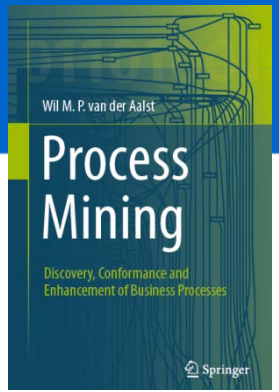
Process Mining: Data Science in Action

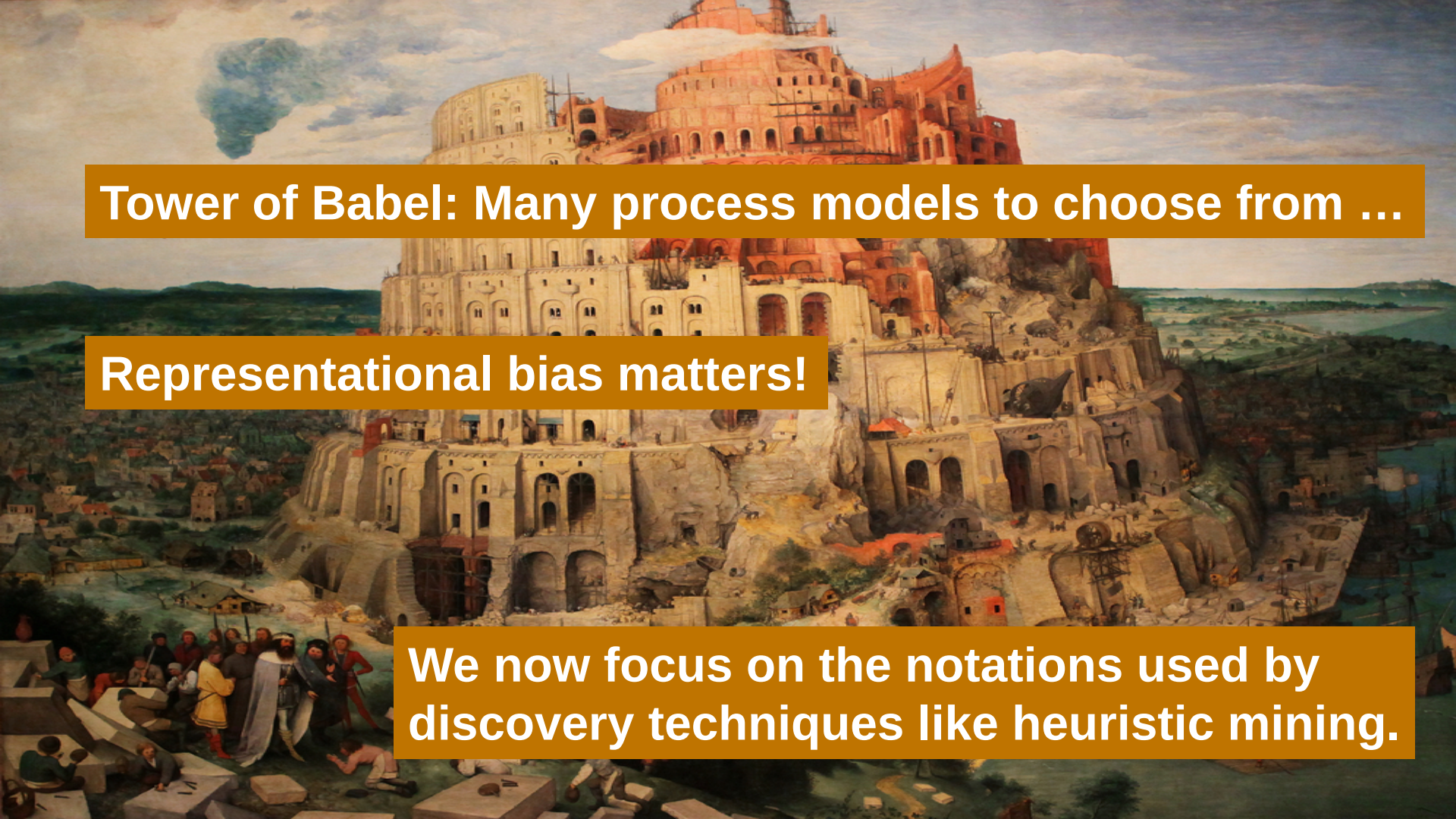
Dependency Graphs and Causal Nets

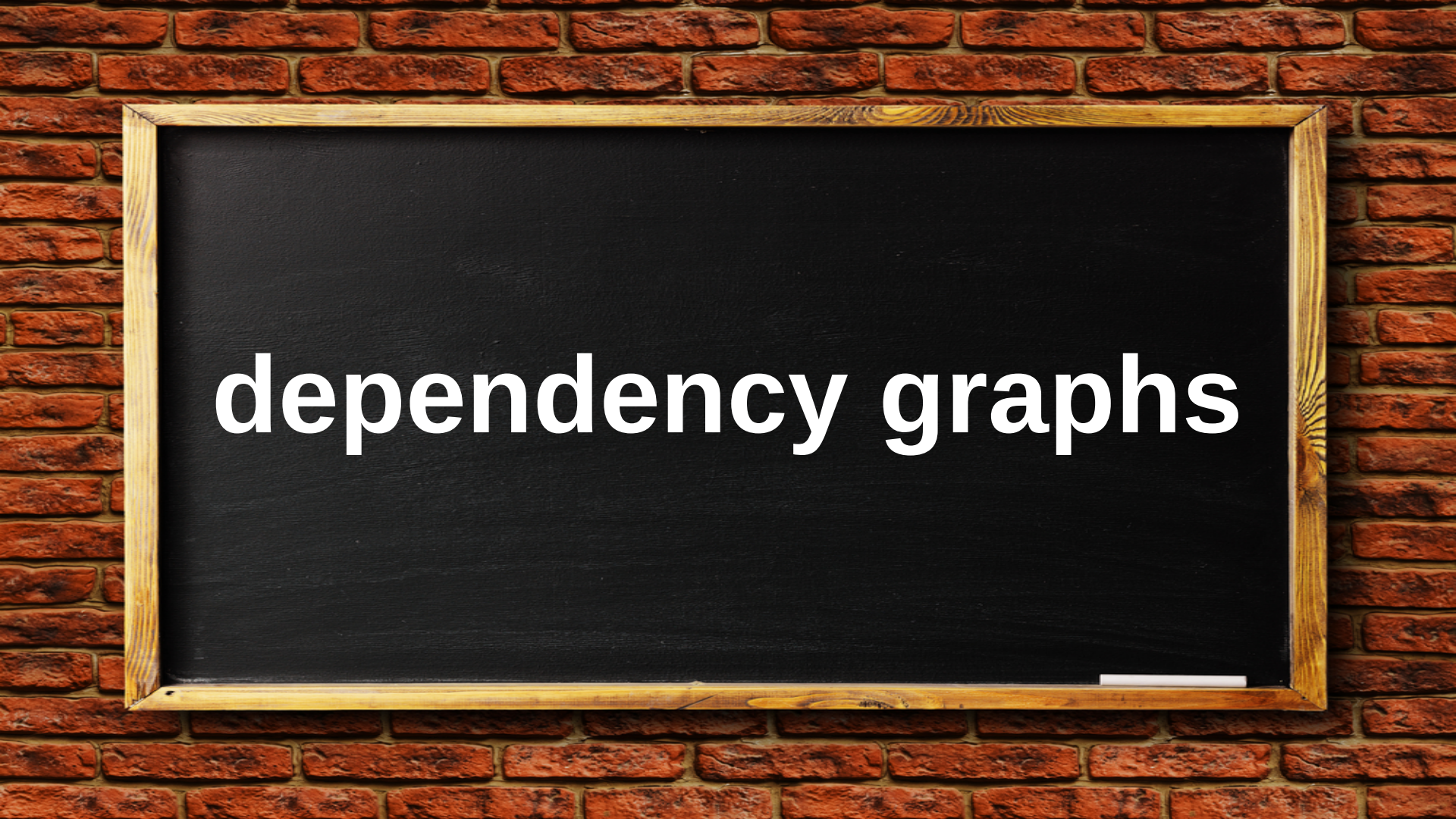


prof.dr.ir. Wil van der Aalst
www.processmining.org

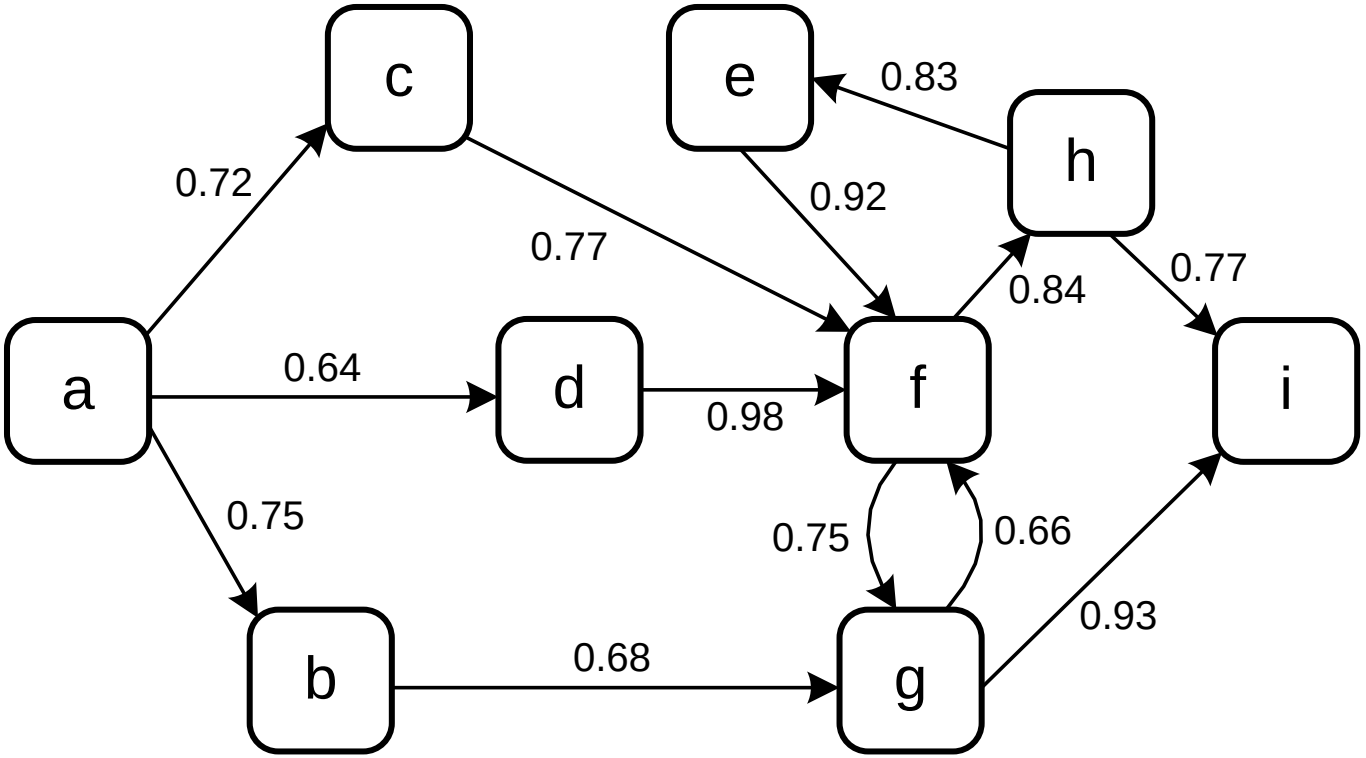
Technische Universiteit Eindhoven University of Technology

Where innovation starts


Tower of Babel: Many process models to choose from …
Representational bias matters!
We now focus on the notations used by discovery techniques like heuristic mining.

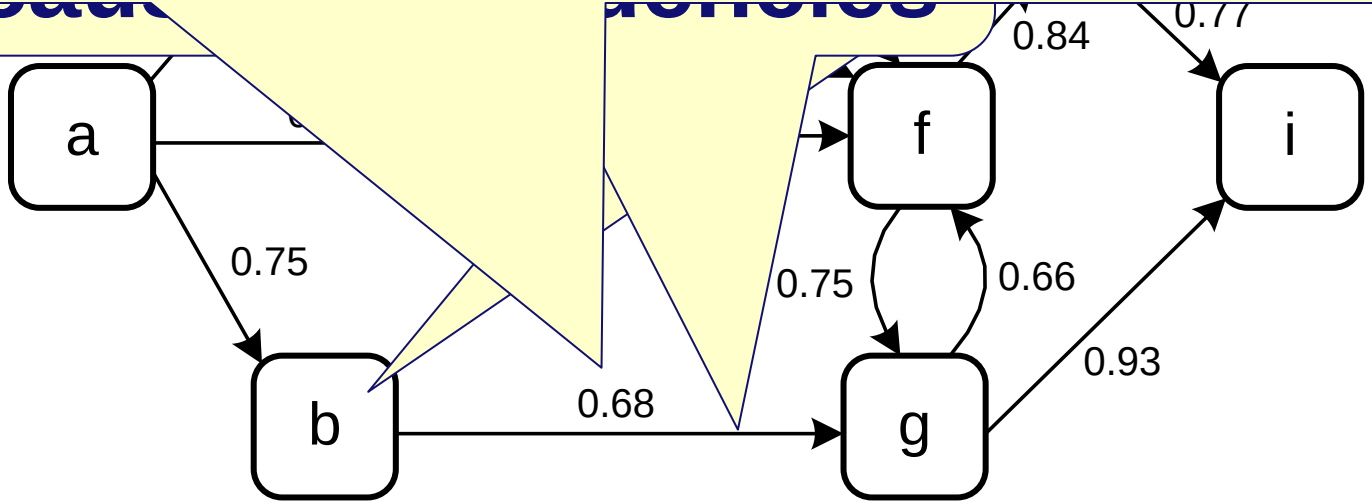

dependency graphs

Dependency graph



Dependency graph

arcs may be annotated with frequency and/or confidence/certainty



TU/e

Challenge: Interleavings should not introduce causal dependencies!

each node can be viewed as a "fuzzy" OR-join & OR-split

executable semantics!

no AND/XOR splits and joins!

