



Master Thesis

Model Repair by Incorporating Negative Instances in Process Enhancement

Author: Kefang Ding

Email: kefang.ding@rwth-aachen.de

Supervisor: Dr. Sebastiaan J. van Zelst

Examiners: Prof. Wil M.P. van der Aalst
Prof. Thomas Rose

Institute: Lehrstuhl Informatik 9
Process and Data Science

Master Thesis







Master Thesis

Model Repair by Incorporating Negative Instances in Process Enhancement

Author: Kefang Ding

Email: kefang.ding@rwth-aachen.de

Supervisor: Dr. Sebastiaan J. van Zelst

Examiners: Prof. Wil M.P. van der Aalst
Prof. Thomas Rose

Institute: Lehrstuhl Informatik 9
Process and Data Science

Outline

- Motivation
 - Problem Definition
 - Approach
 - Demo
 - Evaluation
 - Conclusion

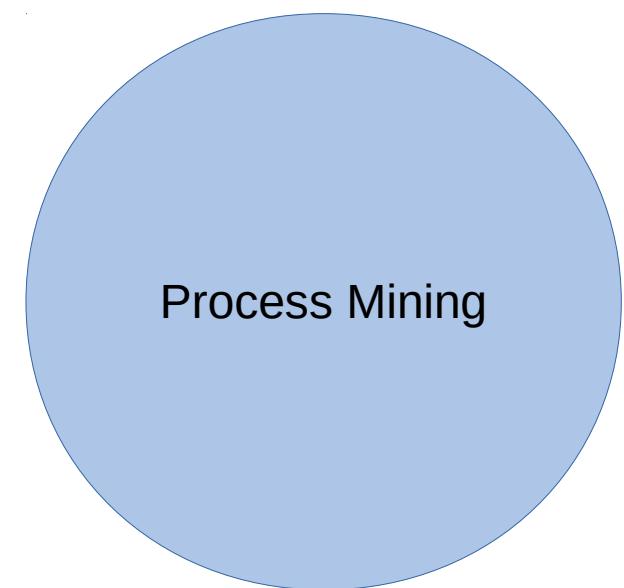
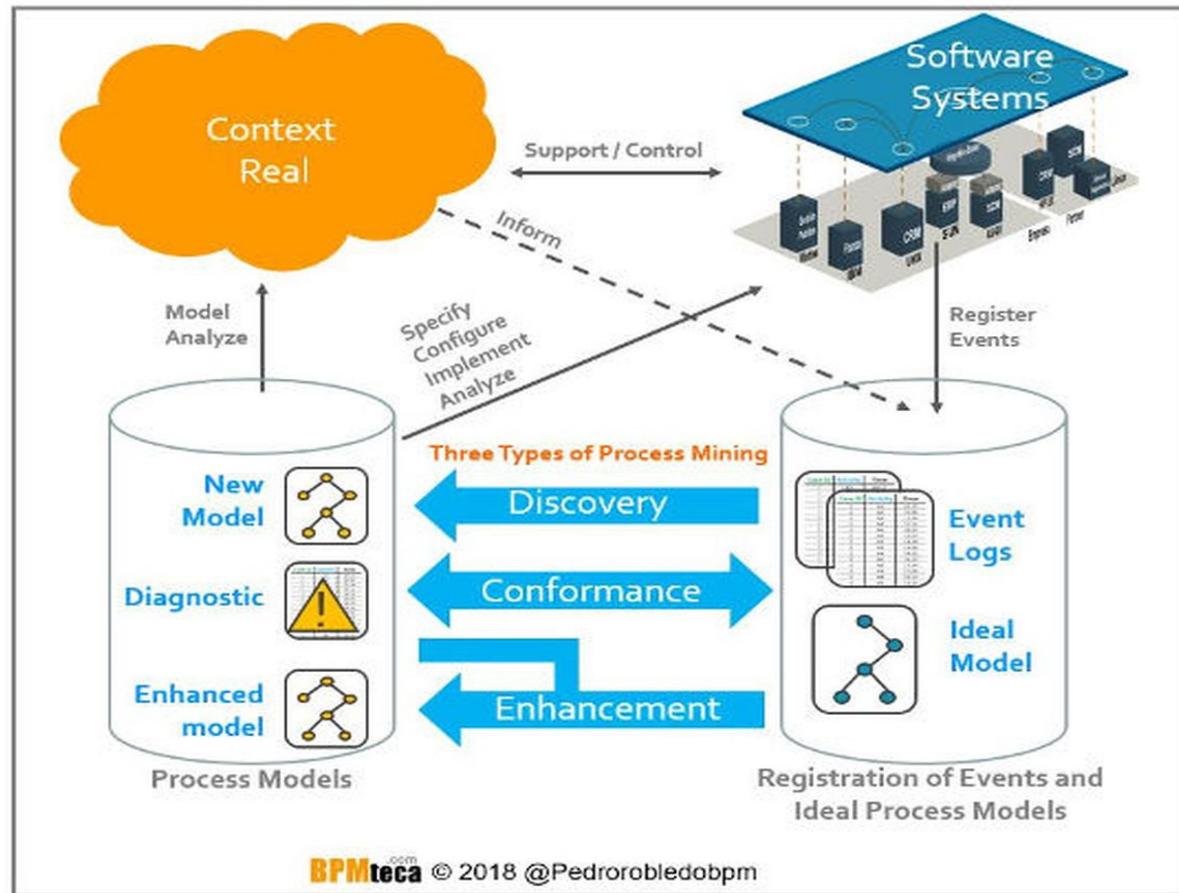


Outline

- **Motivation**
 - Scope
 - Related work
 - Motivating examples
 - **Problem Definition**
 - **Approach**
 - **Demo**
 - **Evaluation**
 - **Conclusion**



Scope

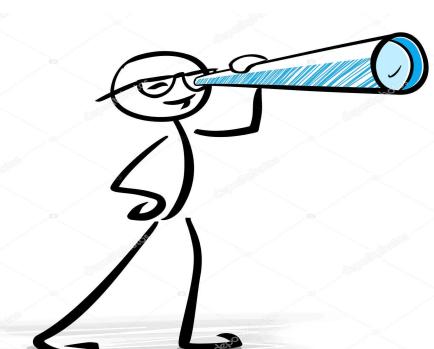
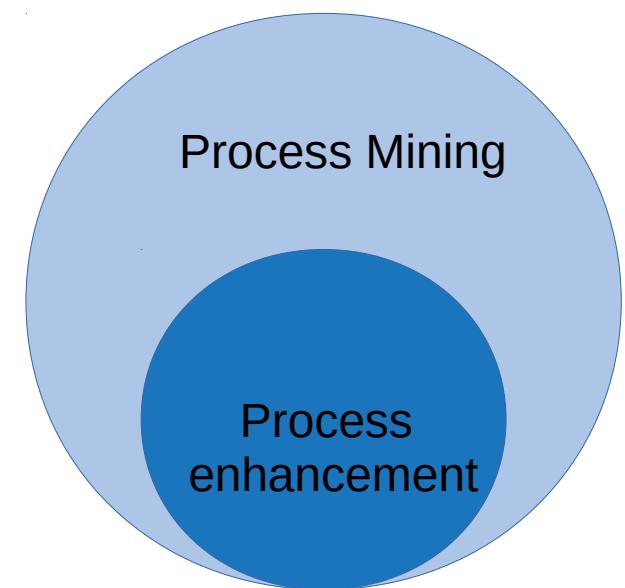
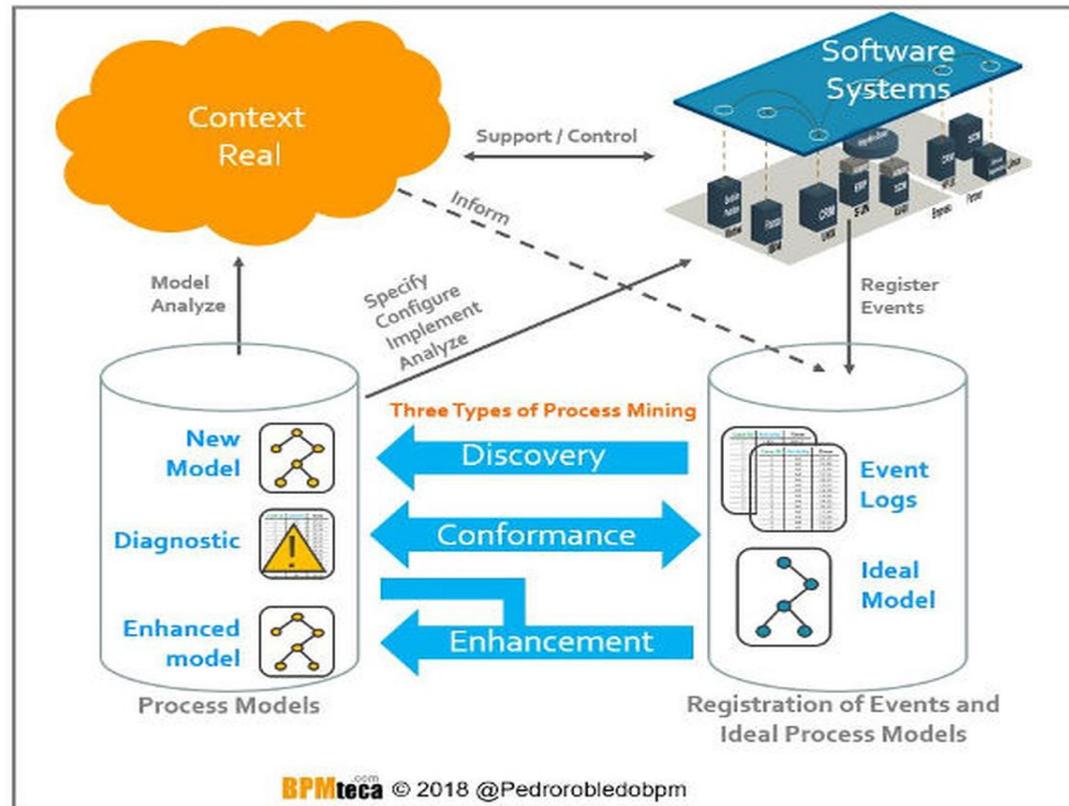


<https://medium.com/@pedrorobledobpm/process-mining-plays-an-essential-role-in-digital-transformation-384839236bbe>

May 29, 2019

7

Scope

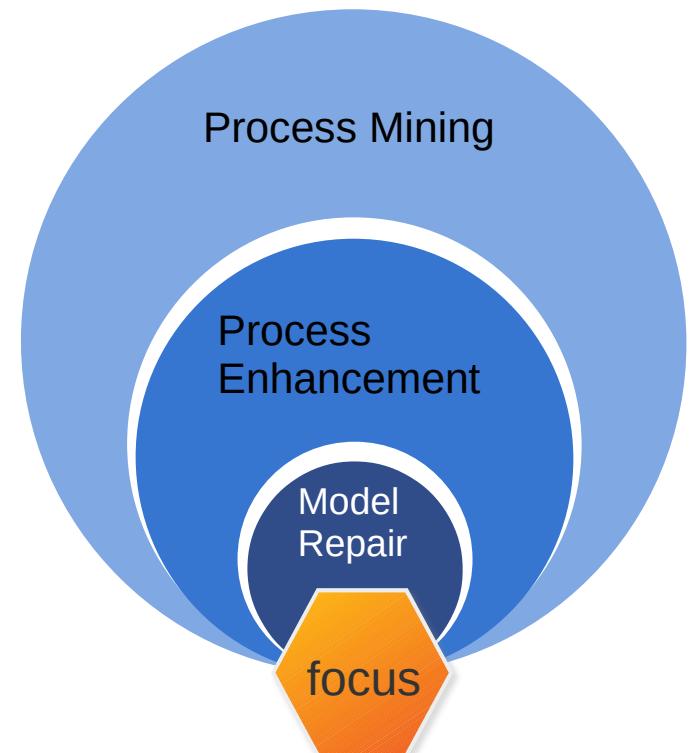
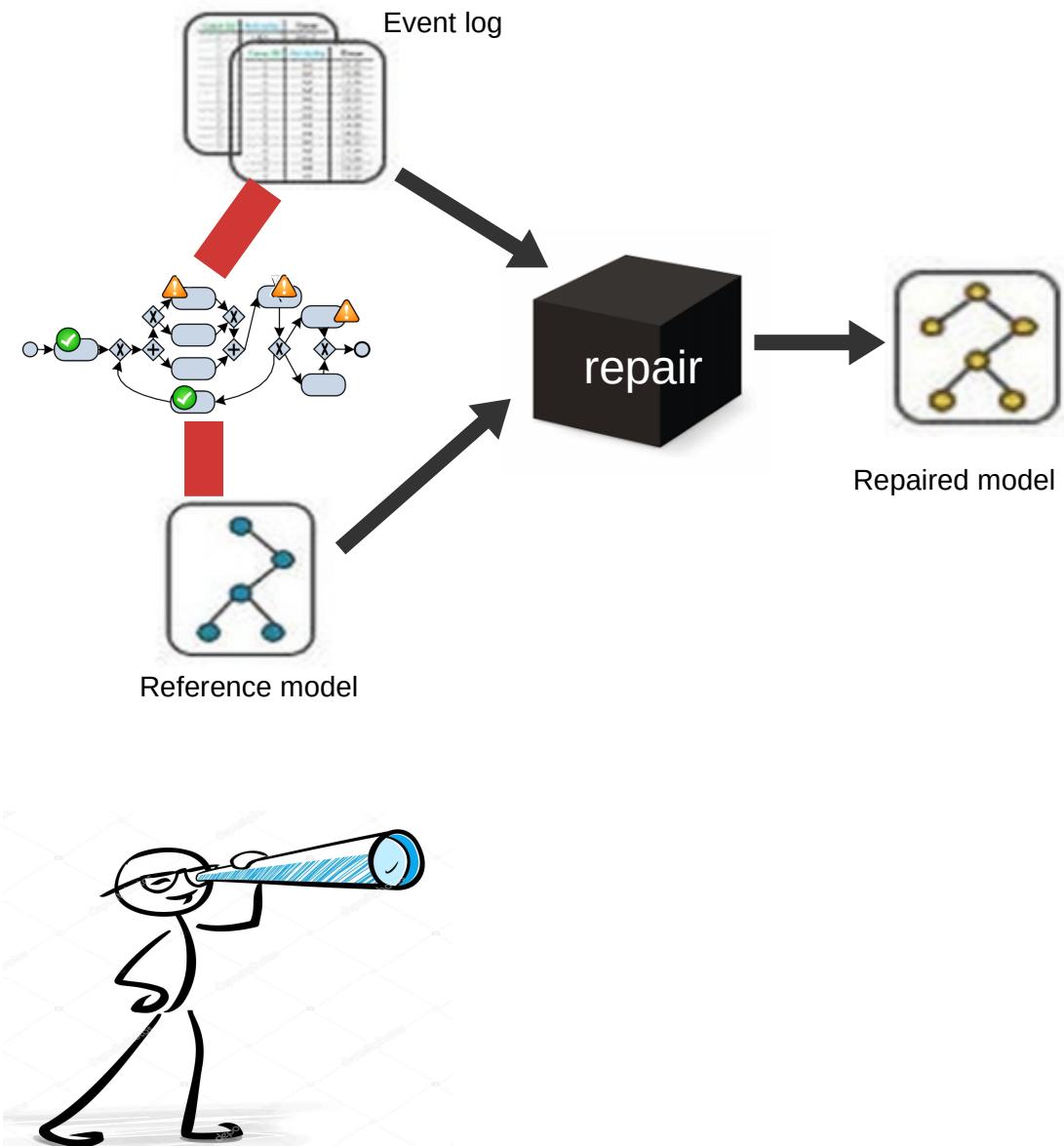


<https://medium.com/@pedrorobledobpm/process-mining-plays-an-essential-role-in-digital-transformation-384839236bbe>

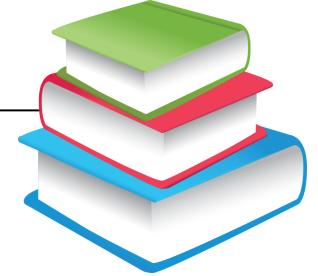
May 29, 2019

8

Scope – model repair

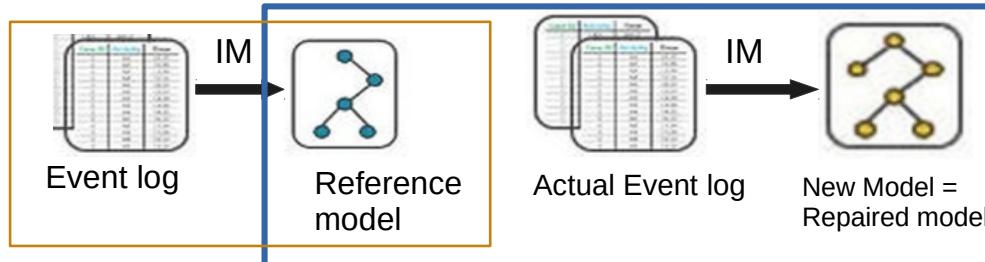


Enhancement related work



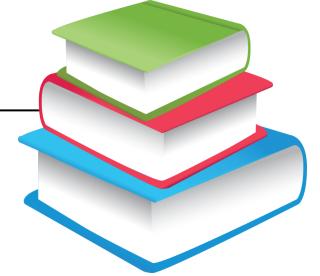
- Rediscovery

Process discovery

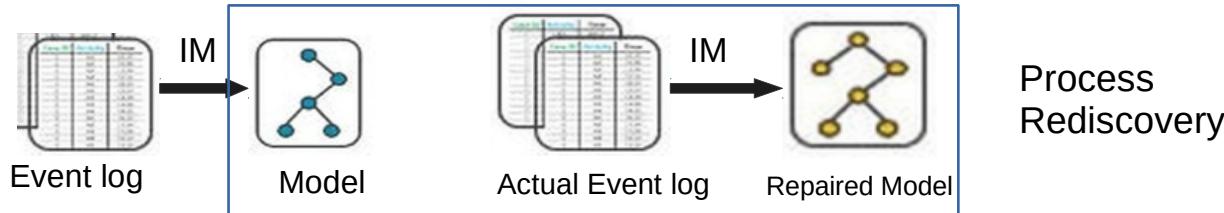


Process Rediscovery

Related Work

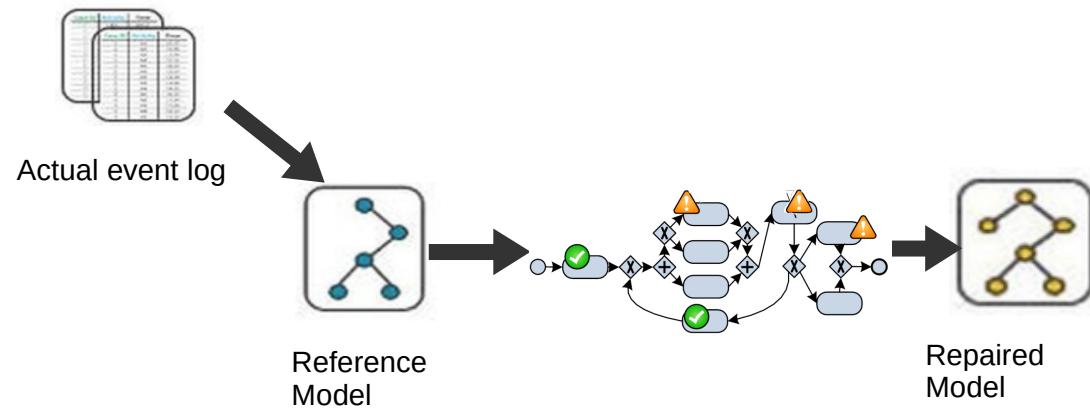


- **Rediscovery**

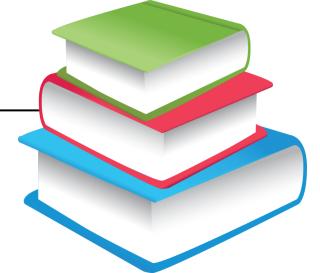


- **Model Repair by Fahland**

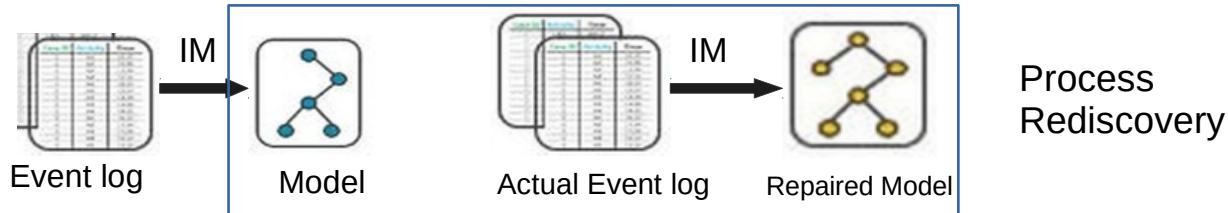
- Deviations
- Subprocesses



Related Work



- **Rediscovery**

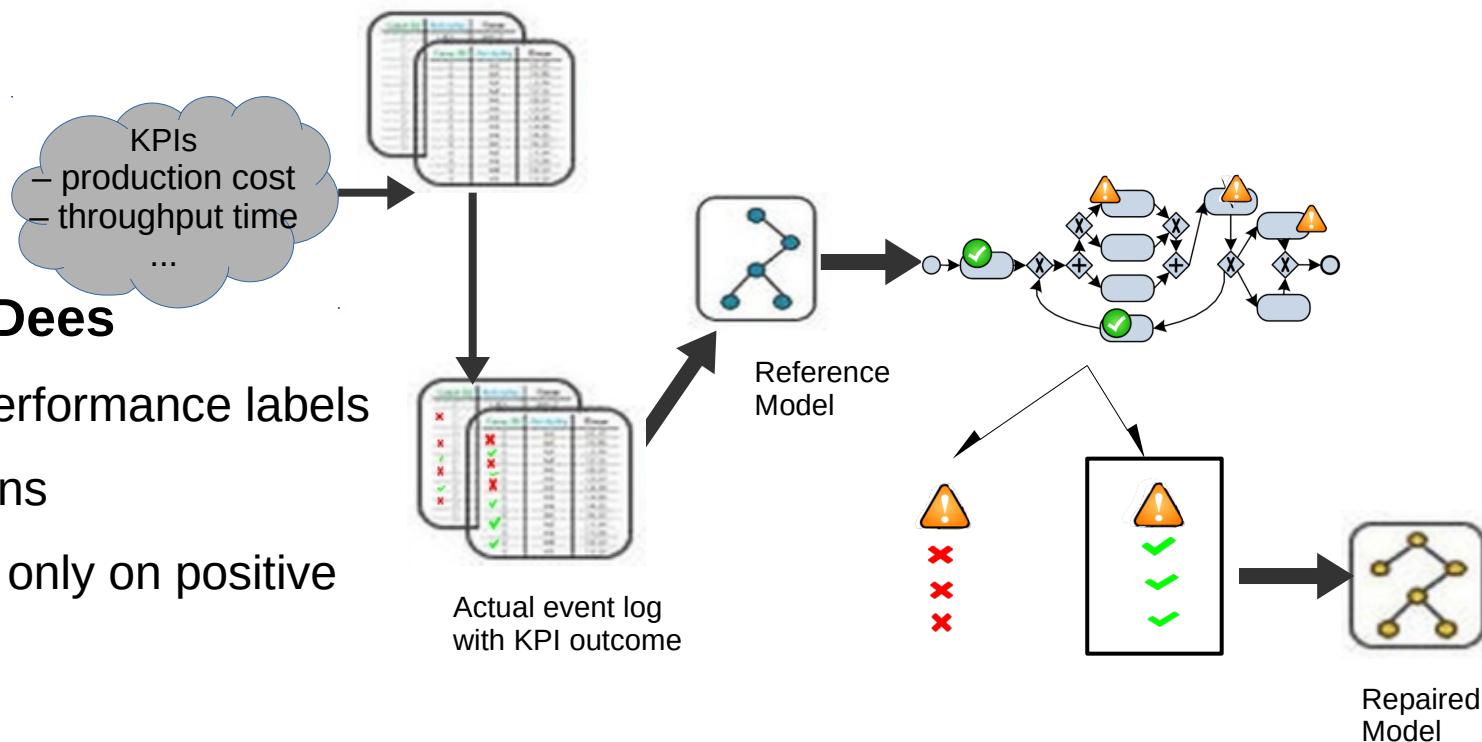


- **Model Repair by Fahland**

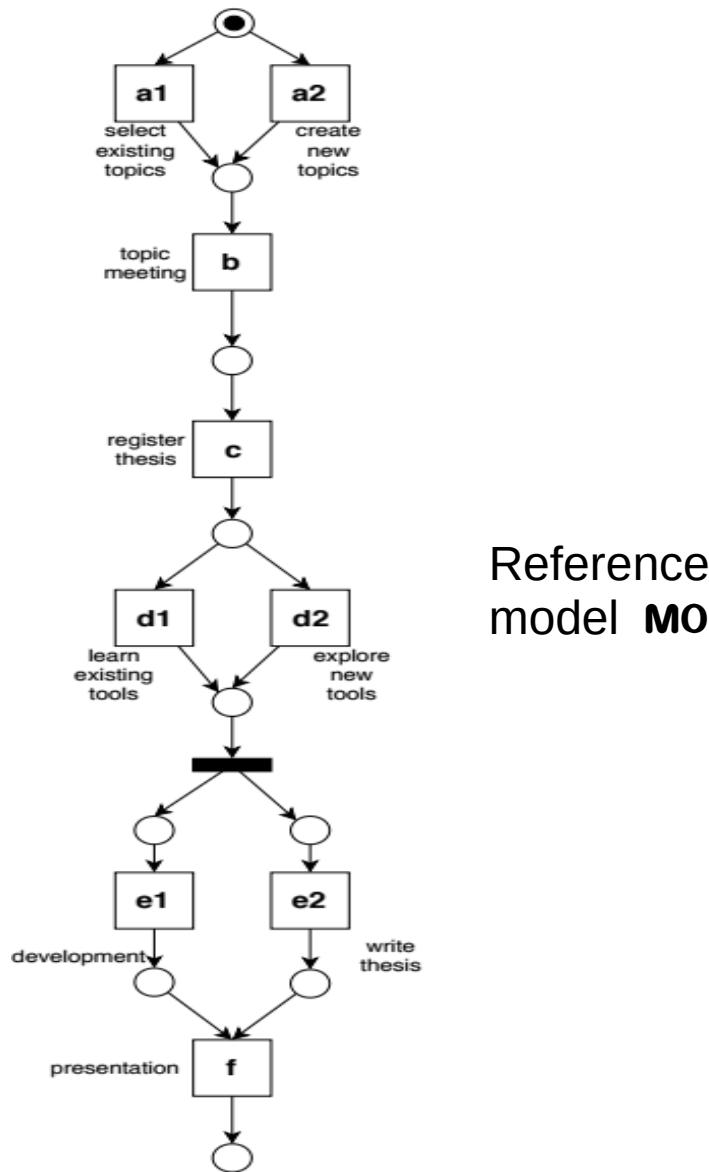
- Deviations
- Subprocesses

- **Model Repair by Dees**

- Event log with performance labels
- Classify deviations
- Fahland's repair only on positive deviations



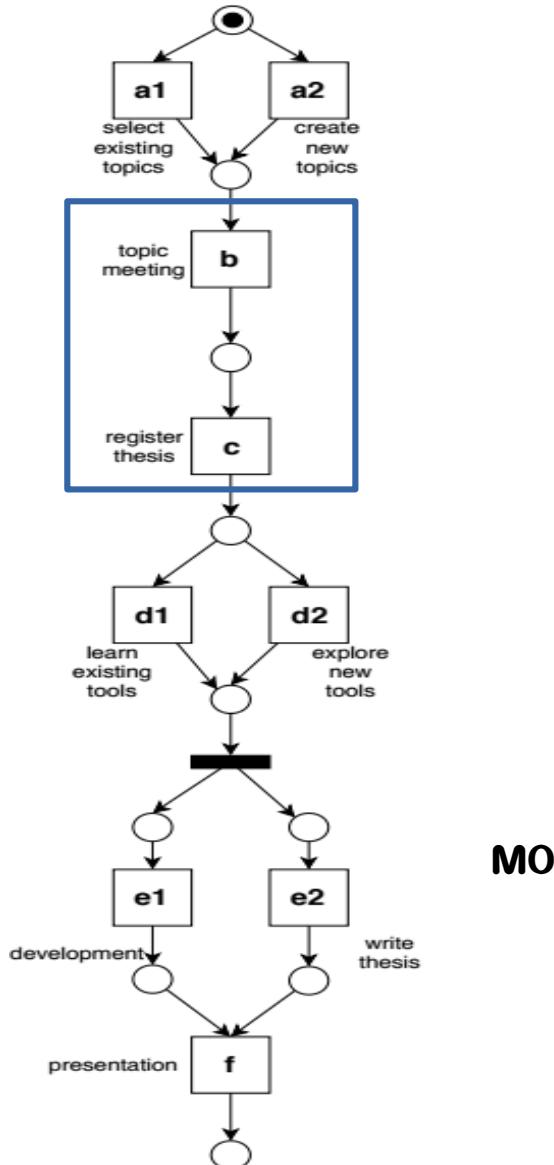
Motivation – reference model



Motivation – shortcomings

x1: write propose
x2: check course requirement

$$L_1 := \{< a1, b, \boxed{x1}, c, d1, e1, e2, f >^{50, pos}, \quad \checkmark \\ < a1, b, \boxed{x2}, c, d2, e1, e2, f >^{50, pos} \} \quad \checkmark$$



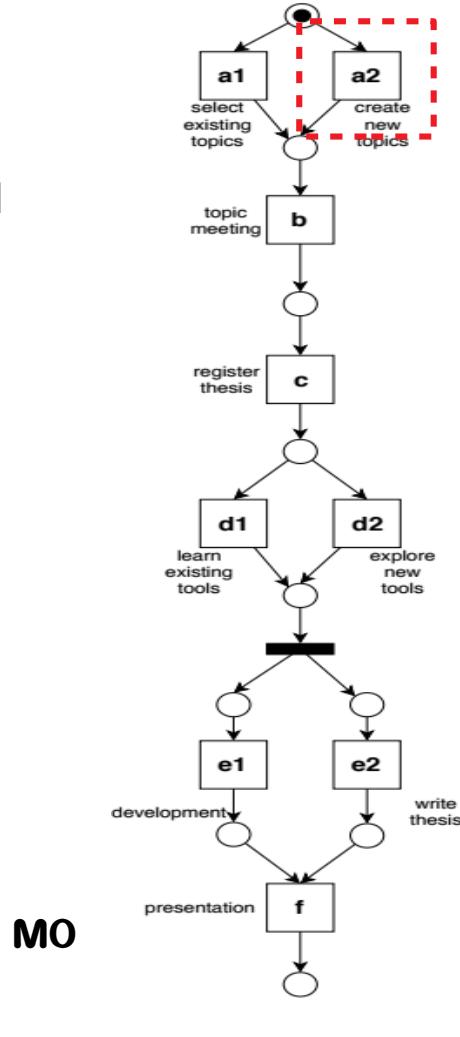
Motivation – shortcomings

$$L_1 := \{< a1, b, \mathbf{x1}, c, d1, e1, e2, f >^{50, pos}, \\ < a1, b, \mathbf{x2}, c, d2, e1, e2, f >^{50, pos}, \}$$

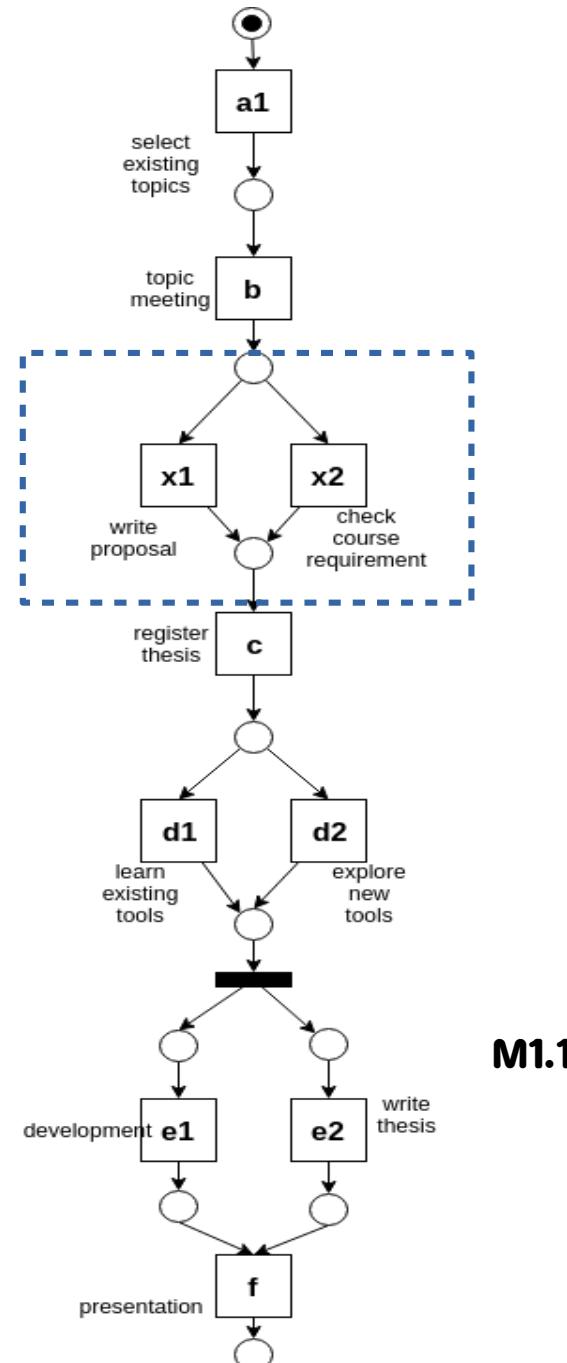
- IM not consider reference model



Similarity decreases!



