Open for Innovation

KNIME

BUPAR

BIARU
Business Informatics
Applied Research Unit

UHASSELT

bupaR 4 KNIME

Manual

Nodes are divided according to package and function

bupaR
- ➥Counters
- ➥Helpers
- ➥IO
- ➥Manipulation
- ➥Summaries
- ➥Verbs

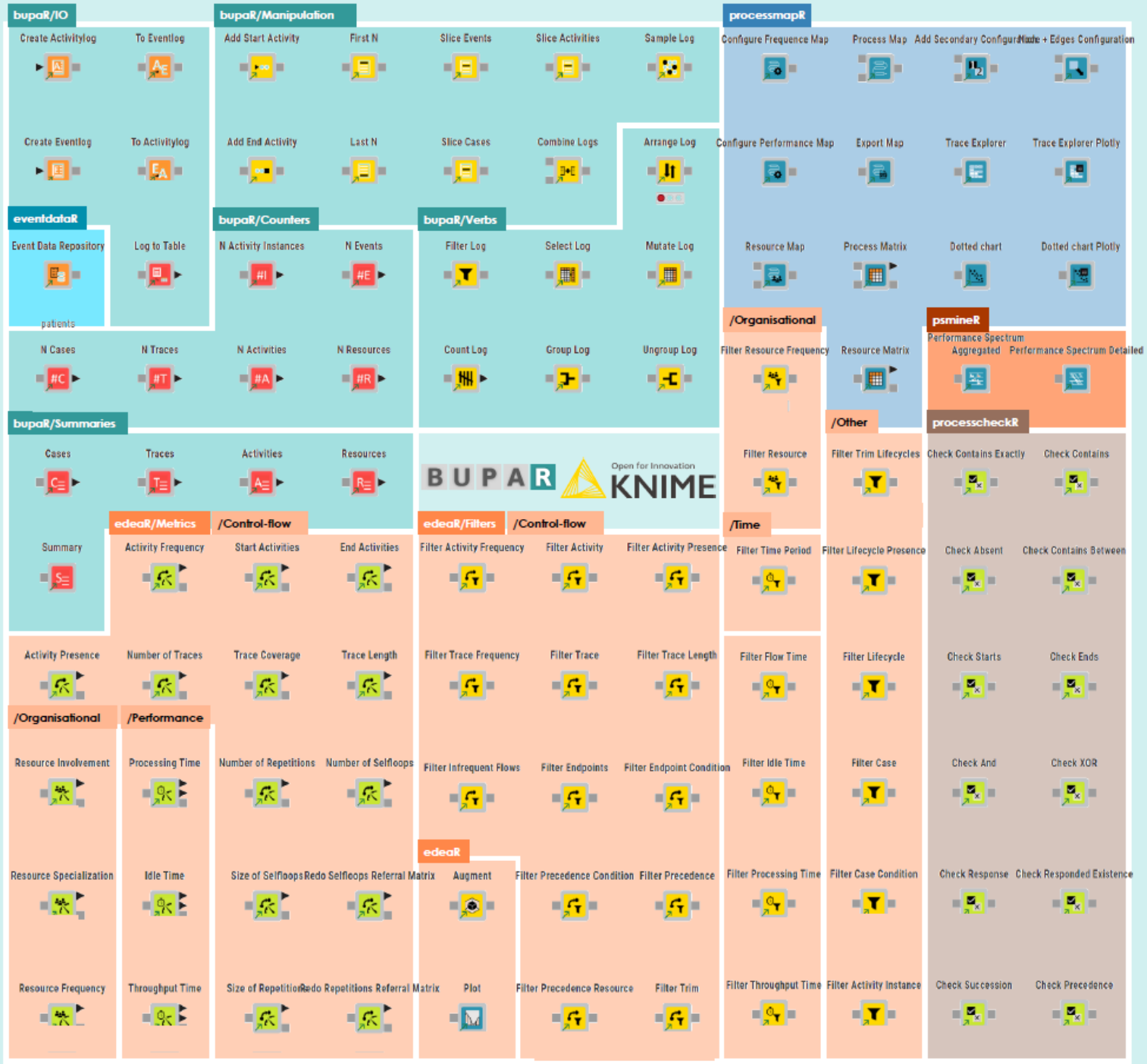
edeaR
- ➥Filters
- ➥Metrics

eventdataR

processcheckR

processmapR

psmineR



bupaR

IO

Create Activitylog
Create Eventlog
To Activitylog
To Eventlog
Log to Table

Counters

N Activities
N Activity Instances
N Cases
N Events
N Resources
N Traces

Summaries

Activities
Cases
Resources
Summary
Traces

Verbs

Arrange Log
Count Log
Filter Log
Group Log
Mutate Log
Select Log
Ungroup Log

Manipulation

Add Start Activity
Add End Activity
Combine Logs
First N
Last N
Sample Log
Slice Activities
Slice Cases
Slice Events

eventdataR

Event Data Respository

edeaR

Plot
Augment

Metrics

Organisational

Resource Frequency
Resource Involvement
Resource Specalization

Performance

Idle Time
Processing Time
Throughput Time

edeaR

Metrics

Control-flow

Activity Frequency
Activity Presence
End Activity
Number of Repetitions
Number of Selfloops
Number of Traces
Redo Repetitions Referal Mtrx
Redo Selfloops Referral Mtrx
Size of Repetitions
Size of Selfoops
Start Activities
Trace Coverage
Trace Length

Filters

Organisational

Filter Resource
Filter Resource Frequency

Performance

Filter Flow Time
Filter Idle Time
Filter Processing Time
Filter Throughput Time

Time

Filter Time Period

Filters

Control-flow

Fitler Activity
Filter Activity Frequency
Filter Activity Presence
Filter Endpoint Condition
Filter Endpoints
Filter Infrequent flow
Filter Precedence
Filter Precendence cond.
Filter Precedence Res.
Filter Trace
Fitler Trace Frequency
Fitler Trace Length
Filter Trim

Other

Filter Activity Instance
Filter Case
Filter Case Condition
Filter Lifecycle
Filter Lifecycle presence
Filter Trim Lifecycles

processcheckR

Check Absent
Check And
Check Contains
Check Contains Beteen
Check Contains Exactly
Check Ends
Check Precedence
Check Responded Existence
Check Response
Check Starts
Check Succession
Check XoR

processmapR

Add Secondary Configuration
Configure Frequency Map
Configue Performance Map
Dotted Chart
Dotted Chart Plotly
Export Map
Node + Edges Configuration
Process Map
Process Matrix
Resource Map
Resource Matrix
Trace Explorer
Trace Explorer Plotly

psmineR

Performance Spectrum Aggr.
Peformance Spectrum Det.

bupaR

IO / Counters / Summaries / Verbs / Manipulation

BUPAR

BUPAR

IO

Creating and exporting logs

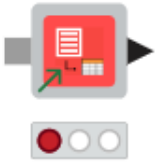
Create Activitylog

To Eventlog

Create Eventlog

To Activitylog

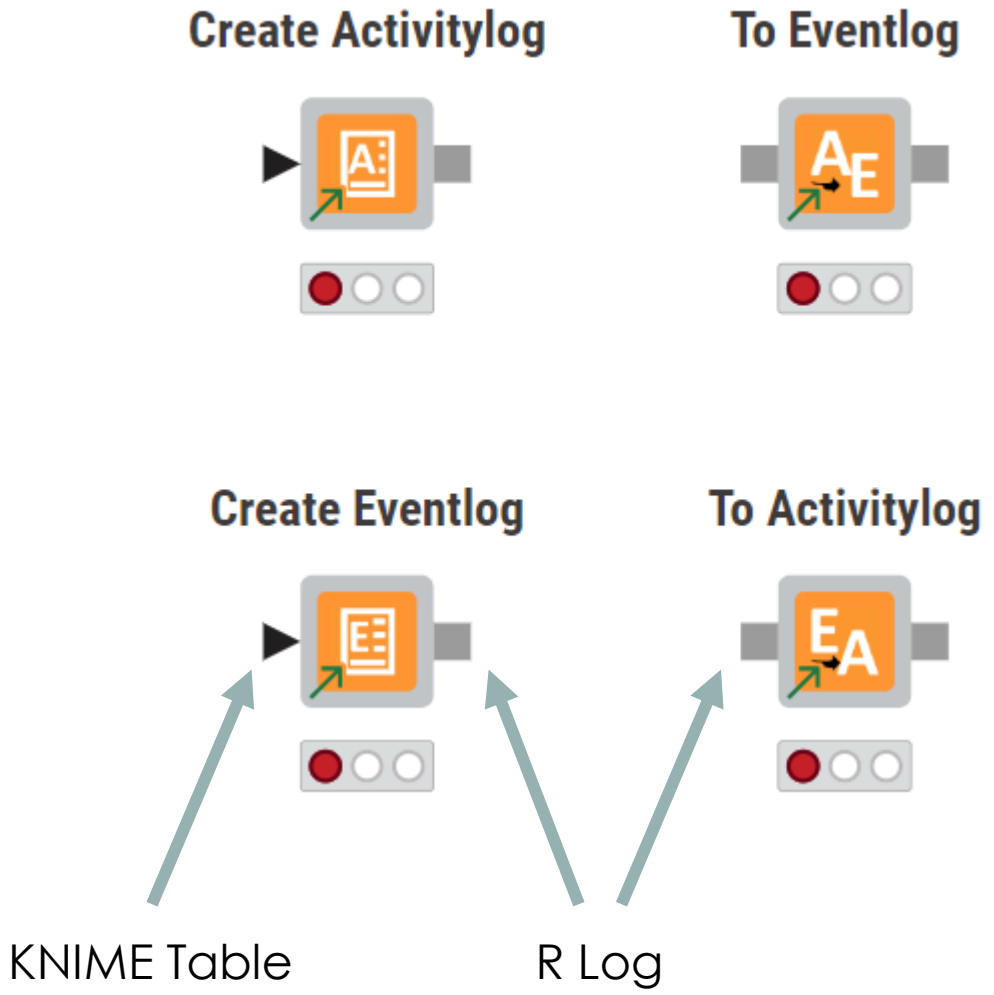
Log to Table



Grey square input and output ports represent R objects. Most of the time, this will be logs, but it can also be derived objects.

The *create* functions convert a regular KNIME Table to an R object that is recognized as a log.

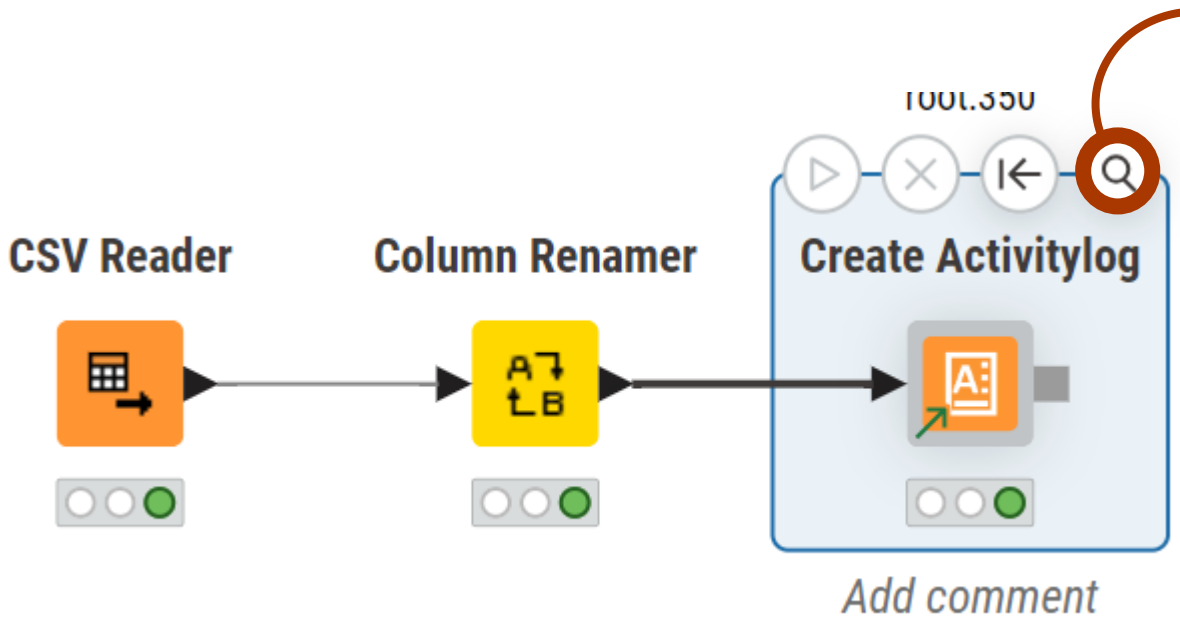
The *To* functions covert one type of log into the other. Most nodes in bupaR4KNIME work for both types of logs, although some only work for event logs, the most flexible format.



Nodes with an R output node have a magnifying glass button. Clicking on this button opens an R view with the main R object at that point (typically the log).

On top, one can see the R functions that have been applied up until that point. This allows you to learn how to replicate the operations in R natively (instead of KNIME).

Note that any necessary manipulations performed before it was an R log (in this example, the column renamer), will not be included in this script.




```
> df %>%
   activitylog(case_id = case_id, activity_id = activity, resource_id = resource_id, timestamps = c('start', 'complete'))

# Log of 1880 events consisting of:
35 traces
35 cases
940 instances of 16 activities
1 resource
Events occurred from 2011-11-28 02:27:59 until 2012-12-03 01:03:12

# Variables were mapped as follows:
Case identifier: case_id
Activity identifier: activity
Resource identifier: resource_id
Timestamps: start, complete

# A tibble: 940 x 6
   case_id activity   start               complete            resource_id .order
     <int> <fct>      <dttm>              <dttm>              <lgl>        <int>
 1      21 Start      2012-11-12 00:48:38 2012-11-12 00:48:38 NA               1
 2      21 washing    2012-11-12 00:48:38 2012-11-12 00:50:12 NA               2
 3      21 watchingtv 2012-11-12 00:50:29 2012-11-12 01:52:12 NA               3
 4      21 toilet     2012-11-12 01:53:19 2012-11-12 01:53:22 NA               4
 5      21 washing    2012-11-12 01:53:27 2012-11-12 01:53:35 NA               5
 6      21 toilet     2012-11-12 01:53:39 2012-11-12 01:53:57 NA               6
 7      21 sleeping   2012-11-12 01:55:09 2012-11-12 09:31:19 NA               7
 8      21 toilet     2012-11-12 09:37:28 2012-11-12 09:41:17 NA               8
 9      21 prepareBr~ 2012-11-12 09:42:02 2012-11-12 09:50:02 NA               9
10      21 eatingBre~ 2012-11-12 09:52:33 2012-11-12 09:55:29 NA              10
# i 930 more rows
# i Use `print(n = ...)` to see more rows
```

