

# Process Mining Project (a.y. 2021/2022)

UNICAM

Professor: Re Barbara

Student: Serpetta Damiano

## Introduction

Project developed by Damiano Serpetta for the Process Mining exam in University of Camerino.

The goal of the project is about to study and analyze events log in order to discover process models and checking conformance of these models.

Process Mining is entirely done by using [ProM Tool](#).

## Process Mining

Process Mining is a methodology used to discover, monitor, and improve processes that already exist within a business by relying on data. The goal of using process mining is to explore where existing business processes are inefficient and address those critical areas.

Process Mining is fundamental for those three applications:

- Process Discovery from events log.
- Conformance of discovered processes related to event log.
- Enhancement of existing discovered processes.

## Data

The 'Hospital Billing' event log was obtained from the financial modules of the **ERP system of a regional hospital**. The event log contains events that are related to the **billing of medical services** that have been provided by the hospital. Each trace of the event log records the activities executed to bill a package of medical services that were bundled together. The event log does not contain

information about the actual medical services provided by the hospital. The 100,000 traces in the event log are a random sample of process instances that were recorded over **three years**. Several attributes such as the 'state' of the process, the 'caseType', the underlying 'diagnosis' etc. are included in the event log. Events and attribute values have been **anonymized**. The time stamps of events have been randomized for this purpose, but the time between events within a trace has not been altered.

Source: [4TU.ResearchData](#)  
Data Repository: [Hospital Billing - Event Log](#)

# Log Description

Total number of process instances: **100000**  
Total number of events: **451359**  
Event classes defined by **concept:name**

## All events

Total number of classes: **18**

Class	Occurrences (absolute)	Occurrences (relative)
NEW	101289	22,441%
FIN	74738	16,558%
RELEASE	70926	15,714%
CODE OK	68006	15,067%
BILLED	67448	14,943%
CHANGE DIAGN	45451	10,07%
DELETE	8225	1,822%
REOPEN	4669	1,034%

CODE NOK	3620	0,802%
STORNO	2973	0,659%
REJECT	2016	0,447%
SET STATUS	705	0,156%
EMPTY	449	0,099%
MANUAL	372	0,082%
JOIN-PAT	358	0,079%
CODE ERROR	75	0,017%
CHANGE END	38	0,008%
ZDBC_BEHAN	1	0,0%

# Start events

Total number of classes: 1

Class	Occurrences (absolute)	Occurrences (relative)
NEW	100000	100,0%

# End events

Total number of classes: 14

Class	Occurrences (absolute)	Occurrences (relative)
BILLED	63498	63,498%
NEW	22407	22,407%

DELETE	8215	8,215%
FIN	3611	3,611%
CODE OK	948	0,948%
SET STATUS	600	0,6%
EMPTY	444	0,444%
RELEASE	107	0,107%
MANUAL	85	0,085%
JOIN-PAT	47	0,047%
CHANGE DIAGN	19	0,019%
CODE NOK	14	0,014%
REJECT	4	0,004%
STORNO	1	0,001%

## Data Visualization

ProM provides some type of data visualization: *traces, events, classes, resources, attributes, and all information about the xes file.*

Data Visualization provide functionalities in order to **relate various informations together**. One example is shown in the screenshot below, where we can see *Events per case* and *Event classes per case*

*Event Log Summary Visualization:*

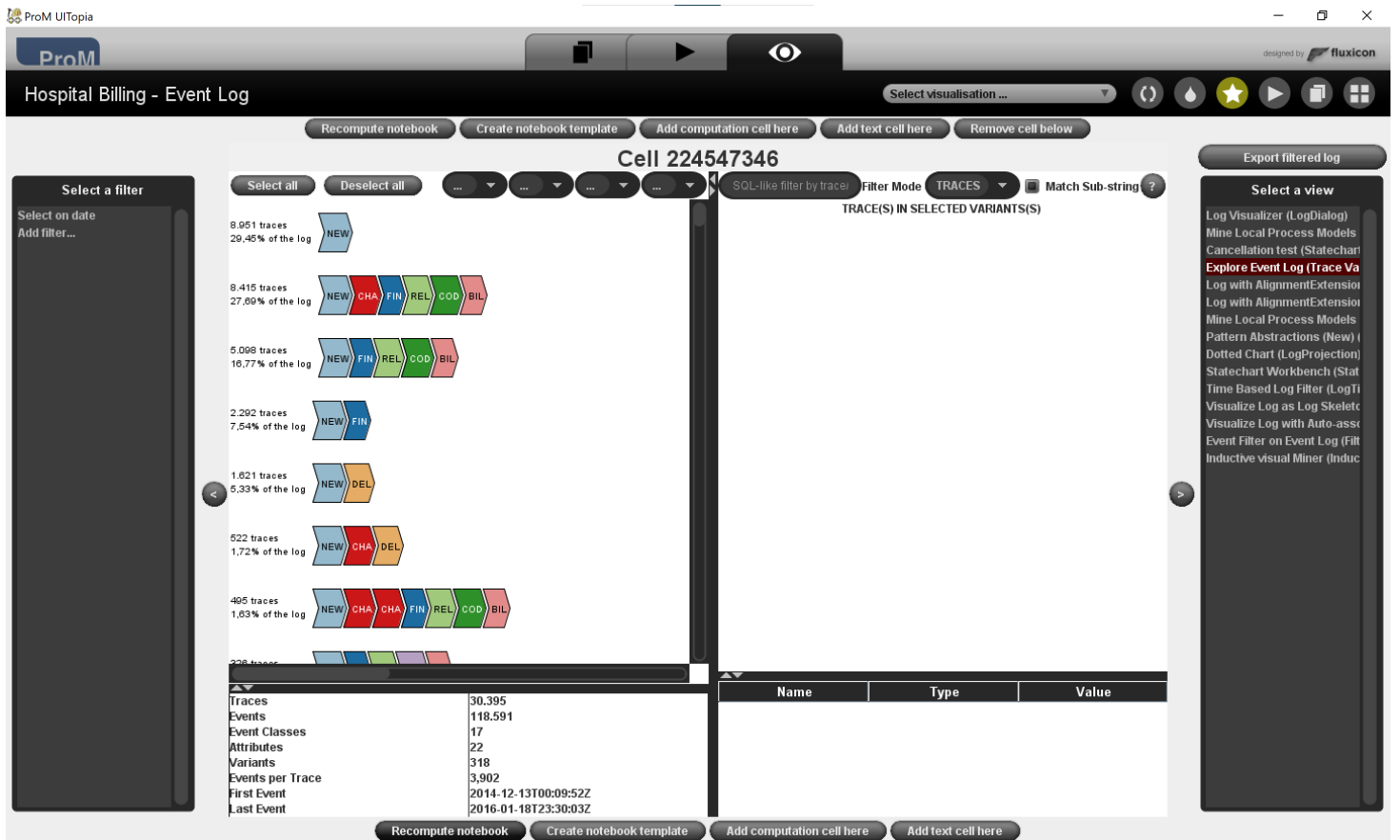


# Filtering Data

In ProM, **event log** can be filtered in some different way, based on various attributes.

One possible filter for the event log can be represented by time: specifying a time window for traces, all traces not contained in that space-time will be deleted.

*Example of filtering data contained between 13/12/2014 and 18/01/2016.*



Example of filtering data based on traces which contain event 'DELETE'.



Filtering Data can be done using the **Notebook Tool** in ProM, and they can be exported so that they can be used in all Process Mining actions.

## Filtering using Simple Heuristic

A way to clean data is to use the action '**Filter Log using Simple Heuristics**'. This type of ProM action has to be configured by events. Classifiers, events and types can be chosen in order to filter data.

In this case:

- All *classifiers* are included.
- All traces which end with events 'BILLED', 'DELETE', 'FIN' o 'NEW', in state **complete**, are included.



## Process Mining Algorithms

Algorithms for **Process Discovery** and **Conformance Checking** are covered in this paragraph. For this project, I choose to discover processes with 3 algorithms: **Alpha**, **Alpha+** and **Heuristic**. In

particular, for the last one algorithm (*Heuristic*), **different approaches**, for *process discovery* and *conformance checking*, are used.

