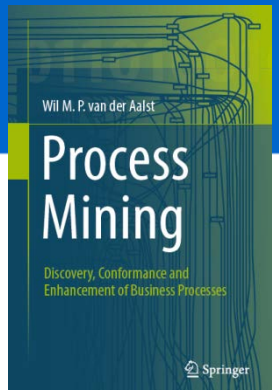
Process Mining: Data Science in Action

Learning Causal Nets and Annotating Them

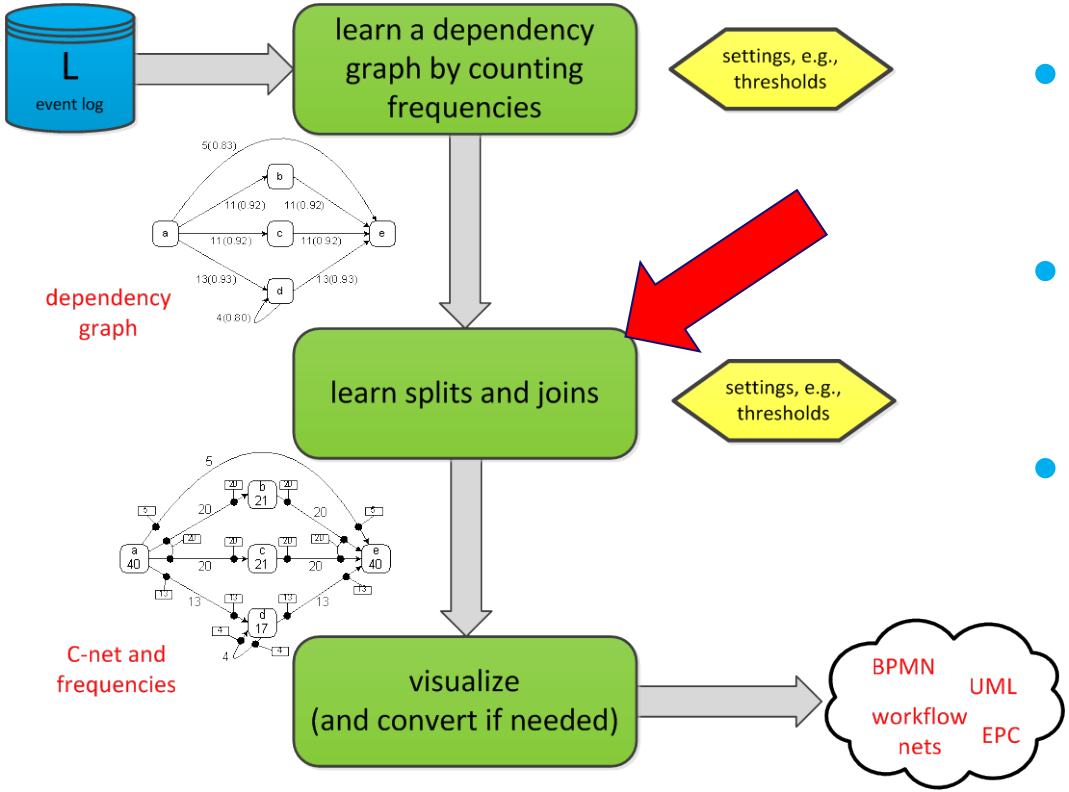


prof.dr.ir. Wil van der Aalst
www.processmining.org

TU/e Technische Universiteit Eindhoven University of Technology

Where innovation starts

Heuristic mining: Two main phases



- Here we focus on the second phase.
- Discovering splits and joins.
- Annotating C-nets with frequencies.

TU/e

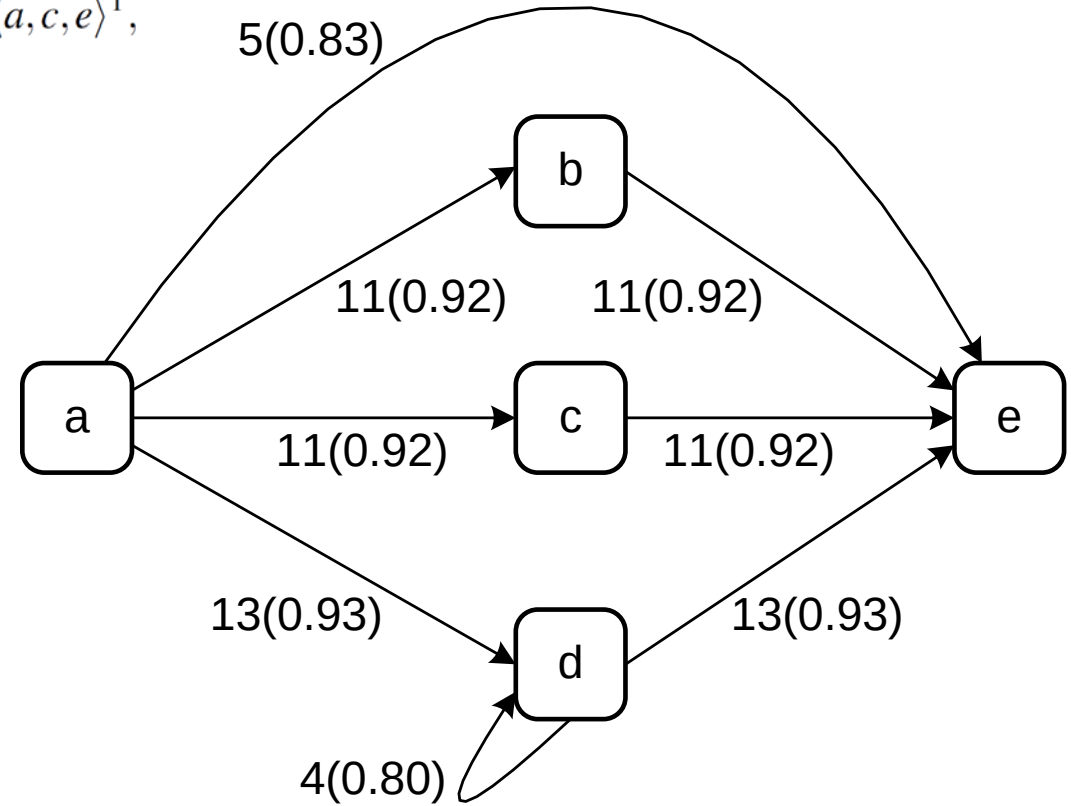
Reminder: First step

Create dependency graph

$$L = [\langle a,e\rangle^{5}, \langle a,b,c,e\rangle^{10}, \langle a,c,b,e\rangle^{10}, \langle a,b,e\rangle^{1}, \langle a,c,e\rangle^{1}, \langle a,d,e\rangle^{10}, \langle a,d,d,e\rangle^{2}, \langle a,d,d,d,e\rangle^{1}]$$