Event *vs* activity log

Event Log

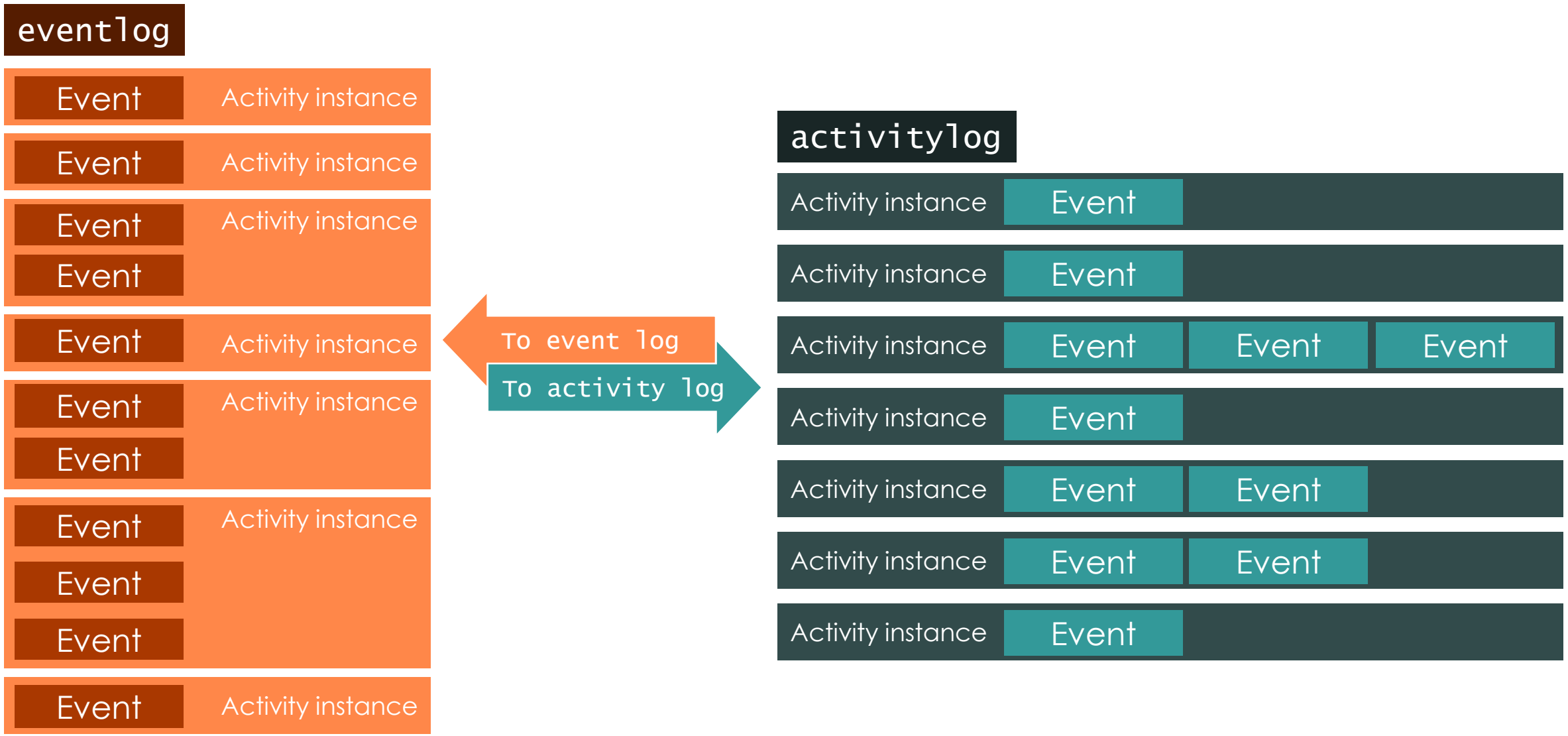
Activity	Case	Resource	Activity Instance	Lifecycle	Timestamp	
handling	patient	employee	handling_id	registration_type	time	
Registration	1	r1	1	start	2/01/2017	11:41:53
Registration	1	r1	1	complete	2/01/2017	12:40:20
Triage and Assessment	1	r2	501	start	2/01/2017	12:40:20
Triage and Assessment	1	r2	501	complete	2/01/2017	22:32:25
Blood test	1	r3	1001	start	5/01/2017	8:59:04
Blood test	1	r3	1001	complete	5/01/2017	14:34:27
MRI SCAN	1	r4	1238	start	5/01/2017	21:37:12
MRI SCAN	1	r4	1238	complete	6/01/2017	1:54:23
Discuss Results	1	r6	1735	start	7/01/2017	7:57:49
Discuss Results	1	r6	1735	complete	7/01/2017	10:18:08
Check-out	1	r7	2230	start	9/01/2017	17:09:43
Check-out	1	r7	2230	complete	9/01/2017	19:45:45

each row is an event, i.e. a single moment in time, thus an event log

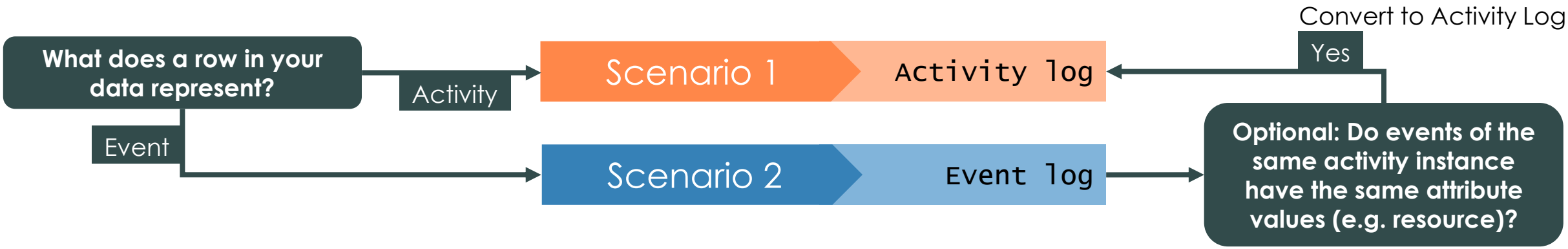
Activity Log

Activity	Case	Resource	Timestamps			
handling	patient	employee	complete		start	
Registration	1	r1	2/01/2017	12:40:20	2/01/2017	11:41:53
Triage and Assessment	1	r2	2/01/2017	22:32:25	2/01/2017	12:40:20
Blood test	1	r3	5/01/2017	14:34:27	5/01/2017	8:59:04
MRI SCAN	1	r4	6/01/2017	1:54:23	5/01/2017	21:37:12
Discuss Results	1	r6	7/01/2017	10:18:08	7/01/2017	7:57:49
Check-out	1	r7	9/01/2017	19:45:45	9/01/2017	17:09:43

each row is an activity, i.e. a single moment in time, thus an activity log


eventlog
Event
Activity instance
Event
Activity instance
Event
Event
Activity instance
Event
Activity instance
Event
Event
Activity instance
Event
Event
Event
Activity instance
Event
Activity instance
To event log
To activity log
activitylog
Activity instance
Event
Activity instance
Event
Activity instance
Event
Event
Event
Activity instance
Event
Activity instance
Event
Event
Activity instance
Event
Event
Activity instance
Event

How to chose?


What does a row in your data represent?
Activity
Event
Scenario 1
Activity log
Scenario 2
Event log
Convert to Activity Log
Yes
Optional: Do events of the same activity instance have the same attribute values (e.g. resource)?

BUPAR

Q1: What does a row in your data represent?

If each row is an activity instance go to **Scenario 1**

If not, go to **Q2**

How to recognize?

- It has more than a single timestamp

or

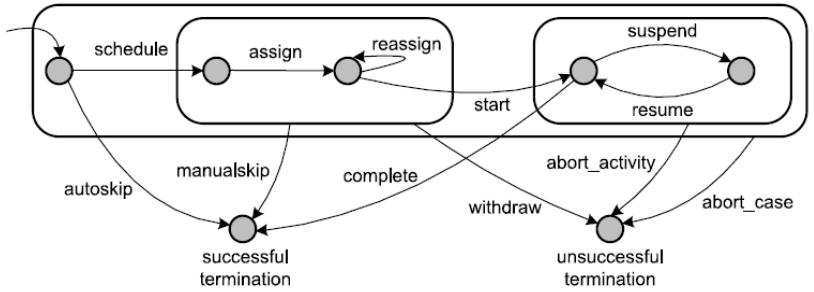
- It has one timestamp but no lifecycle notion (start / complete)

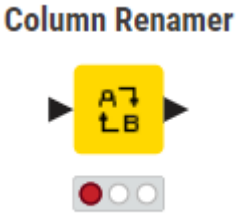
↪ Activity instances

Scenario 1

Each row is an activity instance.

1. Make sure these timestamp columns have appropriate names. If not, fix with *Column Renamer*



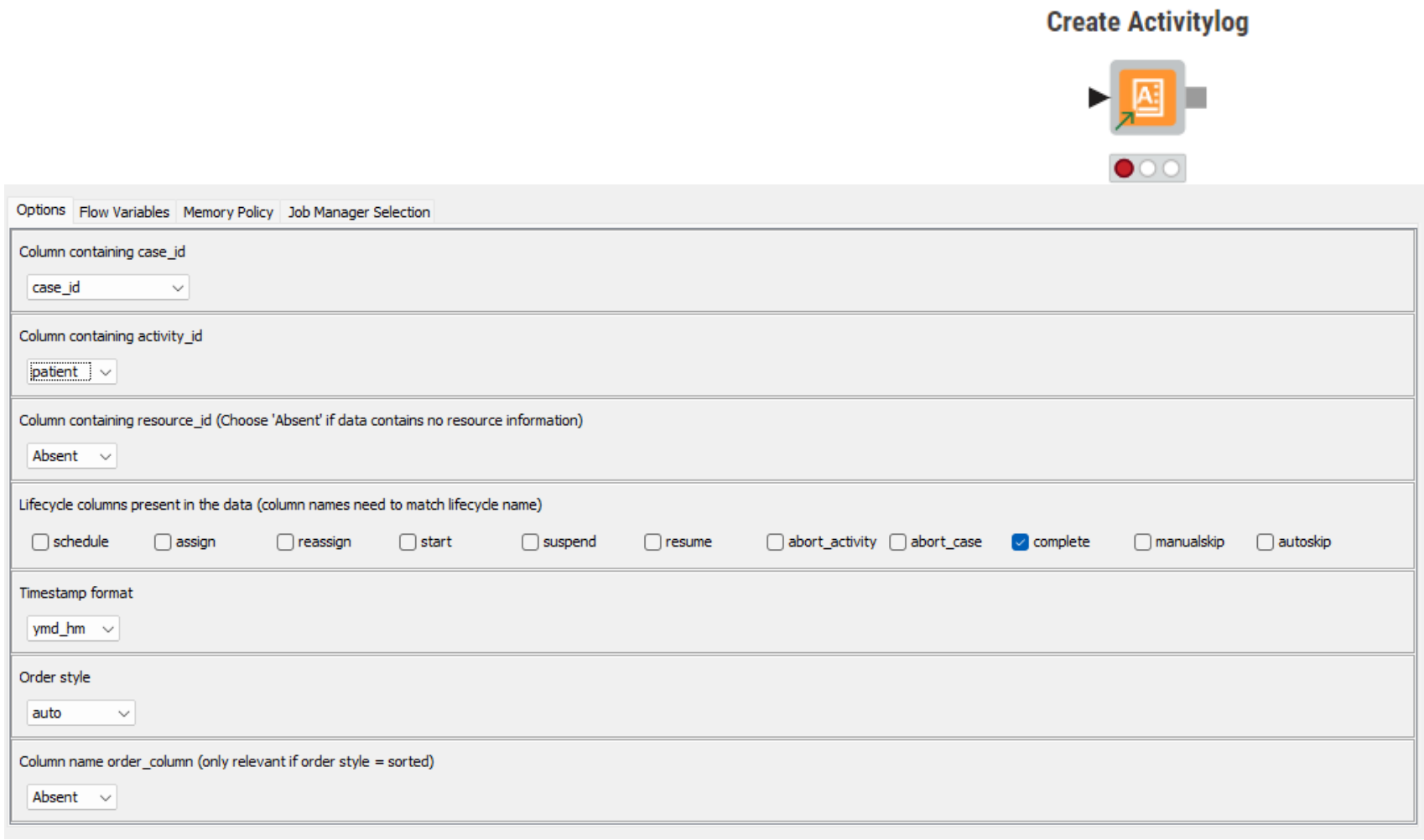


BUPAR

Scenario 1

2. Create activity log by setting the configuration to the correct variables:

- Case identifier
- Activity identifier
- Resource identifier
- Timestamp columns

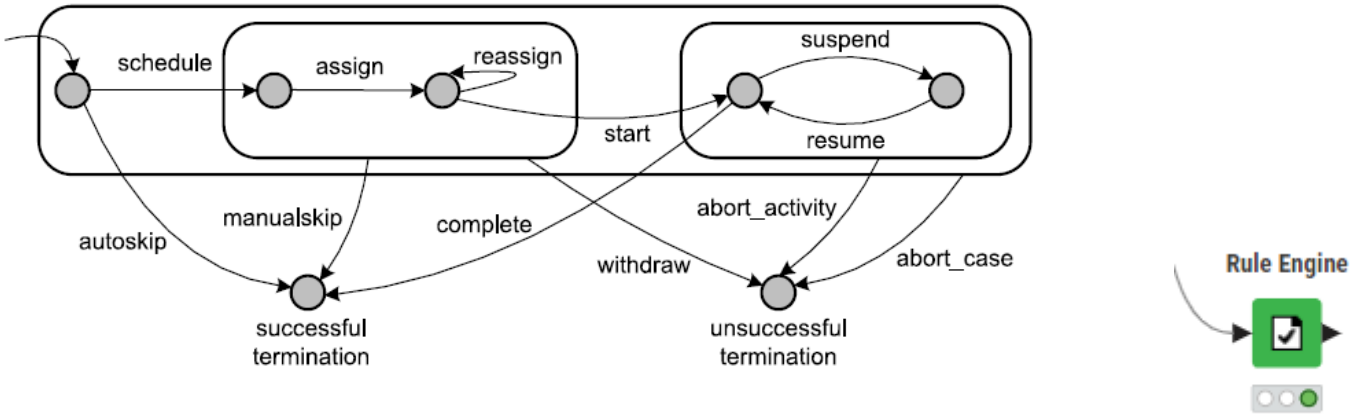


BUPAR

Scenario 2

Each row is an event and events within the same activity instance have the same values.

1. Check whether the lifecycle variable has appropriate values



   If not, adjust with (f.e.) *Rule Engine*

2. Make sure an **activity instance i**d is available that relates event to an activity instance.

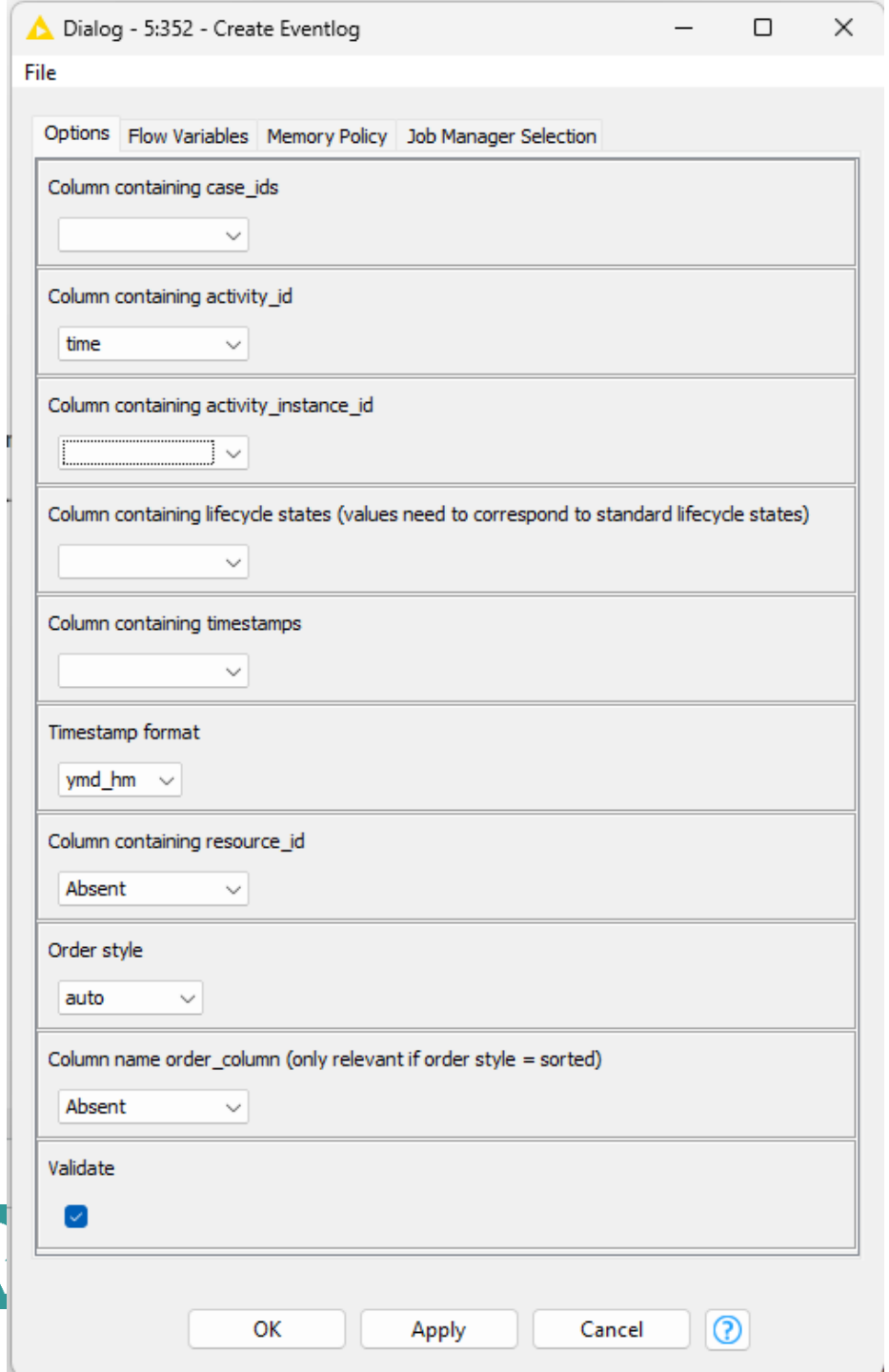
   If not, try to engineer one.

