Methods for the specification and verification of business processes
MPB (6 cfu, 295AA)
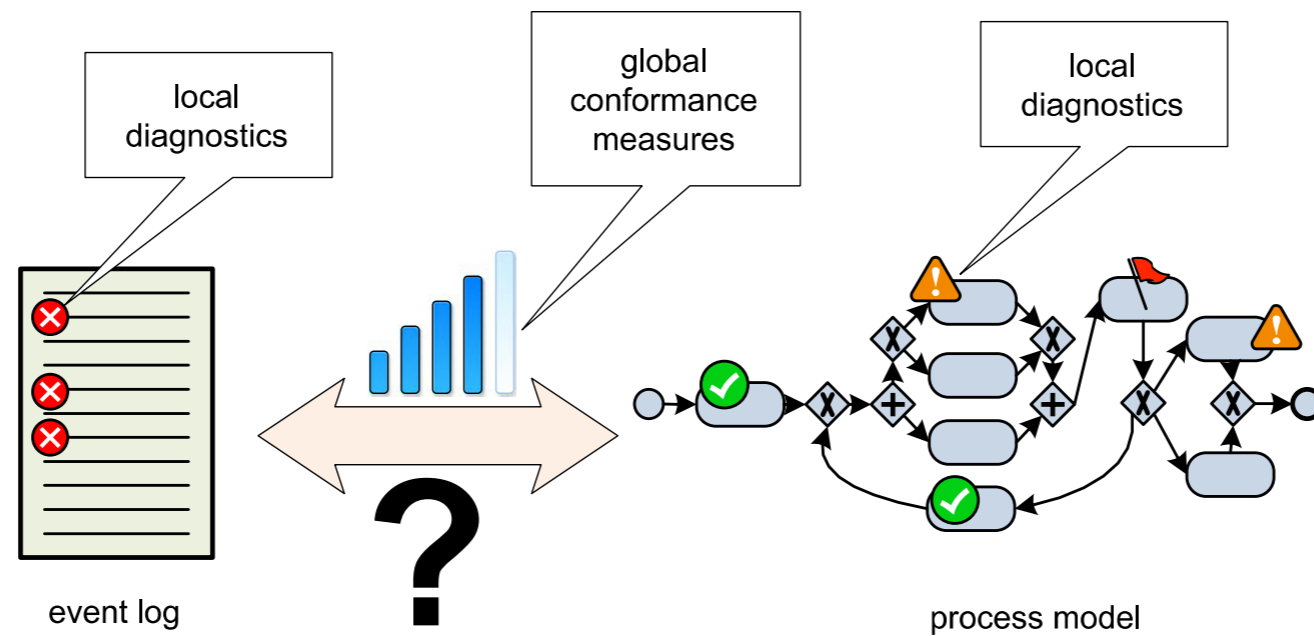
Roberto Bruni

http://www.di.unipi.it/~bruni
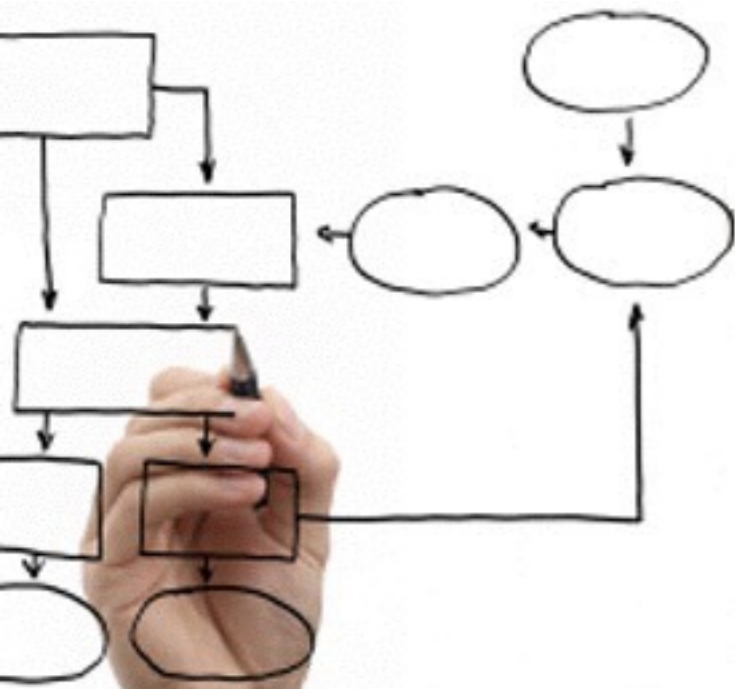
24 - Process Mining

Object

We overview the key principles of process mining

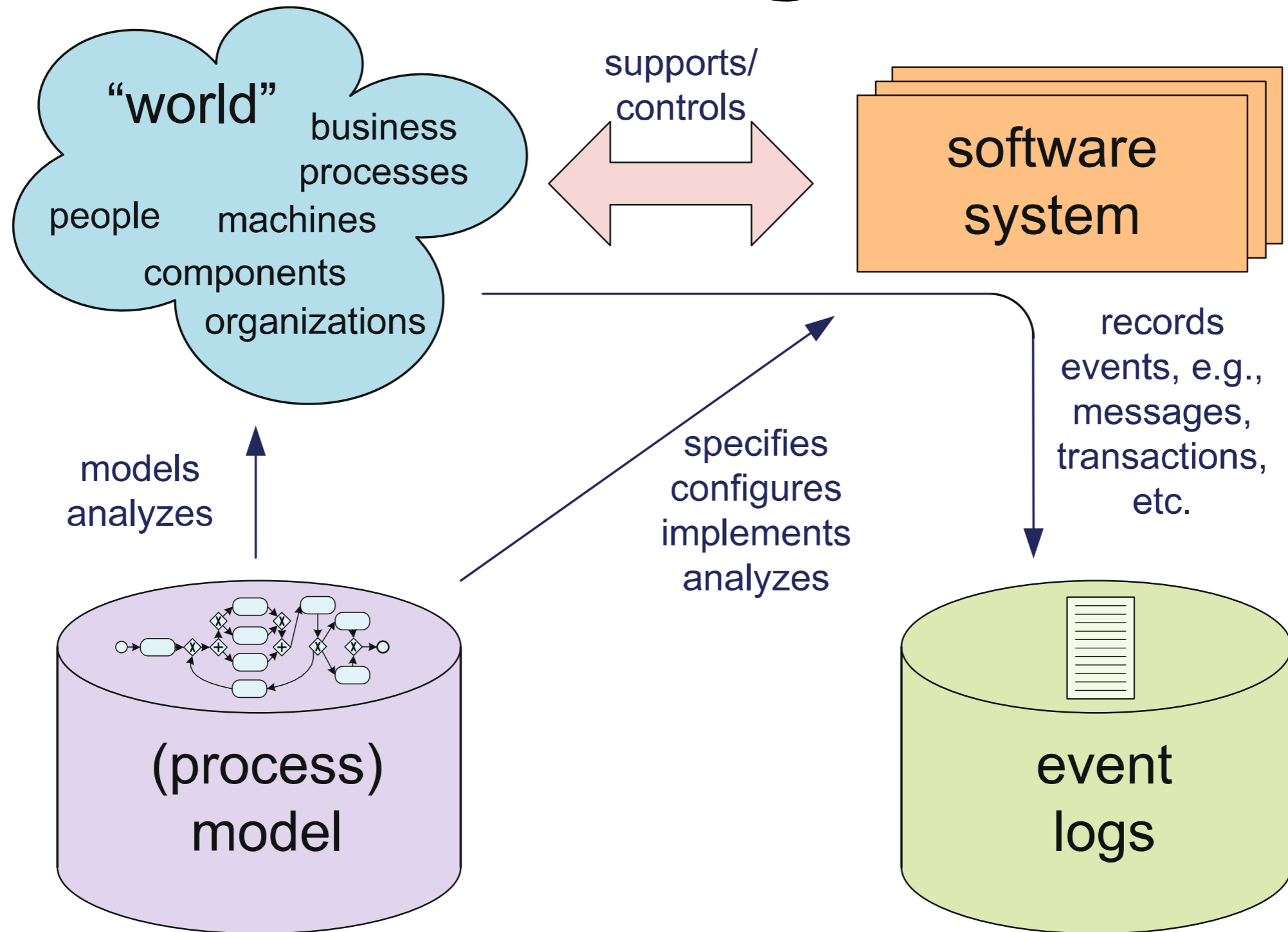Chapters 1, 5, 7. Process Mining. W. van der Aalst

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 assumed ones) by extracting knowledge from event logs readily available in today's systems.

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.

Events can have attributes.

Examples of typical attribute names are activity, time, costs, 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
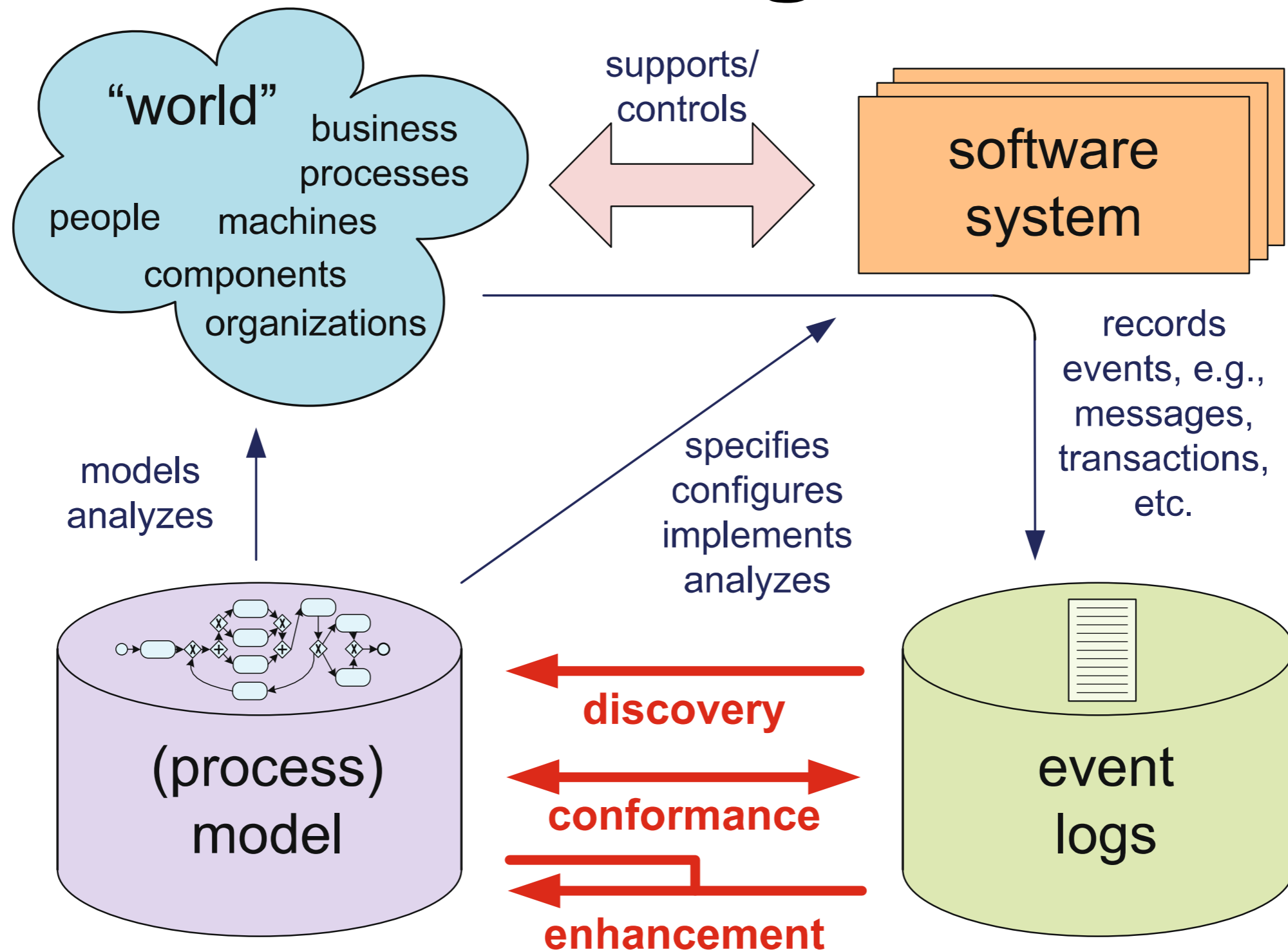
Discovery

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
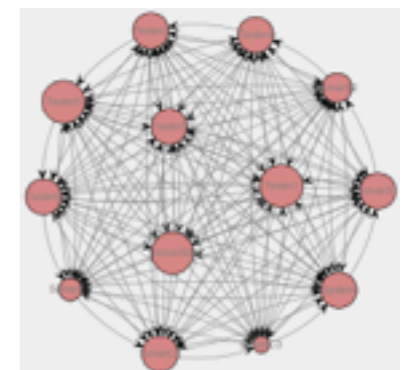showing how people work together in an organization.
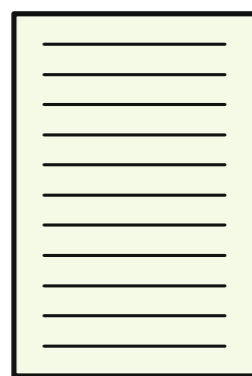
event log
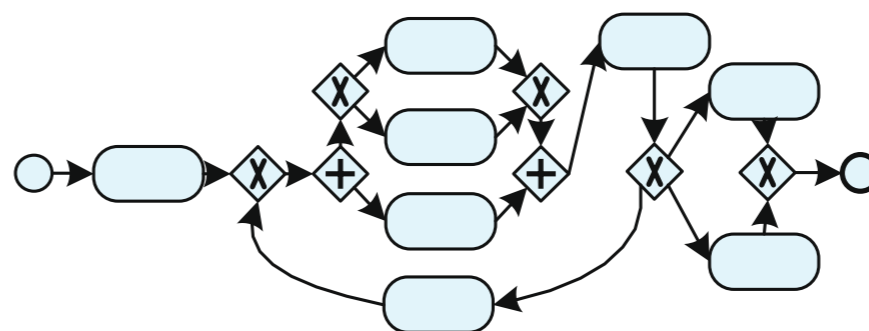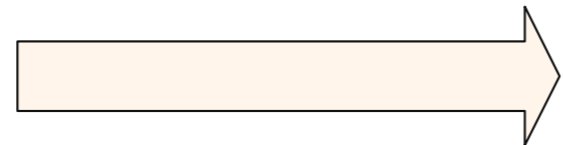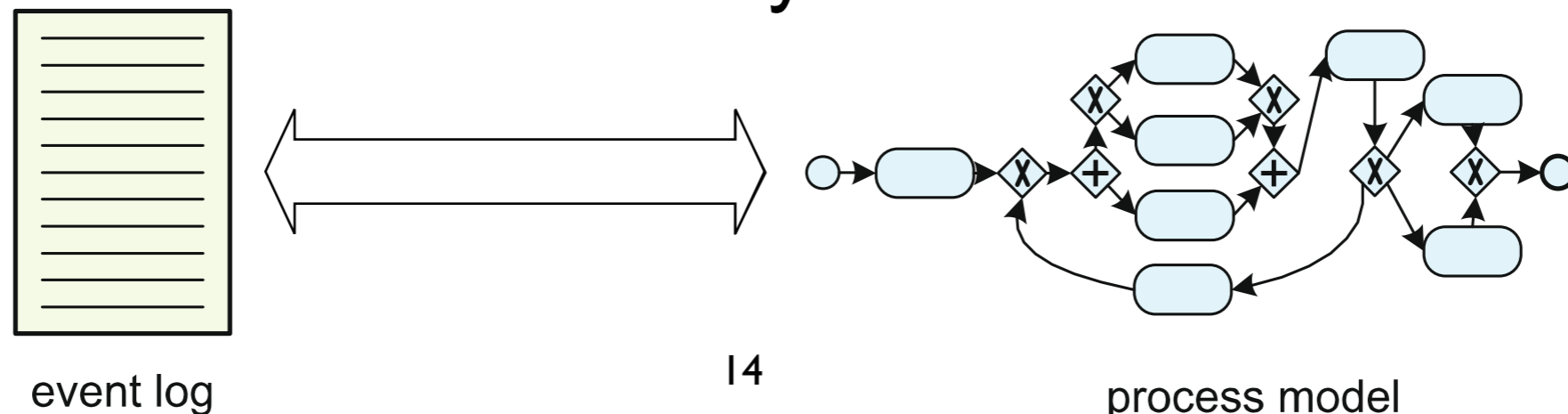
process model

social network

Conformance

An existing process model is compared
with an event log of the same process.

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

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

event log

process model

Enhancement

The idea is to **extend/improve an existing process model** using information about the actual process recorded in some event log.

Whereas conformance checking measures the alignment between a model and reality, this third type of process mining aims at changing or extending the a-priori model.

event log   process model
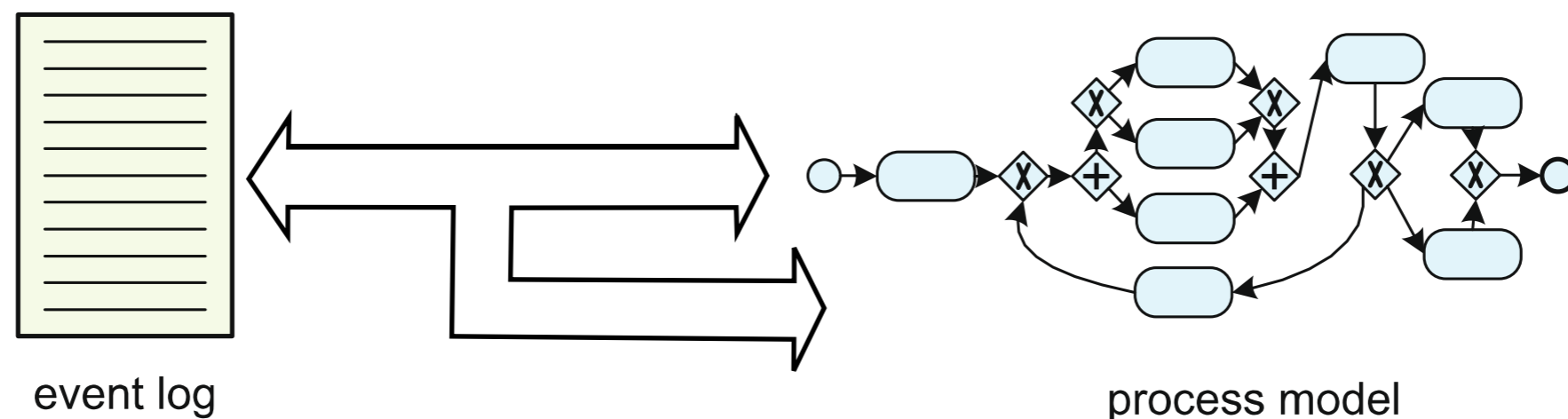
Enhancement: Repair

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.

Two Angles

Conformance check is based on the comparison between an event log and a process model.

(Un)desirable deviations can be detected.

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?").

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

The term auditing refers to the evaluation of organizations and their processes.

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

New Forms of Auditing

However, today detailed information about processes is being recorded in the form of event logs, audit trails, transaction logs, databases, data warehouses, etc.

All events in a business process can be evaluated and this can be done while the process is still running.

The availability of log data and advanced process mining techniques enables new forms of auditing, and conformance checking in particular, provide the means to do so.

Four Perspectives

Organizational Perspective

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

The goal is to either 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, if a case represents a replenishment order, it may be interesting to know the supplier or the number of products ordered.

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
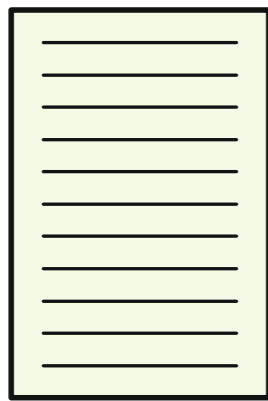running cases.

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).
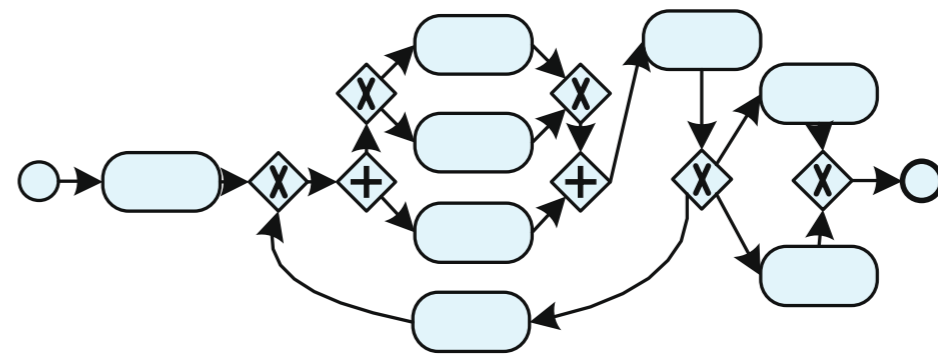
We shall focus on this perspective only

Play-in

Play-In

event log
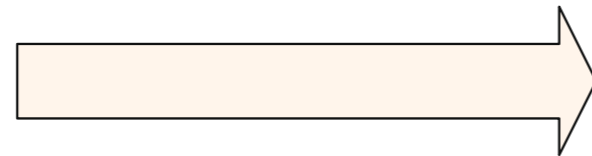
process model

Mining
Discovery

Play-out
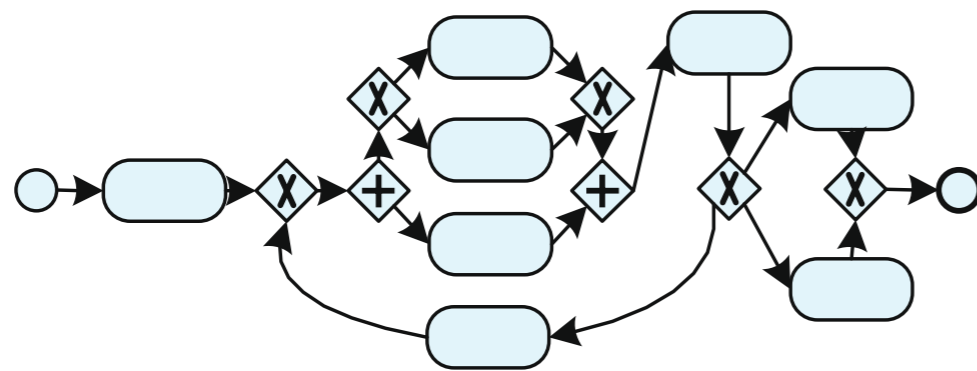
process model
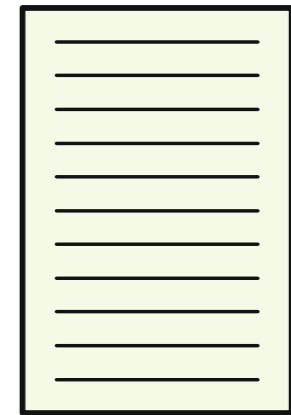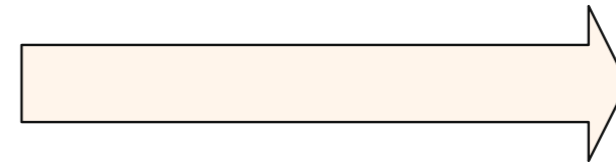
behaviour

Workflow engine
Simulation engine
Trace generation
Model checking

Replay

Replay

event log

process model

- extended model showing times, frequencies, etc.
- diagnostics
- predictions
- recommendations
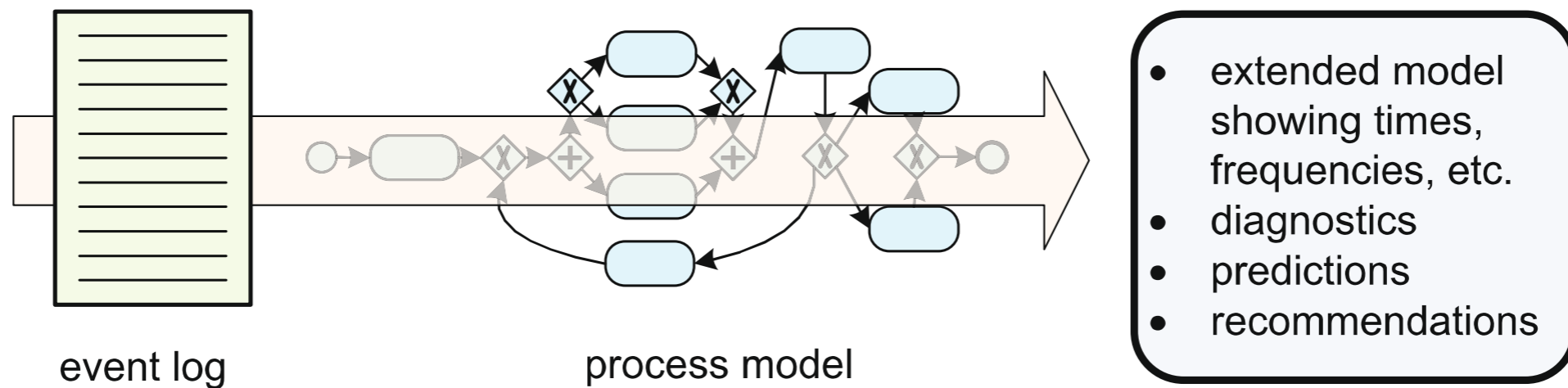
Conformance checking
Performance analysis
Bottlenecks detection
Predictive models (based on past)
Operational support (deviation detection)

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

30

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		

33

Event Log Fragment

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

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	...
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	...

