CS 4072 - Topics in CS
Process Mining

Lecture # 17

April 25, 2022

Spring 2022

FAST - NUCES, CFD Campus

Dr. Rabia Maqsood

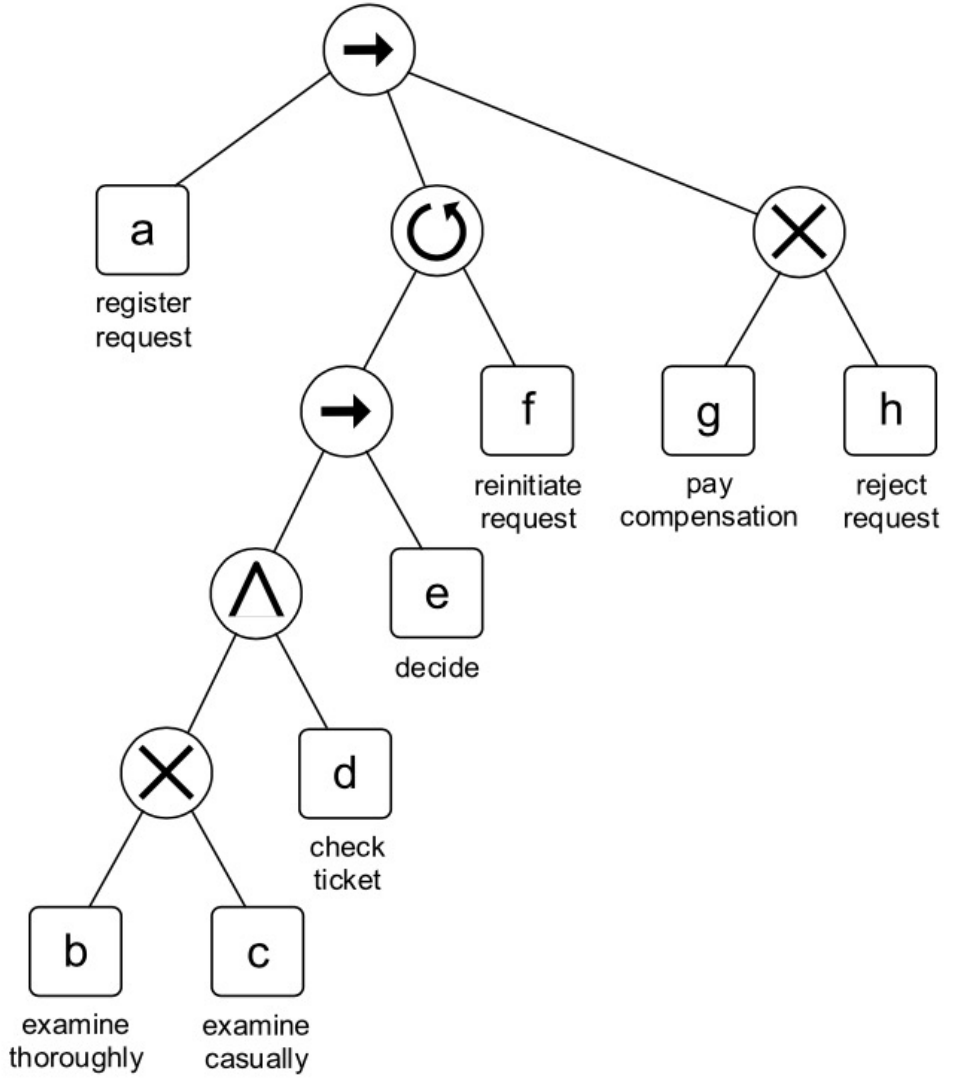
rabia.maqsood@nu.edu.pk

Today's Topics

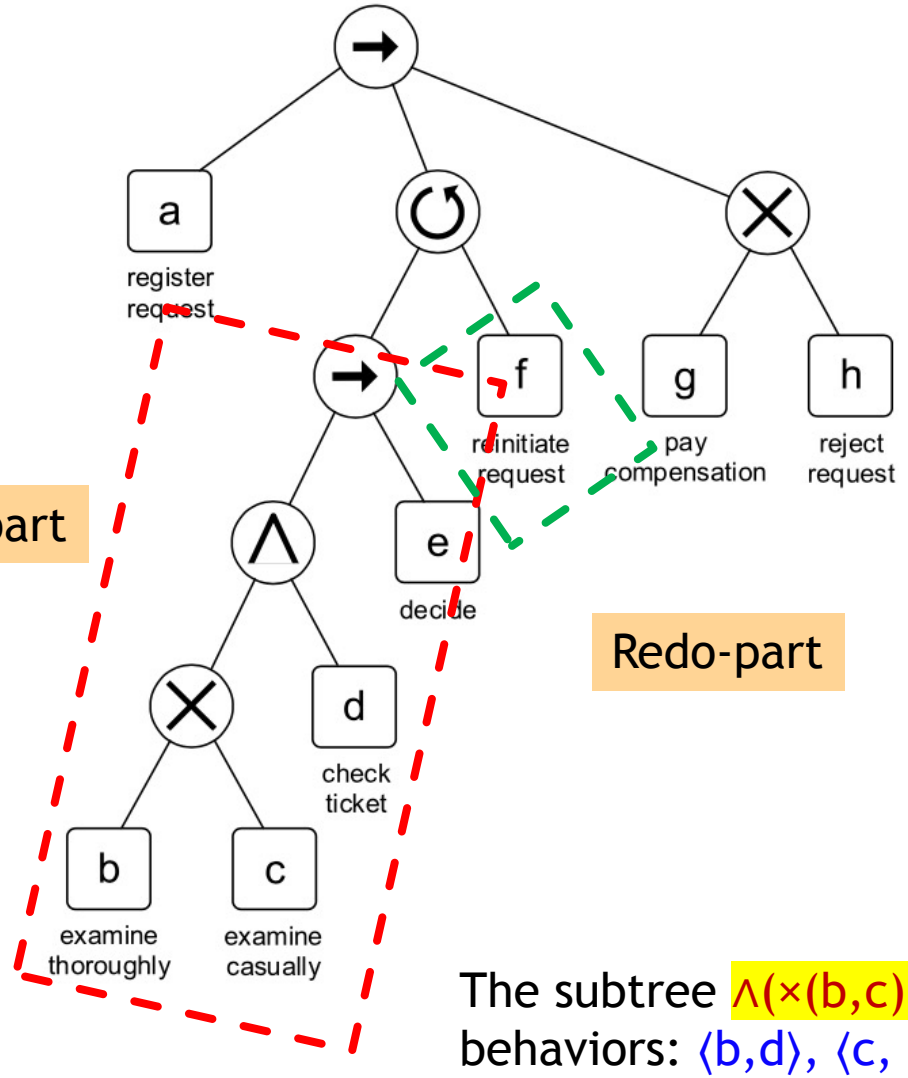
- Inductive Mining Algorithm

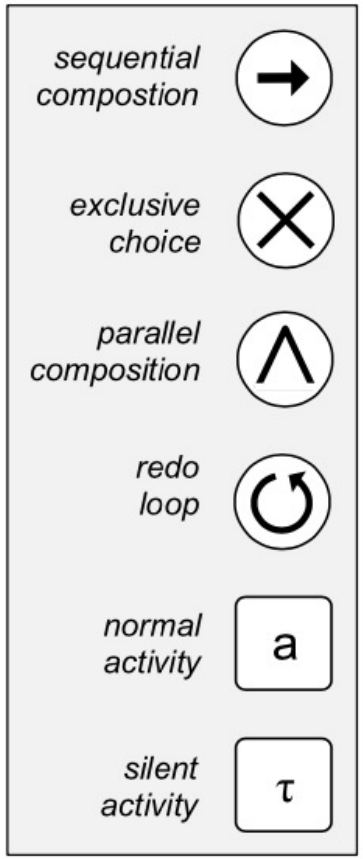
Process Trees

- A process tree is a hierarchical process model where the **(inner) nodes** are operators such as *sequence and choice* and the **leaves** are activities.
- A range of ***inductive process discovery*** techniques exists for process trees, which benefit from the fact that the representation ensures soundness.



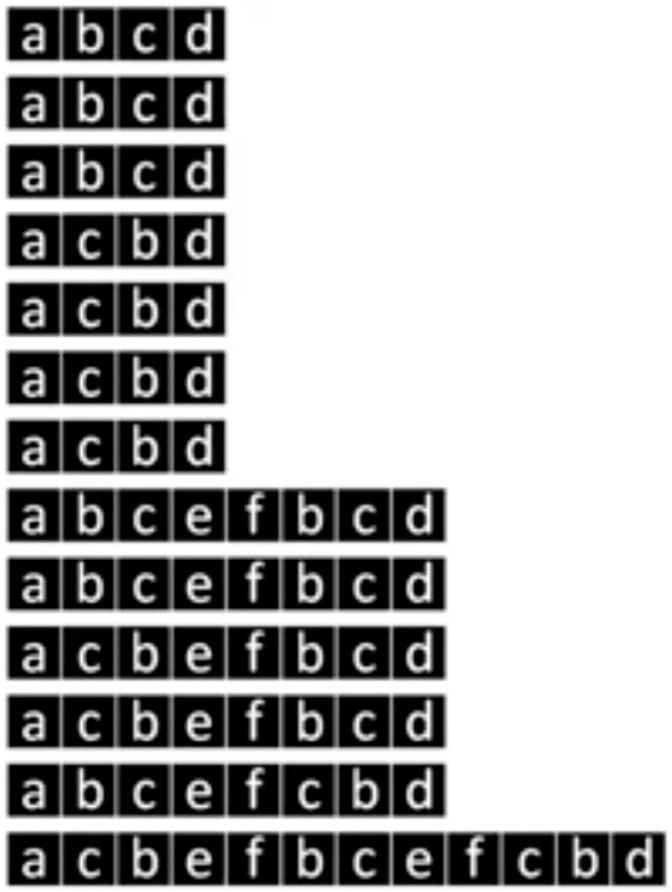
Process Trees: operator & activity nodes

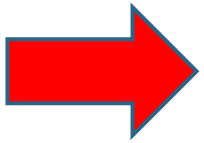


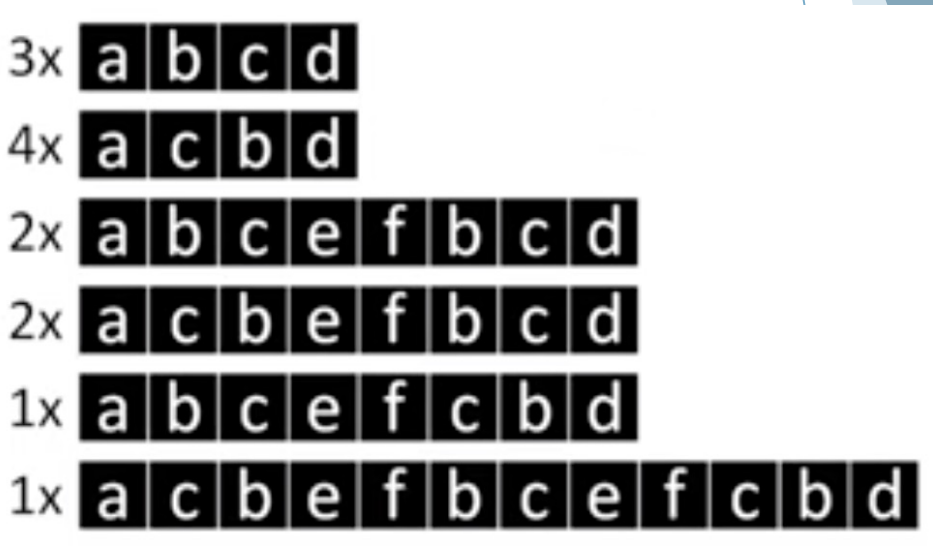


The subtree ∧(×(b,c),d) has four potential behaviors: ⟨b,d⟩, ⟨c, d⟩, ⟨d, b⟩, and ⟨d, c⟩.

