Elements of Machine Learning & Data Science

Process Mining

Lecture 20

Prof. Wil van der Aalst

Marco Pegoraro, M.Sc.
Tsunghao Huang, M.Sc.
Nina Graves, M.Sc.

Part I: Introduction to Process Mining

Event data, process models, software, applications

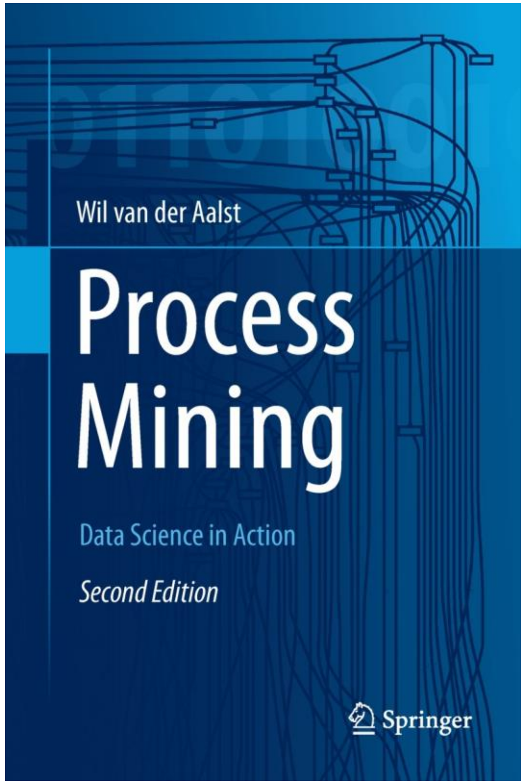
Part II: Unsupervised Process Mining

Process discovery (including Inductive Mining)

Part III: Supervised Process Mining

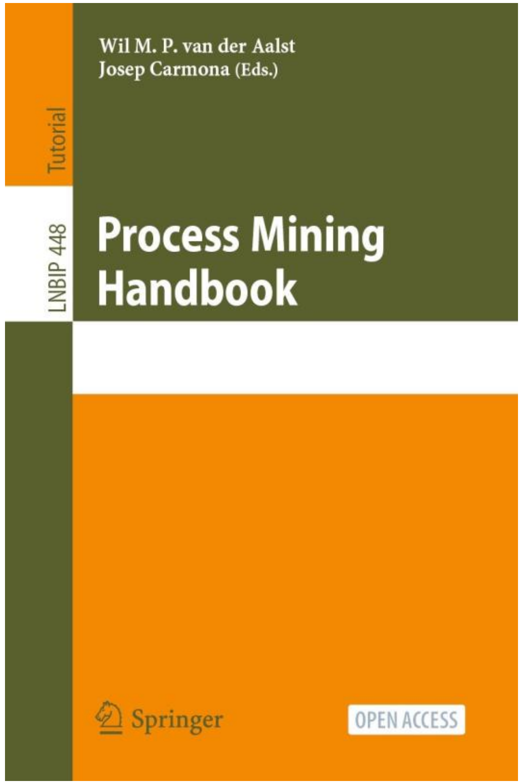
Conformance checking and link to ML (including token-based replay)

Sources for this lecture



W. van der Aalst.
Process Mining: Data Science in Action
2016, Springer
https://link.springer.com/book/10.1007/978-3-662-49851-4

Open Access Online!



W. van der Aalst, Josep Carmona
Process Mining Handbook
2022, Springer
https://link.springer.com/book/10.1007/978-3-031-08848-3

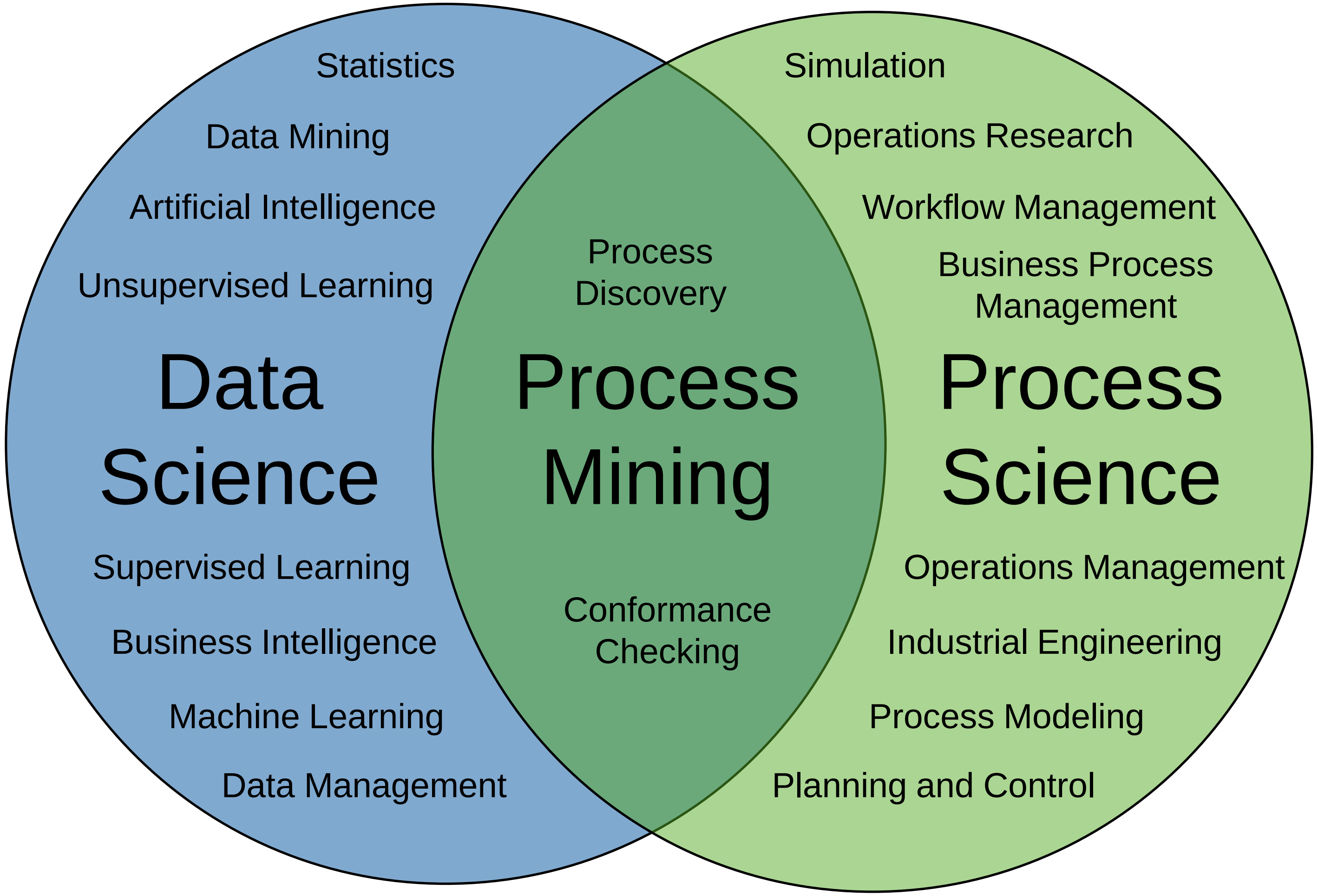
Part I: Introduction to Process Mining

Introduction to Process Mining

1. **Process Mining and Event Data**
2. Process Models
3. Software Tools
4. Applications

Link Between Data and Process Science

Traditionally:
- not process-centric
- focus on specific tasks or decisions



Traditionally:
- not data-driven
- focus on modeling (languages) and automation

Event Data in Process Mining

ID	Activity	Time
11152	Register Order	15.12.22 12:25
11152	Send Invoice	15.12.22 12:45
11152	Pay	15.12.22 13:01
11153	Register Order	15.12.22 13:05
11153	Send Invoice	15.12.22 13:08
11152	Confirm Payment	15.12.22 13:11
11153	Pay	16.12.22 15:03
11152	Make Delivery	17.12.22 8:10

Normal Event Log

We considered timestamped data before. For example:

- **Time series**: numerical features with equidistant timestamps (determined by the sampling rate).
- **Sequence mining**: a very specific setting where we focus on sequences of itemsets.

Event data:

- The occurrence of an event has a meaning, i.e., timestamps are not equidistant.
- Event refers to (at least) a case identifier, activity name, and timestamp.
- Very general!

Event Data in Process Mining

ID	Activity	Time
11152	Register Order	15.12.22 12:25
11152	Send Invoice	15.12.22 12:45
11152	Pay	15.12.22 13:01
11153	Register Order	15.12.22 13:05
11153	Send Invoice	15.12.22 13:08
11152	Confirm Payment	15.12.22 13:11
11153	Pay	16.12.22 15:03
11152	Make Delivery	17.12.22 8:10

Normal Event Log

Each row refers to an event and per event, there are three mandatory attributes:

- Case identifier
- Activity name
- Timestamp

But there can be any number of additional attributes, such as:

- Costs
- Duration
- Location
- Resource
- Etc.

Event Data in Process Mining

ID	Activity	Time
11152	Register Order	15.12.22 12:25
11152	Send Invoice	15.12.22 12:45
11152	Pay	15.12.22 13:01
11153	Register Order	15.12.22 13:05
11153	Send Invoice	15.12.22 13:08
11152	Confirm Payment	15.12.22 13:11
11153	Pay	16.12.22 15:03
11152	Make Delivery	17.12.22 8:10

Normal Event Log

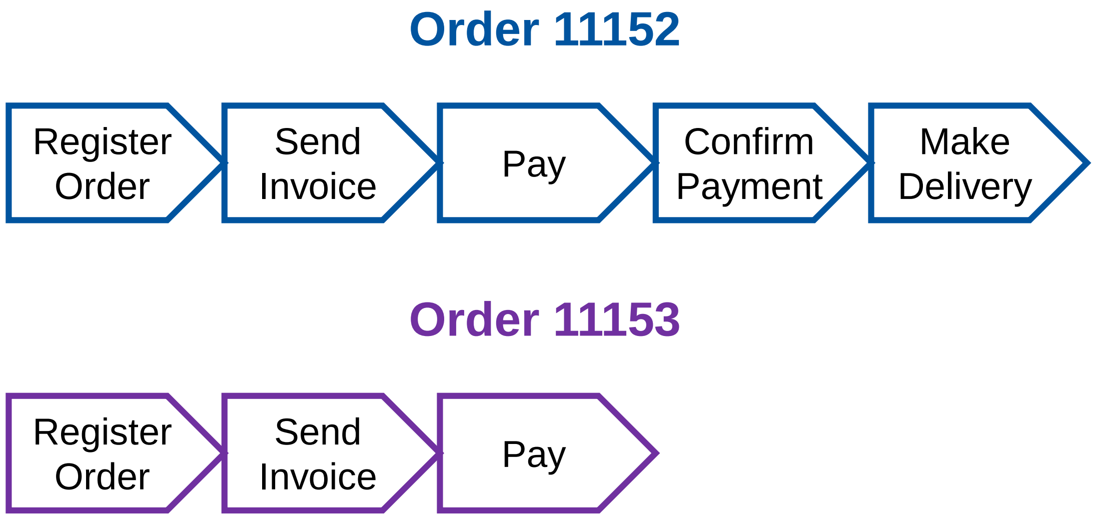
Order 11152

Simplified Event Log

Event Data in Process Mining

ID	Activity	Time
11152	Register Order	15.12.22 12:25
11152	Send Invoice	15.12.22 12:45
11152	Pay	15.12.22 13:01
11153	Register Order	15.12.22 13:05
11153	Send Invoice	15.12.22 13:08
11152	Confirm Payment	15.12.22 13:11
11153	Pay	16.12.22 15:03
11152	Make Delivery	17.12.22 8:10

Normal Event Log

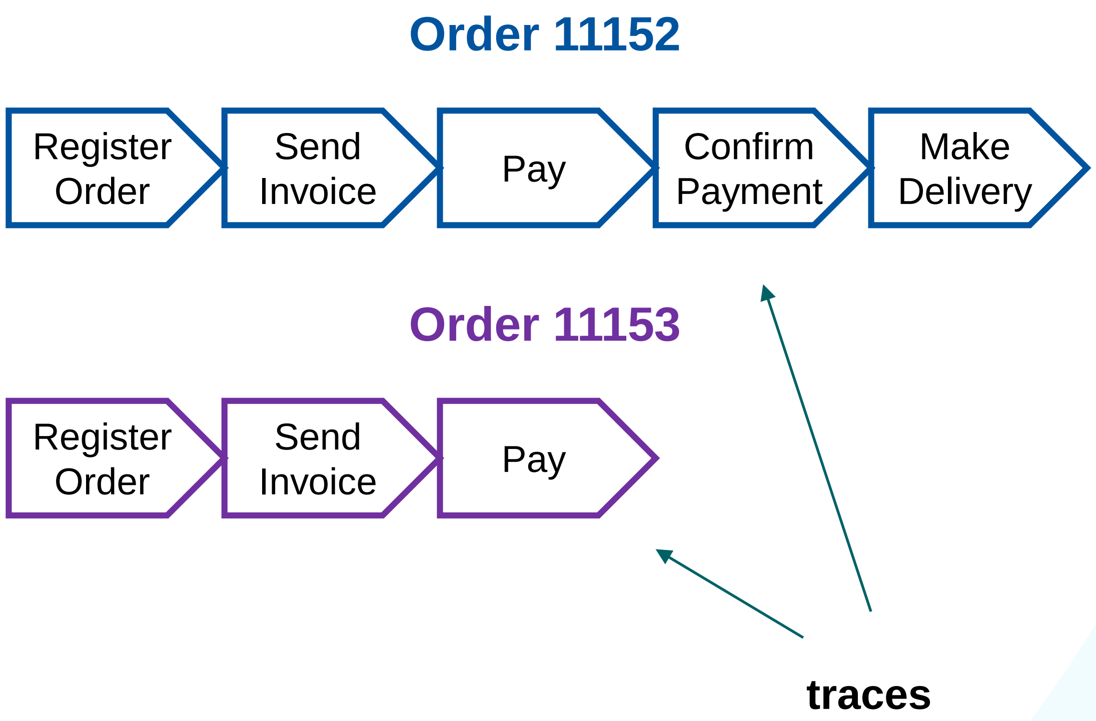


Simplified Event Log

Event Data in Process Mining

ID	Activity	Time
11152	Register Order	15.12.22 12:25
11152	Send Invoice	15.12.22 12:45
11152	Pay	15.12.22 13:01
11153	Register Order	15.12.22 13:05
11153	Send Invoice	15.12.22 13:08
11152	Confirm Payment	15.12.22 13:11
11153	Pay	16.12.22 15:03
11152	Make Delivery	17.12.22 8:10

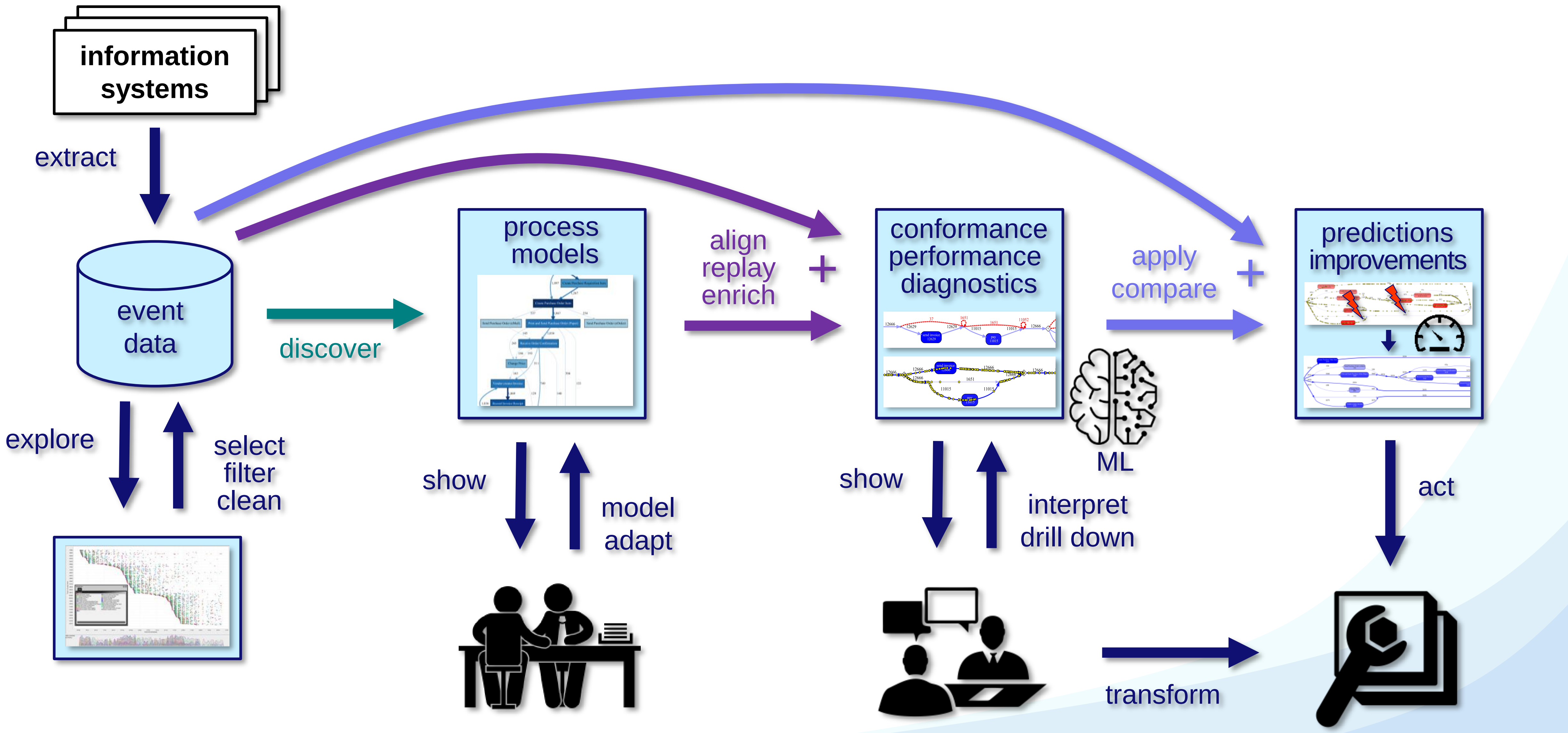
Normal Event Log



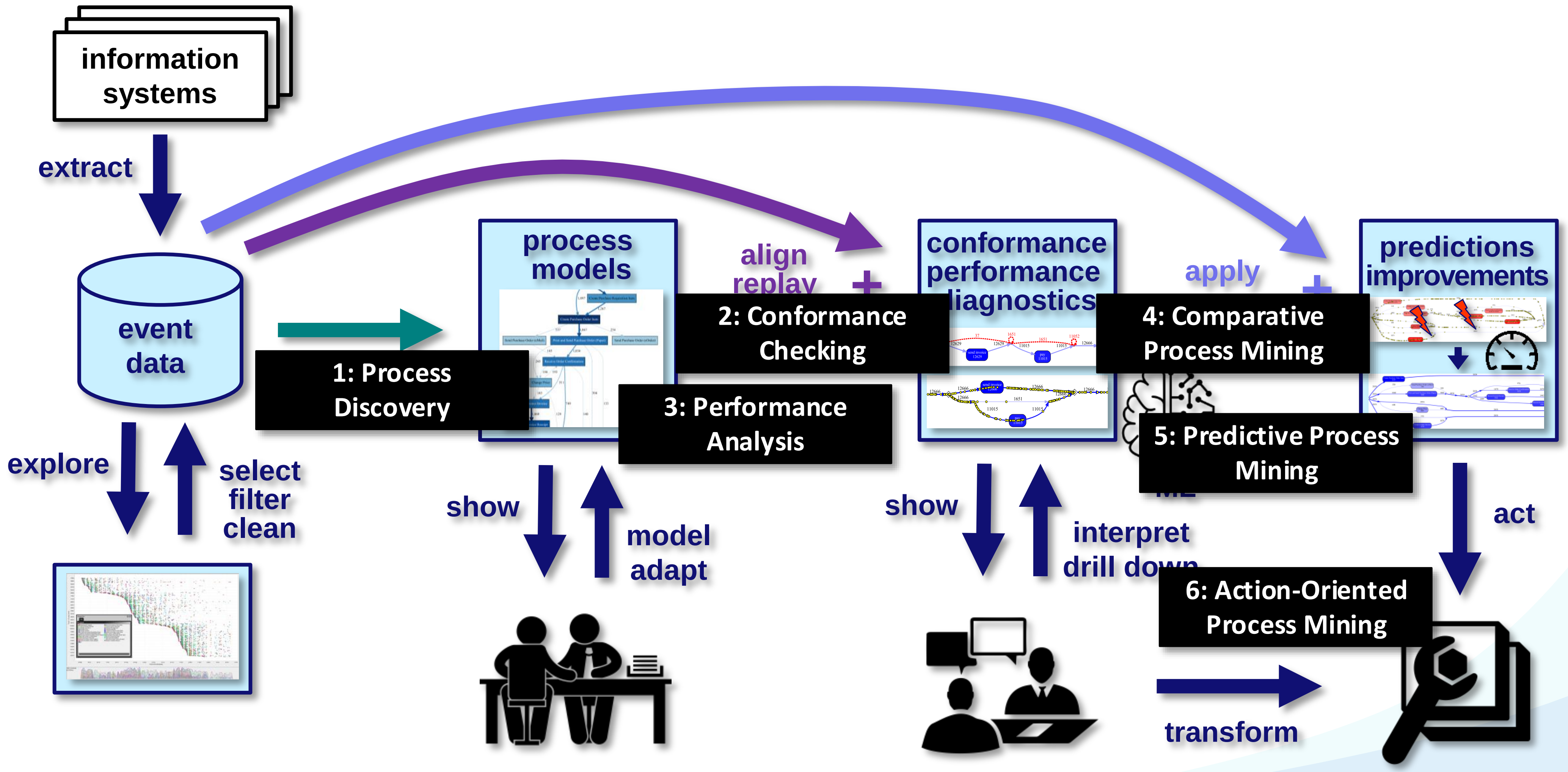
Simplified Event Log

Simplified event log is a multiset of traces

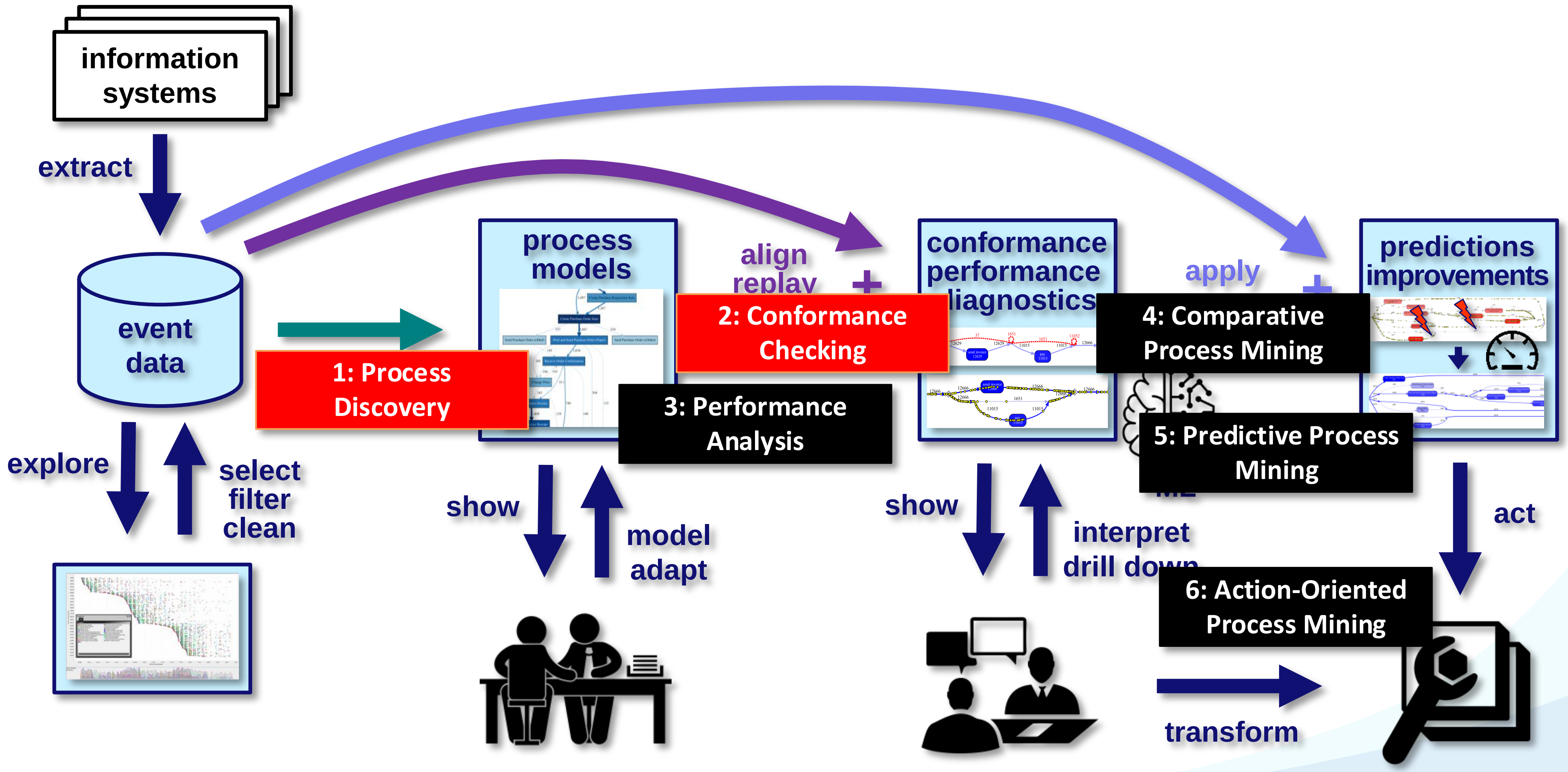
Process Mining Overview



Six Tasks in Process Mining



Six Tasks in Process Mining



Introduction to Process Mining

1. Process Mining and Event Data
2. **Process Models**
3. Software Tools
4. Applications

Four Common Types of Process Models

The same process can be visualized in many ways:

- DFGs (Directly-Follows Graphs)

  Supported by all process mining tools (simple, but no concurrency)

- Petri nets

  The oldest model for concurrent processes and the de facto standard in process mining research

- Process trees

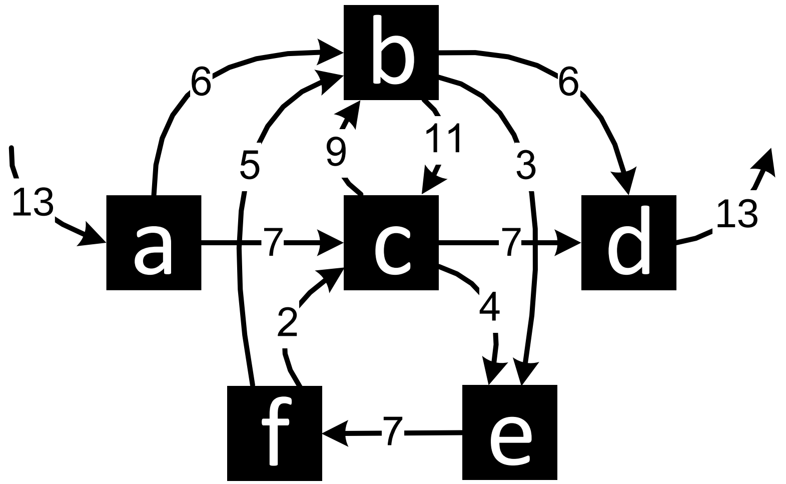
  Frequently used in process mining because it is block structured and sound by construction

- BPMN (Business Process Model and Notation)

  The industry standard (we use a small subset) related to UML Activity Diagrams (not explained in detail)

Directly-Follows Graphs

- Simplest notation
- Marks all edges between activities that occurred
- Helps to get first impression about the data
- Used by process discovery algorithms (e.g., Inductive Miner)



Directly-Follows Graphs – Example

Assume we have a simplified log:

$$L = [\langle a, b, d, e, h\rangle, \langle a, d, c, e, g\rangle, \langle a, c, d, e, f, b, d, e, g\rangle, \langle a, d, b, e, h\rangle, \langle a, c, d, e, f, d, c, e, f, c, d, e, h\rangle, \langle a, c, d, e, g\rangle]$$

$a =$ register request

$b =$ examine thoroughly

$c =$ examine casually

$d =$ check ticket

$e =$ decide

$f =$ reinitiate request

$g =$ pay compensation

$h =$ reject request

Directly-Follows Graphs – Example

Assume we have a simplified log:

$$L = [\langle a, b, d, e, h\rangle, \langle a, d, c, e, g\rangle, \langle a, c, d, e, f, b, d, e, g\rangle, \langle a, d, b, e, h\rangle, \langle a, c, d, e, f, d, c, e, f, c, d, e, h\rangle, \langle a, c, d, e, g\rangle]$$

$a =$ register request

$b =$ examine thoroughly

$c =$ examine casually

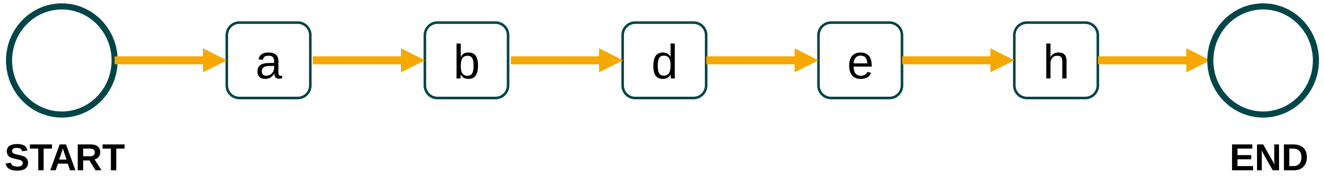
$d =$ check ticket

$e =$ decide

$f =$ reinitiate request

$g =$ pay compensation

$h =$ reject request

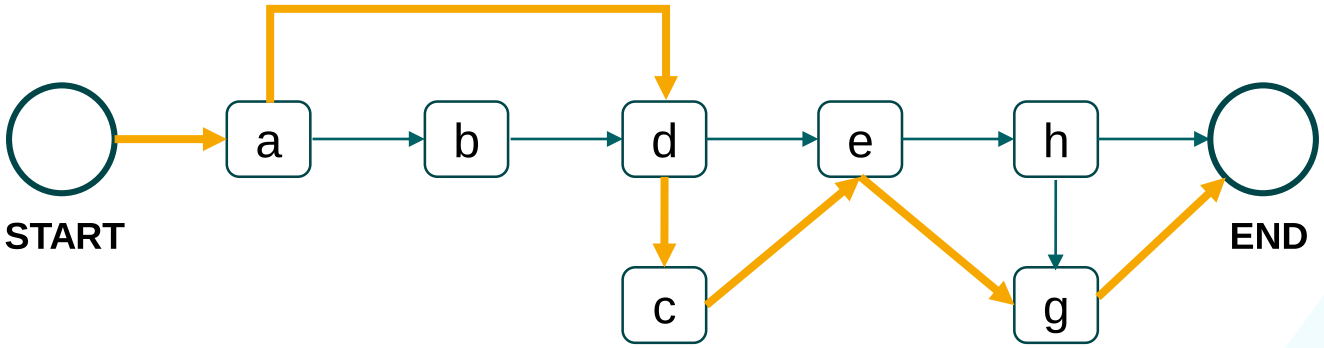


Directly-Follows Graphs – Example

Assume we have a simplified log:

$$L = [\langle a, b, d, e, h\rangle, \langle a, d, c, e, g\rangle, \langle a, c, d, e, f, b, d, e, g\rangle, \langle a, d, b, e, h\rangle, \langle a, c, d, e, f, d, c, e, f, c, d, e, h\rangle, \langle a, c, d, e, g\rangle]$$

$a =$ register request
$b =$ examine thoroughly
$c =$ examine casually
$d =$ check ticket
$e =$ decide
$f =$ reinitiate request
$g =$ pay compensation
$h =$ reject request



Directly-Follows Graphs – Example

Assume we have a simplified log:

$$L = [\langle a,b,d,e,h\rangle, \langle a,d,c,e,g\rangle, \langle a,c,d,e,f,b,d,e,g\rangle, \langle a,d,b,e,h\rangle, \langle a,c,d,e,f,d,c,e,f,c,d,e,h\rangle, \langle a,c,d,e,g\rangle]$$

$a =$ register request

$b =$ examine thoroughly

$c =$ examine casually

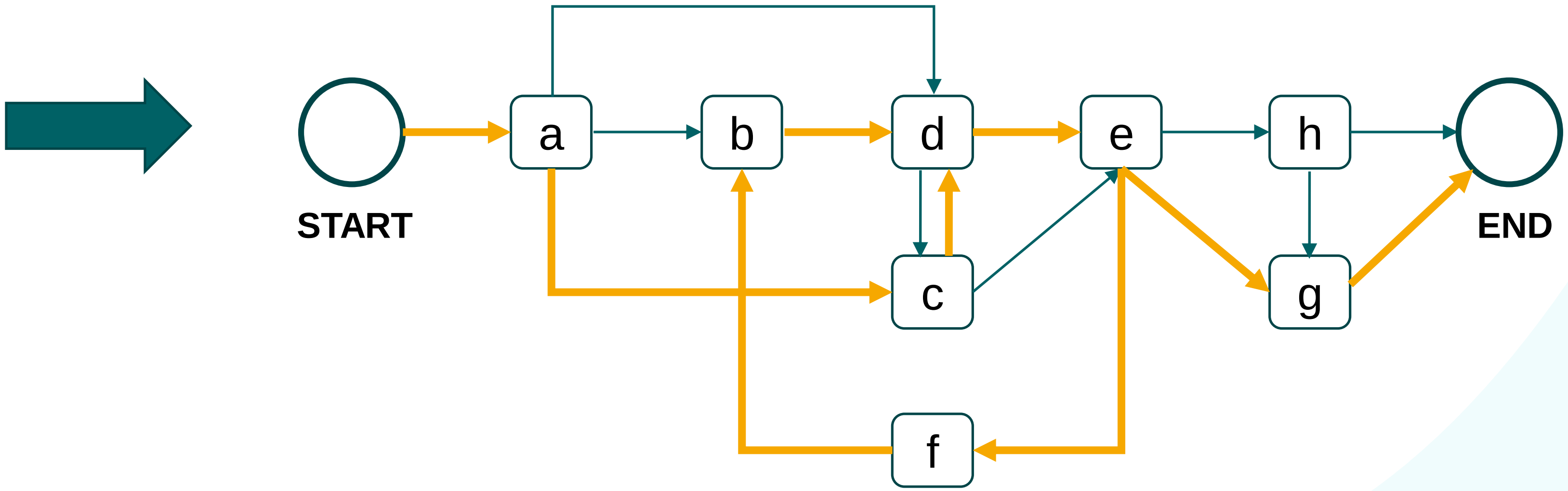
$d =$ check ticket

$e =$ decide

$f =$ reinitiate request

$g =$ pay compensation

$h =$ reject request

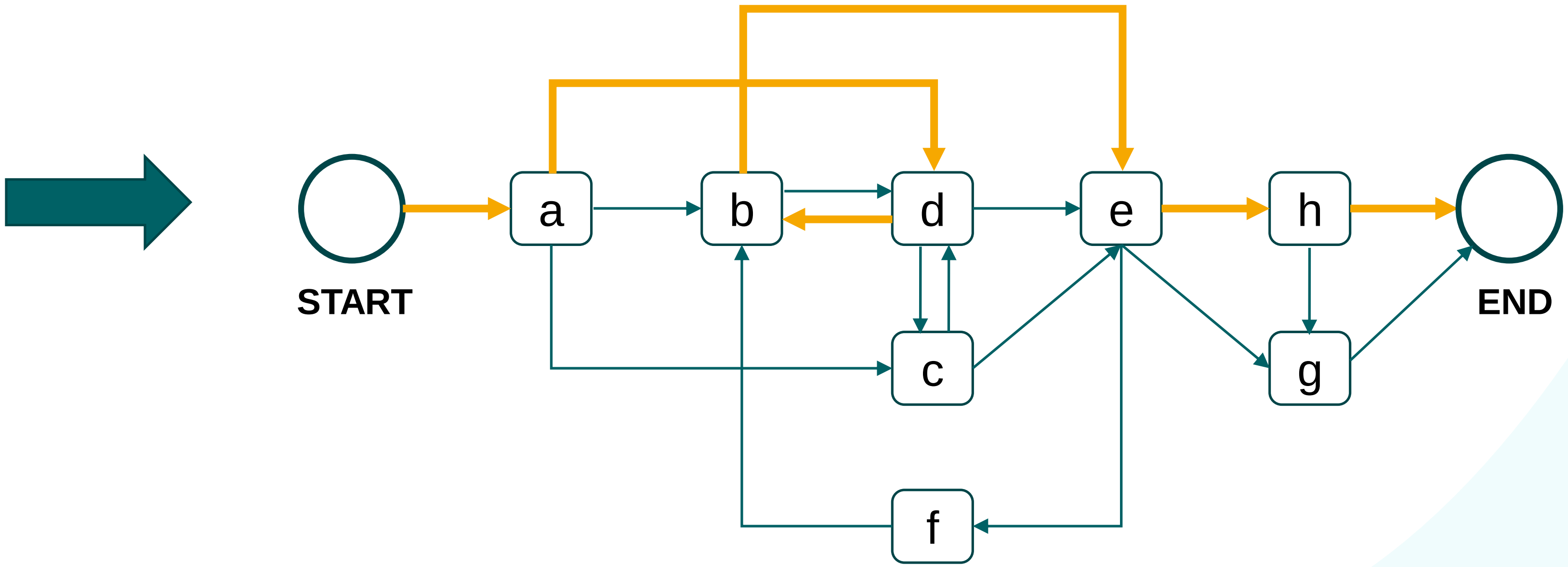


Directly-Follows Graphs – Example

Assume we have a simplified log:

$$L = [\langle a,b,d,e,h\rangle, \langle a,d,c,e,g\rangle, \langle a,c,d,e,f,b,d,e,g\rangle, \langle a,d,b,e,h\rangle, \langle a,c,d,e,f,d,c,e,f,c,d,e,h\rangle, \langle a,c,d,e,g\rangle]$$

$a =$ register request
$b =$ examine thoroughly
$c =$ examine casually
$d =$ check ticket
$e =$ decide
$f =$ reinitiate request
$g =$ pay compensation
$h =$ reject request



Directly-Follows Graphs – Example

Assume we have a simplified log:

$$L = [\langle a,b,d,e,h\rangle, \langle a,d,c,e,g\rangle, \langle a,c,d,e,f,b,d,e,g\rangle, \langle a,d,b,e,h\rangle, \langle a,c,d,e,f,d,c,e,f,c,d,e,h\rangle, \langle a,c,d,e,g\rangle]$$

$a =$ register request

$b =$ examine thoroughly

$c =$ examine casually

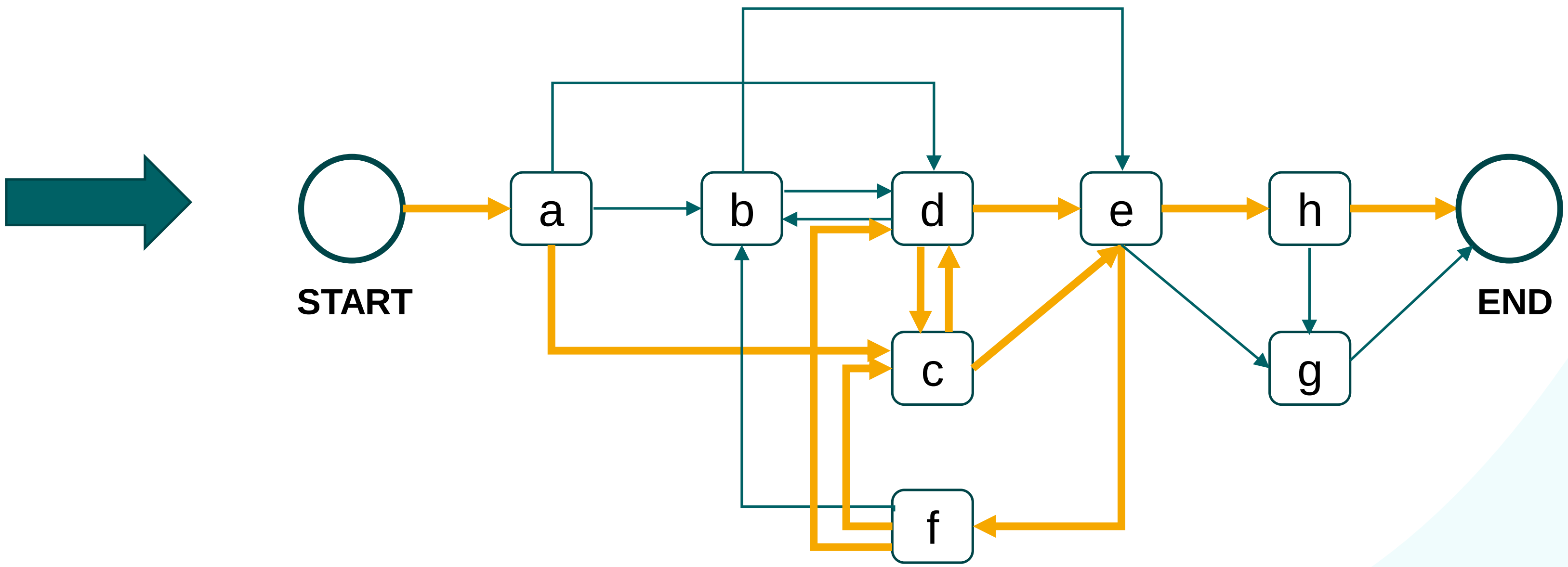
$d =$ check ticket

$e =$ decide

$f =$ reinitiate request

$g =$ pay compensation

$h =$ reject request

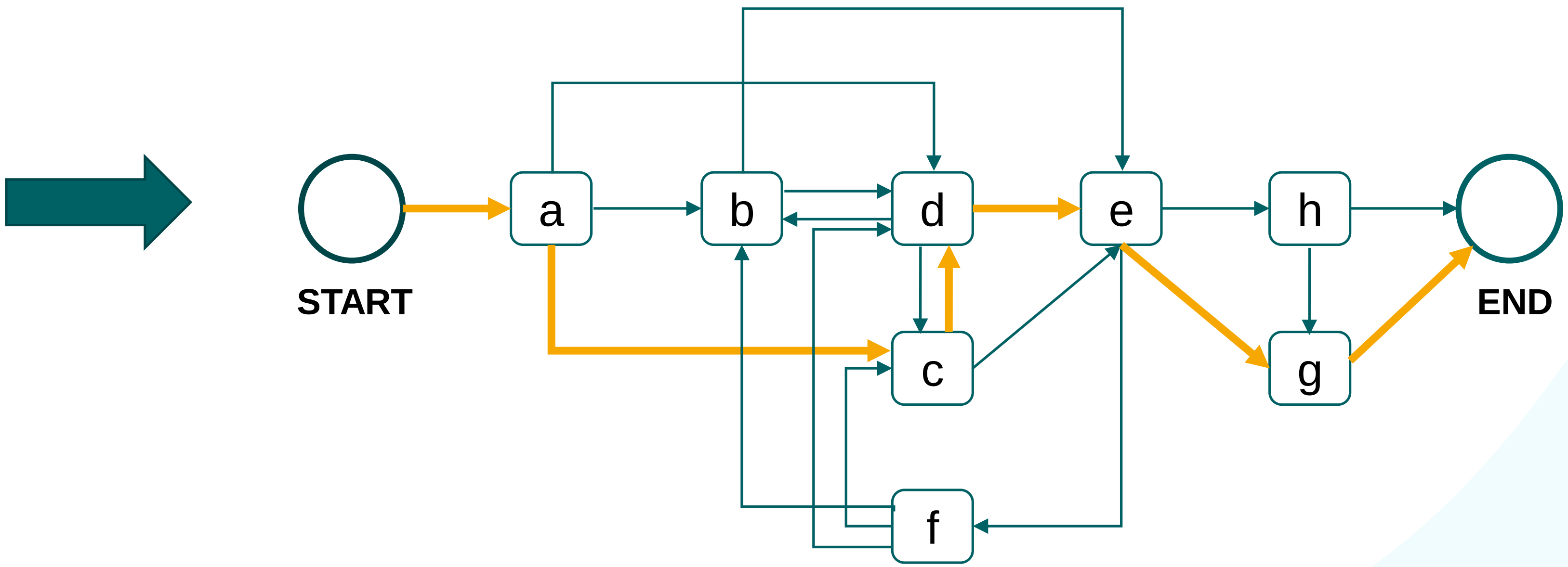


Directly-Follows Graphs – Example

Assume we have a simplified log:

$$L = [\langle a,b,d,e,h\rangle, \langle a,d,c,e,g\rangle, \langle a,c,d,e,f,b,d,e,g\rangle, \langle a,d,b,e,h\rangle, \langle a,c,d,e,f,d,c,e,f,c,d,e,h\rangle, \langle a,c,d,e,g\rangle]$$

$a =$ register request
$b =$ examine thoroughly
$c =$ examine casually
$d =$ check ticket
$e =$ decide
$f =$ reinitiate request
$g =$ pay compensation
$h =$ reject request



Directly-Follows Graphs – Example

Assume we have a simplified log:

$$L = [\langle a, b, d, e, h\rangle, \langle a, d, c, e, g\rangle, \langle a, c, d, e, f, b, d, e, g\rangle, \langle a, d, b, e, h\rangle, \langle a, c, d, e, f, d, c, e, f, c, d, e, h\rangle, \langle a, c, d, e, g\rangle]$$

$a =$ register request

$b =$ examine thoroughly

$c =$ examine casually

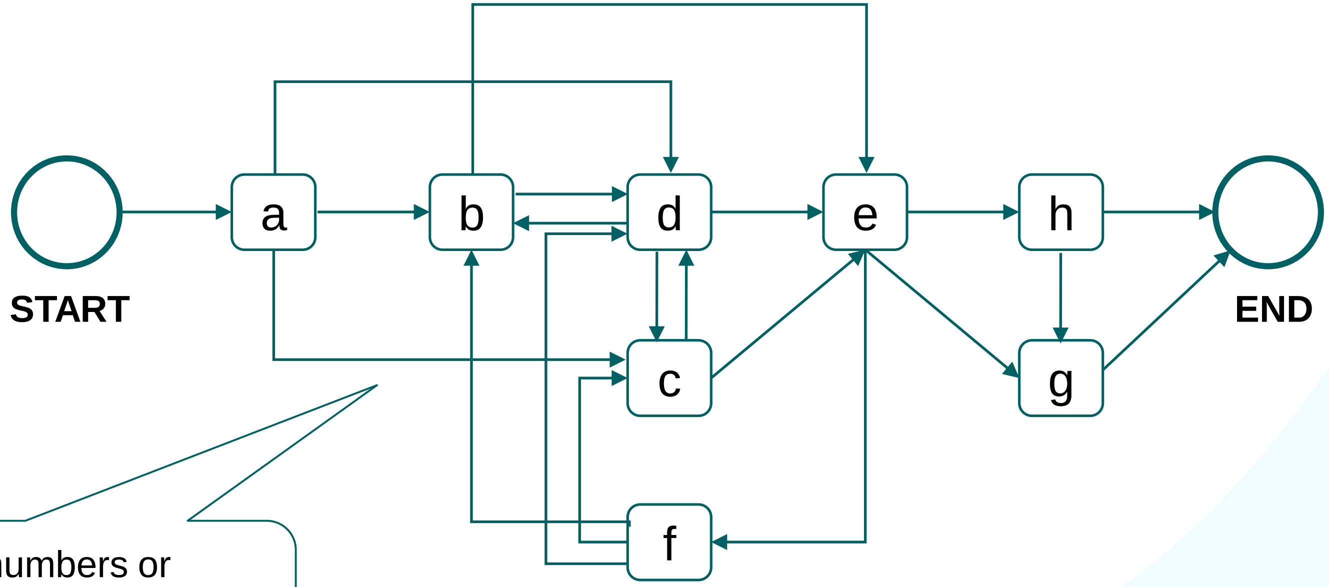
$d =$ check ticket

$e =$ decide

$f =$ reinitiate request

$g =$ pay compensation

$h =$ reject request



Often we also use numbers or width of edges to show the frequency of particular edges

Directly-Follows Graphs – Example

Assume we have a simplified log:

$$L = [\langle a, b, d, e, h\rangle, \langle a, d, c, e, g\rangle, \langle \underline{a, c}, d, e, f, b, d, e, g\rangle, \langle a, d, b, e, h\rangle, \langle \underline{a, c}, d, e, f, d, c, e, f, c, d, e, h\rangle, \langle \underline{a, c}, d, e, g\rangle]$$

