

Process Mining

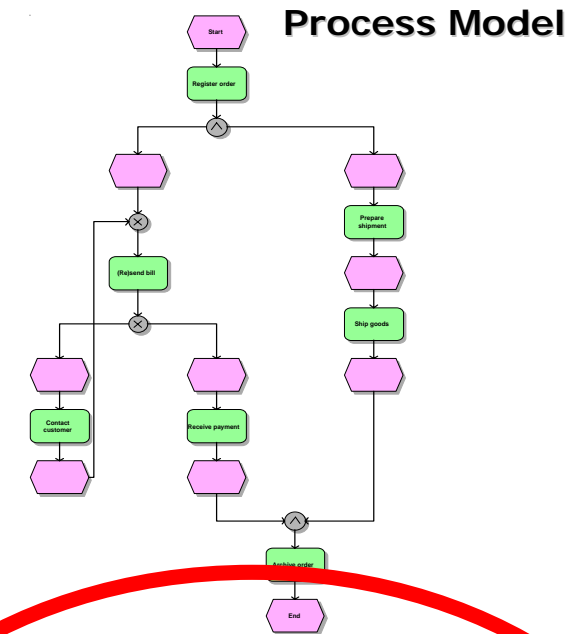
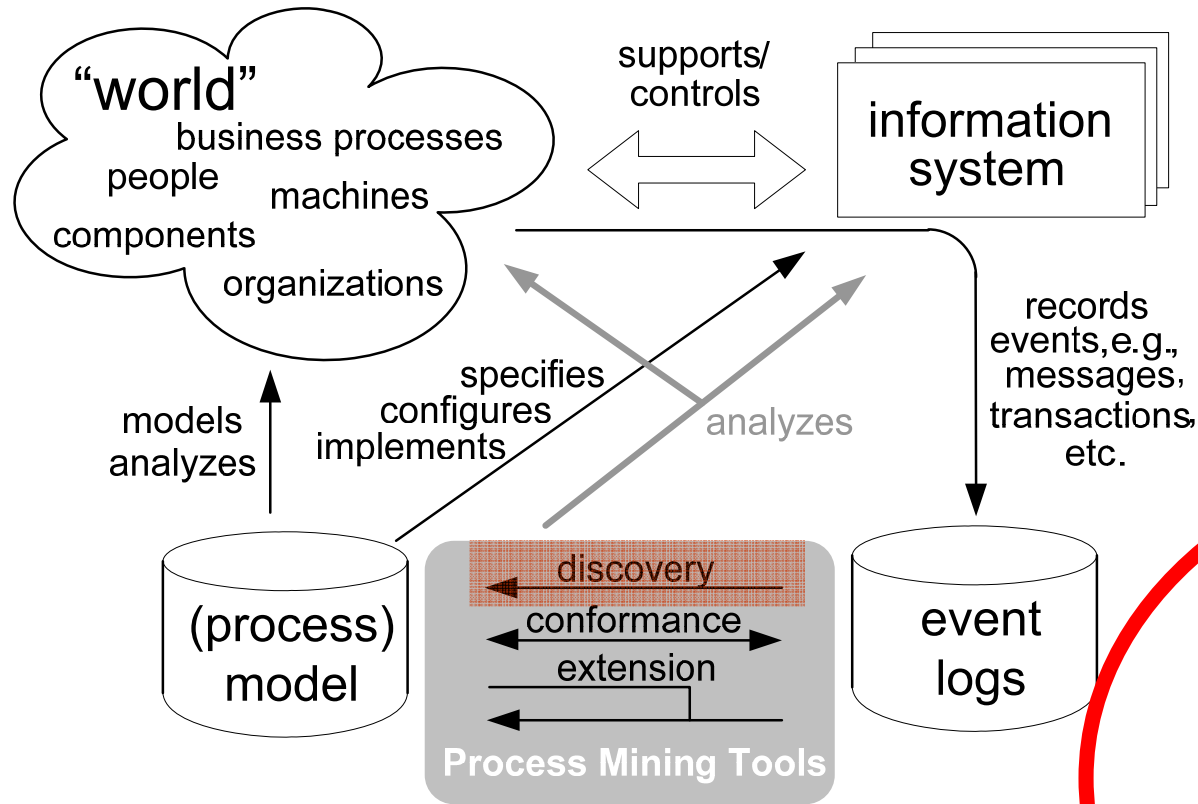
Part III – Beyond control-flow mining

Organizational mining
Discovery of social nets
Extension algorithms

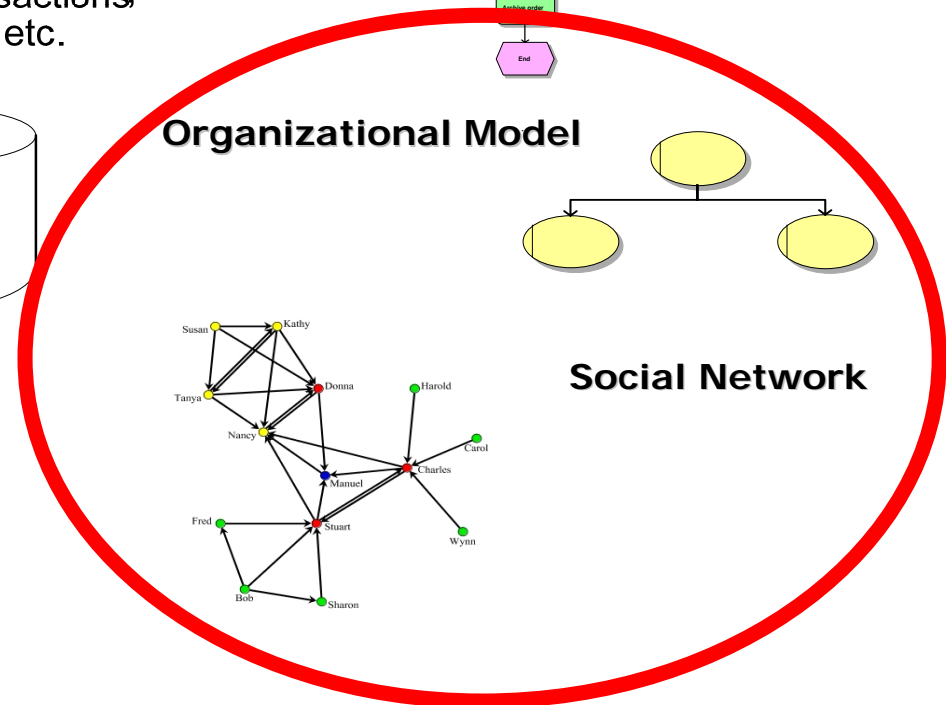


Outline

- **Part I – Introduction to Process Mining**
 - Context, motivation and goal
 - General characteristics of the analyzed processes and logs
 - Classification of Process Mining approaches
- **Part II – Workflow discovery**
 - Induction of basic Control Flow graphs
 - Other techniques (α -algorithm, Heuristic Miner, Fuzzy mining)
- **Part III – Beyond control-flow mining**
 - Organizational mining
 - Social net discovery
 - Extension algorithms
- **Part IV – Evaluation and validation of discovered models**
 - Conformance Check
 - Log-based property verification
- **Part V – Clustering-based Process Mining**
 - Discovery of hierarchical process models
 - Discovery of process taxonomies
 - Outlier detection



Organizational Model



Organizational mining techniques

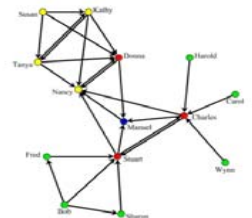
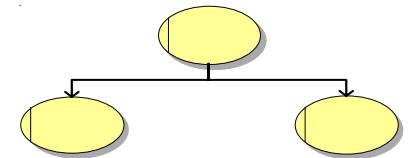
Organizational Mining Algorithms

■ Objective:

- ❑ Discover the organizational model (i.e., roles, departments, etc.) without prior knowledge about the structure of the organization
- ❑ Aid in understanding and improving social and organizational structures

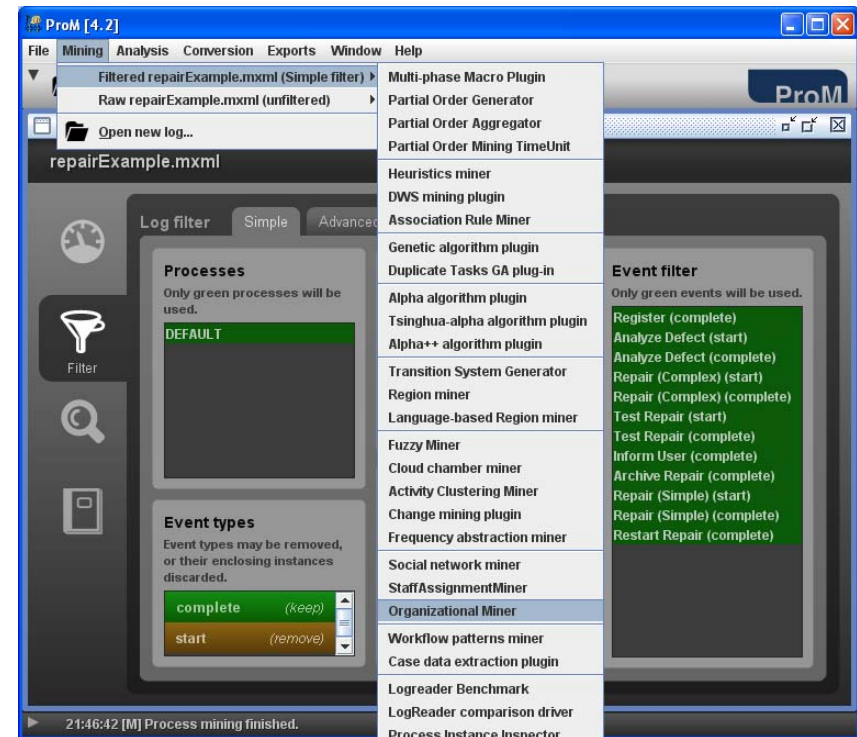
■ Two types of algorithms

- ❑ Organizational Model
 - Mining of roles and teams in organizations
 - ProM Plug-in: Organizational Miner
- ❑ Social Networks
 - Discovery of relationships among originators
 - ProM Plug-ins: *Social Network Miner* and *Analyze Social Network*



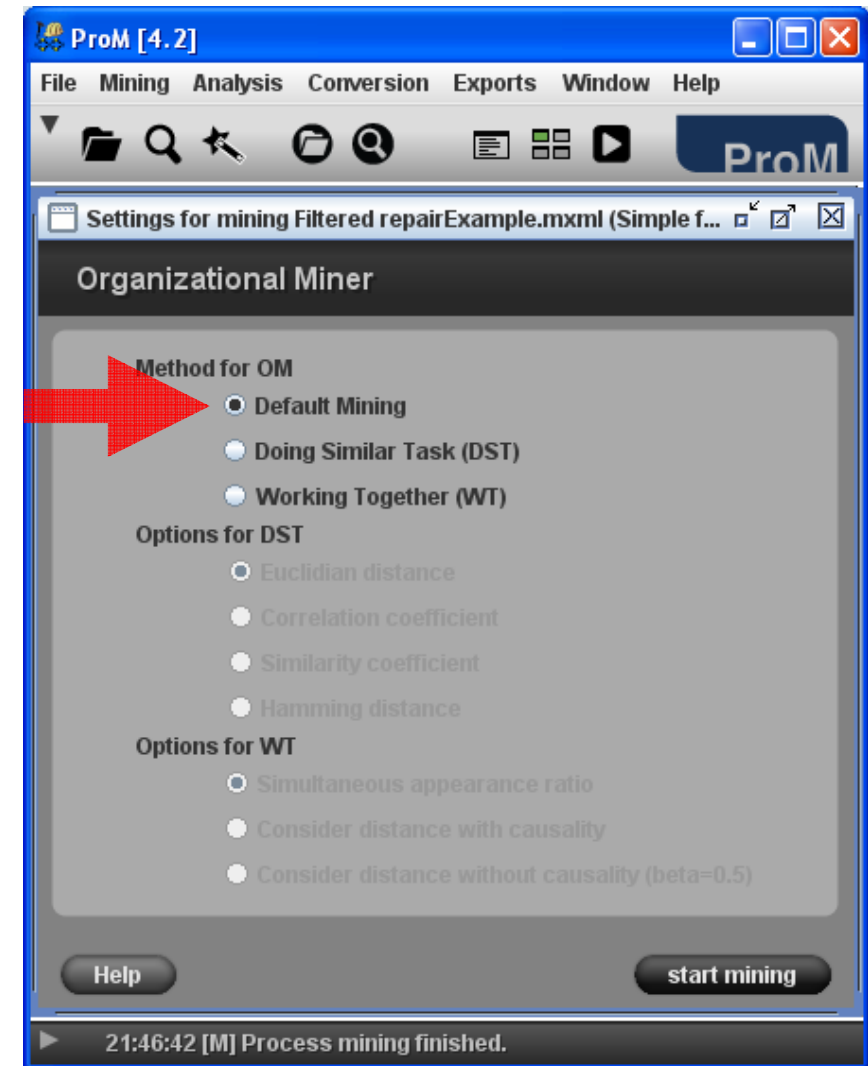
Organizational Miner

- Main idea: Which originators are executing which tasks
- Methods to mine *roles*
 - Default mining
 - Doing Similar Tasks
- Methods to mine *teams*
 - Working together