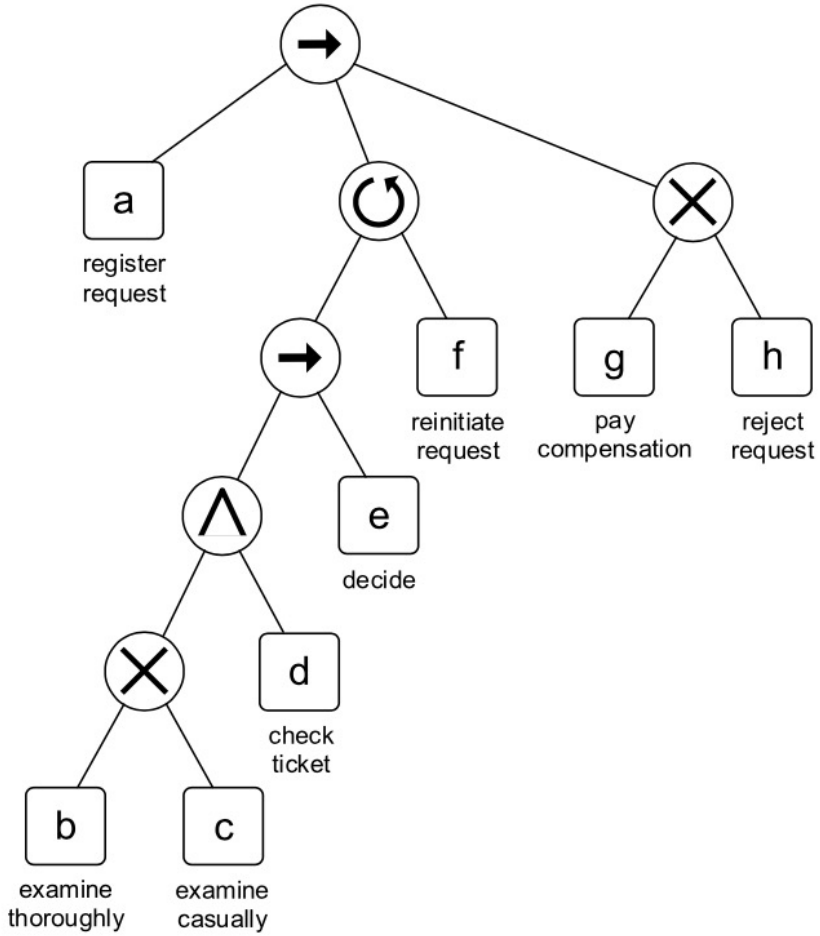
Input: Event log (simplified)

