Introduction to Data Science (IDS) course

Process Mining - Instruction

Lecture 13

IDS-L13-I

Chair of Process and Data Science

RWTH AACHEN UNIVERSITY

Process Discovery

Process Discovery

In the lecture you have seen the key concepts of Process Discovery:

- Process data often comes in form of events (event logs), recorded with a timestamp, case ID, and activity of the event
- Doing Process Discovery means creating a model representing the behavior in the event data
- Many formalisms for models (Petri nets, process trees, others)

Process Discovery

How do people create process models without Process Mining?

By hand!

Process models are often designed by hand. Usually, these models are drawn by experts of a certain process and reflect how thing *should* go in reality (normative process model).

Process Discovery

Process Discovery is hard even when a human and a computer work together. Two (of many) reasons:

- **As it is also the case in Machine Learning, you only get to observe a part of reality. Your model has to balance between fitting both the data you have and also unseen process instances.**

Process Discovery

Process Discovery is hard even when a human and a computer work together. Two (of many) reasons:

- **A problem more specific to Process Mining: you only get to observe positive behavior. Event logs cannot contain negative examples, and you do not have examples of what *cannot happen*.**

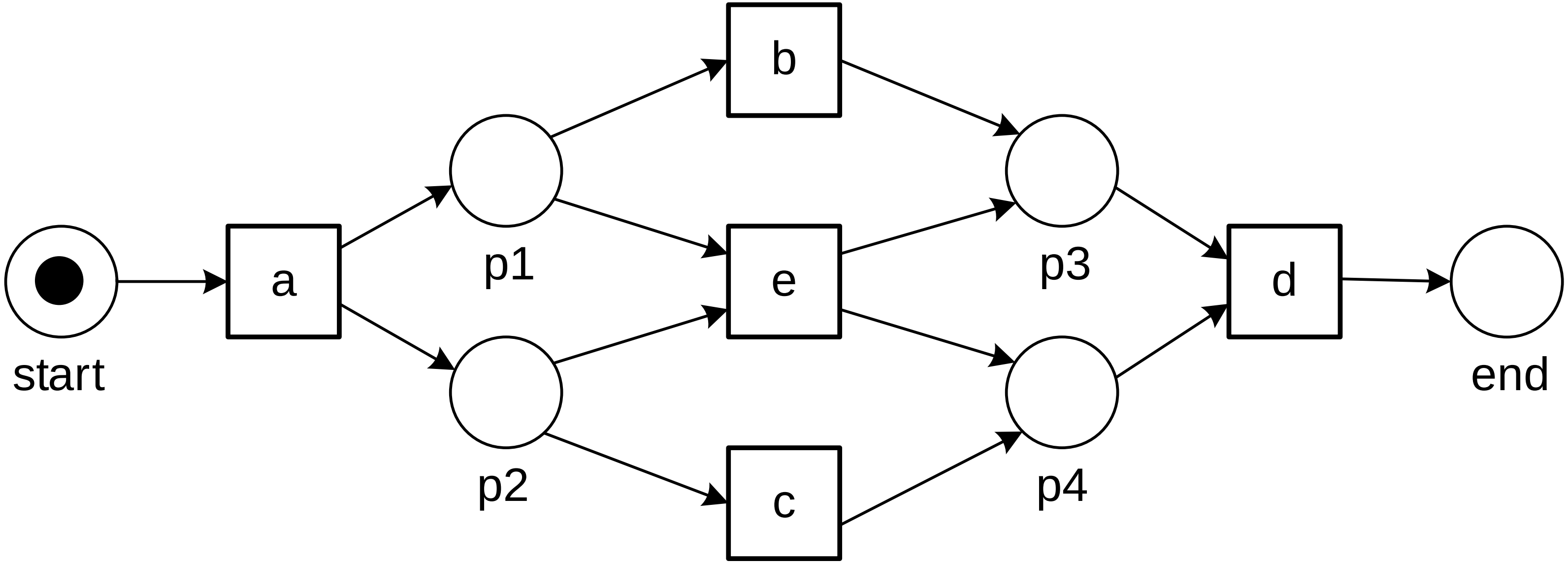
Chair of Process and Data Science

Process Models: Petri Nets

Recall from the lecture:

- Petri nets are composed by directed **arcs**, **places**, and **transitions**.
- Places contain **tokens**. A specific configuration of tokens in a net is called a **marking**.
- Transition can **fire**, consuming a token from the input places, putting a token in the output places, and "producing" and event.
- **Initial** and **final markings** define the start and end of a trace.

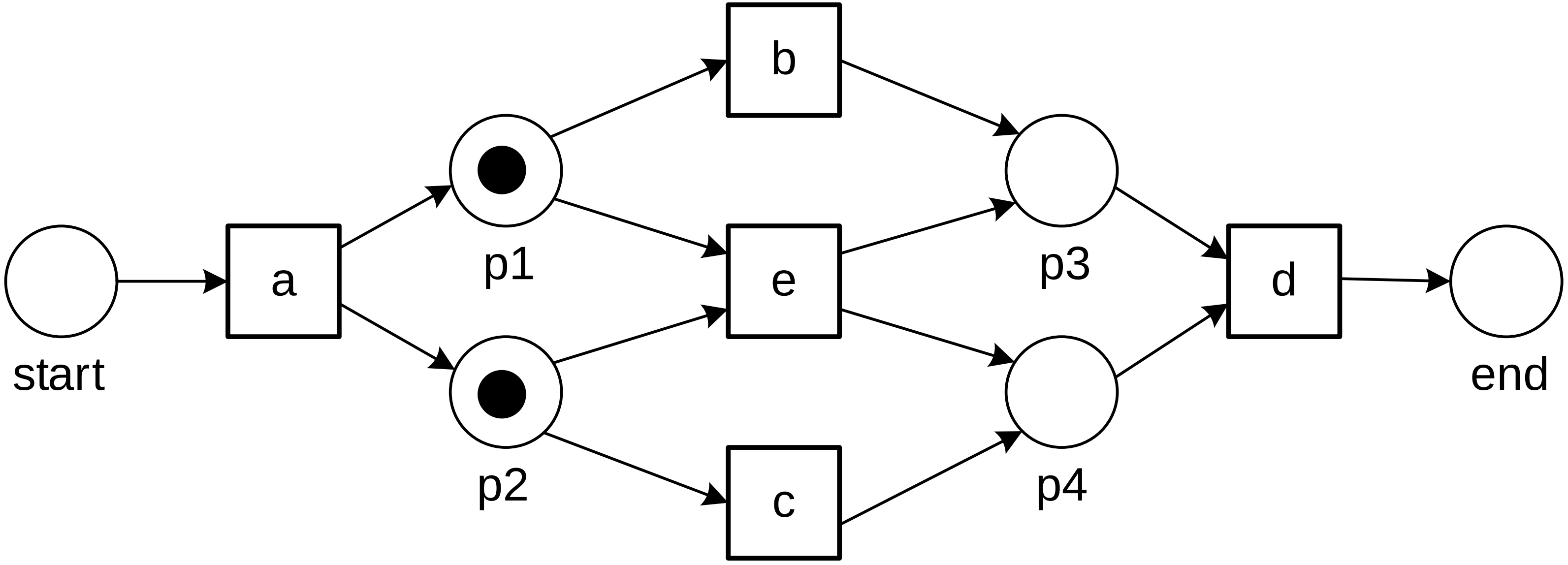
“Playing out”