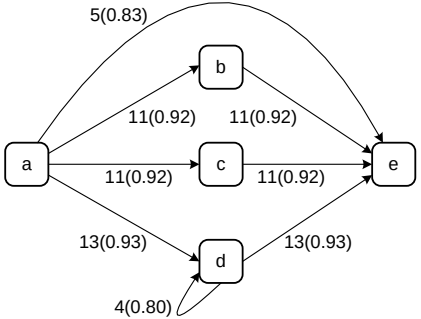
$\lvert >_L \rvert$	a	b	c	d	e
a	0	11	11	13	5
b	0	0	10	0	11
c	0	10	0	0	11
d	0	0	0	4	13
e	0	0	0	0	0

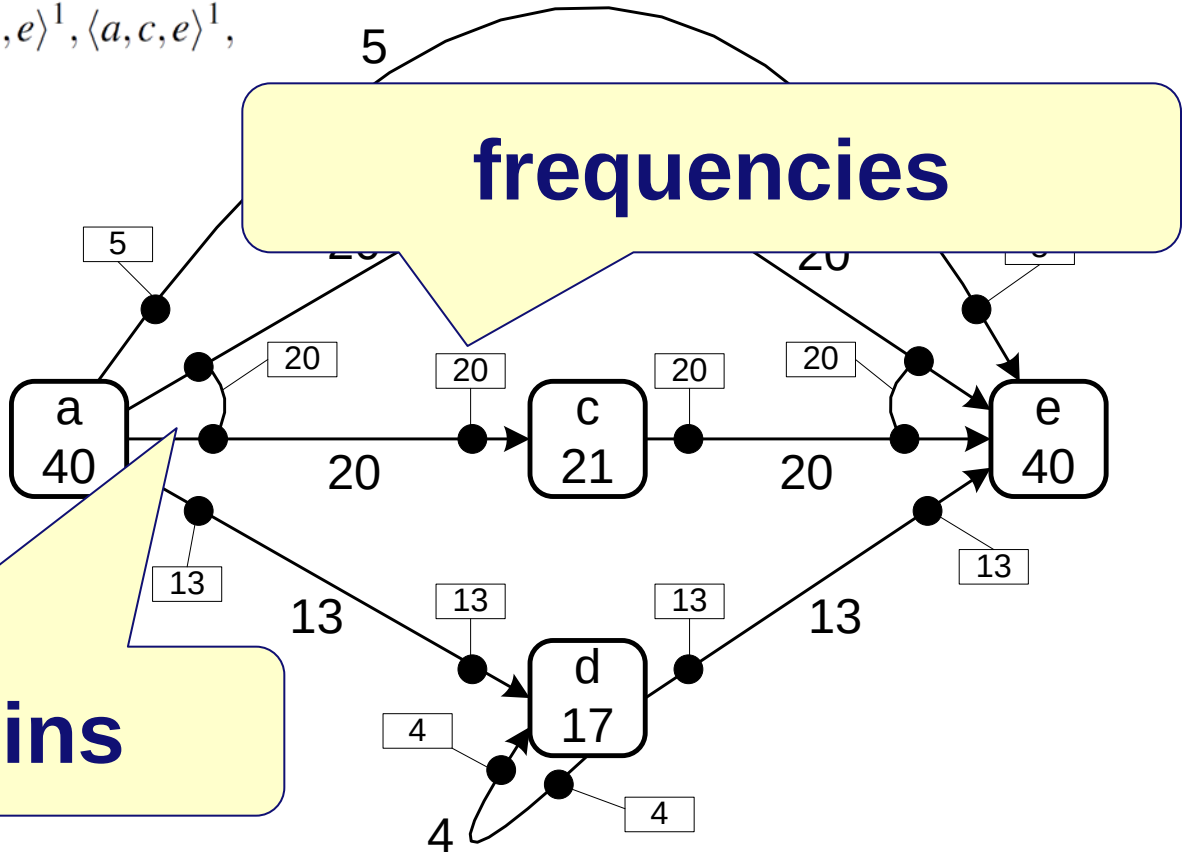
$\lvert \Rightarrow_L \rvert$	a	b	c	d	e
a	$\frac{0}{0+1}=0$	$\frac{11-0}{11+0+1}=0.92$	$\frac{11-0}{11+0+1}=0.92$	$\frac{13-0}{13+0+1}=0.93$	$\frac{5-0}{5+0+1}=0.83$
b	$\frac{0-11}{0+11+1}=-0.92$	$\frac{0}{0+1}=0$	$\frac{10-10}{10+10+1}=0$	$\frac{0-0}{0+0+1}=0$	$\frac{11-0}{11+0+1}=0.92$
c	$\frac{0-11}{0+11+1}=-0.92$	$\frac{10-10}{10+10+1}=0$	$\frac{0}{0+1}=0$	$\frac{0-0}{0+0+1}=0$	$\frac{11-0}{11+0+1}=0.92$
d	$\frac{0-13}{0+13+1}=-0.93$	$\frac{0-0}{0+0+1}=0$	$\frac{0-0}{0+0+1}=0$	$\frac{4}{4+1}=0.80$	$\frac{13-0}{13+0+1}=0.93$
e	$\frac{0-5}{0+5+1}=-0.83$	$\frac{0-11}{0+11+1}=-0.92$	$\frac{0-11}{0+11+1}=-0.92$	$\frac{0-13}{0+13+1}=-0.93$	$\frac{0}{0+1}=0$



Desired output: splits/joins & frequencies

$$L = [\langle a,e\rangle^{5}, \langle a,b,c,e\rangle^{10}, \langle a,c,b,e\rangle^{10}, \langle a,b,e\rangle^{1}, \langle a,c,e\rangle^{1}, \langle a,d,e\rangle^{10}, \langle a,d,d,e\rangle^{2}, \langle a,d,d,d,e\rangle^{1}]$$