M0



M1.1

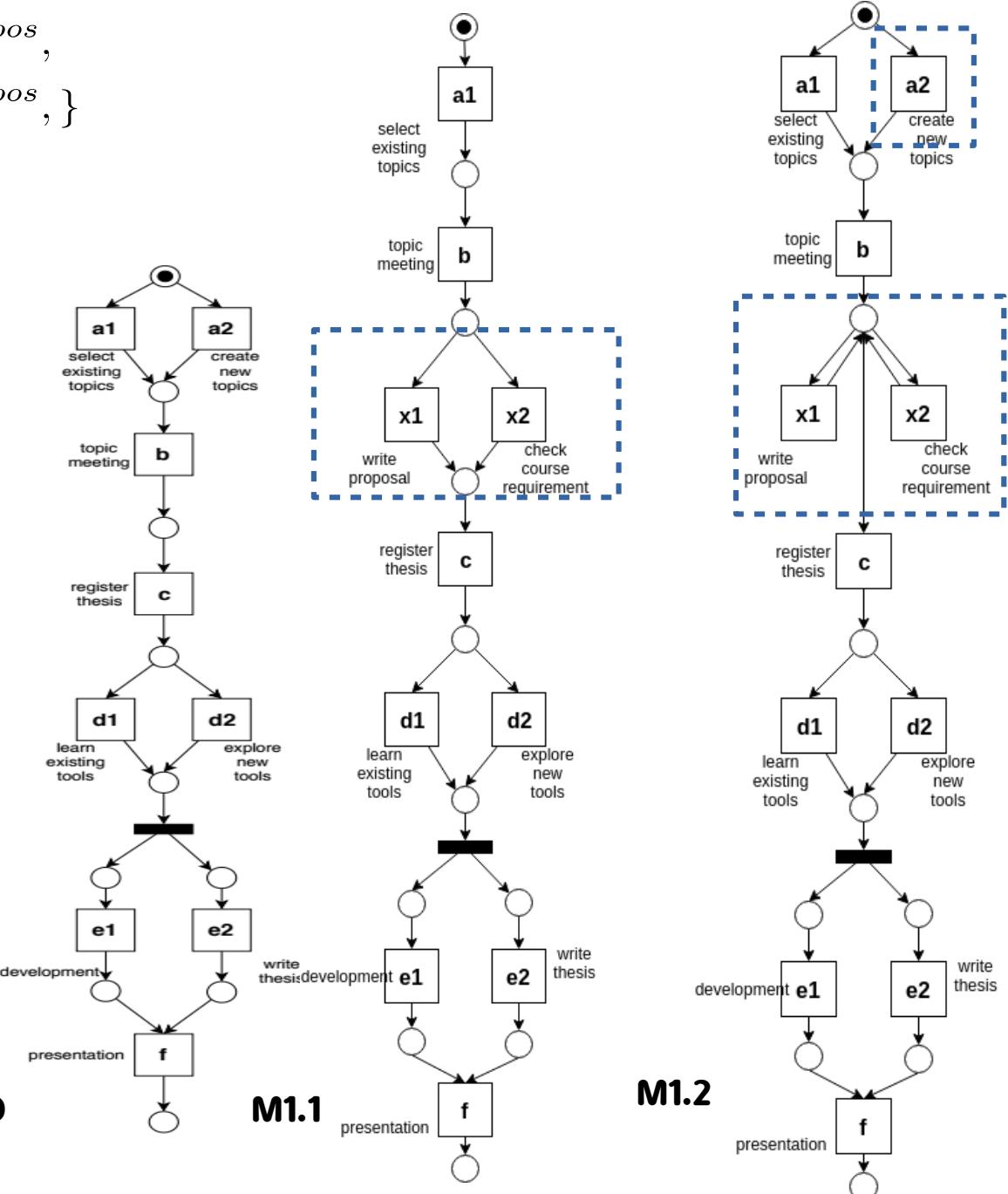
Motivation – shortcomings

$$L_1 := \{ < a1, b, \mathbf{x1}, c, d1, e1, e2, f >^{50, pos}, \\ < a1, b, \mathbf{x2}, c, d2, e1, e2, f >^{50, pos}, \}$$

- Fahland's:** add subprocesses as loops
- Dee's:** same as Fahland's method



Precision decrease by adding subprocess in loops

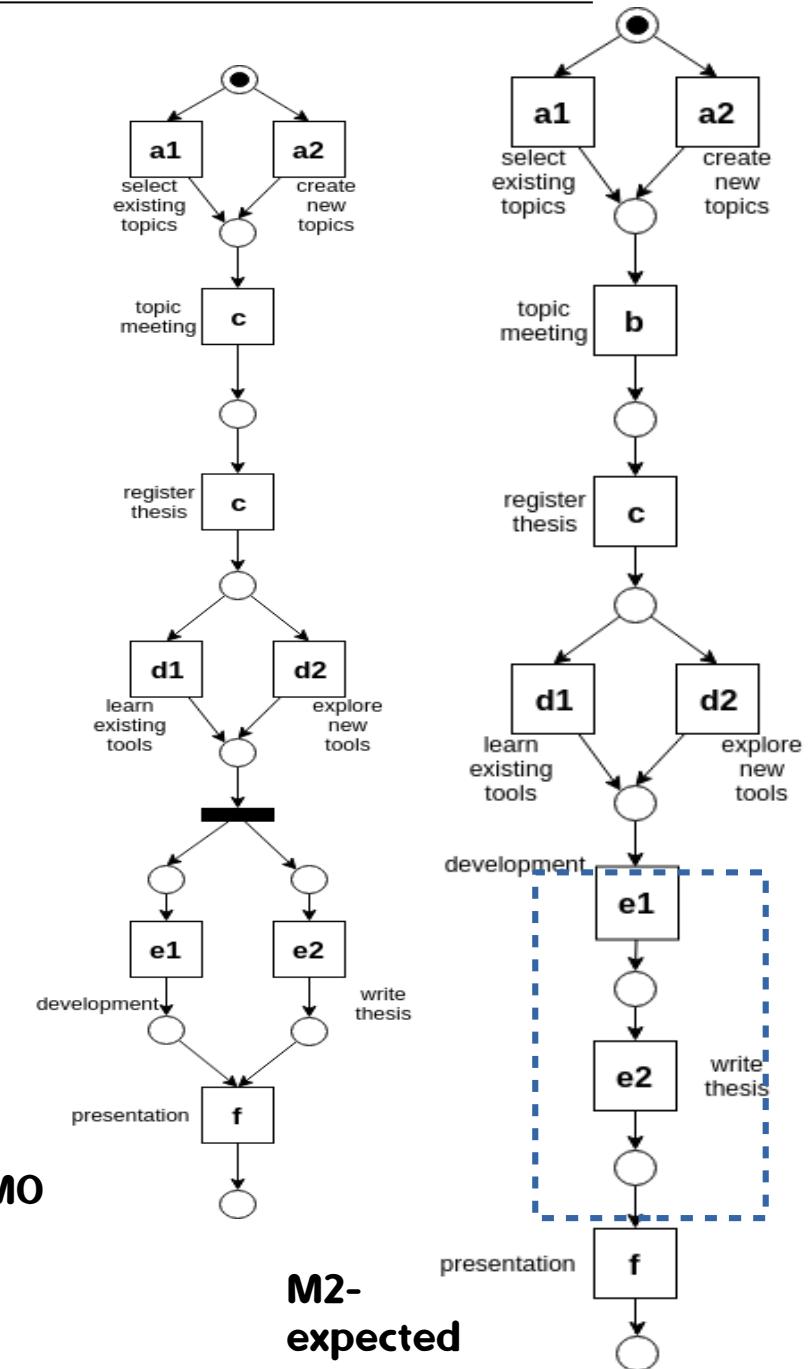


Motivation – shortcomings

$L_2 := \{ < a1, b, c, d2, e1, e2, f >^{30, pos}, \checkmark$
 $< a2, b, c, d1, e1, e2, f >^{20, pos}, \checkmark$
 $< a2, b, c, d2, e2, e1, f >^{10, pos};$
 $< a1, b, c, d2, e2, e1, f >^{20, neg}, \times$
 $< a1, b, c, d1, e2, e1, f >^{20, neg}, \times$
 $< a2, b, c, d1, e1, e2, f >^{5, neg} \}$

- IM keeps the model same
- Fahland's/Dees keep model same

⚠️ Unable to adapt model with negative instances



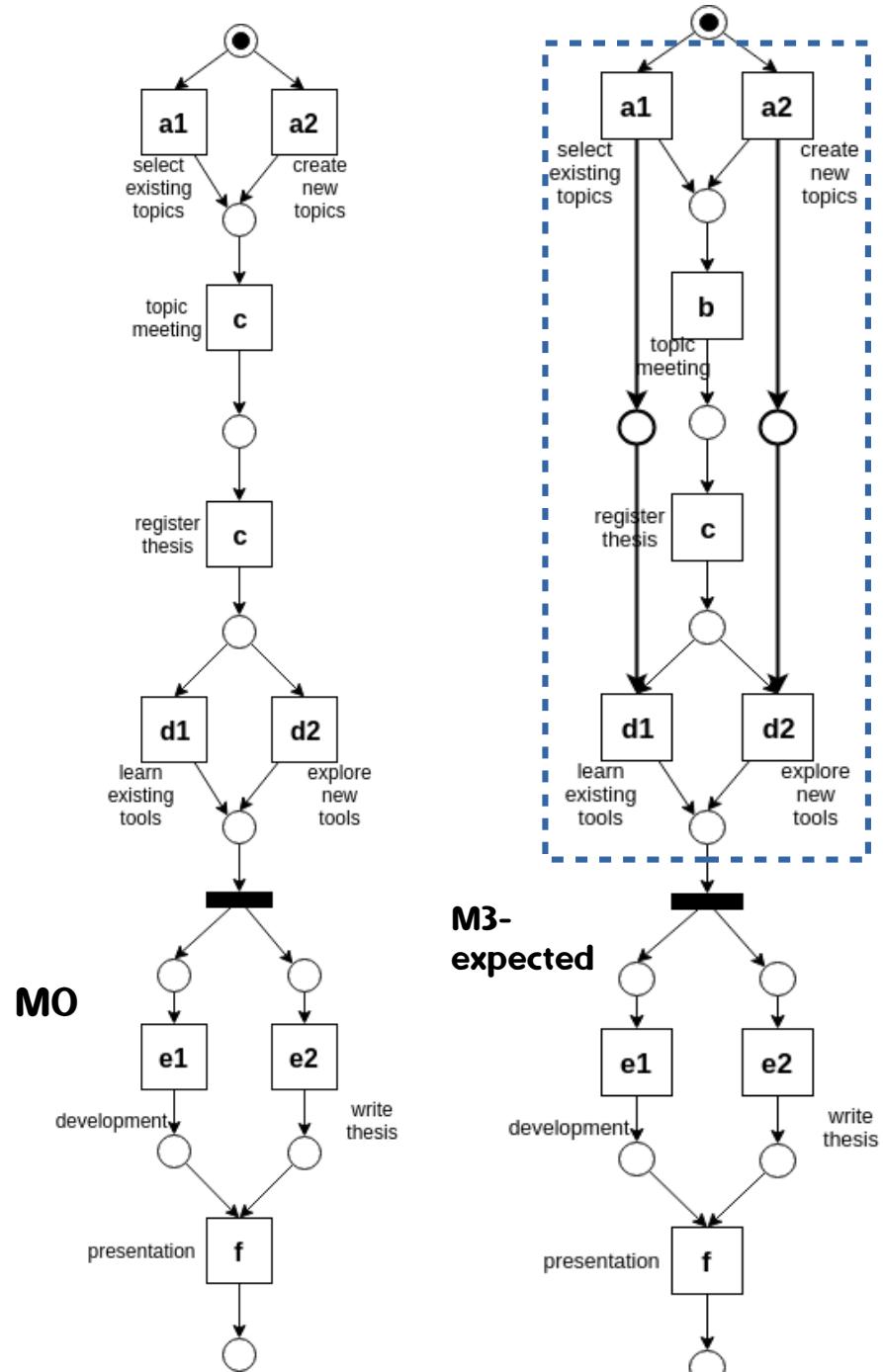
Motivation – shortcomings

$L_3 := \{ < \mathbf{a1}, b, c, \mathbf{d1}, e1, e2, f >^{50, pos}, \checkmark$
 $< \mathbf{a2}, b, c, \mathbf{d2}, e1, e2, f >^{50, pos}, \checkmark$
 $< \mathbf{a1}, b, c, \mathbf{d2}, e1, e2, f >^{50, neg}, \times$
 $< \mathbf{a2}, b, c, \mathbf{d1}, e1, e2, f >^{50, neg} \} \times$

- **Long-term dependency**
 - Choices decide choices

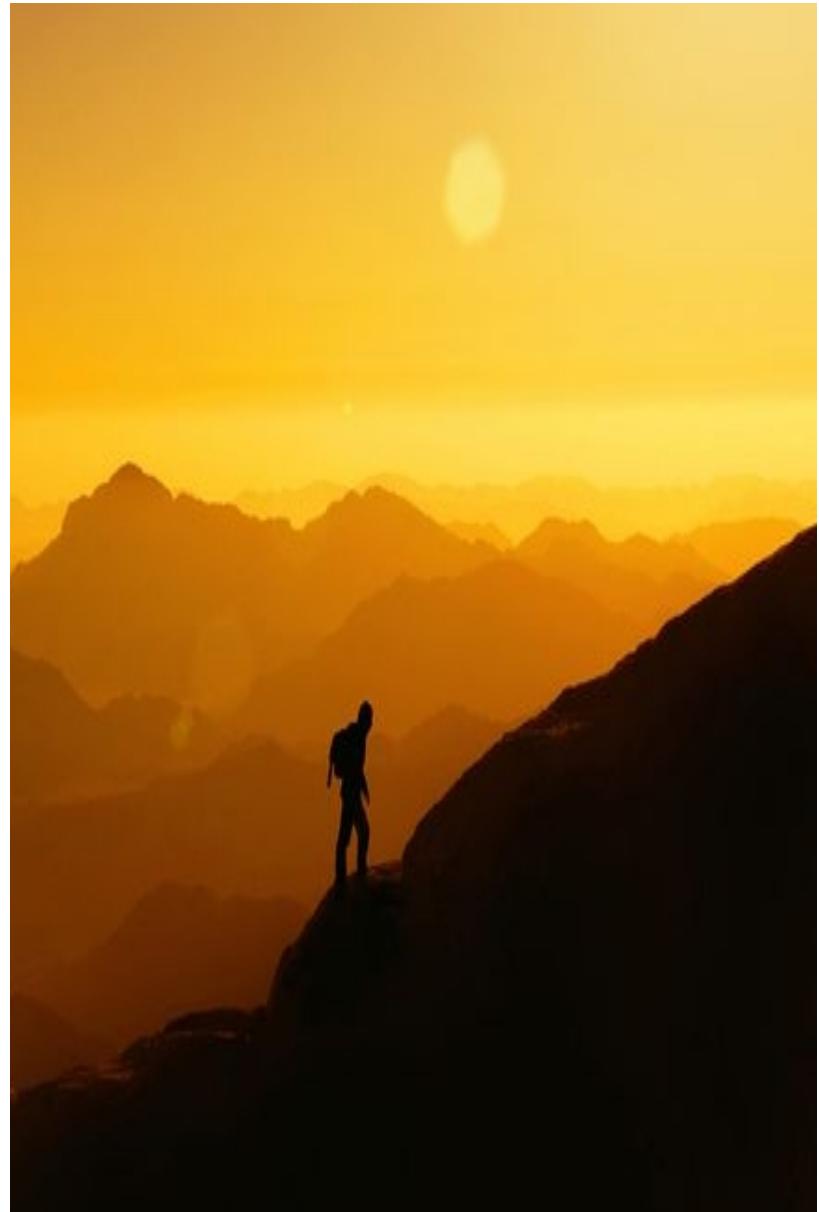


Unable to detect long-term dependency



Motivation – Shortcomings

- Similarity decreases with rediscovery
- Precision decrease by adding subprocess in loops
- Unable to adapt model with negative instances
- Unable to detect long-term dependency

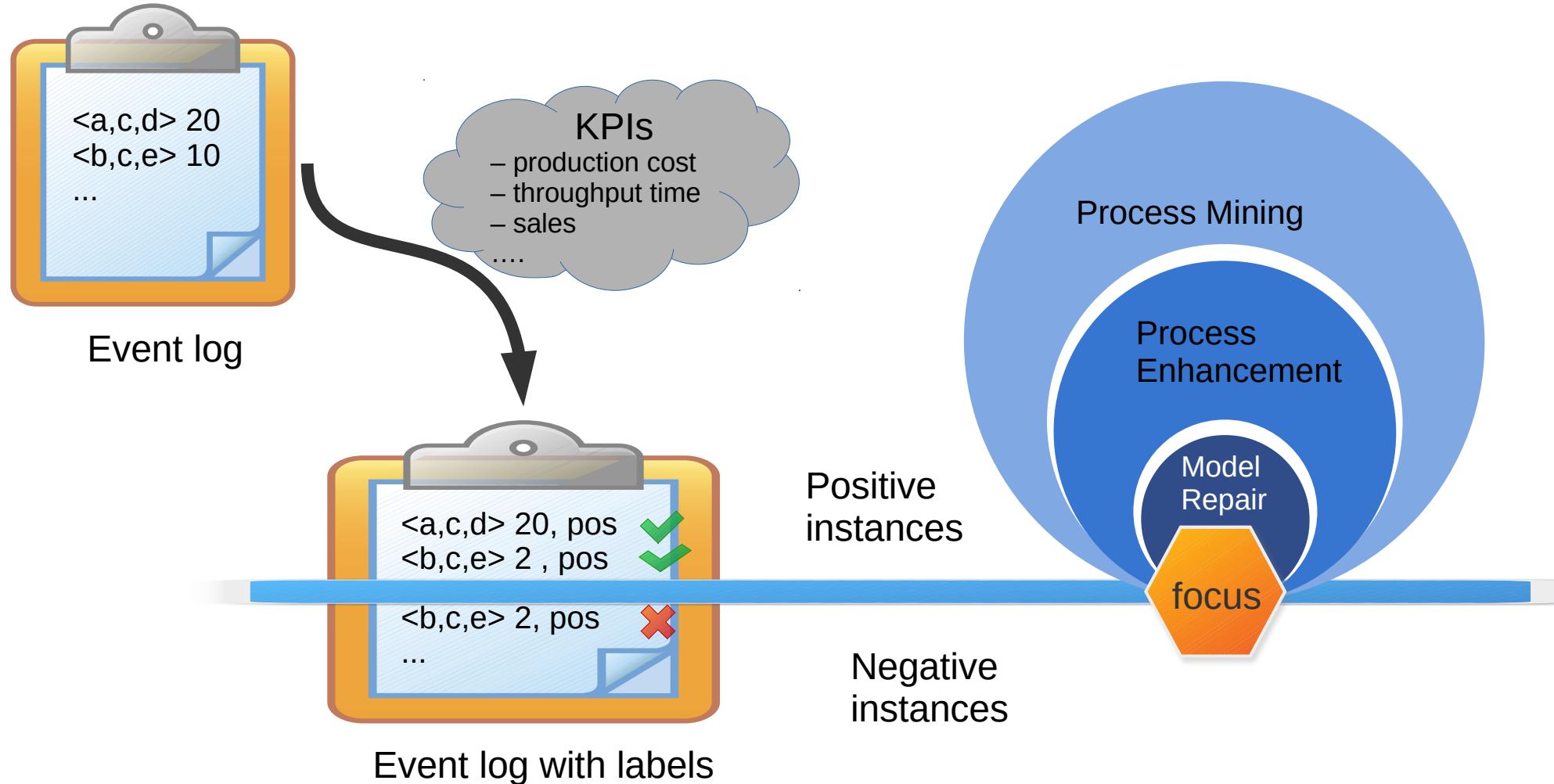


Outline

- Motivation for Research
 - Problem Definition
 - Approach
 - Demo
 - Evaluation
 - Conclusion

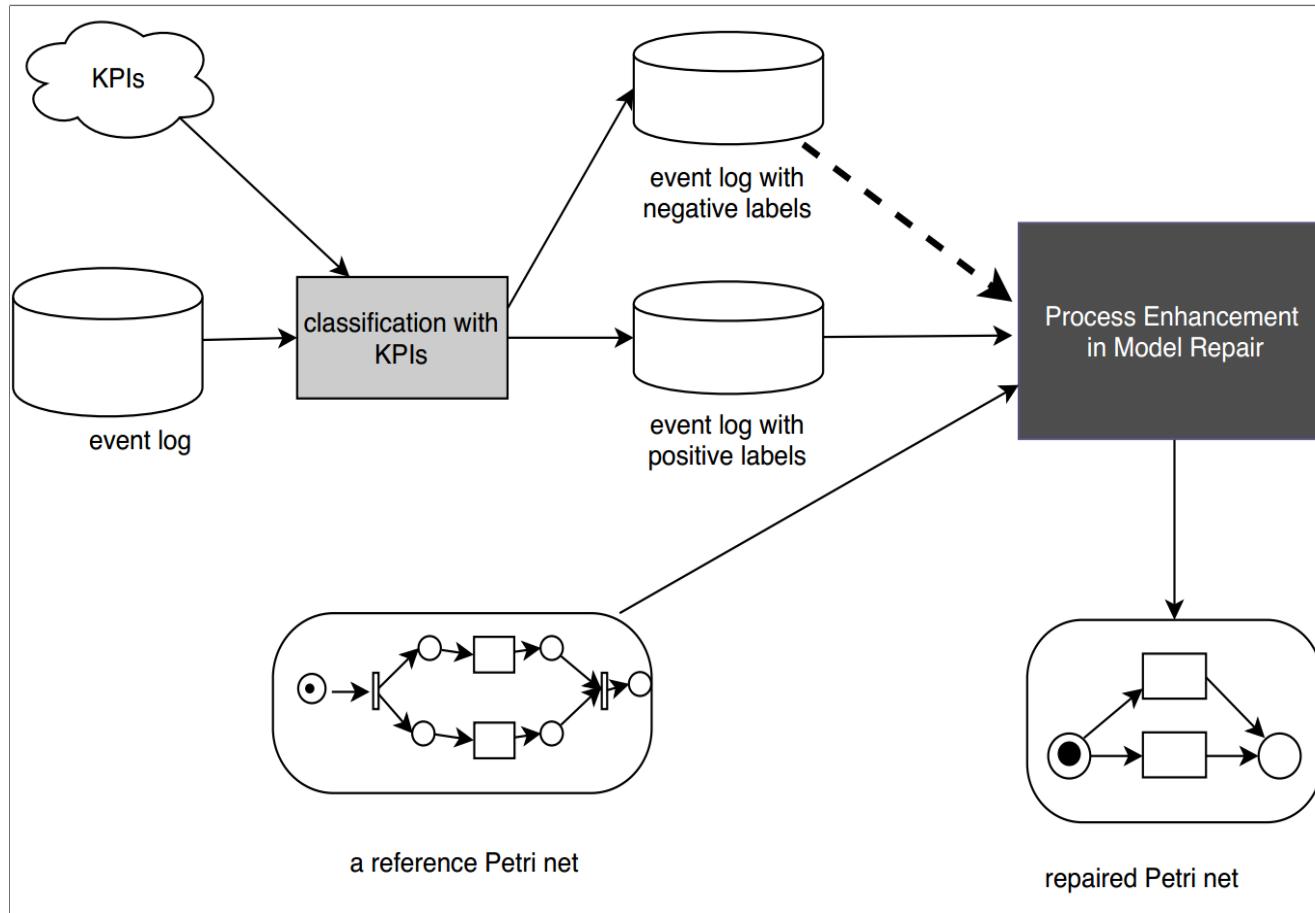


Research Problem



Research Problem

Given an **event log with labels**, a **reference Petri net**,
how to incorporate **negative instances** to generate the
repaired Petri net which supports better performance?



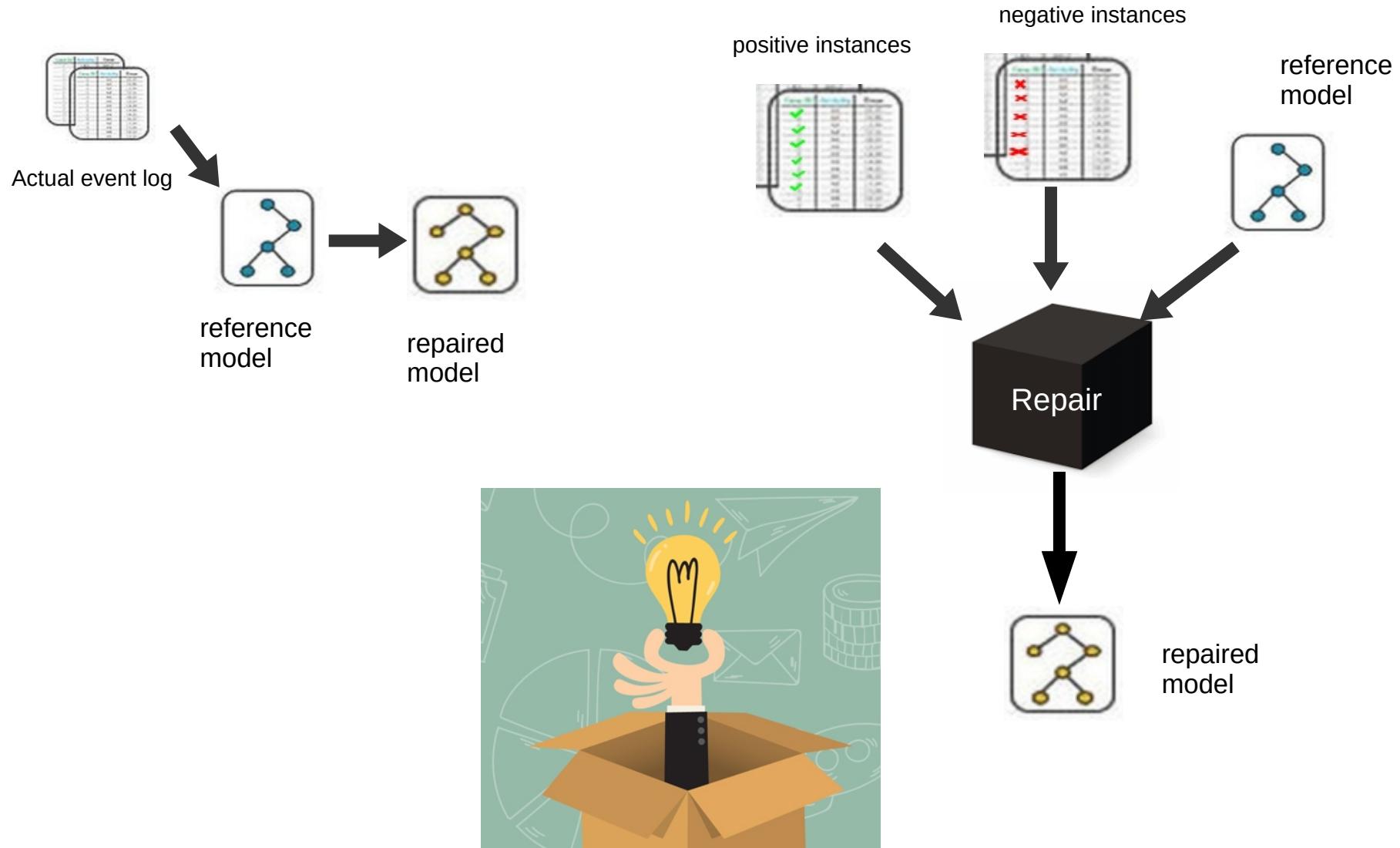
- enforce positive instances
- block negative instances
- similar
- simple
-