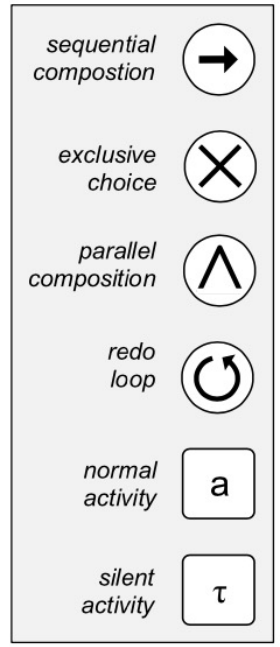




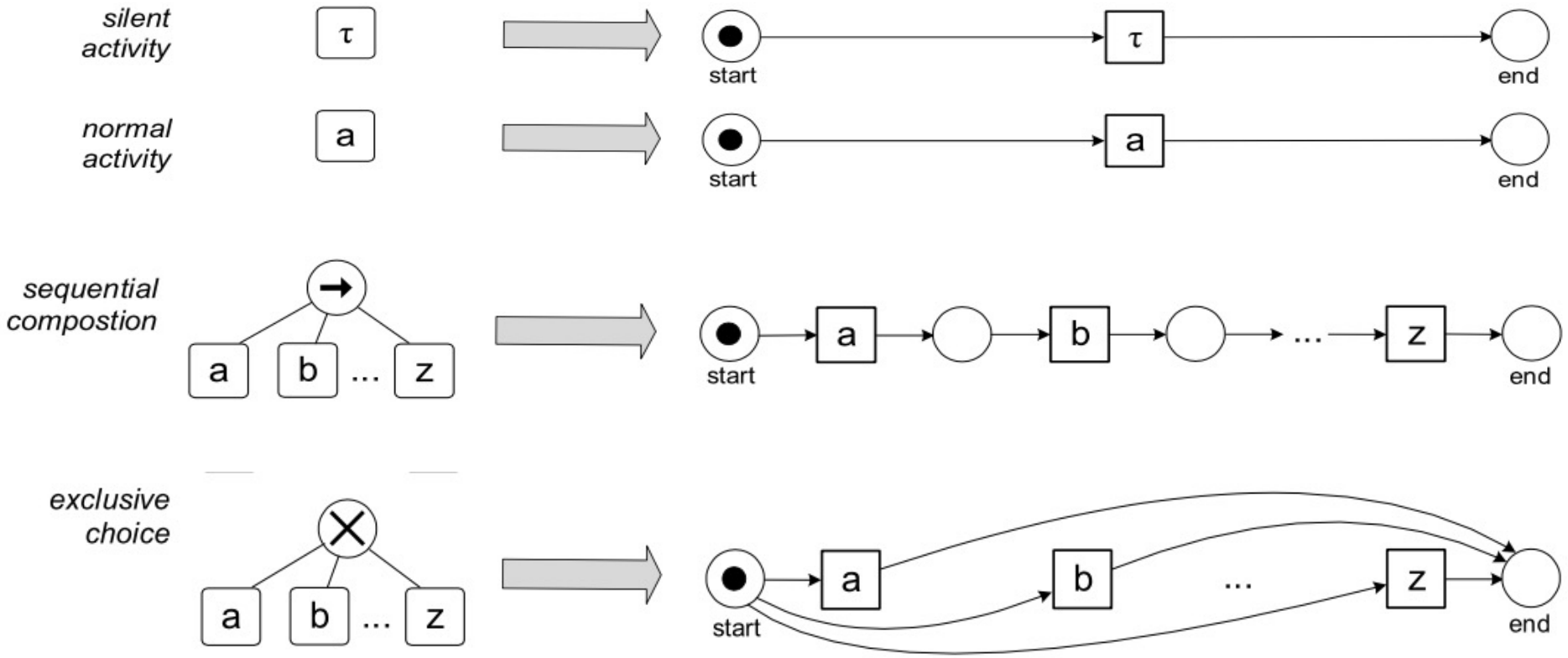


Output: Process Tree

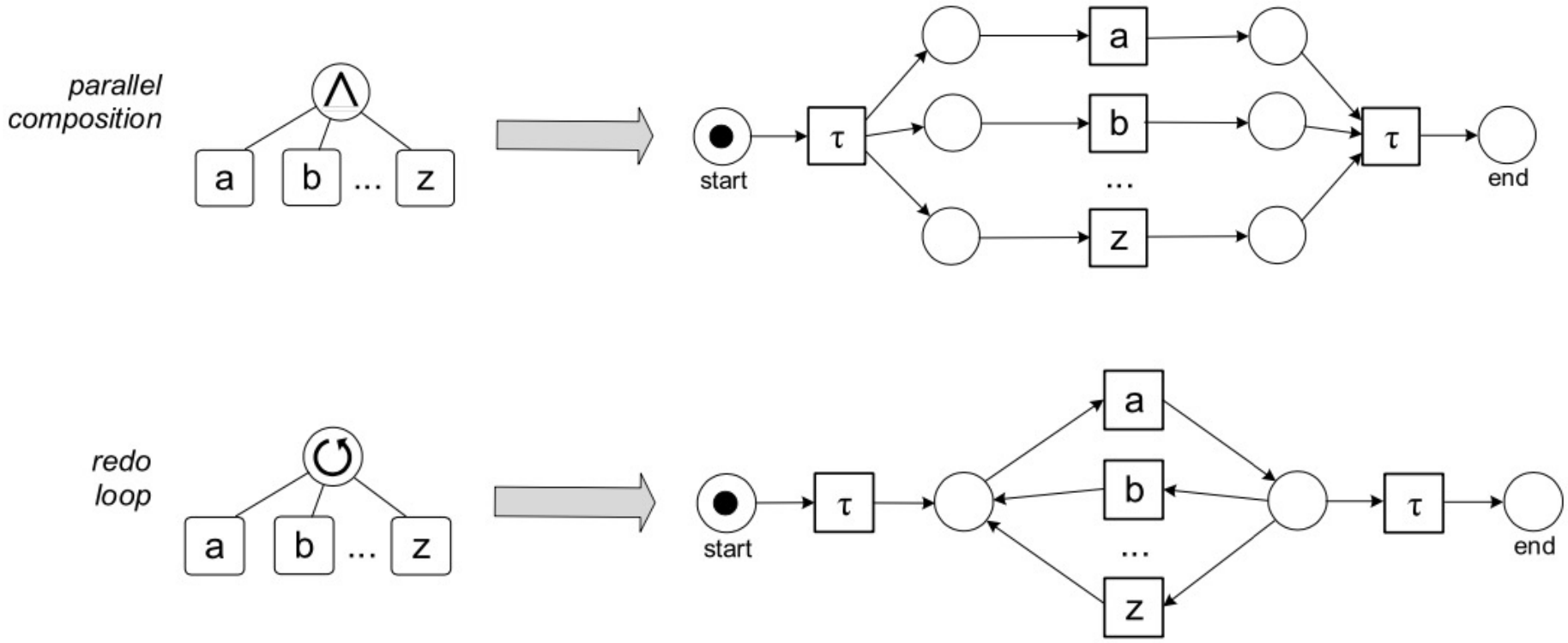




Output: Process Tree



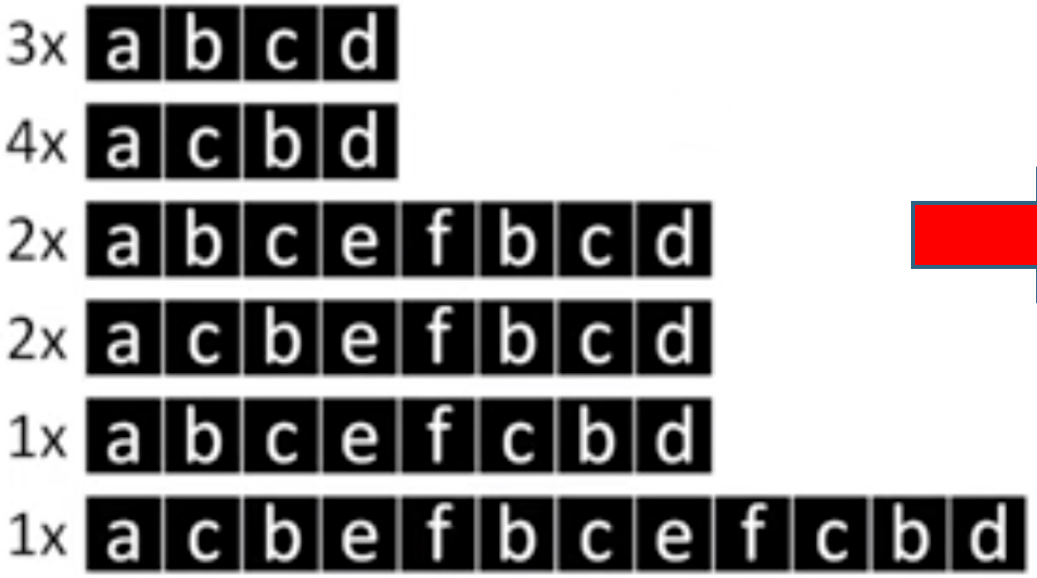
Output: Process Tree

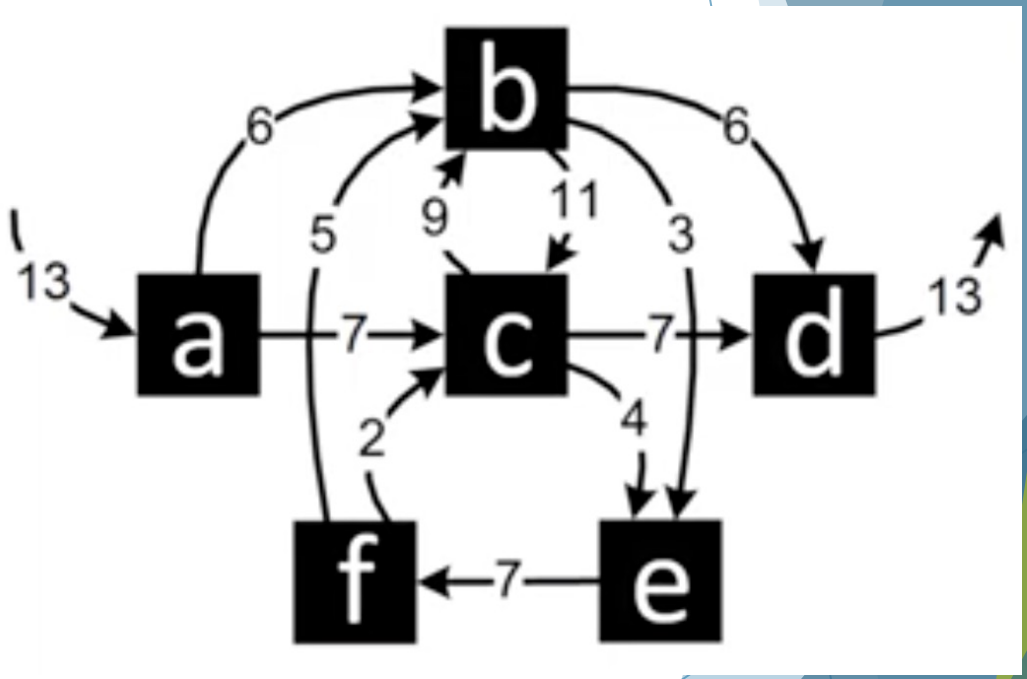


Inductive Miner Algorithm

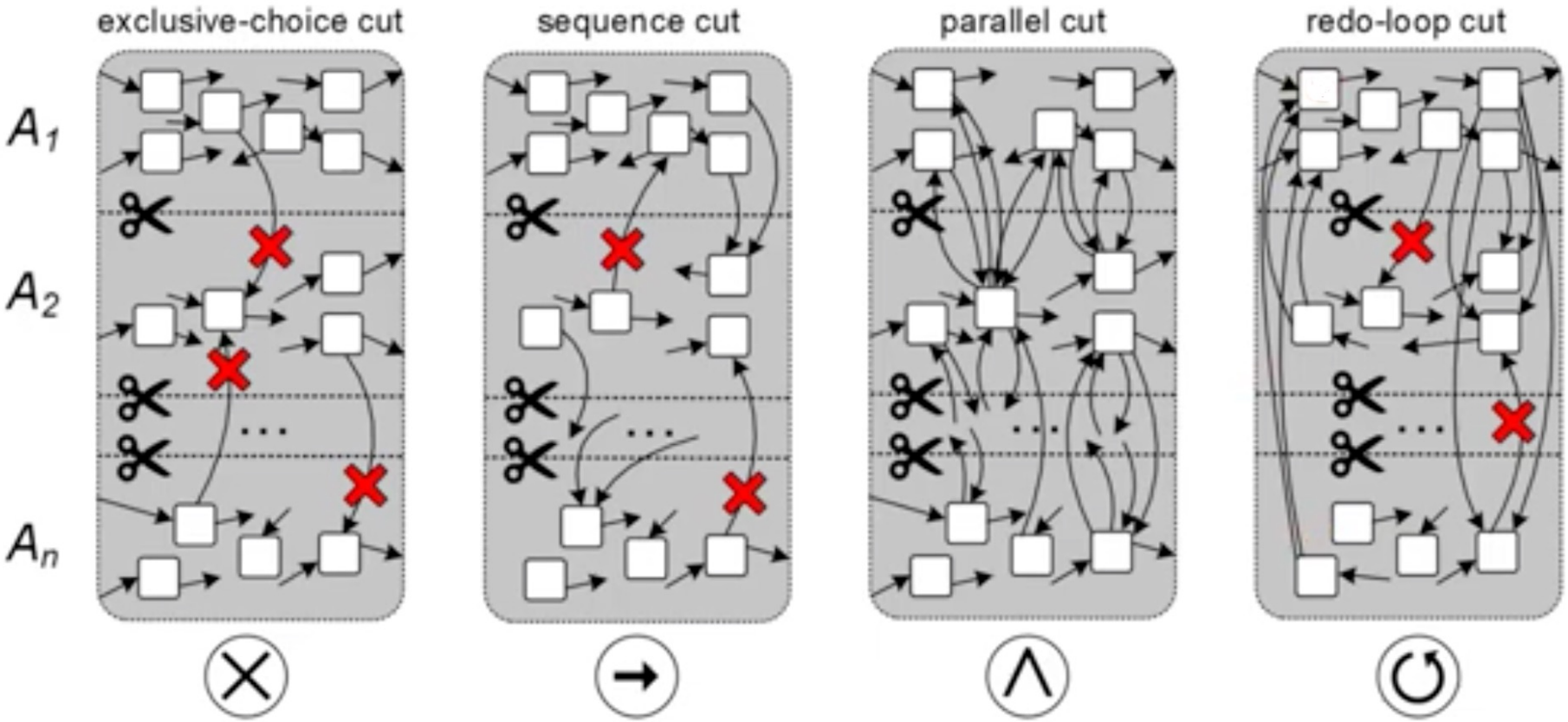
- Basic idea:
  1. Construct a directly-follows graph based on an event log
  2. Detect patterns in the directly-followed graph
     - Identify an appropriate cut that represents one of the four possible operator nodes in the process tree
  3. Divide the event log based on the operator identified in the Step 2
  4. Repeat Steps 2 & 3 until a sub-event log cannot be divided further

Directly-follows graph based on event log



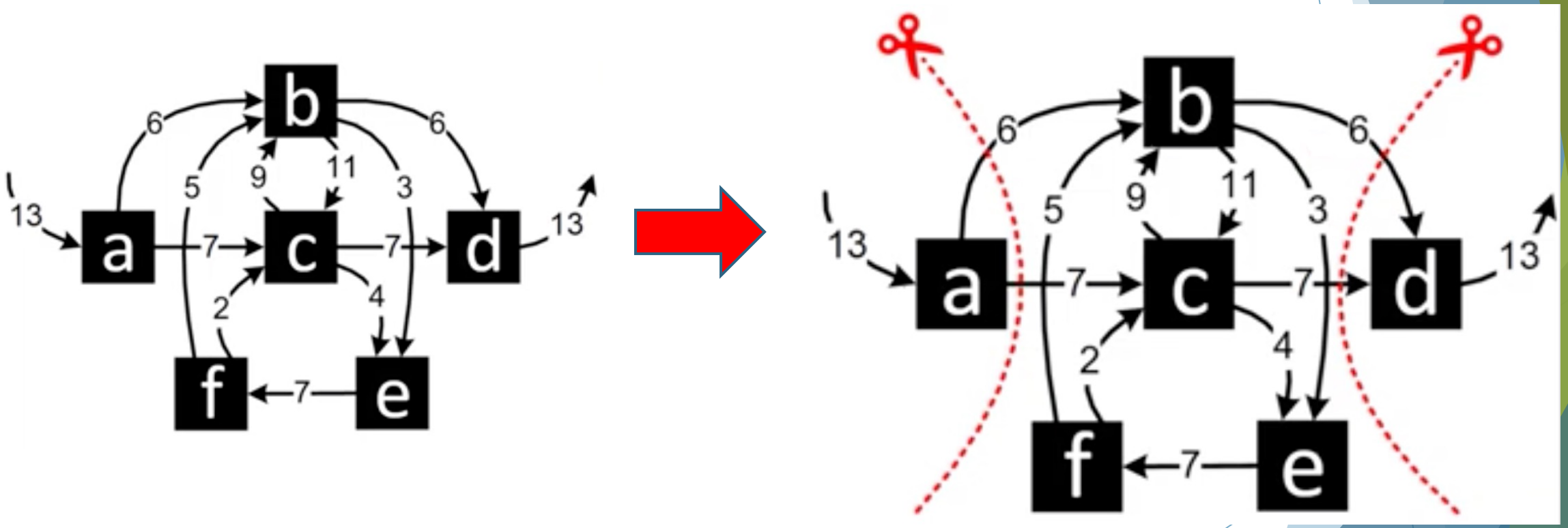


Four types of cuts

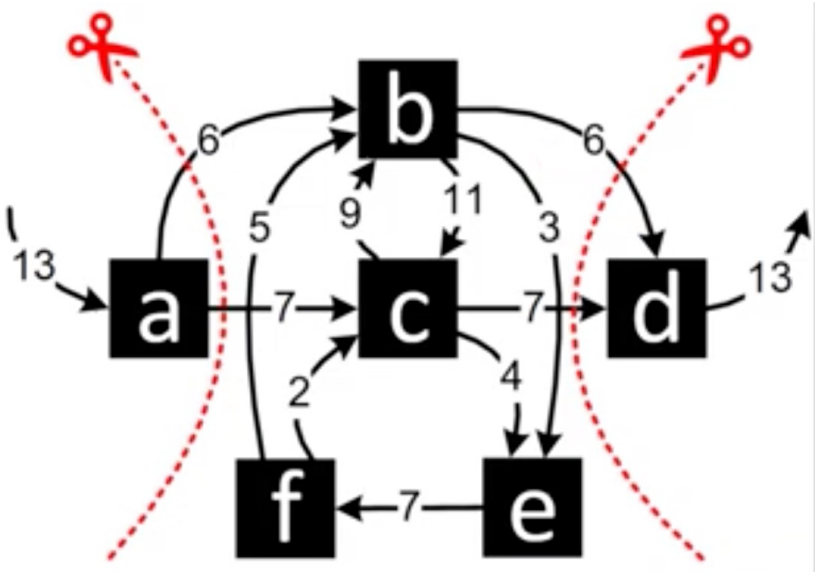


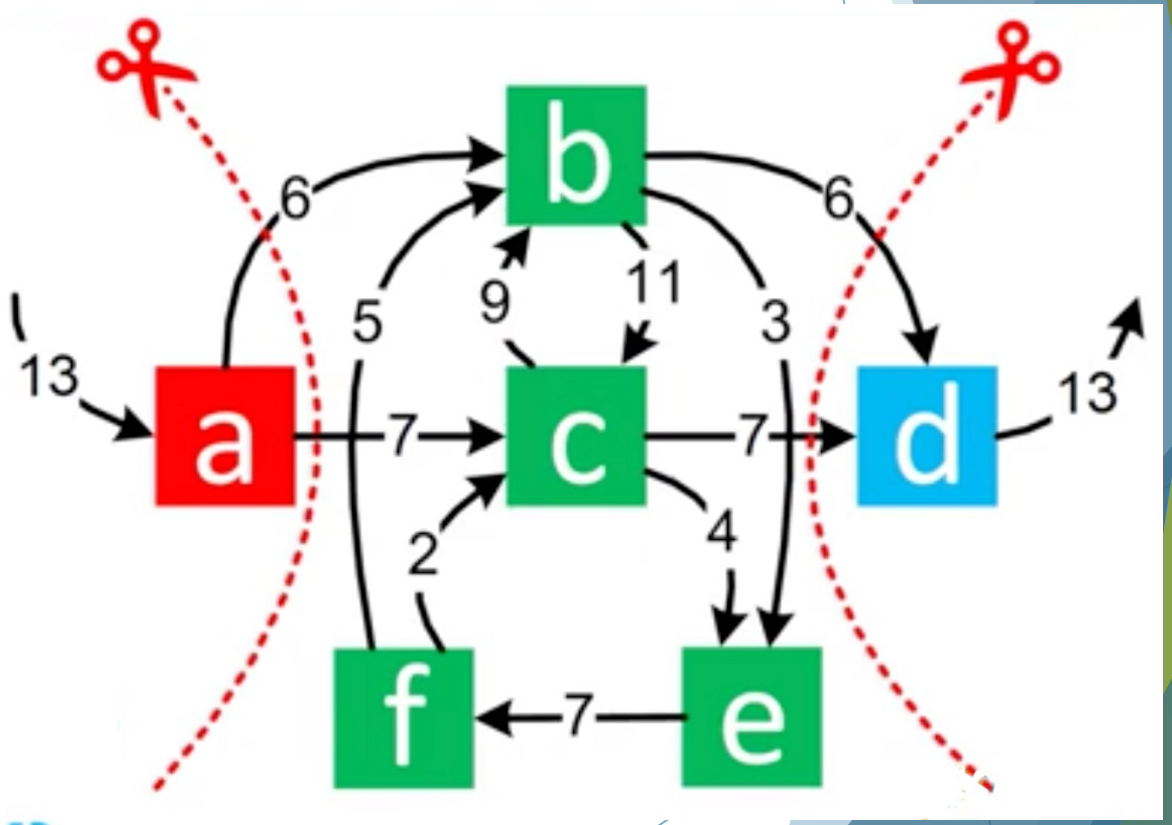
Sequence cut

Partitions the directly-follows graph into parts where arcs are going in one direction



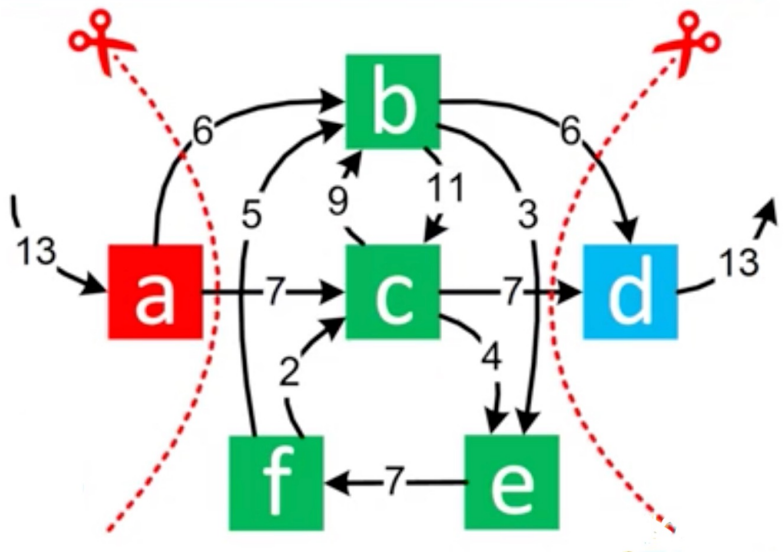
Partition activities based on sequence cut





