Handling Duplicated Tasks in Process Discovery by Refining Event Labels

Xixi Lu (x.lu@tue.nl)[1]

Dirk Fahland[1]

Frank van den Biggelaar[2]

Wil van der Aalst[1]

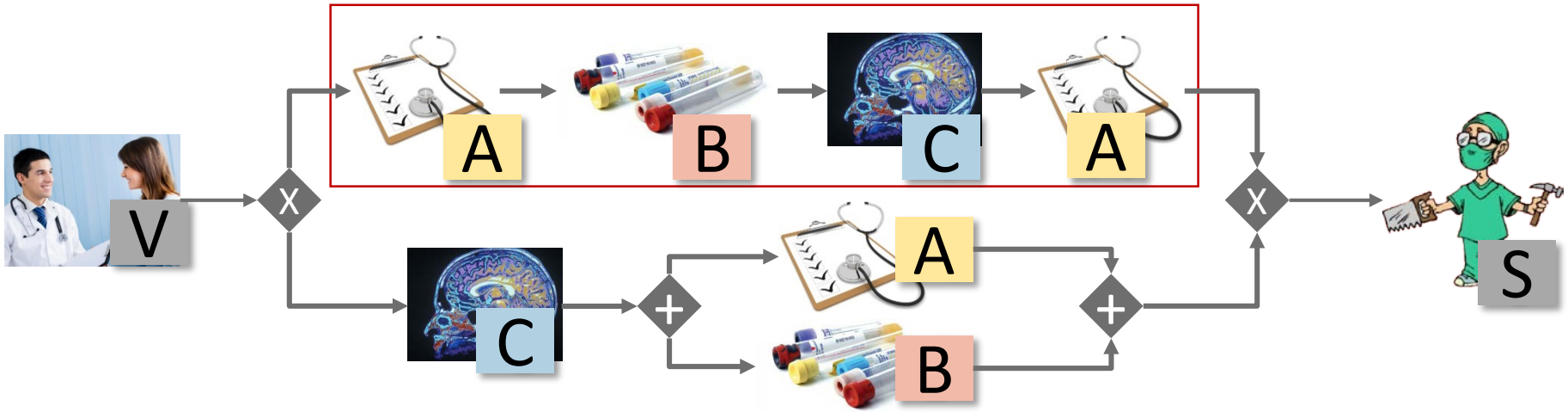
[1]Eindhoven University of Technology

[2]Maastricht University Medical Center

Overview

- (5 min) Research Problem
- (10 min) Approach
- (5 min) Evaluation and Conclusion