BUPAR

Scenario 2

3. Create event log by setting the configuration to the correct variables:

- Case identifier
- Activity identifier
- Activity instance identifier
- Lifecycle states
- Timestamp column
- Timestamp format
- Resource identifier



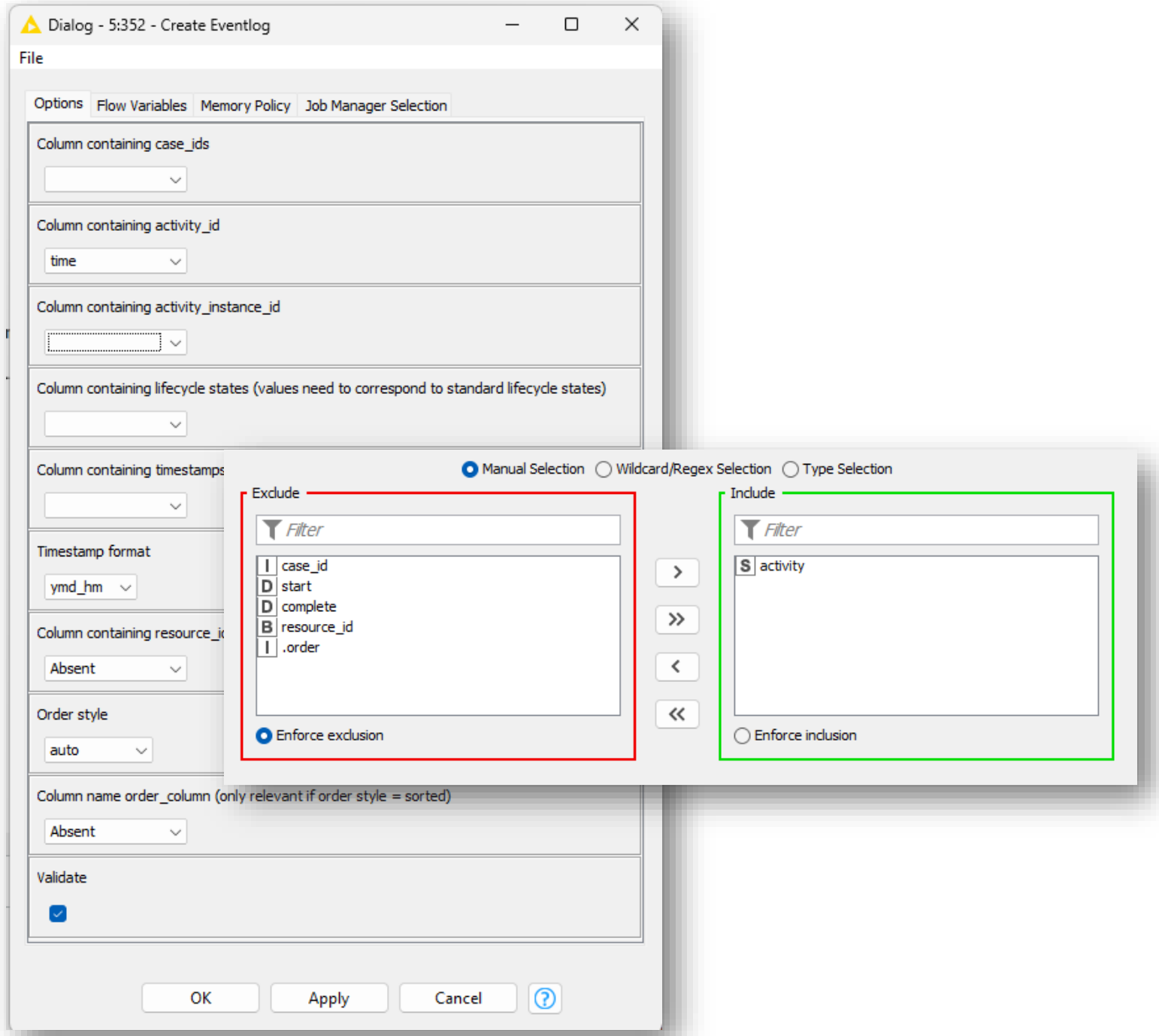
BUPAR

IMPORTANT

Some nodes allow you to chose dataset-specific items such as attribute values or columns from a dropdown menu or using an exclude/include window.

The proper options will not always be shown immediately. You may find incorrect labels or an empty list of options.

In that case, pre-load the configuration pane by already executing the node once before adjusting the configuration



Scenario 2

Optional

Do events from the same activity instance have the same attribute values (resources etc)?

Convert eventlog to activity log
(optional for performance gains)

BUPAR

Scenario 2

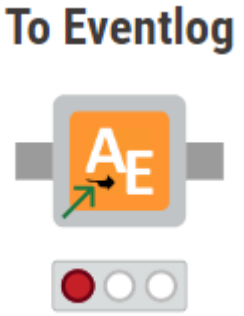
Optional

Do events from the same activity instance have the same attribute values (resources etc)?

Convert eventlog to activity log
(optional for performance gains)

BUPAR

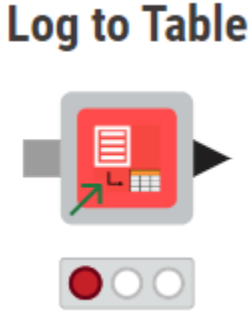
To Event log



Event logs are more flexible than activity logs. Certain methods or analysis might not work for activity logs

In that case, convert to event log

Log to Table



In case you need to do some manipulations to the data that cannot be done by bupaR nodes you can convert the log back to a KNIME Table and us KNIME nodes.

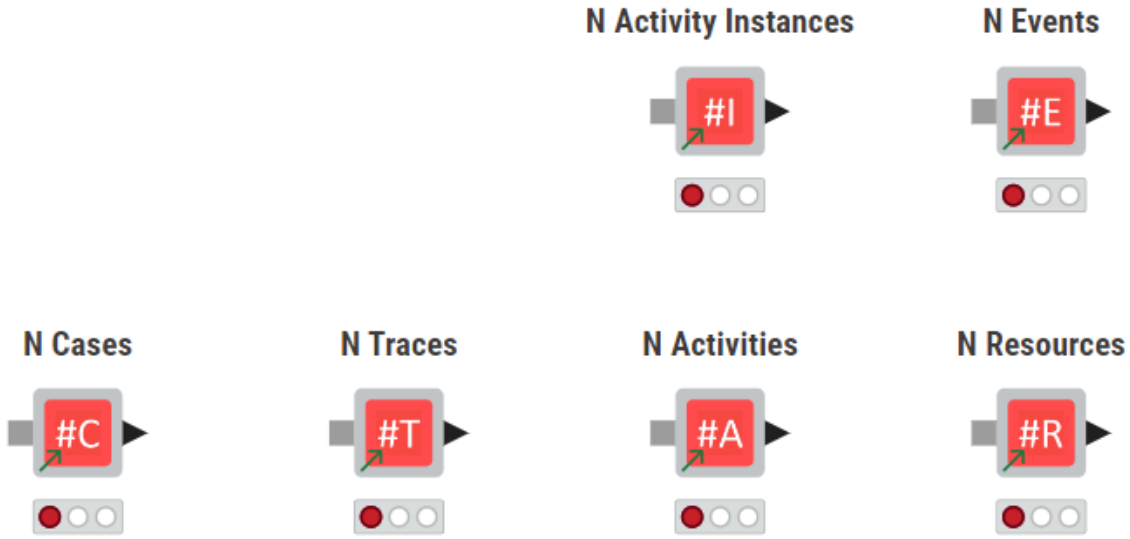
Ultimately, you can always return to a Log format using *Create Activity/Event Log*

BUPAR

Counters

Check log dimensions

Calculcate the number of cases, traces, etc.

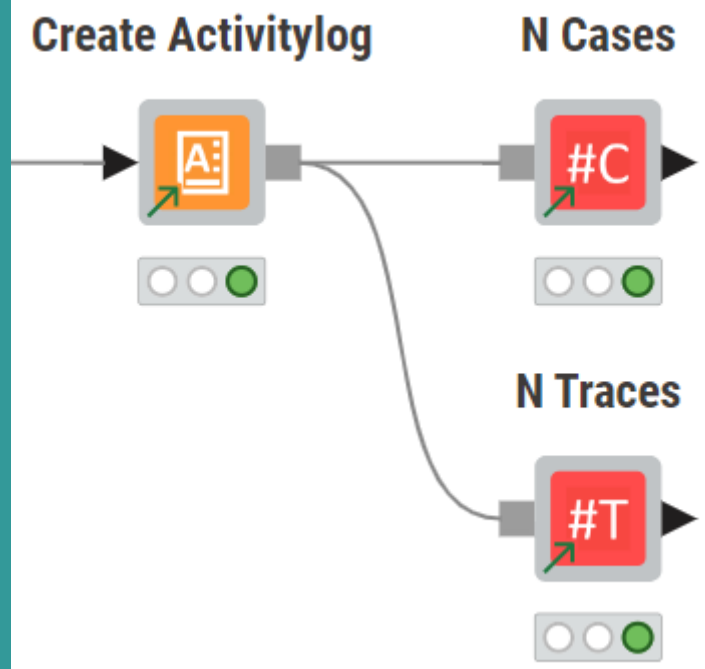


BUPAR

Counters

Check log dimensions

In this example there are 35 cases and 35 traces, i.e. each case is unique.



#	RowID	n_cases *Number (Integer)*
1	1	35

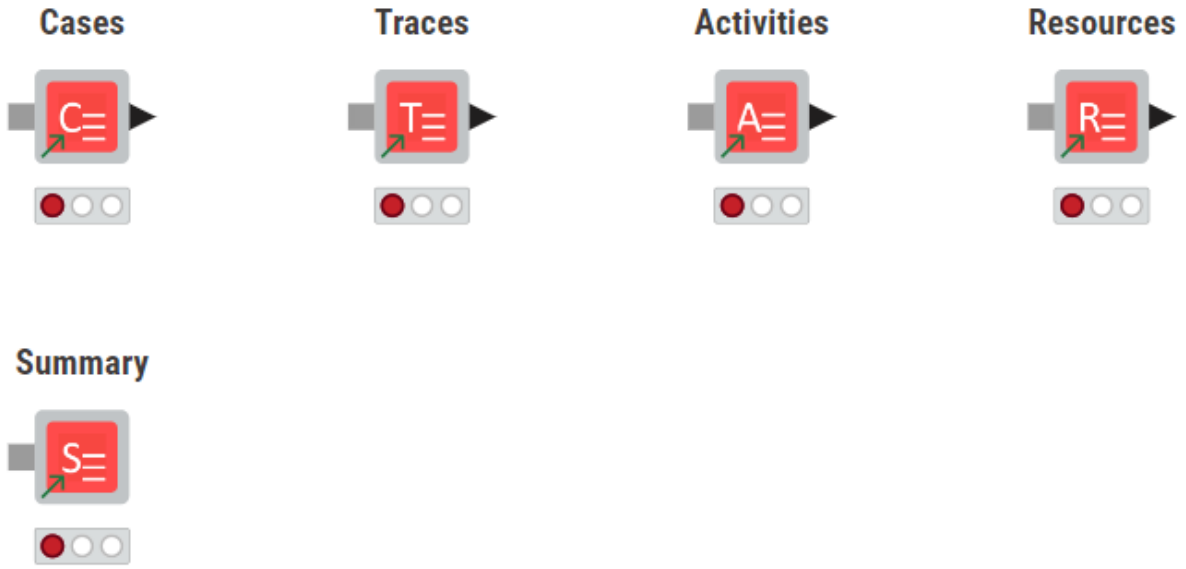
#	RowID	n_traces *Number (Integer)*
1	1	35

BUPAR

Summaries

Descriptive analysis

Descriptive statistics of different process components

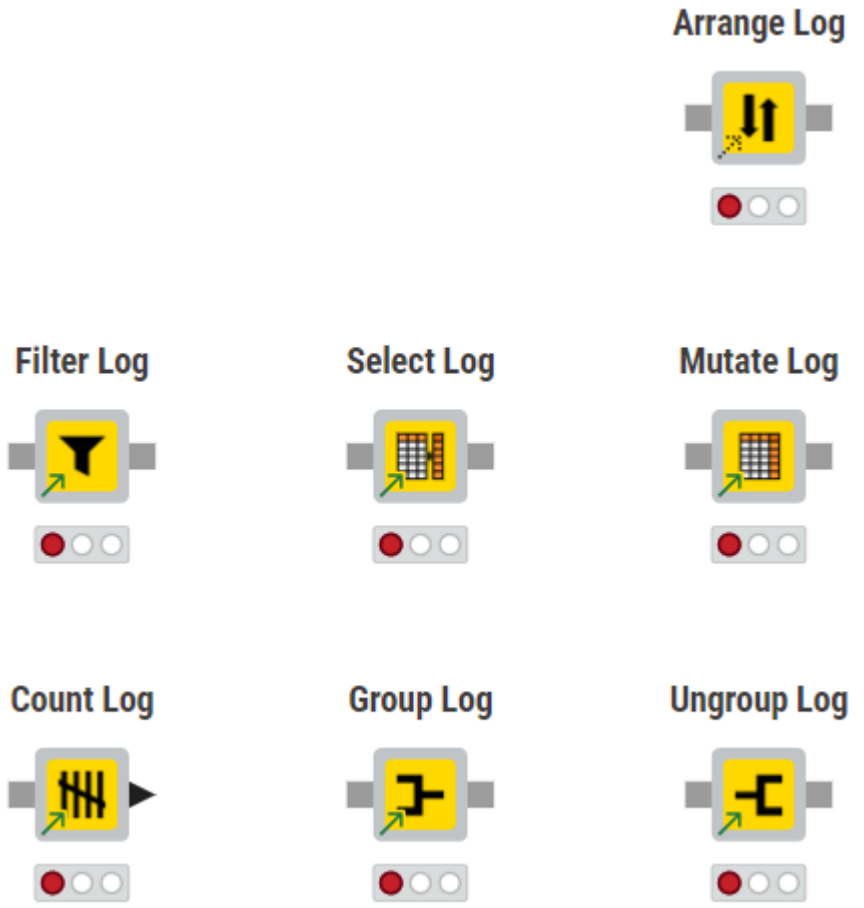


BUPAR

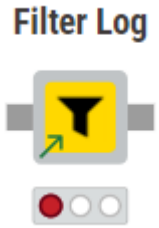
Verbs

Log wrangling

Node for generic log wrangling

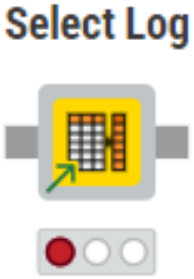


Filter



Remove rows that do not adhere to certain conditions

Select



Select specific columns and remove others.

Will always keep the log-defining columns (case, activity, etc)

Mutate

Add new variables, or make changes to existing variables

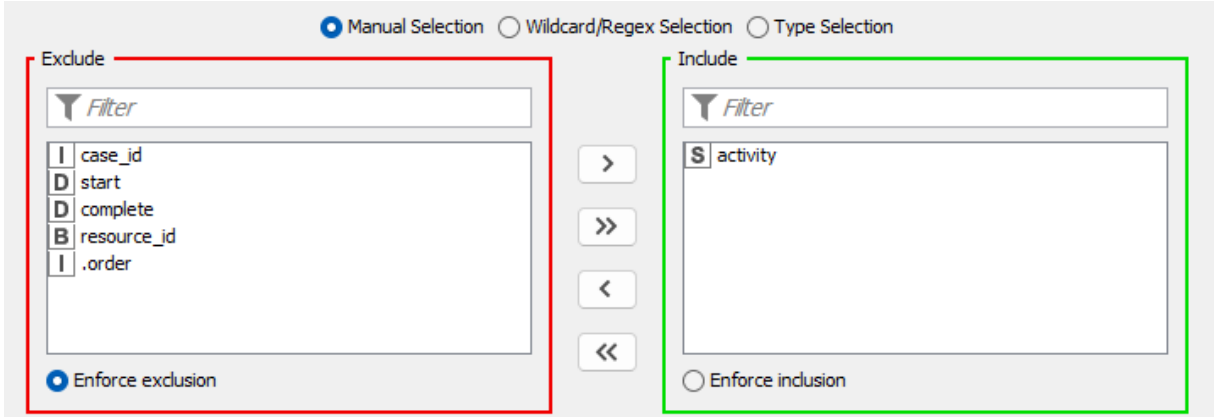
Arrange

Sort log on specified columns

Count

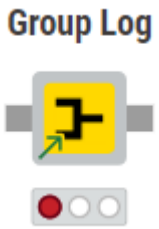
Count occurences of values for any (set of) column(s).

In the example on the right, we count the *activity* column, giving us a list of activity types and their frequency. (I.e. the same we would get if we would use the *Activities* summary node.)



activity *String*	n *Number (Integer)*
Start	35
washing	174
watchingtv	172
toilet	135
sleeping	42
prepareBreakfast	36
eatingBreakfast	34
shower	26
grooming	83

Group



Group the log on a (set of) variables.

Although it will have no immediate effect, any analyses you perform from then onwards will be performed for each group in parallel.

This allows you to perform analyses conditioned on certain variables, which is useful when performing multi-perspective analysis

Use cases

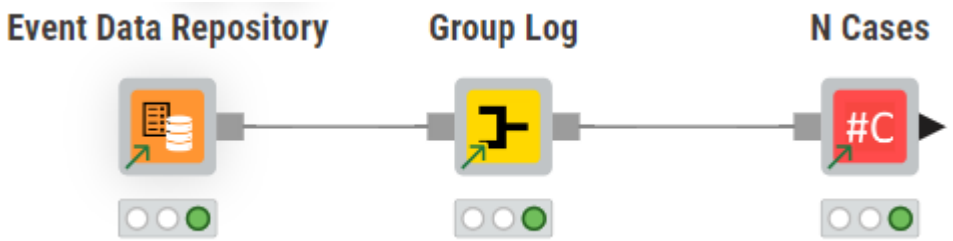
Detailed metrics

```
log ▶ group_by ▶ metric
```

Calculated variables

```
log ▶ group_by ▶ mutate
```

Group – An Example



In this example, the *Traffic Fines* log from the repository was taken and grouped on *VehicleClass.* Subsequently we compute the number of cases, the result of which is shown below.

vehicleclass *String*	n_cases *Number (Integer)*
A	9973
C	21
M	6
?	10000

Ungroup

Remove any groups that might still be attached to a log.