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	...
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 $h = reject\ request$

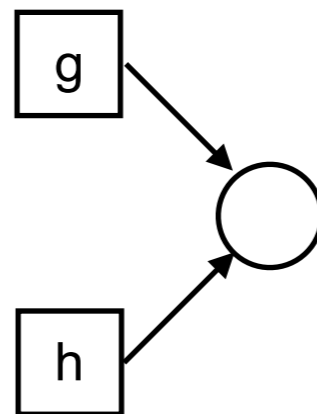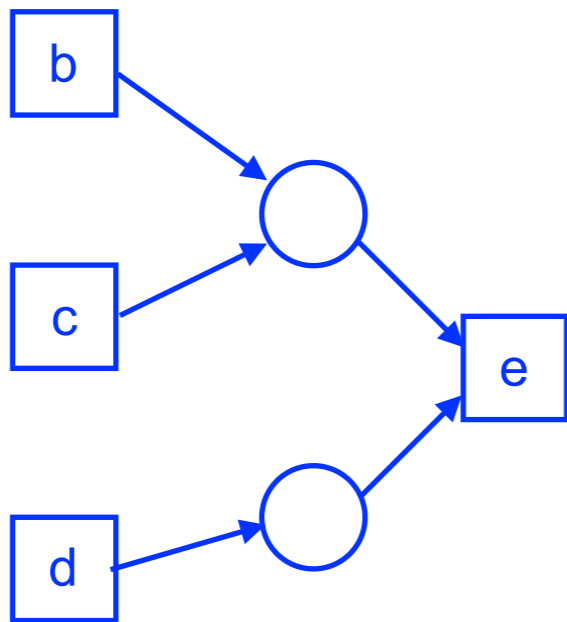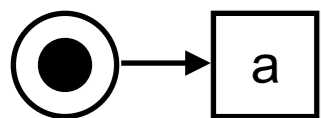| Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
...	...

35

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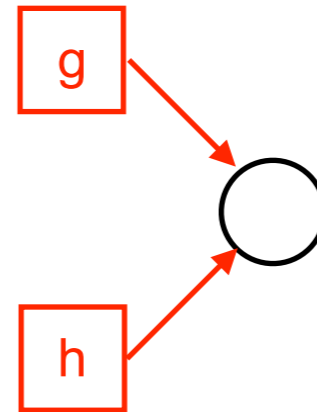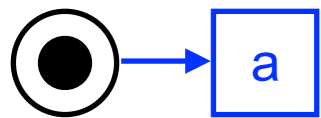
All cases start with a
and end with either g or h.

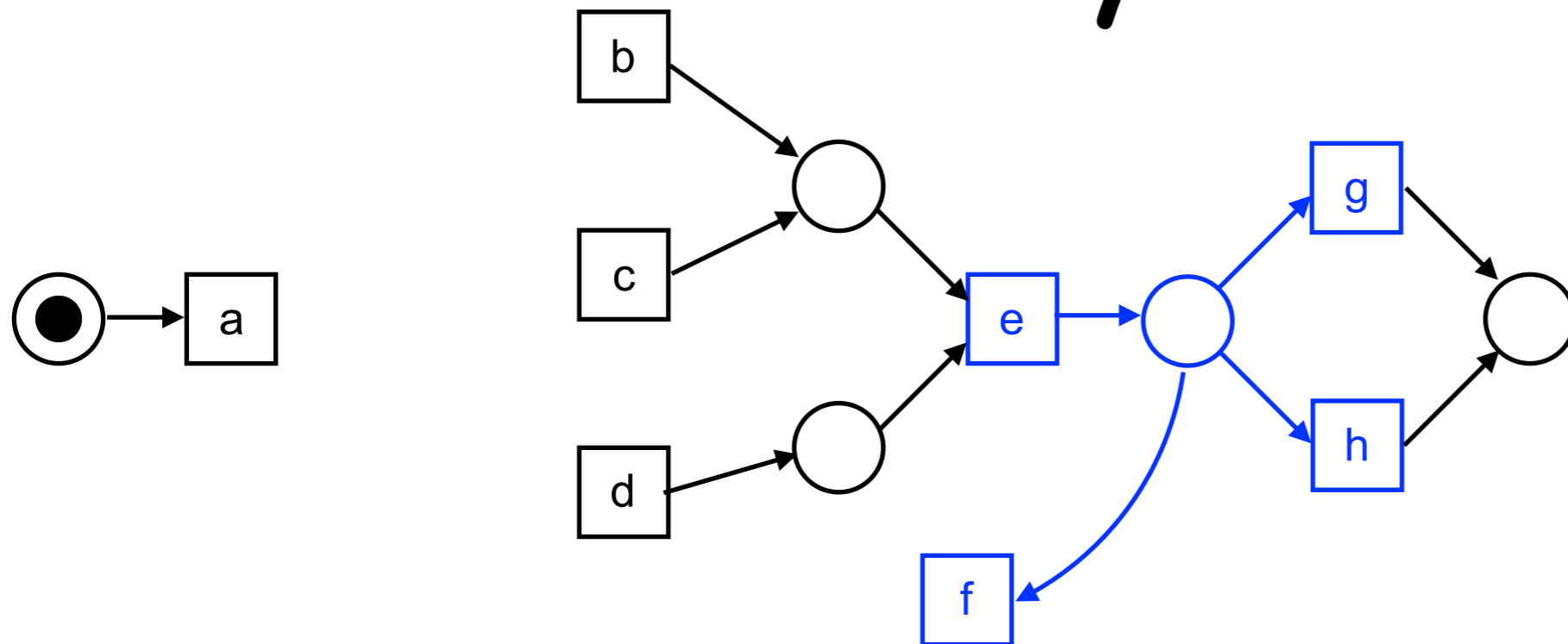| Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
...	...

Discovery Example

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

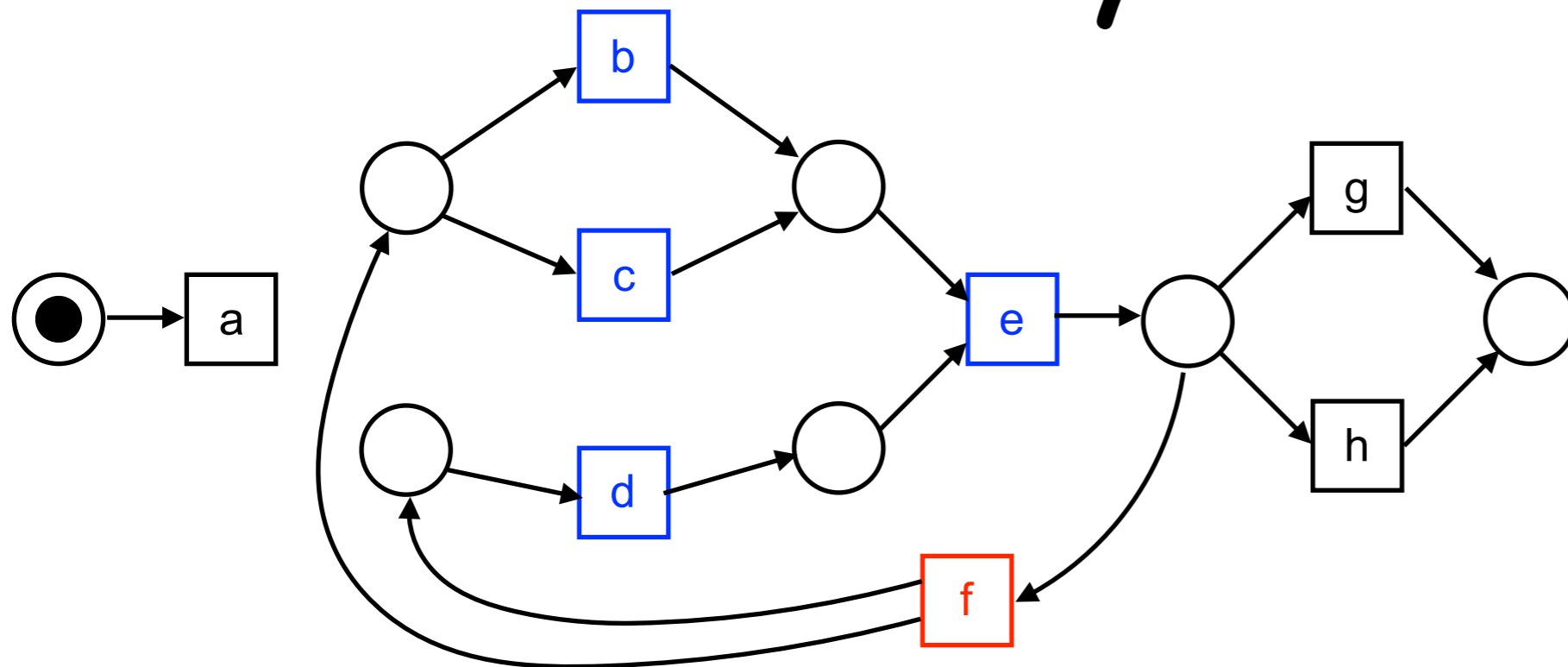| Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
…	…

37

Discovery Example

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

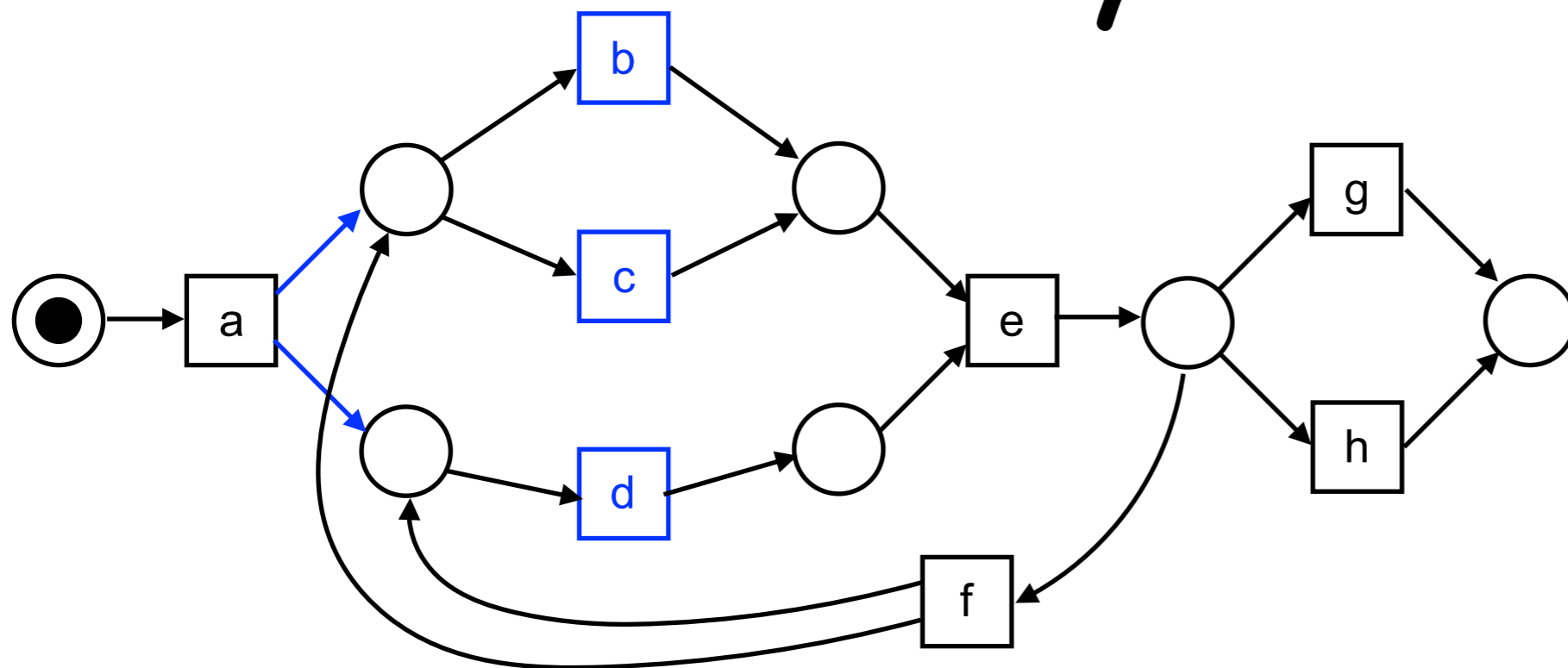| Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
...	...

38

Discovery Example

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

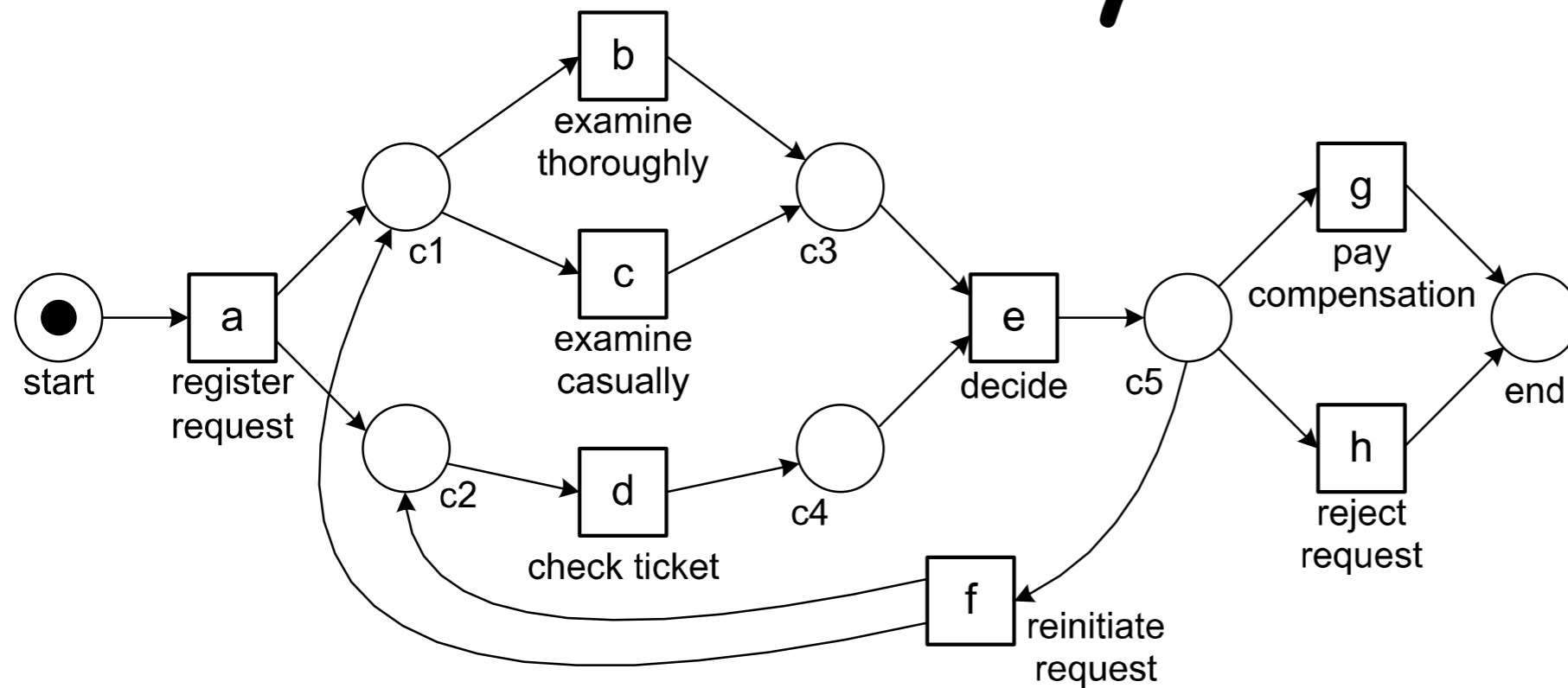| Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
...	...

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	$\langle a, c, d, e, g \rangle$
...	...

40

Discovery Example

Log features are adequately captured by the net

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	$\langle a, c, d, e, g \rangle$
…	…

41

Overfitting and Underfitting

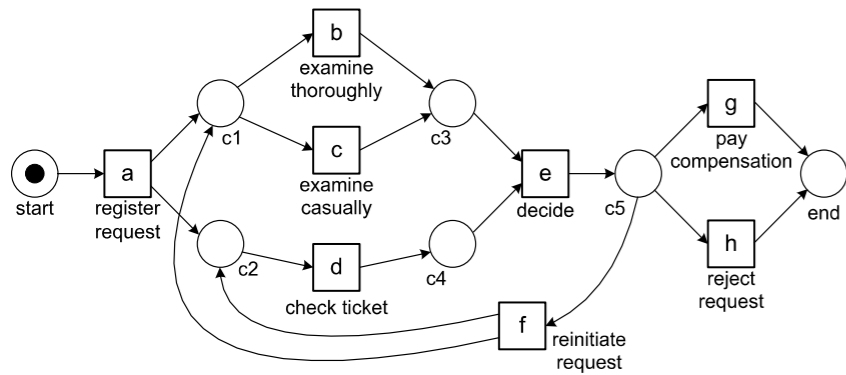One of the challenges of process mining is
to balance between

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

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	$\langle a, c, d, e, g \rangle$
…	…

Discussion

The discovered net also allows for traces not in the log, e.g.
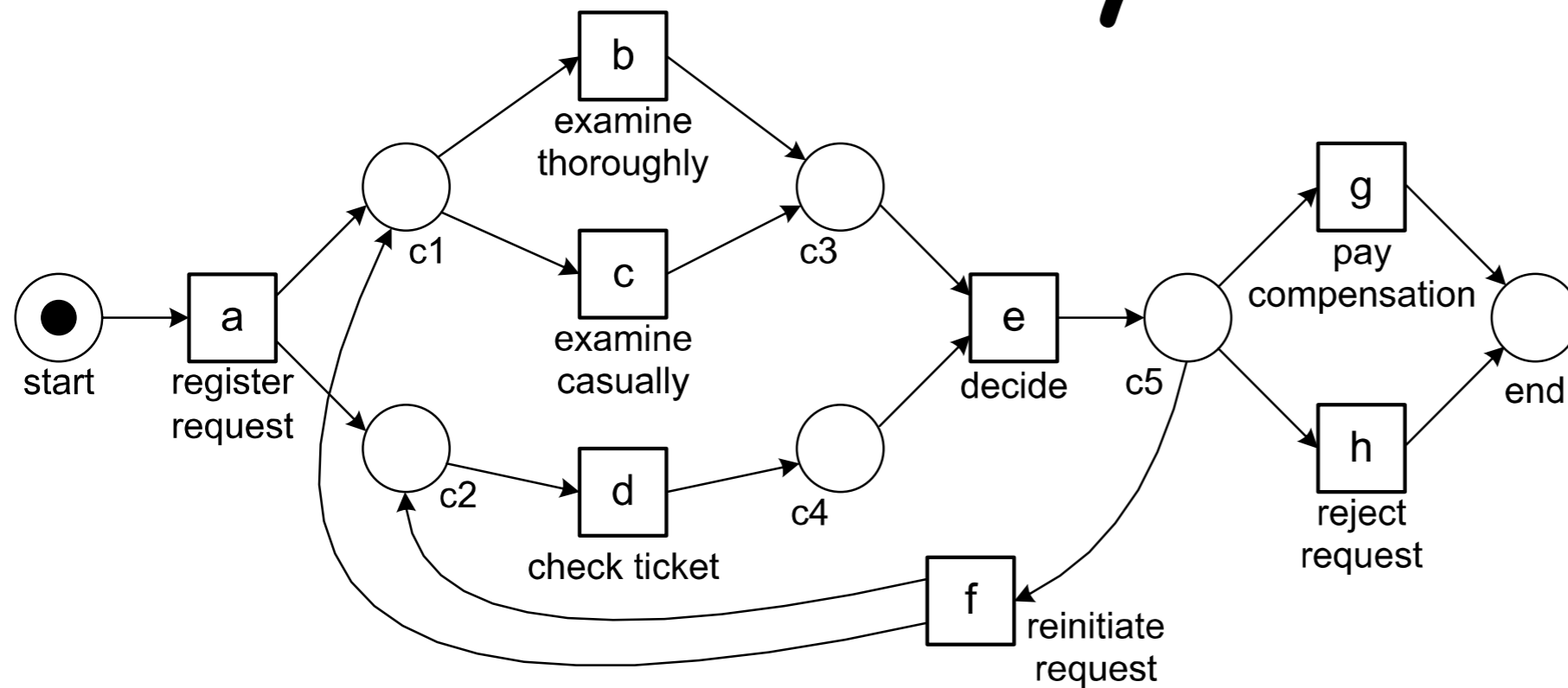
$\langle$ a, d, c, e, f, b, d, e, g $\rangle$

$\langle$ a, c, d, e, f, c, d, e, f, c, d, e, f, b, d, e, g $\rangle$

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
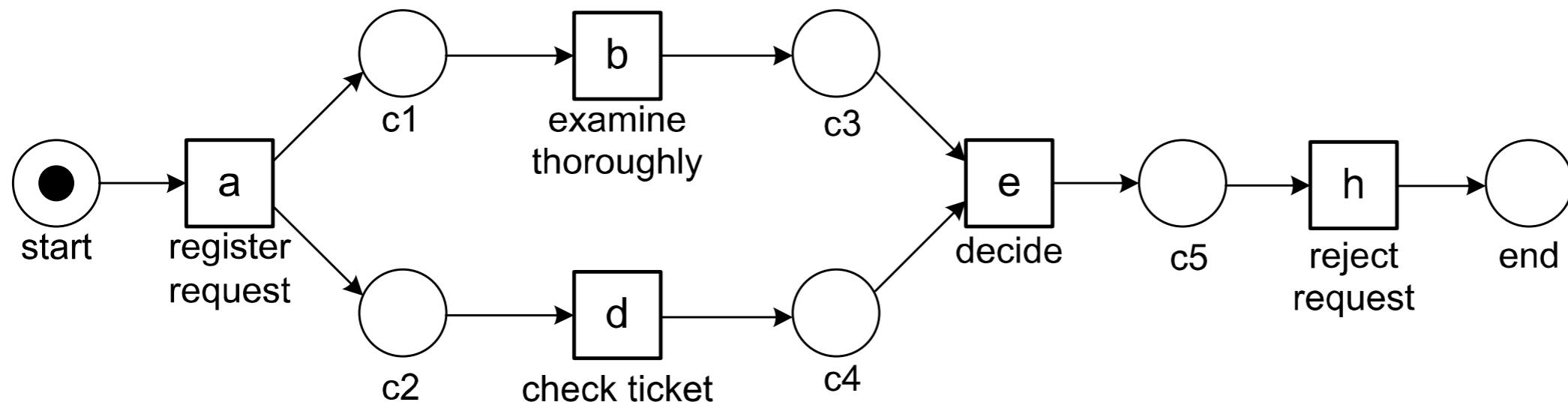
Discovery Example

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	$\langle a, c, d, e, g \rangle$
…	…

