

Sanchayan Bhunia (4849650)

Process Mining

Mining algorithms on Purchase Data

Goals

Conversion of Data into Event Logs

Visualization of the Event Log

Choice of Models and Representational Bias:

- Alpha Miner with petrinet
- Heuristic Miner with Dependency Graphs
- Inductive Miner with Process Trees

Visualization of the Process Model

Compliance Checking




Performance analysis for potential bottlenecks

Organizational view and bottlenecks

Social Network analysis of the Flow of Work





Visualization of Purchase Log

AutoSaveOff









PurchasingExample.csv - Excel

FileHomeInsertDrawPage LayoutFormulasDataReviewViewHelpAcrobatPower PivotSearch















































Clipboard

Calibri11A^A^

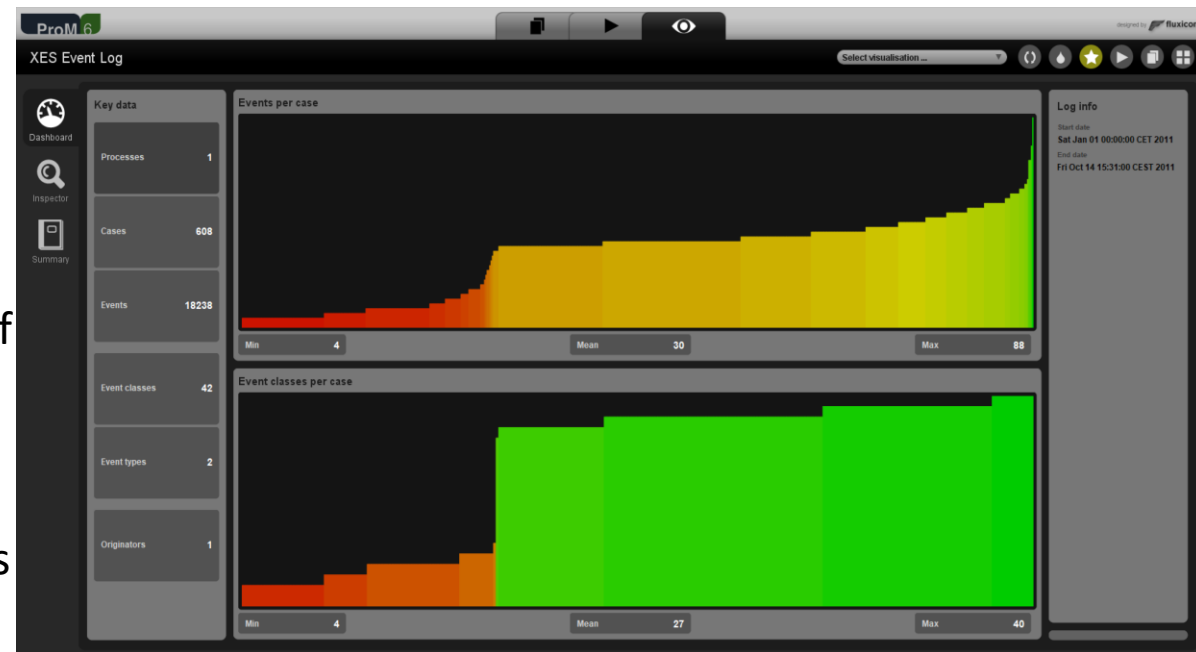


Font



Transformation

- After importing we'll use CSV to XES plugin and select Case Id, Activity, Start Time, Completion Time.
- If any error occurs while parsing a trace, ProM will omit the trace.
- We get a **log dialogue** as an visualizer.
- We have 608 cases and 18238 event logs for these cases.
- Event represents **Activity**.
- Then we have 21 event classes for **start** and **complete** of a class separately accounting 42.
- Minimum length of a trace is 4 events and max is 88 with an average length of 30.
- Combining both of the graphs we can say in some traces some activities are being repeated.



Analysis of the Event Log

- On the right side we have individual traces.
- The first trace has been repeated 88 times in the event log which is 14.47% of the all of the traces.