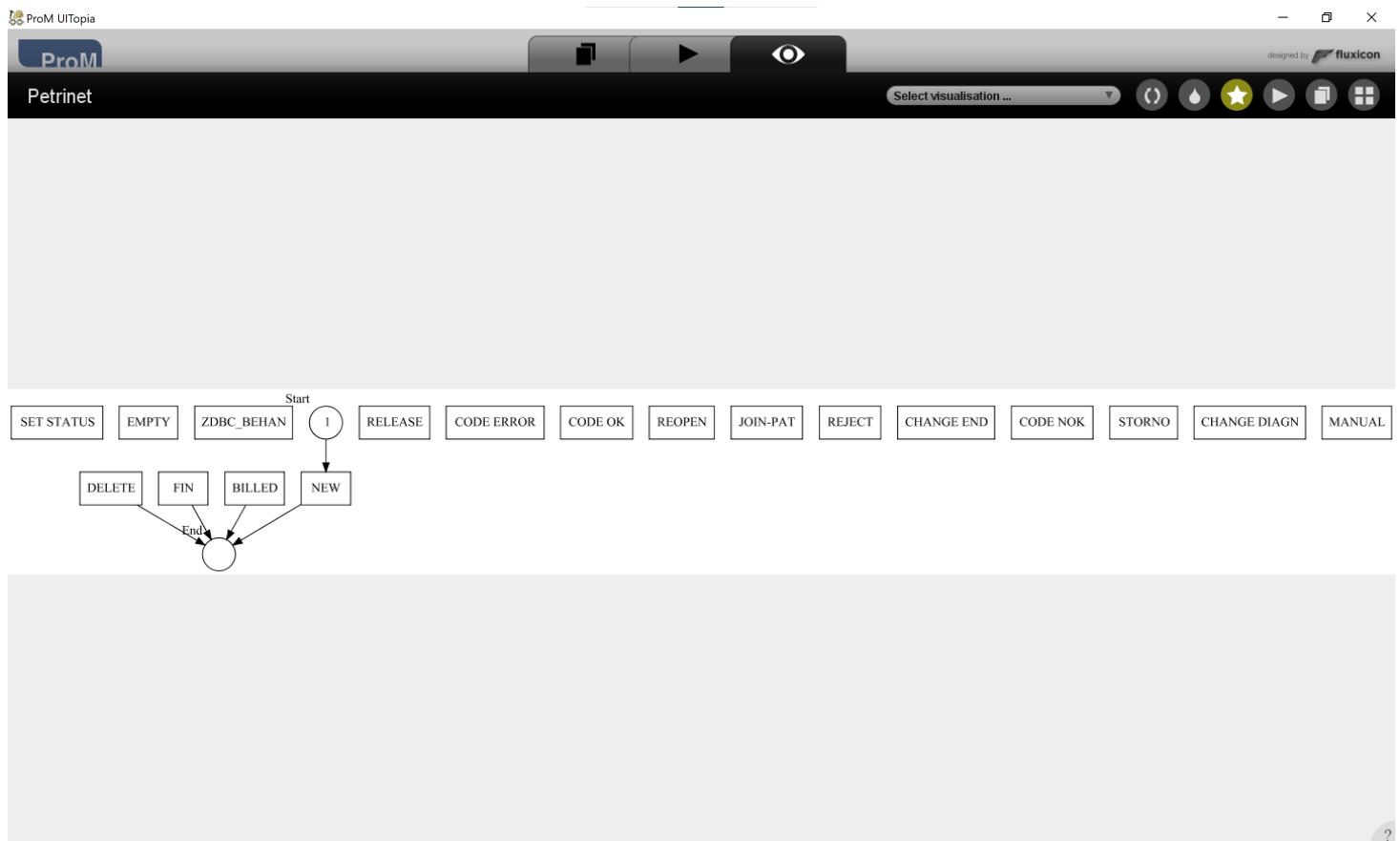
## Alpha

Alpha is an Algorithm which focuses on **Control Flow**.

Relations between activities are discovered, and a **footprint table** of *direct successions*, *causality*, *parallels* and *choices* is calculated.

However, even if Alpha Algorithm is magnificent from a sperimental and mathematic point of view, it's not suitable for real events log.

*Discovered Petri Net using Alpha:*



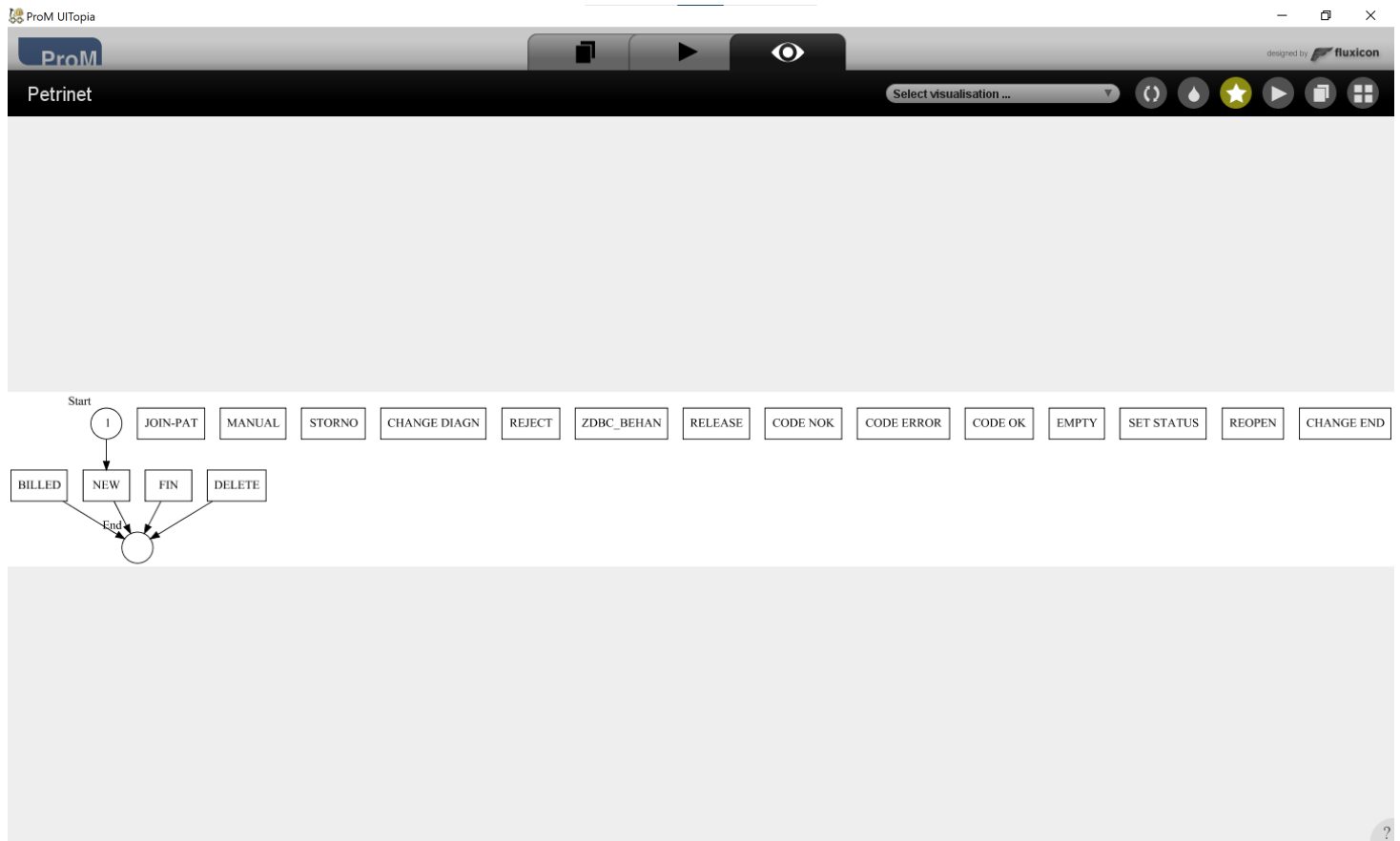
In this case, we can see that it's not even close to our expectations, because of the quantity of **unconnected activities**.

## Alpha+

Alpha+ is an improved Alpha Algorithm which can discover loops of lenght of 1 and 2.



## Discovered Petri Net using Alpha+:



Also in this case there are too much unconnected activities in this Petri Net.

We can assume that Alpha and Alpha+ are not suitable for this Events Log.

## Heuristic

Heuristic Miner is a practical applicable mining algorithm that can deal with noise, and can be used to express the main behavior (i.e. not all details and exceptions) registered in an event log.