44

Another Discovery Example

Another net could disallow some traces

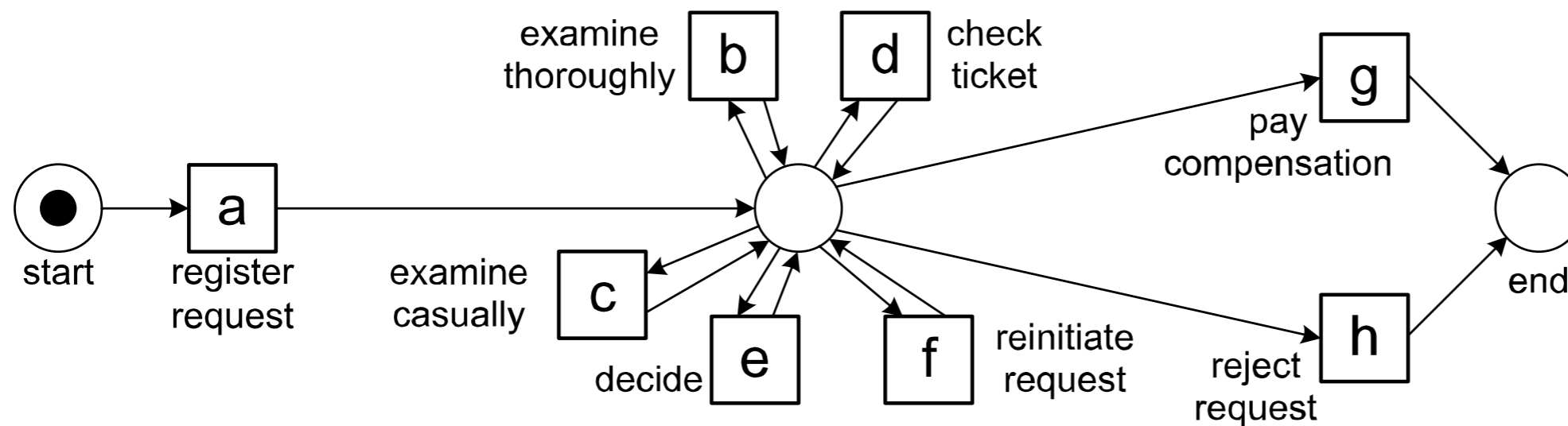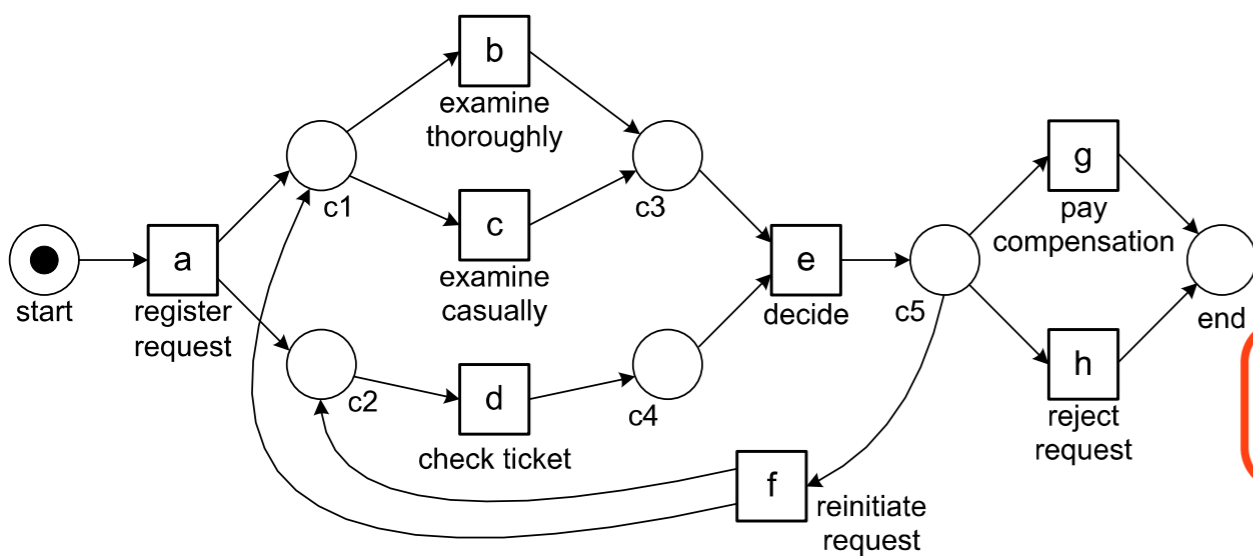| Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
…	…

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	$\langle a, c, d, e, g \rangle$
…	…

46

Conformance Example

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

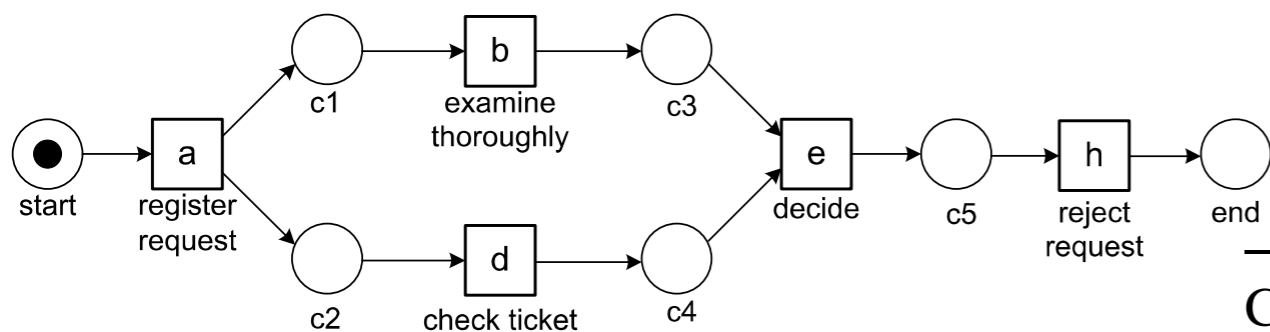| Case id	Trace
1	$\langle a, b, d, e, h \rangle$
2	$\langle a, d, c, e, g \rangle$
3	$\langle a, c, d, e, f, b, d, e, g \rangle$
4	$\langle a, d, b, e, h \rangle$
5	$\langle a, c, d, e, f, d, c, e, f, c, d, e, h \rangle$
6	$\langle a, c, d, e, g \rangle$
7	$\langle \mathbf{a}, \mathbf{b}, \mathbf{e},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

b — examine thoroughly
c1
c — examine casually
c3
a — register request
start
e — decide
c5
g — pay compensation
c2
d — check ticket
c4
h — reject request
f — reinitiate request
end

7 ok out of 10

47

Conformance Example

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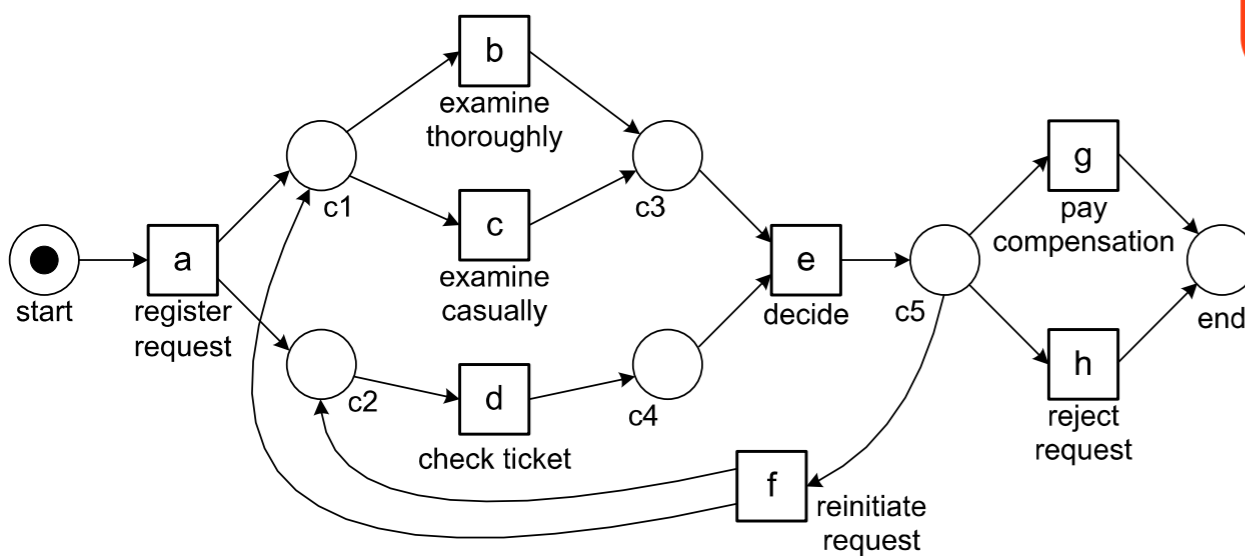(how well they pair together)

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	$\langle \mathbf{a}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}, \mathbf{b}, \mathbf{d}, \mathbf{g} \rangle$

48

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.
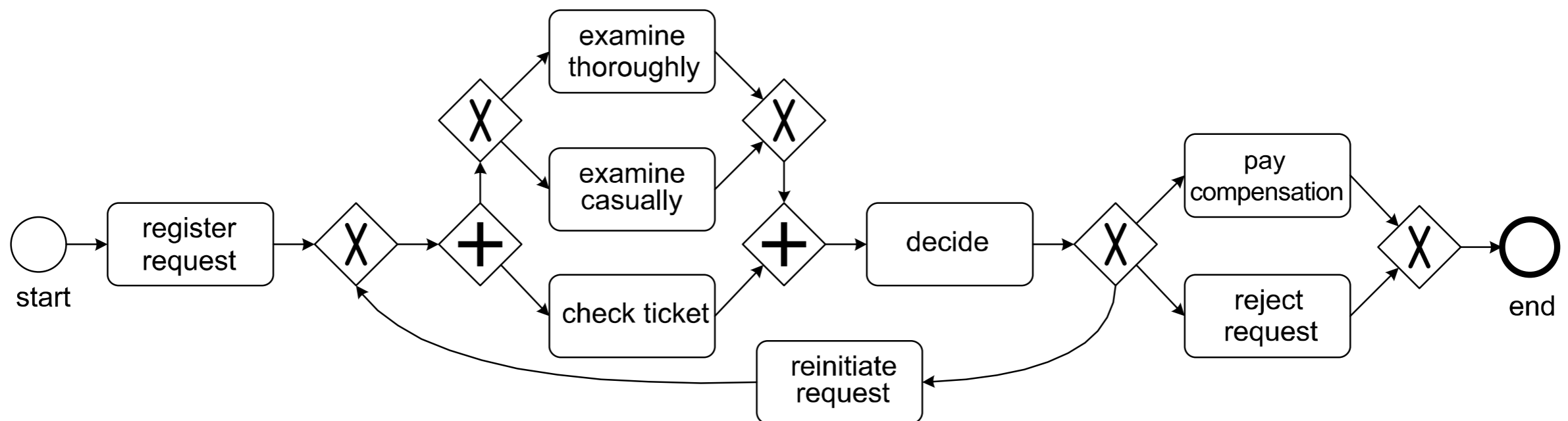
Process Discovery: α-Algorithm

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).

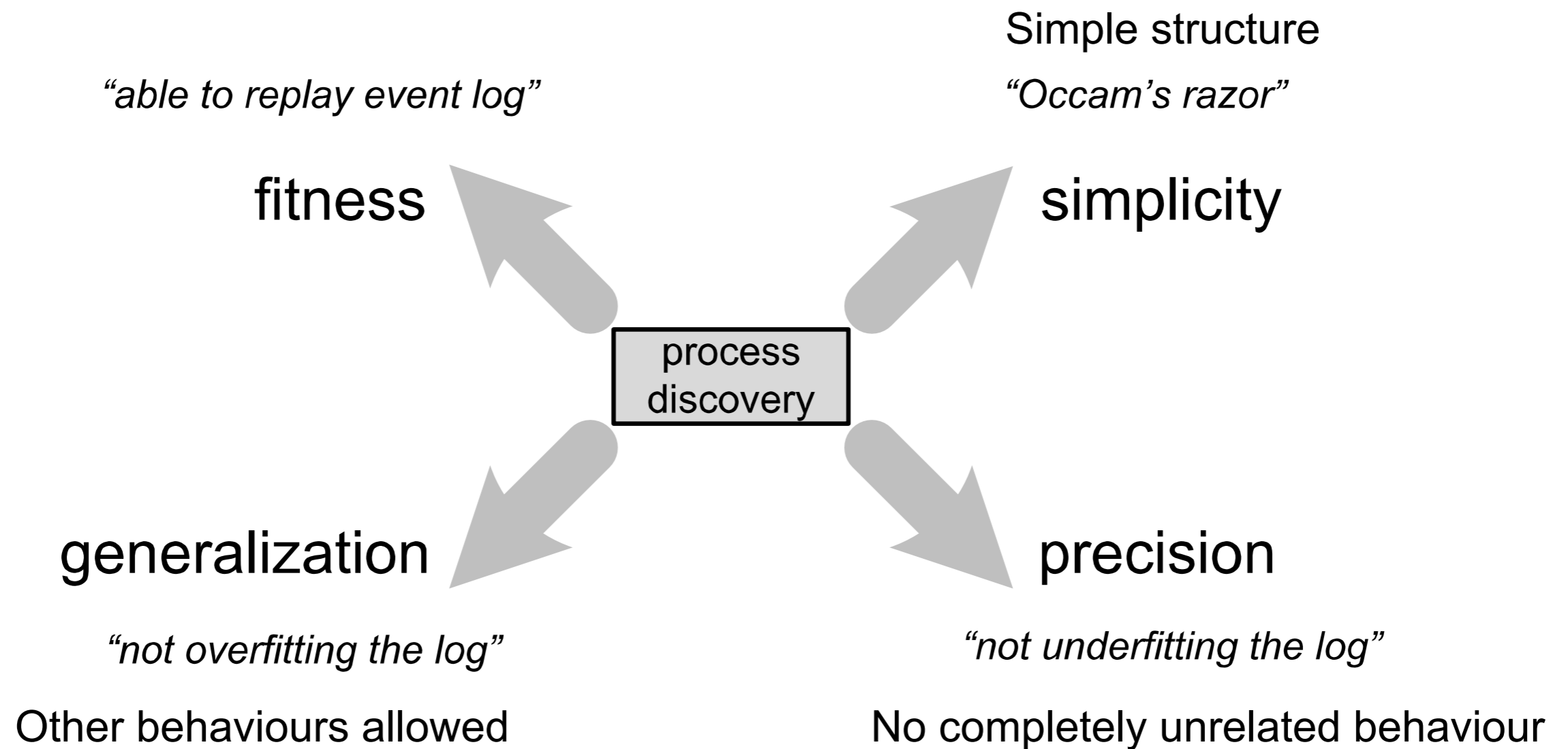# Simple Event Log

Let A be a set of activities.

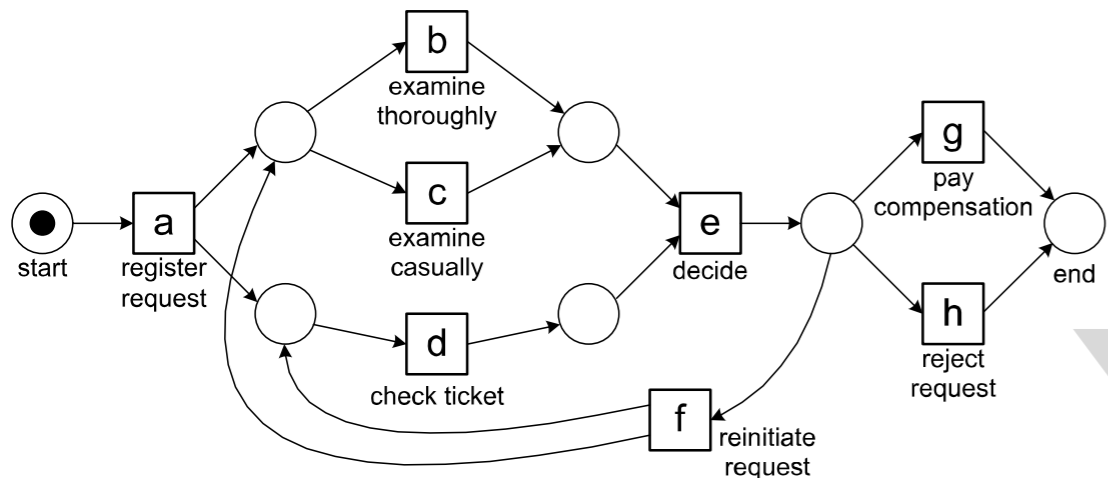A **simple trace** over A is a finite sequence of activities.

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

$$L_1 = \left[\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle \right]$$
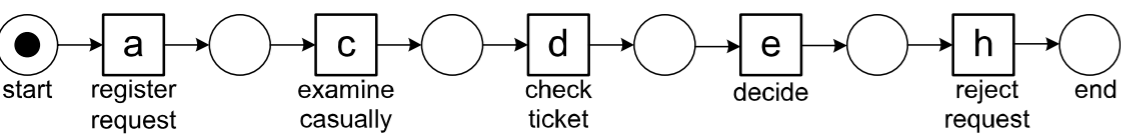
$$L_2 = \big[\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^4, \langle a, b, c, e, f, b, c, d \rangle^2, \langle a, b, c, e, f, c, b, d \rangle,$$
$$\langle a, c, b, e, f, b, c, d \rangle^2, \langle a, c, b, e, f, b, c, e, f, c, b, d \rangle \big]$$
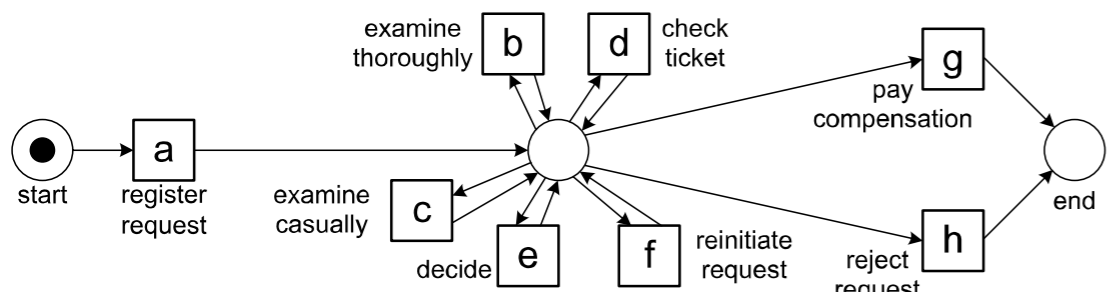
Quality Criteria

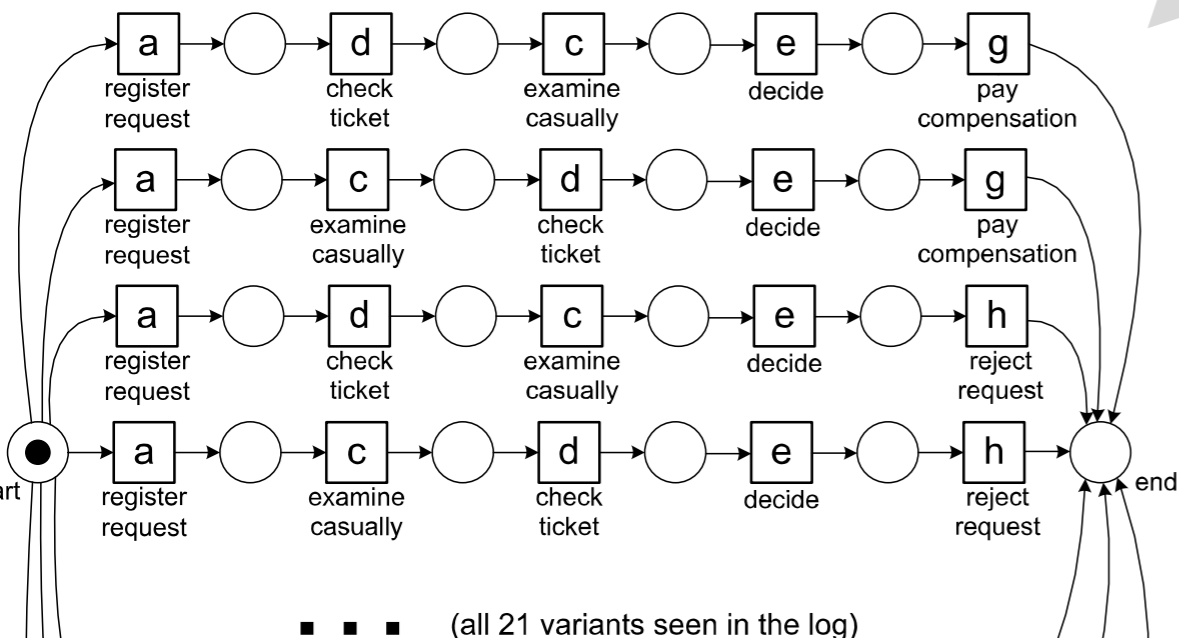*"able to replay event log"*

Simple structure

"Occam's razor"

fitness

simplicity

process discovery

generalization

precision

"not overfitting the log"

"not underfitting the log"

Other behaviours allowed

No completely unrelated behaviour

N₁ : *fitness = +, precision = +, generalization = +, simplicity = +*

Process model N₁ labels: start, a register request, b examine thoroughly, c examine casually, d check ticket, e decide, f reinitiate request, g pay compensation, h reject request, end

N₂ : *fitness = -, precision = +, generalization = -, simplicity = +*

Process model N₂ labels: start, a register request, c examine casually, d check ticket, e decide, h reject request, end

N₃ : *fitness = +, precision = -, generalization = +, simplicity = +*

Process model N₃ labels: start, a register request, b examine thoroughly, d check ticket, c examine casually, e decide, f reinitiate request, g pay compensation, h reject request, end

N₄ : *fitness = +, precision = +, generalization = -, simplicity = -*

Process model N₄ labels: a register request, d check ticket, c examine casually, e decide, g pay compensation (and variants); start, end; (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 — *"able to replay event log"*

simplicity — *"Occam's razor"*

process discovery

generalization — *"not overfitting the log"*

precision — *"not underfitting the log"*

54
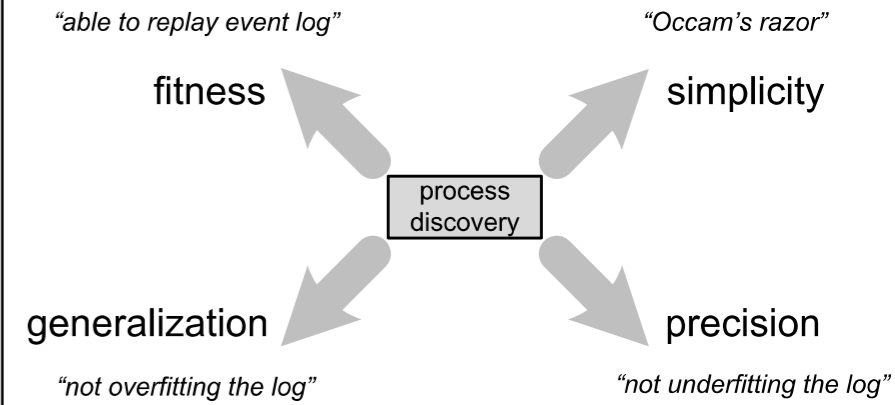
Quality Measures

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

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)

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

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

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

- $a \to_L b$ 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 $b >_L a$

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

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

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

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

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

Log-based Ordering Relations: Example

- $a \to_L b$ if and only if $a >_L b$ and $b \ngtr_L a$
- $a \#_L b$ if and only if $a \ngtr_L b$ and $b \ngtr_L a$
- $a \parallel_L b$ if and only if $a >_L b$ and $b >_L a$

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

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

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

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

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

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
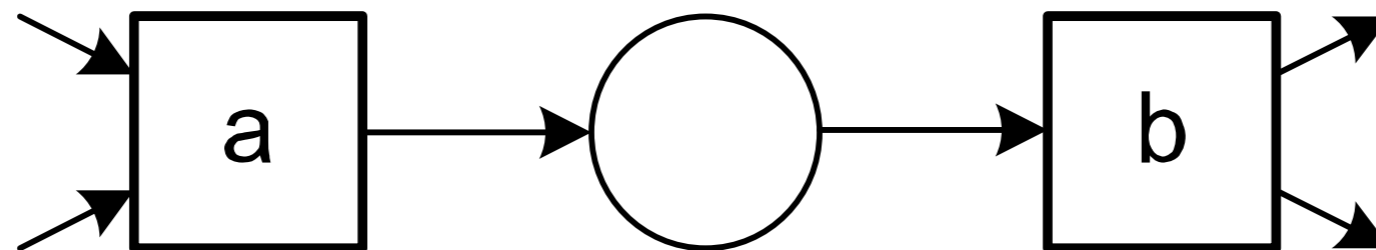
(a) sequence pattern: a→b