Initial marking

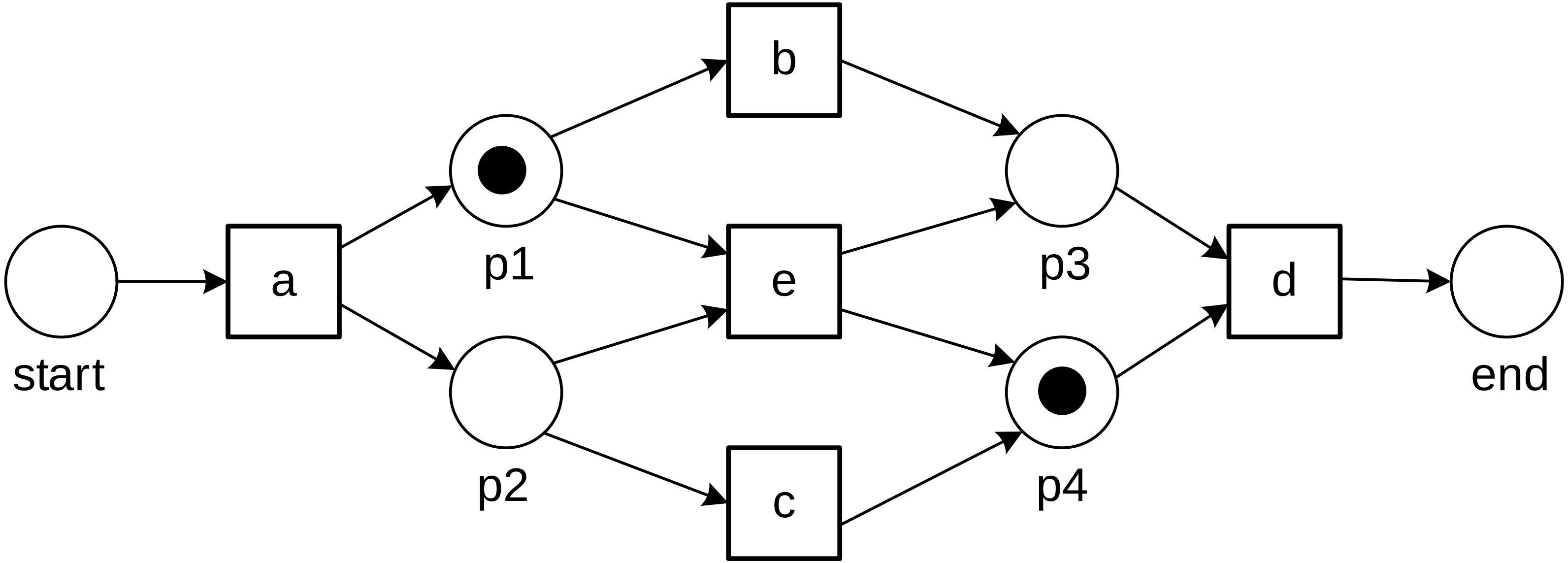
<>

“Playing out”



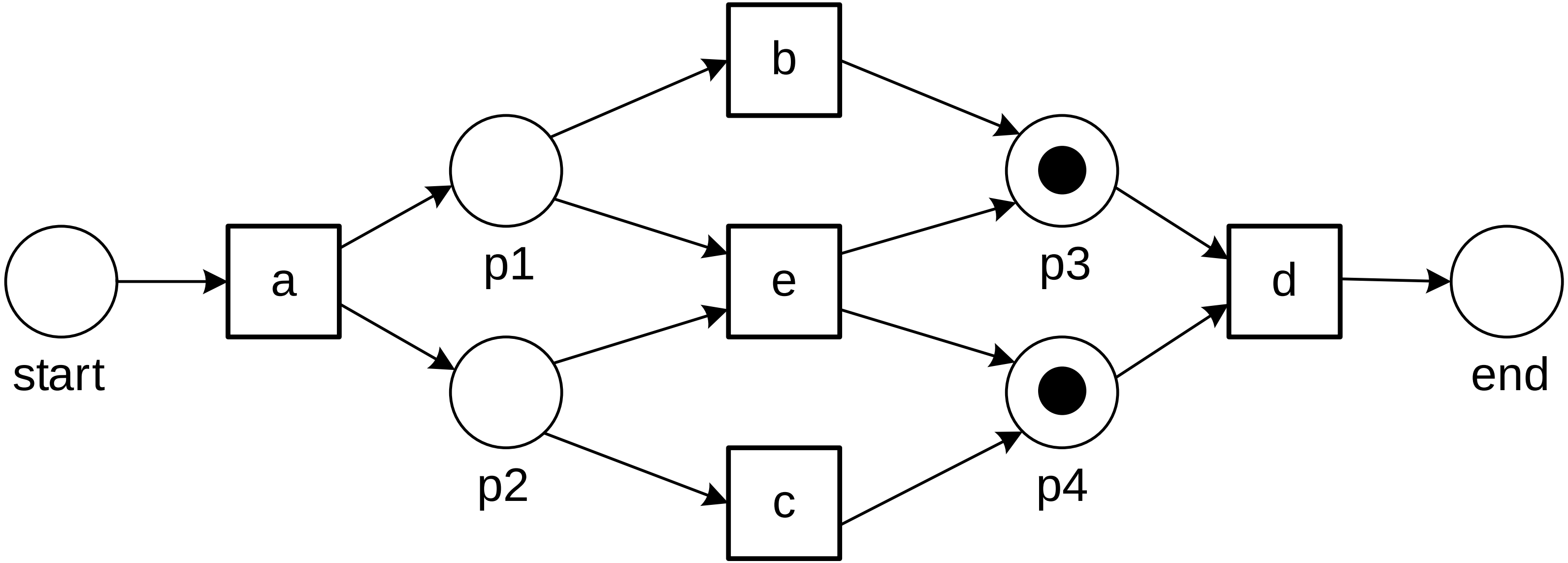
<a>

“Playing out”



