Replay a Log on Petri Net for Conformance Analysis Plug-in

Package: PNetReplayer, PNetAlignmentAnalysis

Arya Adriansyah

Last updated on 8 October 2012 (draft version)

Contents

1. Introduction 2
2. Requirements 2
3. Known Issue (!) 2
4. Example use of the plug-in 2
5. Result 7
    1. Model Projected with Alignments 8
    2. Project Alignments to Log 9
    3. Trace Alignment of Alignments 9
    4. xFrequent Movement Sets Mining for Deviation Analysis 10
    5. Synchronous Transitions Analysis 11
6. References 12

1. Introduction

This plug-in accept a Petri Net/Inhibitor/Reset/ResetInhibitor Net and an event log to create alignments between each trace in the log and the net [1]. As a side-product, fitness value between each trace and the net is calculated. Note that log may not be complete, i.e. it may be retrieved when process executions are not finished yet.

2. Requirements

To get the most benefit of this plugin, these requirements need to be satisfied:

1. The net:
    a. Has a non-empty initial marking, and
    b. Has a final marking. Without any final marking, it is assumed that all deadlock states/reachable states in the net are final markings.
2. The log has more than 0 (zero) number of traces

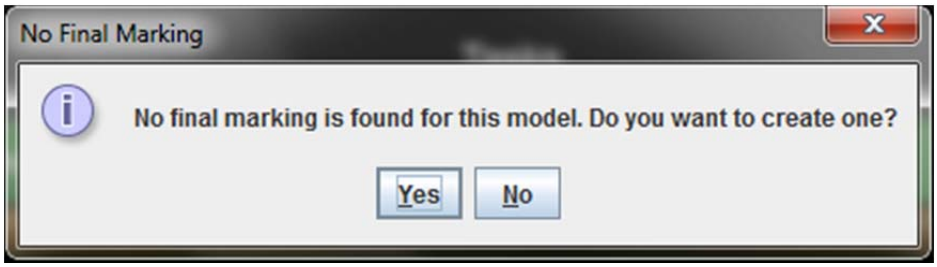
3. Known Issue (!)

ProM 6 "PNetReplayer" package is derived from the **obsolete**-and-will-be-removed "Replayer" package. Existence of both packages within one ProM installation may result in inconsistency problem. Hence, please remove the "Replayer" package using the ProM Package Manager and ensure that the "Replayer" package has been removed from .ProM folder (located in C:\Users\<user name>\.ProM\packages in Windows OS) before you use this plug-in.