Patterns

Footprints are useful to discover typical patterns of activities in the corresponding process model
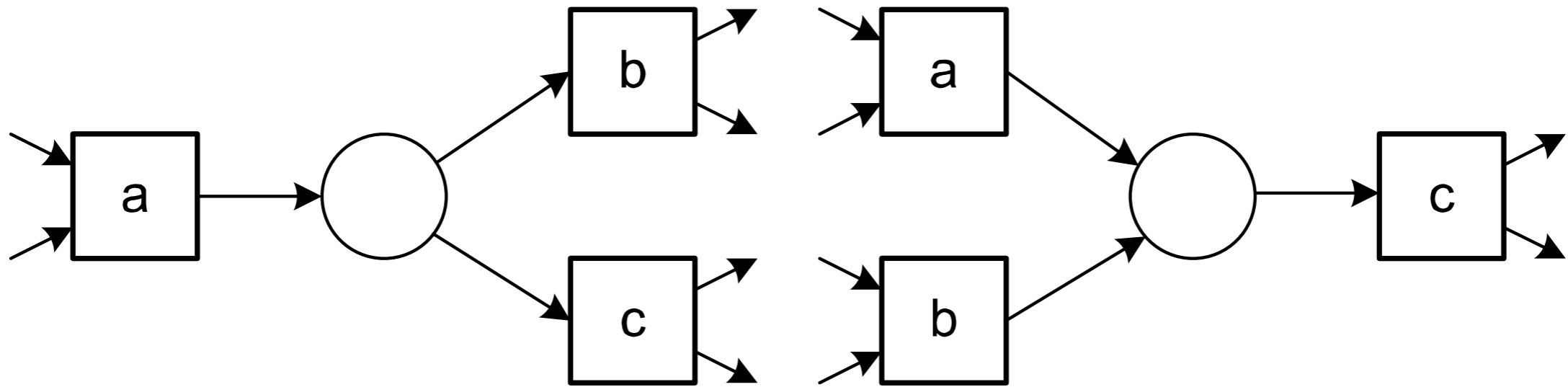
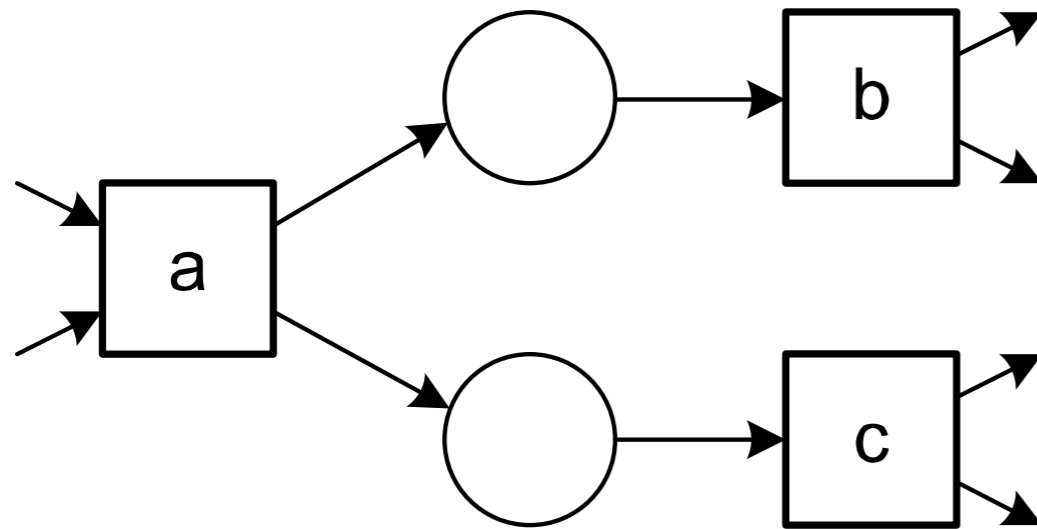(b) XOR-split pattern:
a→b, a→c, and b#c

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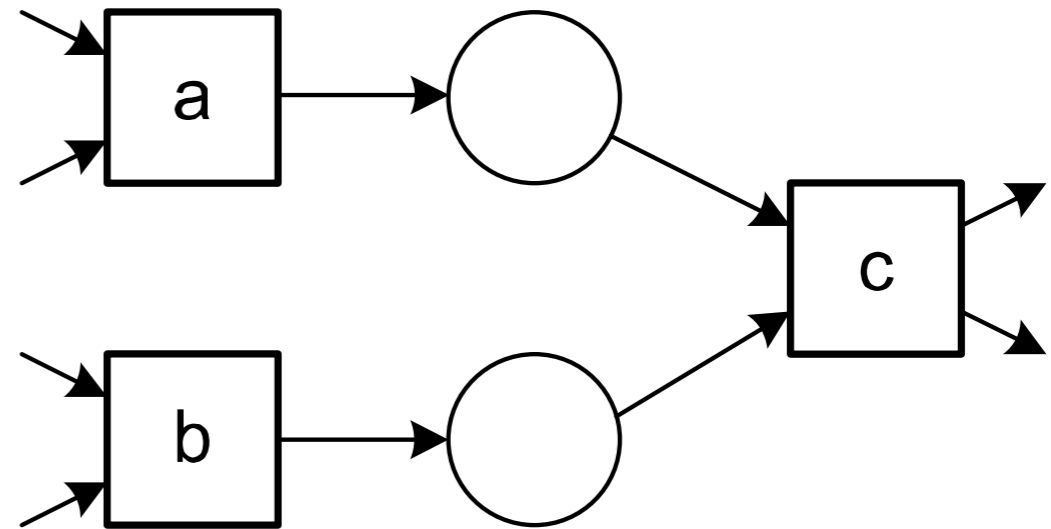a→c, b→c, and a#b

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

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

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

4. $X_L = \left\{ \; (A, B) \;\middle|\; \begin{array}{llll} A, B \subseteq T_L & \wedge & A, B \neq \emptyset & \wedge \\ \forall_{a \in A} \forall_{b \in B} & a \to_L b & & \wedge \\ \forall_{a_1, a_2 \in A} & a_1 \#_L a_2 & & \wedge \\ \forall_{b_1, b_2 \in B} & b_1 \#_L b_2 & & \end{array} \right\}$ 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. $T_L = \{ \ t \in T \ | \ \exists_{\sigma \in L} \ t \in \sigma \ \}$ transitions

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

3. $T_O = \{ \ t \in T \ | \ \exists_{\sigma \in L} \ t = last(\sigma) \ \}$ end events
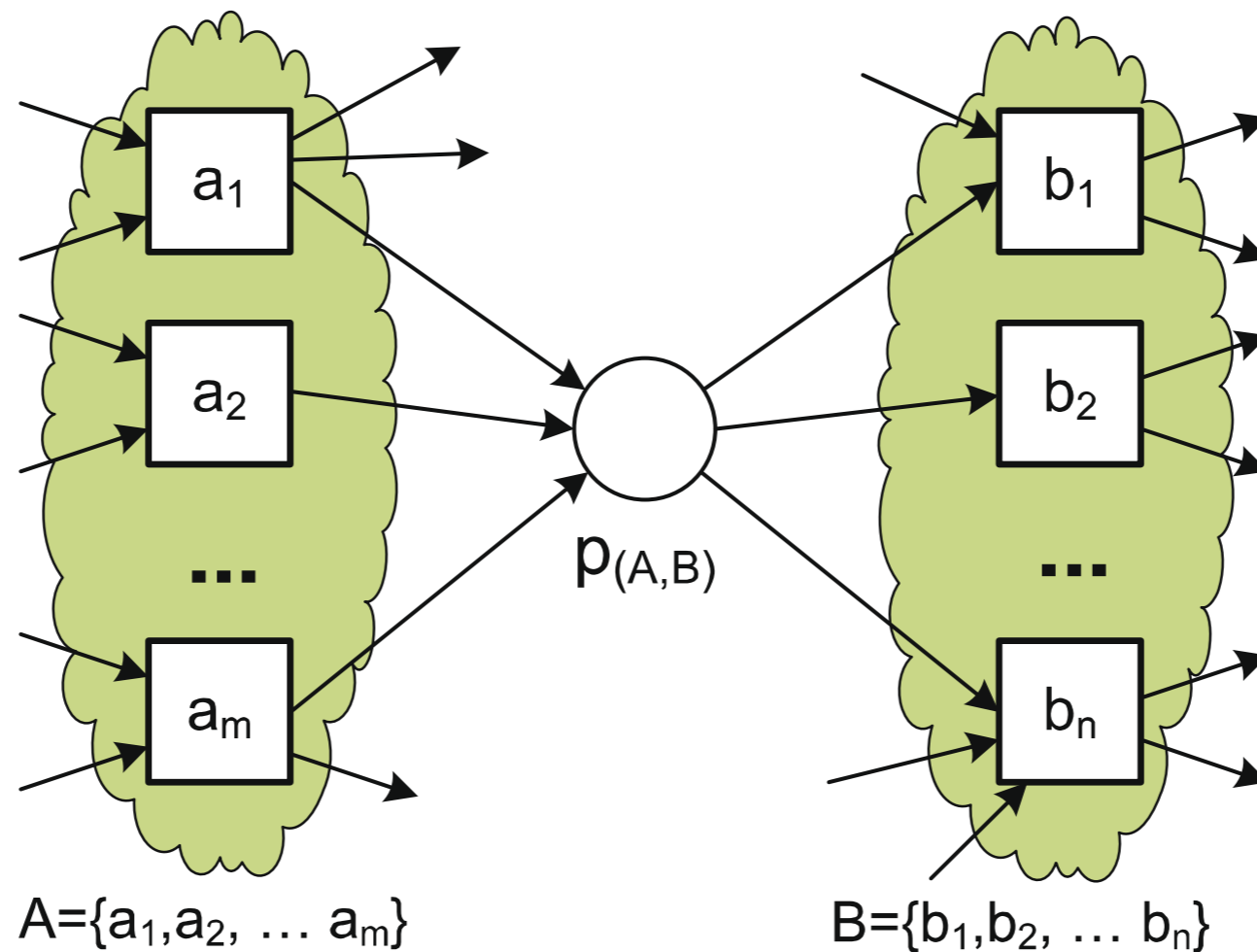
The alpha-Algorithm

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

The Core of the algorithm: Steps 4, 5

$$\forall_{a \in A} \forall_{b \in B} \quad a \rightarrow_L b$$
$$\forall_{a_1, a_2 \in A} \quad a_1 \#_L a_2$$
$$\forall_{b_1, b_2 \in B} \quad b_1 \#_L b_2$$

	a_1	a_2	$\ldots$	a_m	b_1	b_2	$\ldots$	b_n
a_1	#	#	$\ldots$	#	$\rightarrow$	$\rightarrow$	$\ldots$	$\rightarrow$
a_2	#	#	$\ldots$	#	$\rightarrow$	$\rightarrow$	$\ldots$	$\rightarrow$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
a_m	#	#	$\ldots$	#	$\rightarrow$	$\rightarrow$	$\ldots$	$\rightarrow$
b_1	$\leftarrow$	$\leftarrow$	$\ldots$	$\leftarrow$	#	#	$\ldots$	#
b_2	$\leftarrow$	$\leftarrow$	$\ldots$	$\leftarrow$	#	#	$\ldots$	#
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
b_n	$\leftarrow$	$\leftarrow$	$\ldots$	$\leftarrow$	#	#	$\ldots$	#

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

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

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

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	$\leftarrow_{L_1}$	$\#_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$

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

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

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	$\leftarrow_{L_1}$	$\#_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$

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

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

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	$\leftarrow_{L_1}$	$\#_{L_1}$	$\#_{L_1}$	$\rightarrow_{L_1}$	$\#_{L_1}$

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

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

The alpha-Algorithm

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

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

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

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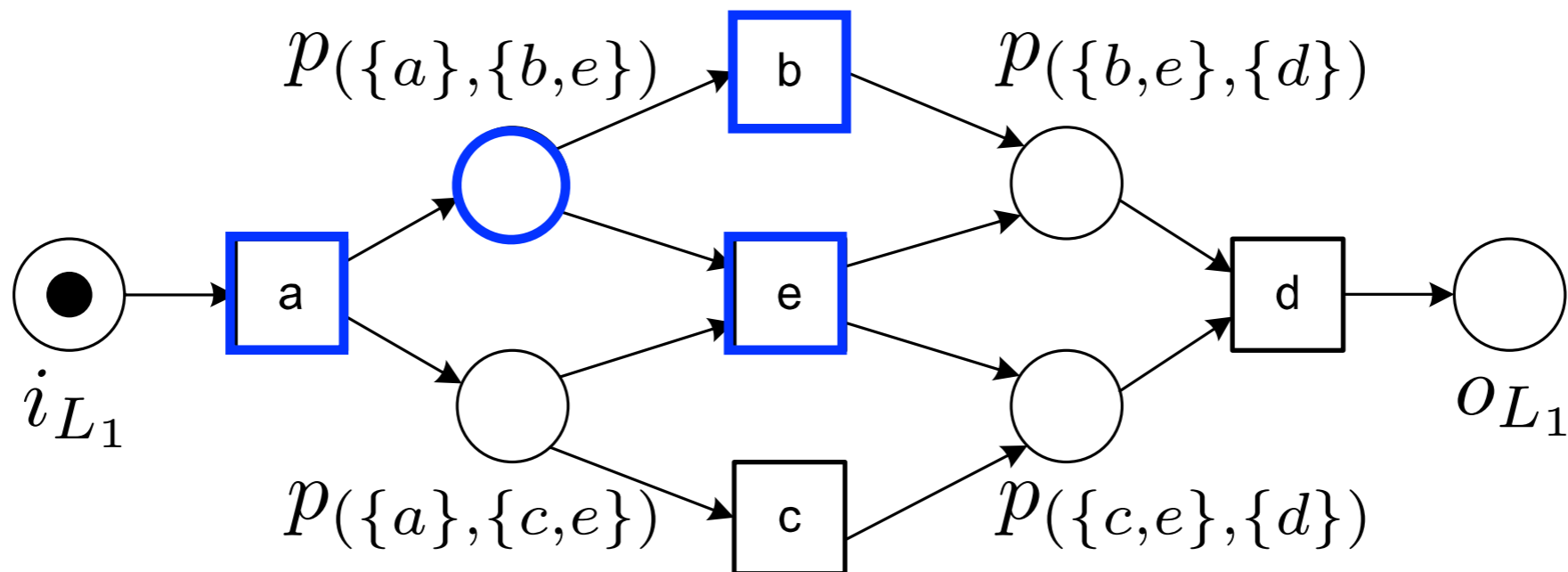
$$L_1 = [\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$$

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

The Algorithm: Example

$$L_1 = [\langle a, b, c, d \rangle^3, \langle a, c, b, d \rangle^2, \langle a, e, d \rangle]$$
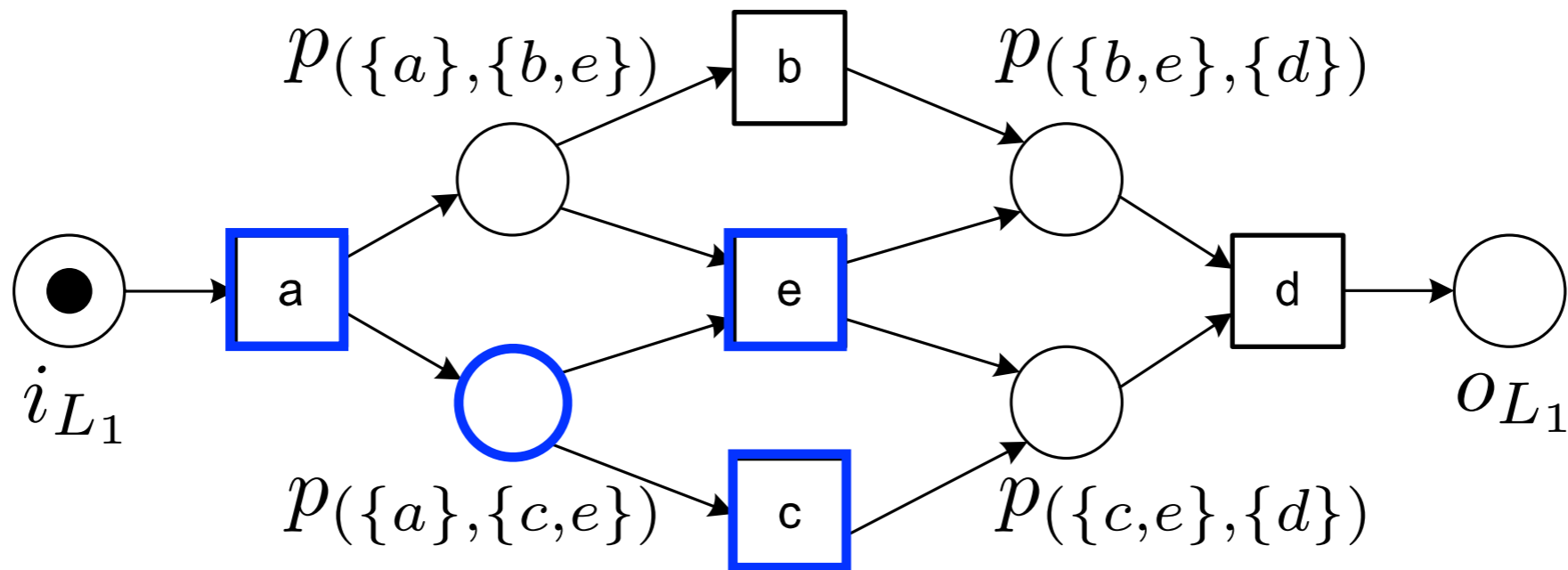
$$Y_{L_1} = \big\{ (\{a\}, \{b, e\}), (\{a\}, \{c, e\}), (\{b, e\}, \{d\}), (\{c, e\}, \{d\}) \big\}$$

The Algorithm: Example

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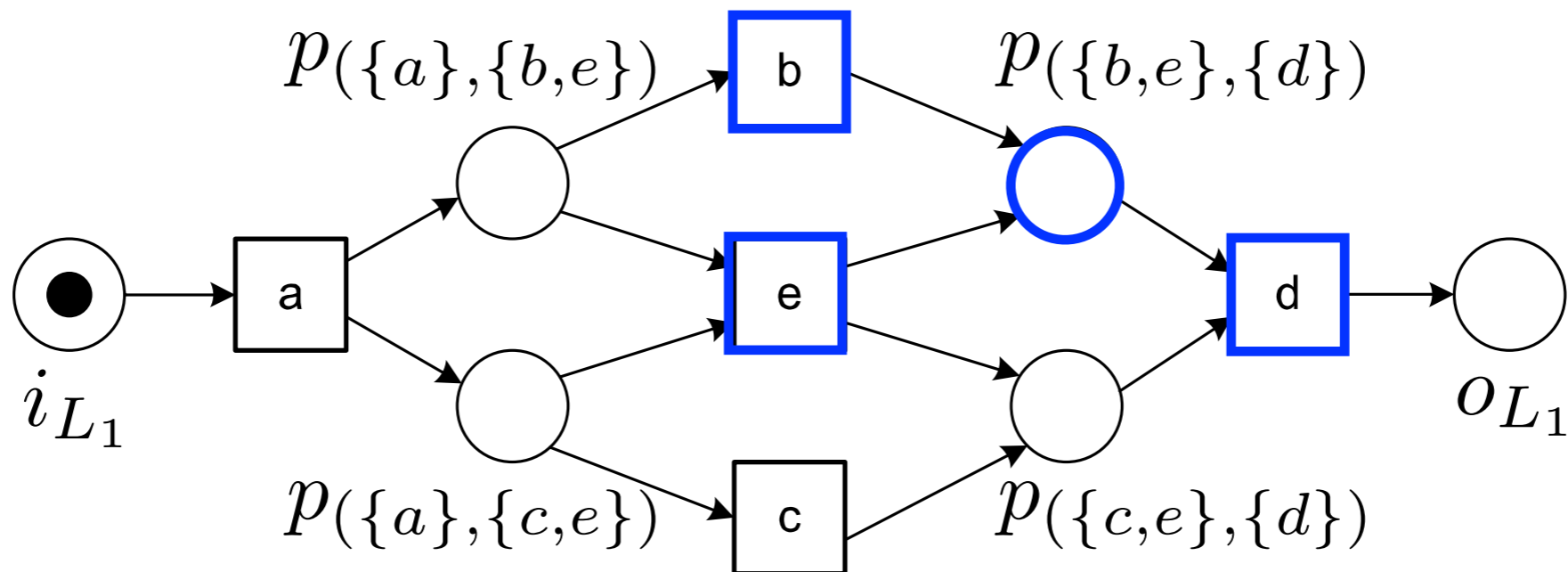
$$Y_{L_1} = \left\{ (\{a\}, \{b, e\}), (\{a\}, \{c, e\}), \boxed{(\{b, e\}, \{d\})}, (\{c, e\}, \{d\}) \right\}$$

The Algorithm: Example

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

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

Another Example

$$L_5 = \left[\langle a, b, e, f \rangle^2, \langle a, b, e, c, d, b, f \rangle^3, \langle a, b, c, e, d, b, f \rangle^2, \right.$$

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

	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	#	$\leftarrow$	#	#	$\leftarrow$	#

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

$T_I = \{a\}$

$T_I = \{f\}$

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

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

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

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

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

86
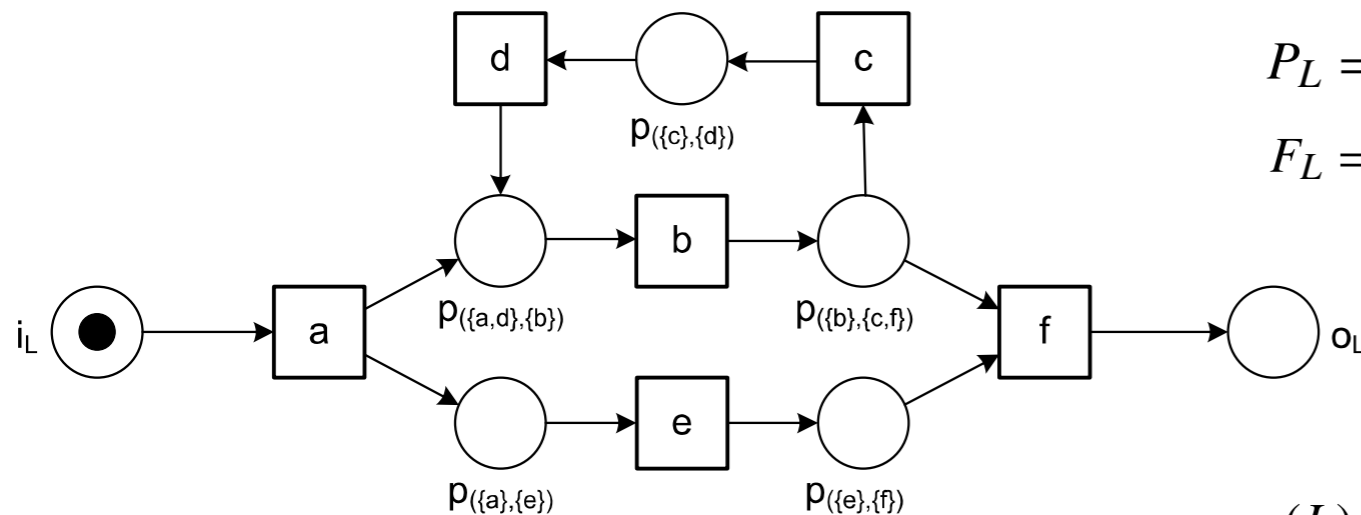
Exercises

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

	a	b	c	d	e
a	#	#	→	#	#
b	#	#	→	#	#
c	←	←	#	→	→
d	#	#	←	#	#
e	#	#	←	#	#

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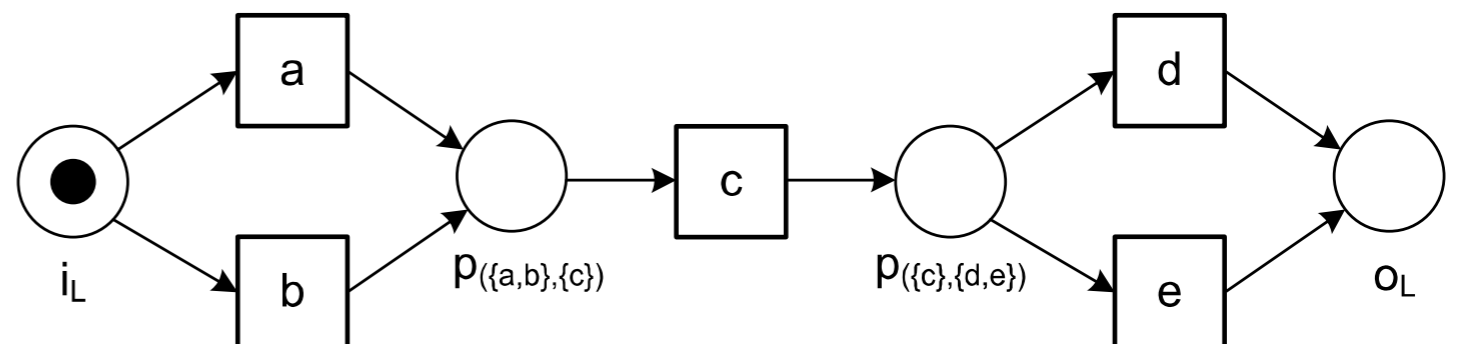
Exercises