<a,c>

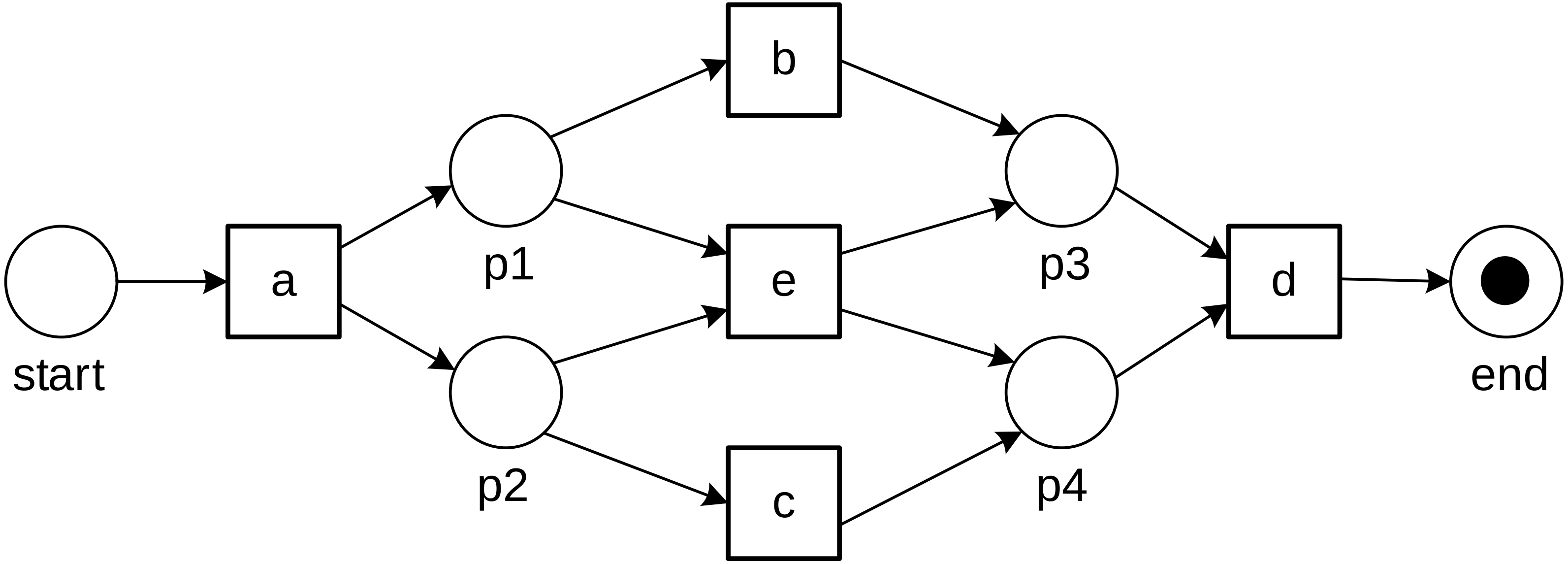
“Playing out”



<a,c,b>

Chair of Process and Data Science

“Playing out”



Final marking

<a,c,b,d>

Process Discovery

Given some traces, can you come up with a Petri net that can replay all of them?

