Homework 2

Miriam Wagner
373045

July 3, 2018

Contents

1 Process Models **1**
1.1 Exploring process . . . 1
1.2 Application data set . . . 1
1.2.1 General details of the data set . . . 2
1.2.2 Discover and evaluate a model of the application lifecycle . . . 2
1.3 Proposal data . . . 4
1.3.1 General Details of the data set . . . 4
1.3.2 Discover and evaluate a model of the proposal lifecycle . . . 5
1.3.3 Analyzing the explored model . . . 7
1.4 Combinded Model . . . 7
1.4.1 Endstate APP_Approved . . . 7
1.4.2 Endstate APP_Cancelled . . . 8
1.4.3 Endstate APP_Rejected . . . 8
1.4.4 All 3 Models . . . 8
1.5 C-net of the proposal process . . . 9
1.5.1 Analysis of the C-net . . . 9
1.6 Own Petri net of the proposal process . . . 10
1.7 Analysis of the performance of Application and work process . . . 10

Introduction

1 Process Models

1.1 Exploring process

For exploring a good model of a data set I first filtered the data with the (**Filter log on event attribute names**) tool for extracting the required. Then I used **Interactive data heuristic miner** and **Inductive Miner** on the filtered data set to discover different models. After comparing the outcomes I decided to concentrate on **Interactive data heuristic miner** and their directly followed graphs and petri nets for understanding the lifecycle. I think the directly followed graph is the best, but for checking the model I also checked the petri net with corresponding frequency filtering and basis configuration. The conformance checking where done with **Replay a log on Petri Net for conformance analysis** tool on the petri net and the filtered data. For the precision check I applied the **Multi-perspective Process Explorer** tool on the petri net and the filtered data and chose "show precision mode" in the tool with basic configuration.

1.2 Application data set

Just the events beginning with "App_..." are required. The resulting data set is saved as "Filtered App".

1.2.1 General details of the data set

The data set is collected between 1st of 0ct 2011 (saturday), 00:38:44 and 14th of Mar 2012 (Wednesday), 15:33:57. It contains 13087 cases with 60849 executed events.
10 different events appear: "App_Fully_Submission" (21.507%), "App_Incomplete_Submission" (21.507%), ("App_Rejection" (12.547%), ("App_Pre_Acceptation" (12.107%), ("App_Acceptation" (8.403%), ("App_Finalization" (8.242%), ("App_Cancellation" (4.613%), ("App_Initiation" (3.691%), ("App_Approving" (3.691%) and ("App_Registration" (3.691%). Just "APP_Fully_Submission" appears to be a start event. However there are 8 end events possible: "App_Rejection" (58.34%), "App_Cancellation" (21.449%), "App_Initiation" (8.573%), "App_Registration" (6.014%), "App_Approving" (2.575%), "App_Finalization" (2.499%), "App_Pre_Acceptation" (0.527%) and "App_Acceptation" (0.023%).
A trace contains maximal 8 different events and minimal 3. The mean is 4.65. In total there are 17 different variants of traces.

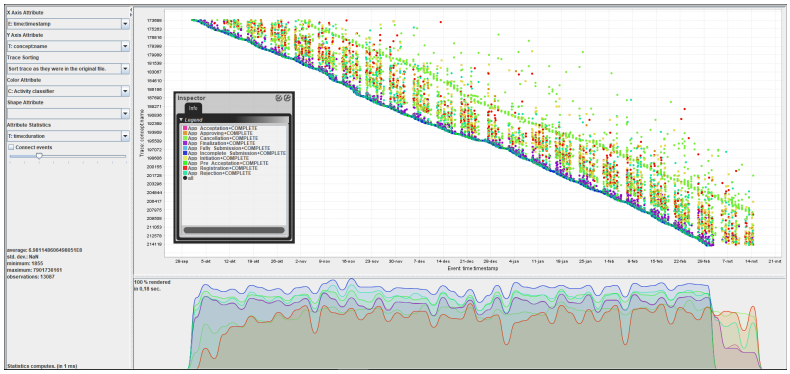
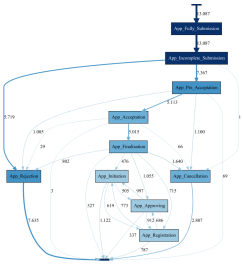