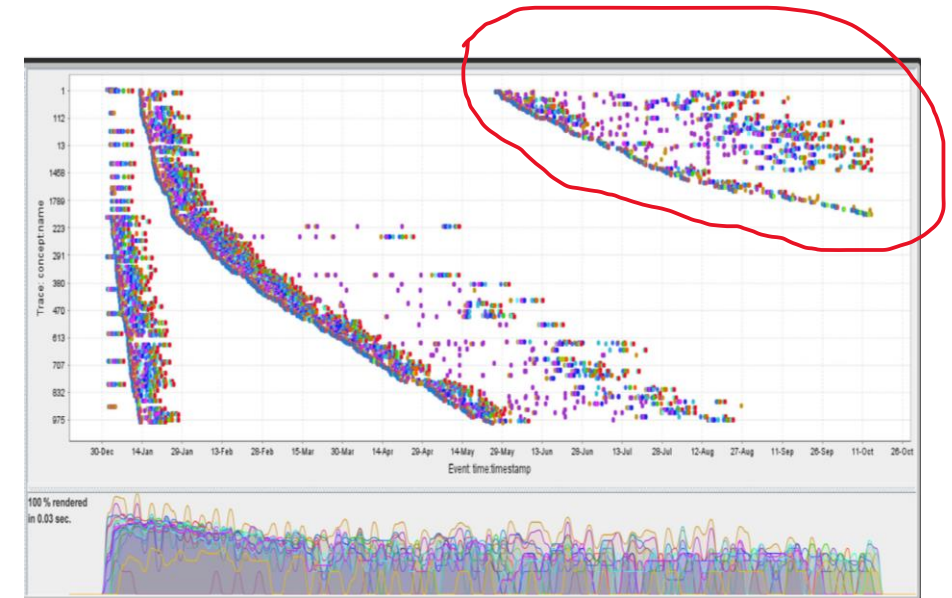
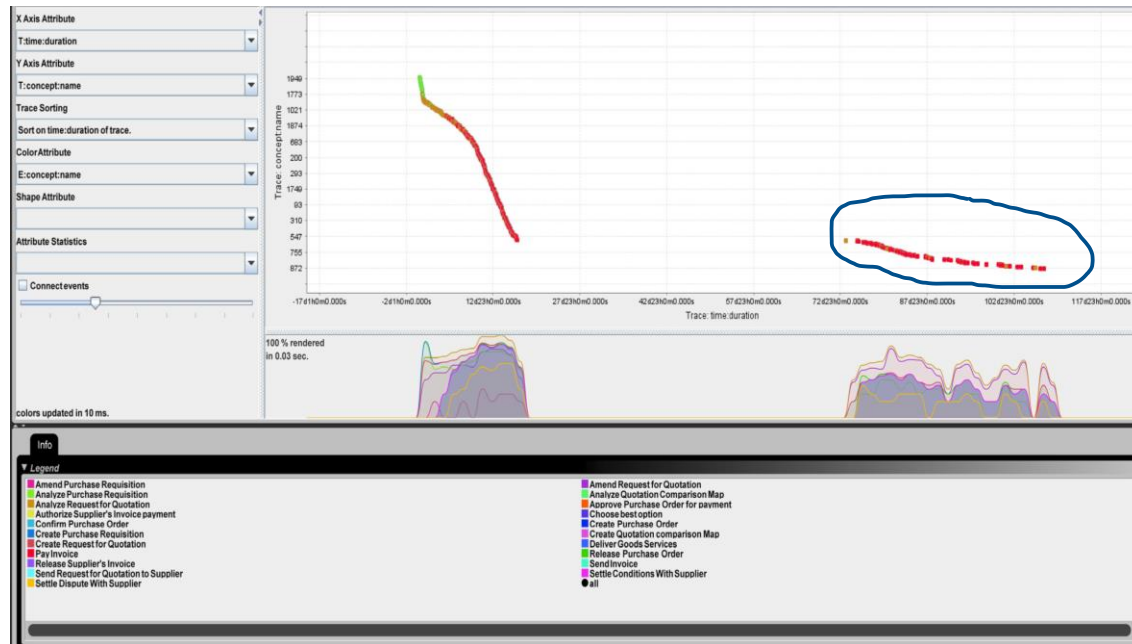
All events		
Total number of classes: 21		
Class	Occurrences (absolute)	Occurrences (relative)
Analyze Request for Quotation	2214	12.139%
Create Purchase Requisition	1216	6.667%
Amend Request for Quotation	1126	6.174%
Create Request for Quotation	1088	5.966%
Release Purchase Order	878	4.814%
Send Invoice	826	4.529%
Confirm Purchase Order	826	4.529%
Send Request for Quotation to Supplier	826	4.529%
Settle Conditions With Supplier	826	4.529%
Create Quotation comparison Map	826	4.529%
Create Purchase Order	826	4.529%
Analyze Quotation Comparison Map	826	4.529%
Pay Invoice	826	4.529%
Deliver Goods Services	826	4.529%
Authorize Supplier's Invoice payment	826	4.529%
Approve Purchase Order for payment	826	4.529%
Choose best option	826	4.529%
Release Supplier's Invoice	806	4.419%
Analyze Purchase Requisition	764	4.189%
Settle Dispute With Supplier	212	1.162%
Amend Purchase Requisition	22	0.121%
Start events		
Total number of classes: 1		
Class	Occurrences (absolute)	Occurrences (relative)
Create Purchase Requisition	608	100.0%
End events		
Total number of classes: 3		
Class	Occurrences (absolute)	Occurrences (relative)
Pay Invoice	413	67.928%
Analyze Request for Quotation	131	21.546%
Analyze Purchase Requisition	64	10.526%



- We can see that we have 21 event classes.
- “Analyze Request for Quotation” occurring 2214 times which is 12% in all of the events.
- There is one starting event and are three ending events with “Pay Invoice” occurring 68% of the times.