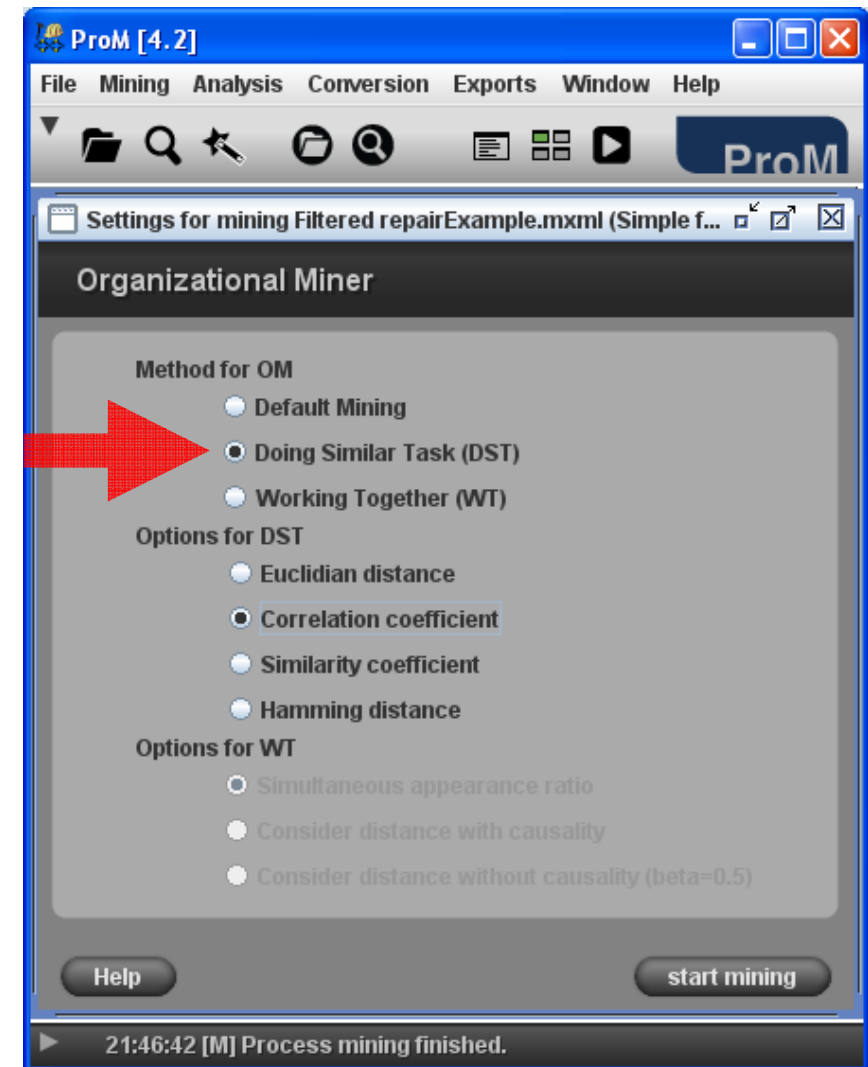
Organizational Miner

- Main idea: Which performers are executing which tasks
- Methods to mine *roles*
 - **Default mining**
 - Doing Similar Tasks
- Methods to mine *teams*
 - Working together



Organizational Miner

- Main idea: Which performers are executing which tasks
- Methods to mine *roles*
 - Default mining
 - **Doing Similar Tasks**
- Methods to mine *teams*
 - Working together



ProM [4.2]

File Mining Analysis Conversion Exports Window Help

Results - Organizational Miner on Filtered repairExample.mxml (Simple filter)

Default Mining

Organizational Model

This diagram shows the default mining result. It features a central 'System' resource node (pink oval) at the top, which branches into six task nodes (teal pentagons): 'Test Repair_complete', 'Analyze Defect_complete', 'Inform User_complete', 'Archive Repair_complete', 'Register_complete', and 'Restart Repair_complete'. To the left, six 'Tester' resource nodes (Tester1 to Tester6, pink ovals) are connected to the first two task nodes. To the right, three 'Solver' resource nodes (SolverS1 to SolverS3, pink ovals) are connected to 'Repair (Simple)_complete', and three more (SolverC1 to SolverC3, pink ovals) are connected to 'Repair (Complex)_complete'. Each task node has a corresponding task node below it (yellow rectangle) with the same name but without the '_complete' suffix. Arrows indicate the flow from resources to tasks and from tasks to their respective outputs.

Results - Organizational Miner on Filtered repairExample.mxml (Simple filter)

Doing Similar Tasks

Mining Result: adjust threshold value

Organizational Model

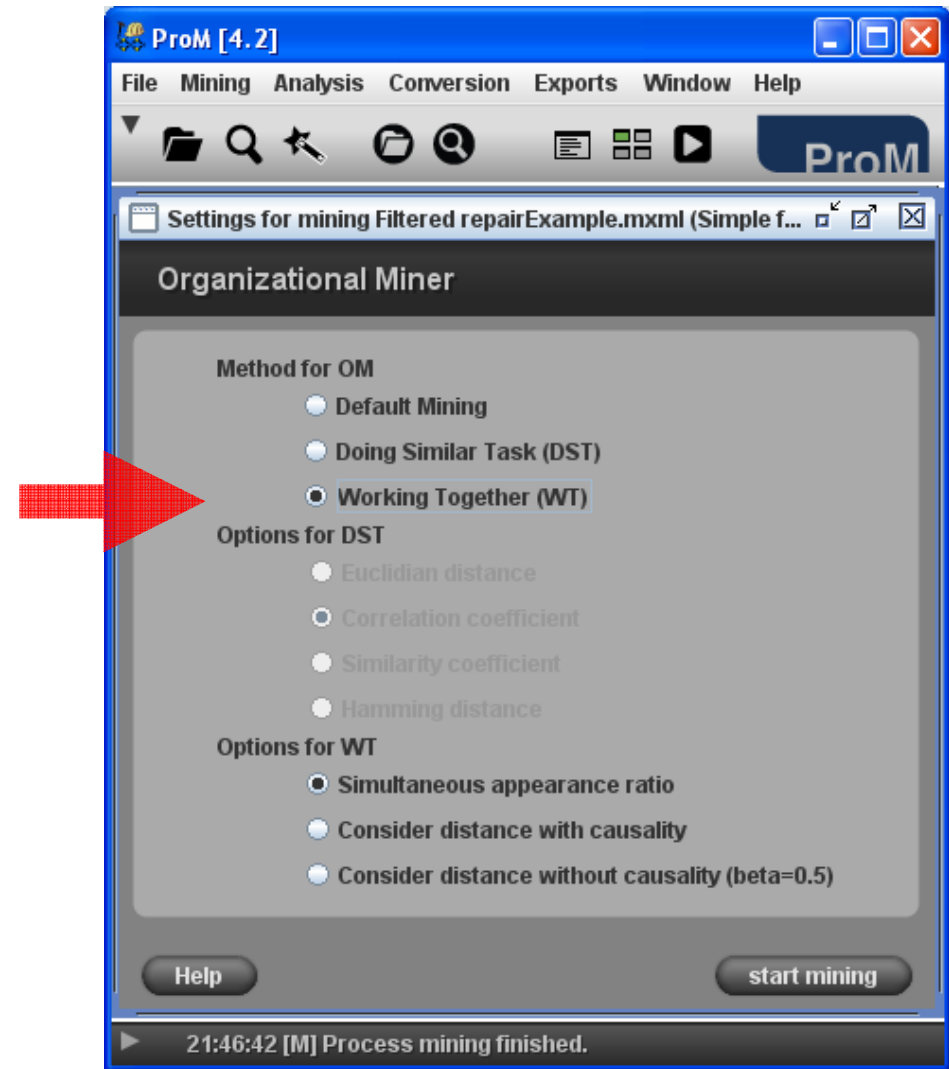
This diagram shows the result of mining with an adjusted threshold. It features four 'minedGroup' nodes (teal pentagons) instead of the 'System' node. 'minedGroup1' is connected to three 'Solver' nodes (SolverS1 to SolverS3) and leads to 'Repair (Simple)_complete'. 'minedGroup0' is connected to three 'Solver' nodes (SolverC1 to SolverC3) and leads to 'Repair (Complex)_complete'. 'minedGroup3' is connected to six 'Tester' nodes (Tester1 to Tester6) and leads to 'Test Repair_complete' and 'Analyze Defect_complete'. 'minedGroup2' is connected to the 'System' node and leads to 'Inform User_complete', 'Restart Repair_complete', 'Archive Repair_complete', and 'Register_complete'. Each group node has a corresponding task node below it (yellow rectangle) with the same name but without the '_complete' suffix. Arrows indicate the flow from resources to tasks and from tasks to their respective outputs.

☒ Show Resource nodes ☒ Show OrgEntity nodes ☒ Show Task nodes Redraw Graph

22:19:01 [D] DOT finished on: C:\DOCUME~1\AMEDEI~1\AKL\LOCALS~1\Temp\pmt60973.dot

Organizational Miner

- Main idea: Which performers are executing which tasks
- Methods to mine *roles*
 - Default mining
 - Doing Similar Tasks
- Methods to mine *teams*
 - **Working together**



Organizational Miner

Why is the notion of process instances necessary to mine teams but unnecessary to mine roles?