Figure 1: Dotted chart showing the time of events

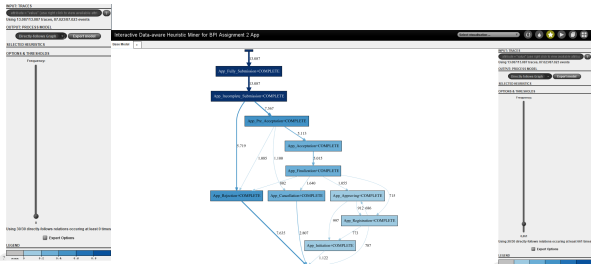
In figure 1 the dotted chart can be seen. Having a closer look at this chart you see gaps, which are always on a sunday. Those gaps do not appear for "APP_Pre_Acceptation" and "APP_Incomplete_Submission". Furthermore is in the left below corner to see the average duration of a case, 8 days 1 hours 55 minutes and 14.86 seconds, and the maximum duration, 91 days 10 hours 55 minutes and 36.16 seconds. Both are given in milliseconds.

1.2.2 Discover and evaluate a model of the application lifecycle

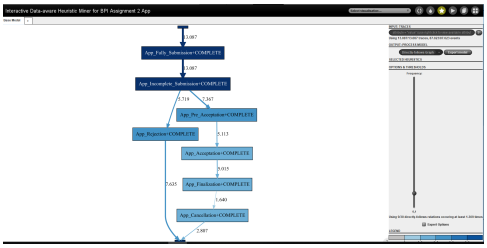
To find an accurate model I tried different frequency filters. First I chose the frequency 0.1 and had a look at the directly followed graph. I also checked 0.2 and 0.51. For comparsion in the end I had also a look at the original directly followed graph found by the **Interactive data heuristic miner**. In figure 2 the 4 considered directly followed graphs can be seen. My first choice was the directly followed graph with 0.1 as threhold for frequency, because it was a simple model that still tells us a lot about the main process and is not to specific (still has an acceptable generalization). Obviously the original graph does not fullfill the criterium of simplicity and also the graph with frequency 0.051 still looks not simple enough.
In the next step I checked the conformance of the corresponding petri nets (using basic configuration just changing the frequency filter), which I exported from the Interactive **Data-aware Heuristic Miner**, by combining the data and the petri net for the conformance checking with the **Replay a log on Petri Net for conformance analysis** tool.
The results are shown in figure 3. To see is that the model with 0.2 also has the same conformanc outcome as 0.1, what is not surprising since I have the same petri-net for the configuration I used. Based on the simplicity and generalization and the fact, that the fitness of 0.1 filtering is still not bad (88.20% fitness) I considered 0.051 and 0.1 for the precision check.
For the precision check I applied the **Multi-perspective Process Explorer** tool and chose "show precision mode" in the tool with simple configuration.
Having a look at the precision, figure 5, I could see, that the precision of 0.51 filtering is 100%, but of the 0.1 filtering just 88.1%. In combination with the results before, I came to the conclusion, that 0.1 filtering is not good enough as model and 0.051 would be good enough, but did not fullfill my simplicity criterium. Starting by this I again tried different frequency filters starting by 0.075 to find a model



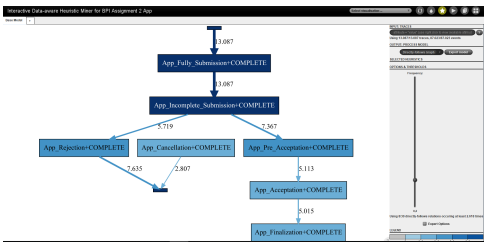
(a) Directly followed graph without frequency filtering



