed, one produced



$$\sigma_1 = \langle a, c, d, e, h \rangle$$

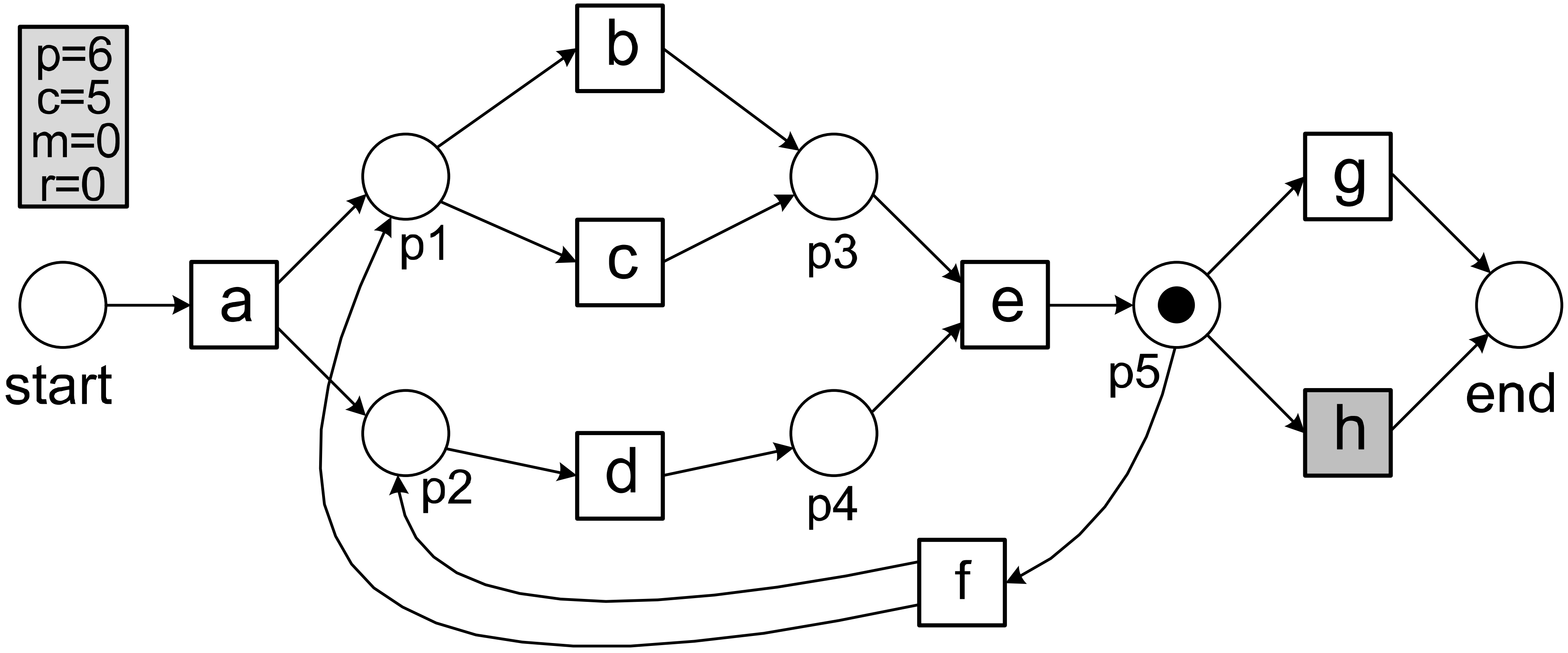
Example



$$\sigma_1 = \langle a, c, d, e, h \rangle$$

Example

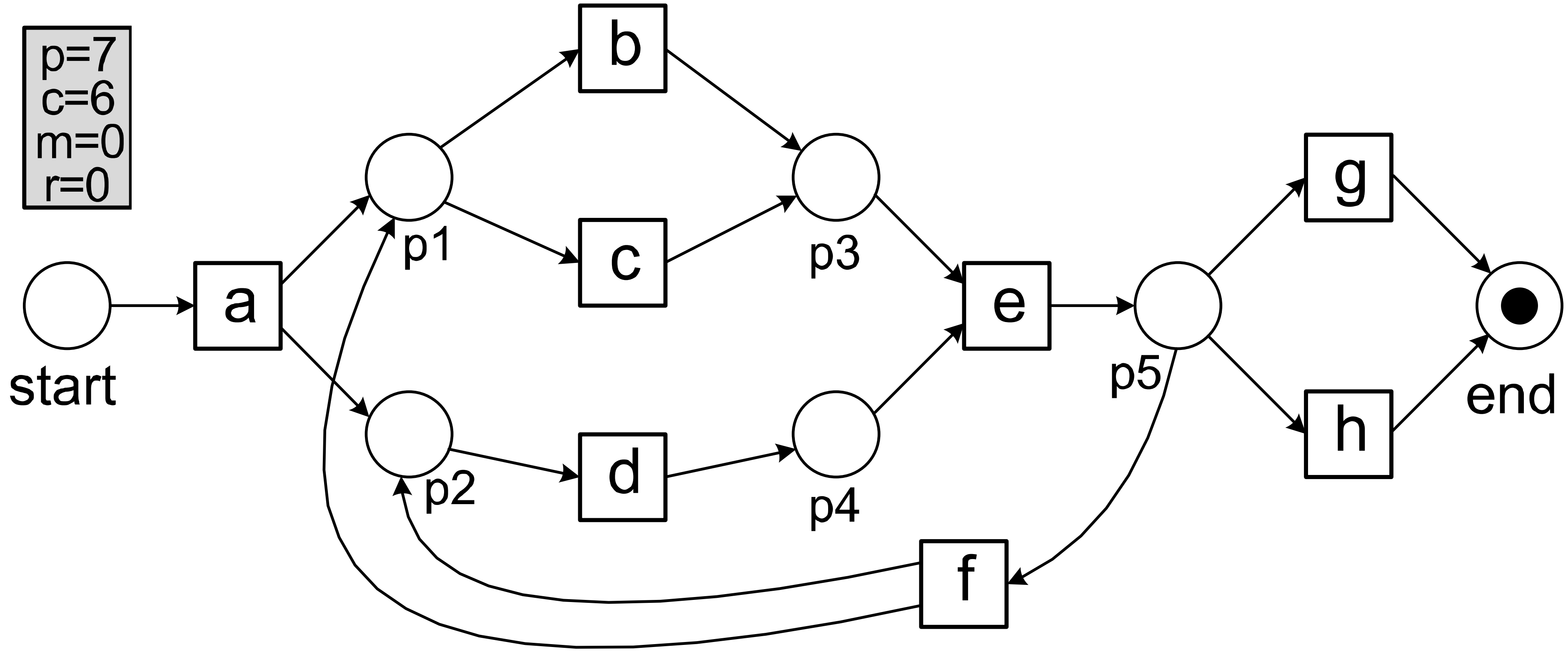
replaying e is possible
two tokens are consumed, one produced