Duplicated Tasks – Events Having Imprecise Labels



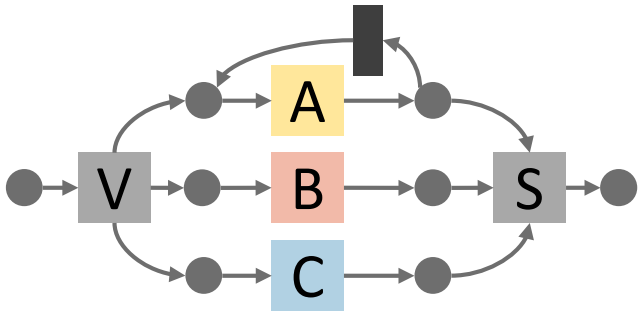
Record

V → A → B → C → A → S

V → C → B → A → S

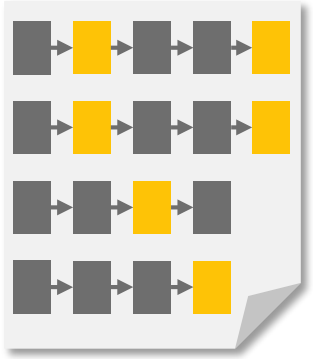
V → C → A → B → S

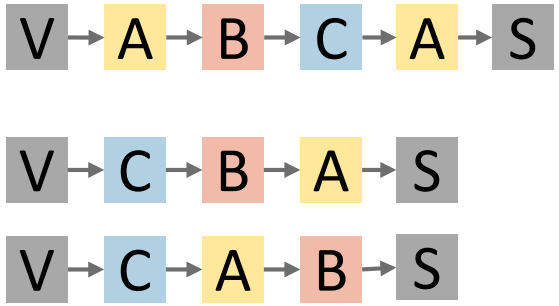
Discover



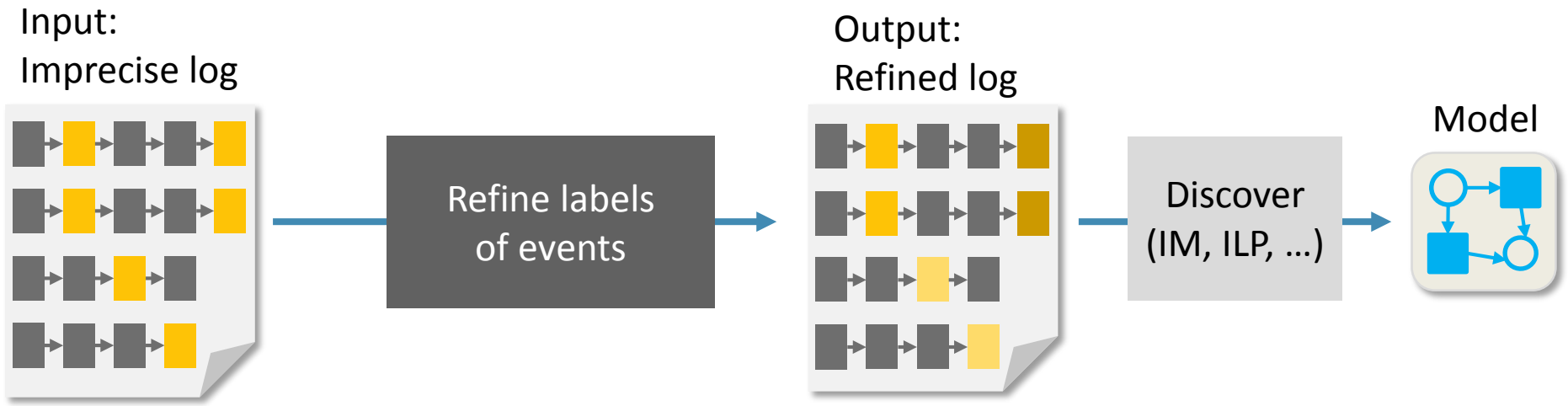
Research Problem

Input:
Imprecise log





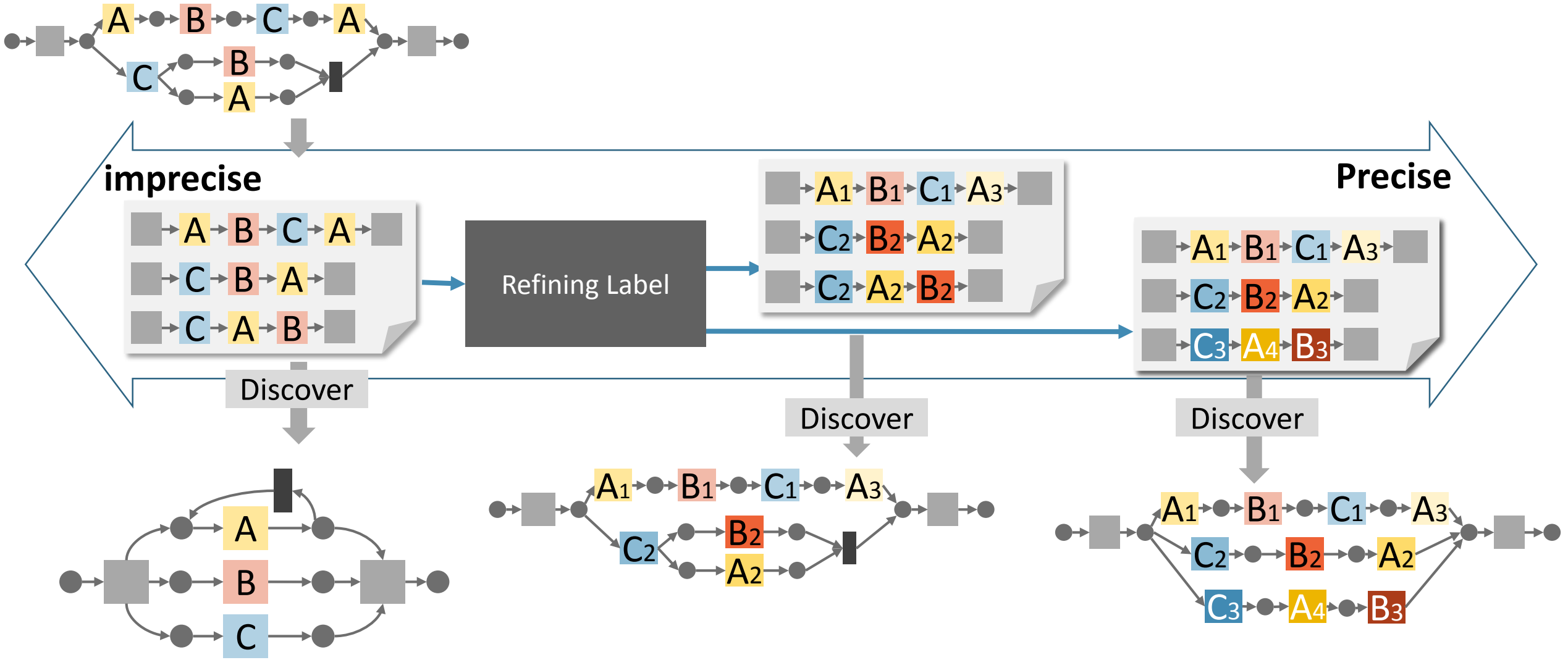
Research Problem



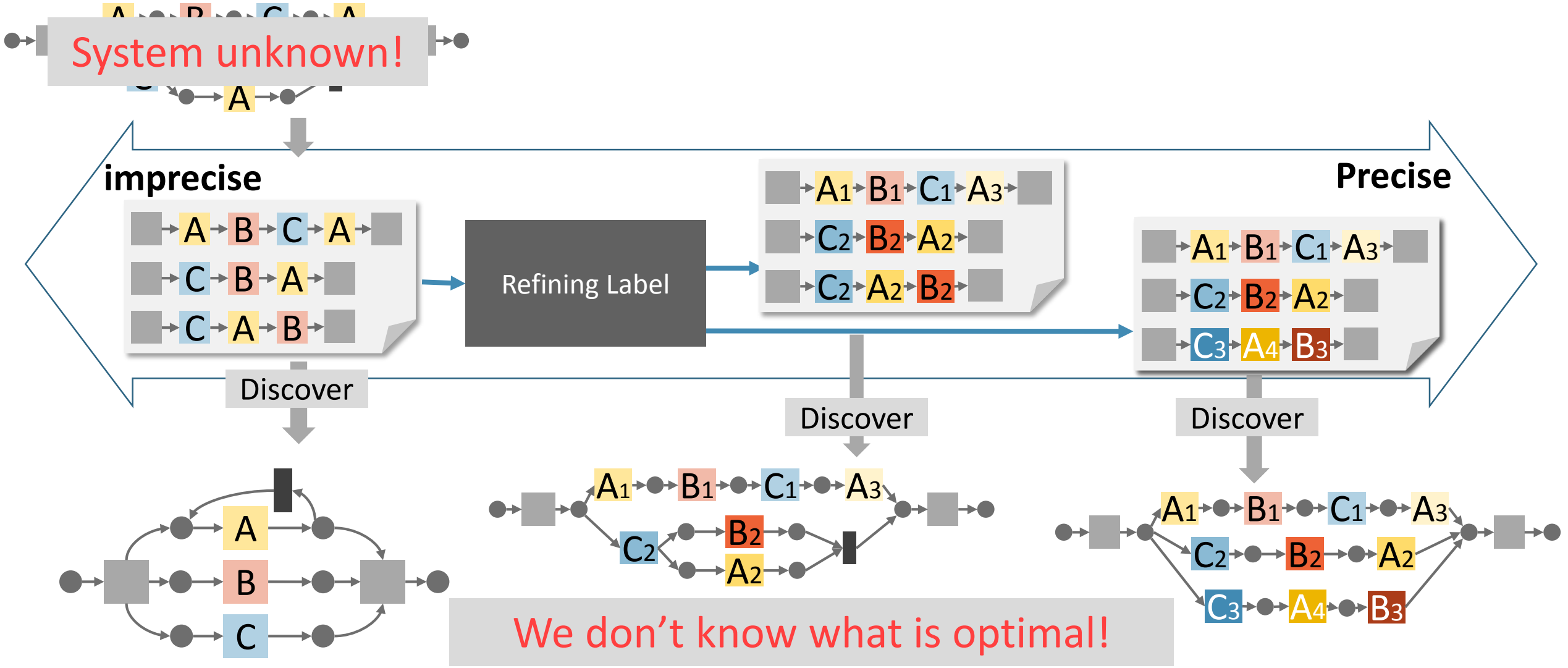
So that

1. all traces & events preserved (no filtering)
2. model is more precise (than without refining)
3. we can explore different refinement of labels interactively (because we don't know correct label refinement from given log)