Event Log Statistics

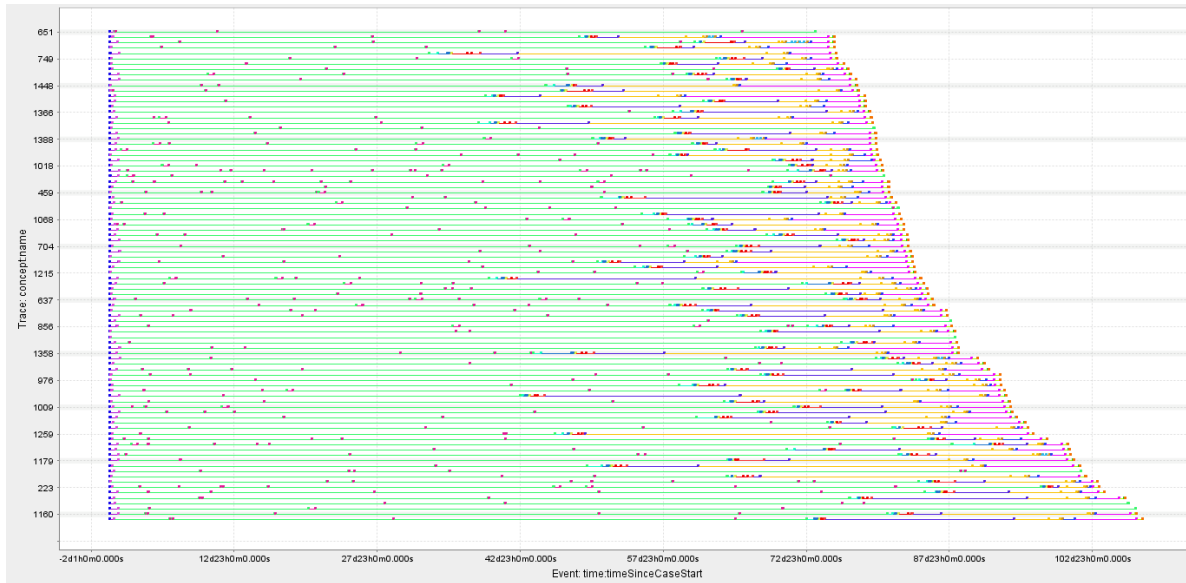
- In the dotted graph x axis is time and y axis is the trace
- Three major bands as most of the process starts in one of the three bands.
- Traces starting in the 3rd band i.e. after approx. 29th May have taken more time to end on an average.



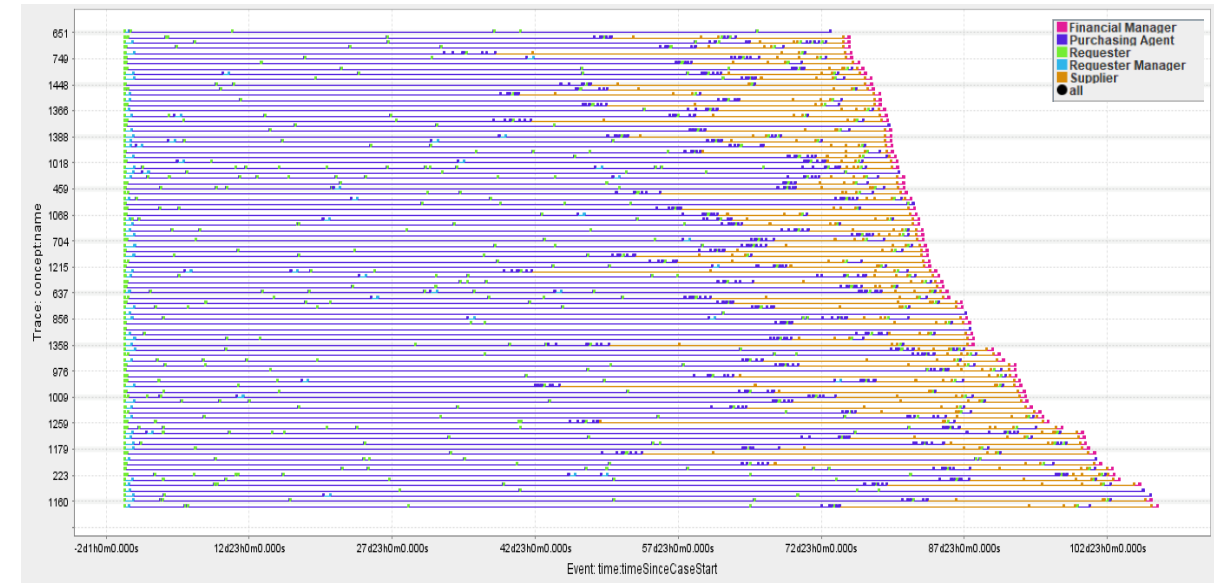
- In the time duration dotted plot x axis is the time duration and y axis is the traces.
- We see most of the traces took about 20 days to end.
- Traces indicated in Blue took more than 72 days to end.
- These are the outliers mostly occurring to a **bottleneck** and to be investigated further.

Analysis of long-time traces

- We will filter traces which took more than 20 days to complete by using “Trace Attribute Filter” in ProM plugin which is about 15% of all traces.
- Each of the horizontal line represent a trace in both of the graphs.
- Both of the graphs shows traces which took more than 70 days to complete and some taking even more than 120 days.