$$\sigma_1 = \langle a, c, d, e, h \rangle$$

Example

replaying h is possible
one token is consumed, one produced

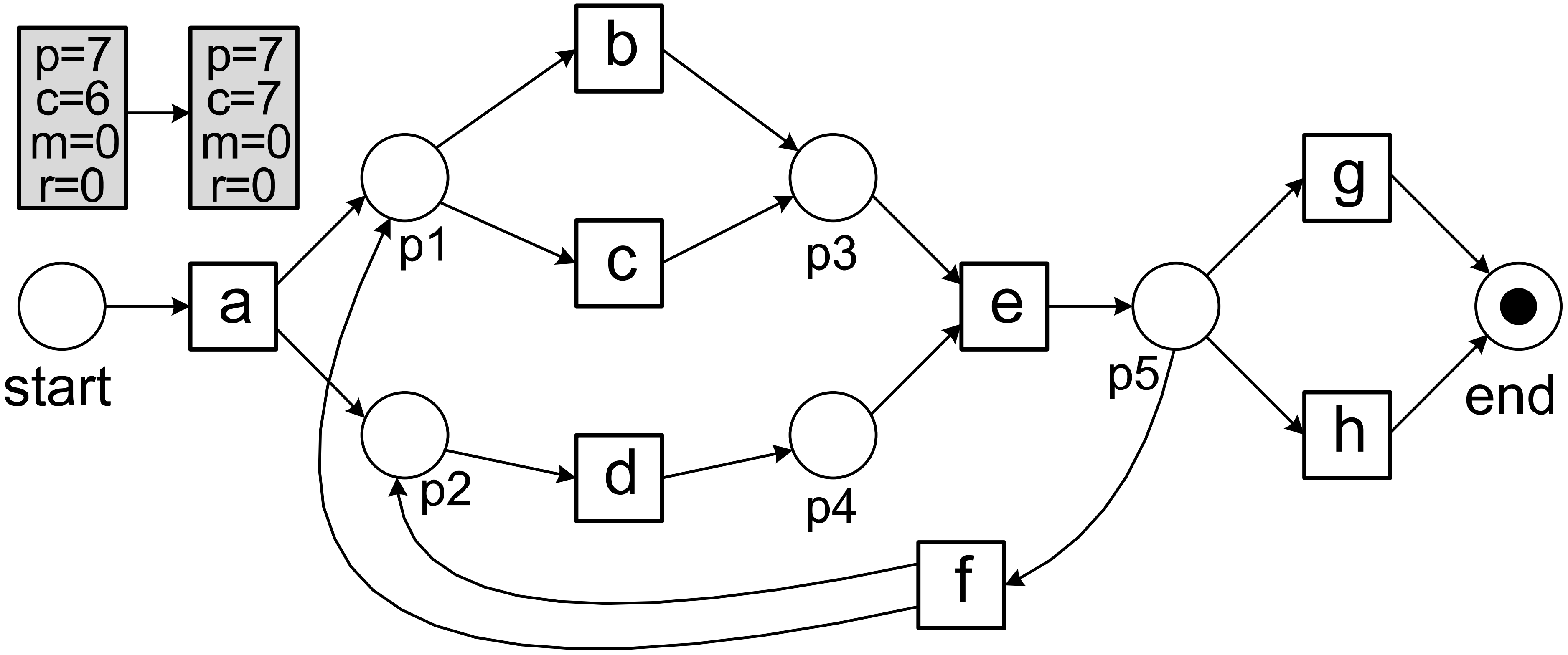


$$fitness(\sigma_1, N_1) = \frac{1}{2}(1 - \frac{0}{7}) + \frac{1}{2}(1 - \frac{0}{7}) = 1$$

$$\sigma_1 = \langle a, c, d, e, h \rangle$$

Example

At the end,
the environment consumes
a token from place end.

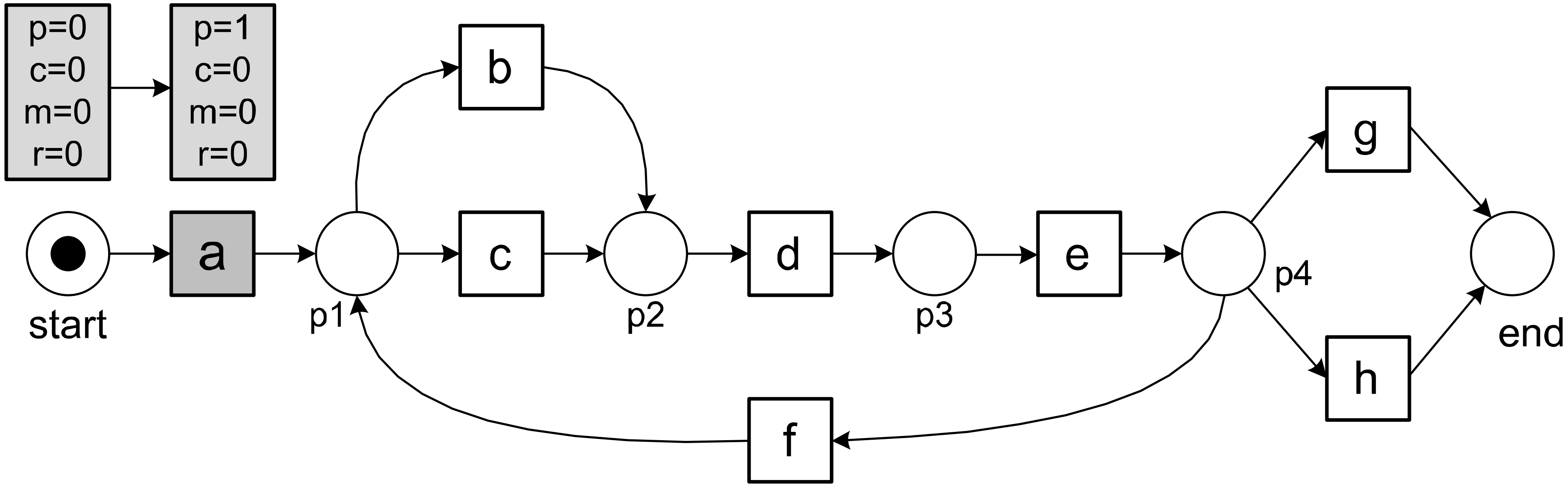


$$fitness(\sigma_1, N_1) = \frac{1}{2}(1 - \frac{0}{7}) + \frac{1}{2}(1 - \frac{0}{7}) = 1$$

$$\sigma_1 = \langle a, c, d, e, h \rangle$$

Example: Missing Token

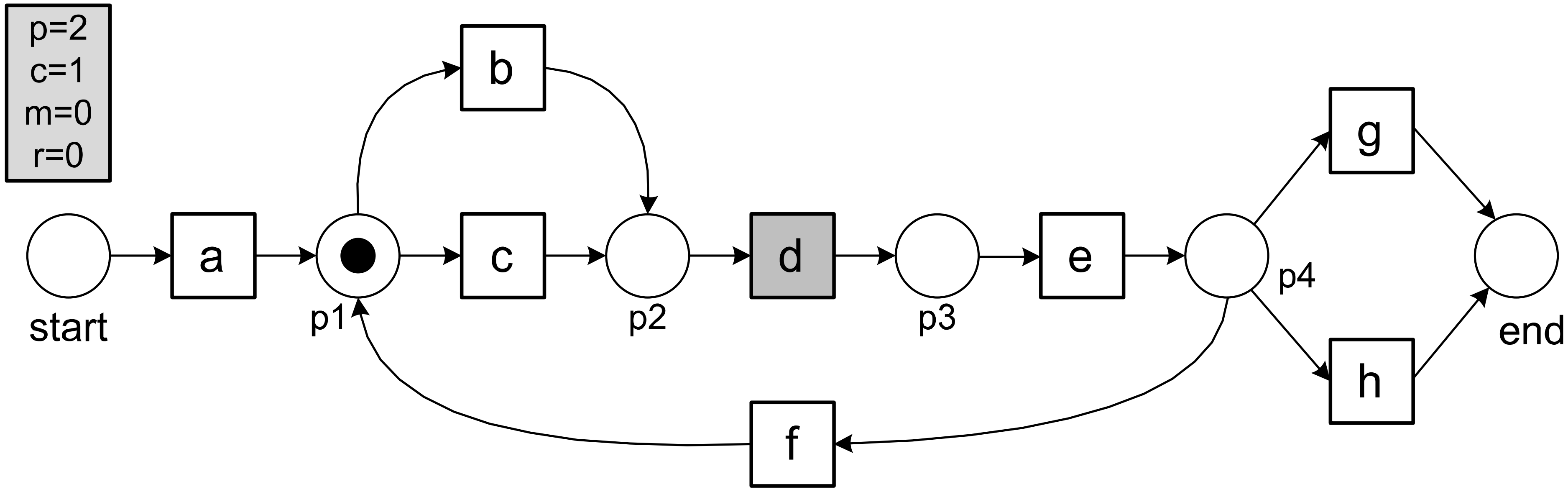
the environment produces a token for place start