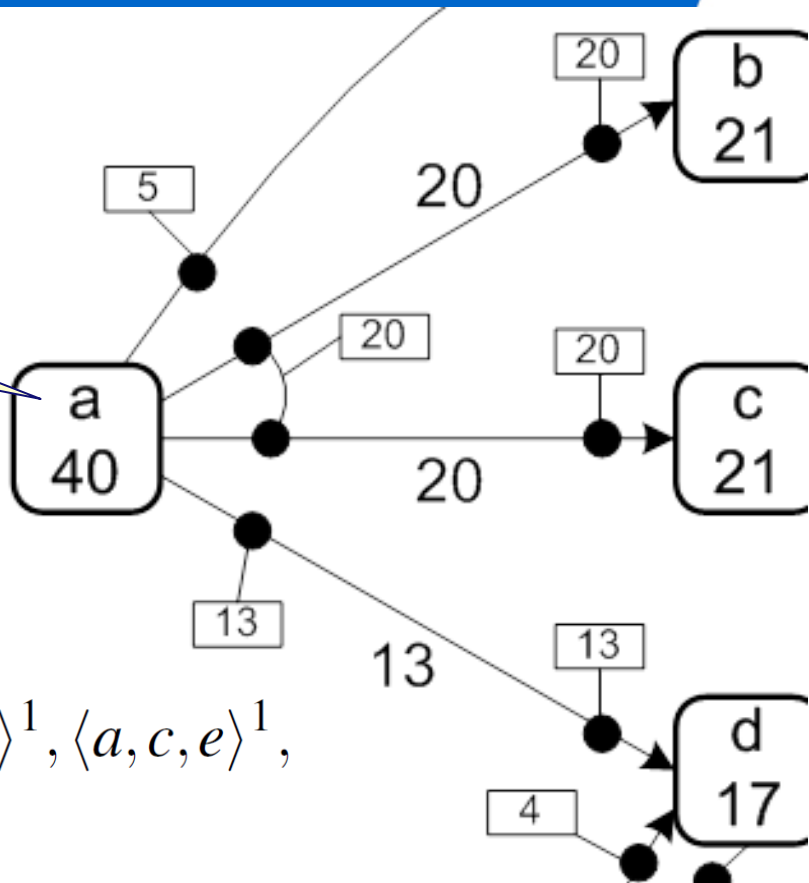


frequencies

splits and joins

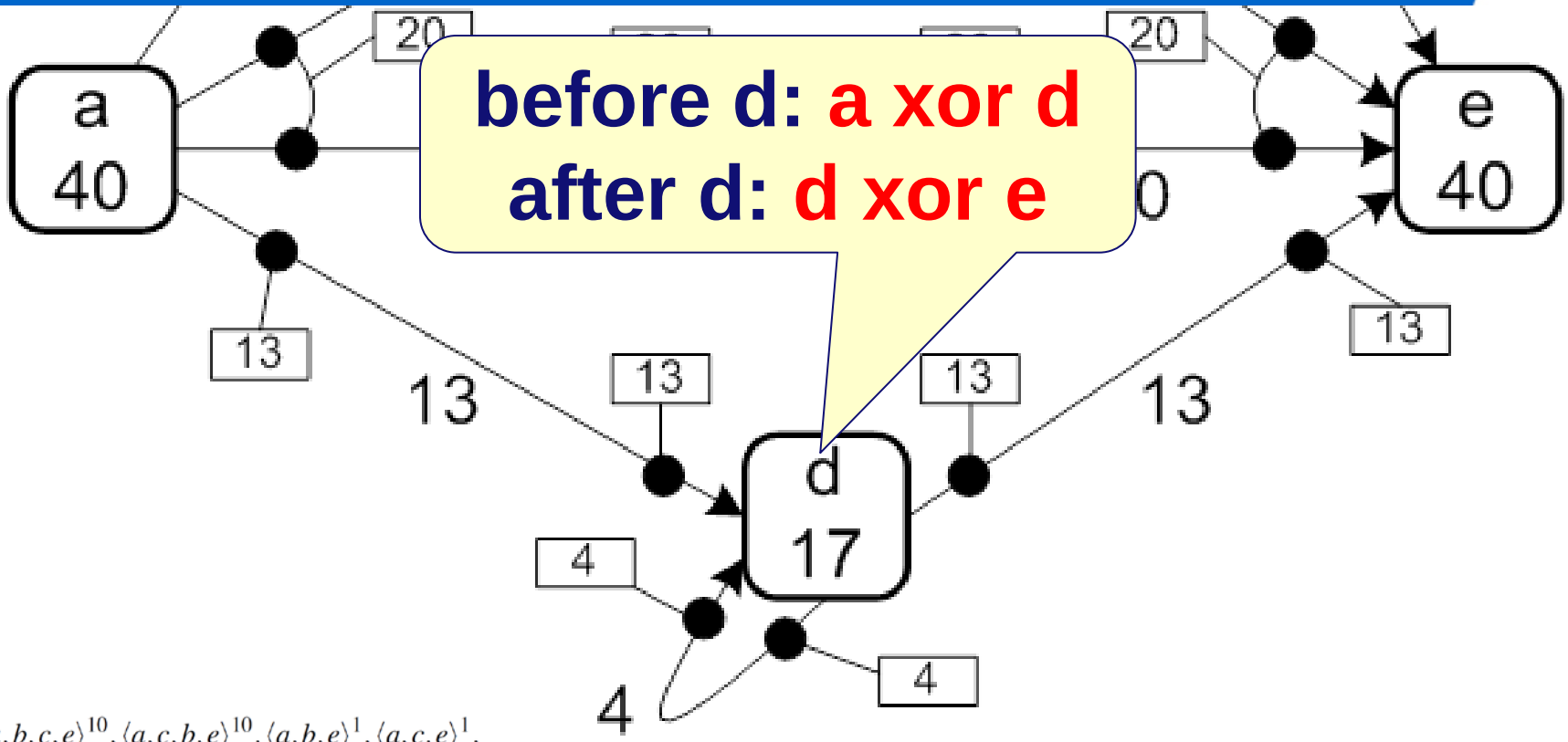
Output bindings of activity a

after a:
e xor (b and c) xor d



$$L = [\langle a,e\rangle^{5}, \langle a,b,c,e\rangle^{10}, \langle a,c,b,e\rangle^{10}, \langle a,b,e\rangle^{1}, \langle a,c,e\rangle^{1},$$
$$\langle a,d,e\rangle^{10}, \langle a,d,d,e\rangle^{2}, \langle a,d,d,d,e\rangle^{1}]$$