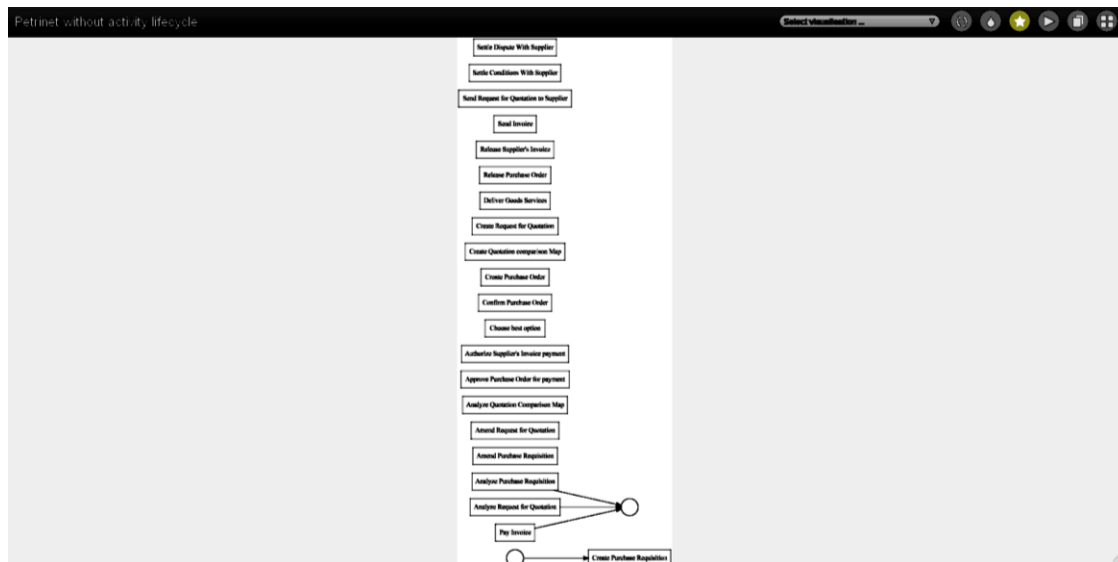
- Different colour shows different activities.
- Green activity caused delay in all of the traces with purple being the second.
- Those two activities are “Analyse Quotation Comparison” and “Analyse Purchase Requisition”.



- Different colour shows different actors performing the activities.
- Purple actor caused delay in all of the traces with yellow being the second.
- Those two actors are “Purchasing Agent” and “supplier”.

Process Mining in ProM: Alpha Algorithm

- From ProM plugins we will import Alpha Miner and apply it on event log.
- Alpha algorithm outputs a petrinet.



- The mined petrinet could only connect directly related transitions and was unable to find loops.
- Which leads to random firing of the parallel transitions leading explosion of reachable states.
- So, the petrinet is not sound and precision is worst.
- We must check for more flexible mining algorithms.