<a,b,d,e,f,h>

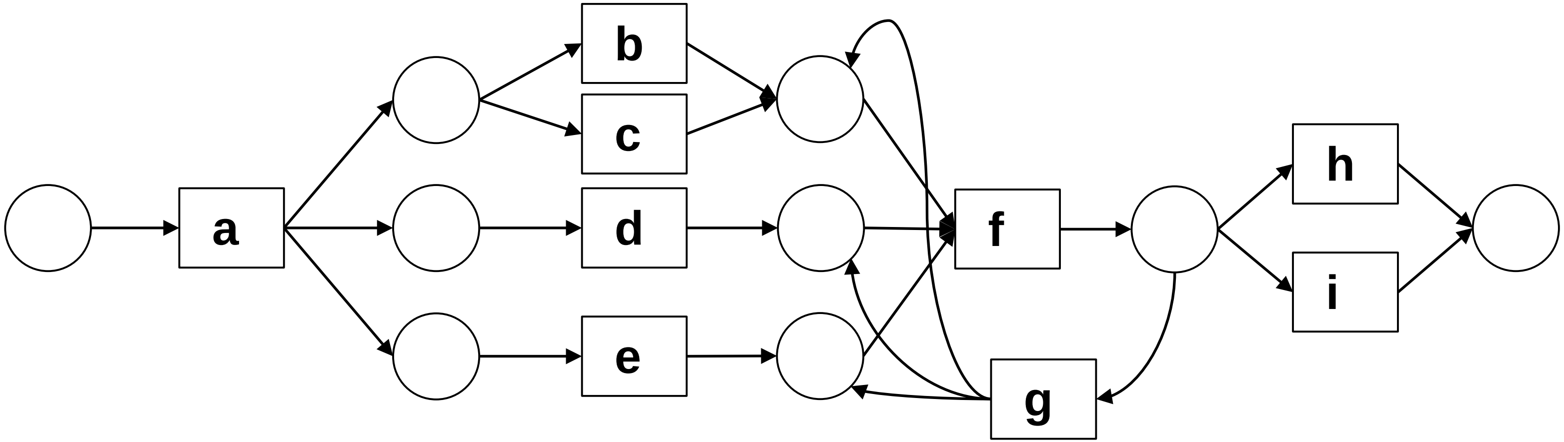
<a,e,c,d,f,g,f,i>

<a,d,e,b,f,g,f,g,f,h>

Chair of Process and Data Science

Process Discovery: solution

<a,b,d,e,f,h>   <a,e,c,d,f,g,f,i>   <a,d,e,b,f,g,f,g,f,h>



Chair of Process and Data Science

Directly-follows graph based on event log

3x a b c d

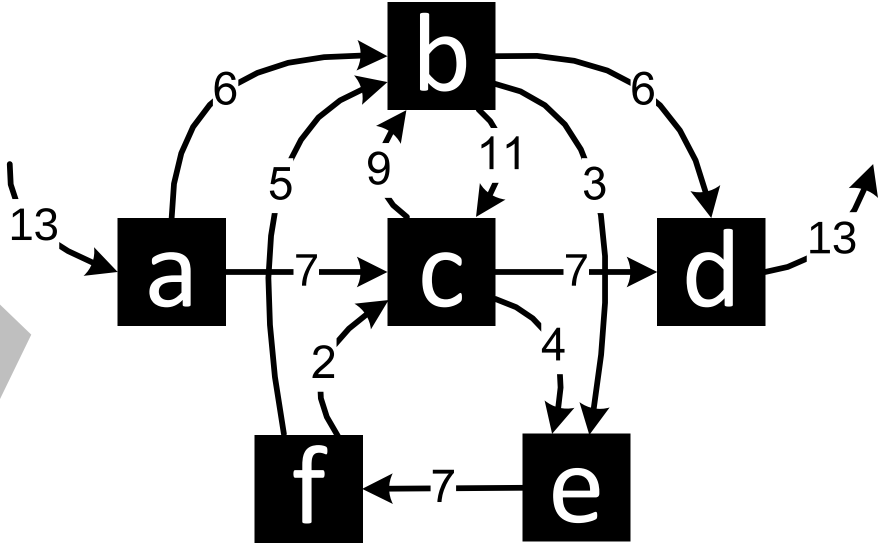
4x a c b d

2x a b c e f b c d

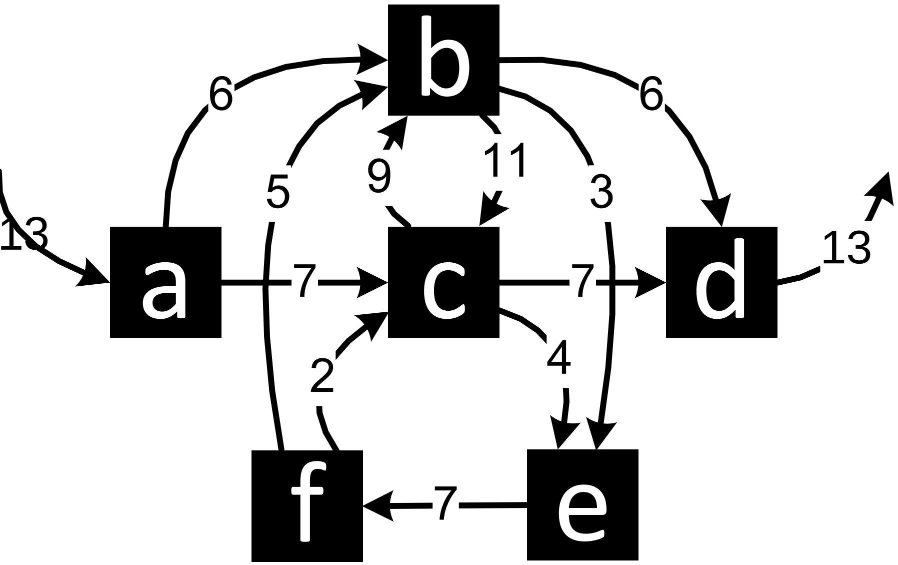
2x a c b e f b c d

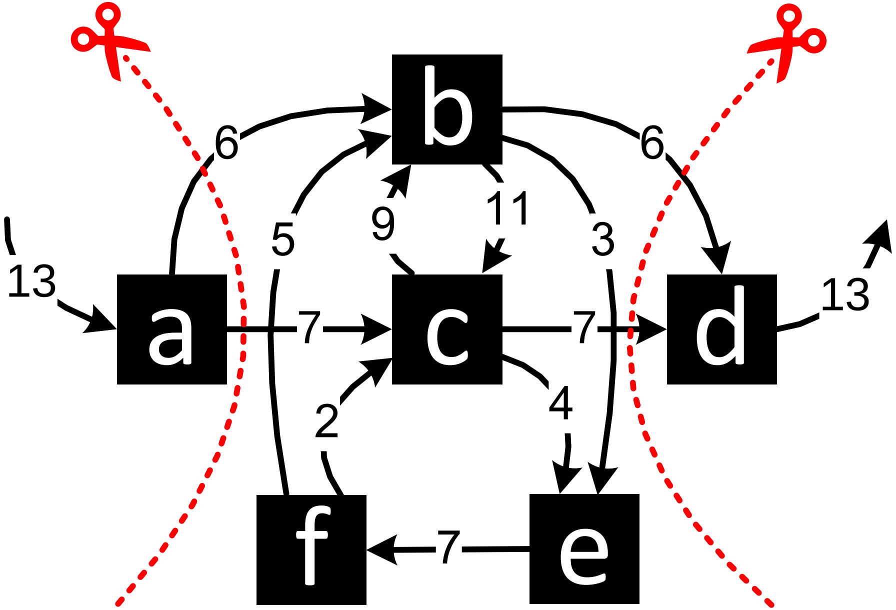
1x a b c e f c b d

1x a c b e f b c e f c b d



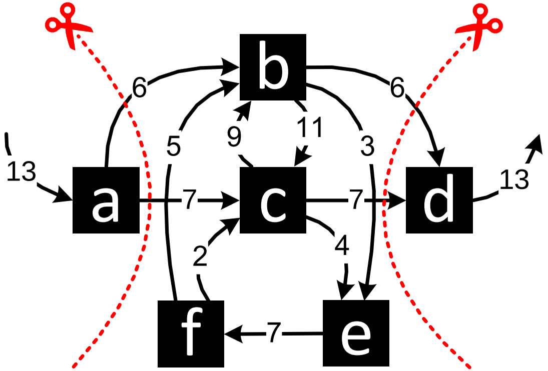