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

$$\langle a,b,c,d,e,f,b,c,d,e,f,b,d,c,e,g\rangle\big]$$

	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

f

c

b

$p_{(\{b\},\{c\})}$

$p_{(\{c\},\{e\})}$

e

g

a

i_L

$p_{(\{a,f\},\{b\})}$

d

$p_{(\{b\},\{d\})}$

$p_{(\{d\},\{e\})}$

$p_{(\{e\},\{f,g\})}$

o_L

88
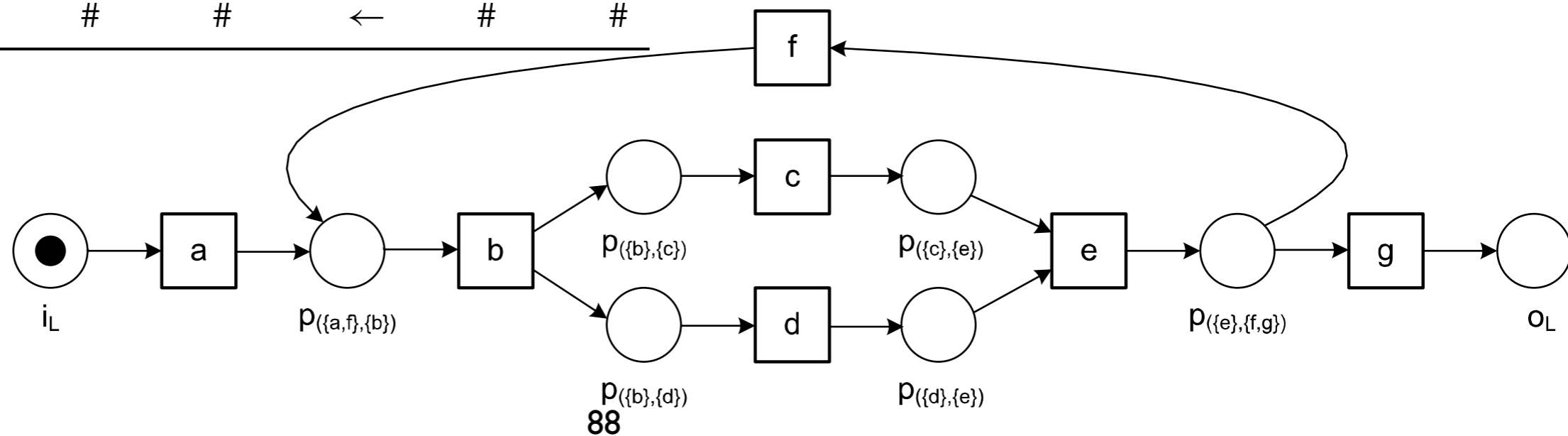
Exercises

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

$$\langle a, c, b, e, f, b, c, d\rangle^2, \langle a, c, b, e, f, b, c, e, f, c, b, d\rangle\big]$$

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

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

89
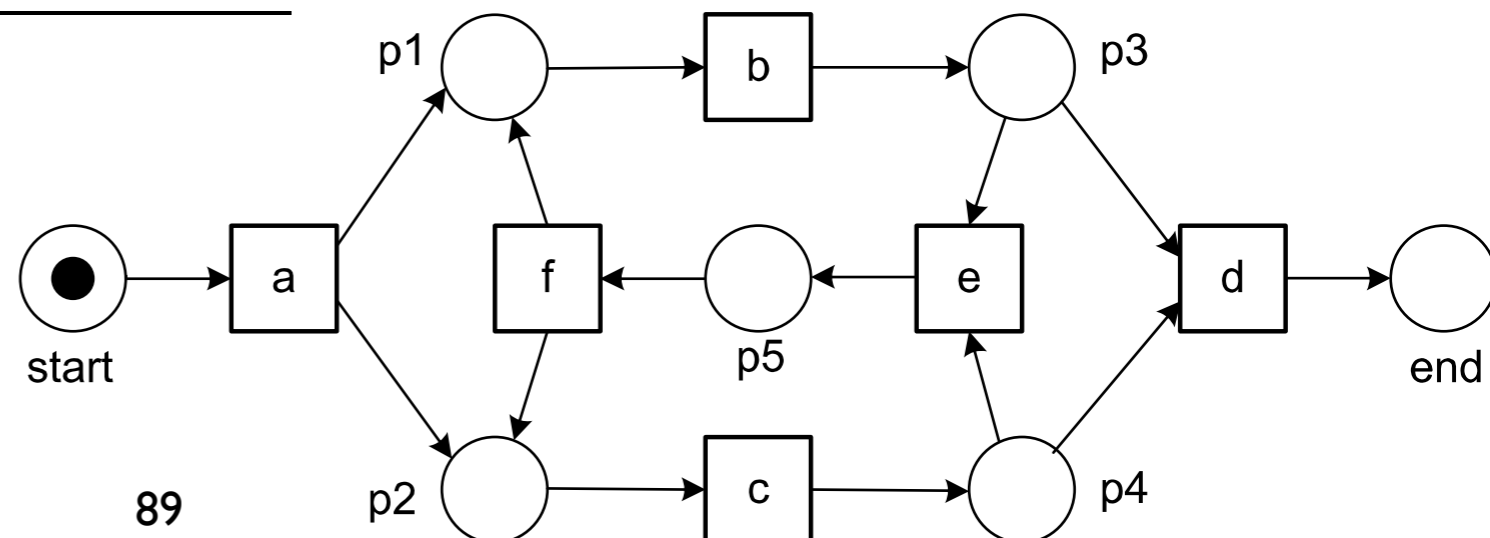
Limitation: Implicit Places

$$L_6 = \left[\langle a, c, e, g \rangle^2, \langle a, e, c, g \rangle^3, \langle b, d, f, g \rangle^2, \langle b, f, d, g \rangle^4 \right]$$
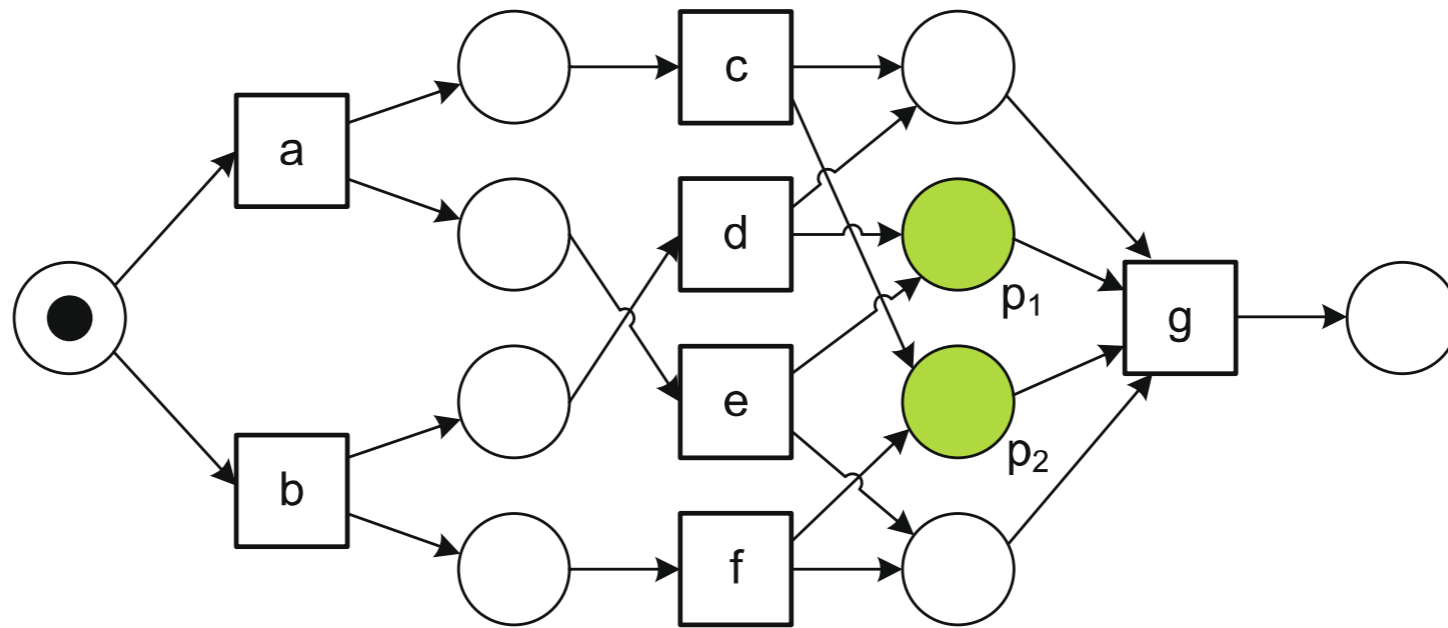
p$_1$ and p$_2$ are redundant

Limitation: Short Loop

$$L_7 = \left[\langle a, c \rangle^2, \langle a, b, c \rangle^3, \langle a, b, b, c \rangle^2, \langle a, b, b, b, b, c \rangle^1 \right]$$
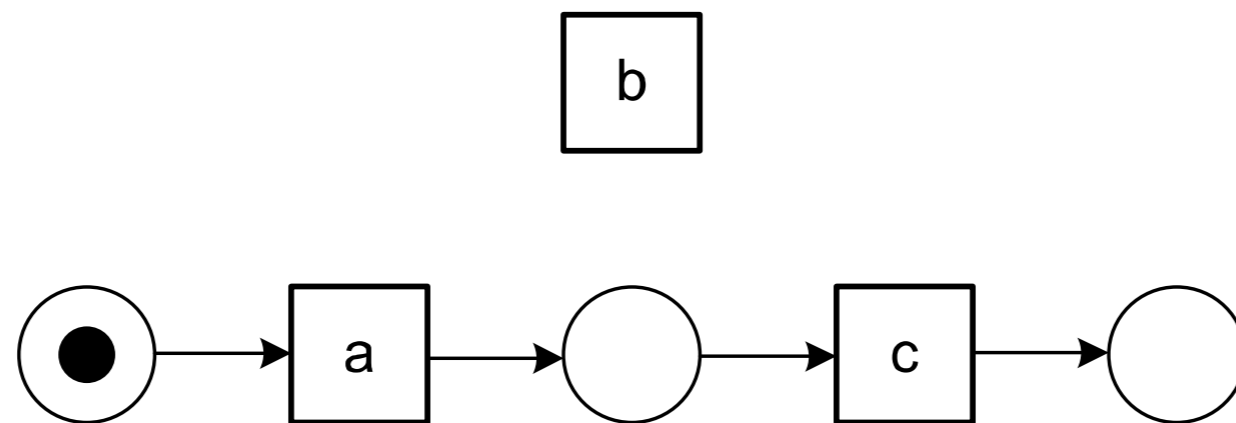
b is disconnected from the model
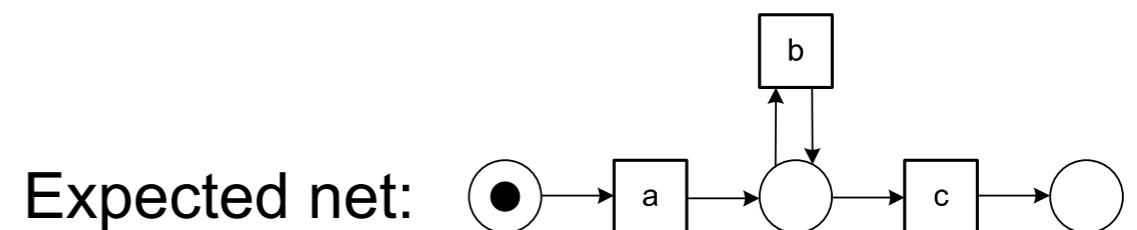
Expected net:

Limitation: Short Loop

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

$a \rightarrow_{L_8} b,$

$b \rightarrow_{L_8} d,$

$b \parallel_{L_8} c.$
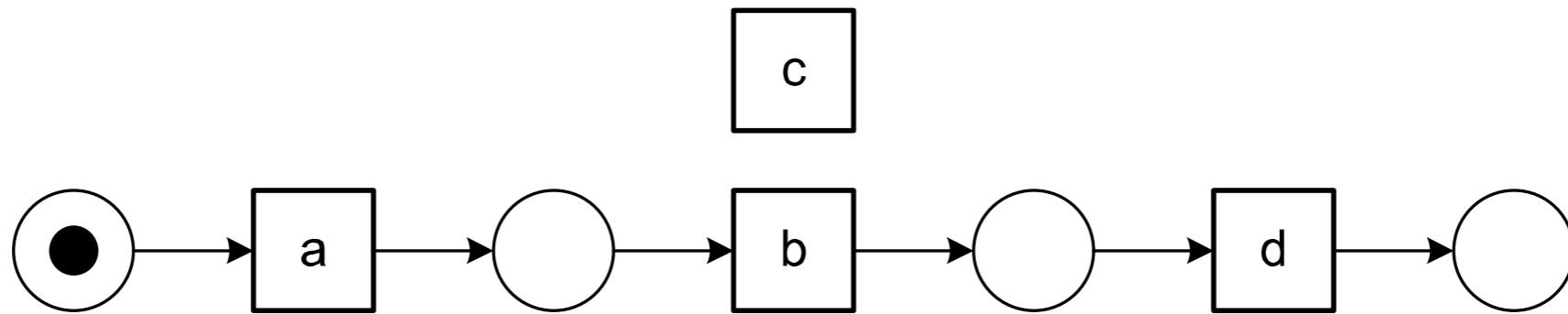
c is disconnected from the model

Expected net:
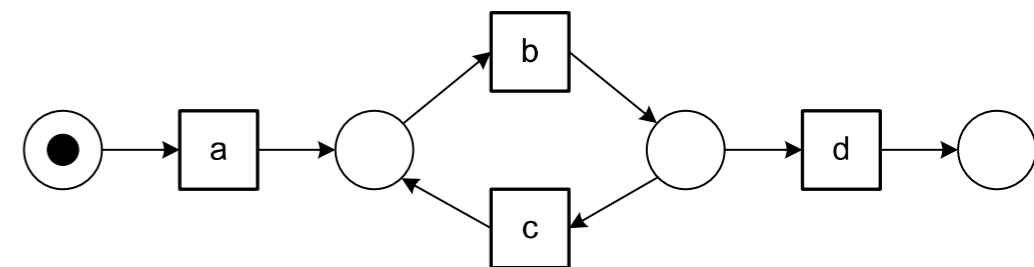
Limitation: Noise

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?).
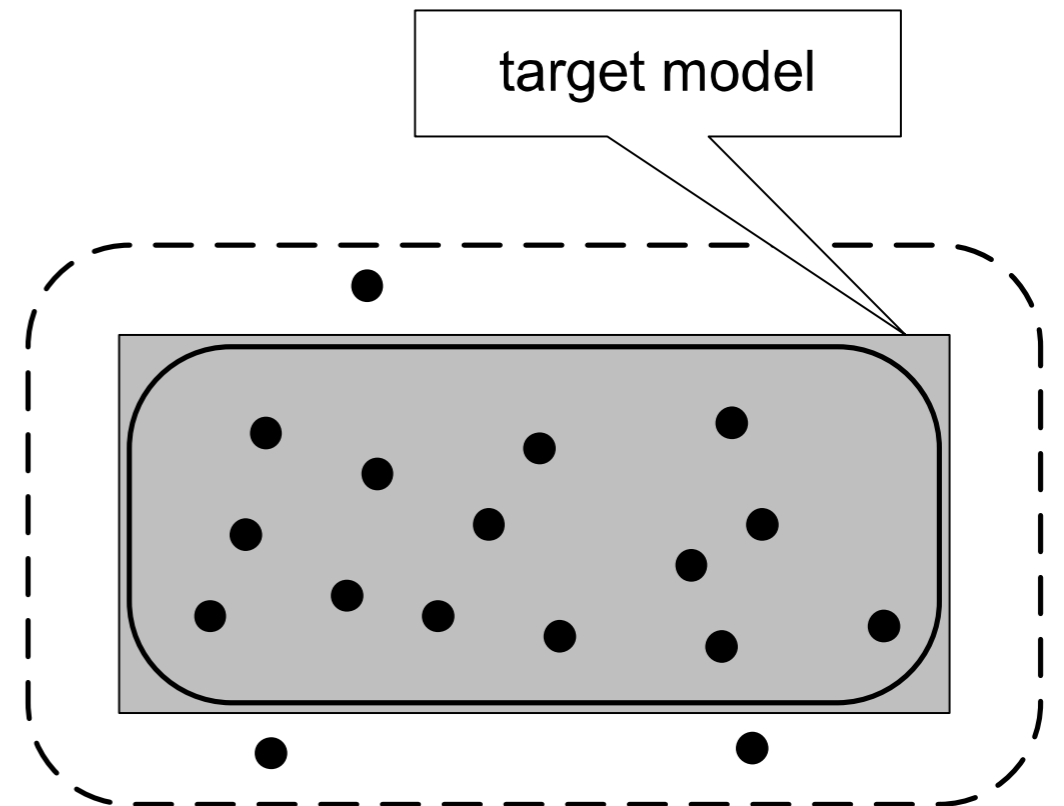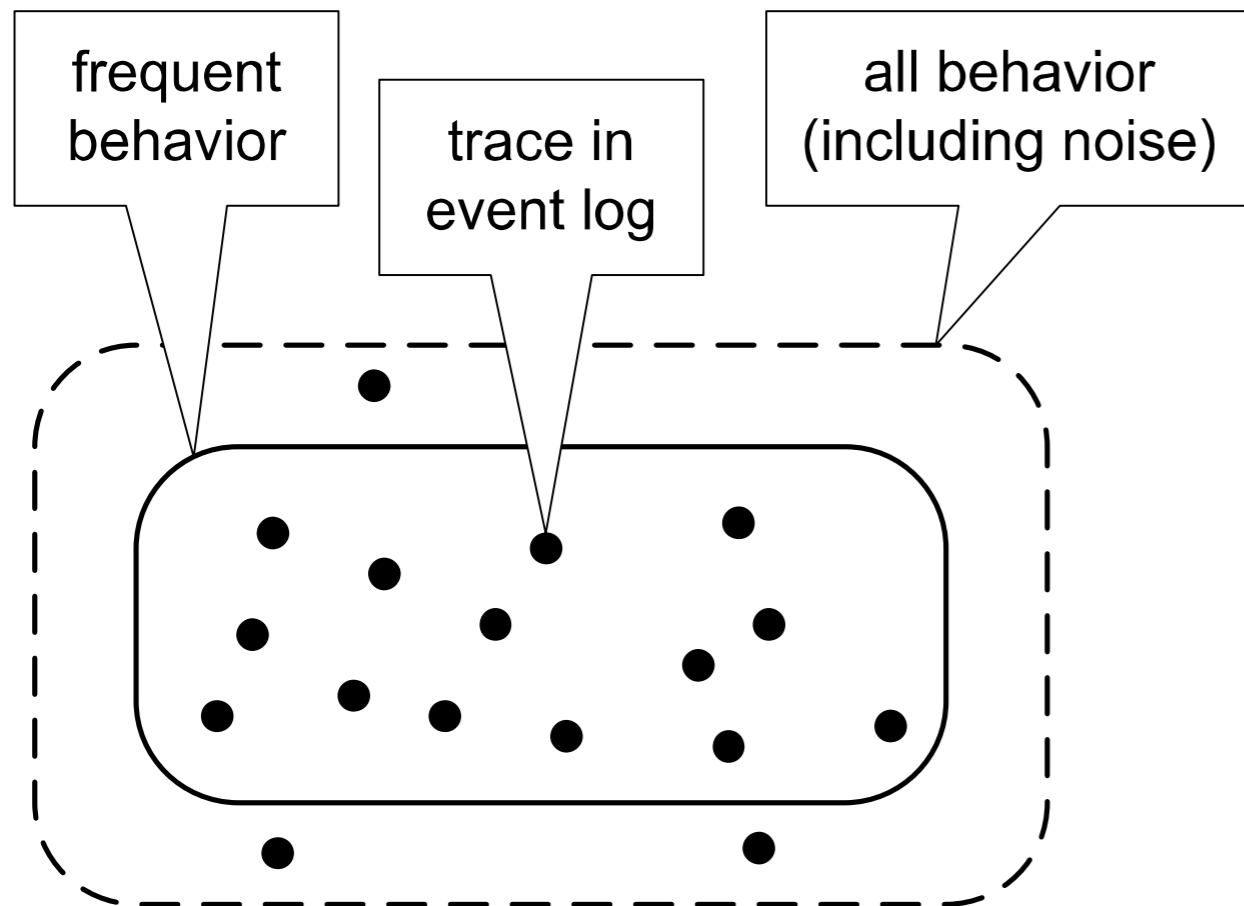
Limitation: Noise

Limitation: Noise

non-fitting model

overfitting model

underfitting model
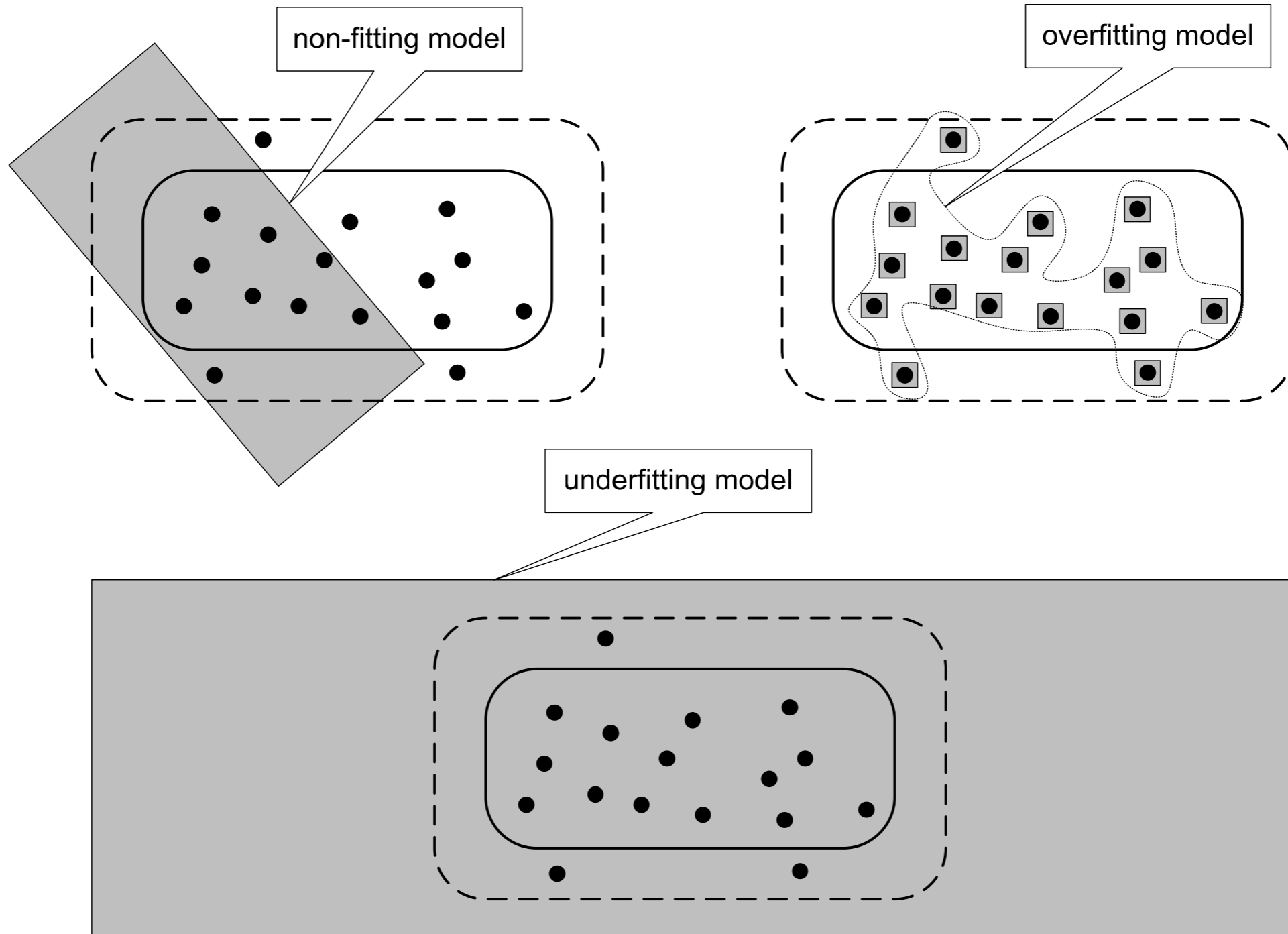
Limitation: Incompleteness