Outline

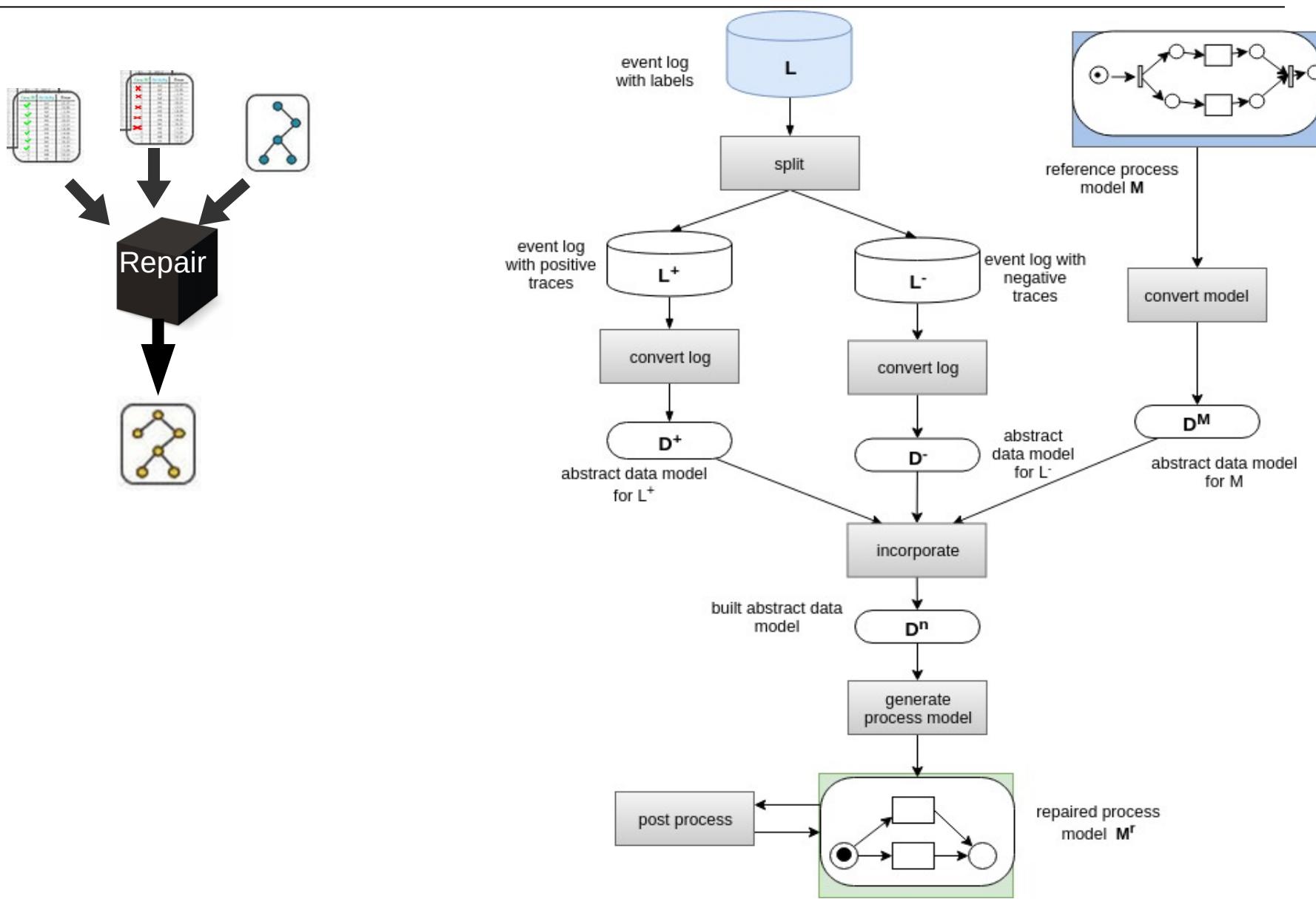
- Motivation for Research
 - Problem Definition
 - Approach
 - Framework
 - Data models
 - Modules
 - Demo
 - Evaluation
 - Conclusion



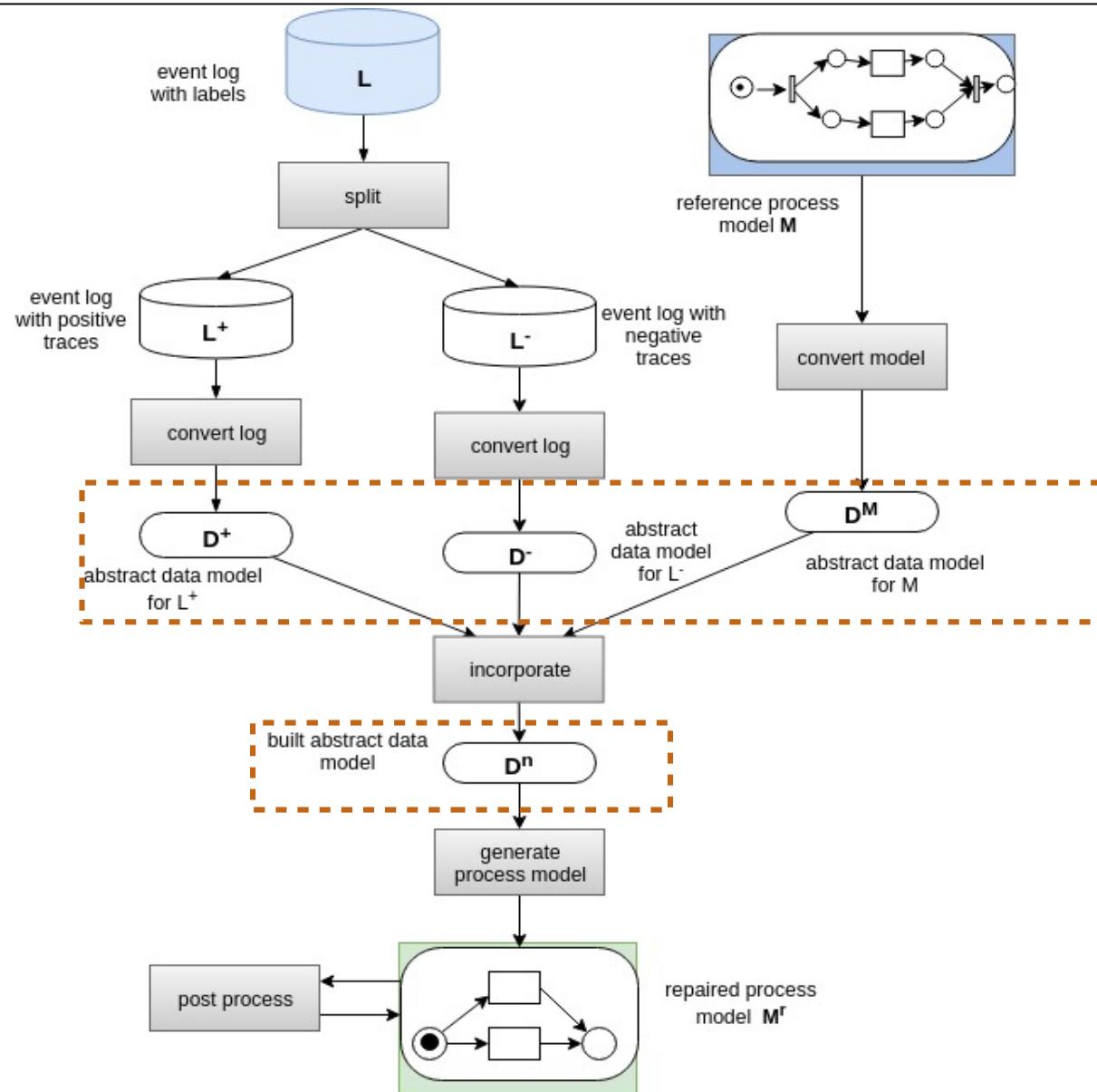
Algorithm – framework



Algorithm – framework



Algorithm – data model

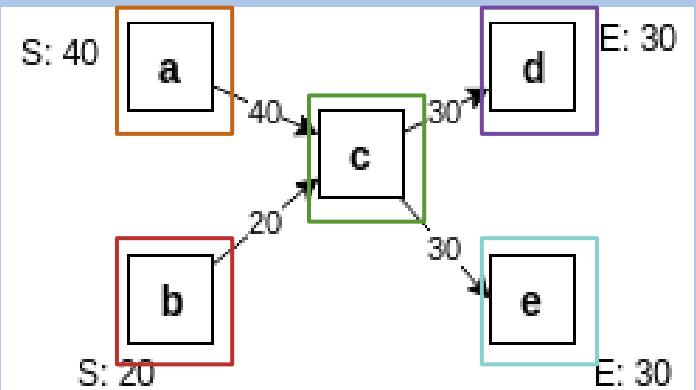


Data Model – directly-follows graph

$$G(L) = (A, F, A_{start}, A_{end})$$

► $A = A_L$

$$L = \{\langle a, [c, d] \rangle^{20}, \langle b, c, [e] \rangle^{10}, \\ \langle a, c, e \rangle^{20}, \langle b, c, d \rangle^{10} \}$$

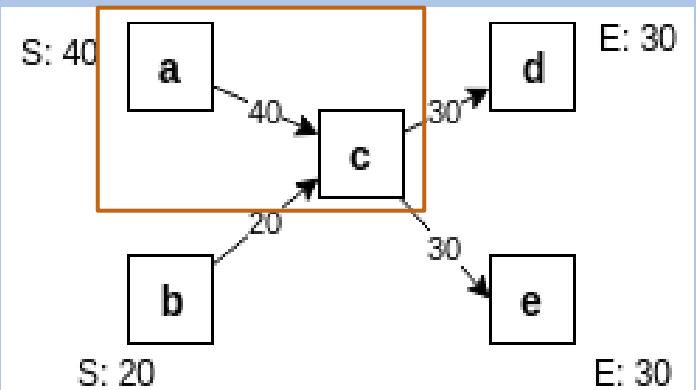


Data Model – directly-follows graph

$$G(L) = (A, F, A_{start}, A_{end})$$

- ▶ $A = A_L$
- ▶ $F = \{(a, b) \in A \times A | a >_L b\}$

$$L = \{\langle a, c, d \rangle^{20}, \langle b, c, e \rangle^{10}, \\ \langle a, c, e \rangle^{20}, \langle b, c, d \rangle^{10}\}$$

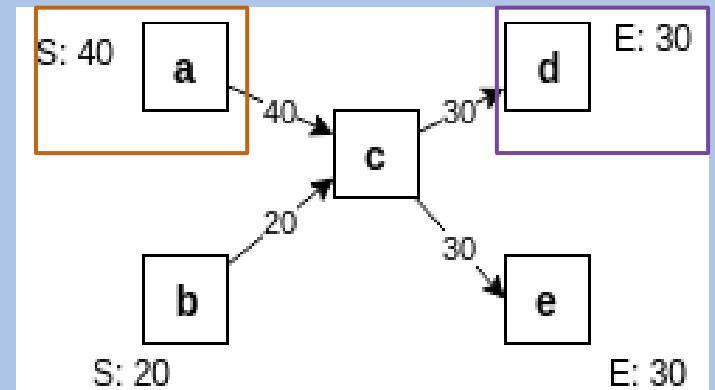


Data Model – directly-follows graph

$$G(L) = (A, F, A_{start}, A_{end})$$

- ▶ $A = A_L$
- ▶ $F = \{(a, b) \in A \times A \mid a >_L b\}$
- ▶ $A_{start} = \{a \mid \exists \sigma \in L, a = \sigma(1)\}$
- ▶ $A_{end} = \{a \mid \exists \sigma \in L, a = \sigma(|\sigma|)\}$

$$L = \{\langle a, c, d \rangle^{20}, \langle b, c, e \rangle^{10}, \\ \langle a, c, e \rangle^{20}, \langle b, c, d \rangle^{10}\}$$

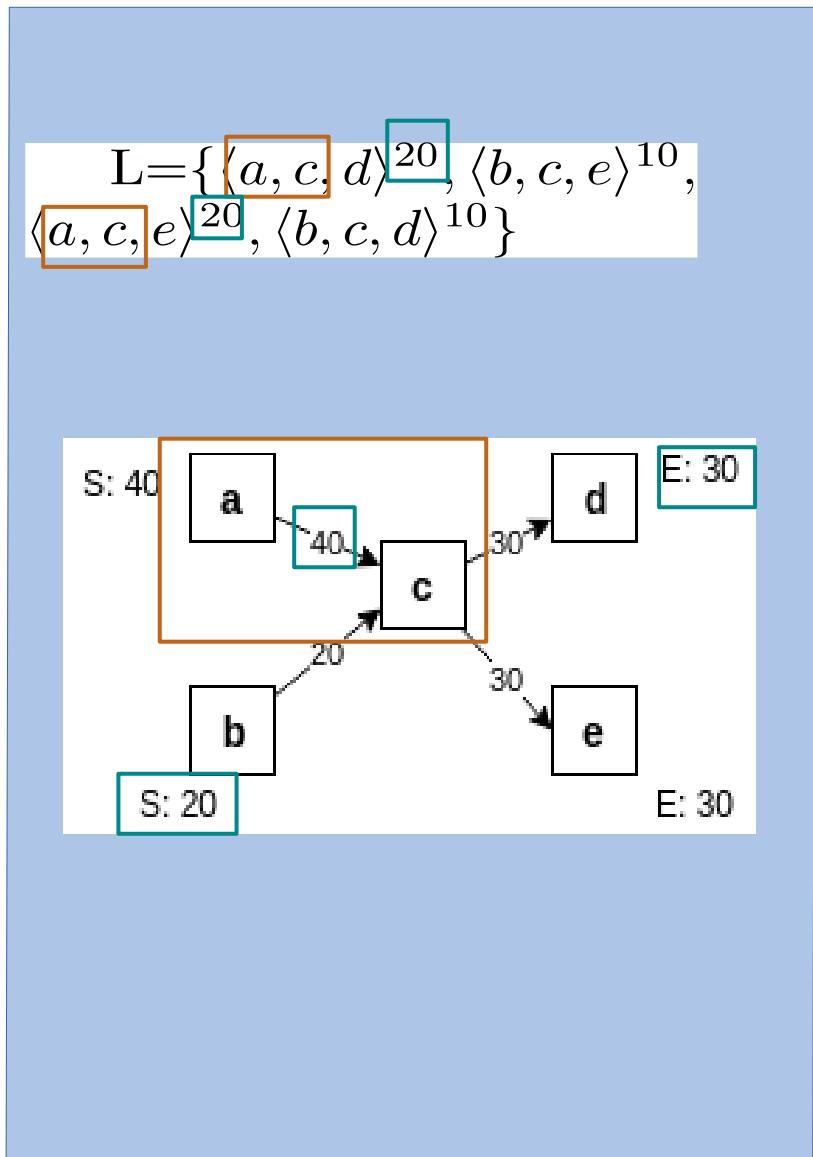


Data Model – directly-follows graph

$$G(L) = (A, F, A_{start}, A_{end})$$

- ▶ $A = A_L$
- ▶ $F = \{(a, b) \in A \times A \mid a >_L b\}$
- ▶ $A_{start} = \{a \mid \exists \sigma \in L, a = \sigma(1)\}$
- ▶ $A_{end} = \{a \mid \exists \sigma \in L, a = \sigma(|\sigma|)\}$
- ▶ Cardinality $c : F \rightarrow N$

$$A_{start} \cup A_{end} \rightarrow N$$

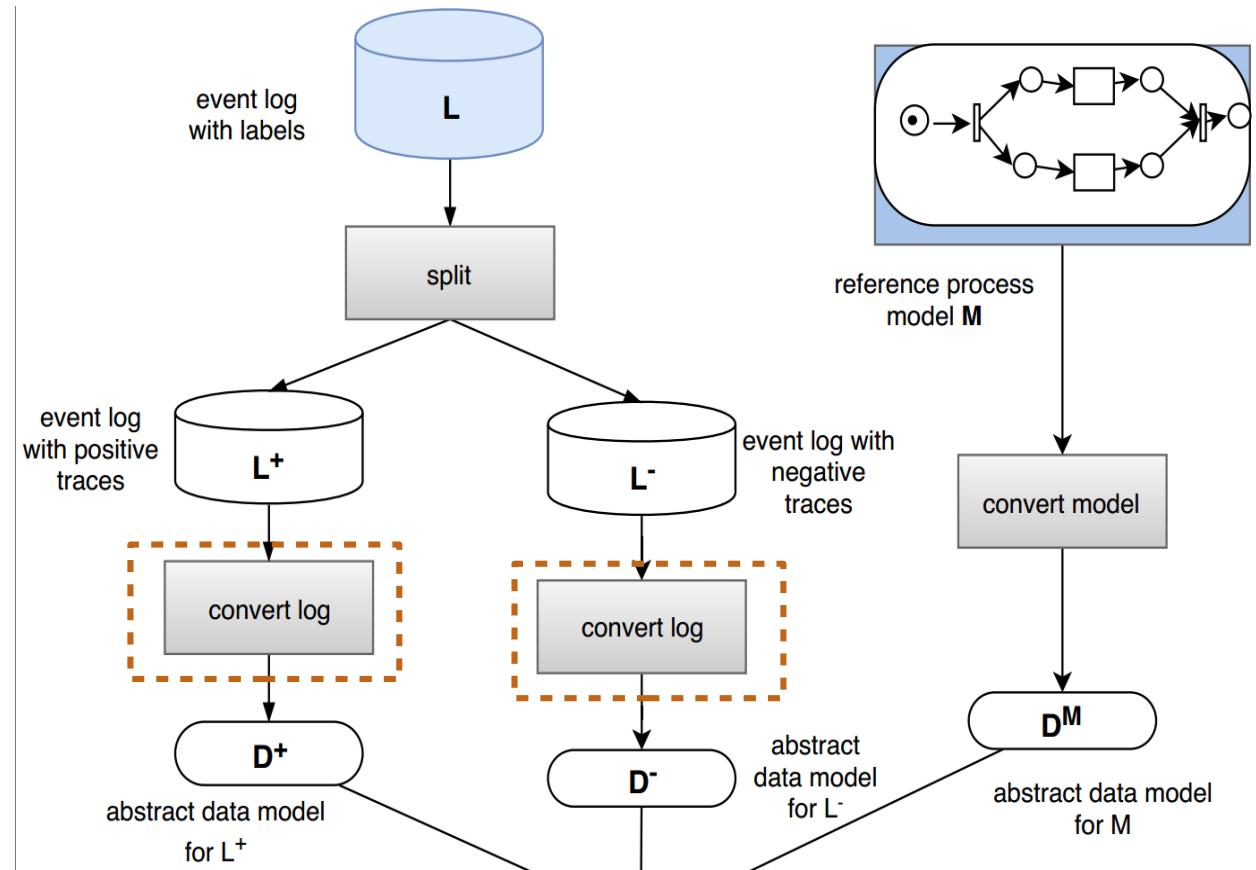
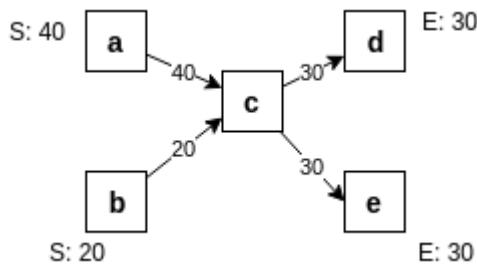


Convert to directly-follows graph

From Event log

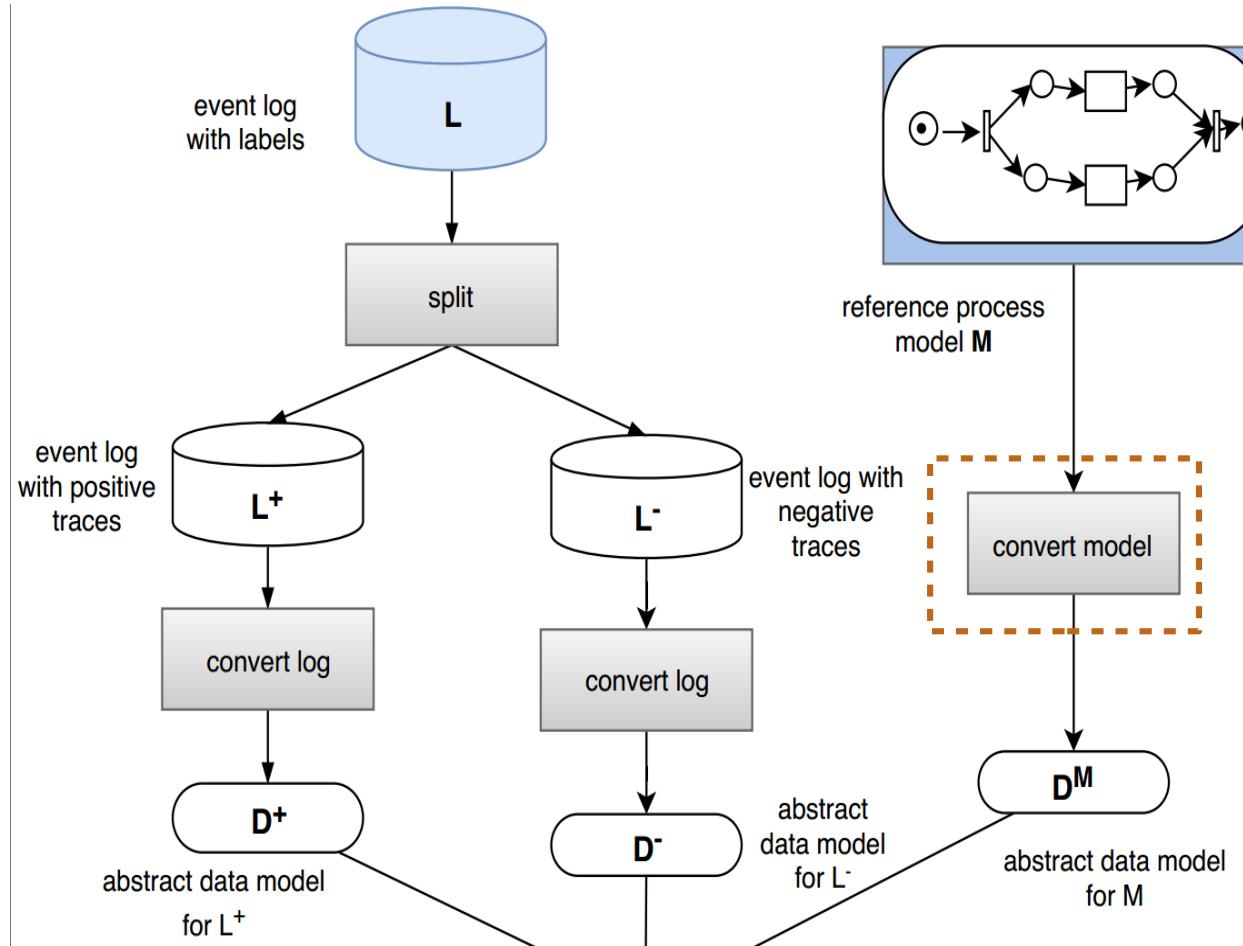
- Directly-follows relation
- Existing plugin

$$L = \{\langle a, c, d \rangle^{20}, \langle b, c, e \rangle^{10}, \\ \langle a, c, e \rangle^{20}, \langle b, c, d \rangle^{10}\}$$



Convert to directly-follows graph

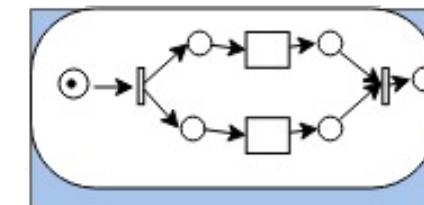
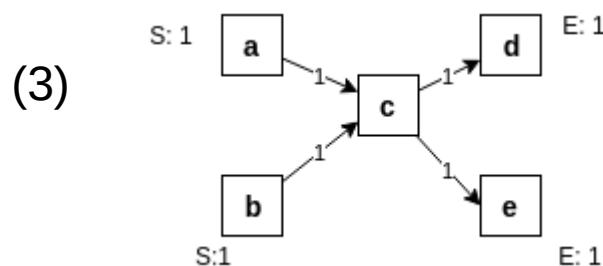
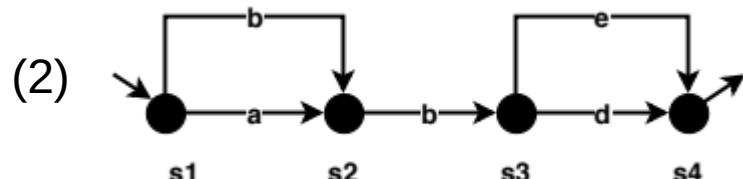
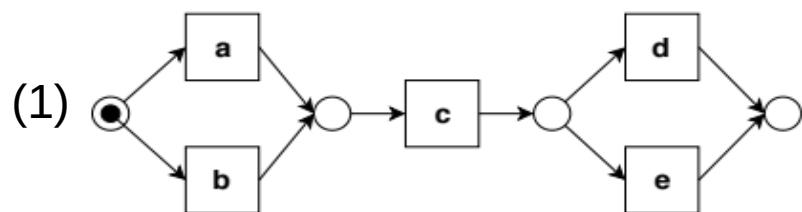
From Petri net



Convert to directly-follows graph

From Petri net

- Transition System
 - Directly-follows relation from transitions before and after states



reference process model M

convert model

D^M

abstract data model
for M

Data Model

- **Unification of cardinality**

- Models from existing model, positive and negative event log
- For any directly-follows relation

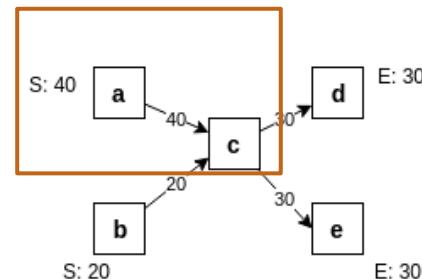
$$u(a, b) = \frac{c(a, b)}{\sum_{(a, b') \in F} c(a, b')}$$

- For any start activity

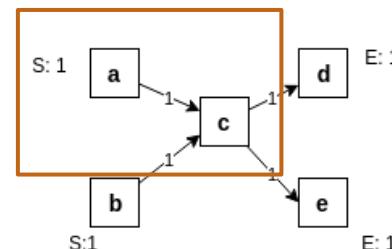
$$u(a) = \frac{c(a)}{\sum_{a' \in A_{start}} c(a')}$$

- For any end activity

$$u(a) = \frac{c(a)}{\sum_{a' \in A_{end}} c(a')}$$



dfg from event log



dfg from model

Data Model

- **Unification of cardinality**

- Models from existing model, positive and negative event log

- For any directly-follows relation

$$u(a, b) = \frac{c(a, b)}{\sum_{(a, b') \in F} c(a, b')}$$

- For any start activity

$$u(a) = \frac{c(a)}{\sum_{a' \in A_{start}} c(a')}$$

- For any end activity

$$u(a) = \frac{c(a)}{\sum_{a' \in A_{end}} c(a')}$$



dfg from event log

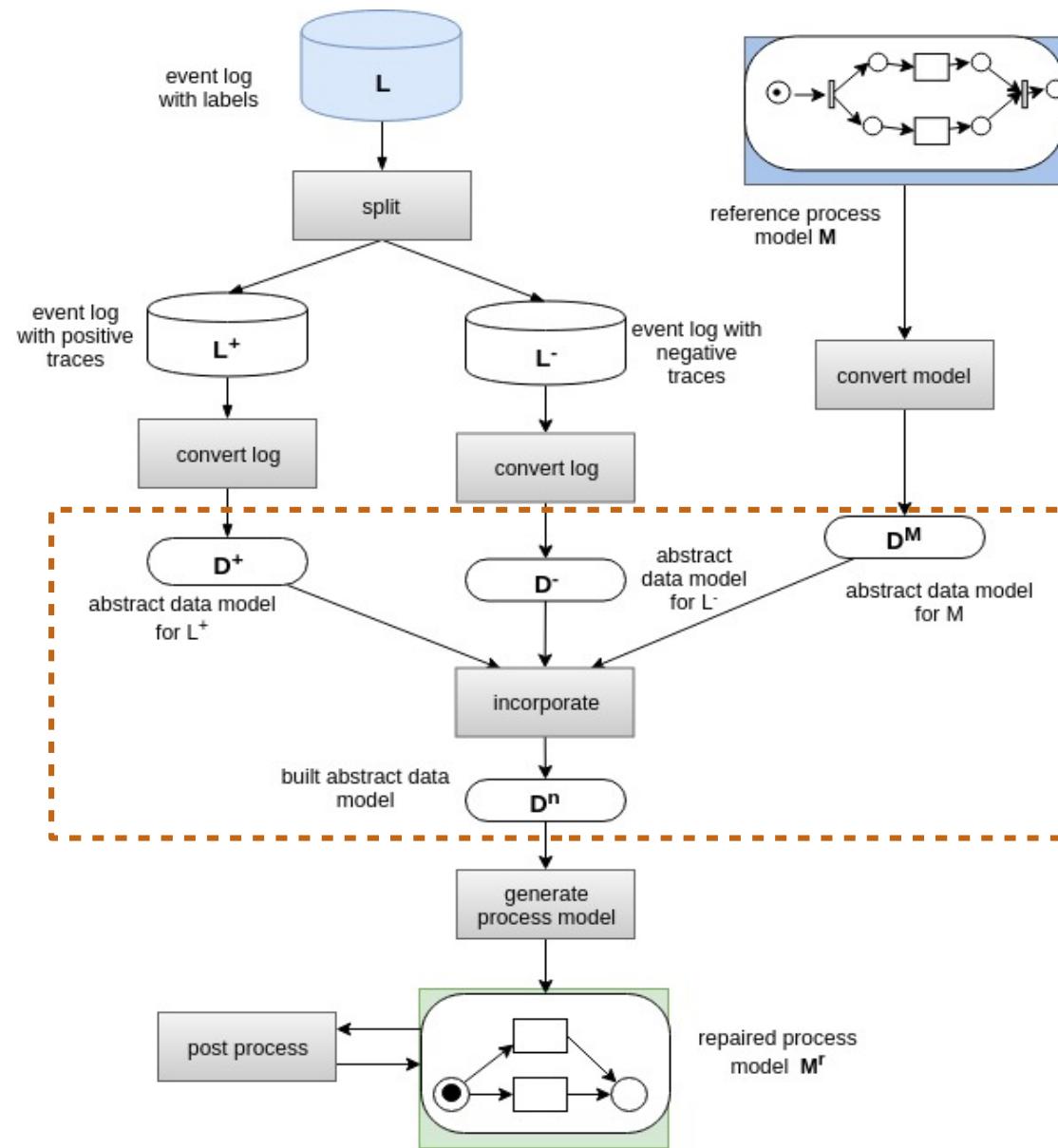
Unified dfg from event log



dfg from model

Unified dfg from model

Algorithm – data model

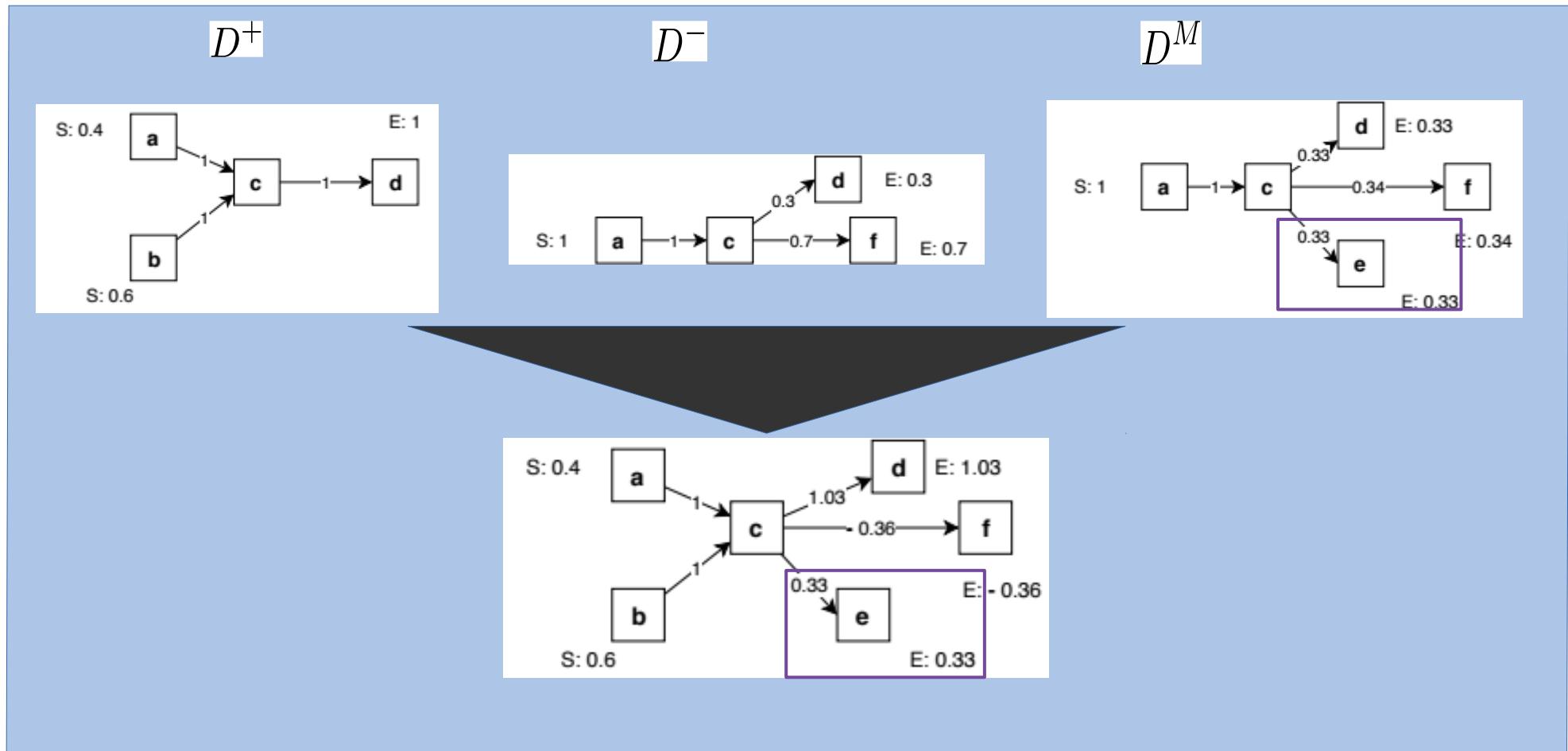
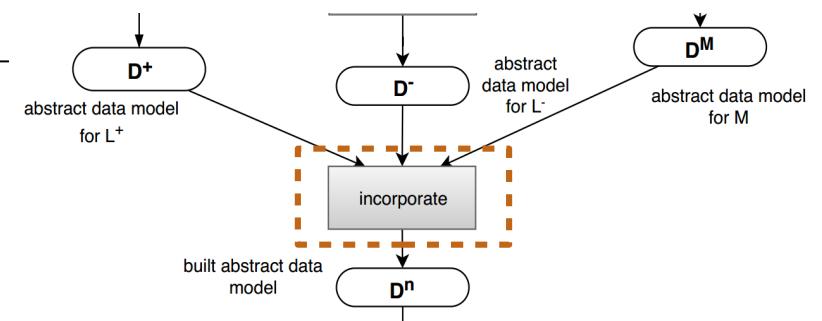


