

Question 1

(a)

Import log-flat.xes to ProM. Click the eye symbol to view resource.



This results in the following view:



From it we can read

- the time period (1) covered by the event log, which is from 13.01.2010 to 03.01.2014
- the number of cases, events and activities of the log (2), being 4580, 21348 and 14 respectively (note that activities appear in ProM as event classes)

To gain more information on the activities we click the summary tab in the left which results in the following view:

The screenshot shows the 'Log Summary' interface. On the left, a sidebar contains 'Dashboard', 'Inspector', and 'Summary' (highlighted with a red arrow). The main area is divided into two sections. The top section, 'Event classes defined by Event Name', shows a table of 14 classes (labeled 3). The bottom section, 'Start events', shows a table of 6 classes (labeled 4). The 'End events' section shows a table of 6 classes (labeled 5).

| Class | Occurrences (absolute) |
|-----------------------|------------------------|
| Take in charge ticket | 5060 |
| Resolve ticket | 4983 |
| Assign seriousness | 4938 |
| Closed | 4574 |
| Wait | 1463 |
| Require upgrade | 119 |
| Insert ticket | 118 |
| Create SW anomaly | 67 |
| Resolve SW anomaly | 13 |
| Schedule intervention | 5 |
| VERIFIED | 3 |
| INVALID | 2 |
| RESOLVED | 2 |
| DUPLICATE | 1 |

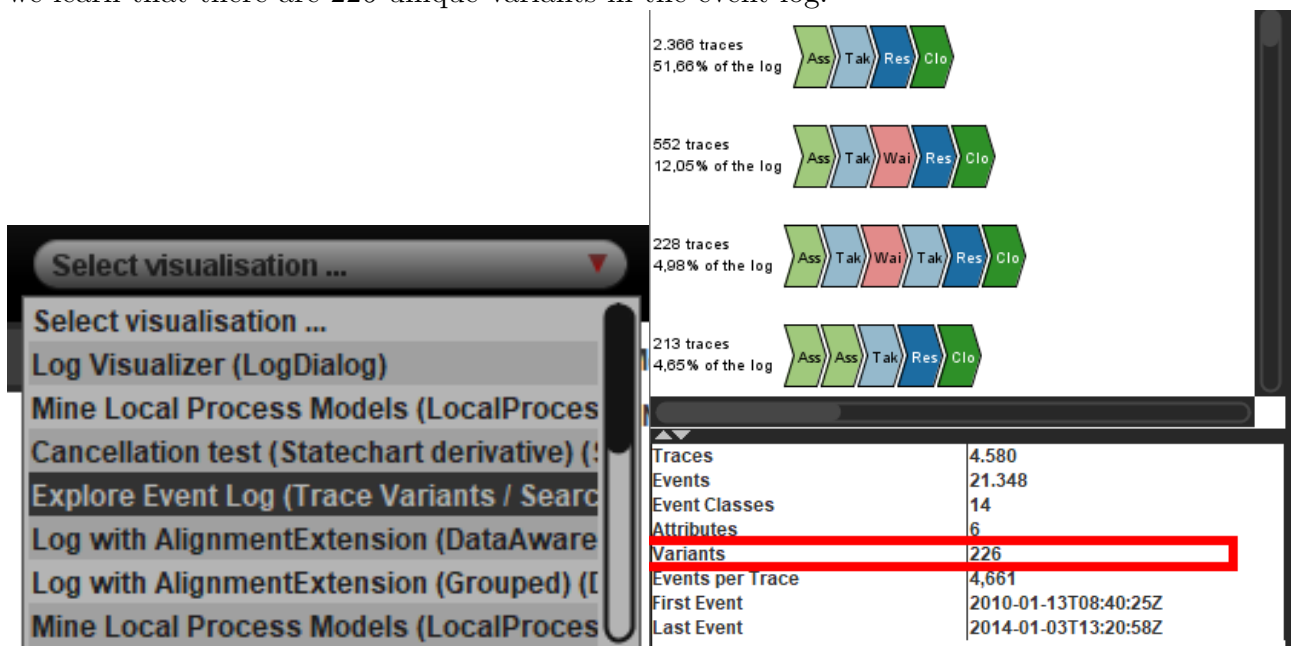
| Class | Occurrences (absolute) |
|-----------------------|------------------------|
| Assign seriousness | 4384 |
| Insert ticket | 118 |
| Take in charge ticket | 74 |
| Resolve ticket | 2 |
| Create SW anomaly | 1 |
| Wait | 1 |

| Class | Occurrences (absolute) |
|-----------------------|------------------------|
| Closed | 4557 |
| Resolve ticket | 10 |
| Wait | 8 |
| Require upgrade | 3 |
| Take in charge ticket | 1 |
| VERIFIED | 1 |

From (3) we get a table of occurrence frequencies for each activity. From (4) we get a table of occurrence frequencies for each start activity. From (5) we get a table of occurrence frequencies for each end activity.

To determine the number of unique trace variants we click on 'Select visualization' and select 'Explore Event Log'

Under this view all trace variants are listed and some further information is given. From this we learn that there are 226 unique variants in the event log.



To-do: Discussion

98% of tickets taken in charge are resolved ($\frac{4983}{5060}$) The variants seem to be quite diverse (1:20 ratio of cases to variants, although distributed very unevenly)

(b)

From the introduction we learned that every trace has to start with 'Insert ticket' or 'Assign seriousness' and ends with 'Closed'. Therefore every trace that does not begin/end with these events must have started/ended outside of our observed time period, making it incomplete.

To filter out incomplete traces we go on the 'Actions' tab, select 'Filter Log using Simple Heuristics' and press 'Start'. In the first dialogue we just click 'Next'. In the next dialogue window we select 'Insert ticket' as our only start event and click 'Next'.

Start events
Only instances starting with a green event will be used.

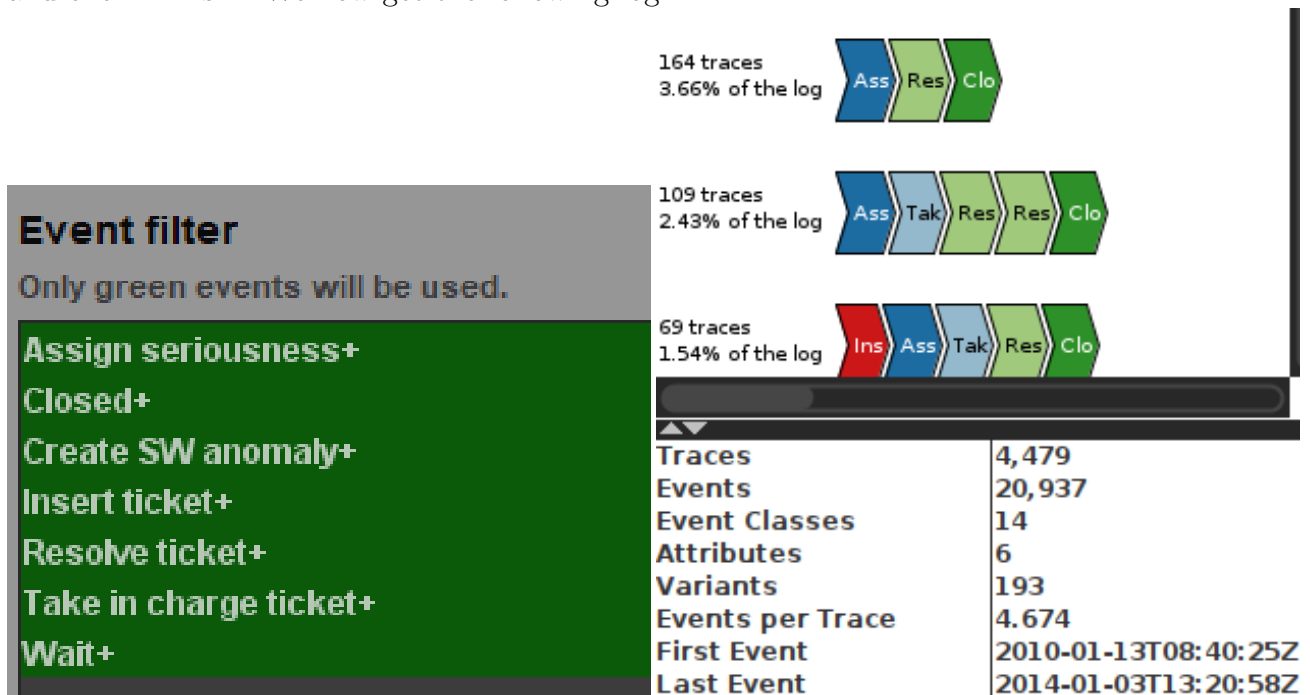
Assign seriousness+
Create SW anomaly+
Insert ticket+
Resolve ticket+
Take in charge ticket+
Wait+

For the end events we only select 'Closed' and click 'Next'.