(b) Directly followed graph 0.051 frequency

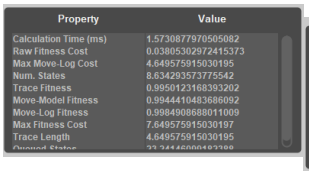


(c) Directly followed graph 0.1 frequency



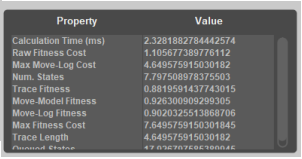
(d) Directly followed graph 0.2 frequency

Figure 2: Considered directly followed graphs



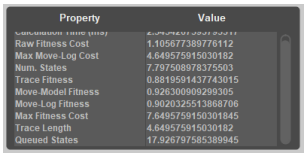
Property	Value
Calculation Time (ms)	1.5730877970505082
Raw Fitness Cost	0.038053029724153 73
Max Move-Log Cost	4.649575915030195
Num. States	8.634293573775542
Trace Fitness	0.9950123168393202
Move-Model Fitness	0.9944410483686092
Move-Log Fitness	0.9984908688011009
Max Fitness Cost	7.649575915030197
Trace Length	4.649575915030195

(a) Conformance for 0.051 frequency



Property	Value
Calculation Time (ms)	2.3281882784442574
Raw Fitness Cost	1.105677389776112
Max Move-Log Cost	4.649575915030182
Num. States	7.797508978375503
Trace Fitness	0.8819591437743015
Move-Model Fitness	0.926300909299305
Move-Log Fitness	0.9020325513868706
Max Fitness Cost	7.649575915030845
Trace Length	4.649575915030182

(b) Conformance 0.1



Property	Value
Raw Fitness Cost	1.105677389776112
Max Move-Log Cost	4.649575915030182
Num. States	7.797508978375503
Trace Fitness	0.8819591437743015
Move-Model Fitness	0.926300909299305
Move-Log Fitness	0.9020325513868706
Max Fitness Cost	7.649575915030845
Trace Length	4.649575915030182
Queued States	17.926797585389945

(c) Conformance for 0.2 frequency

Figure 3: Conformance checking

Avg activity precision	100%
# Moves Observed	144.789
# Moves Possible	144.789
Avg fitness	99,6%
% Violations	0,8%
% Event Violations	0,8%
% Data Violations	0%
# Correct Events	60.849
# Wrong Events	0
# Missing Events	498

(a) Precision for 0.051 as frequency

Avg activity precision	88,1%
# Moves Observed	97.074
# Moves Possible	110.161
Avg fitness	88,6%
% Violations	21,7%
% Event Violations	21,7%
% Data Violations	0%
# Correct Events	52.096
# Wrong Events	8.753
# Missing Events	5.717

(b) Precision for 0.1 as frequency

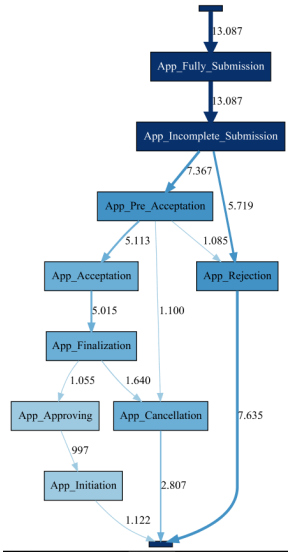
Figure 4: Precision checking

fullfilling both, a similar simplicity as the 0.1 frequency model, but a better conformance and precision than this simpel model. Already the frequency filtering 0.076 gives me the wished result.