<WorkflowLog>

<Process>

<ProcessInstance>

<AuditTrailEntry/>

<AuditTrailEntry/>

<AuditTrailEntry/>

</ProcessInstance>

<ProcessInstance/>

<ProcessInstance/>

</Process>

</WorkflowLog>

<AuditTrailEntry>

<WorkflowModelElement/> Task A **</Wf.M.E.>**

<EventType> complete **</EventType>**

<TimeStamp> 2005-10-26T12:37:33... **</TimeStamp>**

<Originator> John Doe **</Originator>**

<Data>

<Attribute name="x"> 1 **</Attribute>**

<Attribute name="y"> whatever **</Attribute>**

</Data>

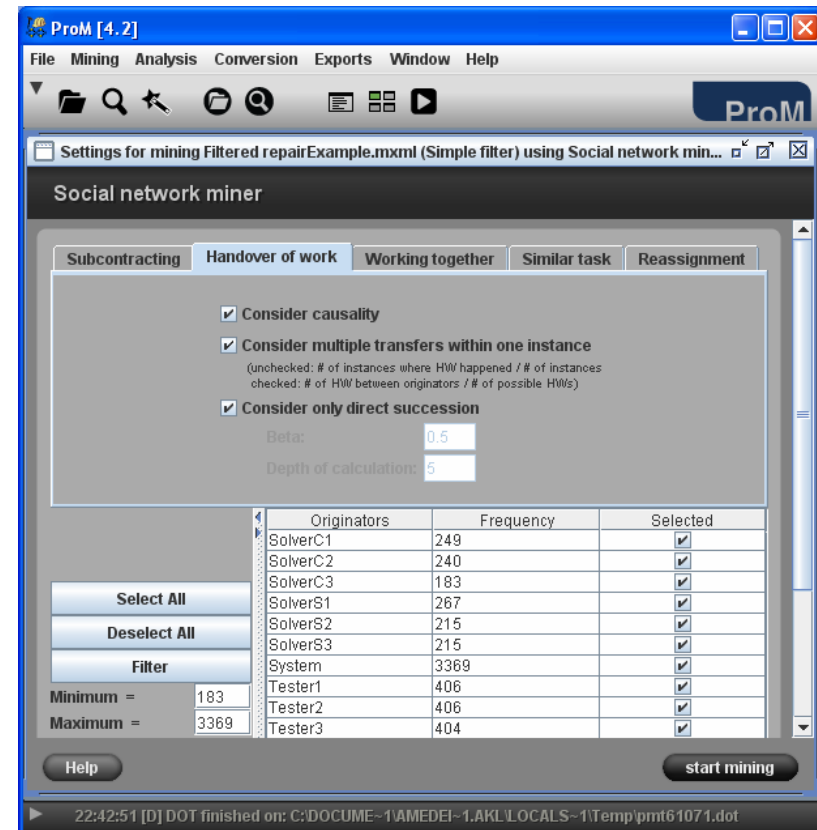
</AuditTrailEntry>

Outline

- **Part I – Introduction to Process Mining**
 - Context, motivation and goal
 - General characteristics of the analyzed processes and logs
 - Classification of Process Mining approaches
- **Part II – Workflow discovery**
 - Induction of basic Control Flow graphs
 - Other techniques (α -algorithm, Heuristic Miner, Fuzzy mining)
- **Part III – Beyond control-flow mining**
 - Organizational mining
 - **Social net discovery**
 - Extension algorithms
- **Part IV – Evaluation and validation of discovered models**
 - Conformance Check
 - Log-based property verification
- **Part V – Clustering-based Process Mining**
 - Discovery of hierarchical process models
 - Discovery of process taxonomies
 - Outlier detection

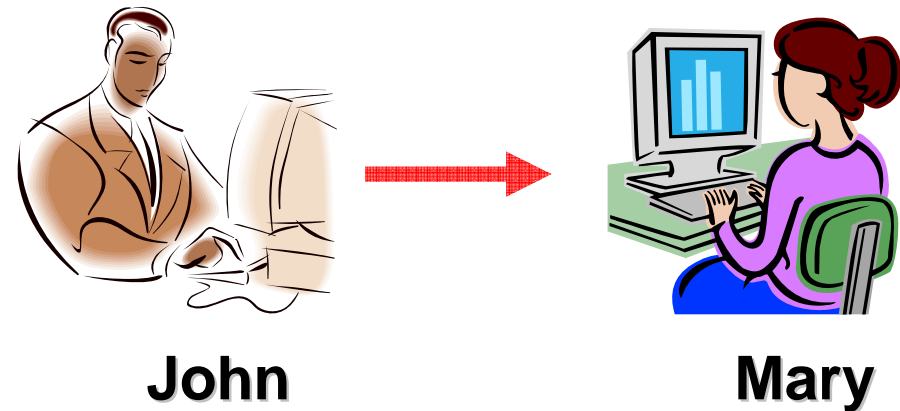
Social Network Miner

- Aim:
 - ❑ Monitor how individual process instances are routed between originators
- Metrics
 - ❑ Handover of work
 - ❑ Subcontracting
 - ❑ Reassignment
 - ❑ Working together
 - ❑ Similar task



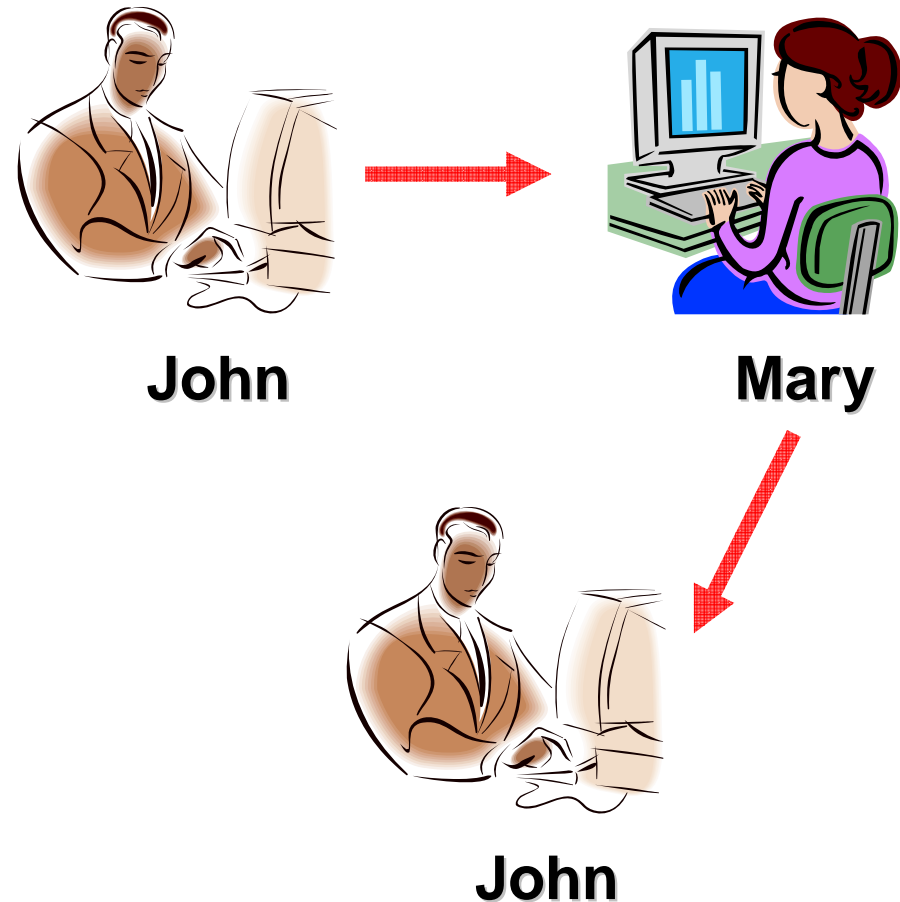
Social Network Miner

- Aim: Monitor how individual process instances are routed between originators
- Metrics
 - ❑ **Handover of work**
 - ❑ Subcontracting
 - ❑ Reassignment
 - ❑ Working together
 - ❑ Similar task



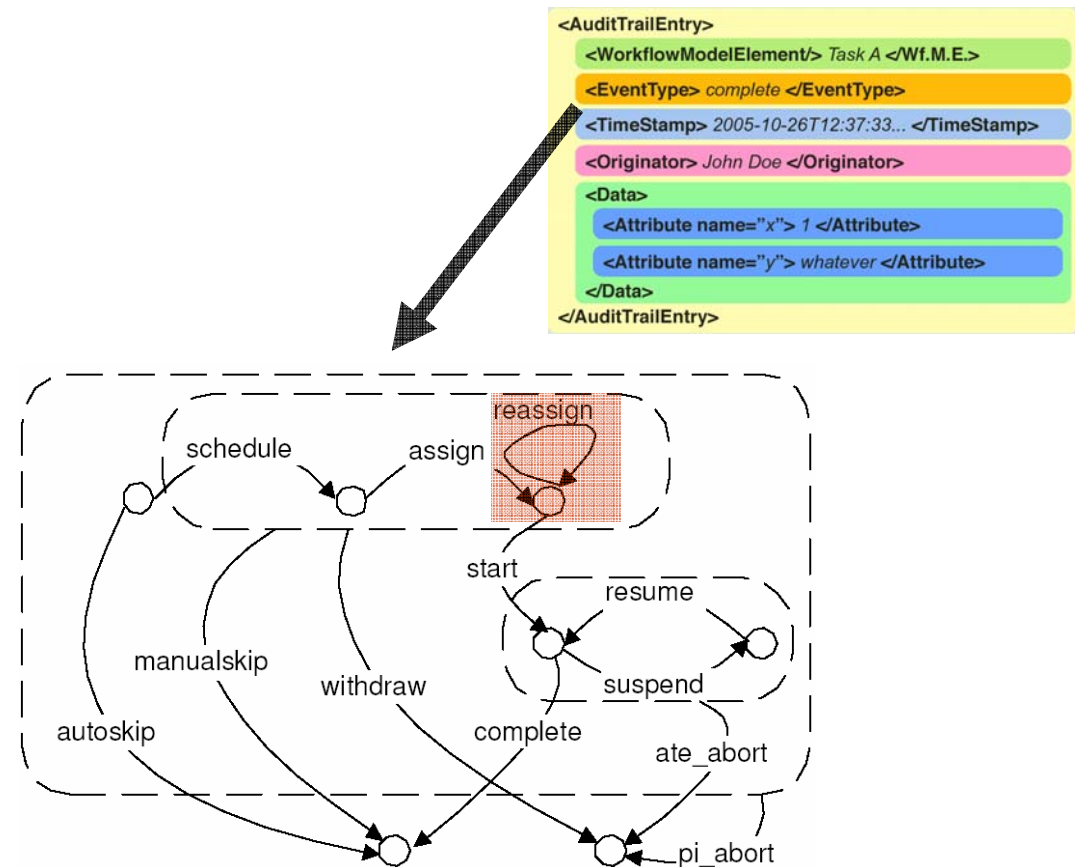
Social Network Miner

- Aim: Monitor how individual process instances are routed between originators
- Metrics
 - Handover of work
 - **Subcontracting**
 - Reassignment
 - Working together
 - Similar task



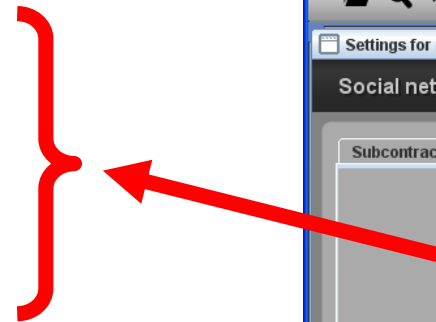
Social Network Miner

- Aim: Monitor how individual process instances are routed between originators
- Metrics
 - ❑ Handover of work
 - ❑ Subcontracting
 - ❑ **Reassignment**
 - ❑ Working together
 - ❑ Similar task

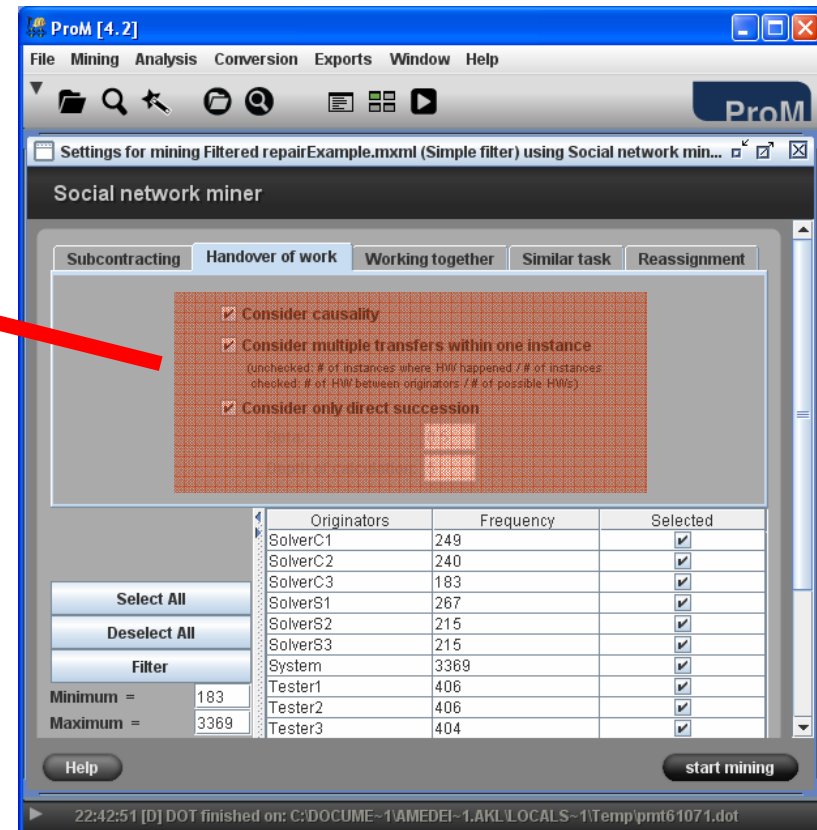


Social Network Miner

- Aim: Monitor how individual process instances are routed between originators
- Metrics
 - ❑ Handover of work
 - ❑ Subcontracting
 - ❑ Reassignment
 - ❑ Working together
 - ❑ Similar task



Based on ordering relations derived from a log!