TU/e

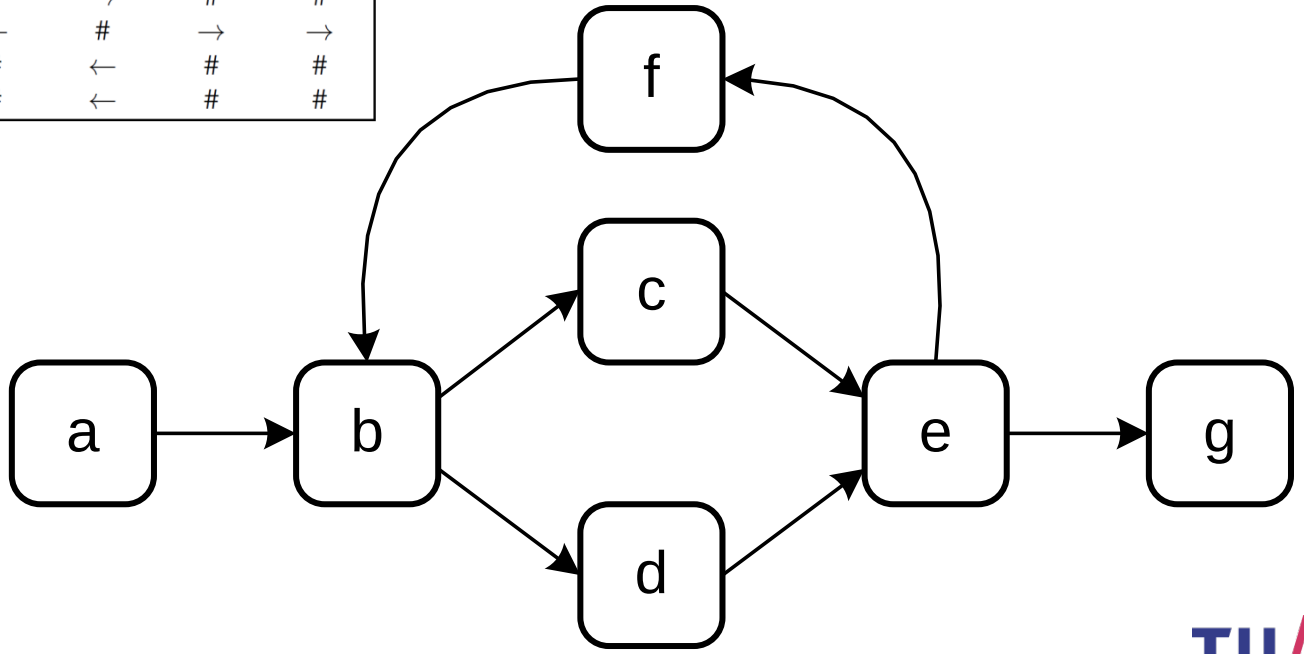
Intuition: Causality relations in footprint

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

TU/e

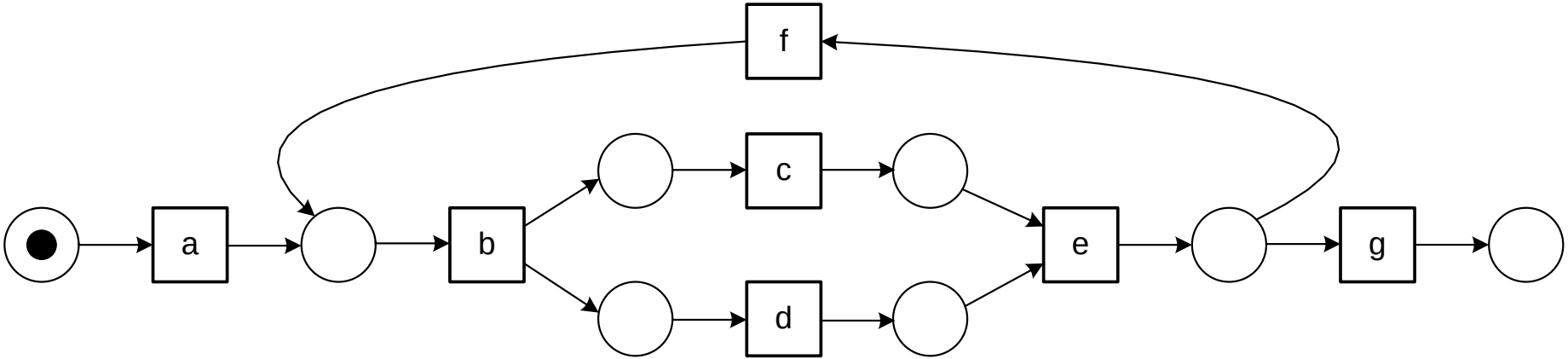
Dependency graph based on footprint

	a	b	c	d	e	f	g
a	#	$\rightarrow$	#	#	#	#	#
b	$\leftarrow$	#	$\rightarrow$	$\rightarrow$	#	$\leftarrow$	#
c	#	$\leftarrow$	#	$\|\|$	$\rightarrow$	#	#
d	#	$\leftarrow$	$\|\|$	#	$\rightarrow$	#	#
e	#	#	$\leftarrow$	$\leftarrow$	#	$\rightarrow$	$\rightarrow$
f	#	$\rightarrow$	#	#	$\leftarrow$	#	#
g	#	#	#	#	$\leftarrow$	#	#



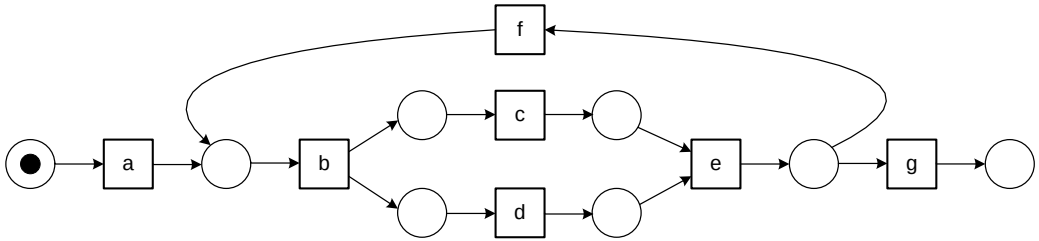
TU/e

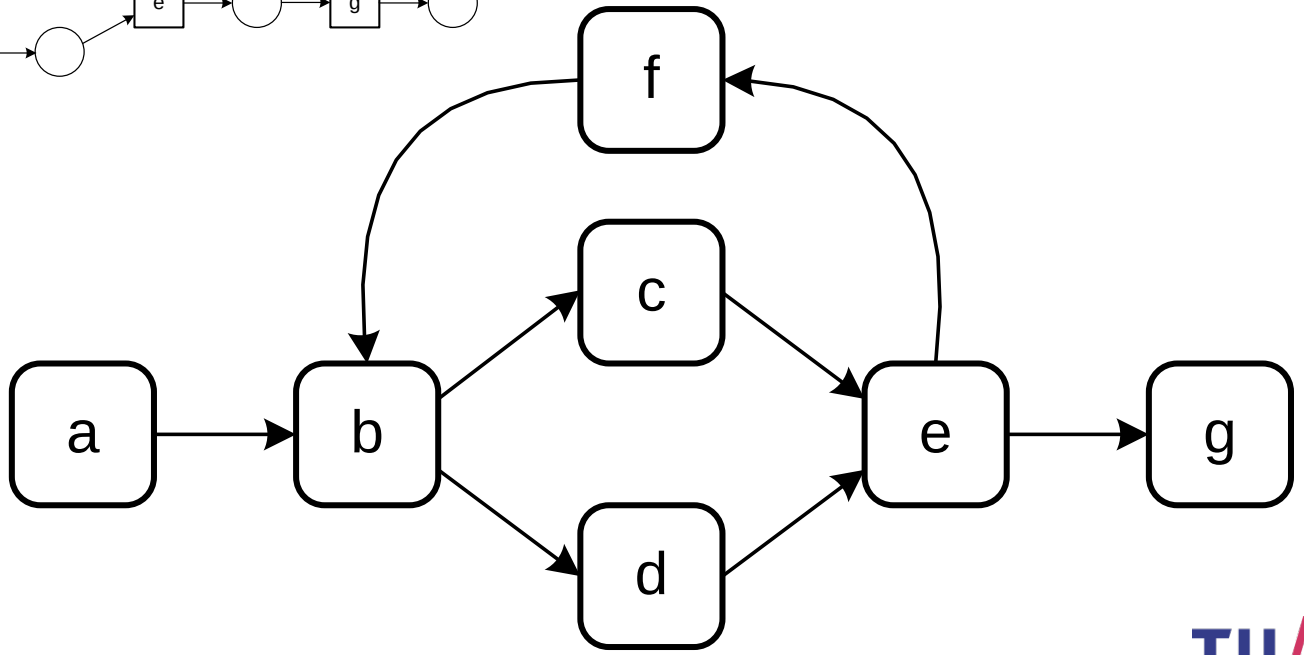
Intuition: Causality relations in Petri net



TU/e

Dependency graph based on Petri net

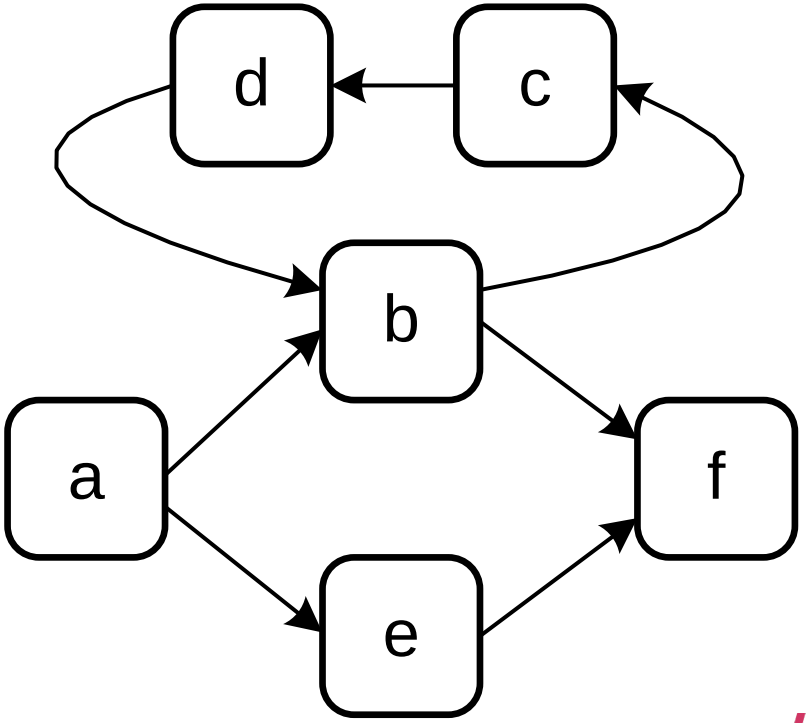




TU/e

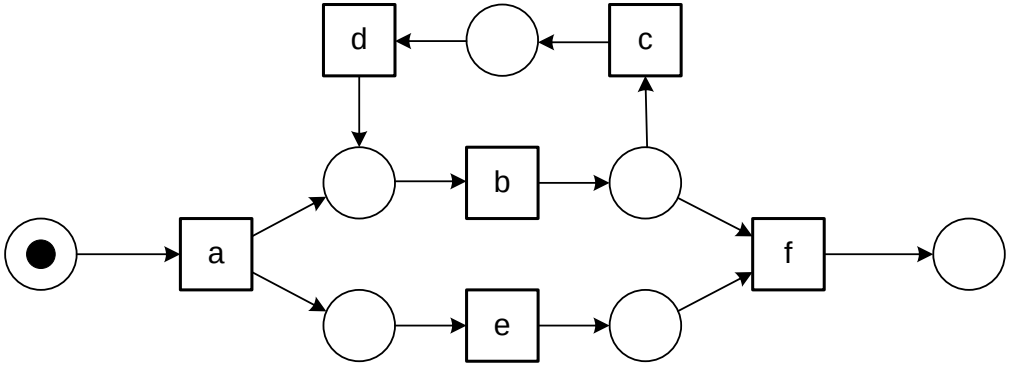
Another example: Dependency graph based on footprint

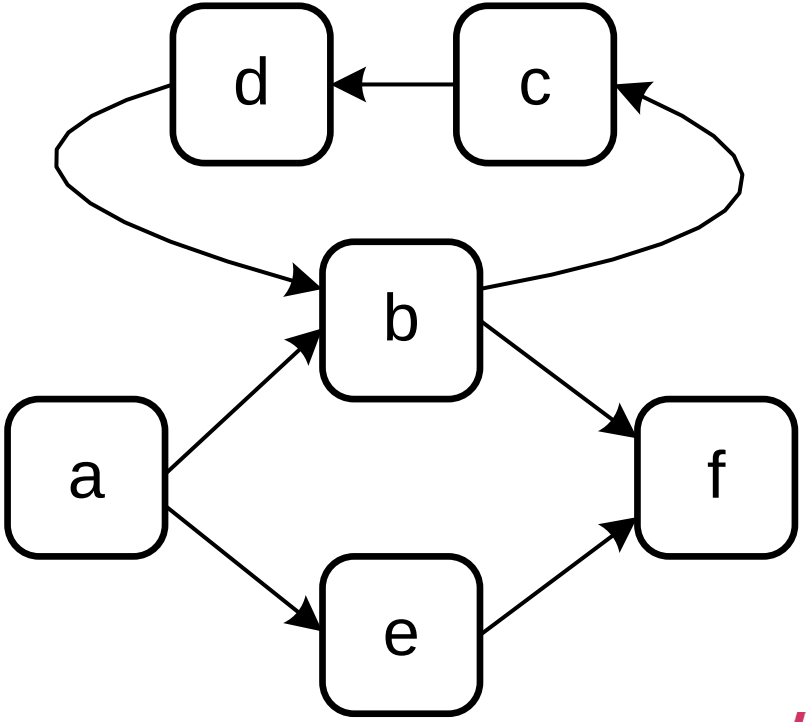
	a	b	c	d	e	f
a	#	$\rightarrow$	#	#	$\rightarrow$	#
b	$\leftarrow$	#	$\rightarrow$	$\leftarrow$	$\|\|$	$\rightarrow$
c	#	$\leftarrow$	#	$\rightarrow$	$\|\|$	#
d	#	$\rightarrow$	$\leftarrow$	#	$\|\|$	#
e	$\leftarrow$	$\|\|$	$\|\|$	$\|\|$	#	$\rightarrow$
f	#	$\leftarrow$	#	#	$\leftarrow$	#