Property	Value
Calculation Time (ms)	2.3191716971039815
Raw Fitness Cost	0.33223809887674755
Max Move-Log Cost	4.64957591503021
Num. States	8.79514021548101
Trace Fitness	0.9657921716262099
Move-Model Fitness	0.9944410483686092
Move-Log Fitness	0.9566108861210915
Max Fitness Cost	7.649575915030202
Trace Length	4.64957591503021
Queued States	22.414409267746 71

Avg activity precision	94,8%
# Moves Observed	133.936
# Moves Possible	141.304
Avg fitness	96,6%
% Violations	7%
% Event Violations	7%
% Data Violations	0%
# Correct Events	57.772
# Wrong Events	3.077
# Missing Events	1.271

(a) Conformance and Precision



(b) Directly followed graph

Figure 5: Frequency 0.076

This model has a good simplicity, but still has a path fitness of 96.58% and precision of 94.8%, so it is still pretty good. Overall I so decided to choose the 0.076 frequency model for the application's lifecycle. The process always starts with "App_Fully_Submission" and "App_Incomplete_Submission". After all 13087 cases completeted this two steps, 7367, roughly spoken 50%, accomplished "App_Pre_Acceptation". They now split again, where the most cases continue with App_Acceptation (5113 cases) and then App_Finalization (5015 cases). App_Pre_Acceptation is second most followed by App_Cancellation (1100 cases, which is a fifth of the amount of cases for App_Acceptation) and the last possibility is App_Rejection (1085 cases). App_Finalization also has two different followed actions, App_Approving (1055 cases) and App_Cancellation (1640 cases coming from App_Finalization). App_Cancellation thus is been performed in 2740 cases by the pathes to see, but in total by 2807. App_Approving is been followed by App_Initiation in 997 cases, but App_Initiation is executed 1122 times. App_Rejection has the predecessor actions as explained above in 6804 cases, but in total is been done in 7635 cases.
All the differences between incoming and outcoming cases can be explained by the filtering of the frequency. Because of this filtering not all possible pathes are visible, but just the 92.4% most common ones.

1.3 Proposal data

1.3.1 General Details of the data set

The data set is collected between 1st of 0ct 2011 (saturday), 10:44:40 and 14th of Mar 2012 (Wednesday), 15:50:59. Having a look at the visualization you can see that there a gaps in the workflow.
In figure 6 the dotted chart can be seen. Having a closer look at this gaps makes clear, that it is always a sunday. What does not show this behavior so clear is "P_Cancellation" and "P_Initiation". Furthermore is in the left below corner to see what is the average duration of a case, 65 days 20 hours 6 minutes and 25.88 seconds, and the maximum duration, 89 days 13 hours 10 minutes and 6.16 seconds. Both is given in milliseconds.
The data set has 7 events: "P_Initiation" (22.5%), "P_Sending" (22.5%), "P_Selection" (22.5%), "P_Cancellation" (11.698%), "P_Returning" (11.055%), "P_Acceptation"(7.179%) and "P_Rejection" (2.567%), where the percentage is the relative occurence. In total this data set contains 13087 cases with in total 31244 events. It always stars with "P_Selection" and has 5 different end events: "P_Acceptation" (44.726%), "P_Cancellation"(32,702%), "P_Rejection" (15.992%), "P_Sending" (4.806%) and "P_Returning" (1.775%).

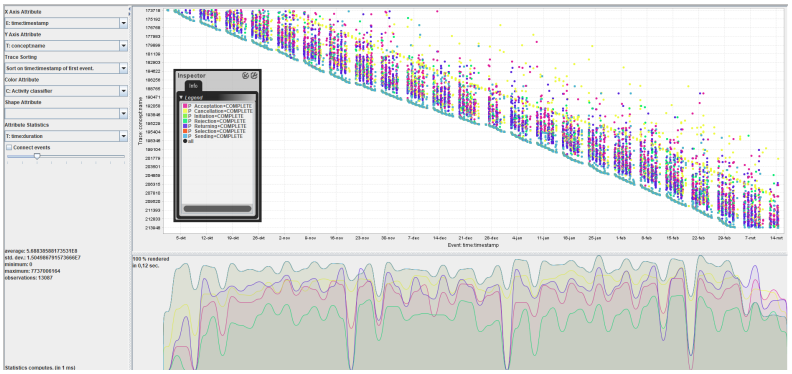
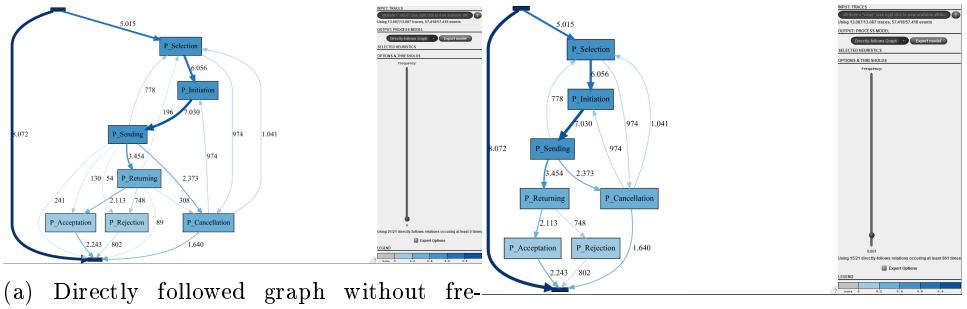