ProM [4.2]

File Mining Analysis Conversion Exports Window Help

Settings for mining Filtered repairExample.mxml (Simple filter) using Social network min...

Social network miner

Subcontracting Handover of work Working together Similar task Reassignment

☒ Consider causality
☒ Consider multiple transfers within one instance
(unchecked: # of instances where HW happened / # of instances checked: # of HW between originators / # of possible HWs)
☒ Consider only direct succession

Originators	Frequency	Selected
SolverC1	249	<input checked="" type="checkbox"/>
SolverC2	240	<input checked="" type="checkbox"/>
SolverC3	183	<input checked="" type="checkbox"/>
SolverS1	267	<input checked="" type="checkbox"/>
SolverS2	215	<input checked="" type="checkbox"/>
SolverS3	215	<input checked="" type="checkbox"/>
System	3369	<input checked="" type="checkbox"/>
Tester1	406	<input checked="" type="checkbox"/>
Tester2	406	<input checked="" type="checkbox"/>
Tester3	404	<input checked="" type="checkbox"/>

Select All
 Deselect All
 Filter
 Minimum = 183
 Maximum = 3369

Help start mining

22:42:51 [D] DOT finished on: C:\DOCUME~1\AMEDEI~1\AKL\LOCALS~1\Temp\pmt61071.dot

Social Network Miner: Example

	John	Alex	Lucia	Peter	Mary
John	0	0	0	0	2
Alex	0	0	0	0	0
Lucia	0	0	0	2	2
Peter	0	0	2	0	2
Mary	2	0	2	2	0

Output Log #2

File Edit

Log File Number 2

CLIQUE

Minimum Set Size: 2
 Input dataset: C:\Program Files\...

WARNING: Valued graph. All values > 0 treated as 1.
 2 cliques found.

1: Lucia Peter Mary
 2: John Mary

Actor-by-Actor Clique Co-Membership Matrix

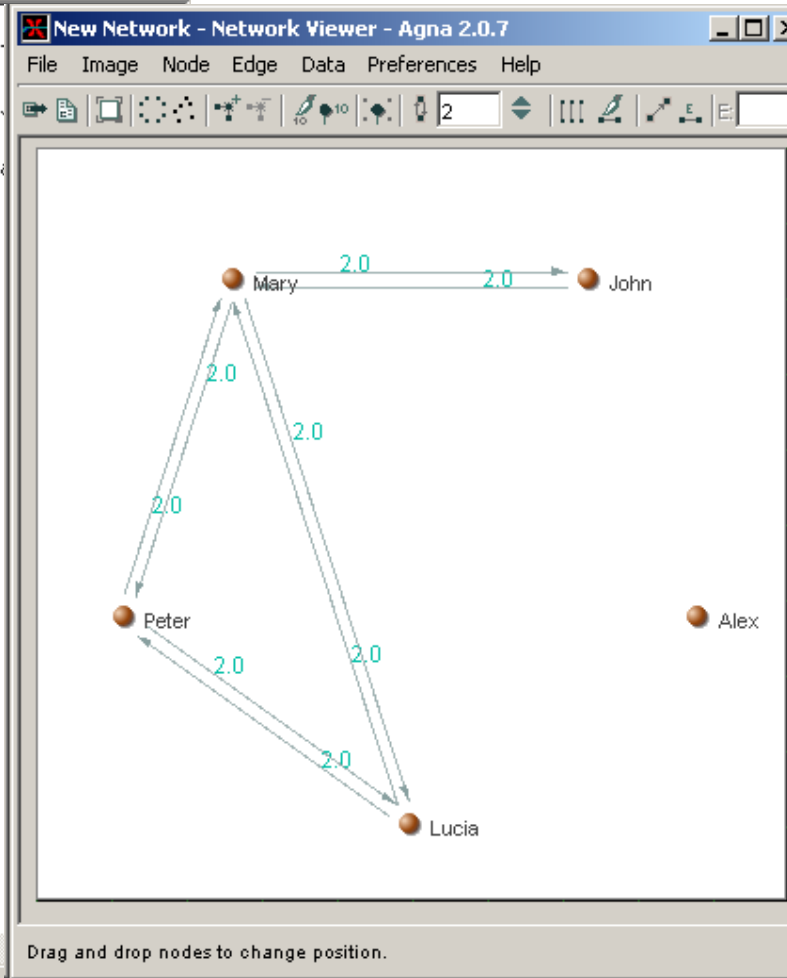
		1	2	3	4	5
	J A L P M					
1	John	1	0	0	0	1
2	Alex	0	0	0	0	0
3	Lucia	0	0	1	1	1
4	Peter	0	0	1	1	1
5	Mary	1	0	1	1	2

HIERARCHICAL CLUSTERING OF EQUIVALENCE MATRIX

	L	P
A	ue	J M
l	ct	oa
e	ie	hr
x	ar	ny

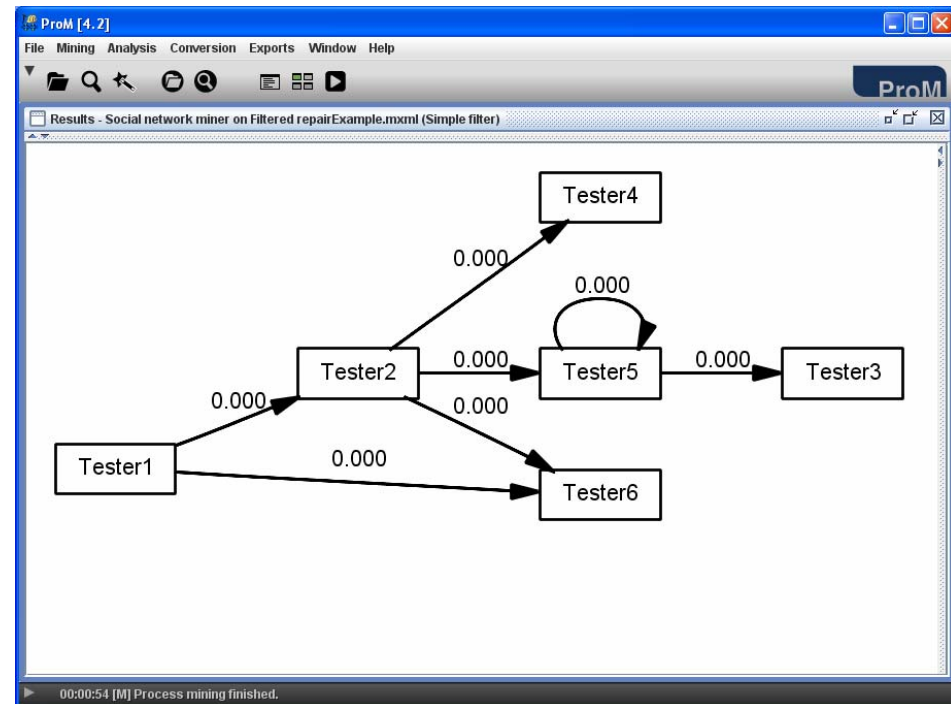
Level 2 3 4 1 5

1.000	.	XXX	XXX
0.667	.	XXXXXXX	
0.000		XXXXXXXXX	



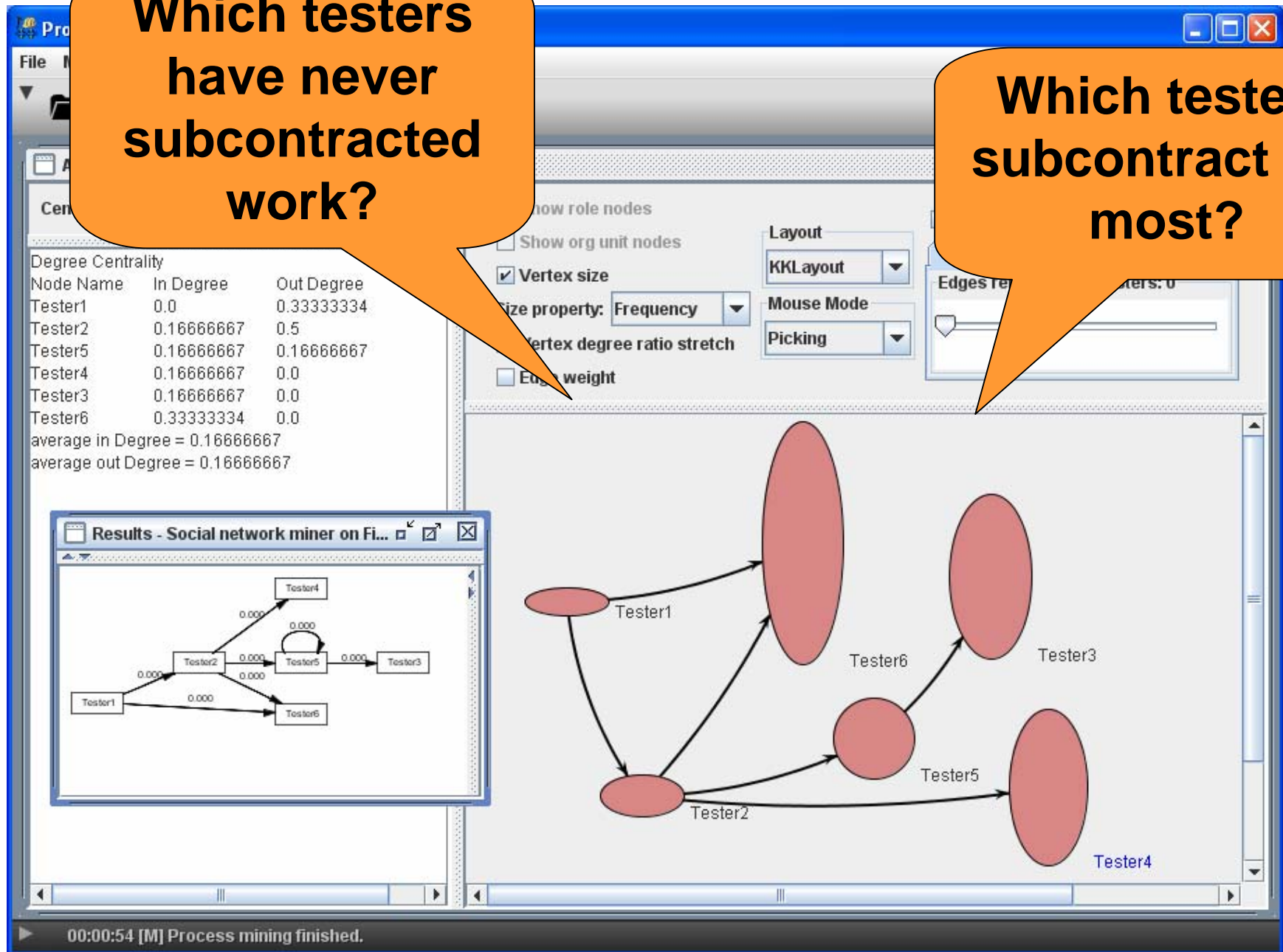
Plugin *Analyze Social Network*

- Better graphical view for the results of the Social Network Miner
- Includes different metrics to measure centrality of nodes
- Example: subcontracting