TU/e

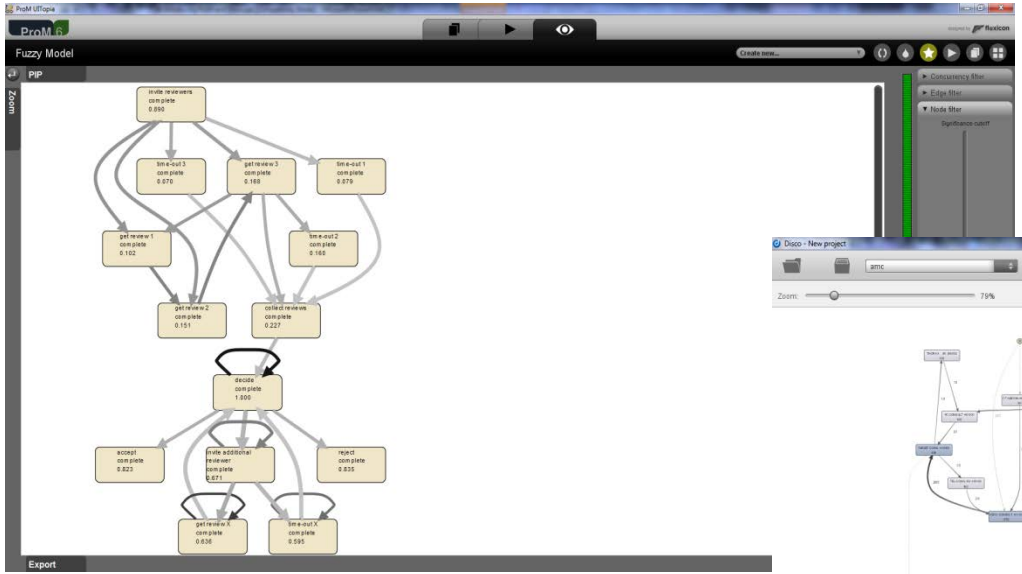
Another example: Dependency graph based on Petri net



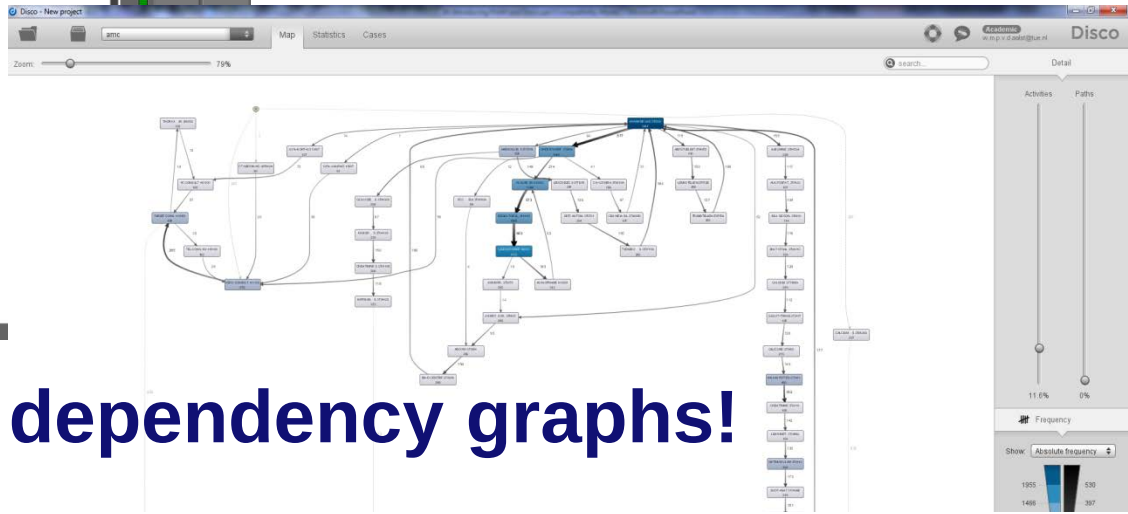


TU/e

Fuzzy models can be viewed as dependency graphs

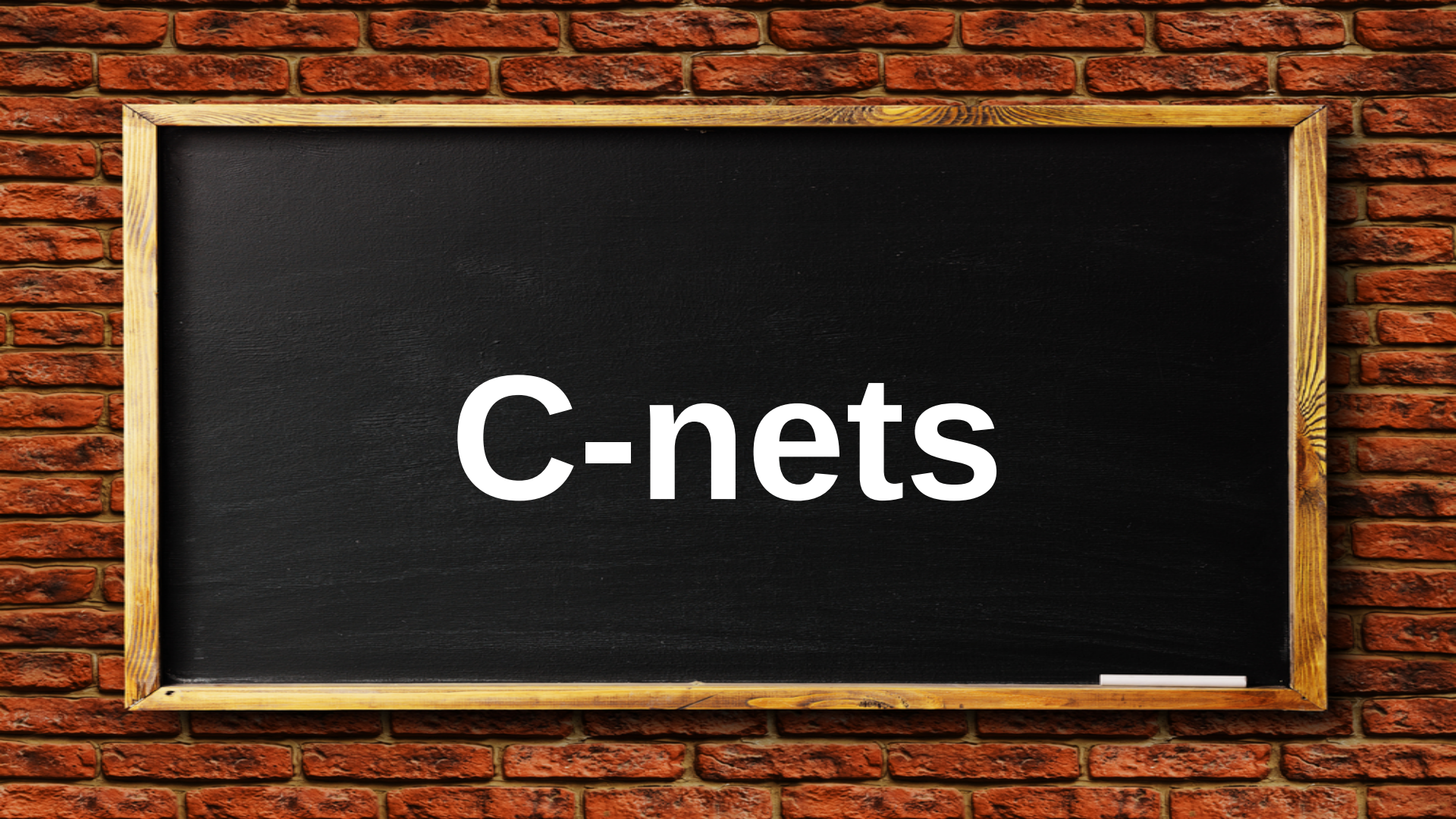


No precise semantics!

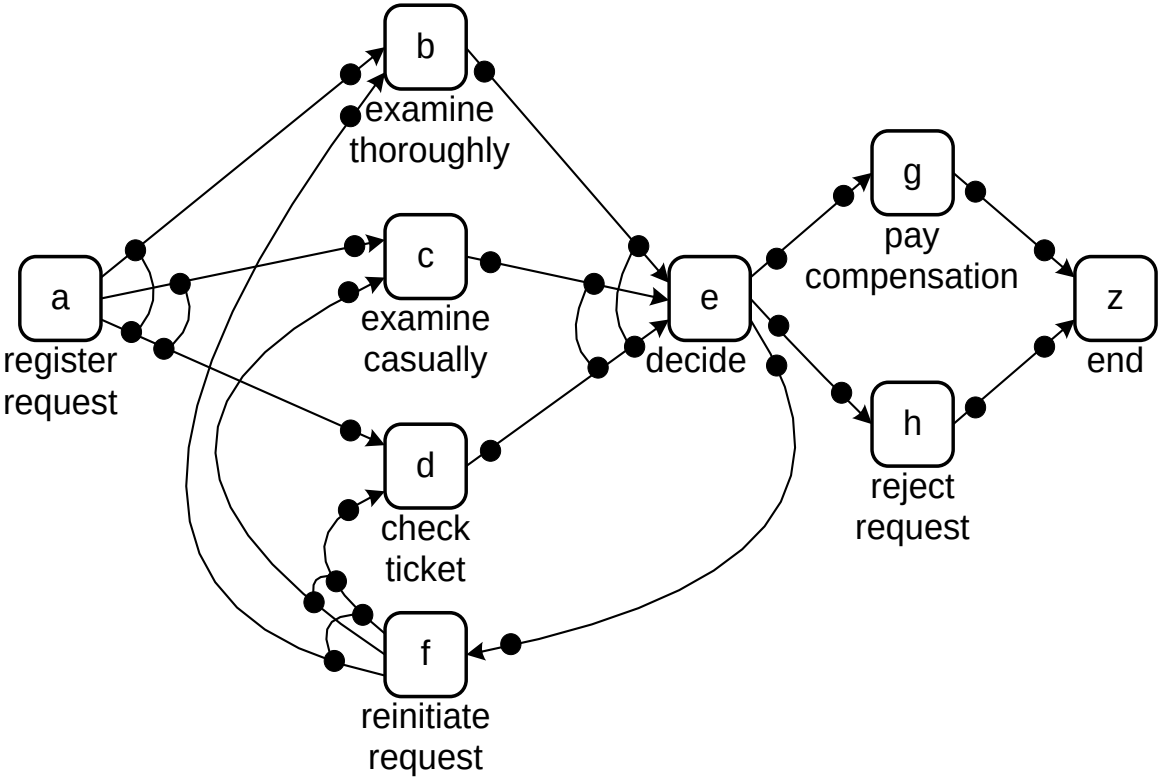


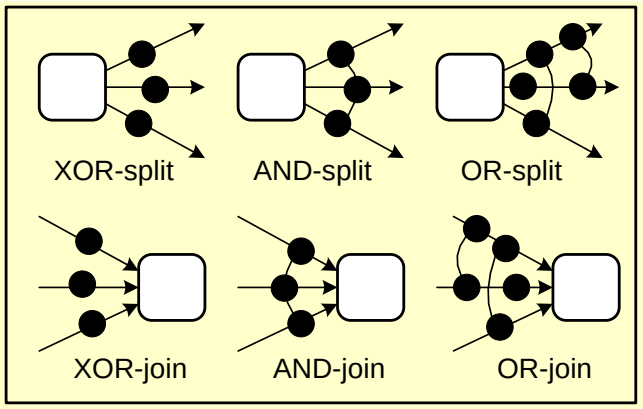
Many ways to create dependency graphs!

Often based on heuristics …


C-nets

Causal nets (C-nets)

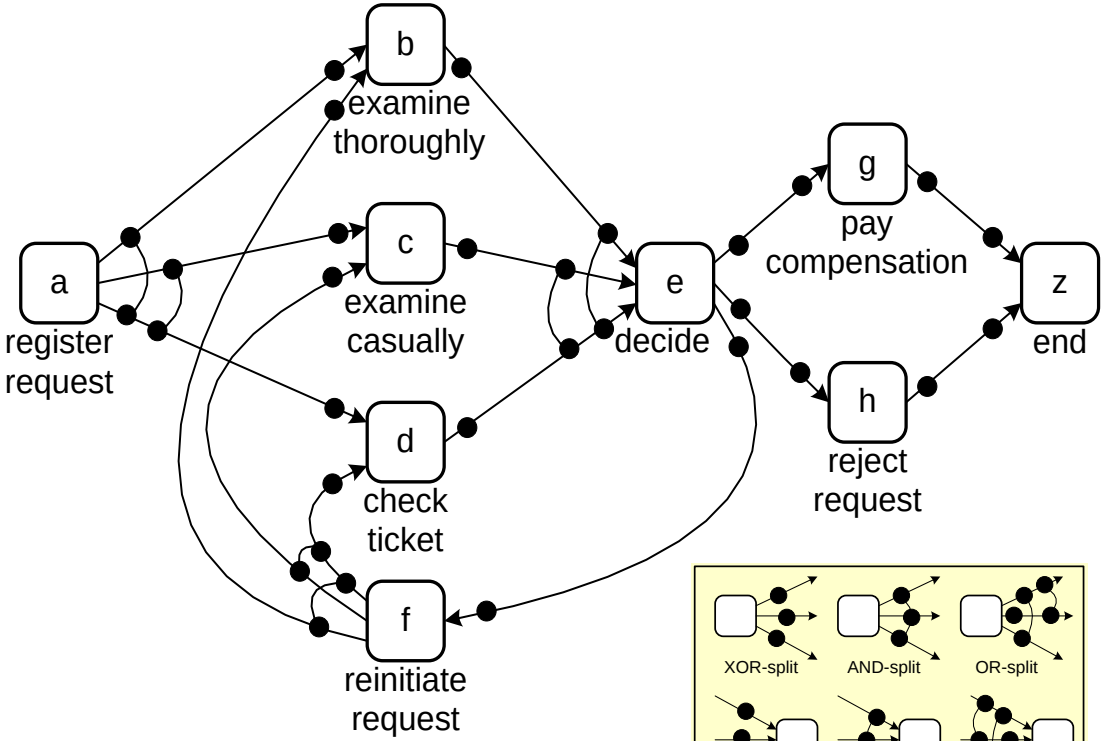




Why C-nets?