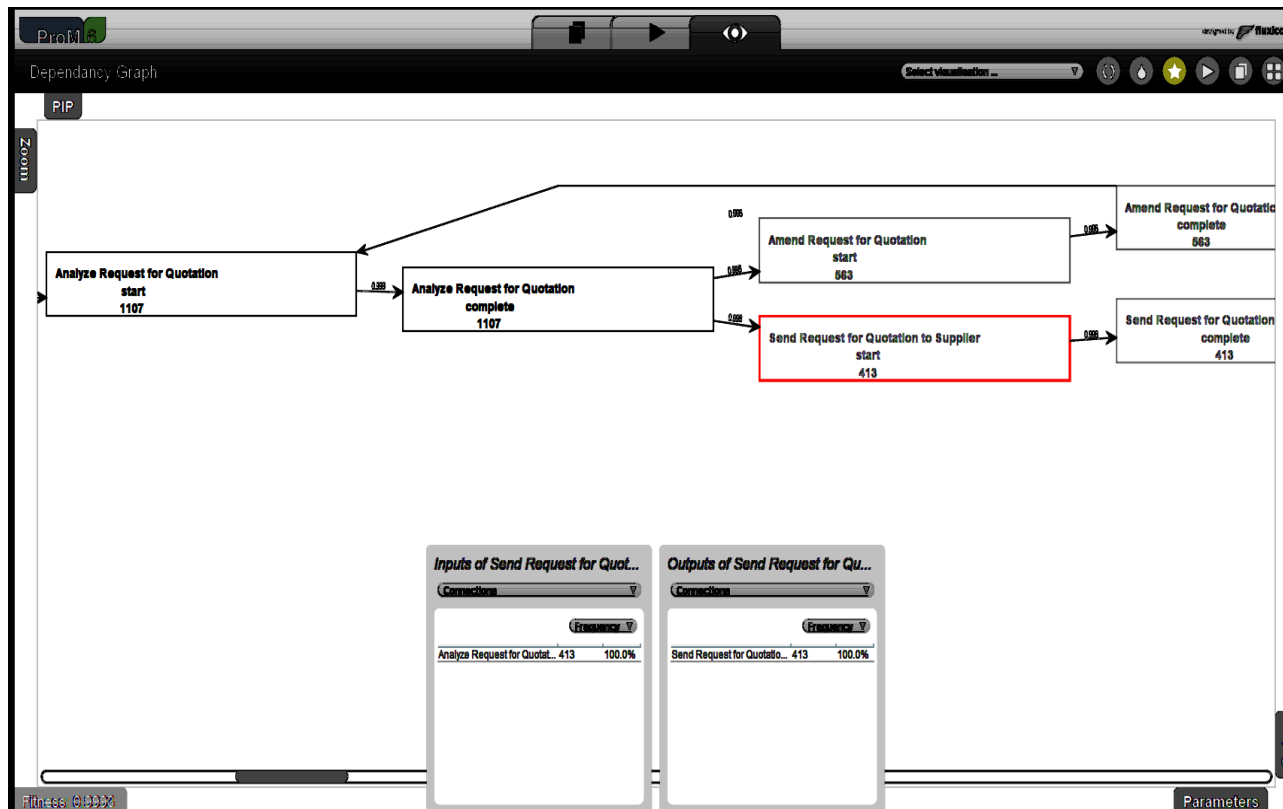


Heuristic Mining

Two step Process Discovery Technique

First step

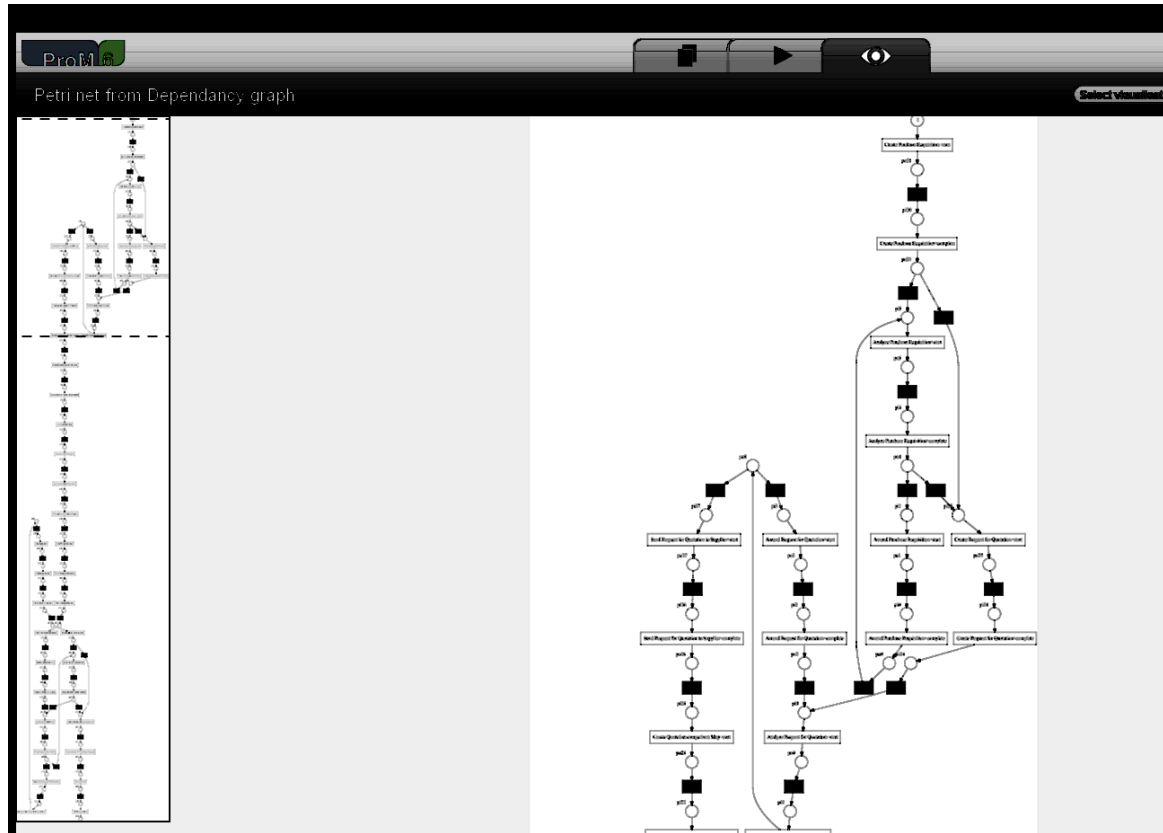
- **Dependency graph** is the representational bias of Heuristic mining.
- It uses frequency matrix to calculate dependencies between two activities.
- We will use Heuristic Miner plugin in ProM and set the dependency about 90%



- We get a model with fitness about 0.99
- All of the events are connected with arrows
- Every arrow contains the dependency of two events.
- Frequency of each activity is also mentioned below the name of the activity.