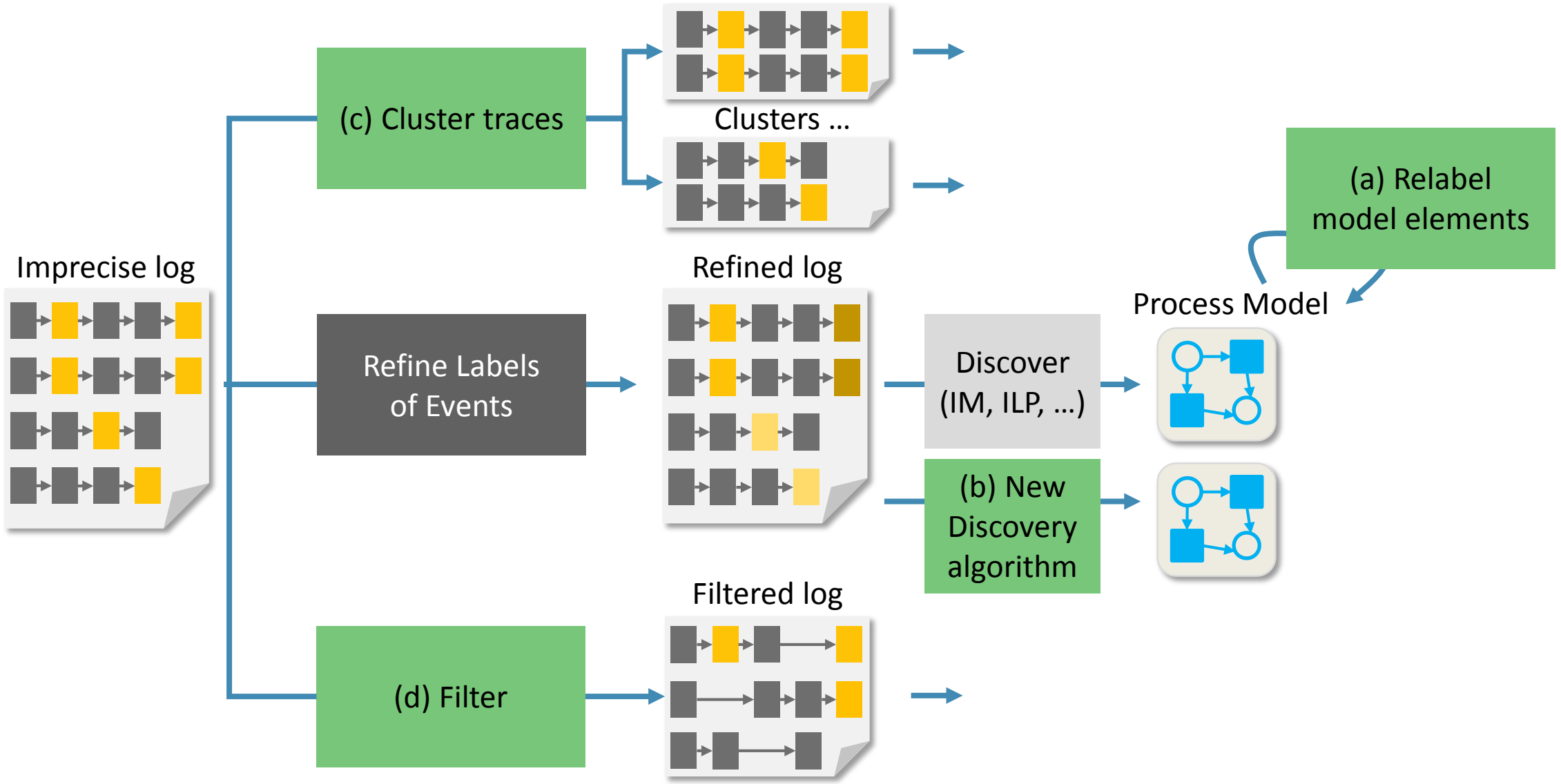
Refining Event Labels



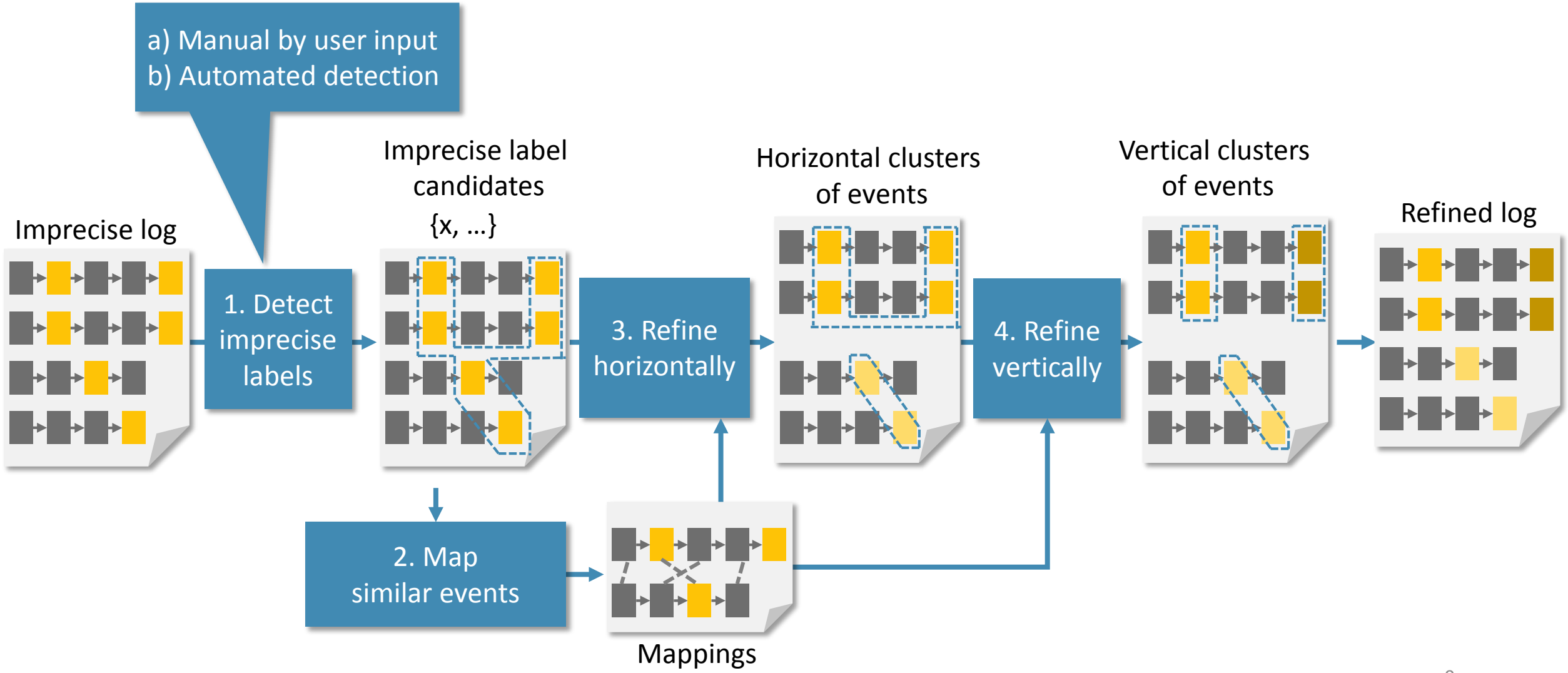
Refining Event Labels



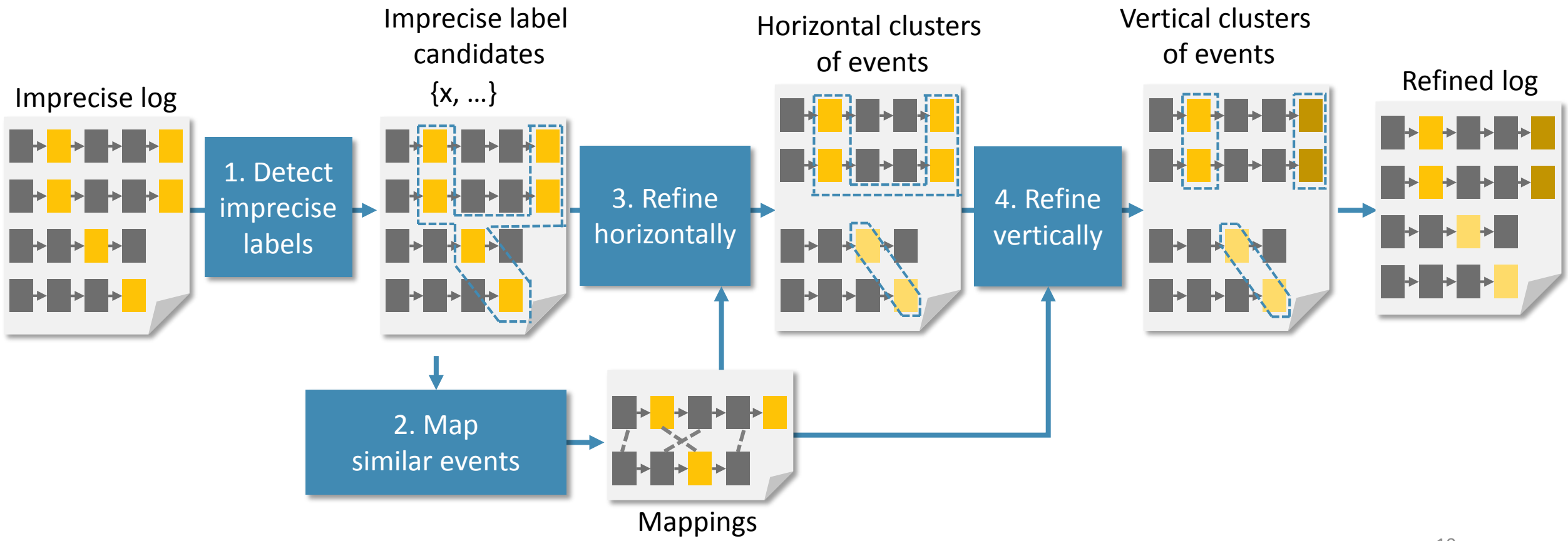
Related Work



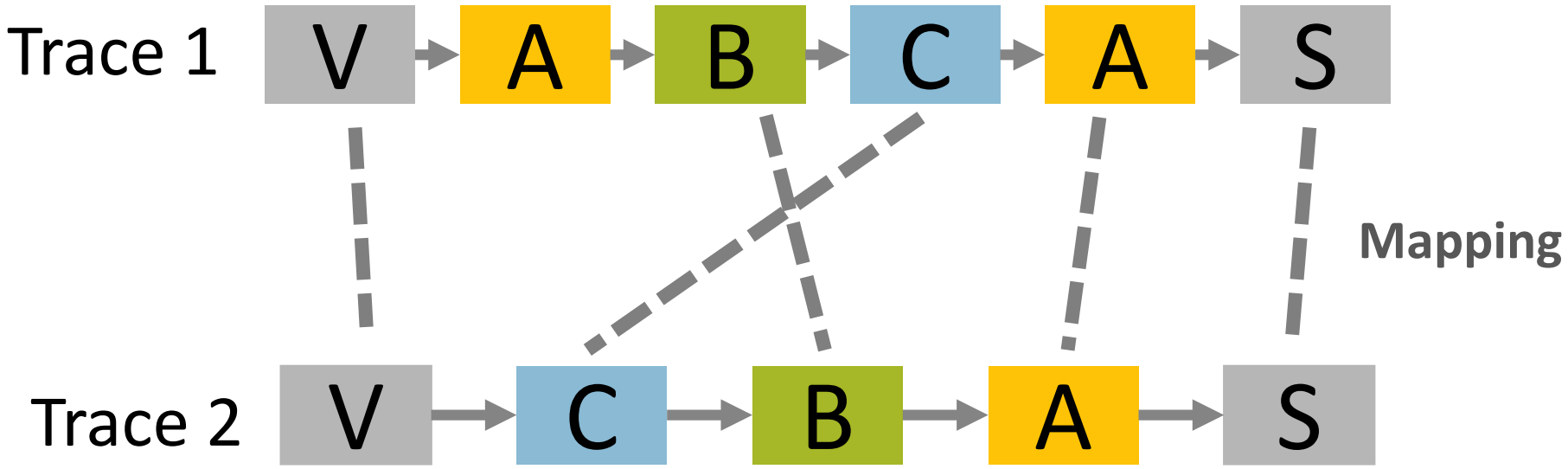
Approach


a) Manual by user input
b) Automated detection
Imprecise log
1. Detect imprecise labels
Imprecise label candidates
{x, …}
2. Map similar events
Mappings
3. Refine horizontally
Horizontal clusters of events
4. Refine vertically
Vertical clusters of events
Refined log

Approach

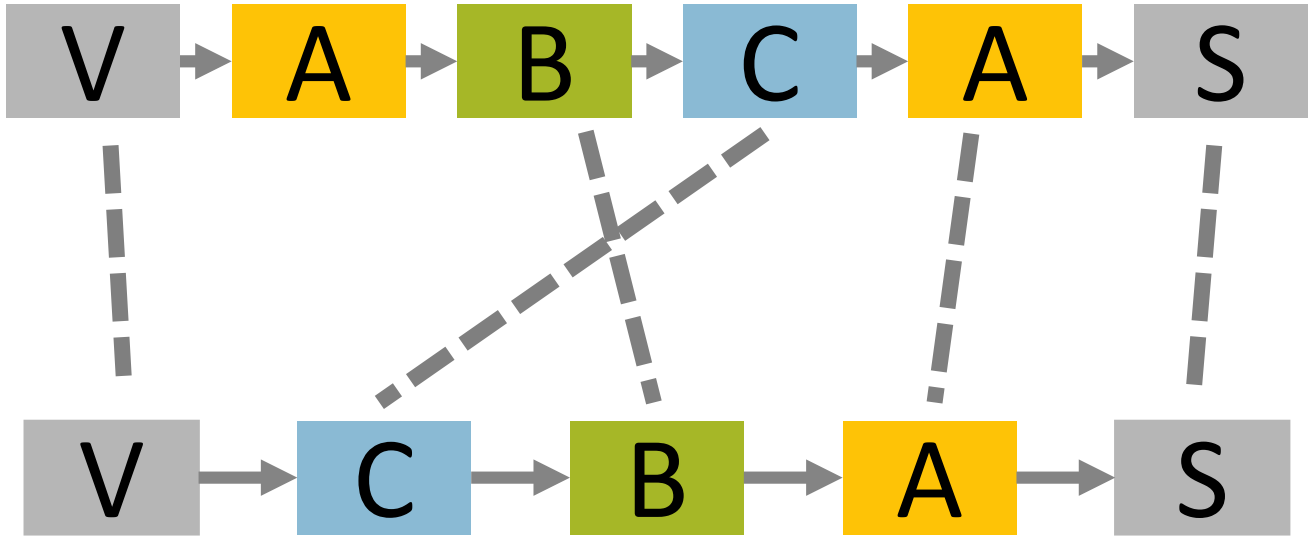

Imprecise log
1. Detect imprecise labels
Imprecise label candidates
{x, …}
2. Map similar events
Mappings
3. Refine horizontally
Horizontal clusters of events
4. Refine vertically
Vertical clusters of events
Refined log

Mapping Between Events



Quantify Similarity of Mapped Events

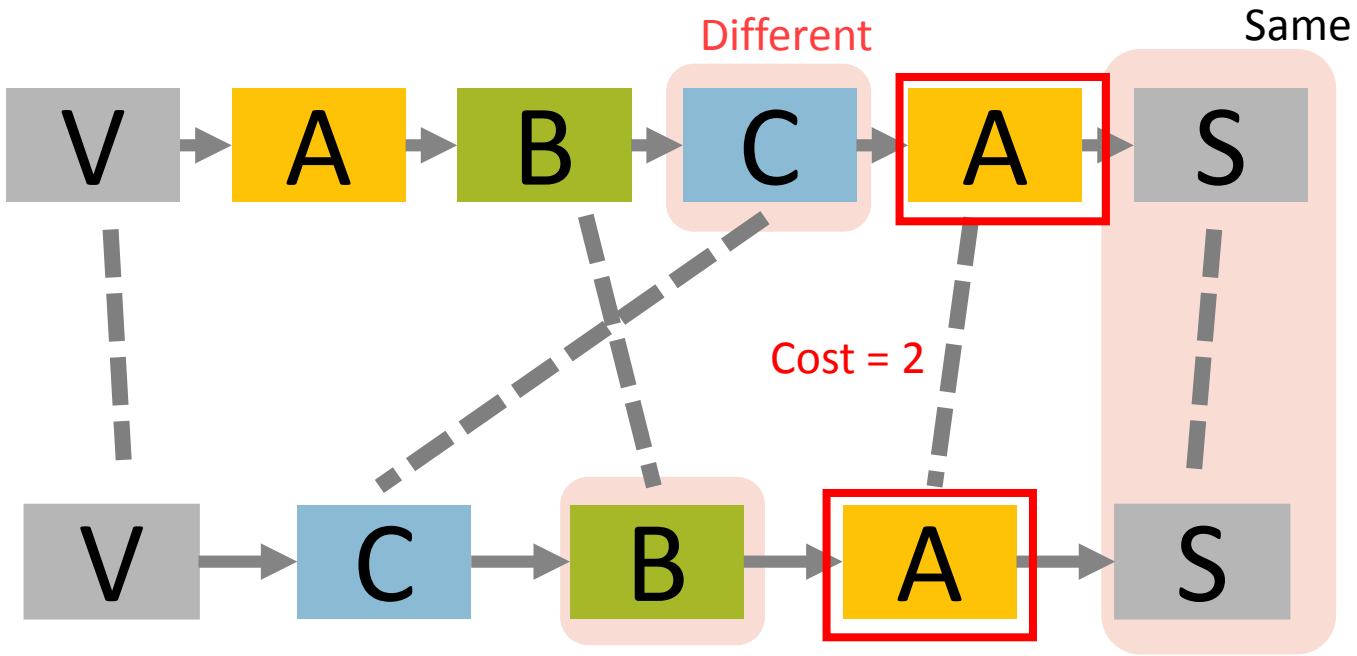
… based on “structural context of events”



Quantify Similarity of Mapped Events

… based on “structural context of events”

(1) Differences in neighbors

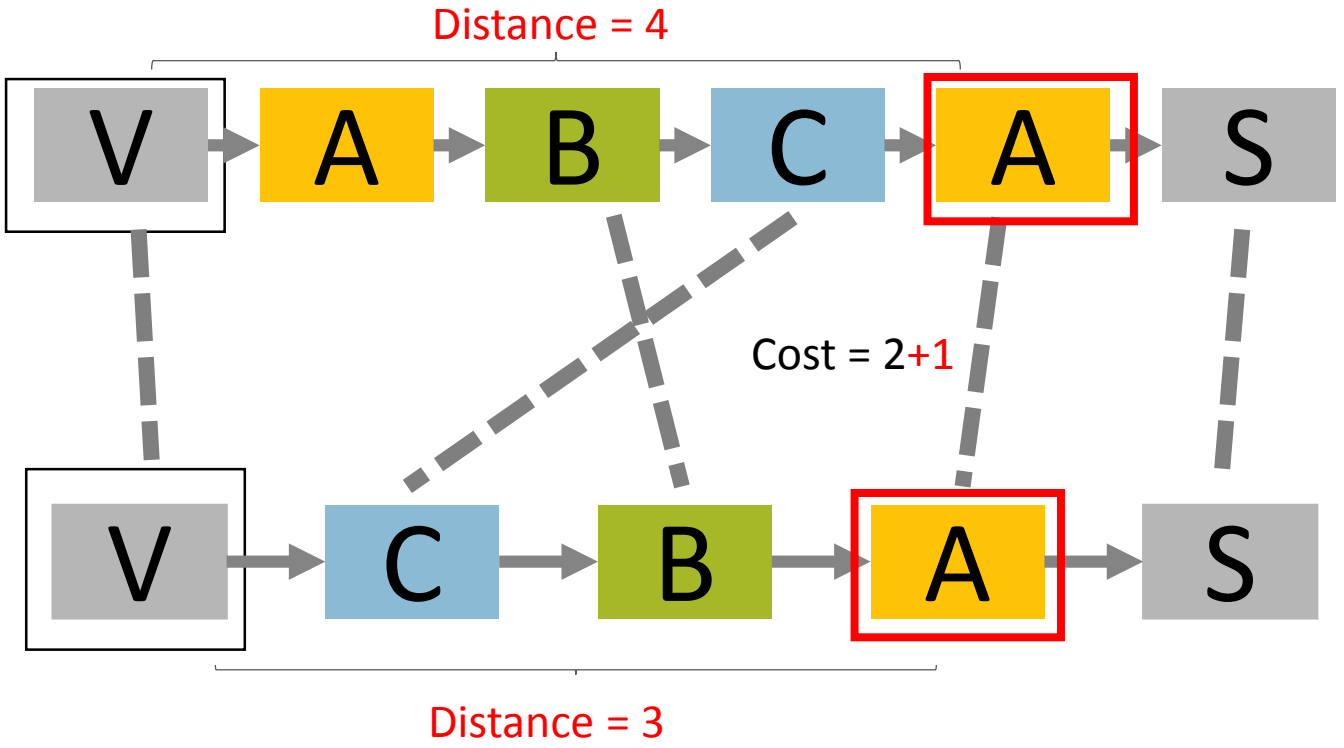


Quantify Similarity of Mapped Events

… based on “structural context of events”