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

Partition events based on sequence cut



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

3x a b c d

4x a c b d

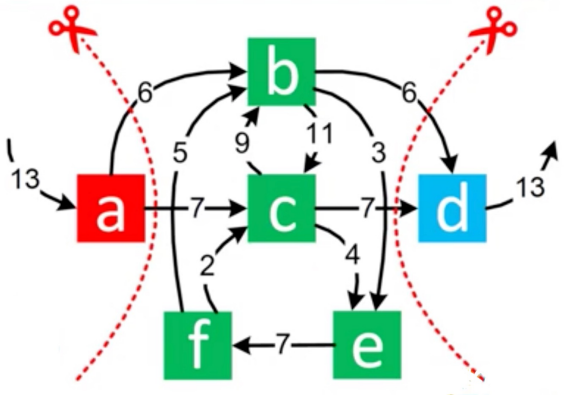
2x a b c e f b c d

2x a c b e f b c d

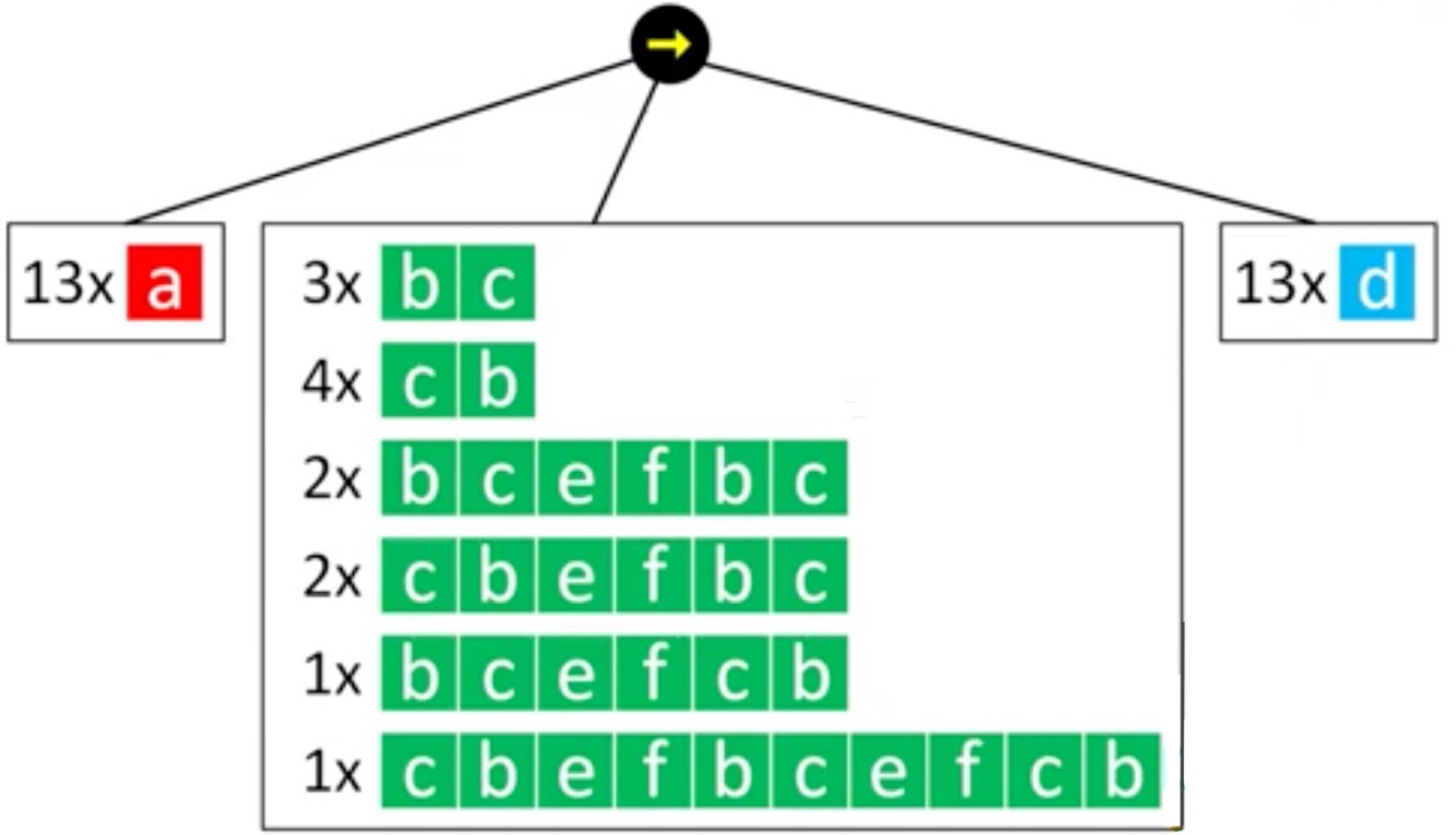
1x a b c e f c b d

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

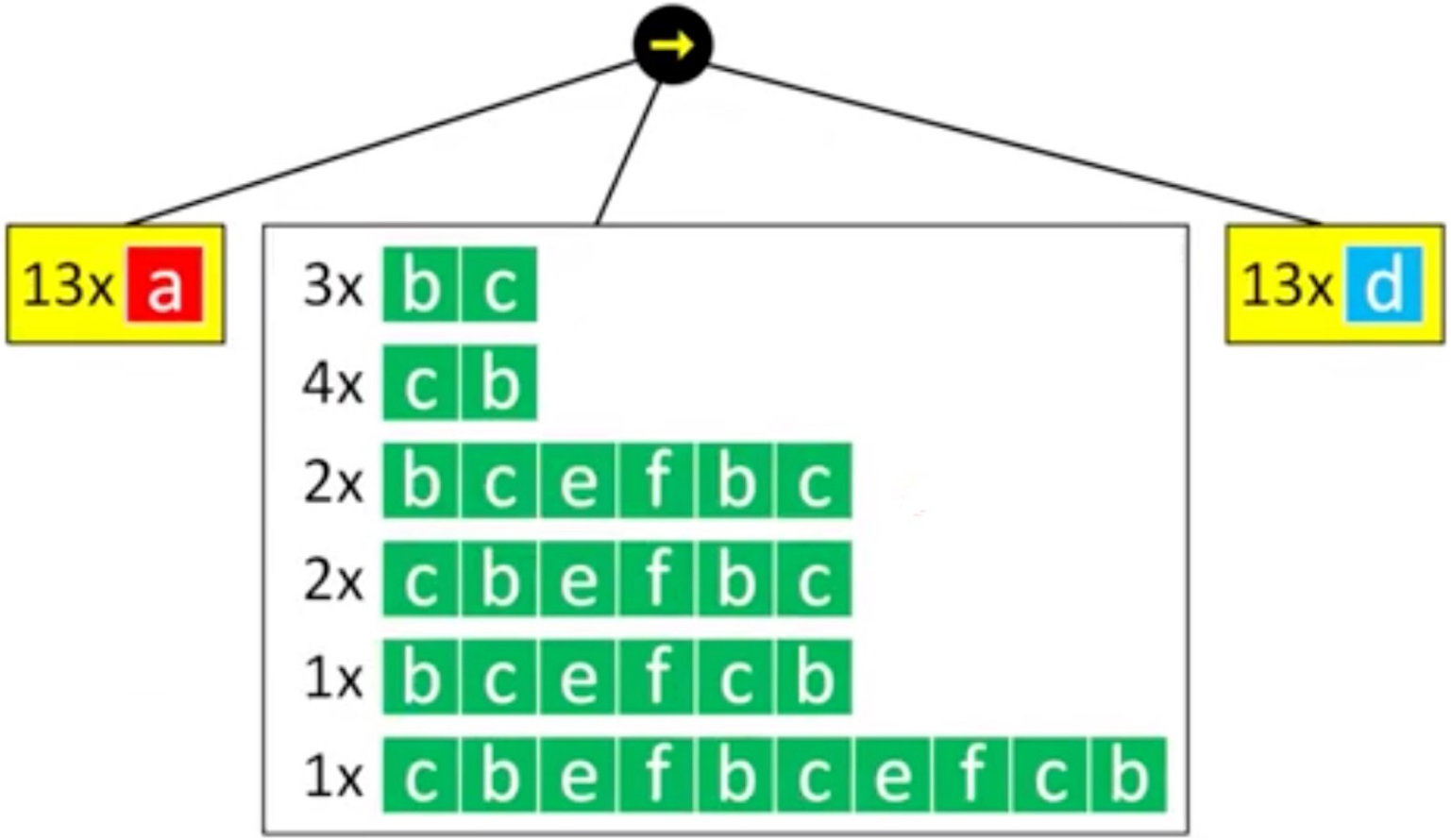
Partition events based on sequence cut



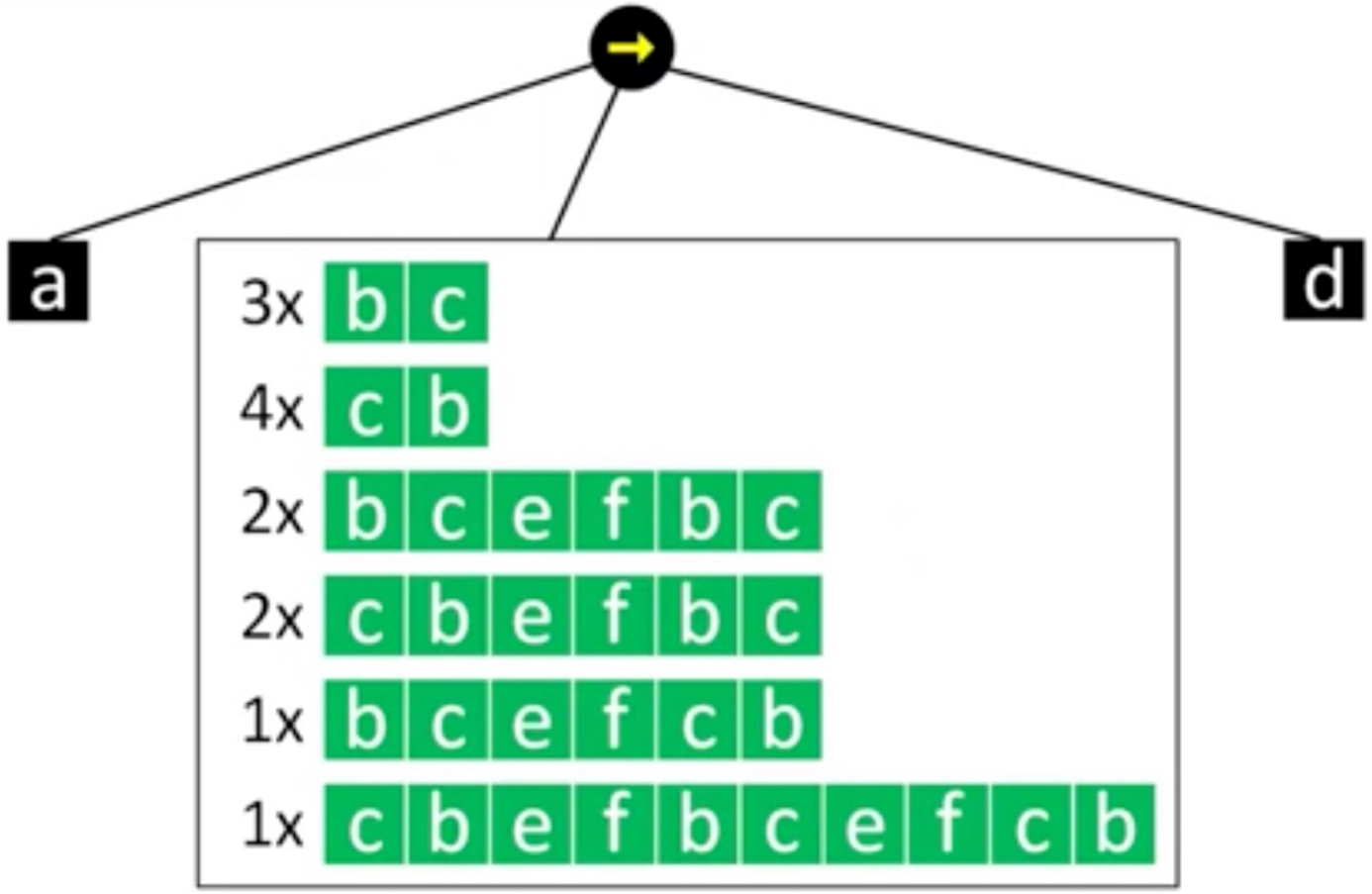
3x a b c d
4x a c b d
2x a b c e f b c d
2x a c b e f b c d
1x a b c e f c b d
1x a c b e f b c e f c b d