**Two approaches** are covered in this project, based on two kind of actions in ProM.

## Approach 1: 'Mine for a Heuristic Net using Heuristic Miner'

*Configuration of Heuristic Miner action:*


Please select classifier to use:

Event Name ▼

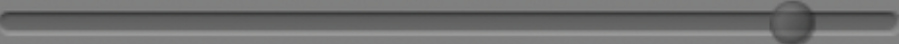
## Thresholds

Relative-to-best:  15.0

Dependency:  70.0

Length-one-loops:  90.0

Length-two-loops:  90.0

Long distance:  90.0

## Heuristics

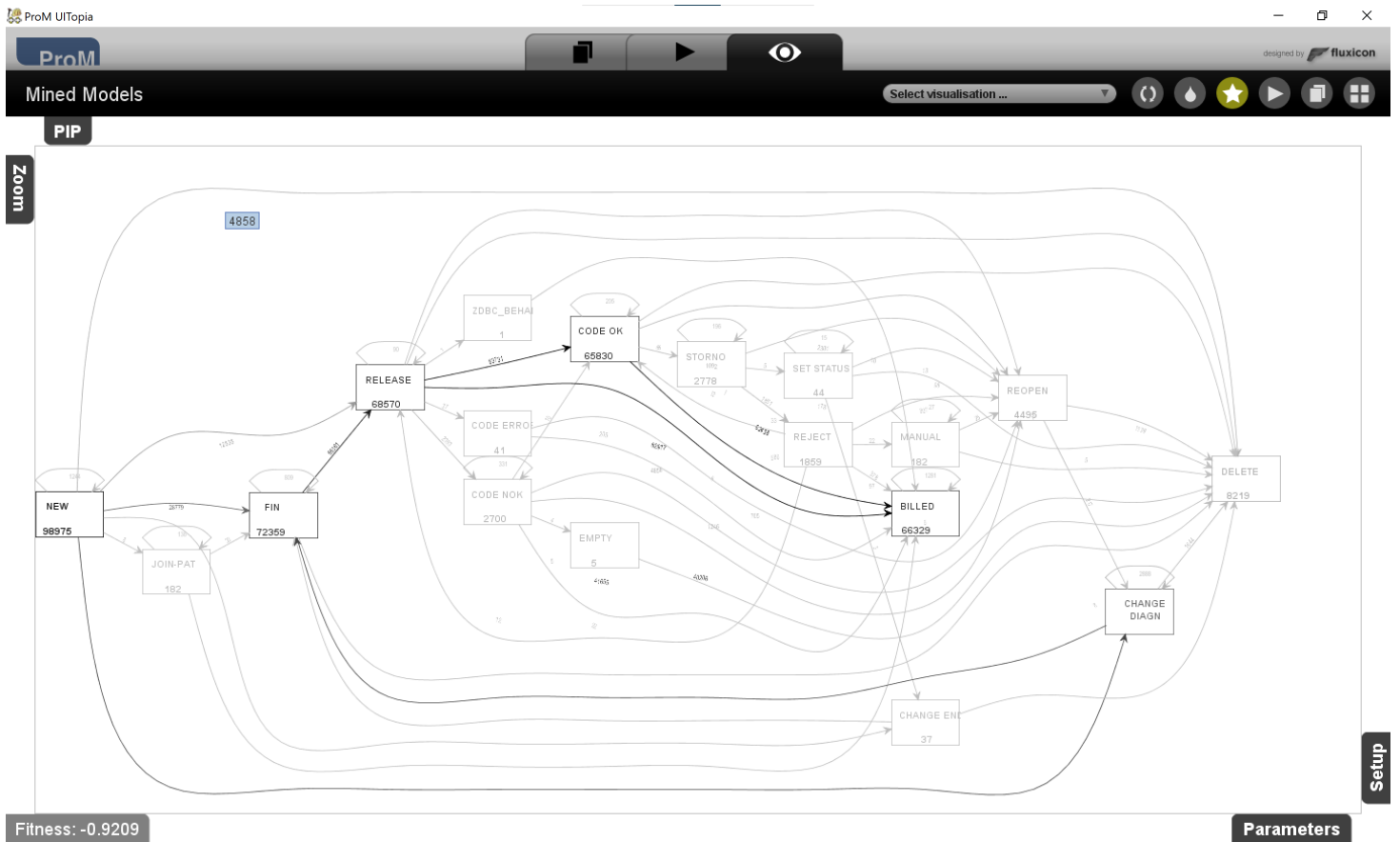
All tasks connected: ☐

Long distance dependency: ☐

Ignore loop dependency thresholds: ☐

Various configurations have been tried in ProM, and this configuration gives rise to the following model.

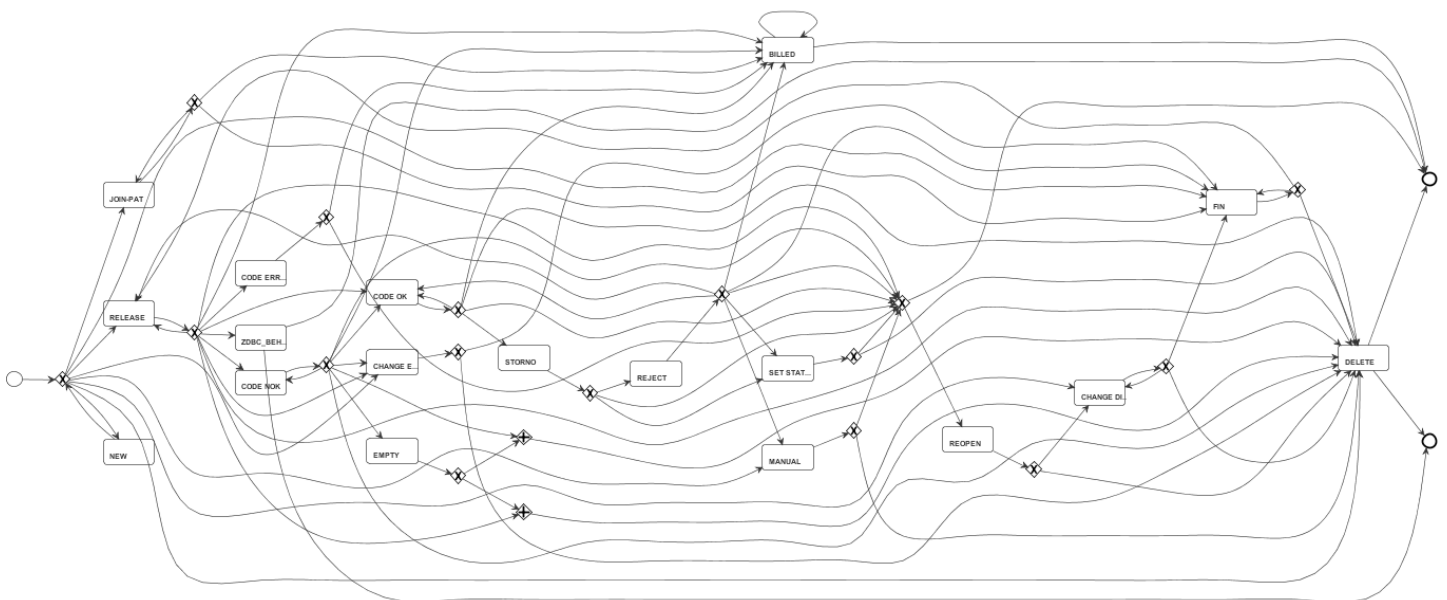
*Discovered Model (Dependency Graph):*



With dependency graph we can also discover the related Petri Net, and consequently mapping it to a BPMN Model.

The discovered Petri Net **is not sound** and this was the main problem with this approach.