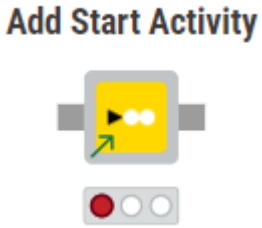
BUPAR

Manipulation

Typical preprocessing tasks

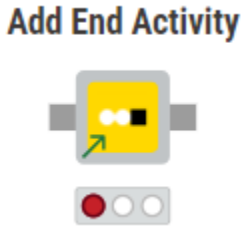
Standard log manipulations

Add Start Activity



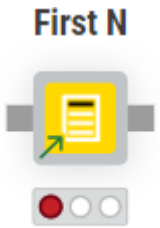
Adds a starting activity instance before the start of each case

Add End Activity



Adds an ending activity instance before the end of each case

First N



Select first n activity instances in the log

Last N

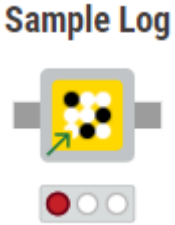
Select last n activity instances in the log

Slice cases

Take a slice of cases from the log, e.g. the cases 5 till 10.

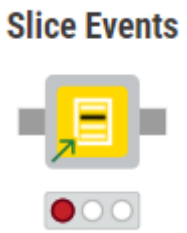
Using the order in the current log

Sample Log



Take a random sample of n cases

Slice events



Take a slice of events from the log, e.g. the events 5 till 10.

Using the order in the current log

Slice activities

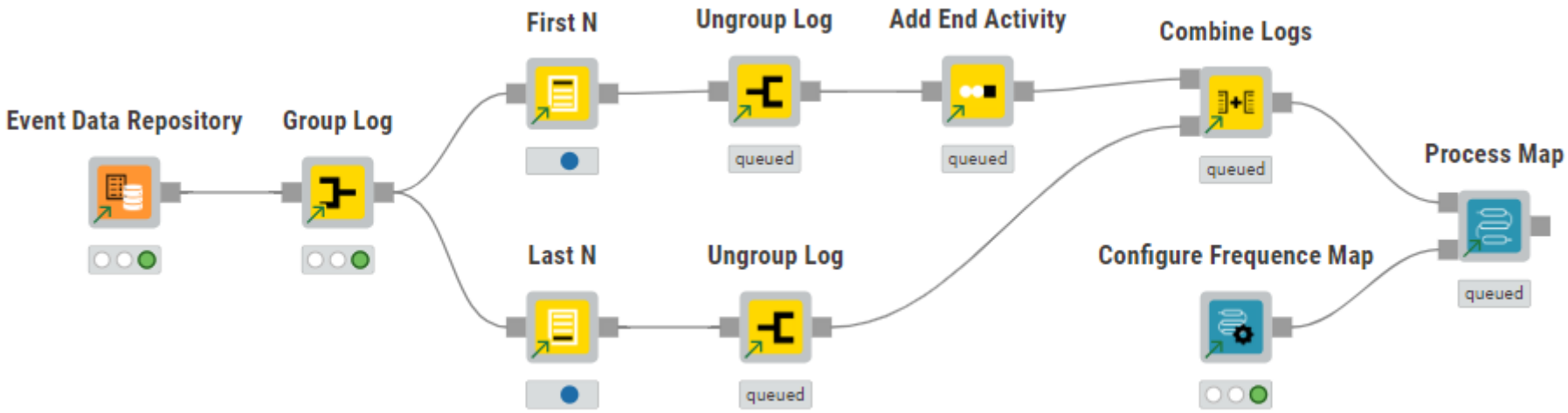
Take a slice of activity instances from the log, e.g. the instances 5 till 10.

Using the order in the current log

Combine Logs

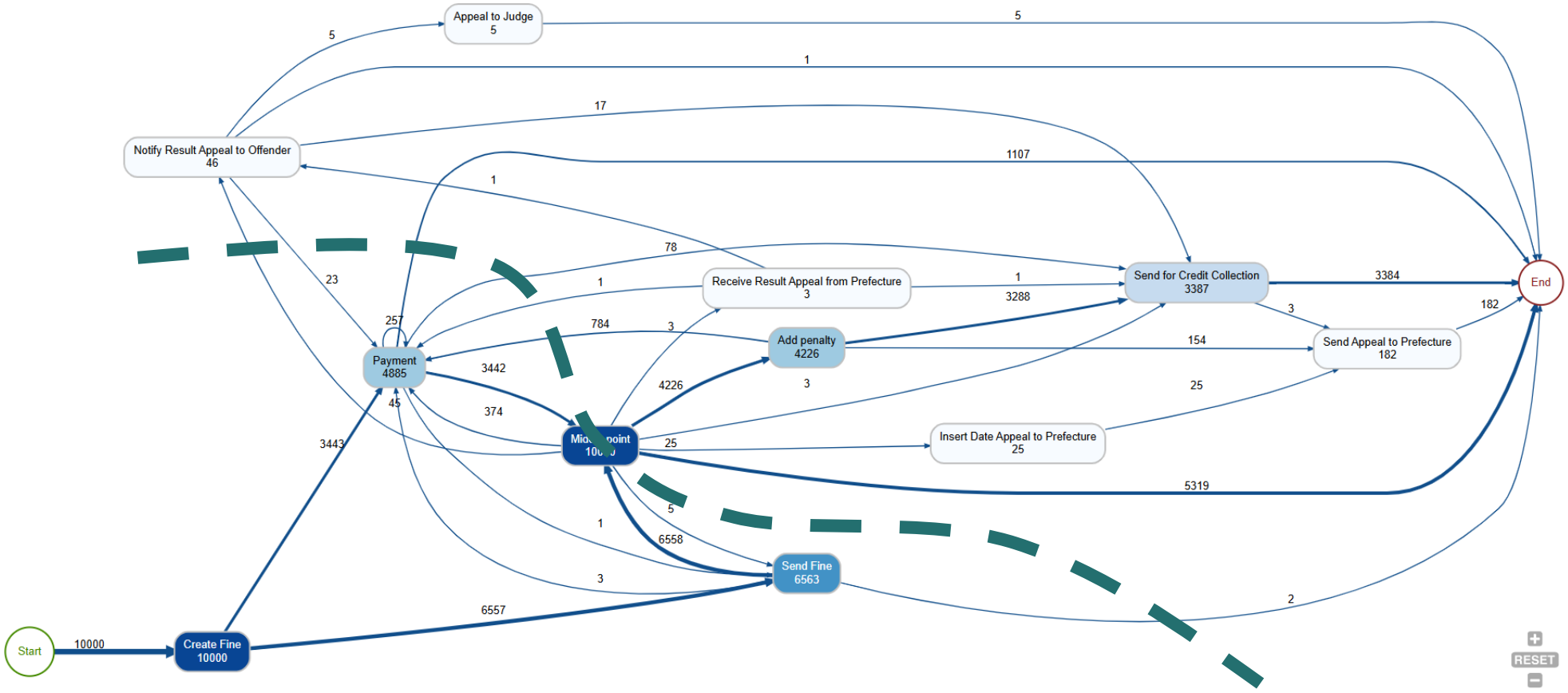
Combine 2 logs that have same configurations (e.g. case, activity, etc.)

Example



In this workflow, we take the *Traffic Fines* log from the repository. We group this log by case, and then take both the first and last three activity instances per case. We then ungroup both subsets of the log. To the first we add an ending activity with the label “Middle” to represent the behaviour occuring between themn. We then combine them together and create a process map (more on process map later in this manual).

Example



The resulting process map focusses on the start and the end of the process, with the *Middle point* activity in between.

eventdataR

BUPAR

BUPAR

eventdataR

Load preconfigured example logs

Load one of the predefined logs that is available in bupaR for testing workflows

- Patients
- Patients activitylog
- Hospital
- Hospital_billing
- Traffic Fines
- Sepsis

edeaR

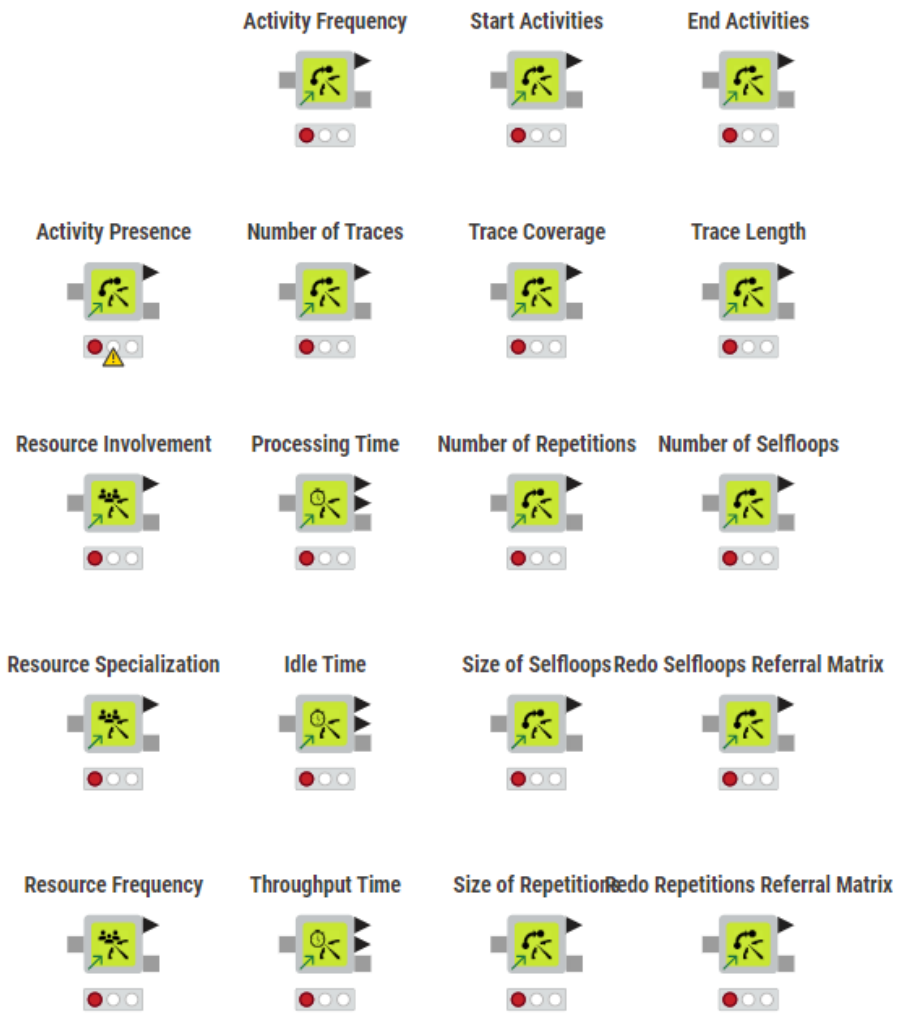
Metrics / Plot/ Augment / Filters

BUPAR

BUPAR

Metrics

Analyse the log



Control-flow

Metrics have different **levels of measurement**.

Check the documentation of the node to see what each available level computes

	Measurement levels					
	Log	**Trace**	**Case**	**Activity**	**Resource**	**Resource Activity**
Activity frequency	■	■	■	■		
Activity presence						
Start activities	■		■	■	■	■
End activities	■		■	■	■	■
Number of repetitions	■		■	■	■	■
Size of repetitions	■		■	■	■	■
Number of selfloops	■		■	■	■	■
Size of selfloops	■		■	■	■	■
Number of traces						
Trace length	■	■	■			
Trace coverage	■	■	■			

Organisational

Metrics have different **levels of measurement**.

Check the documentation of the node to see what each available level computes

	Measurement levels					
	Log	**Trace**	**Case**	**Activity**	**Resource**	**Resource Activity**
Resource frequency	■		■	■	■	■
Resource involvement			■		■	■
Resource specialization	■			■	■	

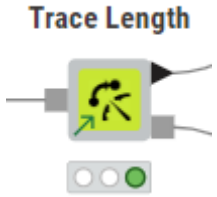
Performance

Metrics have different **levels of measurement**.

Check the documentation of the node to see what each available level computes

	Measurement levels					
	Log	**Trace**	**Case**	**Activity**	**Resource**	**Resource Activity**
Processing time	■	■	■	■	■	■
Throughput time	■	■	■			
Idle Time	■	■	■		■	

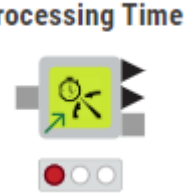
Metric Syntax



Metrics take a (grouped) log as input

They output

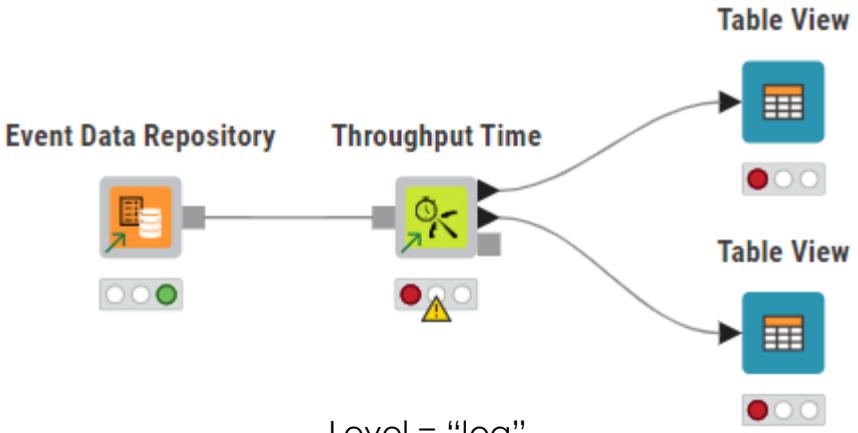
1. The metric as a KNIME Table
2. The metric as an R Table



Some metrics have **two** KNIME Table outputs

1. Summary statistics
2. Raw metric data

Example



Level = “log”

The raw output is only available for performance metrics, and is most suitable for creating custom visualisations with KNIME

Rows: 1 | Columns: 8

RowID	min Number (Float)	q1 Number (Float)	median Number (Float)	mean Number (Float)	q3 Number (Float)	max Number (Float)	st_dev Number (Float)	iqr Number (Float)
1	0.034	15.443	128.241	683.264	431.918	10,135.775	1,452.897	416.475

Rows: 1050 | Columns: 2

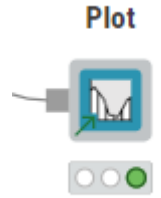
RowID	case_id String	throughput_time Number (Float)
1	A	267.989
2	B	126.927
3	C	134.509
4	D	437.745
5	E	0.35
6	F	40.051
7	G	163.32
8	H	111.166

BUPAR

Plot

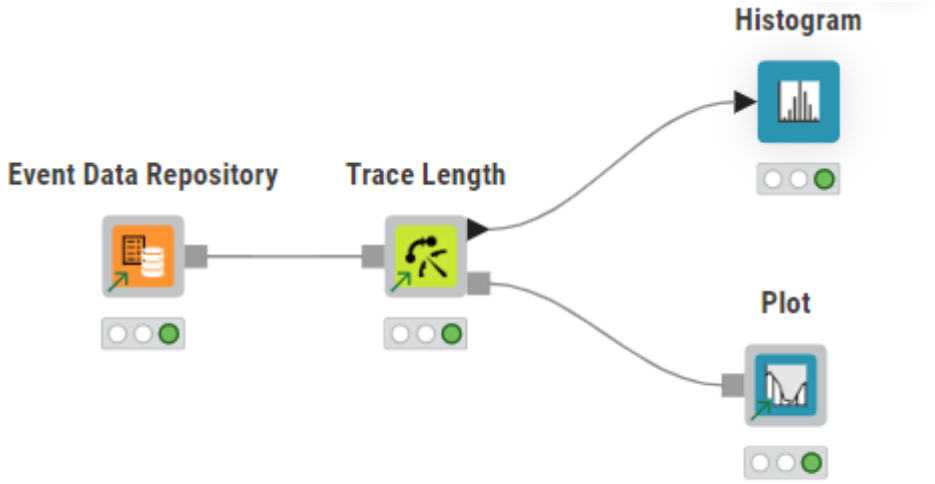
Standard metric visualisations

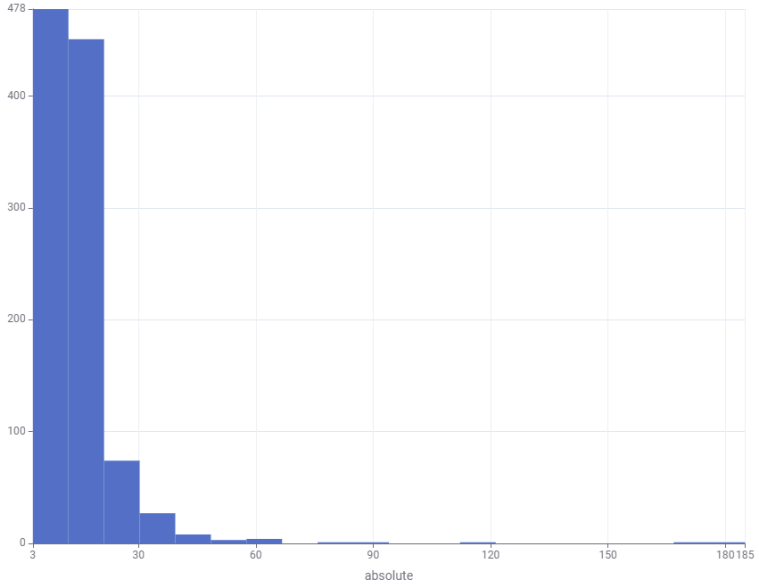
Most metrics have a standard visualization that can be shown with the Plot node.