End events
Only instances ending with a green event will be used.

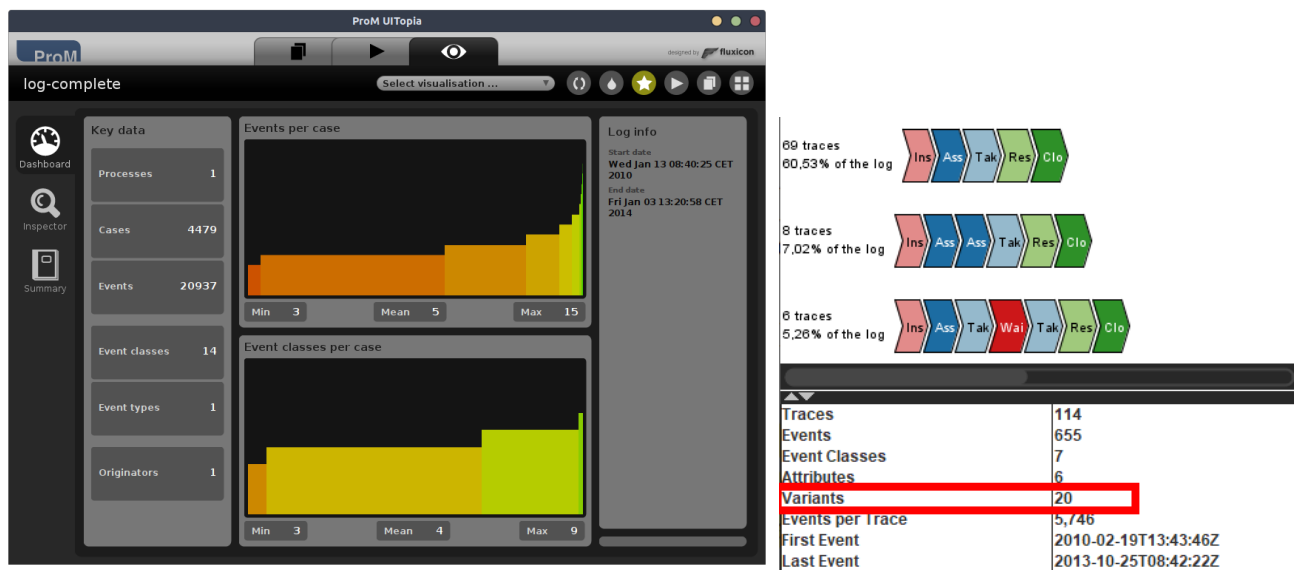
Closed+
Wait+

Since we do not want to filter out any other events we select 100% of events in Event Filter and click 'Finish'. We now get the following log:



(c)

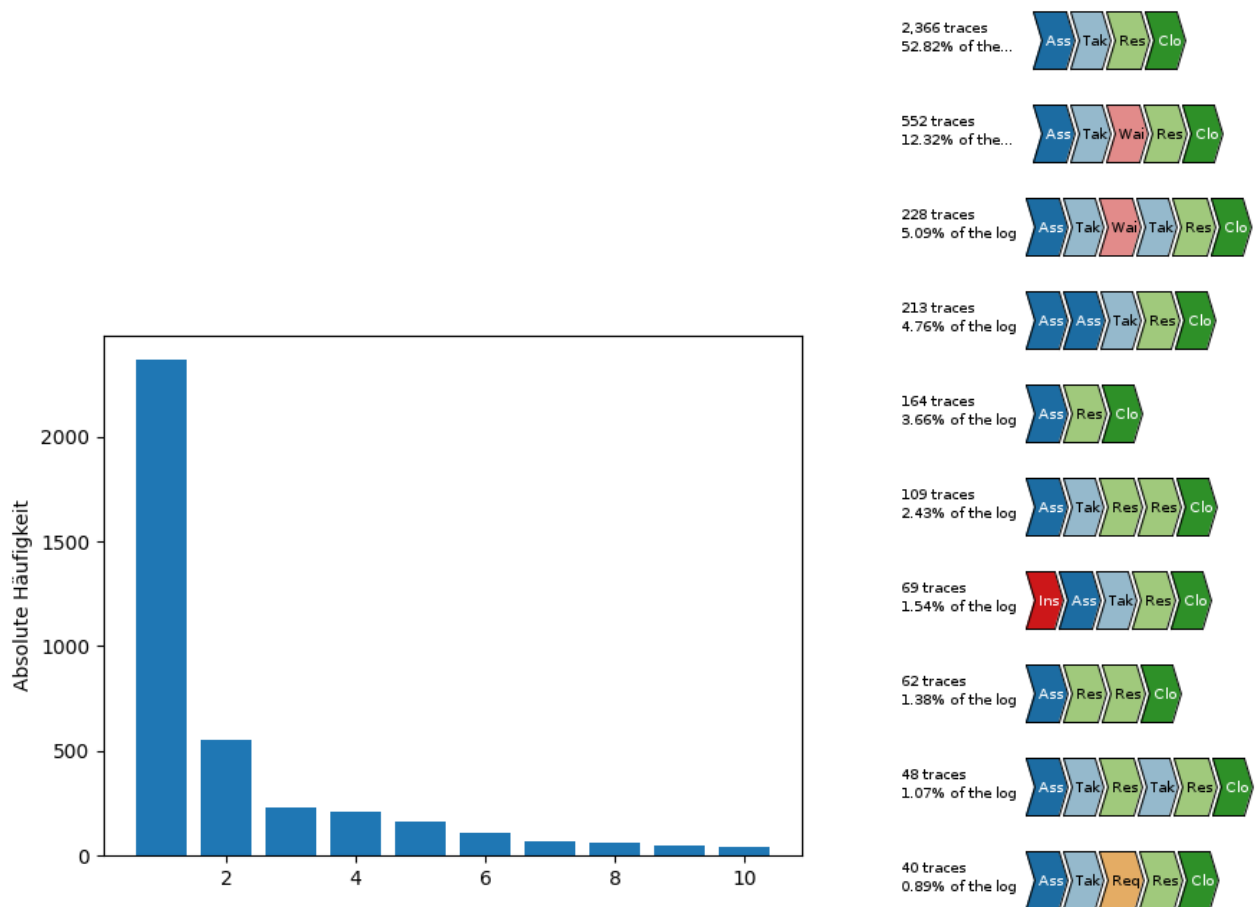
1. As in (a) we inspect the overview of *log-complete* to find 114 cases, 655 events and 7 activities. Also just like in (a) we select the visualization 'Explore Event Log' to find out there are 20 unique trace variants.



2. Just like in (a) we look at 'Summary' to find a table of activities along with their frequency of occurrence:



3. We read the counts of the top 10 traces from the Event Log Explorer in RapidMiner, and plugged those values into a short python script to create the following bar chart.