$$\sigma_3 = \langle a, d, c, e, h \rangle$$

Example: Missing Token

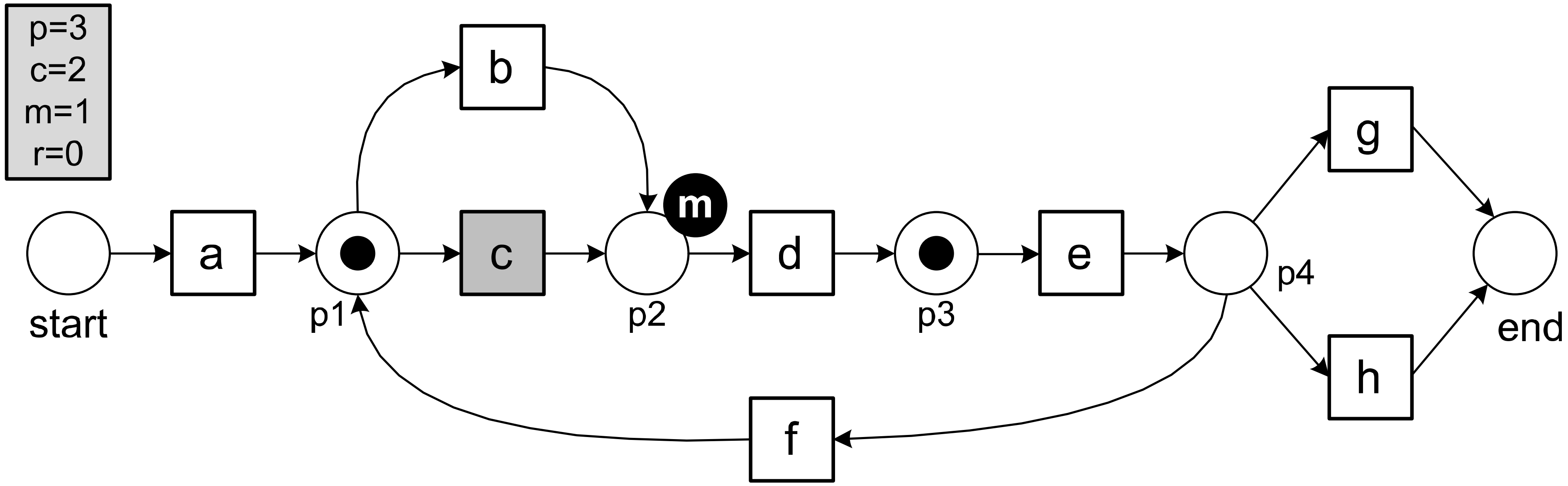
replaying a is possible
one token is consumed, one produced



$$\sigma_3 = \langle a, d, c, e, h \rangle$$

Example: Missing Token

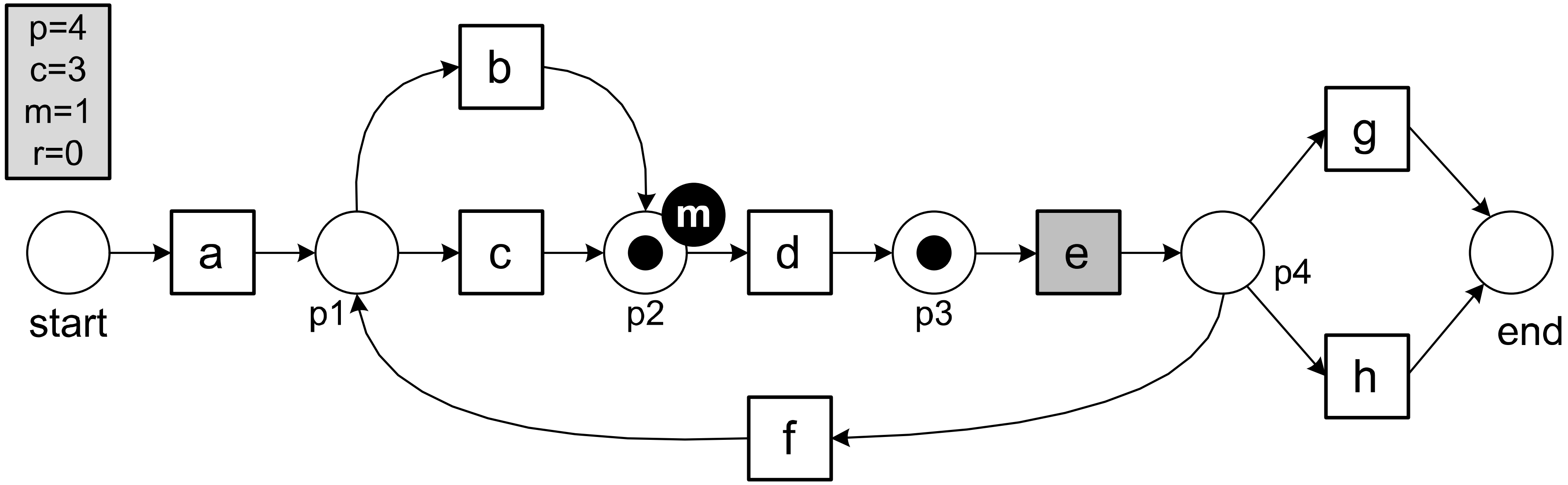
replaying d is NOT possible
one token is missing,
one produced, one consumed



$$\sigma_3 = \langle a, d, c, e, h \rangle$$

Example: Missing Token

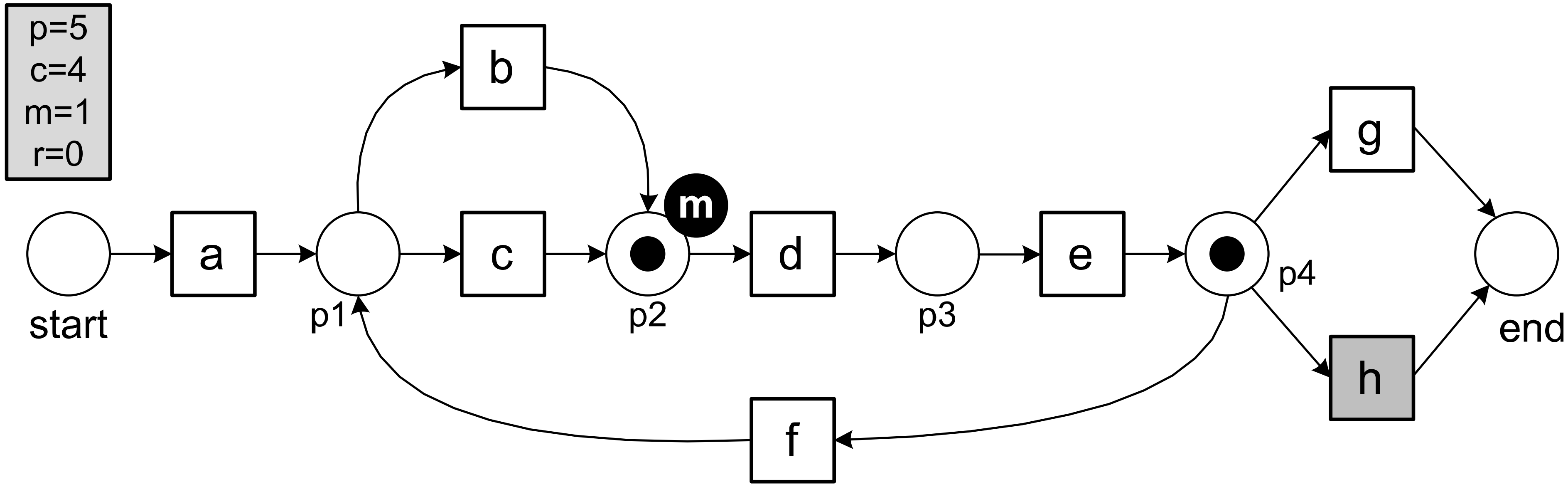
replaying c is possible
one token is produced, one consumed