Alternatively you can use the KNIME Table output of a metric to create your own visualisations native with KNIME

Example

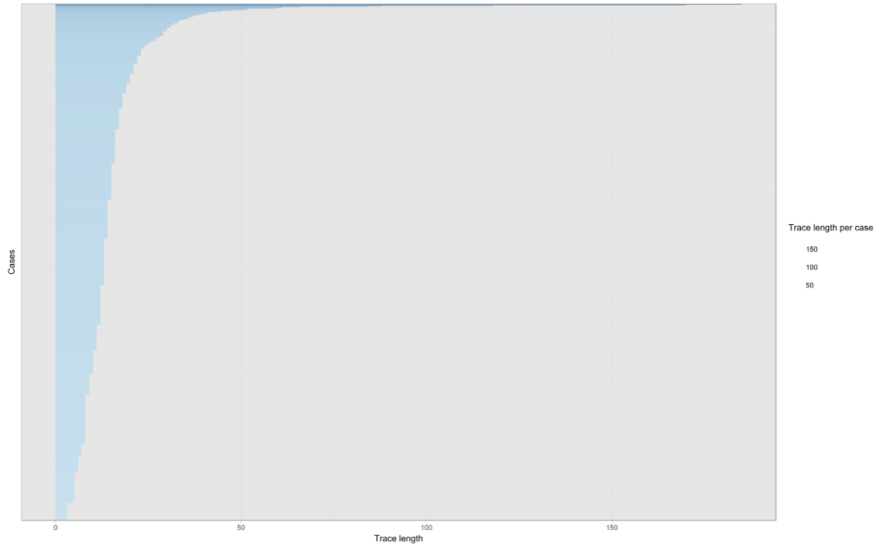




Or make a histogram of trace length with KNIME

This approach is more flexible and allows interactivity

We can go with the default plot that shows the trace length for all cases



BUPAR

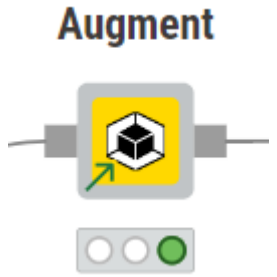
Augment

Enrich the log

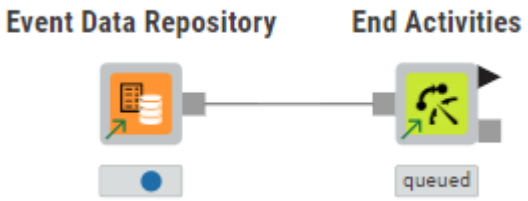
The R output of a metric can be used to augment the log.

This means that the computed metric is added to the log to enrich the data.

The newly available attributes can be used in further analyses



Example



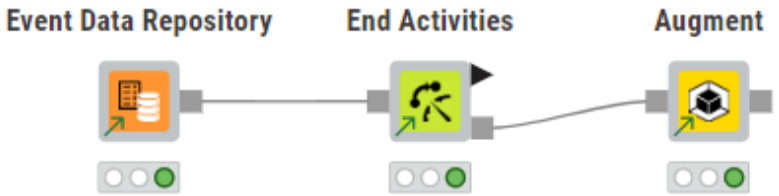
Rows: 10000 | Columns: 2

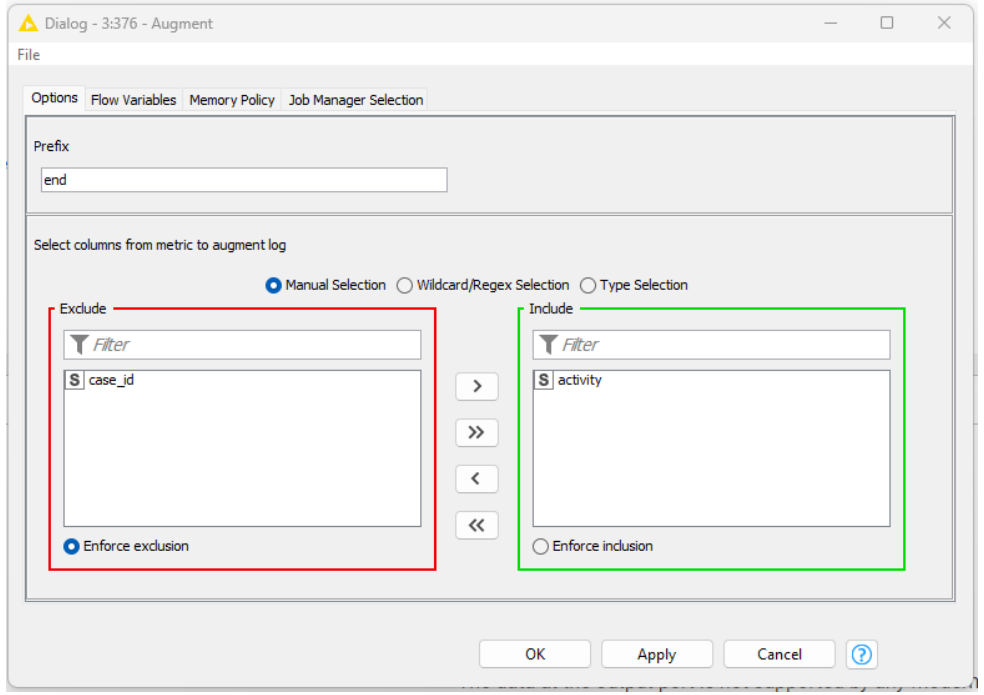
Table | Statistics

	#	RowID	case_id *String*	activity *String*
☐	1	1	A1	Send Fine
☐	2	2	A100	Send for Credit Collection
☐	3	3	A10000	Payment
☐	4	4	A10001	Send Appeal to Prefecture
☐	5	5	A10004	Send for Credit Collection
☐	6	6	A10005	Payment
☐	7	7	A10007	Payment
☐	8	8	A10008	Send for Credit Collection
☐	9	9	A10009	Payment
☐	10	10	A1001	Send for Credit Collection

For *Traffic Fines*, we find the last activity for each case (level = case)

Example

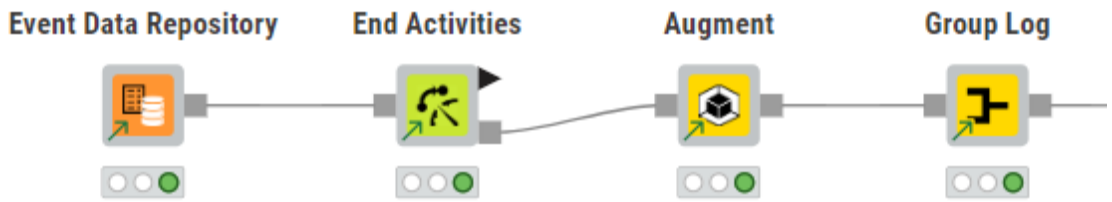


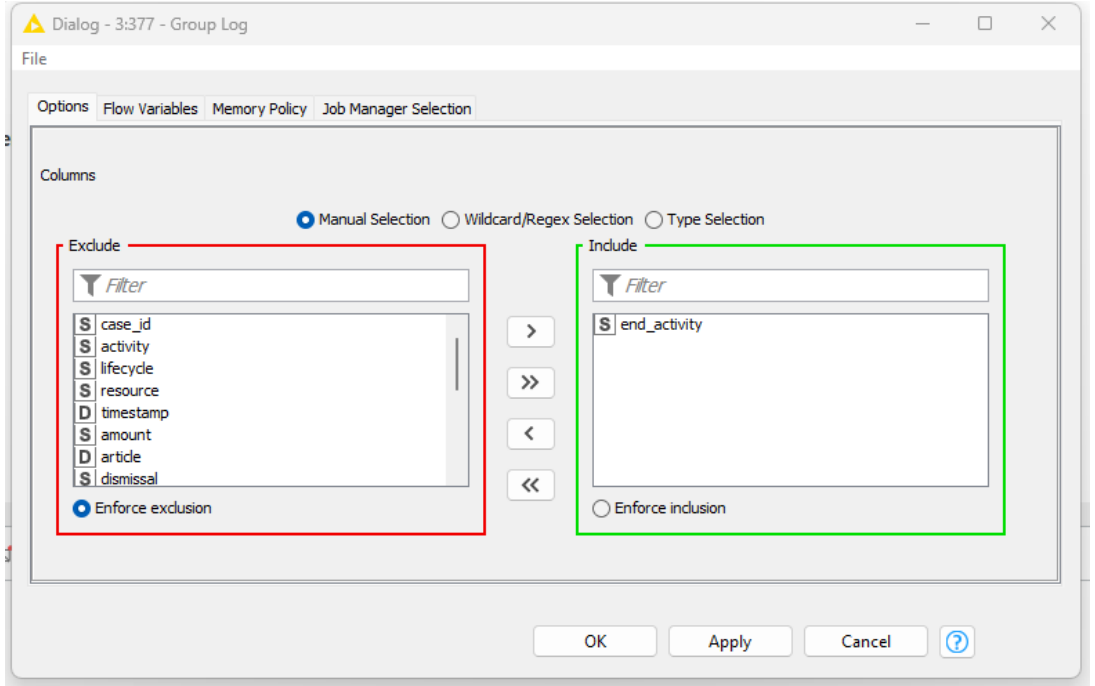


For *Traffic Fines*, we find the last activity for each case (level = case)

From this metric, we take the activity column to augment to log, prefixing the new column with 'end'.

Example



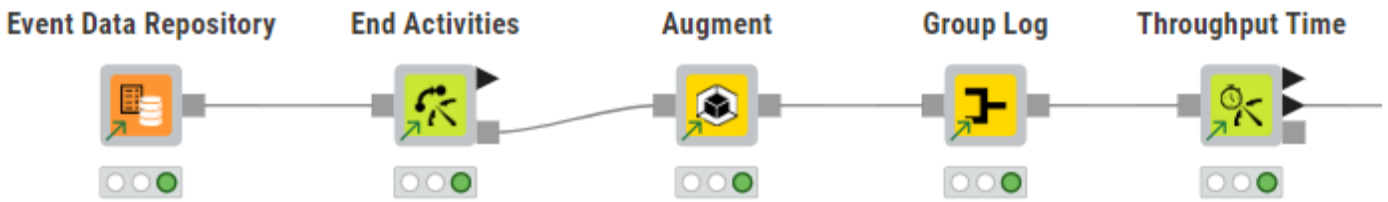


For *Traffic Fines*, we find the last activity for each case (level = case)

From this metric, we take the activity column to augment to log, prefixing the new column with 'end'.

We group the augmented log on the new 'end activity' column

Example

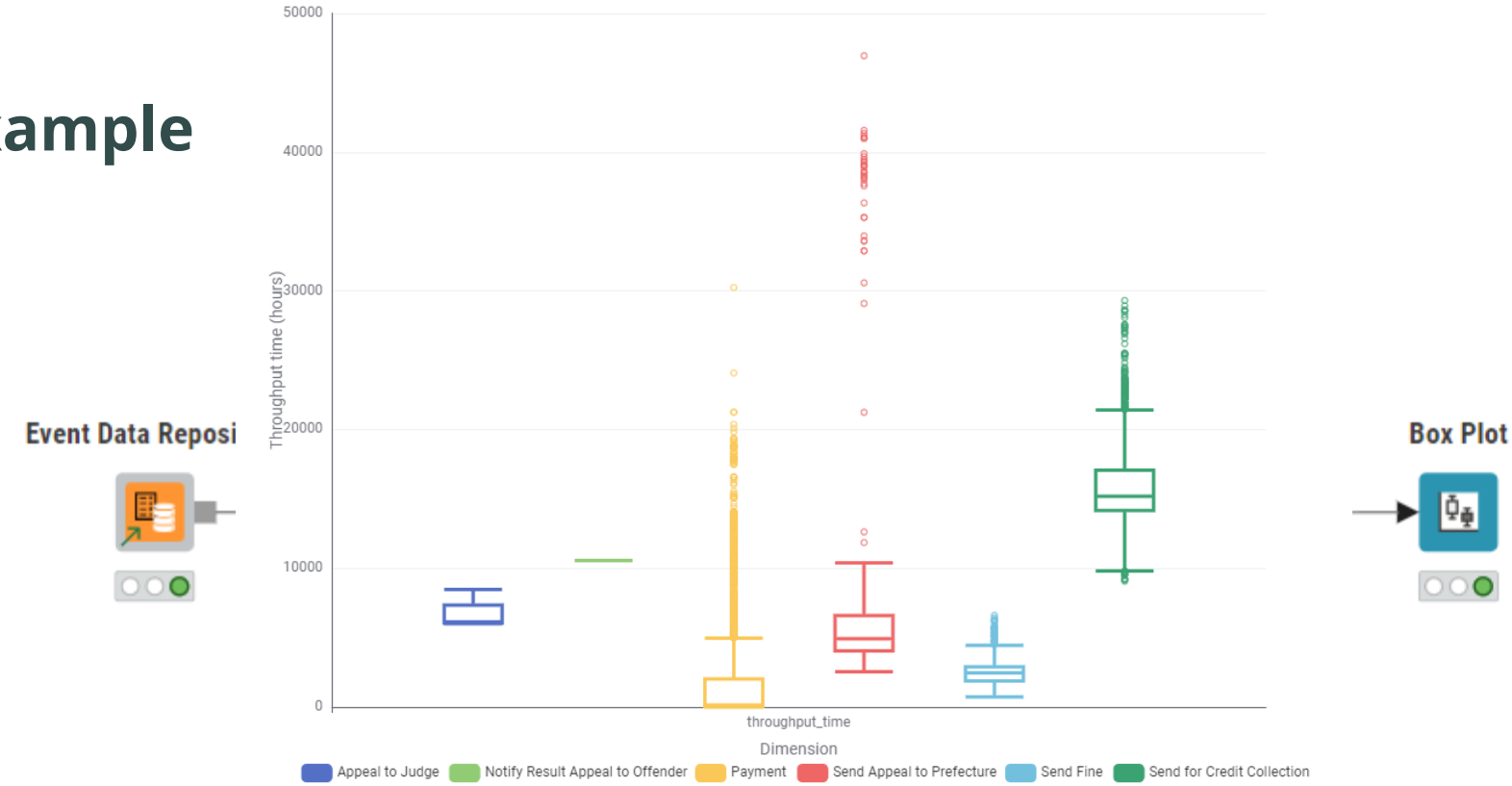


From this metric, we take the activity column to augment to log, prefixing the new column with ‘end’.