Figure 6: Dotted chart showing the time of events

There are 169 different variants of traces.
Maximal 30 events are executed in a class of cases and minimal 0, what is a hint, that there are cases ending directly. The mean of events per class is 2.387.

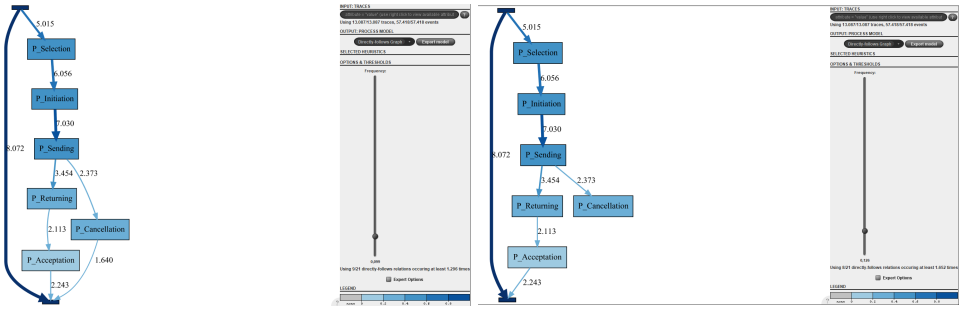
1.3.2 Discover and evaluate a model of the proposal lifecycle

Applying the steps on the proposal data set it gave me 4 models I wanted to have a closer look at.



(a) Directly followed graph without frequency filtering

(b) Directly followed graph 0.051 frequency



(c) Directly followed graph 0.1 frequency

(d) Directly followed graph 0.126 frequency

Figure 7: Considered directly followed graphs

Because of simplicity reasons I piced out of those, figure 7, just the 0.051 and 0.1 filtered and checked the conformance and precision for them.
The conformance and precision outcomes, to see in figure 8, told me, that the 0.1 frequency model (fitness 91.55%, precision 80%) has not a good enough precision. And also the precision of the 0.051 filtering is not really high. So I searched for an other model, which has a similar simplicity as the 0.1 and 0.051 filtered models and has maybe a better performance. The models fullfilling the simplicity criterium had 0.08 or 0.025 frequency filtered.
After having a look at all results in 9 and comparing it with earlier results I chose the model with 0.025. I decided, that the precision is still comparable and better than for 0.051, but it is still a simple enough model to understand the main traces. Choosing a lower frequency threshold made the model to

Property	Value
Raw Fitness Cost	0.07778711698632224
Max Move-Log Cost	2.3874073508061433
Num. States	6.670130664017713
Trace Fitness	0.9882694969197188
Move-Model Fitness	0.9937748638349697
Move-Log Fitness	0.9959403350088936
Max Fitness Cost	2.3874073508061433
Trace Length	2.3874073508061433
Queued States	16.723542446702886

(a) Conformance for 0.051 frequency

Property	Value
Raw Fitness Cost	0.7597615954764291
Max Move-Log Cost	2.387407350806137
Num. States	6.684114006265772
Trace Fitness	0.9154572453183918
Move-Model Fitness	0.9867960571559562
Move-Log Fitness	0.9298519372518611
Max Fitness Cost	2.387407350806137
Trace Length	2.387407350806137
Queued States	12.33177962863913