4. We apply the 'Add Throughput Time as Trace Attribute (In place)' plugin to *log-complete* and select 'DAYS' as the resolution to be used for the elapsed time. We then select the visualization 'Dotted Chart' on the result. To retrieve minimum, maximum and average trace durations we select 'T: throughputtime' as Attribute Statistics and get the following rounded result:

- minimum trace duration: 31 days
- maximum trace duration: 60 days
- average trace duration: 41 days

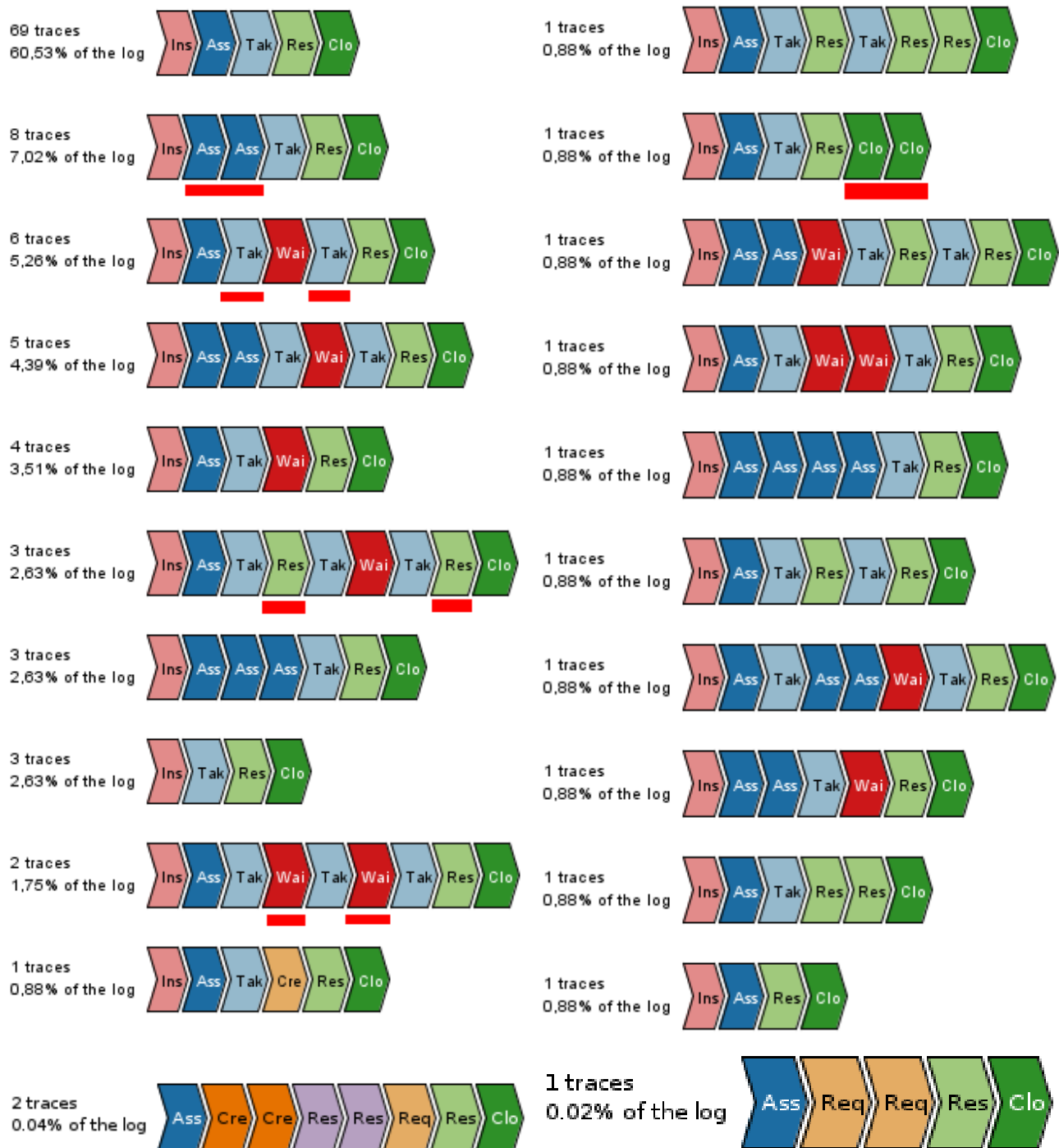


5. To find out which activities appear more than once in at least one trace we again take a look at the 'Explore Event Log' visualization:

Assignment 1

Business Process Intelligence

July 3, 2022



The tasks that appear more than once (as marked in the visualization) are

- Ass: Assign seriousness
- Tak: Take in charge ticket
- Res: Resolve ticket
- Wai: Wait
- Clo: Closed
- Cre: Create SW Anomaly
- Req: Require Update

(d)

1. We choose Pie Charts for these visualization. For *Ticket type* distribution we choose **TICKET TYPE** as dimension:

```
"case_table_csv"."TICKET TYPE"
```

and **COUNT(TICKET TYPE)** as KPI:

```
COUNT("case_table_csv"."TICKET TYPE")
```

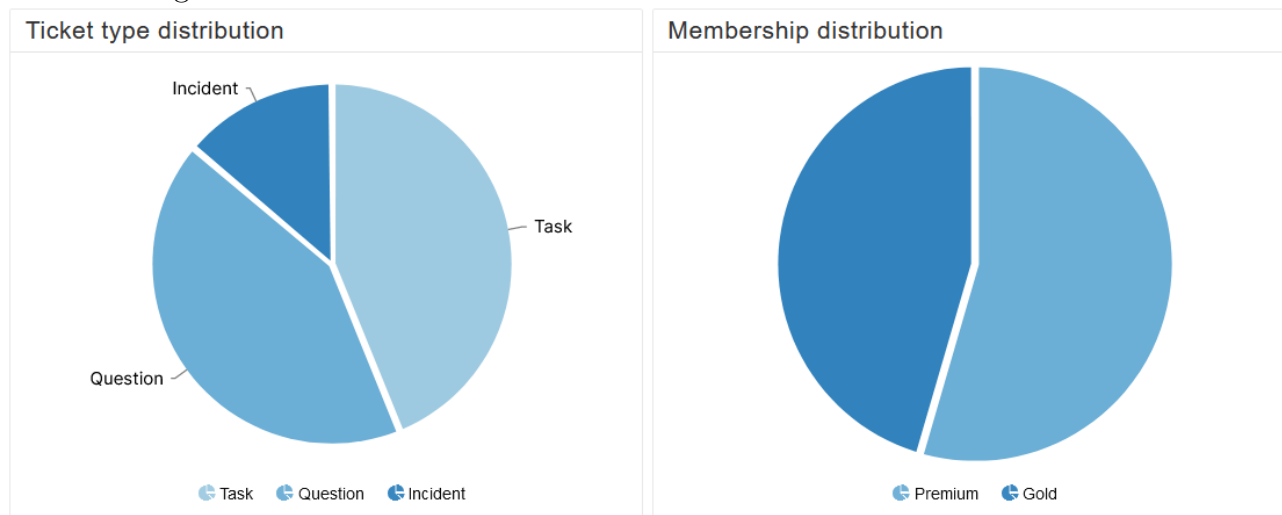
For *Membership* distribution we choose **MEMBERSHIP** as dimension:

```
"case_table_csv"."MEMBERSHIP"
```

and **COUNT(MEMBERSHIP)** as KPI:

```
COUNT("case_table_csv"."MEMBERSHIP")
```

The resulting distribution visualization can be seen below.



2. We obtained the column chart titled 'Total workload per ressource' by using

```
"event_table_csv"."RESOURCE"
```

as dimension and

```
COUNT("event_table_csv"."ACTIVITY")
```

as KPI. The x-Axis shows the resources 1-8 in ascending order, the y-Axis shows the summed number of activities handled by the given resource.