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
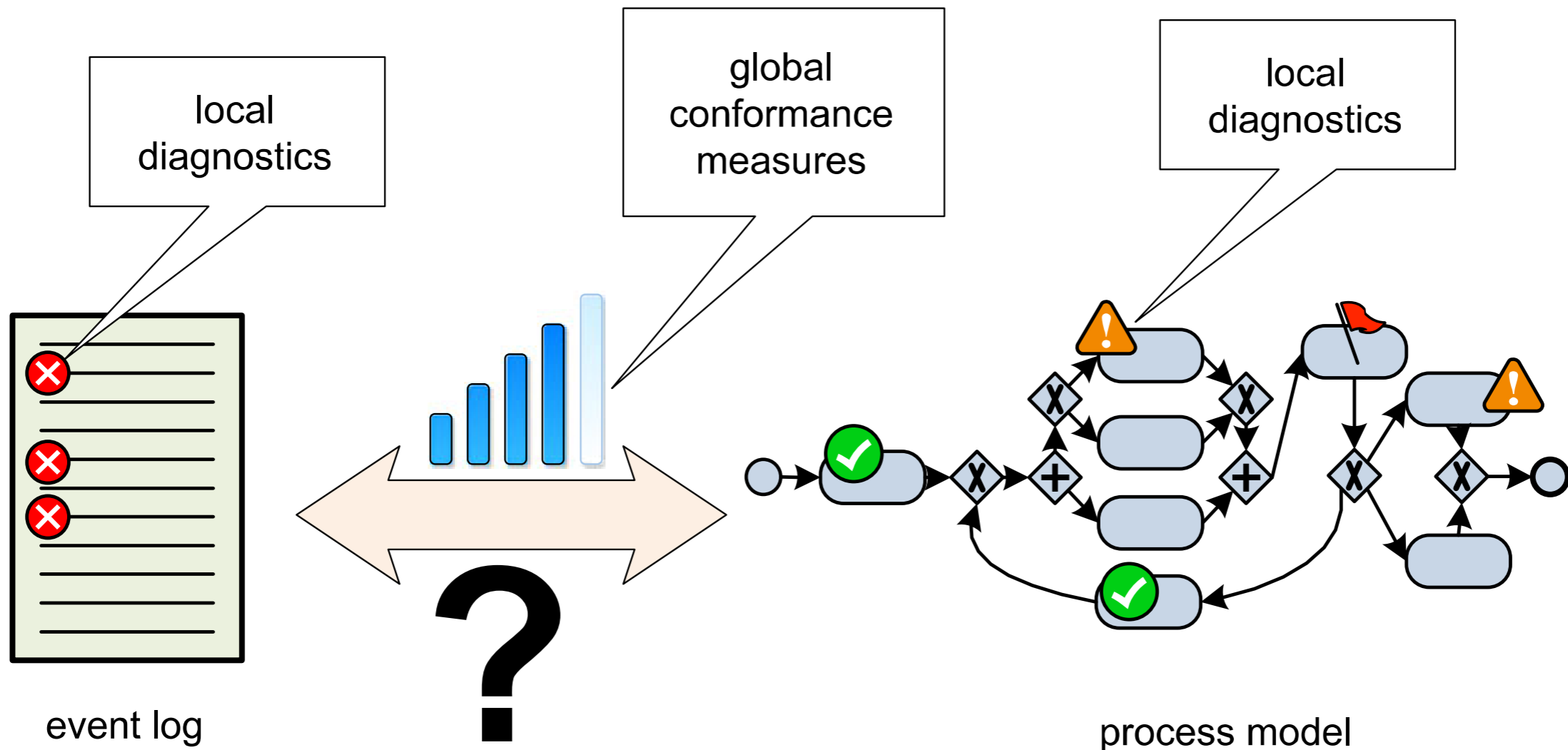
Fig. 7.1 Conformance checking: comparing observed behavior with modeled behavior. Global conformance measures quantify the overall conformance of the model and log. Local diagnostics are given by highlighting the nodes in the model where model and log disagree. Cases that do not fit are highlighted in the visualization of the log   99

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

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

N_1

b — examine thoroughly

c — examine casually

a — register request

d — check ticket

e — decide

g — pay compensation

h — reject request

f — reinitiate request

start

end

p1, p2, p3, p4, p5
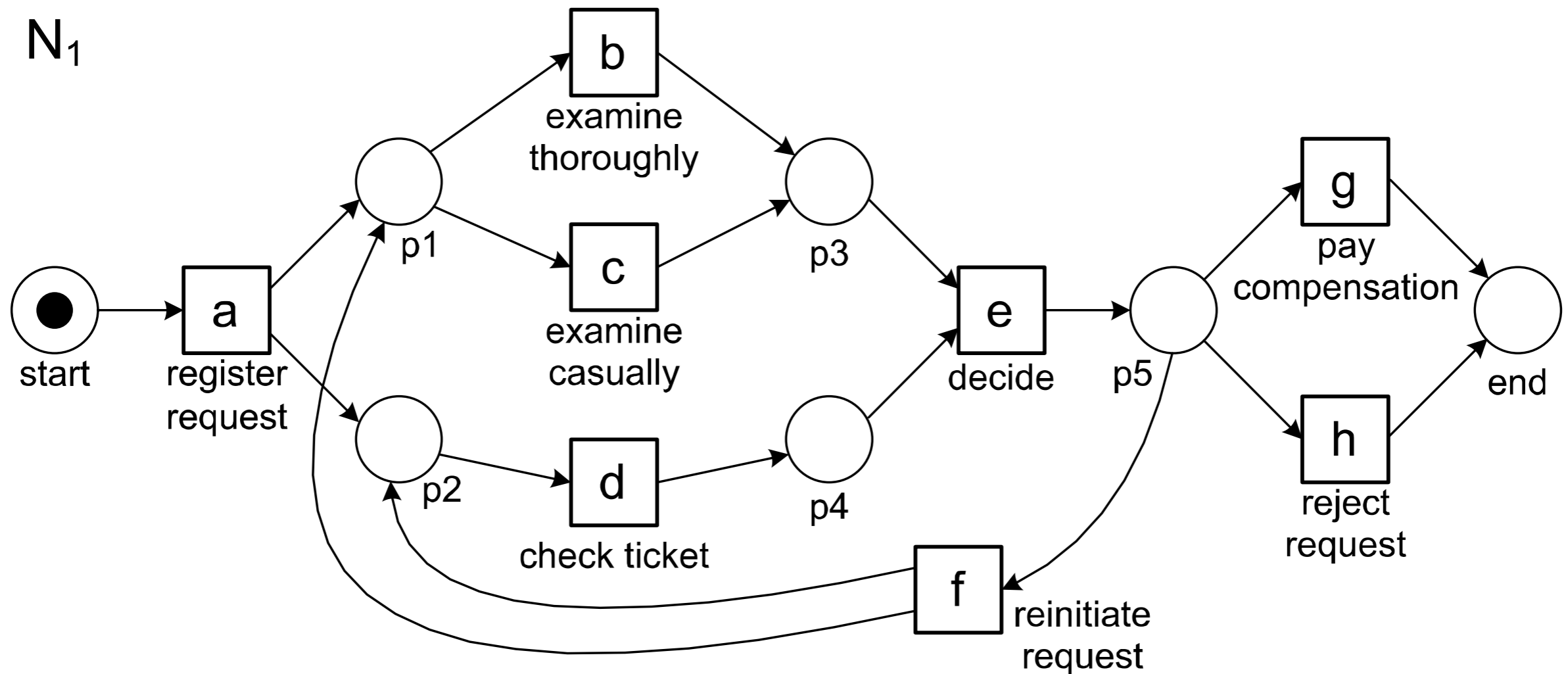
naïve fitness $\frac{1391}{1391} = 1$ The net can ``replay'' any trace

Example N2

N_2

b
examine
thoroughly

start · register request · p1 · c examine casually · p2 · d check ticket · p3 · e decide · p4

pay compensation
g

h
reject request

a register request

f
reinitiate request

end
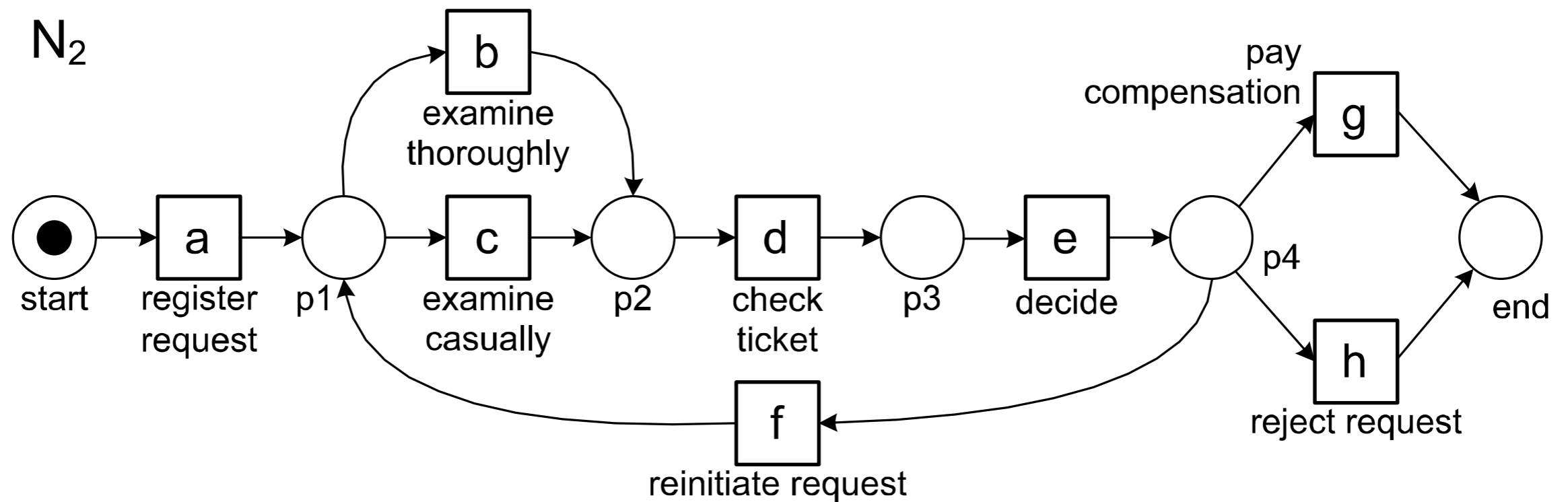
443 cases do not correspond to a firing sequence

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

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

Example N3

N_3

start — register request (a) — p1 — examine casually (c) — p3 — decide (e) — p5 — reject request (h) — end

p2 — check ticket (d) — p4
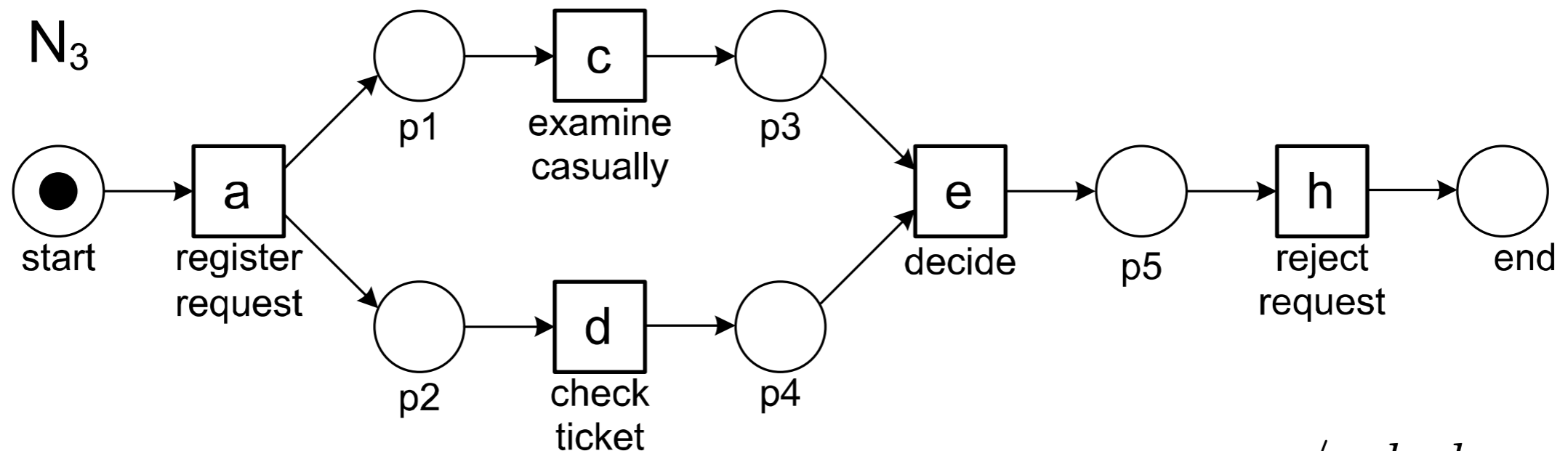
759 cases do not correspond to a firing sequence

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

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

Example N4

N_4

examine thoroughly — b

d — check ticket

start — a — register request

examine casually — c

decide — e

f — reinitiate request

p1

g — pay compensation

h — reject request

end
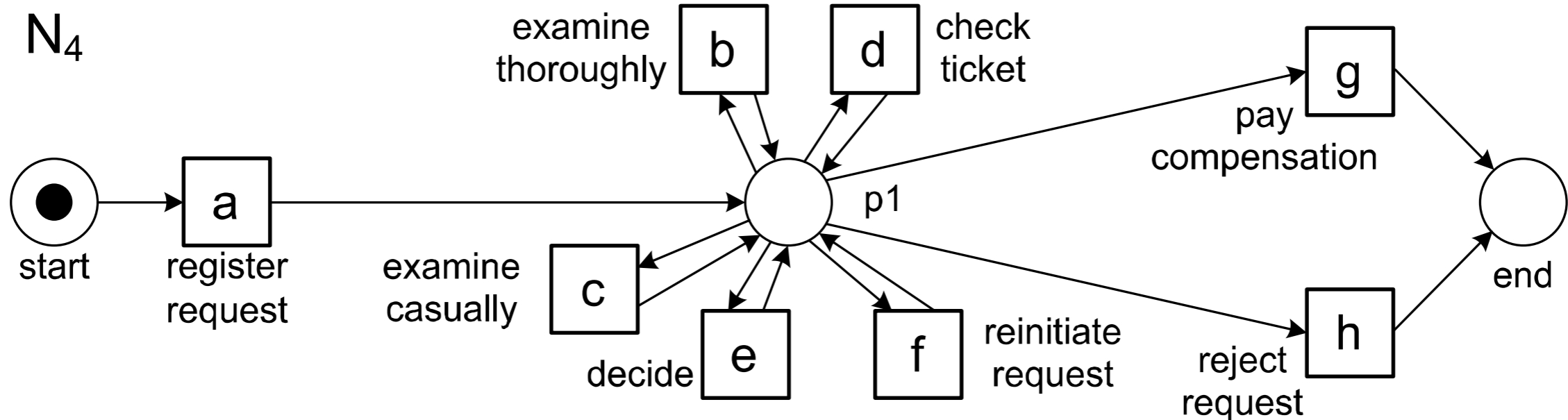
"flower model" (poorly structured)

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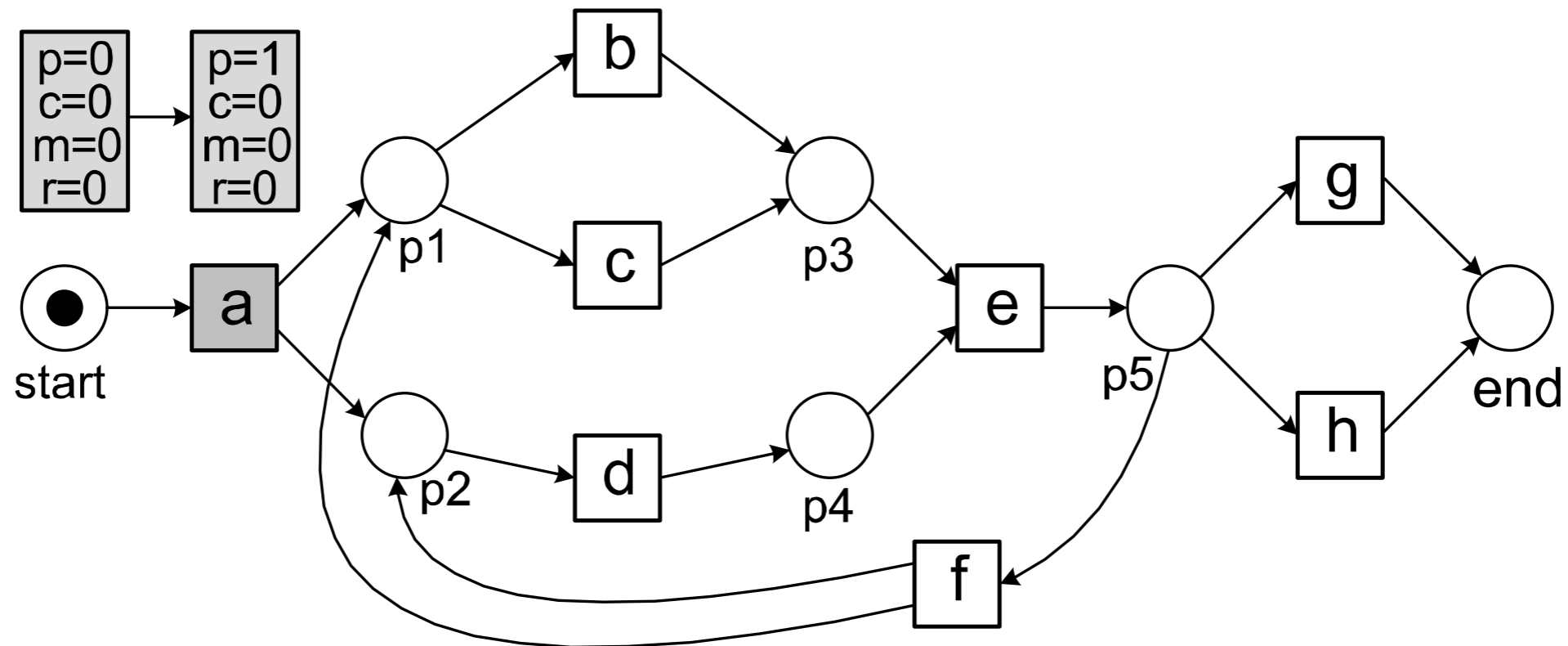
$$\sigma_1 = \langle a, c, d, e, h \rangle$$

Example
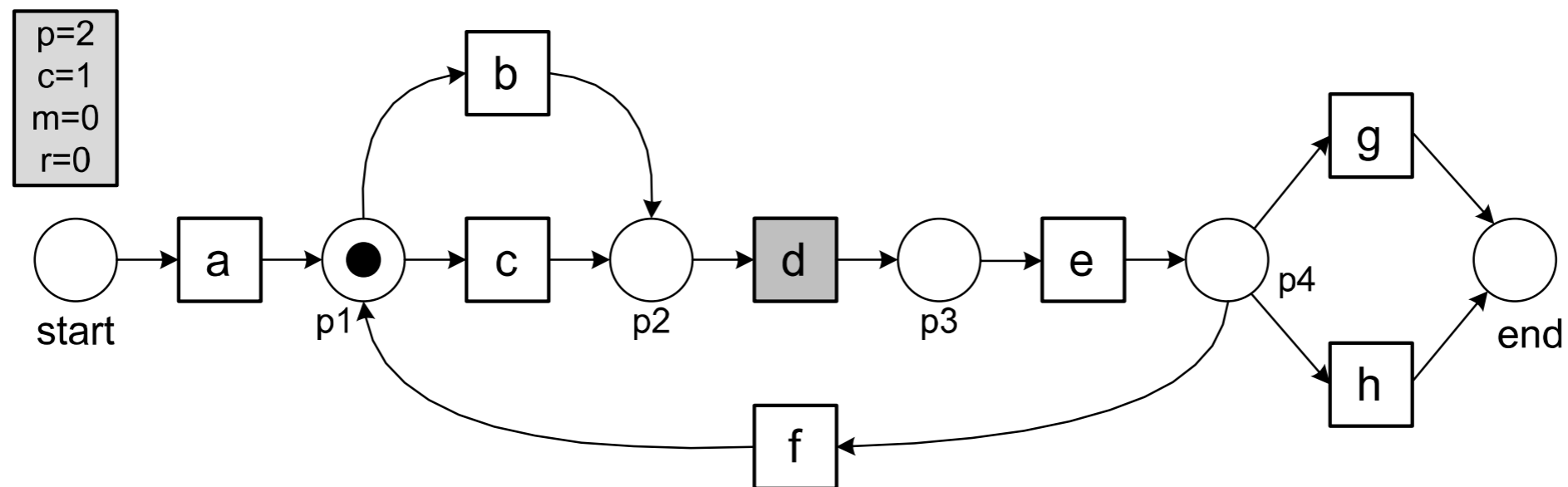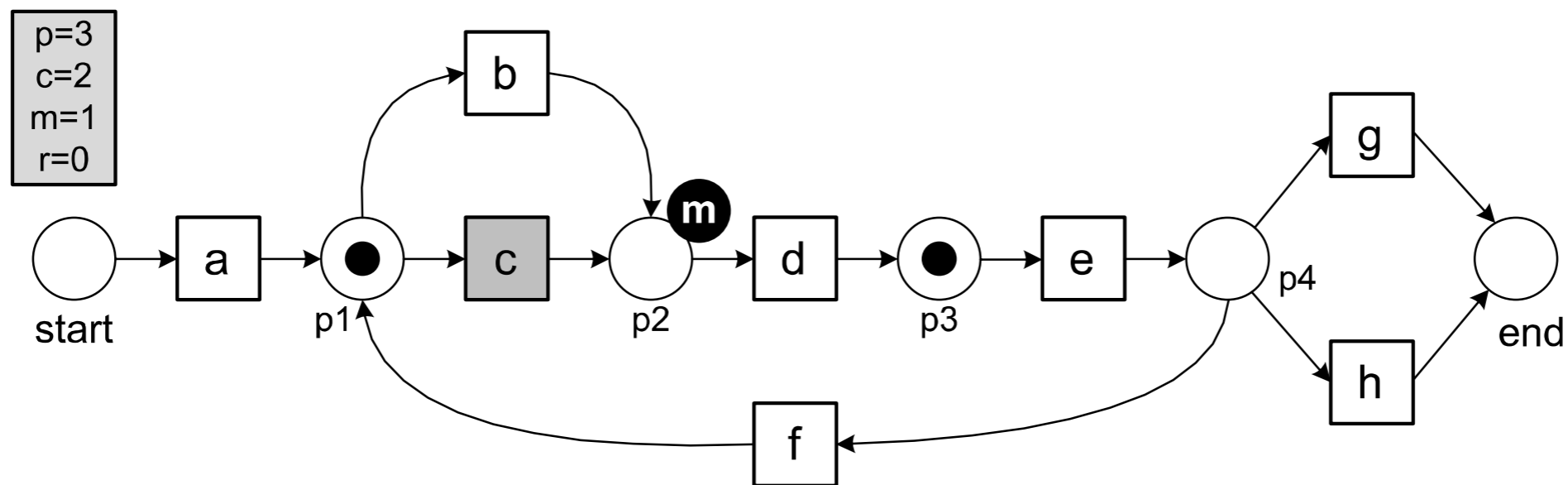
replaying a is possible
one token is consumed, two produced
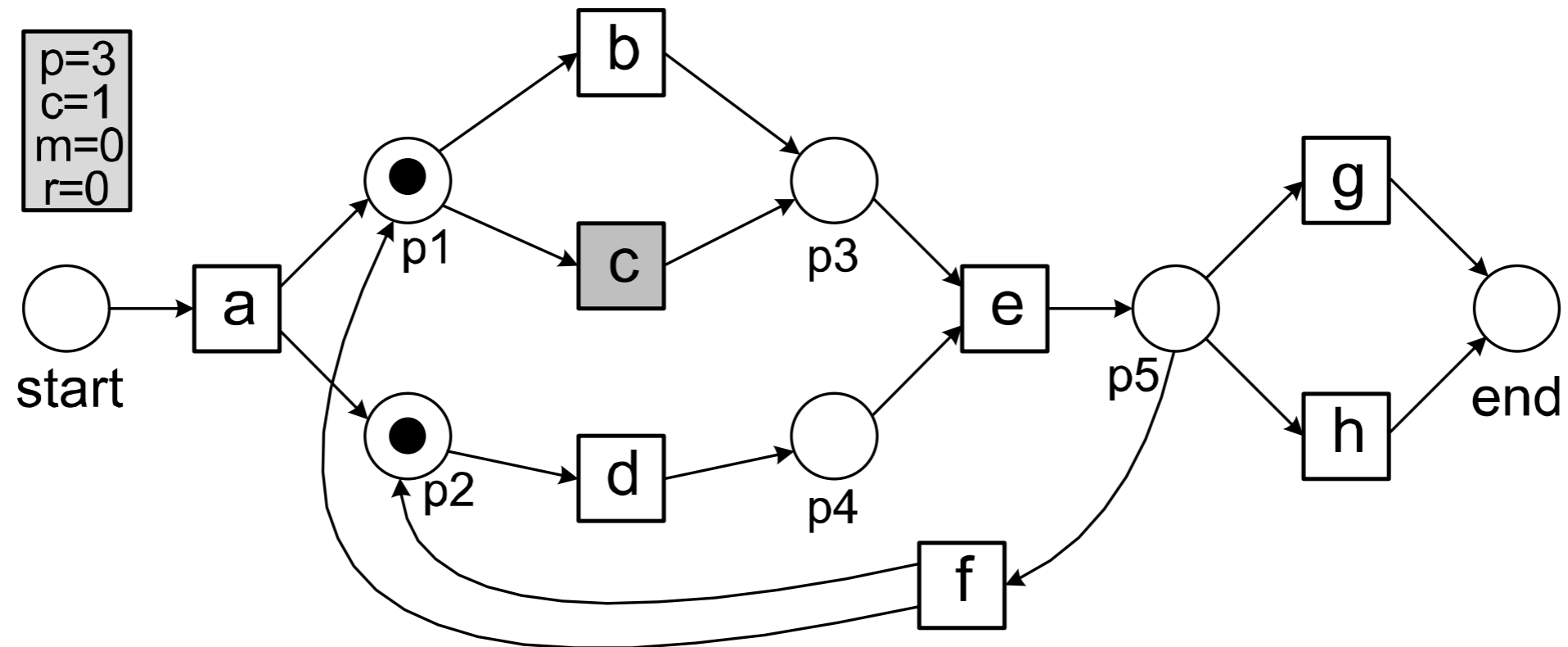
$$\sigma_1 = \langle a, c, d, e, h \rangle$$