**Which testers
have never
subcontracted
work?**

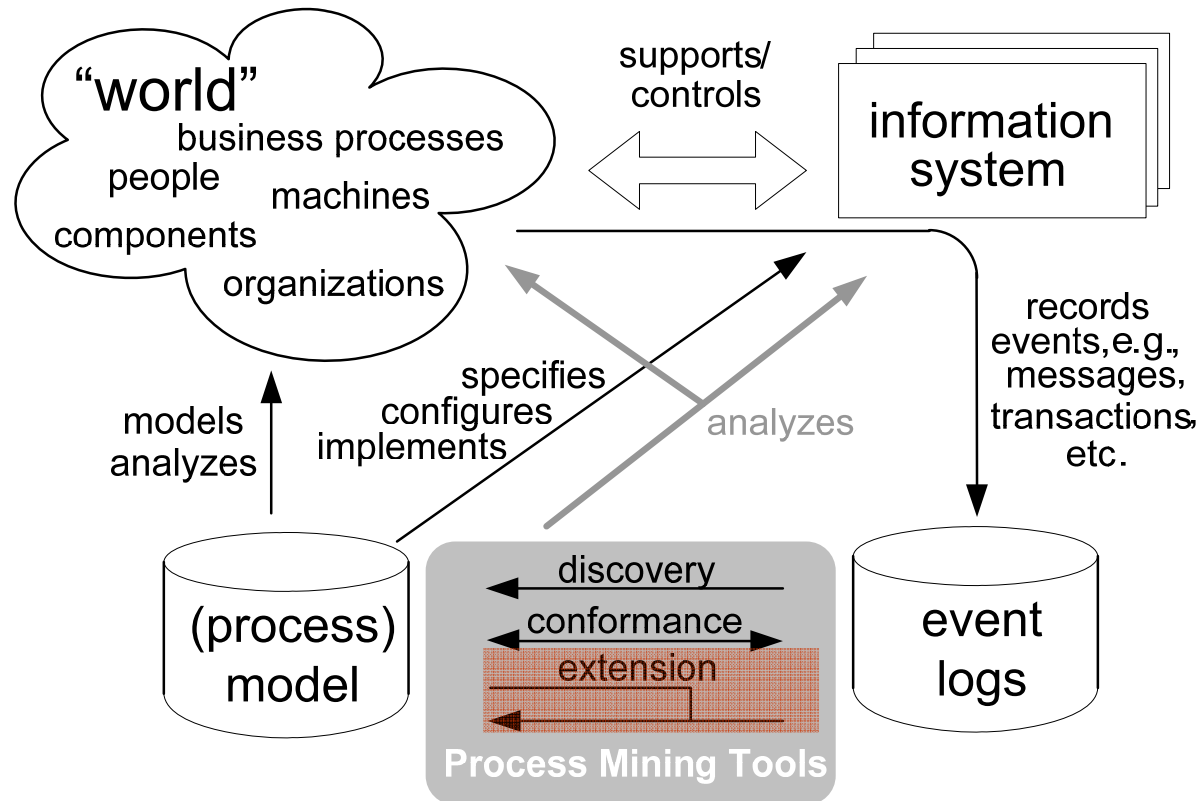
**Which testers
subcontract the
most?**



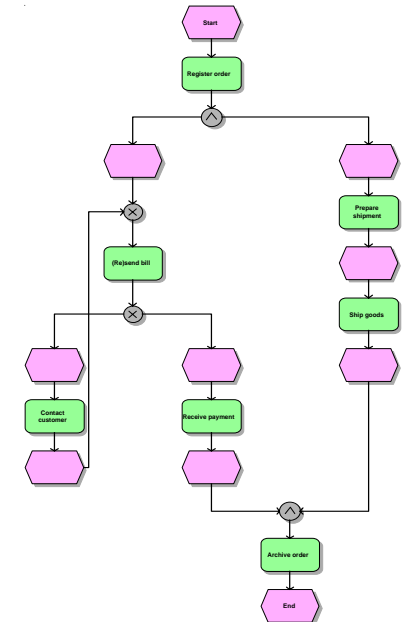
Outline

- Part I – Introduction to Process Mining
 - Context, motivation and goal
 - General characteristics of the analyzed processes and logs
 - Classification of Process Mining approaches
- Part II – Workflow discovery
 - Induction of basic Control Flow graphs
 - Other techniques (α -algorithm, Heuristic Miner, Fuzzy mining)
- **Part III – Beyond control-flow mining**
 - Organizational mining
 - Social net discovery
 - **Extension algorithms**
- **Part IV – Evaluation and validation of discovered models**
 - Conformance Check
 - Log-based property verification
- **Part V – Clustering-based Process Mining**
 - Discovery of hierarchical process models
 - Discovery of process taxonomies
 - Outlier detection in a process mining setting

Extension techniques



Bottlenecks/ Business Rules Process Model



Enhance existing models with information discovered from logs

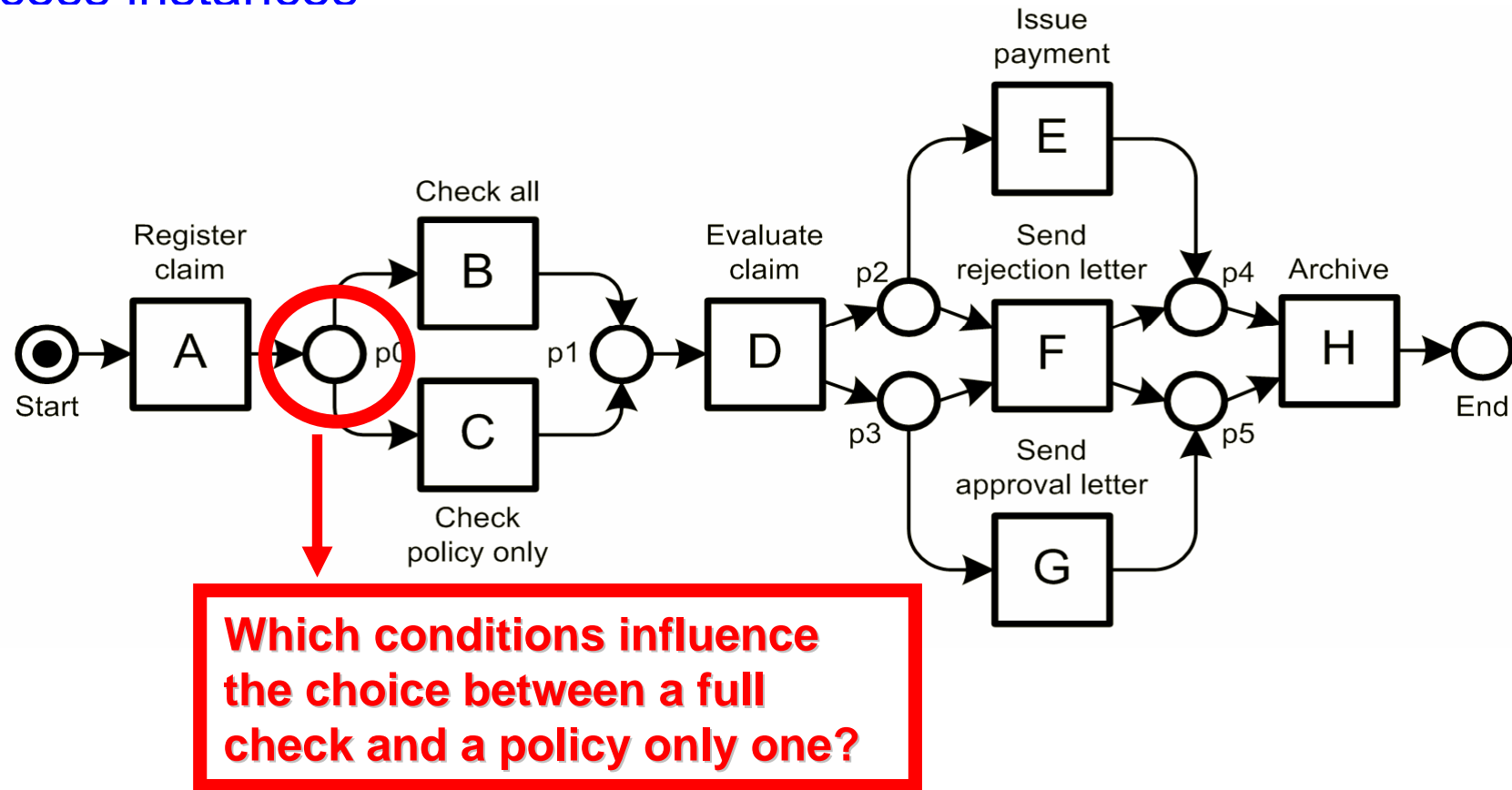
- ❑ The *Decision Point Analysis* plug-in can discover the “business rules” for the moments of choice in a process model
- ❑ The *Performance Analysis with Petri Nets* plug-in provides various KPIs w.r.t. the execution of processes

Performance Analysis



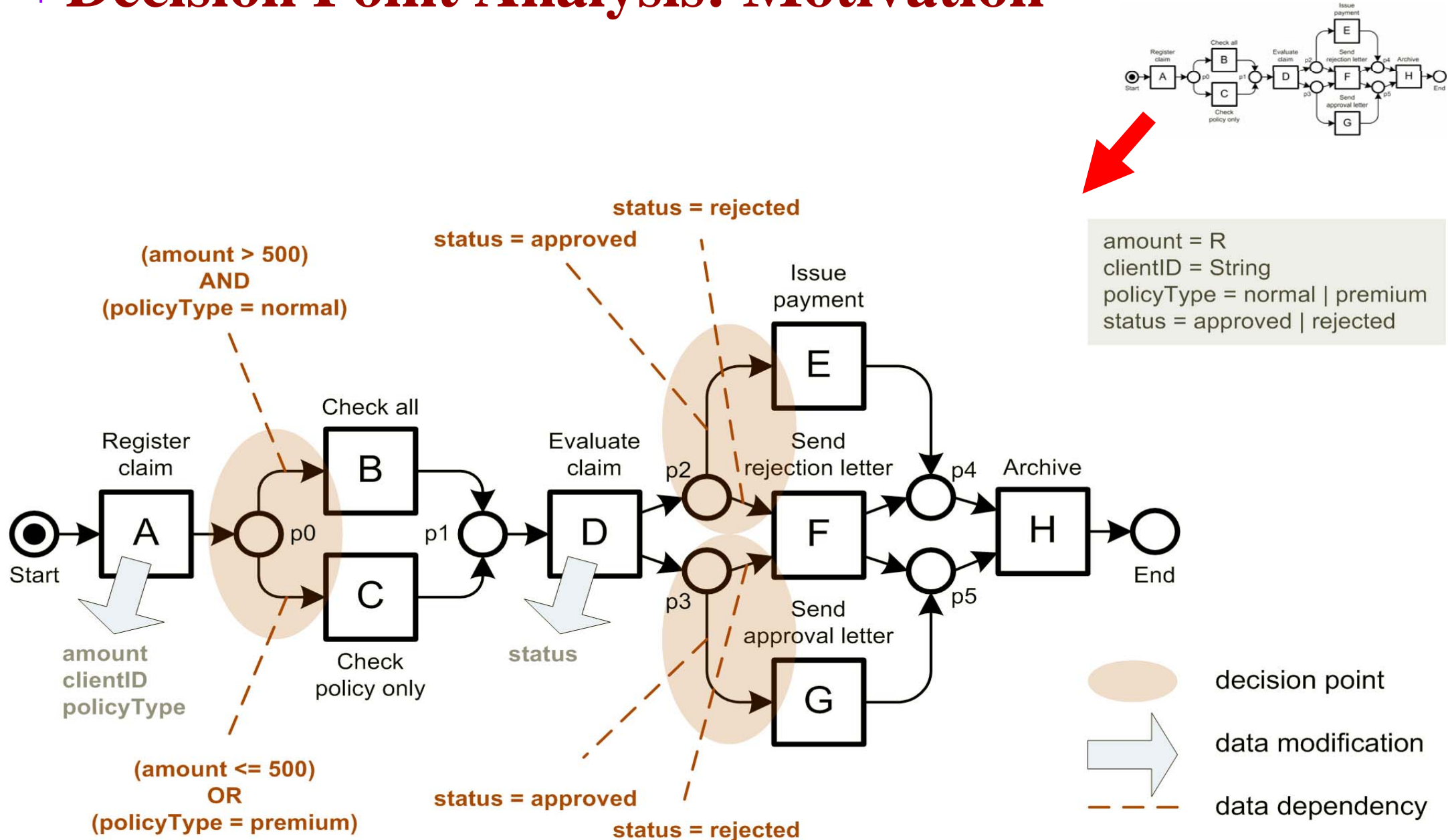
Decision Point Analysis: Main Idea

- Detection of data dependencies that affect the routing the routing of process instances