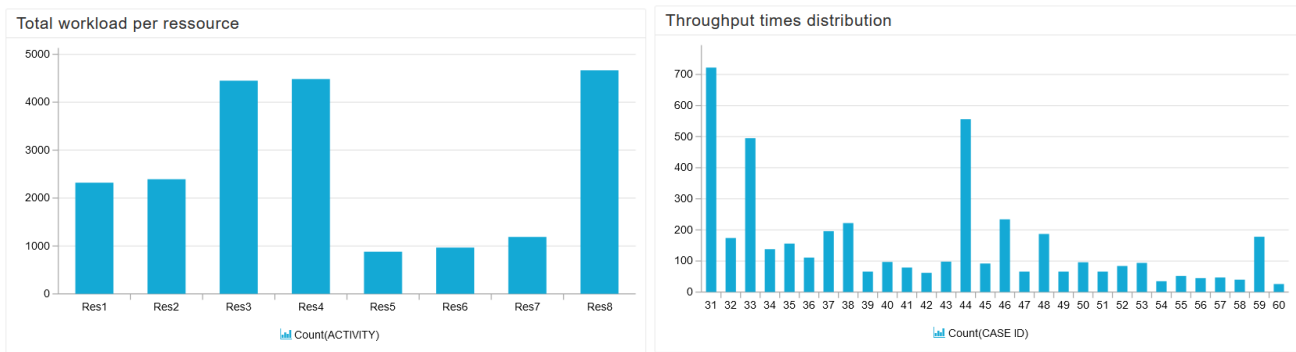
3. We created the column chart named 'Throughput times distribution' by selecting the total throughput time in days as our dimension:

```
CALC_THROUGHPUT(ALL_OCCURRENCE['Process Start'] TO ALL_OCCURRENCE['Process End'], REMAP_
```

and selecting

```
COUNT("case_table_csv"."CASE ID")
```

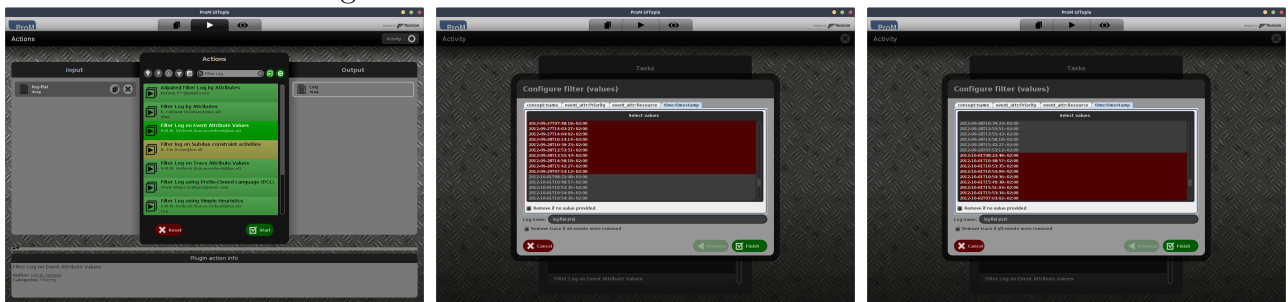
as our KPI. The x-Axis shows the throughput time in days and the y-Axis shows the number of cases with a given throughput time.



Question 2

(a)

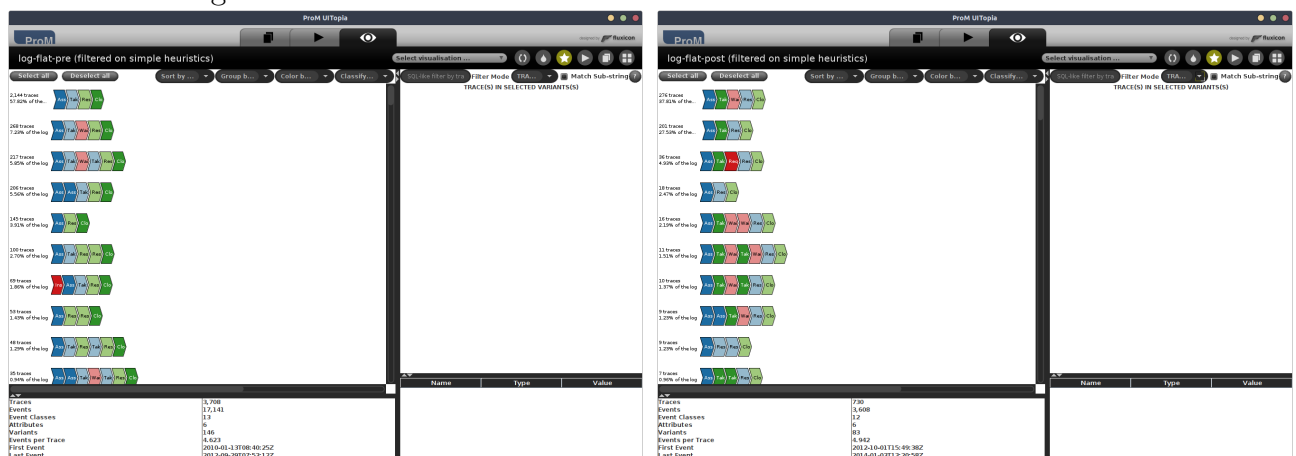
We can split the event log into two event logs using the plugin 'Filter Log on Event Attribute Values' with the following selections:



Afterwards, we can apply our filtering to only allow valid traces, as seen before.

For the event log pre 01.10.2012 we get 13 unique activities across 3708 cases, as a whole consisting of 6 variants.

For the event log post (including) 01.10.2012 we get 12 unique activities across 730 cases, as a whole consisting of 6 variants.



(b)

1. We apply the following workflow both for *log-pre-complete* and *log-post-complete*:

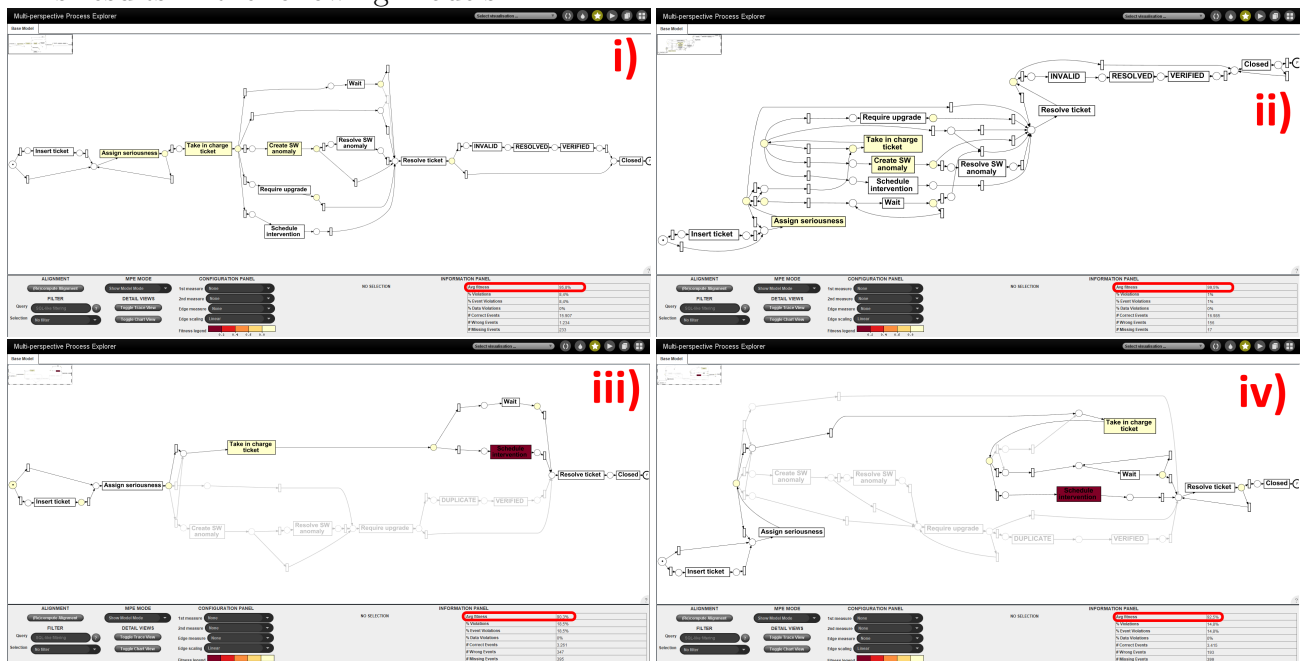
1. Apply plugin 'Interactive Data-aware Heuristic Miner (iDHM)'