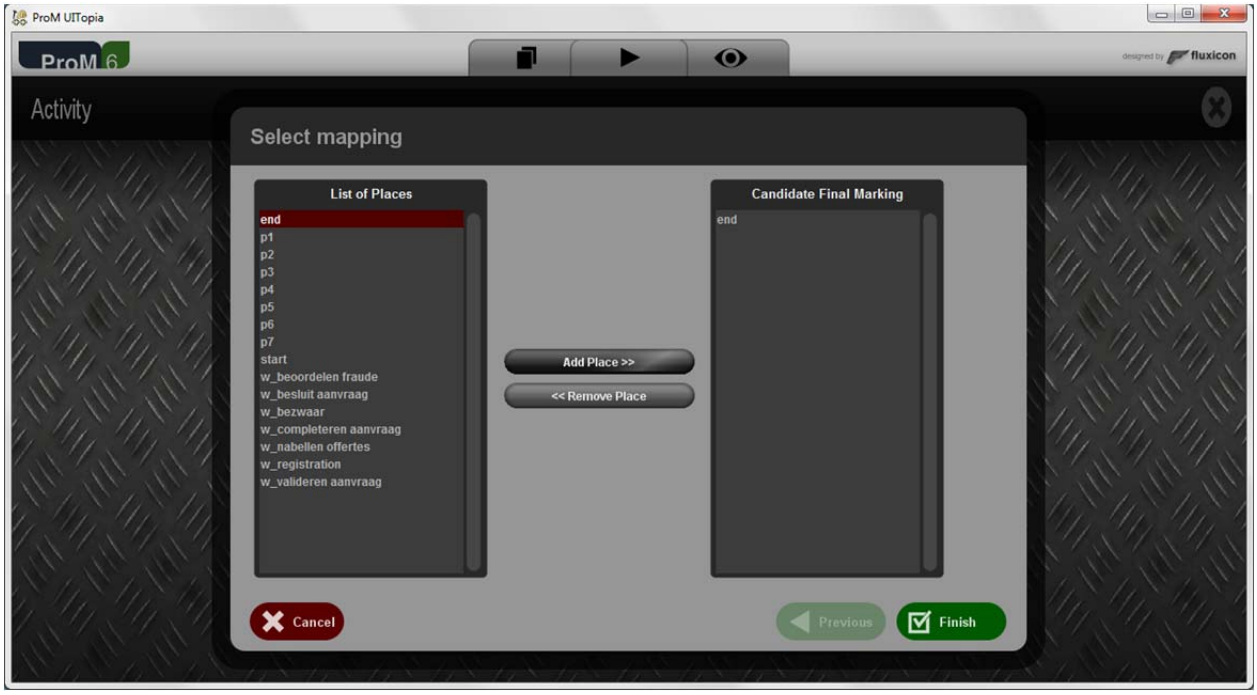
4. Example use of the plug-in

The plug-in requires a Petri Net/Inhibitor/Reset/ResetInhibitor Net and an event log. Therefore, we first need to load both of them (see figure below). Click the "Start" button

If the net does not have any final marking, (i.e. no FinalMarkingConnection object that connects the net to a marking object), a popup panel that asks whether you want to create one is shown.



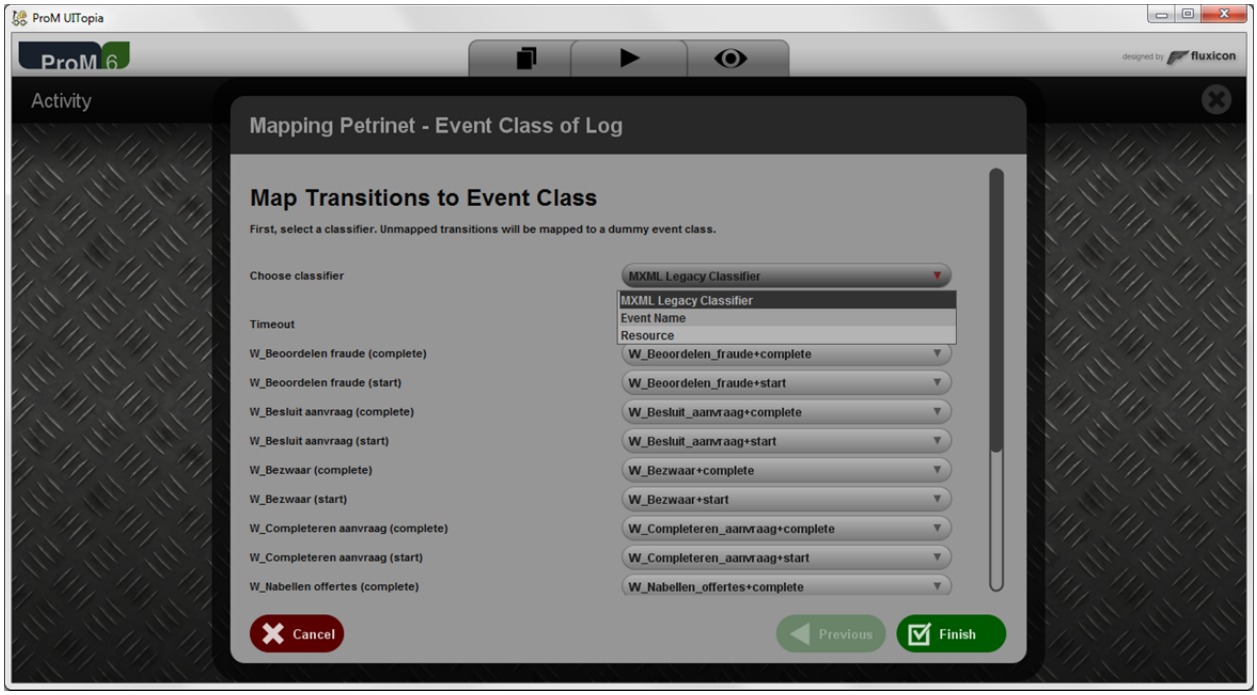
Answering yes to the question triggers the “Create Final Marking” plugin. This plug-in can also be executed separately from the “Replay a Log on Petri Net for Conformance Analysis” plug-in. Answering no to the question means that you assume that all deadlock/reachable states are final markings.



After finish creating a final marking, click “Finish”.