(1) Differences in neighbors

(2) Differences in structure

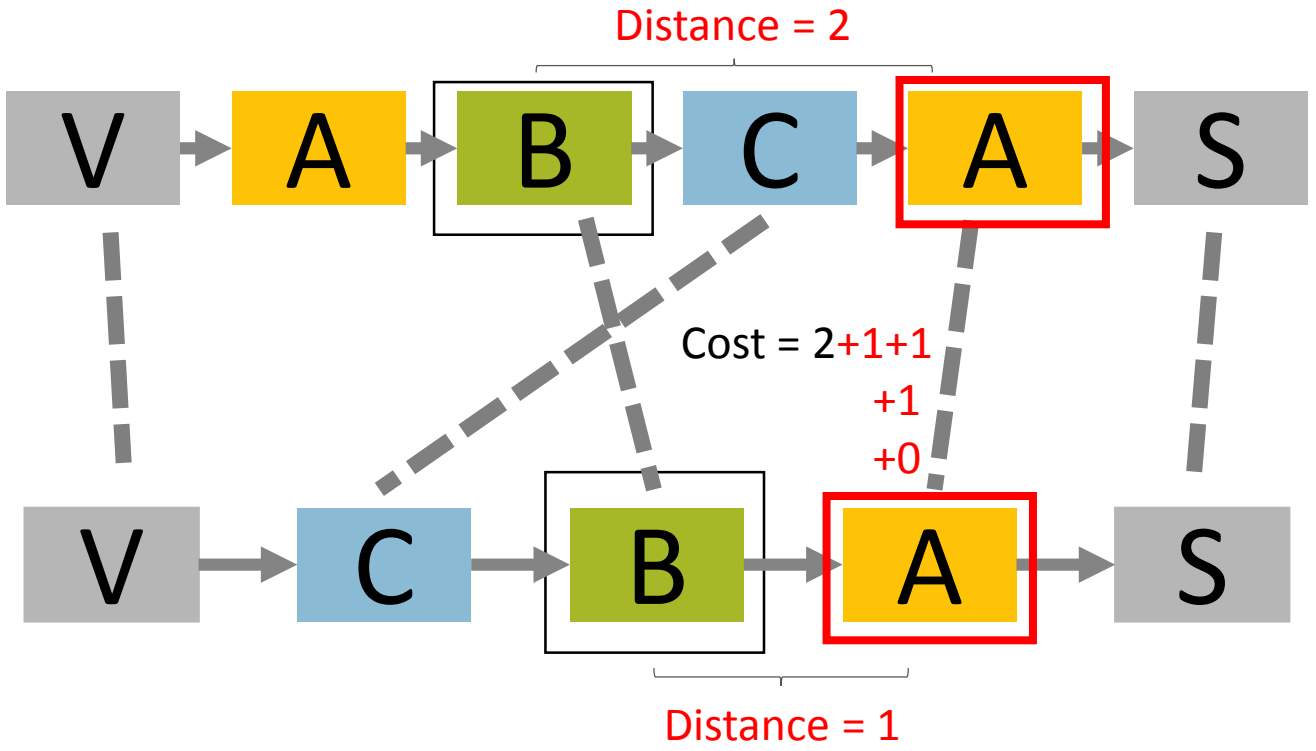


Quantify Similarity of Mapped Events

… based on “structural context of events”

(1) Differences in neighbors

(2) Differences in structure

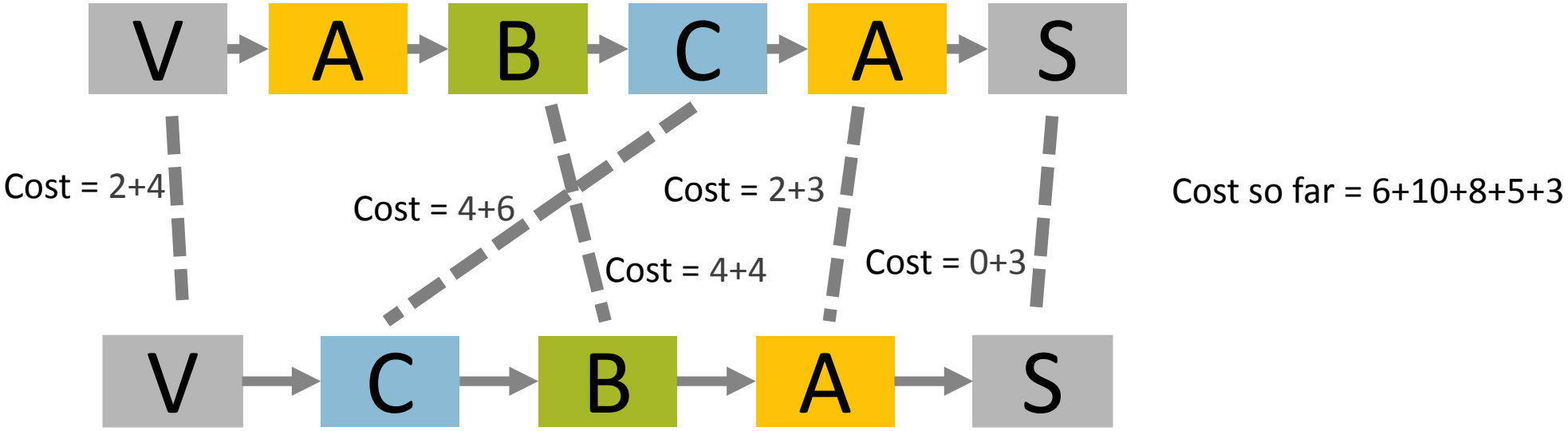


Quantify Similarity of Mapped Events

… based on “structural context of events”

(1) Differences in neighbors

(2) Differences in structure



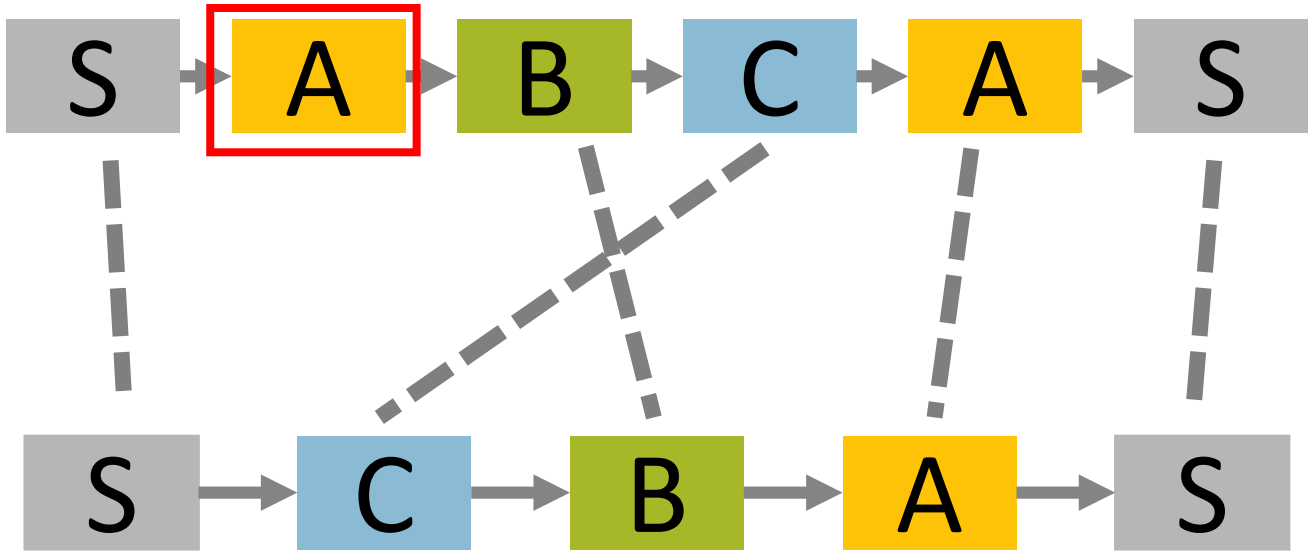
Quantify Similarity of Mapped Events

… based on “structural context of events”