(b) Conformance 0.1

Avg activity precision	83,6%
# Moves Observed	103.538
# Moves Possible	123.899
Avg fitness	37,4%
% Violations	3,2%
% Event Violations	3,2%
% Data Violations	0%
# Correct Events	30.740
# Wrong Events	504

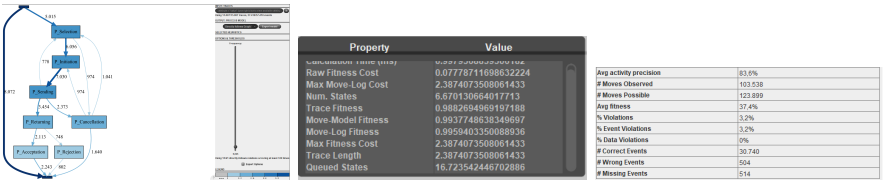
(c) Precision for 0.051 as frequency

Avg activity precision	80%
# Moves Observed	72.228
# Moves Possible	90.330
Avg fitness	30,3%
% Violations	31%
% Event Violations	31%
% Data Violations	0%
# Correct Events	22.119

(d) Precision for 0.1 as frequency

Figure 8: Conformance and precision checking

(a) 0.08 frequency



(b) 0.025 frequency

Figure 9: Directly followeg graphs, conformance and precision

complicated in my opinion.

1.3.3 Analyzing the explored model

This model has the conspicuousness, that 8072 cases directly end, but recheck it with the original graph this can be found, too. The other 5015 cases start with "P_Selection", then the most cases perform the "P_Initiation" action (6056 cases), while in 974 cases "P_Selection" is followed by "P_Cancellation". "P_Cancellation" builds a loop with "P_Selection" (10141 cases go to "P_Selection"). "P_Initiation" is followed by "P_Sending" (7030 cases), which has 3 different following actions. In 2373 cases as next action "P_Cancellation" is performed, so is "P_Cancellation" in total 3347 times followed like to see in the model. "P_Cancellation" is in 1640 cases the last action or it leads to "P_Initiation" (974 times) or "P_Selection" (1041 times). So it is executed in total 3655 times. "P_Sending" has also as followed action 778 times "P_Selection" and in the highest amount of times it is followed by "P_Returning" (3454 times). "P_Returning" now lead of to "P_Acceptation" (2113 cases) or "P_Rejection" (748 cases). In total "P_Acceptation" happens in 2243 cases and "P_Rejection" in 802 cases. Like in the aplication model the differences of cases are a result of the filtering.
What is special to see is the loop behavior containing "P_Selection", "P_Initiation" and "P_Sending" as loop, or "P_Selection" and "P_Cancellation", or "P_Selection", "P_Cancellation", "P_Initiation" and "P_Sending" in different orders.

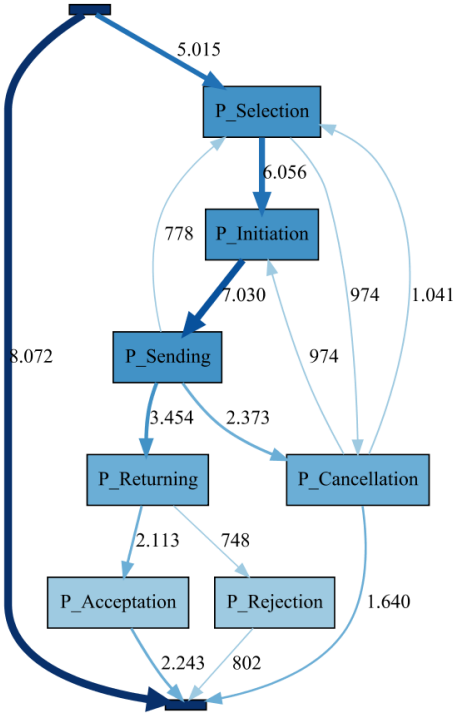


Figure 10: 0.025 filtered frequency

1.4 Combinded Model

For combinded Models I first filtered the data to have a dataset with all proposal data combined with the application data. This data I filtered with Heuristic filter (all configurations to 100% and just deciding what the endstate is) with the outcomes/endstates "APP_rejected" or "APP_cancelled" or "APP_approved". I saved them under the names "Filtered P App with approv", "Filtered P App with canc" and "Filtered P App with rej".