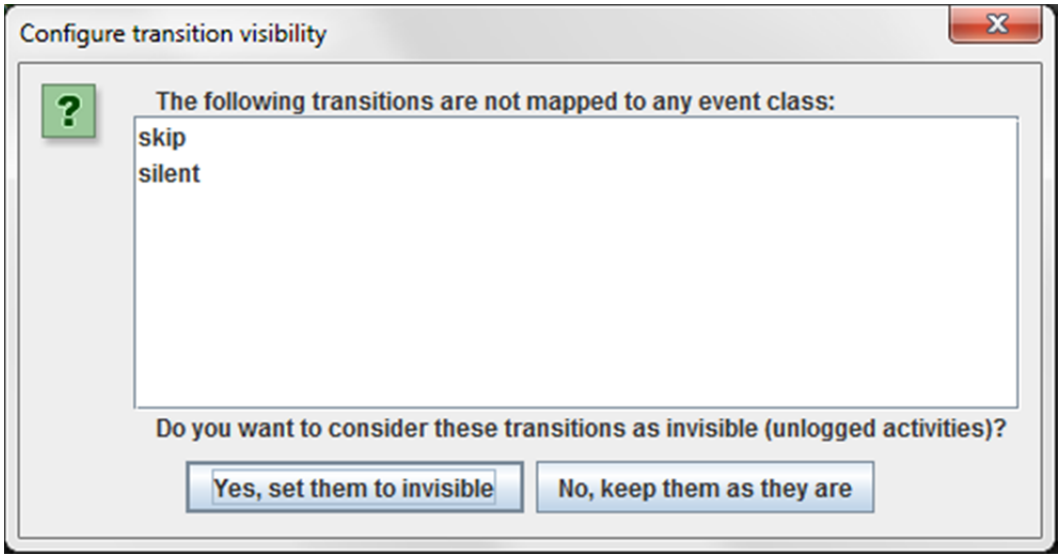
If the net and log is not mapped before (i.e. no EvClassLogPetrinetConnection object that connects the net and the log), a mapping panel will be shown (see below). Choose an appropriate classifier and map transitions in the left side to event class in the right side. Apart of standard classifiers, the plugin also identified possible classifiers from the provided log. Click “Finish” when you’re done.The mapping between transitions

(left) to event classes (right) is one-to-one, i.e. a transition can only be mapped to maximum one event class.