Incorporate Data Models

- Incorporate method**

- For any directly-follows relation

$$u^n(a, b) = u^M(a, b) + u^+(a, b) - u^-(a, b)$$

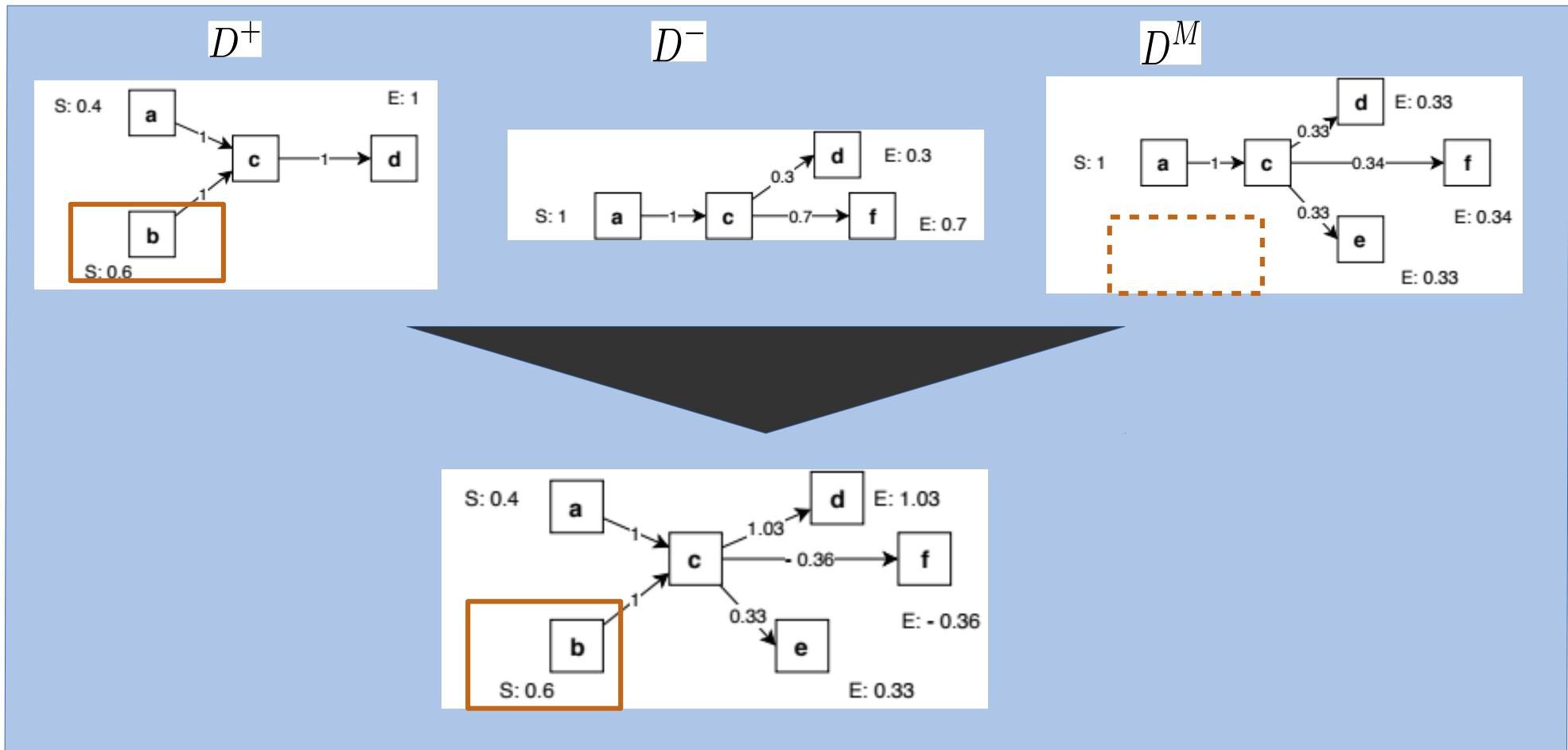
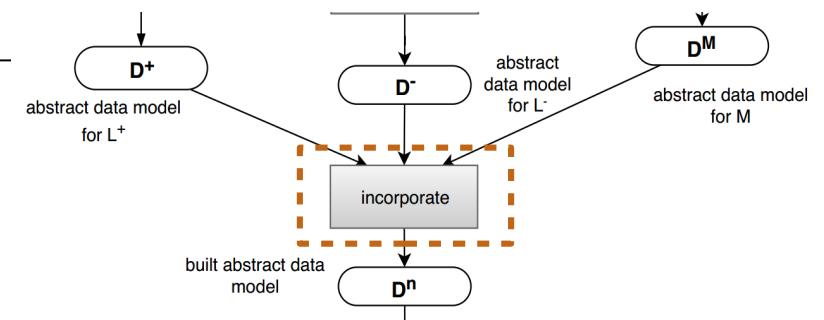


Incorporate Data Models

- Incorporate method**

- For any directly-follows relation

$$u^n(a, b) = u^M(a, b) + u^+(a, b) - u^-(a, b)$$

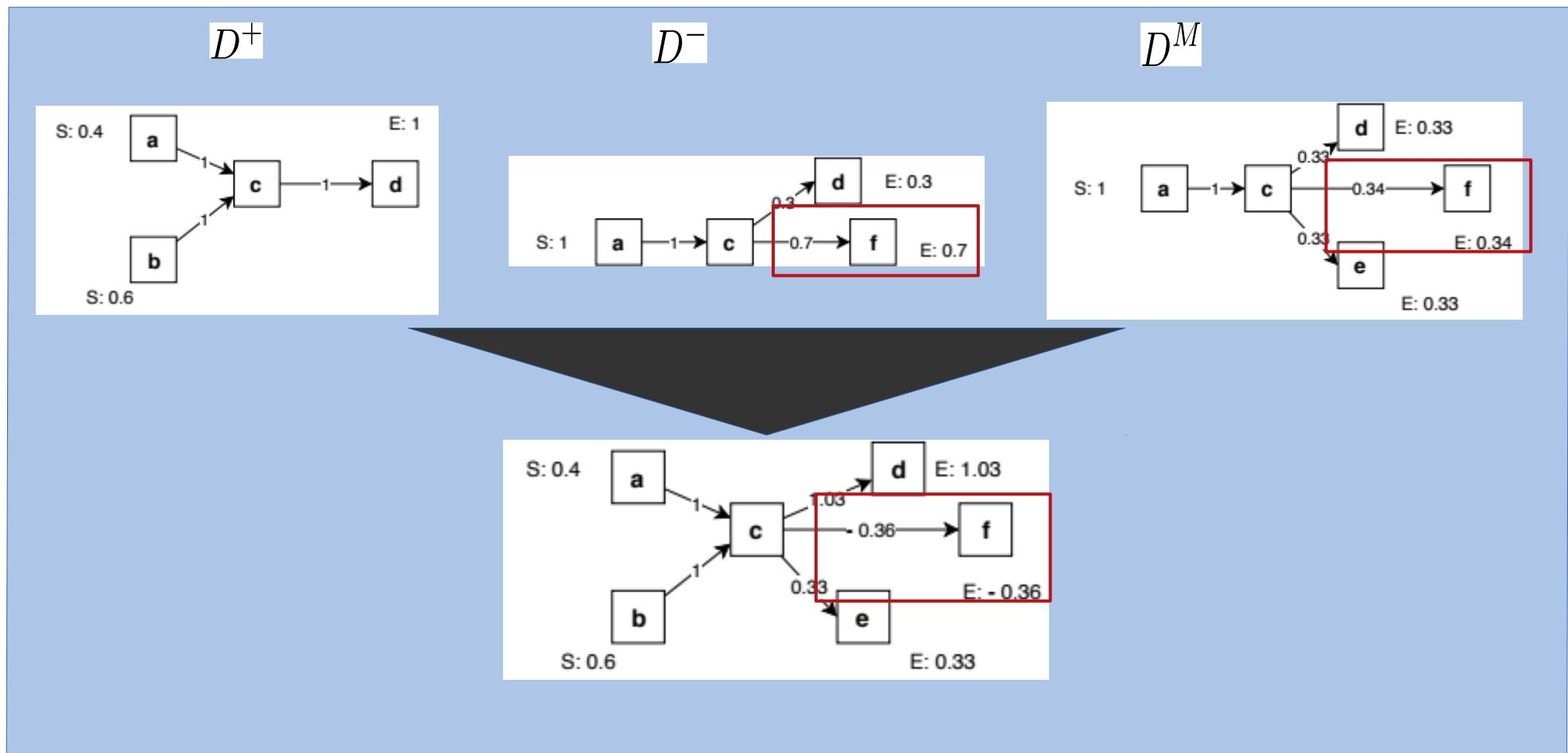
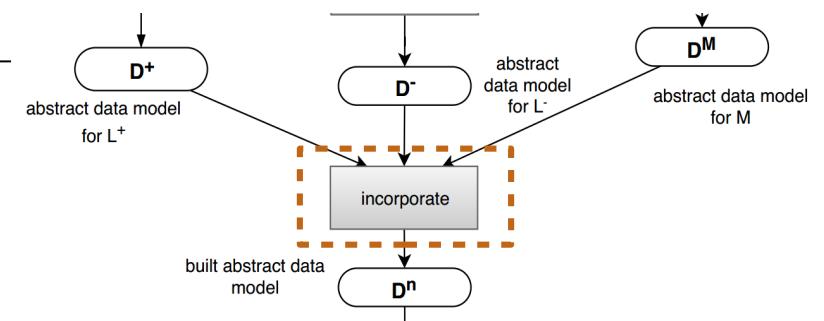


Incorporate Data Models

- Incorporate method**

- For any directly-follows relation

$$u^n(a, b) = u^M(a, b) + u^+(a, b) - u^-(a, b)$$

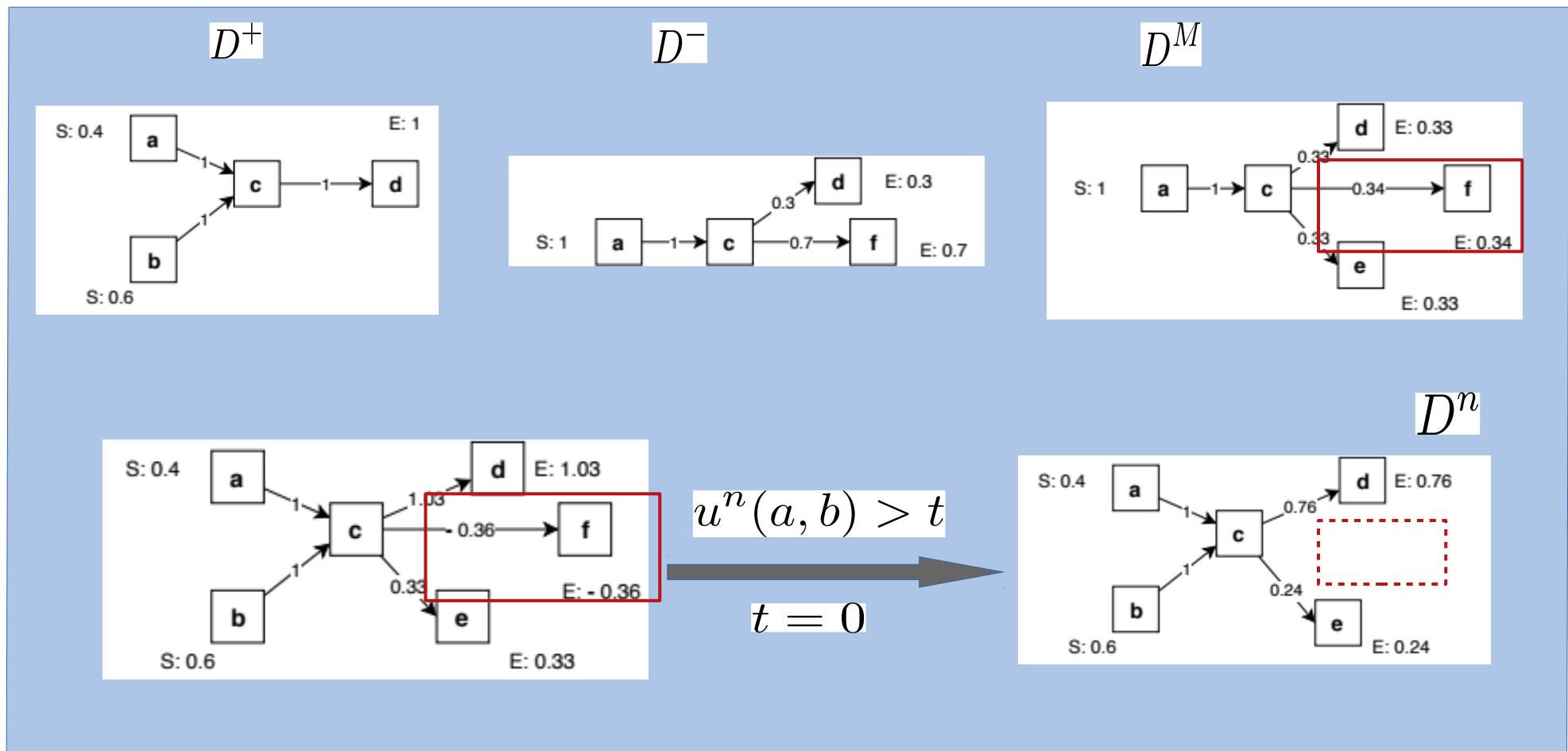
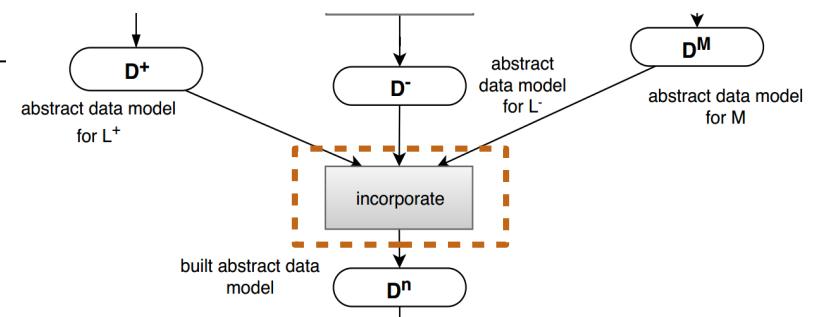


Incorporate Data Models

- Incorporate method**

- For any directly-follows relation

$$u^n(a, b) = u^M(a, b) + u^+(a, b) - u^-(a, b)$$

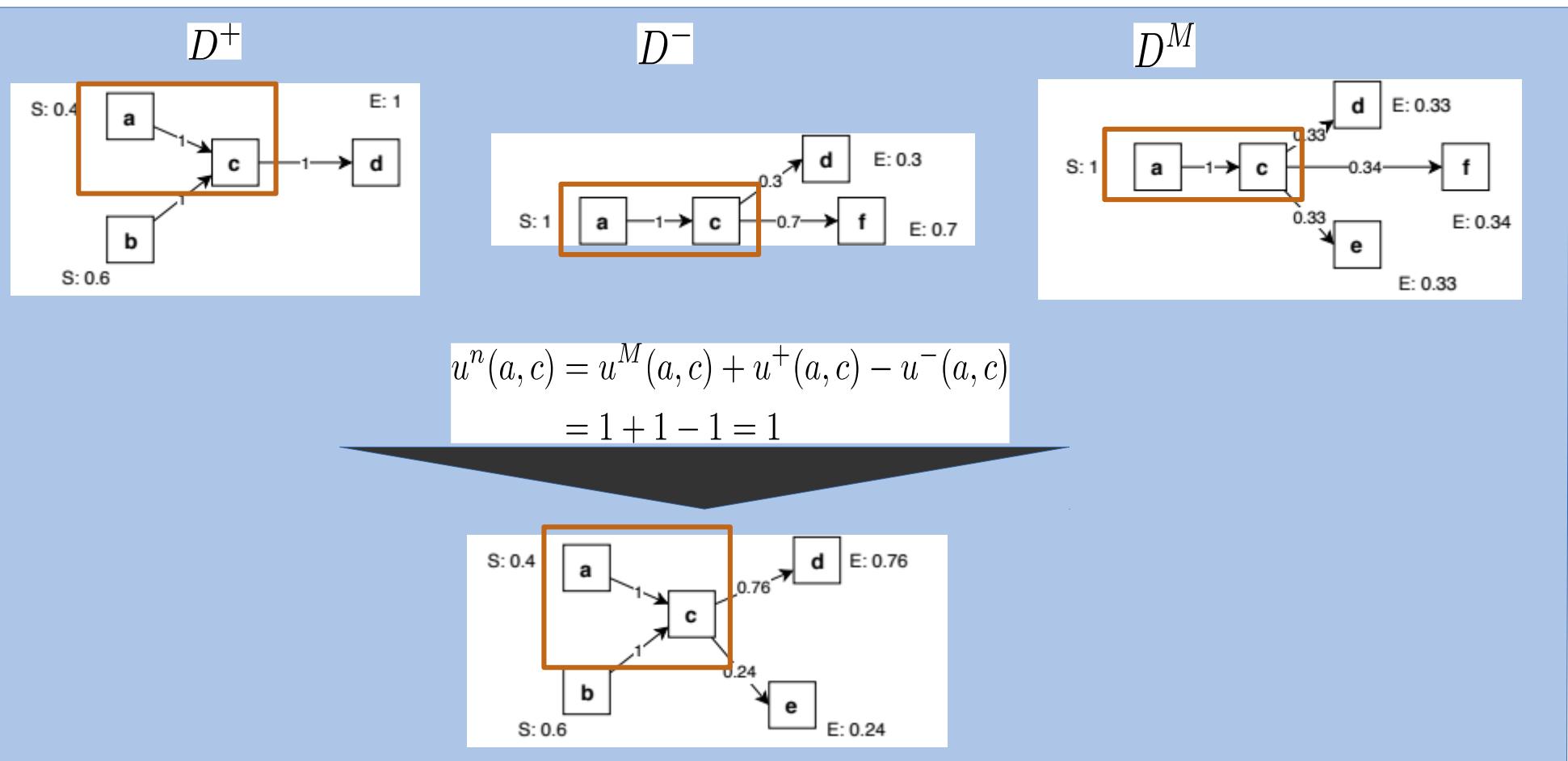
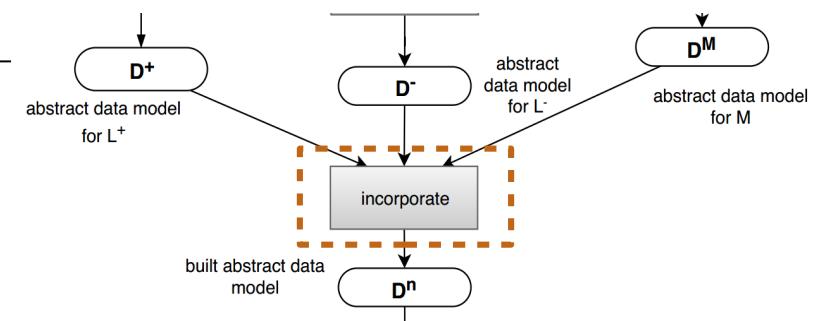


Incorporate Data Models

- Incorporate method**

- For any directly-follows relation

$$u^n(a, b) = u^M(a, b) + u^+(a, b) - u^-(a, b)$$



Incorporate Data Models

- Incorporate method with weights** w^+, w^-, w^M

- For any directly-follows relation

$$u_w^n(a, b) = w^M \cdot u^M(a, b) + w^+ \cdot u^+(a, b) - w^- \cdot u^-(a, b)$$

- For any start activity,

$$a \in A_{start}^M \cup A_{start}^+ \cup A_{start}^-, u^n(a) = w^M \cdot u^M(a) + w^+ \cdot u^+(a) - w^- \cdot u^-(a)$$

- For any end activity,

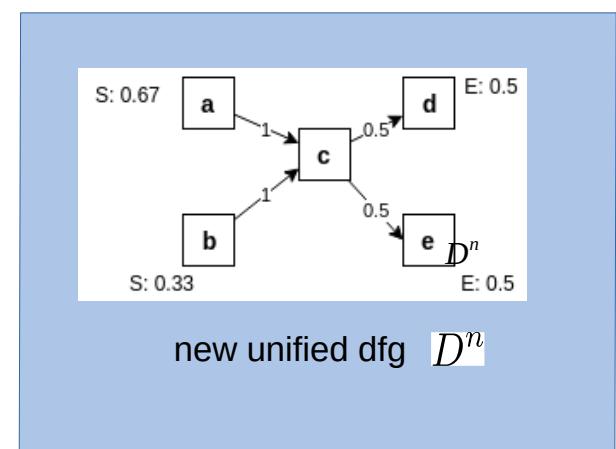
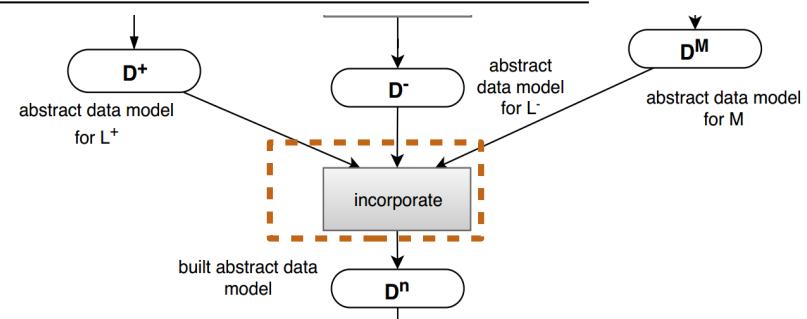
$$a \in A_{end}^M \cup A_{end}^+ \cup A_{end}^-, u^n(a) = w^M \cdot u^M(a) + w^+ \cdot u^+(a) - w^- \cdot u^-(a)$$

$$w^M = 1, w^+ = w^- = 0$$

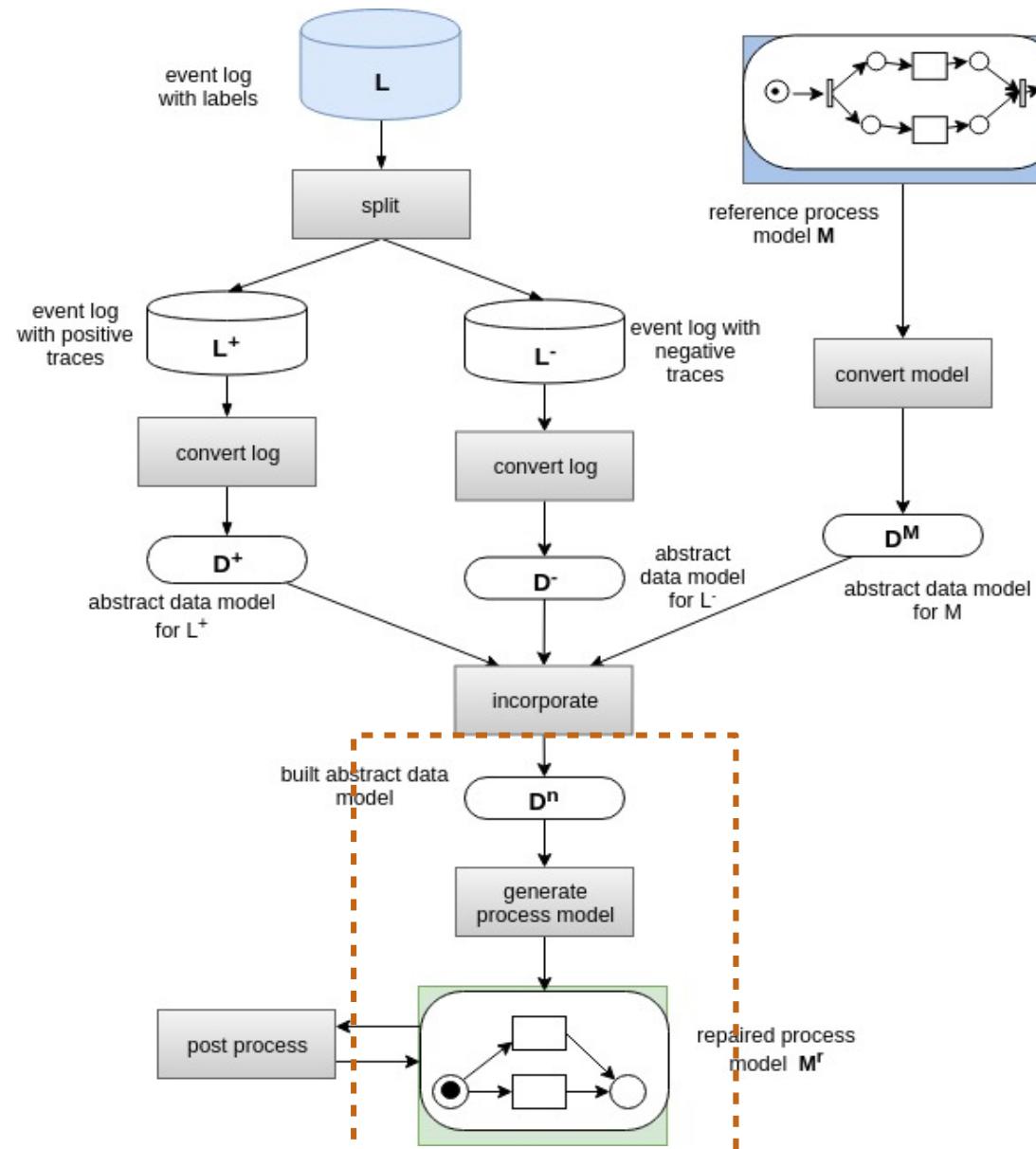
→ Keep the reference model

$$w^+ = 1, w^M = w^- = 0$$

→ Mine a new model from positive instances



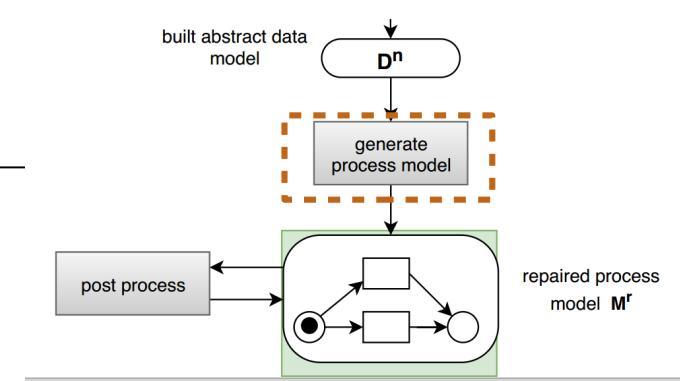
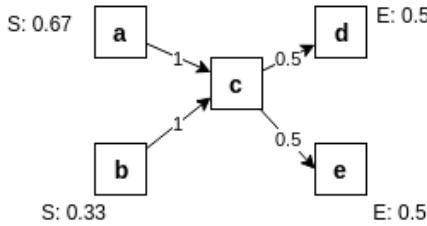
Algorithm – data model