Input and output bindings of activity d

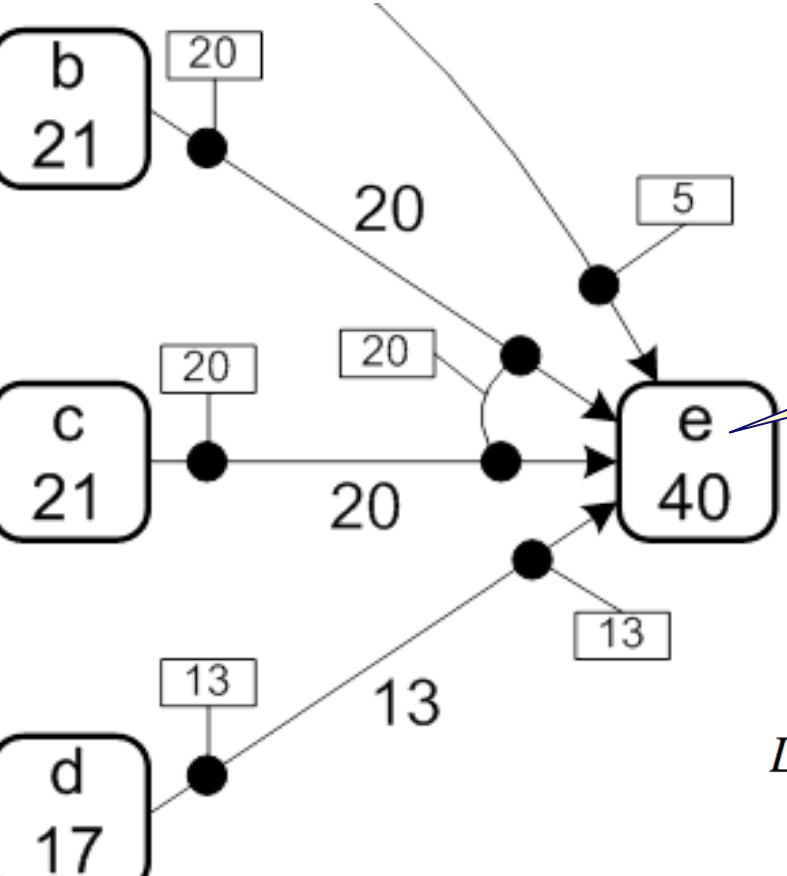


before d: a xor d

after d: d xor e

$$L = [\langle a,e\rangle^{5}, \langle a,b,c,e\rangle^{10}, \langle a,c,b,e\rangle^{10}, \langle a,b,e\rangle^{1}, \langle a,c,e\rangle^{1},$$
$$\langle a,d,e\rangle^{10}, \langle a,d,d,e\rangle^{2}, \langle a,d,d,d,e\rangle^{1}]$$

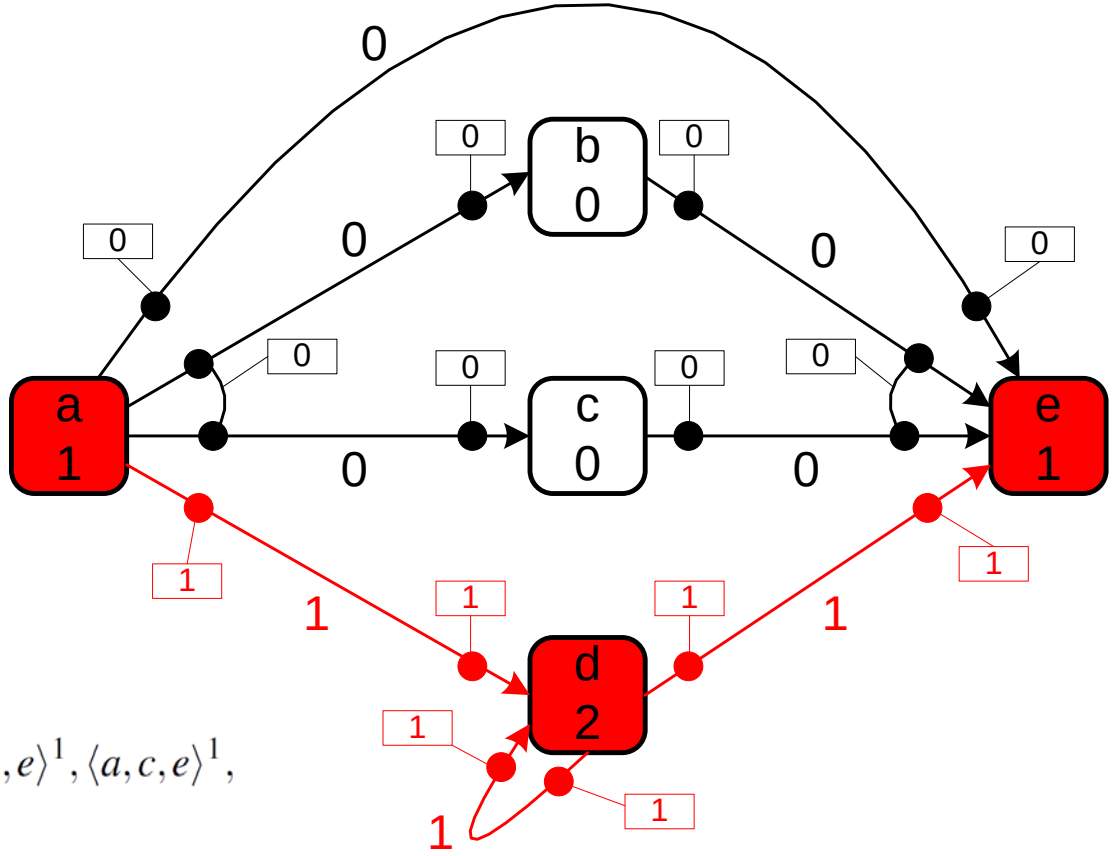
Input bindings of e



before e:
a xor (b and c) xor d

$$L = [\langle a,e\rangle^{5}, \langle a,b,c,e\rangle^{10}, \langle a,c,b,e\rangle^{10}, \langle a,b,e\rangle^{1}, \langle a,c,e\rangle^{1},$$
$$\langle a,d,e\rangle^{10}, \langle a,d,d,e\rangle^{2}, \langle a,d,d,d,e\rangle^{1}]$$