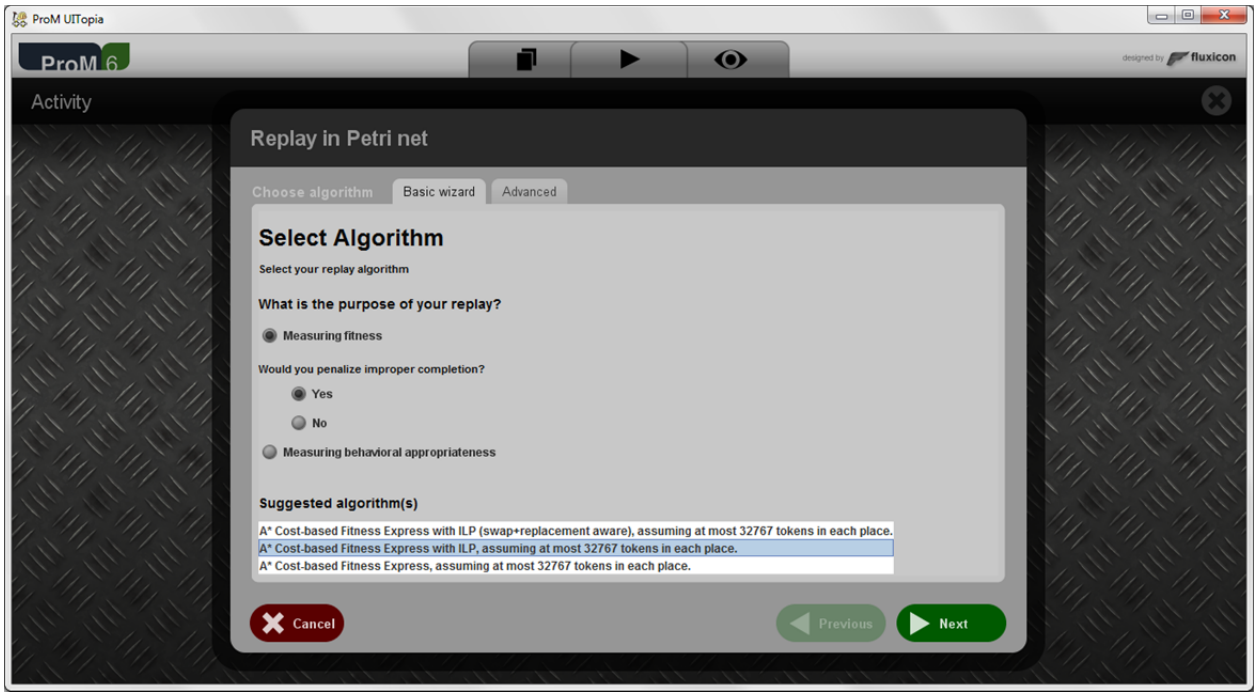


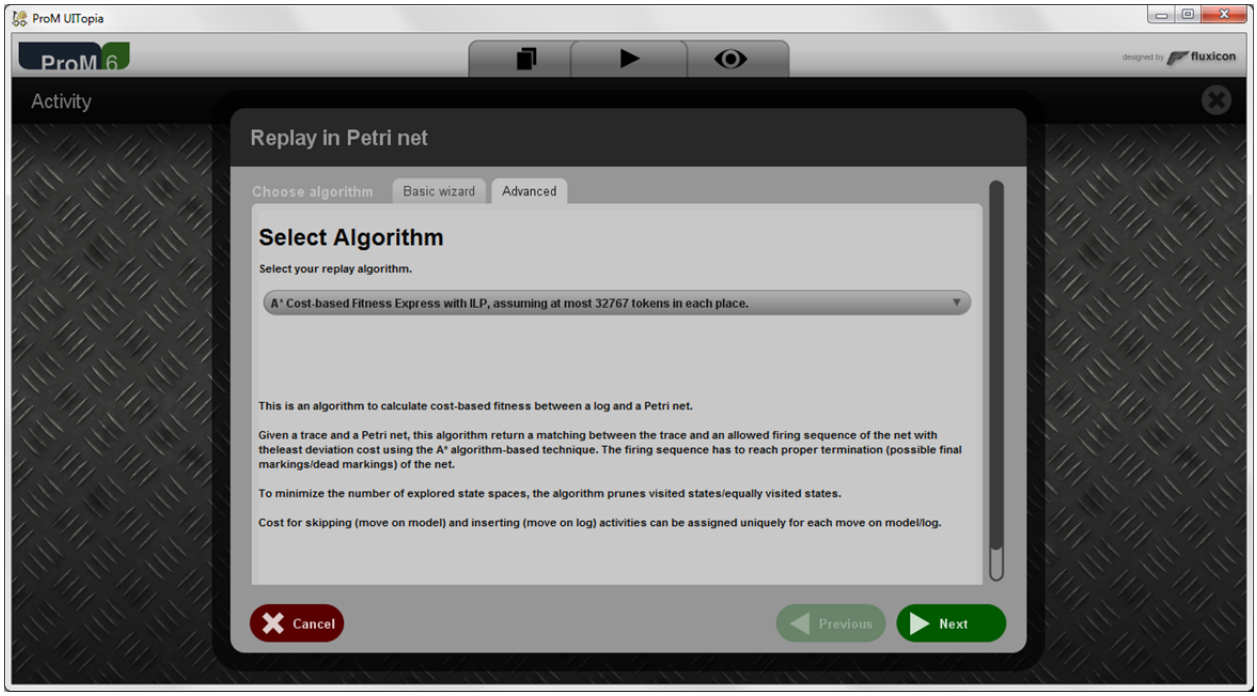
Note: The mapping panel does not check whether the mapping requirements for replay algorithms are satisfied. If there is an event class that is not mapped to any transition in the net and the replay algorithm require all event classes are mapped to transitions, exceptions may occur in the later stage of the replay.

If there are transitions that are not mapped to any event class, a popup is shown, asking whether such transitions should be marked as invisible transitions (transitions whose execution is unlogged) or not (transitions whose execution should be logged). Note the only difference between invisible transitions and transitions whose events are unlogged is their default cost value for deviating, i.e. 0 for invisible transitions and 1 for unlogged transitions. You can always change this later. Setting transitions to be invisible is done by calling another plugin “Invisible transitions”.