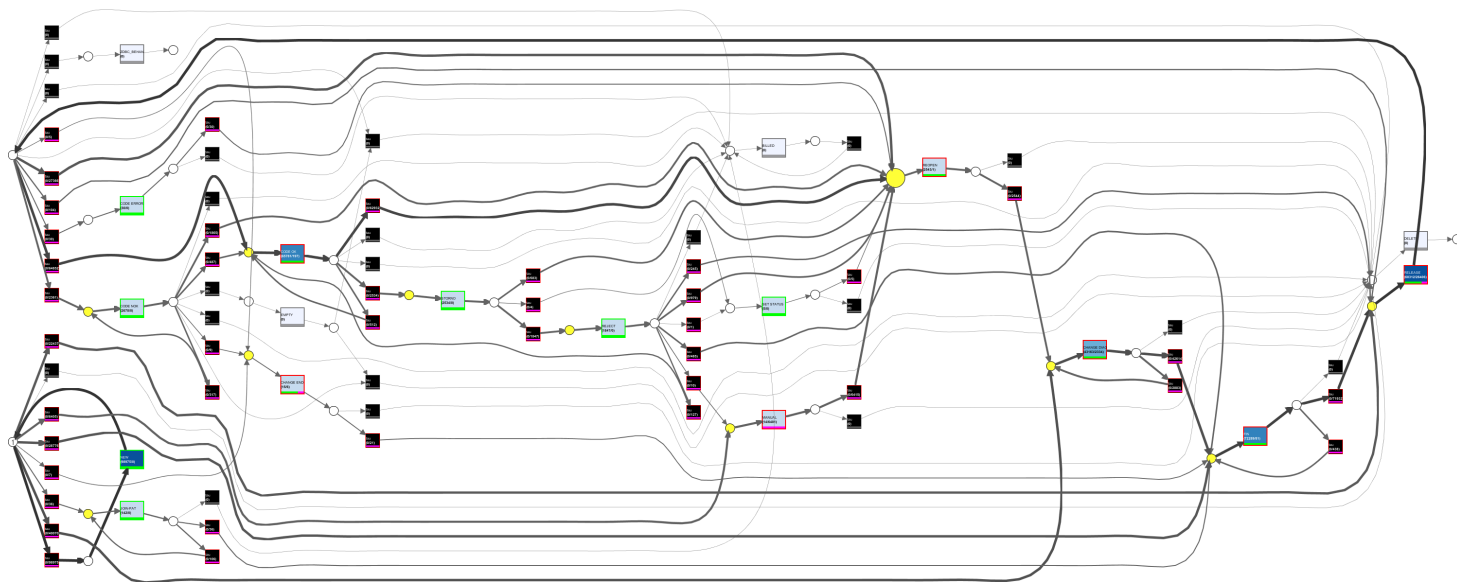
*Discovered Petri Net mapped to BPMN:*



# Conformance Checking

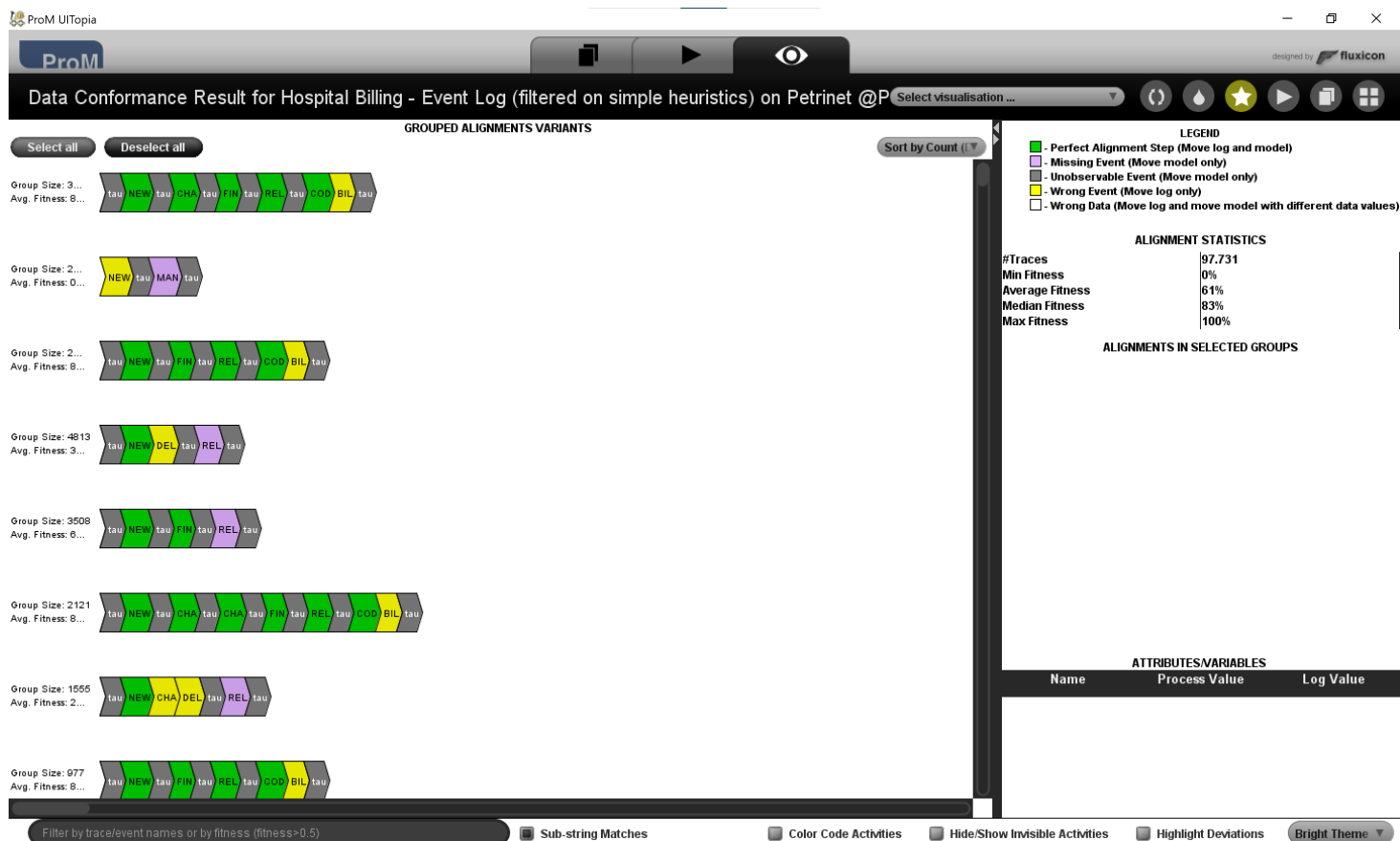
Conformance Checking is done by using the action '*Replay a Log on Petri Net for Conformance Analysis*', which provides an implementation of **Alignment**.

*Conformance Checking between Model and Log:*



In ProM we can see, for each trace, the correspondent **alignment** and **moves**.

*Alignments Sample Data:*



Properties of Conformance Checking:

Property	Value
Traversed Arcs	214.59343504108227
Calculation Time (ms)	3.0687806325526177
Raw Fitness Cost	1.1712557939650672
Max Move-Log Cost	4.47524326979157
Num. States	86.05312541568175
Trace Fitness	0.7211971276570514
Move-Model Fitness	0.8368869751317306
Move-Log Fitness	0.8425838800962936
Max Fitness Cost	5.47524326979157
Trace Length	4.47524326979157

Here we can see that **Trace Fitness** is around **0.72**.

## Approach 2: 'Interactive Data-aware Heuristic Miner (iDHM)'

Second approach for Heuristic Mining is achieved by using the action '*Interactive Data-aware Heuristic Miner (iDHM)*'.