(1) Differences in neighbors

(2) Differences in structure

(3) #Dissimilar events

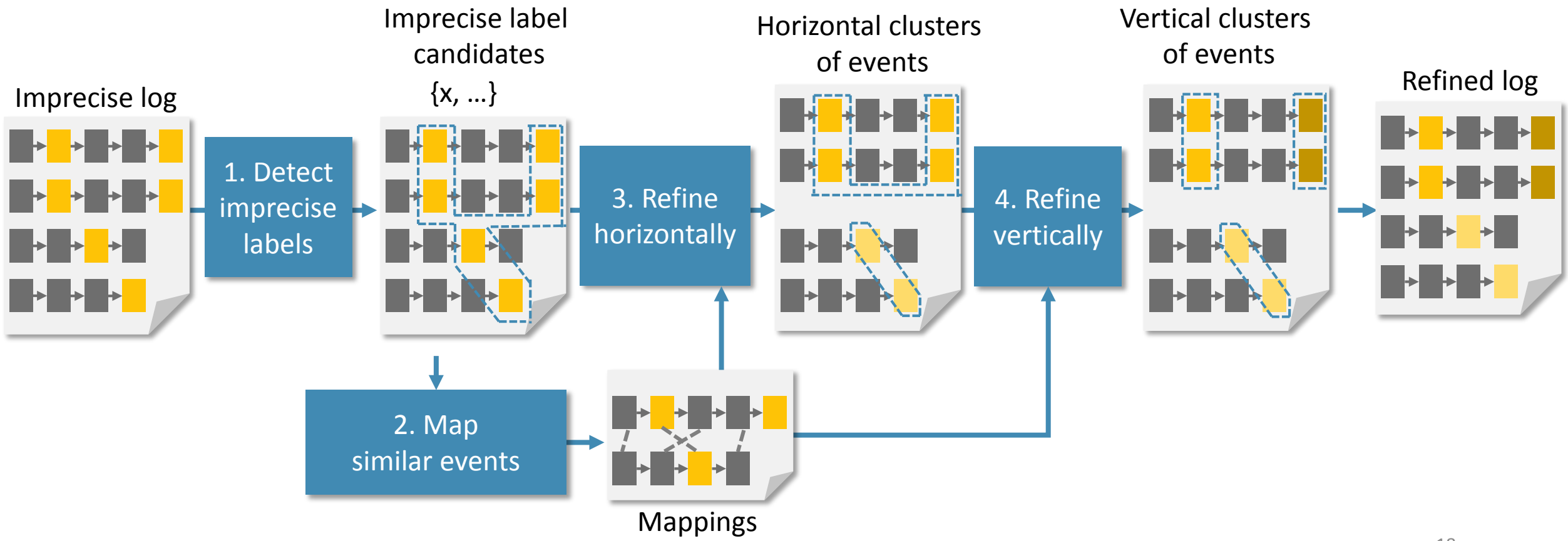


Total Cost = 6+10+8+5+3 +1

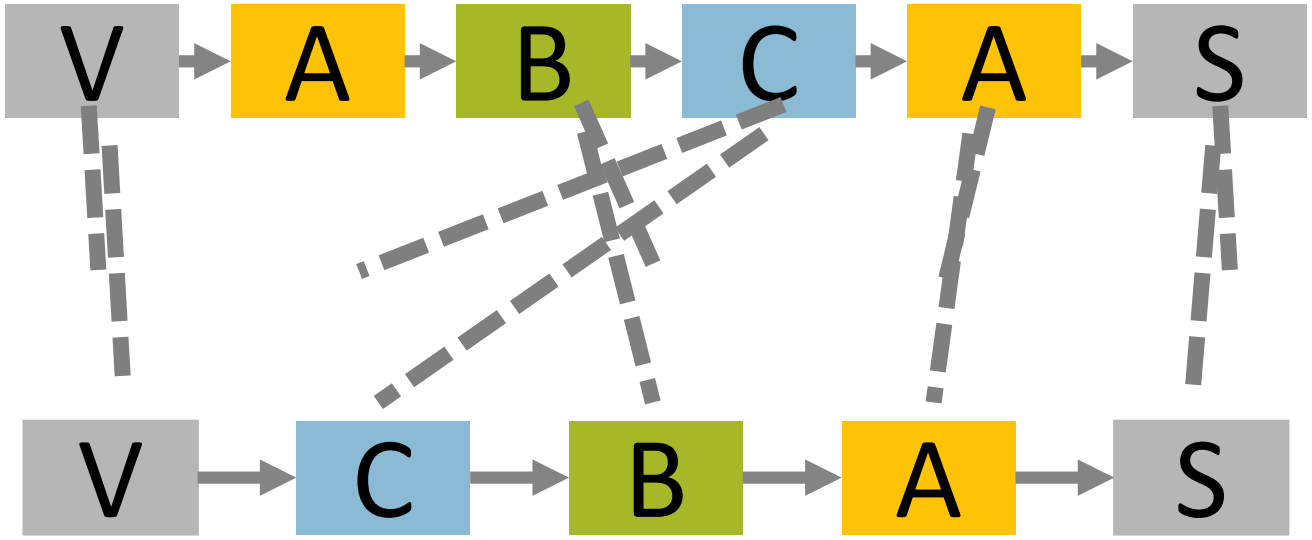
There is an algorithm to compute an optimal mapping :

Xixi Lu, Dirk Fahland, Frank J.H.M. van den Biggelaar, and Wil M.P. van der Aalst. **Detecting Deviating Behavior without Models**. In *BPM Workshops* 2015, volume 256, pp.126–139, Springer International Publishing, 2015.

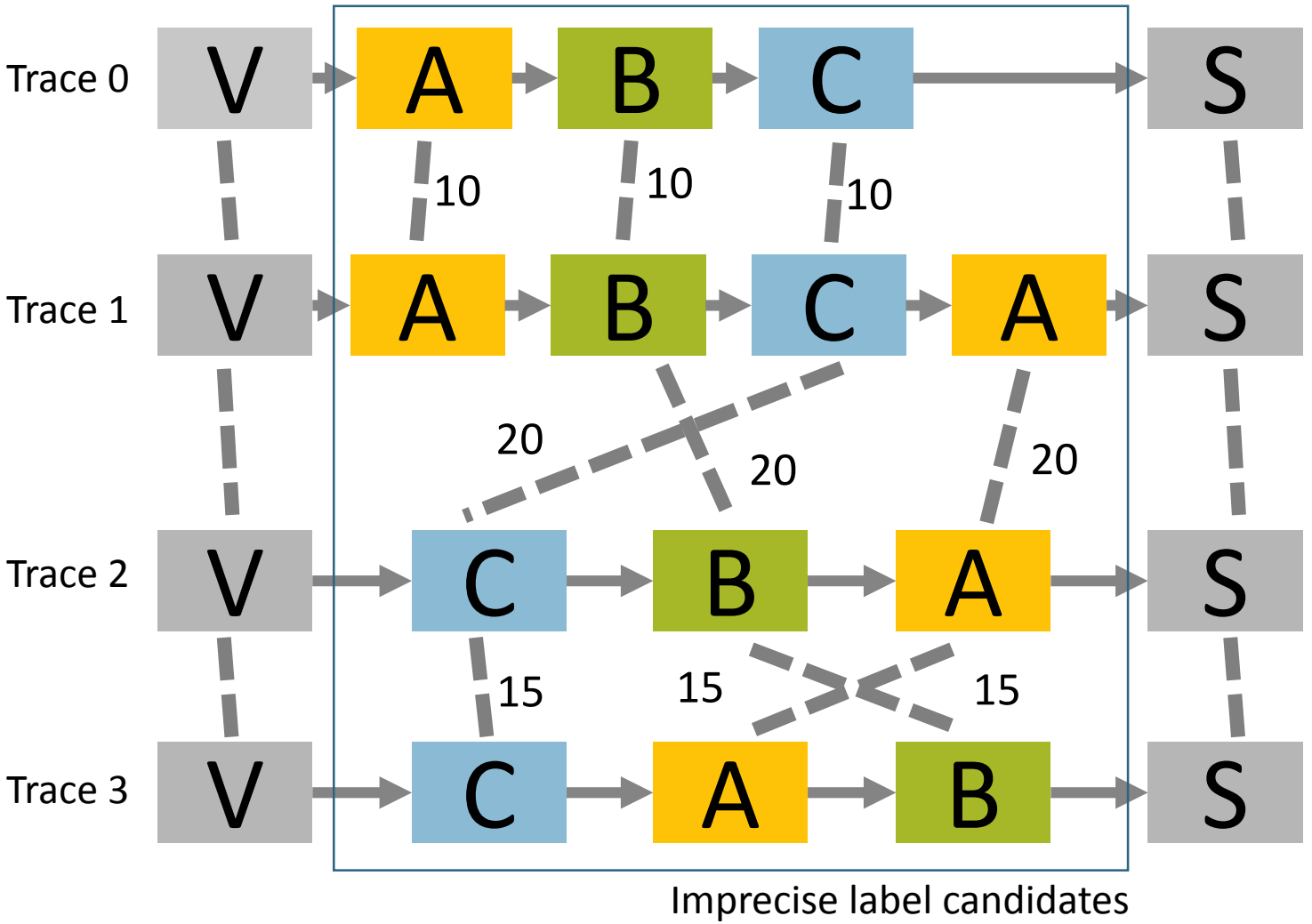
Approach


Imprecise log
1. Detect imprecise labels
Imprecise label candidates
{x, …}
2. Map similar events
Mappings
3. Refine horizontally
Horizontal clusters of events
4. Refine vertically
Vertical clusters of events
Refined log