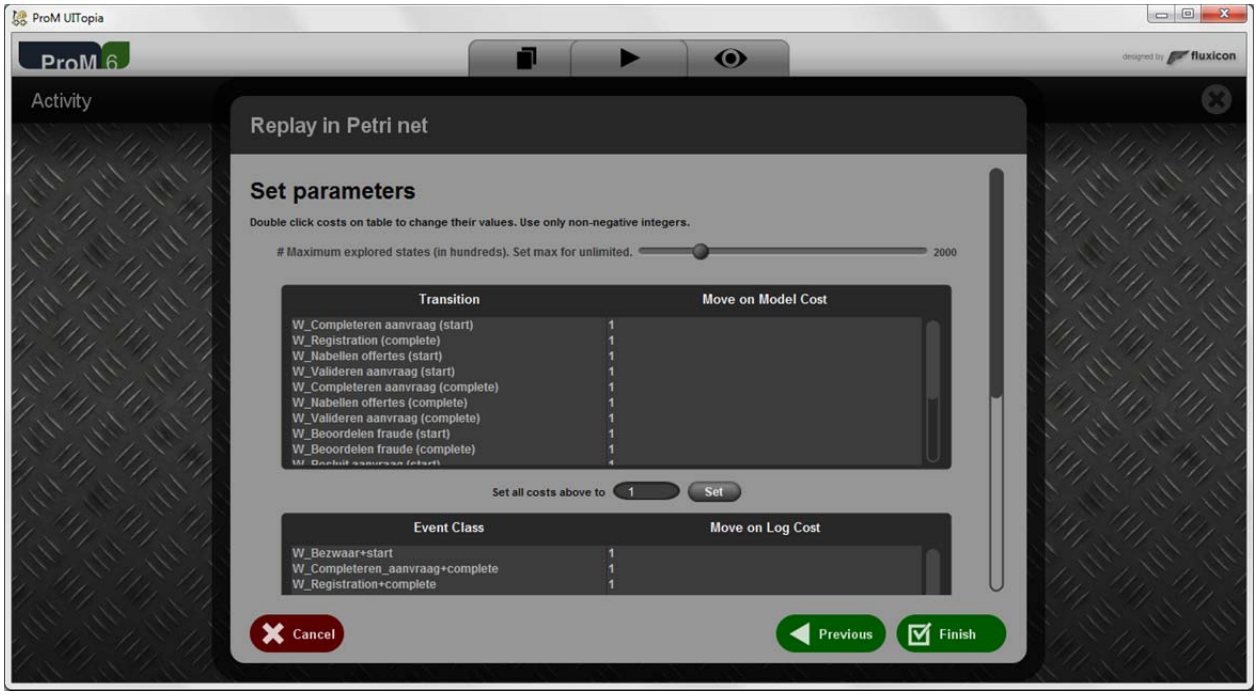


Choose a replay algorithm. By default, the fastest (and the most advanced) algorithm will be chosen for you. Only if you really know what you're up to, change the default settings of the algorithm or use other algorithms provided in the "Advanced" tab. A short explanation of any selected algorithm is also provided in the "Advanced" tab.



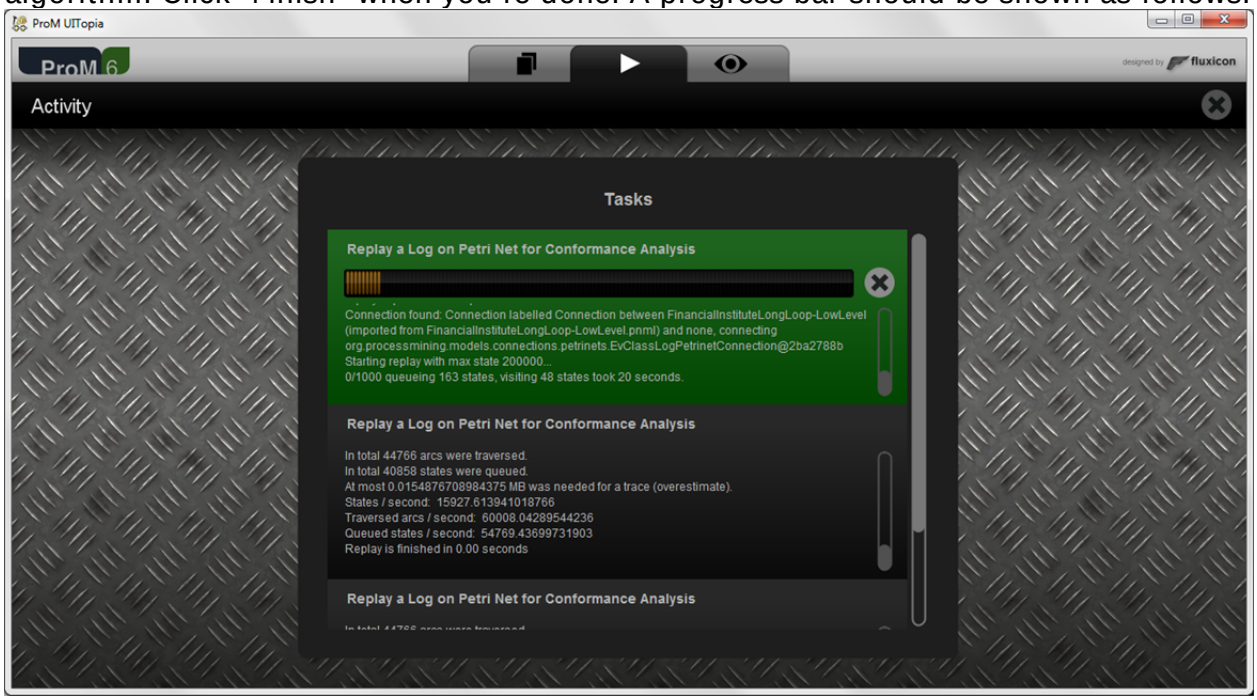


Assuming that you choose the default algorithm (A* Cost-based Fitness Express with ILP), you can configure the cost of violation and the limit of calculation that you allow to come up with reliable results (see below).