Example

replaying c is possible
one token is consumed, one produced
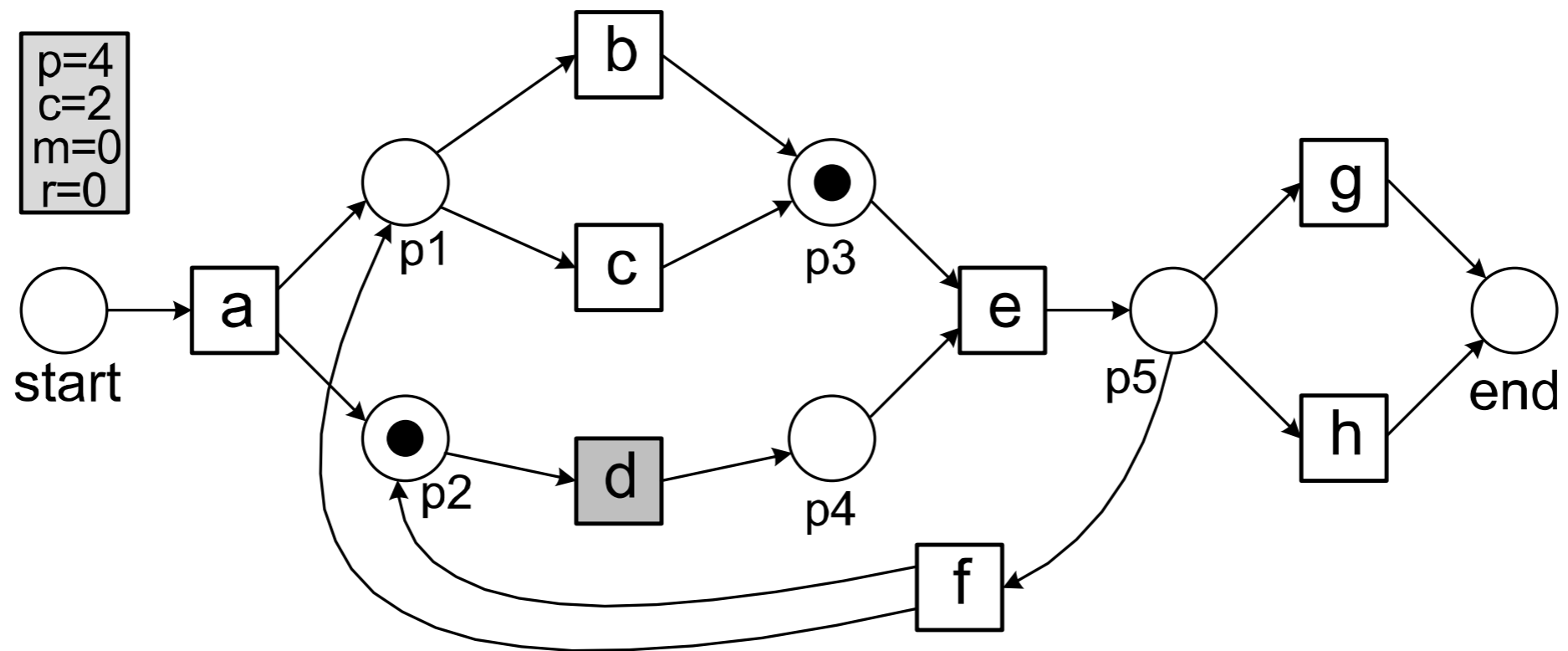
$$\sigma_1 = \langle a, c, d, e, h \rangle$$

Example

replaying d is possible
one token is consumed, one produced
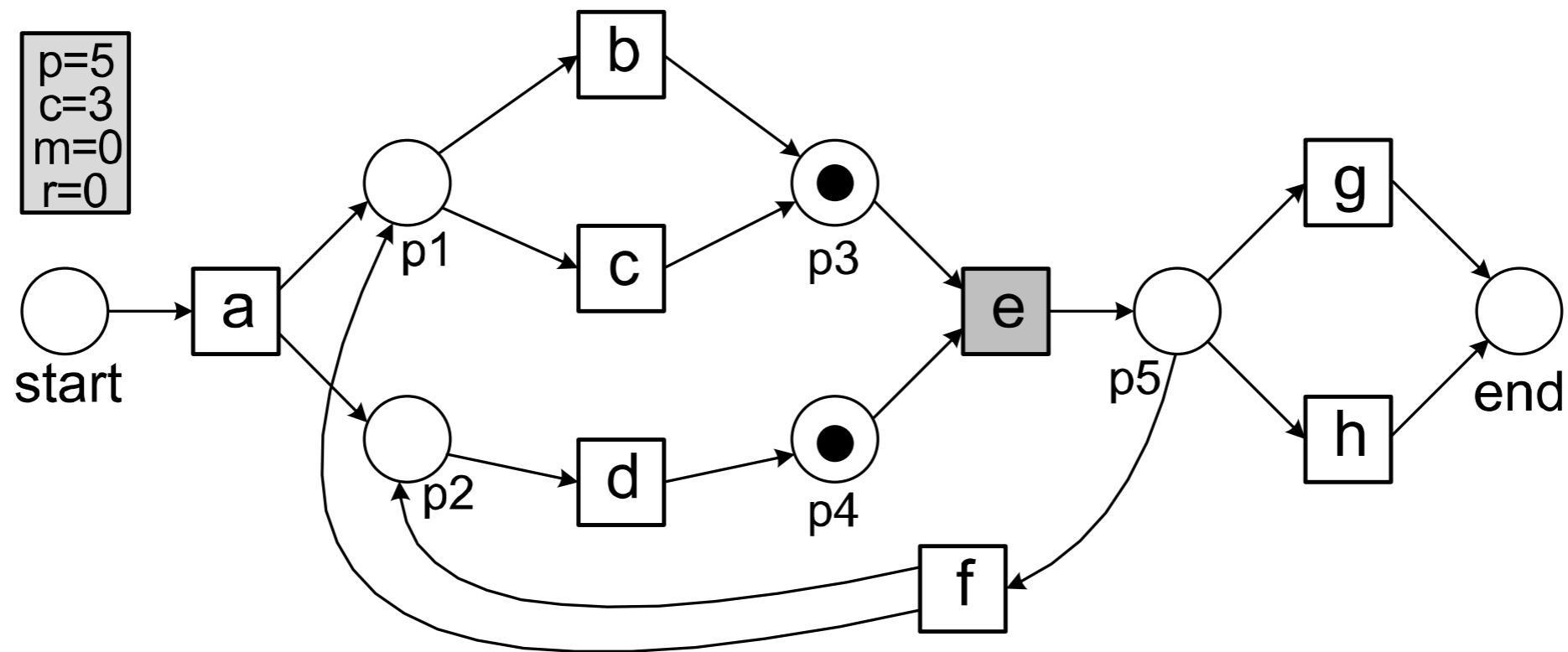
$$\sigma_1 = \langle a, c, d, e, h \rangle$$

Example

replaying e is possible
two tokens are consumed, one produced
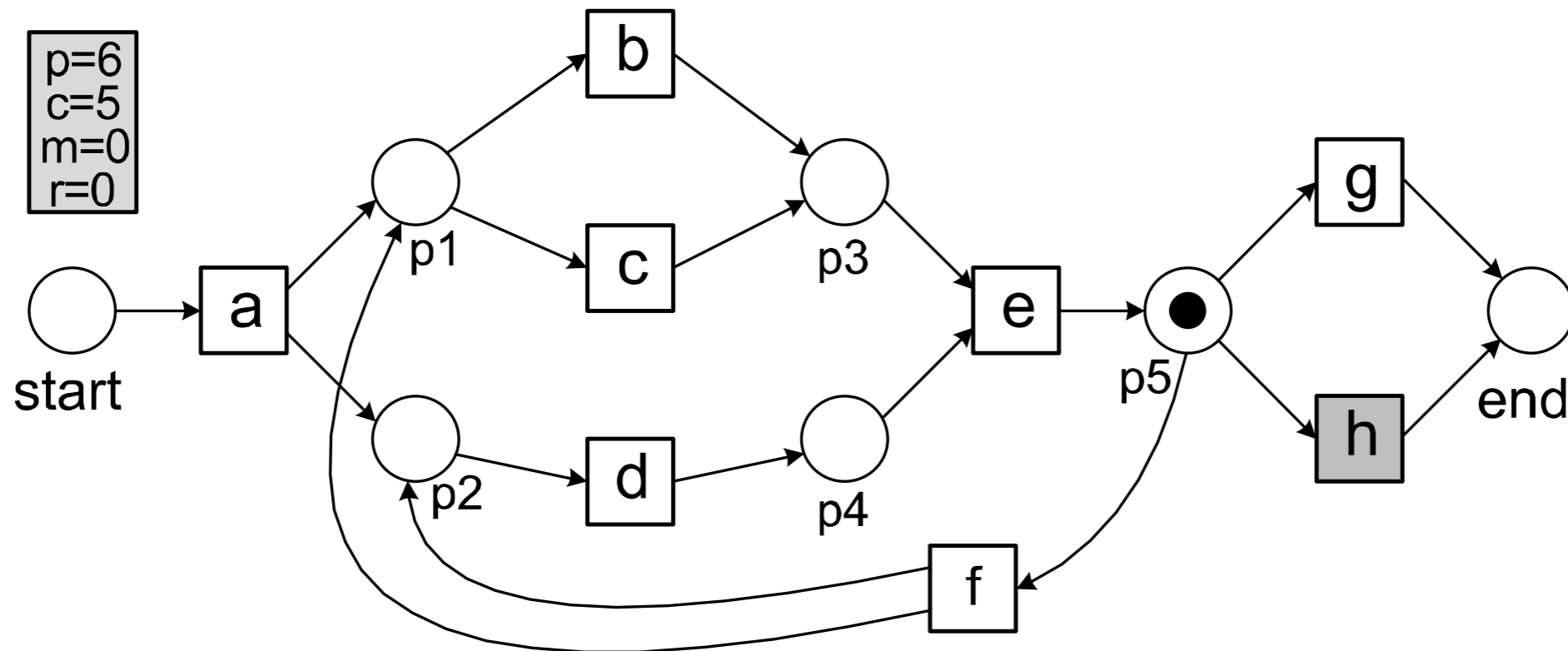
$$\sigma_1 = \langle a, c, d, e, h \rangle$$

Example

replaying h is possible
one token is consumed, one produced
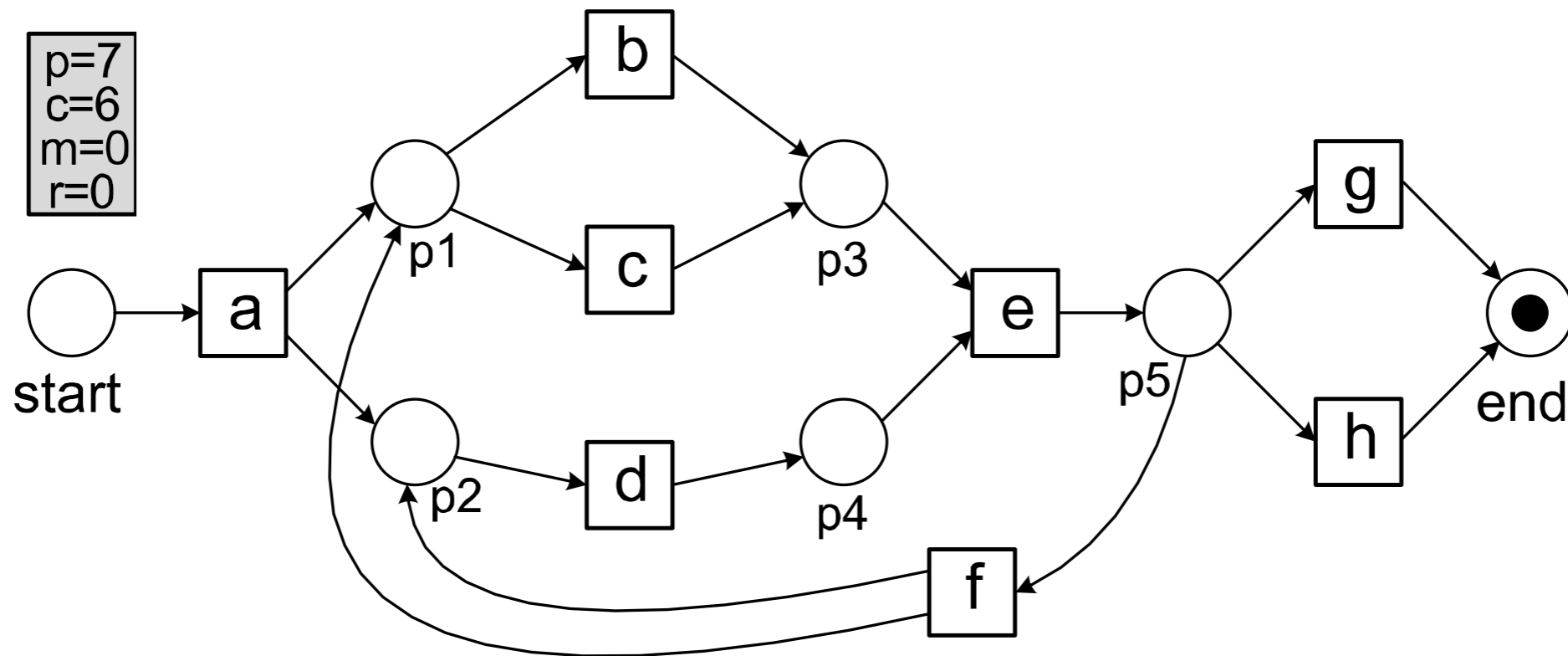
$$fitness(\sigma_1, N_1) = \tfrac{1}{2}(1 - \tfrac{0}{7}) + \tfrac{1}{2}(1 - \tfrac{0}{7}) = 1$$

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

Example

At the end,
the environment consumes
a token from place end.
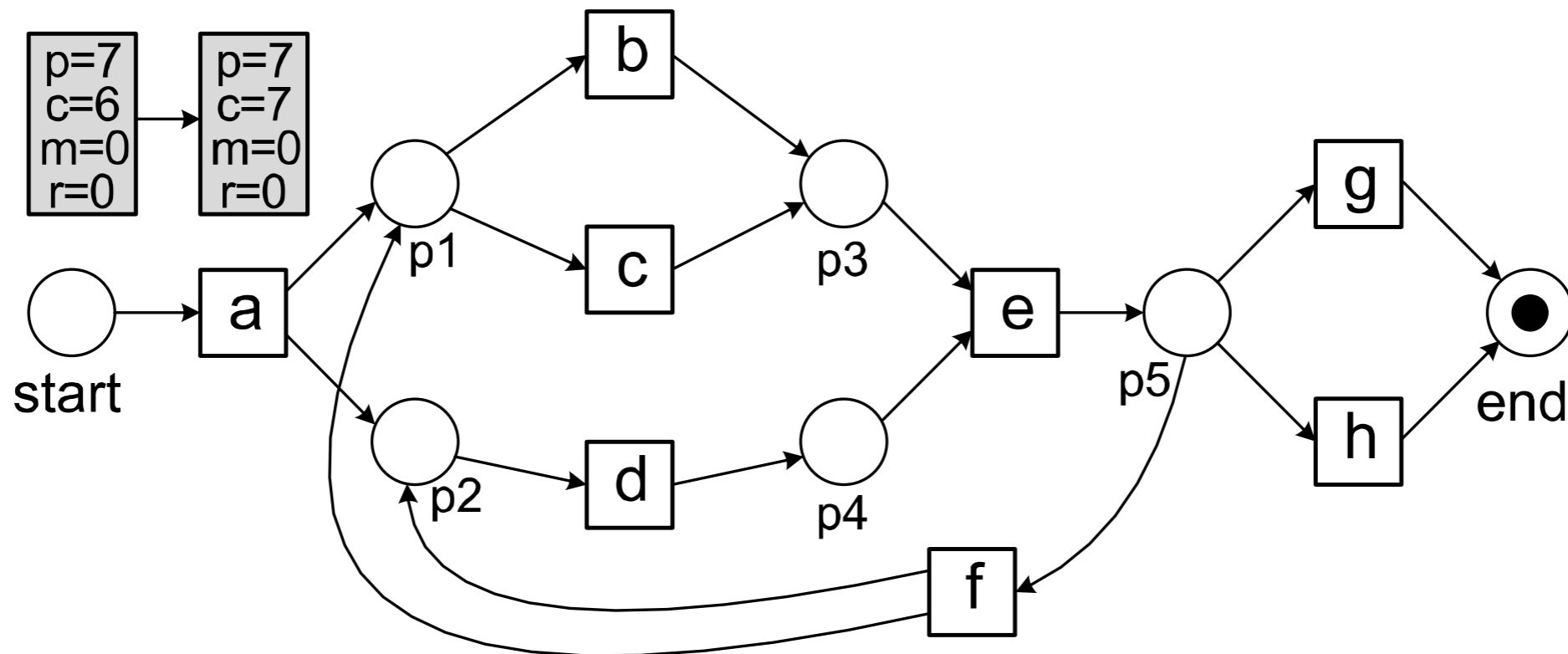
$$fitness(\sigma_1, N_1) = \tfrac{1}{2}(1 - \tfrac{0}{7}) + \tfrac{1}{2}(1 - \tfrac{0}{7}) = 1$$

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

Example: Missing Token
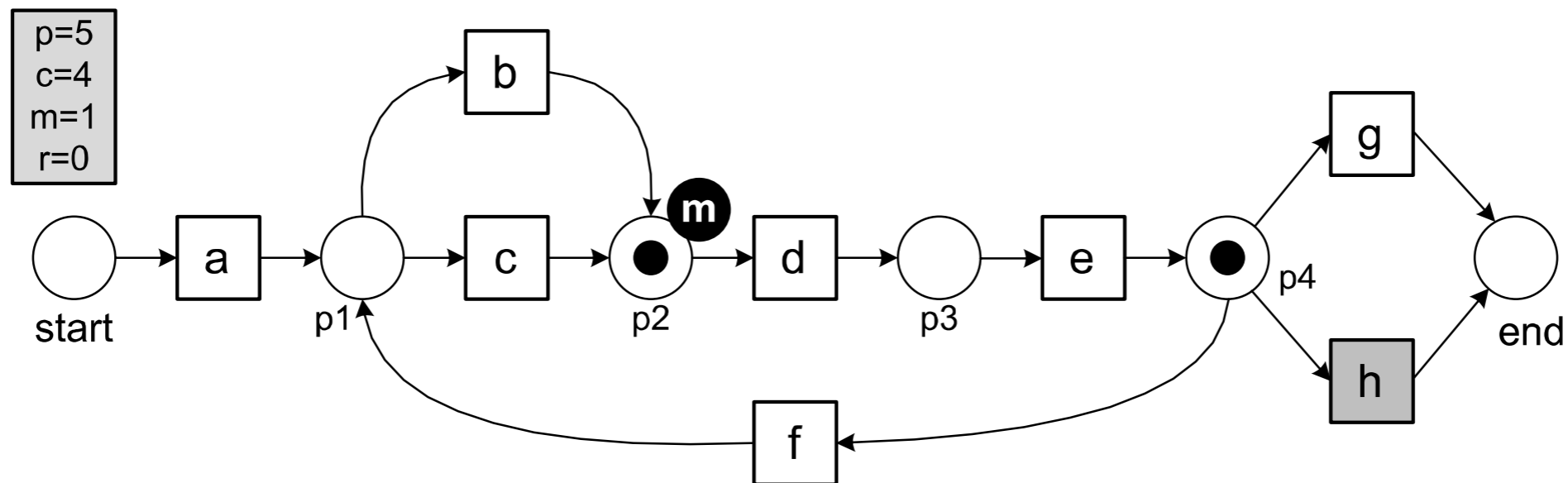
the environment produces a
token for place start
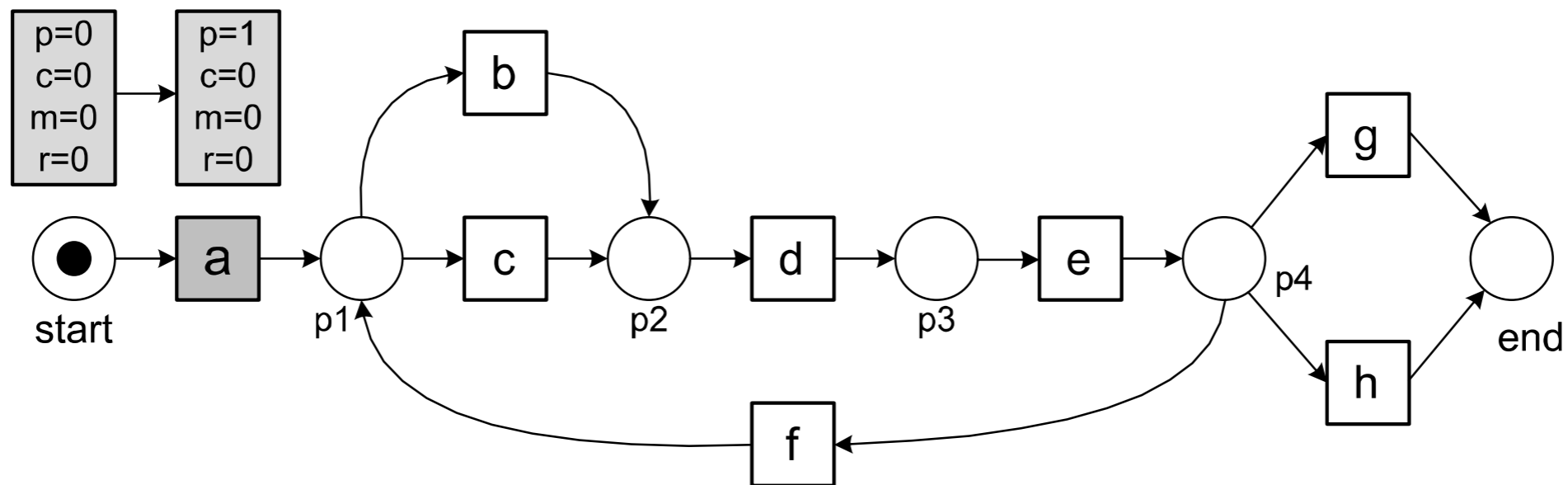
$$\sigma_3 = \langle a, d, c, e, h \rangle$$