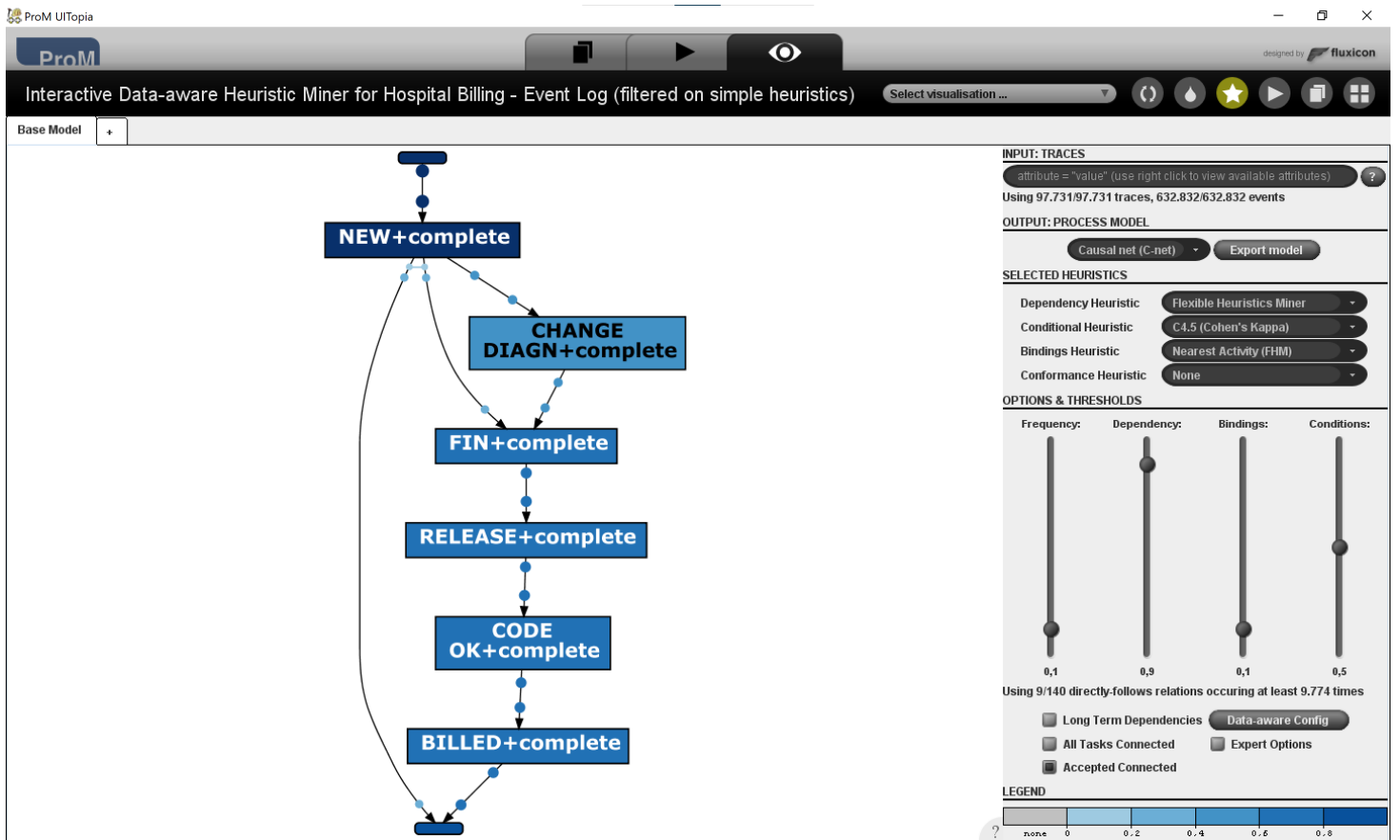
Unlike the first approach, this method is more configurable and provide to create a Petri Net which is **Sound**.

The **Approach 2** is also divided in **2 tries**.

### First Try

First Try consists in discovering and create a simpler Model, with minor activities.

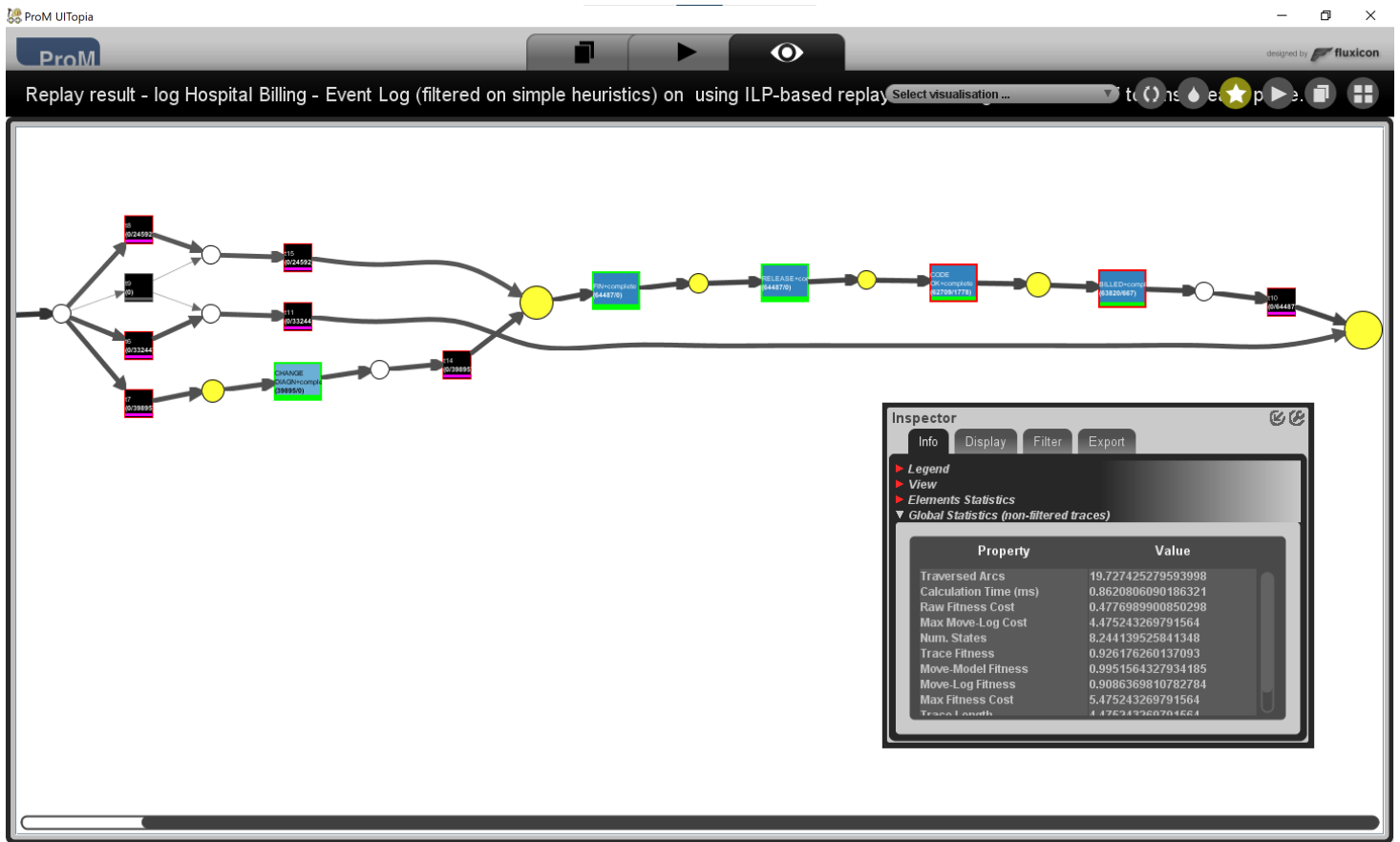
*First Model Discovered:*



In the right part of the screenshot, there is the **configuration** for obtaining this model, with the relative *thresholds*.