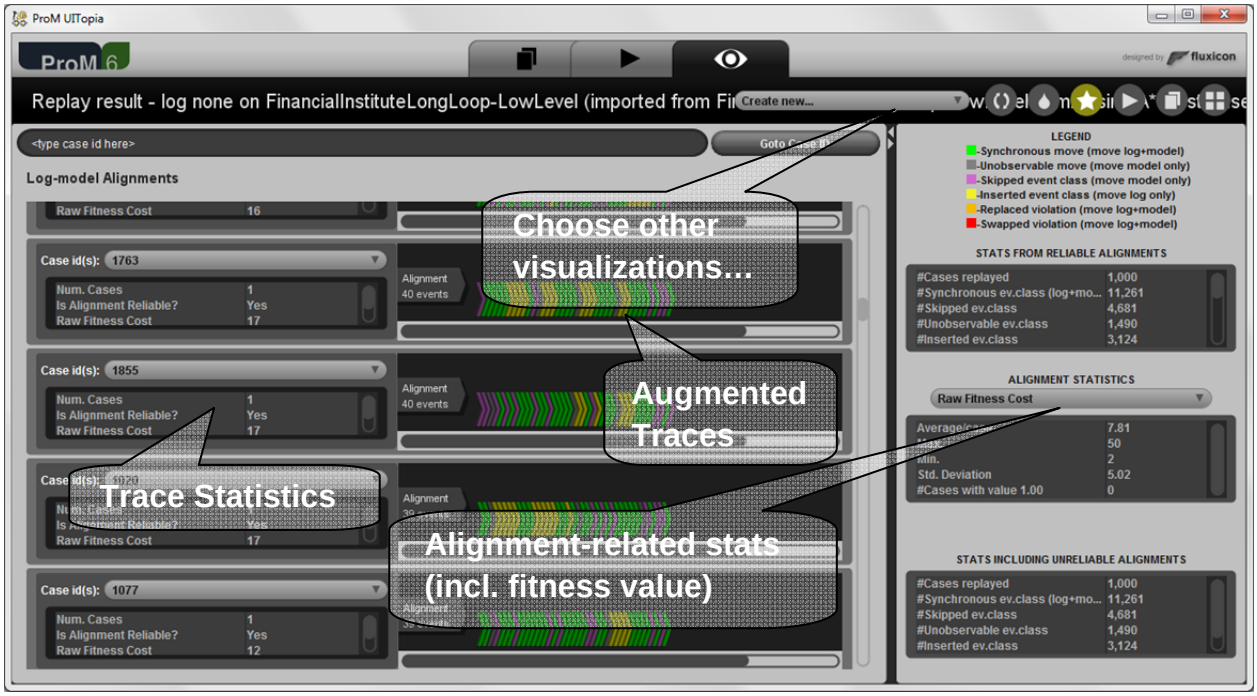
Setting the maximum explored states to maximum (unlimited) means that you allow the algorithm to explore up to $(2^{31} - 1)$ states to find an optimal alignment for each trace in the log. This may cause OutOfMemory exception in most personal computer after a long computation time (may take hours!) if the net and trace is really complex.

The limit of 200,000 states is typically good enough to deal with real-life logs and models with moderate complexity, using the A* Cost-based Fitness Express with ILP algorithm. Click “Finish” when you’re done. A progress bar should be shown as follows.



5. Result

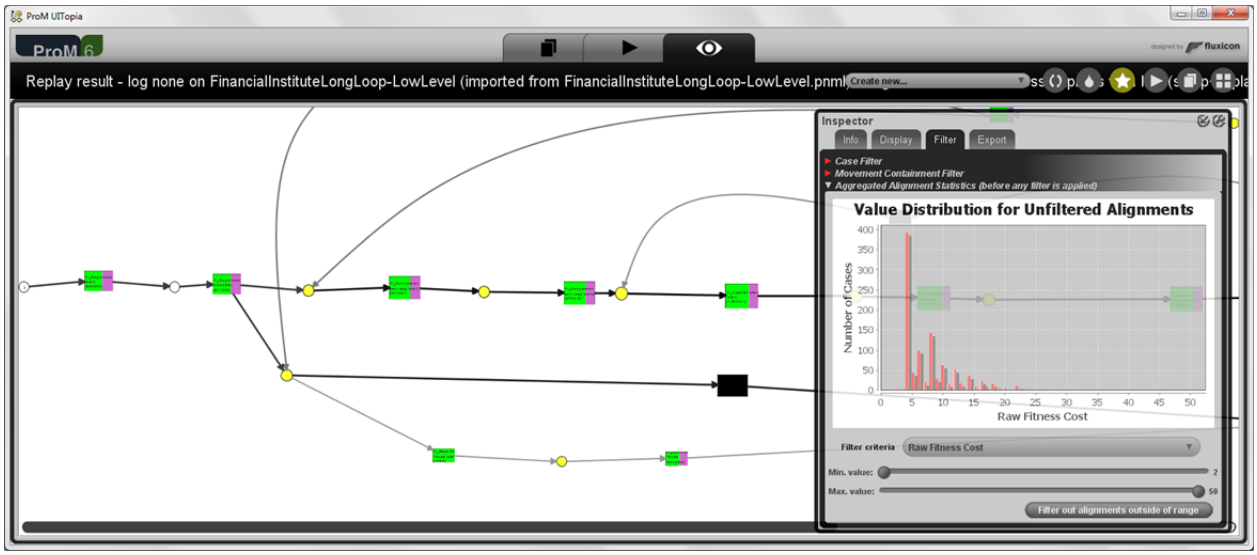
Result of the plug-in is a PNRepResult object that contains all alignments between the log and net. If you only installed the PNetReplayer package, the alignments are visualized with “Projects Alignments to Log” visualization, similar to the one shown below.