Sequence cut



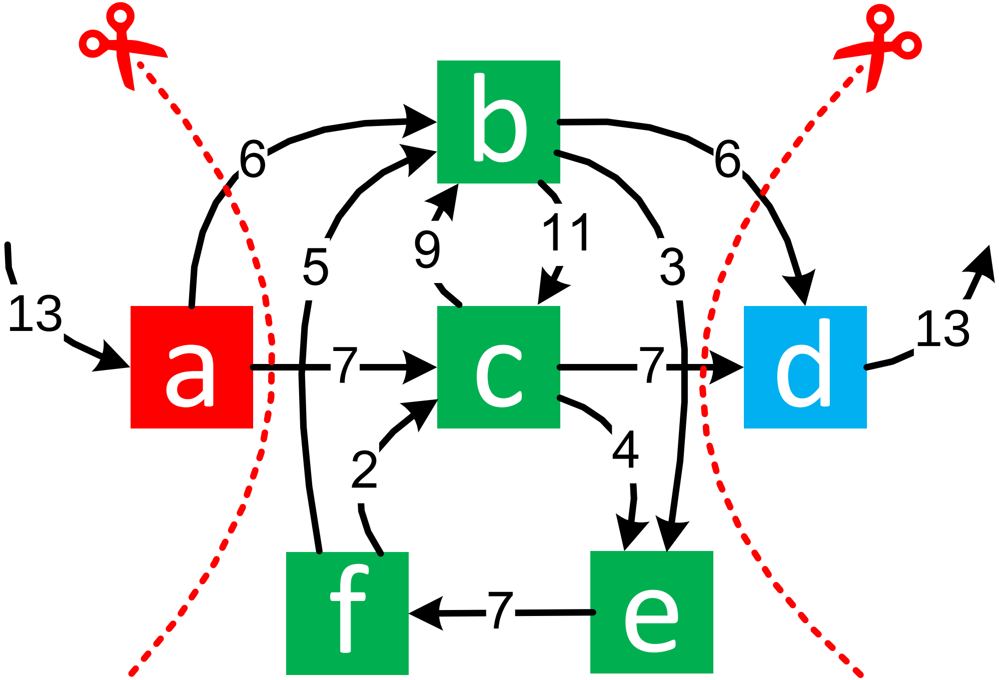


Chair of Process and Data Science

Partition activities based on sequence cut

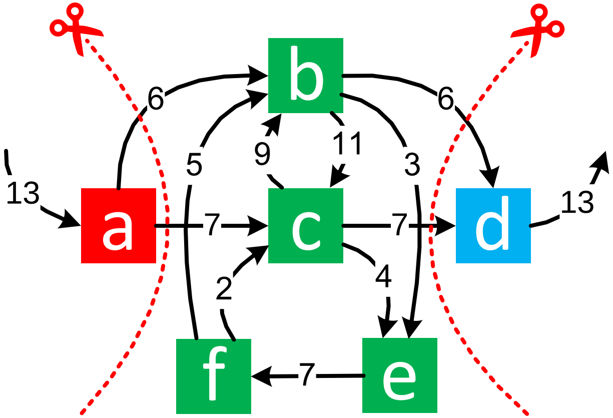


{a} , {b,c,e,f} , {d}



Chair of Process and Data Science

Partition events based on sequence cut



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

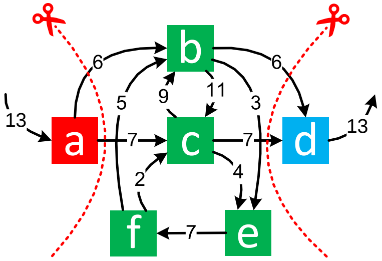
1x a b c e f c b d

1x a c b e f b c e f c b d

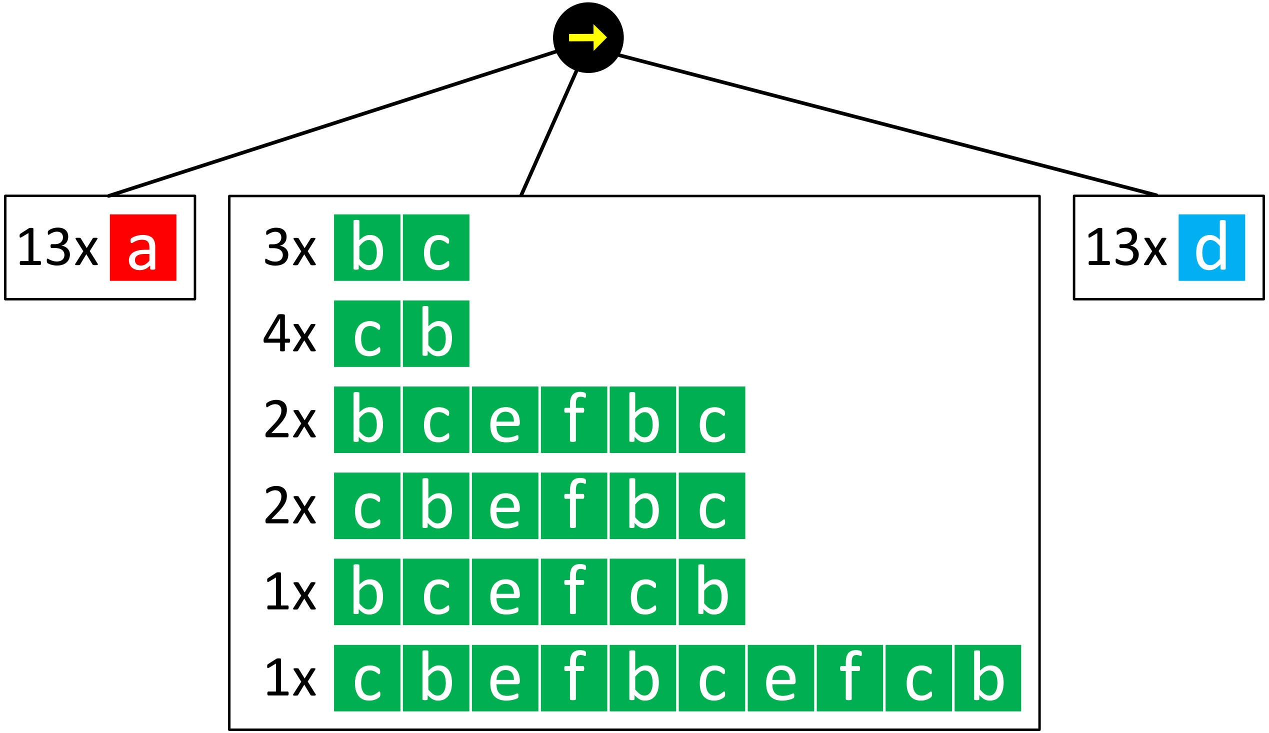
{a} , {b,c,e,f} , {d}

Chair of Process and Data Science

Partition events based on sequence cut