Example path: adde

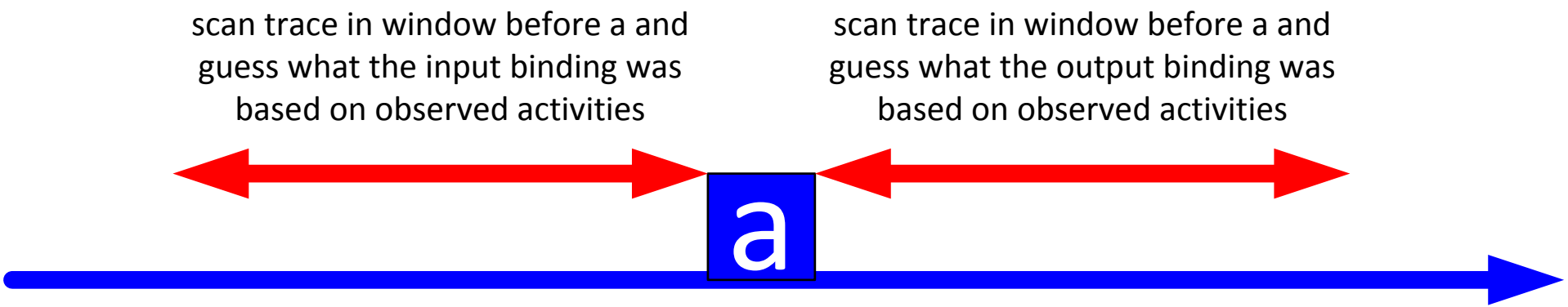


$$L = [\langle a,e\rangle^{5}, \langle a,b,c,e\rangle^{10}, \langle a,c,b,e\rangle^{10}, \langle a,b,e\rangle^{1}, \langle a,c,e\rangle^{1},$$
$$\langle a,d,e\rangle^{10}, \langle a,d,d,e\rangle^{2}, \langle a,d,d,d,e\rangle^{1}]$$

How to discover splits and joins?

- **Two classes of approaches:**
  1. **Heuristics using a time window before and after each activity. By counting sets of input and output activities the bindings can be determined (local decision).**
  2. **Optimization approaches based on replay. Given a set of possible input and output bindings one can see whether reality can be replayed properly. The set of possible input and output bindings is finite, so a "best set of bindings" can be determined using some goal function.**
- **Many variations are possible!**

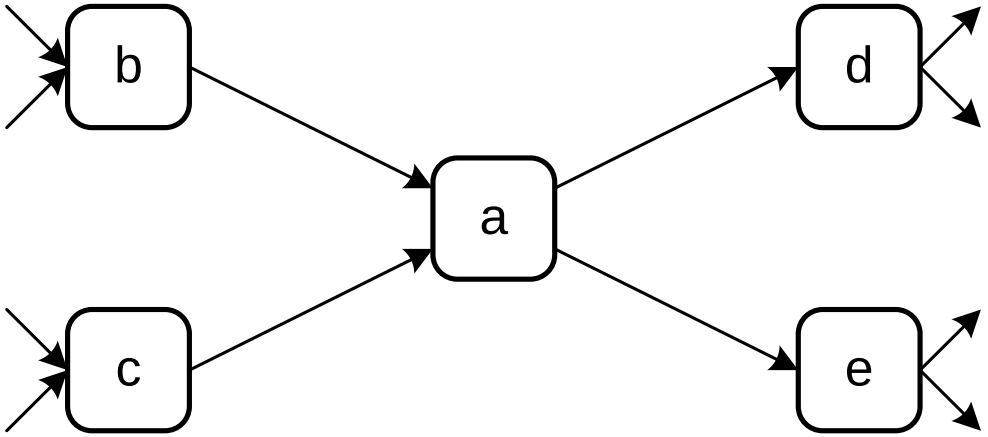
Approach 1: Based on heuristics



- **Activities have possible inputs and outputs (based on dependency graph).**
- **Count how often they appear in a window before (for input bindings) and a window after (for output bindings)**

Example: Window size 4

1….klbgadhek…
2….lkgcahedl…
3….kblgaehdk…
4….klgbadehk…
5….klkcadkeh…



input output bindings

- {b}: • {d,e}: 5 times
- {c}: 2 times

Adding bindings and frequencies

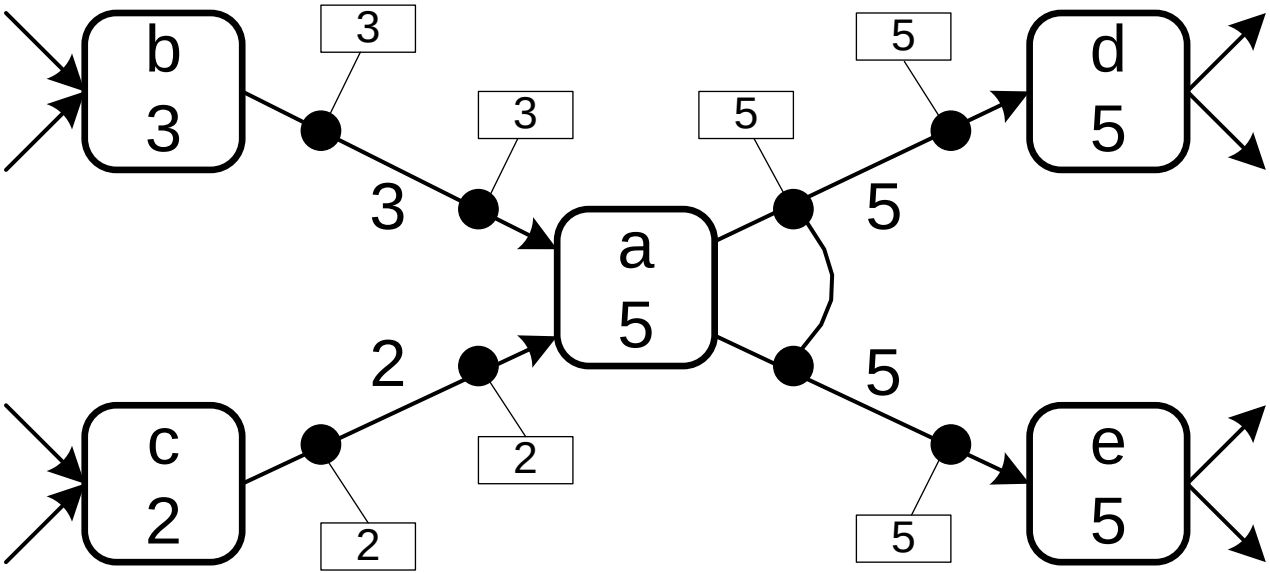
1. …klbgadhek…
2. …lkgcahedl…
3. …kblgaehdk…
4. …klgbadehk…
5. …klkcadkeh…