## Conformance Checking

*ILP Alignment:*



## ILP Alignment Sample Data:

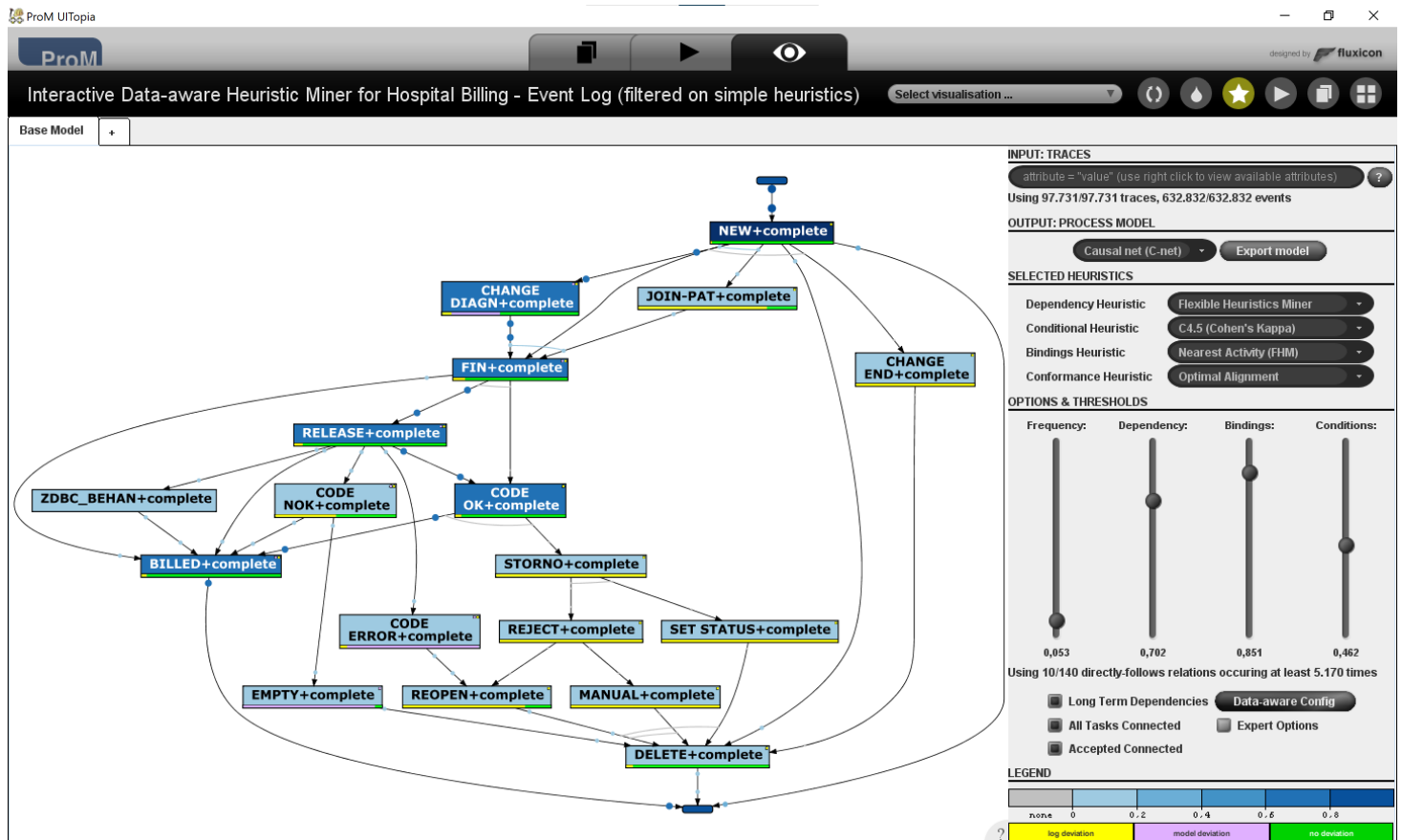


Second Try

Second try is way more complicated than the first and it contains more activities.

In this case some parameters have been modified. In particular, the **Dependency Threshold** was lower, in order to get more activities in the model.

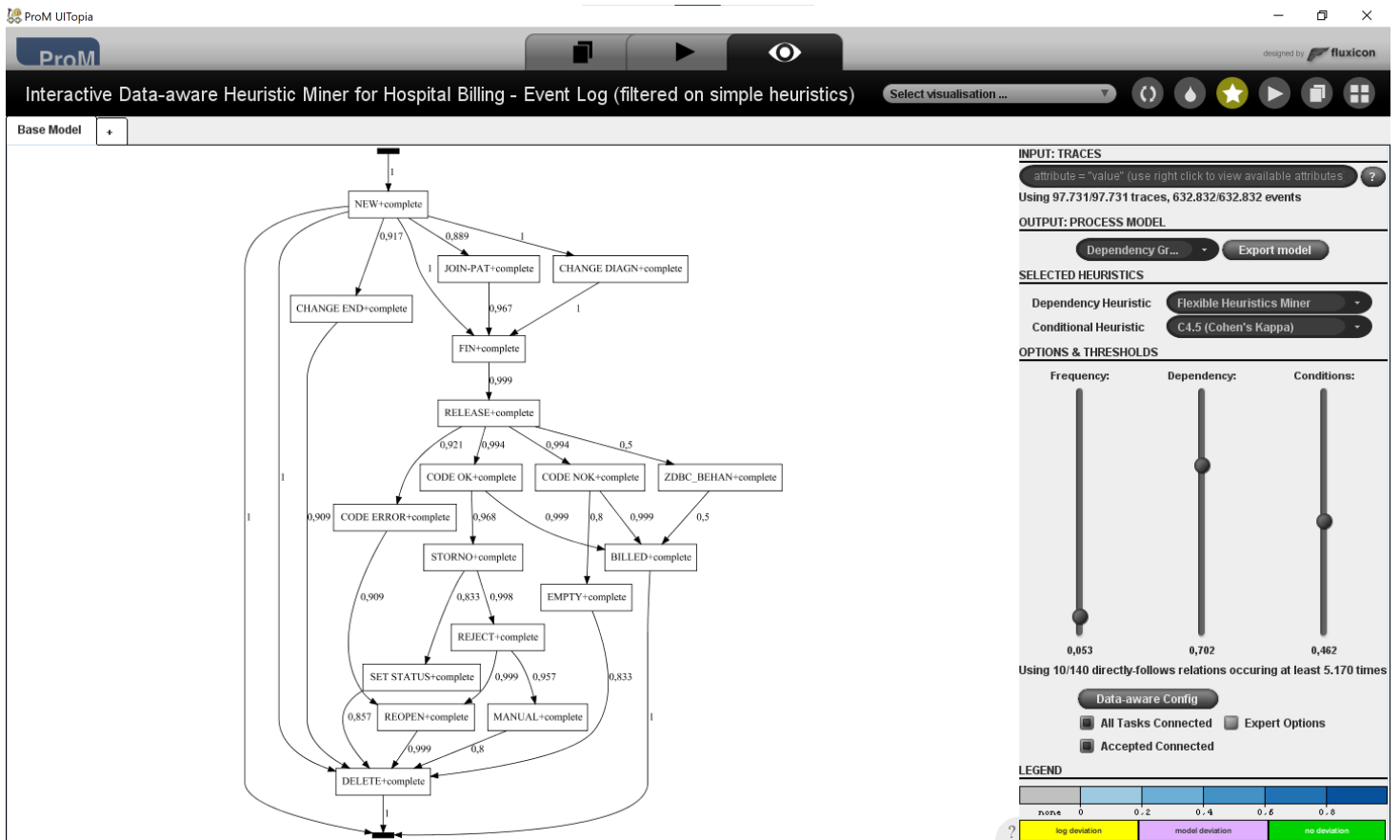
*Discovered Model with Optimal Alignments:*



The relative Dependency Graph for this discovered model is obtained.