Handle base cases

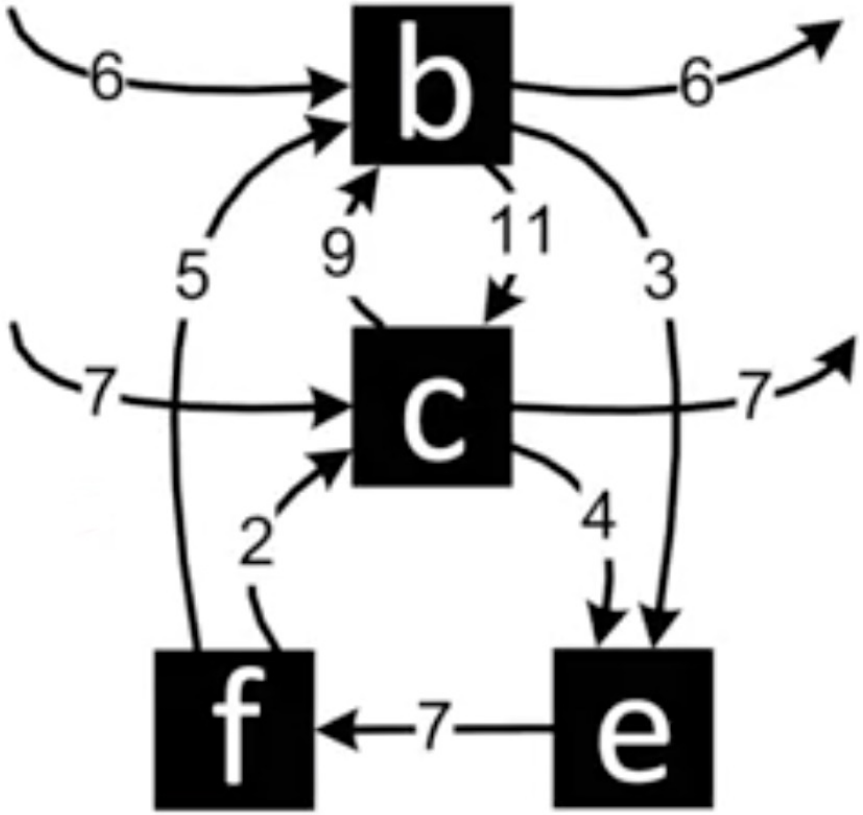


Recurse on non-base cases



Directly-follows graph based on sublog

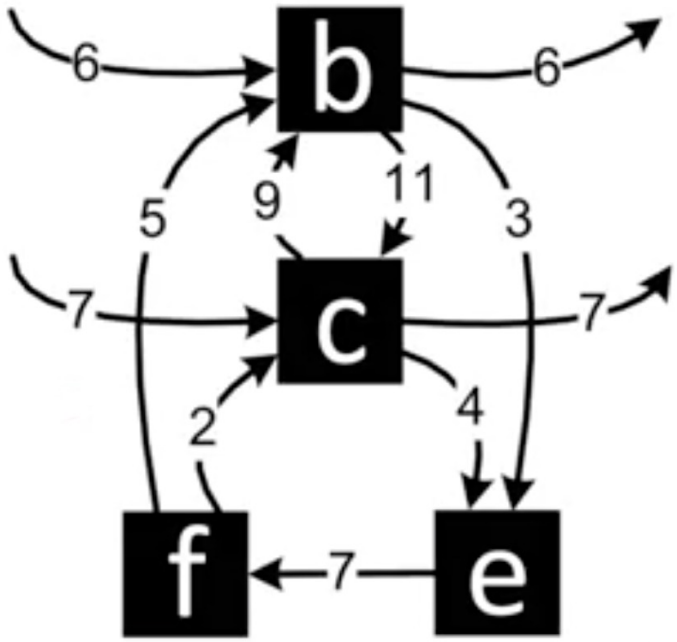
3x b c
4x c b
2x b c e f b c
2x c b e f b c
1x b c e f c b
1x c b e f b c e f c b

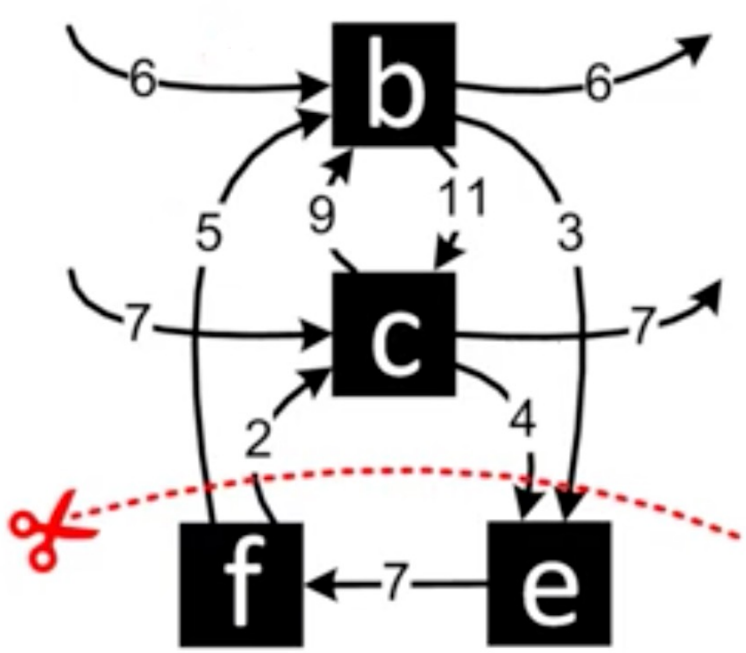


Loop cut

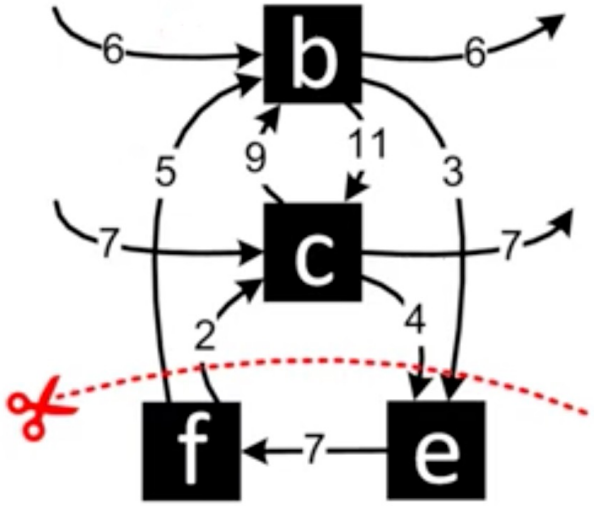
We need **do** and **redo** parts:

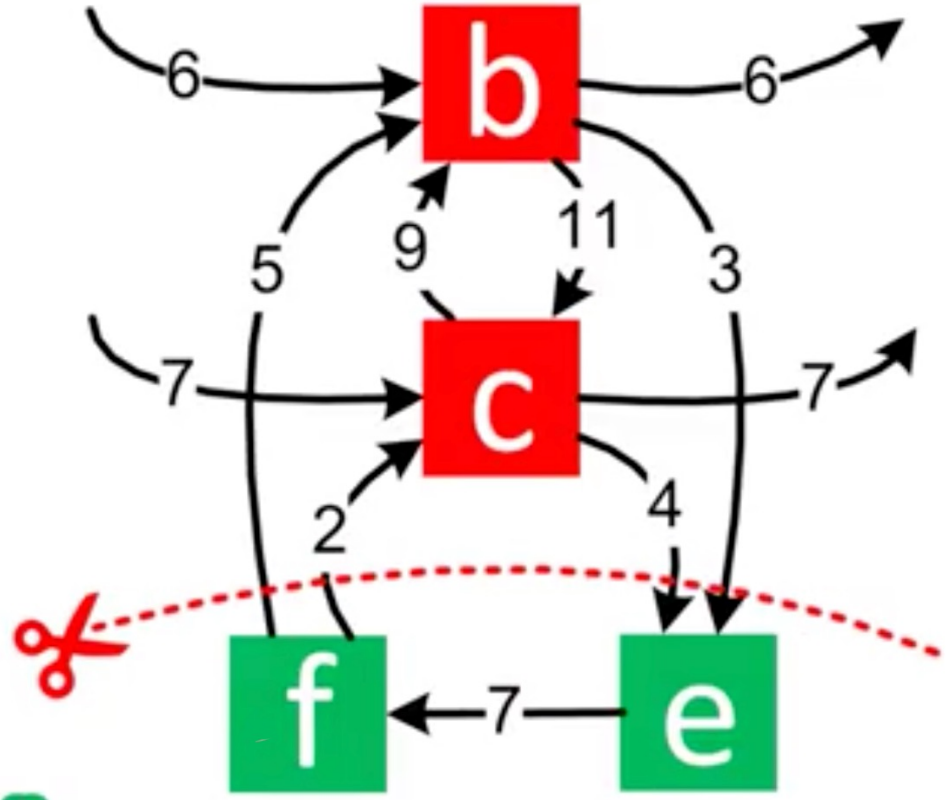
- Everything should **begin** and **end** in do-part
- From all the end activities, we should be able to **move to redo-part** & we should be able to **move to the start activities in do-part** from the redo-part