We group the augmented log on the new ‘end activity’ column

We compute the throughput time (log level) on the grouped augmented log

Example



We group the augmented log on the new ‘end activity’ column

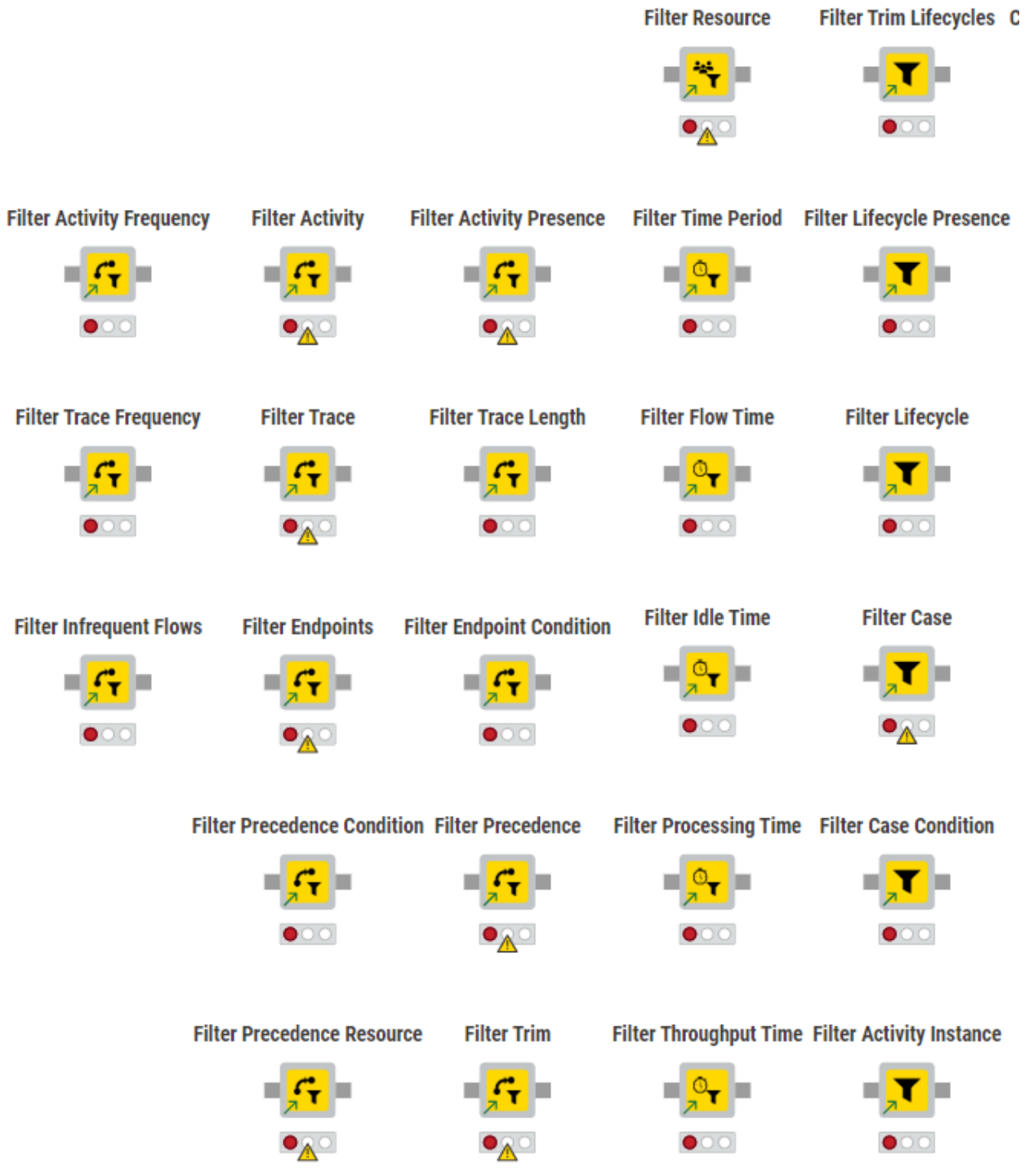
We compute the throughput time (log level) on the grouped augmented log

We visualize the result with a boxplot, allowing us the see the different distribution of throughput time for each end state

BUPAR

Filters

Filter the log



Control-Flow

Filter Activity Frequency

Filter Activity

Filter Activity Presence

Filter Trace Frequency

Filter Trace

Filter Trace Length

Filter Infrequent Flows

Filter Endpoints

Filter Endpoint Condition

Filter Precedence Condition

Filter Precedence

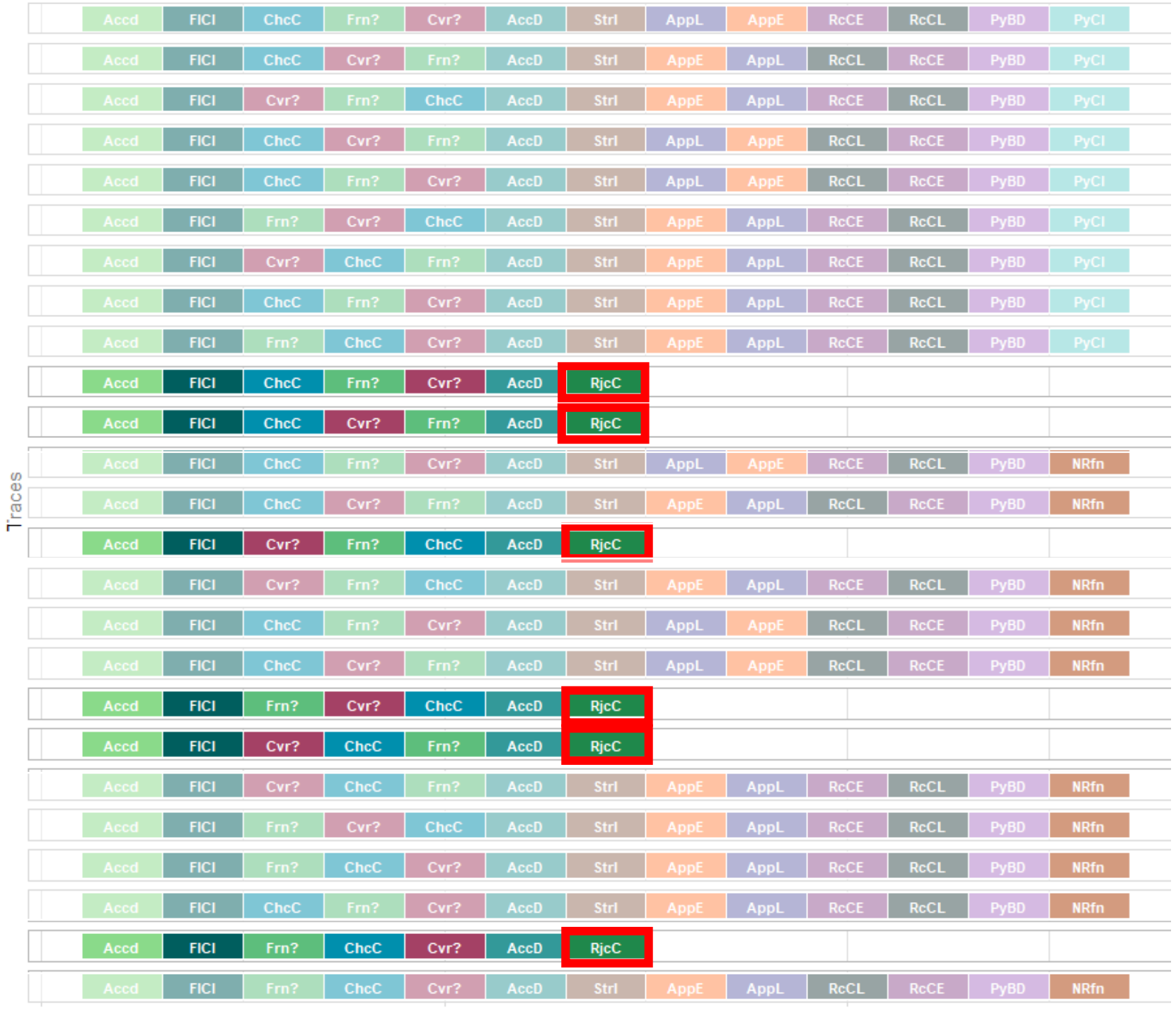
Filter Precedence Resource

Filter Trim

BUPAR

Filter Activity Presence

Selects cases where a(n) (set of) activity(ies) occurs. F.e. Reject Claim



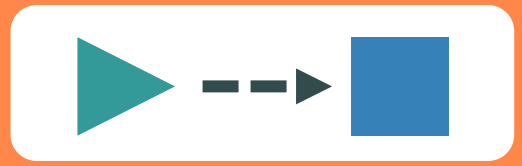
Activity presence

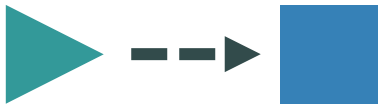
If more than one activity listed, use **`method`**

"all"	each of the activities must occur
"one_of"	at least one must occur
"none"	none must occur

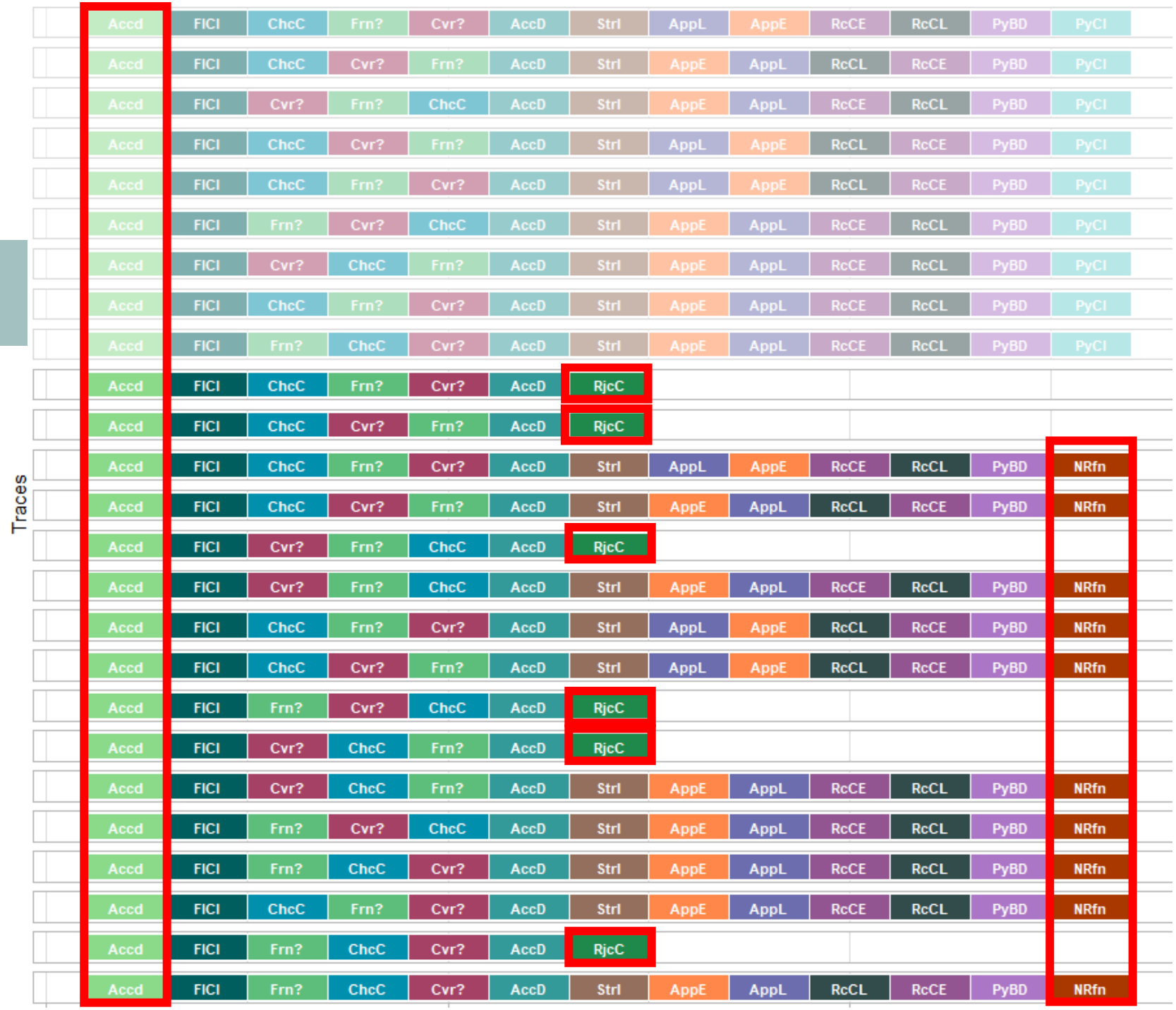
BUPAR

Filter Endpoints



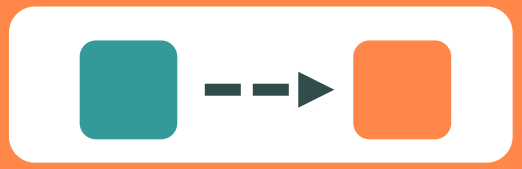


Select cases with specific start and/or end activity



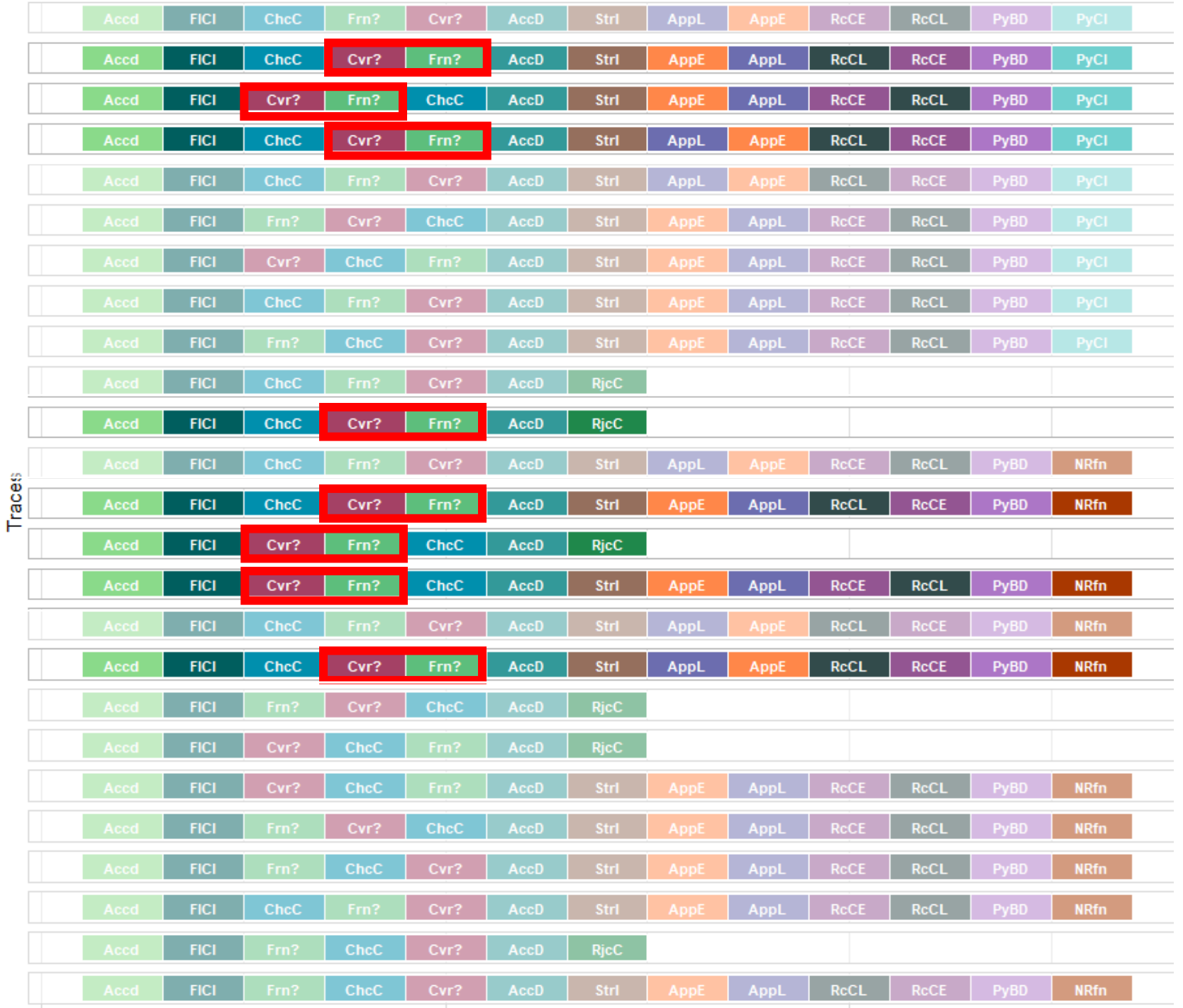
BUPAR

Filter Precendence



Select cases where activities are followed directly or eventually by each other

F.e. Coverage and Franchise directly follows



BUPAR

Filter Trace Frequency

Trace frequency

Select cases based on frequency of variant

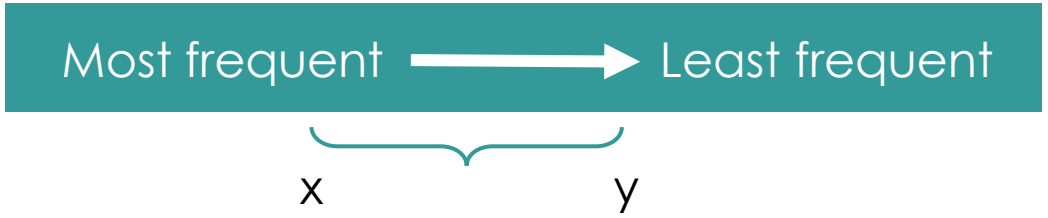
Two approaches

Favor most (in)frequent traces

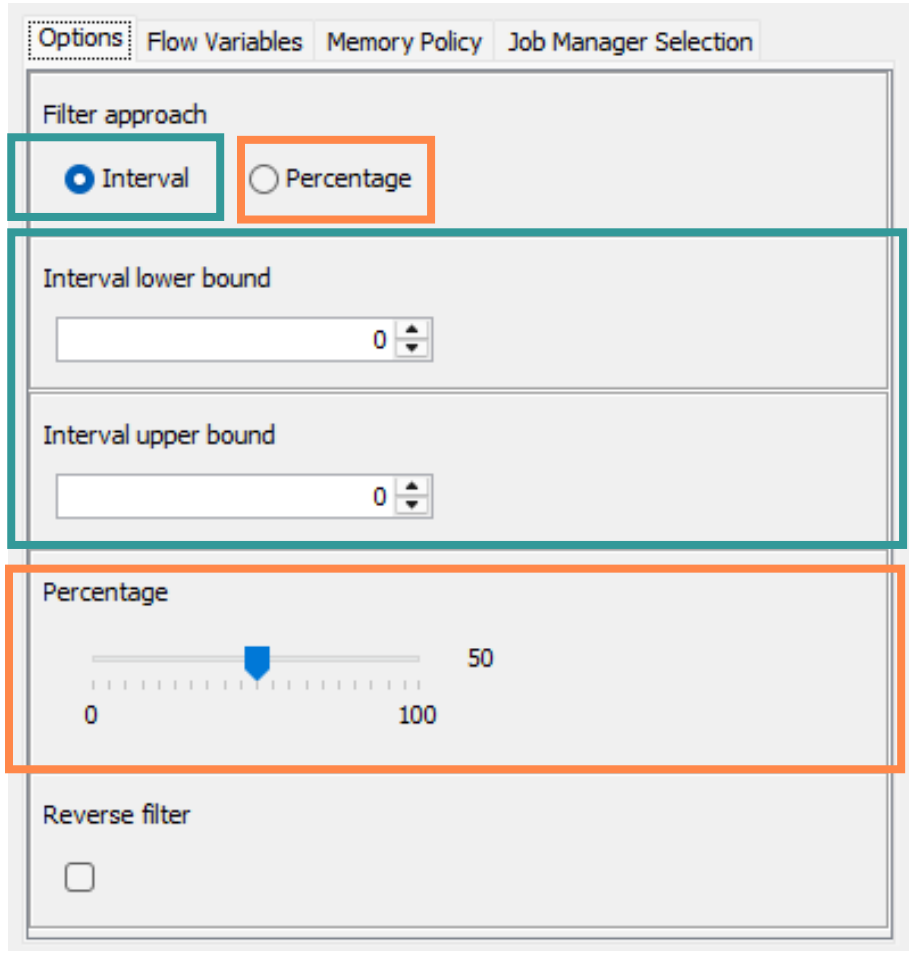
'percentage' approach

Select specific trace frequencies

'interval' approach



Trace frequency



BUPAR

Filter Infrequent flows

Trace frequency vs Flow frequency

Traces

↓

Global

Infrequent = end-to-end sequence is infrequent, but activity-to-activity-components might not be infrequent