$$\sigma_3 = \langle a, d, c, e, h \rangle$$

Example: Missing Token

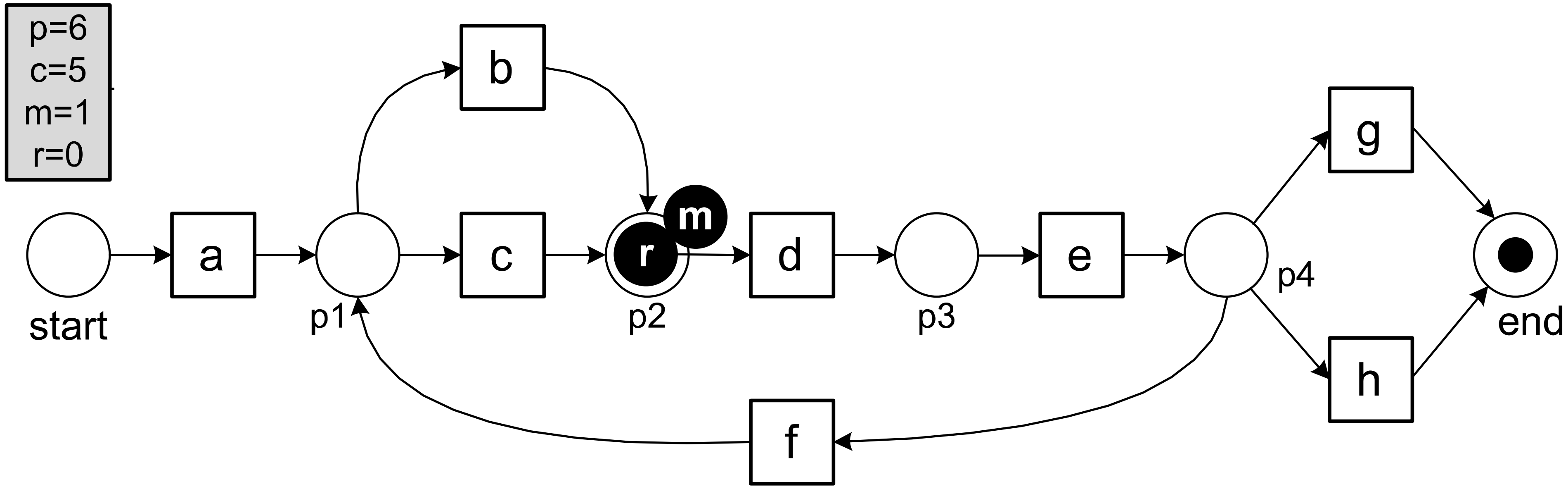
replaying e is possible
one token is produced, one consumed



$$\sigma_3 = \langle a, d, c, e, h \rangle$$

Example: Missing Token

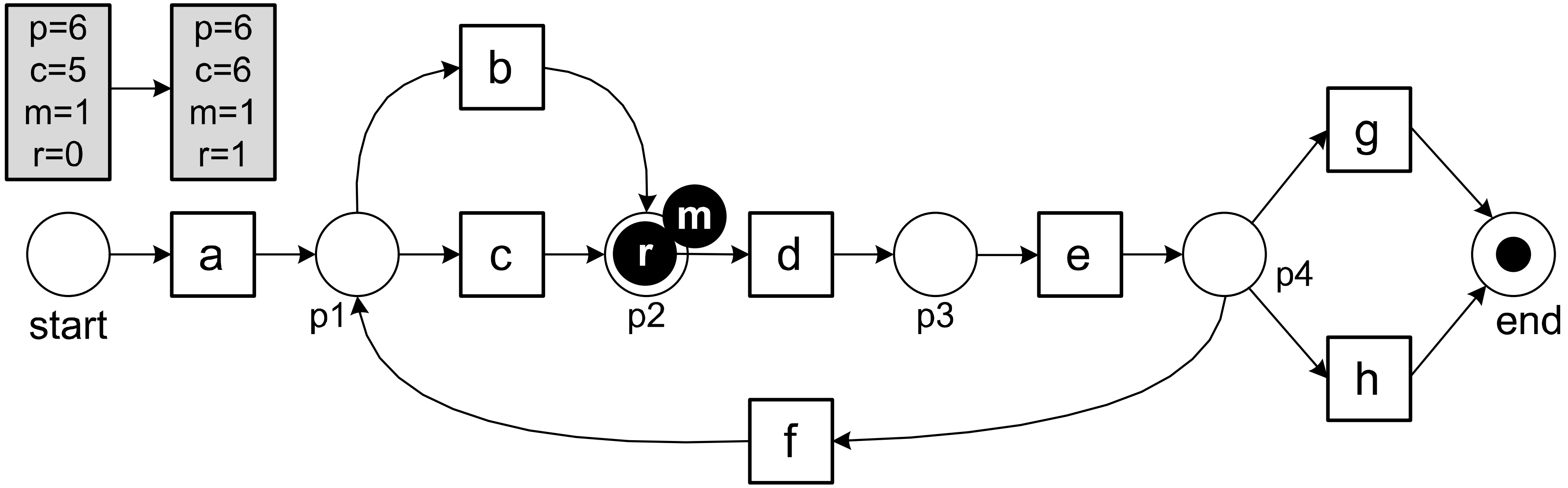
replaying h is possible
one token is produced, one consumed



$$\sigma_3 = \langle a, d, c, e, h \rangle$$

Example: Missing Token

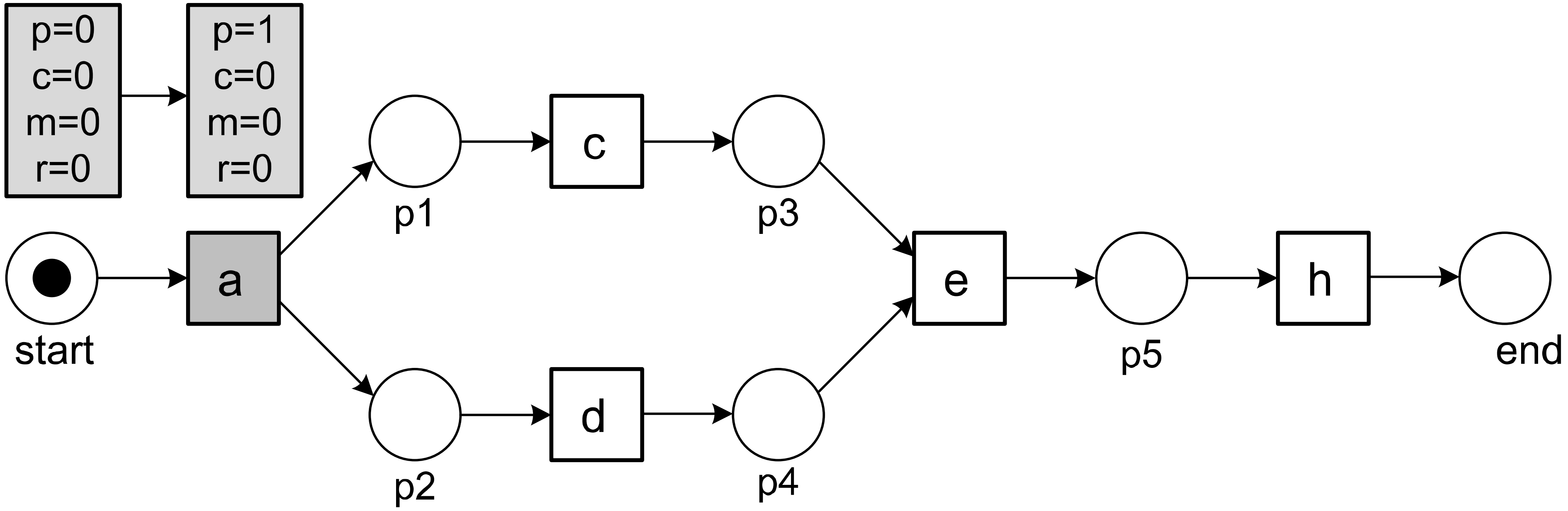
At the end,
the environment consumes
a token from place end.