3. Refine horizontally using cost

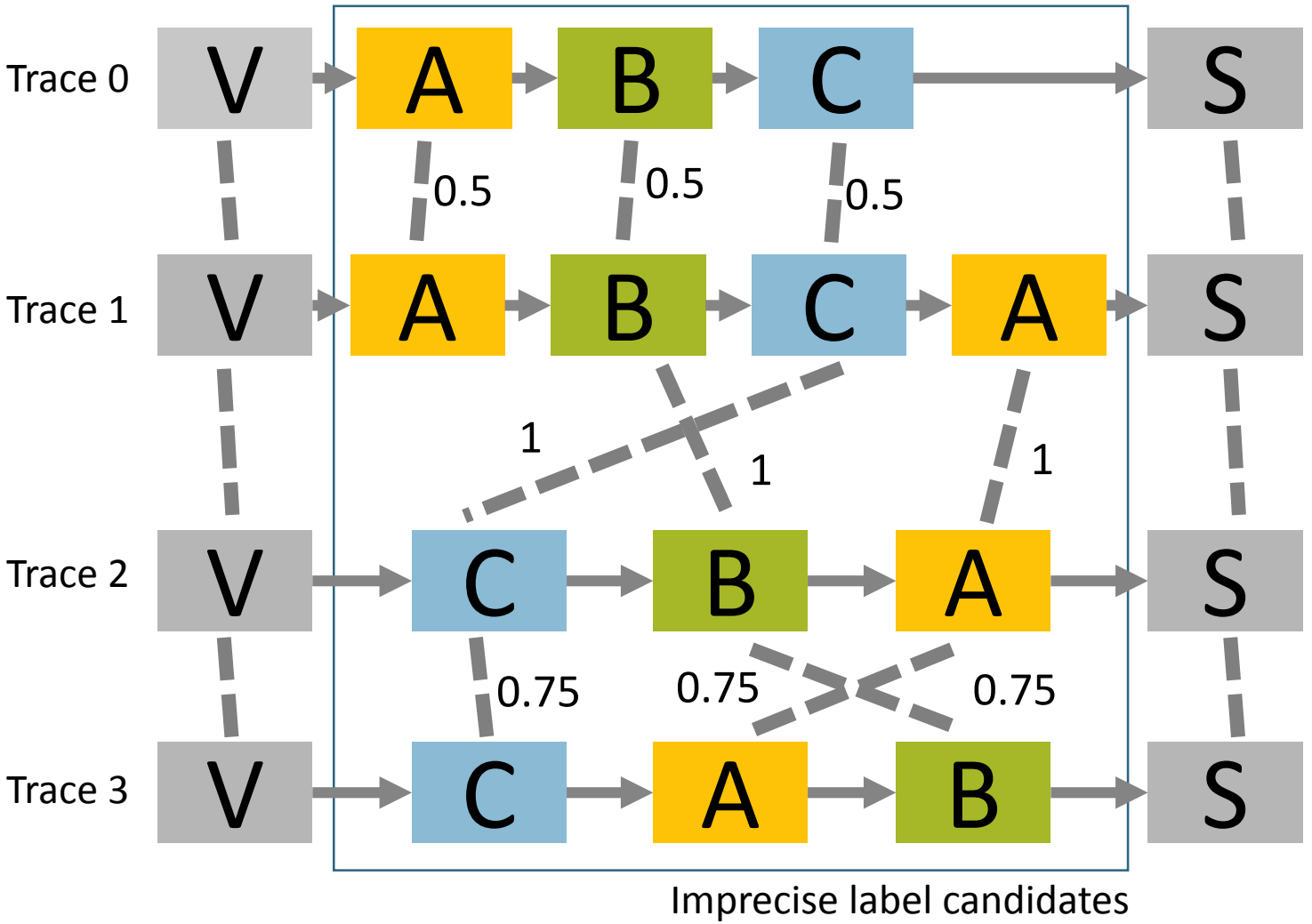


3. Refine horizontally using cost



a) Normalize costs w.r.t. maximal cost seen in the log

3. Refine horizontally using cost

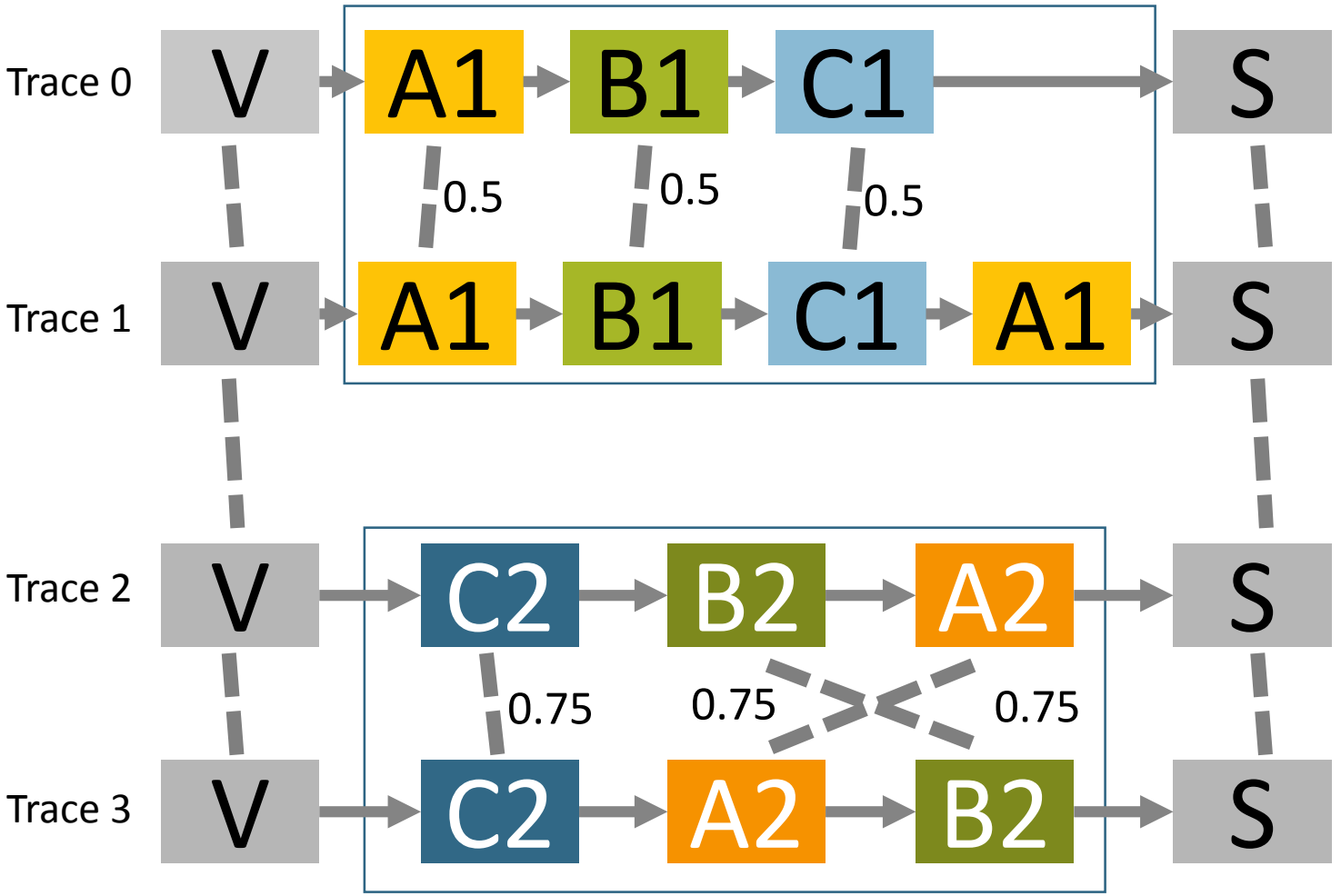


a) Normalize costs w.r.t. maximal cost seen in the log

b) Set variant threshold,
say, 0.8

c) Remove edges
if cost > variant threshold

3. Refine horizontally using cost

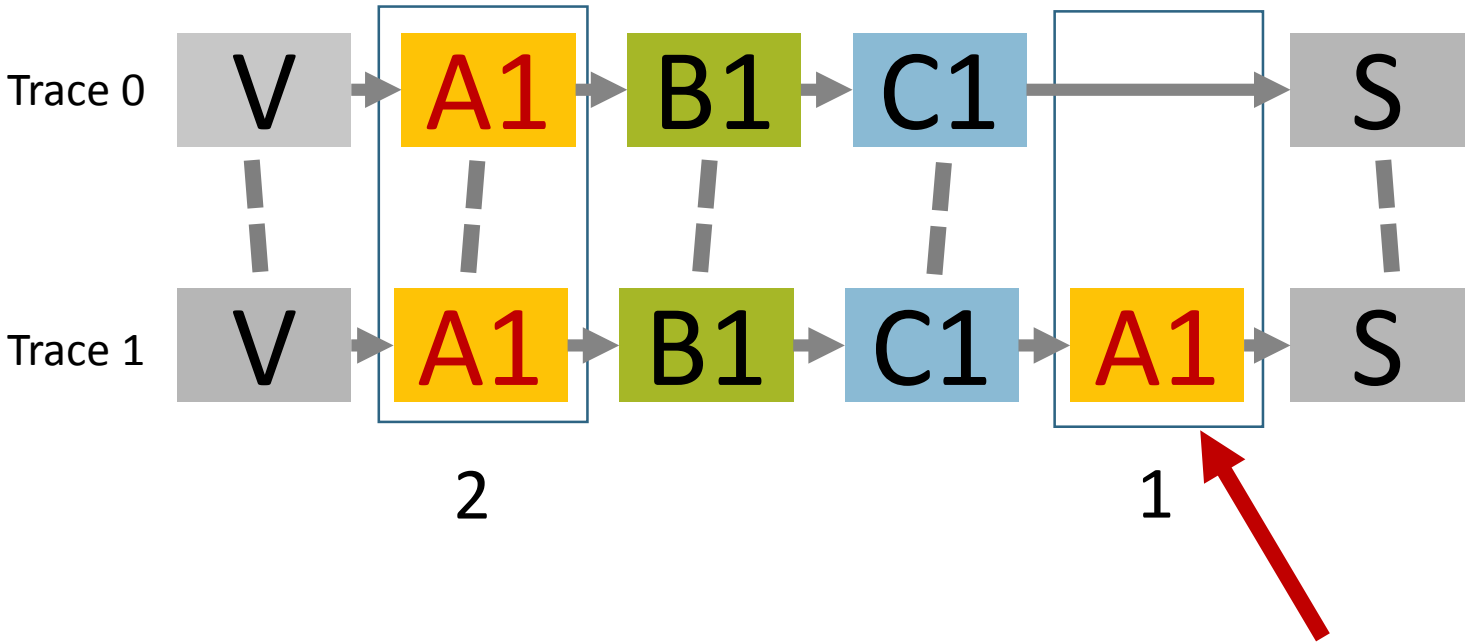


a) Normalize costs w.r.t. maximal cost seen in the log

b) Set variant threshold say, 0.8

c) Remove edges if cost > variant threshold

4. Refine vertically using frequency

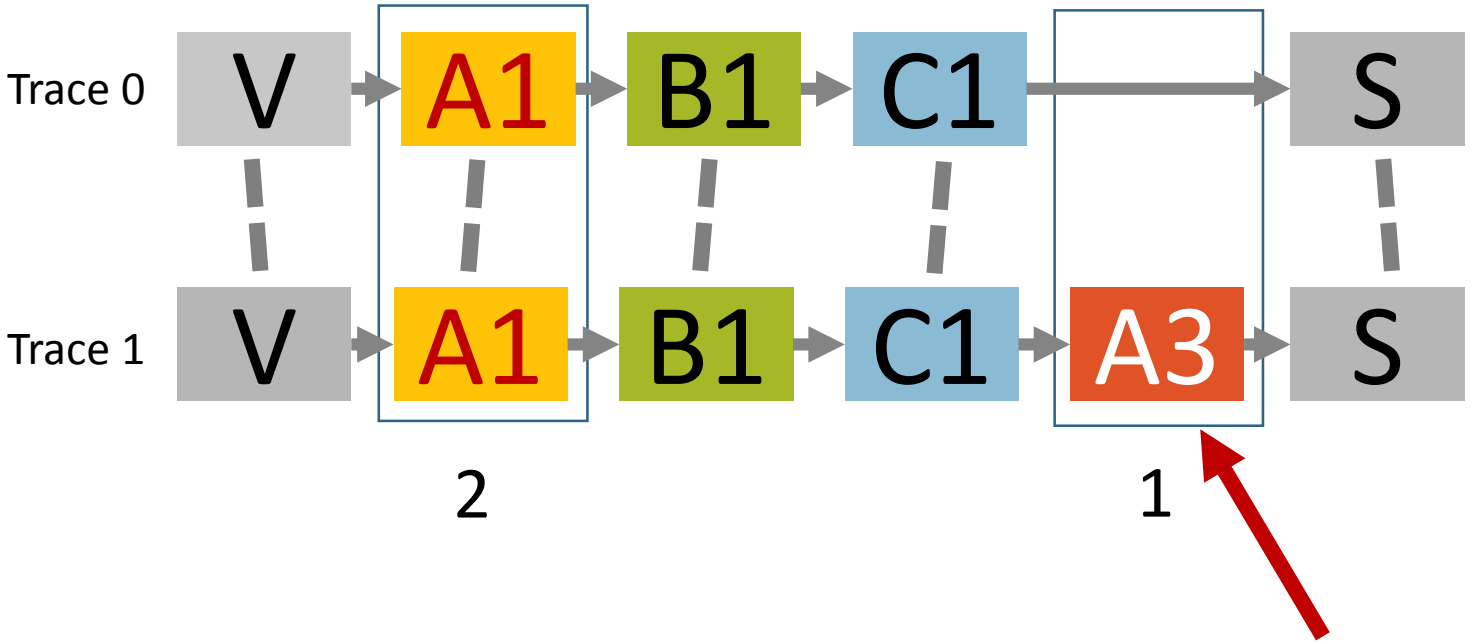


I) Set unfolding threshold,
say, 60%

II) Refine labels
if freq > unfolding threshold

Is this A1 a 2nd iteration of a loop?