$a =$ register request

$b =$ examine thoroughly

$c =$ examine casually

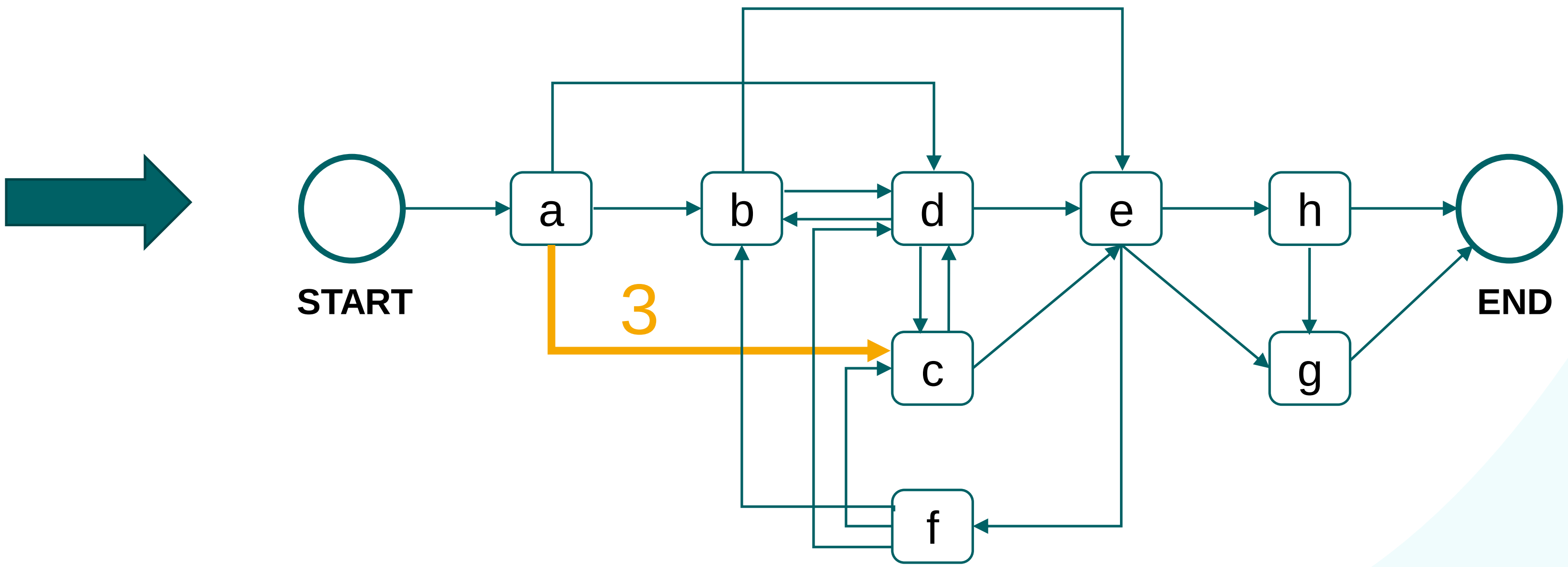
$d =$ check ticket

$e =$ decide

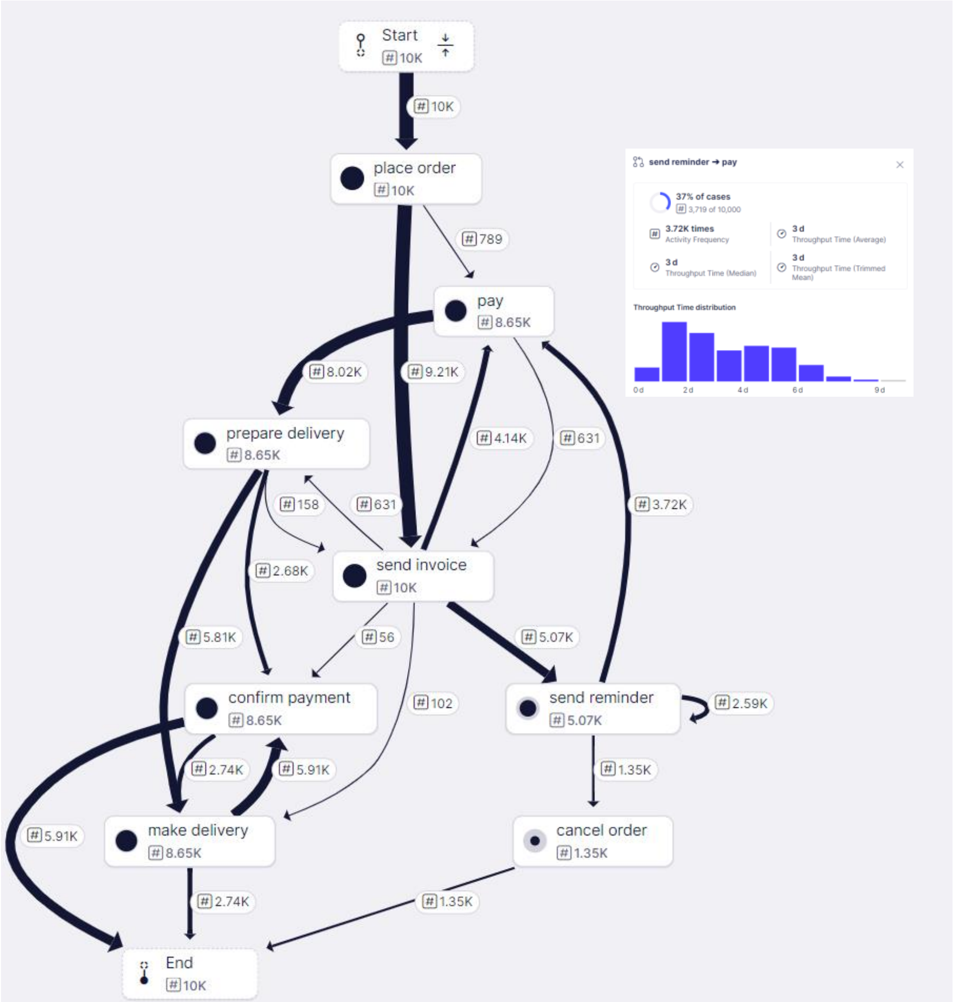
$f =$ reinitiate request

$g =$ pay compensation

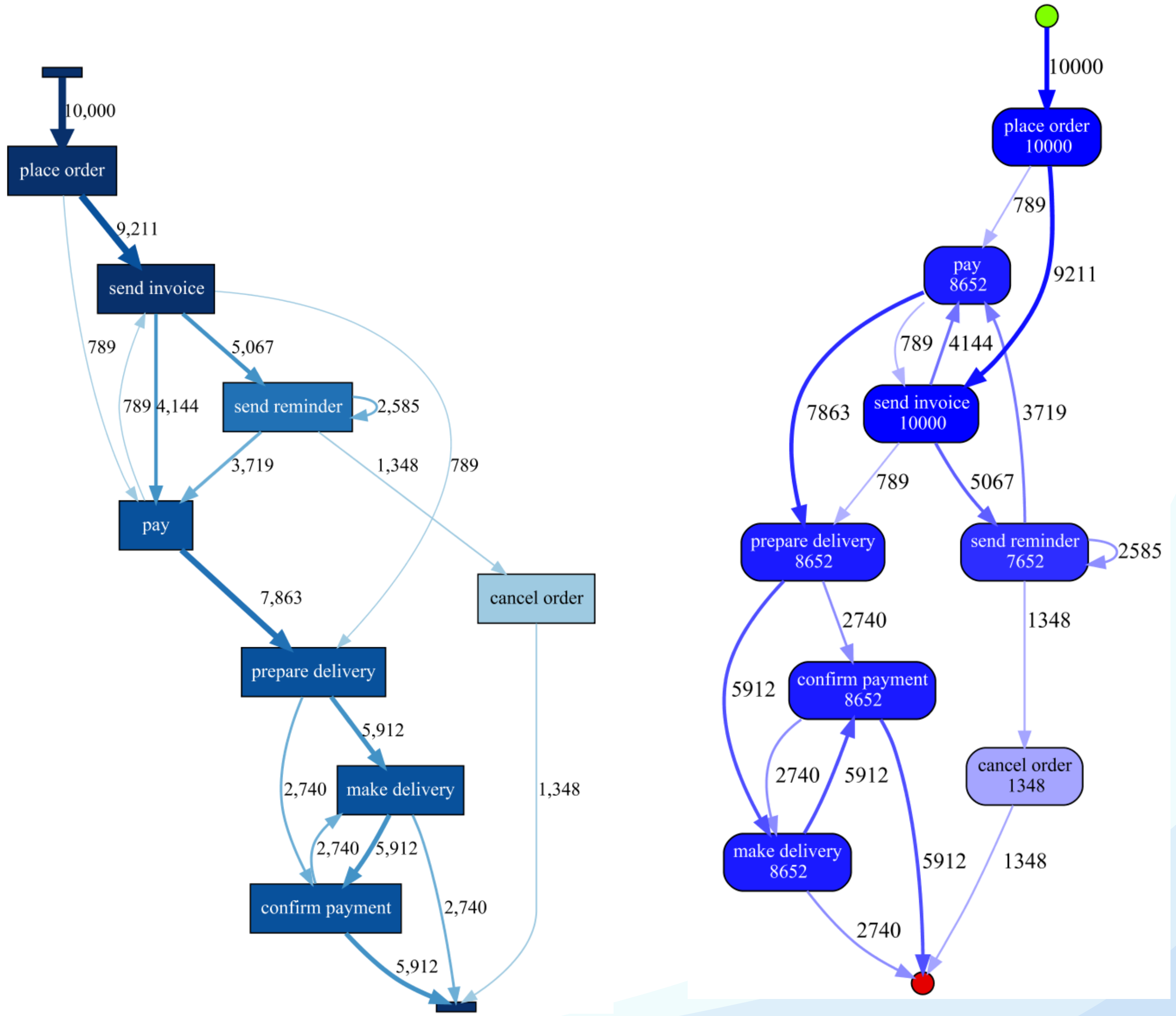
$h =$ reject request



Directly-Follows Graphs – Example Visualization in Tools



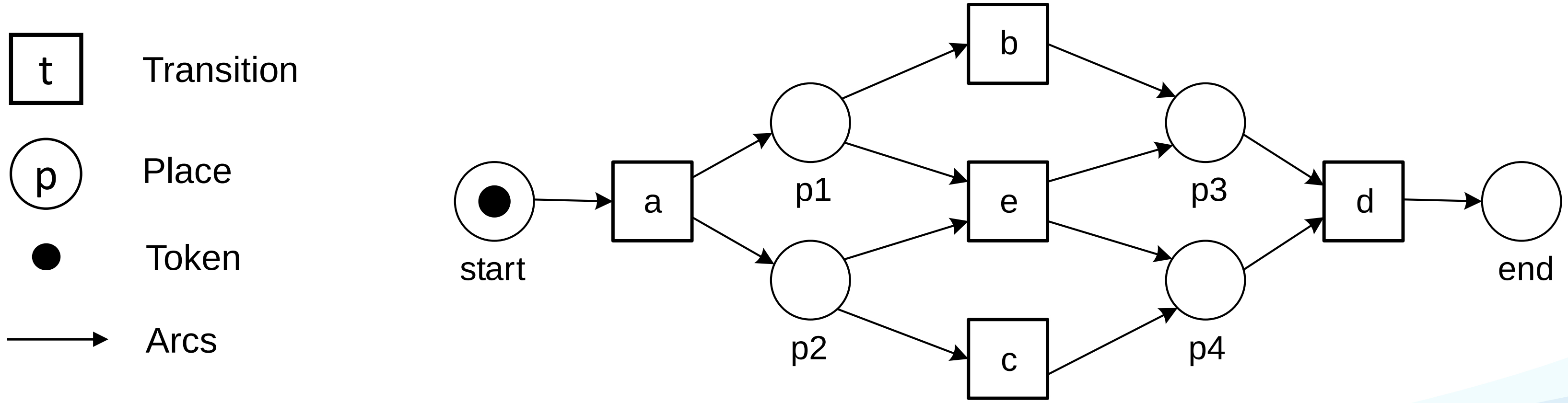
DFG in Celonis



DFGs in ProM

Petri Nets

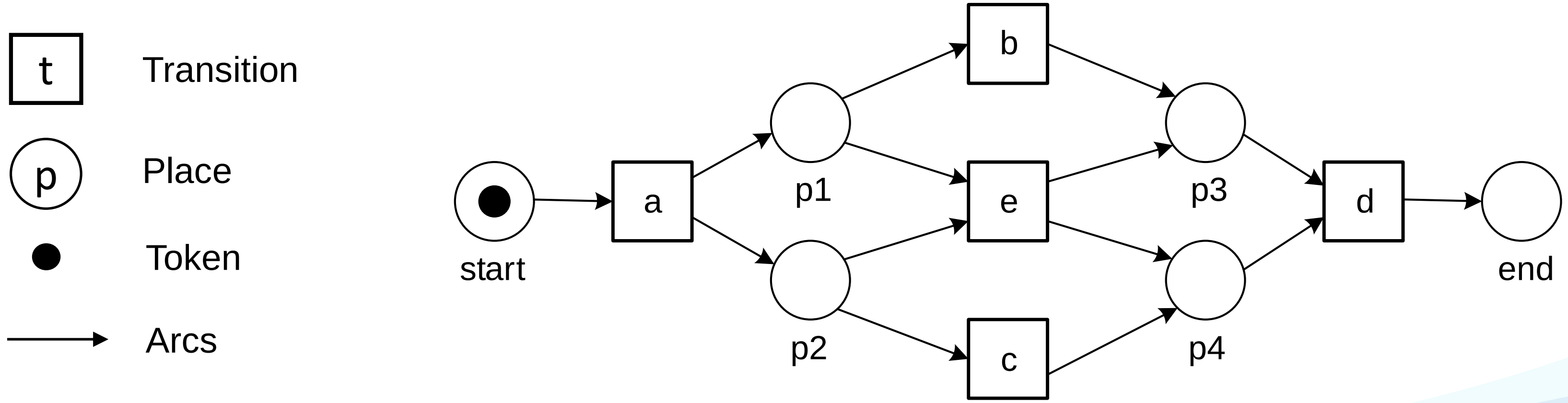
- Wide variety of application domains
- Oldest and most investigated process modeling language
- Petri net is a bipartite graph consisting of places and transitions



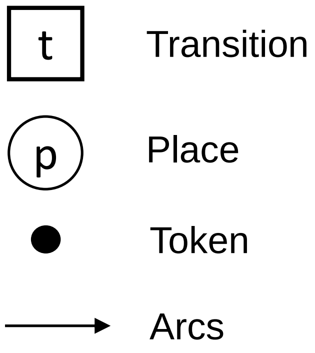
Petri Nets

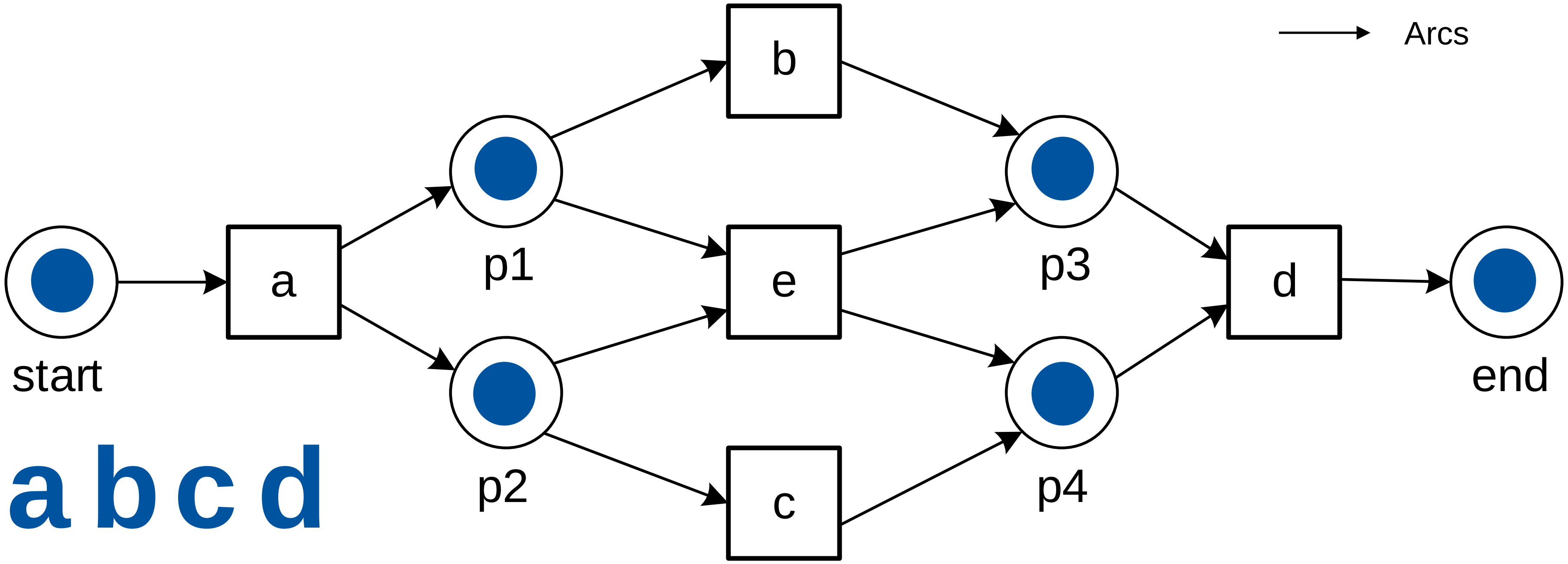
- Transitions can fire if all input places have tokens (when firing they produce tokens in all output places and consume tokens from all input places)
- In process mining we use mainly accepting Petri nets

  (accepting Petri nets have a defined START and END state)



One Possible Execution of Petri Net



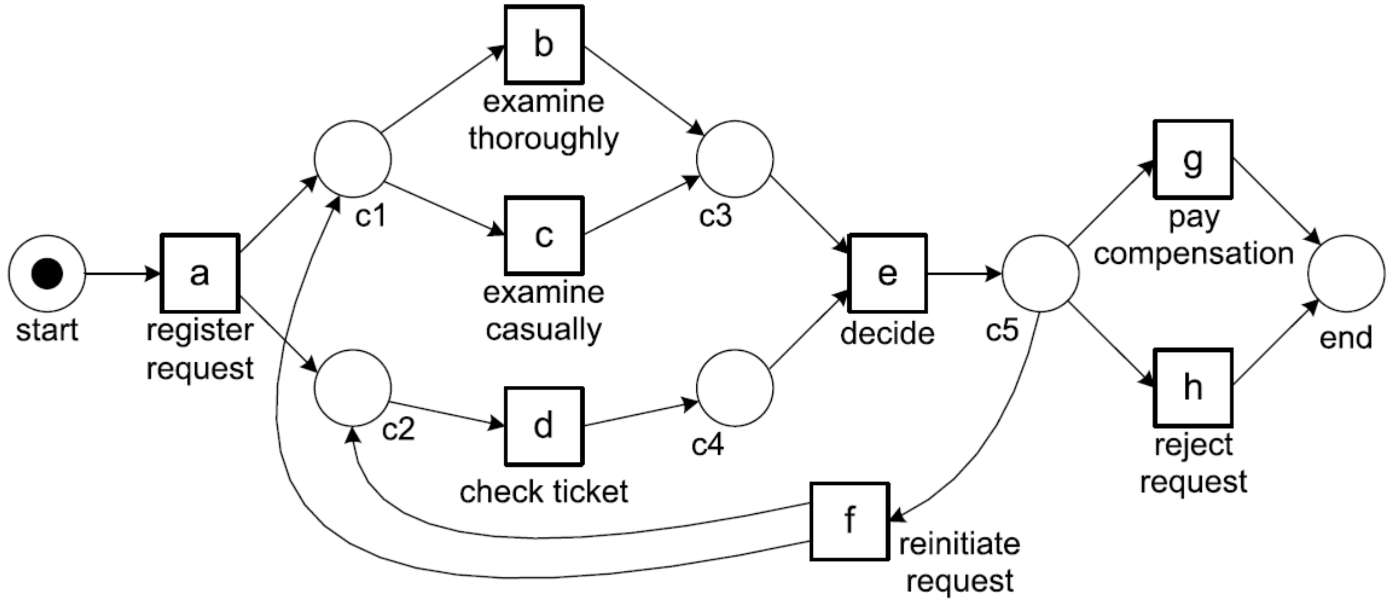


Initial marking [start], final marking [end]

Petri Nets – Example

The same log previously used to build a DFG can be used to discover the following Petri net:

$$L = [\langle a, b, d, e, h\rangle, \langle a, d, c, e, g\rangle, \langle a, c, d, e, f, b, d, e, g\rangle, \langle a, d, b, e, h\rangle, \langle a, c, d, e, f, d, c, e, f, c, d, e, h\rangle, \langle a, c, d, e, g\rangle]$$

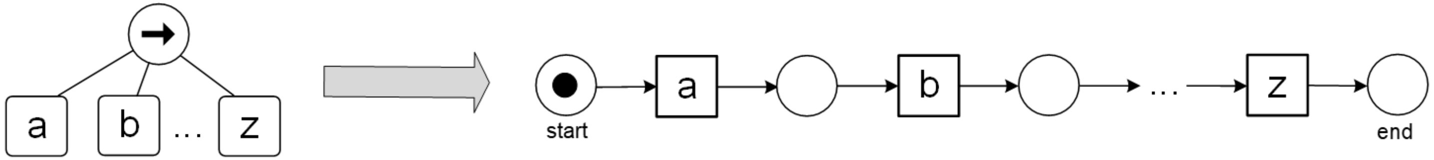


Process Tree – Four Types of Operators

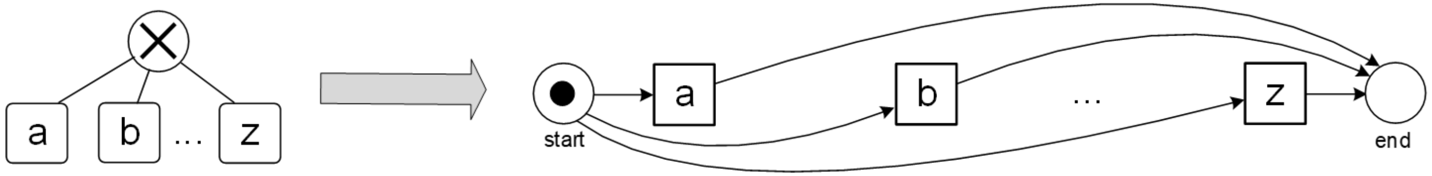
- Often used in process mining because it guarantees favourable properties simplifying many applications
- Hierarchically structures the process into behavioural blocks (represented as a tree)
- The behaviour of a block is defined by operators

Process Tree – Four Types of Operators

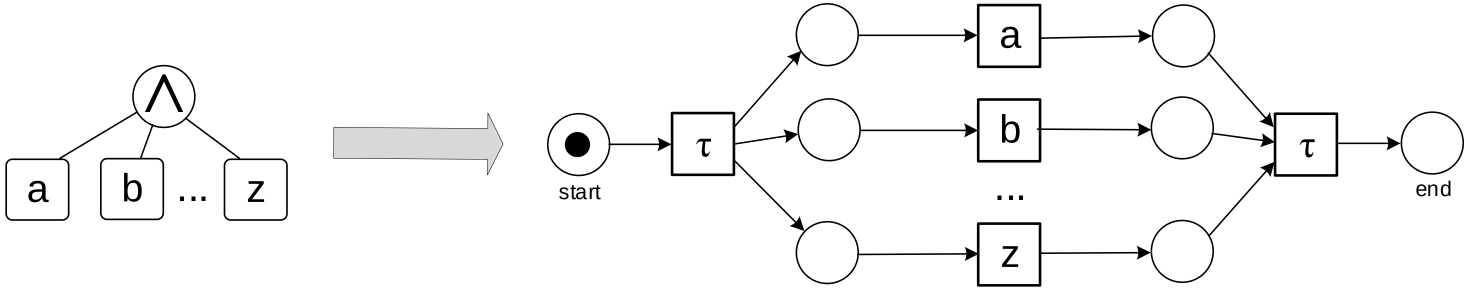
sequential compostion



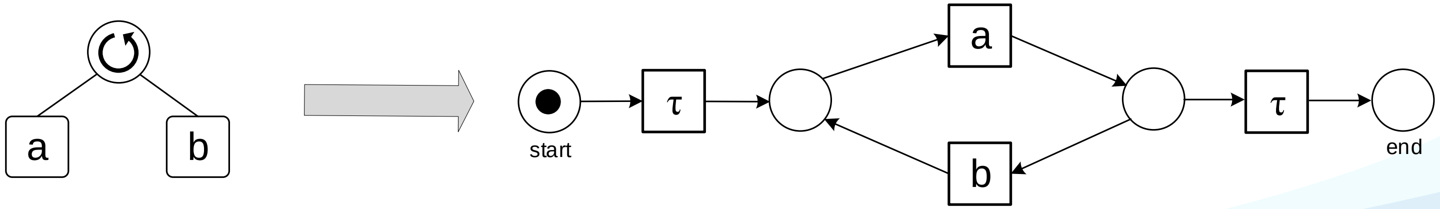
exclusive choice



parallel composition



redo loop

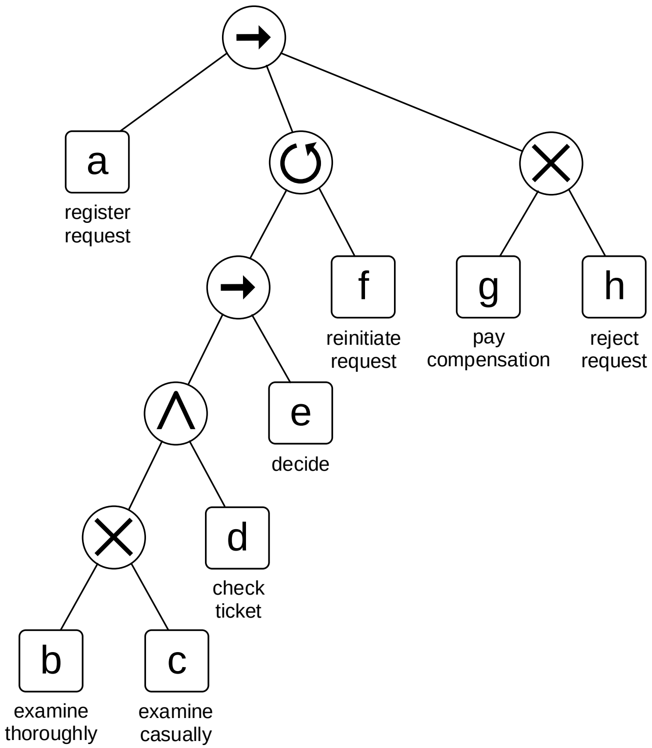


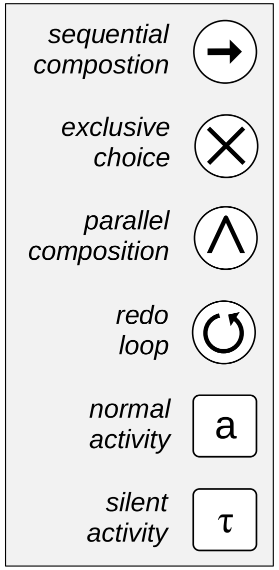
τ always refers to a silent activity, i.e., skip.

Process Tree – Example

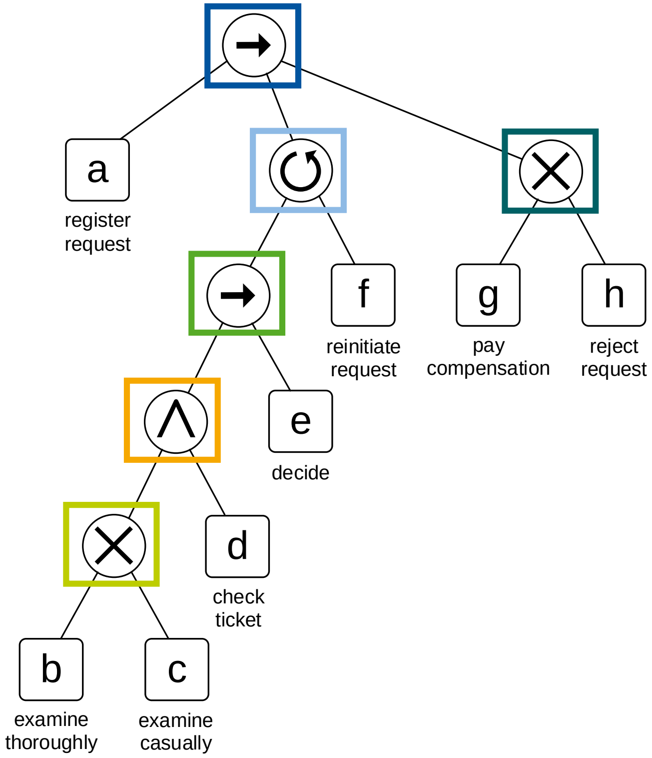
The same log previously used to build a DFG can be used to discover the following process tree:

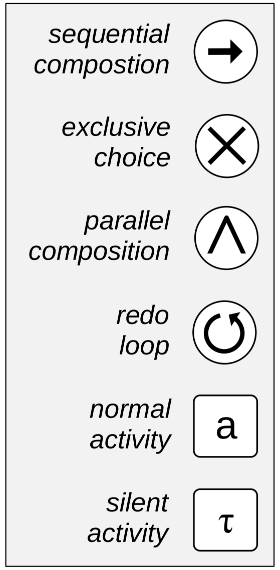
$$L = [\langle a,b,d,e,h\rangle, \langle a,d,c,e,g\rangle, \langle a,c,d,e,f,b,d,e,g\rangle, \langle a,d,b,e,h\rangle, \langle a,c,d,e,f,d,c,e,f,c,d,e,h\rangle, \langle a,c,d,e,g\rangle]$$

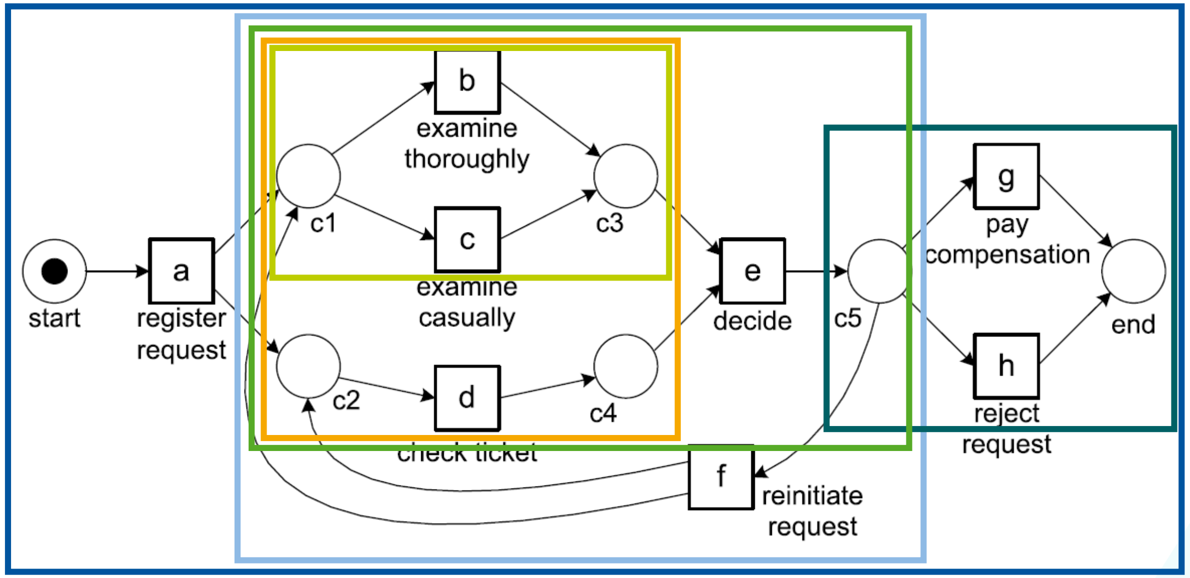




Process Tree Semantics



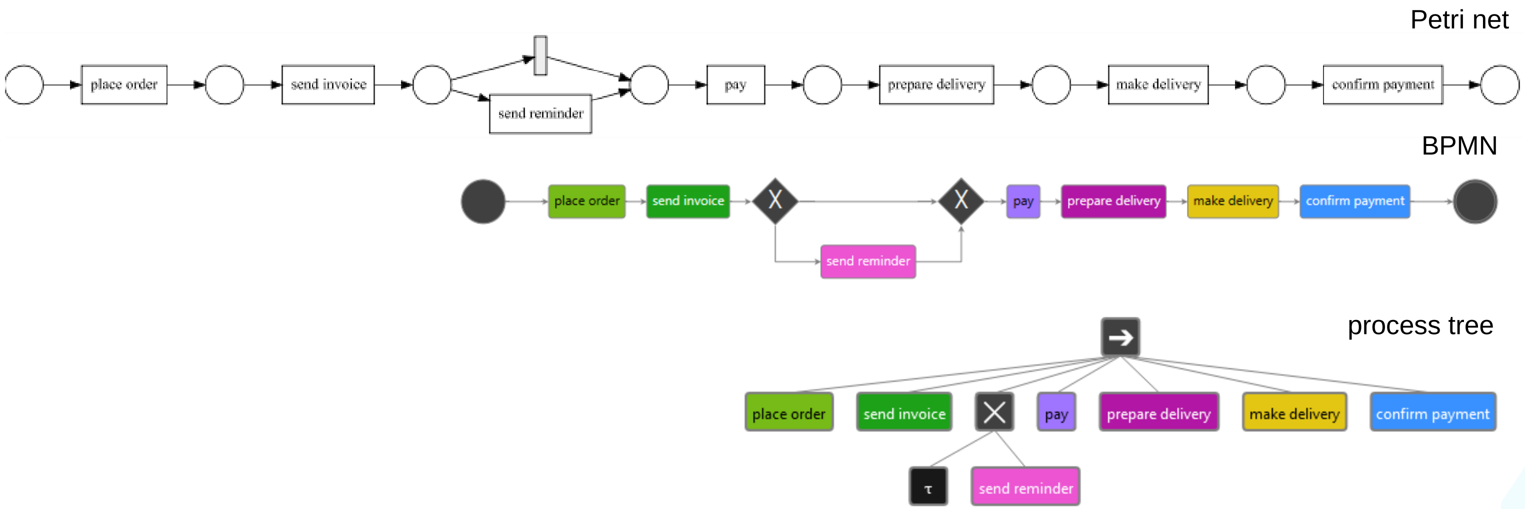




Process Models

The same process can be visualized in many ways:

- DFGs
- Petri Nets
- Process Trees
- BPMN models

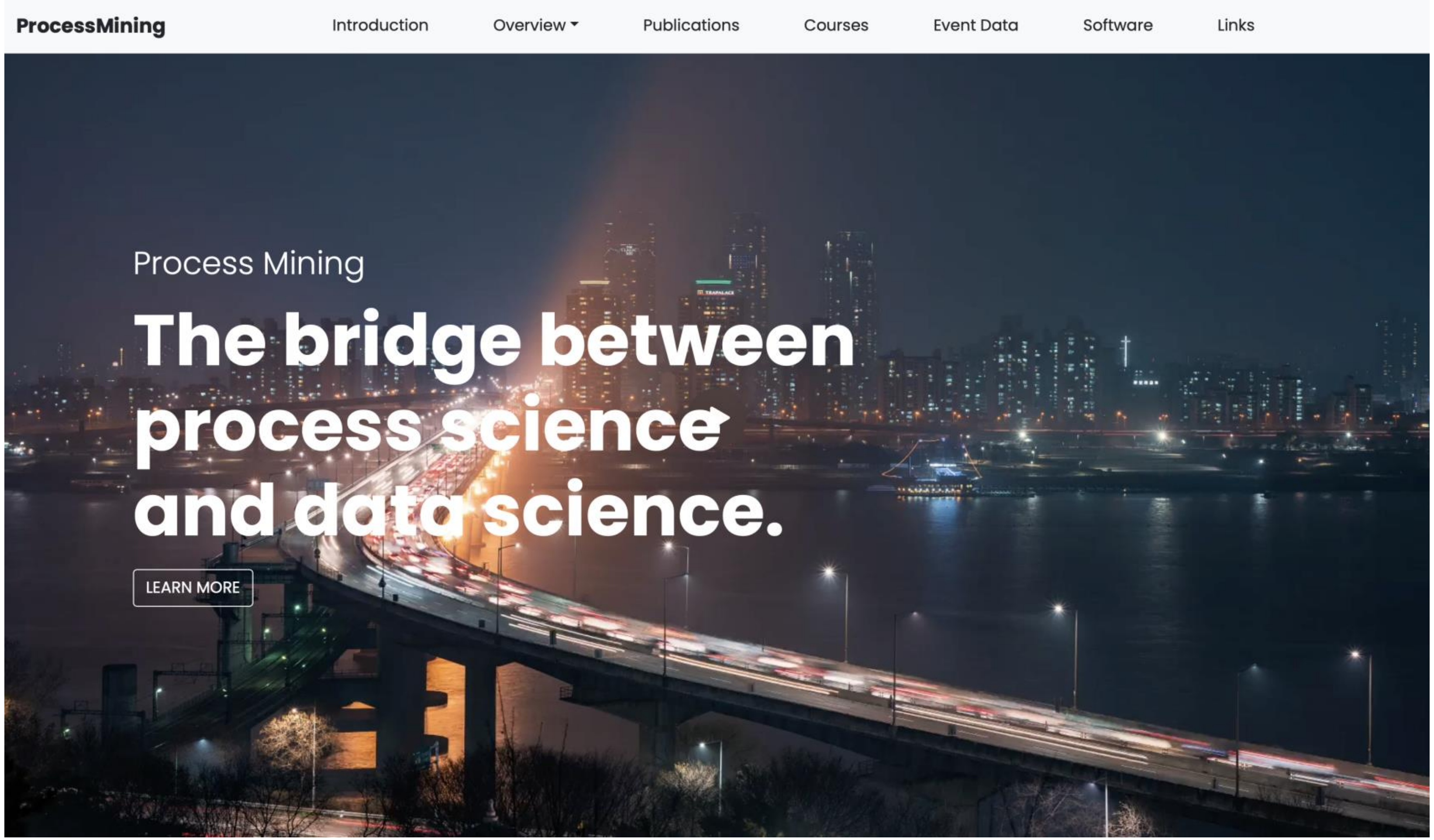


The conversion of process mining results into desired notations is relatively easy.

Introduction to Process Mining

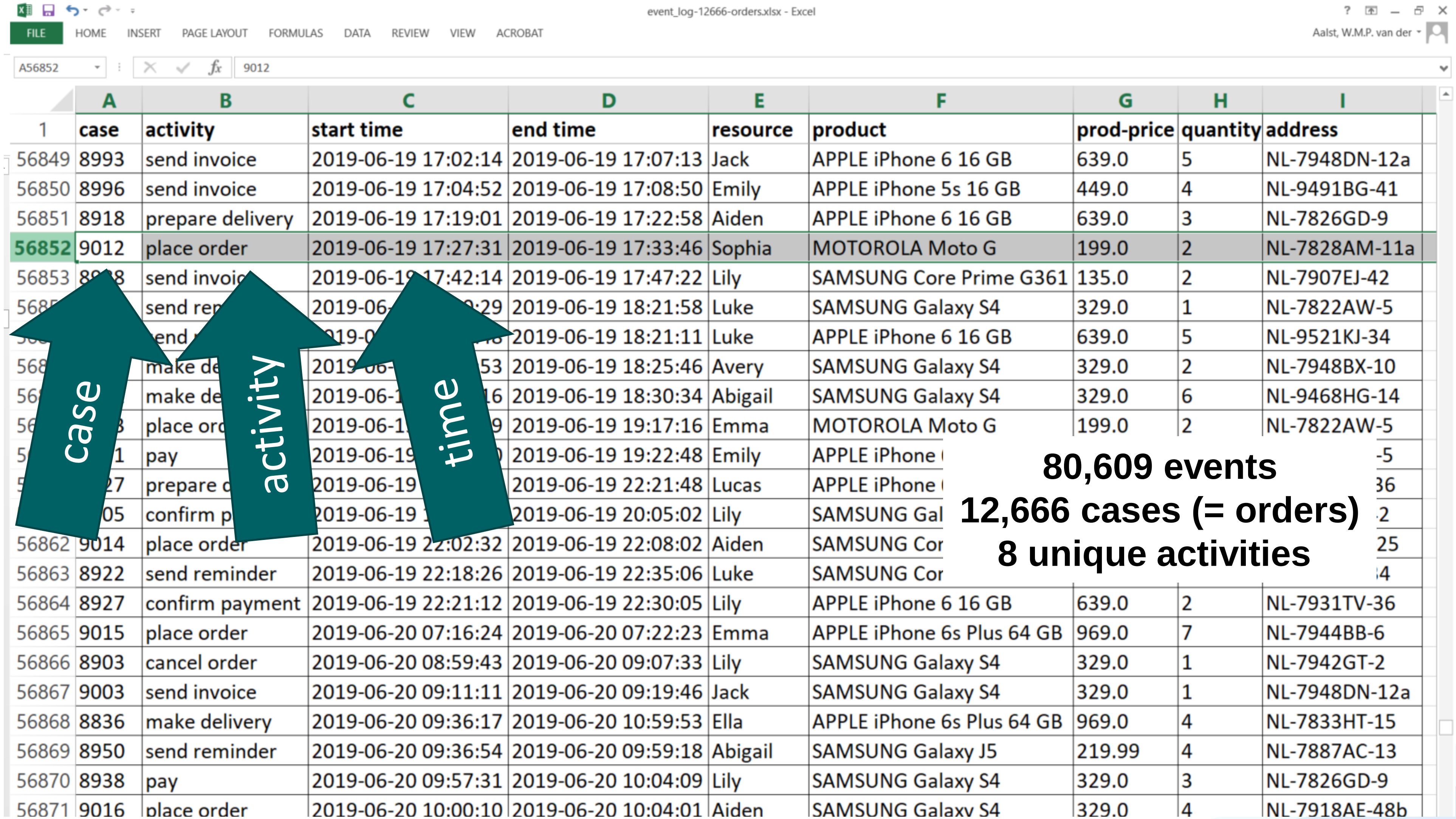
1. Process Mining and Event Data
2. Process Models
3. **Software Tools**
4. Applications

Over 40 commercial process mining tools (see processmining.org)



For learning resources and more information about tools check out processmining.org

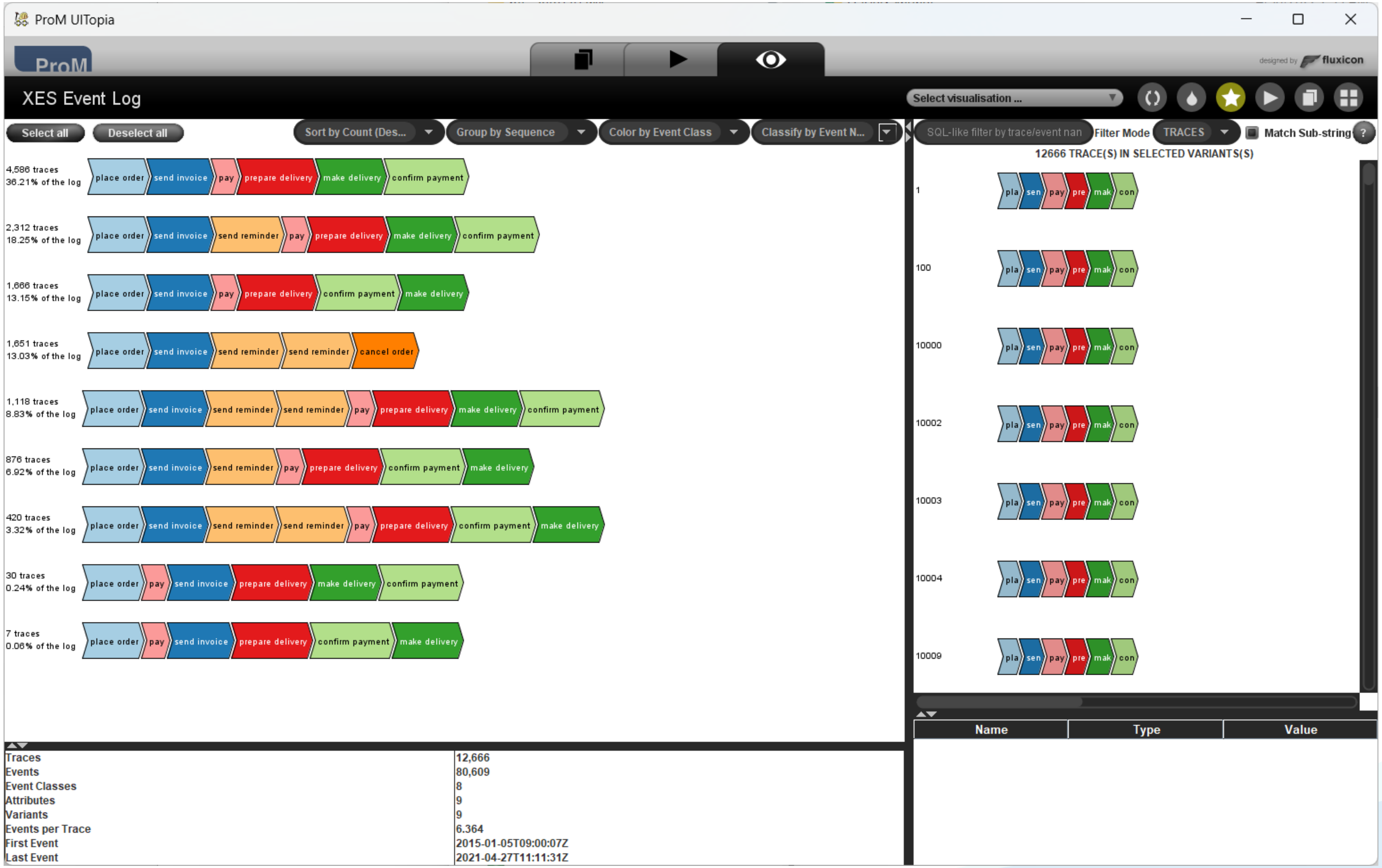
Process Mining Demo



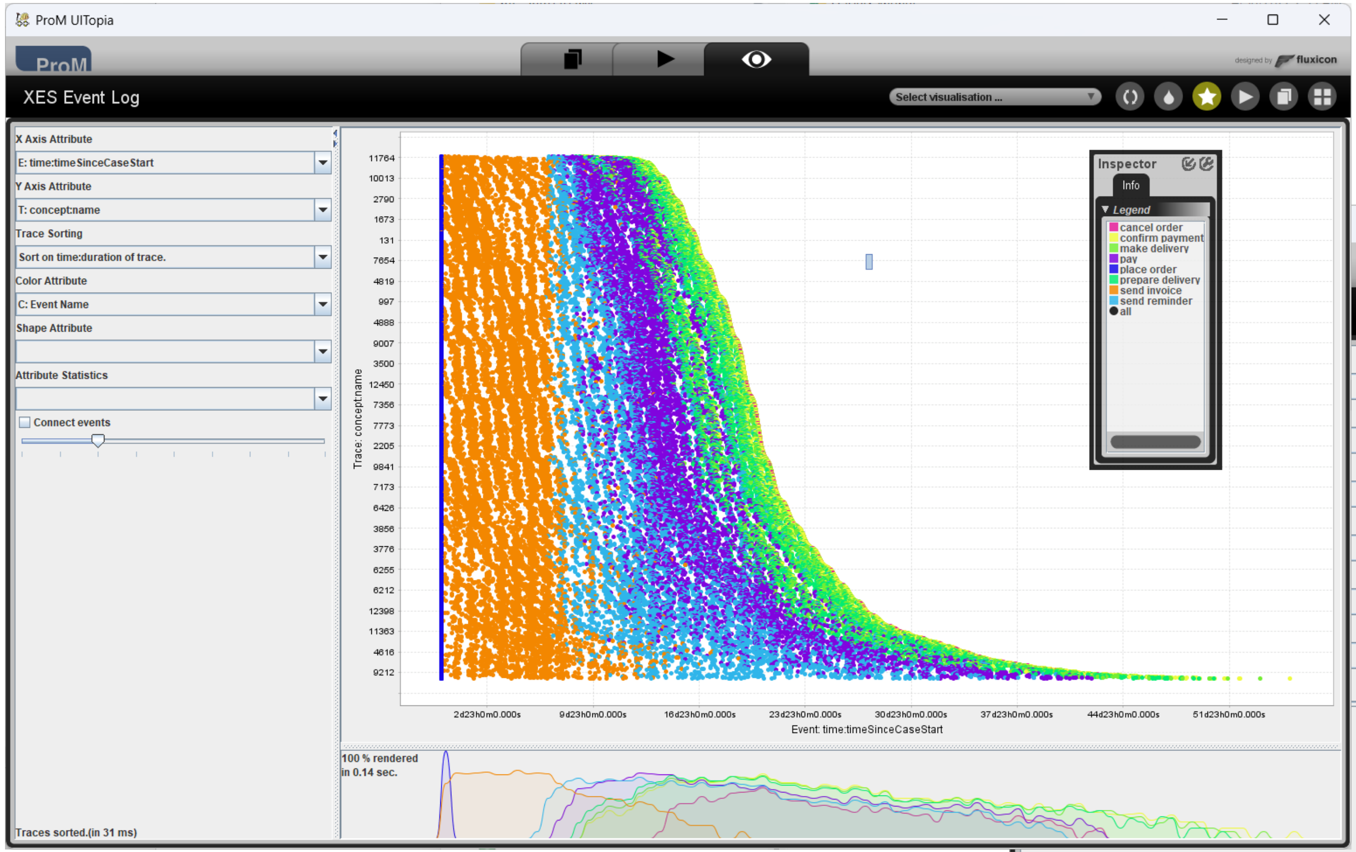
	case	activity	start time	end time	resource	product	prod-price	quantity	address
56849	8993	send invoice	2019-06-19 17:02:14	2019-06-19 17:07:13	Jack	APPLE iPhone 6 16 GB	639.0	5	NL-7948DN-12a
56850	8996	send invoice	2019-06-19 17:04:52	2019-06-19 17:08:50	Emily	APPLE iPhone 5s 16 GB	449.0	4	NL-9491BG-41
56851	8918	prepare delivery	2019-06-19 17:19:01	2019-06-19 17:22:58	Aiden	APPLE iPhone 6 16 GB	639.0	3	NL-7826GD-9
56852	9012	place order	2019-06-19 17:27:31	2019-06-19 17:33:46	Sophia	MOTOROLA Moto G	199.0	2	NL-7828AM-11a
56853	8 8	send invoi	2019-06-1 17:42:14	2019-06-19 17:47:22	Lily	SAMSUNG Core Prime G361	135.0	2	NL-7907EJ-42
568		send re	2019-06- 29	2019-06-19 18:21:58	Luke	SAMSUNG Galaxy S4	329.0	1	NL-7822AW-5
		end	019-0	2019-06-19 18:21:11	Luke	APPLE iPhone 6 16 GB	639.0	5	NL-9521KJ-34
568		make de	2019 53	2019-06-19 18:25:46	Avery	SAMSUNG Galaxy S4	329.0	2	NL-7948BX-10
56		make de	2019-06-1 16	2019-06-19 18:30:34	Abigail	SAMSUNG Galaxy S4	329.0	6	NL-9468HG-14
56	3	place or	2019-06-1 9	2019-06-19 19:17:16	Emma	MOTOROLA Moto G	199.0	2	NL-7822AW-5
5	1	pay	2019-06-19	2019-06-19 19:22:48	Emily	APPLE iPhone			-5
5	27	prepare	2019-06-19	2019-06-19 22:21:48	Lucas	APPLE iPhone			36
	05	confirm p	2019-06-19	2019-06-19 20:05:02	Lily	SAMSUNG Gal			2
56862	9014	place order	2019-06-19 22:02:32	2019-06-19 22:08:02	Aiden	SAMSUNG Cor			25
56863	8922	send reminder	2019-06-19 22:18:26	2019-06-19 22:35:06	Luke	SAMSUNG Cor			4
56864	8927	confirm payment	2019-06-19 22:21:12	2019-06-19 22:30:05	Lily	APPLE iPhone 6 16 GB	639.0	2	NL-7931TV-36
56865	9015	place order	2019-06-20 07:16:24	2019-06-20 07:22:23	Emma	APPLE iPhone 6s Plus 64 GB	969.0	7	NL-7944BB-6
56866	8903	cancel order	2019-06-20 08:59:43	2019-06-20 09:07:33	Lily	SAMSUNG Galaxy S4	329.0	1	NL-7942GT-2
56867	9003	send invoice	2019-06-20 09:11:11	2019-06-20 09:19:46	Jack	SAMSUNG Galaxy S4	329.0	1	NL-7948DN-12a
56868	8836	make delivery	2019-06-20 09:36:17	2019-06-20 10:59:53	Ella	APPLE iPhone 6s Plus 64 GB	969.0	4	NL-7833HT-15
56869	8950	send reminder	2019-06-20 09:36:54	2019-06-20 09:59:18	Abigail	SAMSUNG Galaxy J5	219.99	4	NL-7887AC-13
56870	8938	pay	2019-06-20 09:57:31	2019-06-20 10:04:09	Lily	SAMSUNG Galaxy S4	329.0	3	NL-7826GD-9
56871	9016	place order	2019-06-20 10:00:10	2019-06-20 10:04:01	Aiden	SAMSUNG Galaxy S4	329.0	4	NL-7918AE-48b