$$fitness(\sigma_3, N_2) = \frac{1}{2}\left(1 - \frac{1}{6}\right) + \frac{1}{2}\left(1 - \frac{1}{6}\right) = 0.8333$$

$$\sigma_3 = \langle a, d, c, e, h \rangle$$

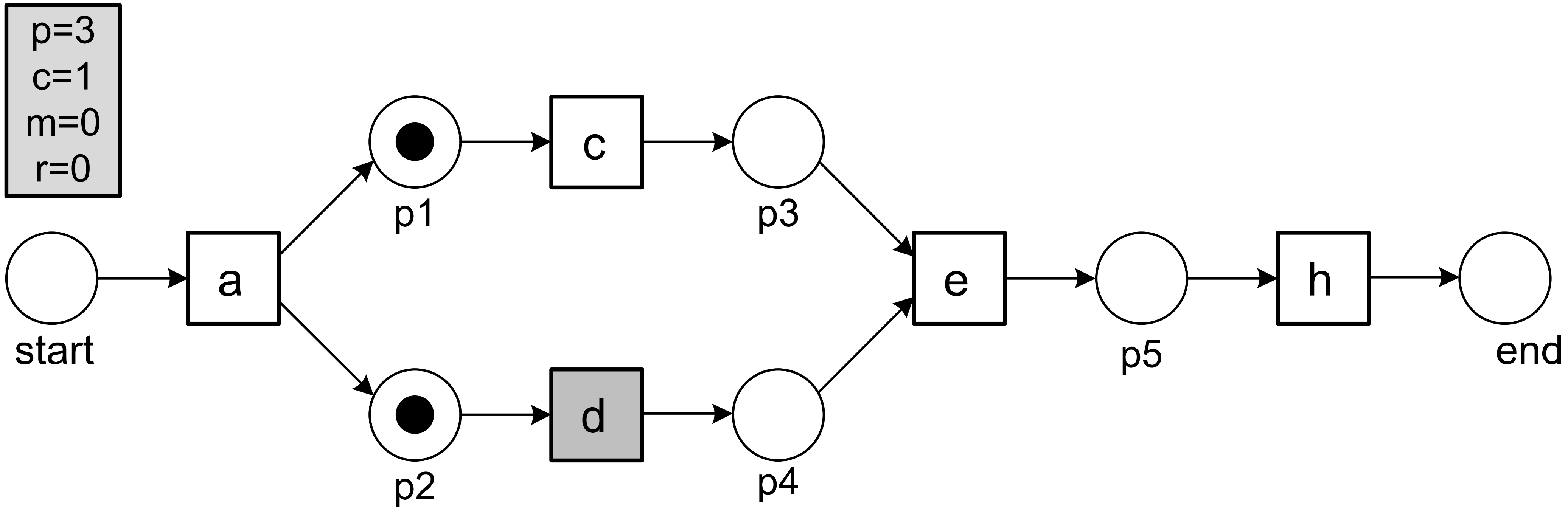
Example: Event Removal



events b and g are not present in the net
therefore we remove them from the trace

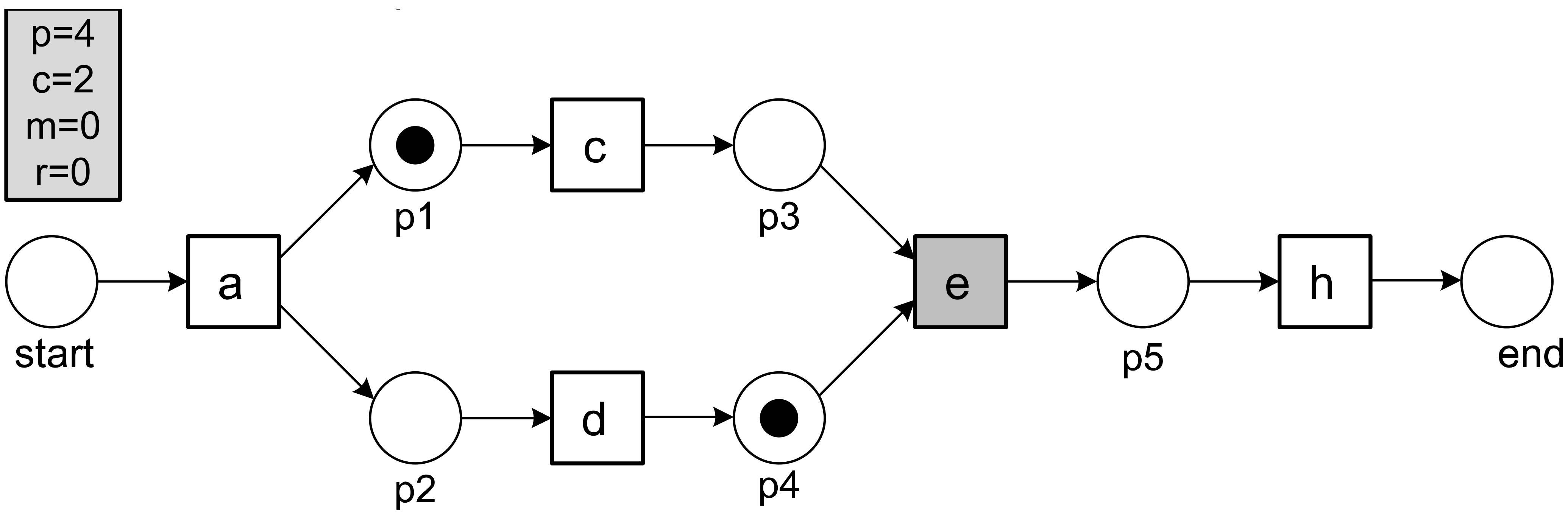
$$\sigma_2 = \langle a, b, d, e, g \rangle \qquad \sigma_2' = \langle a, d, e \rangle$$