4. Refine vertically using frequency

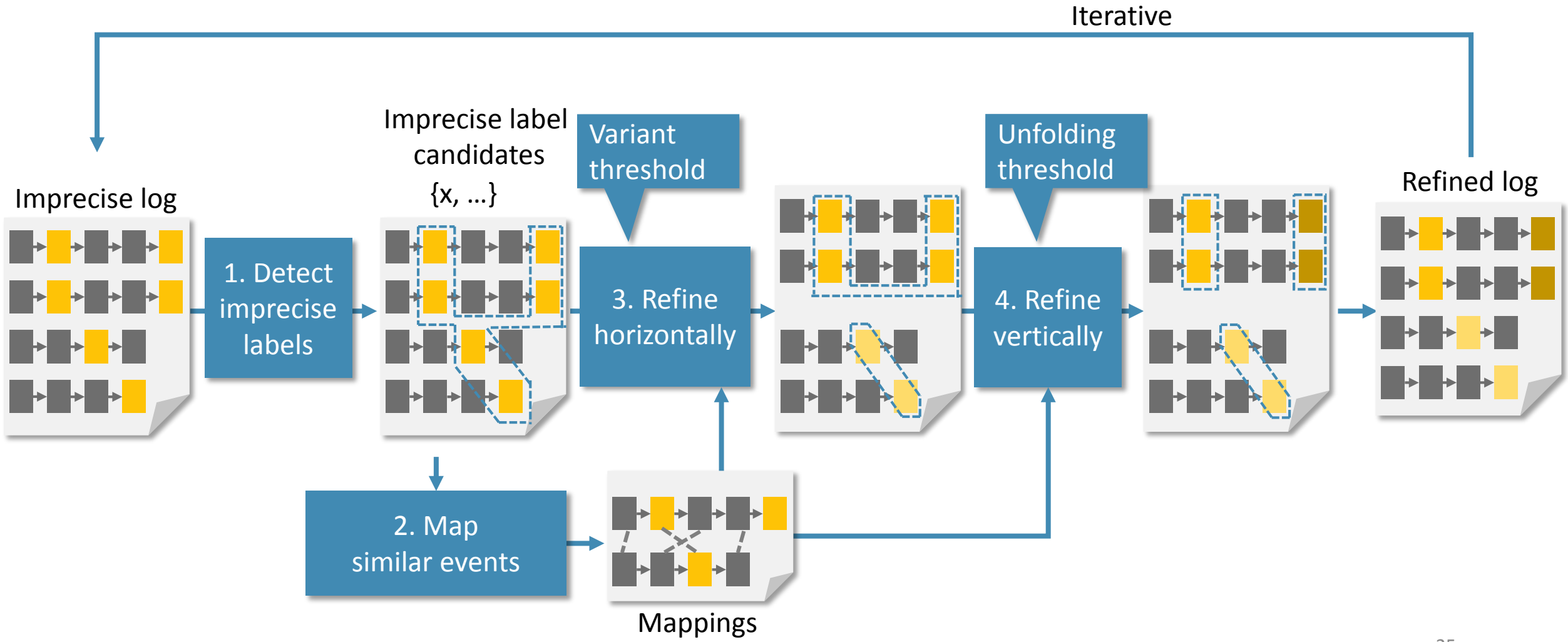


I) Set unfolding threshold,
say, 40%

II) Refine labels
if freq > unfolding threshold

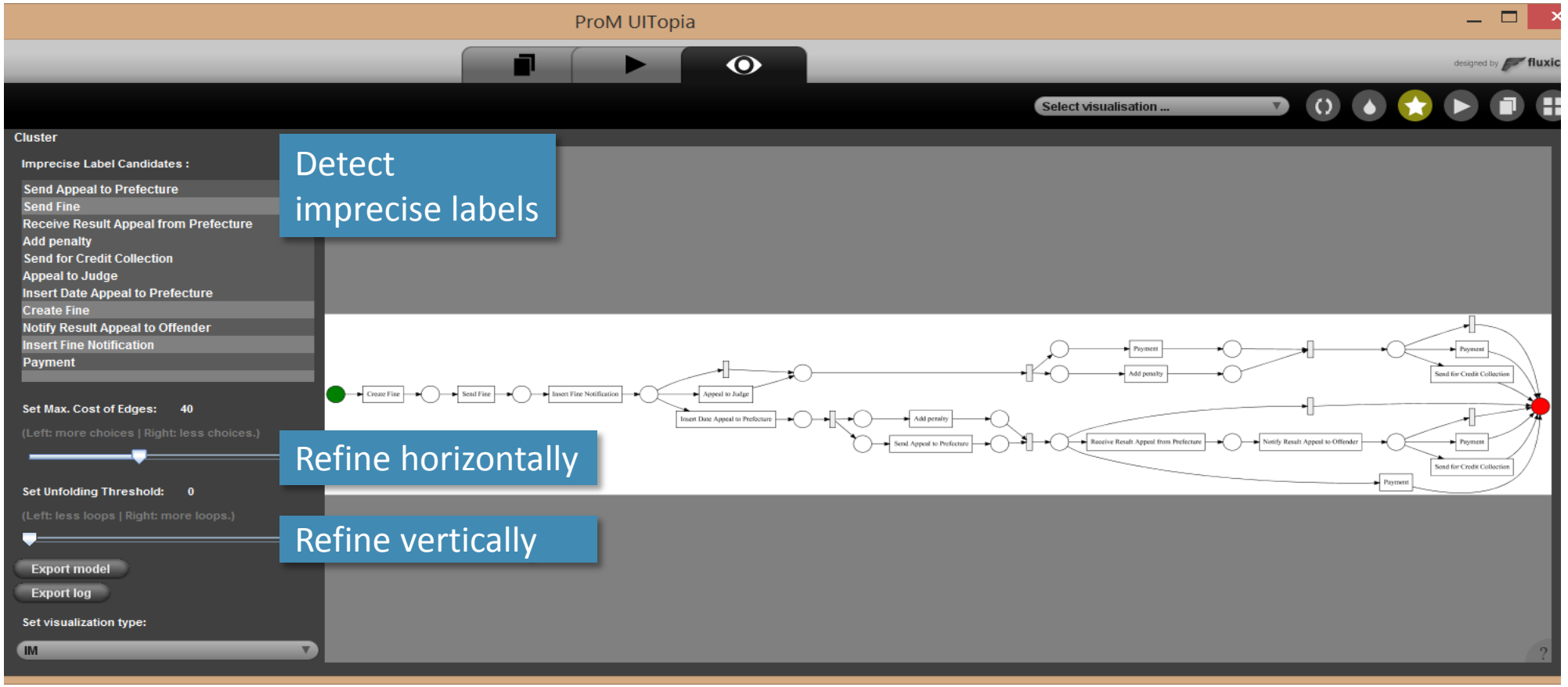
Or is this a different task?

Approach


Iterative
Imprecise log
1. Detect imprecise labels
Imprecise label candidates
{x, …}
2. Map similar events
Variant threshold
3. Refine horizontally
Mappings
Unfolding threshold
4. Refine vertically
Refined log

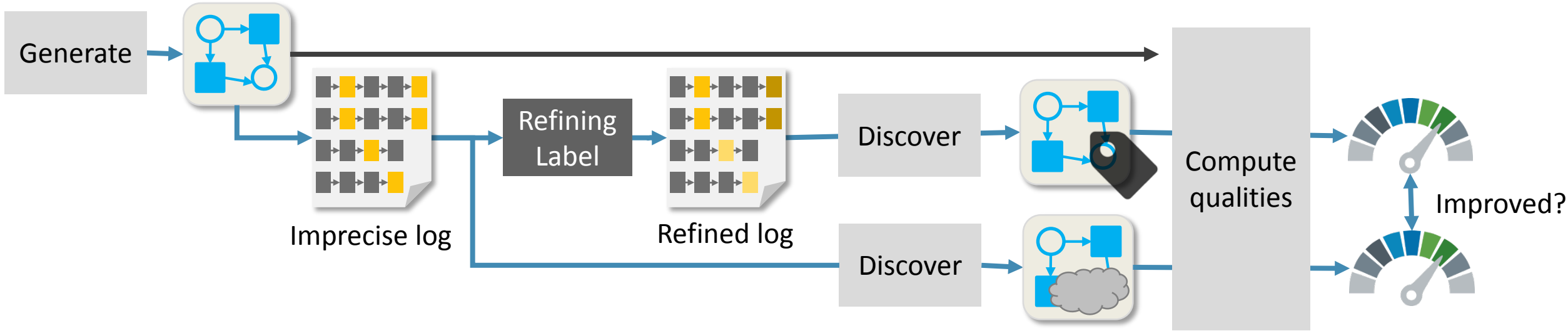
Implemented in ProM

- There will be a demo in the demo track.

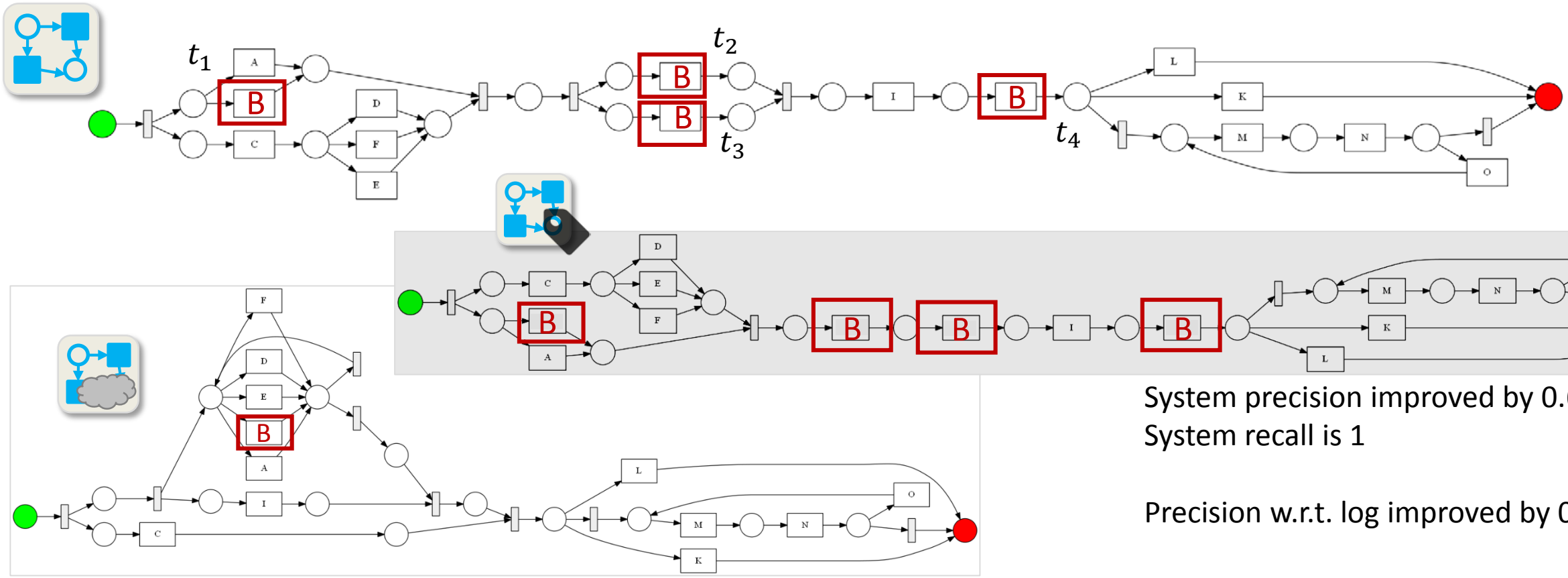


Evaluation

- 3 experiments, each 600 models, k transitions w/ same label
  - E1) Default parameters, k = 4
  - E2) Adaptive parameters, k = 4
  - E3) 1 in loop, k = 2