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

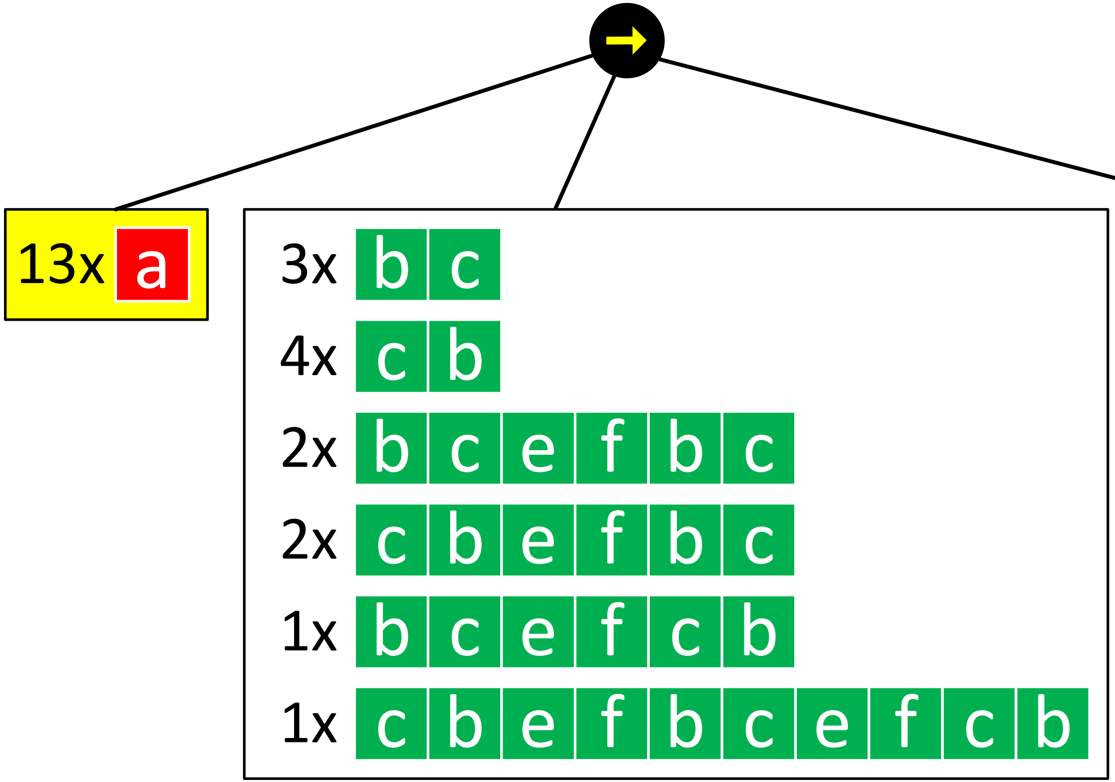


13x a

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

13x d

Handle base cases



13x a

3x b c

4x c b

2x b c e f b c

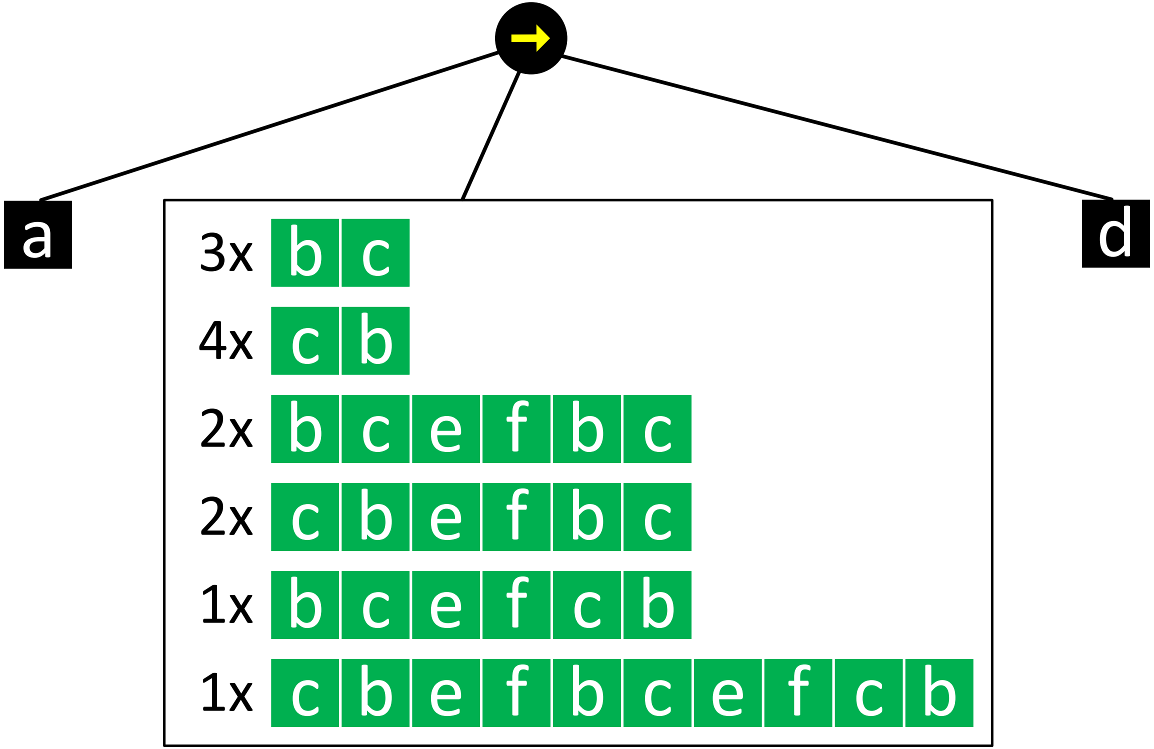
2x c b e f b c

1x b c e f c b

1x c b e f b c e f c b

13x d

Recurse on non-base cases



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

Directly-follows graph based on sublog

3x b c

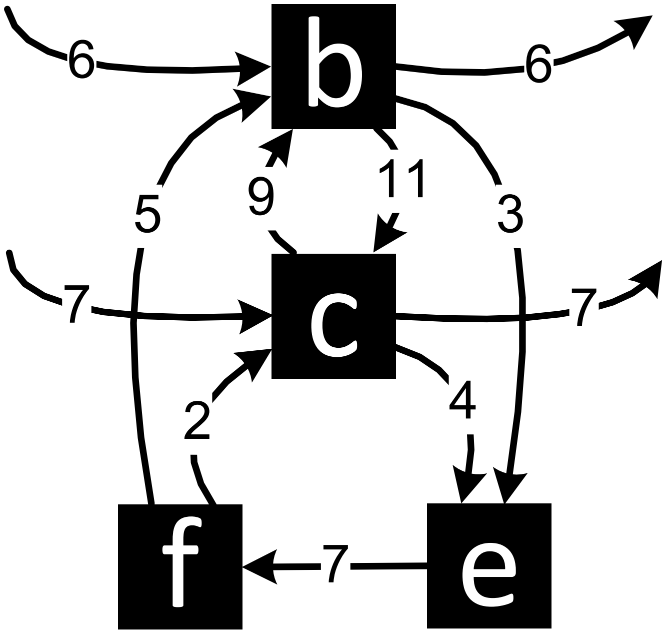
4x c b

2x b c e f b c

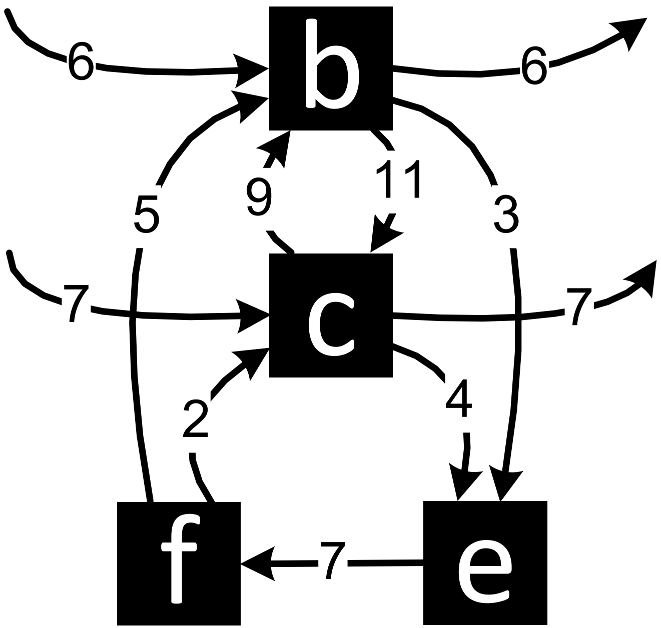
2x c b e f b c