- Output of several mining techniques, e.g., the well-known heuristics miner.
- Fits well with mainstream languages (BPMN, EPCs, YAWL, BPEL, etc.).
- Able to model XOR, AND, and OR, but no silent steps or duplicate activities needed.
- Avoiding non-sound models.

Semantics: Loose interpretation



Provides replay semantics rather than execution semantics, e.g., the moment of choice is not fixed.

TU/e

Formal semantics (Chapter 2)

Definition 2.8 (Causal net) A *Caus*
where:

- $A \subseteq \mathscr{A}$ is a finite set of *activities*
- $a_i \in A$ is the *start activity*
- $a_o \in A$ is the *end activity*
- $D \subseteq A \times A$ is the *dependency re*
- $AS = \{X \subseteq \mathscr{P}(A) \mid X = \{\emptyset\} \vee \emptyset$
- $I \in A \to AS$ defines the set of po
- $O \in A \to AS$ defines the set of p

such that

- $D = \{(a_1, a_2) \in A \times A \mid a_1 \in \bigcup_a$
- $D = \{(a_1, a_2) \in A \times A \mid a_2 \in \bigcup_a$
- $\{a_i\} = \{a \in A \mid I(a) = \{\emptyset\}\}$
- $\{a_o\} = \{a \in A \mid O(a) = \{\emptyset\}\}$
- All activities in the graph (A, D)

Definition 2.9 (Binding) Let $C = (A, a_i, a_o, D, I, O)$ be a C-net. $B = \{(a, as^I, as^O) \in A \times \mathscr{P}(A) \times \mathscr{P}(A) \mid as^I \in I(a) \wedge as^O \in O(a)\}$ is the set of *activity bindings*. A *binding sequence* σ is a sequence of activity bindings, i.e., $\sigma \in B^*$

Definition 2.10 (State) Let $C = (A, a_i, a_o, D, I, O)$ be a C-net. $S = \mathbb{B}(A \times A)$ is the *state space* of C. $s \in S$ is a *state*, i.e., a multi-set of pending *obligations*. Function $\psi \in B^* \to S$ is defined inductively: $\psi(\langle\,\rangle) = [\,]$ and $\psi(\sigma \oplus (a, as^I, as^O)) = (\psi(\sigma) \setminus (as^I \times \{a\})) \uplus (\{a\} \times as^O)$ for any binding sequence $\sigma \oplus (a, as^I, as^O) \in B^*$. [3] $\psi(\sigma)$ is the state after executing binding sequence σ

Definition 2.11 (Valid) Let $C = (A, a_i, a_o, D, I, O)$ be a C-net and $\sigma = \langle (a_1, as_1^I, as_1^O), (a_2, as_2^I, as_2^O), \ldots, (a_n, as_n^I, as_n^O)\rangle \in B^*$ a binding sequence. σ is a *valid sequence* of C if and only if:

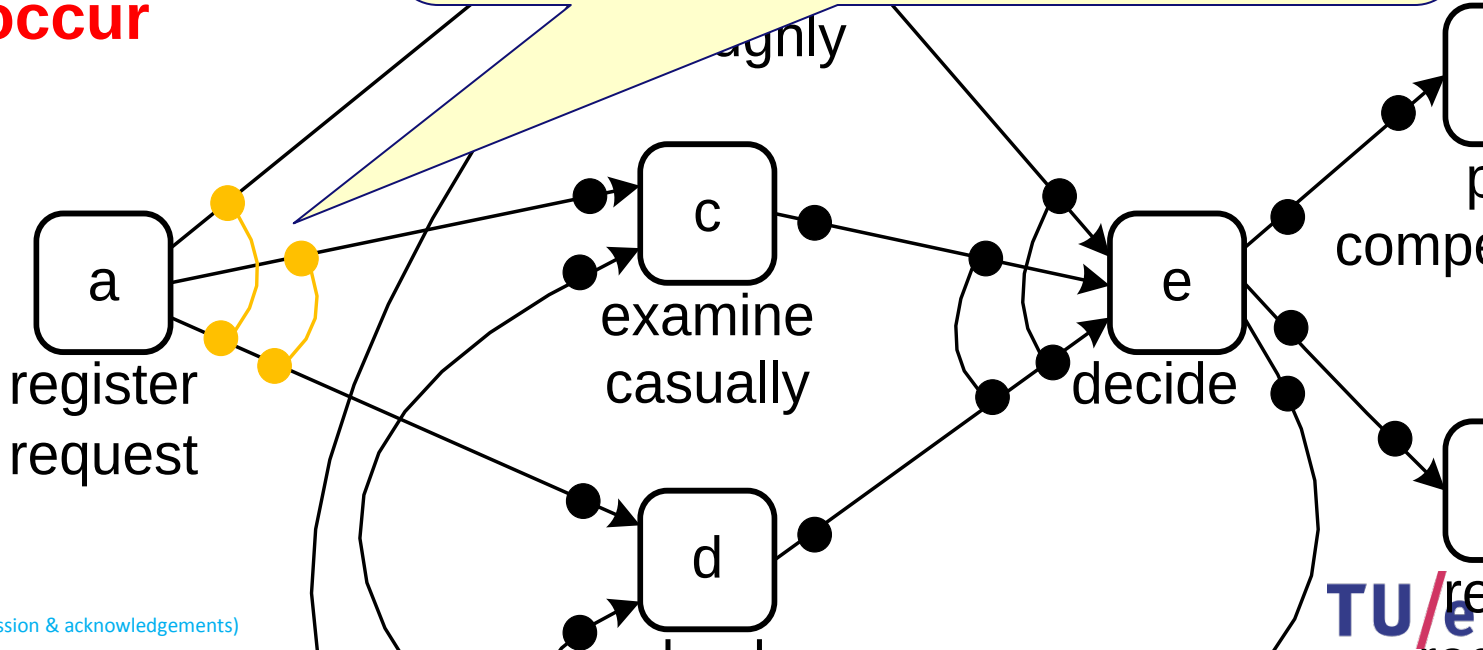
- $a_1 = a_i$, $a_n = a_o$, and $a_k \in A \setminus \{a_i, a_o\}$ for $1 < k < n$
- $\psi(\sigma) = [\,]$
- For any prefix $\langle (a_1, as_1^I, as_1^O), (a_2, as_2^I, as_2^O), \ldots, (a_k, as_k^I, as_k^O)\rangle = \sigma' \oplus (a_k, as_k^I, as_k^O) \in pref(\sigma)$: $(as_k^I \times \{a_k\}) \leq \psi(\sigma')$

$V(C)$ is the set of all valid sequences of C.

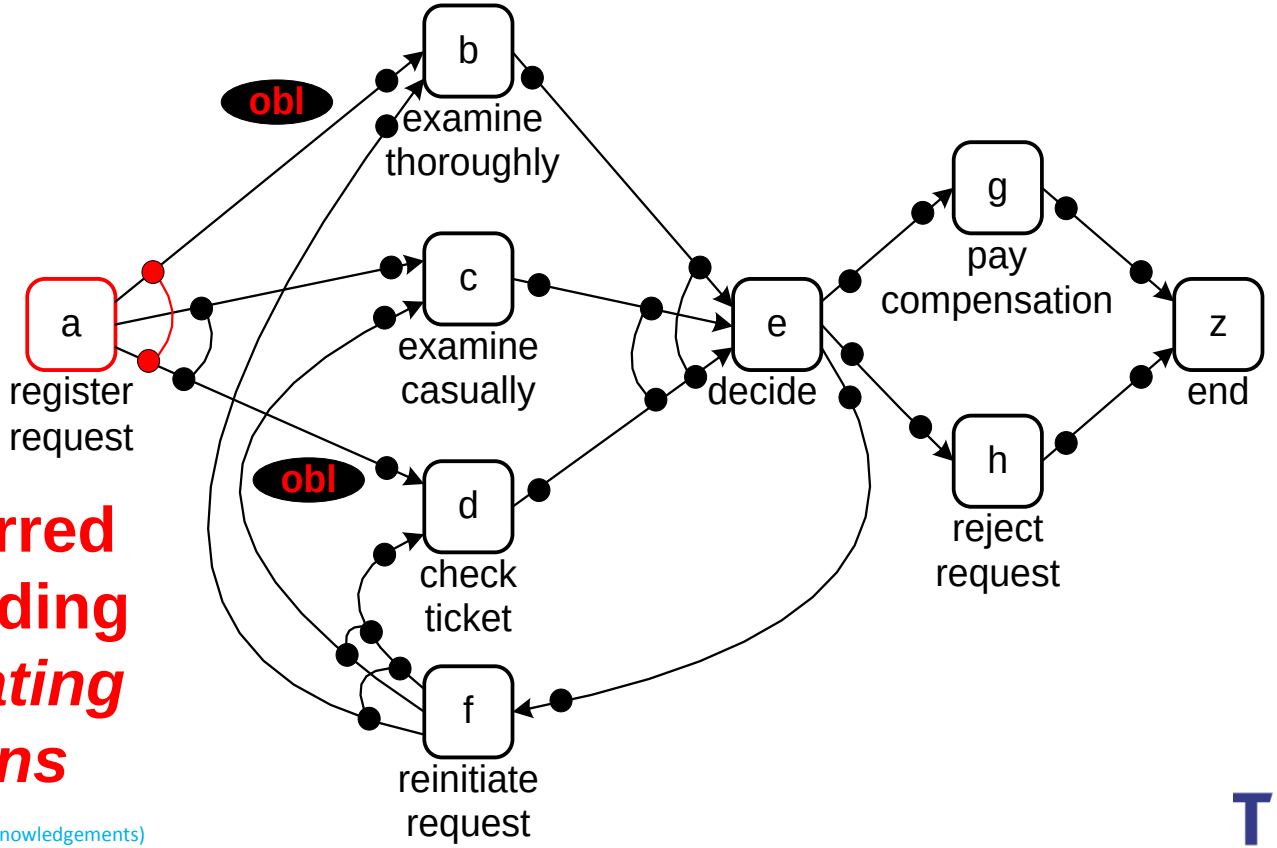
Initial state: Only activity *a* can occur

activity *a* is the start activity and will be the first to occur

activity a has two output bindings



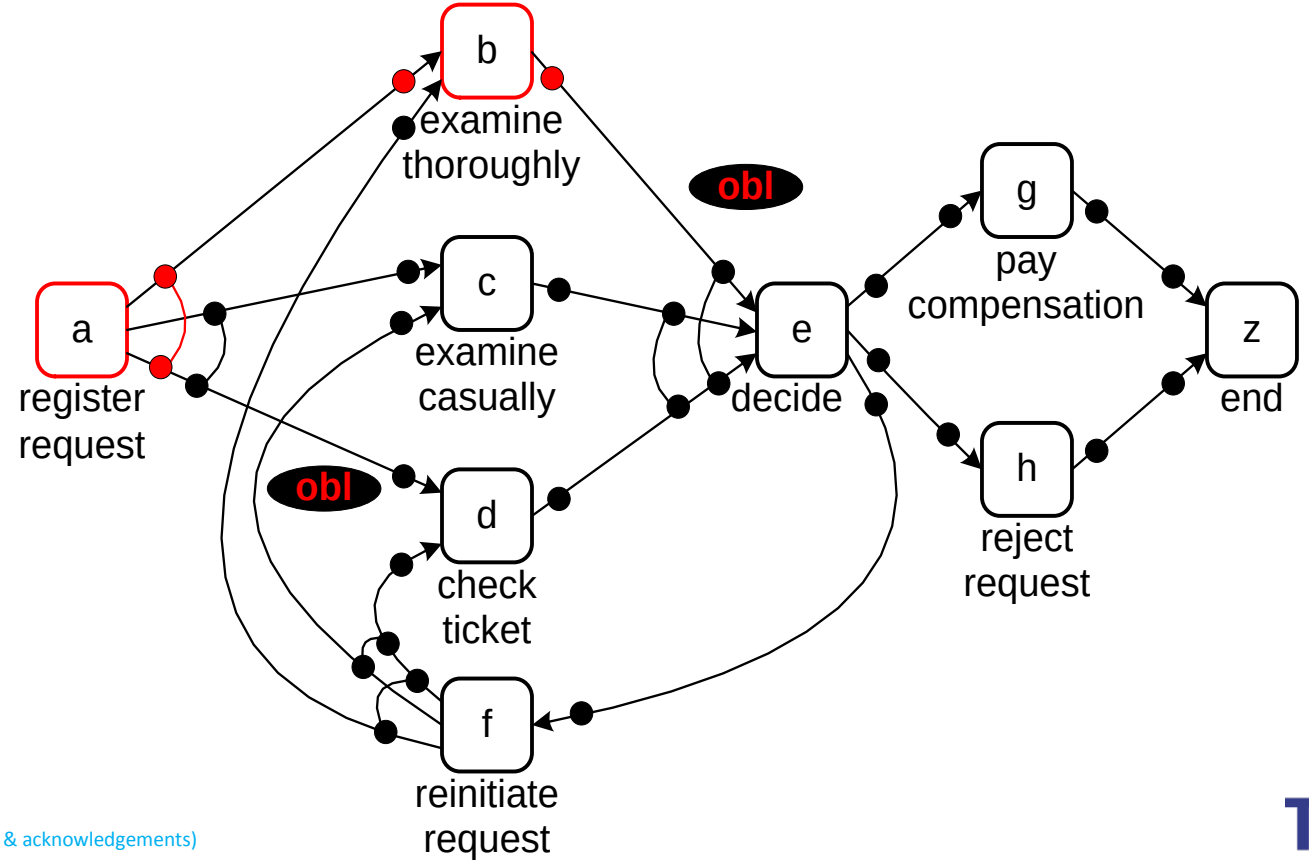
Assume activity *a* occurs with an output binding enabling *b* and *d*