1.4.1 Endstate APP_Approved

Like for approved and proposal I first checked different frequency filter setting to have a first idea, which models fullfill simplicity. Then for every chosen frequency in begin I checked conformance and precision.

	Frequency			
	0	0.1	0.2	0.3
Simplicity	-	-	+	++
Fitness	99.84	99.47	93.93	93.93
Precision	93.3	94.5	91.7	94.8

Figure 11: Results for approved as endstate

Based on the results, 11, I chose the model with 0.3 filtering. This one has a high simplicity, but still has surprisingly good results.

	Frequency		
	0	0.049	0.1
Simplicity	-	+	++
Fitness	99.96	99.39	96.84
Precision	92.5	91.8	94.6

Figure 12: Results for cancelled as endstate

1.4.2 Endstate APP_Cancelled

Like for APP_Approved I checked different configurations and based on the results, 12, I decided to pick 0.1 filtered frequency model. The fitness and precision is still higher than 90%, but it is also the most simpel model.

1.4.3 Endstate APP_Rejected

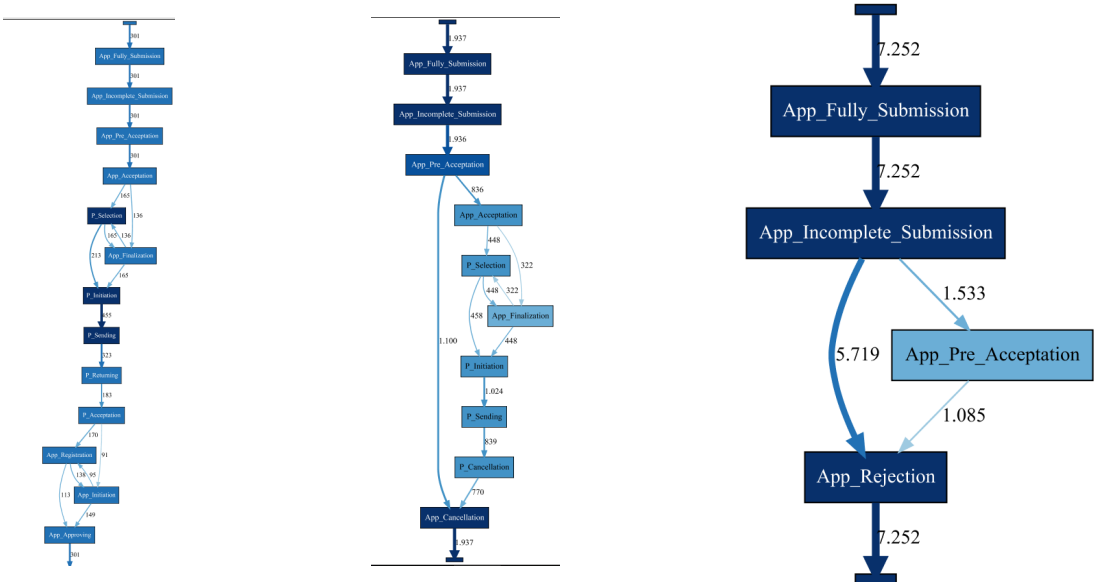
The last analysis is of the models ending in APP_Rejected. In the first step I checked different frequency filters and decided based on simplicity and traceability I had a closer look at 0, 0.025 and 0.1.