2. Set 'Options & Thresholds' to
All tasks connected: True

| | i) (<i>log-pre-complete</i>) | ii) (<i>log-pre-complete</i>) | iii) (<i>log-post-complete</i>) | iv) (<i>log-post-complete</i>) |
|------------|--------------------------------|---------------------------------|-----------------------------------|----------------------------------|
| Frequency | 0.1 | 0 | 0.1 | 0 |
| Dependency | 0.9 | 0.9 | 0.9 | 0.9 |
| Bindings | 0.1 | 0.1 | 0.1 | 0.1 |
| Conditions | 0.5 | 0.5 | 0.5 | 0.5 |

3. Select 'Petri net' as 'Output: Process Model' and click 'Export model'
4. Use the log and the newly created Petri net as input for the 'Multi-perspective Process Miner' plugin

This results in the following Models:



As we can see all these models exceed our fitness threshold of 90% and contain all activities occurring in their respective logs (*log-pre-complete* doesn't contain the activity DUPLICATE and *log-post-complete* doesn't contain the activities INVALID and RESOLVED).

(c)

(d)

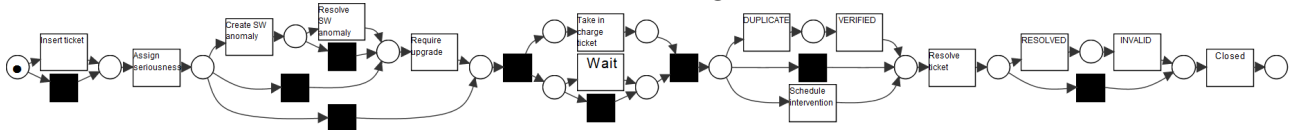
Question 3

(a)

We use the Plugin 'Use Trace Attribute Values' to only select Tickets of type 'Task'. As visualization we then choose 'Explore Event Log'. This gives us the table below:

| | |
|------------------|----------------------|
| Traces | 2.018 |
| Events | 10.283 |
| Event Classes | 14 |
| Attributes | 6 |
| Variants | 201 |
| Events per Trace | 5,096 |
| First Event | 2010-01-13T12:30:37Z |
| Last Event | 2014-01-03T13:20:58Z |

From it we can take that there are 2018 traces, 201 trace variants and 10283 events. We then apply the plugin 'Mine Petri net with Inductive Miner' on this filtered log and make sure we choose 'Inductive Miner - Infrequent (IMf)' as our variant and set it to 20% by assigning the Noise threshold to 0.20. This results in the following Model:



(b)

We use the plugin 'Multi-perspective Process Explorer' on our Petri net from (a). This gives us the following information on fitness and precision:

| | | | |
|--------------------|-------|------------------------|--------|
| Avg fitness | 92,3% | Avg activity precision | 82,9% |
| % Violations | 14,2% | # Moves Observed | 33.237 |
| % Event Violations | 14,2% | # Moves Possible | 40.096 |
| % Data Violations | 0% | Avg fitness | 92,3% |
| # Correct Events | 8.966 | % Violations | 14,2% |
| # Wrong Events | 1.317 | % Event Violations | 14,2% |
| # Missing Events | 166 | % Data Violations | 0% |
| | | # Correct Events | 8.966 |
| | | # Wrong Events | 1.317 |
| | | # Missing Events | 166 |

We obtain the percentage of fitting traces by calculating 100% minus the percentage of violations (100% - 14.2%), resulting in 85,8% fitting traces. Alignment-based fitness (92.3%) and precision (82.9%) can be read from the tables above.

By using the inductive miner again and adjusting the infrequency parameter to 10% we obtain a process model with better fitness, precision and more perfectly fitting traces than before. The stats of the new model can be seen below:

| | |
|------------------------|--------|
| Avg activity precision | 83,6% |
| # Moves Observed | 35.324 |
| # Moves Possible | 42.242 |
| Avg fitness | 92,4% |
| % Violations | 14% |
| % Event Violations | 14% |
| % Data Violations | 0% |
| # Correct Events | 8.985 |
| # Wrong Events | 1.298 |
| # Missing Events | 169 |

(c)

Question 4

(a)

We created the following OLAP Table:

| CASE ID | TICKET TYPE | PRIORITY | RESOURCE OF STARTING ACTI... | NUMBER OF ACTIVE CASES AT ... | Decision |
|-----------|-------------|----------|------------------------------|-------------------------------|---------------|
| Case 1 | Question | Normal | Res4 | 18 | False/No-Wait |
| Case 10 | Task | Normal | Res4 | 18 | False/No-Wait |
| Case 100 | Task | Normal | Res3 | 18 | False/No-Wait |
| Case 1000 | Incident | Urgent | Res8 | 18 | False/No-Wait |
| Case 1001 | Task | Normal | Res4 | 18 | False/No-Wait |
| Case 1002 | Task | High | Res3 | 18 | False/No-Wait |
| Case 1003 | Task | Normal | Res4 | 18 | False/No-Wait |
| Case 1004 | Incident | Normal | Res4 | 18 | False/No-Wait |
| Case 1005 | Task | Normal | Res3 | 18 | False/No-Wait |
| Case 1006 | Question | High | Res4 | 18 | False/No-Wait |
| Case 1007 | Task | Normal | Res3 | 18 | False/No-Wait |
| Case 1008 | Task | High | Res8 | 18 | False/No-Wait |
| Case 1009 | Task | High | Res8 | 18 | True/Wait |
| Case 101 | Question | Normal | Res3 | 18 | False/No-Wait |
| Case 1010 | Task | Normal | Res4 | 18 | False/No-Wait |
| Case 1011 | Question | Normal | Res8 | 18 | False/No-Wait |
| Case 1012 | Incident | Normal | Res8 | 18 | False/No-Wait |
| Case 1013 | Question | Normal | Res4 | 18 | False/No-Wait |
| Case 1014 | Task | Normal | Res4 | 18 | True/Wait |
| Case 1015 | Incident | Normal | Res8 | 18 | False/No-Wait |
| Case 1016 | Question | High | Res8 | 18 | False/No-Wait |
| Case 1017 | Task | Normal | Res4 | 18 | True/Wait |
| Case 1018 | Task | Normal | Res4 | 18 | False/No-Wait |
| Case 1019 | Task | Normal | Res8 | 18 | True/Wait |
| Case 102 | Question | Normal | Res3 | 18 | False/No-Wait |
| Case 1020 | Task | Normal | Res4 | 18 | True/Wait |
| Case 1021 | Question | Normal | Res3 | 18 | False/No-Wait |
| Case 1022 | Question | Normal | Res8 | 18 | False/No-Wait |
| Case 1023 | Task | Normal | Res3 | 18 | False/No-Wait |
| Case 1024 | Task | High | Res4 | 18 | False/No-Wait |
| Case 1025 | Task | Normal | Res4 | 18 | True/Wait |
| Case 1026 | Task | Normal | Res8 | 18 | False/No-Wait |
| Case 1027 | Task | Normal | Res3 | 18 | True/Wait |
| Case 1028 | Task | High | Res4 | 18 | False/No-Wait |
| Case 1029 | Question | Normal | Res3 | 18 | False/No-Wait |
| Case 103 | Task | Normal | Res4 | 18 | True/Wait |