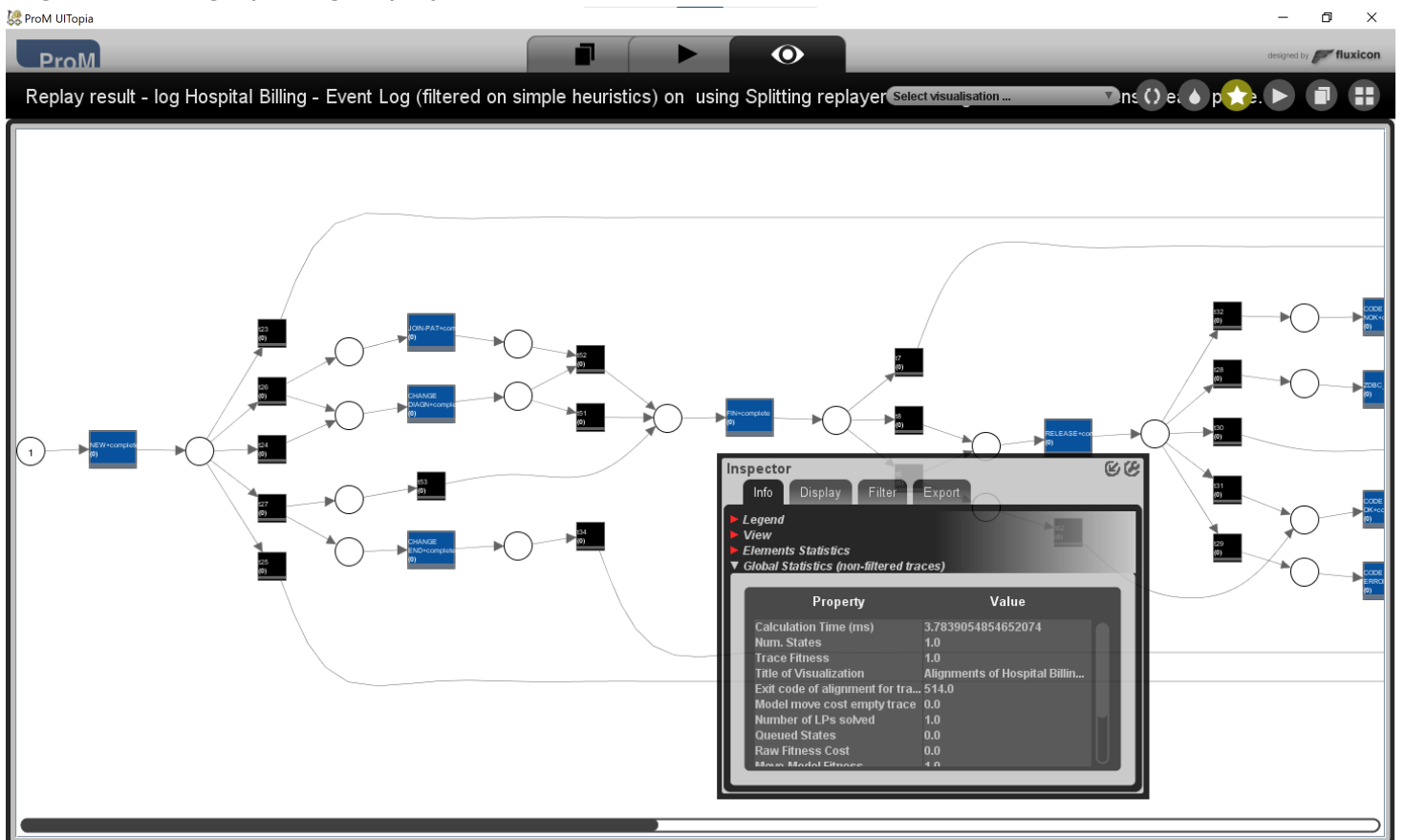
*Dependency Graph:*



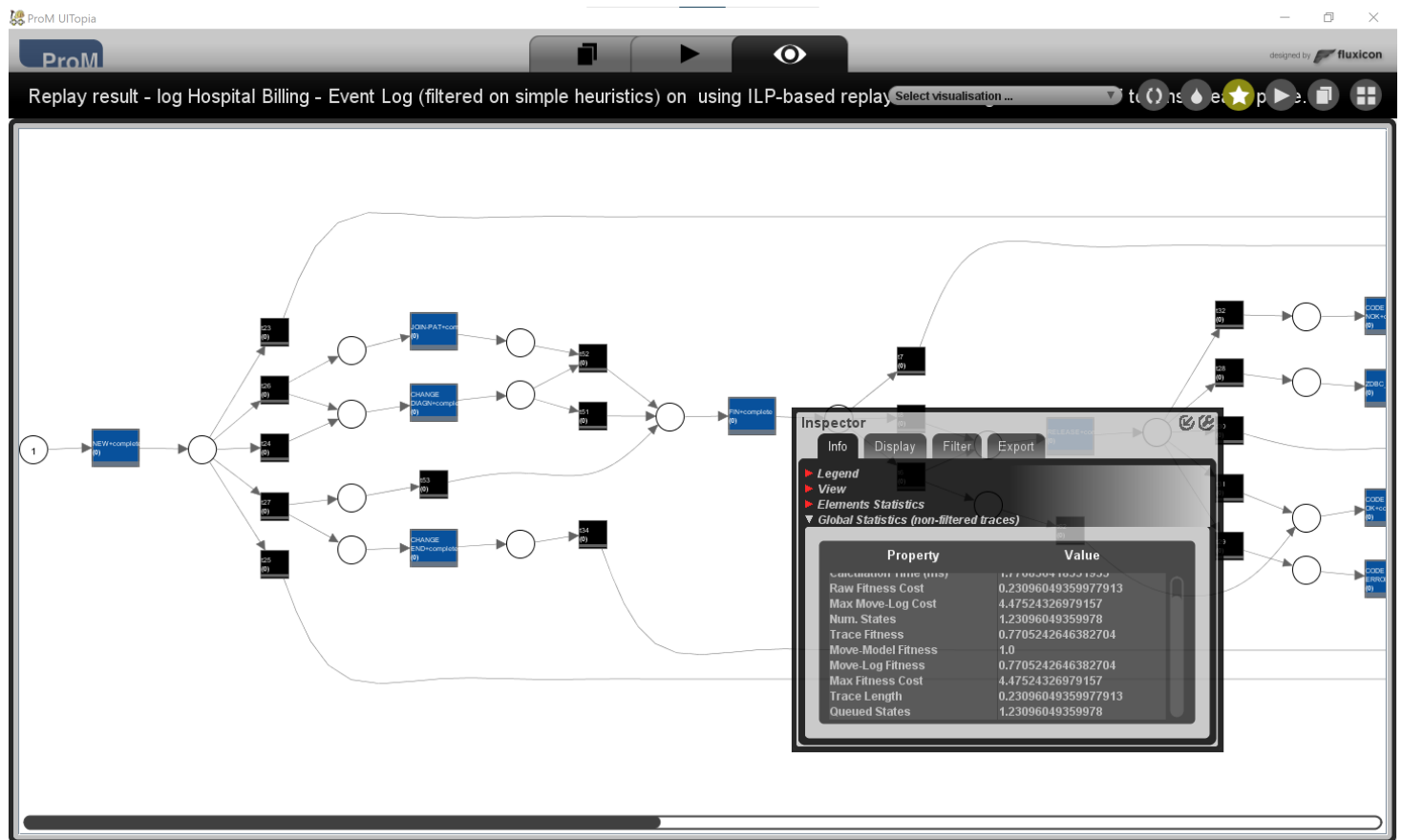


## Conformance Checking

### Alignment using Splitting Replayer:



## Alignment using ILP:



## ILP Alignments Sample Data:

ProM UI Topia

Replay result - log Hospital Billing - Event Log (filtered on simple heuristics) on using ILP-based replay

<type case id here>

Log-model Alignments

Case id(s)	#Num. Cases	#Is Alignment Reliable?	Calculation Time (ms)	Alignment
AYPA	21	No	3	0 events
BXAA	19	No	2	0 events
DPGD	18	No	2	0 events
ALKD	18	No	2	0 events
ALDC	17	No	2	0 events
AVMC	16	No		0 events

LEGEND

- Synchronous move (move log+model)
- Unobservable move (move model only)
- Skipped event class (move model only)
- Inserted event class (move log only)
- Replaced violation (move log+model)
- Swapped violation (move log+model)

STATS FROM RELIABLE ALIGNMENTS

Queued States	1.23096049359...
Raw Fitness Cost	0.23096049359...
Trace Fitness	0.77052426463...
Trace Length	0.23096049359...
Traversed Arcs	3.00339707973...

ALIGNMENT STATISTICS

Traversed Arcs

Average/case	<NaN>
Max.	<NaN>
Min.	<NaN>
Std. Deviation	0
#Cases with value 1.00	0

STATS INCLUDING UNRELIABLE ALIGNMENTS

#Cases replayed	97.731
#Synchronous ev.class (log+mo...	0
#Skipped ev.class	0
#Unobservable ev.class	0
#Inserted ev.class	22.572

In this second approach, we seen that we obtained a great fitness (in the first case, alignment of 1), but giving a look to alignments data, we can see that there are only *Swapped Violations*, which means that the given fitness **is not correct**.