	Frequency		
	0	0.025	0.1
Simplicity	-	+	+++
Fitness	1.00	99.60	96.91
Precision	99.3	98.7	100

Figure 13: Results for cancelled as endstate

Based on 13 I chose 0.1 filtered frequency model as the best. It is really easy to follow and has a good fitness.

1.4.4 All 3 Models

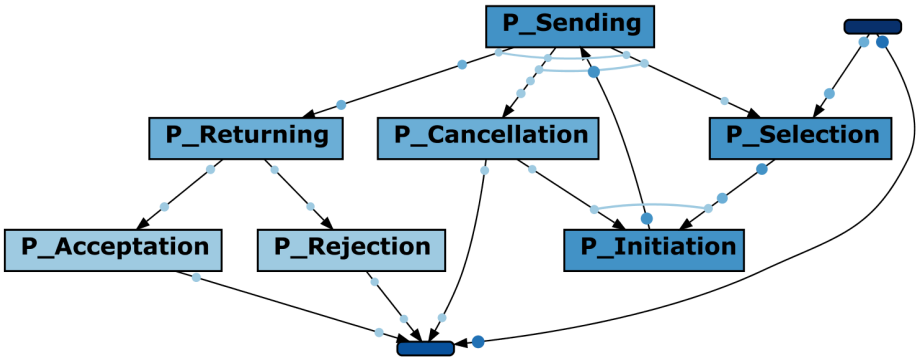


(a) Endstate APP_Approved (b) Endstate APP_Cancelled (c) Endstate APP_Rejected

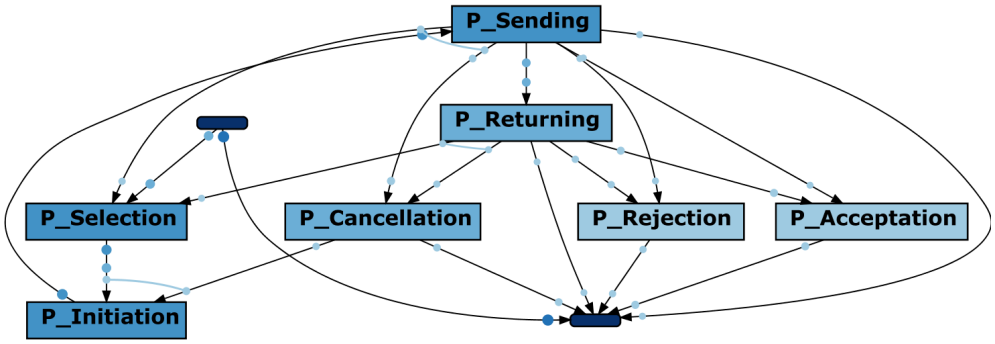
Figure 14: Models for the different endstates

1.5 C-net of the proposal process

Based on the results of my analysis of the model before I chose the same frequency filtering for the C-net of the proposal process.



(a) C-net with 0.025 filtered frequency



(b) C-net without filtered frequency

Figure 15: C-Nets of the proposal process

In 15 I show two C-nets of the proposal process for comparsion. 15a is the one I picked and I will discuss it later in detail. In 15b the original C-net of the process is to see and obviously it is much more complicated and not so intuitive than the filtered one.

1.5.1 Analysis of the C-net

For simplicity reasons I will not write "P_" as prefix of every activity. The first thing I did is having a look at the maximal number of bindings. This is for sending with 6 possible bindings. The input is always from initiation, but there are 4 different outputs possible. Having a look at all possible traces and the corresponding procedures. Just checking the possible traces shows you, that there are infinite many possible traces, because of a loop between sending and initiation.

- Done
- Selection →Initiation →Sending →
  - Returning →
    - Acceptation →Done
    - Rejection →Done
  - ((Cancellation →Selection) or (Selection →Cancellation)) →Initiation →Sending ...
  - Cancellation →Done

This pathes are also what you would sort of expect. The sending step has the biggest variance in the next step and we have a fixed prefix for the case, that we not directly end.

1.6 Own Petri net of the proposal process

1.7 Analysis of the performance of Application and work process

Questions

Conclusion