For this, we used the following PQL Queries in the order of columns in the image left to right:

```
"case_table_csv"."CASE ID"
```

```
"case_table_csv"."TICKET TYPE"
```

```
"event_table_csv"."PRIORITY"
```

```
PU_FIRST ( "case_table_csv", "event_table_csv"."RESOURCE")
```

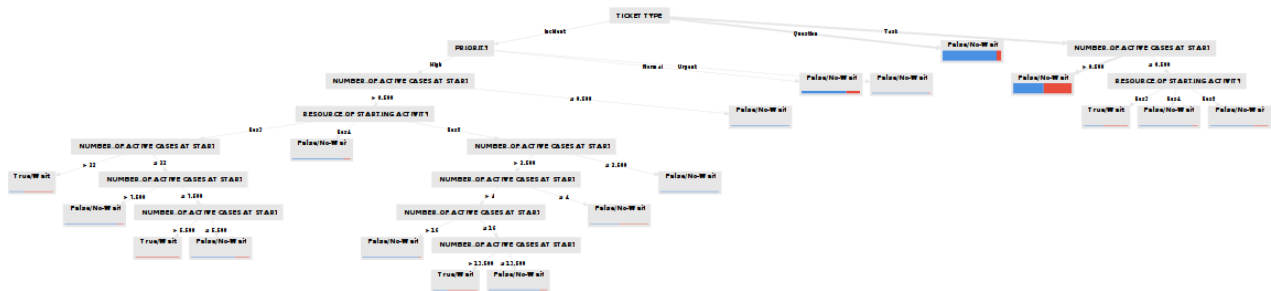
```
RUNNING_SUM (
  CASE WHEN MATCH_PROCESS_REGEX("event_table_csv"."ACTIVITY", 'Closed'$) = 1 THEN 0
  ELSE 1
  END
)
```

```
CASE WHEN
MATCH_PROCESS_REGEX ( "event_table_csv"."ACTIVITY", 'Wait' ) = 1 THEN 'True/Wait'
ELSE 'False/No-Wait'
END
```

(b)

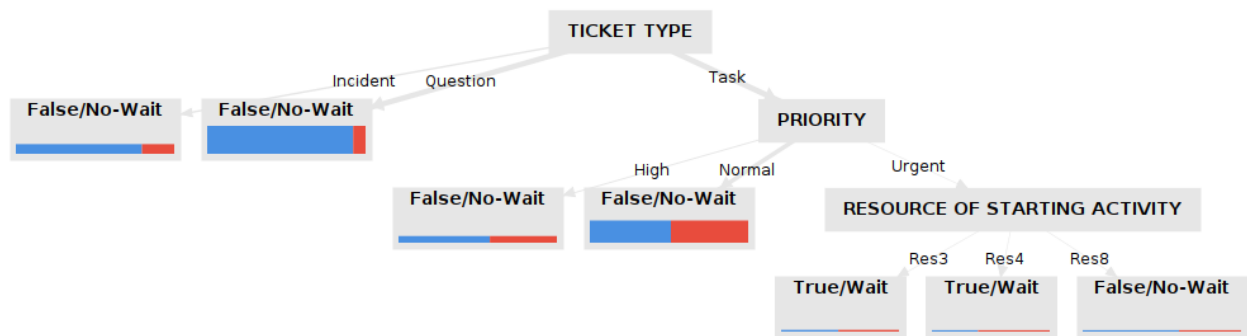
First, we export the OLAP Table and import it into RapidMiner as described in Instruction 2. After setting the attribute Decision as our label, we had to adjust the minimal gain ratio to 0.006 in order to see more than just one Wait-Leaf. The resulting decision tree is evidently too large to fit into a PDF, thus we have added the description in the Appendix:

July 3, 2022



We found that some tasks using Resource 3 would wait, even if they were the only task running at creation. From the tree we can also observe that some high priority incidents (using Resource 3 or 8) would have to wait depending on the number of active cases at start, although there does not seem to be any connection to the prior predictor variables.

After removing the attribute 'number of active cases at start' and further lowering the minimal gain ratio to 0.001, we get a much more simplified, but comprehensible Decision Tree:



Here we see that urgent tasks using the starting resource 3 or 4 are quite likely to be set into waiting mode.

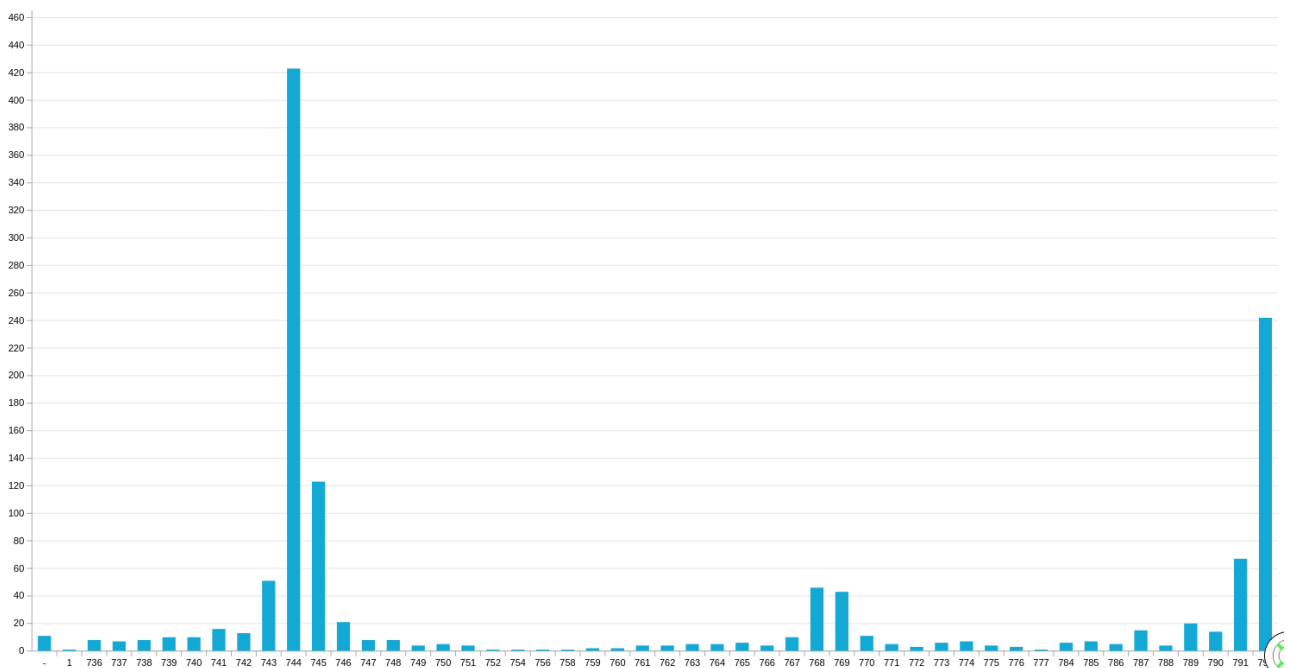
Question 5

(a)

Similarly to 1d), we created a new column chart. Here, we used the following Dimension:

```
CASE WHEN MATCH_PROCESS_REGEX("event_table_csv"."ACTIVITY", 'Closed'$) = 1
THEN CALC_THROUGHPUT(
CASE_START TO CASE_END,
REMAP_TIMESTAMPS("event_table_csv"."TIMESTAMP", HOURS)
)
ELSE CALC_THROUGHPUT(
CASE_START TO LAST_OCCURRENCE['Resolve ticket'],
REMAP_TIMESTAMPS("event_table_csv"."TIMESTAMP", HOURS)
)
END
```

Using the Case Count as a Dimension, we got the following chart. Note that this is just a tiny section of the entire chart. There are cases with throughput times up to 1440h:



(b)

We applied the `Quantile` function on the real throughput times (see above) for 0.3 and 0.7, respectively (here for 0.3):

```

QUANTILE(CASE WHEN MATCH_PROCESS_REGEX("event_table_csv"."ACTIVITY", 'Closed'$) = 1
THEN CALC_THROUGHPUT(
CASE_START TO CASE_END,
REMAP_TIMESTAMPS("event_table_csv"."TIMESTAMP", HOURS)
)
ELSE CALC_THROUGHPUT(
CASE_START TO LAST_OCCURRENCE['Resolve ticket'],
REMAP_TIMESTAMPS("event_table_csv"."TIMESTAMP", HOURS)
)
END, 0.3)

```

For the 0.3-quantile we got 795h, for the 0.7-quantile 1079h.