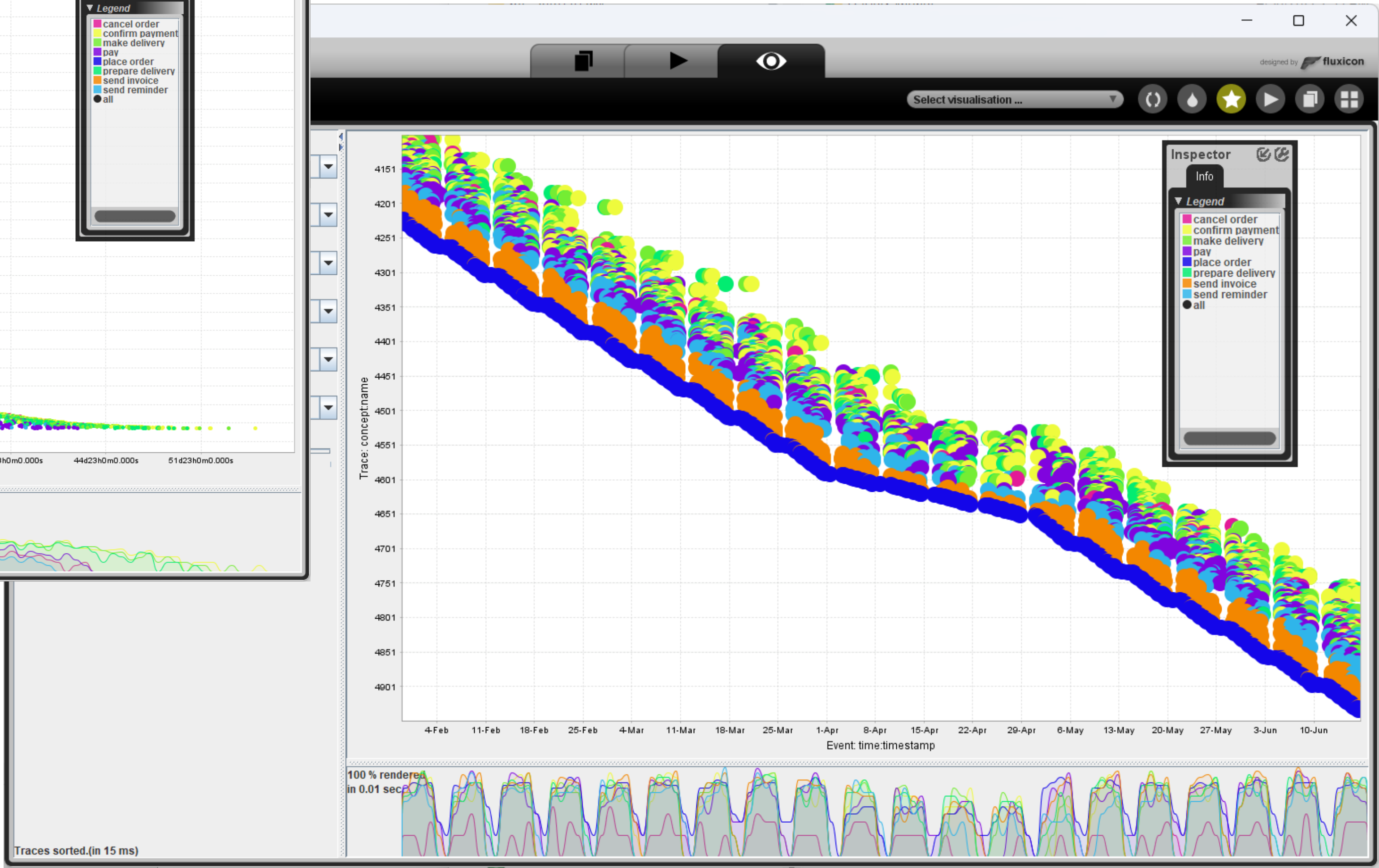
80,609 events
12,666 cases (= orders)
8 unique activities

Process Mining Demo – ProM



Process Mining Demo – ProM

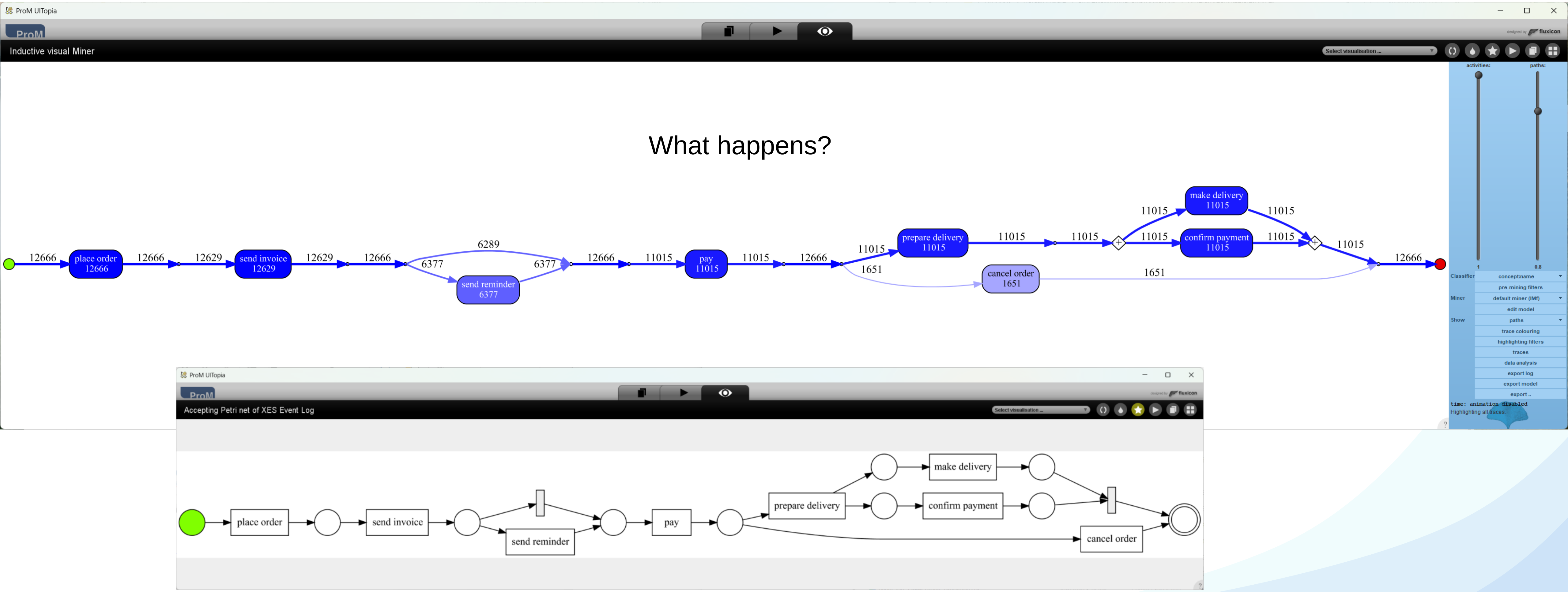




80,609 dots, one for each event

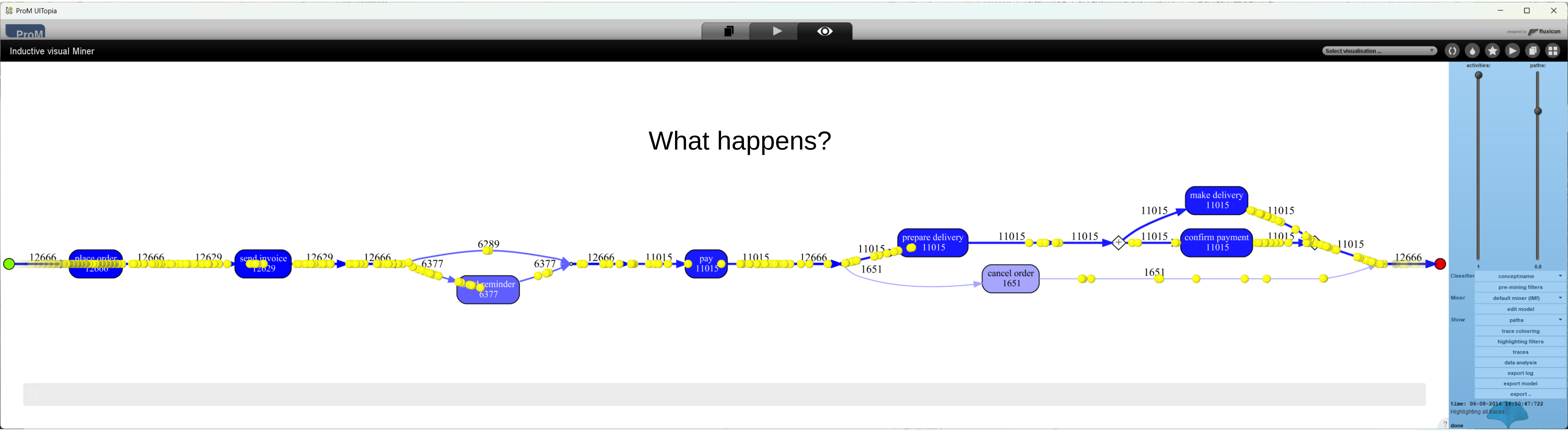
Process Mining Demo – ProM

Process Model Discovered Using the Inductive Miner



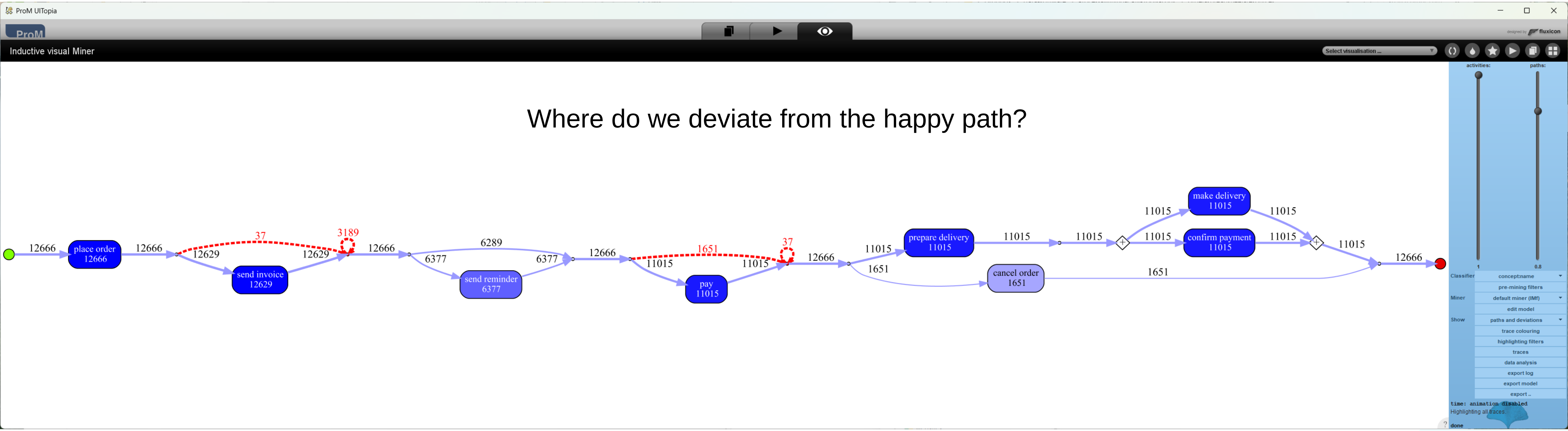
Process Mining Demo – ProM

Process Model Discovered Using the Inductive Miner



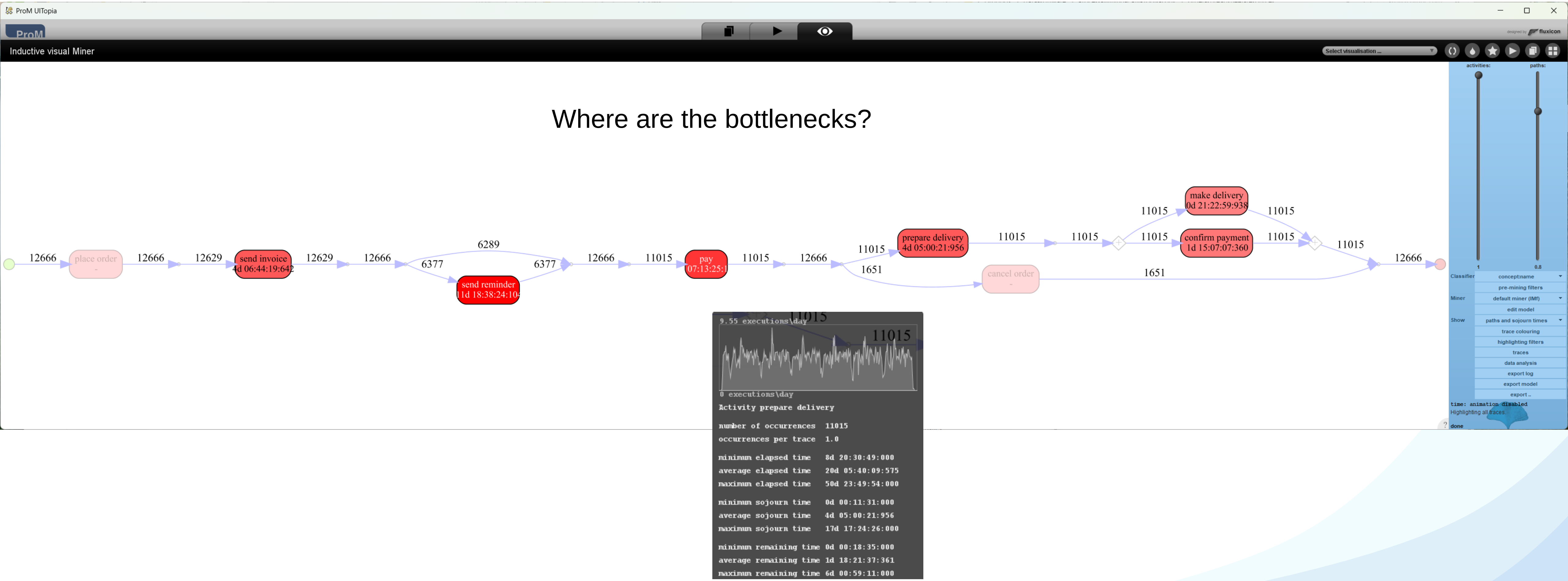
Process Mining Demo – ProM

Using Conformance Checking to See Process Deviations



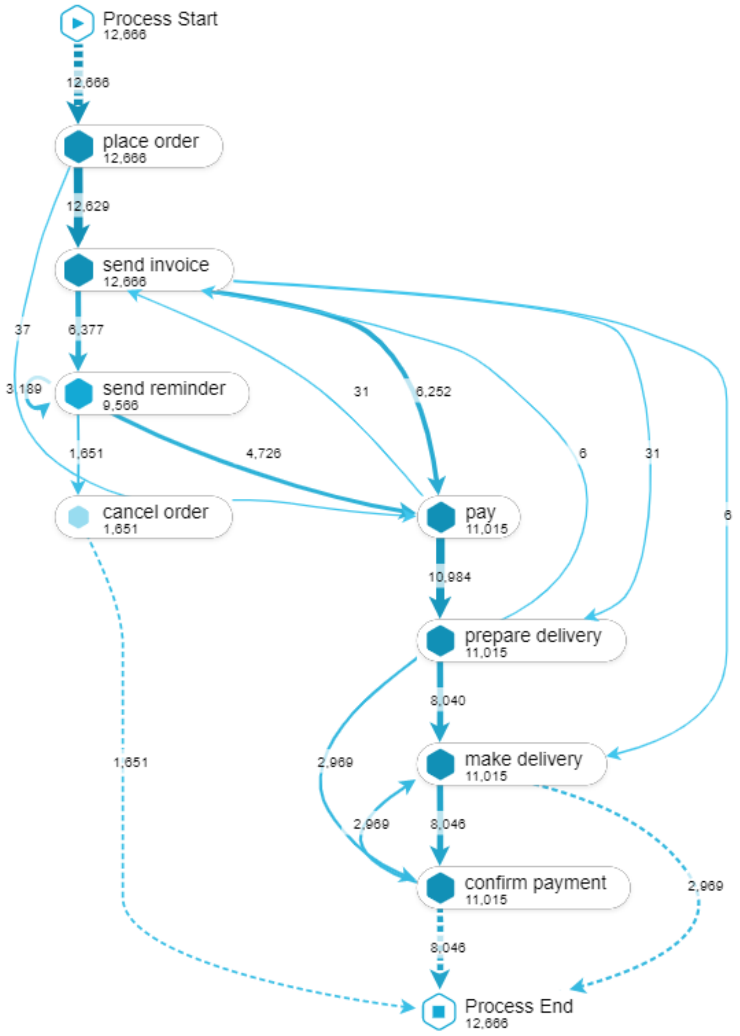
Process Mining Demo – ProM

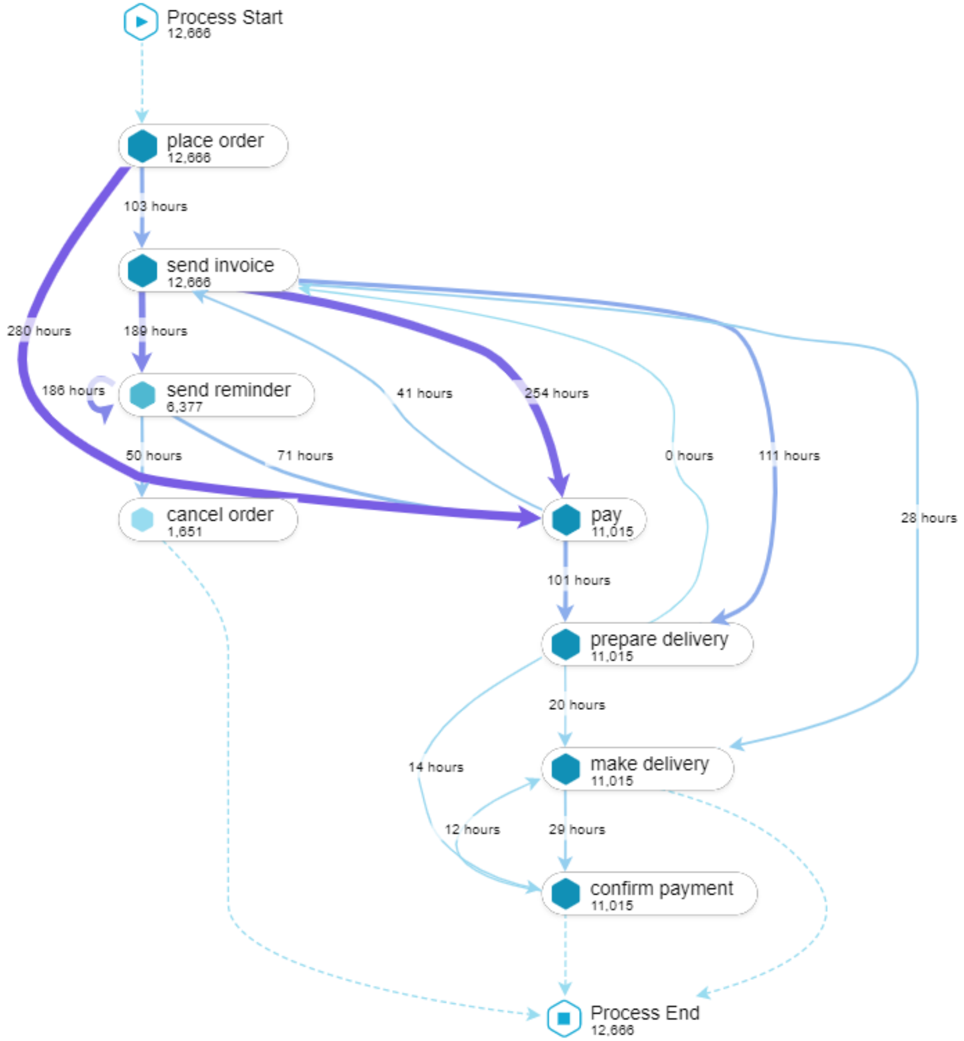
Bottleneck Analysis – Enriching the Model with Performance Information



Process Mining Demo – Celonis

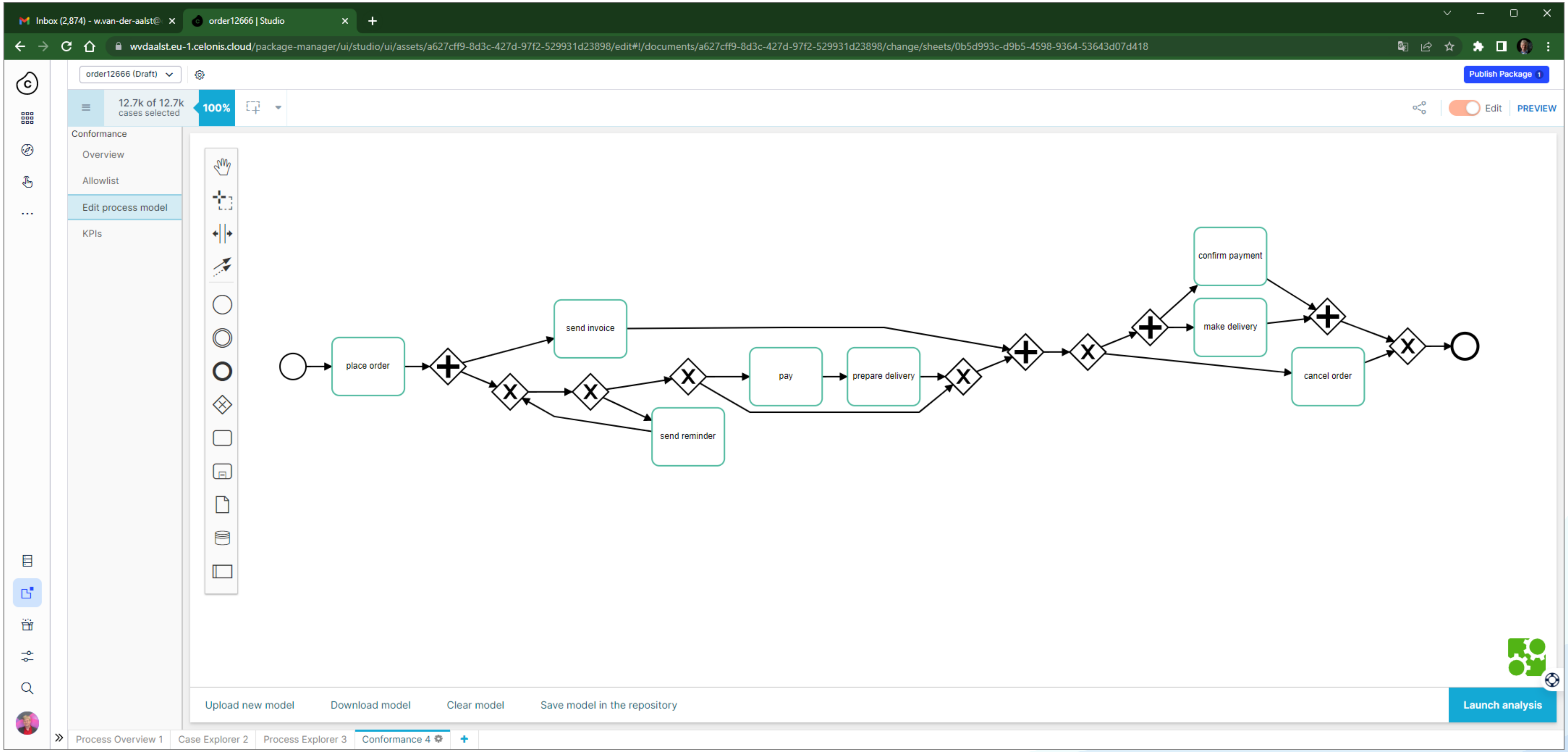
Frequencies and Times (Using the same event data)





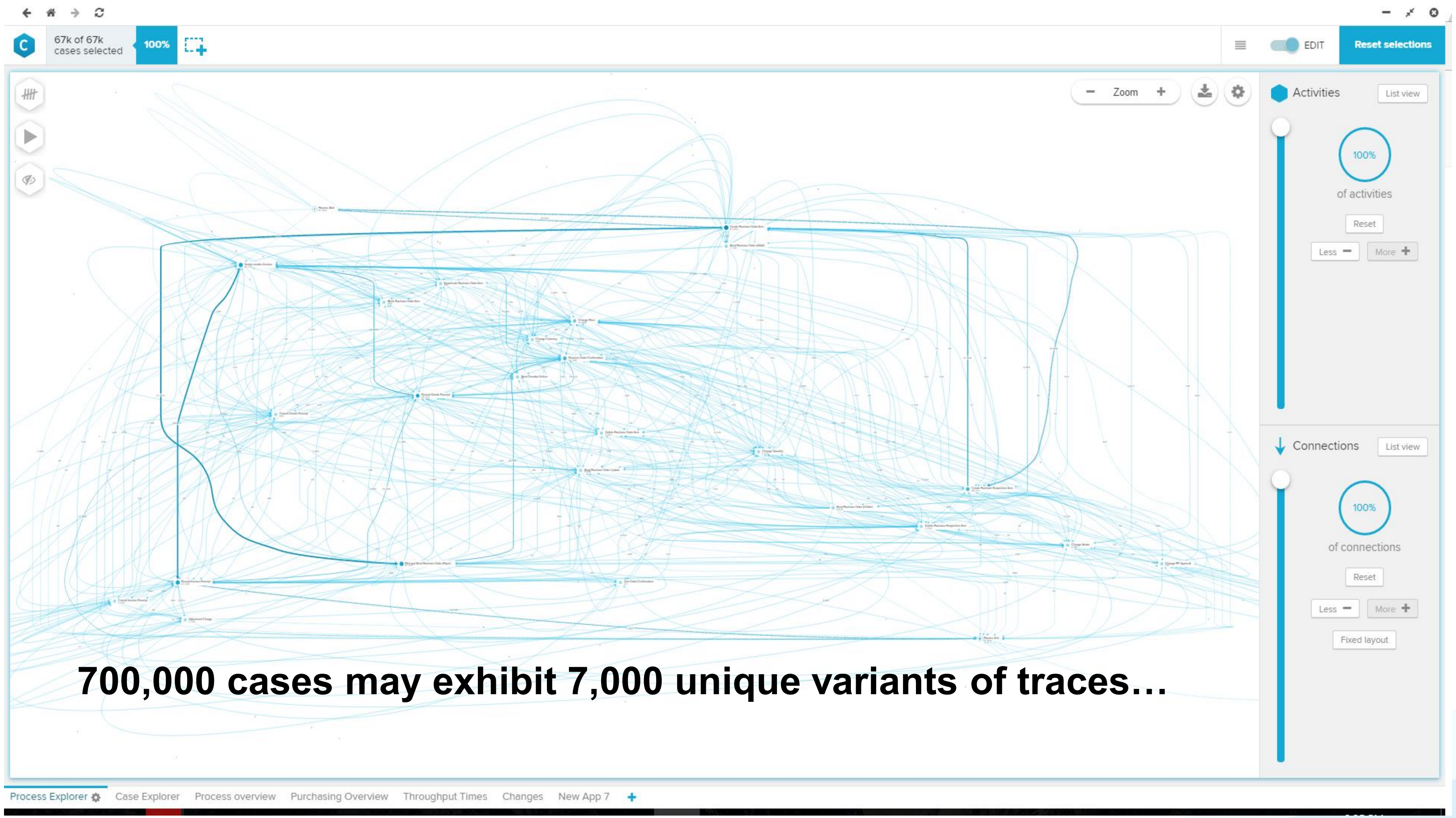
Process Mining Demo – Celonis

Process Model Discovered Using the Inductive Mining Algorithm



Reality Is Not So Simple

Real Processes May Look Like This



Examples of Tools

- **ProM** is the most complete open-source process tool that served as an example for all later tools
  - Download from https://promtools.org/
- **Celonis** is the leading commercial tool (there are 40+ other commercial tools)
  - Get via https://signup.celonis.com/
  - Free course: https://www.celonis.com/wils-process-mining-class/
- In this course, we will mostly use **PM4Py**
  - Python-based process mining library
  - Easy to combine with other data science techniques
  - Collaborative effort PADS@RWTH and Fraunhofer FIT

Introduction to Process Mining

1. Process Mining and Event Data

2. Process Models

3. Software Tools

4. **Applications**

Process Mining is Used in All Domains!

- finance and insurance (Rabobank, Wells Fargo, ADAC, APG, Suncorp, VTB, etc.)
- logistics and transport (Uber, Deutsche Bahn, Lufthansa, Airbus, Vanderlande, etc.)
- production (ABB, Siemens, BMW, Fiat, Bosch, AkzoNobel, Bayer, Neste, etc.)
- healthcare, biomedicine, and pharmacy (Uniklinik RWTH Aachen, Charite University Hospital, GE Healthcare, Philips, Medtronic, Pfizer, Bayer, AstraZeneca, etc.)
- telecom (Deutsche Telekom, Vodafone, A1 Telekom Austria, Telekom Italia, etc.)
- food and retail (Edeka, MediaMarkt, Globus, Zalando, AB InBev, etc.)
- energy (Uniper, Chevron, Shell, BP, E.ON, etc.)
- IT services (Dell, Xerox, IBM, Nokia, ServiceNow, etc.)
- consultancy (Deloitte, Ernst & Young, KPMG, PwC, etc.)

Process Mining Example – Airports

- Why do bags miss a plane?
- Why do I need to wait so long for my bags?
- When and why does the system break down?
- Am I using the available capacity properly?

Part II: Unsupervised Process Mining

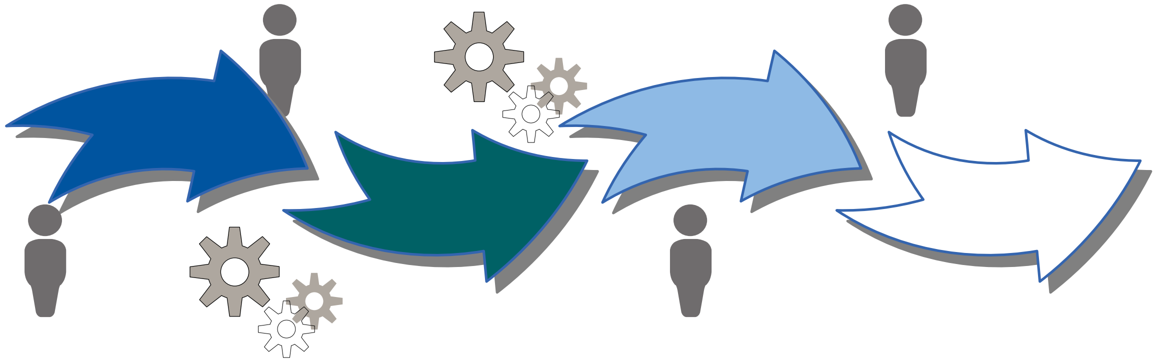
Process Discovery

Unsupervised Process Mining

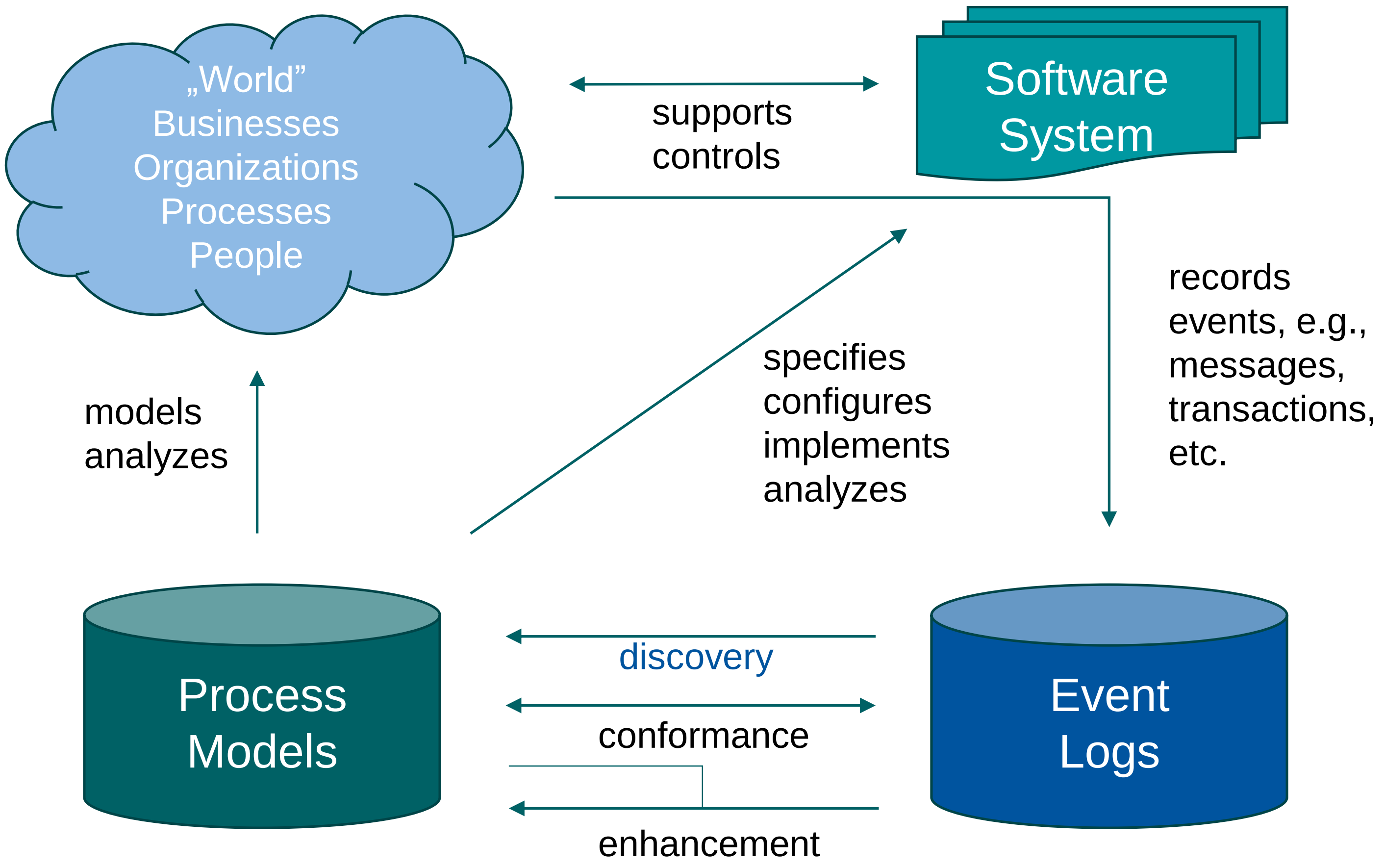
1. **Process Discovery**

2. Bottom-Up Discovery (very brief)

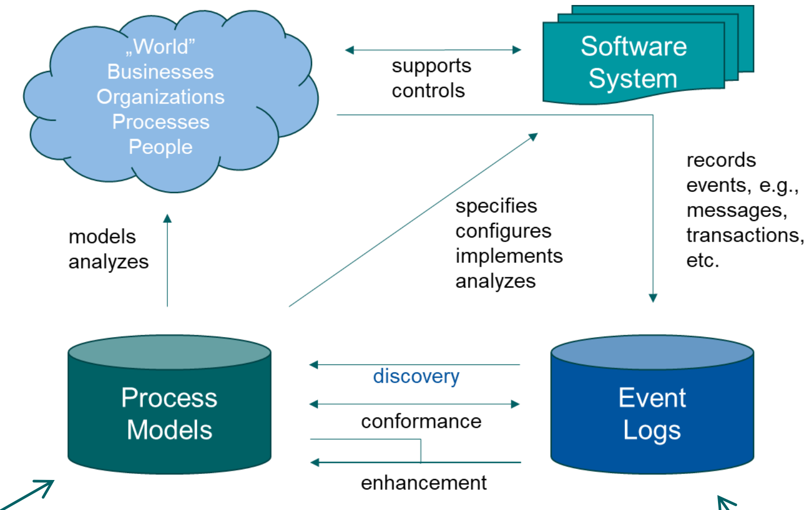
3. Top-Down Discovery (IM)



Positioning Process Discovery



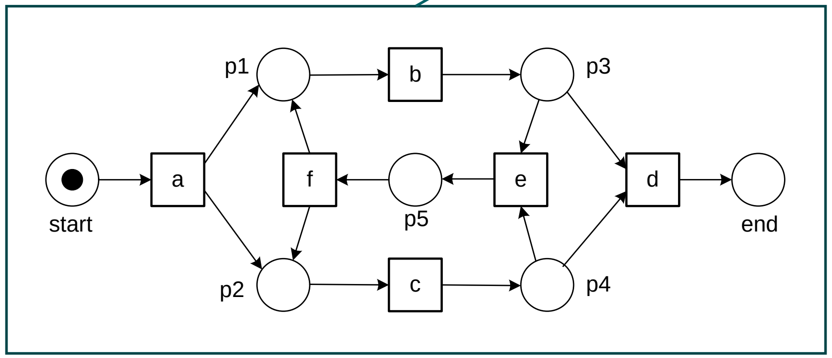
Let’s Consider the Simplest Setting Possible



[1]

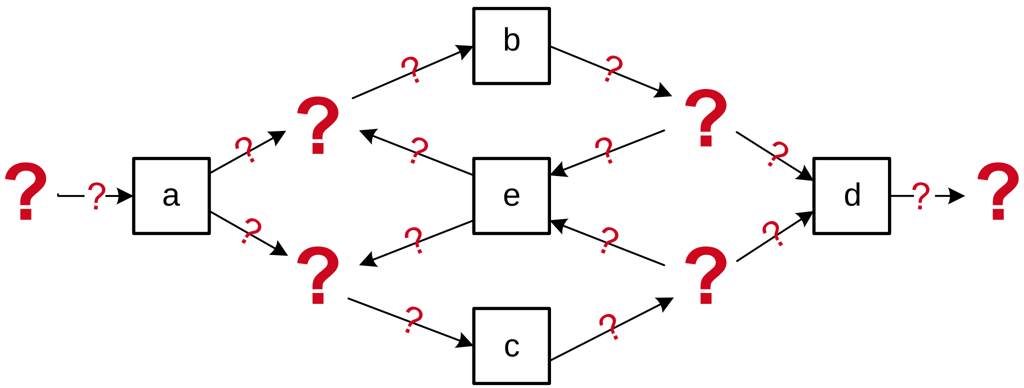
A process model describes a (possibly infinite) set of traces.

A simplified event log is just a multiset of traces, and each trace is a sequence of activities.

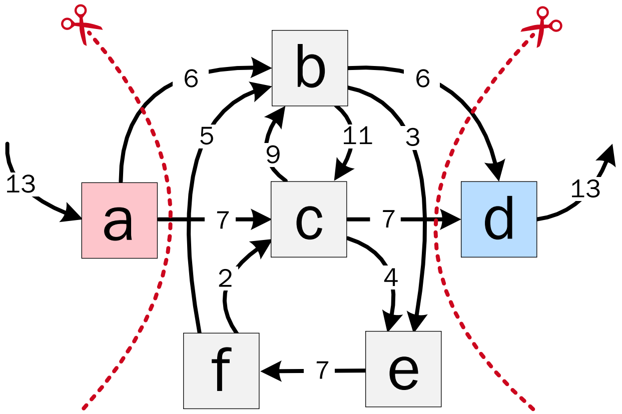


$$Log = [\langle a,b,c,d\rangle^3, \langle a,c,b,d\rangle^4, \langle a,b,c,e,f,b,c,d\rangle^2, \langle a,b,c,e,f,c,b,d\rangle, \langle a,c,b,e,f,b,c,d\rangle^2, \langle a,c,b,e,f,b,c,e,f,c,b,d\rangle]$$

Process Discovery Approaches



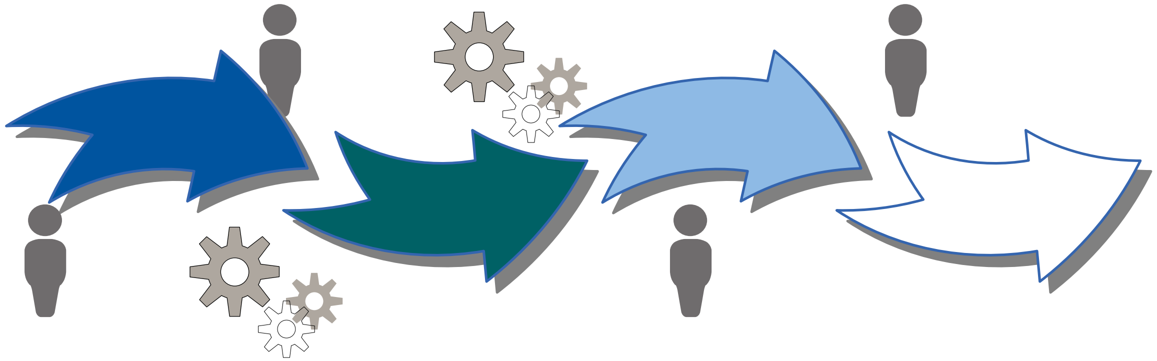
Bottom-Up



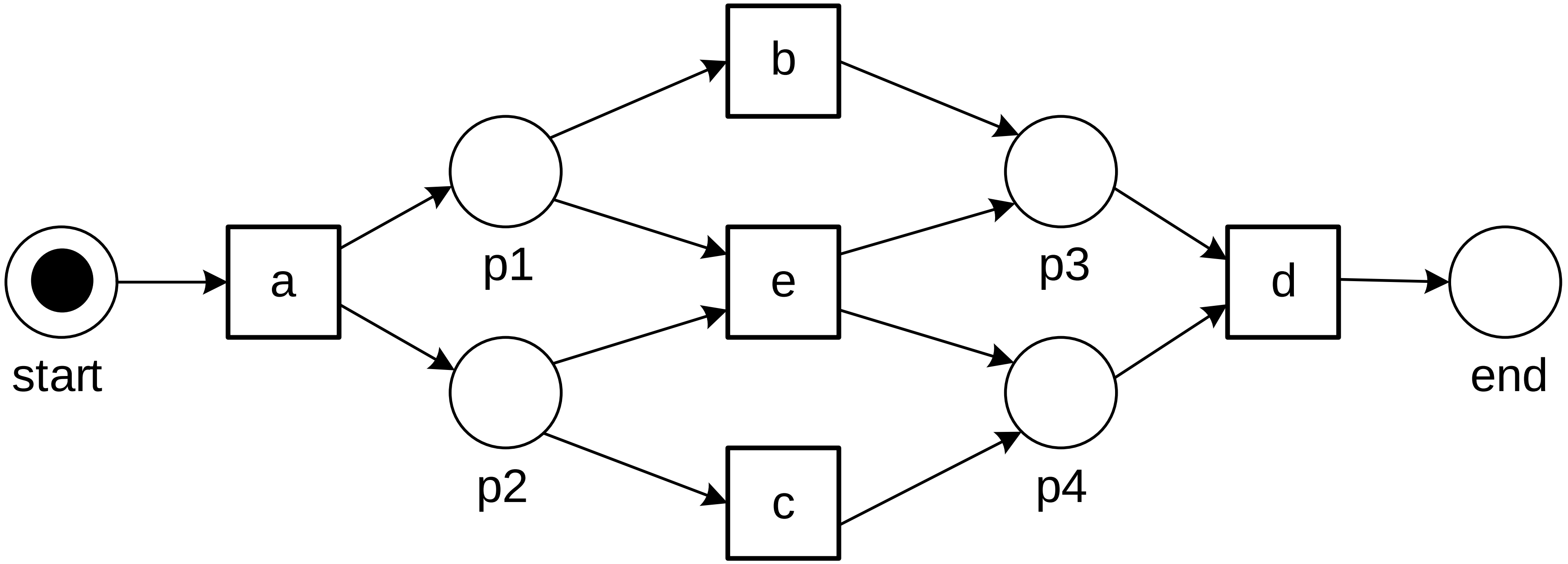
Top-Down

Unsupervised Process Mining

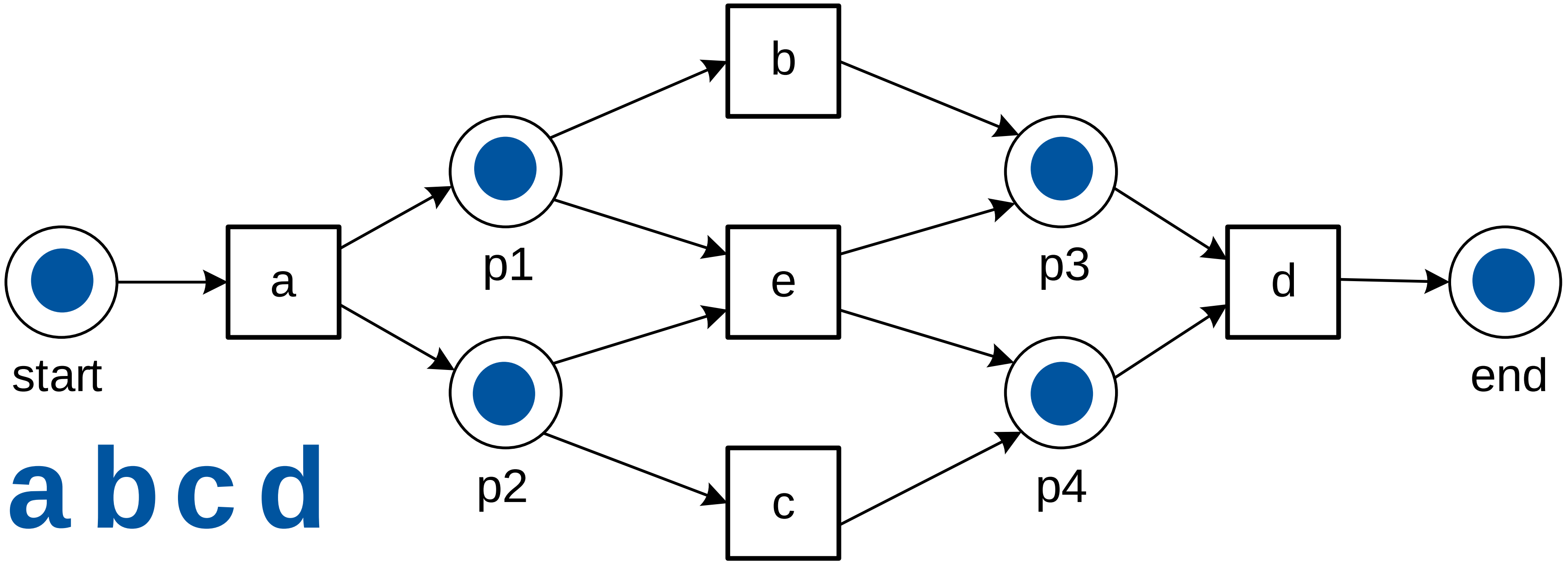
1. Process Discovery
2. **Bottom-Up Discovery**
3. Top-Down Discovery



Explaining Bottom-Up Approach Using Accepting Petri Nets



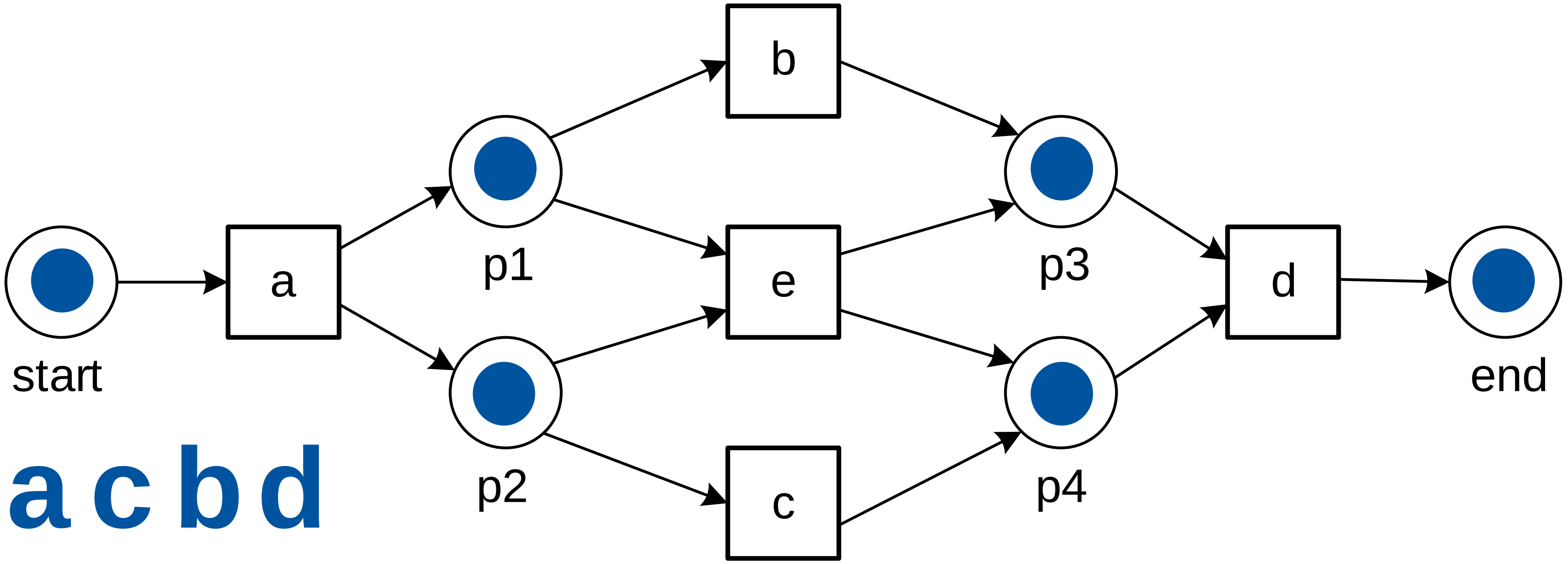
Example Trace (1/3)



a b c d

Initial marking [start], final marking [end]