Generate Petri net

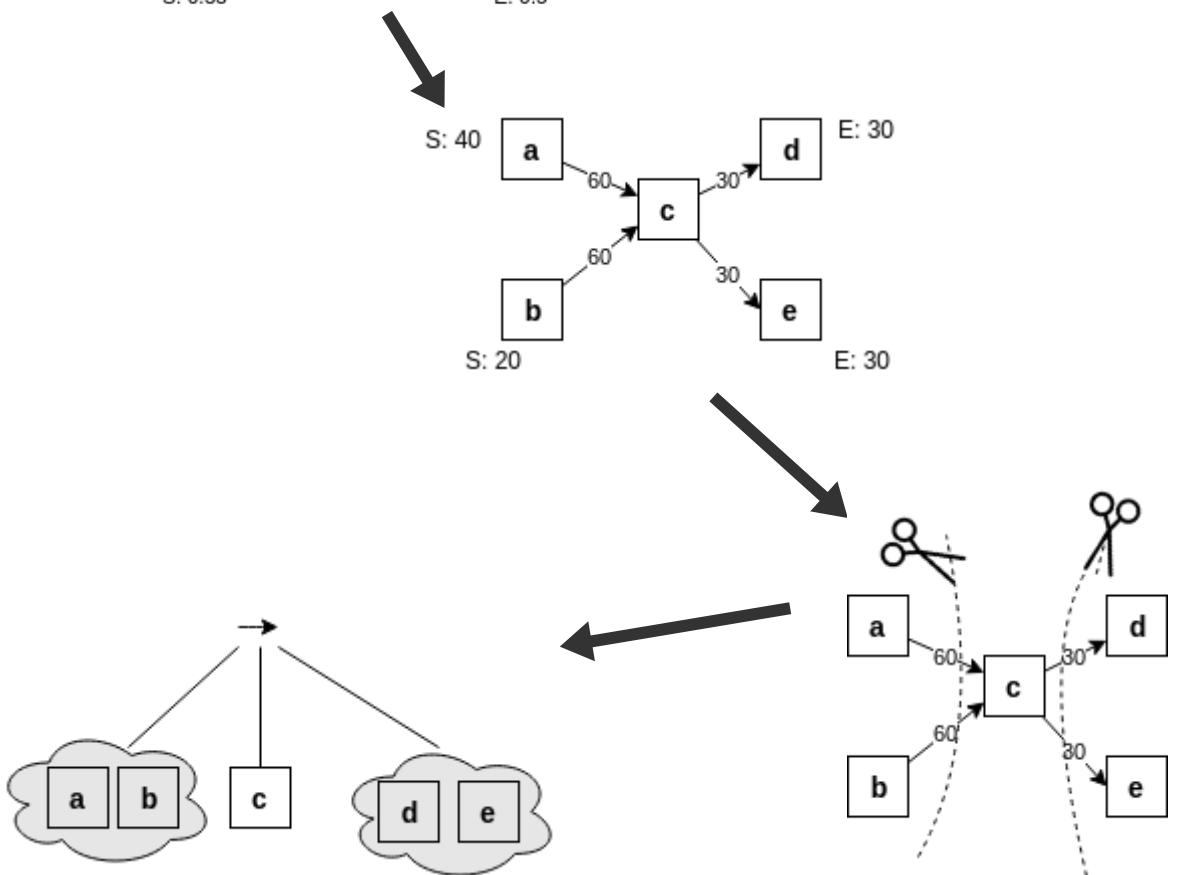
- Transform into normal dfg

$$c^n(a, b) = u_w^n(a, b) \cdot (|L^+| + |L^-|)$$



- IM algorithm

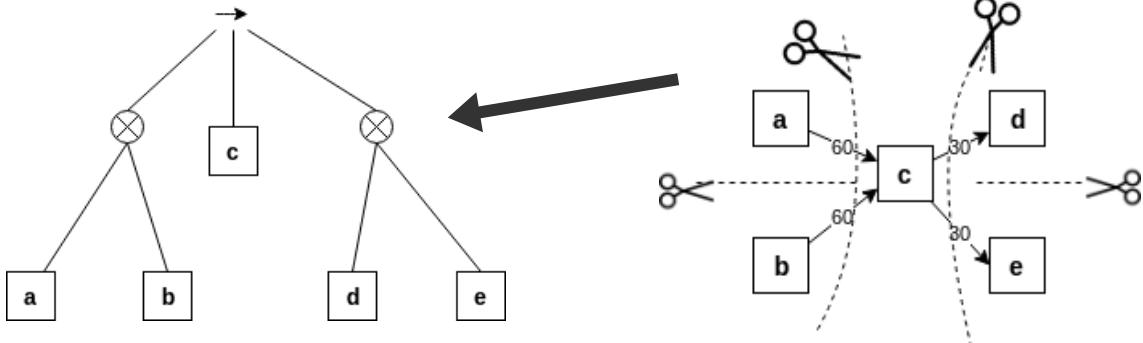
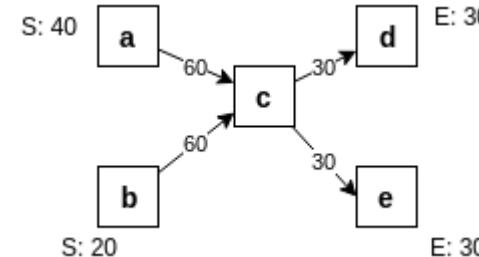
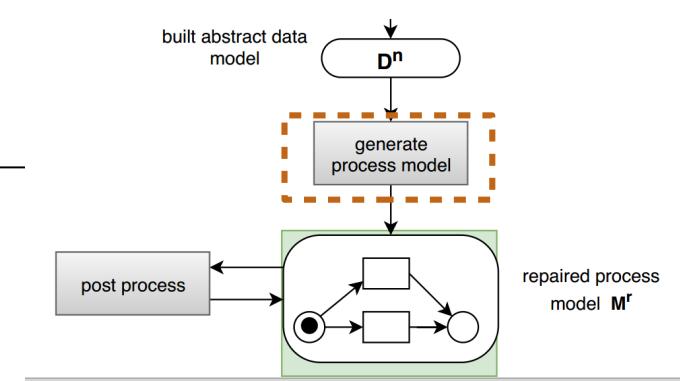
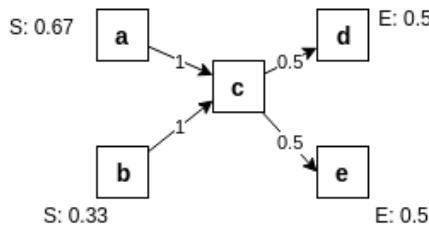
- Directly-follows graph
- Process tree
- Petri net



Generate Petri net

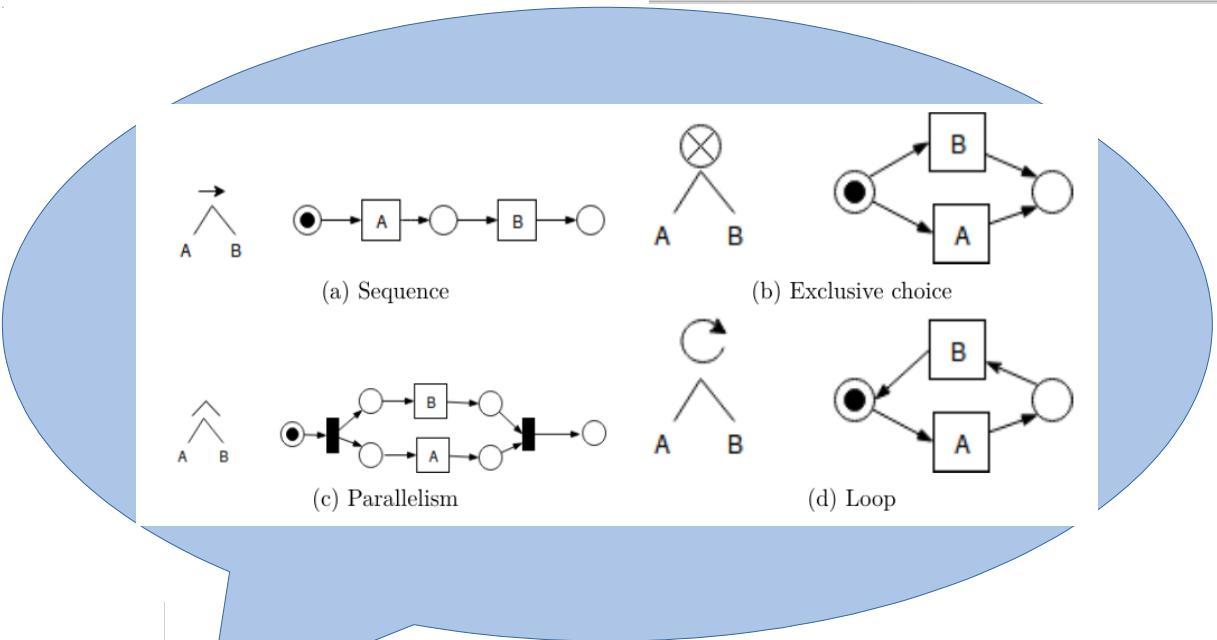
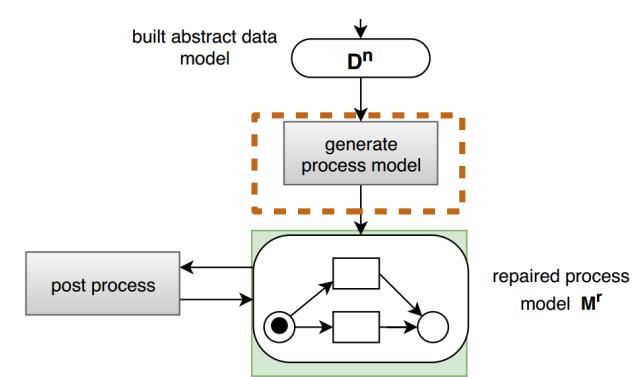
- Transform into normal dfg

$$c^n(a, b) = u_w^n(a, b) \cdot (|L^+| + |L^-|)$$



Generate Petri net

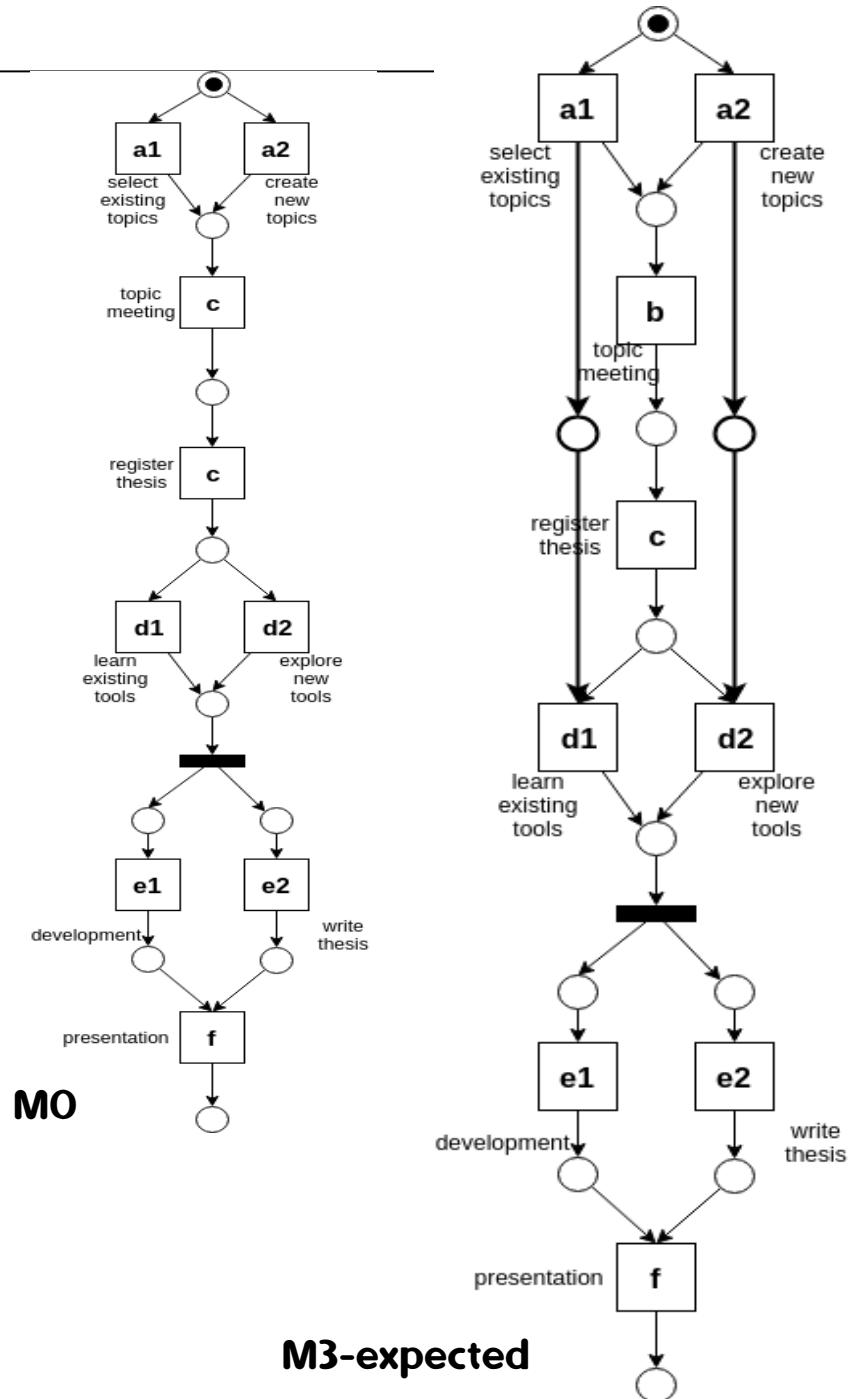
- Transform into normal dfg
- IM algorithm
 - Directly-follows graph
 - Process tree
 - Petri net



Post Process Petri net

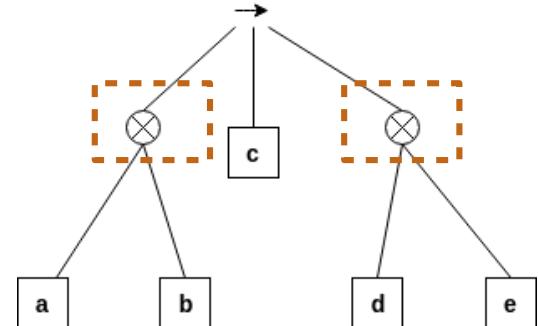
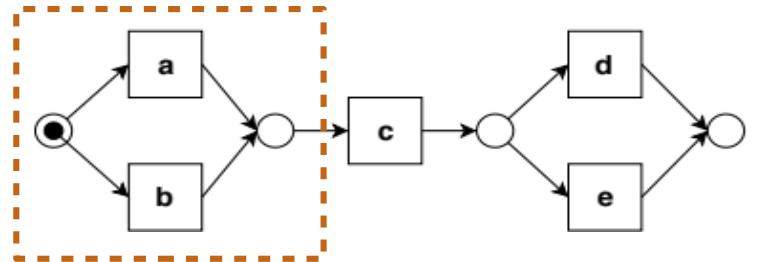
- Long-term dependency

$$L_3 := \{ < \mathbf{a1}, b, c, \mathbf{d1}, e1, e2, f >^{50, pos}, \\ < \mathbf{a2}, b, c, \mathbf{d2}, e1, e2, f >^{50, pos}; \\ < \mathbf{a1}, b, c, \mathbf{d2}, e1, e2, f >^{50, neg}, \\ < \mathbf{a2}, b, c, \mathbf{d1}, e1, e2, f >^{50, neg} \}$$



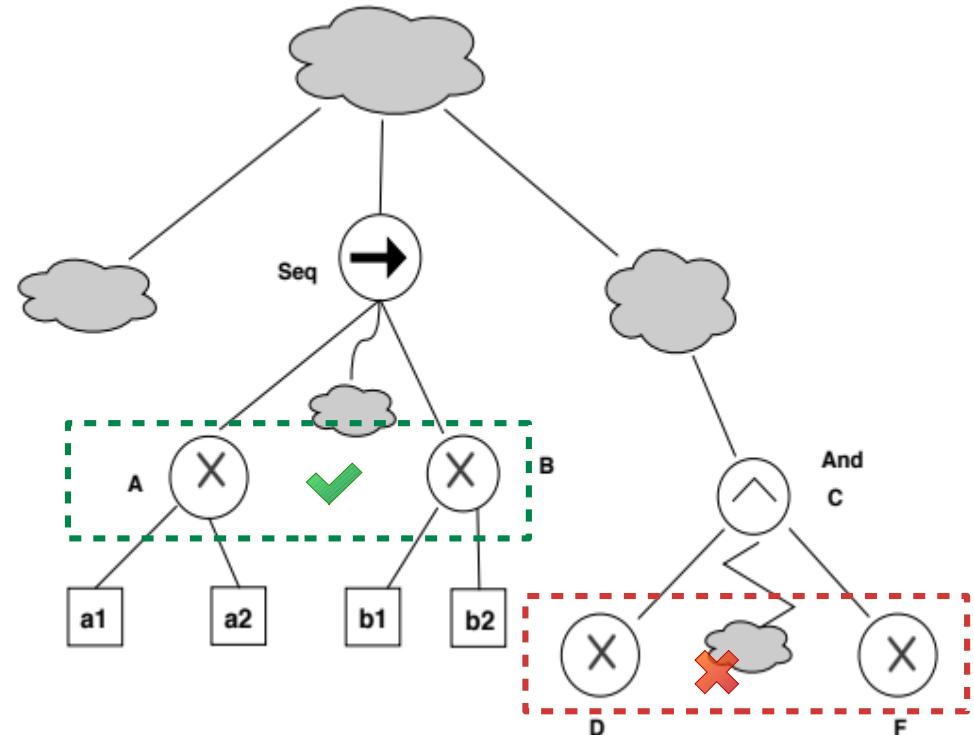
Post Process Petri net

- Long-term dependency
 - Exclusive choices
 - ✓ xor blocks/branches



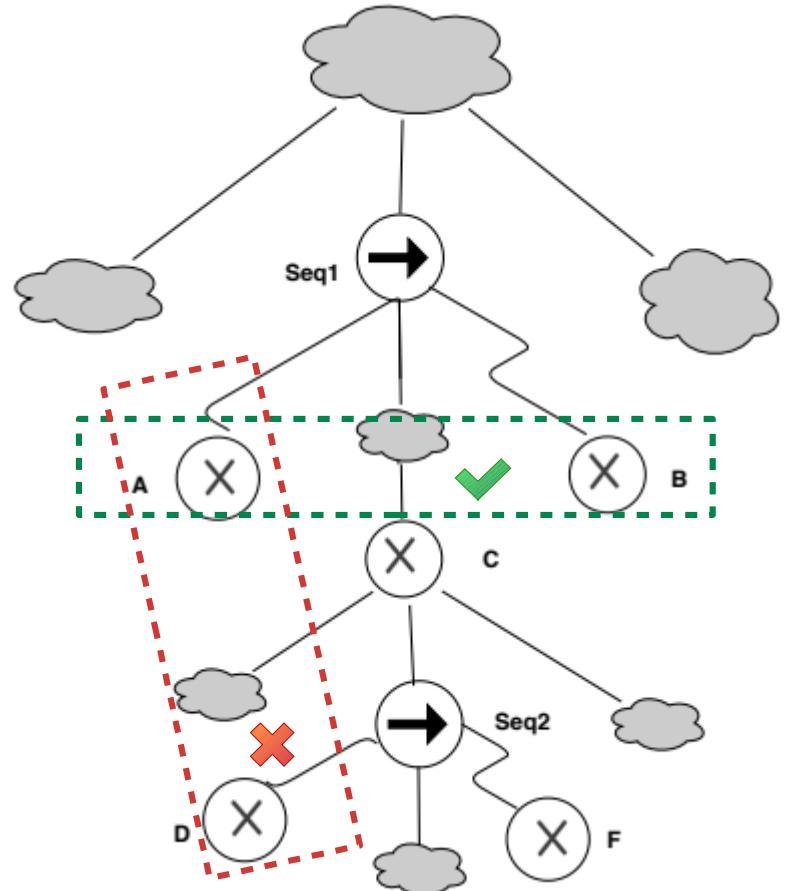
Post Process Petri net

- Long-term dependency
 - Exclusive choices
 - ✓ xor blocks/branches
 - ✓ In order: Least common ancestor is Seq
- A < B, !D< F



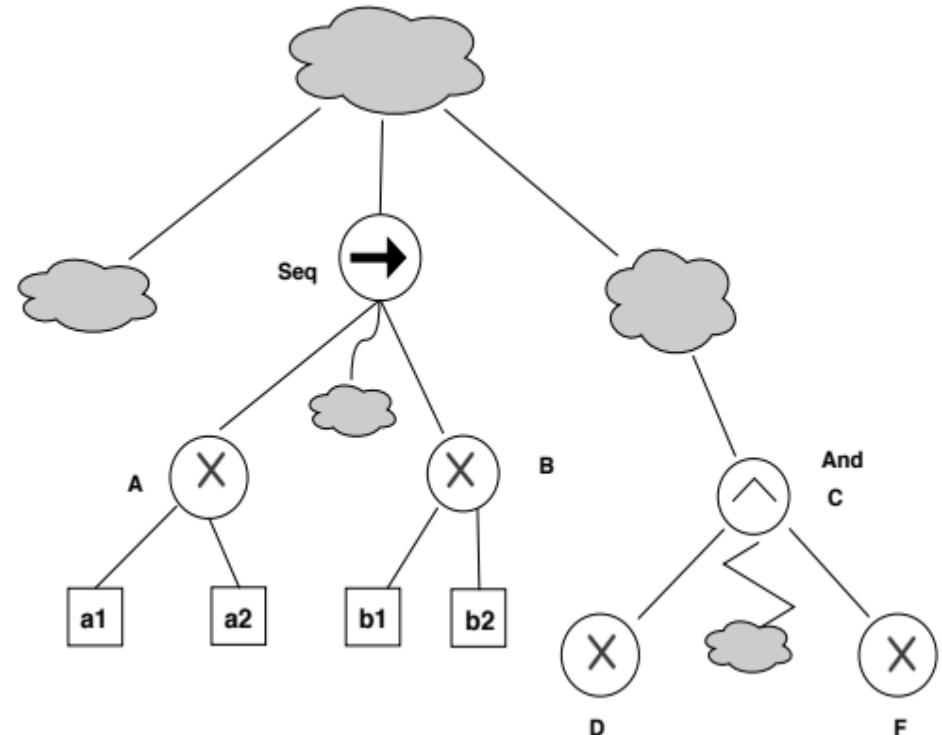
Post Process Petri net

- Long-term dependency
 - Exclusive choices
 - ✓ xor blocks/branches
 - ✓ In order: Least common ancestor is Seq
 $A < C < B, D < F$
 - ✓ In same level
 - A, B pair, D,F not pair



Post Process Petri net

- Long-term dependency
 - Exclusive choices
 - ✓ xor blocks/branches
 - ✓ In order: Least common ancestor is Seq
 $A < C < B, D < F$
 - ✓ In same level
 - A, B pair, D,F not pair
 - Strong correlation



Post Process Petri net

- Long-term dependency
 - Strong correlation: frequently coexist

Frequency of multiple xor branches is
 $f_L(x_i, y_j) = \sum_{\sigma \in L} |\{\sigma | \sigma \models x_i \wedge \sigma \models y_j\}|$

$$d^+(x_i, y_j) = \begin{cases} 0, & \text{if } \sum_{y_k \in T} f_{L^+}(x_i, y_k) = 0; \\ \frac{f_{L^+}(x_i, y_j)}{\sum_{y_k \in T} f_{L^+}(x_i, y_k)}, & \text{otherwise} \end{cases}$$

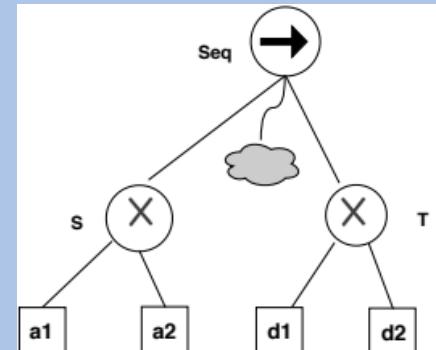
$$d^-(x_i, y_j) = \begin{cases} 0, & \text{if } \sum_{y_k \in T} f_{L^-}(x_i, y_k) = 0; \\ \frac{f_{L^-}(x_i, y_j)}{\sum_{y_k \in T} f_{L^-}(x_i, y_k)}, & \text{otherwise} \end{cases}$$

$$d(x_i, y_j) = w^+ \cdot d^+(x_i, y_j) - w^- \cdot d^-(x_i, y_j) > t$$

$$\begin{aligned} L_3 := & \{<\mathbf{a1}, b, c, \mathbf{d1}, e1, e2, f>^{50, pos}, \\ & <\mathbf{a2}, b, c, \mathbf{d2}, e1, e2, f>^{50, pos}; \\ & <\mathbf{a1}, b, c, \mathbf{d2}, e1, e2, f>^{50, neg}, \\ & <\mathbf{a2}, b, c, \mathbf{d1}, e1, e2, f>^{50, neg}\} \end{aligned}$$

$$\begin{aligned} w^+ &= 1, w^- = 1, t = 0 \\ d(a1, d1) &= d^+(a1, d1) - d^-(a1, d1) \\ &= 50/50 - 0 = 1; \\ d(a1, d2) &= d^+(a1, d2) - d^-(a1, d2) \\ &= 0 - 50/50 = -1; \\ d(a2, d1) &= -1; \\ d(a2, d2) &= 1. \end{aligned}$$

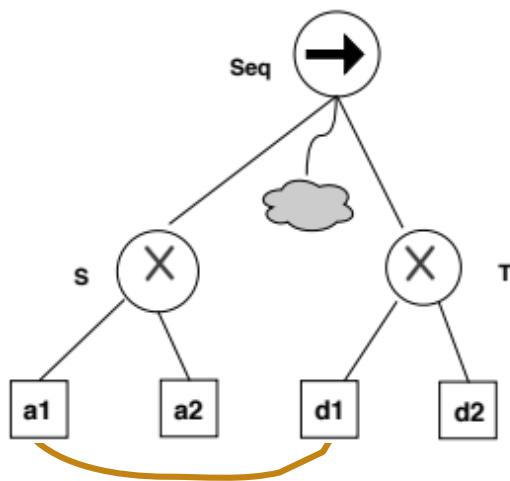
$$LT = \{ a1 \rightsquigarrow d1, \\ a2 \rightsquigarrow d2 \}.$$



Algorithm – add long-term dependency

- express on process tree

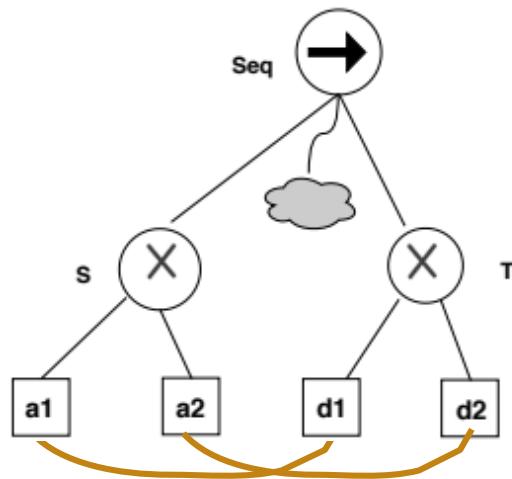
$LT = \{ a1 \rightsquigarrow d1,$
 $a2 \rightsquigarrow d2 \}.$



Algorithm – add long-term dependency

- express on process tree

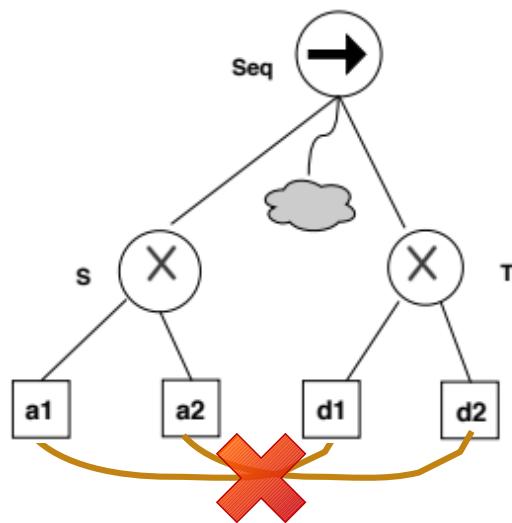
$LT = \{ a1 \rightsquigarrow d1,$
 $a2 \rightsquigarrow d2 \}.$



Algorithm – add long-term dependency

- express on process tree

$LT = \{ a1 \rightsquigarrow d1,$
 $a2 \rightsquigarrow d2 \}.$

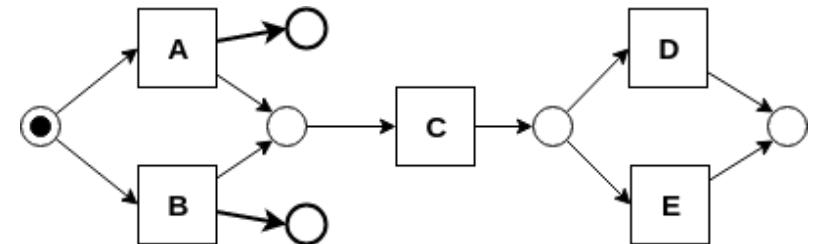
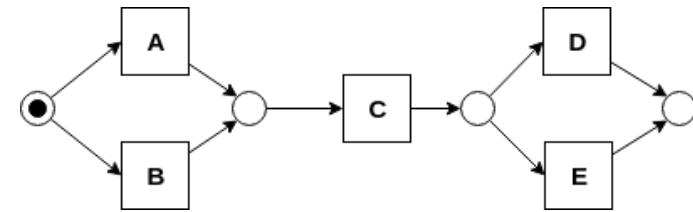


Algorithm – add long-term dependency

- express on Petri net
 - ✓ Add silent transition

- Add control place as post-place post after $S=\{A,B\}$

$$S = \{A, B\}, T = \{D, E\}, LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow E\}.$$

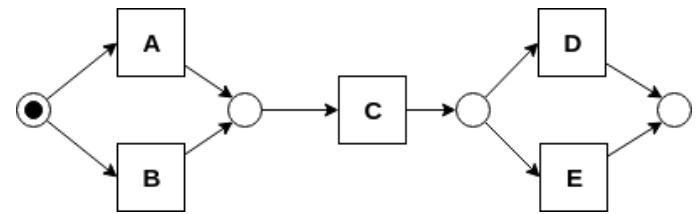


Algorithm – add long-term dependency

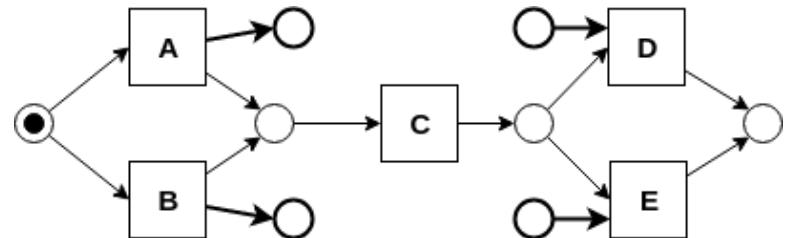
- How to express on Petri net

- ✓ Add silent transition
 - Add control place as post-place post after $S=\{A,B\}$

- Add control place as pre-place before $T=\{D,E\}$



$$LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow E\}.$$



Algorithm – add long-term dependency

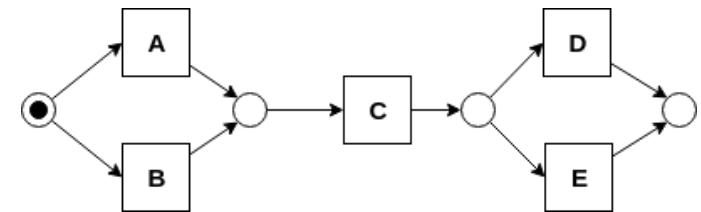
- How to express on Petri net

- ✓ Add silent transition

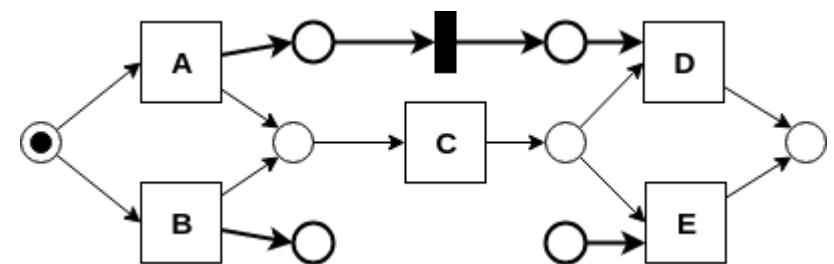
- Add control place as post-place post after $S=\{A,B\}$

- Add control place as pre-place before $T=\{D,E\}$

- Add silent transitions for each long-term dependency



$$LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow E\}.$$



Algorithm – add long-term dependency

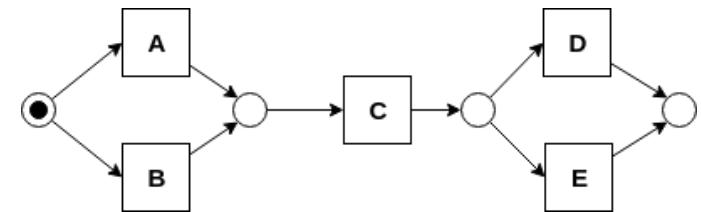
- How to express on Petri net

- ✓ Add silent transition

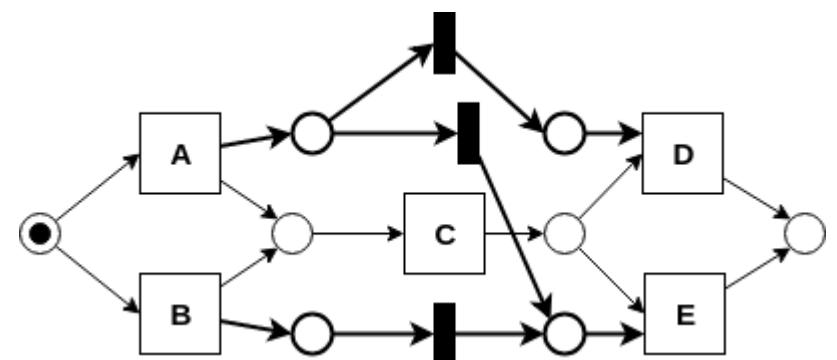
- Add control place as post-place post after S

- Add control place as pre-place before T

- Add silent transitions for each long-term dependency



$$LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow E\}.$$



Algorithm – add long-term dependency

- Long-term dependency Situations

1. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow D, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\}, |LT| = |S| \cdot |T|$.

2. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\} LT_S = S$ and $LT_T = T, |LT| < |S| \cdot |T|$.

3. $LT = \{A \rightsquigarrow D, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\} LT_S = S$ and $LT_T = T, |LT| < |S| \cdot |T|$.

4. $LT = \{A \rightsquigarrow D, B \rightsquigarrow D\}$.

$LT_S = S, LT_T \subsetneq T$.

5. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E\}$.

$LT_S \subsetneq S, LT_T = T$.

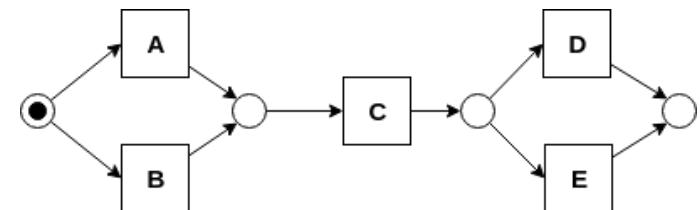
6. $LT = \{A \rightsquigarrow E\}$.

$LT_S \subsetneq S, LT_T \subsetneq T$.

7. $LT = \emptyset$

$$LT_S := \{X_i \mid \exists Y_j, X_i \rightsquigarrow Y_j \in LT\}$$

$$LT_T := \{Y_j \mid \exists X_i, X_i \rightsquigarrow Y_j \in LT\}$$



- Situation 1 is full dependency ==> no consideration
- Situation 7 is empty. ==> no consideration

Algorithm – add long-term dependency

- Long-term dependency Situations

1. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow D, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\}, |LT| = |S| \cdot |T|$.

2. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\} LT_S = S$ and $LT_T = T, |LT| < |S| \cdot |T|$.

3. $LT = \{A \rightsquigarrow D, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\} LT_S = S$ and $LT_T = T, |LT| < |S| \cdot |T|$.

4. $LT = \{A \rightsquigarrow D, B \rightsquigarrow D\}$.

$LT_S = S, LT_T \subsetneq T$.

5. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E\}$.

$LT_S \subsetneq S, LT_T = T$.

6. $LT = \{A \rightsquigarrow E\}$.

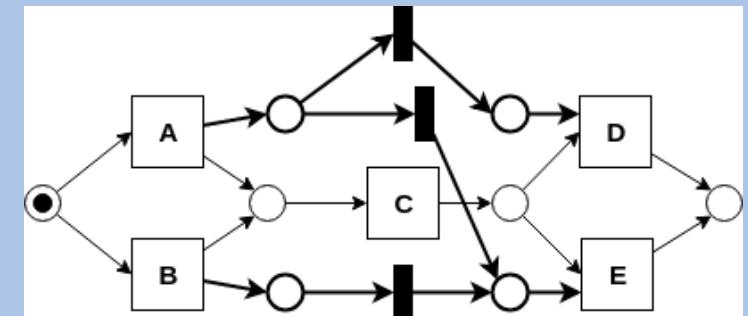
$LT_S \subsetneq S, LT_T \subsetneq T$.

7. $LT = \emptyset$

$$LT_S := \{X_i \mid \exists Y_j, X_i \rightsquigarrow Y_j \in LT\}$$

$$LT_T := \{Y_j \mid \exists X_i, X_i \rightsquigarrow Y_j \in LT\}$$

$$LT_S = S, LT_T = T$$



Soundness

- ✓ Safeness.
- ✓ Proper completion.
- ✓ Option to complete.
- ✓ No dead parts.

Algorithm – add long-term dependency

- **Long-term dependency Situations**

1. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow D, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\}, |LT| = |S| \cdot |T|$.

2. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\} LT_S = S$ and $LT_T = T, |LT| < |S| \cdot |T|$.

3. $LT = \{A \rightsquigarrow D, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\} LT_S = S$ and $LT_T = T, |LT| < |S| \cdot |T|$.

4. $LT = \{A \rightsquigarrow D, B \rightsquigarrow D\}$.

$LT_S = S, LT_T \subsetneq T$.

5. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E\}$.

$LT_S \subsetneq S, LT_T = T$.

6. $LT = \{A \rightsquigarrow E\}$.

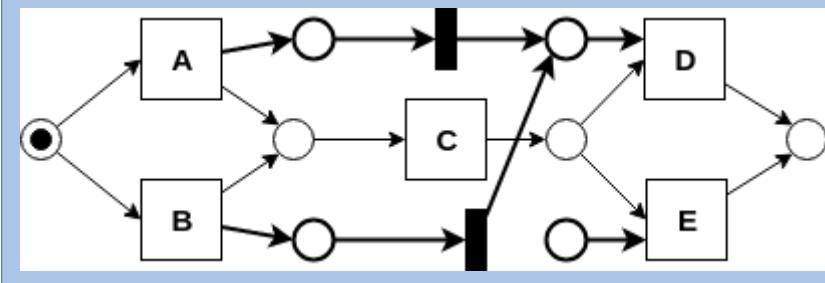
$LT_S \subsetneq S, LT_T \subsetneq T$.

7. $LT = \emptyset$

$$LT_S := \{X_i \mid \exists Y_j, X_i \rightsquigarrow Y_j \in LT\}$$

$$LT_T := \{Y_j \mid \exists X_i, X_i \rightsquigarrow Y_j \in LT\}$$

$$LT_S \subsetneq S, LT_T \subsetneq T$$



Soundness

- ✓ Safeness.
- ✓ Proper completion.
- ✓ Option to complete.
- ✓ No dead parts.

Algorithm – add long-term dependency

- **Long-term dependency Situations**

1. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow D, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\}, |LT| = |S| \cdot |T|$.

2. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\} LT_S = S$ and $LT_T = T, |LT| < |S| \cdot |T|$.

3. $LT = \{A \rightsquigarrow D, B \rightsquigarrow E\}$.

$LT_S = \{A, B\}, LT_T = \{D, E\} LT_S = S$ and $LT_T = T, |LT| < |S| \cdot |T|$.

4. $LT = \{A \rightsquigarrow D, B \rightsquigarrow D\}$.

$LT_S = S, LT_T \subsetneq T$.

5. $LT = \{A \rightsquigarrow D, A \rightsquigarrow E\}$.

$LT_S \subsetneq S, LT_T = T$.

6. $LT = \{A \rightsquigarrow E\}$.

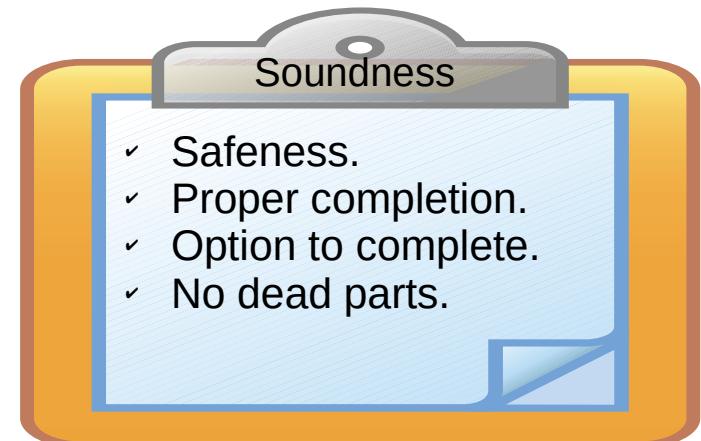
$LT_S \subsetneq S, LT_T \subsetneq T$.

7. $LT = \emptyset$

$$LT_S := \{X_i \mid \exists Y_j, X_i \rightsquigarrow Y_j \in LT\}$$

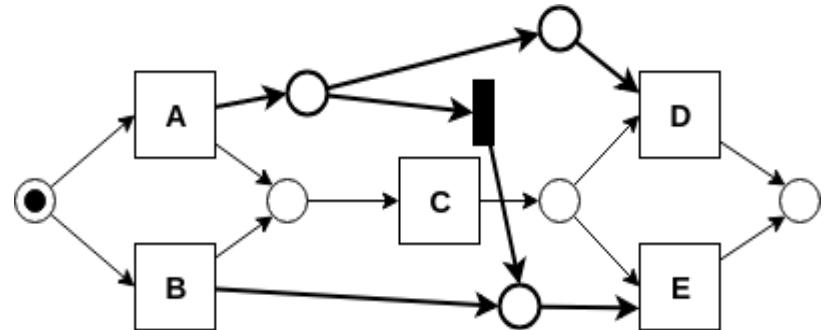
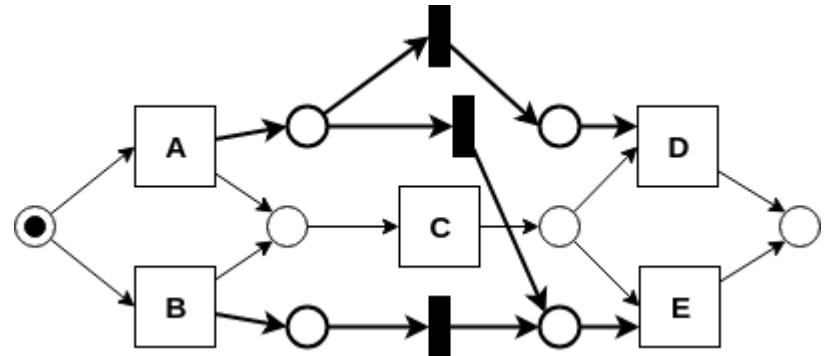
$$LT_T := \{Y_j \mid \exists X_i, X_i \rightsquigarrow Y_j \in LT\}$$

$LT_S = S, LT_T = T, |LT| < |S| \cdot |T|$

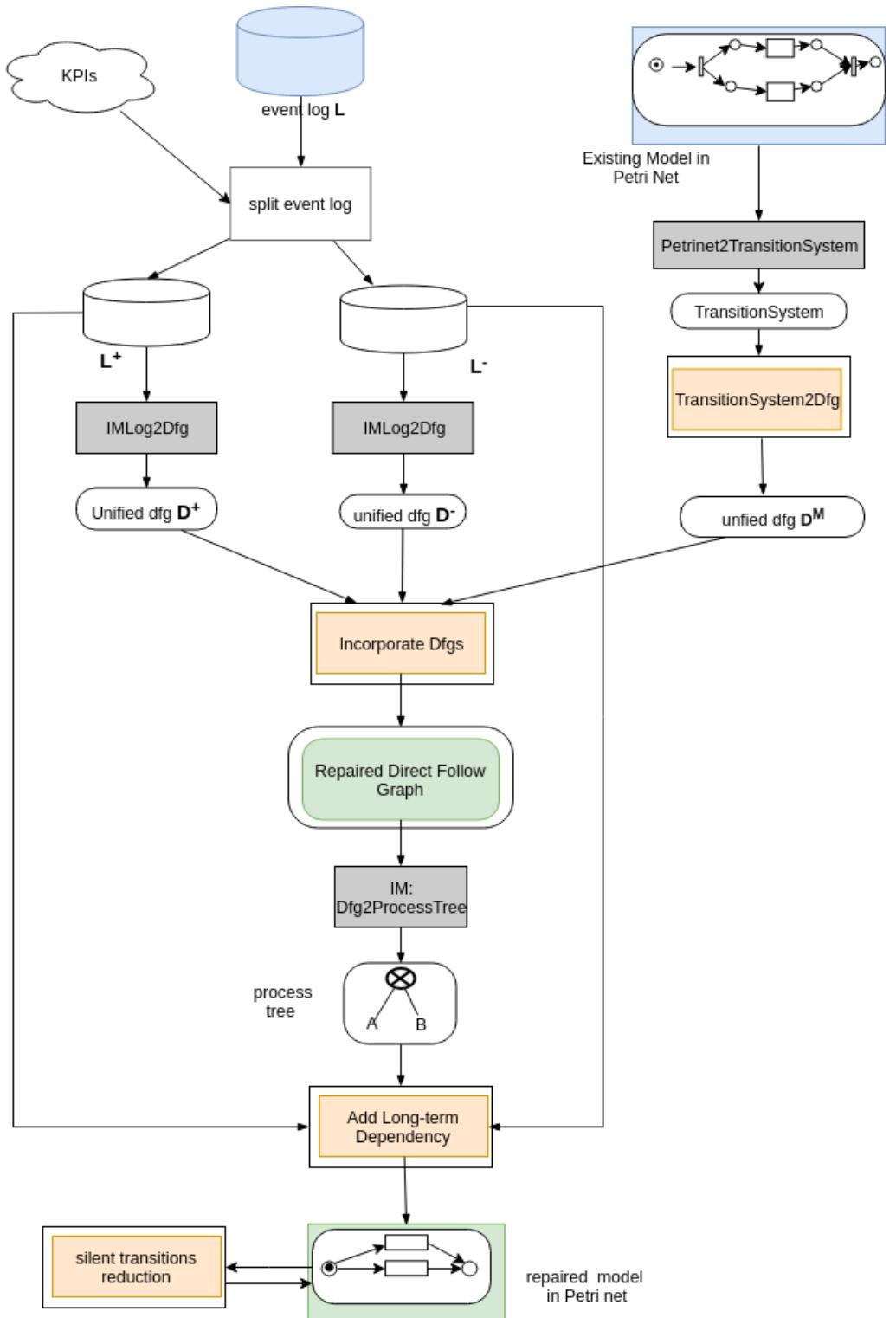


Post process

- Delete redundant silent transitions



Algorithm – architecture



Outline

- Motivation for Research
- Problem Definition
- Approach
- Demo Show
 - ProM
 - KNIME
- Evaluation
- Conclusion



Demo

(3) change Model type

ProM

Directly follows graph of seq_3_xor

Select visualisation ...

Generated Model

(1). repaired model view

(4) change weights

Show Process Tree
Show Petri net
Show Petri net with It
Show Petri net with LT After Reduc...

Set Weights

Weight for Existing Model	Weight for Pos Examples	Weight for Neg Examples
0.7	1.0	1.0

Reset Submit

Add Long-term Dependency on Petri...
Select Add Method
Add All In Order
Add XOR Pair By Choice
Choose XOR Pair To Add Or Remove
Chosse XOR Pair To Add LT
Add this pair

(2). control panel

Demo

Directly follows graph of seq_3_xor

Generated Model:

Select visualisation ...

Show Process Tree
Show Petri net
Show Petri net with LT
Show Petri net with LT After Reduc...

Set Weights

Weight for Existing Model	weight for Pos Examples	Weight for Neg Examples
0.7	1.0	1.0

Reset Submit

Add Long-term Dependency on Petri...
Select Add Method
 Add All In Order
 Add XOR Pair By Choice

Choose XOR Pair To Add Or Remove
Choose XOR Pair To Add LT
Add this pair
Choose Source
Choose Target

Choose Pair to Remove
Remove this pair
Choose Source
Choose Target

(5) long-term dependency control panel

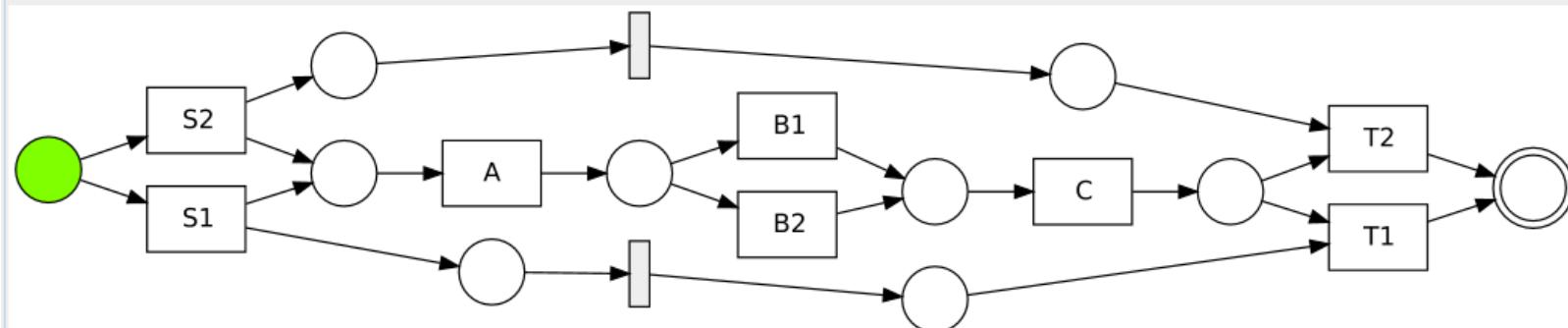
Demo

Directly follows graph of seq_3_xor

Select visualisation ...



Generated Model



(5) long-term
dependency
control panel
manually

Show Process Tree
Show Petri net
 Show Petri net with Lt
Show Petri net with LT After Reduc...

Set Weights

Weight for Existing Model	Weight for Pos Examples	Weight for Neg Examples
0.4	1.0	0.8

Add Long-term Dependency on Petri...
Select Add Method
 Add All In Order
 Add XOR Pair By Choice

Choose XOR Pair To Add Or Remove
Chosse XOR Pair To Add LT

Choose Source Xor(B2, B1)
Choose Target

Choose Pair to Remove

Choose Source Xor(S2,...)
Choose Target Xor(T2,...)

Result In Confusion Matrix

Demo

Directly follows graph of seq_3_xor

Select visualisation ...

(Generated Model)

(6)reduce silent transitions

Show Process Tree
Show Petri net
Show Petri net with It
Show Petri net with LT After Reduc...

Set Weights

Weight for Existing Model	Weight for Pos Examples	Weight for Neg Examples
0.7	1.0	1.0

Reset Submit

Add Long-term Dependency on Petri...
Select Add Method
 Add All In Order
 Add XOR Pair By Choice

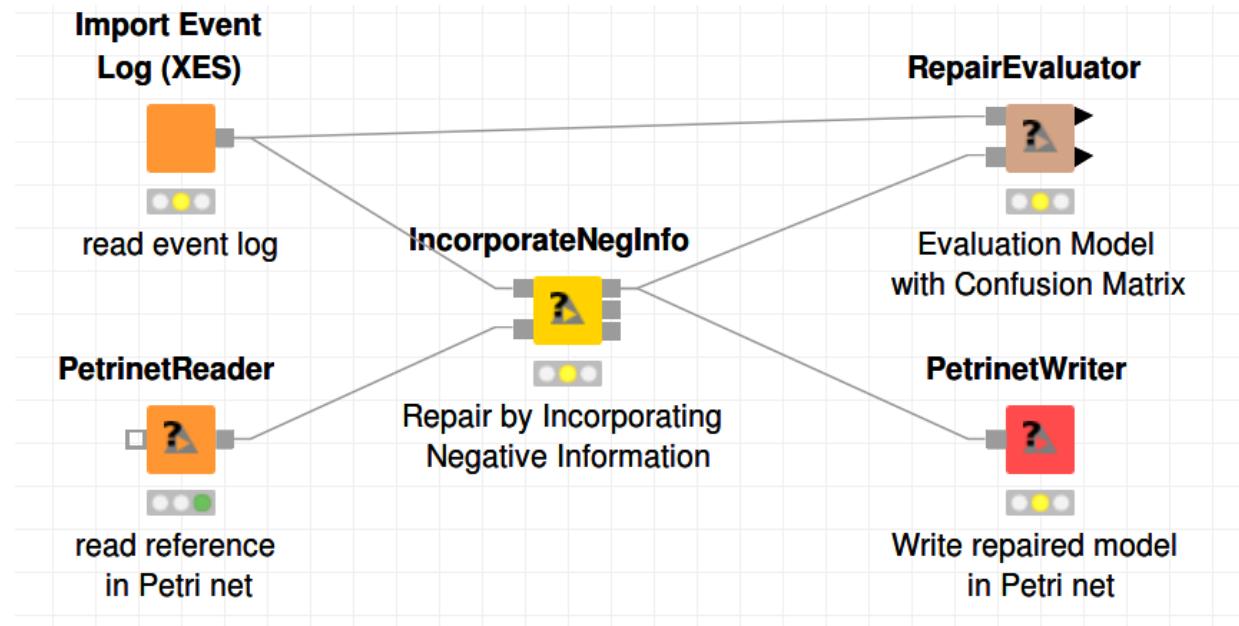
Choose XOR Pair To Add Or Remove
Chosse XOR Pair To Add LT
Add this pair

The diagram illustrates a Petri net model with tokens (green circles), places (white circles), and transitions (rectangular boxes). The model consists of several XOR pairs and long-term dependencies:

- XOR Pairs:** S1 and S2 are XORed before transitioning to A. A and B1 are XORed before transitioning to B1. B1 and B2 are XORed before transitioning to B2. C and T1 are XORed before transitioning to T1.
- Long-term Dependencies:** There are two long-term dependencies:
 - A dependency from S1 to T2.
 - A dependency from B2 to T2.
- Transitions:** Transitions A, B1, B2, C, and T1 are shown as rectangular boxes. Transition T2 has a self-loop.
- Initial State:** A green token is present in place S1.

Demo

Integration with KNIME



Outline

- Motivation for Research
 - Problem Definition
 - Approach
 - Demo
 - Evaluation
 - Synthetic data
 - Real life data
 - Conclusion



Evaluation

Confusion matrix

	Allowed behavior	Not allowed behavior
positive	TP	FN
negative	FP	TN

- **Confusion matrix**

- Recall

$$Recall = \frac{TP}{TP + FN}$$

- Precision

$$Precision = \frac{TP}{TP + FP}$$

- Accuracy

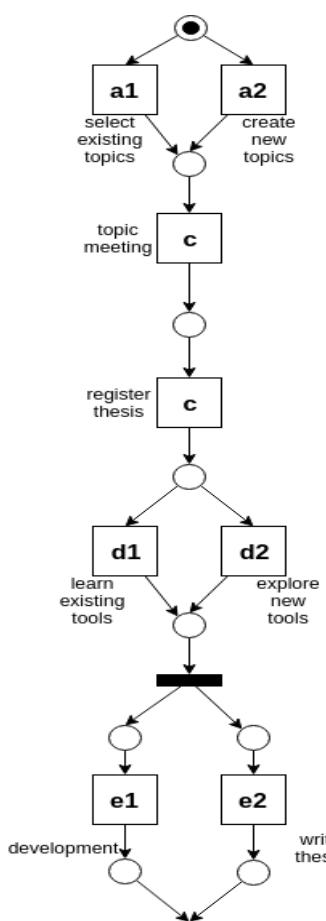
$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- F1

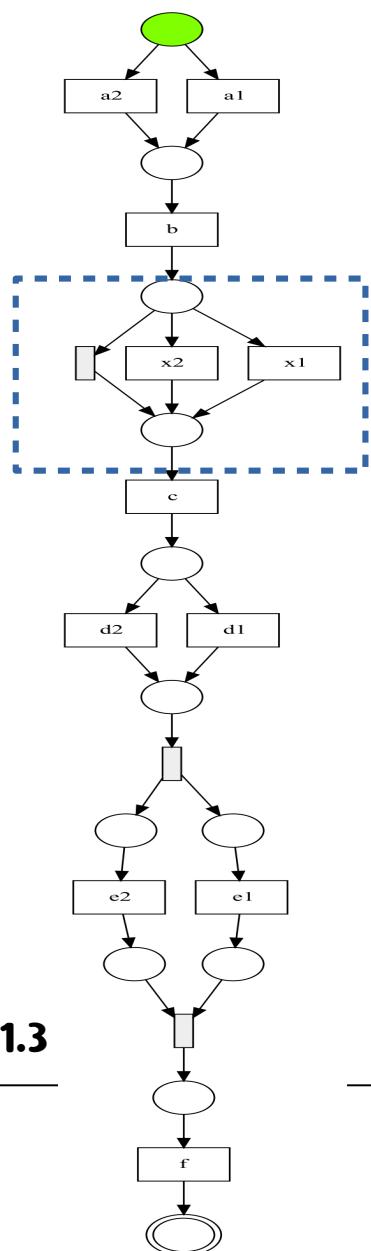
$$F_1 = \frac{2 * Recall * Precision}{Precision + Recall}$$

Demo --result

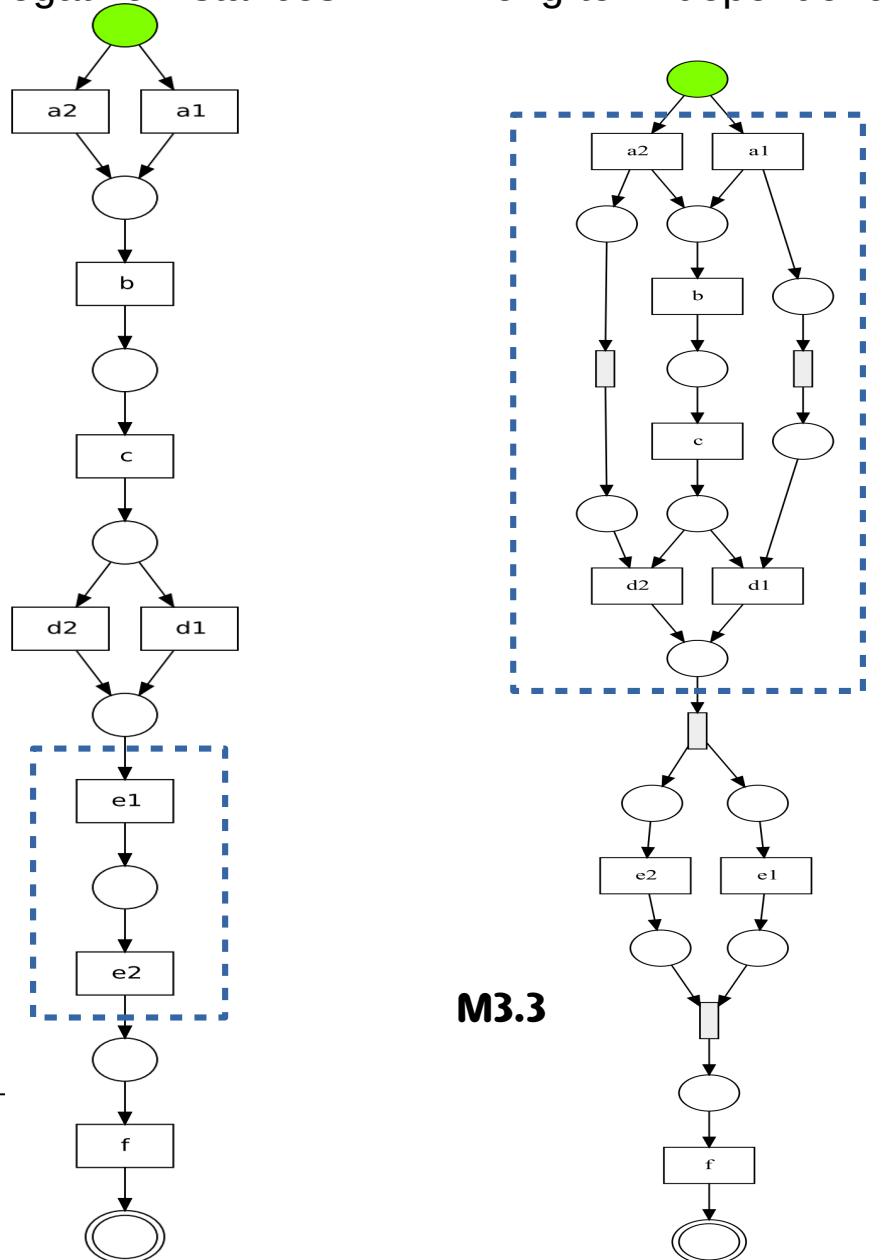
Overcome s1:
add subprocesses
as loops



Overcome s2:
unable to adapt model
With negative instances



Overcome s3:
unable to detect
long-term dependency



Demo --result

Situation	Method	Generated Model	Confusion matrix measurements							
			TP	FP	TN	FN	recall	precision	accuracy	F1
S1	IM-Infrequent Noise threshold: 20%	M1.1	50	50	0	0	1	0.5	0.5	0.667
	Fahland's Repair Model	M1.2	50	50	0	0	1	0.5	0.5	0.667
	Dfg-repair	M1.3	50	50	0	0	1	0.5	0.5	0.667
S2	IM/Fahland's	M0	60	45	0	0	1	0.571	0.571	0.727
	Dfg-repair	M2.3	50	5	40	10	0.833	0.909	0.857	0.870
S3	IM/Fahland repair	M0	100	100	0	0	1	0.5	0.5	0.667
	Dfg-repair	M3.3	100	0	100	0	1	1	1	1

Conclusion:

- ✓ Conquer shortcomings of current techniques in listed situations,
 - ✓ Better precision, accuracy, F1 score
-