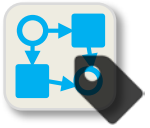
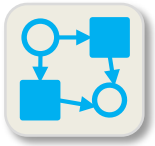


Experiment Result - Example

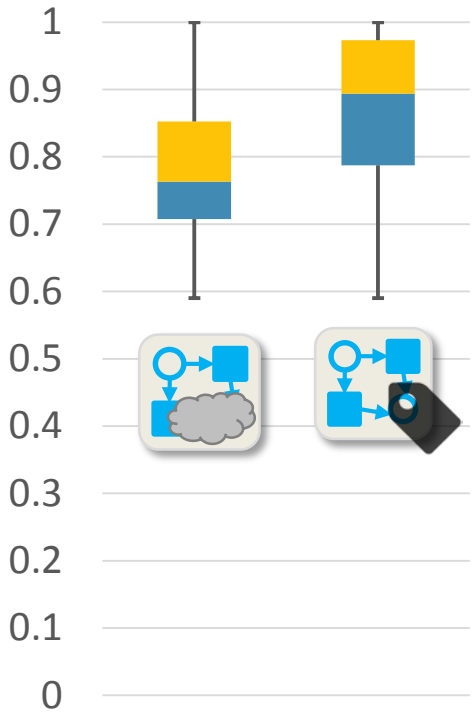


System precision improved by 0.68
System recall is 1

Precision w.r.t. log improved by 0.55

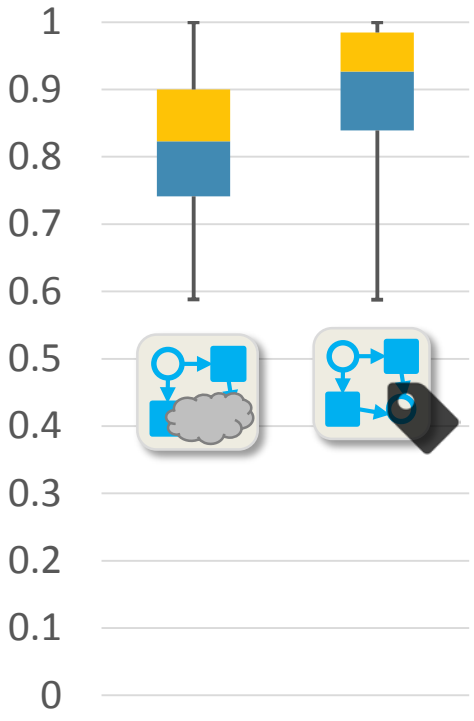
Experiment Result - F1-score of  and  w.r.t. 

E1) Default



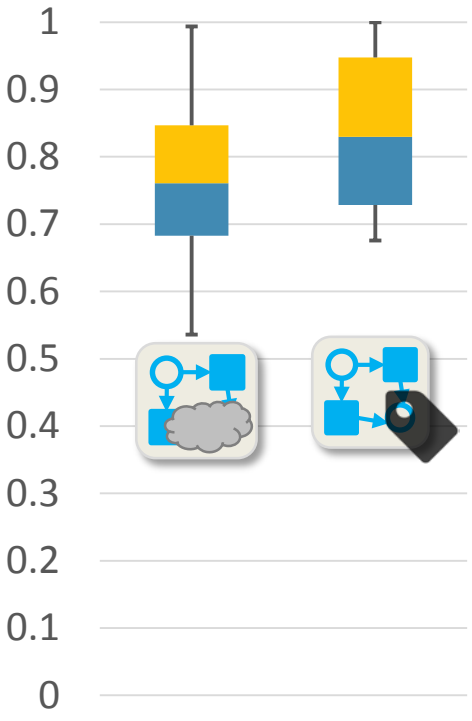
Improved 35% ~ 46% Models

E2) Adaptive



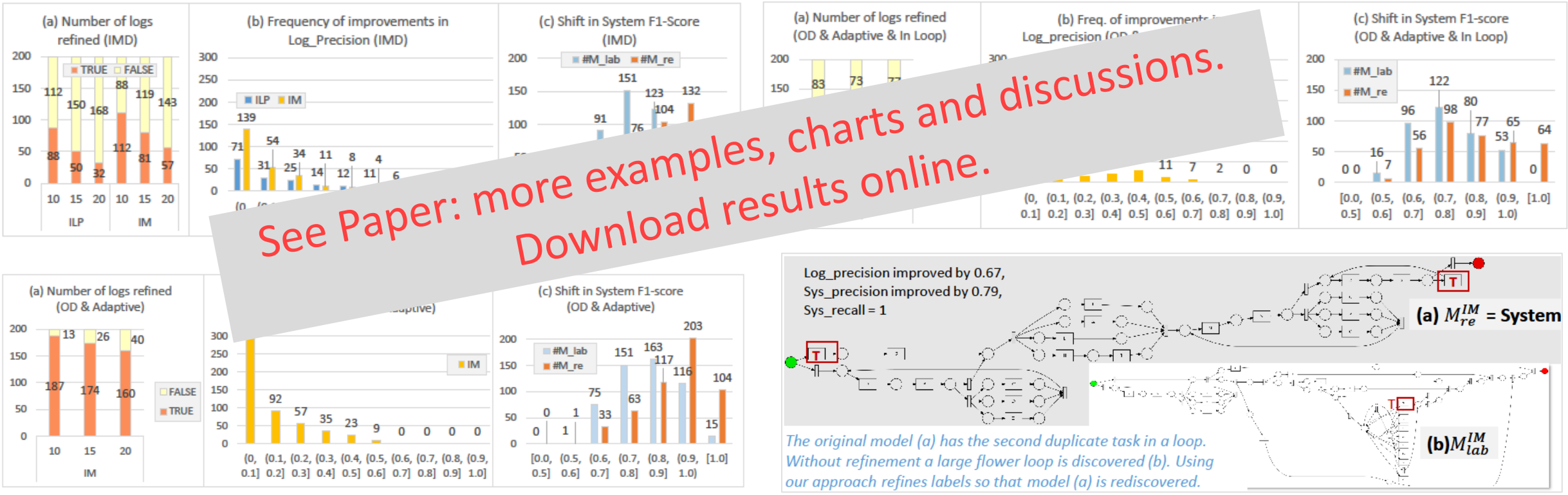
Improved 89% Models

E3) 1 in loop



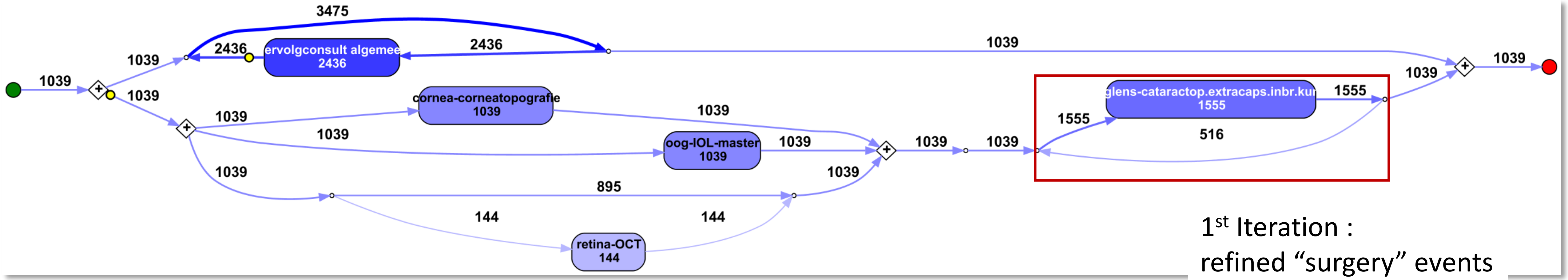
Improved 60% Models

Experiment Results and Limitations

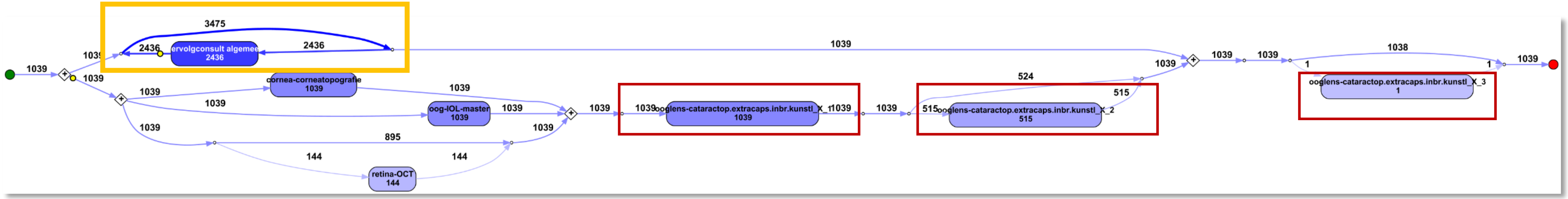


The original model (a) has the second duplicate task in a loop. Without refinement a large flower loop is discovered (b). Using our approach refines labels so that model (a) is rediscovered.

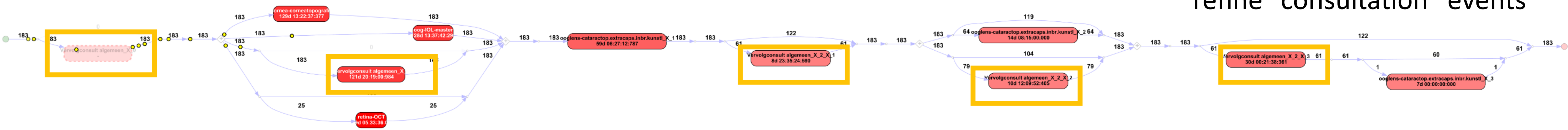
Real-life Example : Hospital Data



1st Iteration :
refined "surgery" events



2nd & 3rd Iterations:
refine "consultation" events



Conclusion and Future Work

- Improved logs (and process discovery) by refining labels for up to 87%
- Interactive and explorative

- Future work
  - Different ways to compute similar events using context of events
  - Log preprocessing framework

- Integrated in “Log to Model Explorer”
  - Supporting clustering, filtering and label refining
- Come to the Demo!

Questions?