If you installed the PNetAlignmentAnalysis package, there are other visualizations to analyze conformance/performance from various perspectives. Readers are referred to [1] for a study case that shows the benefit of using various visualizations of alignments. To date, these are the list of available visualizations:

1. Model Projected with Alignments

This visualization shows in the original model, which tasks are often skipped, and when extra activities that should not be performed according to the model is performed in reality.



Use this visualization to get an overview of where deviations are located, separate severely deviating cases from non-deviating ones, and analyze deviations that often co-occur together. For example, you can export cases where deviations are severe as a new log by first filter out non-severely deviating cases using the filter tab, and then export shown cases as a new Log using the export tab.

2. Project Alignments to Log

This is the standard visualization of alignments. In this perspective, you can inspect per trace, which deviations occurred, and what are their possible causes. Use the “Goto Case ID” button to jump into your case of interest.

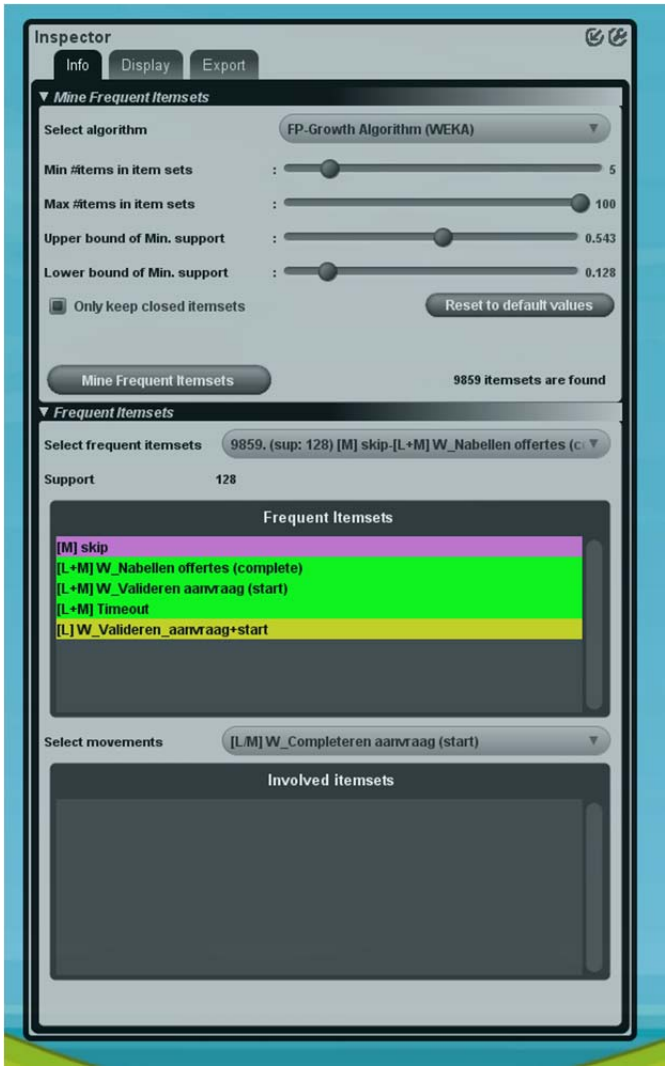
3. Trace Alignment of Alignments

This visualization uses the trace alignment technique of [1]. Use this visualization to get an overview of the context in which deviations occur.