activity a* occurred with output binding *{b,d}*, *thus creating two obligations

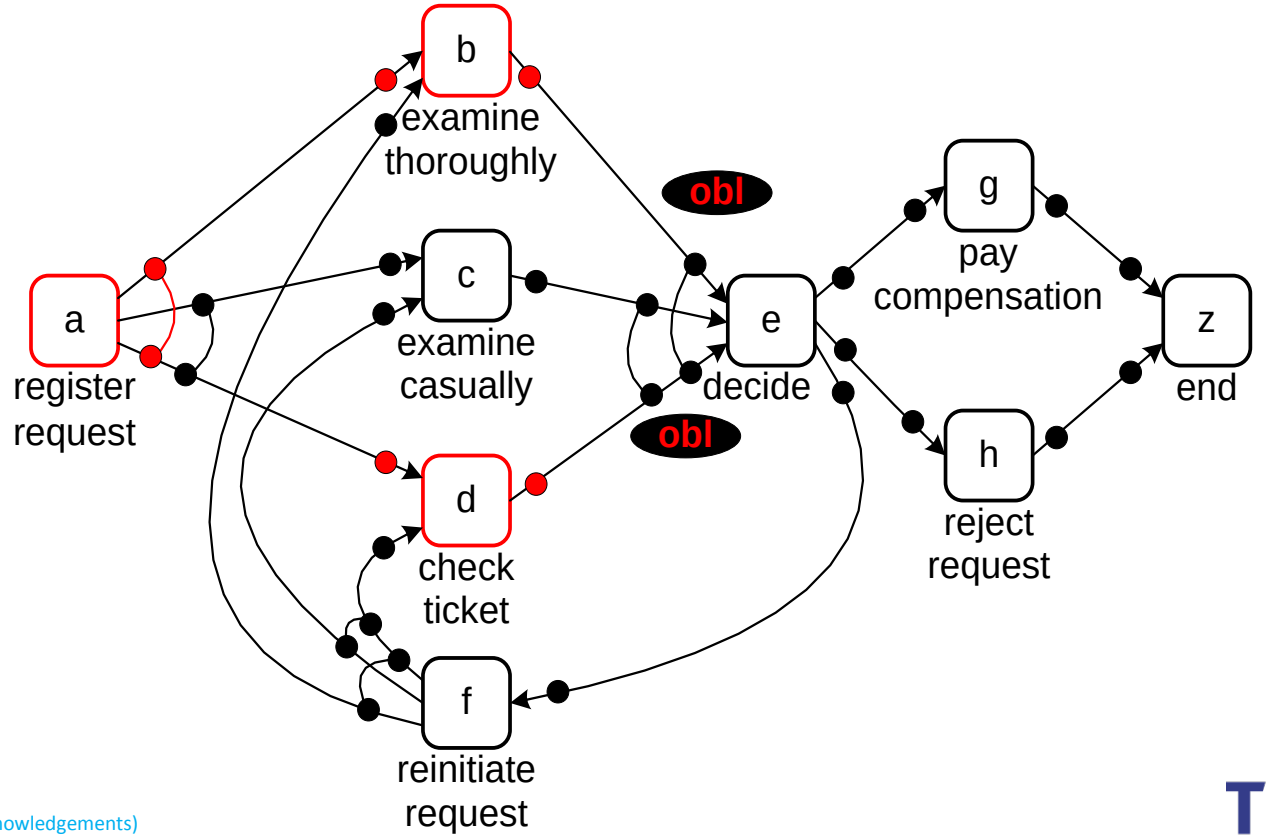
TU/e

Activity *b* occurs, removes obligation *(a,b)* and creates obligation *(b,e)*

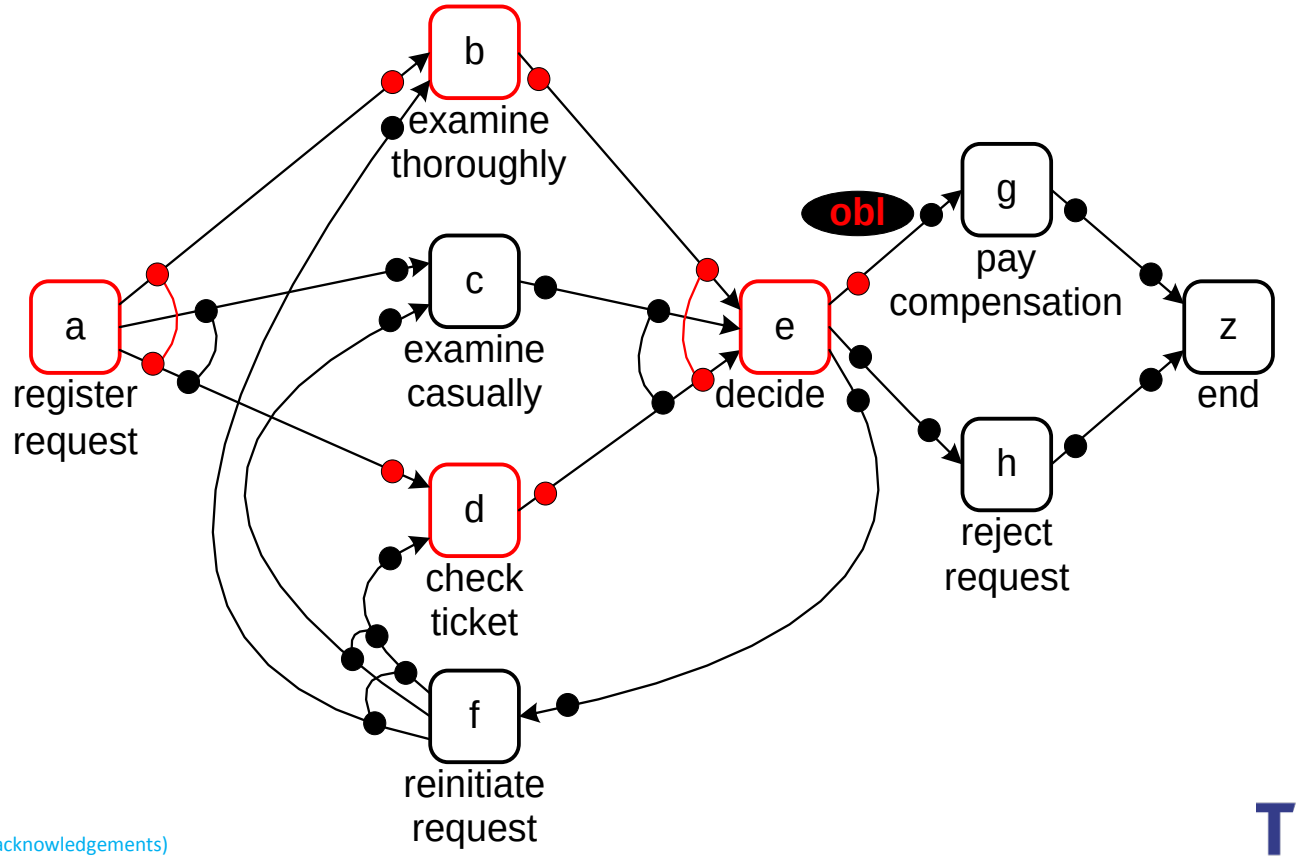


TU/e

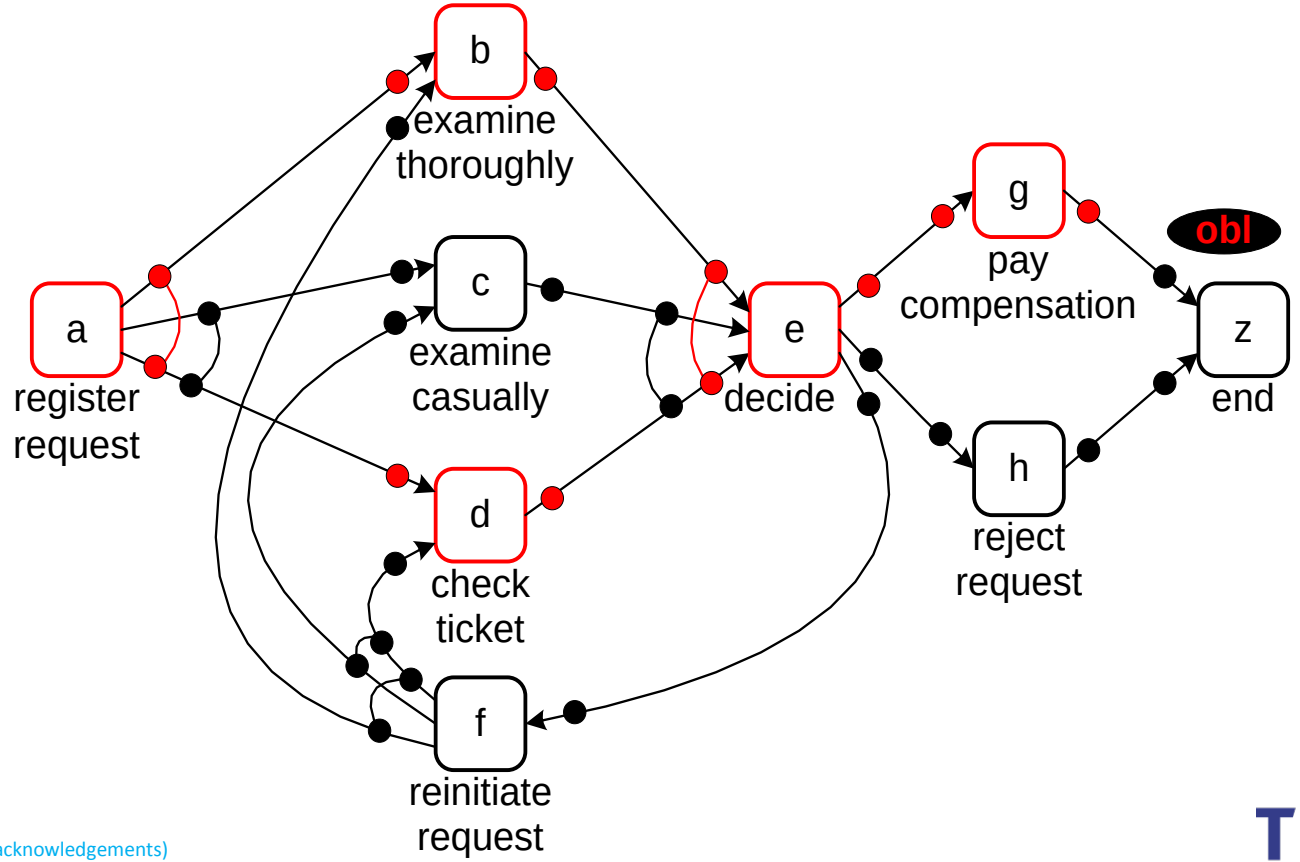
Activity *d* occurs, removes obligation *(a,d)* and creates obligation *(d,e)*



Activity *e* occurs, removes obligations *(b,e)* *and (d,e)* and creates obligation *(e,g)*

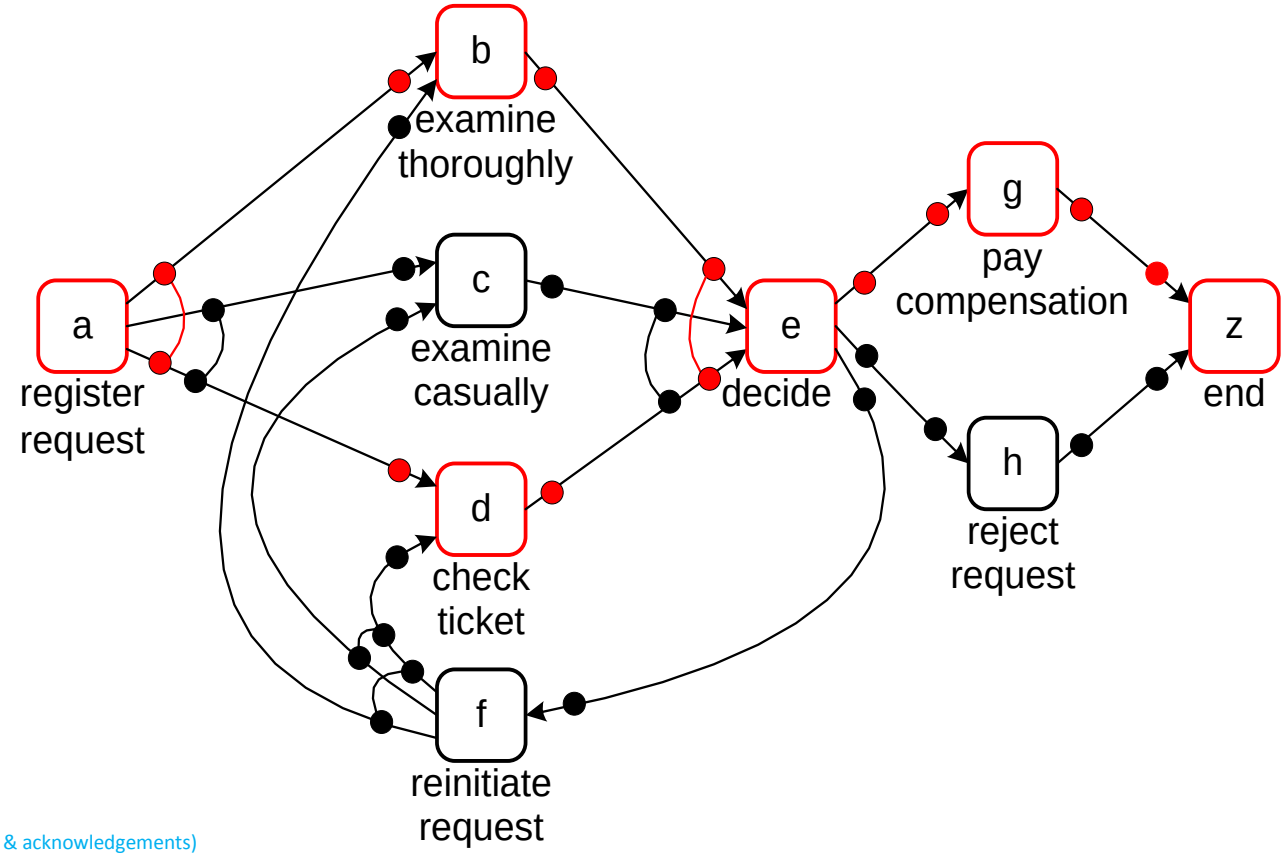


Activity *g* occurs, removes obligation *(e,g)* and creates obligation *(g,z)*



TU/e

Activity *z* occurs and removes obligation *(g,z)* while leaving no other obligations