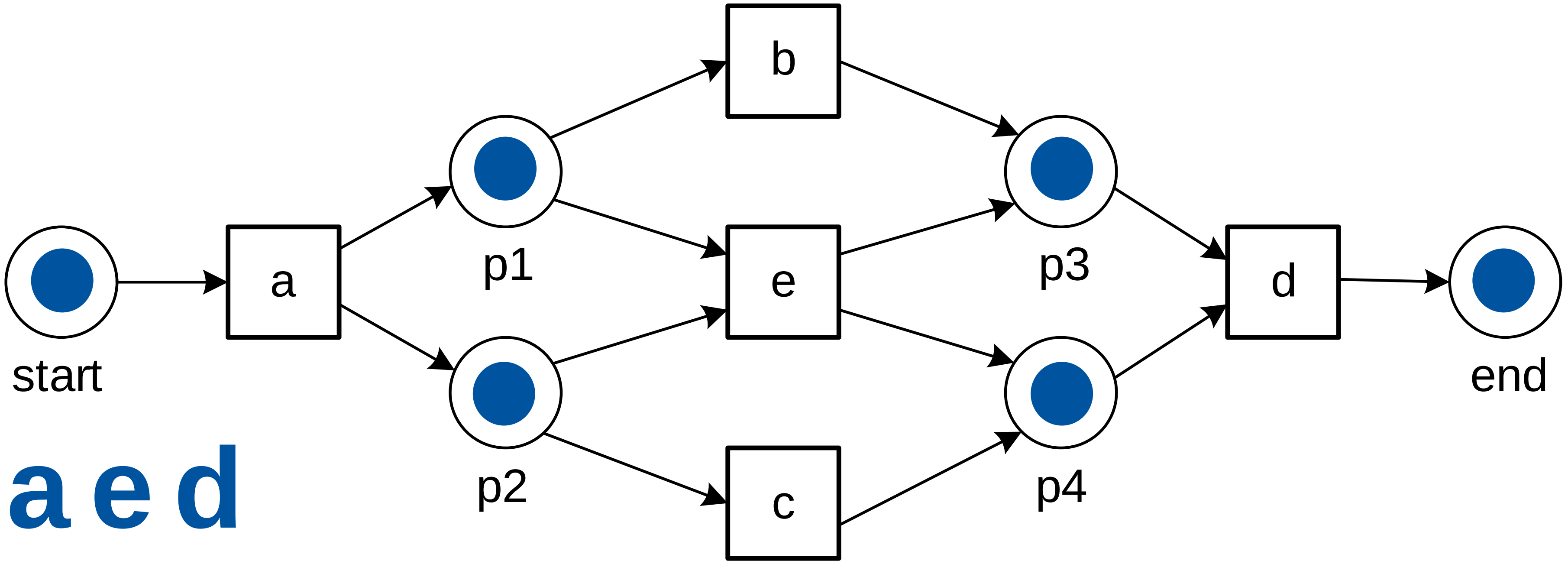
Example Trace (2/3)



a c b d

Initial marking [start], final marking [end]

Example Trace (3/3)

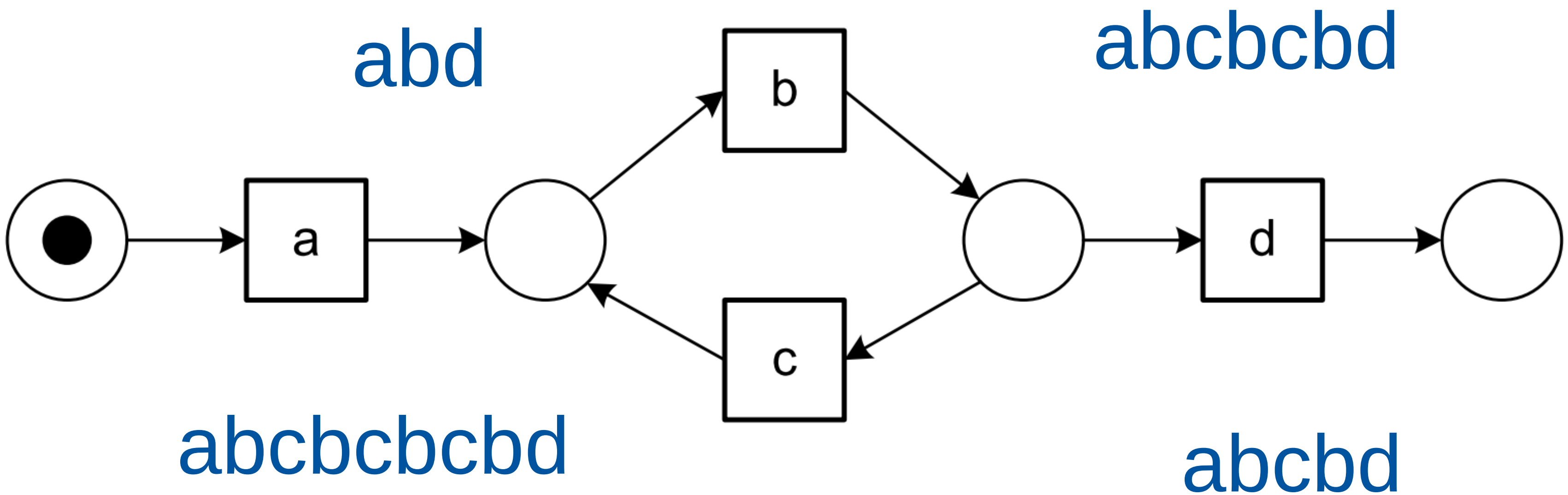


a e d

Initial marking [start], final marking [end]

Another Example – Loop

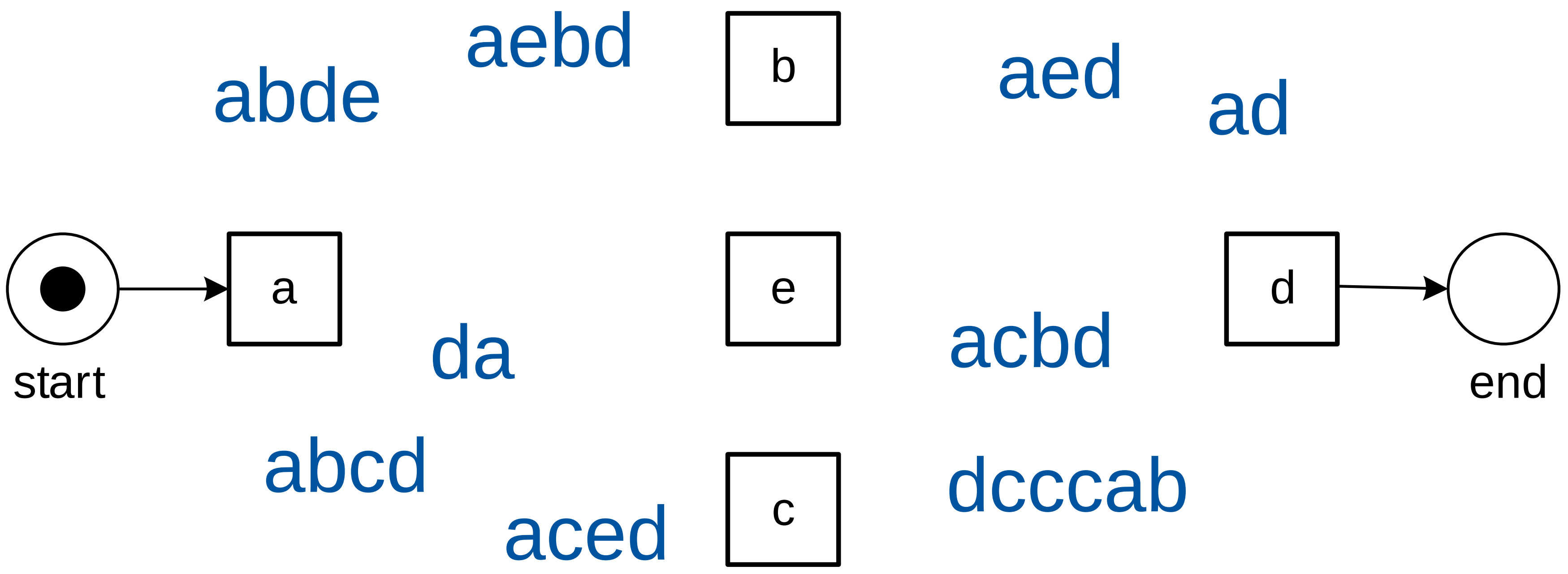
Infinitely many possible traces



Initial marking [start], final marking [end]

Places are Constraints

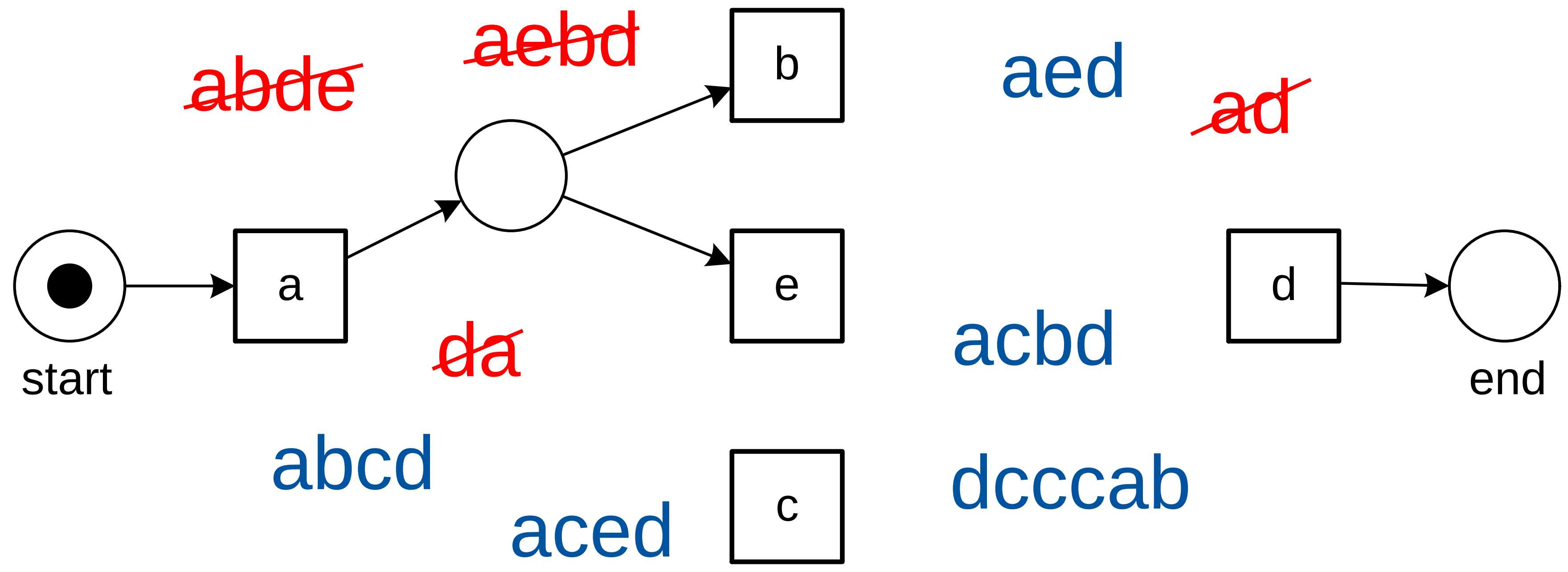
- Places cannot have ‘negative tokens’
- Must have the correct number of tokens in the end (indicated by final marking)



Initial marking [start], final marking [end]

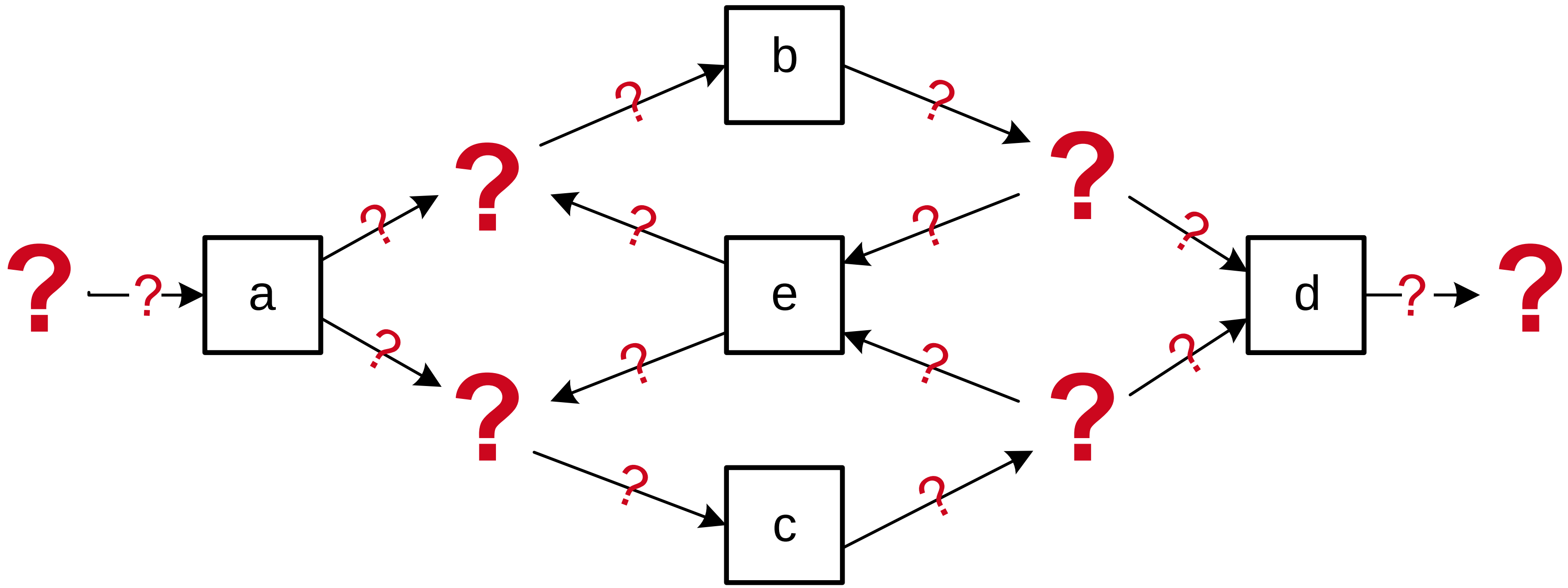
Places are Constraints

- Places cannot have ‘negative tokens’
- Must have the correct number of tokens in the end (indicated by final marking)



Initial marking [start], final marking [end]

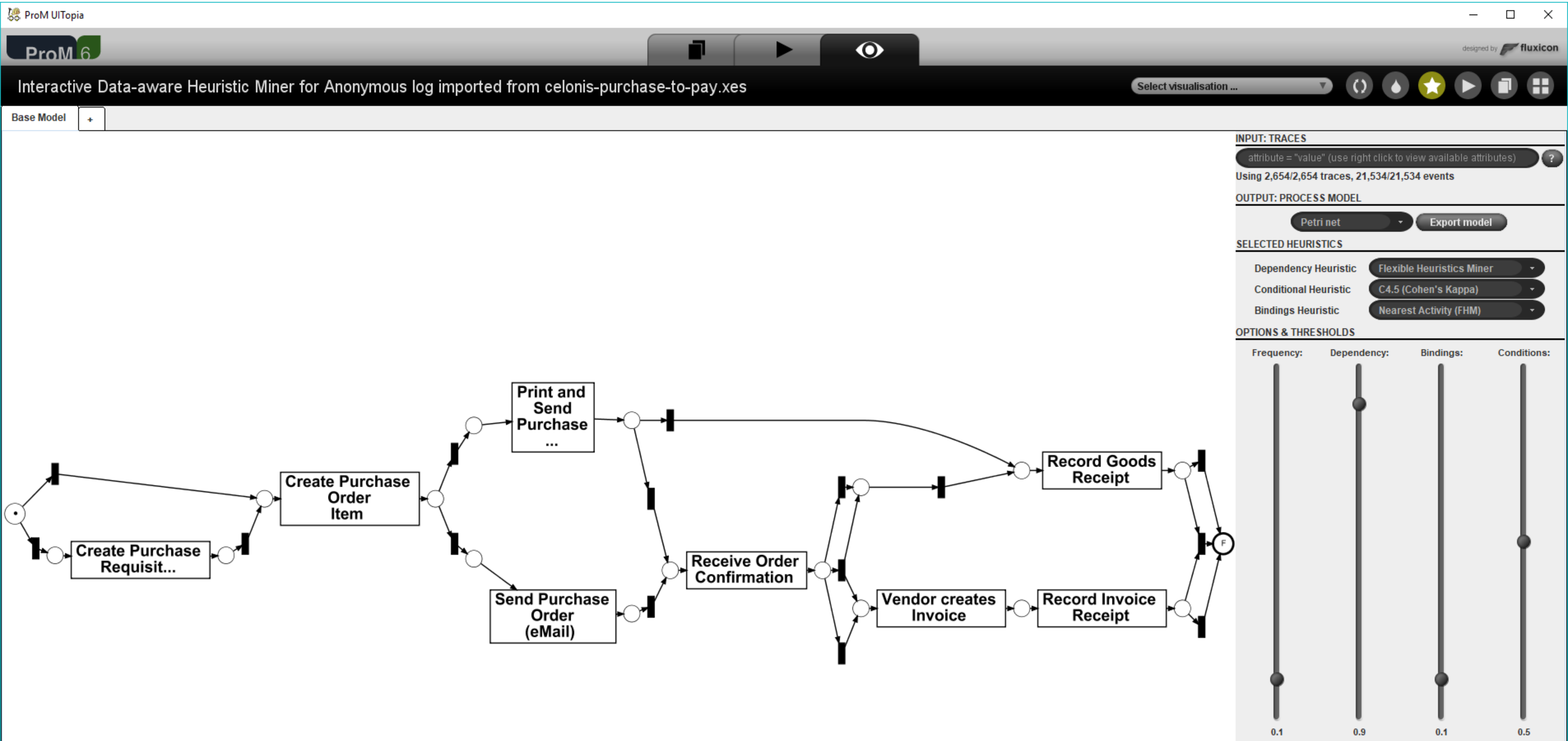
Process Discovery – Finding Places



Many Approaches Possible

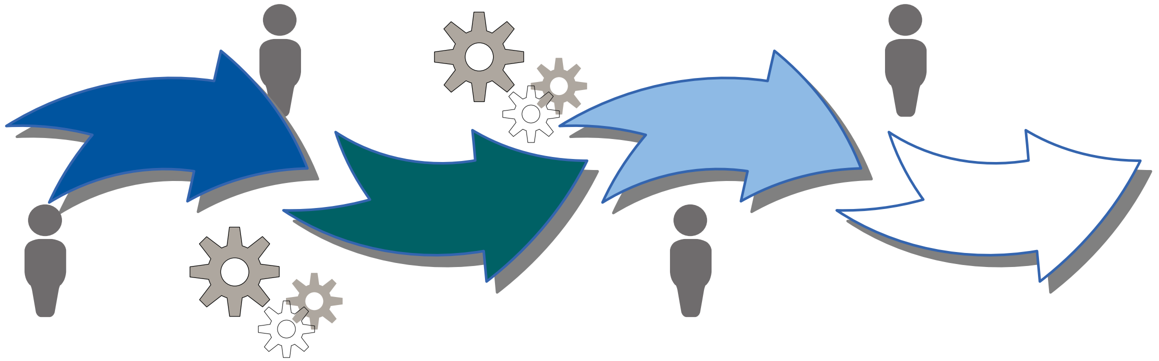
- Heuristics that provide only guarantees for limited classes of models (e.g., Alpha algorithm and heuristic miner)
- Approaches that formally guarantee perfect replayability of the event log (e.g., state-based regions)
- Genetic and other evolutionary approaches (very flexible)
- Optimization-based approaches that turn discovery into an optimization problem (e.g., ILP miner)
- Brute-force approaches that exploit monotonicity properties (apriori-style algorithms)

Example – Heuristic Miner Applied to SAP Data



Unsupervised Process Mining

1. Process Discovery
2. Bottom-Up Discovery
3. **Top-Down Discovery**

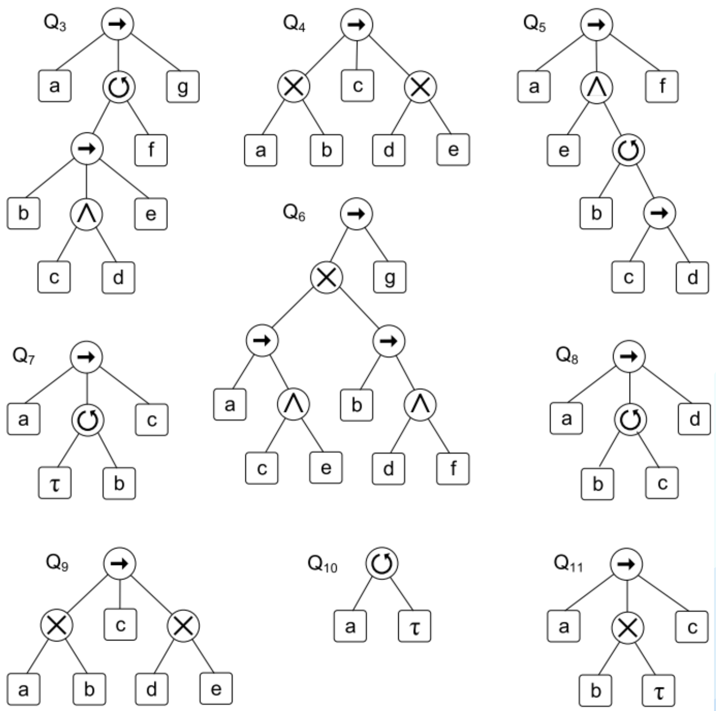


Example Top-Down Algorithm Approach: Inductive Mining

- Based on work done by Sander Leemans, Dirk Fahland, and Wil van der Aalst
- Family of approaches with different guarantees and scalability characteristics

  (all can ensure replayability of the whole event log)

$$
\begin{aligned}
L_3 &= [\langle a,b,c,d,e,f,b,d,c,e,g\rangle, \langle a,b,d,c,e,g\rangle^2 \\
&\quad \langle a,b,c,d,e,f,b,c,d,e,f,b,d,c,e,g\rangle] \\
L_4 &= [\langle a,c,d\rangle^{45}, \langle b,c,d\rangle^{42}, \langle a,c,e\rangle^{38}, \langle b,c,e\rangle^{22}] \\
L_5 &= [\langle a,b,e,f\rangle^2, \langle a,b,e,c,d,b,f\rangle^3, \langle a,b,c,e,d,b,f\rangle^2, \langle a,b,c,d,e,b,f\rangle^4, \\
&\quad \langle a,e,b,c,d,b,f\rangle^3] \\
L_6 &= [\langle a,c,e,g\rangle^2, \langle a,e,c,g\rangle^3, \langle b,d,f,g\rangle^2, \langle b,f,d,g\rangle^4] \\
L_7 &= [\langle a,c\rangle^2, \langle a,b,c\rangle^3, \langle a,b,b,c\rangle^2, \langle a,b,b,b,b,c\rangle] \\
L_8 &= [\langle a,b,d\rangle^3, \langle a,b,c,b,d\rangle^2, \langle a,b,c,b,c,b,d\rangle] \\
L_9 &= [\langle a,c,d\rangle^{45}, \langle b,c,e\rangle^{42}] \\
L_{10} &= [\langle a,a\rangle^{55}] \\
L_{11} &= [\langle a,b,c\rangle^{20}, \langle a,c\rangle^{30}]
\end{aligned}
$$



[1]

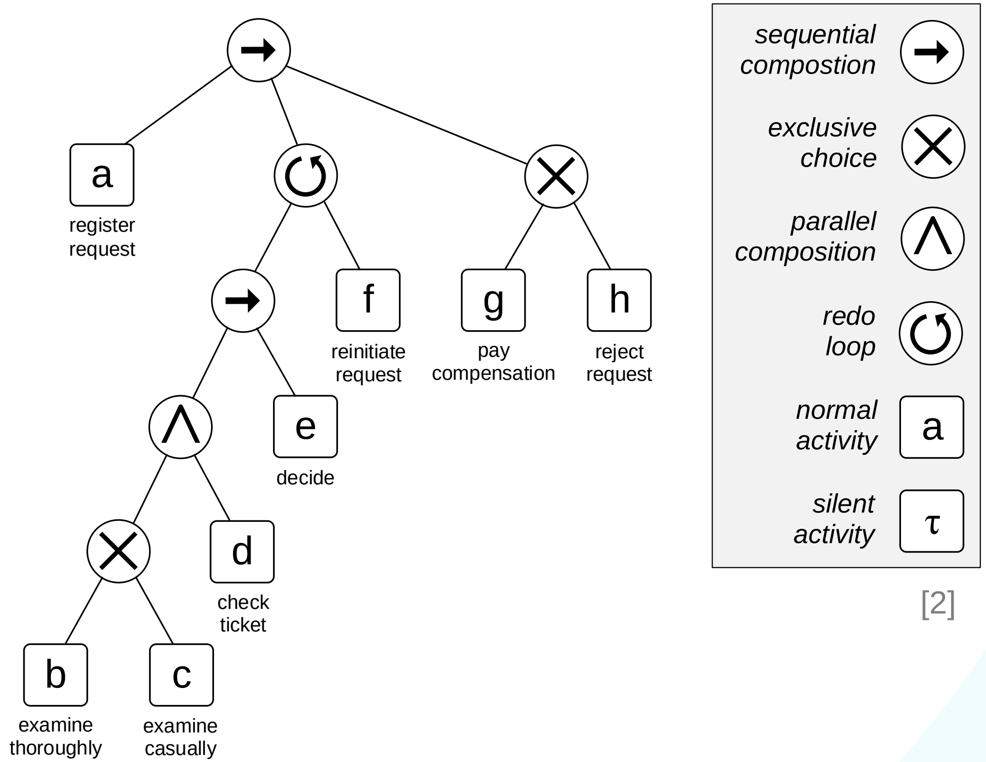
Inductive Mining

$$
\begin{aligned}
L = [&\langle a, b, d, e, h\rangle^3, \\
&\langle a, d, c, e, g\rangle^4, \\
&\langle a, c, d, e, f, b, d, e, g\rangle^2, \\
&\langle a, d, b, e, h\rangle^2, \\
&\langle a, c, d, e, f, d, c, e, f, c, d, e, h\rangle^2, \\
&\langle a, c, d, e, g\rangle]
\end{aligned}
$$

Input – simplified event log

Inductive Mining

$$
\begin{aligned}
L = [&\langle a,b,d,e,h\rangle^3,\\
&\langle a,d,c,e,g\rangle^4,\\
&\langle a,c,d,e,f,b,d,e,g\rangle^2,\\
&\langle a,d,b,e,h\rangle^2,\\
&\langle a,c,d,e,f,d,c,e,f,c,d,e,h\rangle^2,\\
&\langle a,c,d,e,g\rangle]
\end{aligned}
$$



[2]

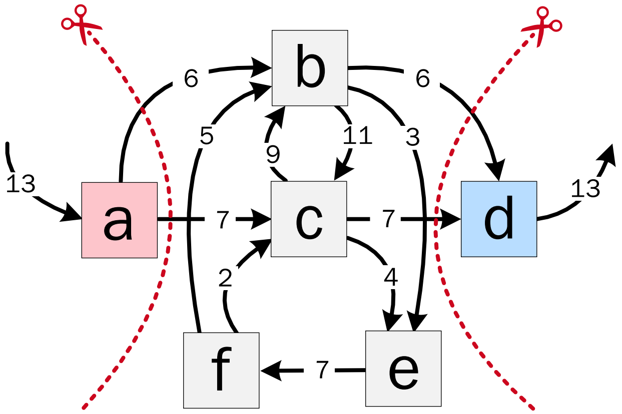
Input – simplified event log

Output – process tree

Inductive Mining in Steps

Apply recursively (split into multiple sublogs):

1. Create DFG based on the event log
2. Find a cut in the DFG
3. Partition event log based on chosen cut
4. Handle base cases
5. Recurse on non-base cases

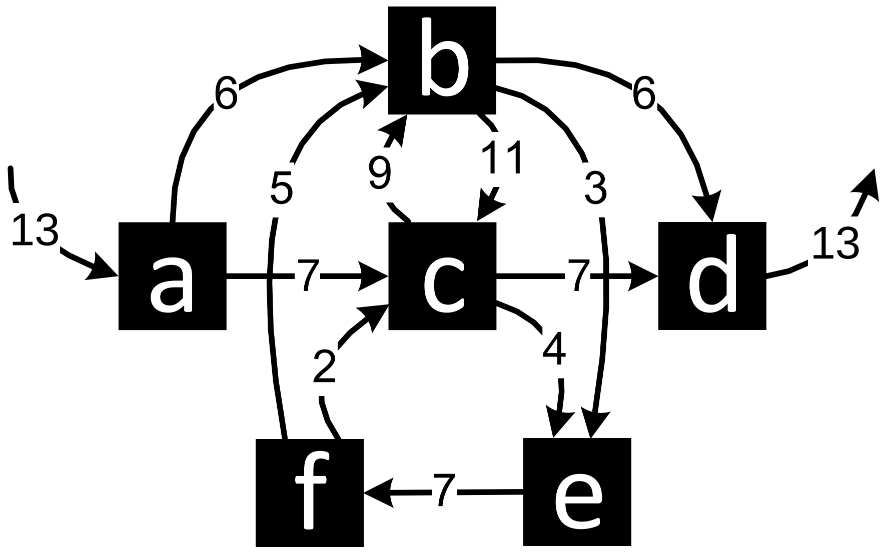


Top-Down

Applying Inductive Mining Recursively

Step 1 – Create DFG Based On the Event Log

3x a b c d
4x a c b d
2x a b c e f b c d
2x a c b e f b c d
1x a b c e f c b d
1x a c b e f b c e f c b d

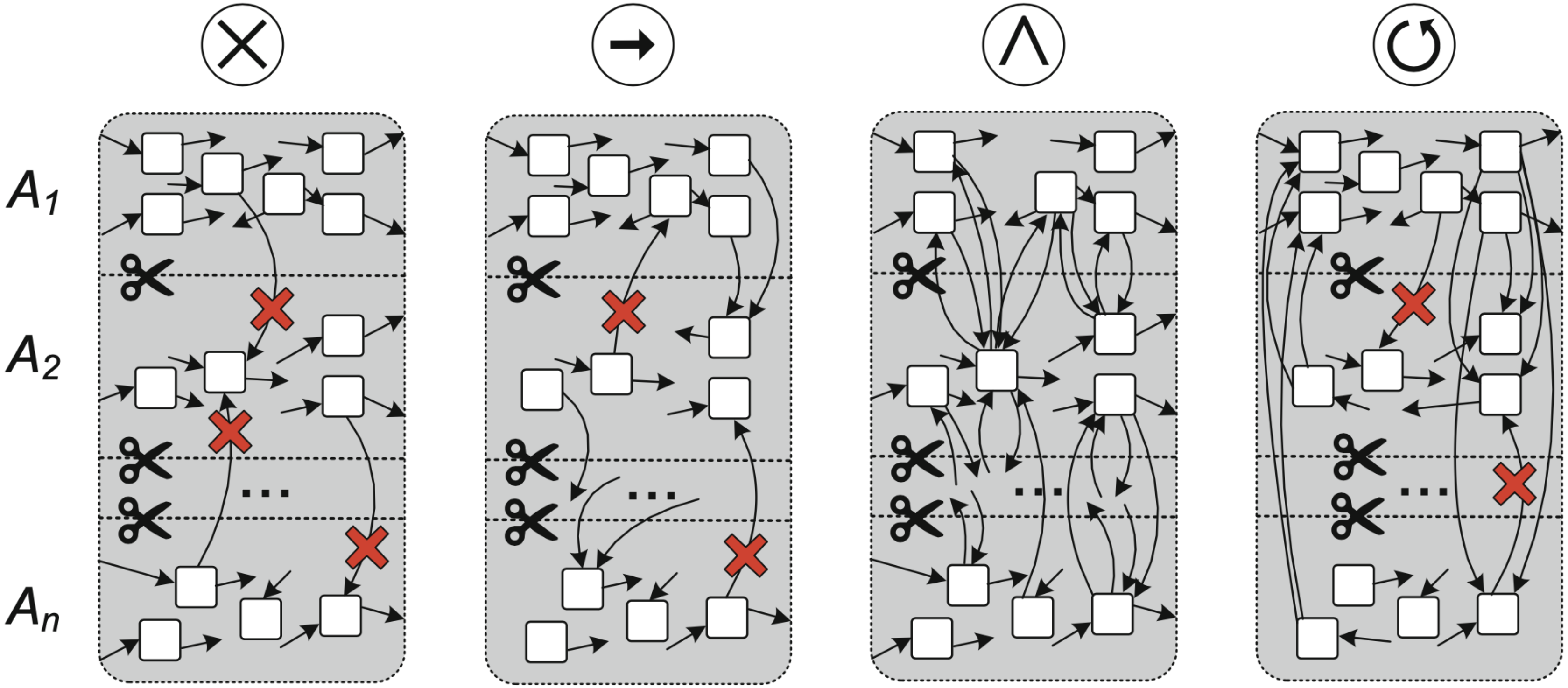


[3]

Input – simplified event log

Directly-follows graph

Inductive Mining – Possible Cuts



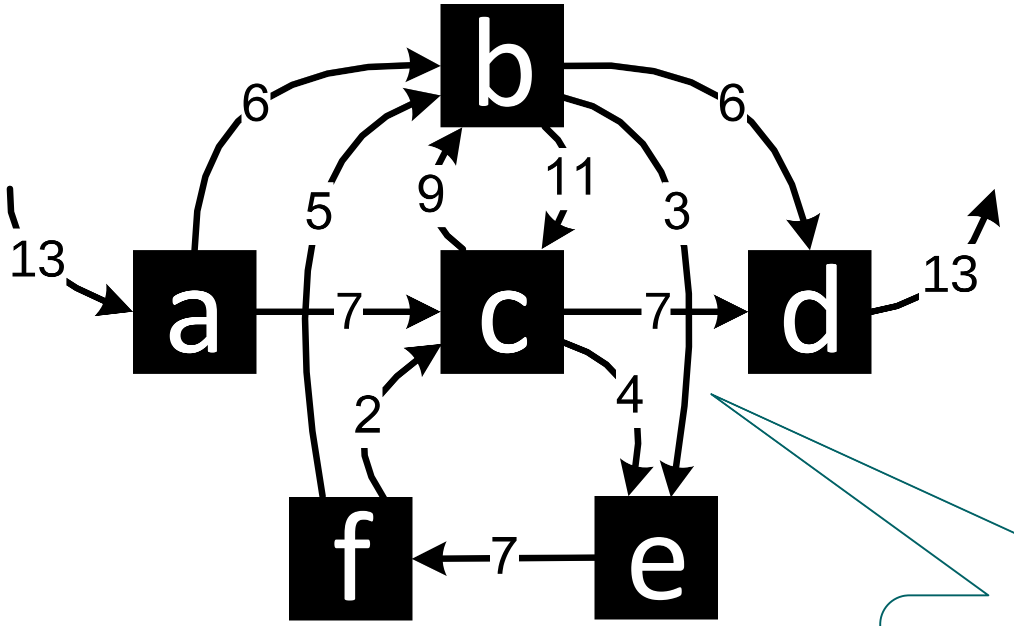
(a) exclusive-choice cut (b) sequence cut (c) parallel cut (d) redo-loop cut

[4]

1 Order 4

Applying Inductive Mining Recursively

Step 2 – Choose Cut

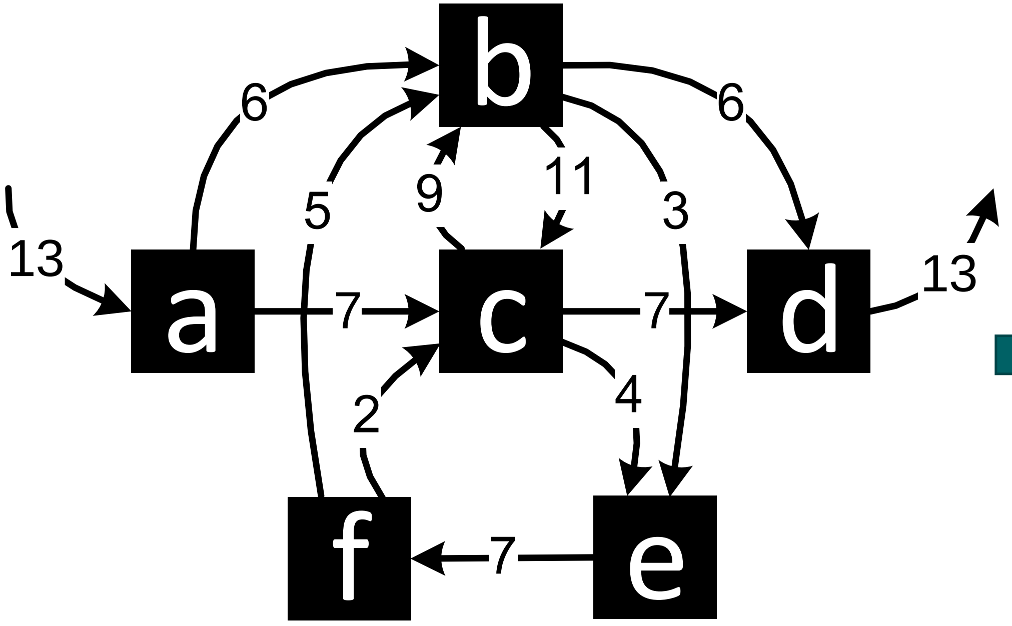


Directly-follows graph

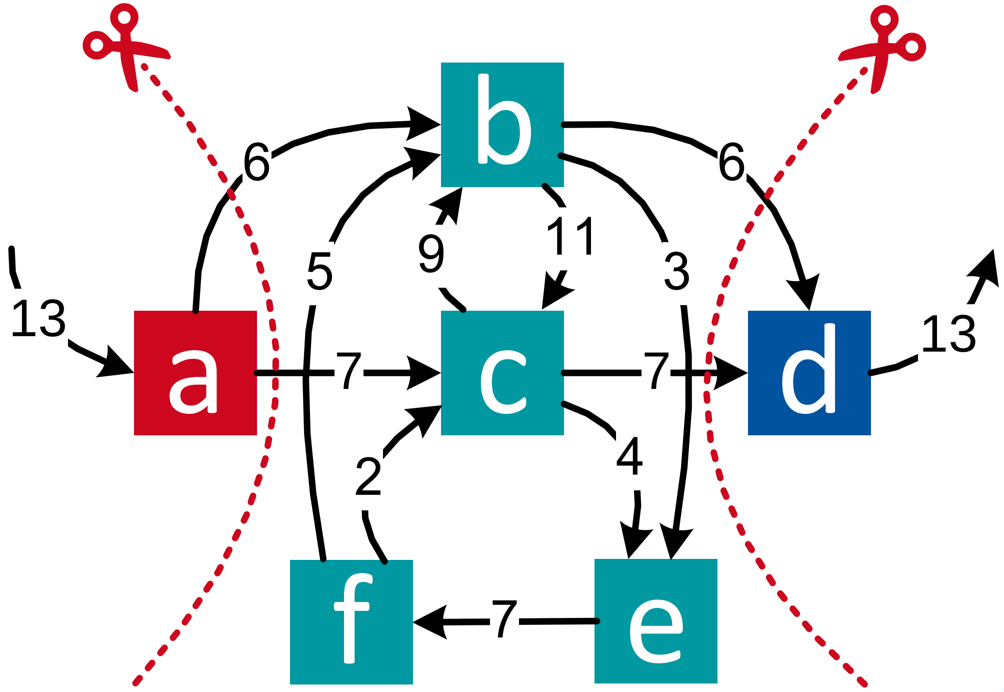
Exclusive choice cut not possible, but we can apply sequence cut!

Applying Inductive Mining Recursively

Step 2 – Choose Cut



Directly-follows graph

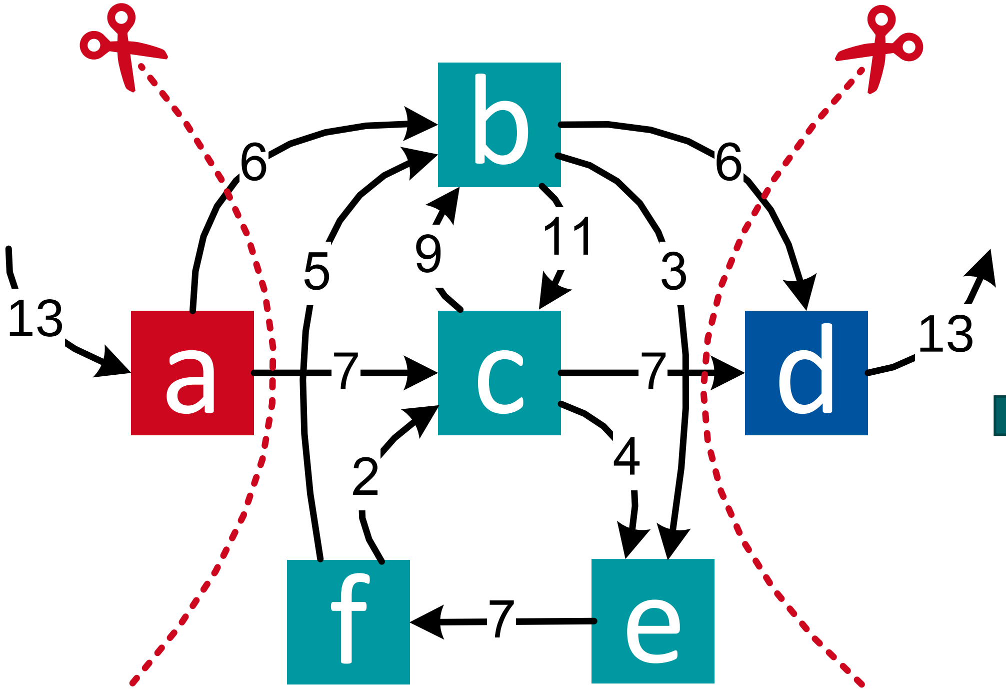


Sequence cut

{a} , {b,c,e,f} , {d}

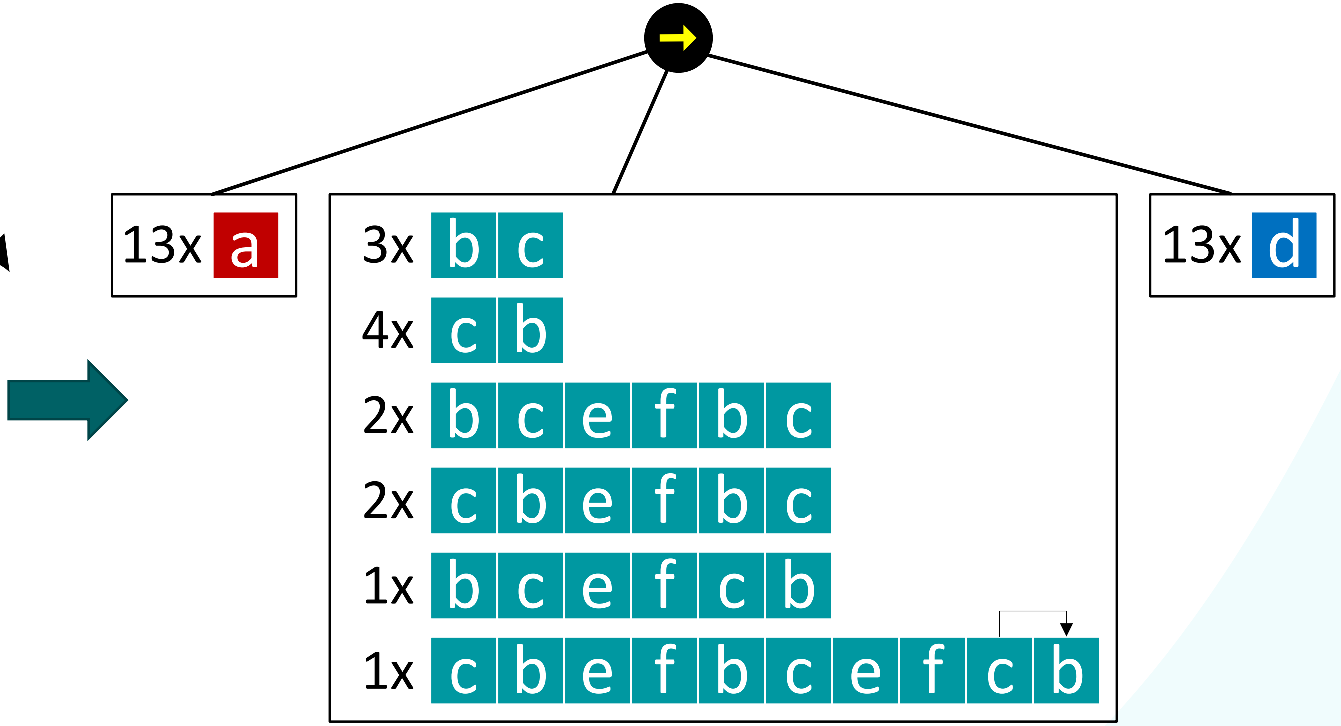
Applying Inductive Mining Recursively

Step 3 – Partition Event Log Based on Chosen Cut



Sequence cut

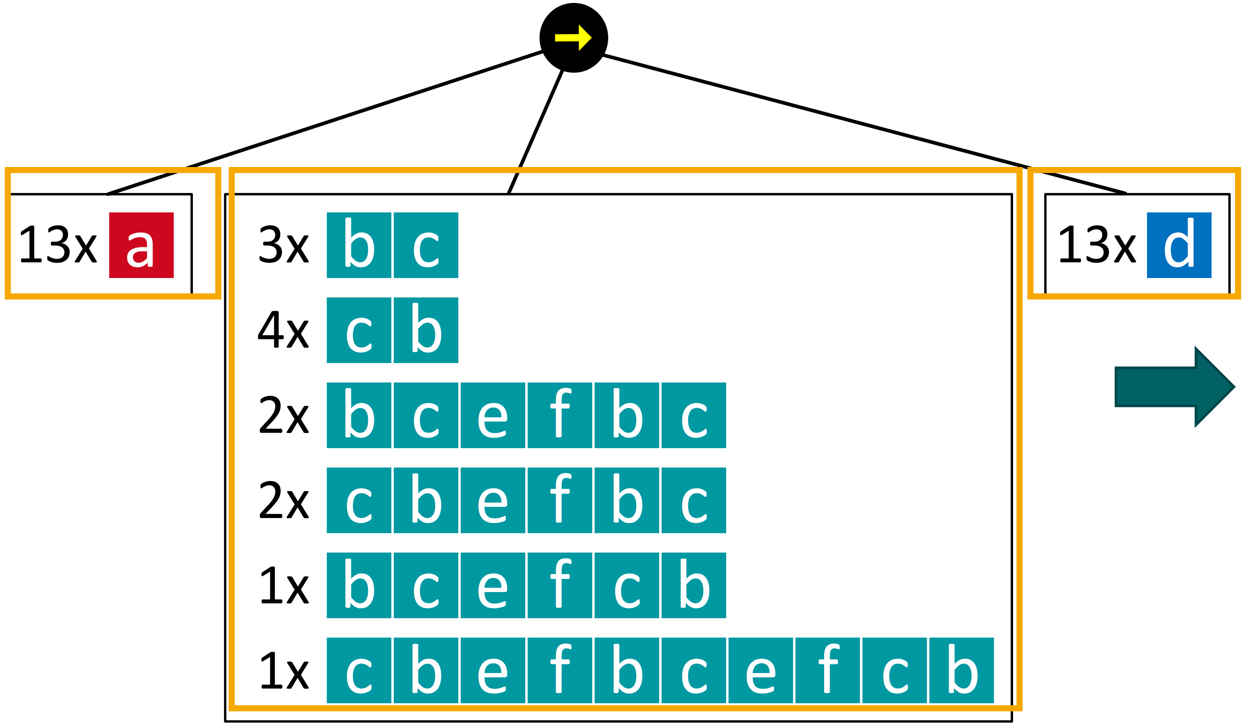
{a} , {b,c,e,f} , {d}



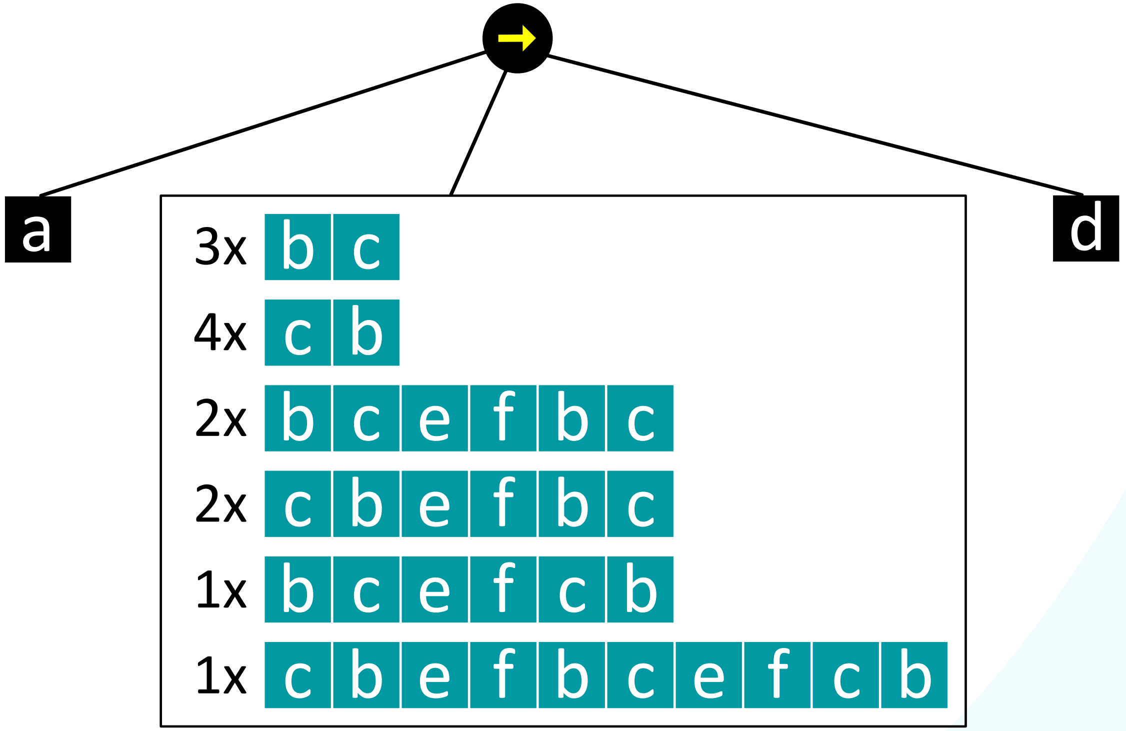
Partitioned event log

Applying Inductive Mining Recursively

Step 4 – Handble Base Cases



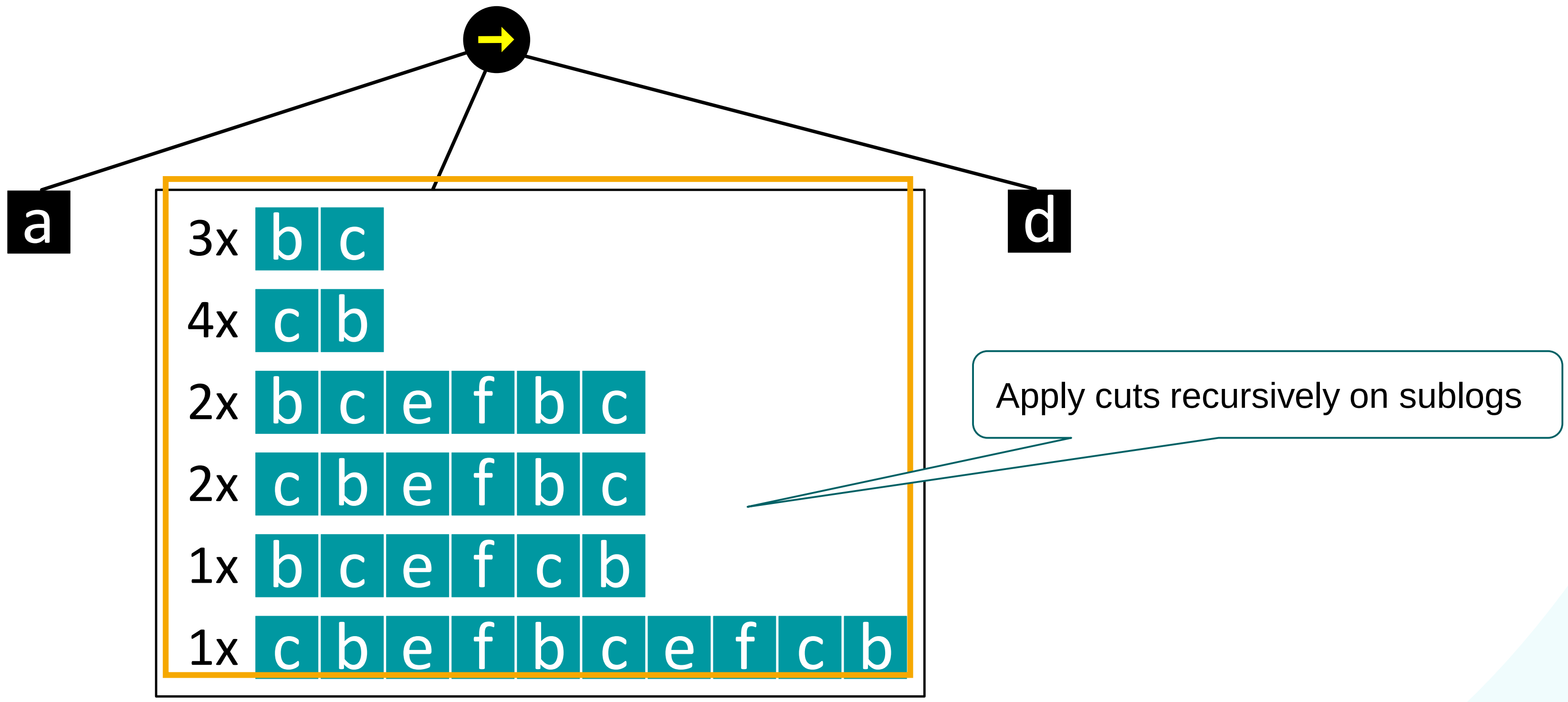
Partitioned event log



Process tree after first cut

Applying Inductive Mining Recursively

Step 5 – Recurse on Non-Base Cases

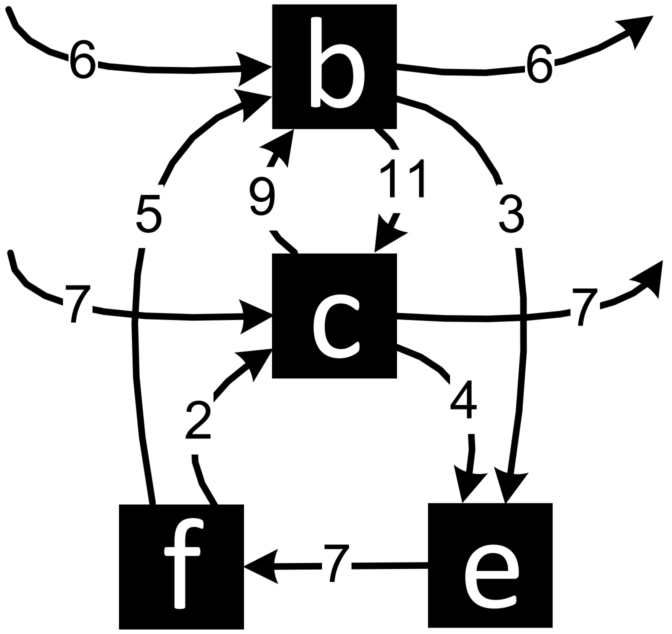


Process tree after first cut

Applying Inductive Mining Recursively

Step 1 – Create DFG Based On the Event Log

3x	b	c								
4x	c	b								
2x	b	c	e	f	b	c				
2x	c	b	e	f	b	c				
1x	b	c	e	f	c	b				
1x	c	b	e	f	b	c	e	f	c	b