Second step

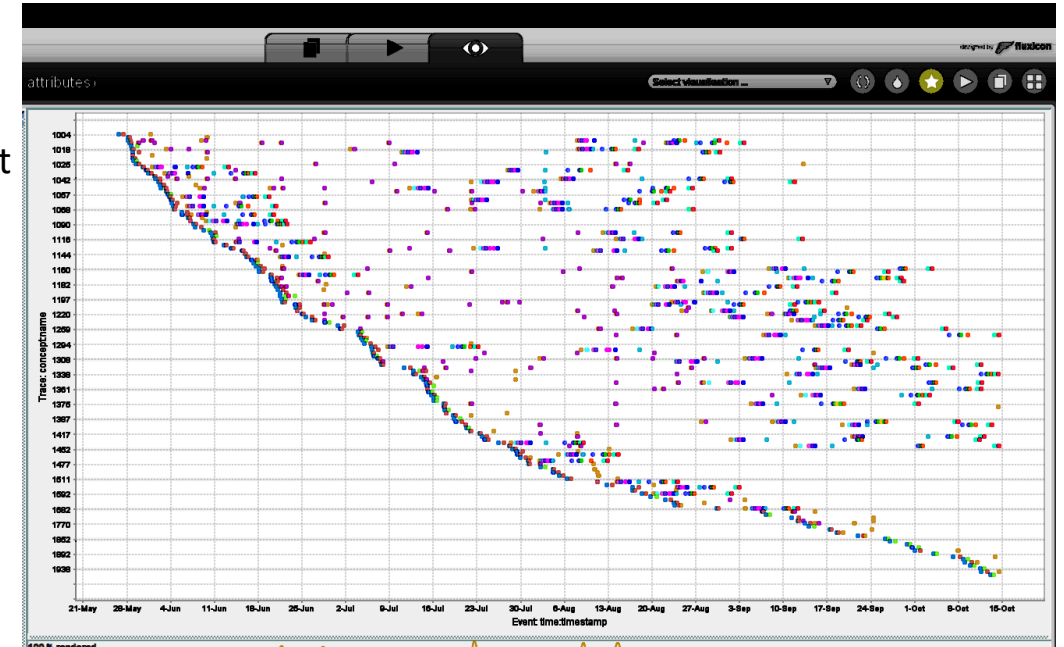
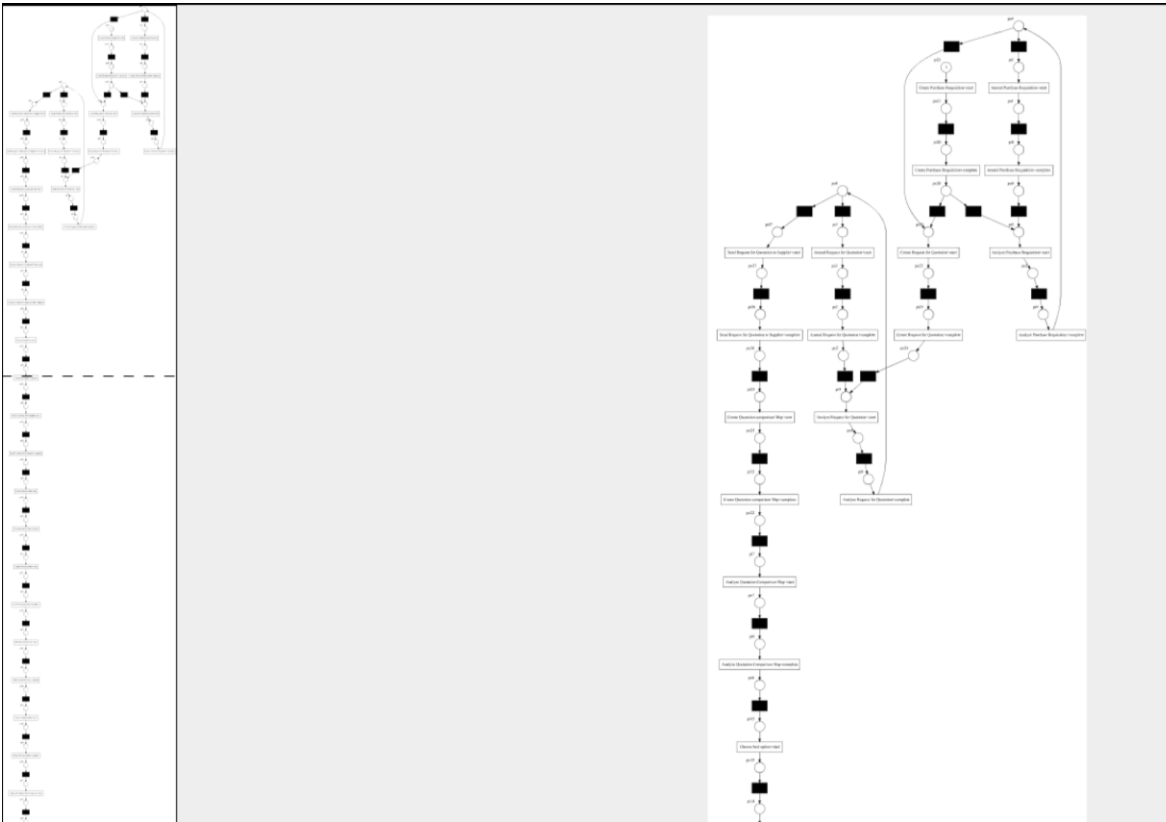
- Now we can convert the dependency graph into a C-net using a ProM plugin to find out joins.