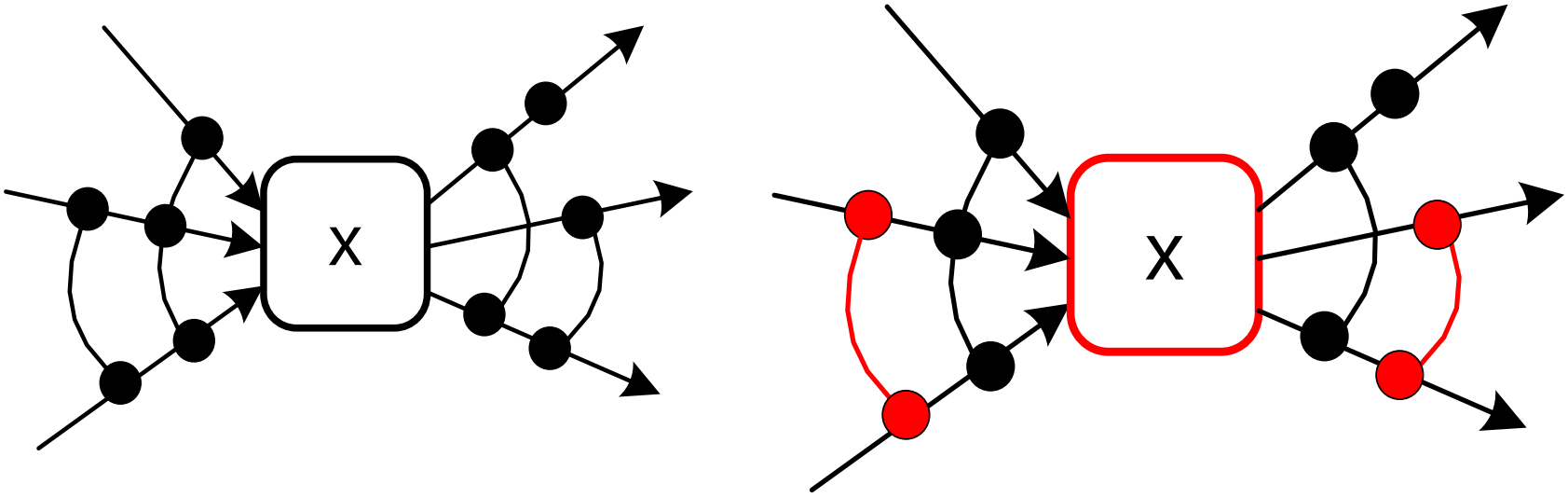


TU/e

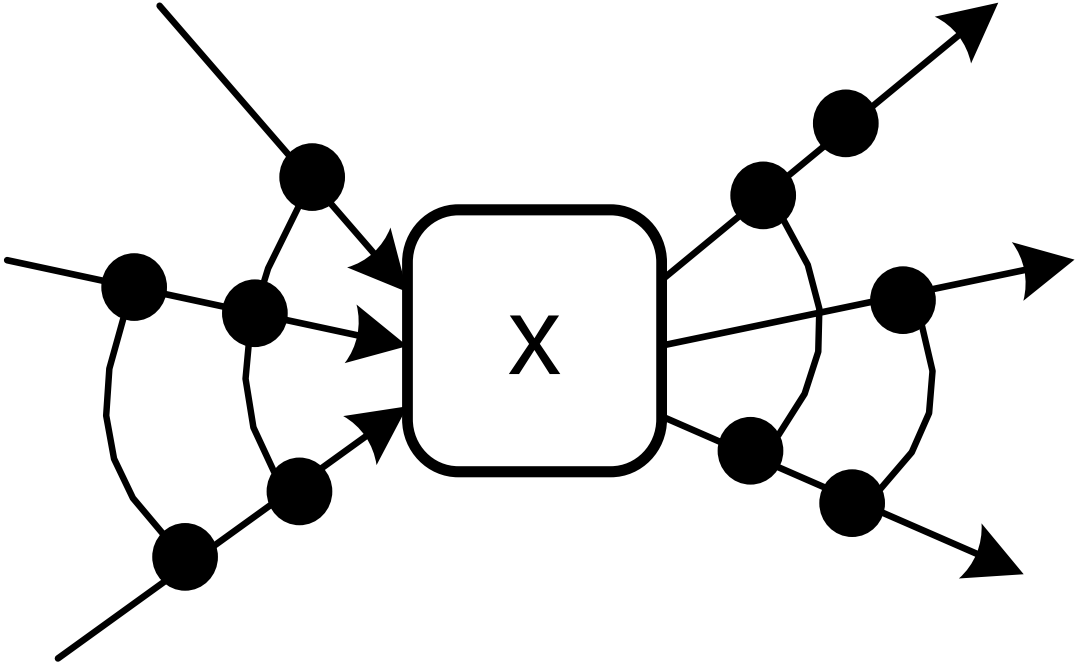
Rules of the game

- Start with the start activity of the C-net.
- End with the end activity of the C-net.
- The start and end activities cannot also happen in-between start and end.
- Obligations are like tokens (need to be there in order to be consumed).
- At the end there should be no remaining obligations.

Example binding

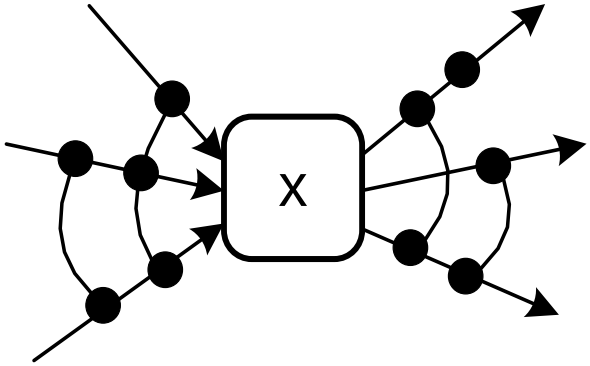


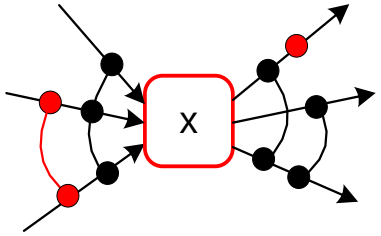
Question: How many bindings are possible?

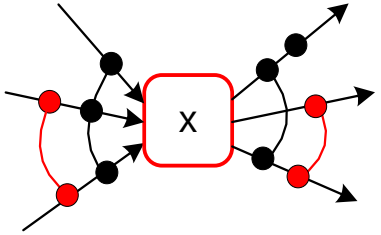


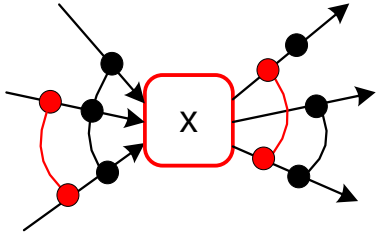
TU/e

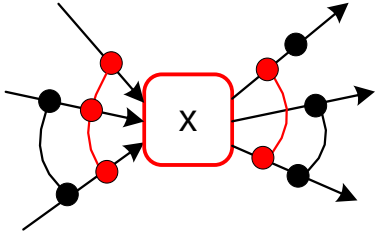
Six bindings are possible

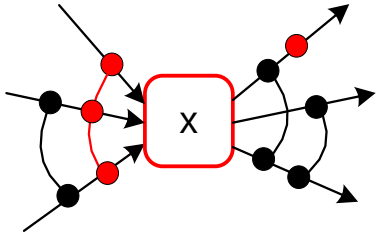


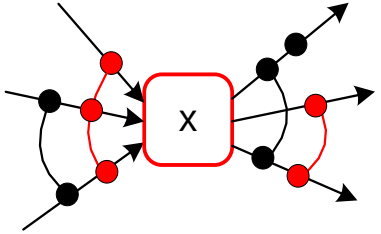




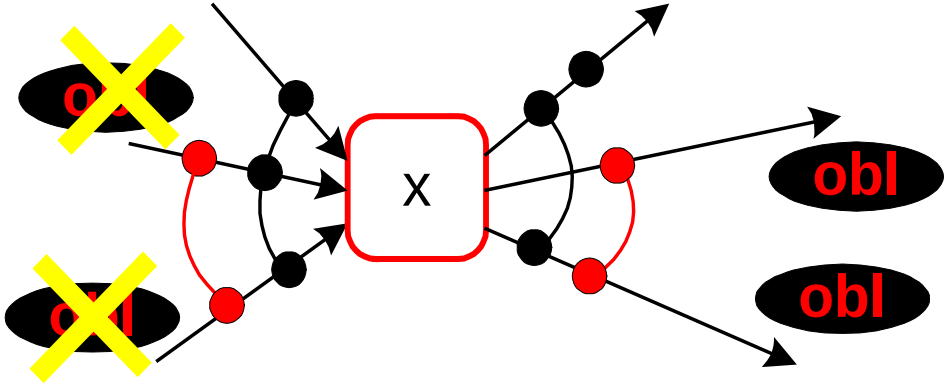








Valid binding sequences

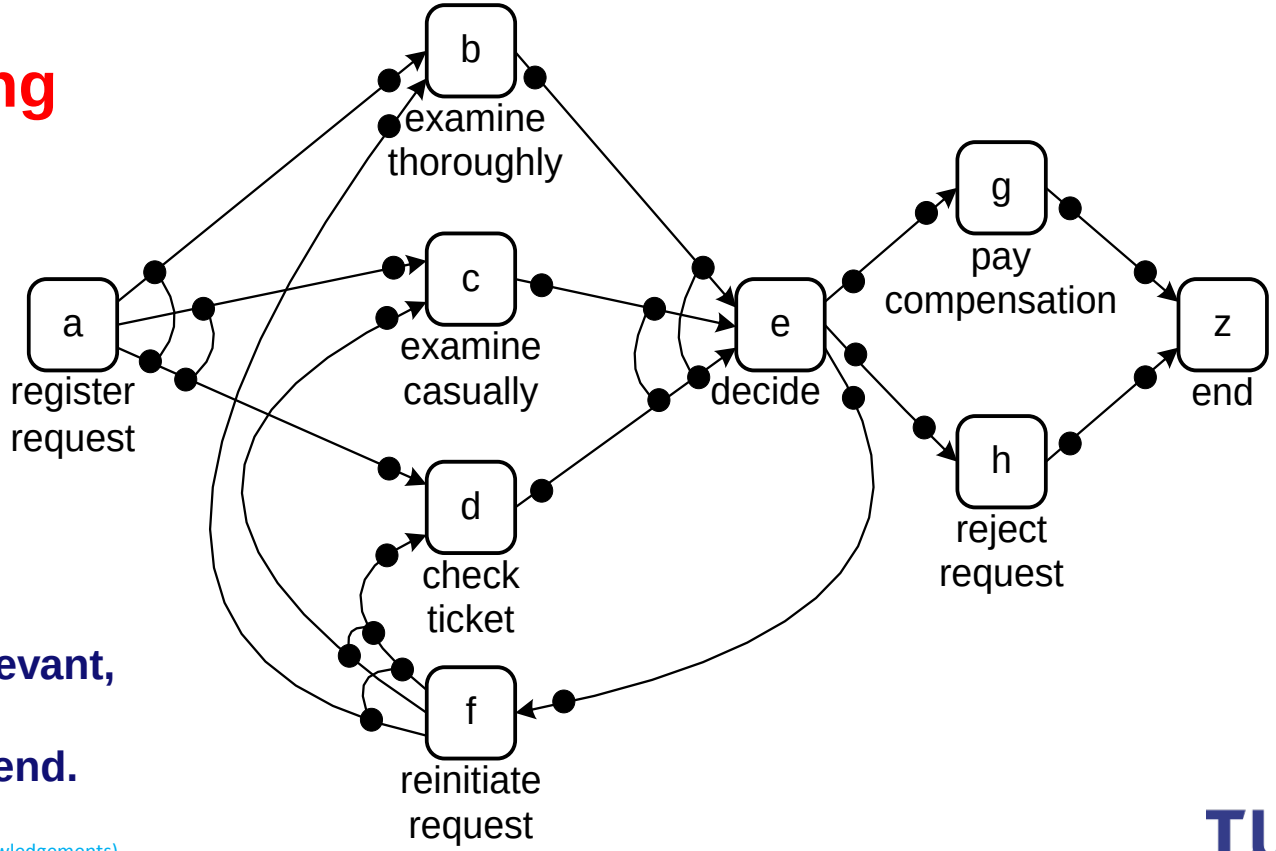


- Start with start activity without any obligations and end with end activity without any pending obligations.
- Input bindings remove existing obligations and output bindings create obligations.

TU/e

Semantics are declarative!

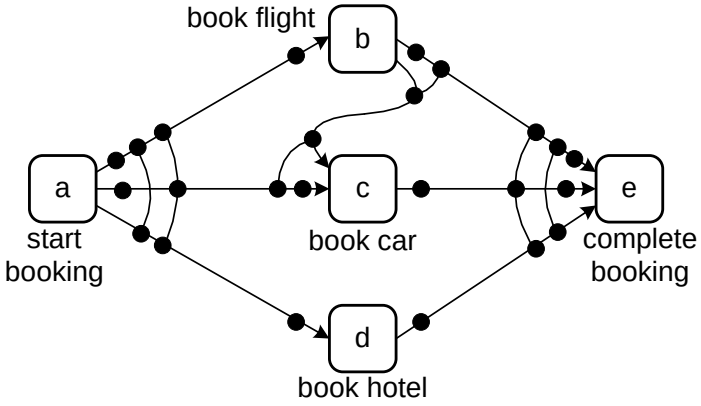
Only valid binding sequences are considered !!!