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
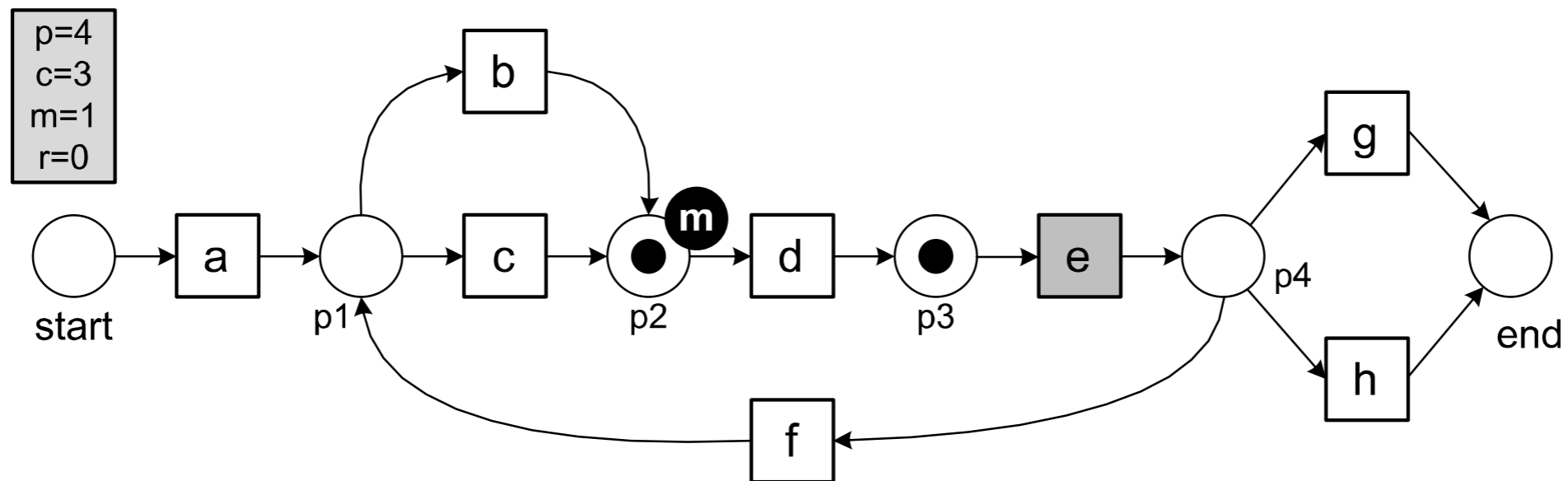
$$\sigma_3 = \langle a, d, c, e, h \rangle$$
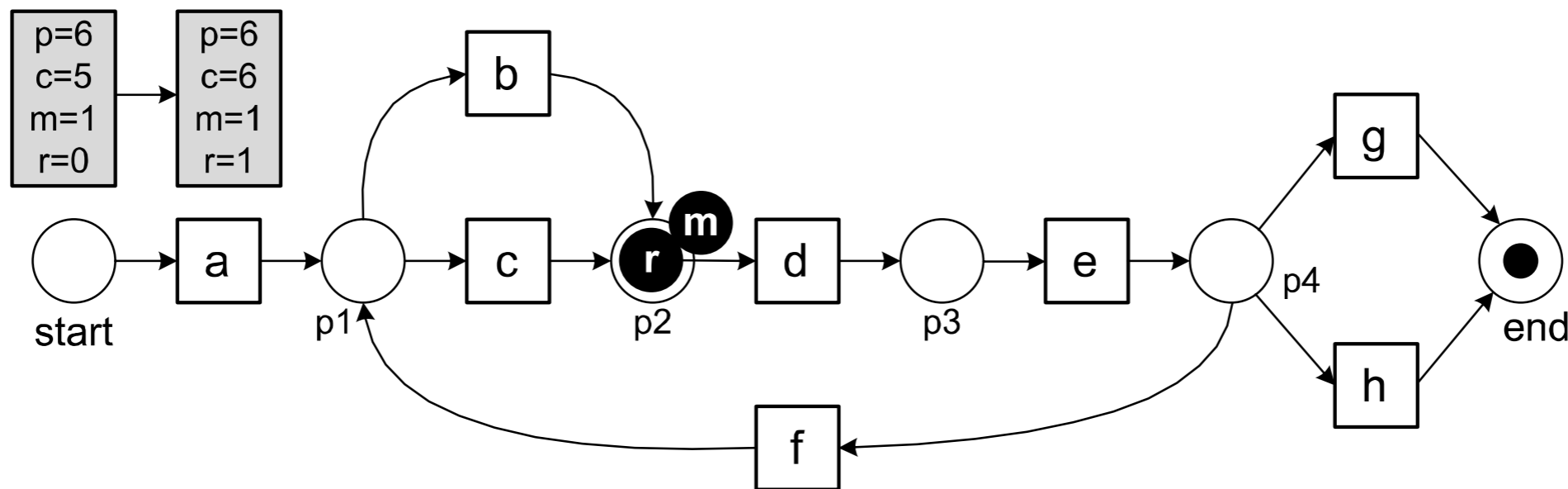
Example: Missing Token

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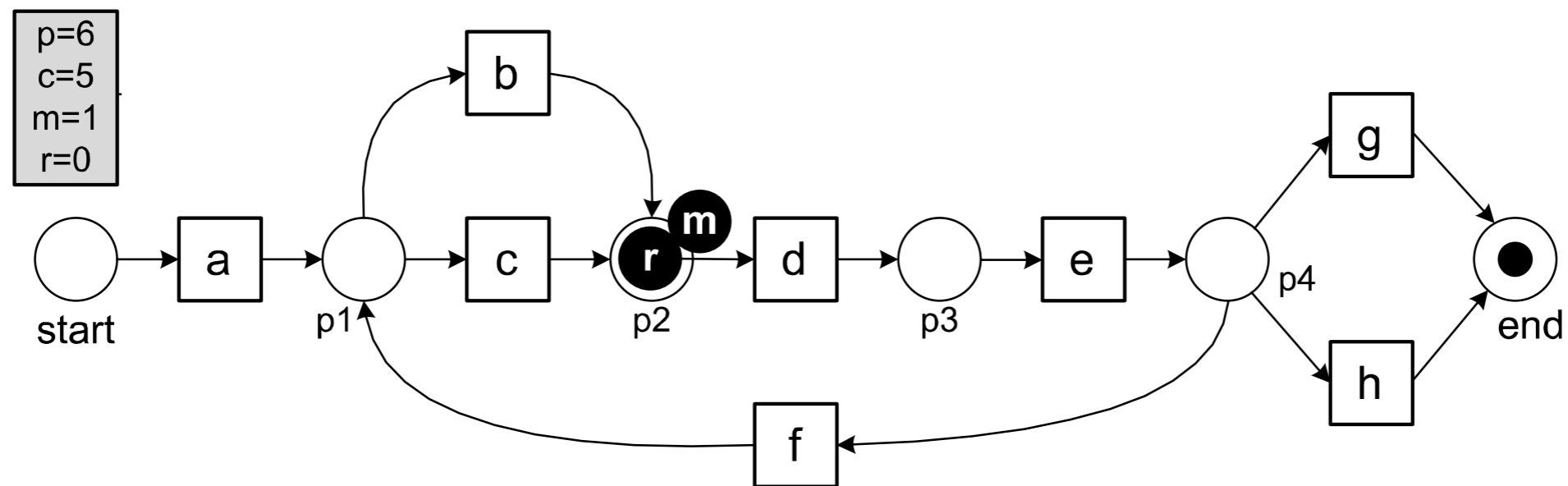
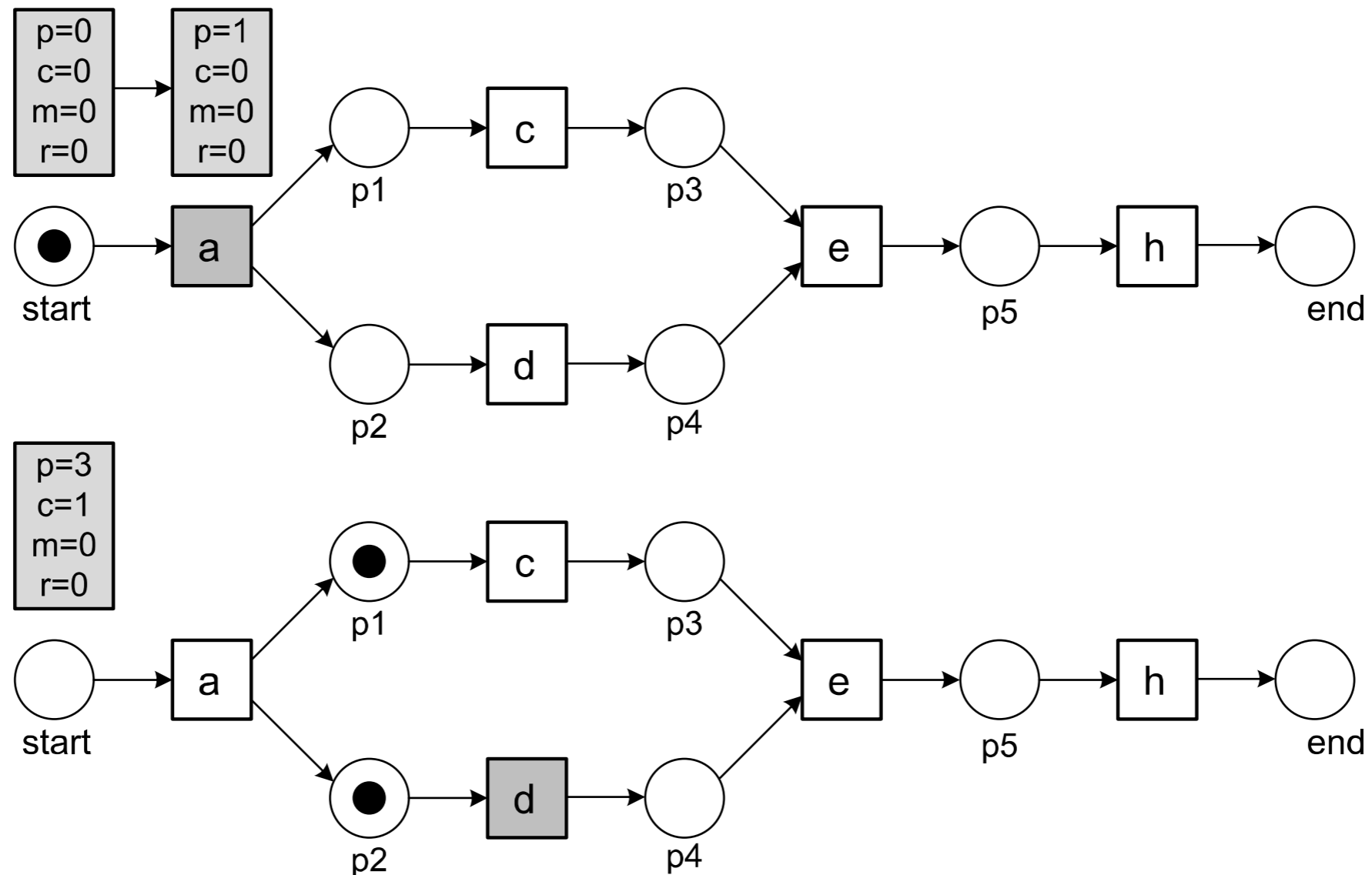$$\sigma_3 = \langle a, d, c, e, h \rangle$$

Example: Missing Token

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

$$fitness(\sigma_3, N_2) = \frac{1}{2}\left(1 - \frac{1}{6}\right) + \frac{1}{2}\left(1 - \frac{1}{6}\right) = 0.8333$$
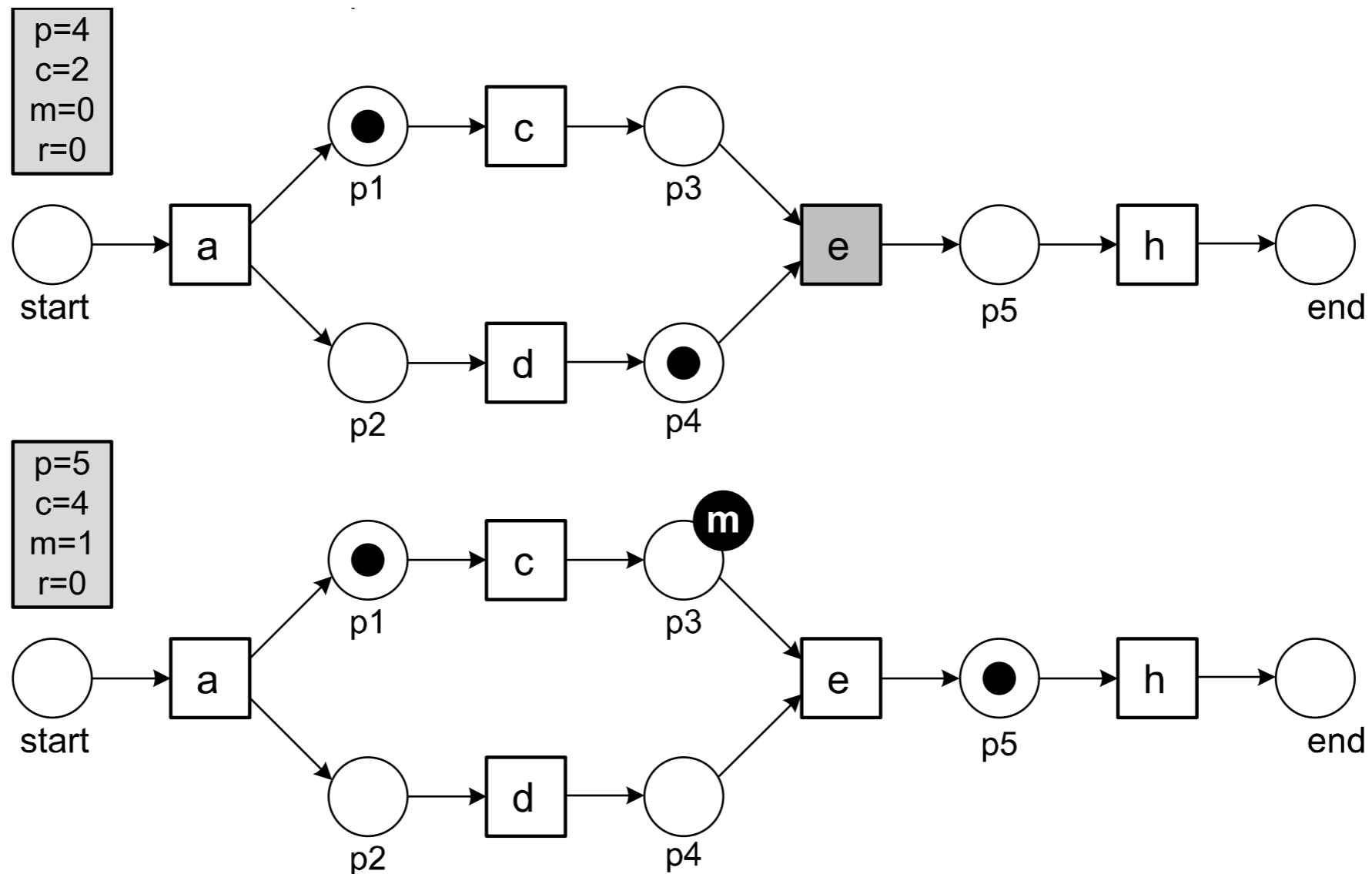
$$\sigma_3 = \langle a, d, c, e, h \rangle$$

Example: Event Removal

$$\sigma_2 = \langle a, b, d, e, g \rangle \qquad \sigma_2' = \langle a, d, e \rangle$$
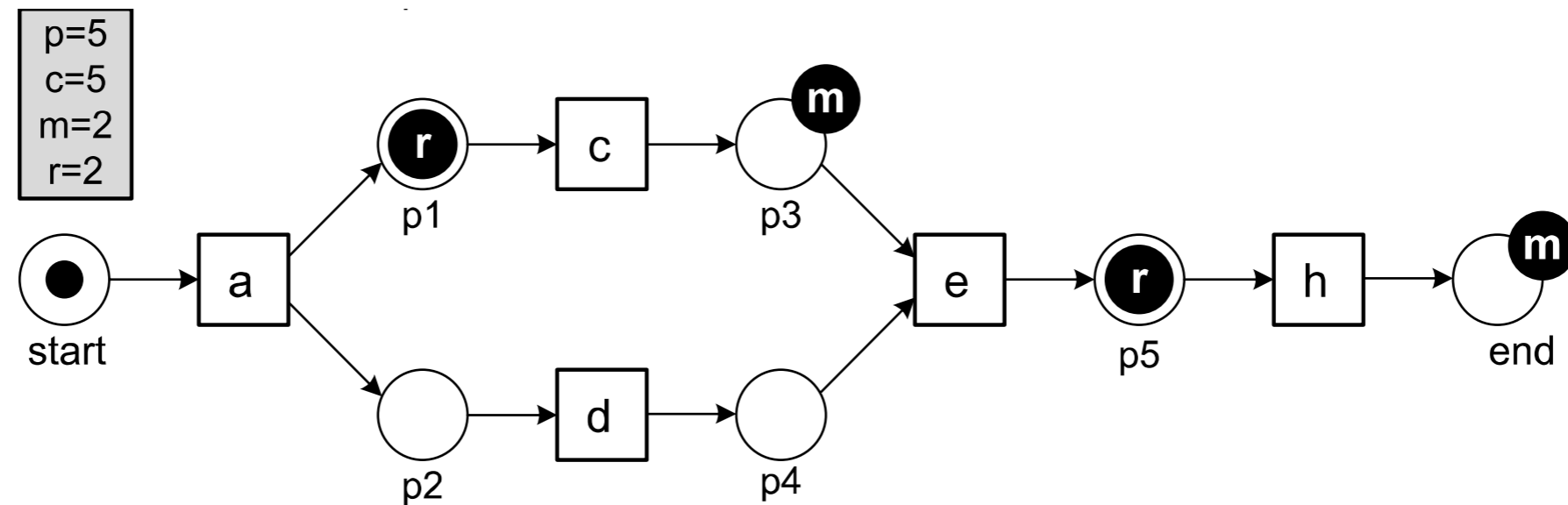
Example: Event Removal

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

Example: Event Removal

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

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

Fitness of a Log

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

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

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

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

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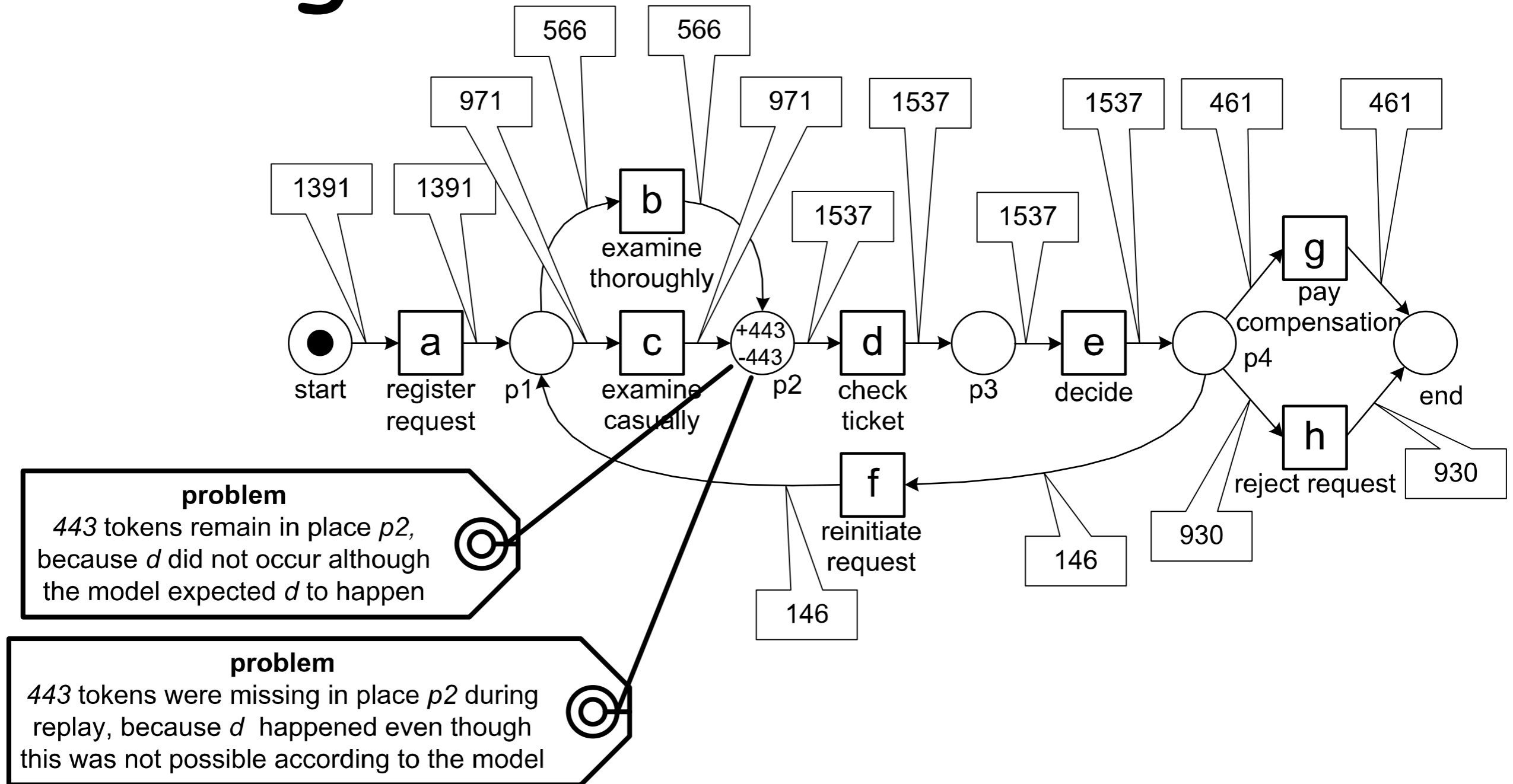
Diagnostic Information

Fig. 7.6 Diagnostic information showing the deviations ($fitness(L_{full}, N_2) = 0.9504$)
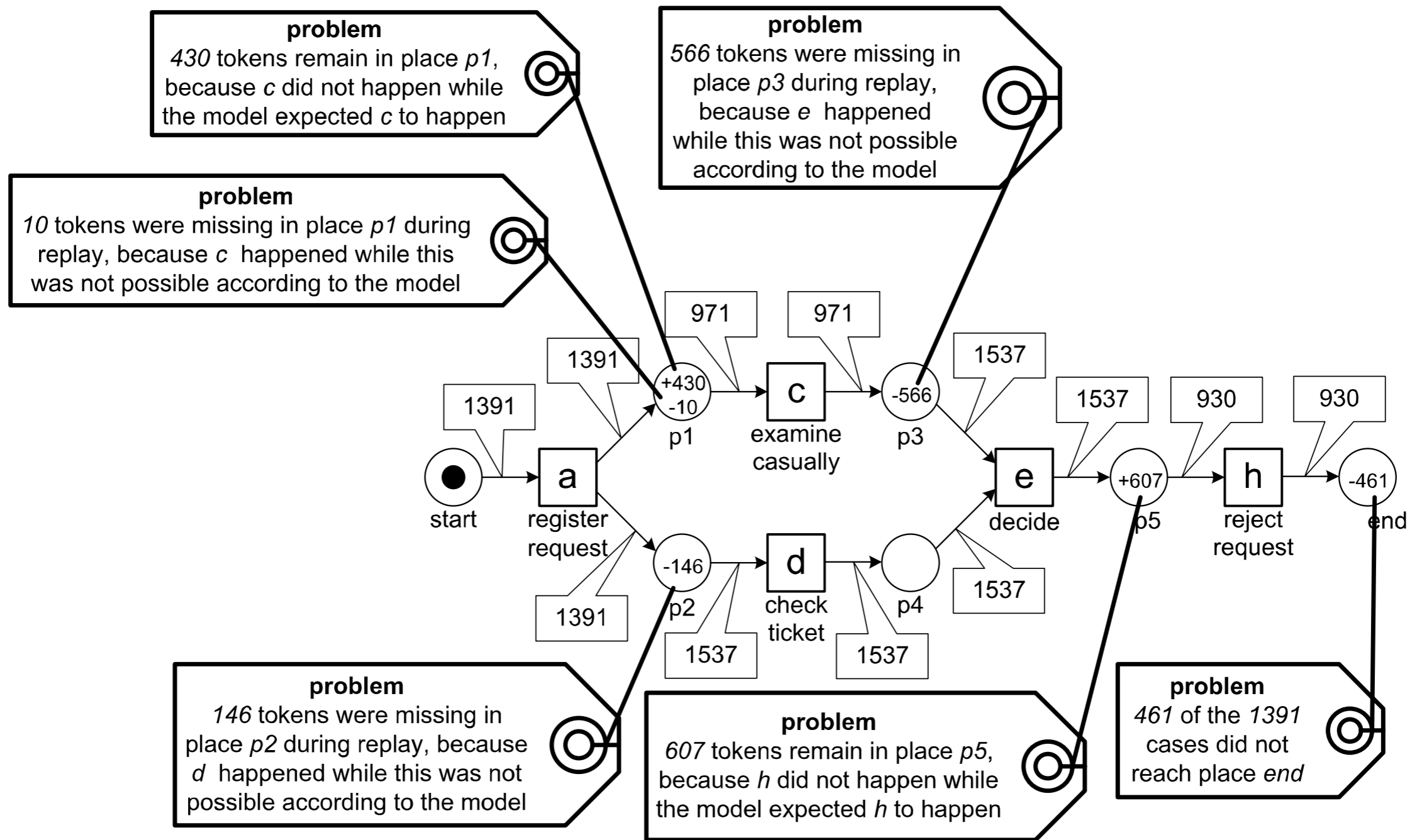
127

Diagnostic Information

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

128

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

130