(c)

Approach equivalent to Question 4a). We used the following PQL queries:

```
"case_table_csv"."CASE ID"
```

```
"case_table_csv"."TICKET TYPE"
```

```
"case_table_csv"."MEMBERSHIP"
```

```
"event_table_csv"."PRIORITY"
```

```
CASE
```

```

WHEN (CASE WHEN MATCH_PROCESS_REGEX("event_table_csv"."ACTIVITY", 'Closed'$) = 1
THEN CALC_THROUGHPUT(
CASE_START TO CASE_END,

```

```

REMAP_TIMESTAMPS("event_table_csv"."TIMESTAMP",HOURS)
)
ELSE CALC_THROUGHPUT(
CASE_START TO LAST_OCCURRENCE['Resolve ticket'],
REMAP_TIMESTAMPS("event_table_csv"."TIMESTAMP",HOURS)
)
END) < 795 THEN 'Short'

WHEN (CASE WHEN MATCH_PROCESS_REGEX("event_table_csv"."ACTIVITY", 'Closed'$) = 1
THEN CALC_THROUGHPUT(
CASE_START TO CASE_END,
REMAP_TIMESTAMPS("event_table_csv"."TIMESTAMP",HOURS)
)
ELSE CALC_THROUGHPUT(
CASE_START TO LAST_OCCURRENCE['Resolve ticket'],
REMAP_TIMESTAMPS("event_table_csv"."TIMESTAMP",HOURS)
)
END) < 1079 THEN 'Medium'

ELSE 'Long'

END

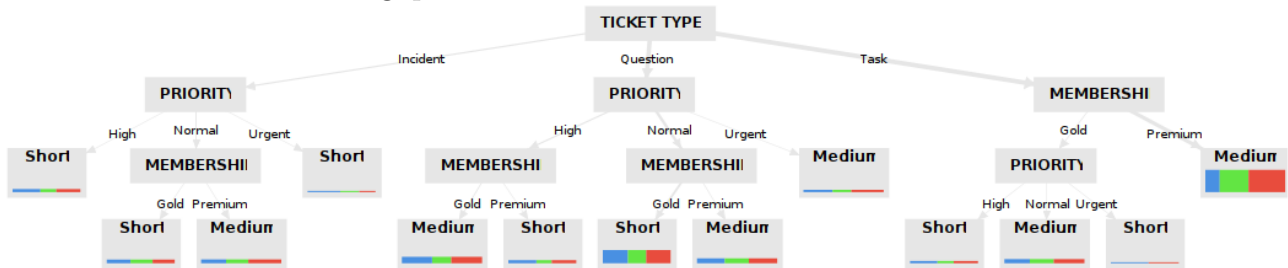
```

| CASE ID | TICKET TYPE | MEMBERSHIP | PRIORITY | Performance class |
|-----------|-------------|------------|----------|-------------------|
| Case 1 | Question | Gold | Normal | Short |
| Case 10 | Task | Premium | Normal | Long |
| Case 100 | Task | Premium | Normal | Long |
| Case 1000 | Incident | Gold | Urgent | Medium |
| Case 1001 | Task | Premium | Normal | Medium |
| Case 1002 | Task | Gold | High | Short |
| Case 1003 | Task | Premium | Normal | Medium |
| Case 1004 | Incident | Premium | Normal | Short |
| Case 1005 | Task | Premium | Normal | Long |
| Case 1006 | Question | Gold | High | Long |
| Case 1007 | Task | Premium | Normal | Long |
| Case 1008 | Task | Gold | High | Medium |
| Case 1009 | Task | Premium | High | Medium |
| Case 101 | Question | Premium | Normal | Long |
| Case 1010 | Task | Premium | Normal | Medium |
| Case 1011 | Question | Gold | Normal | Long |
| Case 1012 | Incident | Premium | Normal | Short |
| Case 1013 | Question | Premium | Normal | Medium |
| Case 1014 | Task | Premium | Normal | Short |
| Case 1015 | Incident | Gold | Normal | Short |
| Case 1016 | Question | Gold | High | Long |
| Case 1017 | Task | Premium | Normal | Medium |
| Case 1018 | Task | Premium | Normal | Long |
| Case 1019 | Task | Premium | Normal | Short |
| Case 102 | Question | Gold | Normal | Long |
| Case 1020 | Task | Premium | Normal | Long |
| Case 1021 | Question | Premium | Normal | Medium |
| Case 1022 | Question | Premium | Normal | Short |
| Case 1023 | Task | Premium | Normal | Short |
| Case 1024 | Task | Gold | High | Medium |
| Case 1025 | Task | Premium | Normal | Long |
| Case 1026 | Task | Premium | Normal | Medium |
| Case 1027 | Task | Premium | Normal | Medium |
| Case 1028 | Task | Gold | High | Short |
| Case 1029 | Question | Gold | Normal | Short |
| Case 103 | Task | Premium | Normal | Medium |

(d)

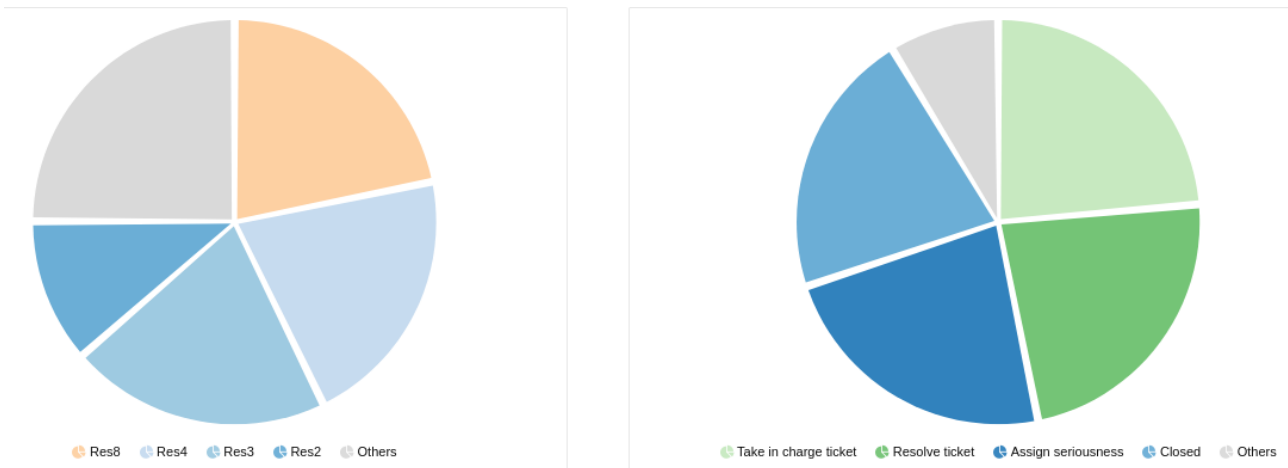
First, we export the OLAP Table and import it into RapidMiner as described in Instruction 2. Even after setting the parameters of the Decision Tree algorithm to extreme values (minimal gain at 1.0E-6), we could not find any clear variable that predicts the outcome of the performance class in any way. Thus, we conclude that the performance class (and thus the real

throughput time) has no strong correlation with neither Priority nor Ticket Type. The only thing noticeable was, that close to half the Tasks stemming from Premium Memberships were Medium rated in real throughput time.