# Social Network Analysis

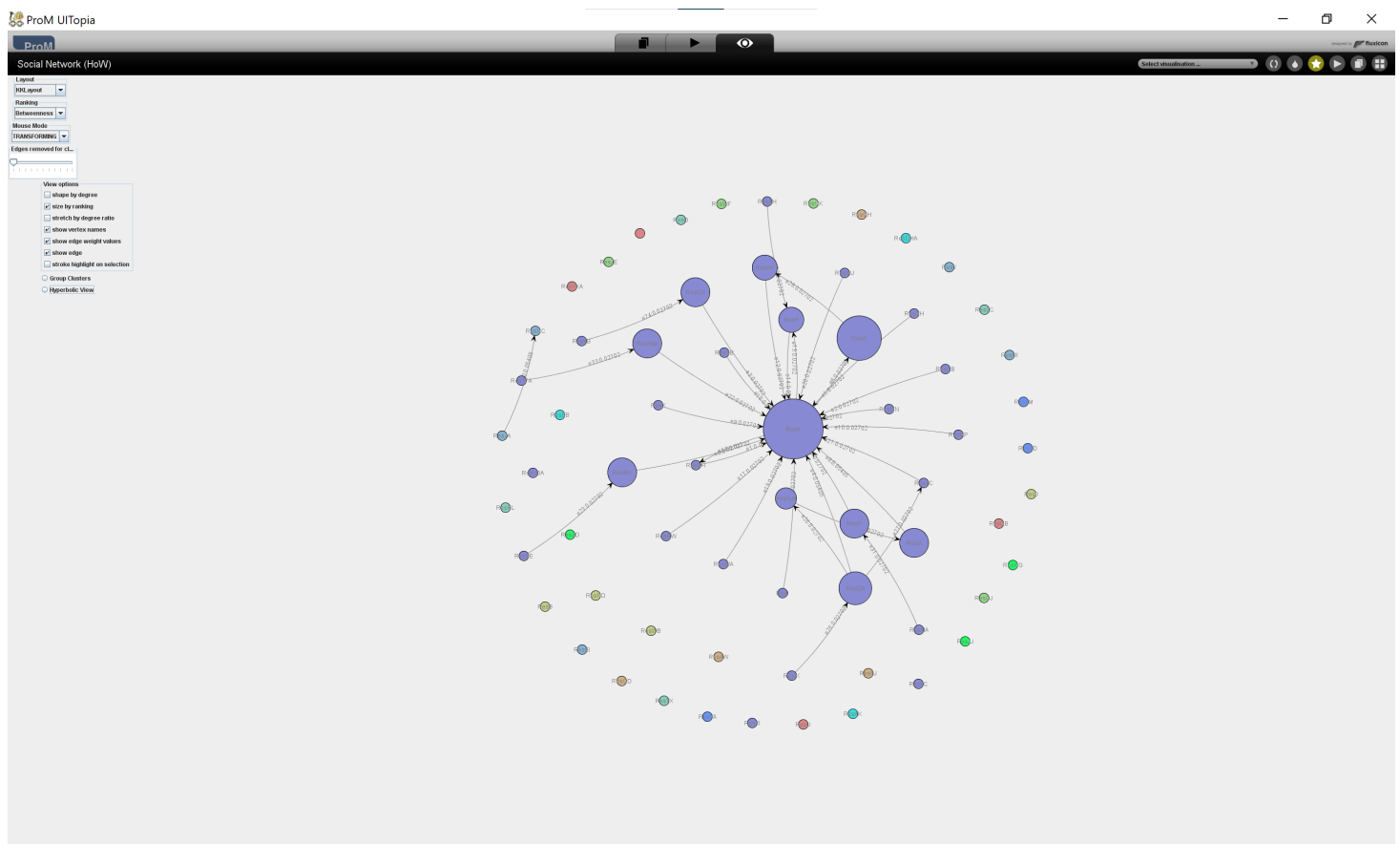
In order to do **Social Network Analysis**, data has been filtered for simplicity.

In fact, in a event log composed by **100.000 traces** and **451.359 events** was a little bit difficult to interpretate resources relationships.

For the purpose of Social Network Analysis, events log was filtered taking only **100 traces** randomly, from an action of ProM.

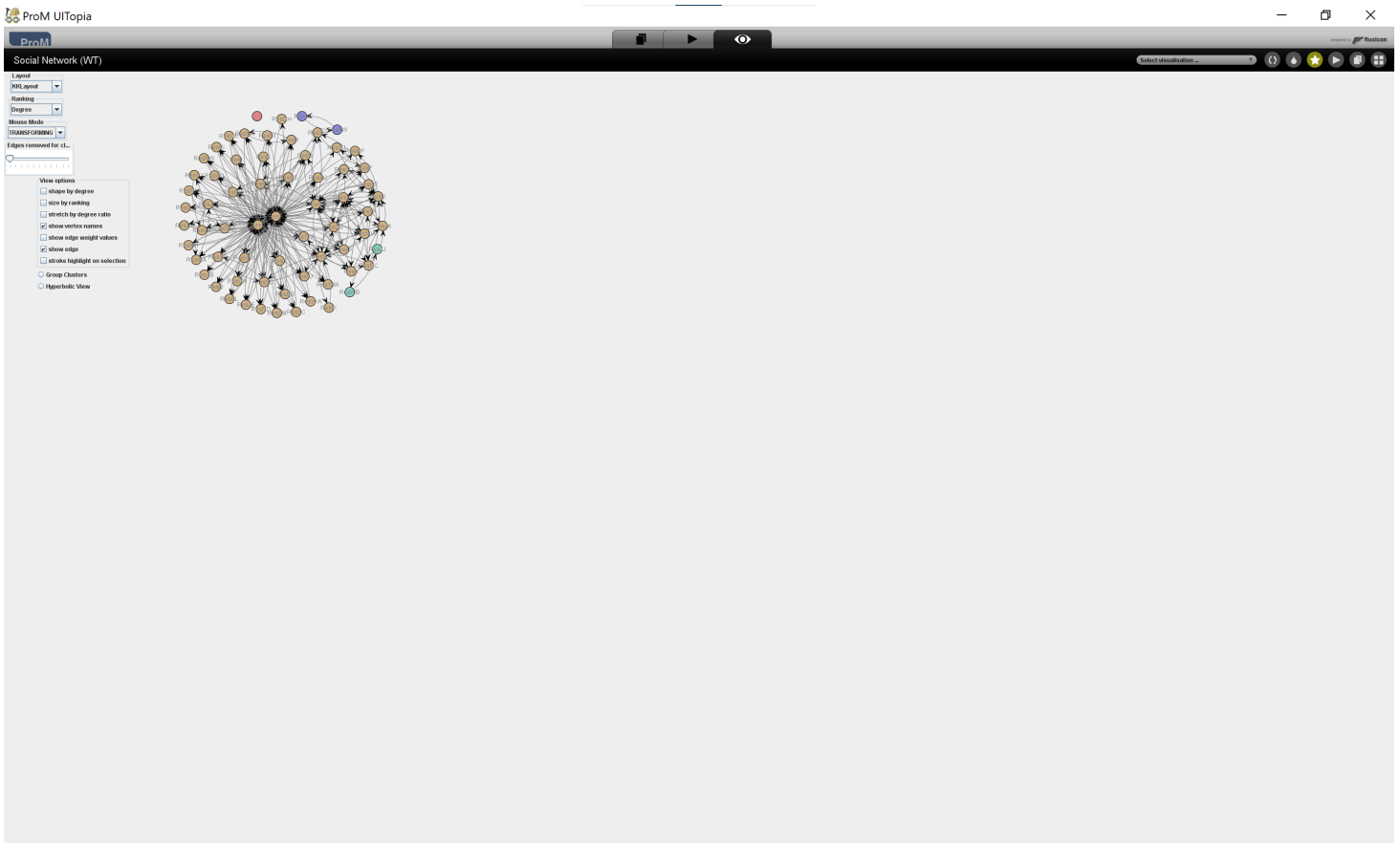
First discovered Social Network was the Handover of Work, and in the following screenshot, we can see relationships between resources, and the respective properties like betweenness, closeness, centrality.

*Handover of Work:*



Second discovered Social Network was the Working Together Relationship.

*Working Together Relationship:*



# Conclusions

In conclusion, I improved my capacity of understanding Logs and doing Process Mining over those. This project helps me to understand how Process Mining **works** applied to real events log and I explored the **power and functionalities** of analyzing logs to **reate Models, checking its Conformance** and doing **Social Network Analysis** on it.

I learned how a real log is formed and structured, which are its attributes and resources and which are the capabilities of *filtering data*.

I seen also the various **Process Discovery Algorithms** in practice, which are the model discovered **playing** with the **configuration**, their **conversion** to other model such as Petri Nets, and the relative **conformance checking** applied with **Alignments** between the model and the Log.

About **Process Discovery Algorithms**, i found out the differences between Alpha, Alpha+ and Heuristic.

I figured the Alpha and Alpha+ **were not suitable** for the chosen log, probably caused by *noise, incomplete log* and *loops*.

The different approaches of Heuristic Miner caught me, because of the amount of **different** possible **configurations** which lead to different models, based upon chosen **thresholds**.

About **Conformance Checking**, it helps me to understand the differences between discovered models, based on quality parameters such as **fitness**, **simplicity**, **precision** and **generalization**.