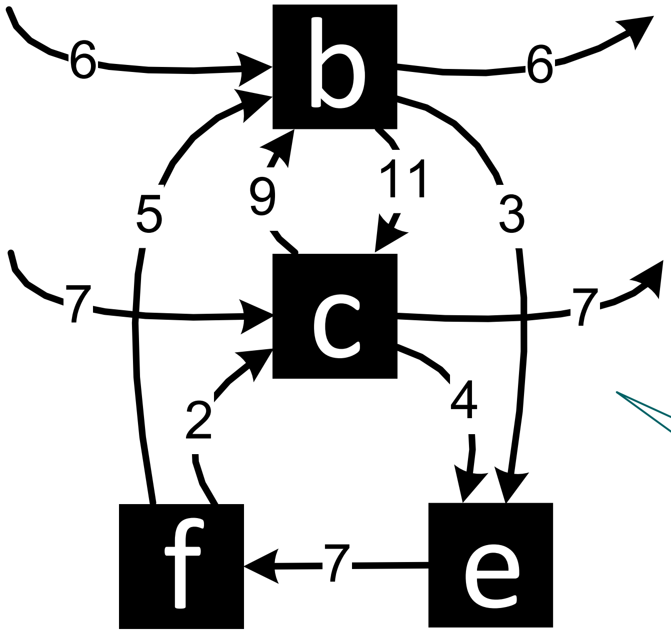


Input – simplified event log

Directly-follows graph

Applying Inductive Mining Recursively

Step 2 – Choose Cut

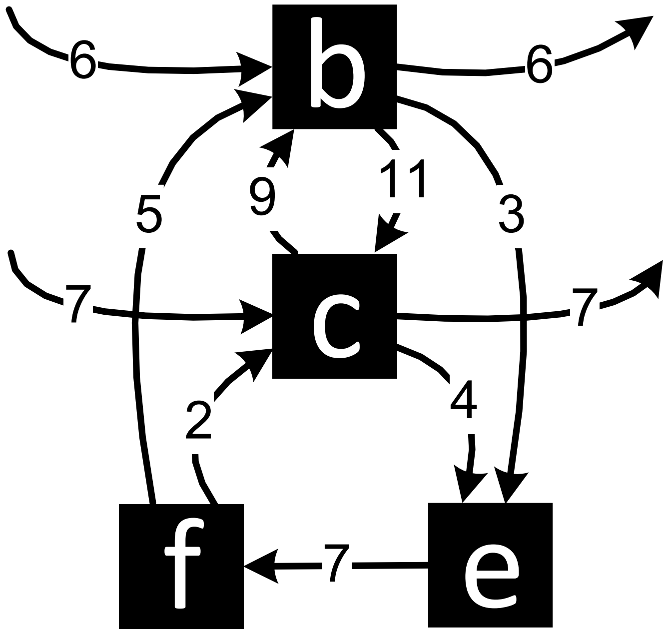


Directly-follows graph

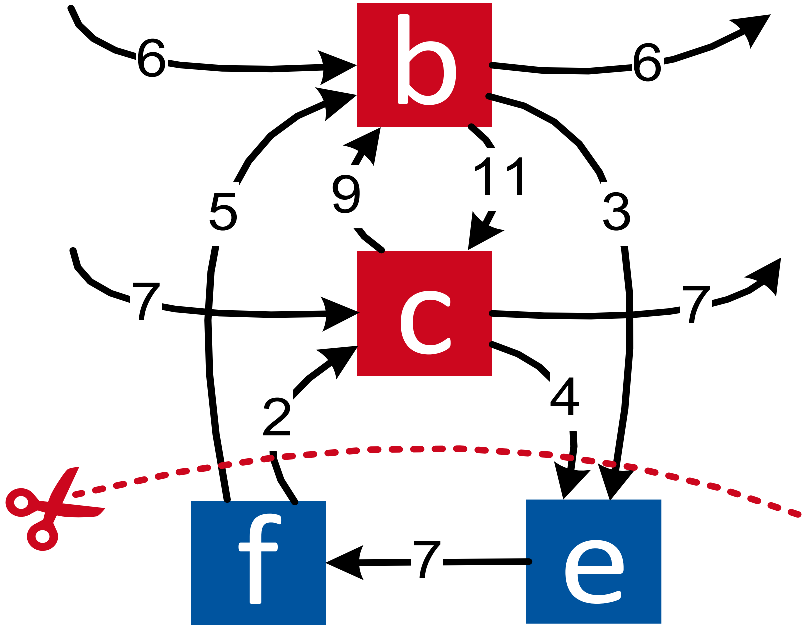
Exclusive choice, sequence cut and parallel cuts not possible, but we can apply loop cut!

Applying Inductive Mining Recursively

Step 2 – Choose Cut



Directly-follows graph

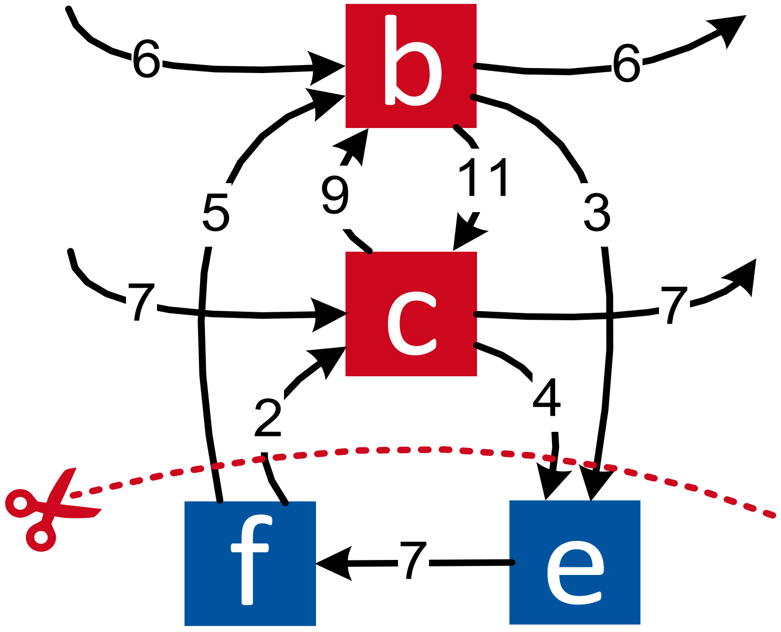


Loop cut

{b, c} , {f, e}

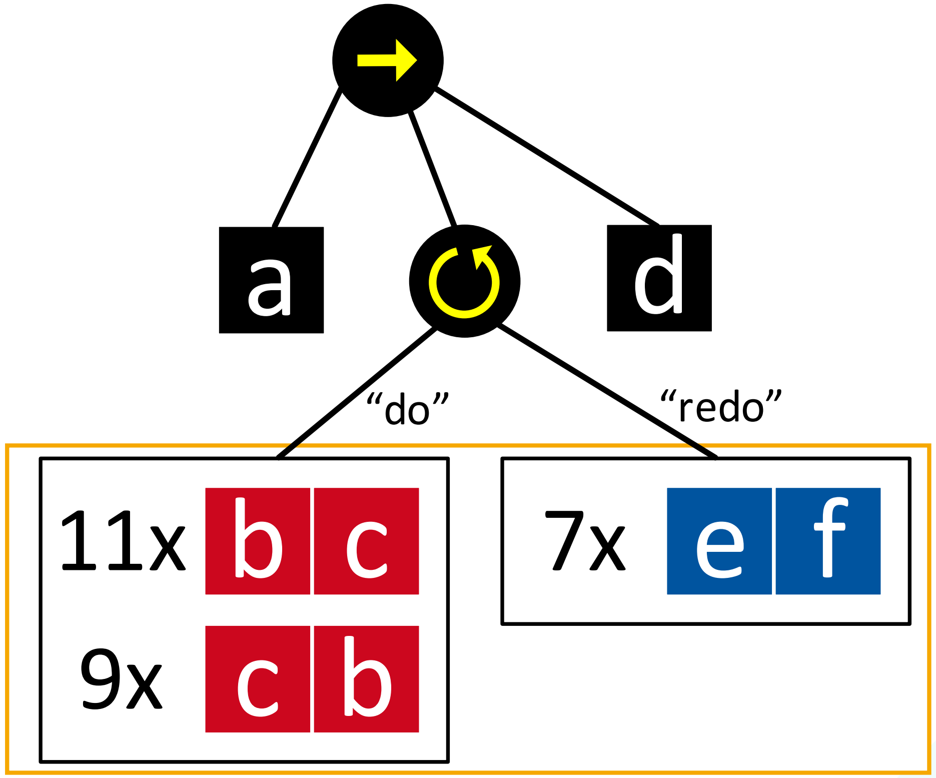
Applying Inductive Mining Recursively

Step 3 – Partition Event Log Based on Chosen Cut



Loop cut

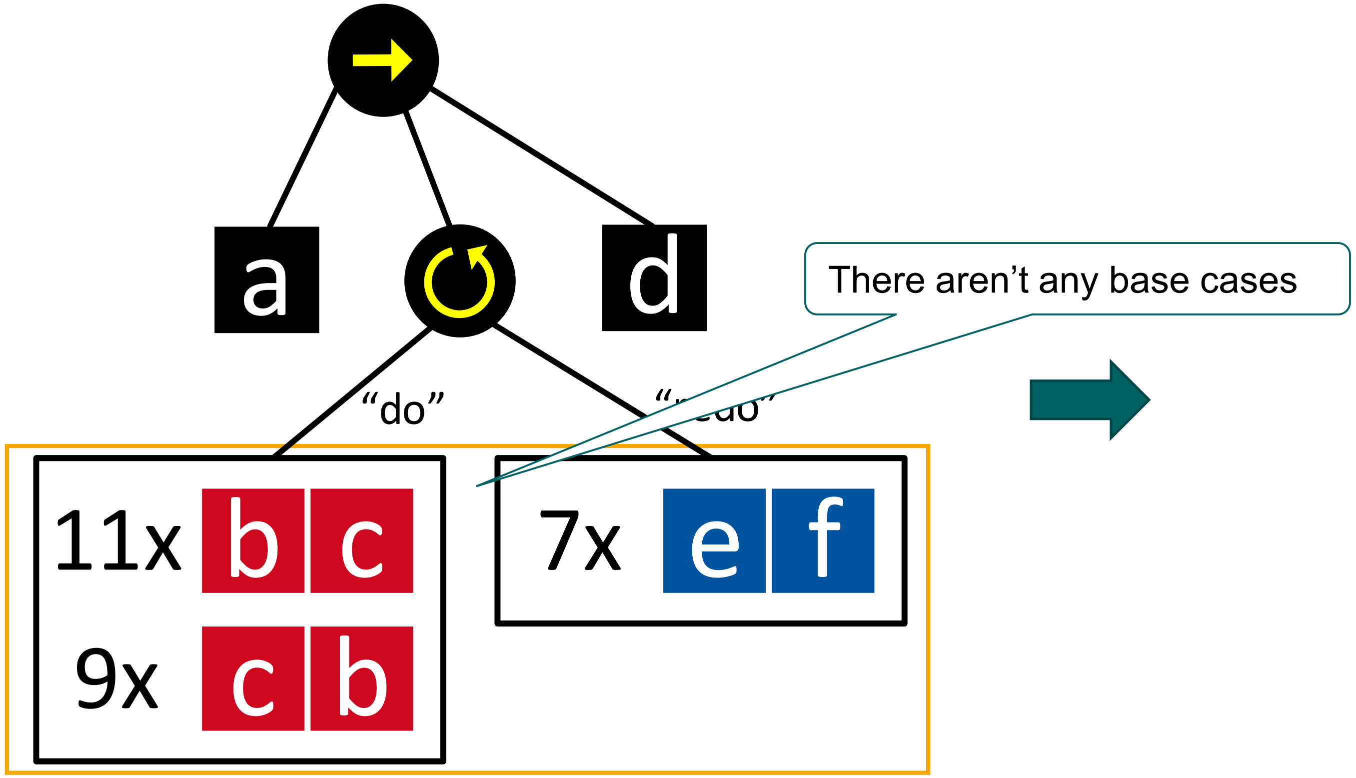
{b, c} , {f, e}



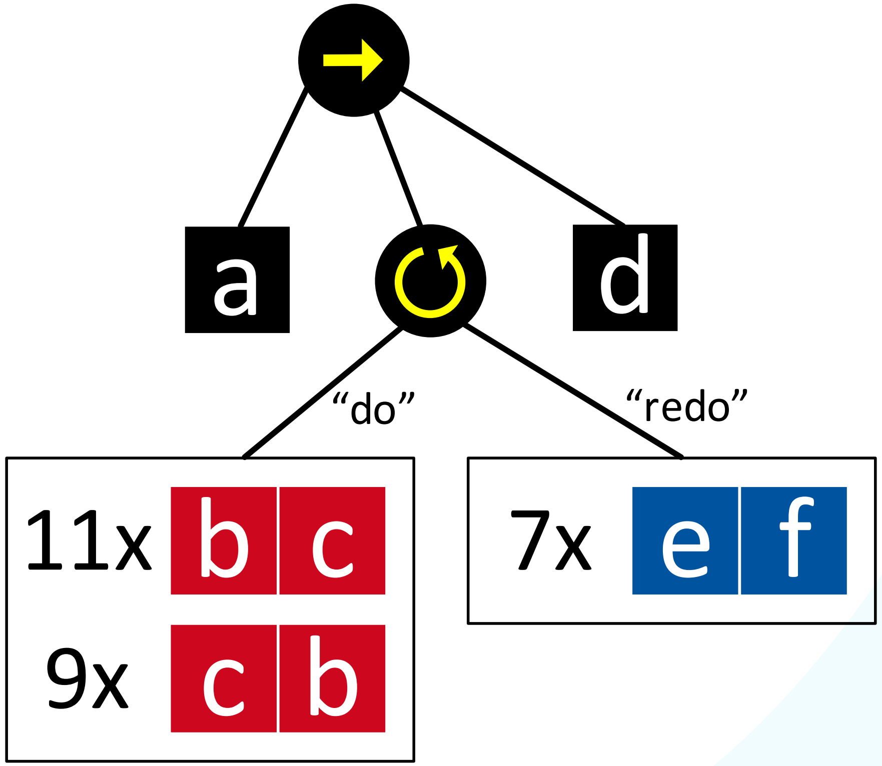
Partitioned event log

Applying Inductive Mining Recursively

Step 4 – Handble Base Cases



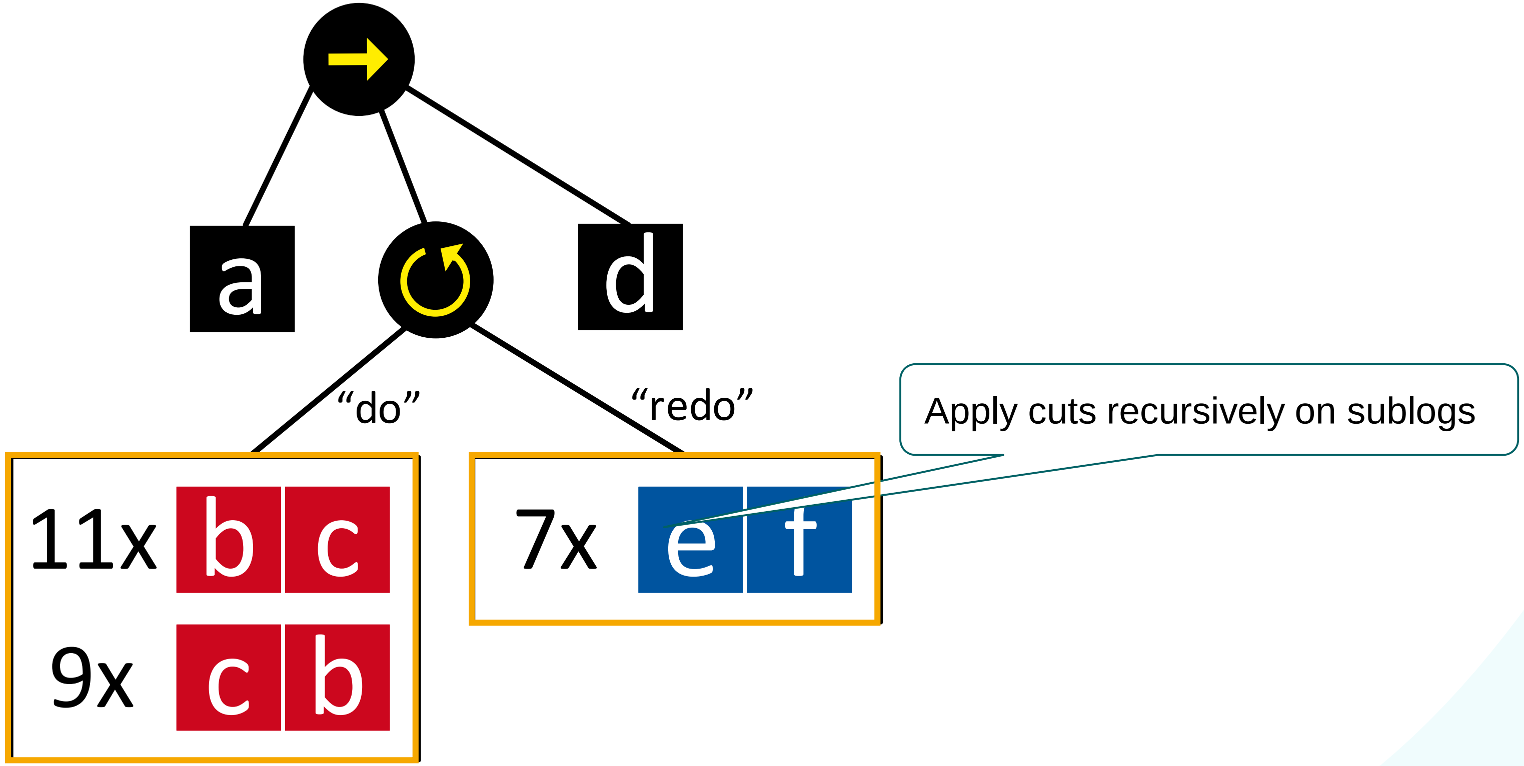
Partitioned event log



Process tree after second cut

Applying Inductive Mining Recursively

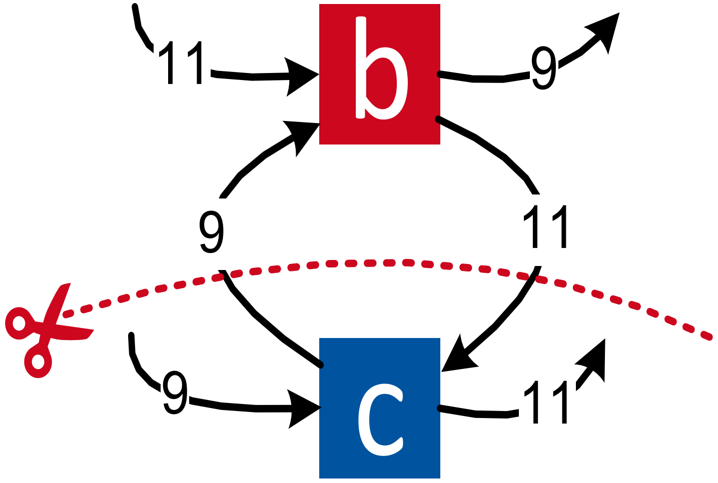
Step 5 – Recurse on Non-Base Cases



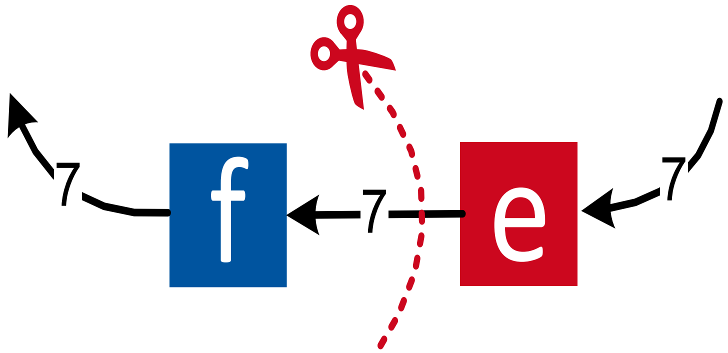
Process tree after second cut

Applying Inductive Mining Recursively

Repeat All These Steps on the Sublogs



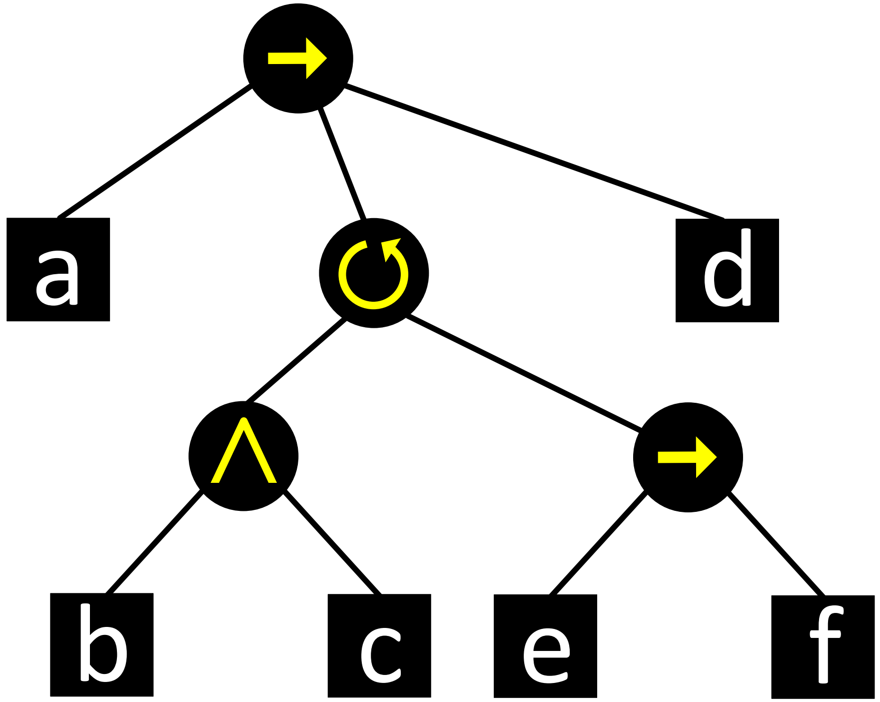
Parallel cut



Sequence cut

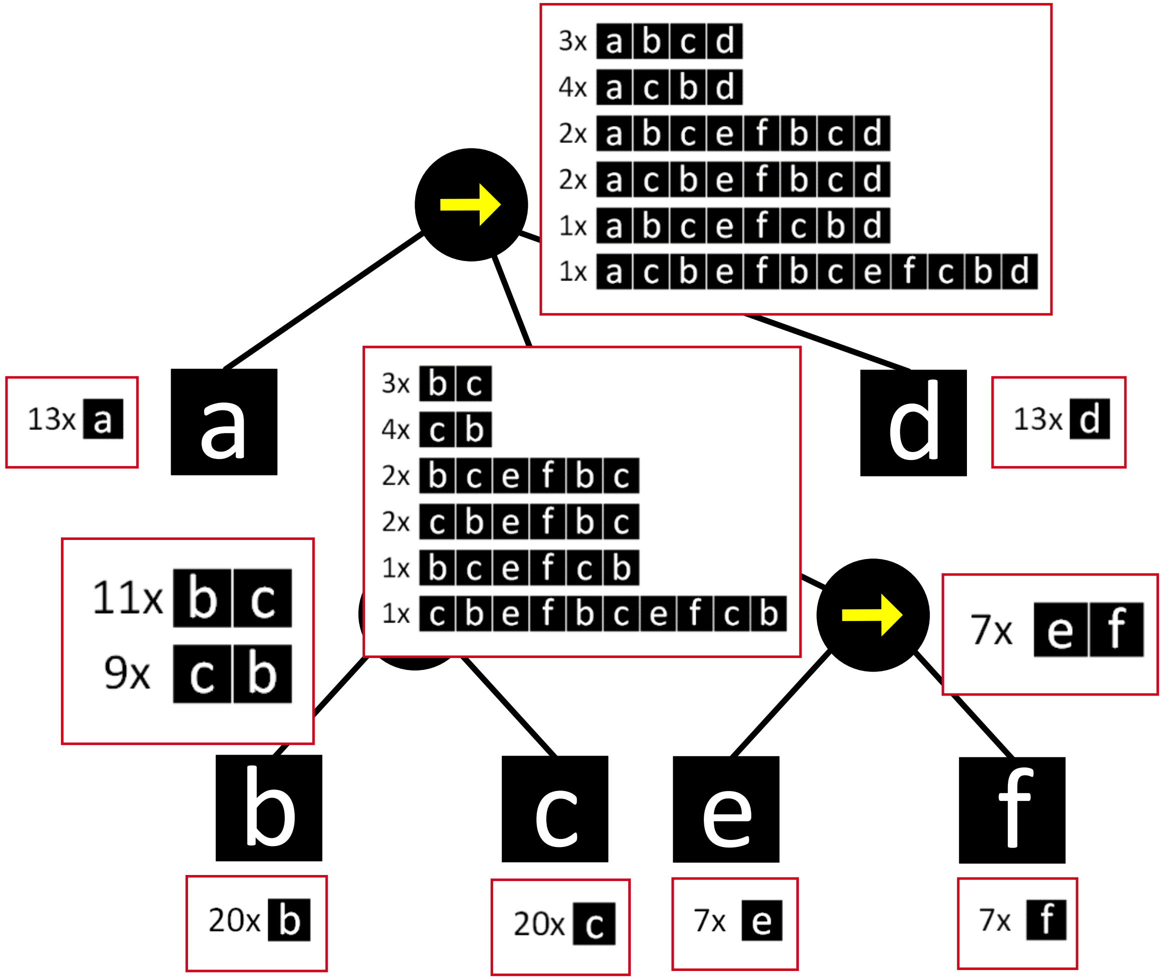
Applying Inductive Mining Recursively

Final Process Tree

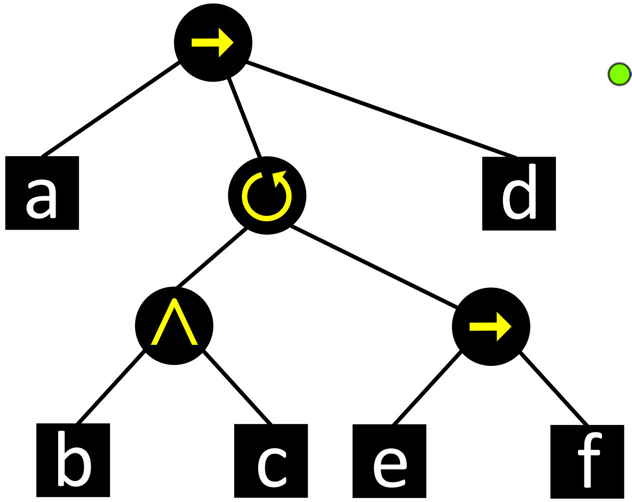


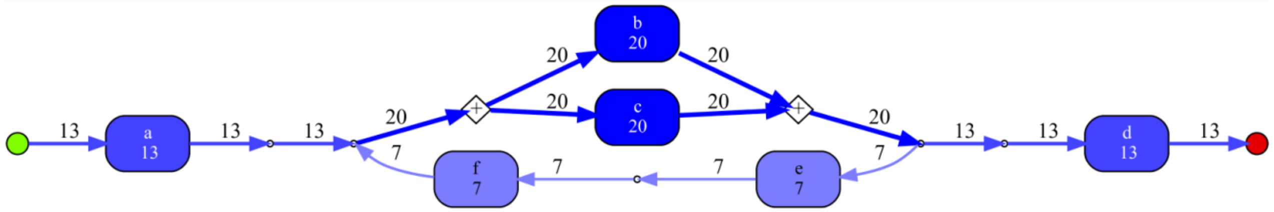
Applying Inductive Mining Recursively

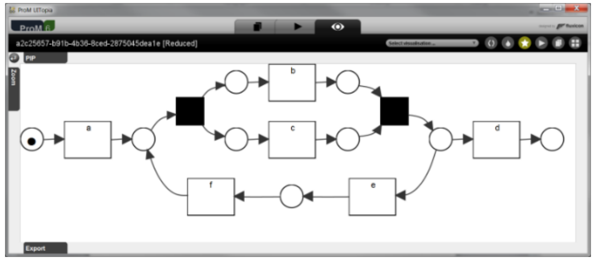
Top-Down Process

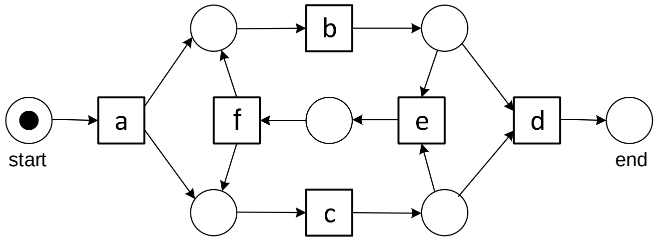


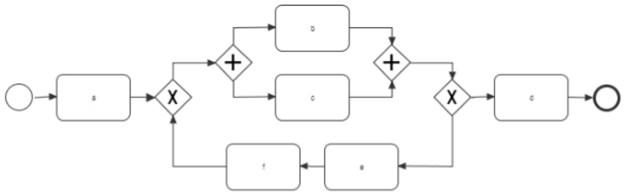
Alternative Notations







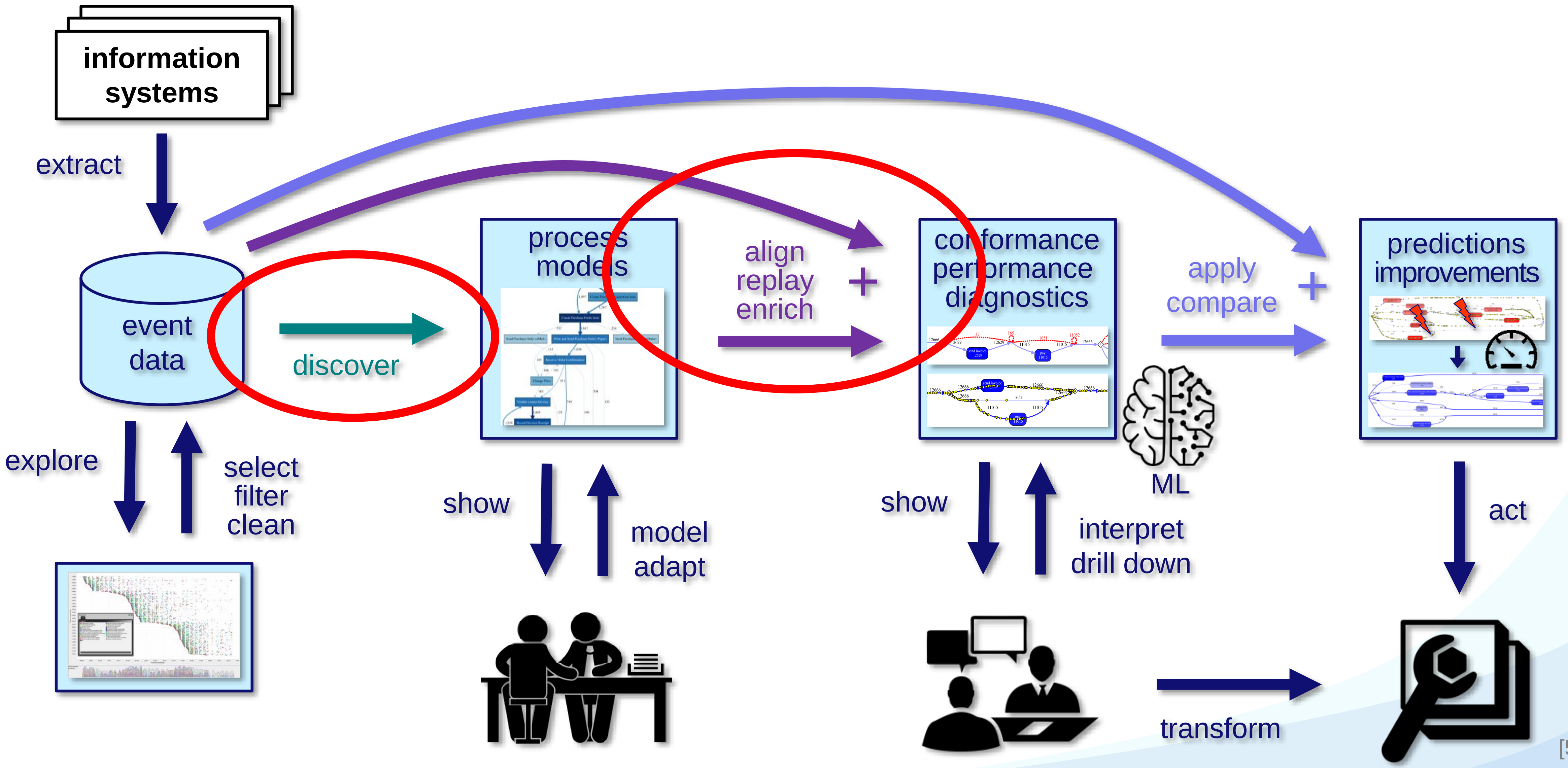




Inductive Mining Properties

- Basic algorithm formally guarantees that the original event log can be fully replayed
- Models satisfy formal properties that greatly facilitate further analysis (soundness)
- If the event log was generated from a basic process tree (no duplication of labels), then an equivalent model will be found
- Extensions exist to deal with infrequent behavior and incomplete event logs
- Highly scalable – dealing with billions of events, millions of cases, and thousands of unique activities
- Allows for distribution, streaming, etc.

From Process Discovery to Conformance Checking



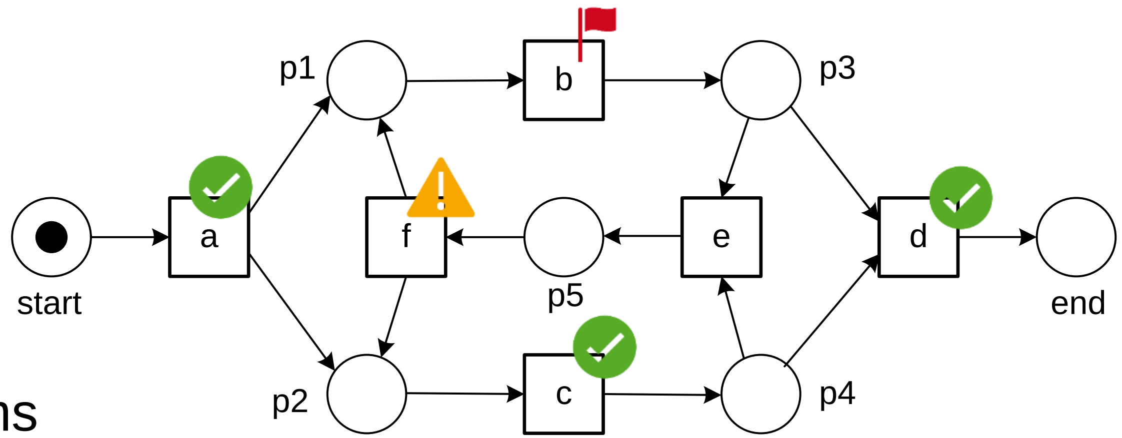
[5]

Part III: Supervised Process Mining

Conformance Checking and the Connection to “Mainstream ML”

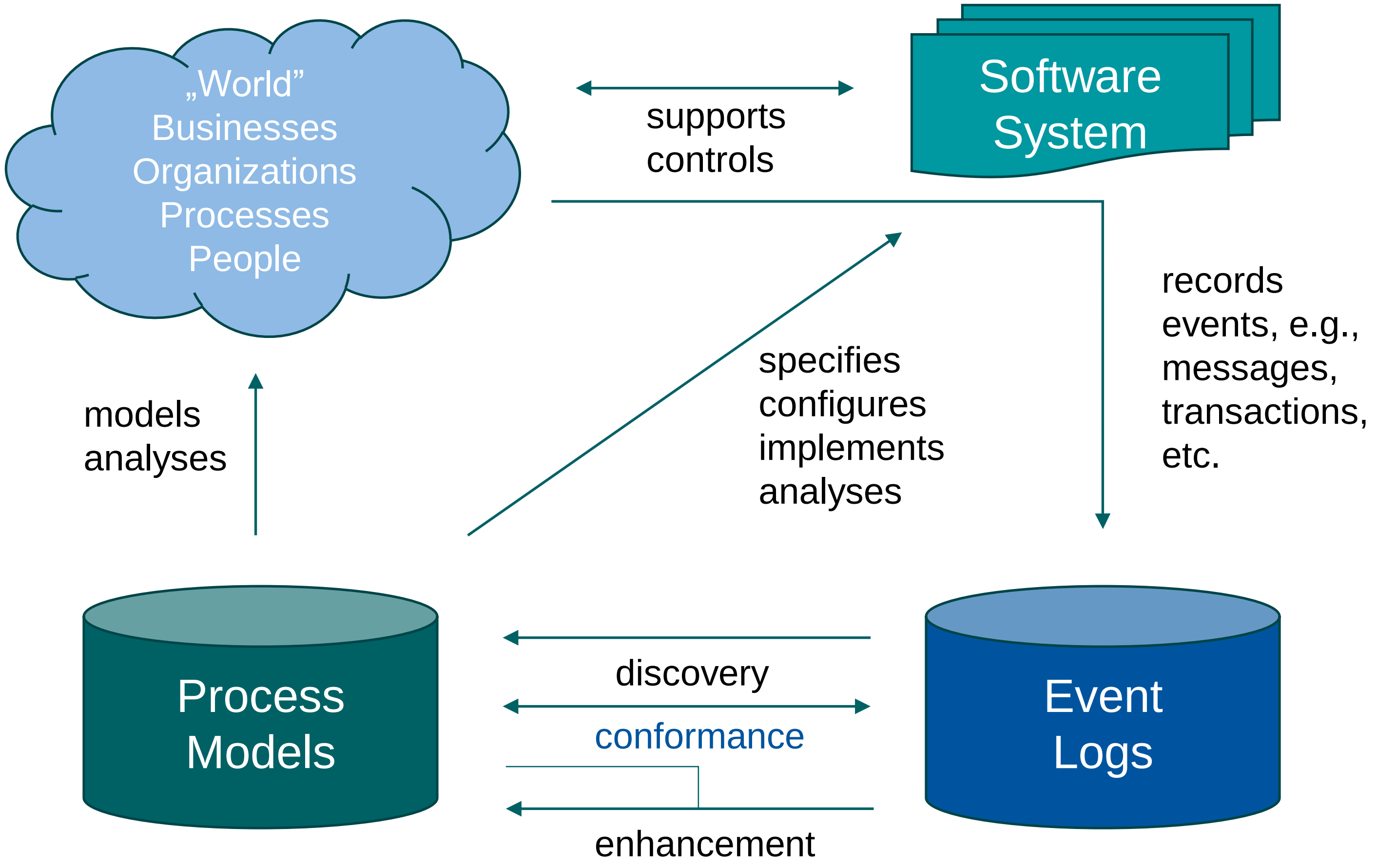
Supervised Process Mining

1. **Token-Based Replay**
2. Token-Based Replay Examples
3. Fitness at the Log Level
4. Generating Supervised Learning Problems



Conformance checking and the link to other data science techniques (e.g., machine learning).

Positioning Conformance Checking

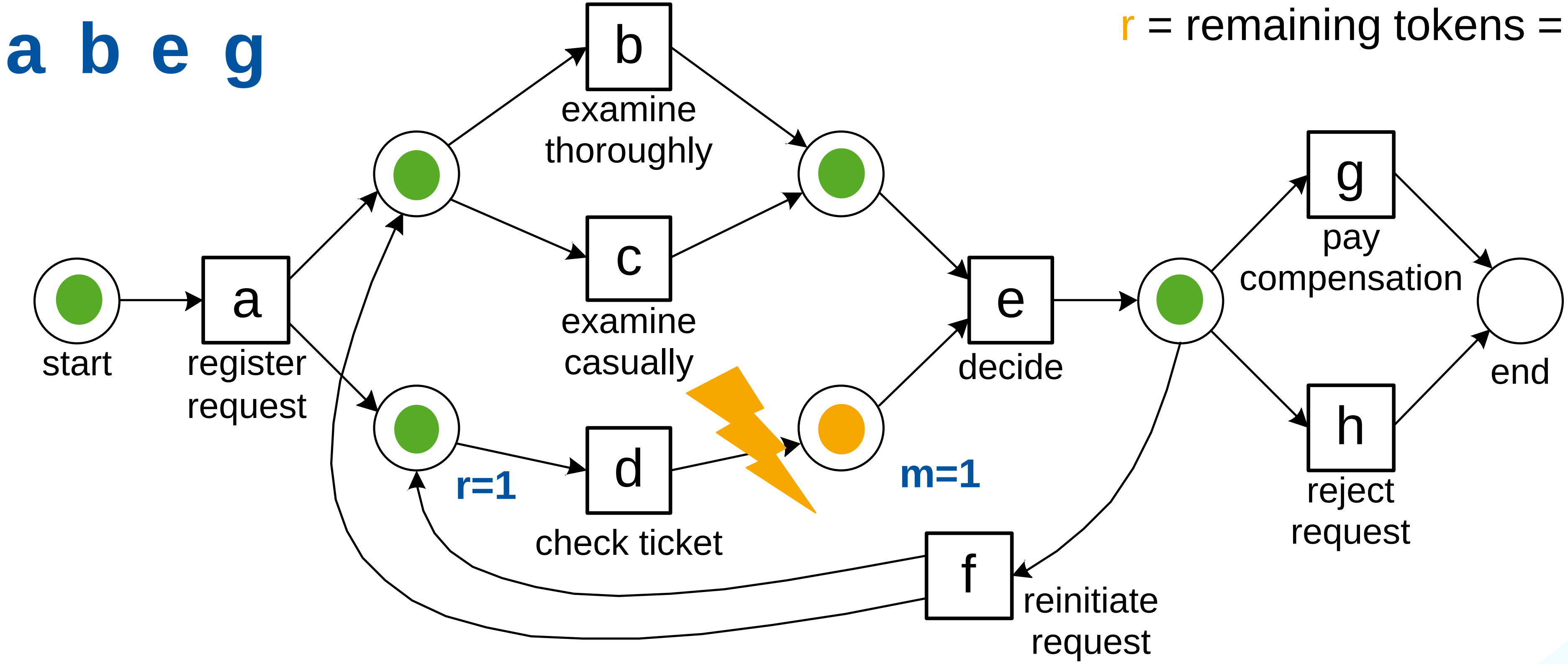


[1]

There are several conformance-checking techniques,
here we focus on "token-based replay".