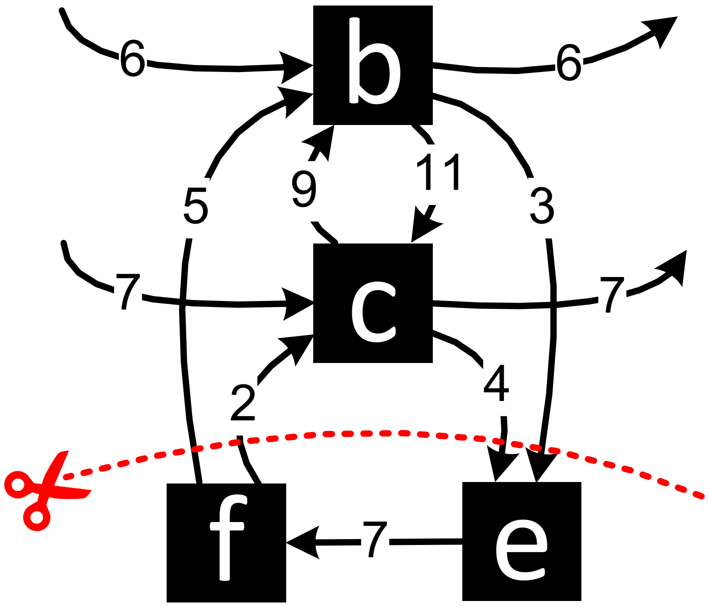
1x b c e f c b

1x c b e f b c e f c b



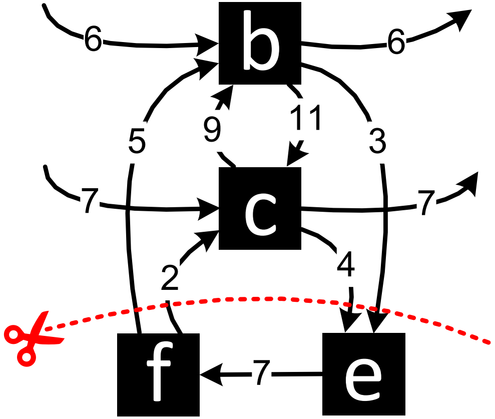
Loop cut



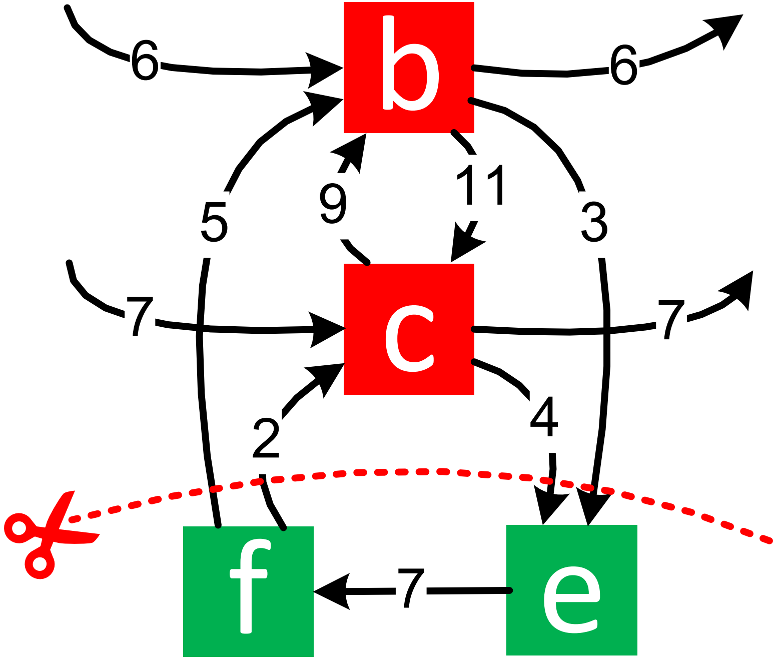


Chair of Process and Data Science

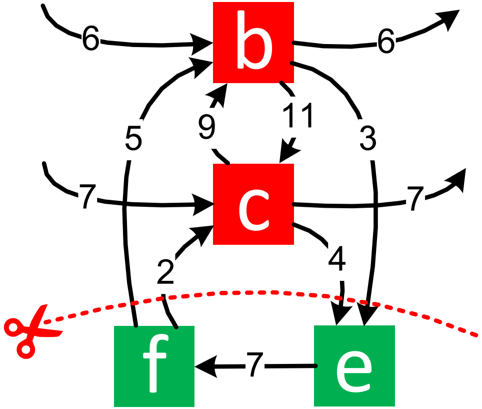
Partition activities based on loop cut



{b,c} , {e,f}



Partition events based on loop cut



3x b c

4x c b

2x b c e f b c

2x c b e f b c

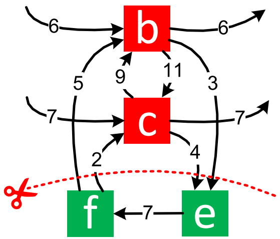
1x b c e f c b

1x c b e f b c e f c b

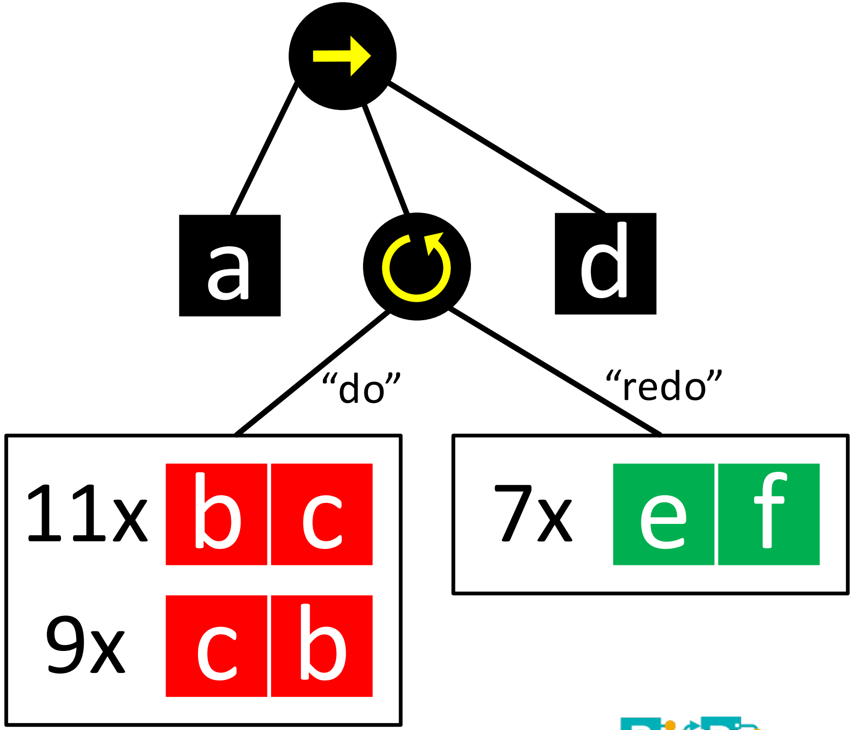
{b,c} ,
{e,f}

© PADS (use only with permission & acknowledgements)