Example: Event Removal



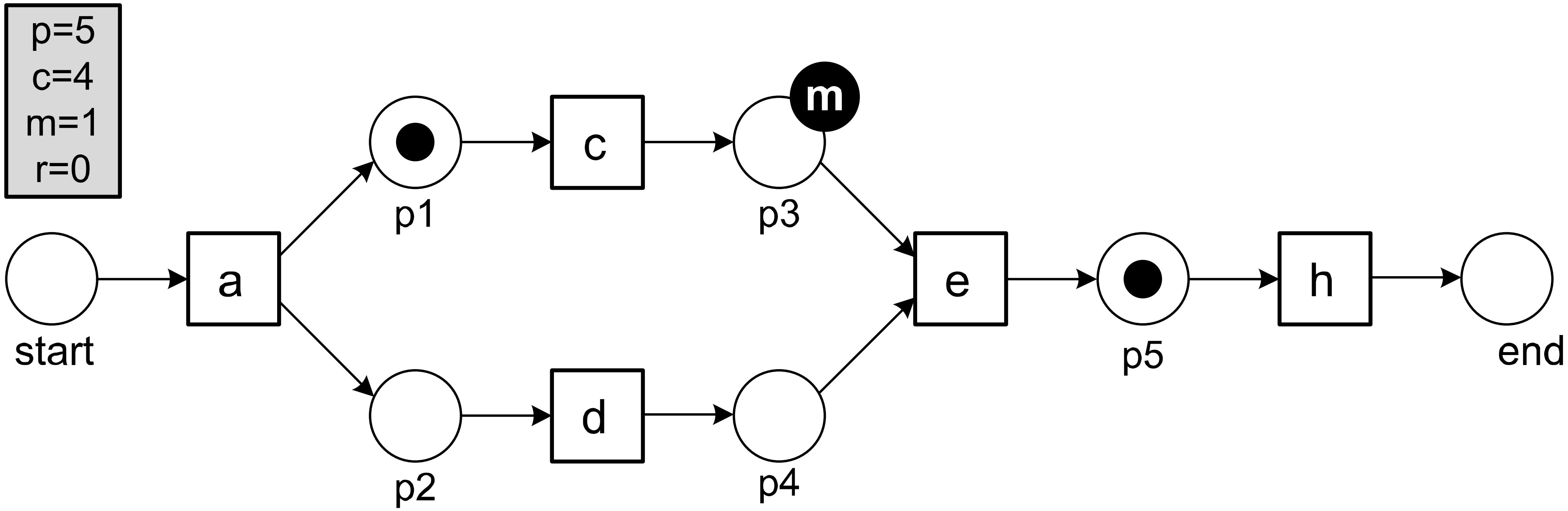
$$\sigma_2' = \langle a, d, e \rangle$$

Example: Event Removal



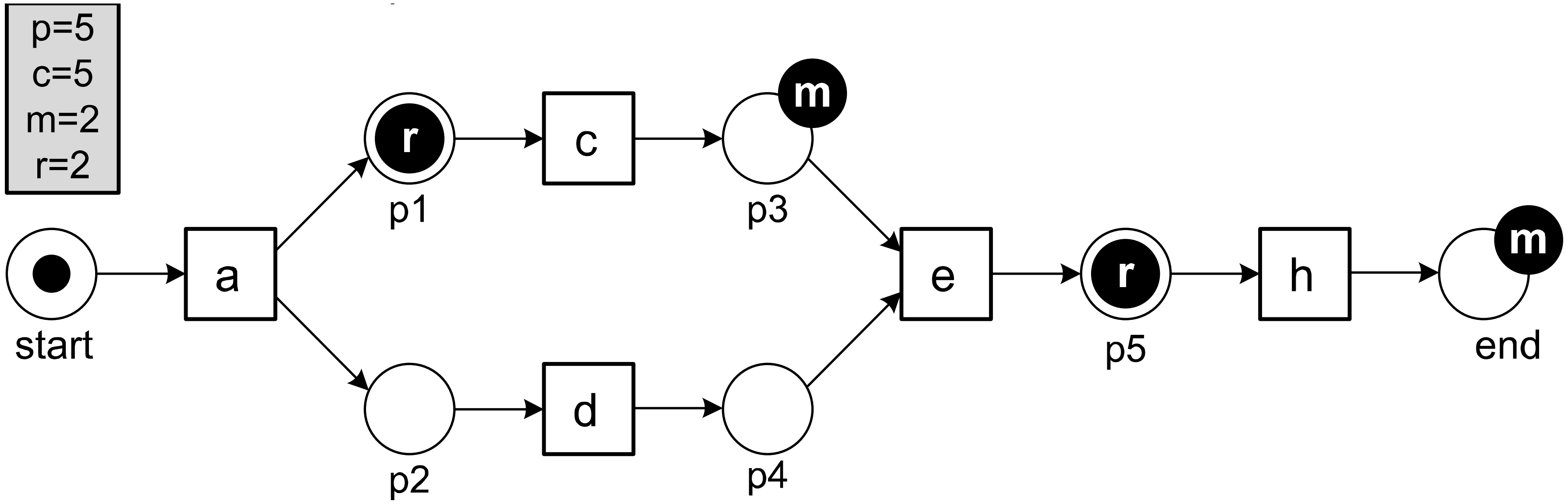
$$\sigma_2' = \langle a, d, e \rangle$$

Example: Event Removal



$$\sigma_2' = \langle a, d, e \rangle$$

Example: Event Removal



$$fitness(\sigma_2, N_3) = \frac{1}{2}\left(1 - \frac{2}{5}\right) + \frac{1}{2}\left(1 - \frac{2}{5}\right) = 0.6$$

$$\sigma_2' = \langle a, d, e \rangle$$

Fitness of a Log

$$fitness(L, N) = \frac{1}{2}\left(1 - \frac{\sum_{\sigma \in L} L(\sigma) \times m_{N,\sigma}}{\sum_{\sigma \in L} L(\sigma) \times c_{N,\sigma}}\right) + \frac{1}{2}\left(1 - \frac{\sum_{\sigma \in L} L(\sigma) \times r_{N,\sigma}}{\sum_{\sigma \in L} L(\sigma) \times p_{N,\sigma}}\right)$$