4. xFrequent Movement Sets Mining for Deviation Analysis

This visualization use Weka's data mining library to mine frequently occurring deviations from all alignments. Typical analysis that you may obtain with this visualization is: "If task A is skipped, task B is also skipped", or "If event class C occurs (although it should not according to the model), task D is skipped".

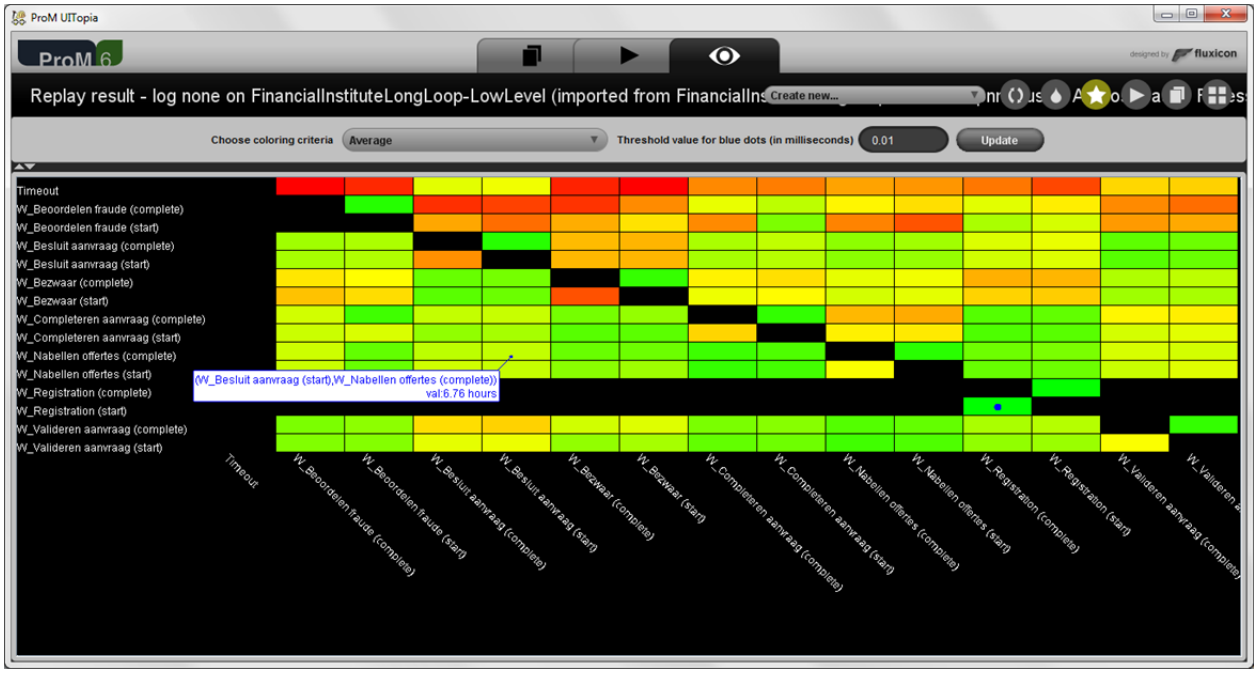
Use the "Info" tab, the Mine Frequent Itemsets pane to mine frequently occurring patterns (shown below). Then, choose an itemset in the Frequent Itemsets panel to see its projection onto the process model. This way, you know where is the itemset.



The projection onto the model only shows the synchronous moves and move on model (skipped tasks).

5. Synchronous Transitions Analysis

This visualization is useful to identify time elapsed between tasks and tasks which are synchronous/batched. Given a collection of alignments, this visualization only take **the first occurrences** of all tasks (only their synchronous moves) in each alignment, and measure time between the moment a task occurs and the moment others occur. The color of cells indicate the value of the measured time compared to other (red = high, green = low). Read the matrix from absis then ordinate, e.g. in the example below, the time elapsed between W_besluit aanvraag (start) and W_Nabellen offertes (complete) is 6.76 hours.



Set a dot threshold value from the textfield on the top right of the screen. If the measured time of a cell in the visualization goes below the threshold, a blue dot is shown on top of the cell. Such blue dots are useful to identify automated tasks/tasks that always occur within short period.

6. References

[1] W. van der Aalst, A. Adriansyah and B. van Dongen, "Replaying history on process models for conformance checking and performance analysis," *WIREs Data Mining and Knowledge Discovery,* vol. 2, no. 2, pp. 182-192, 2012.

[2] A. Adriansyah and J. Buijs, "Mining Process Performance from Event Logs: The BPI Challenge 2012 Case Study," BPM Center Report BPM-12-15, BPMcenter.org, 2012.

[3] J. Bose and W. van der Aalst, "Process diagnostics using trace alignment : opportunities, issues, and challenges," *Information Systems,* vol. 37, no. 2, pp. 117-141, 2012.