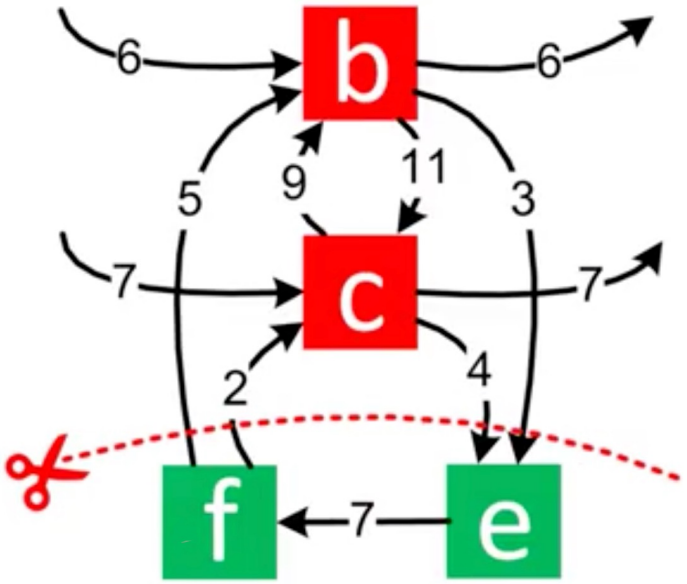
Partition activities based on loop cut





{b,c} , {e,f}

Partition events based on loop cut



{b,c} , {e,f}

3x b c

4x c b

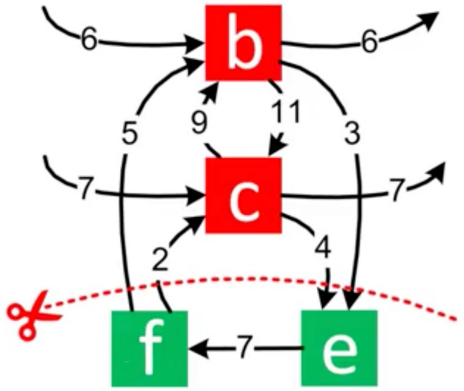
2x b c e f b c

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

Partition events based on loop cut



3x b c

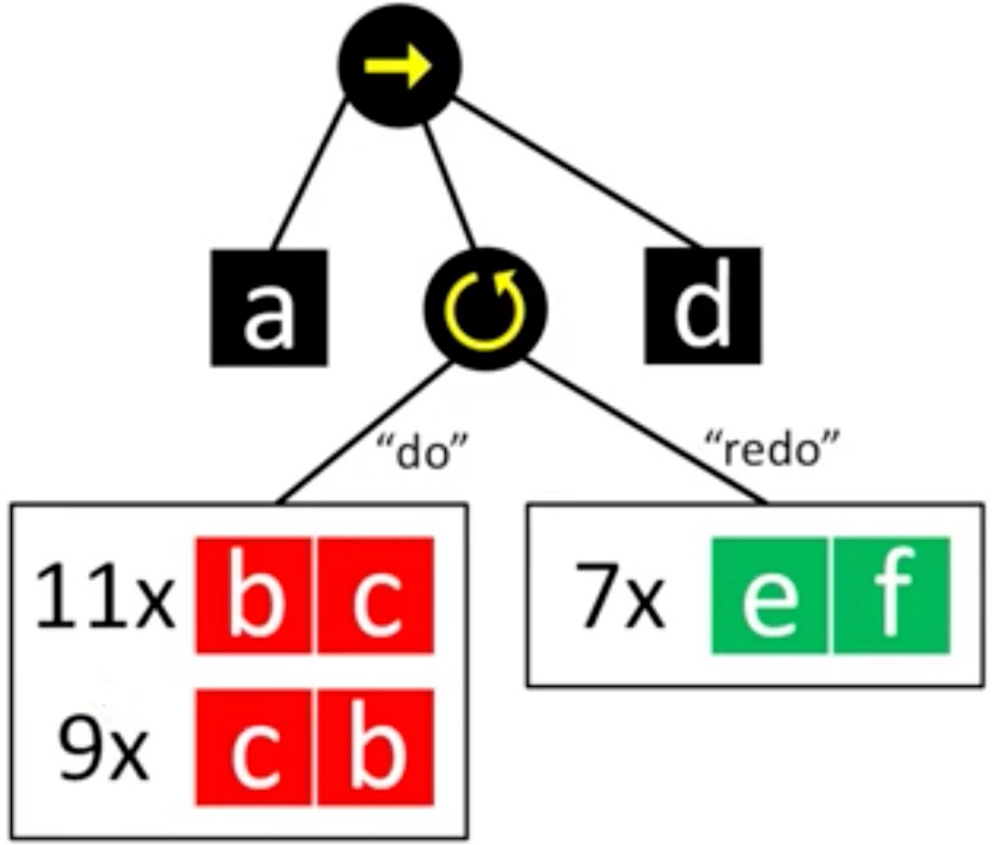
4x c b

2x b c e f b c

2x c b e f b c

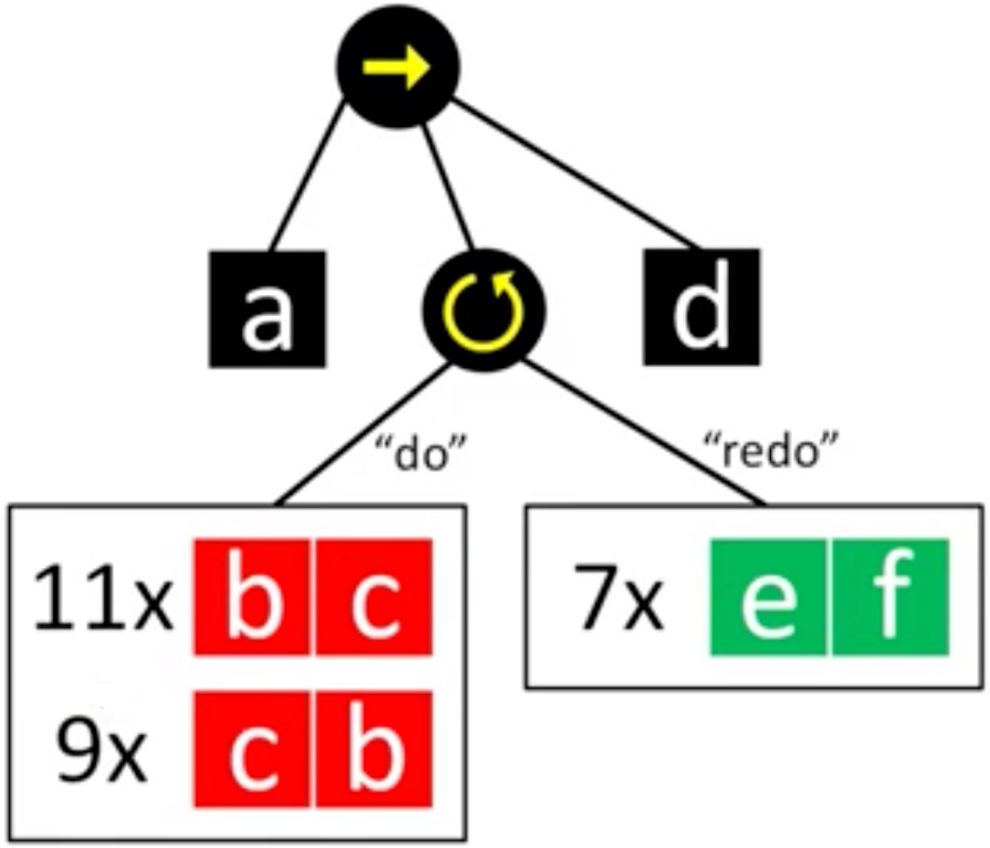
1x b c e f c b

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



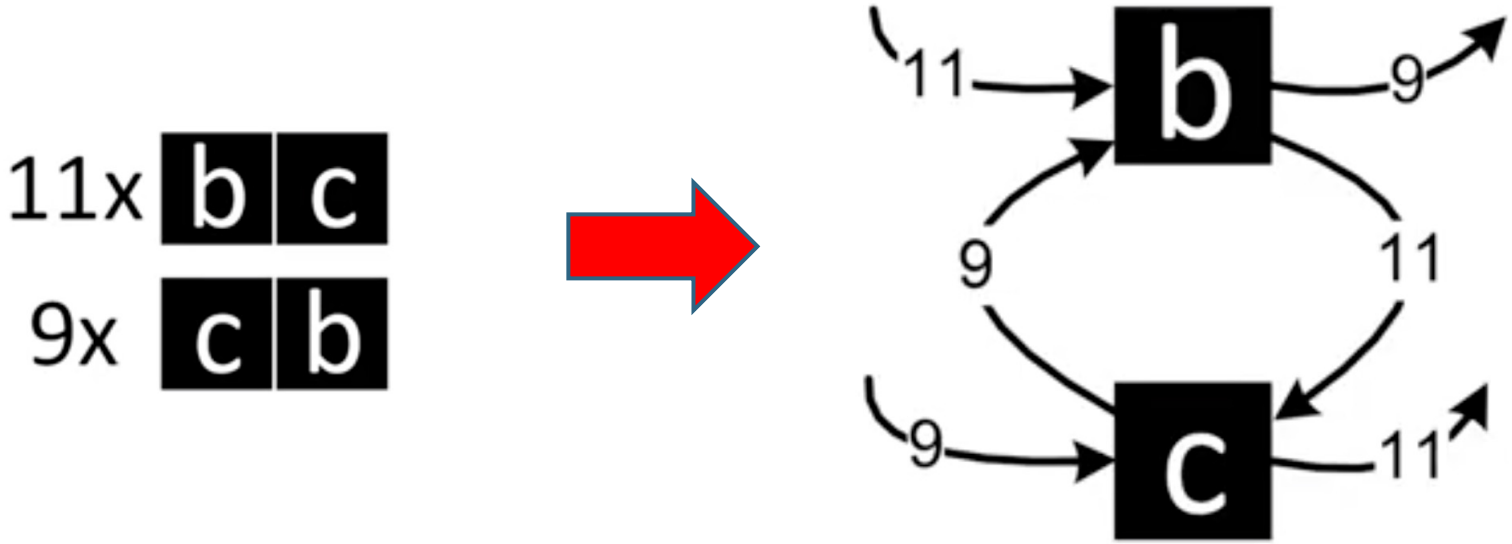
NOTE: each trace will be considered as a new instance on performing a do-part in cycle

Recurse on the two sublogs



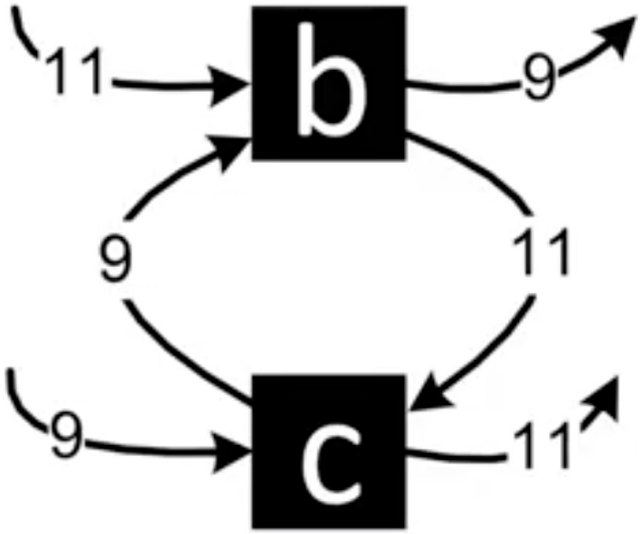
NOTE: logs are getting simpler

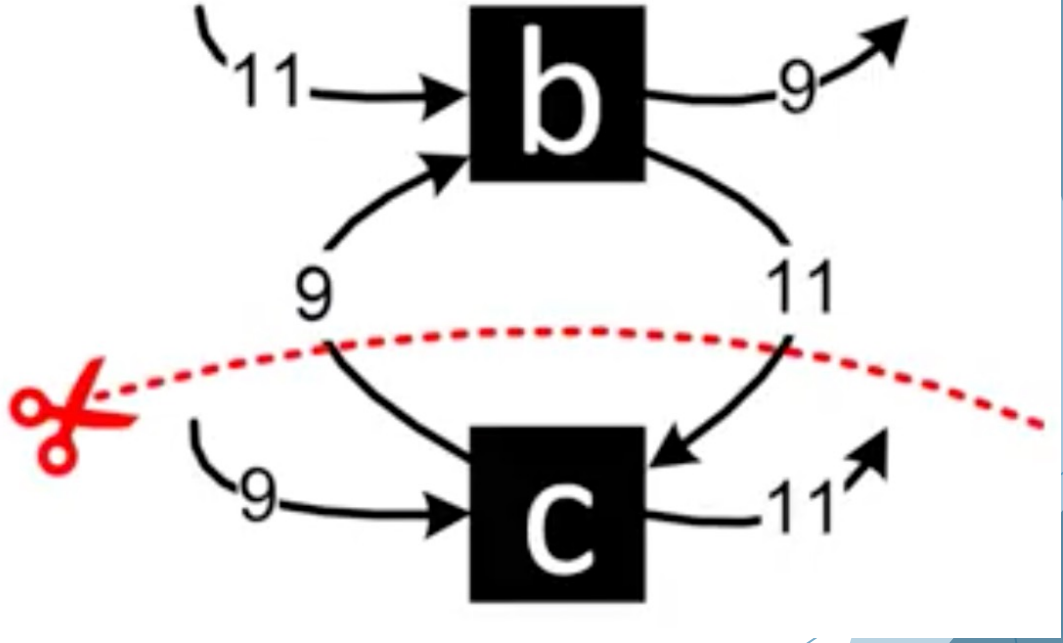
Directly-follows graph based on sublogs



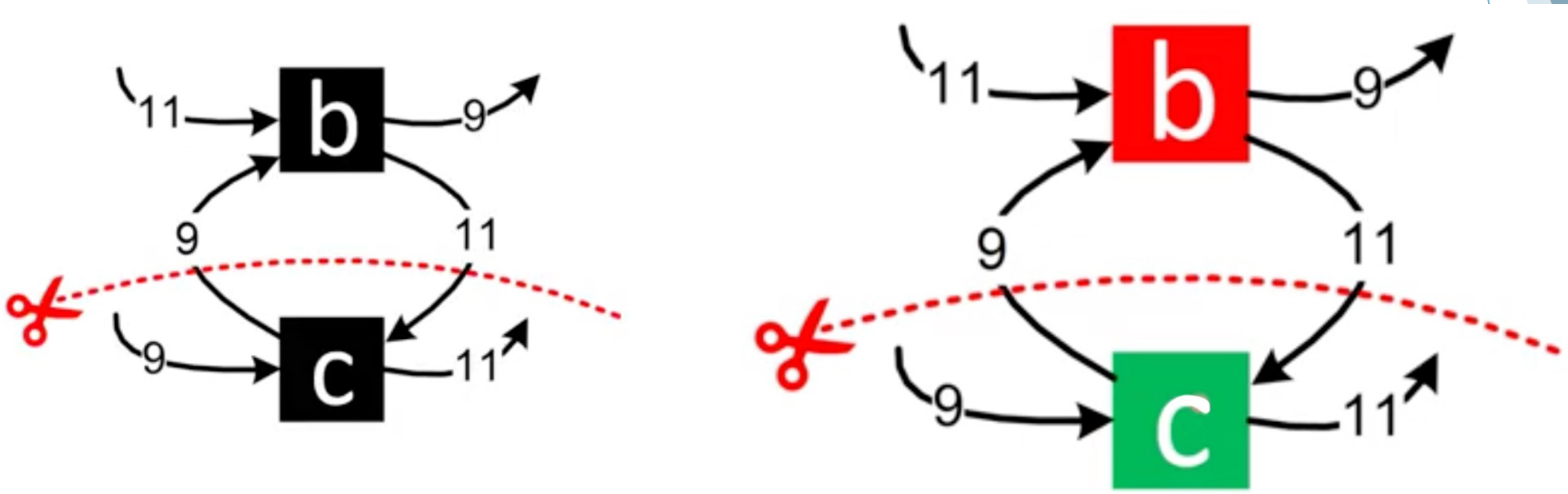
Parallel cut

Any activity in one subset should be followed by any activity in the second subset, then we can split the two subsets.
Also, all the subsets should have **start** and **end** activities.