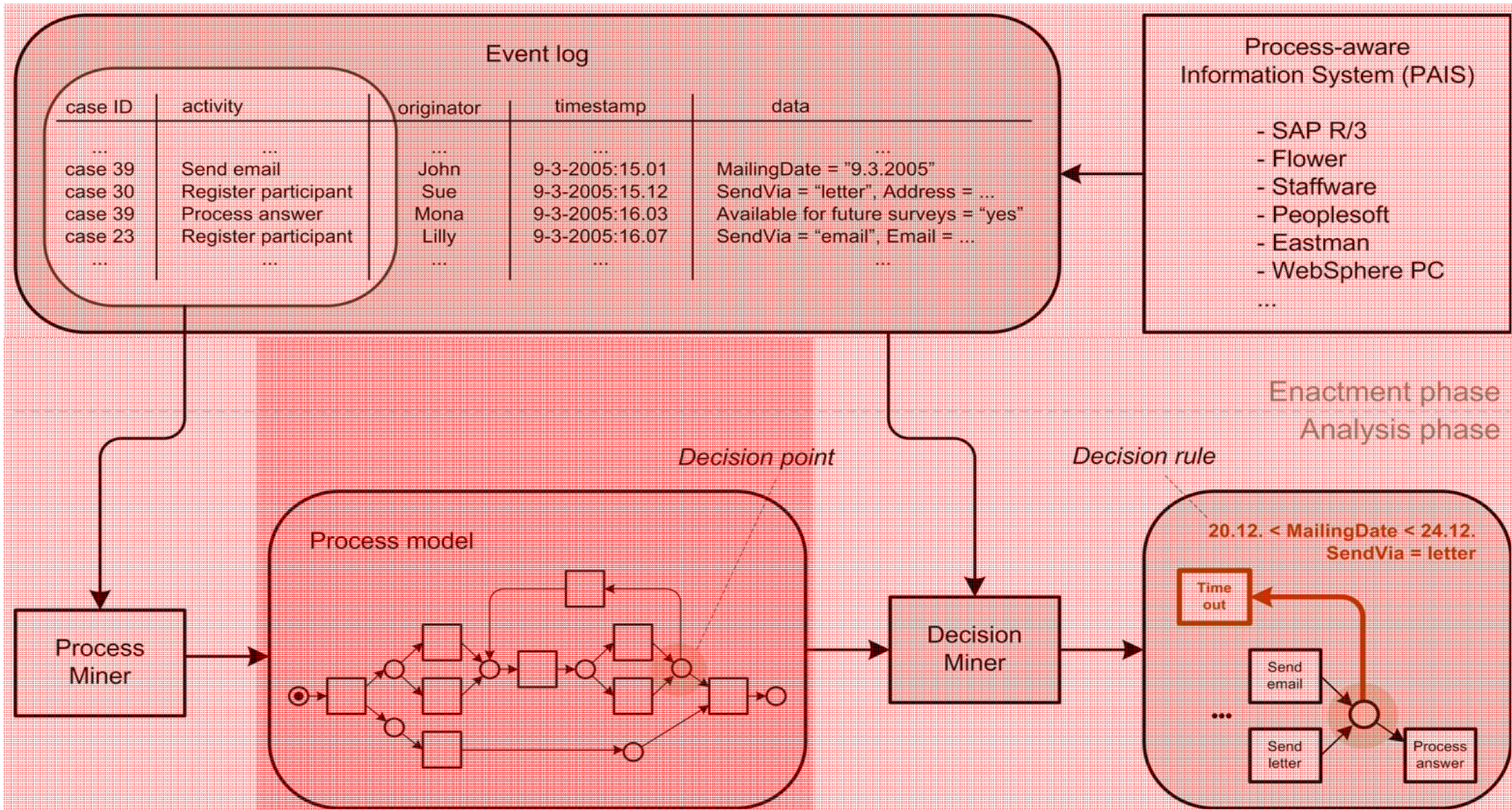


- Motivations
 - Make tacit knowledge explicit
 - Better understand the process model

Decision Point Analysis: Motivation



Decision Point Analysis: Approach



Decision Point Analysis

1. Read a log + model
2. Identify the decision points in a model
3. Find out which alternative branch has been taken for a given process instance and decision point
4. Discover the rules for each decision point
5. Return the enhanced model with the discovered rules

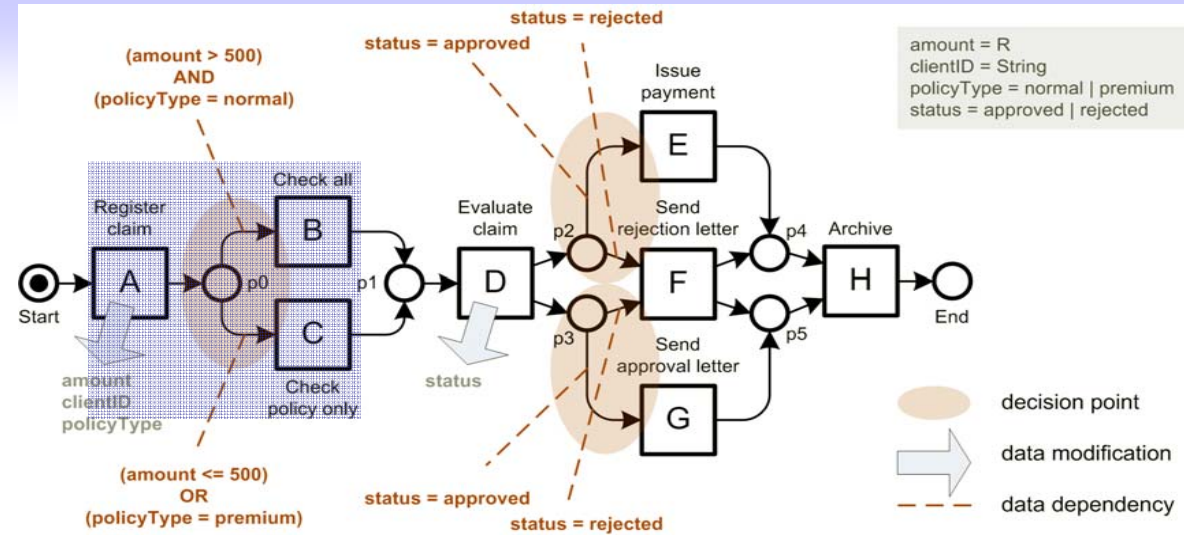
Decision Point Analysis

1. Read a log + model
2. Identify
3. Find out **Which elements are the classes and which are the attributes?** en taken for a given process
4. **Discover the rules for each decision point**
5. Return the enhanced model with the discovered rules

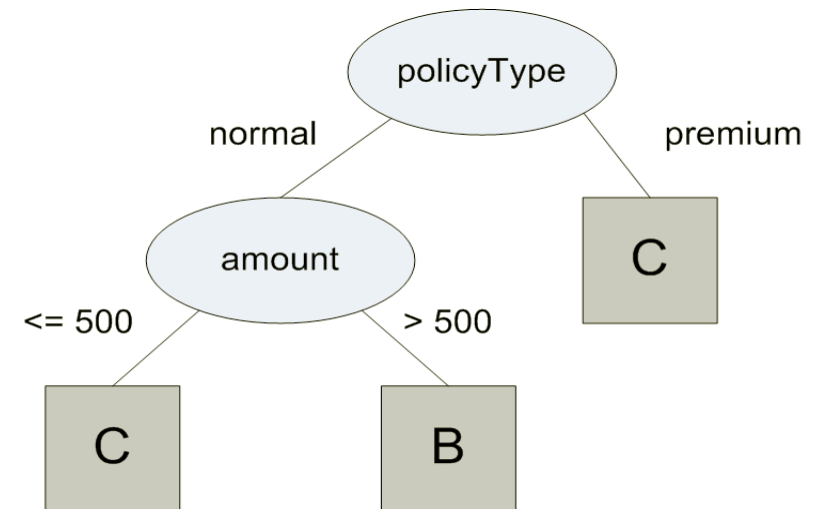
Step 4

Training examples for decision point "p0"

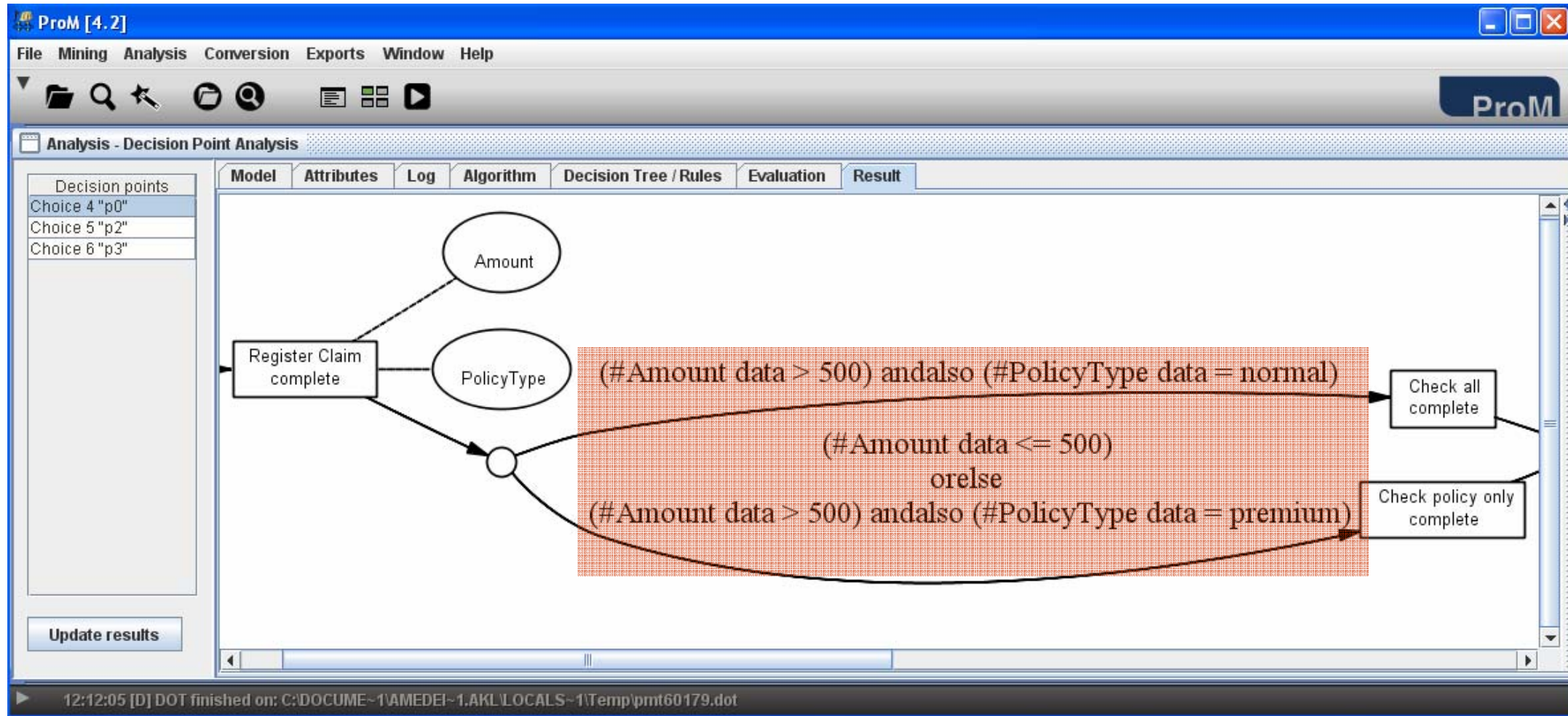
amount	clientID	policyType	class
1000	C567894938	premium	C
700	C938609223	normal	B
550	C135697567	normal	B
500	C568120443	normal	C
50	C493823084	normal	C
200	C945675110	premium	C