Flows

↓

Local

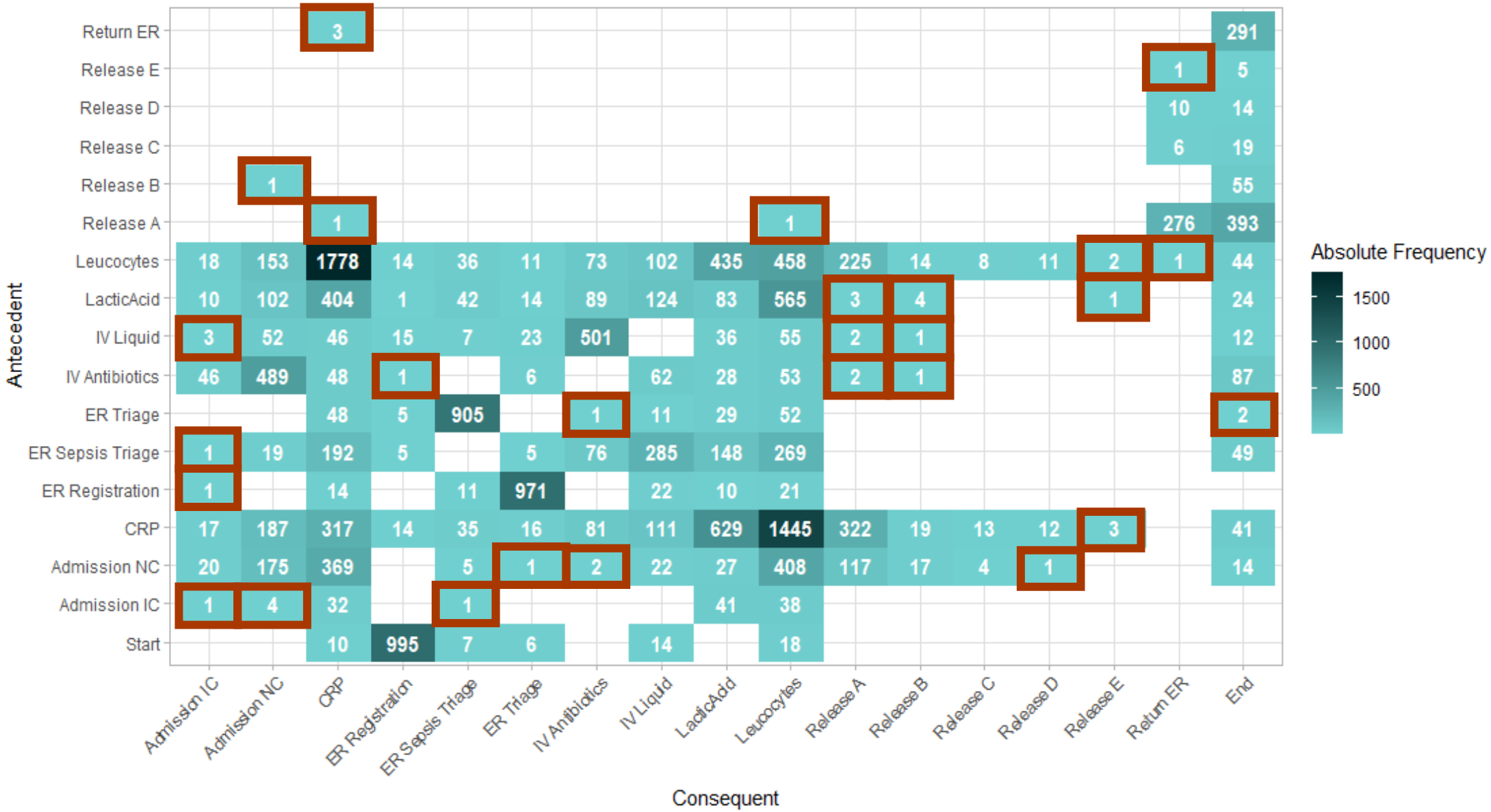
Infrequent = activity-to-activity flow is infrequent

Trace frequency vs Flow frequency

Infrequent traces are not necessarily a problem for a processmap

Every flow in an infrequent trace might be frequent, because it occurs in other traces as well. Removing that infrequent trace, will have a minor impact on a process map

Removing infrequent flows has a **direct impact** on the process map

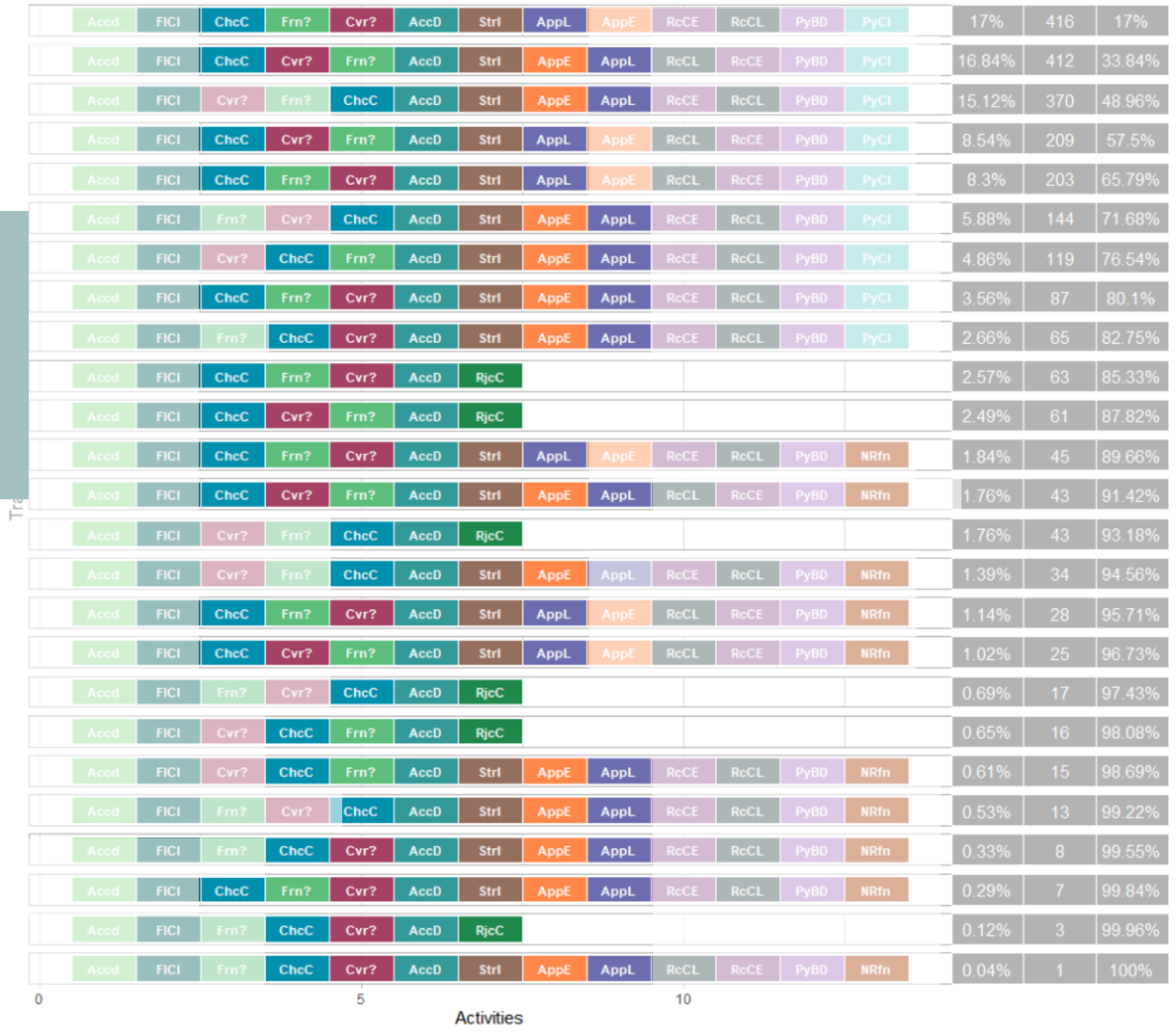


F.e. min flow frequency = 5 > All cases that have one of these infrequent flows are removed

Filter Trim

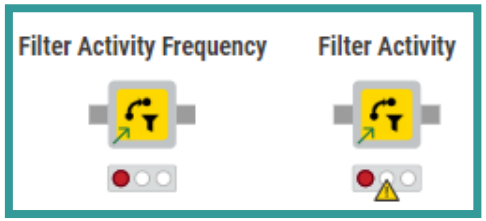
Remove head and/or tails from cases

F.e. from Check Contract to Reject Claim *or* Appoint Lawyer



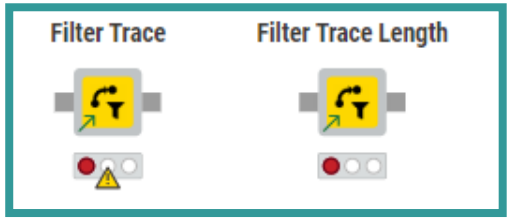
Control-Flow

Filter activities by frequency or name



Filter Activity Presence

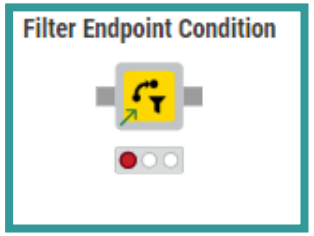
Filter Trace Frequency



Filter traces by id or by length

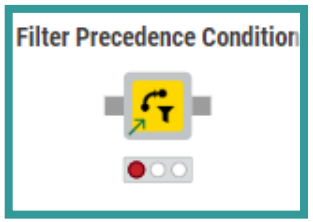
Filter Infrequent Flows

Filter Endpoints



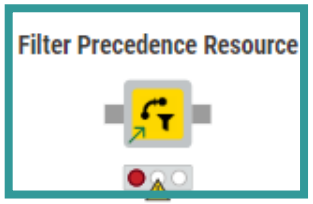
Filter endpoints using logical conditions (using data attributes)

Filter precedence using logical conditions (using data attributes)