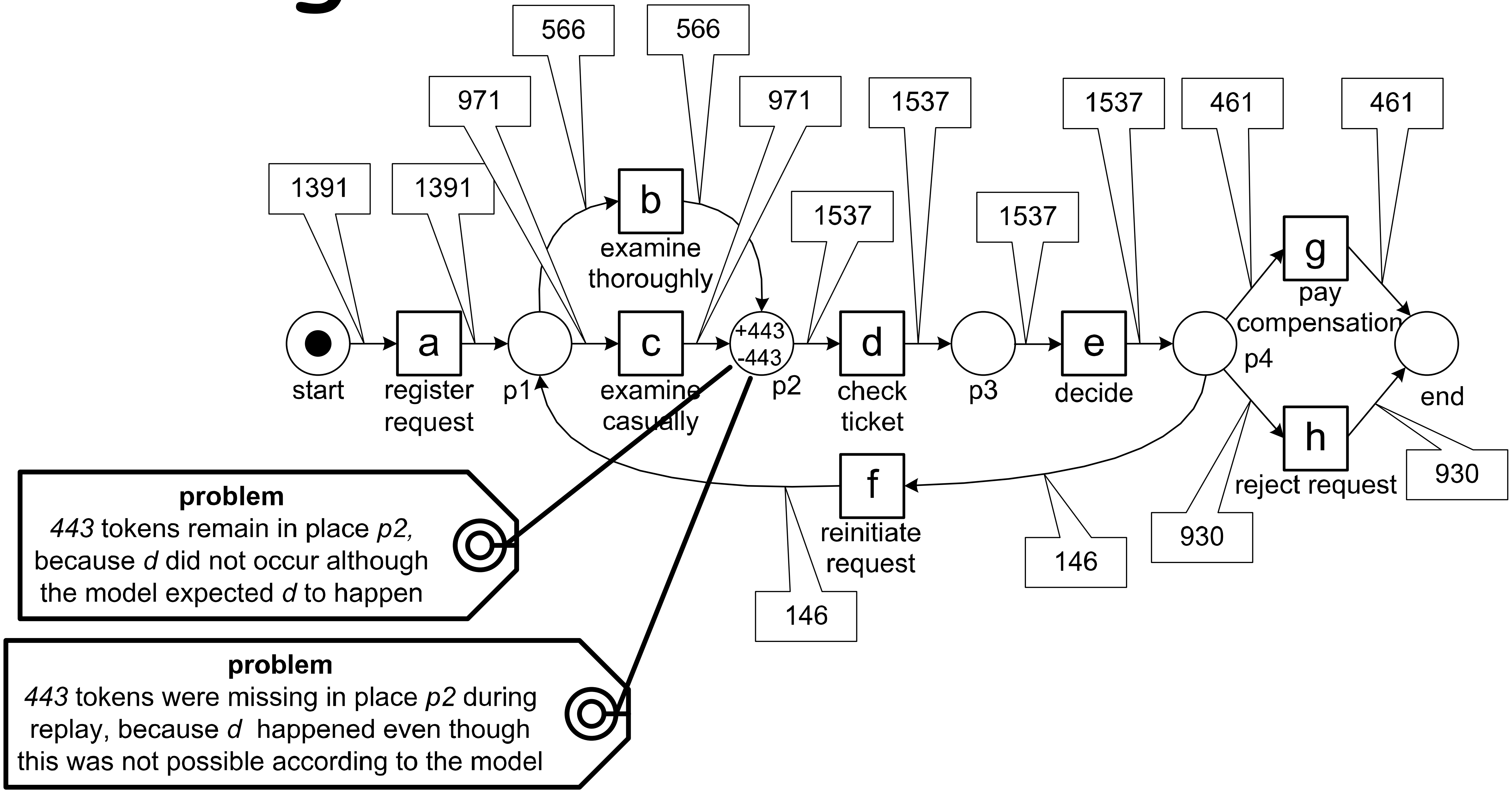
$$fitness(L_{full}, N_1) = 1$$

$$fitness(L_{full}, N_2) = 0.9504$$

$$fitness(L_{full}, N_3) = 0.8797$$

$$fitness(L_{full}, N_4) = 1$$

Diagnostic Information

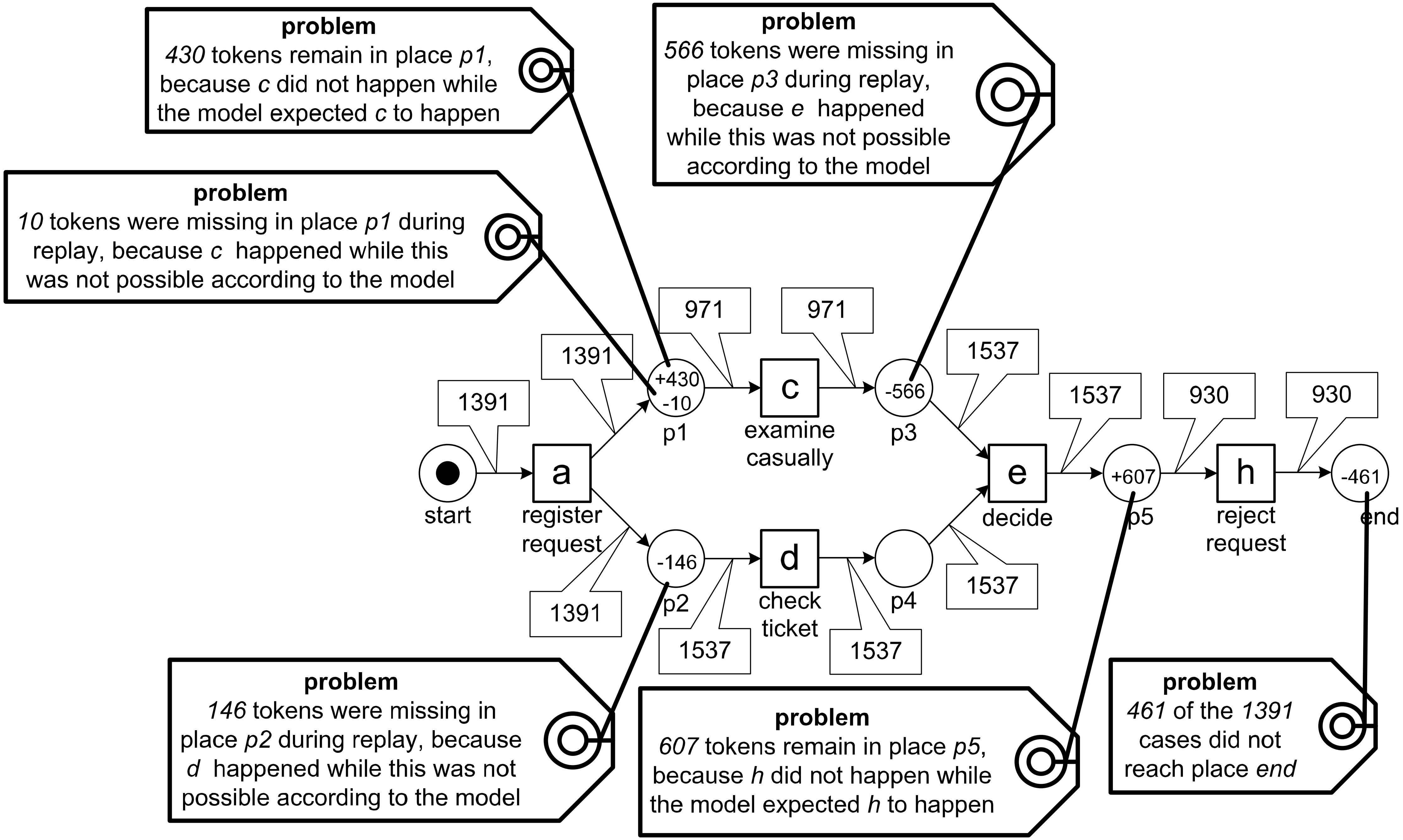


problem
443 tokens remain in place *p2,* because *d* did not occur although the model expected *d* to happen

problem
443 tokens were missing in place *p2* during replay, because *d* happened even though this was not possible according to the model

Fig. 7.6 Diagnostic information showing the deviations ($\mathit{fitness}(L_{full}, N_2) = 0.9504$)

Diagnostic Information



problem
430 tokens remain in place *p1*, because *c* did not happen while the model expected *c* to happen

problem
10 tokens were missing in place *p1* during replay, because *c* happened while this was not possible according to the model

problem
566 tokens were missing in place *p3* during replay, because *e* happened while this was not possible according to the model

problem
146 tokens were missing in place *p2* during replay, because *d* happened while this was not possible according to the model

problem
607 tokens remain in place *p5*, because *h* did not happen while the model expected *h* to happen

problem
461 of the *1391* cases did not reach place *end*

Fig. 7.7 Diagnostic information showing the deviations ($\mathit{fitness}(L_{\mathit{full}}, N_3) = 0.8797$)

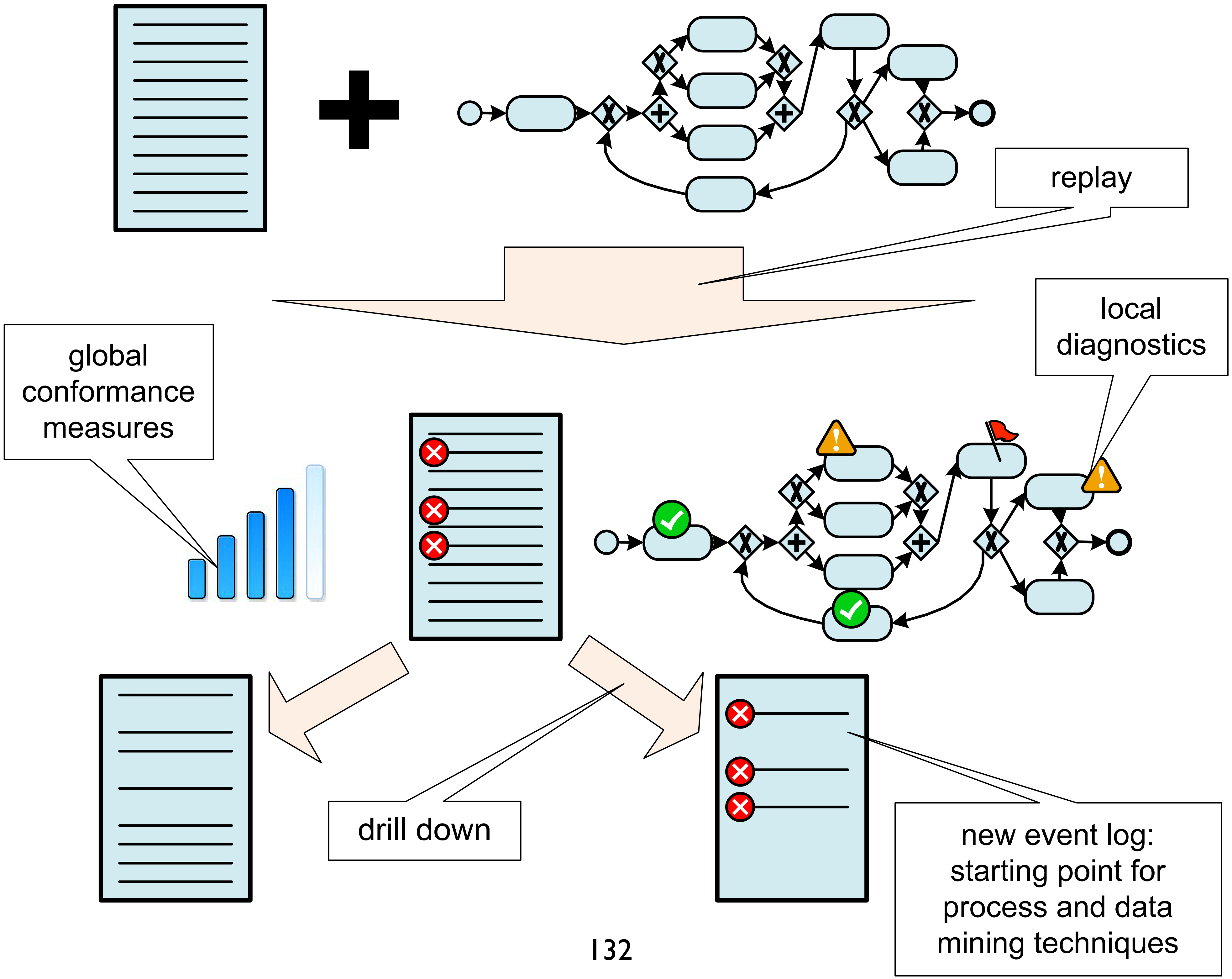
Drill Down

An event log can be split into two sublogs:
one event log containing only fitting cases and
one event log containing only non-fitting cases.

The second event log can be used to discover a different process model.

Also other data and process mining techniques can be used. For instance, it is interesting to know which people handled the deviating cases and whether these cases took longer or were more costly.
In case fraud is suspected, one may create a social network based on the event log with deviating cases.

Drill Down



replay

global conformance measures

local diagnostics

drill down

new event log: starting point for process and data mining techniques

Comparing Footprints

Footprint from Play-out

Given a workflow net, the play-out technique can be used to extract a local complete set of traces.

If we see the set of traces as an event log (without multiplicities), then we can derive the relation >.

Then, we can construct the footprint (i.e. a matrix showing causal dependencies between events) of the net model based on such relation >.

(From the viewpoint of a footprint matrix, an event log is complete if and only if all activities that can follow one another do so at least once in the log.)