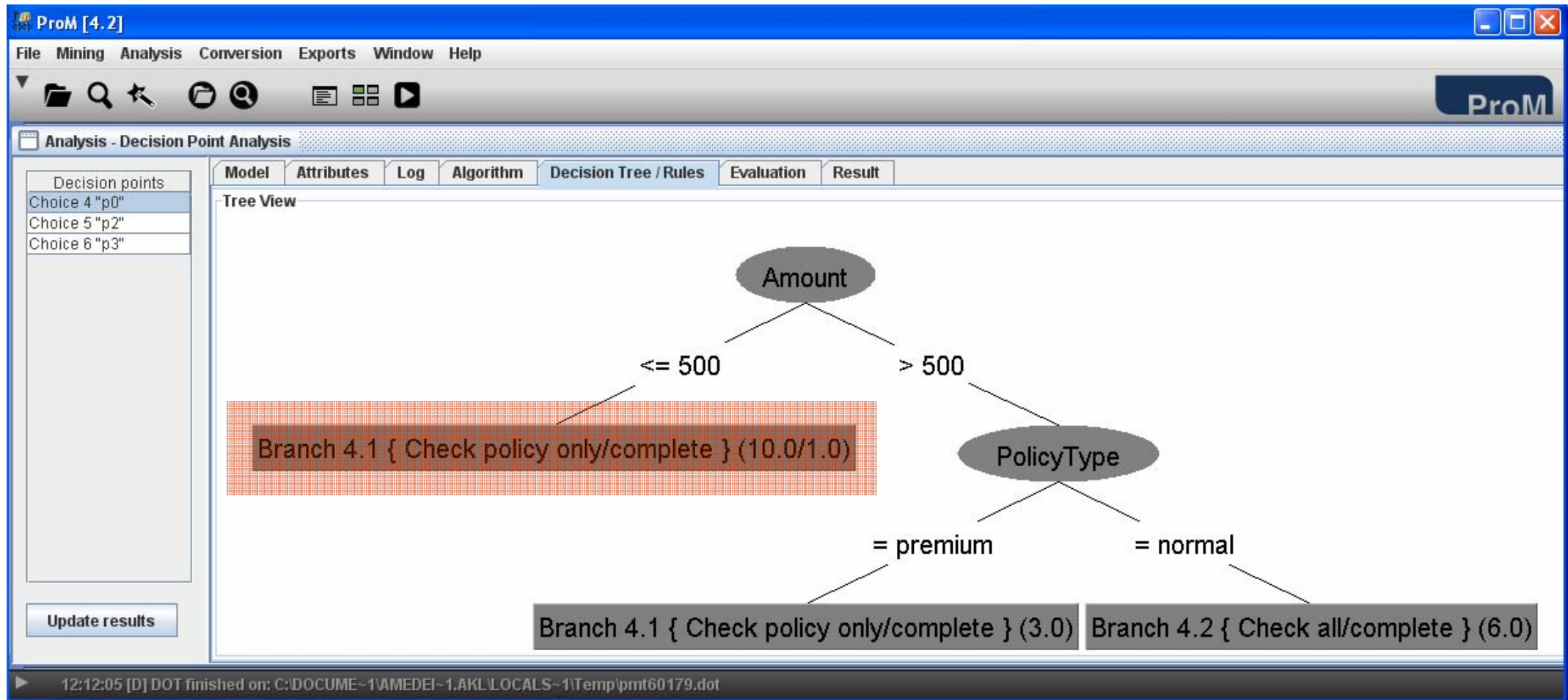
Discovered decision tree for point "p0"



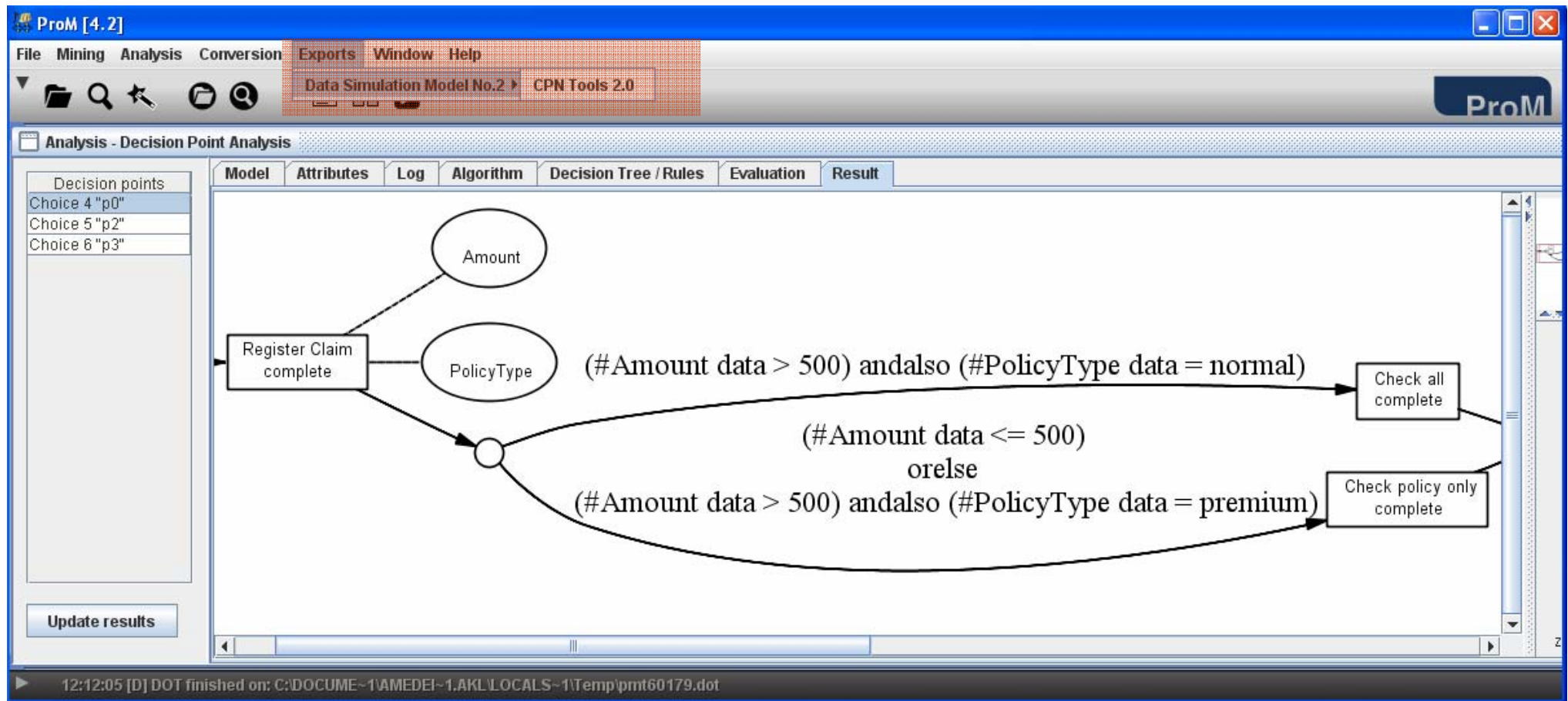
Decision Point Analysis: Example in ProM



Decision Point Analysis: Example in ProM



Decision Point Analysis



Extension techniques

- Decision Miner
- Performance Analysis

Performance analysis: pattern visualization



Performance Analysis with Petri Nets

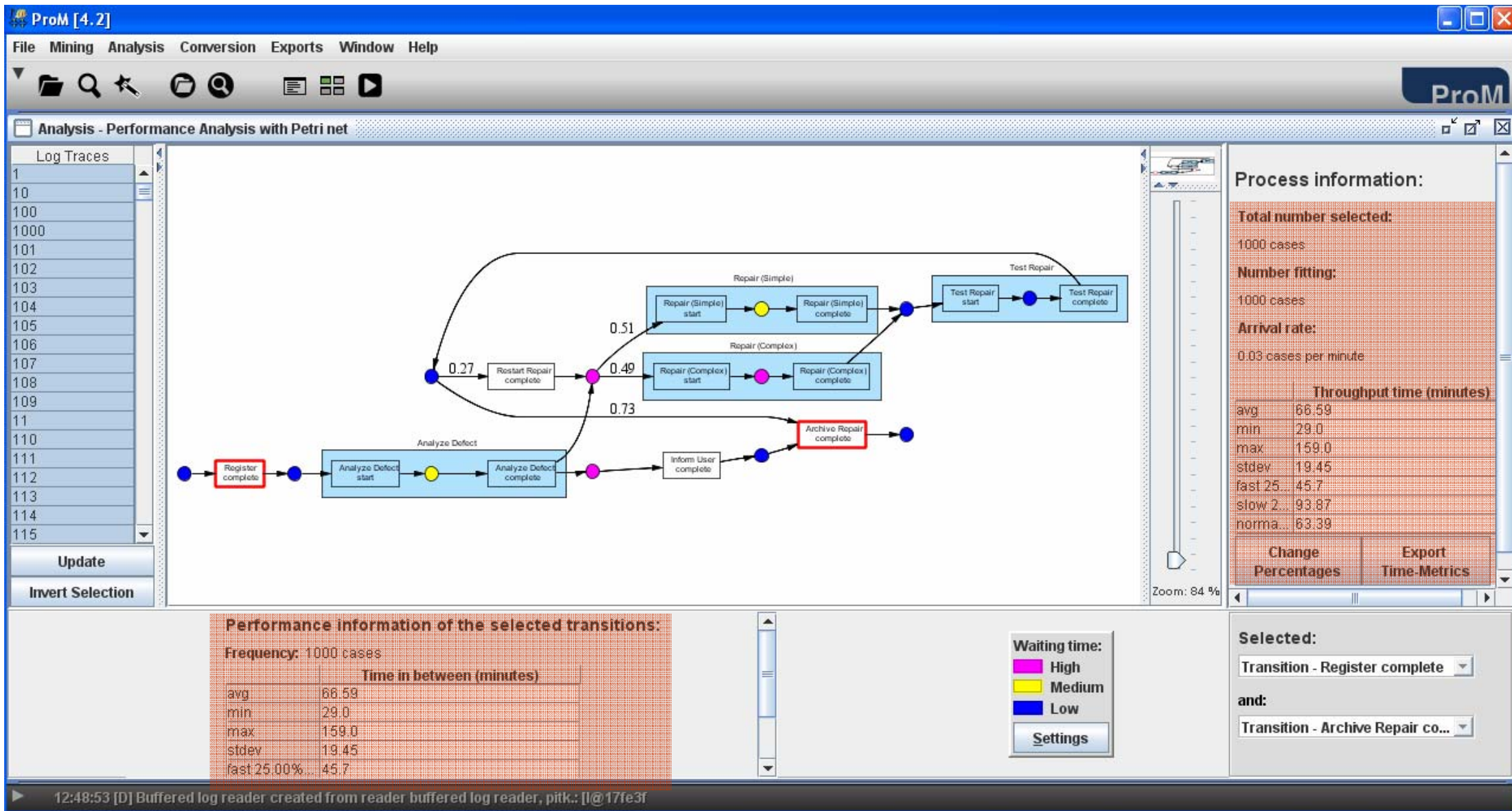
■ Motivation

- Provide different Key Performance Indicators (KPIs) relating to the execution of processes