Experiments -- Real life data

- Data description

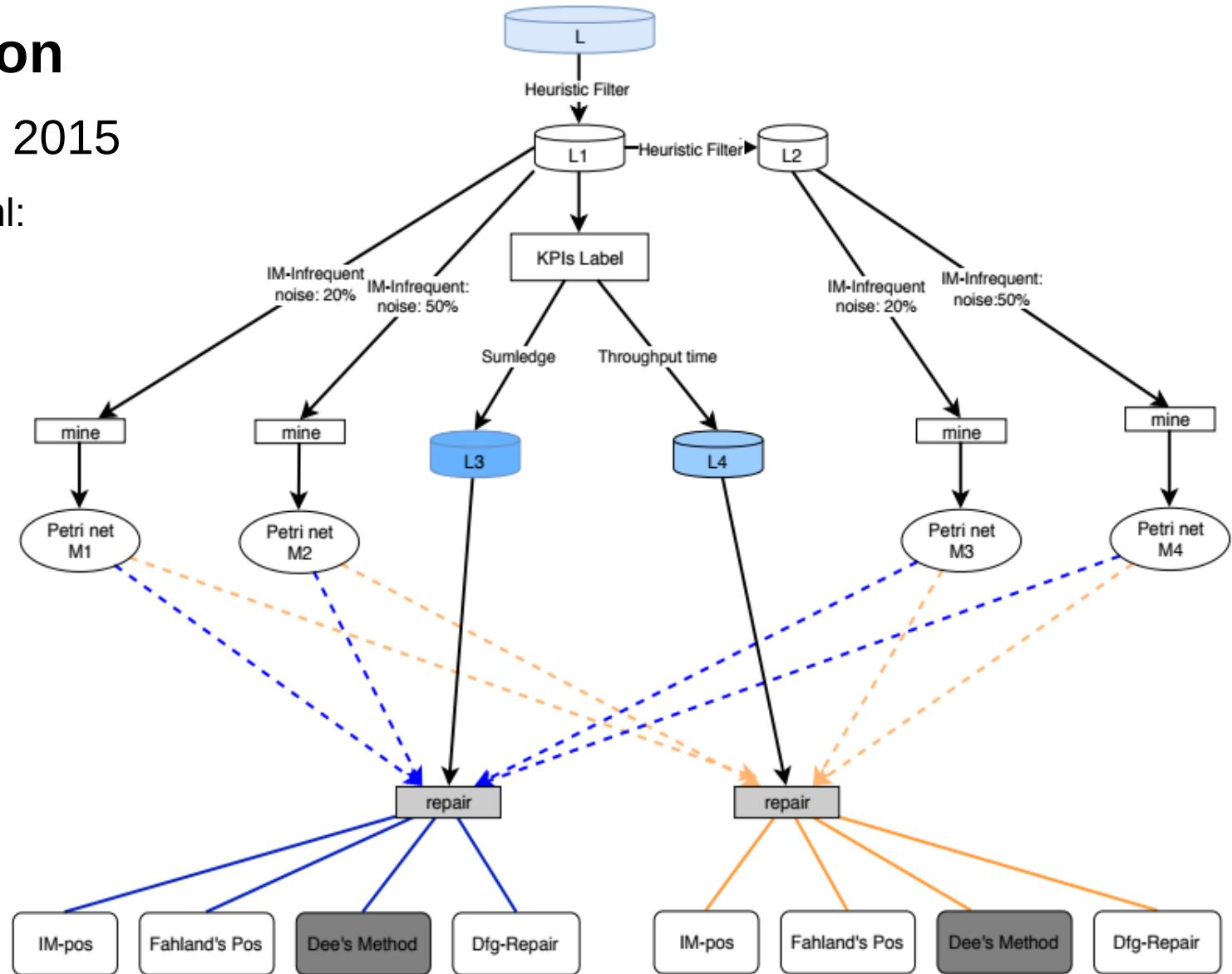
- BPI Challenge 2015

BPIC15_1.xes.xml:

1199 cases,

52217 events

398 event classes



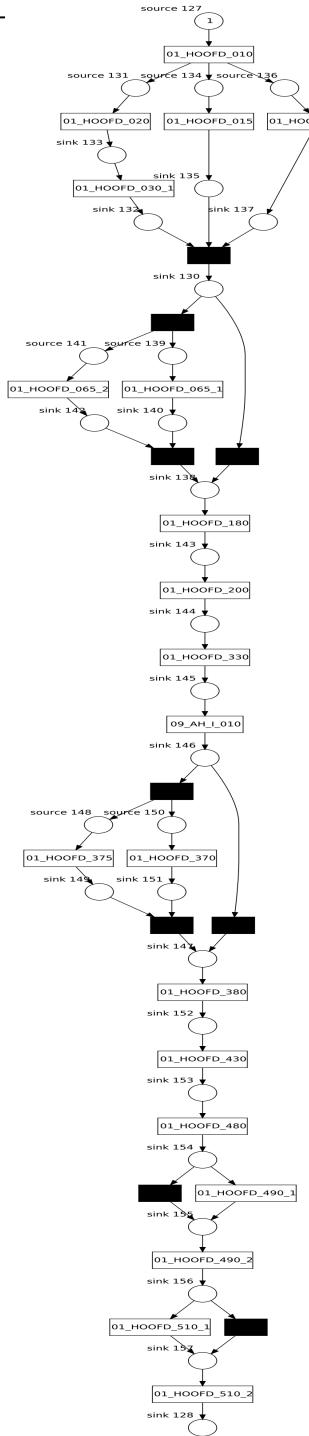
Experiments

- Event logs

ID	Description	Traces Num	Events Num	Event Class
D1	Heuristic filter 40%	495	9565	20
D2	Heuristic filter 60% on D1	378	4566	12
D3.1	Classify on Sumledge; Below 70% as positive	349	6744	20
D3.2	Classify on Sumledge; over 70% as negative	146	2811	20
D3.3	Union of D3.1 and D3.2	495	9565	20
D4.1	Classify on throughput time; Below 70% as positive	346	6719	20
D4.2	Classify on Sumledge; over 70% as negative	146	2846	20
D4.3	Union of D4.1 and D4.2	495	9565	20

Experiments

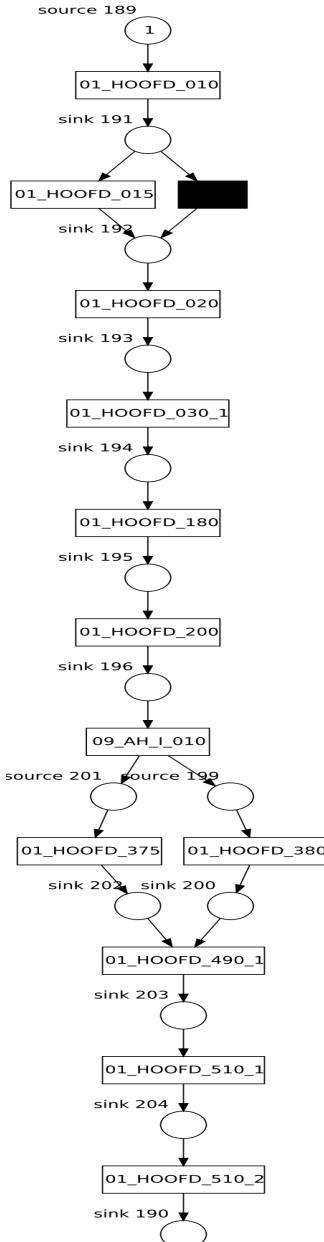
M1



Model ID	Data ID	Confusion matrix							
		TP	FP	TN	FN	recall	Precision	Accuracy	F1
M1	D3.3	112	40	106	237	0.321	0.737	0.440	0.447
	D4.3	131	21	128	215	0.379	0.862	0.523	0.526
M2	D3.3	106	39	107	243	0.304	0.731	0.430	0.429
	D4.3	125	20	129	221	0.361	0.862	0.513	0.509
M3	D3.3	0	0	146	349	0	NaN	0.295	0
	D4.3	0	0	149	346	0	NaN	0.301	0
M4	D3.3	0	0	146	349	0	NaN	0.295	0
	D4.3	0	0	149	346	0	NaN	0.301	0

Experiments

- Petri net Models

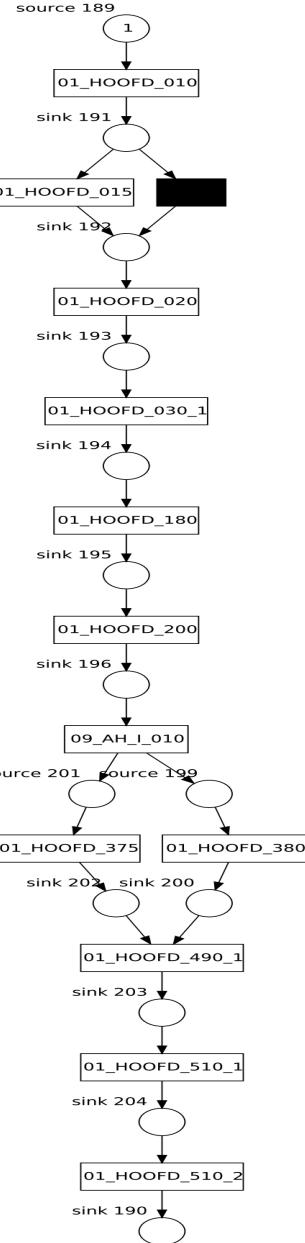


M3

Model ID	Data ID	Confusion matrix							
		TP	FP	TN	FN	recall	Precision	Accuracy	F1
M1	D3.3	112	40	106	237	0.321	0.737	0.440	0.447
	D4.3	131	21	128	215	0.379	0.862	0.523	0.526
M2	D3.3	106	39	107	243	0.304	0.731	0.430	0.429
	D4.3	125	20	129	221	0.361	0.862	0.513	0.509
M3	D3.3	0	0	146	349	0	NaN	0.295	0
	D4.3	0	0	149	346	0	NaN	0.301	0
M4	D3.3	0	0	146	349	0	NaN	0.295	0
	D4.3	0	0	149	346	0	NaN	0.301	0

Experiments- result

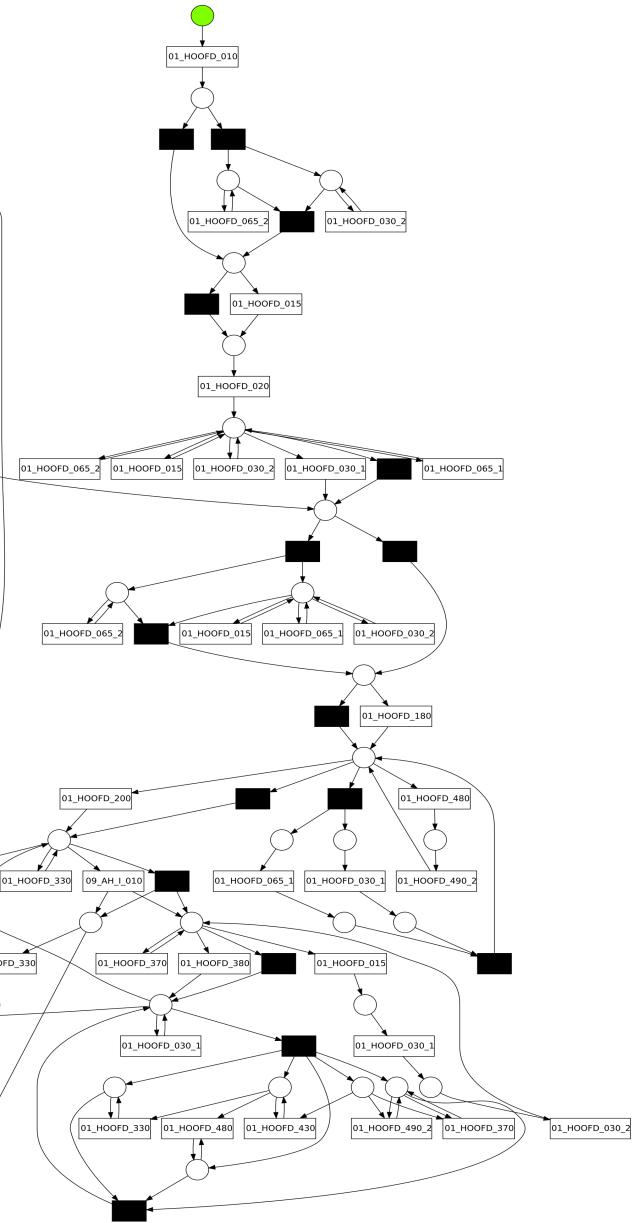
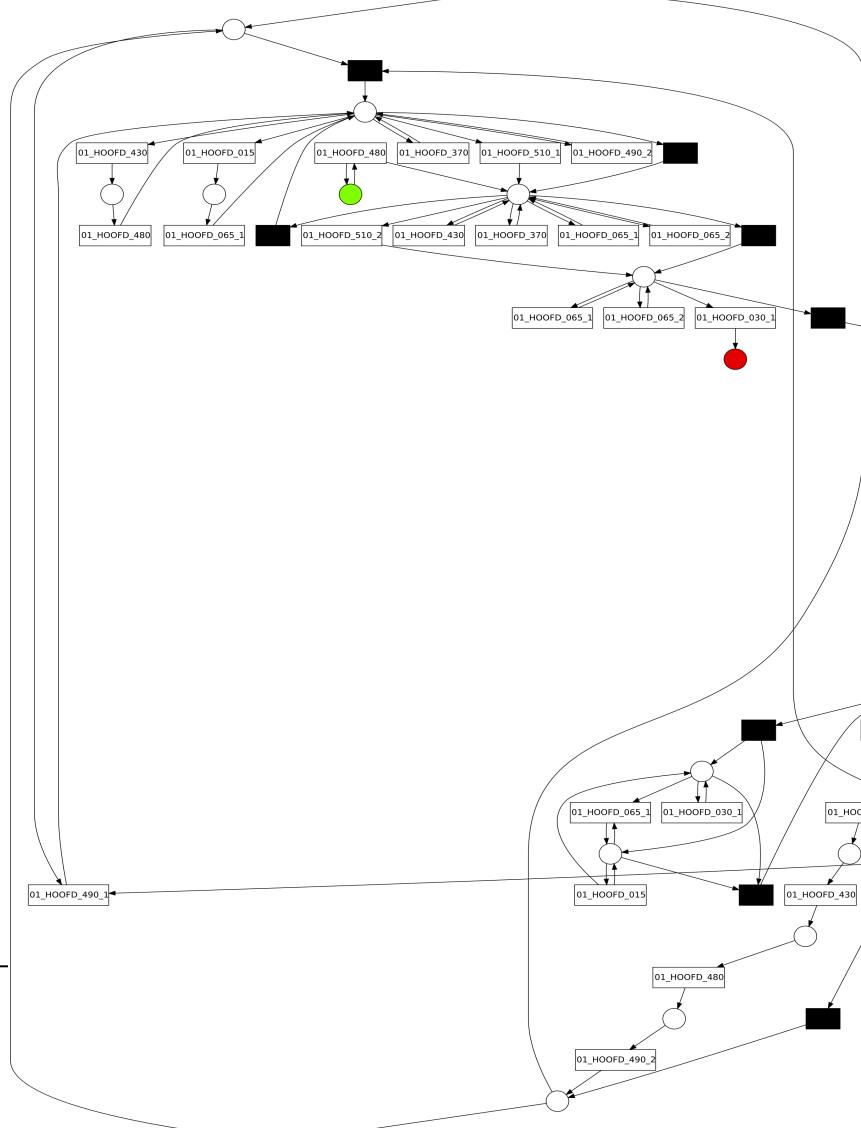
Repaired model M3.2 with Fahland's method on M3 with default setting



M3.2

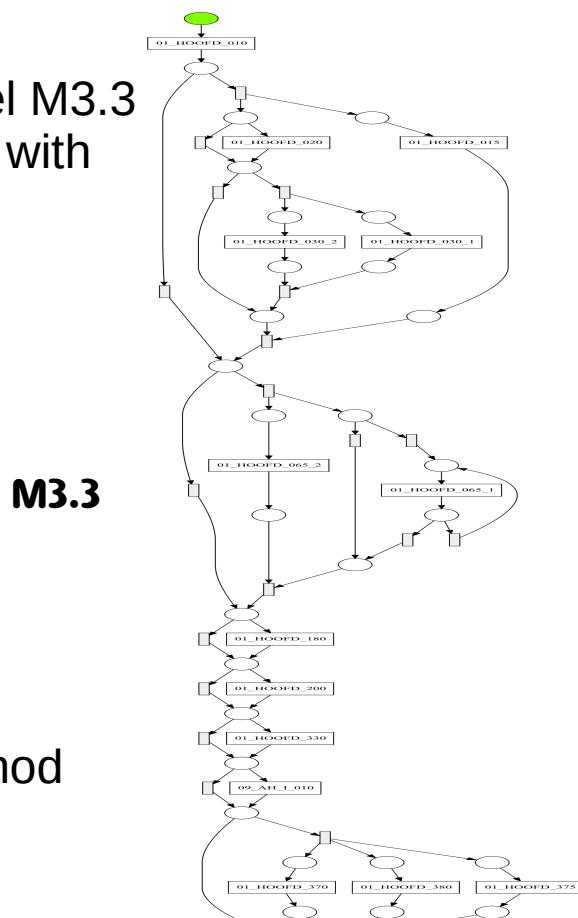
M3

30

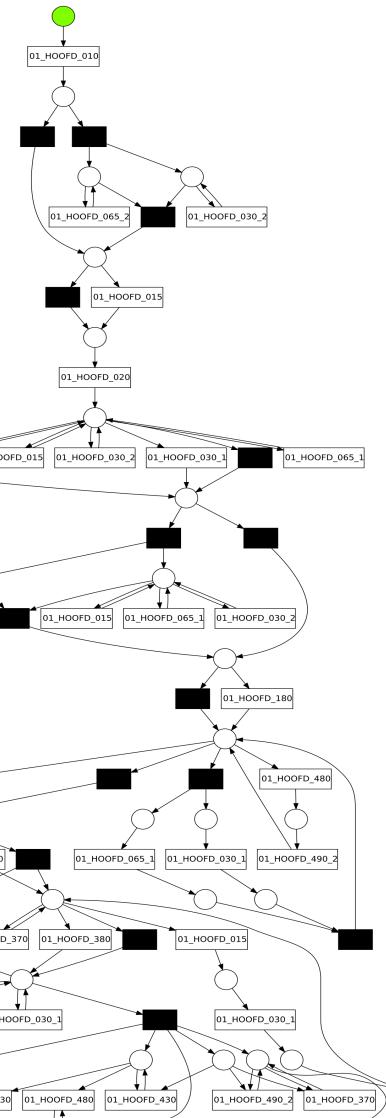
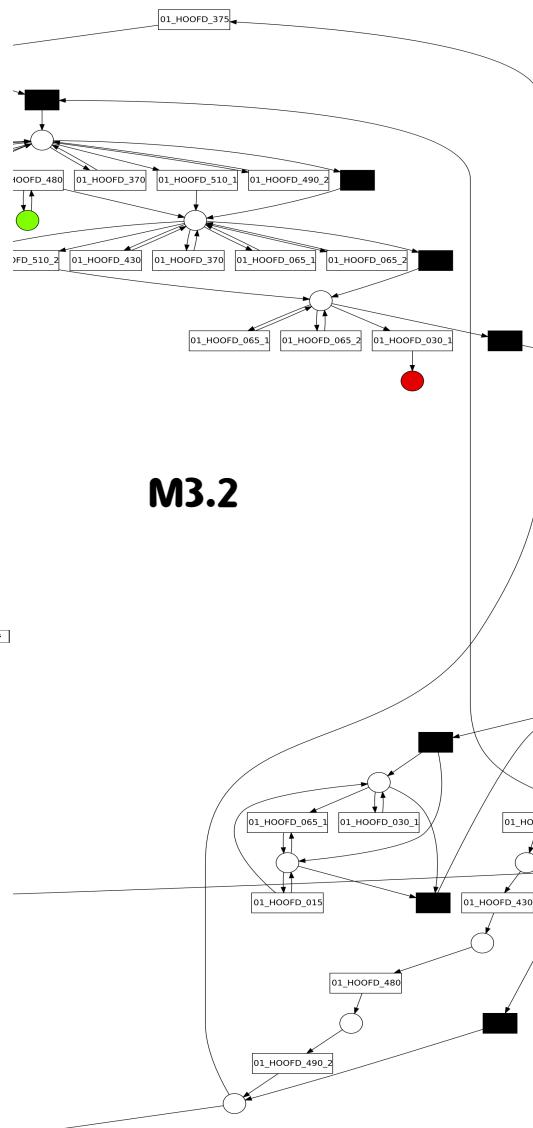
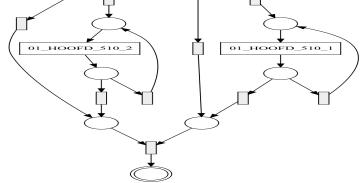


Experiments results

Repaired model M3.3
from dfg-repair with
default setting



Simpler than Fahland's method

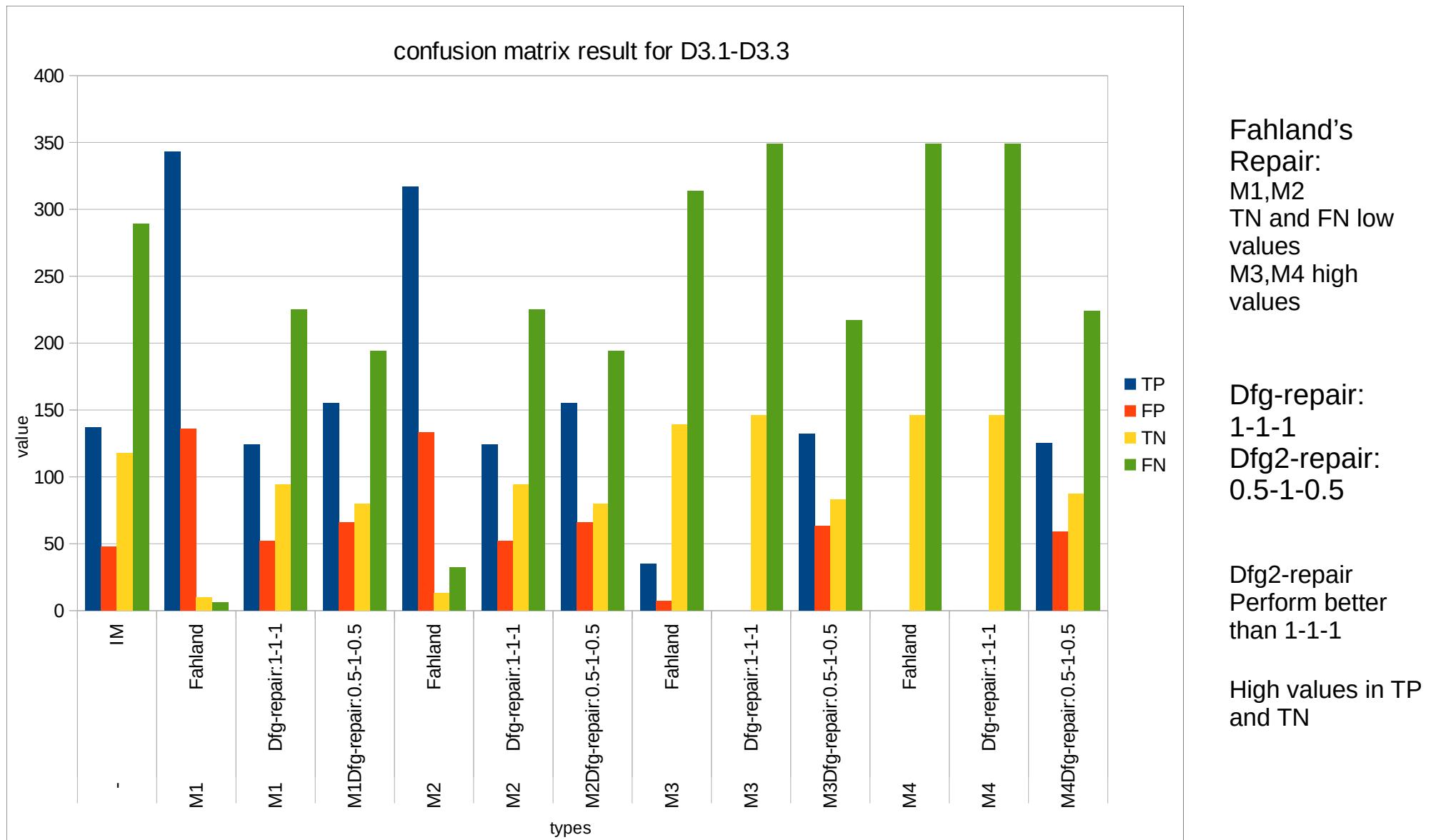


Experiment result

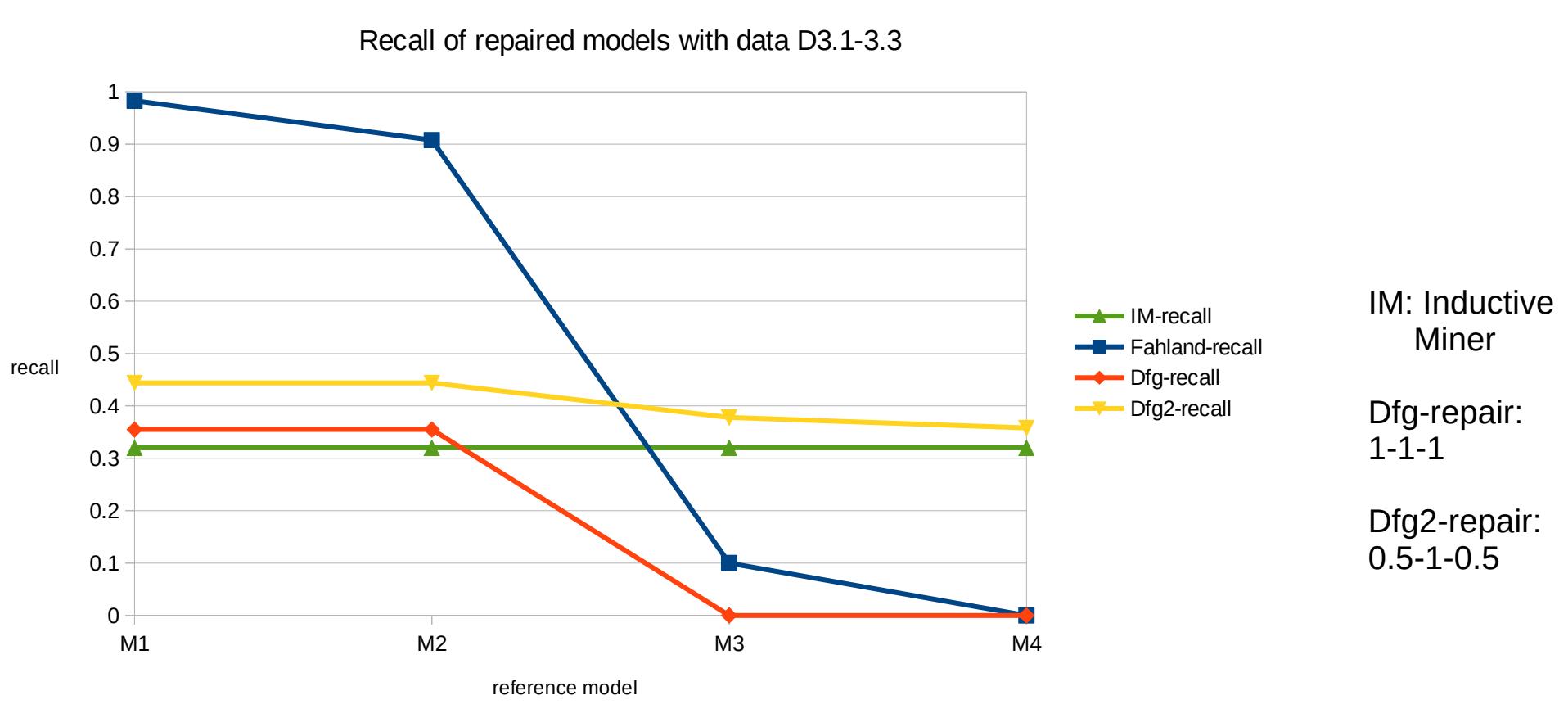
Table 6.5: Test Results on BPI15-M1 data

event log	reference model	method	confusion matrix metrics						
			TP	FP	TN	FN	recall	precision	accuracy
D3.1	-	IM	137	48	118	289	0.32	0.74	0.43
D3.1	M1	Fahland	343	136	10	6	0.983	0.716	0.713
D3.3	M1	Dfg-repair:1-1-1	124	52	94	225	0.355	0.705	0.44
D3.3	M1	Dfg-repair:0.5-1-0.5	155	66	80	194	0.444	0.701	0.474
D3.1	M2	Fahland	317	133	13	32	0.908	0.704	0.667
D3.3	M2	Dfg-repair:1-1-1	124	52	94	225	0.355	0.705	0.44
D3.3	M2	Dfg-repair:0.5-1-0.5	155	66	80	194	0.444	0.701	0.475
D3.1	M3	Fahland	35	7	139	314	0.100	0.833	0.352
D3.3	M3	Dfg-repair:1-1-1	0	0	146	349	0	NaN	0.295
D3.3	M3	Dfg-repair:0.5-1-0.5	132	63	83	217	0.378	0.677	0.434
D3.1	M4	Fahland	0	0	146	349	0	NaN	0.294
D3.3	M4	Dfg-repair:1-1-1	0	0	146	349	0	NaN	0.294
D3.3	M4	Dfg-repair:0.5-1-0.5	125	59	87	224	0.358	0.679	0.428
D4.1	-	IM	131	21	128	215	0.379	0.862	0.523
D4.1	M1	Fahland	325	133	16	21	0.939	0.710	0.689
D4.3	M1	Dfg-repair:1-1-1	139	36	113	207	0.402	0.794	0.509
D4.3	M1	Dfg-repair:0.5-1-0.5	172	48	101	174	0.497	0.782	0.552
D4.1	M2	Fahland	325	130	19	21	0.939	0.714	0.695
D4.3	M2	Dfg-repair:1-1-1	139	36	113	207	0.402	0.794	0.509
D4.3	M2	Dfg-repair:0.5-1-0.5	172	48	101	174	0.497	0.782	0.552
D4.1	M3	Fahland	10	20	129	336	0.029	0.333	0.281
D4.3	M3	Dfg-repair:1-1-1	0	0	346	149	0	NaN	0.303
D4.3	M3	Dfg-repair:0.5-1-0.5	182	49	164	100	0.526	0.788	0.70
D4.1	M4	Fahland	5	14	135	341	0.014	0.263	0.283
D4.3	M4	Dfg-repair:1-1-1	0	0	346	149	0	NaN	0.303
D4.3	M4	Dfg-repair:0.5-1-0.5	172	48	101	174	0.497	0.782	0.552