Question 6

(a)



For the left Pie Chart we used "event_table_csv"."RESOURCE" as the Dimension and COUNT("event_table_csv"."RESOURCE") as the KPI.

For the right Pie Chart we used "event_table_csv"."ACTIVITY" as the Dimension and COUNT("event_table_csv"."RESOURCE") as the KPI.

(b)

Appendix

Question 4b) Decision Tree description:

```

1 TICKET TYPE = Incident
2 |   PRIORITY = High
3 | |   NUMBER OF ACTIVE CASES AT START > 0.500
4 | | |   RESOURCE OF STARTING ACTIVITY = Res3
5 | | | |   NUMBER OF ACTIVE CASES AT START > 22: True/Wait {False/No-
        Wait=1, True/Wait=2}
6 | | | |   NUMBER OF ACTIVE CASES AT START ? 22
7 | | | | |   NUMBER OF ACTIVE CASES AT START > 7.500: False/No-Wait {
        False/No-Wait=34, True/Wait=4}
8 | | | | |   NUMBER OF ACTIVE CASES AT START ? 7.500
9 | | | | |   NUMBER OF ACTIVE CASES AT START > 5.500: True/Wait {
        False/No-Wait=0, True/Wait=3}

```



```

10 | | | | | | | NUMBER OF ACTIVE CASES AT START ? 5.500: False/No-
    | | | | | | | Wait {False/No-Wait=6, True/Wait=2}
11 | | | | | | | RESOURCE OF STARTING ACTIVITY = Res4: False/No-Wait {False/No-
    | | | | | | | Wait=45, True/Wait=6}
12 | | | | | | | RESOURCE OF STARTING ACTIVITY = Res8
13 | | | | | | | NUMBER OF ACTIVE CASES AT START > 2.500
14 | | | | | | | NUMBER OF ACTIVE CASES AT START > 4
15 | | | | | | | NUMBER OF ACTIVE CASES AT START > 16: False/No-Wait {
    | | | | | | | False/No-Wait=23, True/Wait=1}
16 | | | | | | | NUMBER OF ACTIVE CASES AT START ? 16
17 | | | | | | | NUMBER OF ACTIVE CASES AT START > 13.500: True/
    | | | | | | | Wait {False/No-Wait=1, True/Wait=2}
18 | | | | | | | NUMBER OF ACTIVE CASES AT START ? 13.500: False/No
    | | | | | | | -Wait {False/No-Wait=20, True/Wait=3}
19 | | | | | | | NUMBER OF ACTIVE CASES AT START ? 4: False/No-Wait {False
    | | | | | | | /No-Wait=1, True/Wait=1}
20 | | | | | | | NUMBER OF ACTIVE CASES AT START ? 2.500: False/No-Wait {False/
    | | | | | | | No-Wait=5, True/Wait=0}
21 | | | | | | | NUMBER OF ACTIVE CASES AT START ? 0.500: False/No-Wait {False/No-
    | | | | | | | Wait=3, True/Wait=0}
22 | | | | | | | PRIORITY = Normal: False/No-Wait {False/No-Wait=330, True/Wait=101}
23 | | | | | | | PRIORITY = Urgent: False/No-Wait {False/No-Wait=24, True/Wait=1}
24 | | | | | | | TICKET TYPE = Question: False/No-Wait {False/No-Wait=1792, True/Wait=151}
25 | | | | | | | TICKET TYPE = Task
26 | | | | | | | NUMBER OF ACTIVE CASES AT START > 0.500: False/No-Wait {False/No-Wait
    | | | | | | | =1040, True/Wait=951}
27 | | | | | | | NUMBER OF ACTIVE CASES AT START ? 0.500
28 | | | | | | | RESOURCE OF STARTING ACTIVITY = Res3: True/Wait {False/No-Wait=4,
    | | | | | | | True/Wait=5}
29 | | | | | | | RESOURCE OF STARTING ACTIVITY = Res4: False/No-Wait {False/No-Wait
    | | | | | | | =9, True/Wait=1}
30 | | | | | | | RESOURCE OF STARTING ACTIVITY = Res8: False/No-Wait {False/No-Wait
    | | | | | | | =6, True/Wait=2}TICKET TYPE = Incident
31 | | | | | | | PRIORITY = High
32 | | | | | | | NUMBER OF ACTIVE CASES AT START > 0.500
33 | | | | | | | RESOURCE OF STARTING ACTIVITY = Res3
34 | | | | | | | NUMBER OF ACTIVE CASES AT START > 22: True/Wait {False/No-
    | | | | | | | Wait=1, True/Wait=2}
35 | | | | | | | NUMBER OF ACTIVE CASES AT START ? 22
36 | | | | | | | NUMBER OF ACTIVE CASES AT START > 7.500: False/No-Wait {
    | | | | | | | False/No-Wait=34, True/Wait=4}
37 | | | | | | | NUMBER OF ACTIVE CASES AT START ? 7.500
38 | | | | | | | NUMBER OF ACTIVE CASES AT START > 5.500: True/Wait {
    | | | | | | | False/No-Wait=0, True/Wait=3}
39 | | | | | | | NUMBER OF ACTIVE CASES AT START ? 5.500: False/No-
    | | | | | | | Wait {False/No-Wait=6, True/Wait=2}
40 | | | | | | | RESOURCE OF STARTING ACTIVITY = Res4: False/No-Wait {False/No-
    | | | | | | | Wait=45, True/Wait=6}
41 | | | | | | | RESOURCE OF STARTING ACTIVITY = Res8
42 | | | | | | | NUMBER OF ACTIVE CASES AT START > 2.500

```

```

43 | | | | | NUMBER OF ACTIVE CASES AT START > 4
44 | | | | | | NUMBER OF ACTIVE CASES AT START > 16: False/No-Wait {
    False/No-Wait=23, True/Wait=1}
45 | | | | | | NUMBER OF ACTIVE CASES AT START ? 16
46 | | | | | | | NUMBER OF ACTIVE CASES AT START > 13.500: True/
    Wait {False/No-Wait=1, True/Wait=2}
47 | | | | | | | | NUMBER OF ACTIVE CASES AT START ? 13.500: False/No
    -Wait {False/No-Wait=20, True/Wait=3}
48 | | | | | | NUMBER OF ACTIVE CASES AT START ? 4: False/No-Wait {False
    /No-Wait=1, True/Wait=1}
49 | | | | | NUMBER OF ACTIVE CASES AT START ? 2.500: False/No-Wait {False/
    No-Wait=5, True/Wait=0}
50 | | | NUMBER OF ACTIVE CASES AT START ? 0.500: False/No-Wait {False/No-
    Wait=3, True/Wait=0}
51 | | | | PRIORITY = Normal: False/No-Wait {False/No-Wait=330, True/Wait=101}
52 | | | | PRIORITY = Urgent: False/No-Wait {False/No-Wait=24, True/Wait=1}
53 | | | | TICKET TYPE = Question: False/No-Wait {False/No-Wait=1792, True/Wait=151}
54 | | | | TICKET TYPE = Task
55 | | | | | NUMBER OF ACTIVE CASES AT START > 0.500: False/No-Wait {False/No-Wait
    =1040, True/Wait=951}
56 | | | | | NUMBER OF ACTIVE CASES AT START ? 0.500
57 | | | | | | RESOURCE OF STARTING ACTIVITY = Res3: True/Wait {False/No-Wait=4,
    True/Wait=5}
58 | | | | | | RESOURCE OF STARTING ACTIVITY = Res4: False/No-Wait {False/No-Wait
    =9, True/Wait=1}
59 | | | | | | RESOURCE OF STARTING ACTIVITY = Res8: False/No-Wait {False/No-Wait
    =6, True/Wait=2}

```