- This is a cropped out section of the discovered model.
- The model has too many silent transitions making it difficult to understand.
- We can apply different filters on our event log to find out a simplistic model.

Mining on Traces after 29th May

- Using trace attribute filter we will filter out the traces after 29th May.
- Then we will follow two step Heuristic Mining on the filtered log to find out the petrinet from dependency graph as we did in the previous section.



- Dependency graph had a fitness of 99%.
- But we do not find simpler model.
- The model is different from the model we got considering the entire event log.
- Model still has many silent transitions and might lack soundness.
- We have to use a process miner which is more interactive and guarantees sound process model.

A dark blue, irregular ink splash or watercolor blotch serves as the background for the text. It has a textured, painterly appearance with some lighter blue and white speckles around its edges.

Inductive Miner

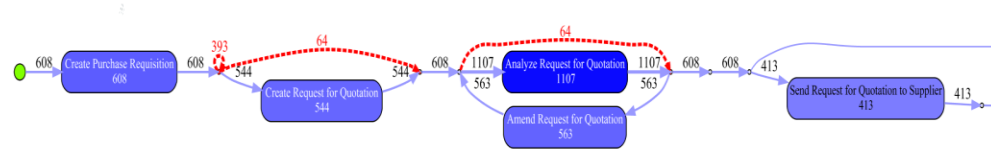
Mining using Process Trees

Visual Inductive Miner

- Inductive miner uses **process tree** as representational bias.
- Visual Inductive Miner converts that process tree into a dependency graph for easy visualization and animations.
- After running the visual inductive miner from ProM plugins on our **complete event log** we discover a model.
- We get this representation of the model by selecting 80% of all of the paths and Events.



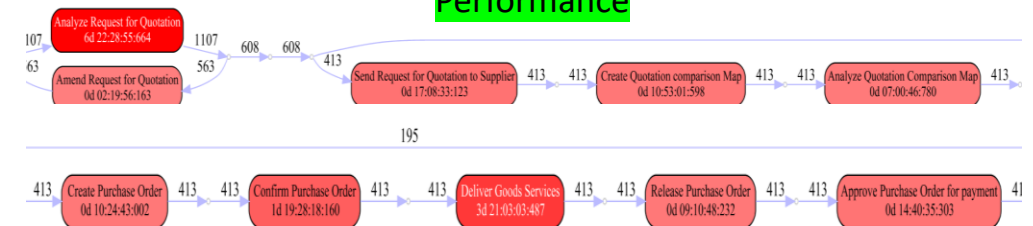
Compliance



- The numbers corresponds to the occurrence of the event or path in the log.
- The red dotted line is the representation where reality deviates from the model.
- For example, “Create Request for Quotation” is avoided 64 times out of 608 times.
- We can either change the operational system to prevent the violation or provide targeted training.

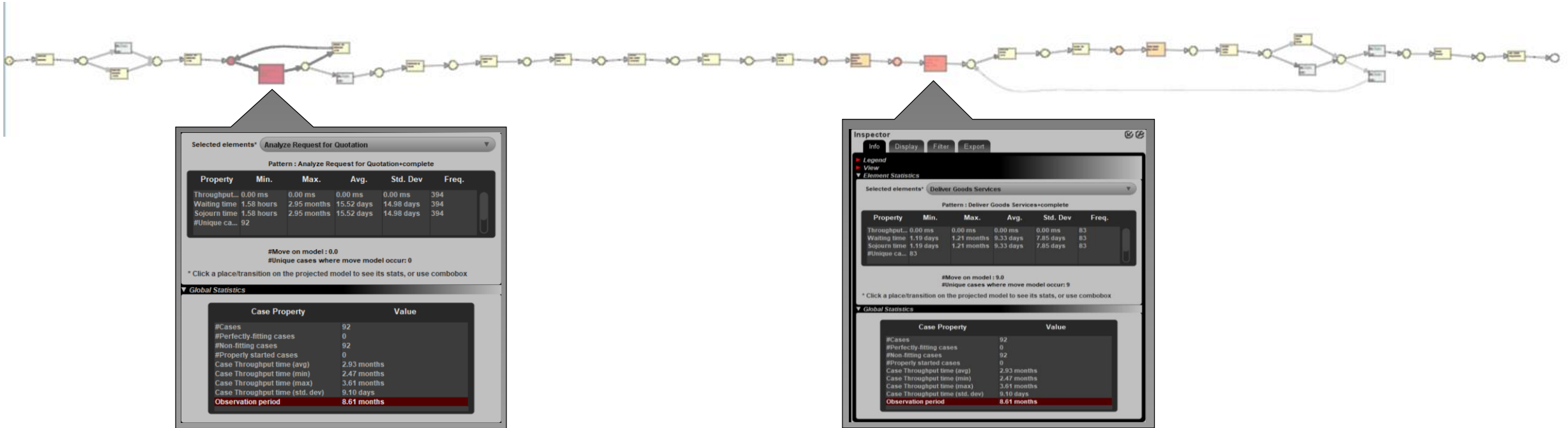
- We can see the bottle neck in the activity by choosing “Paths and Sojourn time” in the settings.
- “*Analyse Request for Quotation*” on average takes almost 7 days to process
- “*Deliver Goods Services*” which takes approx. 4 days on average.
- For further analysis we want to look into the cases which took more than 20 days to complete.