input bindings

- {a}: 3 times
- {c}: 2 times

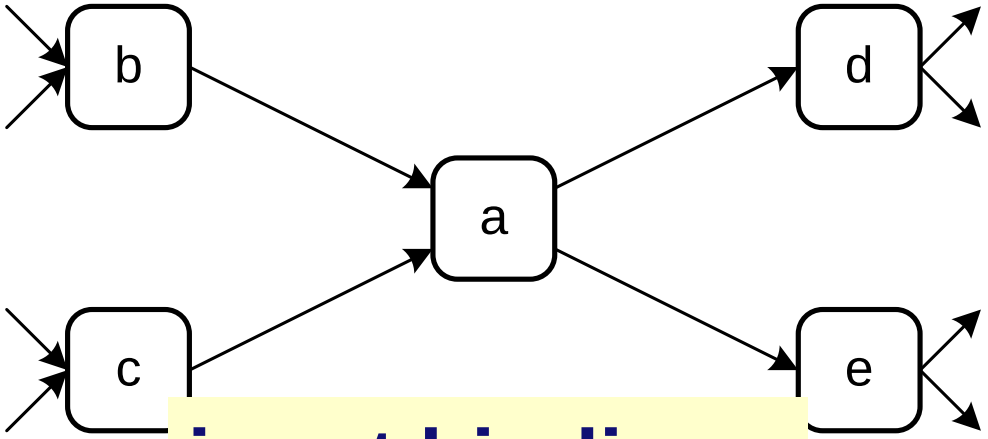
output bindings

- {d,e}: 5 times



Another example: Window size 4

1….klbgadhek…
2….lkgcahhdl…
3….kbcgaehdk…
4….klcbadkhk…
5….klkcadkeh…



input bindings
- {l
- {(

output bindings
- {d}: 2 times
- {d,e}: 3 times

Adding bindings and frequencies

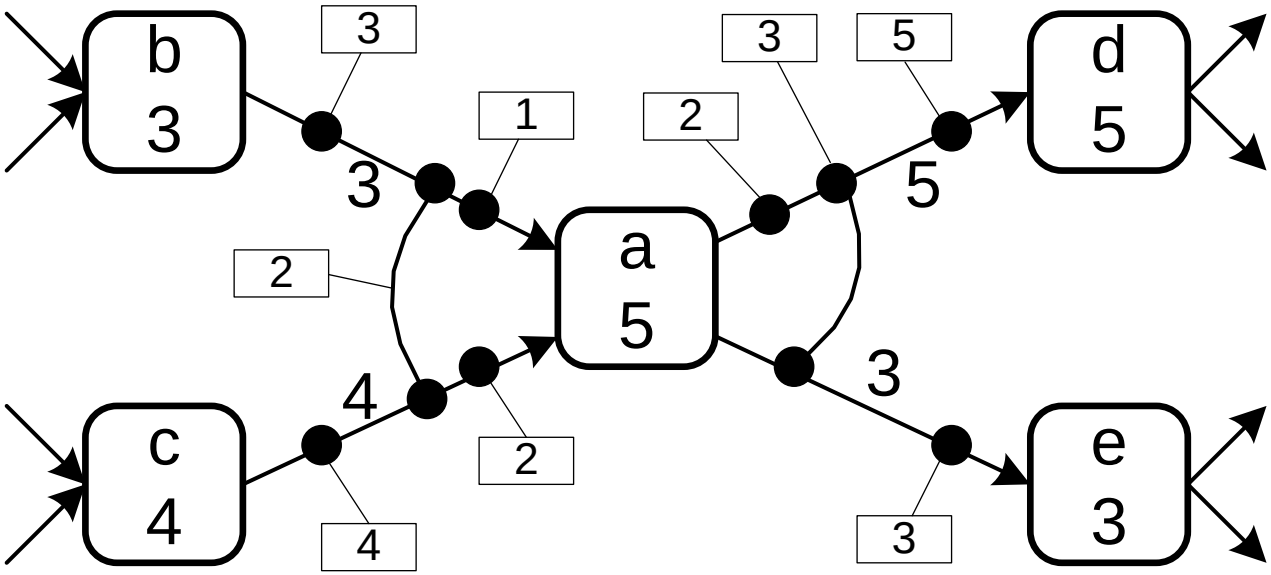
1. …klbgadhek…
2. …lkgcahhdl…
3. …kbcgaehdk…
4. …klcbadkhk…
5. …klkcadkeh…

input bindings

- {b}: 1 time
- {c}: 2 times
- {b,c}: 2 times

output bindings

- {d}: 2 times
- {d,e}: 3 times



Question

Determine input and output bindings (window size 4)

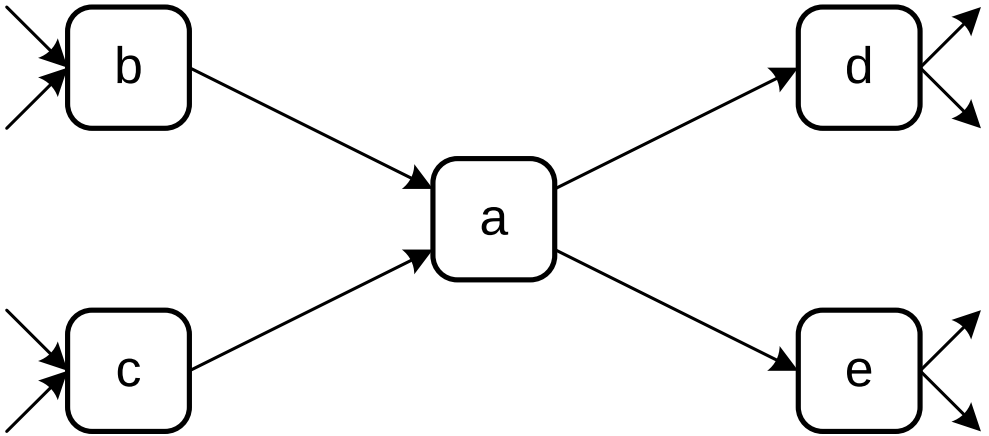
1….kcbgadhek…

2….lbgcaehdl…

3….bkcgaehhk…

4….cklbakkhe…

5….kbkcadkeh…



Answer

Input and output bindings (window size 4)