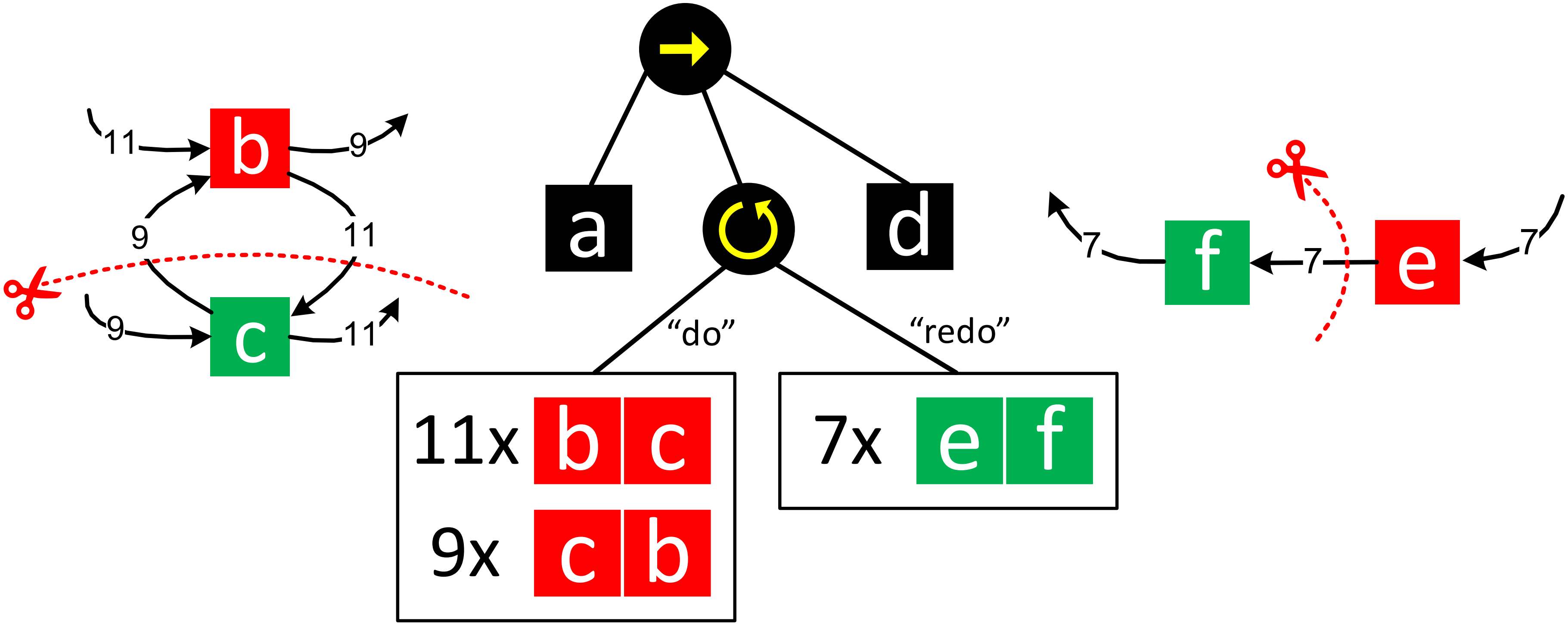
Partition events based on loop cut



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



Recurse on the two sublogs



"do"

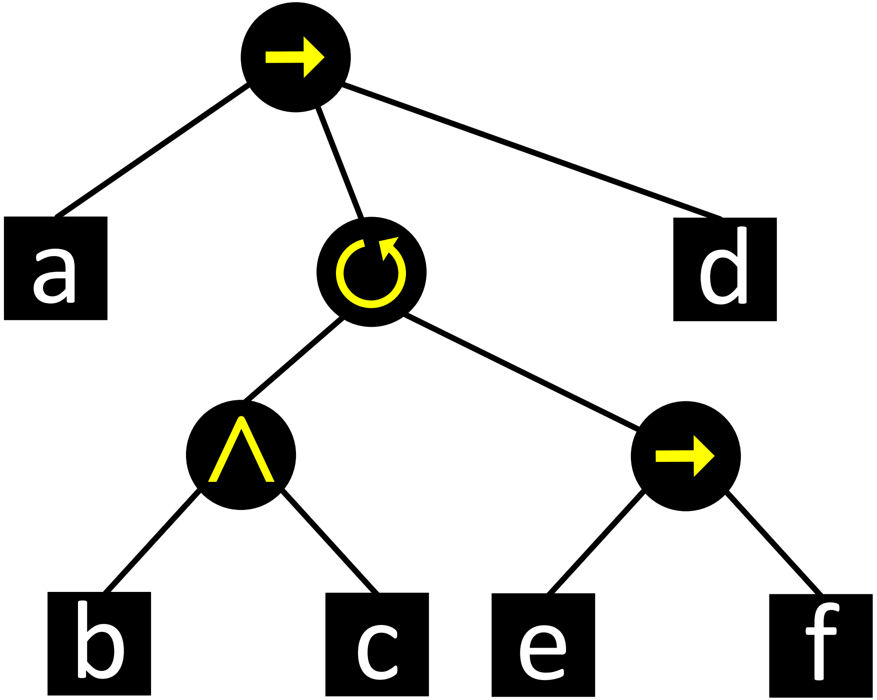
11x b c

9x c b

"redo"

7x e f

Final model



Your turn!

<a, d>

<a, d, e, d>

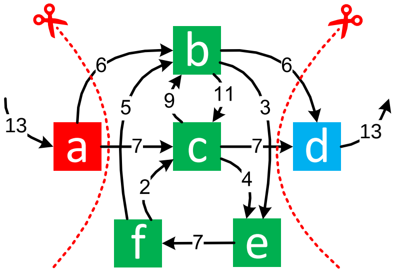
<b, c, d>

<b, c, d, e, d>

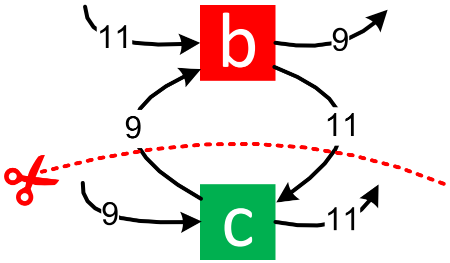
<c, b, d, e, d>

<c, b, d, e, d, e, d>

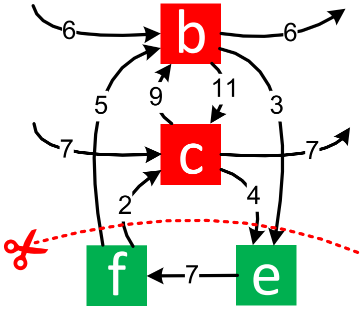
Sequence



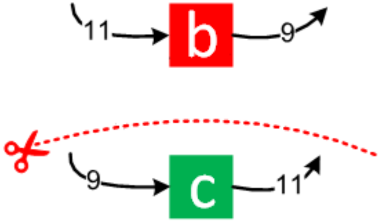
AND



Loop



XOR



Chair of Process and Data Science

Solution

Sequence cut: →({a, b, c}, {d, e})

XOR cut: →(×(a, {b, c}), {d, e})

AND cut: →(×(a, ∧(b, c)), {d, e})

Loop cut: →(×(a, ∧(b, c)), ⟲(d, e))