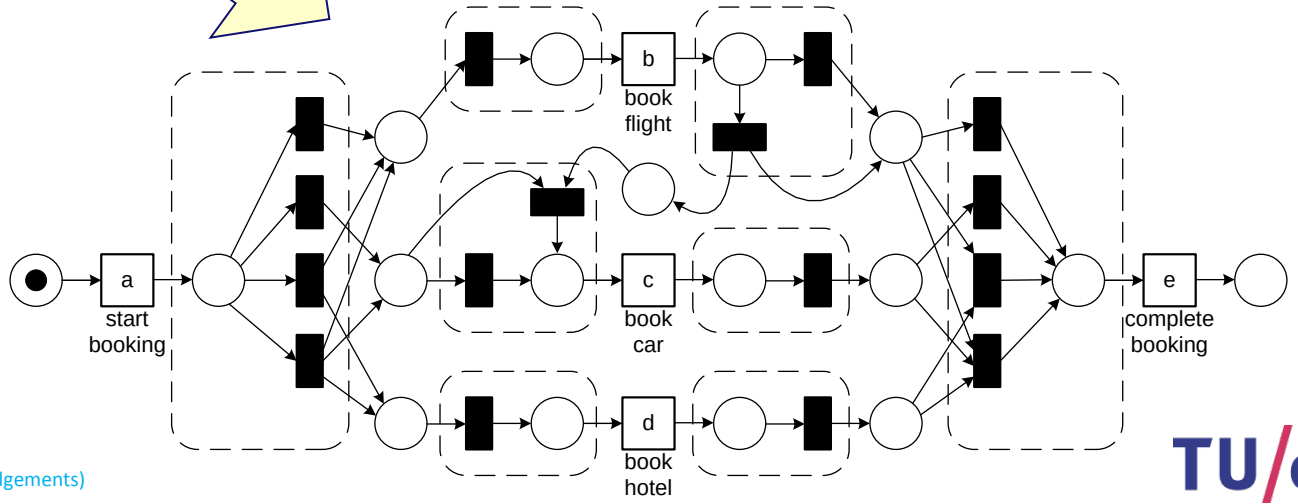
Replay semantics:
Moment of choice is irrelevant, i.e., bindings can be "reconsidered" until the end.

TU/e

Relating C-nets to WF-nets



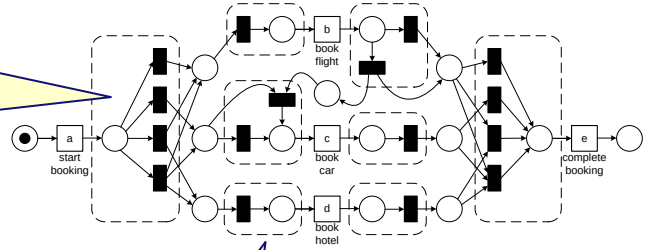
WF-net may deadlock, leave tokens behind, etc.



TU/e

Relating C-nets to WF-nets

also deadlocking or livelocking firing sequences are considered



a
start bookin

book hotel

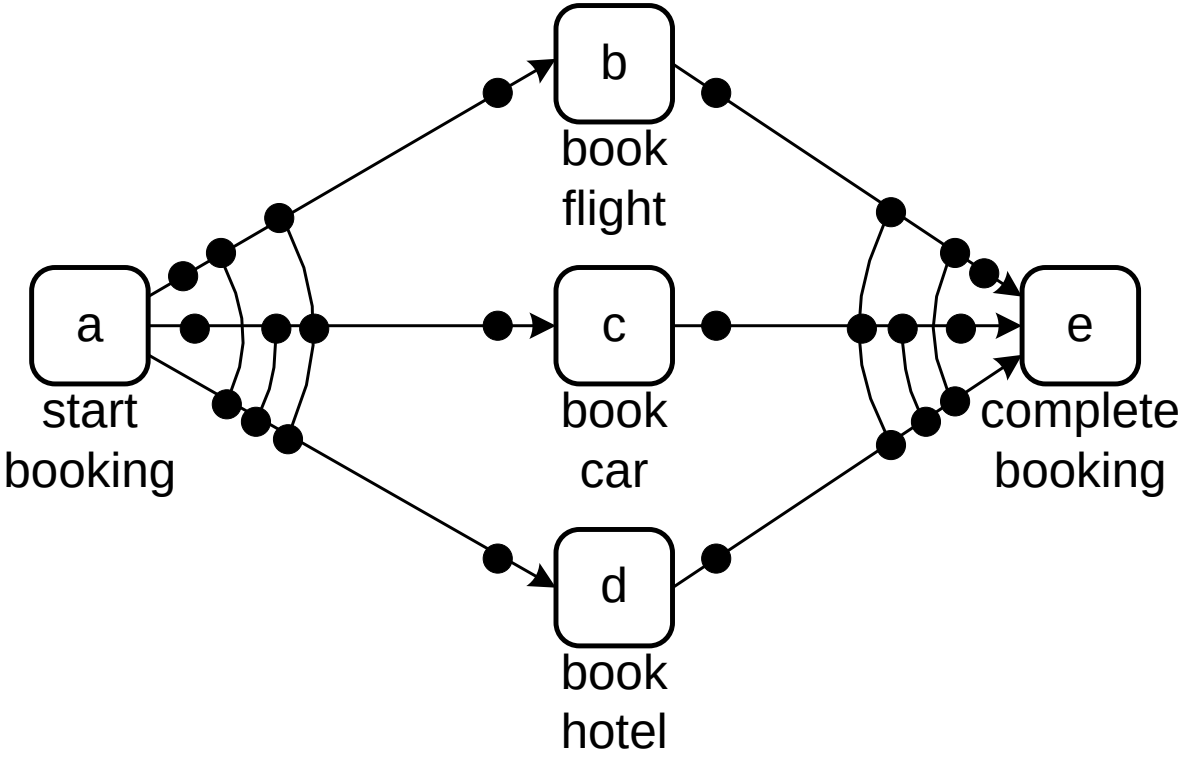
valid binding sequence of C-net

valid firing sequence of WF-net

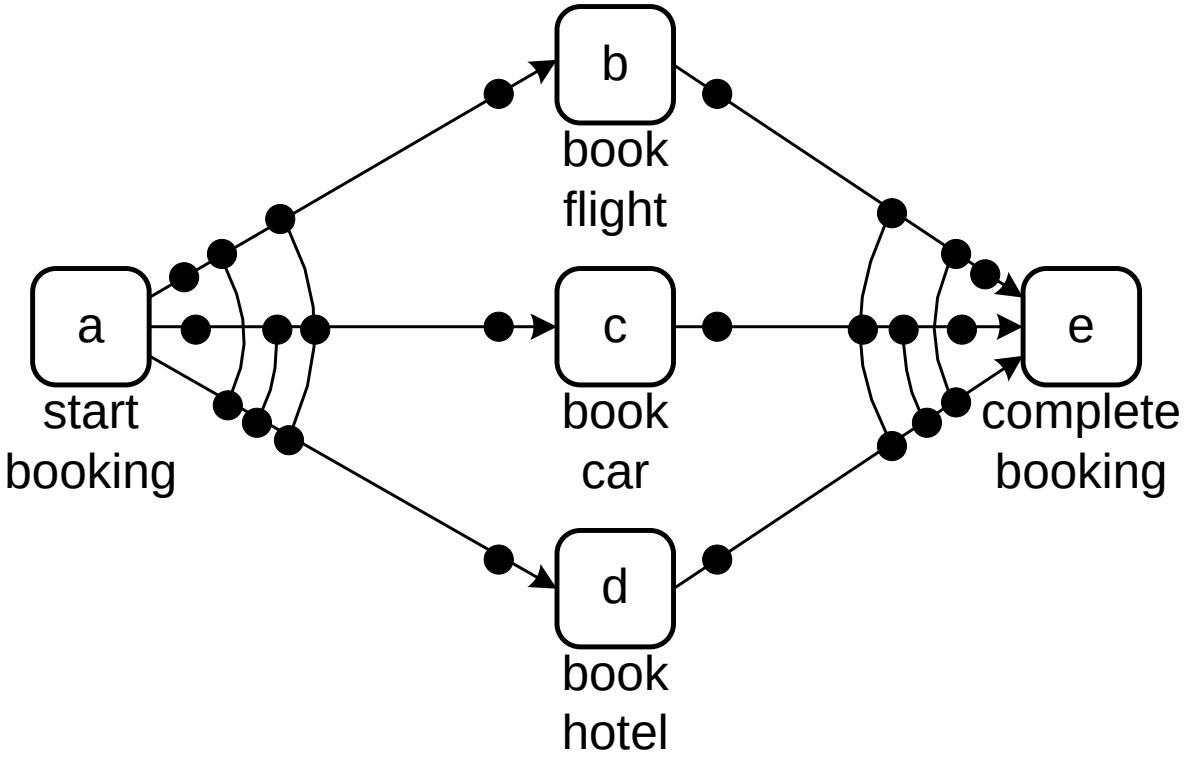
only valid binding sequences are considered

WF-net does not need to be sound

Question
How many valid binding sequences?

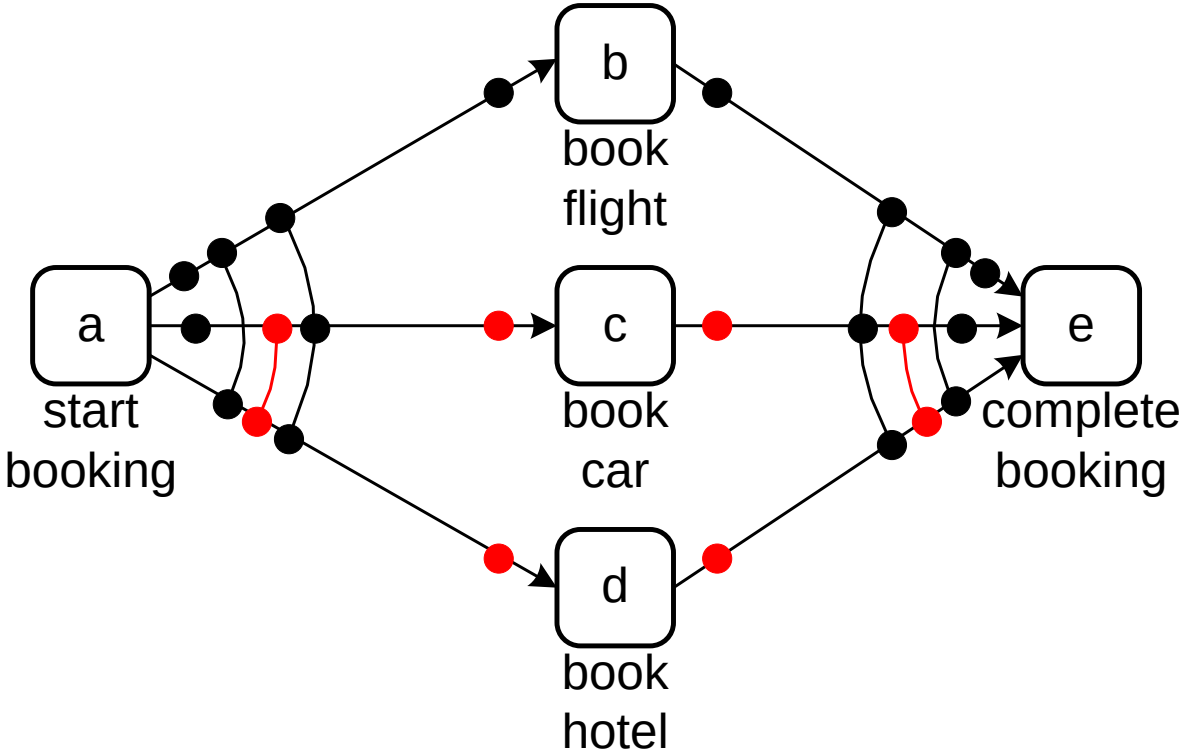


Twelve valid binding sequences are possible



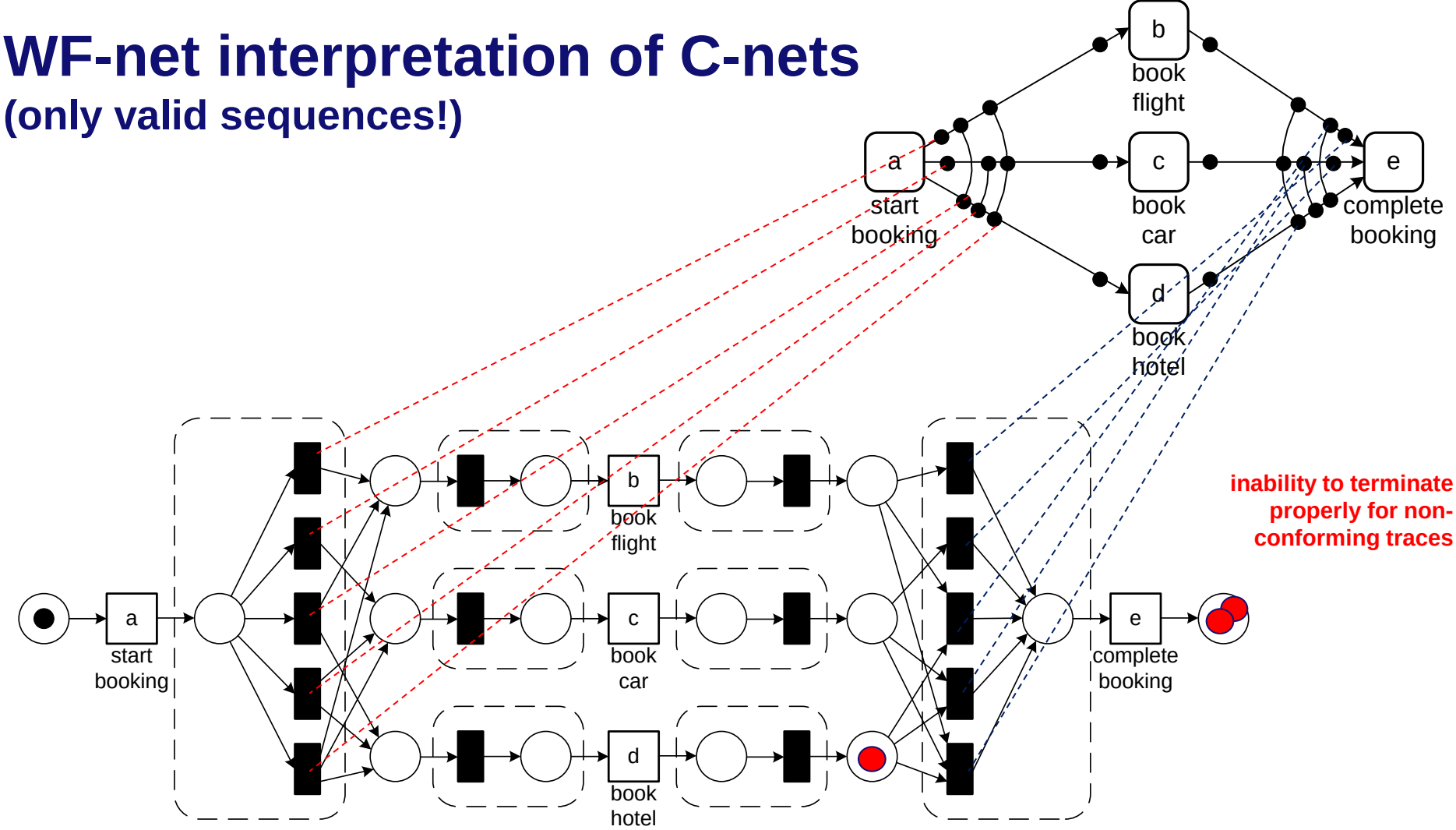
abe
ace
abde
adbe
acde
adce
abcde
abdce
acbde
acdbe
adbce
adcbe