Footprint-based Conformance

Footprints are available for logs and models (nets).
This allows for:

log vs model conformance
(do the log and the model agree on the ordering of activities?)

model vs model conformance
(quantification of their similarities)

log vs log comparison
(*concept drift*: how does the work changes in sub-logs?)

Conformance based on footprints

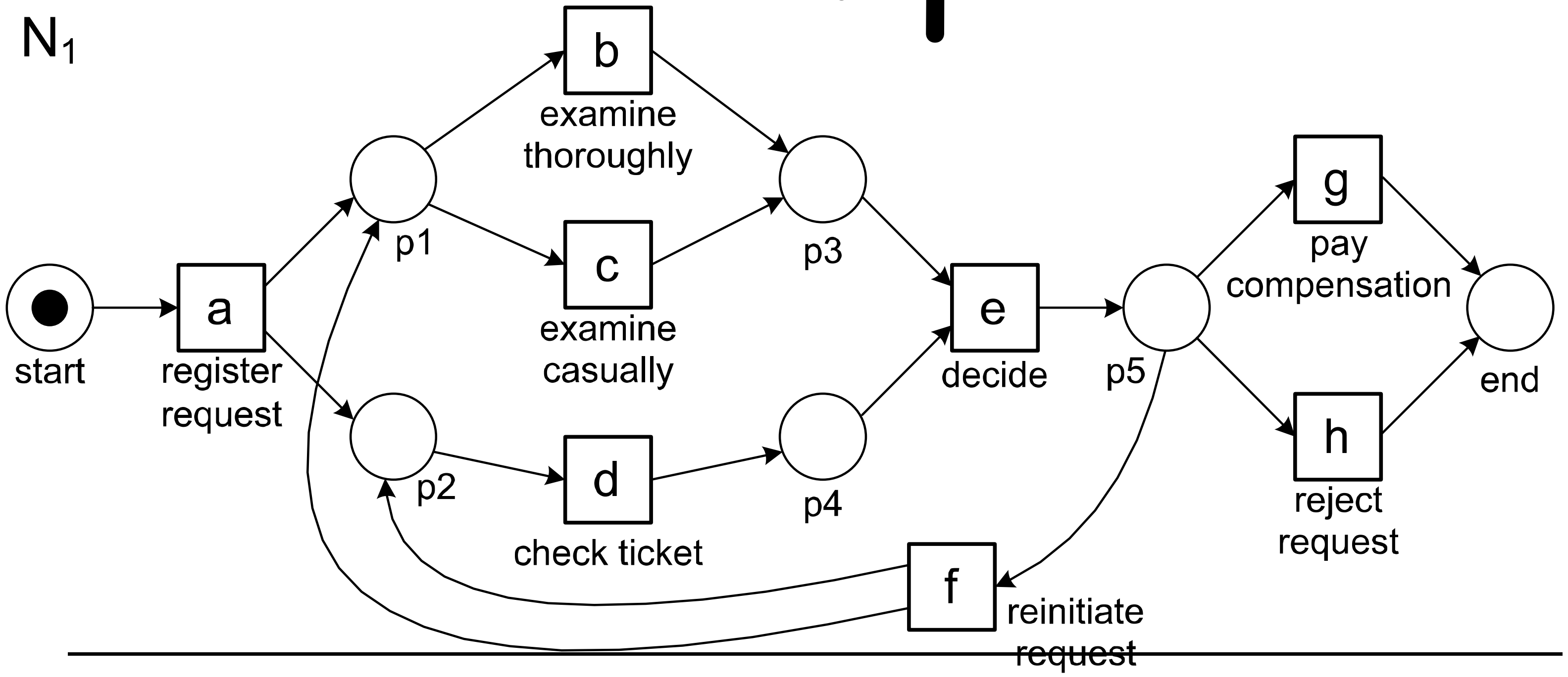
The conformance based on footprints can be computed by taking:

n: total number of cells in the footprint matrix

d: number of cells with different content between the two matrices

$$1 - \frac{d}{n}$$

Example

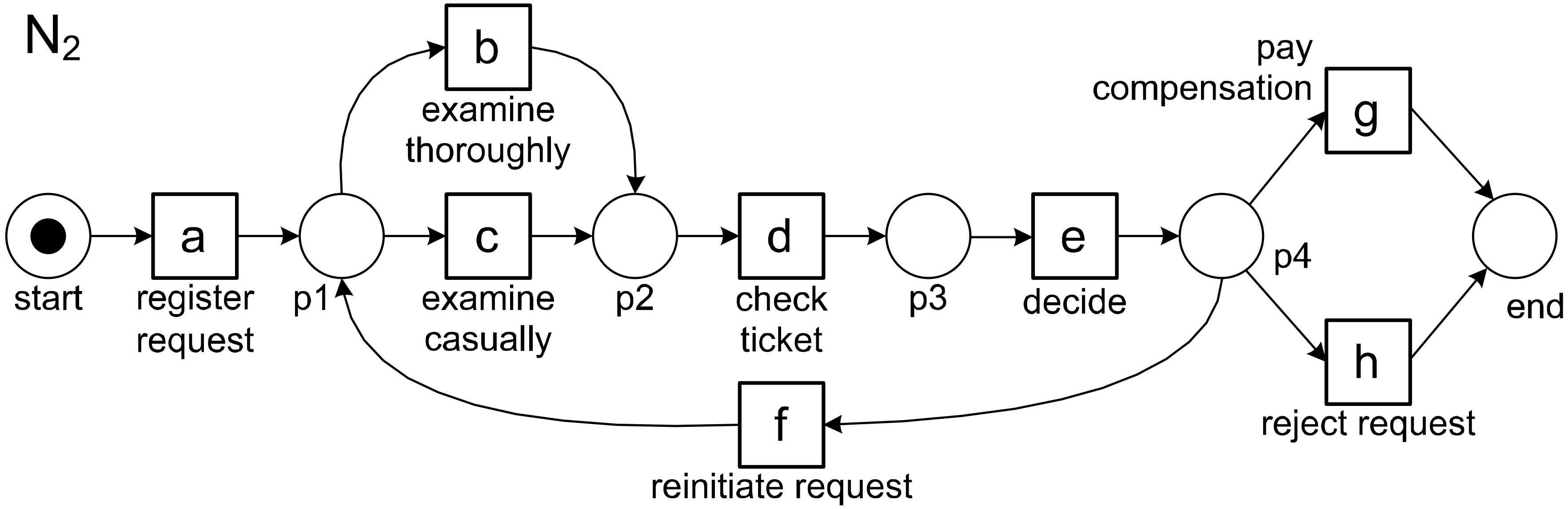


	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
a	#	→	→	→	#	#	#	#
b	←	#	#	∥	→	←	#	#
c	←	#	#	∥	→	←	#	#
d	←	∥	∥	#	→	←	#	#
e	#	←	←	←	#	→	→	→
f	#	→	→	→	←	#	#	#
g	#	#	#	#	←	#	#	#
h	#	#	#	#	←	#	#	#

Also

Footprint of L_{full}

Example



	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
a	#	→	→	#	#	#	#	#
b	←	#	#	→	#	←	#	#
c	←	#	#	→	#	←	#	#
d	#	←	←	#	→	#	#	#
e	#	#	#	←	#	→	→	→
f	#	→	→	#	←	#	#	#
g	#	#	#	#	←	#	#	#
h	#	#	#	#	←	#	#	#

Example

	a a	b b	c c	d d	e e	f f	g g	h h
a a	# #	→→	→→	→#	# #	# #	# #	# #
b b	←←	# #	# #	‖ →	→#	←←	# #	# #
c c	←←	# #	# #	‖ →	→#	←←	# #	# #
d d	←#	‖ ←	‖ ←	# #	→→	←#	# #	# #
e e	# #	←#	←#	←←	# #	→→	→→	→→
f f	# #	→→	→→	→#	←←	# #	# #	# #
g g	# #	# #	# #	# #	←←	# #	# #	# #
h h	# #	# #	# #	# #	←←	# #	# #	# #

Example

$$1 - \frac{12}{64} = 0.8125$$

	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
a				→: #				
b				‖ :→	→: #			
c				‖ :→	→: #			
d	←: #	‖ :←	‖ :←			←: #		
e		←: #	←: #					
f				→: #				
g								
h								