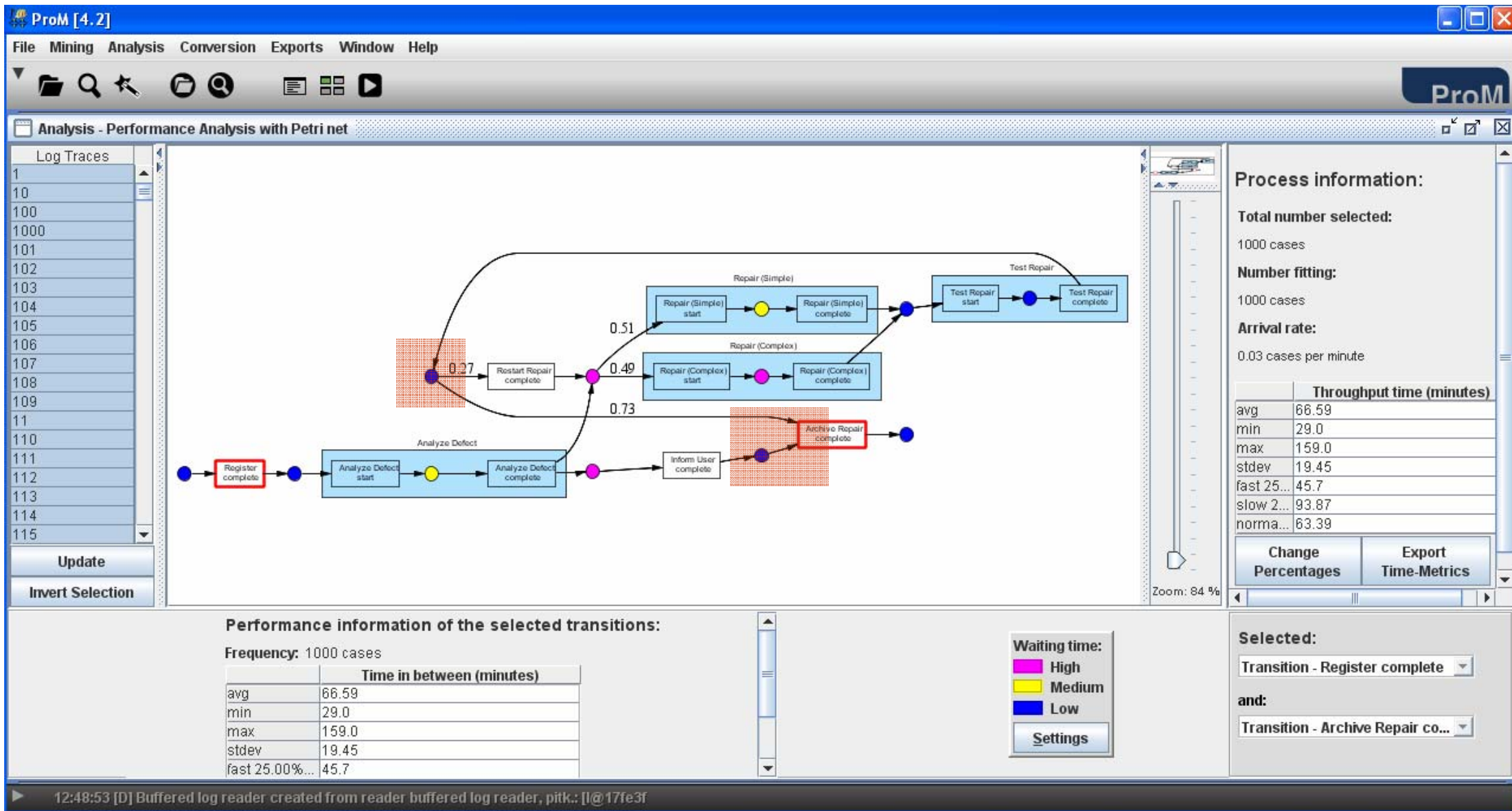
■ Main idea

- Replay the log in a model and detect
 - Bottlenecks
 - Throughput times
 - Execution times
 - Waiting times
 - Synchronization times
 - Path probabilities etc

Bottlenecks – Throughput Times

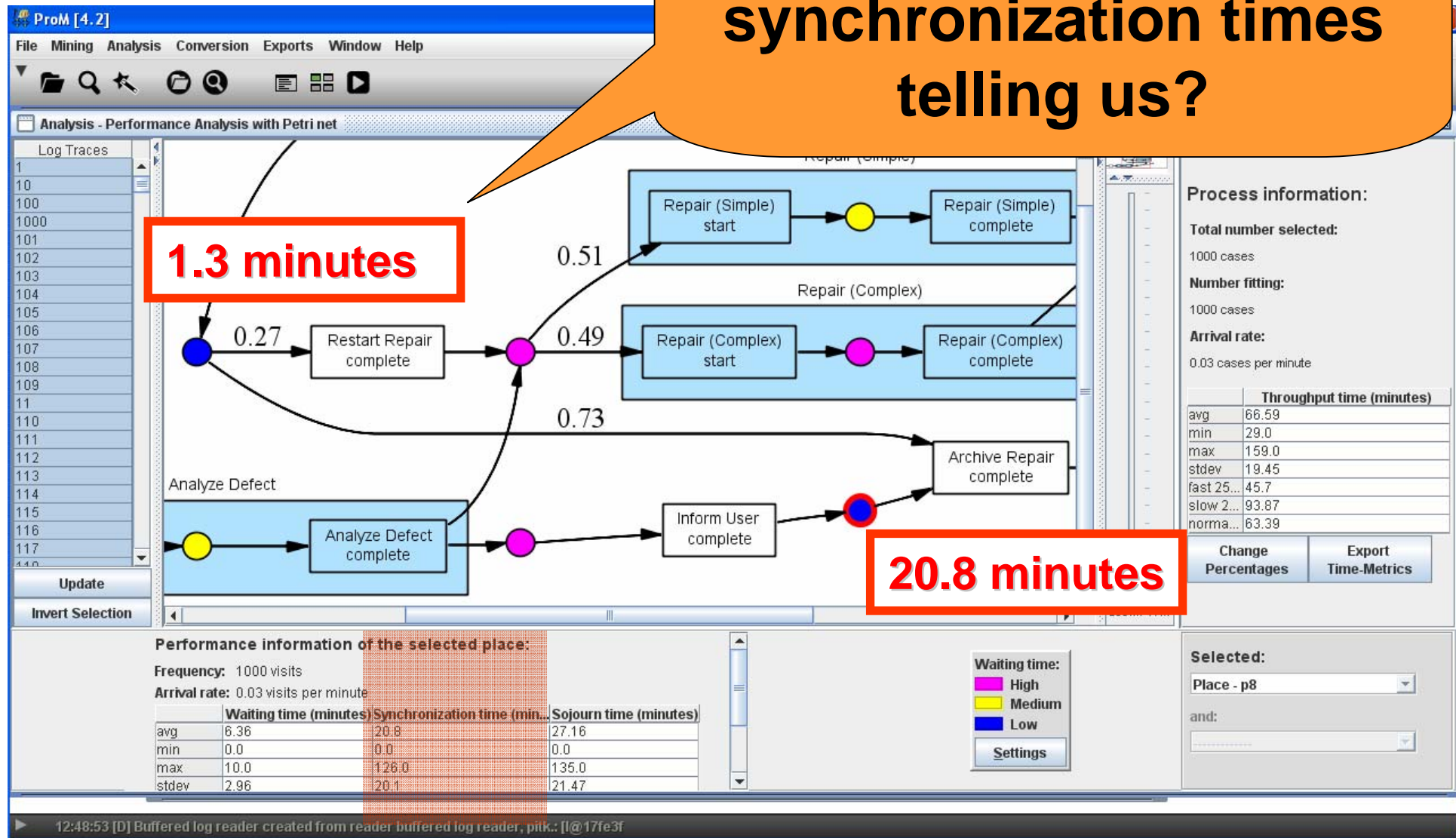


Bottlenecks – Synchronization Times

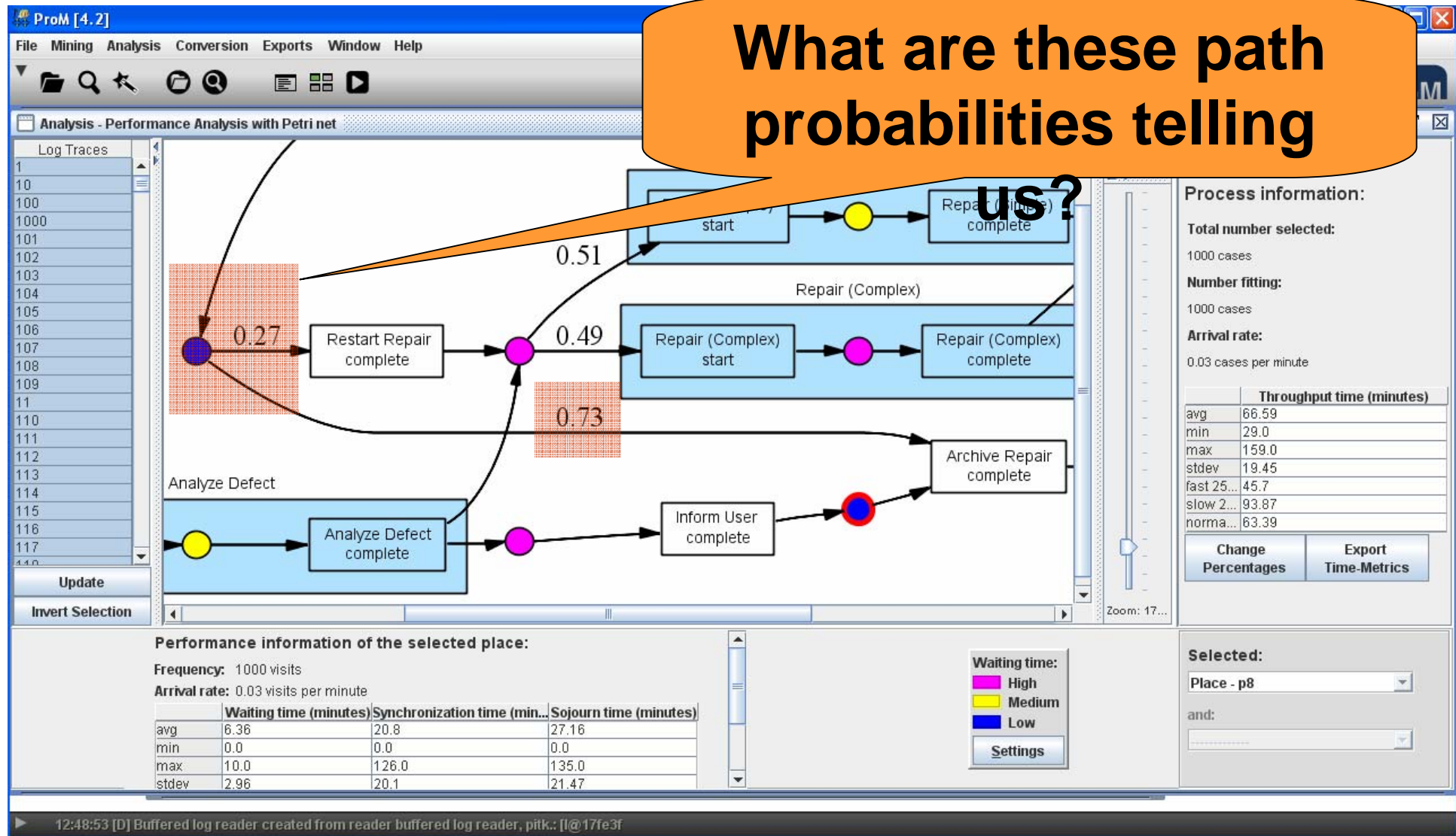


Bottlenecks – Synchronization Times

What are these average synchronization times telling us?



Bottlenecks – Path Probabilities



Performance Analysis with Petri Nets