Performance



long-time traces

- Now we run Inductive Miner on traces which took more than 20 days to complete.
- After mining we save the model and use “Conformance checking and performance analysis” plugin in order to find out the bottlenecks.



Bottlenecks

- We find that “Analyze Request for Quotation” and “Delivery Goods Services” are two main bottlenecks.
- All of the 92 traces pass through “Analyse Request for Quotation” activity.
- 83 out of 92 cases have to pass through “Delivery Goods Services” activity.

Answered Questions



1. How does the process actually look like?

- ❖ Objective process map discovered using Heuristic and Inductive Miner while Alpha Algorithm failed to do so.



2. Are there deviations from the prescribed process?

- ❖ Yes, training or system change needed

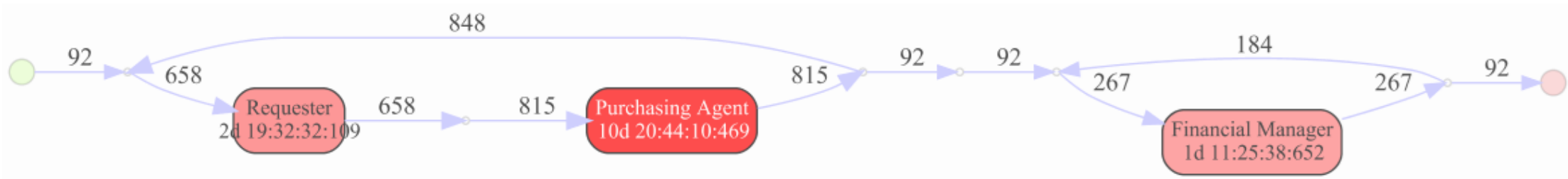


3. Do we meet the performance targets?

- ❖ Not by all (some take longer than 20 days)
- ❖ The 'Analyze Request for Quotation' activity is a huge bottleneck: Process change is needed

Organizational View

In order to get an organizational view of the process model we will select “Role” in the Activity log as our classifying activity in Inductive miner.



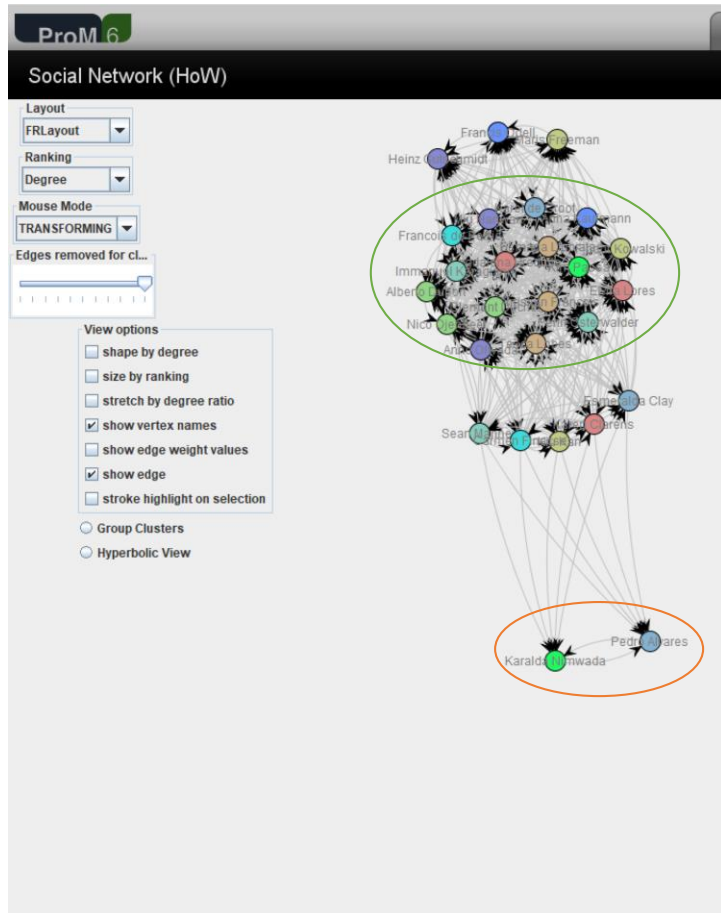
- In the discovered model is represented in performance view.
- Here it is clear that the “Purchasing Agents” are causing the biggest delays in the process on an average 10 days.



Social Network

Analysis of Purchase Log

Hand over of Work (HoW)



- Hand over of work as name suggest shows a connected graph of actors with their works connecting them.
- After using the dedicated ProM plugin we filter out infrequent edges.
- Then we select “FRLayout” with “Degree” of workload as ranking to get the connected graph.
- The cluster in the centre circled in green usually have much of the work hence most of the role in the process.
- The cluster furthest out circled in Orange have least work assigned, hence have least role in the entire process.



Thank You