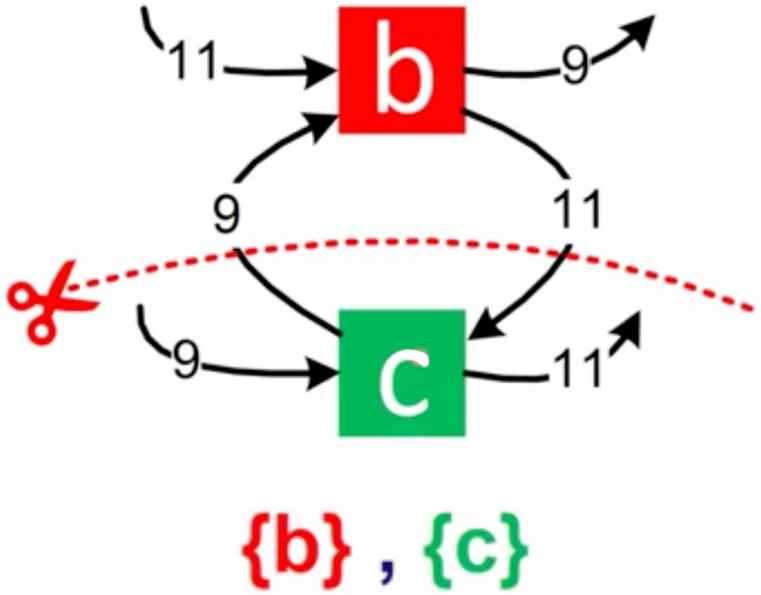


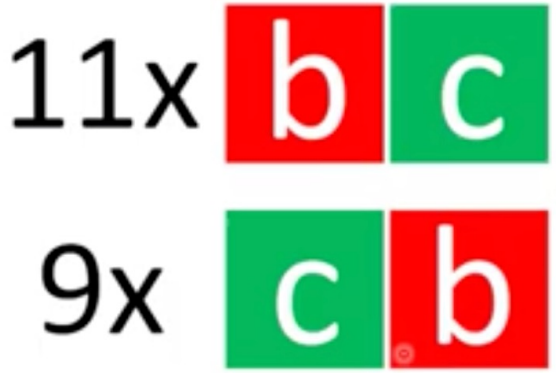
Partition activities based on parallel cut



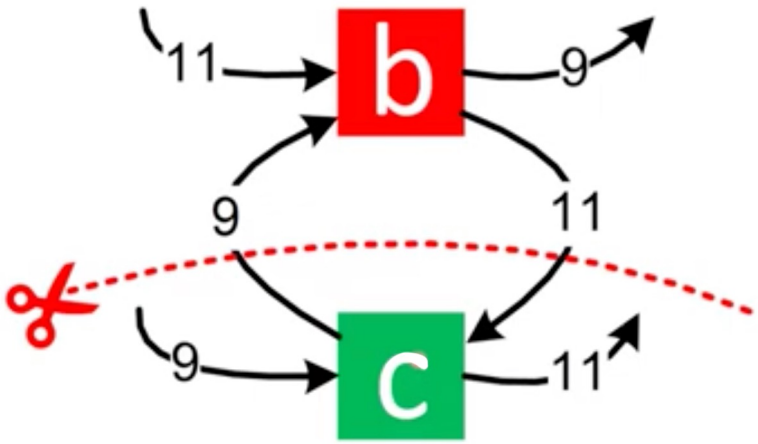
{b} , {c}

Partition events based on parallel cut



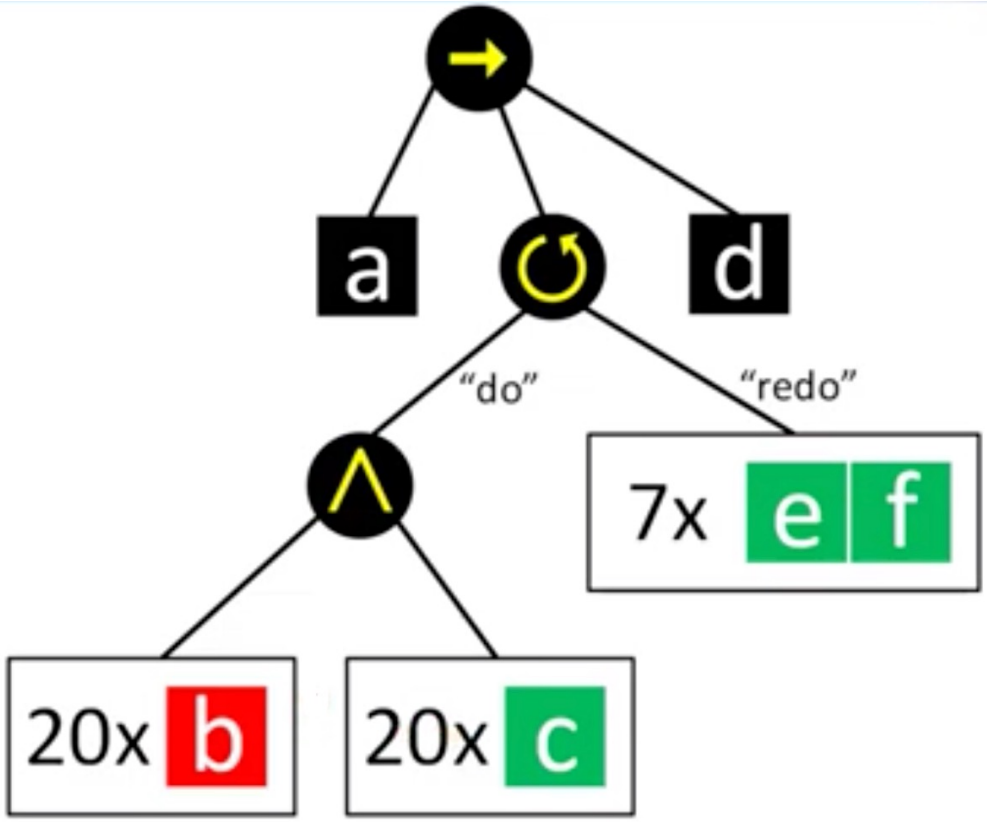


Partition events based on parallel cut

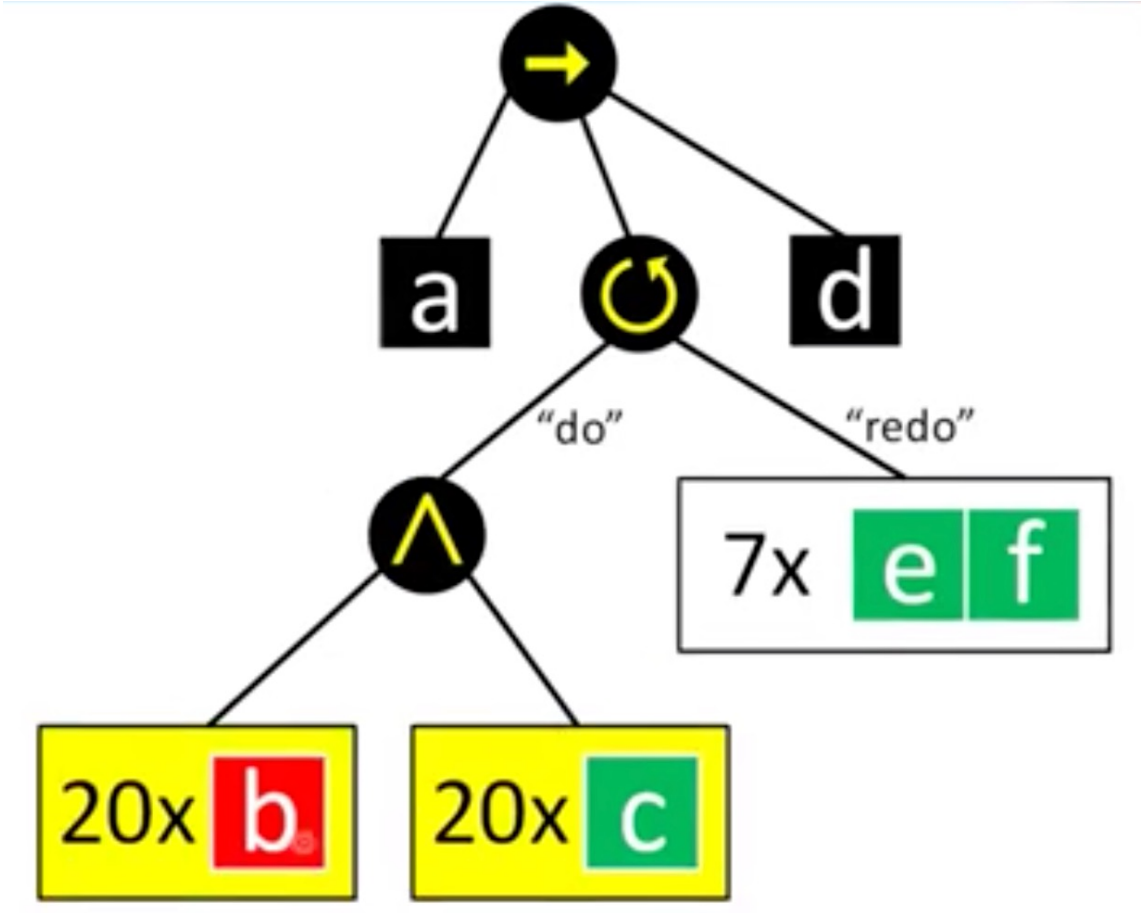


11x b c

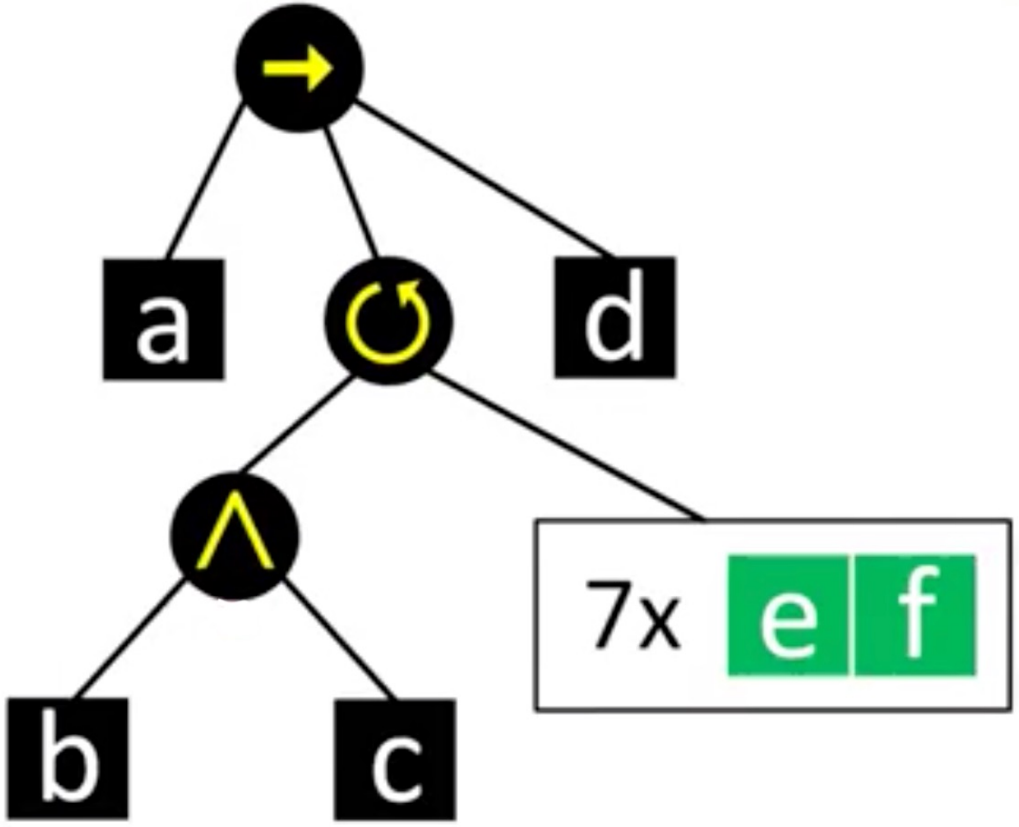
9x c b



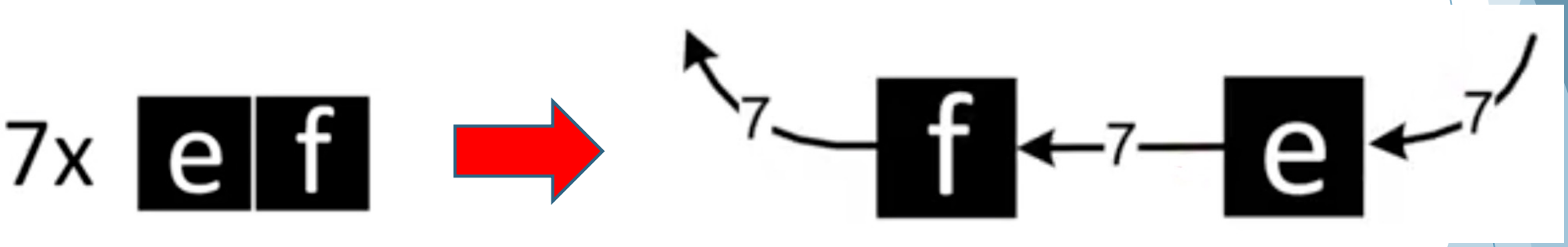
Handle base cases



Recurse on the remaining sublog(s)