Example: Valid binding sequence generating trace adce

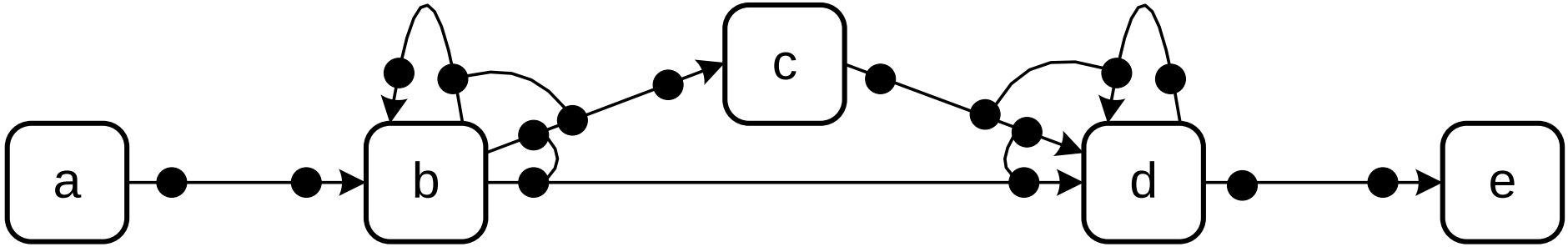


WF-net interpretation of C-nets

(only valid sequences!)



C-nets are more expressive
(due to declarative semantics)

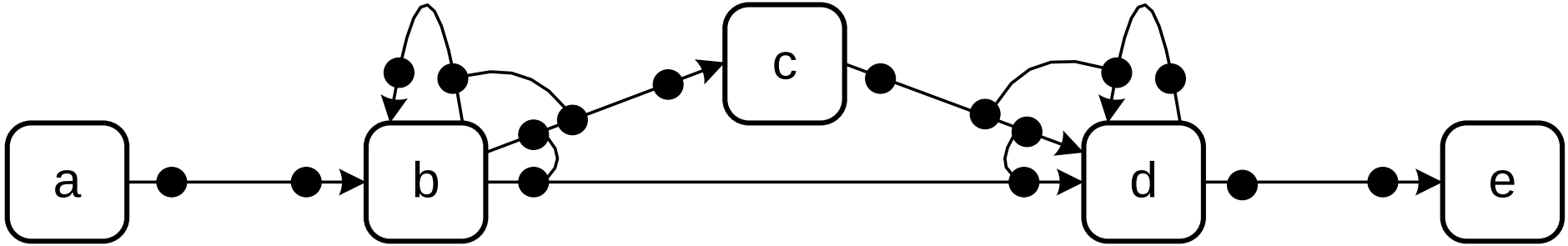


There is no Petri net that has a set of *full* firing sequences corresponding *exactly* to the valid binding sequences of this C-net.

Note that b, c, and d occur the same (unbounded) number of times.

TU/e

Question



Which trace does not correspond to a valid binding sequence?

a) abcde

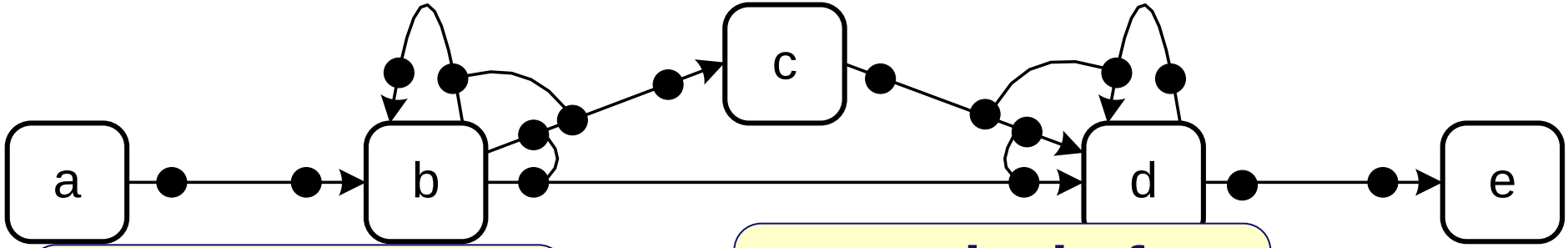
b) abcdcbde

c) abcbcdde

d) abbbbccccddddde

TU/e

Trace abcdcbde does not correspond to a valid binding sequence



after *d* still a *b*

second *c* before second *b*

a) abcde

b) abcdcbde

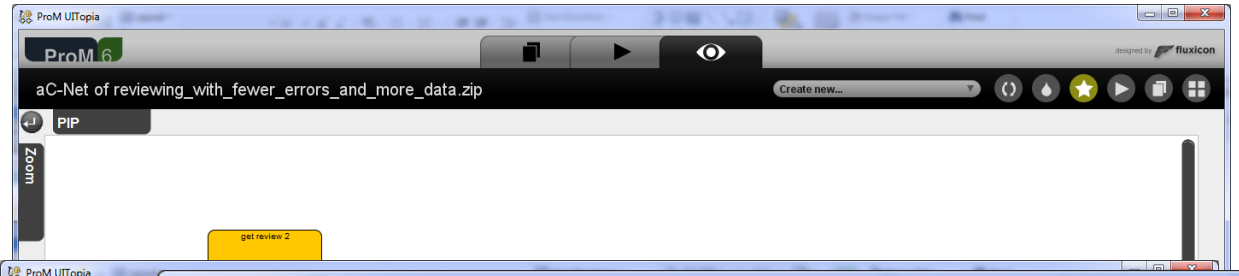
c) abcbcdde

d) abbbbccccddddde

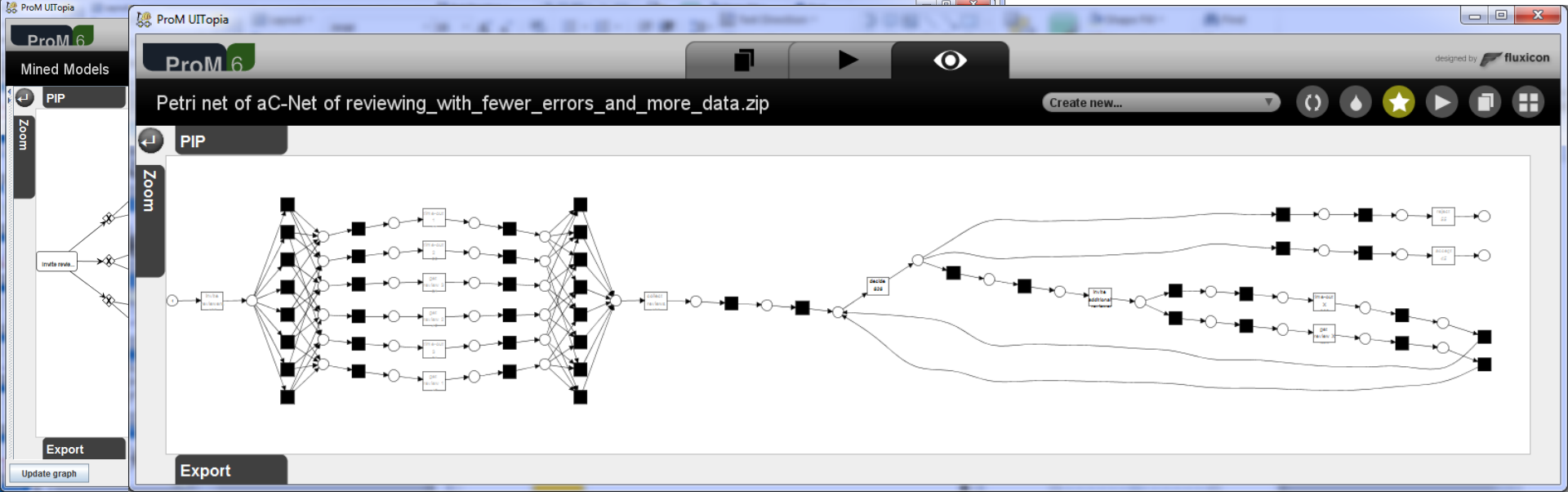
TU/e

Many plug-ins in ProM use C-nets

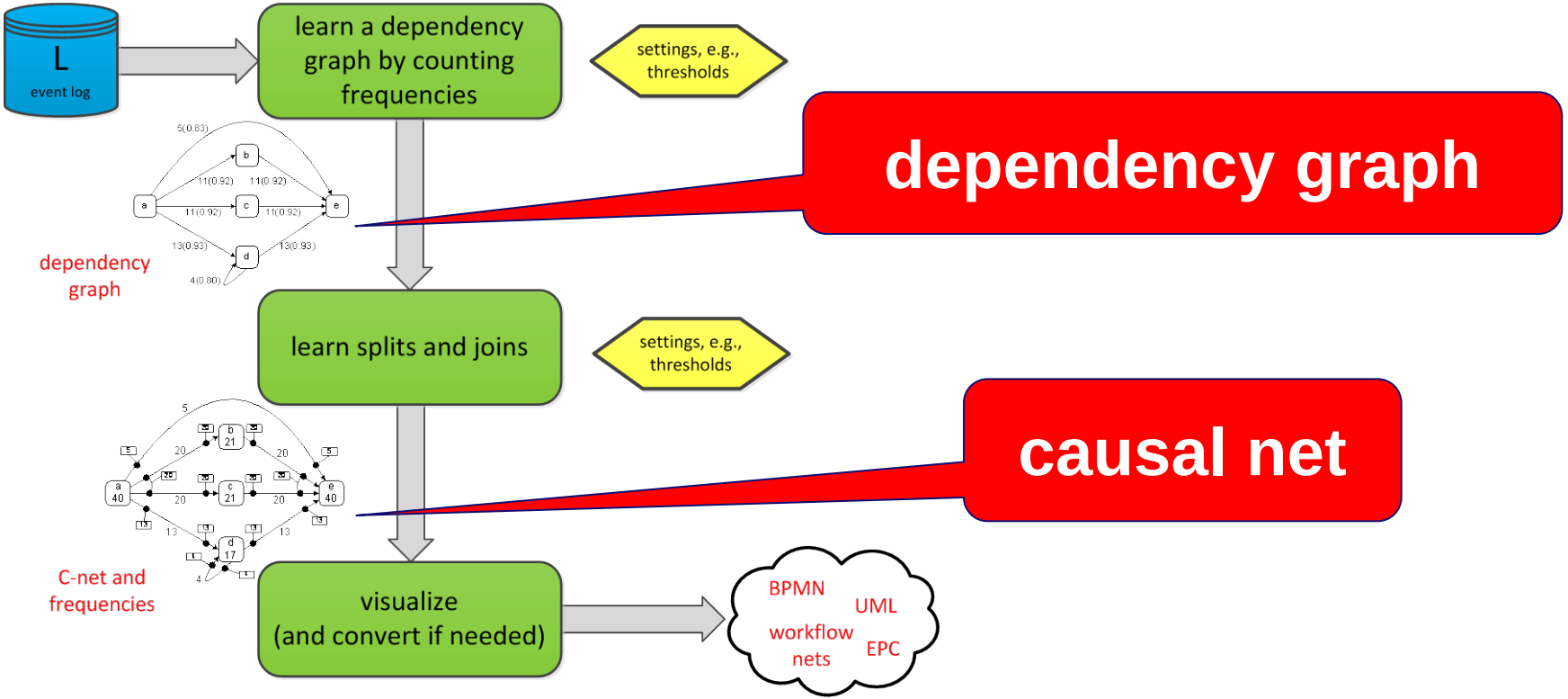
(or variants of C-nets like flexible models and heuristic nets)



- C-net discovery
- C-net conformance checking
- C-net performance analysis
- C-net conversions
- …



Next: Heuristic Mining



TU/e

Part I: Preliminaries

- Chapter 1: Introduction
- Chapter 2: Process Modeling and Analysis
- Chapter 3: [illegible]

Part II: From Event Logs to Process Models

- Chapter 4: Getting the Data
- Chapter 5: Process Discovery: An Introduction
- Chapter 6: Advanced Process Discovery [illegible]

Part III: Beyond Process Discovery

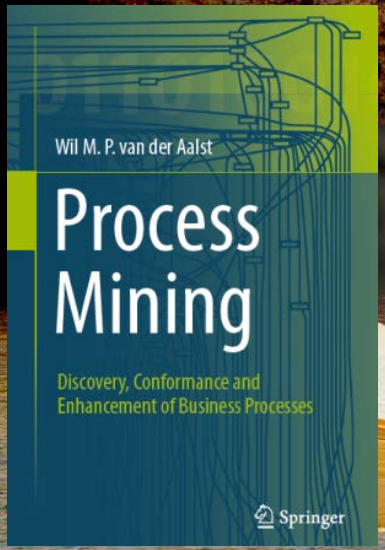
- Chapter 7: Conformance Checking
- Chapter 8: Mining Additional Perspectives
- Chapter 9: Operational Support

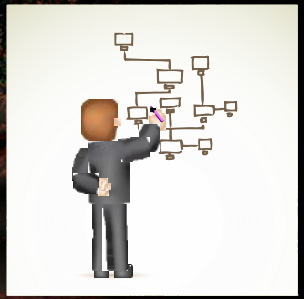
Part IV: Putting Process Mining to Work

- 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