Counting Tokens While Replaying

p = produced tokens = 6
c = consumed tokens = 6
m = missing tokens = 1
r = remaining tokens = 1



Fitness at the Trace Level

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

Fitness at the Trace Level

$$\text{fitness}(\sigma, N) = \frac{1}{2}\left(1 - \frac{1}{6}\right) + \frac{1}{2}\left(1 - \frac{1}{6}\right) = \frac{5}{6} \approx 0.83$$

p = produced tokens = 6
c = consumed tokens = 6
m = missing tokens = 1
r = remaining tokens = 1

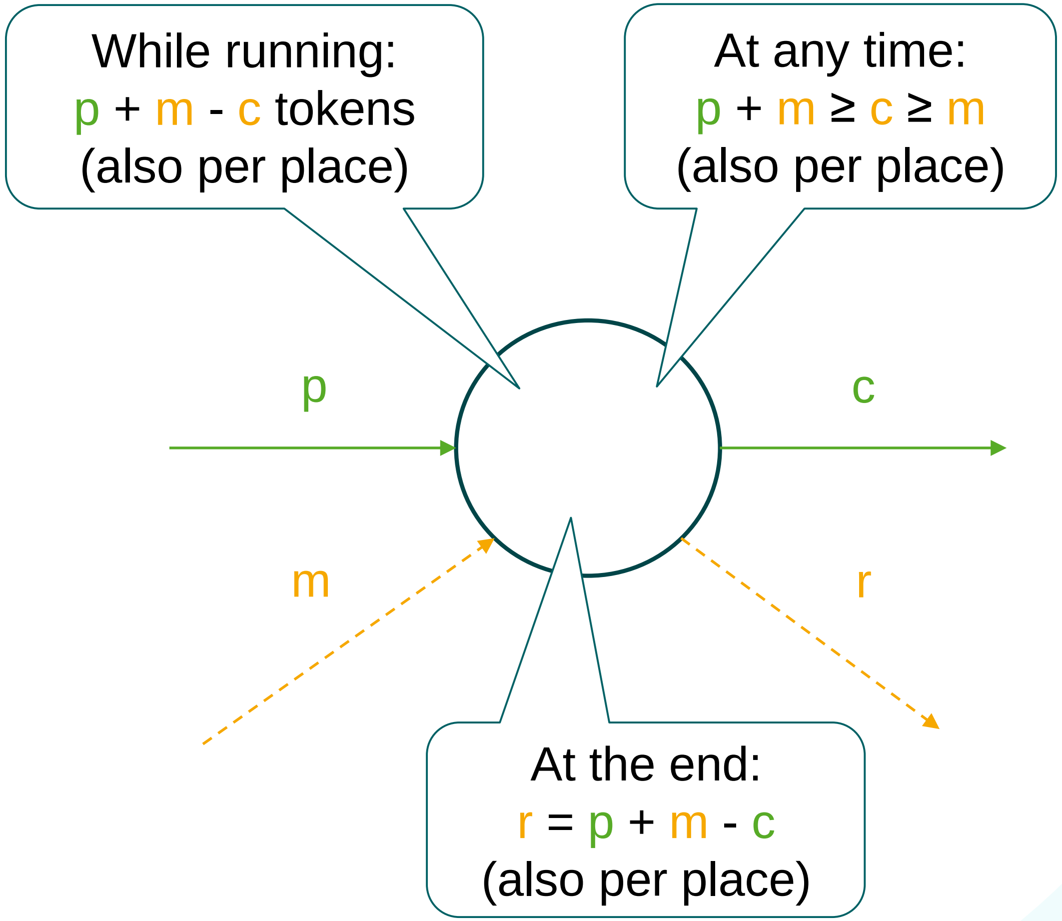
Token-Based Replay Approach

Use four counters:

- p = produced tokens
- c = consumed tokens
- m = missing tokens

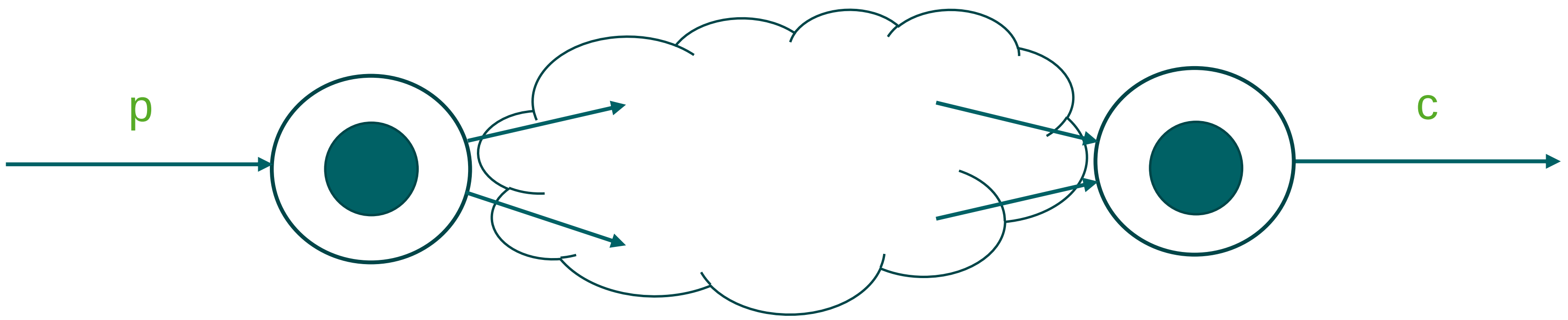
  (consumed while not there)
- r = remaining tokens

  (produced but not consumed)



Token-Based Replay Approach

- Initially, a token is produced for the start place – increment p
- Finally, a token is consumed from the end place (also if it is missing) – increment c (possibly also m)

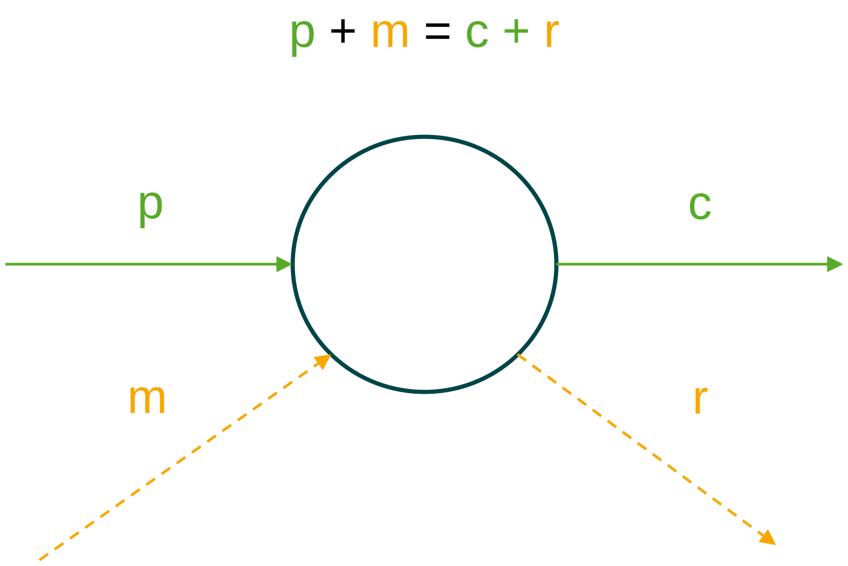


Diagnostics

Dimensions

- Per place or sum for all places in the model
- Per trace or sum for all traces in the event log

Four possible combinations



Four counters:

- p = produced tokens
- c = consumed tokens
- m = missing tokens
- r = remaining tokens

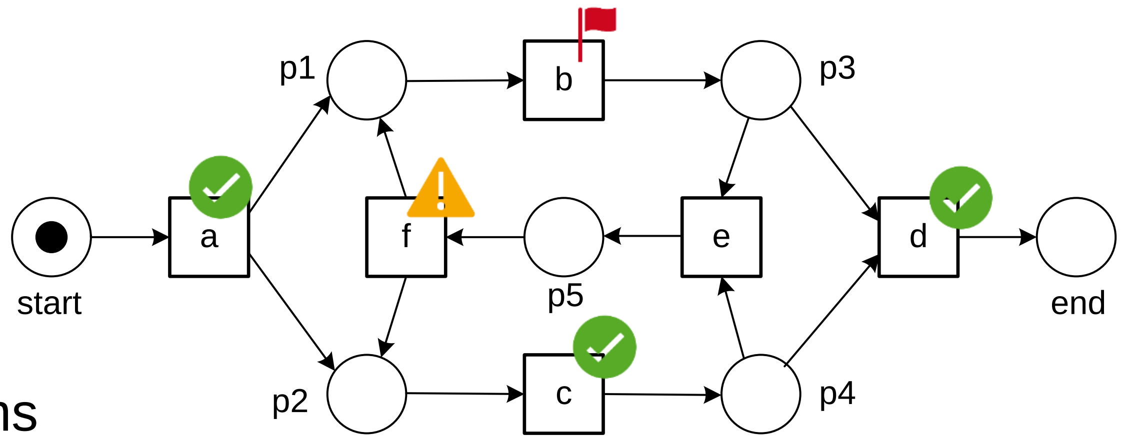
Summary Token-Based Replay Approach

- Pick a trace and initialize the process model by producing a token for the start place
- Fire the transition that corresponds to the next activity in the trace and update the counters for produced and consumed tokens. If not possible, add the required missing tokens first and update the missing tokens counter
- Repeat the above step until the end of the trace is reached
- Consume a token from the end place. If not possible, add the required missing token first and update the missing tokens counter
- Update the remaining tokens counter for each remaining token
- Compute:

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

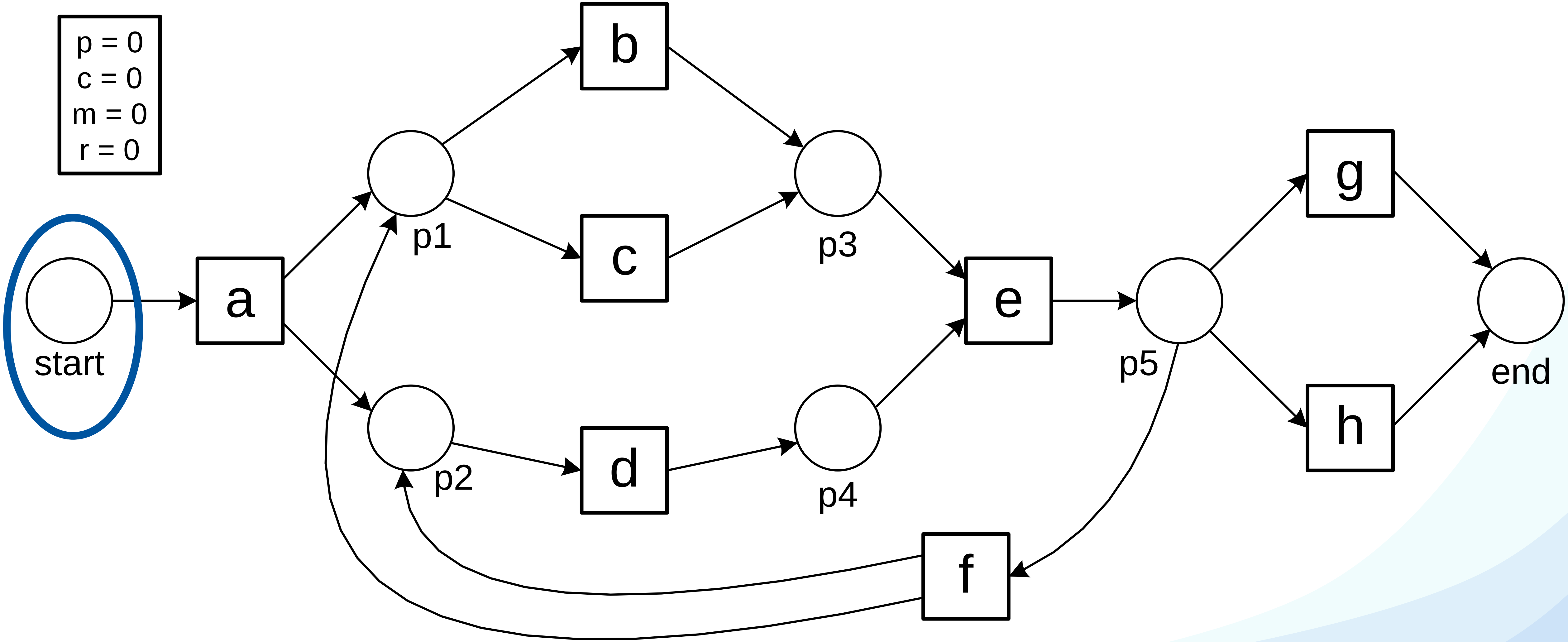
Supervised Process Mining

1. Token-Based Replay
2. **Token-Based Replay Examples**
3. Fitness at the Log Level
4. Generating Supervised Learning Problems

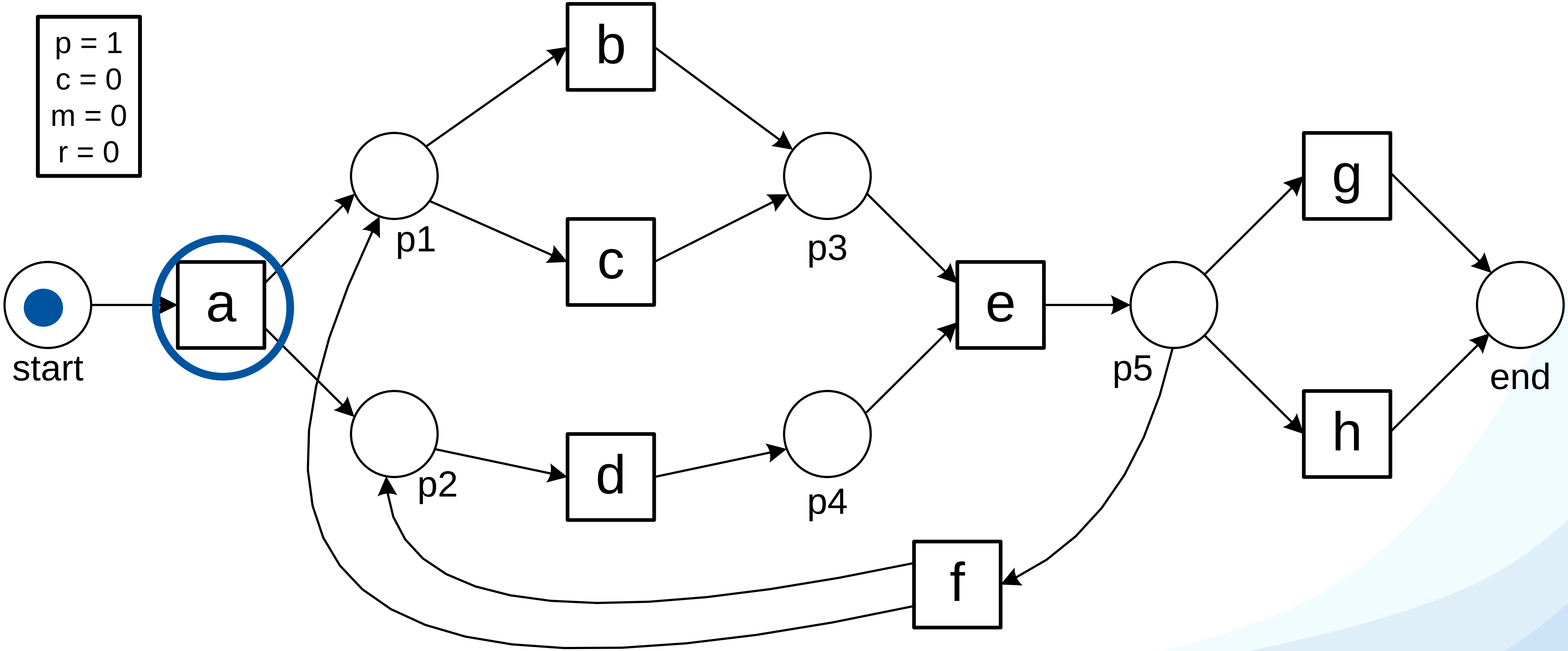


Replaying $\sigma_1 = \langle a, c, d, e, h \rangle$

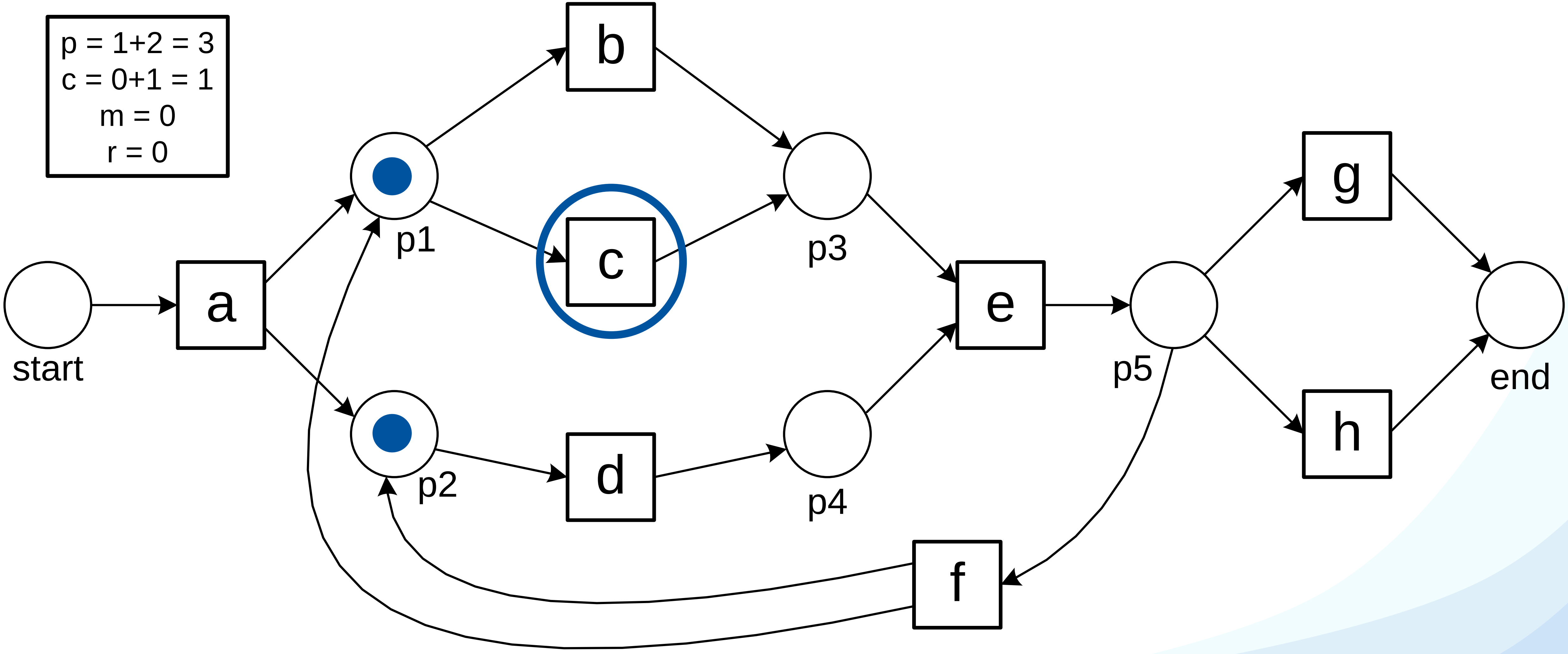
p = 0
c = 0
m = 0
r = 0



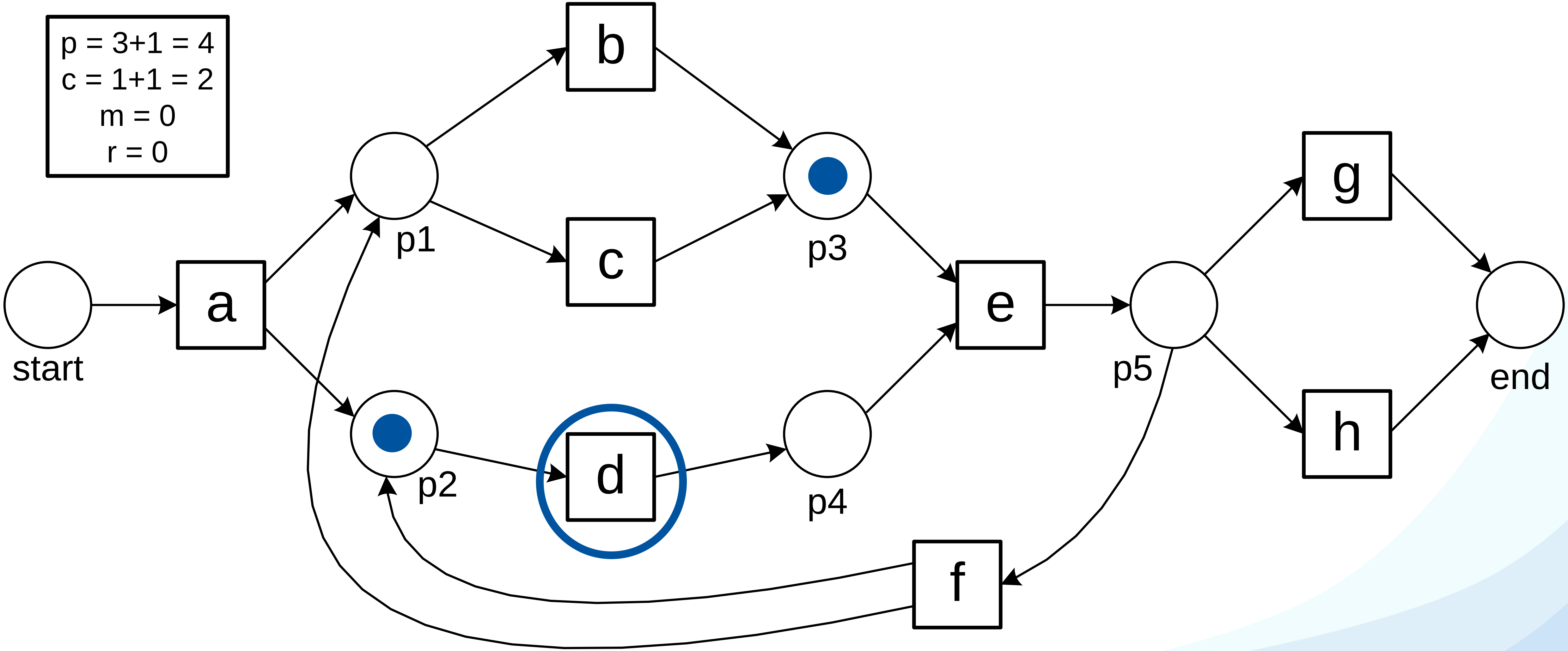
Replaying $\sigma_1 = \langle a, c, d, e, h \rangle$



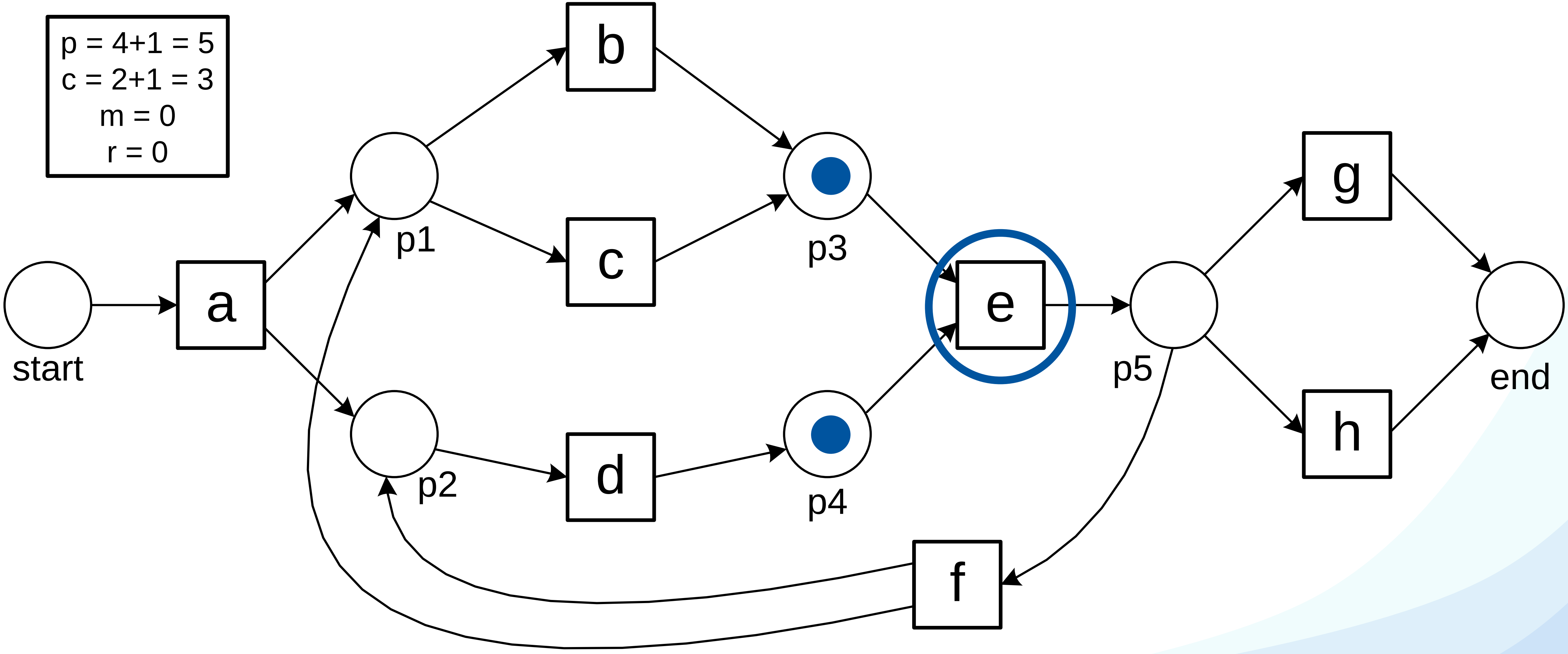
Replaying $\sigma_1 = \langle a, c, d, e, h \rangle$



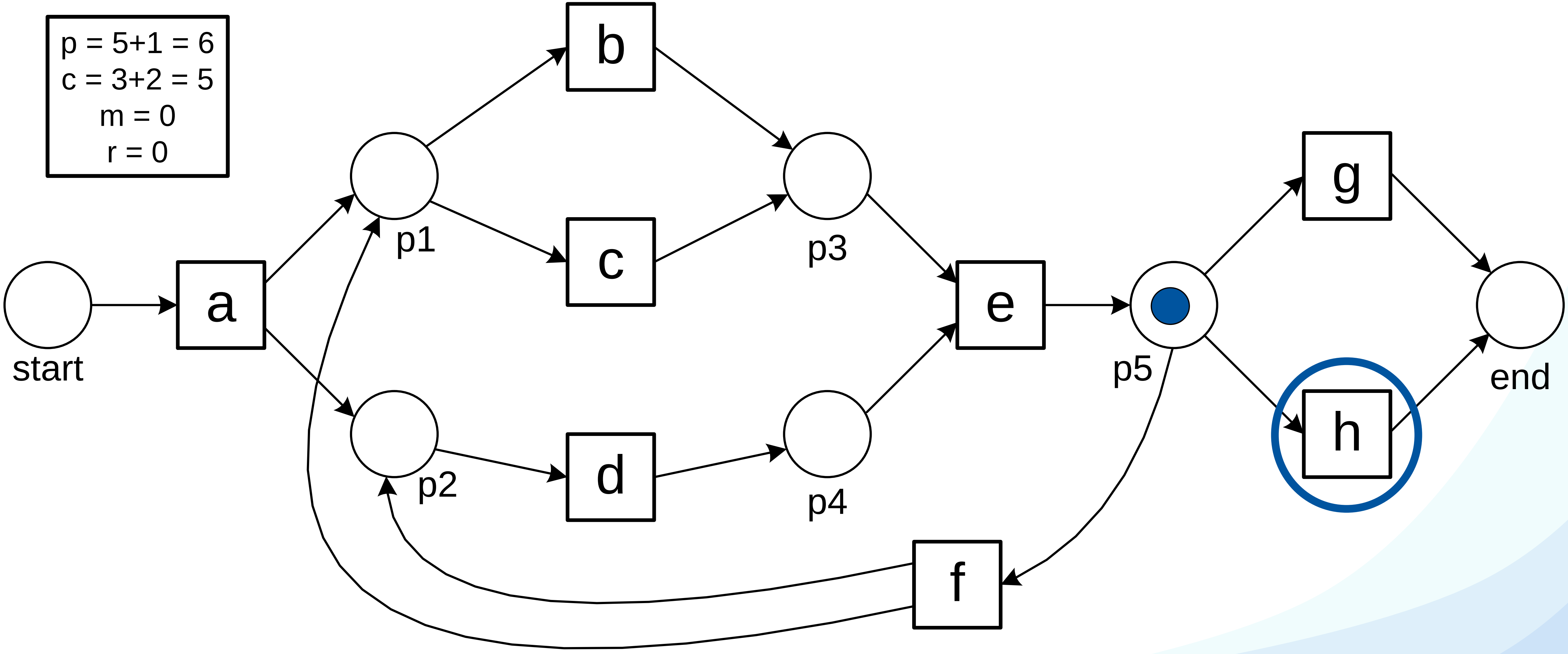
Replaying $\sigma_1 = \langle a, c, d, e, h \rangle$



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

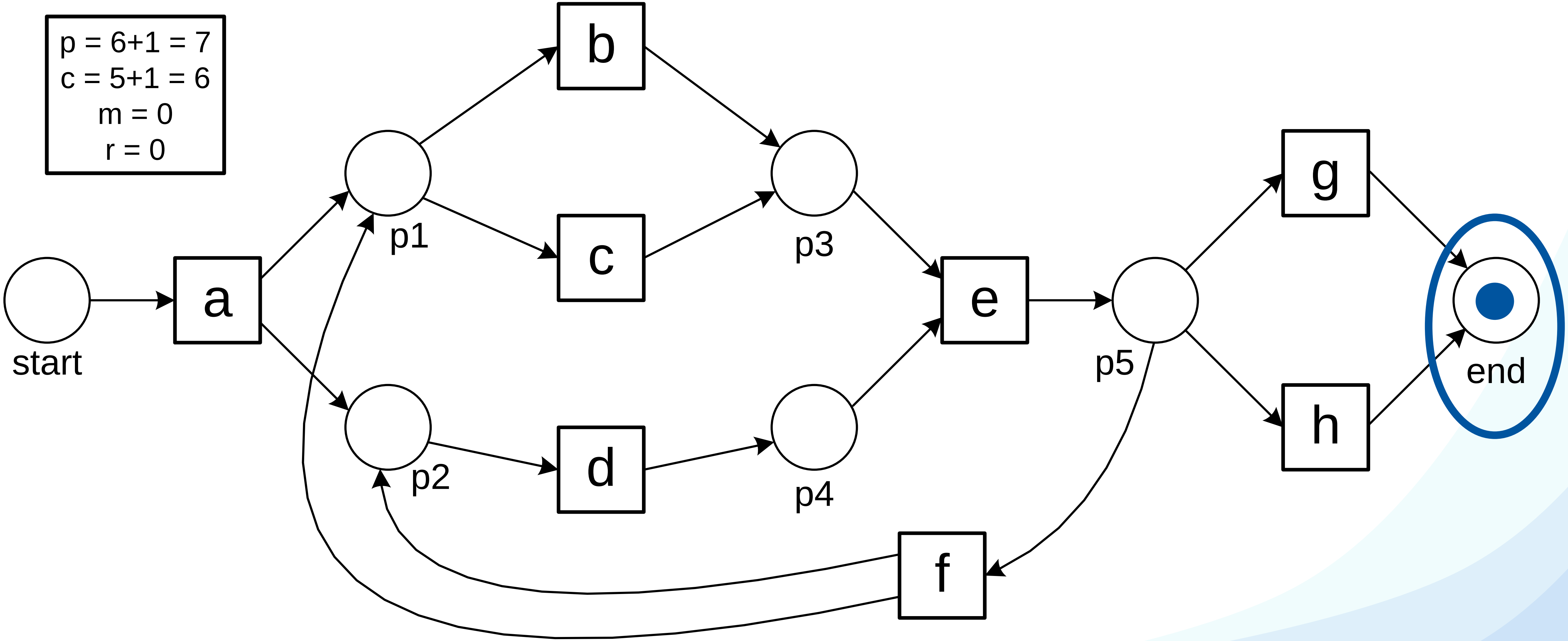


Replaying $\sigma_1 = \langle a, c, d, e, h \rangle$



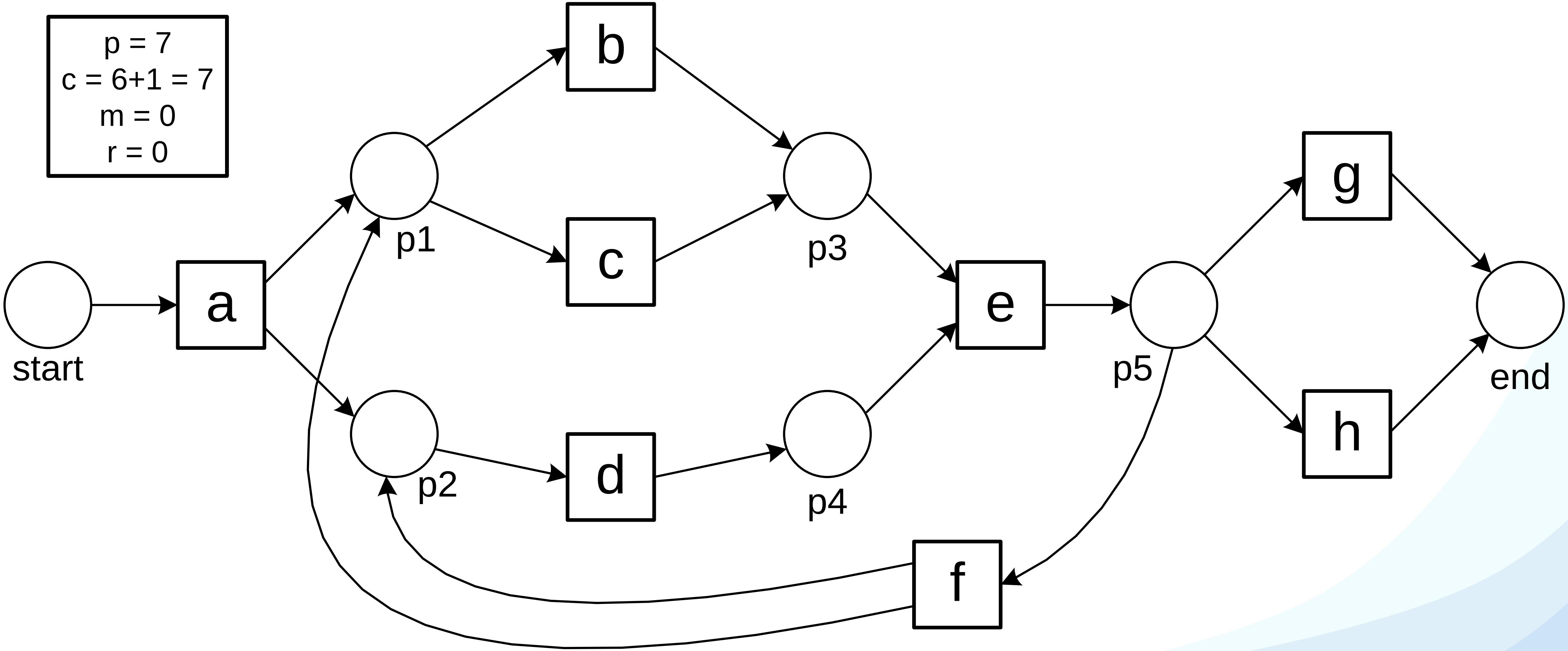
Replaying $\sigma_1 = \langle a, c, d, e, h \rangle$

p = 6+1 = 7
c = 5+1 = 6
m = 0
r = 0



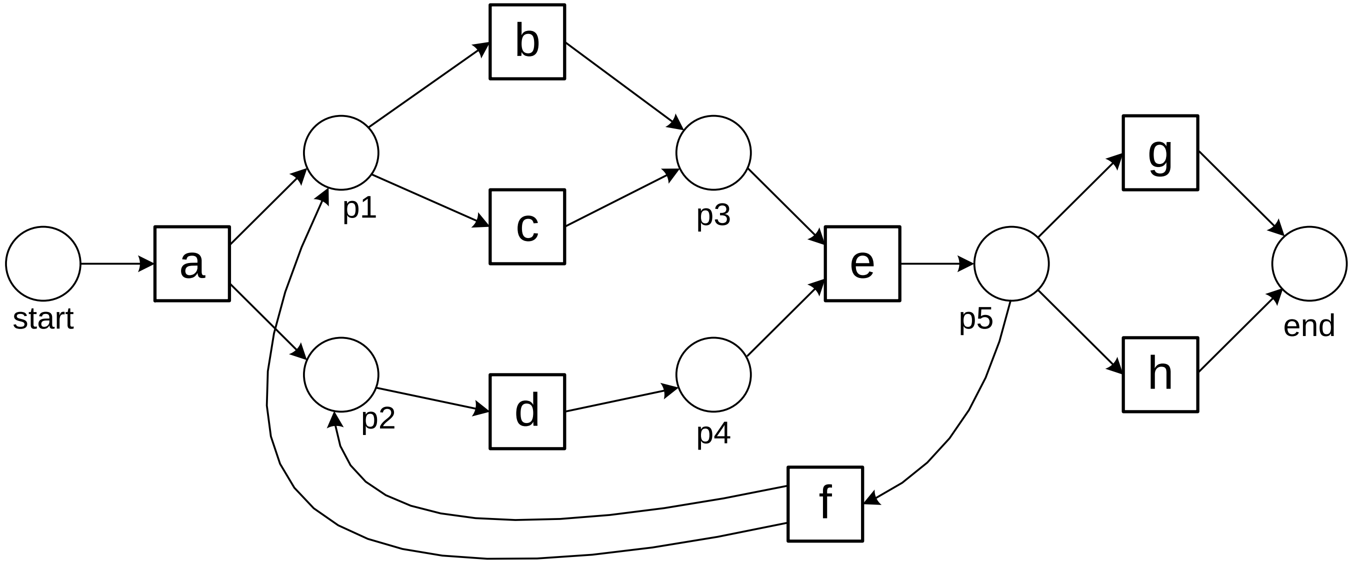
Replaying $\sigma_1 = \langle a, c, d, e, h \rangle$

p = 7
c = 6+1 = 7
m = 0
r = 0



Fitness at the Trace Level

p = 7
c = 7
m = 0
r = 0

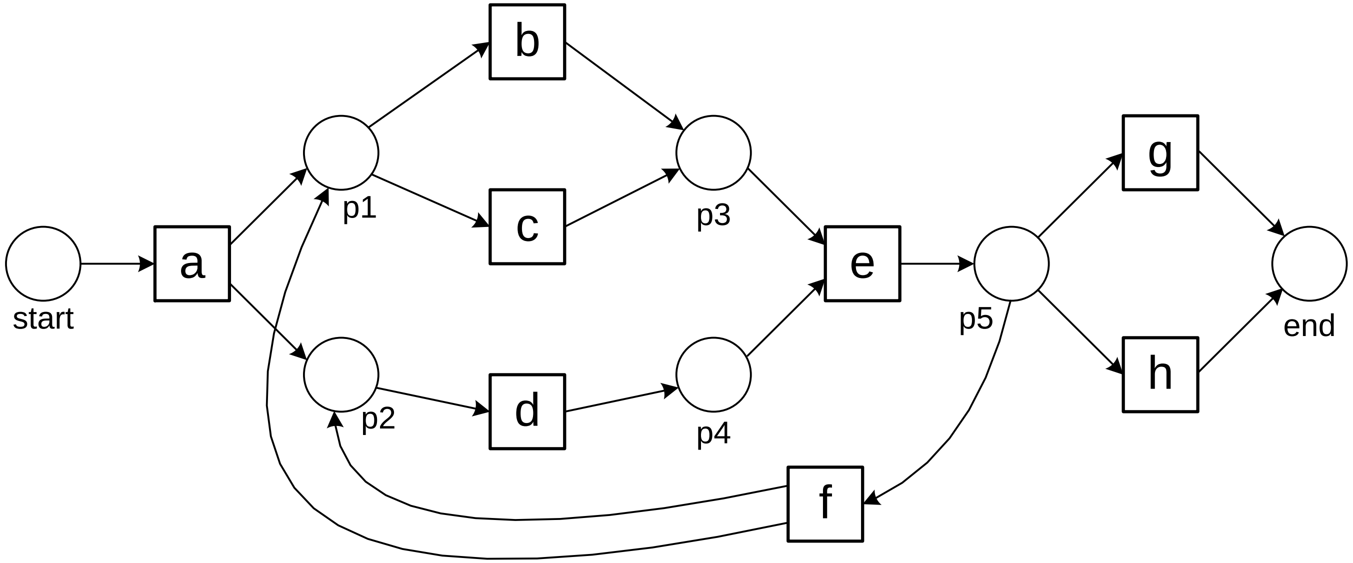


$$\sigma_1 = \langle a, c, d, e, h \rangle$$

$$\text{fitness}(\sigma_1, N) = \frac{1}{2}\left(1 - \frac{m}{c}\right) + \frac{1}{2}\left(1 - \frac{r}{p}\right)$$

Fitness at the Trace Level

p = 7
c = 7
m = 0
r = 0

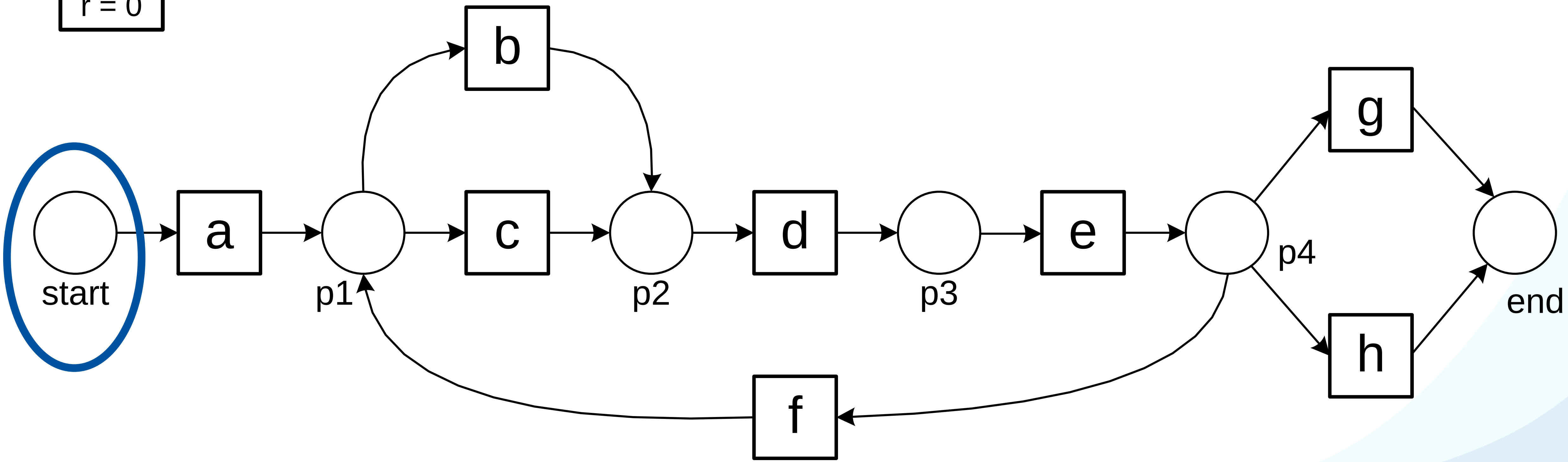


$$\sigma_1 = \langle a, c, d, e, h \rangle$$

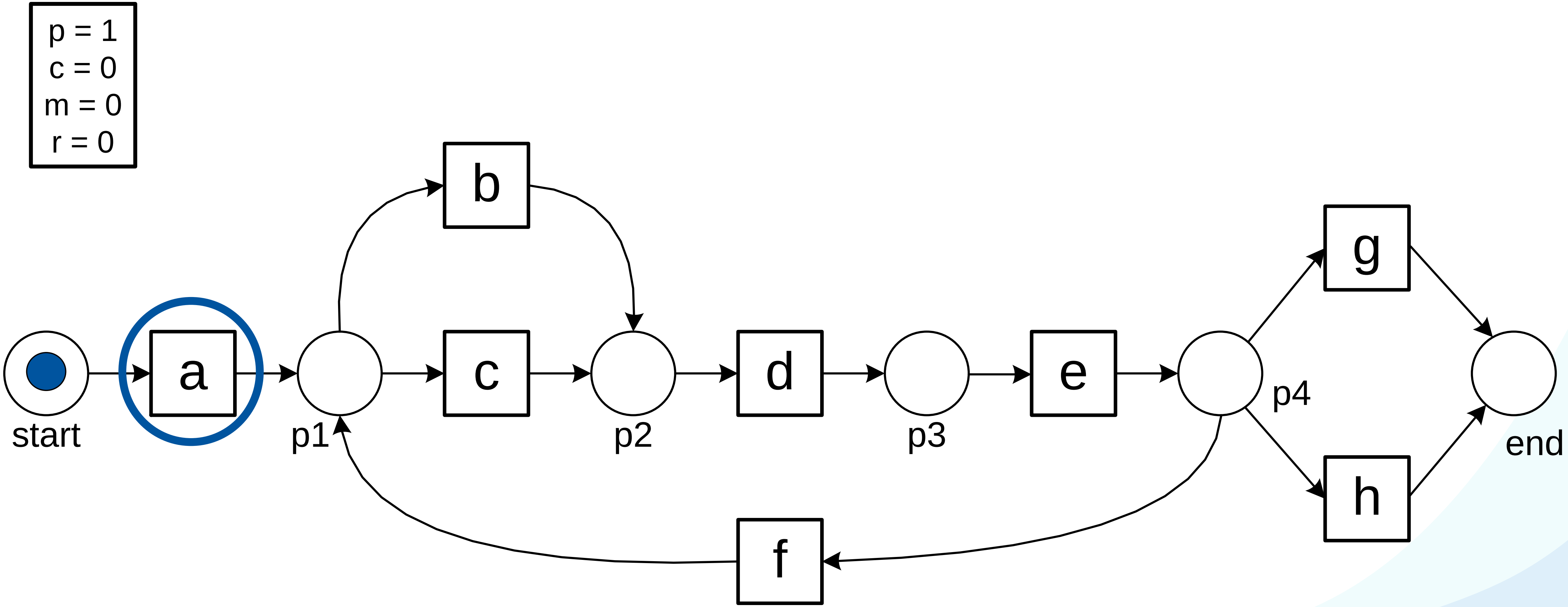
$$\text{fitness}(\sigma_1, N) = \frac{1}{2}\left(1 - \frac{0}{7}\right) + \frac{1}{2}\left(1 - \frac{0}{7}\right) = 1$$