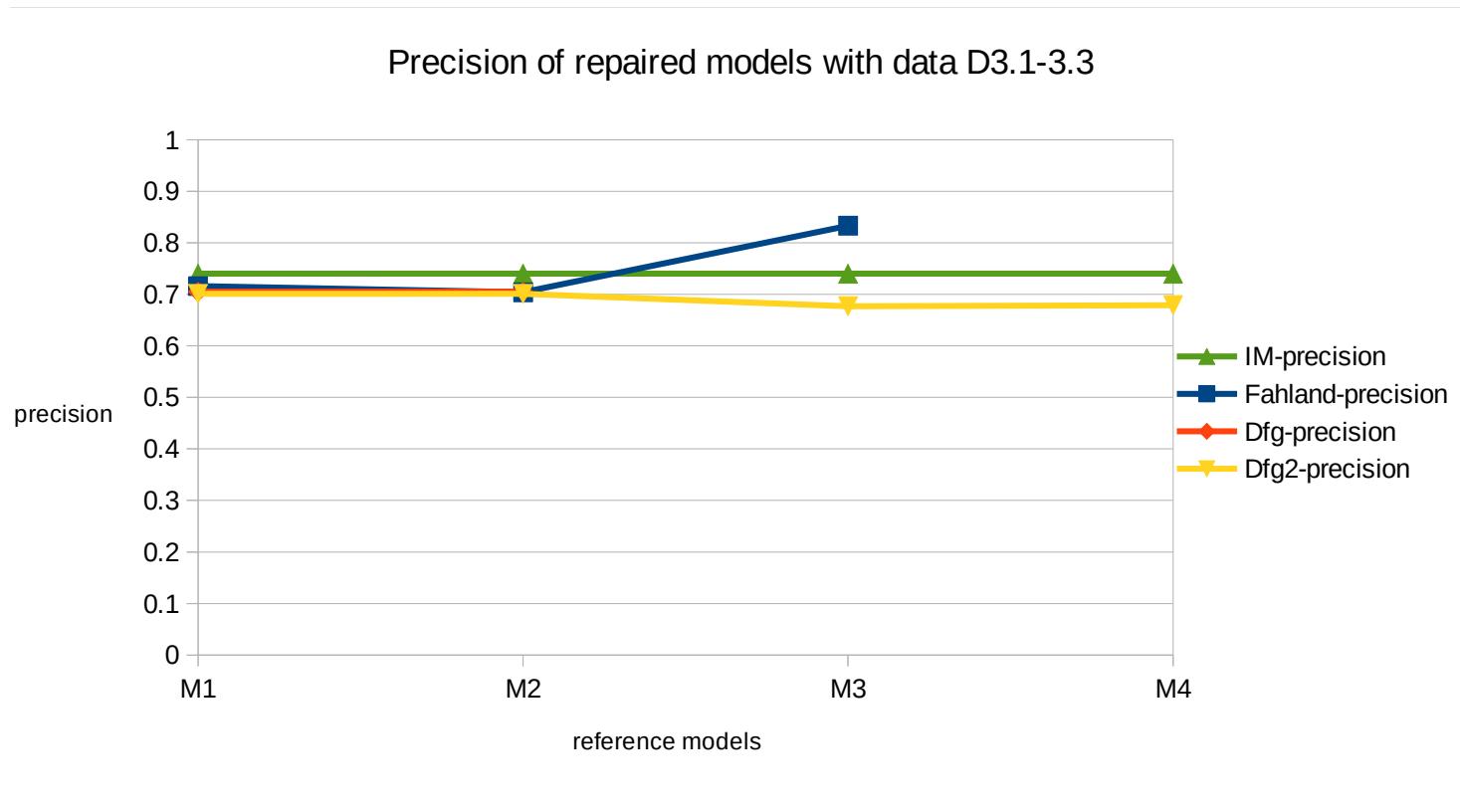
Experiment result



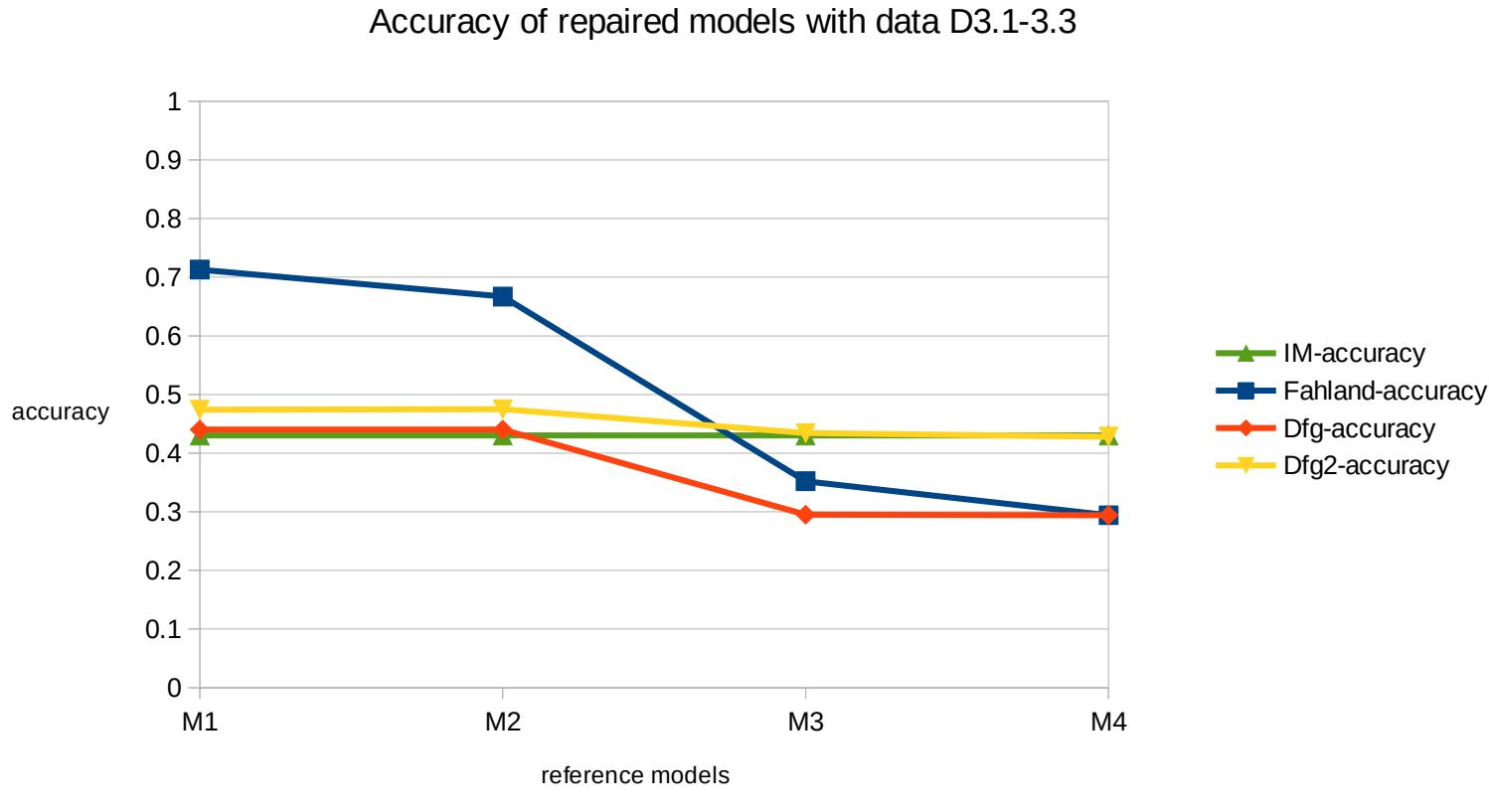
Experiment result



Experiment result



Experiment result

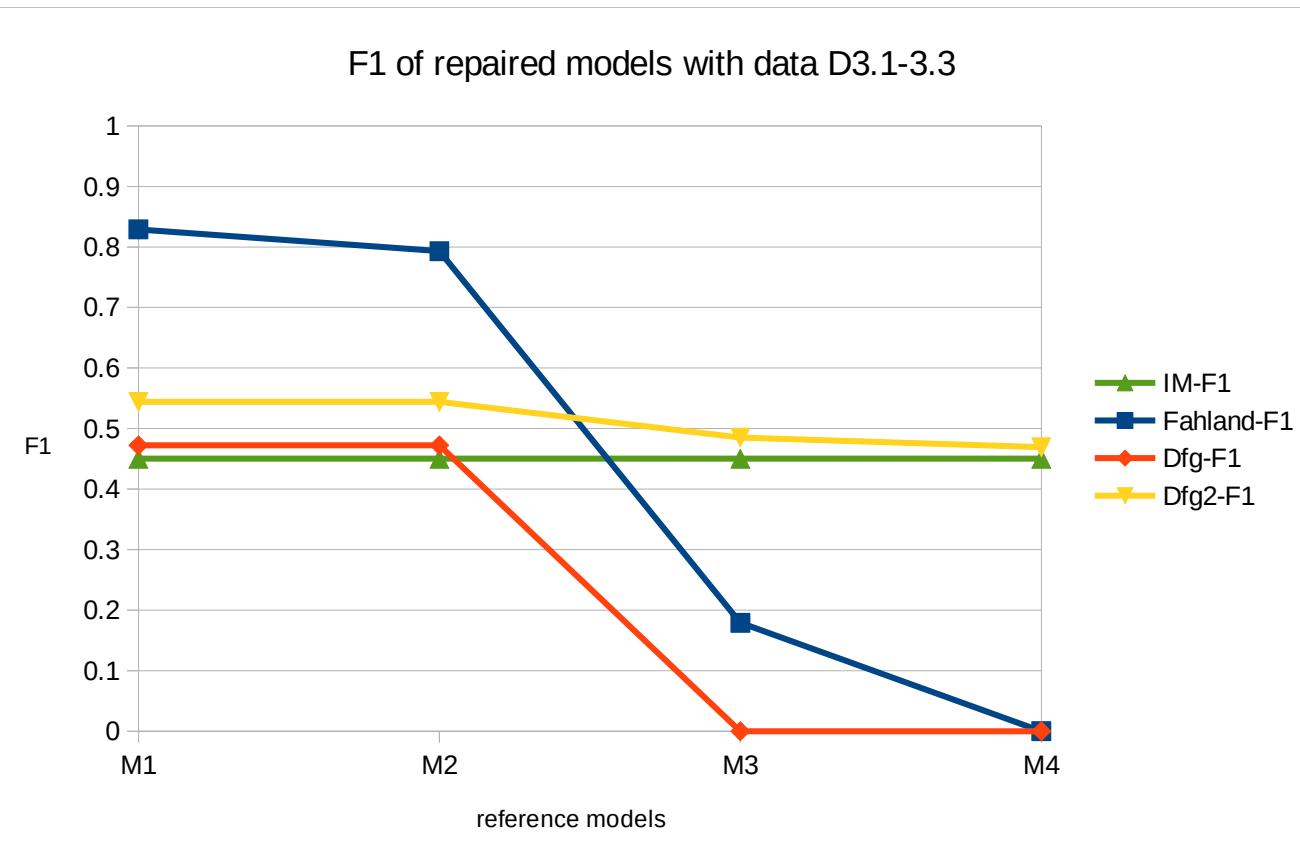


IM: Inductive Miner

Dfg-repair:
1-1-1

Dfg2-repair:
0.5-1-0.5

Experiment result

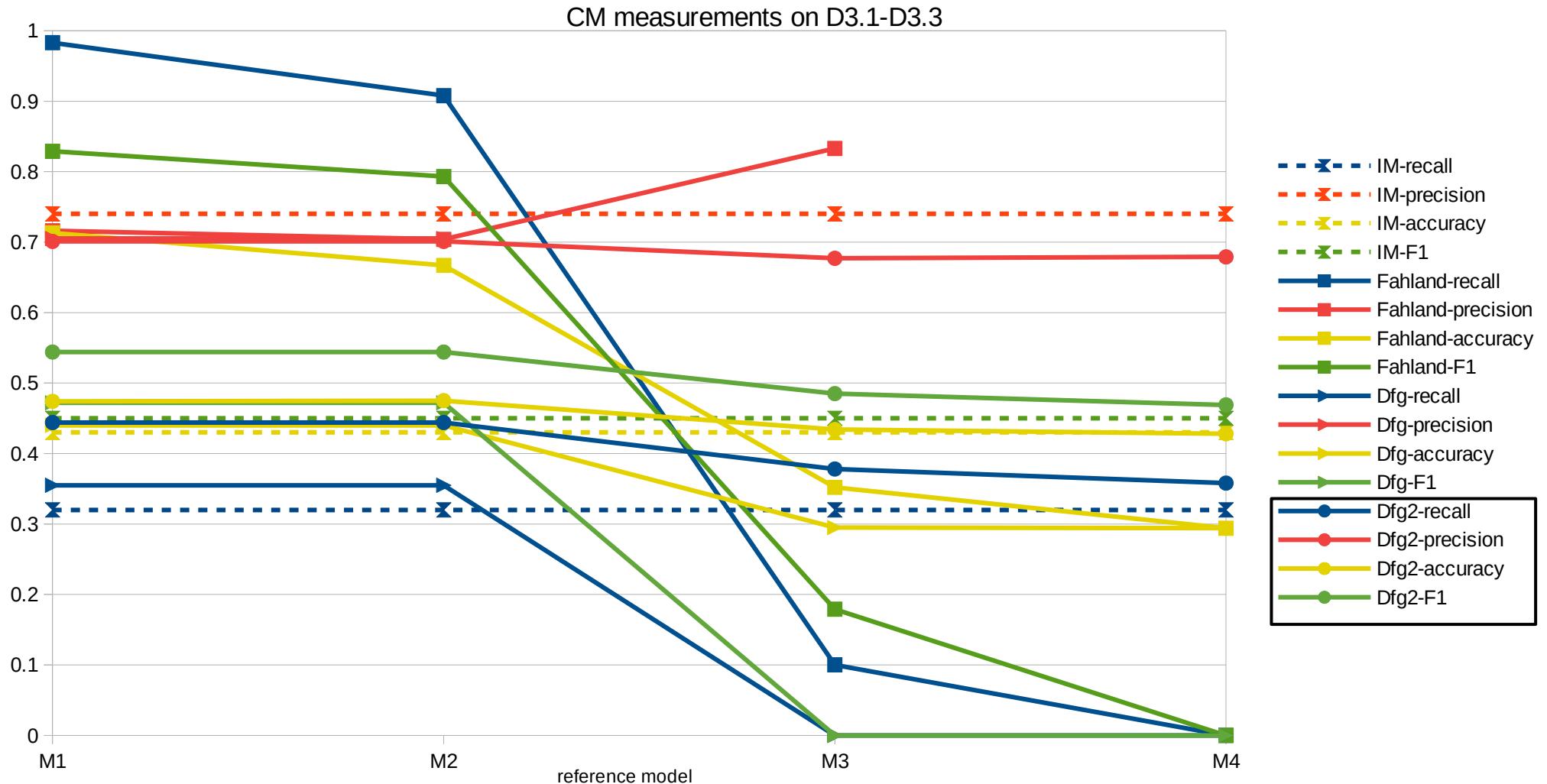


IM: Inductive Miner

Dfg-repair:
1-1-1

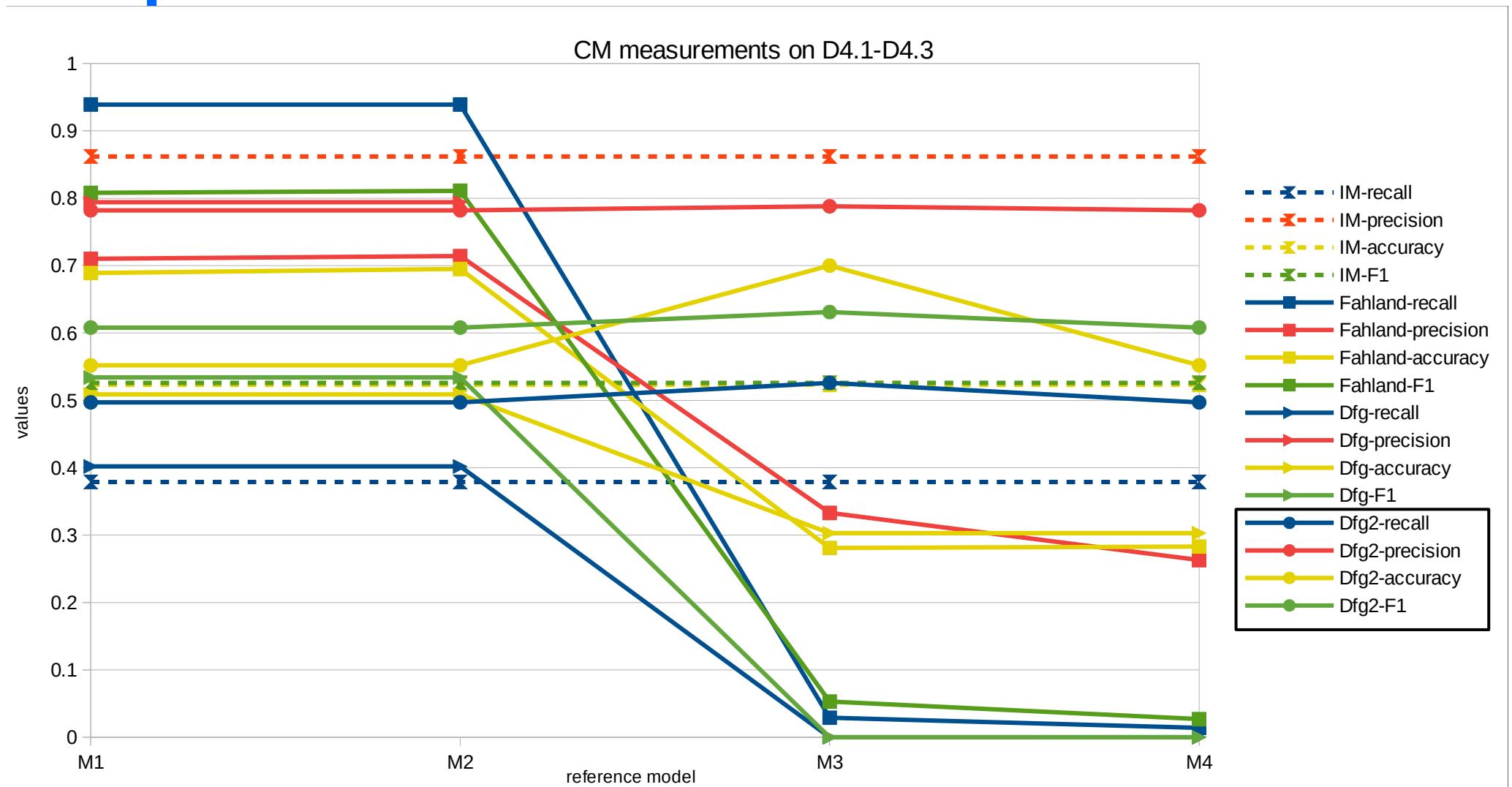
Dfg2-repair:
0.5-1-0.5

Experiment result



d_{fg2}-repair with setting 0.5-1-0.5, rank higher in precision, accuracy and F1 than other techniques; Different setting leads to different results

Experiment result



dfg2-repair with setting 0.5-1-0.5, rank higher in precision, accuracy and F1 than other techniques; Different setting leads to different results

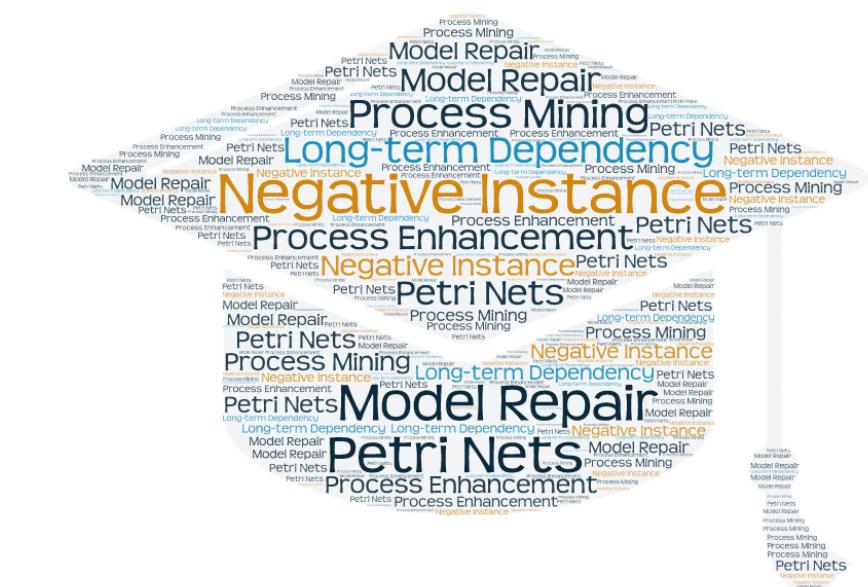
Outline

- Motivation for Research
- Problem Definition
- Approach
- Demo
- Evaluation
- Conclusion



Conclusion

- ✓ Innovative to incorporate negative instances
 - ✓ Innovative to add long-term dependence
 - ✓ In demo, conquer the shortcomings
 - repair model with better precision, accuracy, F1
 - ✓ in practice, feasible to use
 - ✓ In observation, repaired model simpler
 - run fast



Conclusion – Future work

- ★ Improve the method to incorporate different data models
- ★ Improve rules to decide long-term dependency
- ★ Drop process tree as intermediate result
- ★ Extend to other data models



Questions & Answers



Master Thesis

Model Repair by Incorporating Negative Instances
in Process Enhancement

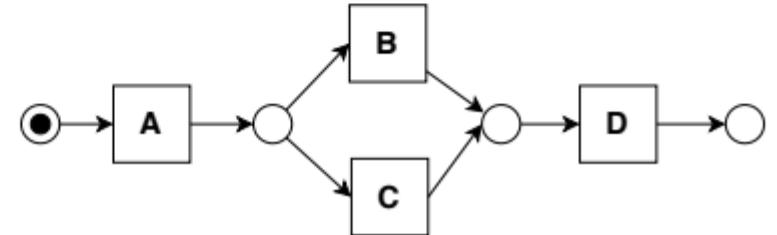
Kefang Ding



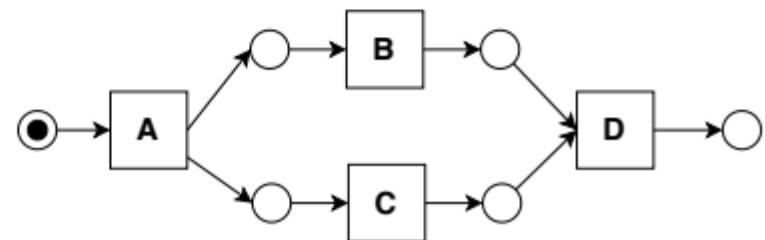
Thank you!!!

Why transition system

- Simple extraction can't get directly-follows graph

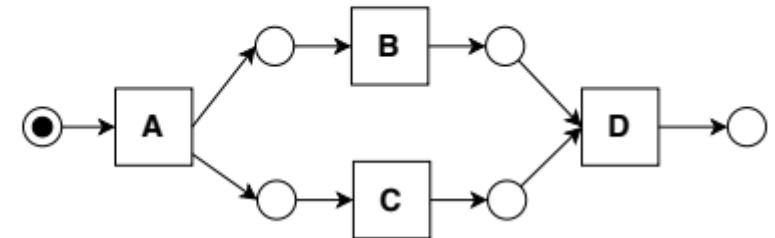
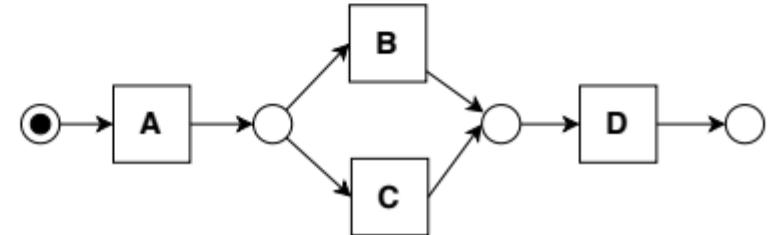


- Tool exists to transform a Petri net into Transition systems

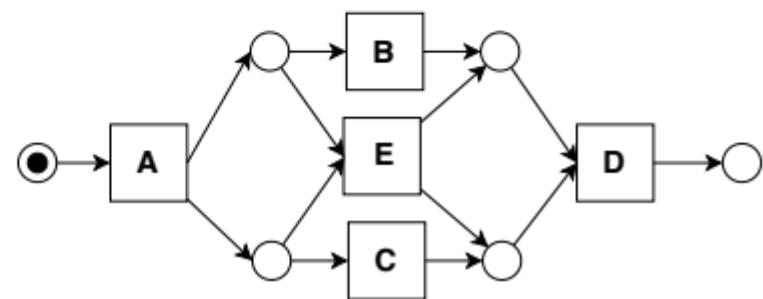


Why transition system

- Simple extraction can't get directly-follows graph



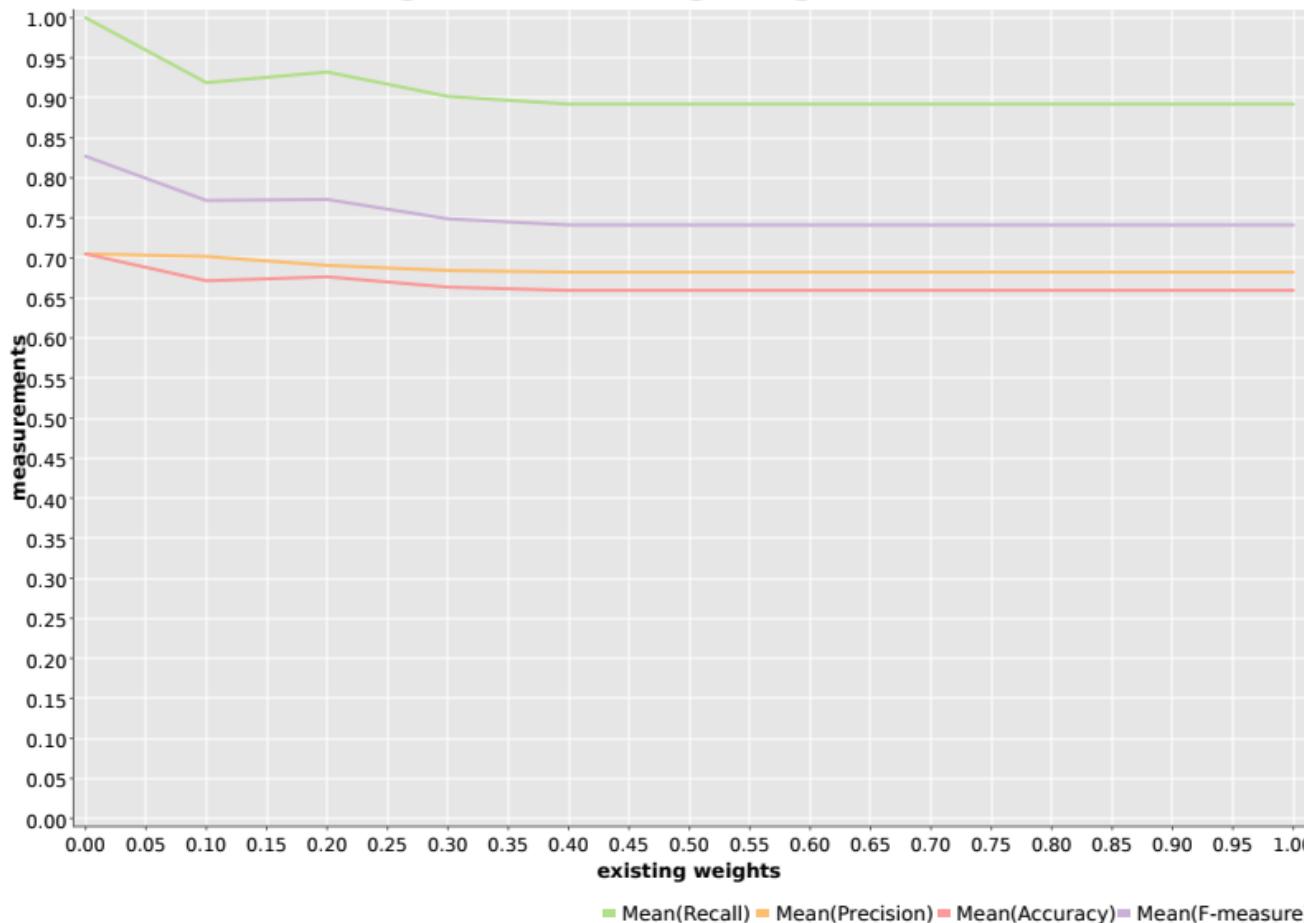
- Tool exists to transform a Petri net into Transition systems



Experiment result

- Weight for the reference model

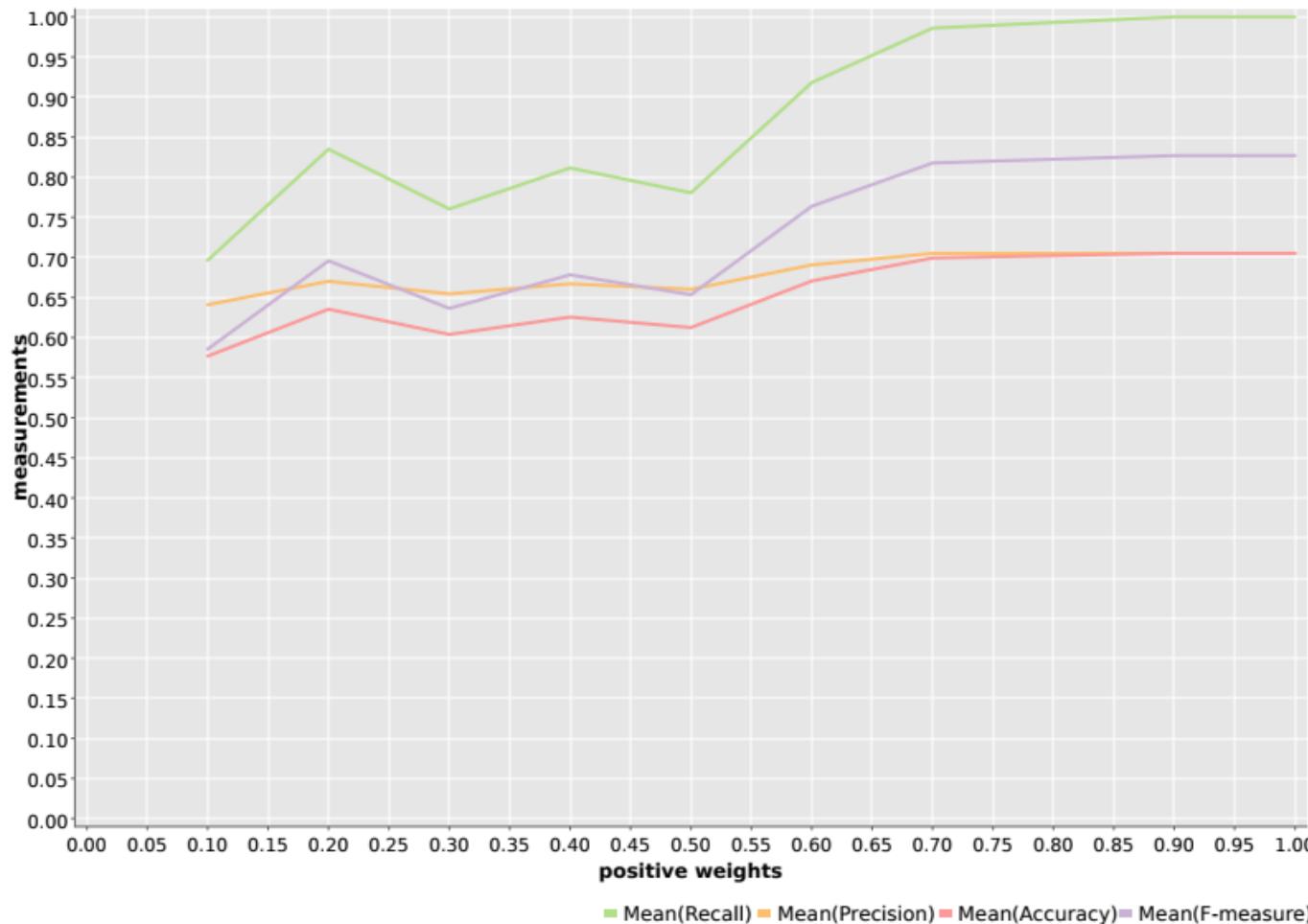
Measurements change with existing weight



Experiment result

- **Weight for positive instance**

Measurements change with positive weight



Experiment result

- Weight for negative instance

Measurements change with negative weight