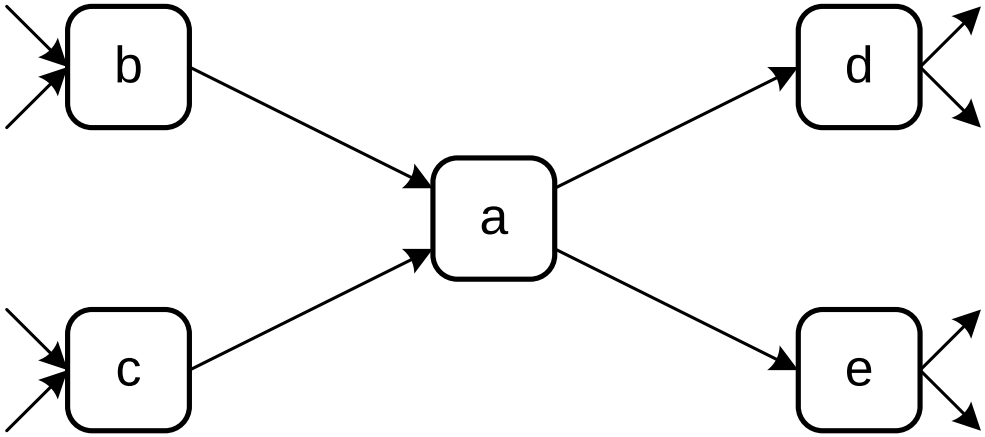
1….kcbgadhek…

2….lbgcaehdl…

3….bkcgaehhk…

4….cklbakkhe…

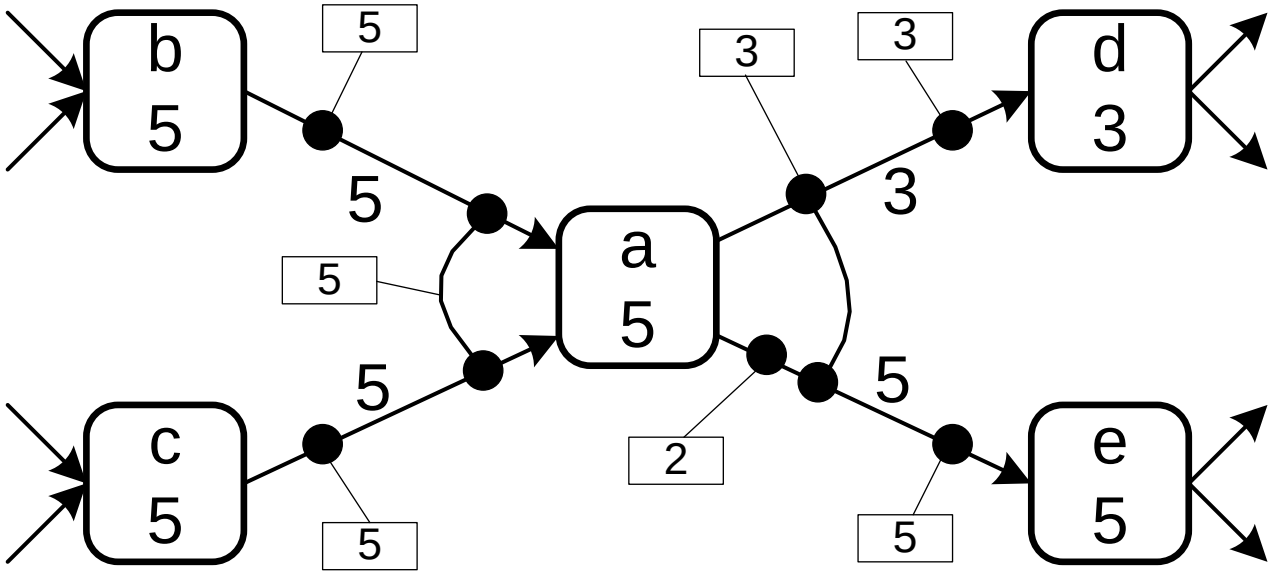
5….kbkcadkeh…



input	output bindings
• {b,c}	• {e}: 2 times
	• {d,e}: 3 times

Adding bindings and frequencies

1. …kcbgadhek…
2. …lbgcaehdl…
3. …bkcgaehhk…
4. …cklbakkhe…
5. …kbkcadkeh…



input bindings

- {b,c}: 5 times

output bindings

- {e}: 2 times
- {d,e}: 3 times

Refinements needed!

- What if there are no corresponding activities in the input or output window?
- Noise filtering (remove infrequent bindings).
- Handling repeating activities (e.g., cut off window size).
- Details are out of scope, but be aware of such complications when interpreting results!

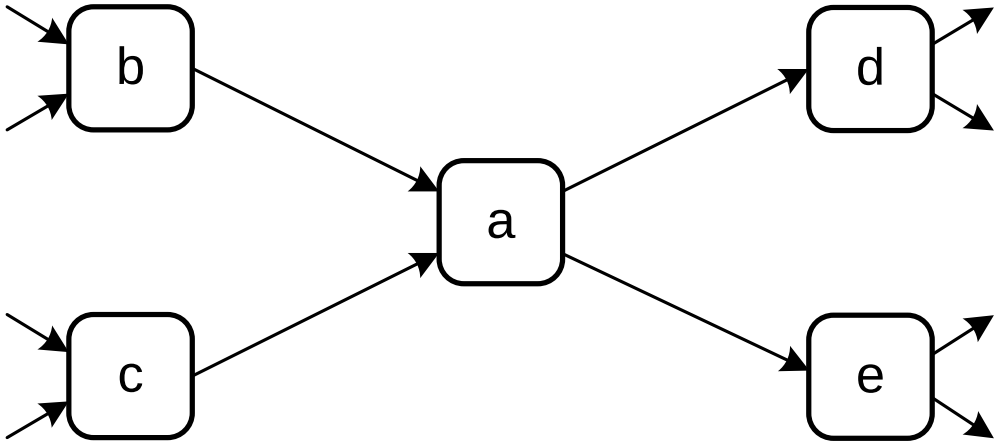
Approach 2: Optimization problem

- **Evaluate all possible activity bindings and take best one.**
- **Based on the idea that ideally a trace can be replayed from the initial state to the final state.**
- **This can be checked precisely using various replay approaches (will be discussed later).**
- **Hence, one can use approaches that simply "try bindings" exhaustively.**

TU/e

Example: Sets of input and output bindings

- **Each input/output arc needs to be involved in at least one binding.**



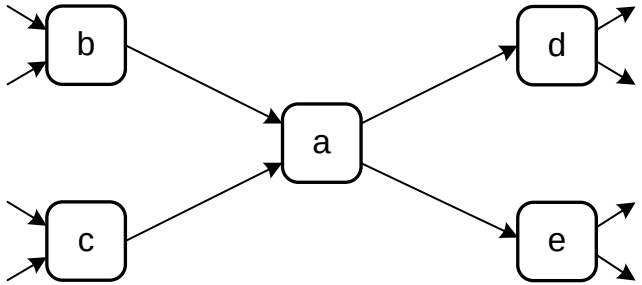
There are

|{ {{b},{c}} , {{b},{b,c}} , {{c},{b,c}} , {{b},{c},{b,c}} }| x
|{ {{d},{e}} , {{d},{d,e}} , {{e},{d,e}} , {{d},{e},{d,e}} }|
= 4 x 4 = 16 possible a activities.