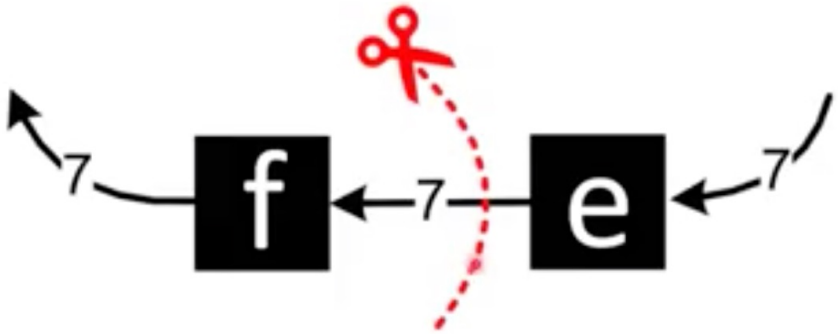


Directly follows graph based on sublog

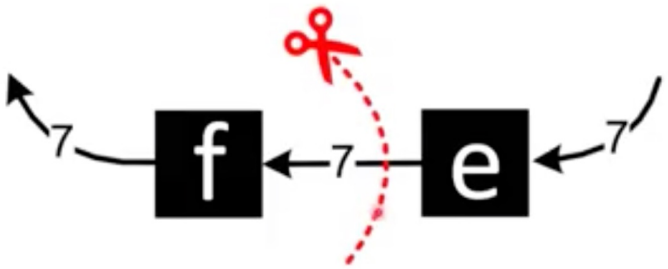


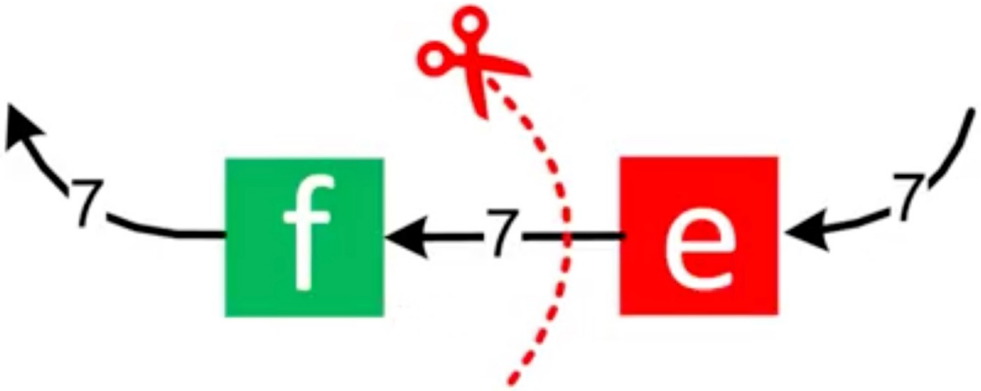
Sequence cut

Partitions the directly-follows graph into parts where arcs are going in one direction



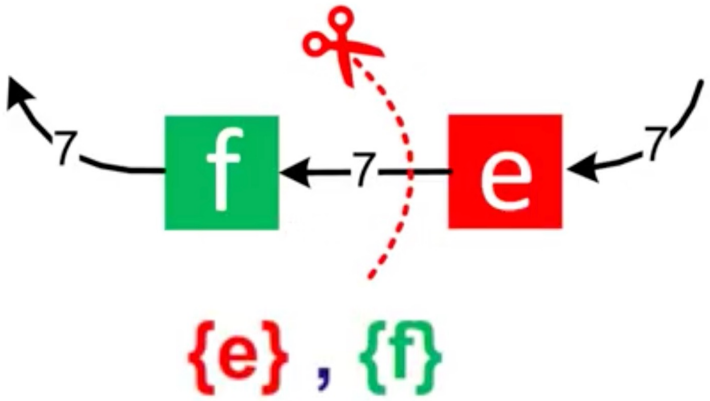
Partition activities based on sequence cut



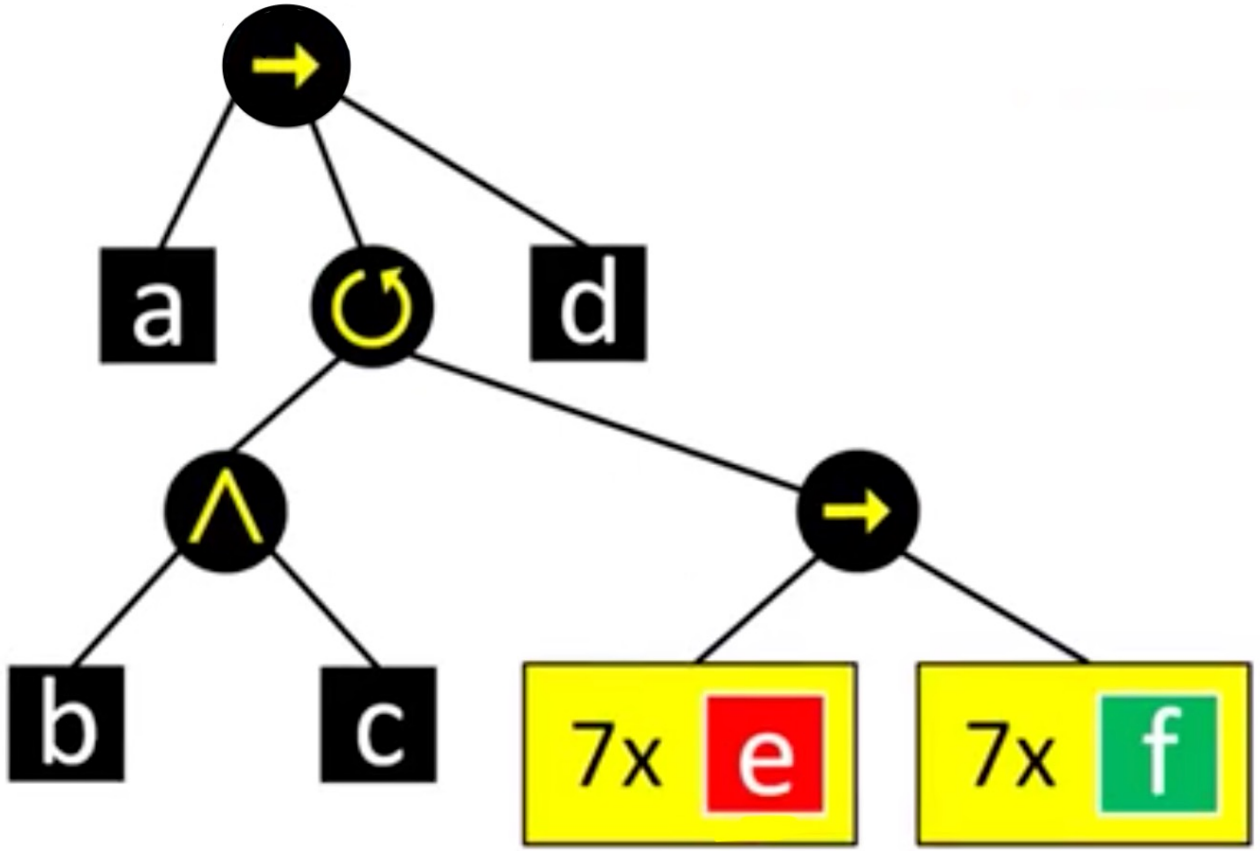


{e} , {f}

Partition events based on sequence cut



Handle base cases



Final model

3x a b c d

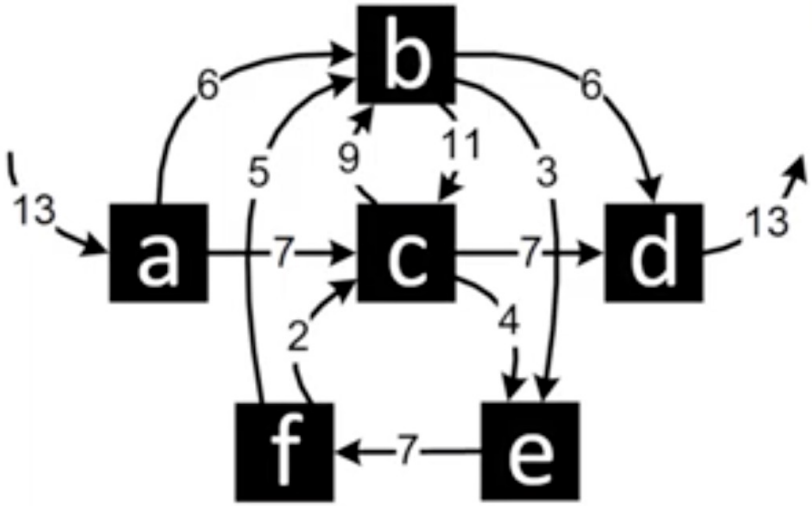
4x a c b d

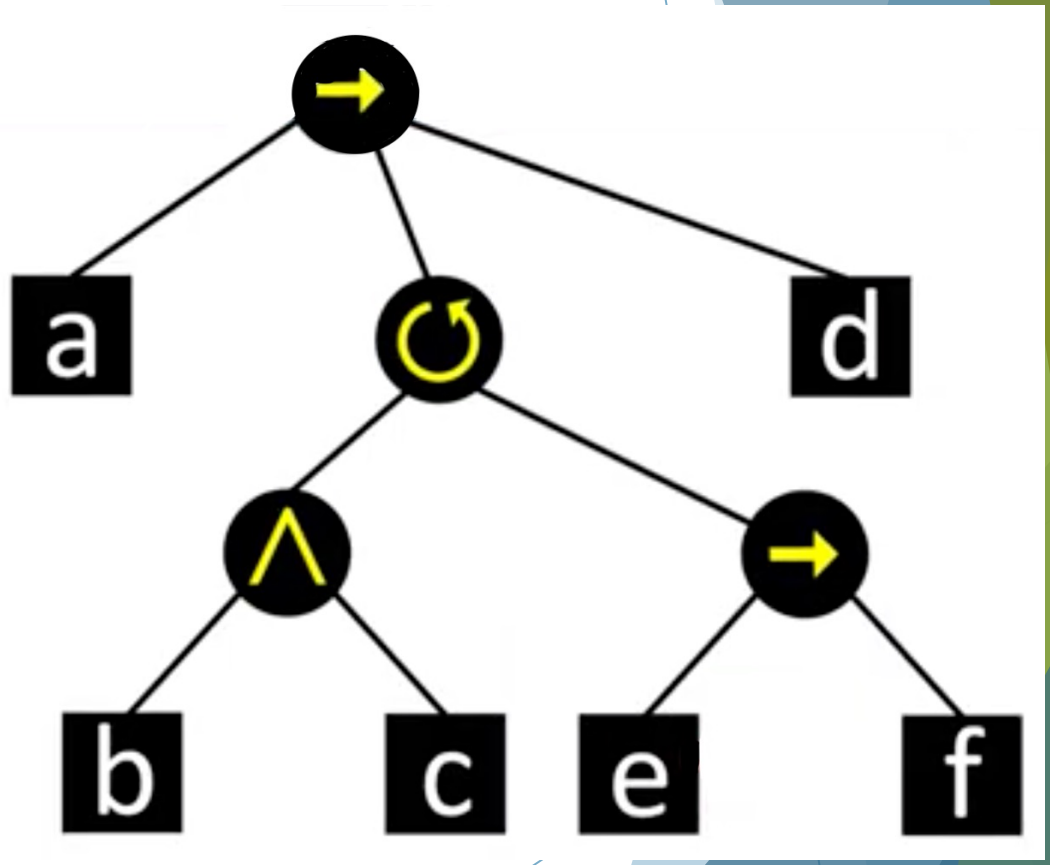
2x a b c e f b c d

2x a c b e f b c d

1x a b c e f c b d

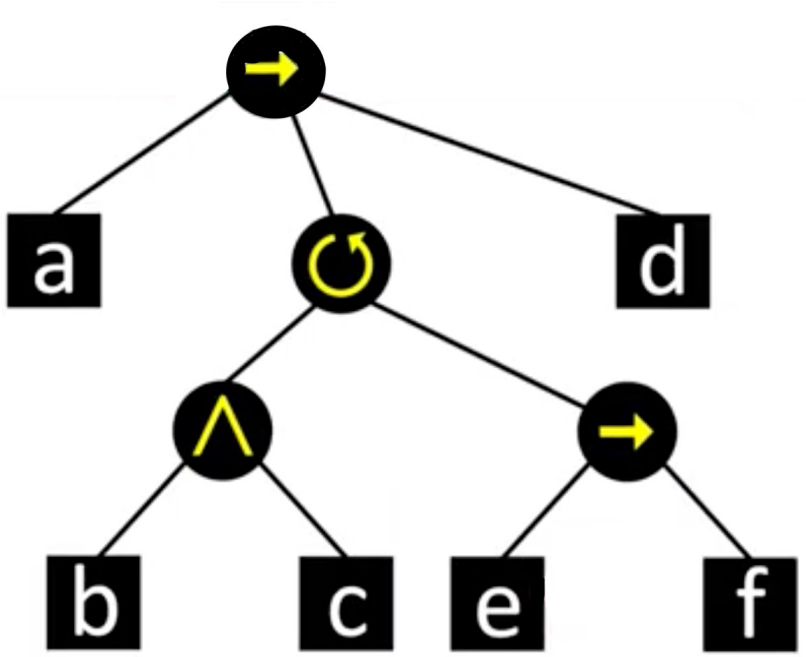
1x a c b e f b c e f c b d





Practice Work

- Convert this process tree into an equivalent WF-net.



Reading Material

- Chapter 7: Aalst