Replaying $\sigma_2 = \langle a, d, c, e, h \rangle$

p = 0
c = 0
m = 0
r = 0

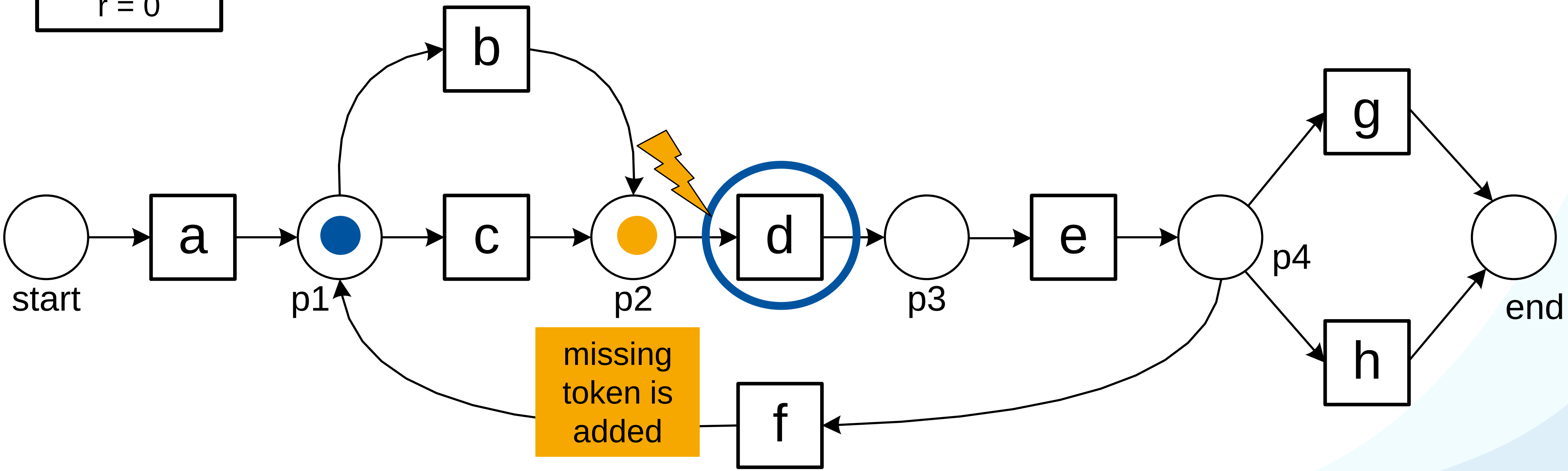


Replaying $\sigma_2 = \langle a, d, c, e, h \rangle$



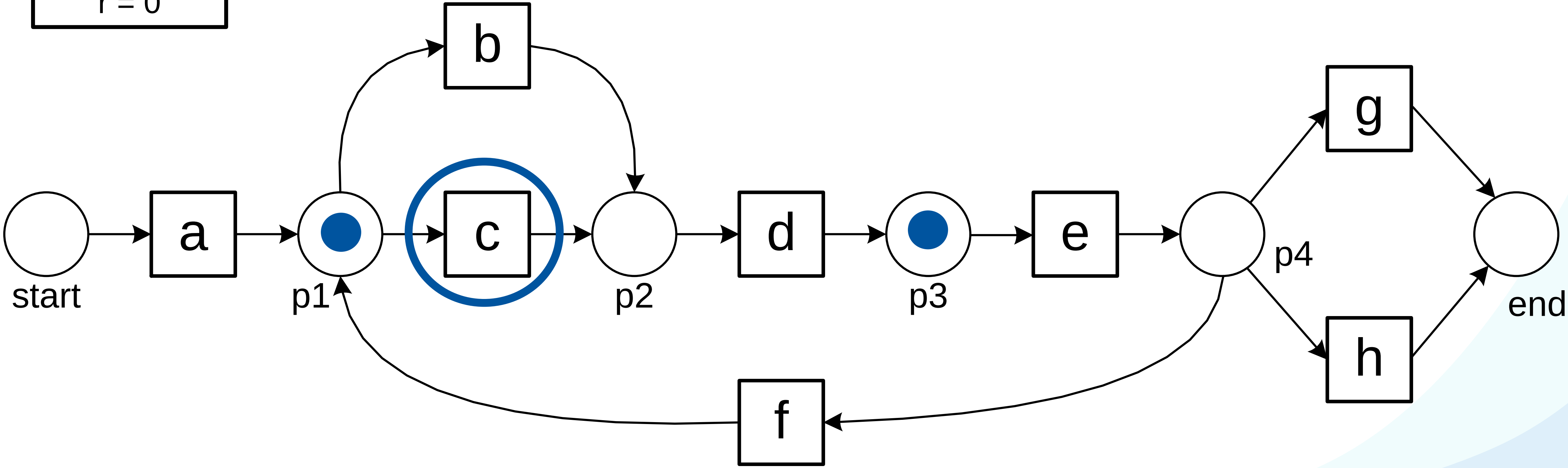
Replaying $\sigma_2 = \langle a, d, c, e, h \rangle$

p = 1+1 = 2
c = 0+1 = 1
m = 0
r = 0



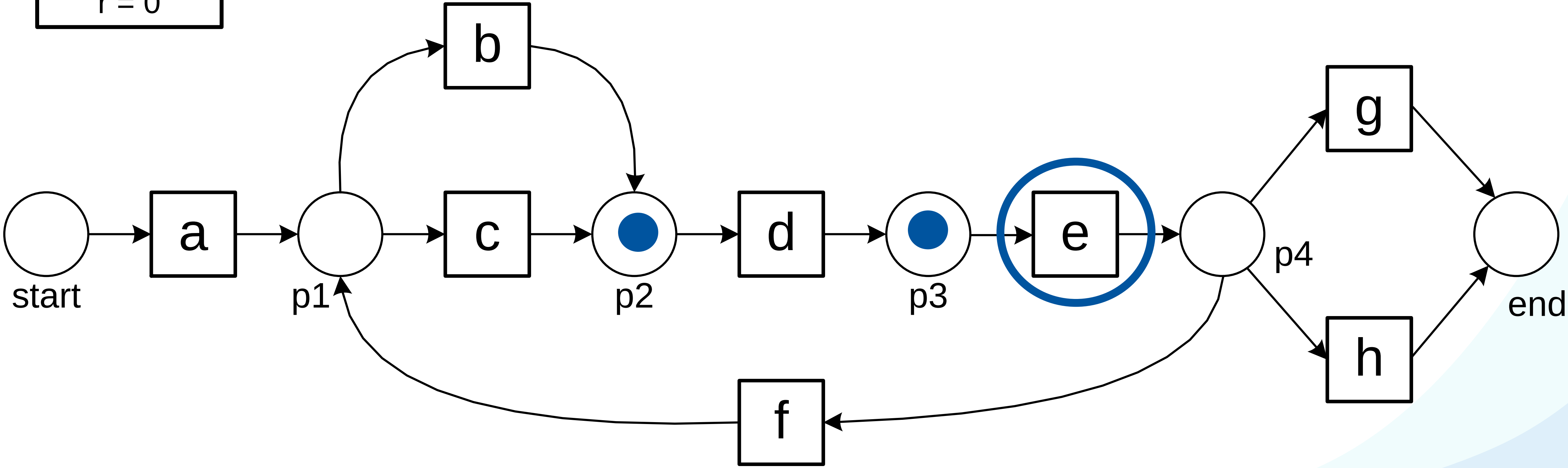
Replaying $\sigma_2 = \langle a, d, c, e, h \rangle$

p = 2+1 = 3
c = 1+1 = 2
m = 0+1 = 1
r = 0



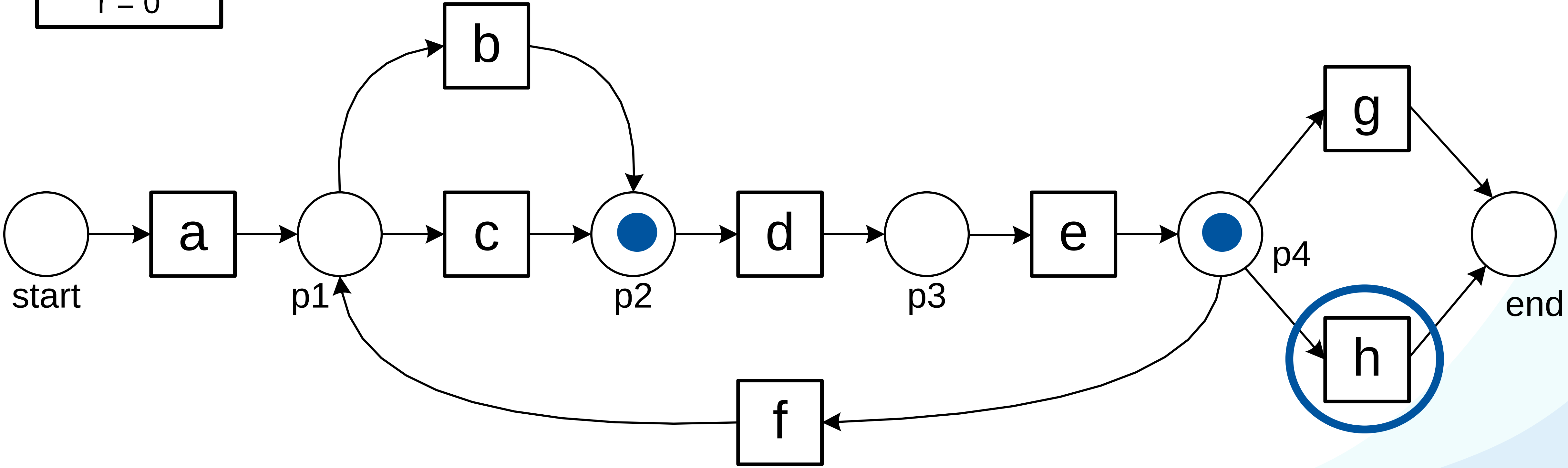
Replaying $\sigma_2 = \langle a, d, c, e, h \rangle$

p = 3+1 = 4
c = 2+1 = 3
m = 1
r = 0



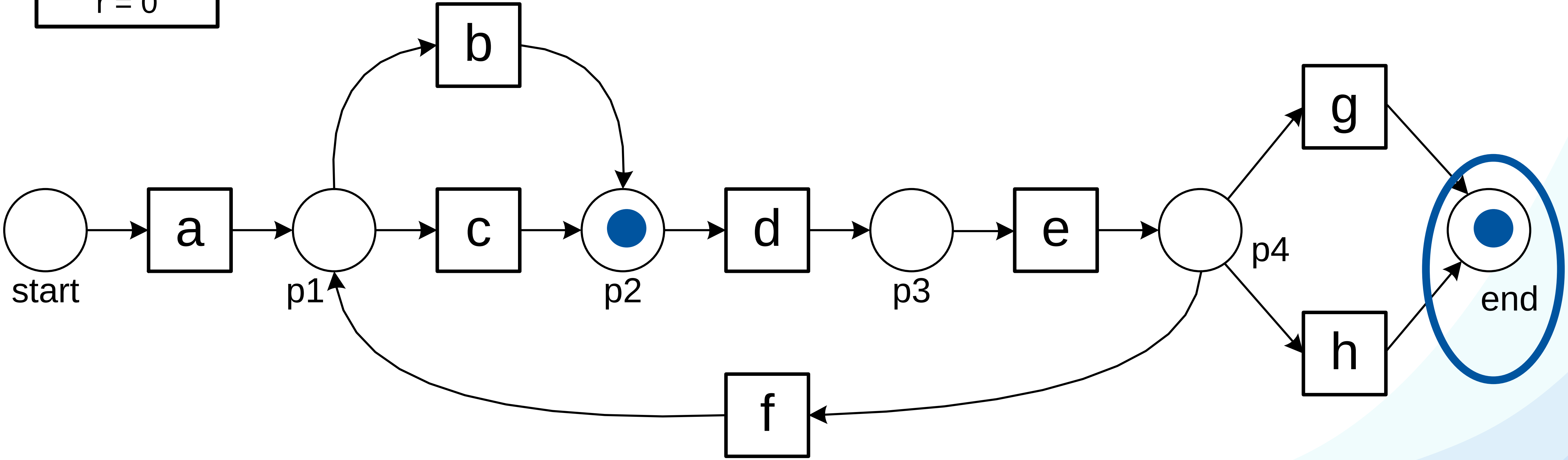
Replaying $\sigma_2 = \langle a, d, c, e, h \rangle$

p = 4+1 = 5
c = 3+1 = 4
m = 1
r = 0



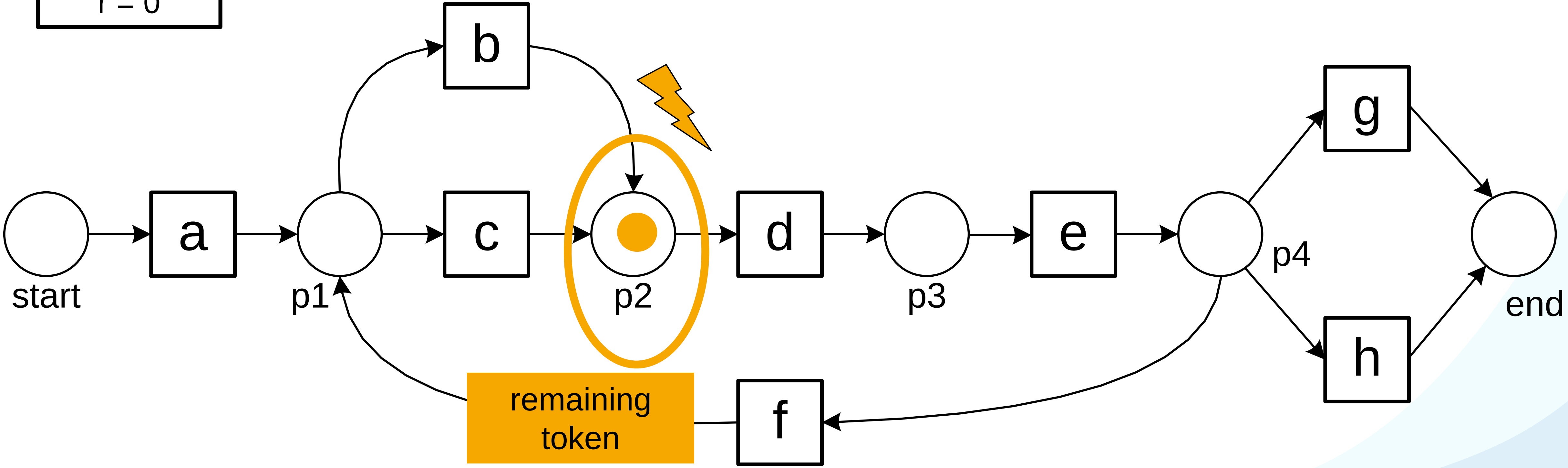
Replaying $\sigma_2 = \langle a, d, c, e, h \rangle$

p = 5+1 = 6
c = 4+1 = 5
m = 1
r = 0



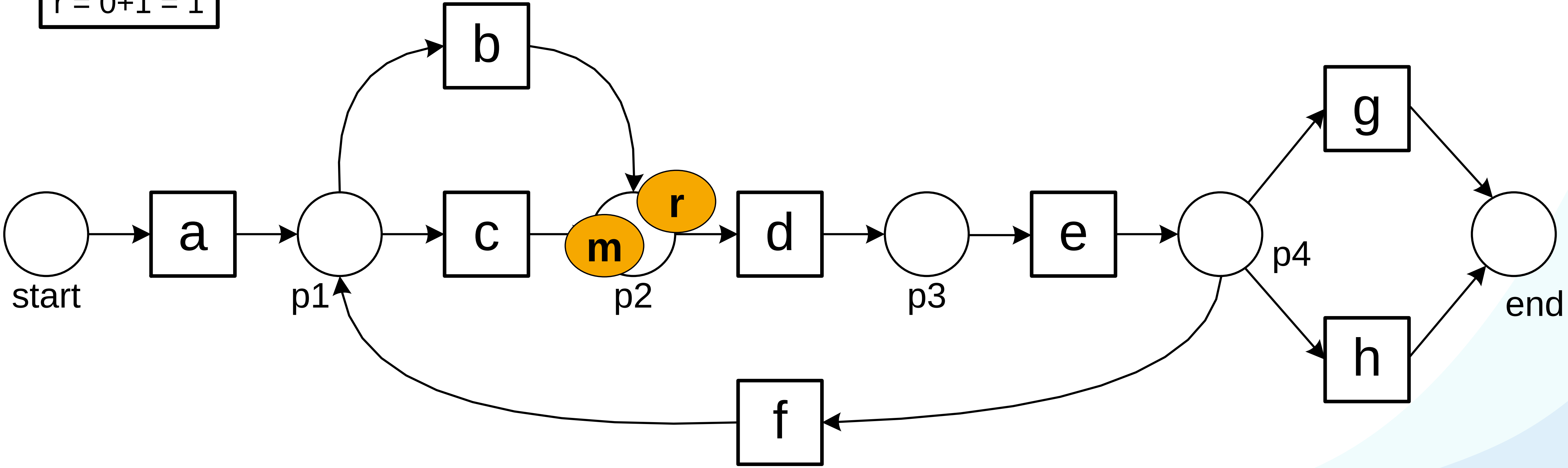
Replaying $\sigma_2 = \langle a, d, c, e, h \rangle$

p = 6
c = 5+1 = 6
m = 1
r = 0



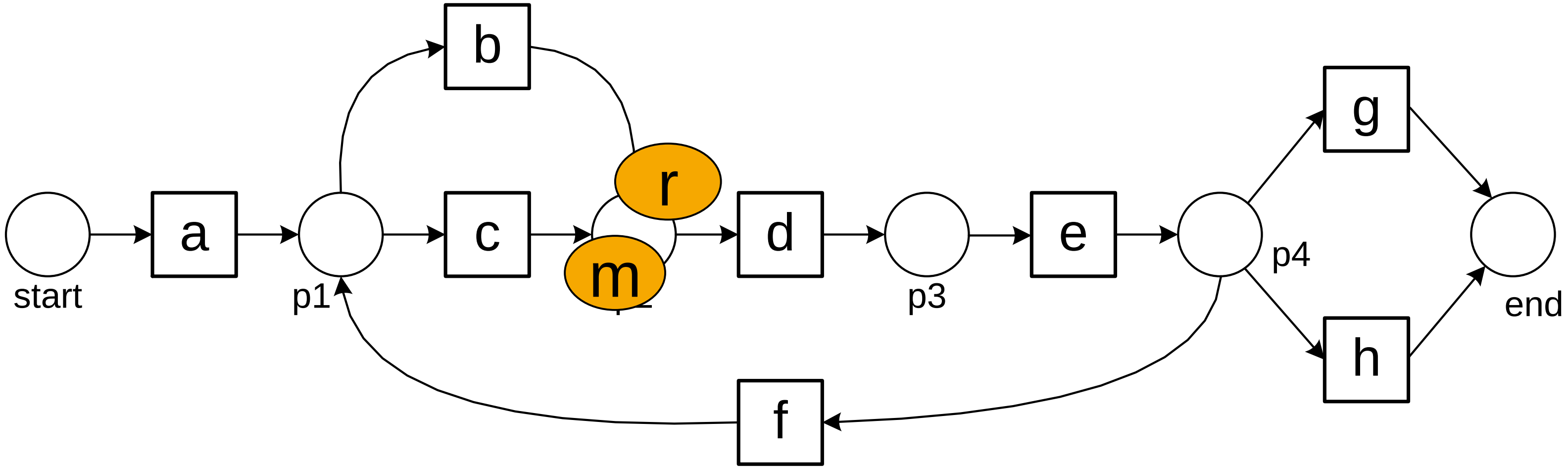
Replaying $\sigma_2 = \langle a, d, c, e, h \rangle$

p = 6
c = 6
m = 1
r = 0+1 = 1



Fitness at the Trace Level

p = 6
c = 6
m = 1
r = 1

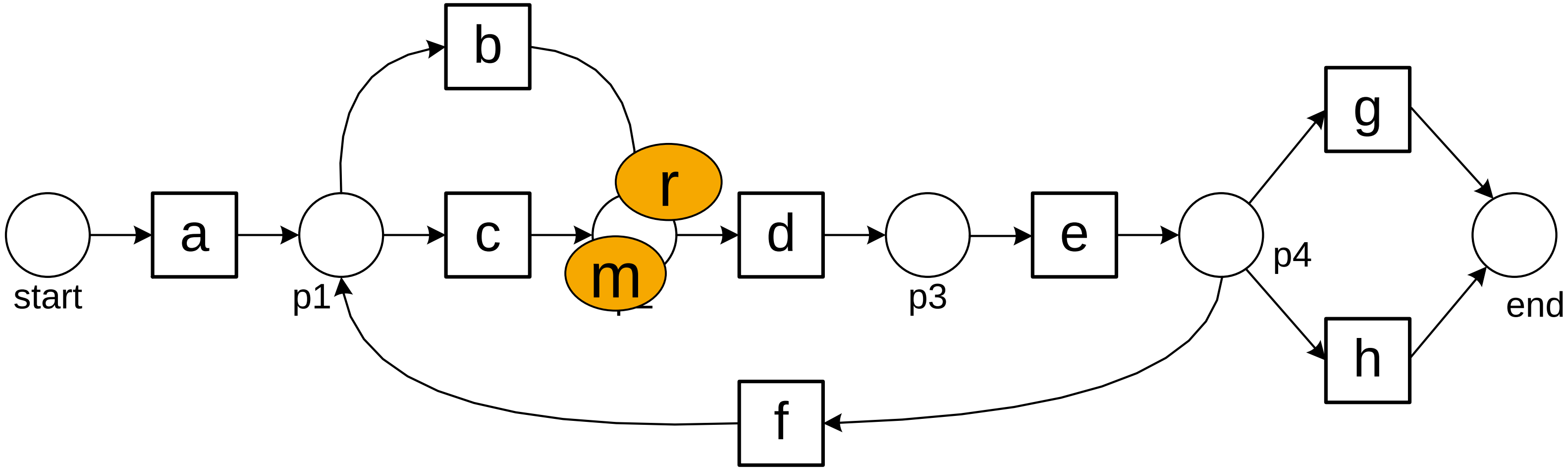


$$\sigma_2 = \langle a, d, c, e, h \rangle$$

$$\text{fitness}(\sigma_2, N) = \frac{1}{2}\left(1 - \frac{m}{c}\right) + \frac{1}{2}\left(1 - \frac{r}{p}\right)$$

Fitness at the Trace Level

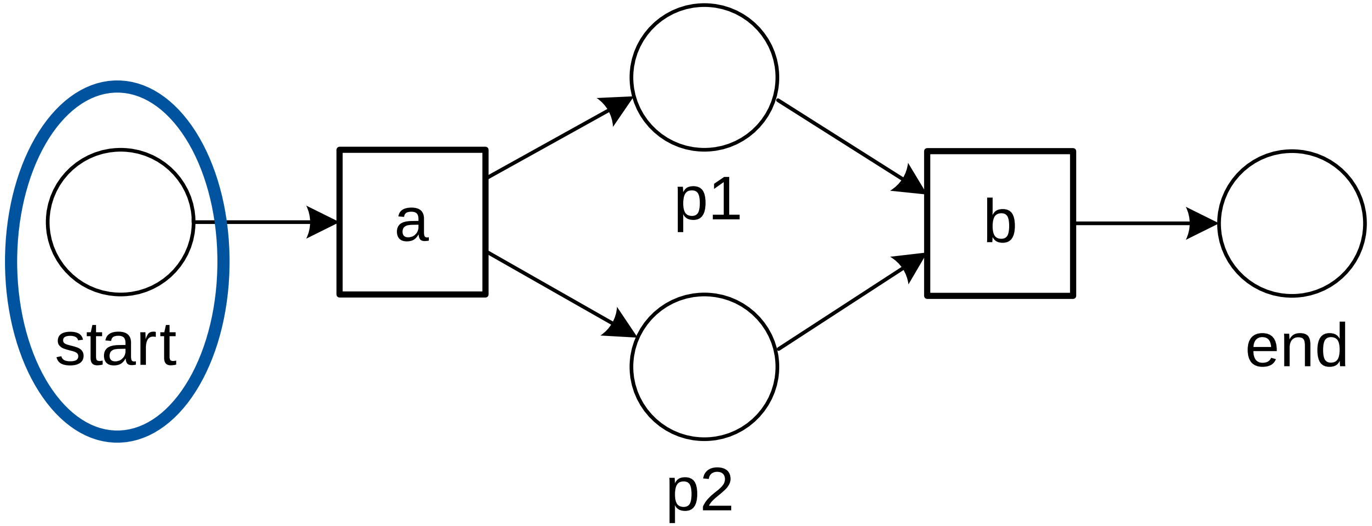
p = 6
c = 6
m = 1
r = 1



$$\sigma_2 = \langle a, d, c, e, h \rangle$$

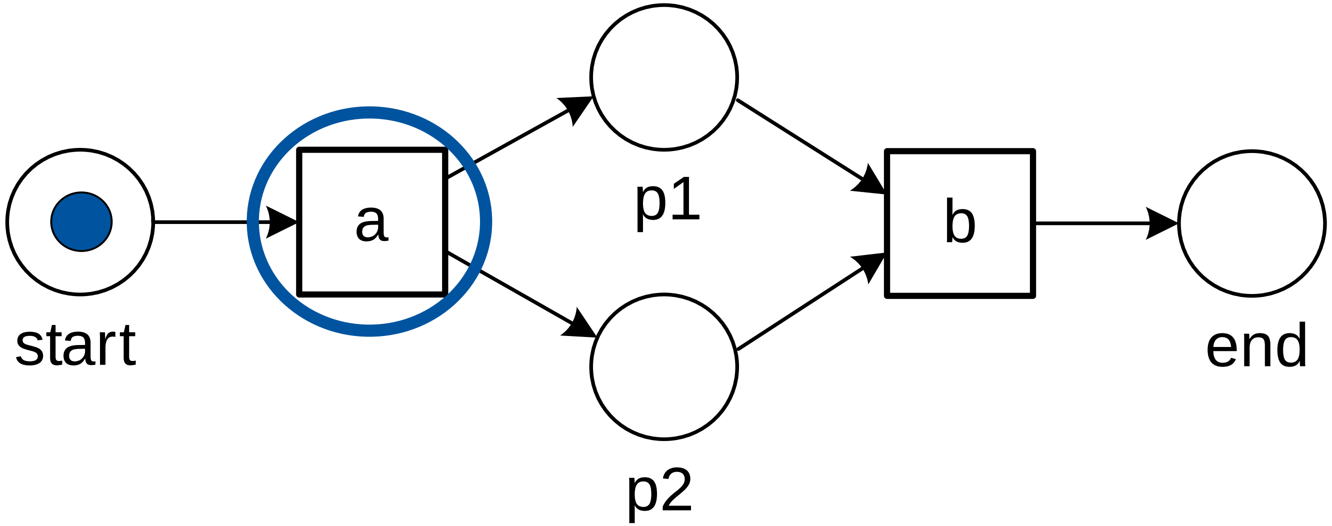
$$\text{fitness}(\sigma_2, N) = \frac{1}{2}\left(1 - \frac{1}{6}\right) + \frac{1}{2}\left(1 - \frac{1}{6}\right) = \frac{5}{6} \approx 0.83$$

Replaying $\sigma_3 = \langle a \rangle$



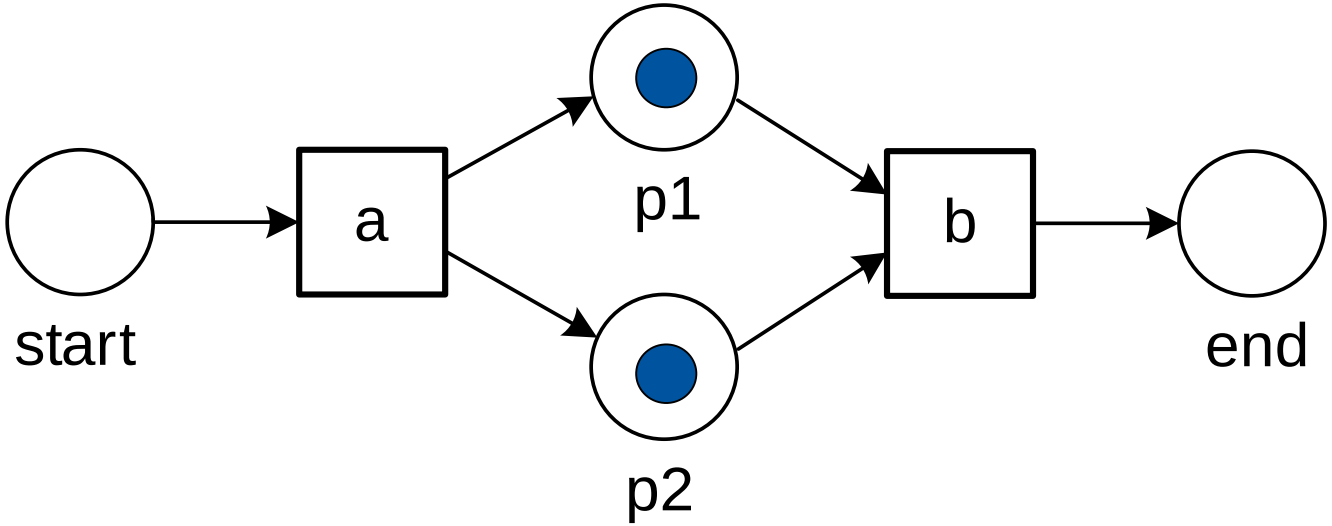
p = 0
c = 0
m = 0
r = 0

Replaying $\sigma_3 = \langle a \rangle$



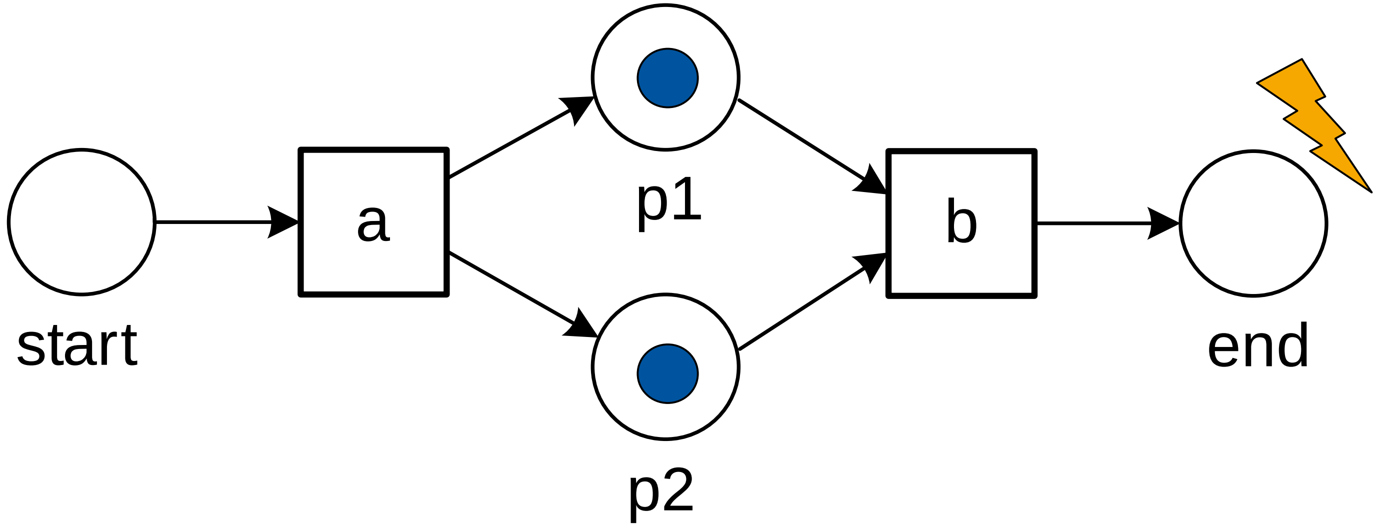
p = 0 + 1 = 1
c = 0
m = 0
r = 0

Replaying $\sigma_3 = \langle a \rangle$



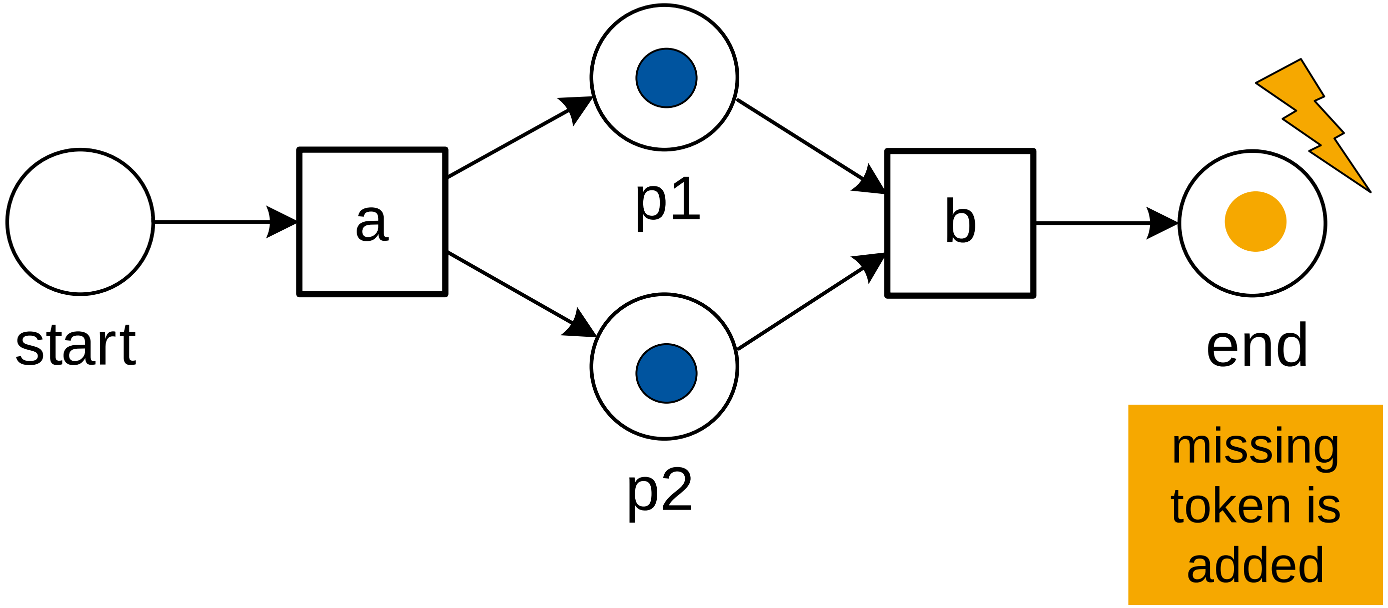
$p = 1 + 2 = 3$
$c = 0 + 1 = 1$
$m = 0$
$r = 0$

Replaying $\sigma_3 = \langle a \rangle$



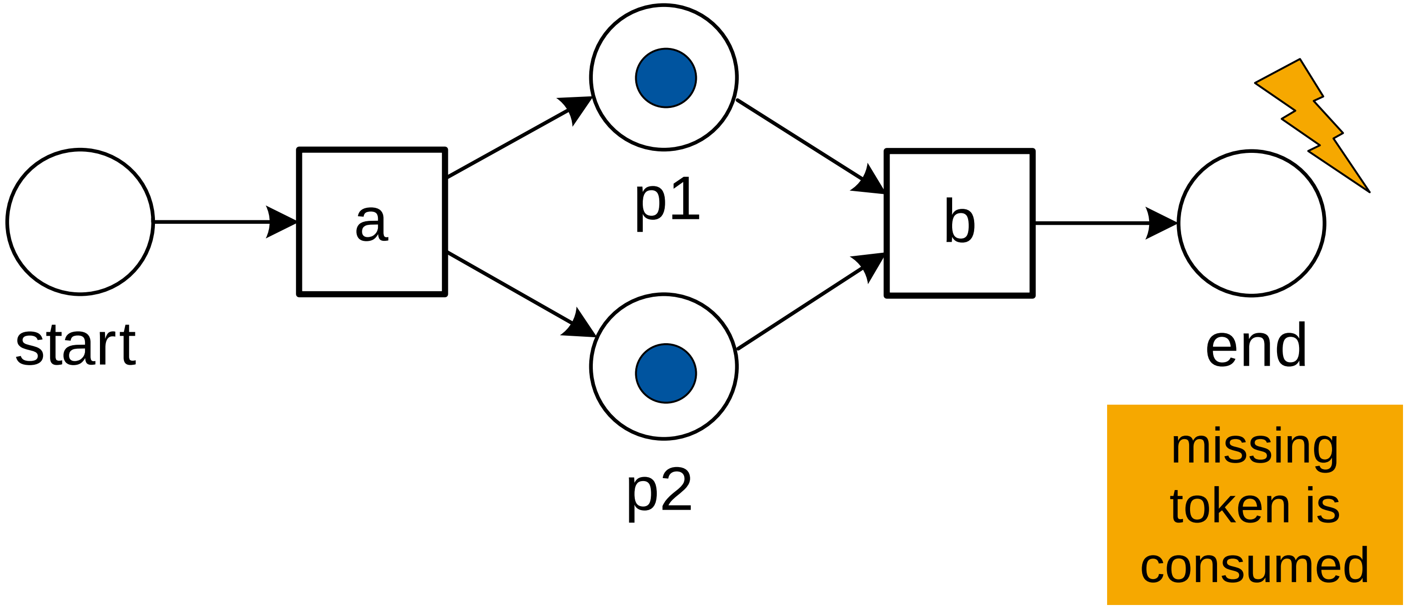
p = 3
c = 1
m = 0
r = 0

Replaying $\sigma_3 = \langle a \rangle$



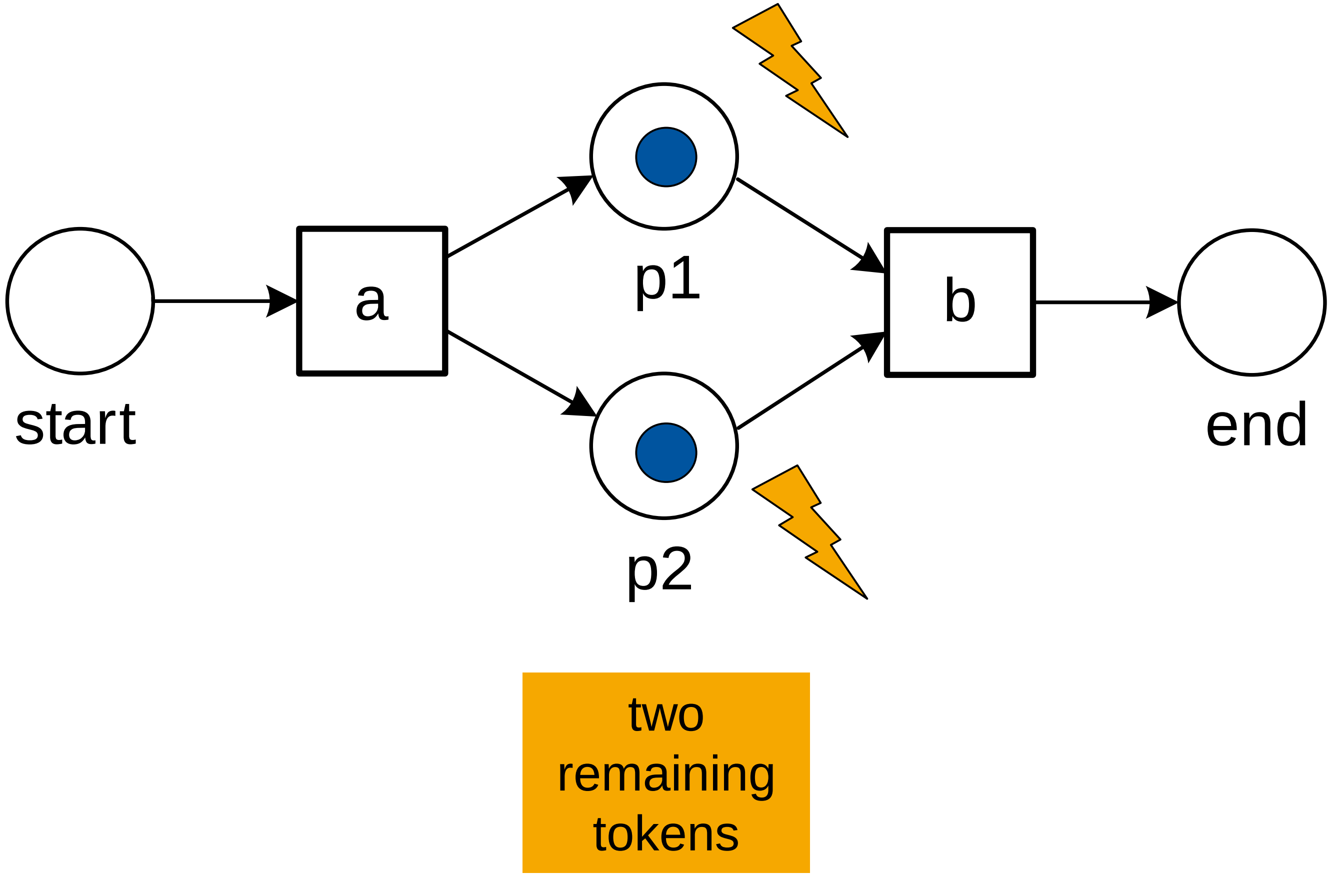
p = 3
c = 1
m = 0 + 1 = 1
r = 0

Replaying $\sigma_3 = \langle a \rangle$



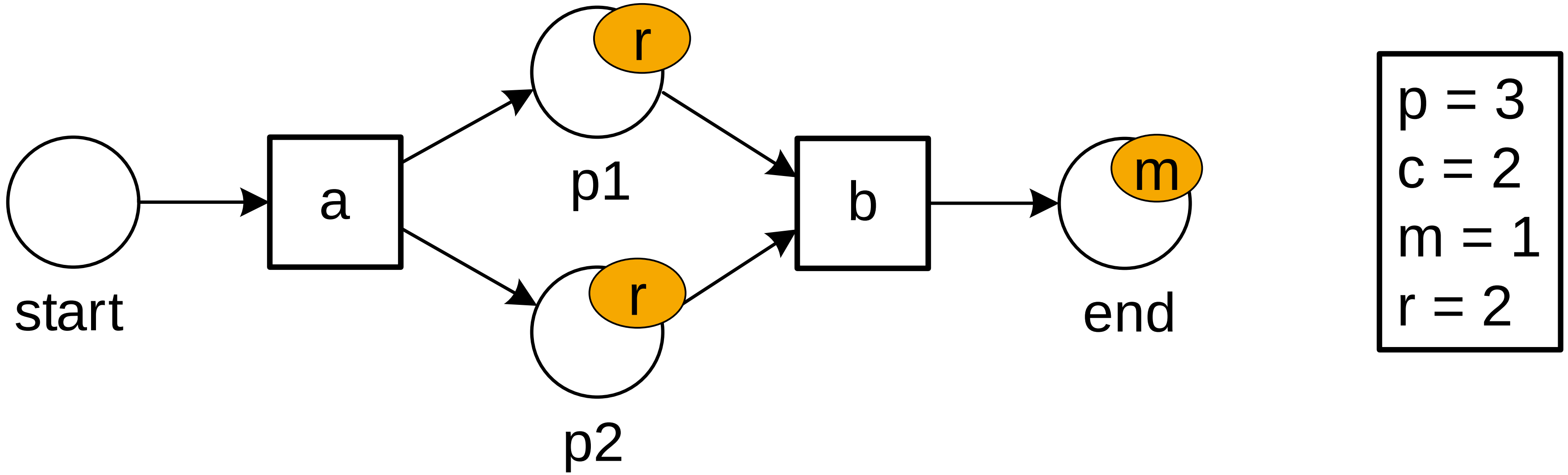
p = 3
c = 1 + 1 = 2
m = 1
r = 0

Replaying $\sigma_3 = \langle a \rangle$



p = 3
c = 2
m = 1
r = 0 + 2 = 2

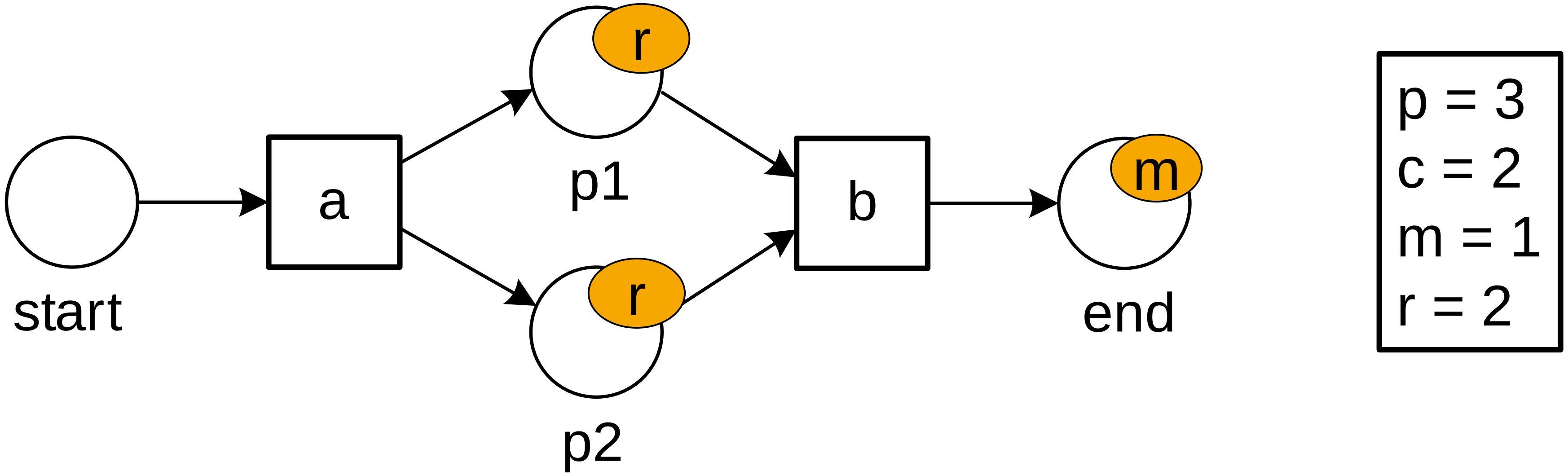
Fitness at the Trace Level



$$\sigma_3 = \langle a \rangle$$

$$\text{fitness}(\sigma_3, N) = \frac{1}{2}\left(1 - \frac{m}{c}\right) + \frac{1}{2}\left(1 - \frac{r}{p}\right)$$

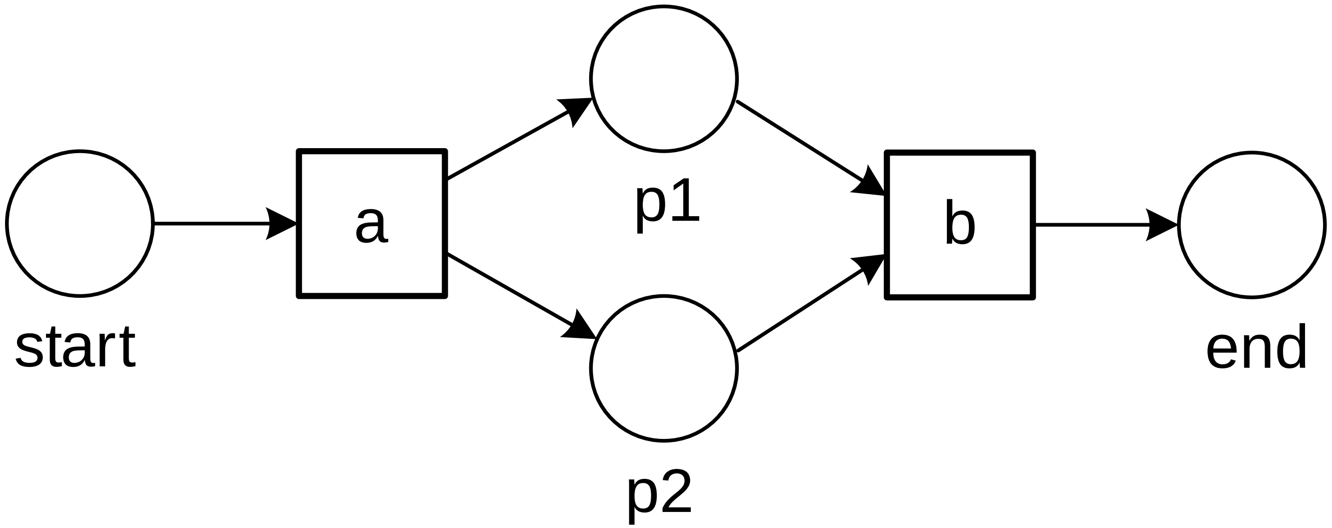
Fitness at the Trace Level



$$\sigma_3 = \langle a \rangle$$

$$\text{fitness}(\sigma_3, N) = \frac{1}{2}\left(1 - \frac{1}{2}\right) + \frac{1}{2}\left(1 - \frac{2}{3}\right) = \frac{5}{12} \approx 0.42$$

What is the Worst Case Scenario?



$$\text{fitness}(\sigma_{bad}, N) = \frac{1}{2}\left(1 - \frac{m}{c}\right) + \frac{1}{2}\left(1 - \frac{r}{p}\right) = 0$$

$$\boxed{\begin{array}{l} p = r \\ c = m \end{array}}$$

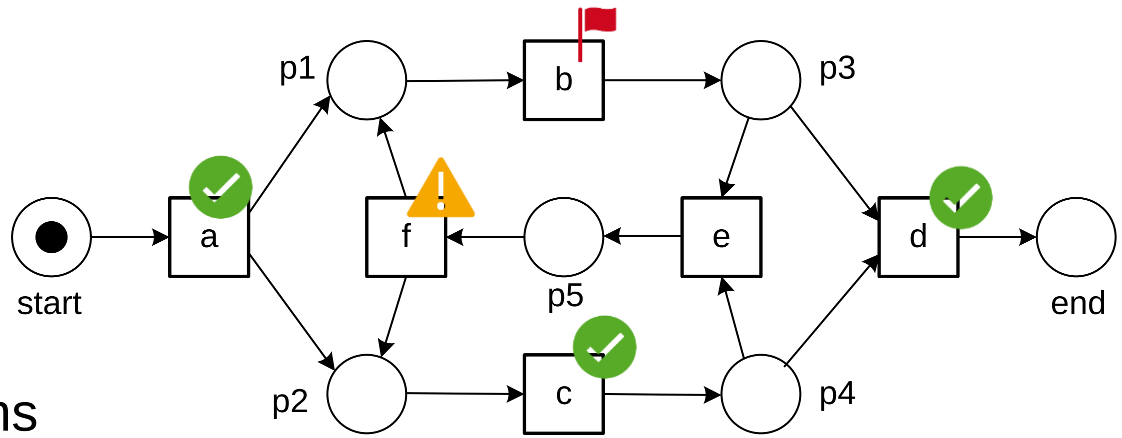
Supervised Process Mining

1. Token-Based Replay

2. Token-Based Replay Examples

3. **Fitness at the Log Level**

4. Generating Supervised Learning Problems



Fitness at the Log Level

All missing tokens

All consumed tokens

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

All remaining tokens

All produced tokens

Less scary than it looks:

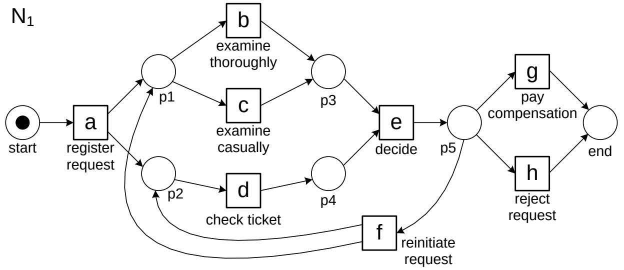
The sums of p, c, m, and r over all traces in the entire event log …

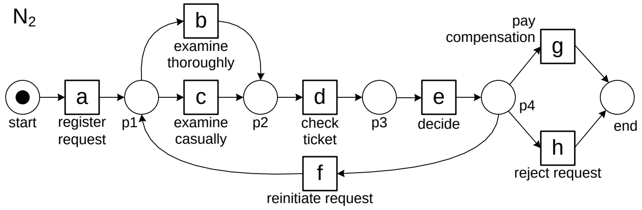
Computing Fitness

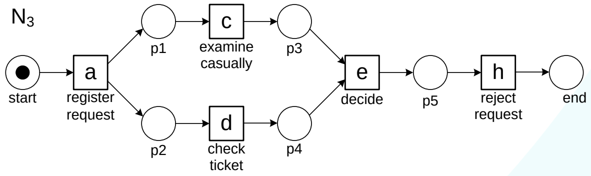
#	trace
455	acdeh
191	abdeg
177	adceh
144	abdeh
111	acdeg
82	adceg
56	adbeh
47	acdefdbeh
38	adbeg
33	acdefbdeh
14	acdefbdeg
11	acdefdbeg
9	adcefcdeh
8	adcefdbeh
5	adcefbdeg
3	acdefbdefdbeg
2	adcefdbeg
2	adcefbdefbdeg
1	adcefdbefbdeh
1	adbefbdefdbeg
1	adcefdbefcdefdbeg
1391	

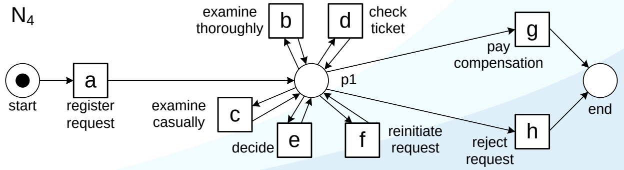
$$\text{fitness}\ (L, N) = \frac{1}{2}\left(1 - \frac{\sum_{\sigma\in L} L(\sigma)\times m_{N,\sigma}}{\sum_{\sigma\in L} L(\sigma)\times c_{N,\sigma}}\right) + \frac{1}{2}\left(1 - \frac{\sum_{\sigma\in L} L(\sigma)\times r_{N,\sigma}}{\sum_{\sigma\in L} L(\sigma)\times p_{N,\sigma}}\right)$$

$$\boxed{\begin{aligned} \text{fitness}\ (L_{\text{full}}\ , N_1) &= 1 \\ \text{fitness}\ (L_{\text{full}}\ , N_2) &= 0.9504 \\ \text{fitness}\ (L_{\text{full}}\ , N_3) &= 0.8797 \\ \text{fitness}\ (L_{\text{full}}\ , N_4) &= 1 \end{aligned}}$$





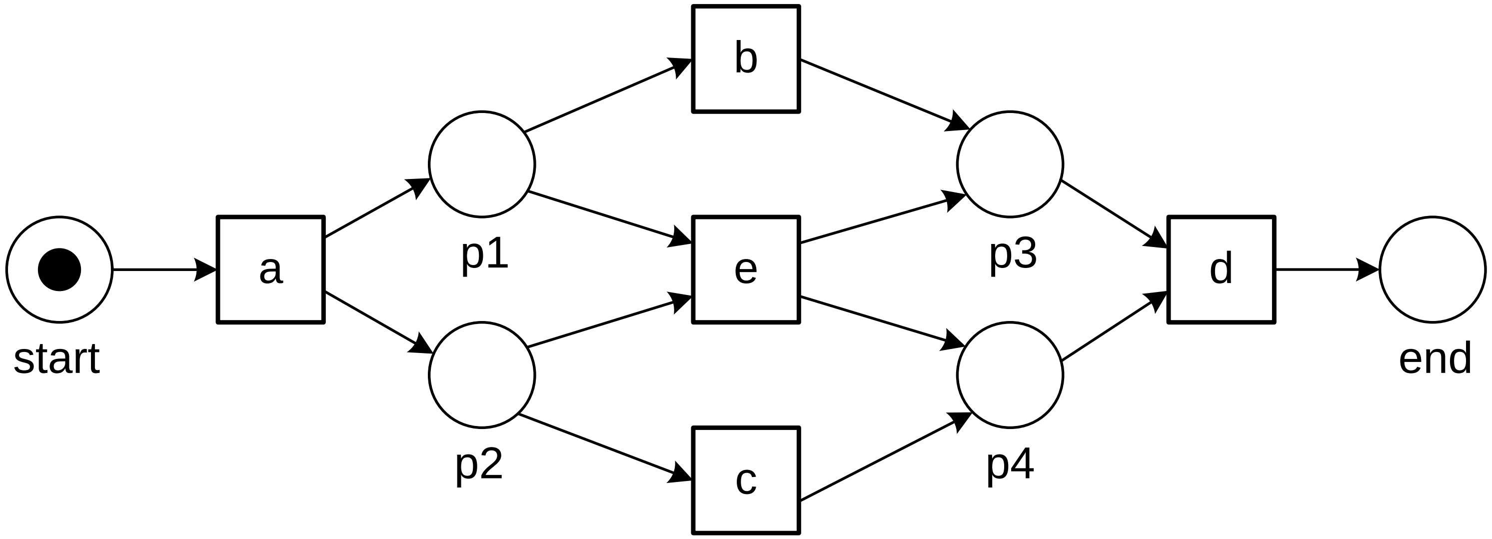




Computing Fitness – Example

Trace	Frequency
abcd	10
acbd	10
aed	10
abd	2
acd	1
ad	1
abbd	1

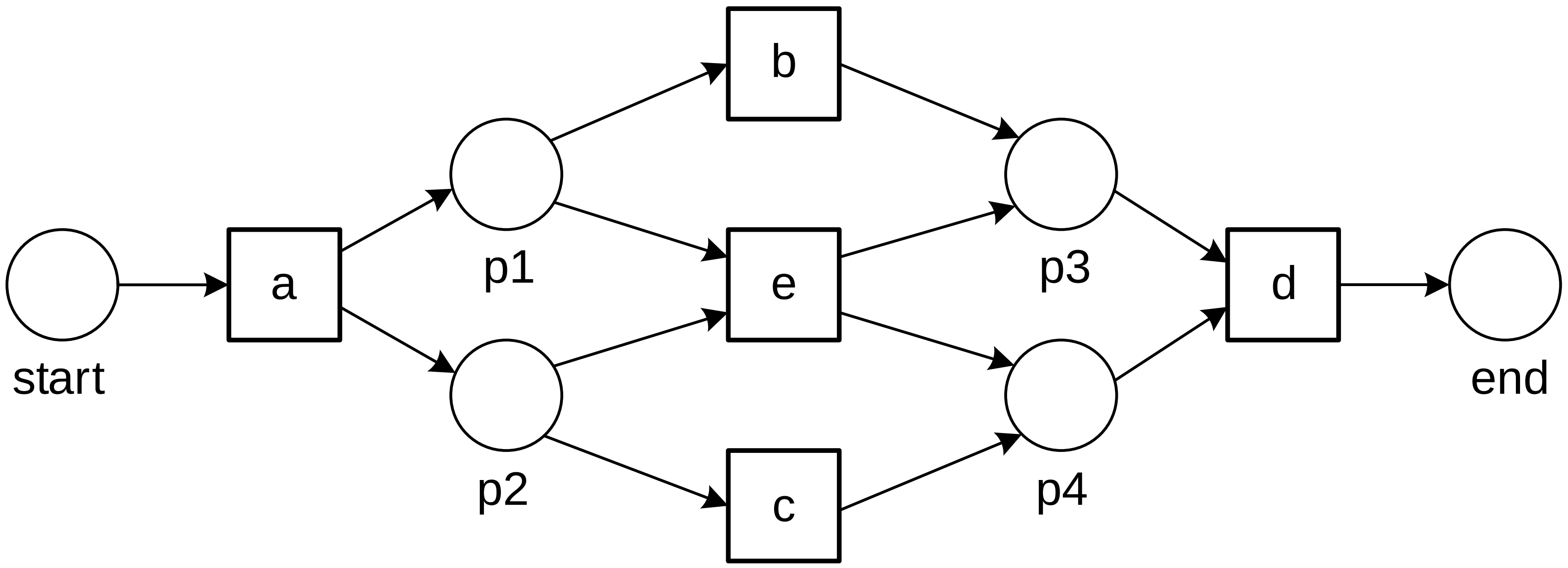
multiset of traces in tabular format



- Consider the event log containing 35 cases
- What is the fitness of this process model?