TU/e

Optimization approach

- For each activity select one of the input-output binding combinations.
- One can do this exhaustively and try all combinations.
- Evaluation can be done using replay.
- Take best one (taking into account fitness, precision, generalization, and simplicity).



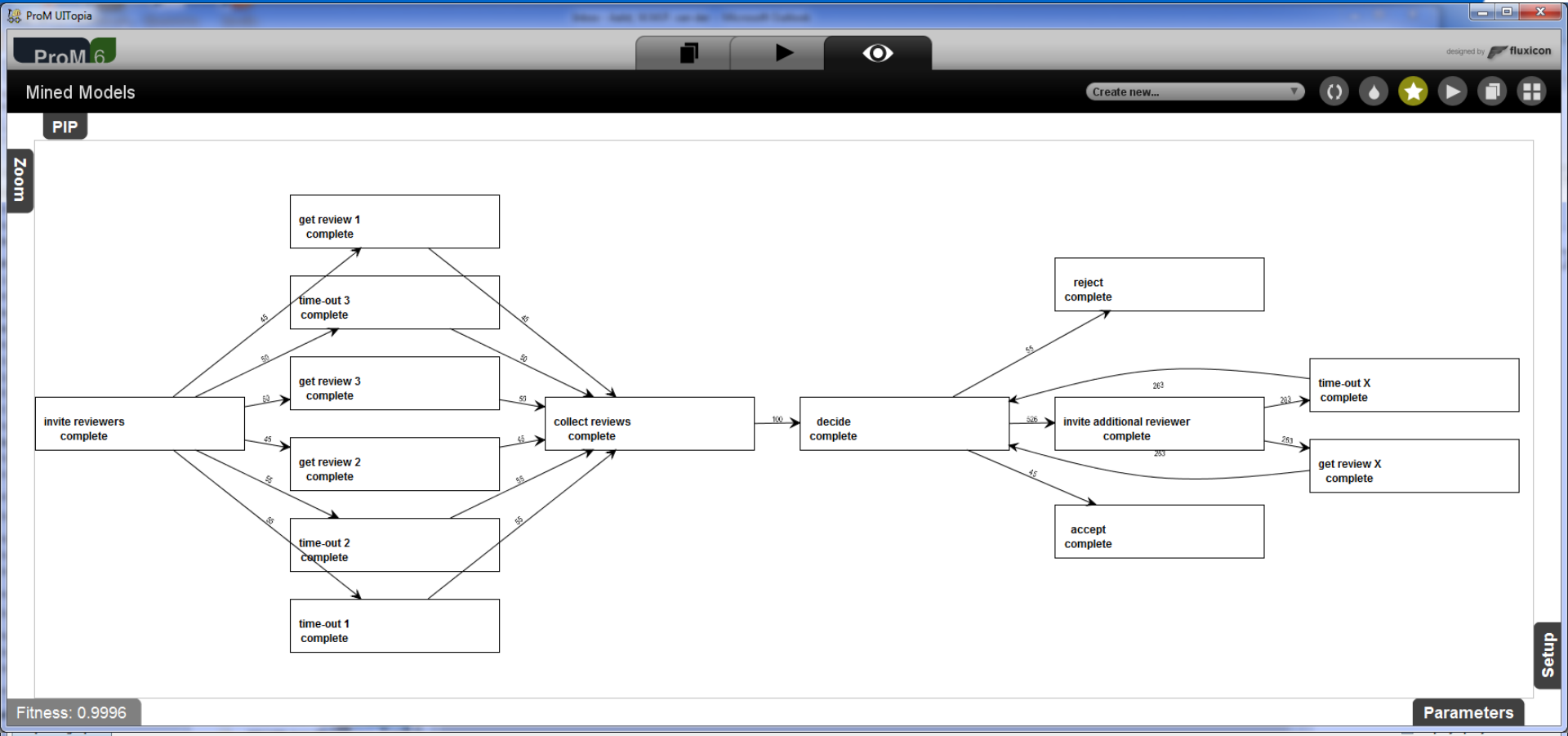
|{ {{b},{c}} , {{b},{b,c}} ,
{{c},{b,c}} , {{b},{c},{b,c}} }| x
|{ {{d},{e}} , {{d},{d,e}} ,
{{e},{d,e}} , {{d},{e},{d,e}} }|
= 4 x 4 = 16 possible a activities

TU/e

If too time consuming, …

- **Randomize**
- **Use a genetic algorithm**

Example: Heuristic miner



Part I: Preliminaries

- Chapter 1: Introduction
- Chapter 2: Process Modeling and Analysis
- Chapter 3: Data Mining

Part II: From Event Logs to Process Models

- Chapter 4: Getting the Data
- Chapter 5: Process Discovery: An Introduction
- Chapter 6: Advanced Process Discovery Techniques

Part III: Beyond Process Discovery

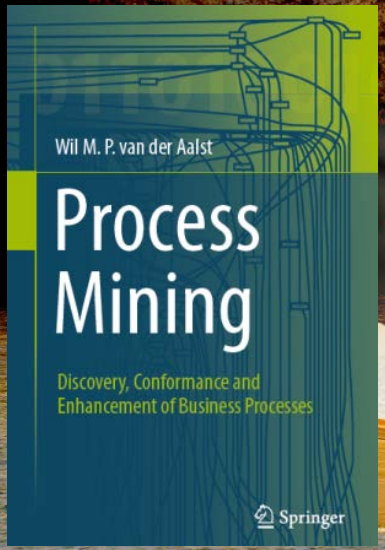
- Chapter 7: Conformance Checking
- Chapter 8: Mining Additional Perspectives
- Chapter 9: Operational Support

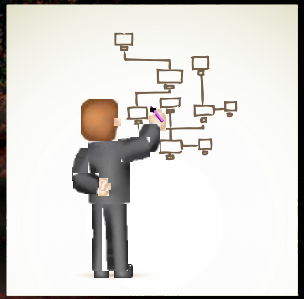
Part IV: Putting Process Mining to Work

- Chapter 10: Tool Support
- Chapter 11: Analyzing "Lasagna Processes"
- Chapter 12: Analyzing "Spaghetti Processes"

Part V: Reflection

- Chapter 13: Cartography and Navigation
- Chapter 14: Epilogue





TU/e