Computing Fitness – Example

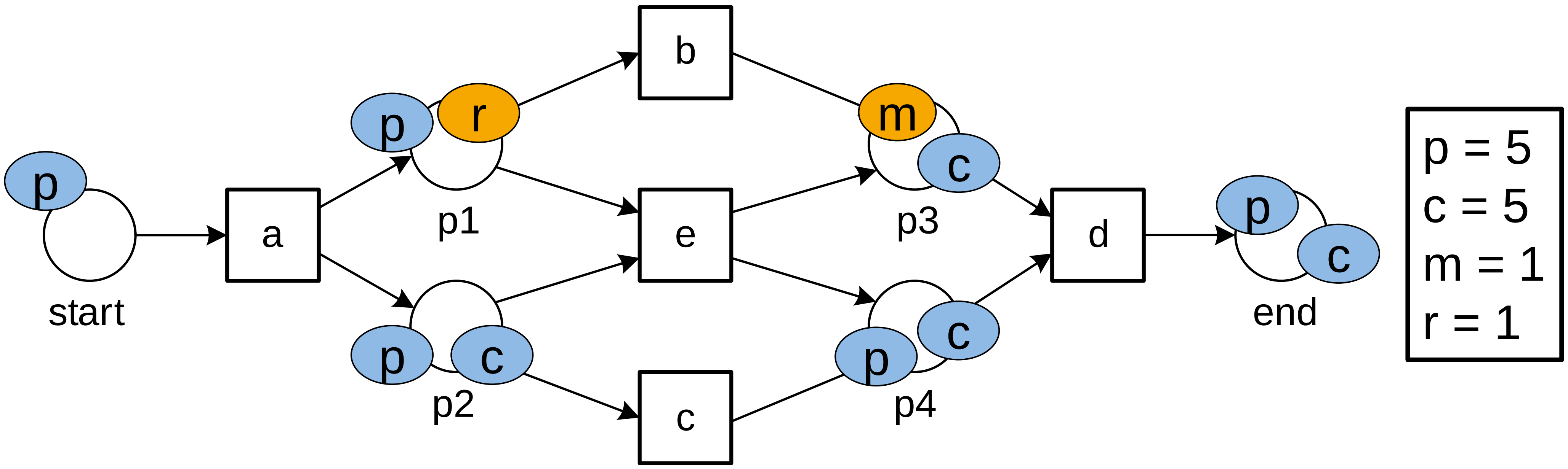
Consider Trace **acd** $(\sigma = \langle a, c, d \rangle)$



p = 0
c = 0
m = 0
r = 0

Computing Fitness – Example

Consider Trace **acd** $(\sigma = \langle a, c, d \rangle)$

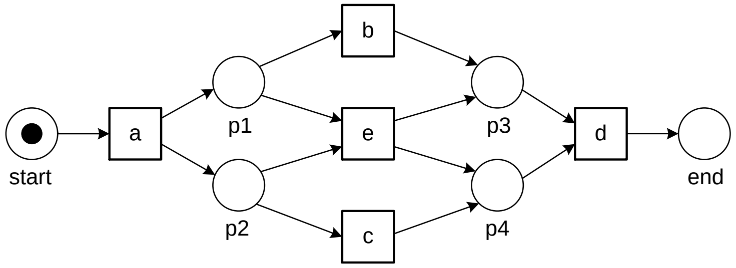


Computing Fitness – Example

Trace	Frequency	Produced (p)	Remaining (r)	Consumed (c)	Missing (m)	Produced (all)	Remaining (all)	Consumed (all)	Missing (all)
abcd	10	6	0	6	0	60	0	60	0
acbd	10	6	0	6	0	60	0	60	0
aed	10	6	0	6	0	60	0	60	0
abd	2	5	1	5	1	10	2	10	2
acd	1	**5**	**1**	**5**	**1**	5	1	5	1
ad	1	4	2	4	2	4	2	4	2
abbd	1	6	2	6	2	6	2	6	2

acd:

$p = 5$
$c = 5$
$m = 1$
$r = 1$

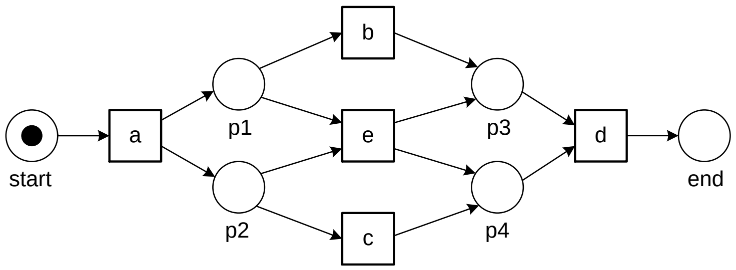


Computing Fitness – Example

Trace	Frequency	Produced (p)	Remaining (r)	Consumed (c)	Missing (m)	Produced (all)	Remaining (all)	Consumed (all)	Missing (all)
abcd	10	6	0	6	0	60	0	60	0
acbd	10	6	0	6	0	60	0	60	0
aed	10	6	0	6	0	60	0	60	0
abd	2	5	1	5	1	10	2	10	2
acd	**1**	5	1	5	1	**5**	**1**	**5**	**1**
ad	1	4	2	4	2	4	2	4	2
abbd	1	6	2	6	2	6	2	6	2

acd:

p = 5
c = 5
m = 1
r = 1



Computing Fitness – Example

Trace	Frequency	Produced (p)	Remaining (r)	Consumed (c)	Missing (m)	Produced (all)	Remaining (all)	Consumed (all)	Missing (all)
abcd	10	6	0	6	0	60	0	60	0
acbd	10	6	0	6	0	60	0	60	0
aed	10	6	0	6	0	60	0	60	0
abd	2	5	1	5	1	10	2	10	2
acd	1	5	1	5	1	5	1	5	1
ad	1	4	2	4	2	4	2	4	2
abbd	1	6	2	6	2	6	2	6	2
Sum						205	7	205	7

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

Computing Fitness – Example

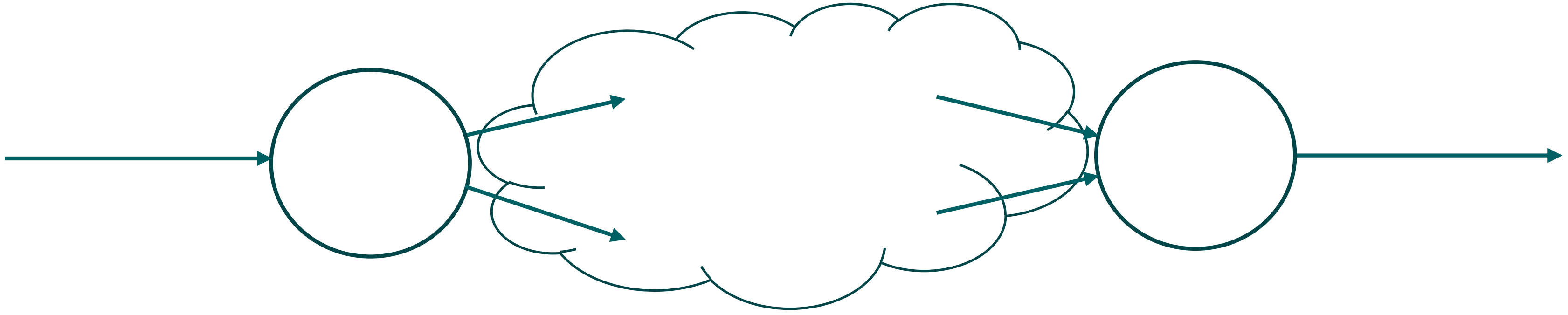
Trace	Frequency	Produced (p)	Remaining (r)	Consumed (c)	Missing (m)	Produced (all)	Remaining (all)	Consumed (all)	Missing (all)
abcd	10	6	0	6	0	60	0	60	0
acbd	10	6	0	6	0	60	0	60	0
aed	10	6	0	6	0	60	0	60	0
abd	2	5	1	5	1	10	2	10	2
acd	1	5	1	5	1	5	1	5	1
ad	1	4	2	4	2	4	2	4	2
abbd	1	6	2	6	2	6	2	6	2
Sum						205	7	205	7

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

$$= \frac{1}{2}(1 - \frac{7}{205}) + \frac{1}{2}(1 - \frac{7}{205}) \approx 0.966$$

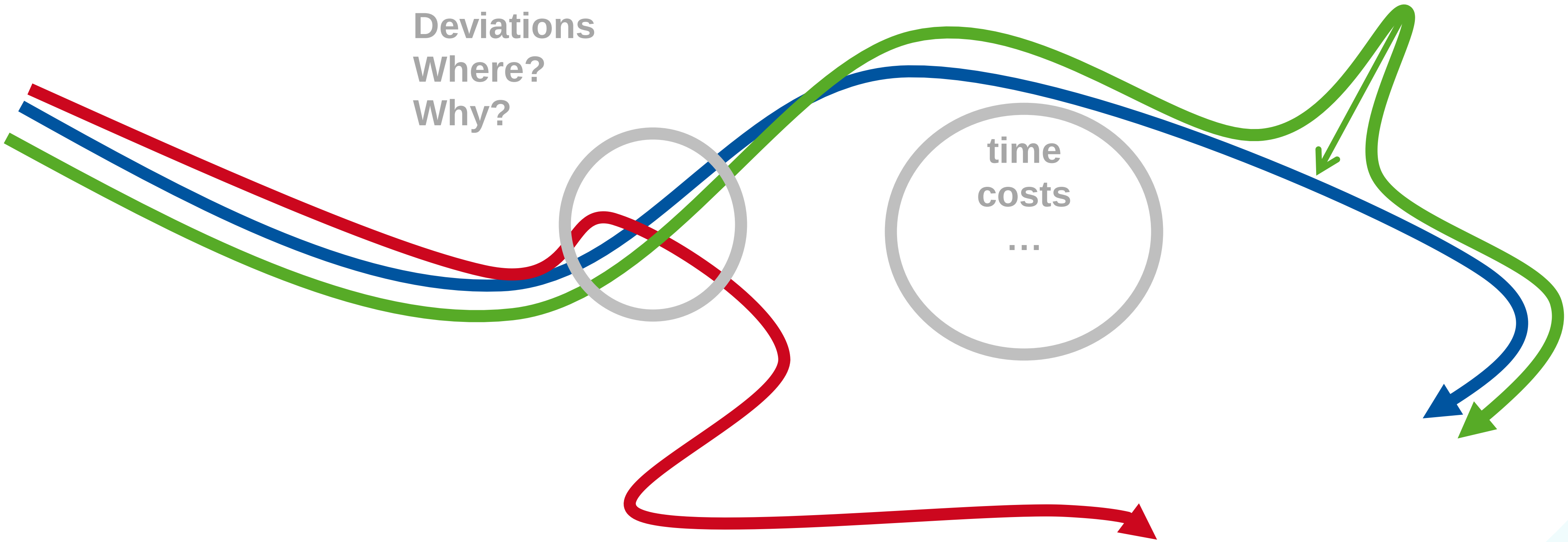
Limitations of Token-Based Approach

- Basic replay approach assumes visible & uniquely labeled transitions.
- Most implementations (ProM, PM4Py, Celonis, etc.) use heuristics to deal with silent transitions and multiple transitions having the same label
- Conformance values are sometimes too optimistic
- Local decision-making may lead to misleading results



Alignments

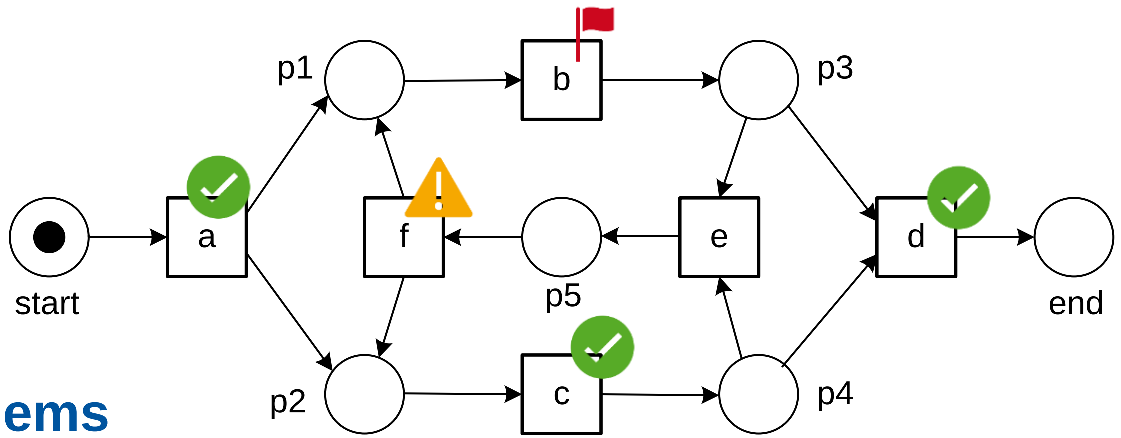
A Better, but More Expensive Way to Check Compliance



- Find the “closest path” in the model
- Outside of the scope of this course

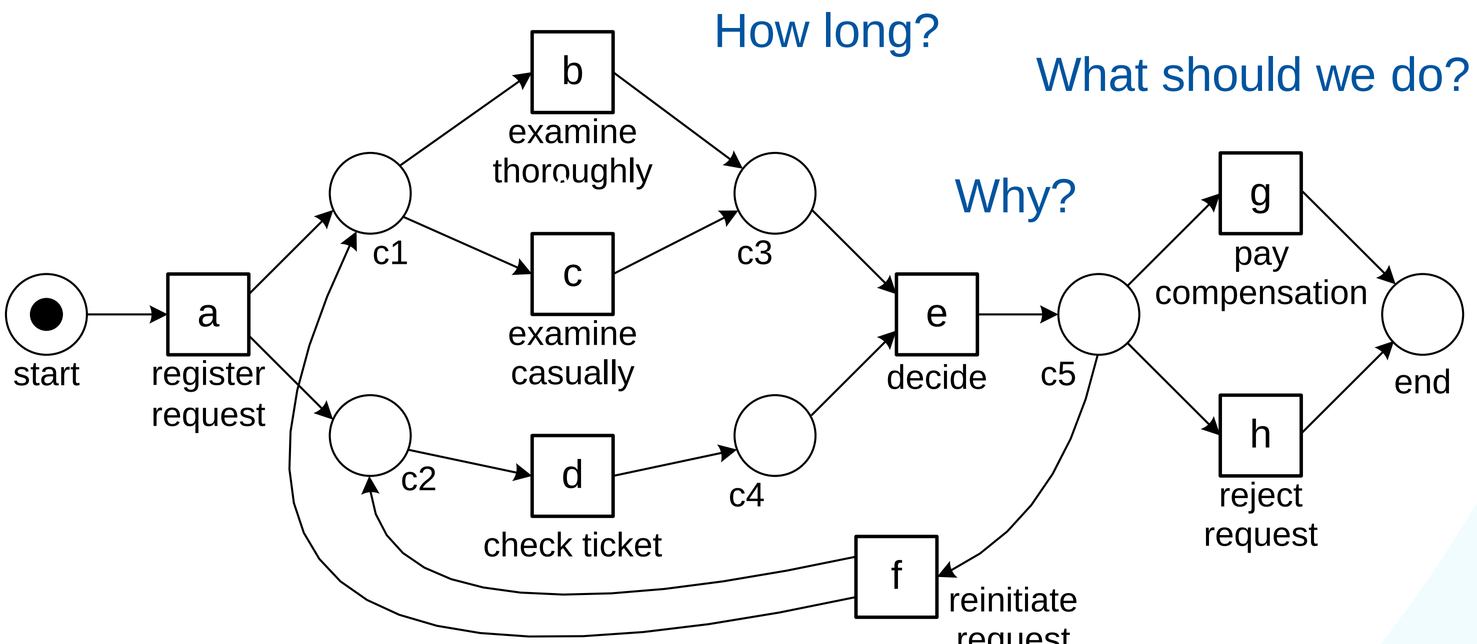
Supervised Process Mining

1. Token-Based Replay
2. Token-Based Replay Examples
3. Fitness at the Log Level
4. **Generating Supervised Learning Problems**

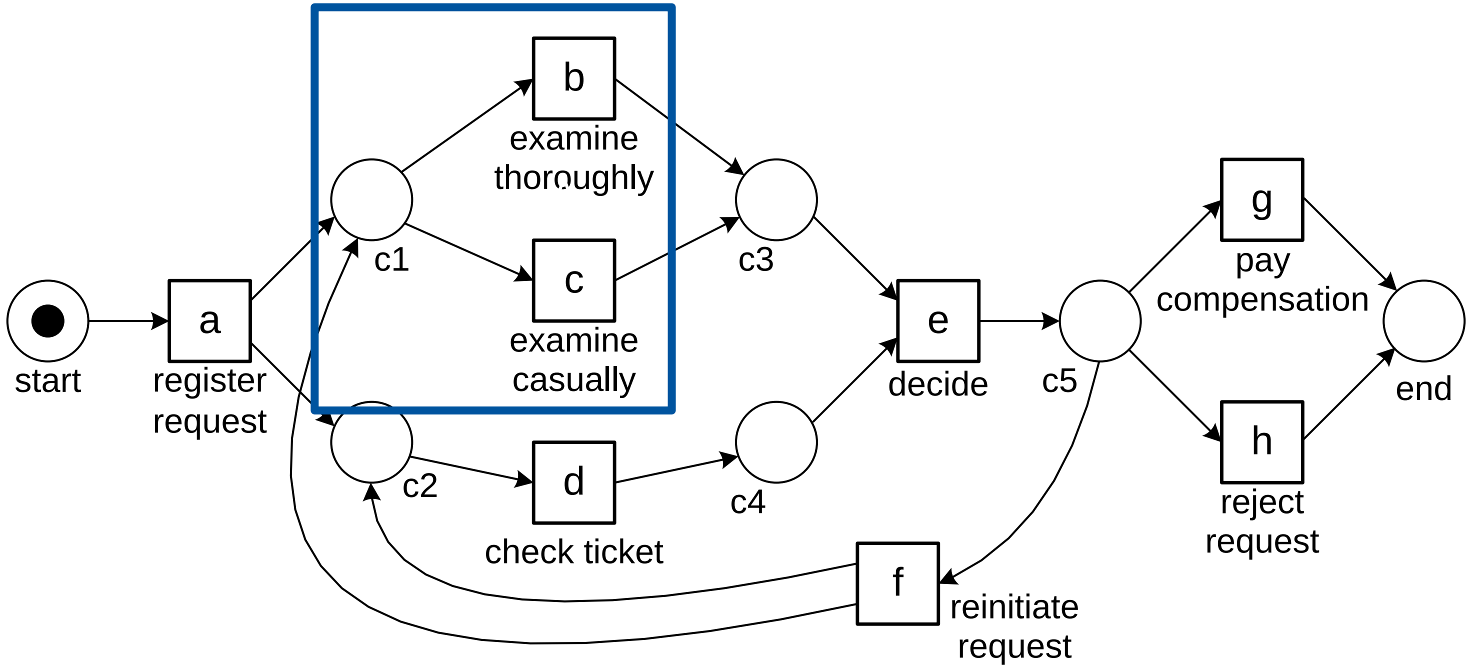


Connection to Machine Learning

- Machine learning and many other data science techniques are not process-centric
- Consider an information system with thousands of tables. How to get started?
- Process mining can generate valuable machine-learning problems



Decision Mining

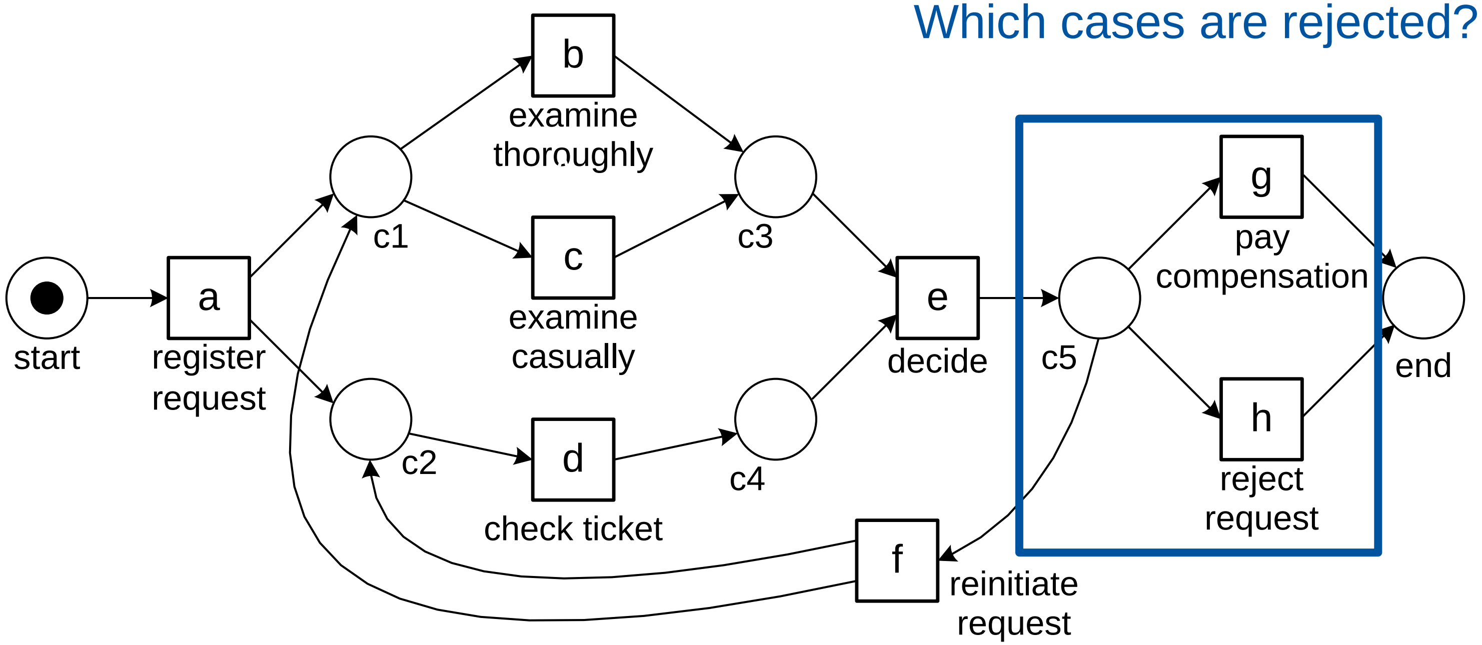


For example:

- Cases handled by John
- Cases handled in January
- Cases that were submitted late
- Cases of new customers
- …

Which cases require a thorough examination?

Decision Mining

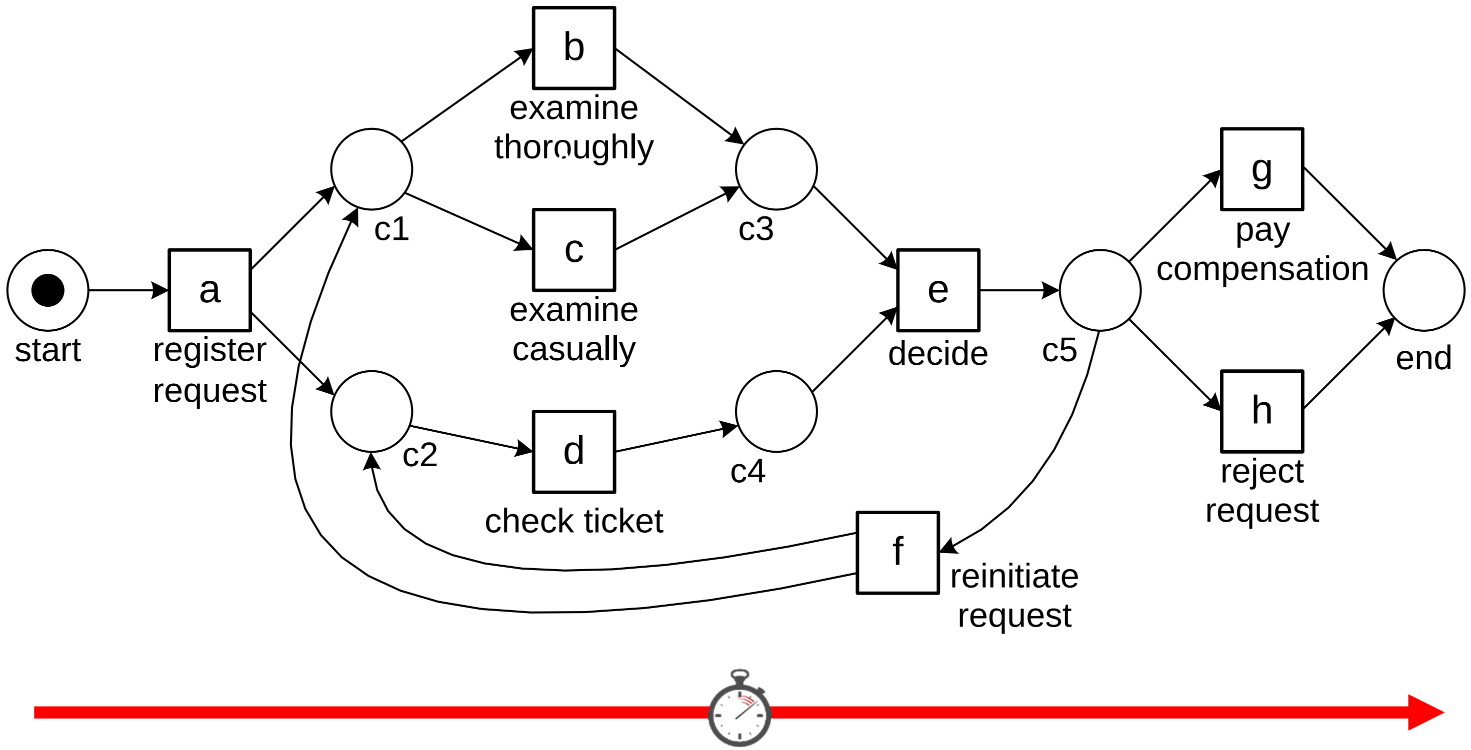


Which cases are rejected?

For example:

- Cases above €500
- Cases that required multiple checks
- Cases that got delayed
- …

Performance Mining

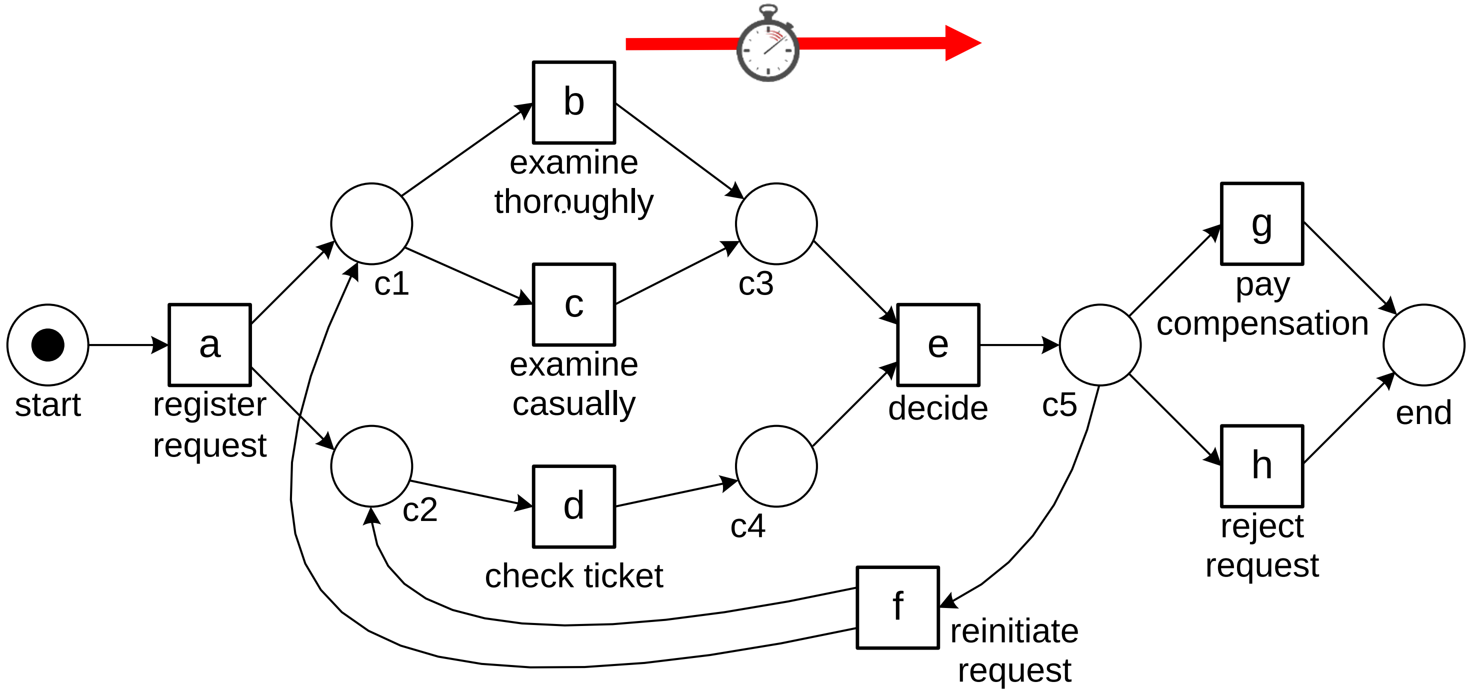


For example:

- Cases handled by Mary
- Cases that required multiple checks
- …

Which cases took more than two months?

Performance Mining

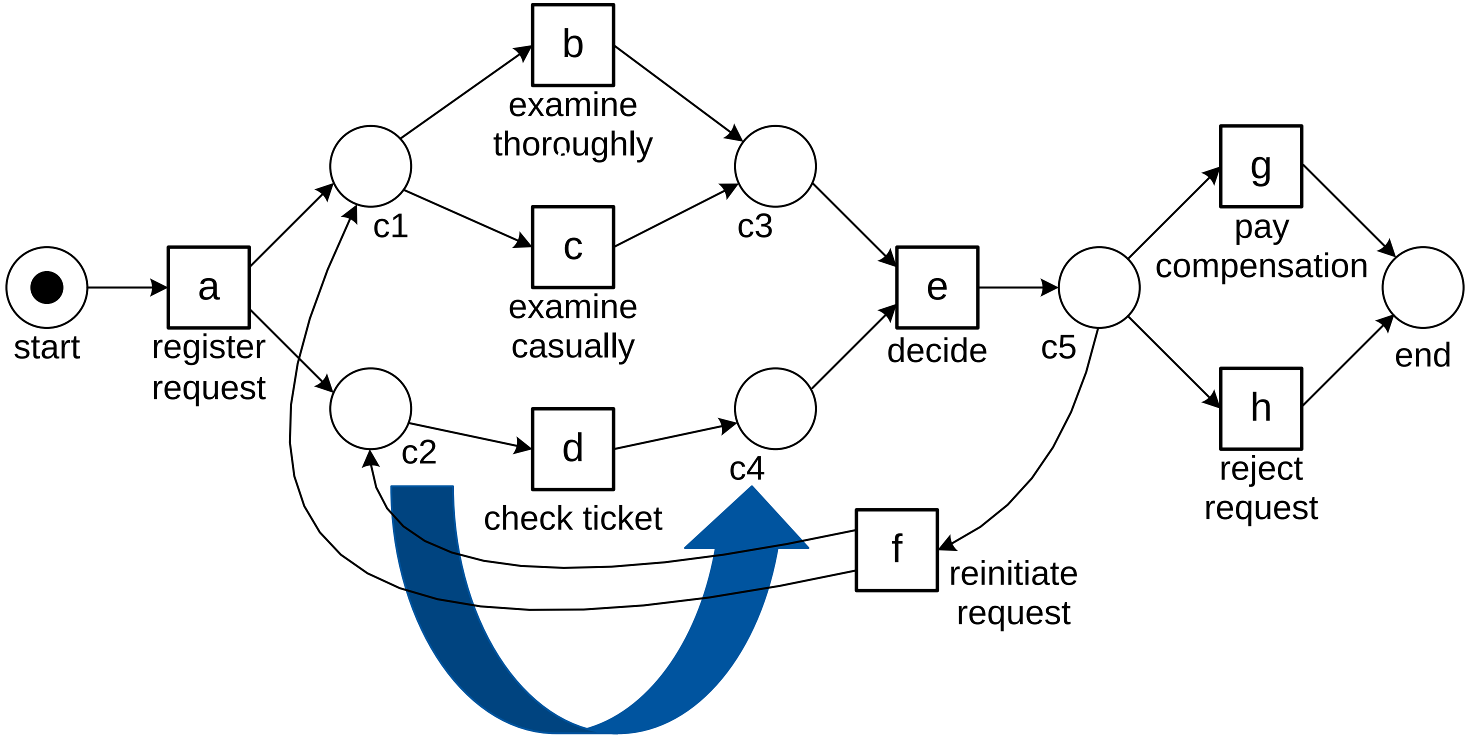


For example:

- A lack of resources due to illness
- An unusual percentage or rework
- …

What caused the delays in decision making in May?

Deviation Mining



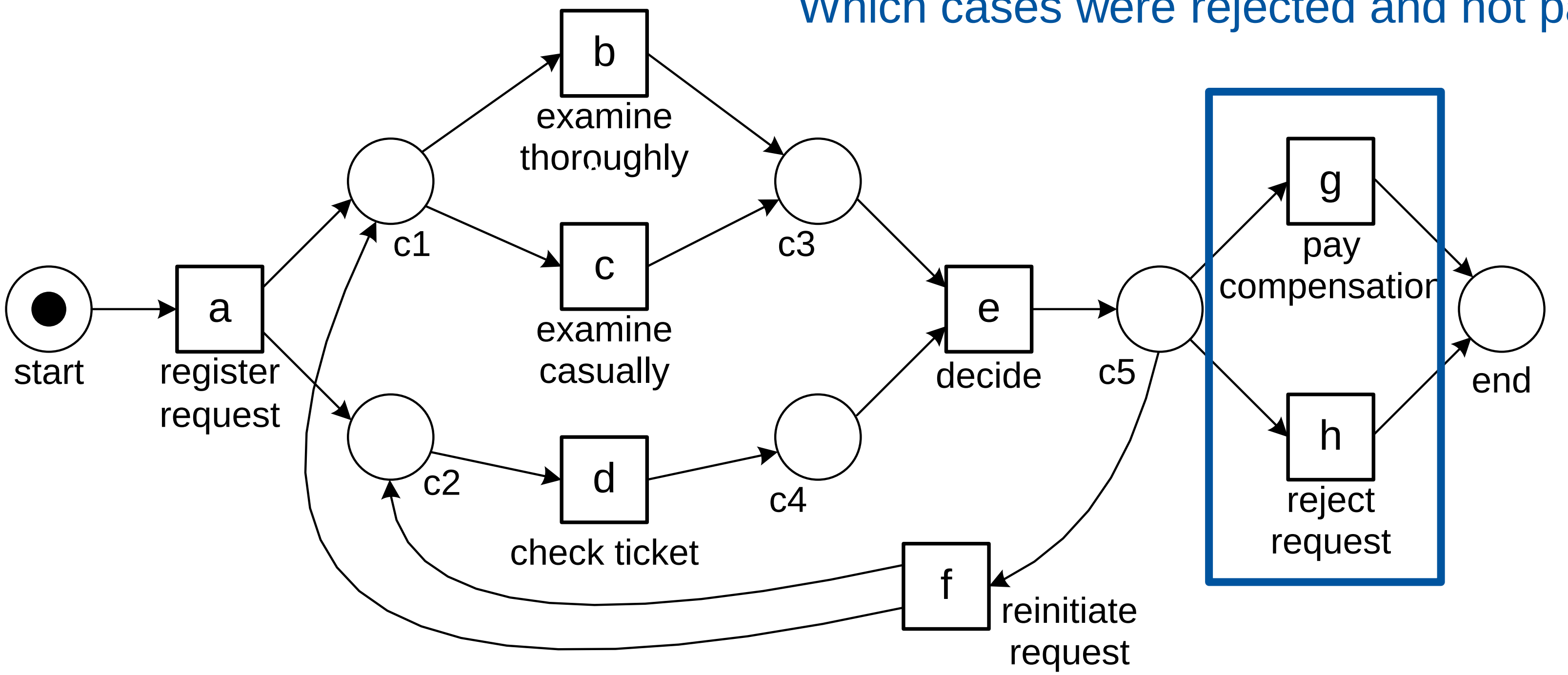
For example:

- Cases handled by Mary
- Cases initiated by the downtown office
- …

For which cases was the ticket not checked?

Deviation Mining

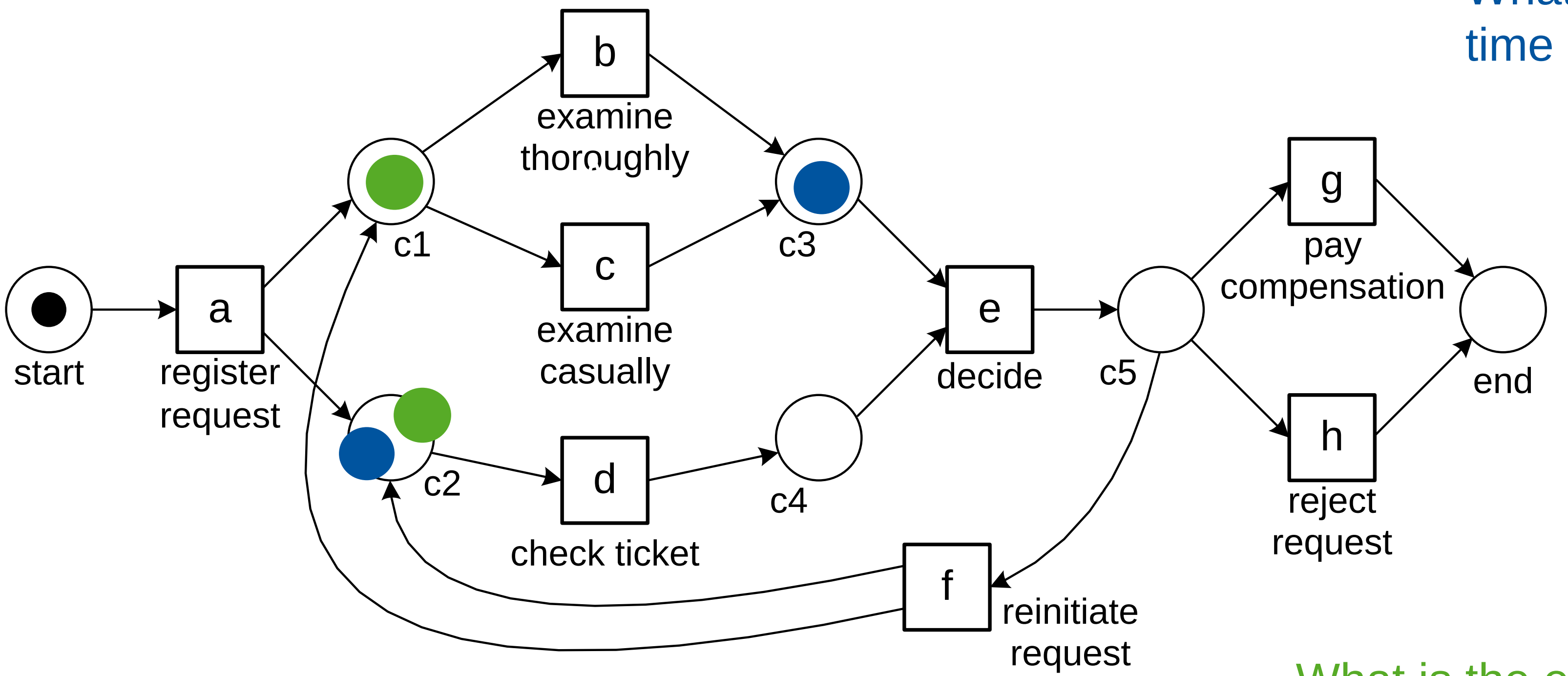
Which cases were rejected and not paid?



For example:

- Cases handled by Pete
- Cases that arrived in June
- …

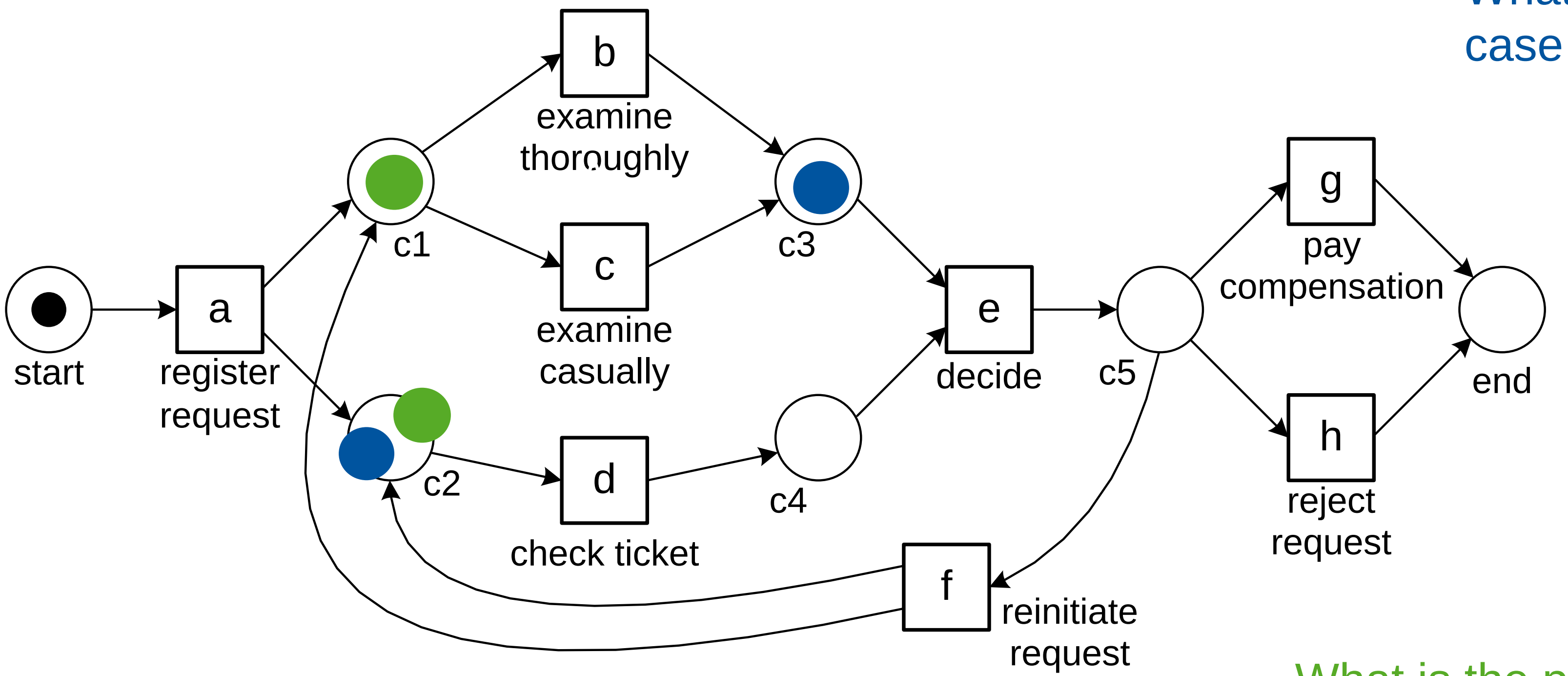
Operational Support Using Process Mining



What is the expected remaining flow time of the blue case?

What is the expected remaining flow time of the green case?

Operational Support Using Process Mining



What is the probability that the blue case will be rejected?

What is the probability that the green case will need two decisions?

General Pattern

process mining

standard ML problems in tabular format

Process Mining

- Event data are omnipresent (just like text data or image data).
- A very interdisciplinary field! Connections with traditional data science, process management, simulation, machine learning, …
- We focused on two tasks:
  - **Process Discovery**: obtain a process model from historic event data
  - **Conformance Checking**: obtain a measure of deviation between expected and actual behavior
- A relatively young field of research: Many foundational questions are still open, but already widely adopted in industry.

Learn More About Process Mining?

- Consider taking Business Process Intelligence (BPI) in the next semester.
- Many opportunities to go deeper, e.g., Advanced Process Mining (APM), seminars, etc.
- Also, we created several online courses on Coursera and edX.
- Visit https://www.pads.rwth-aachen.de/ for thesis projects, etc.