Filter Precedence

Filter precedence with identical resource requirement



Filter Trim

Performance

Filter Flow Time

Filter Idle Time

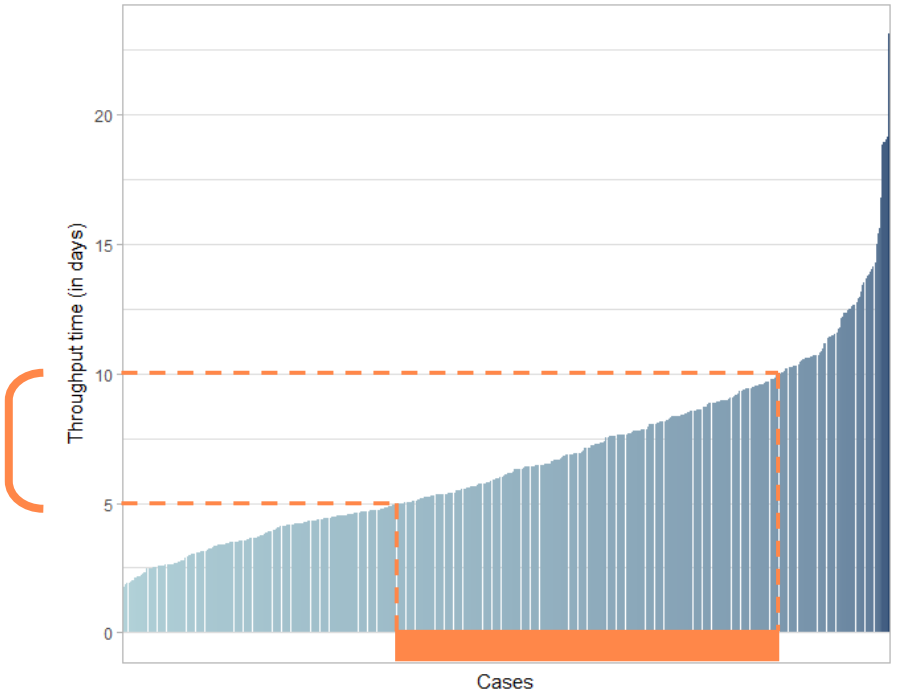
Filter Processing Time

Filter Throughput Time

Idle / Processing / Throughput Time

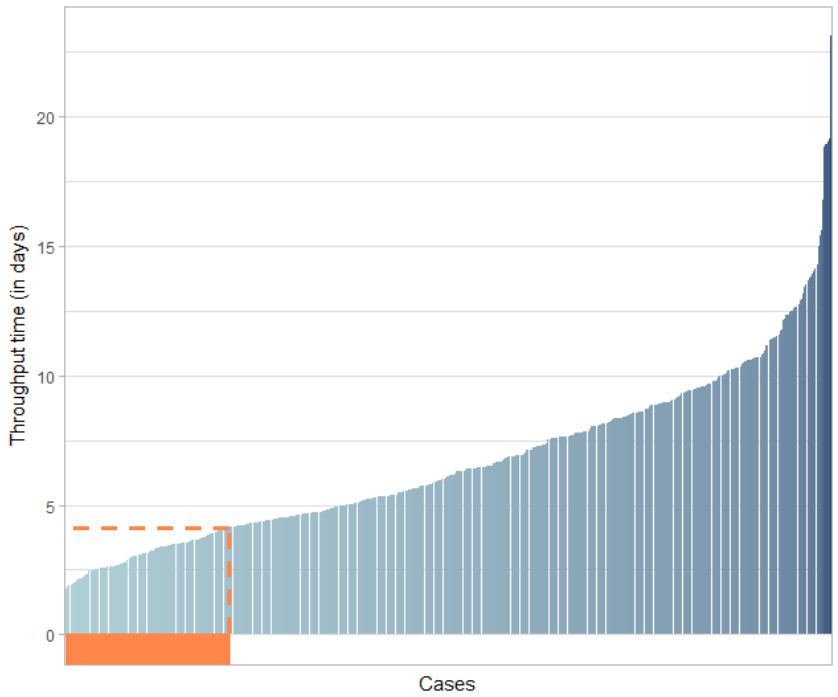
Interval appraoch

E.g. cases with throughput time between 5 and 10 days



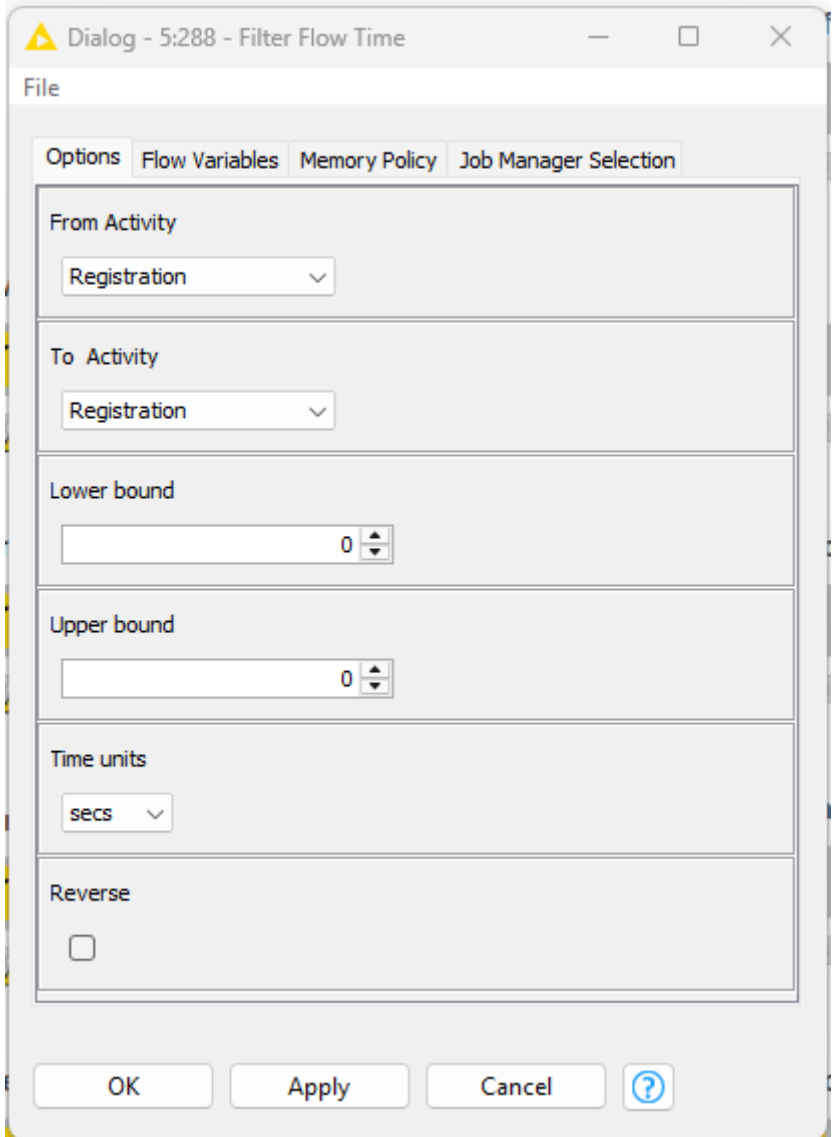
Percentage approach

E.g. 20% shortest cases according to throughput time



Flow time

Specify a directly-follows pair



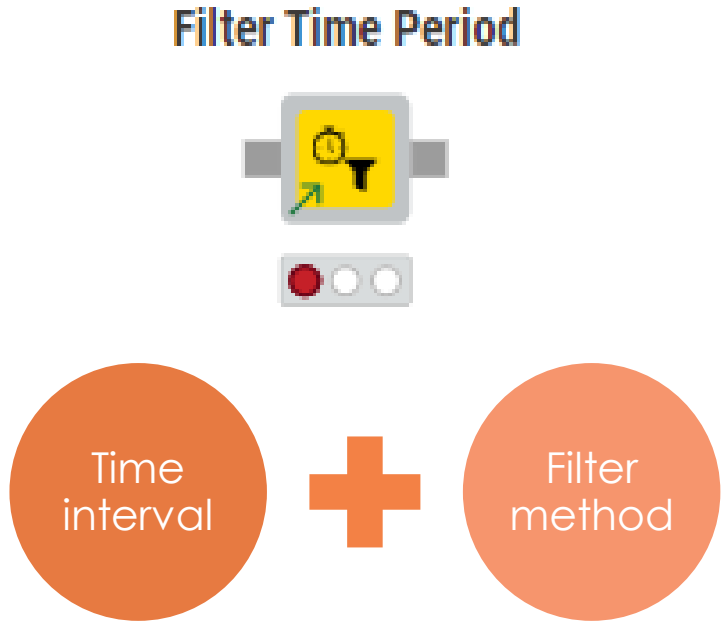
Define an allowed interval of elapsed time

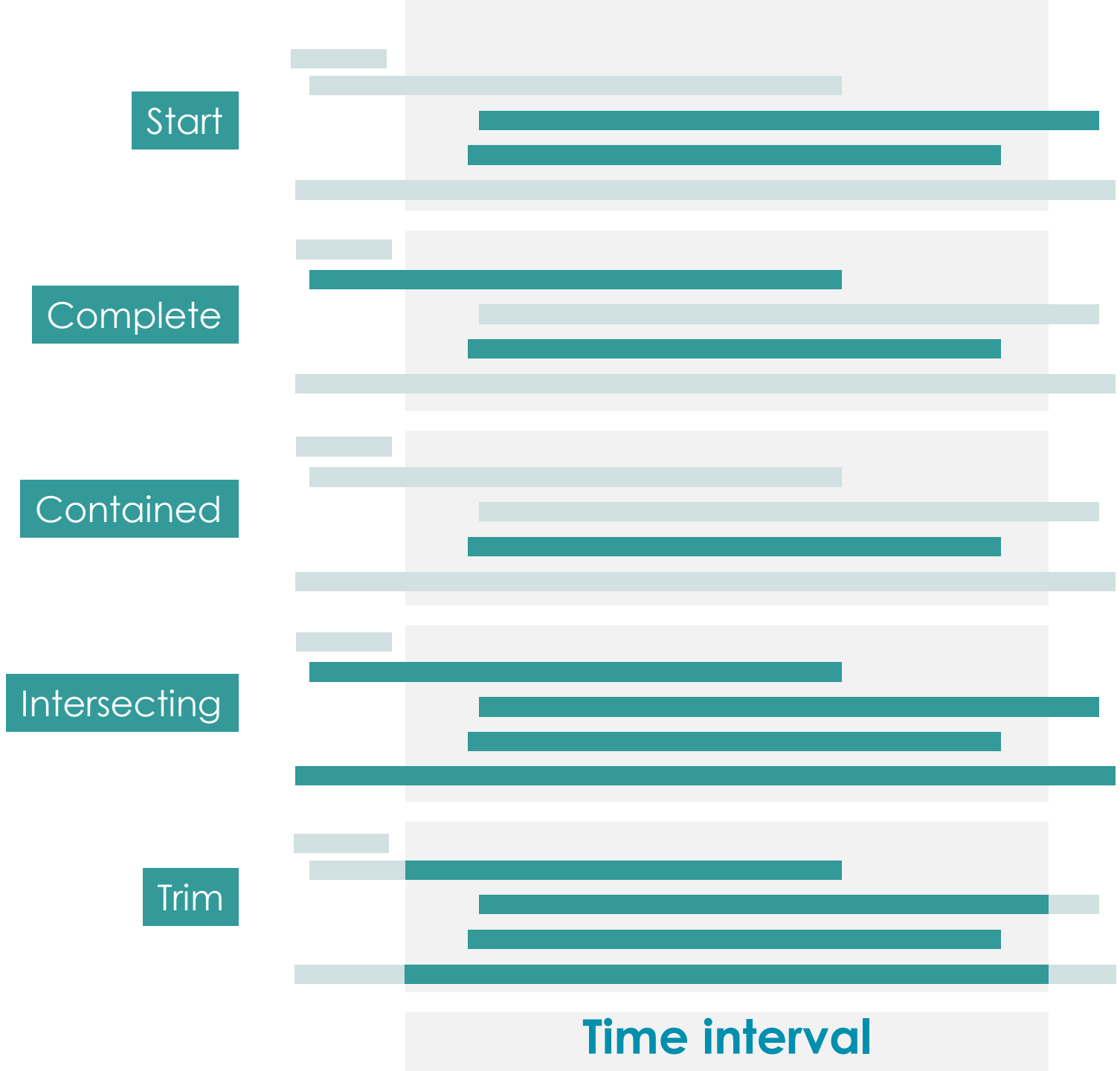
Organisational

Filter resources by frequency (interval or percentage approach)

Filter resources by name

Time




Start
Complete
Contained
Intersecting
Trim
Time interval

Other

Filter Trim Lifecycles

Trim activity instances to within specific lifecycles
(event log only)

Filter Lifecycle Presence

Filter activity instances based on presence/absence of lifecycles

Filter Lifecycle

Fitler lifecycles by name
(event log only)

Filter Case

Fitler cases by id

Filter Case Condition

Filter cases where logical data condition holds

Filter Activity Instance

Filter activity instances by id

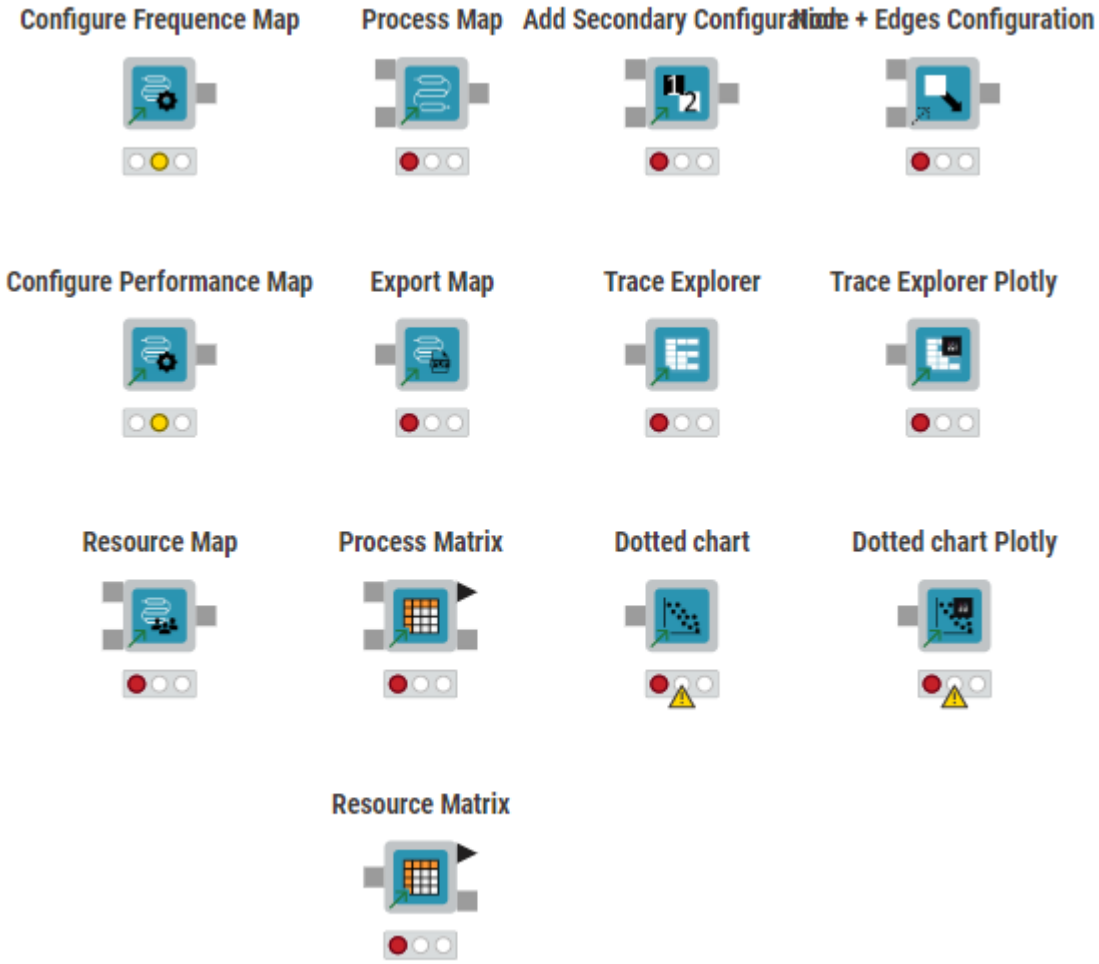
processmapR

BUPAR

BUPAR

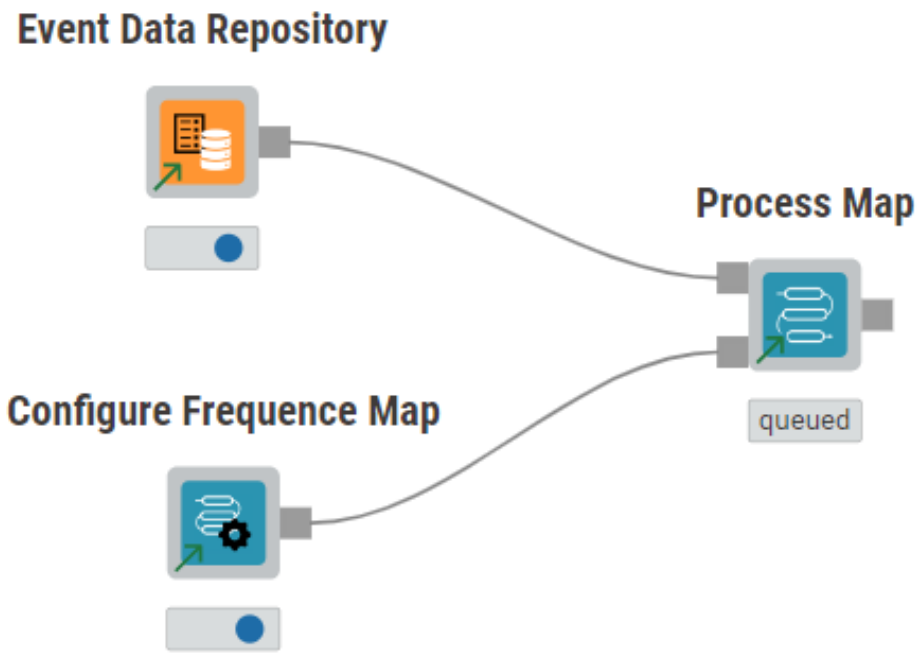
processmapR

Specialized process visualisations



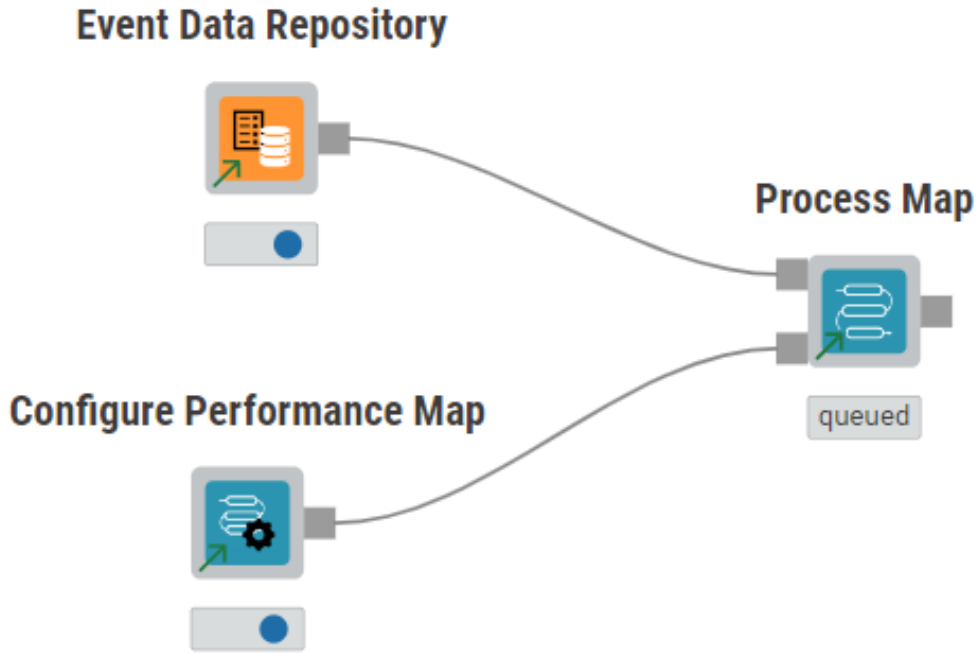
Process maps

Frequency map



A process map always needs a configuration as the second input

Performance map

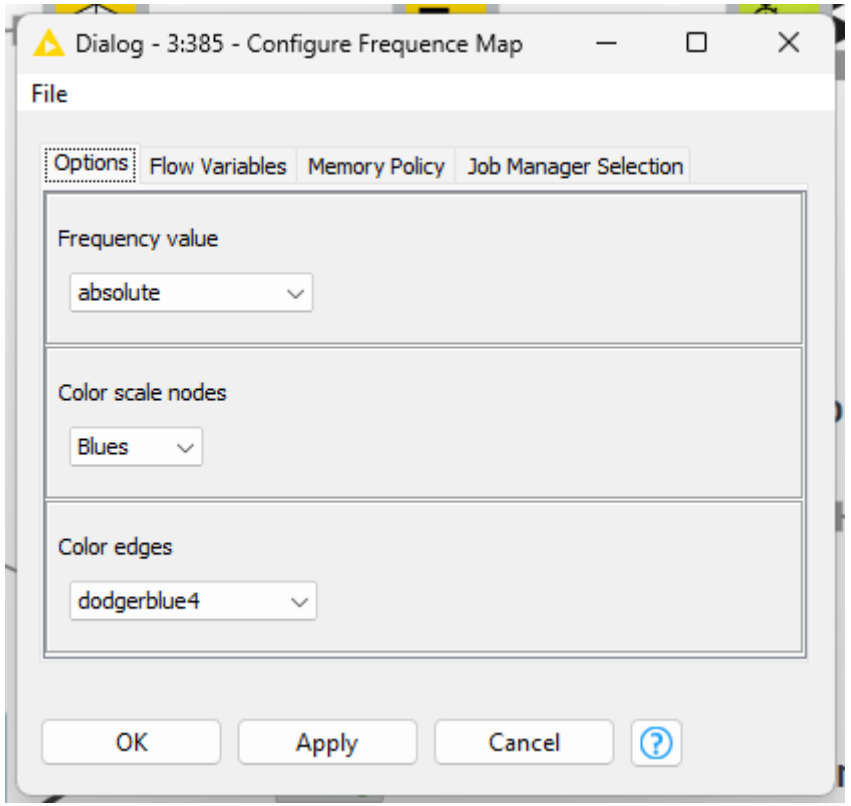


When executed, a process map opens in a browser window

BUPAR

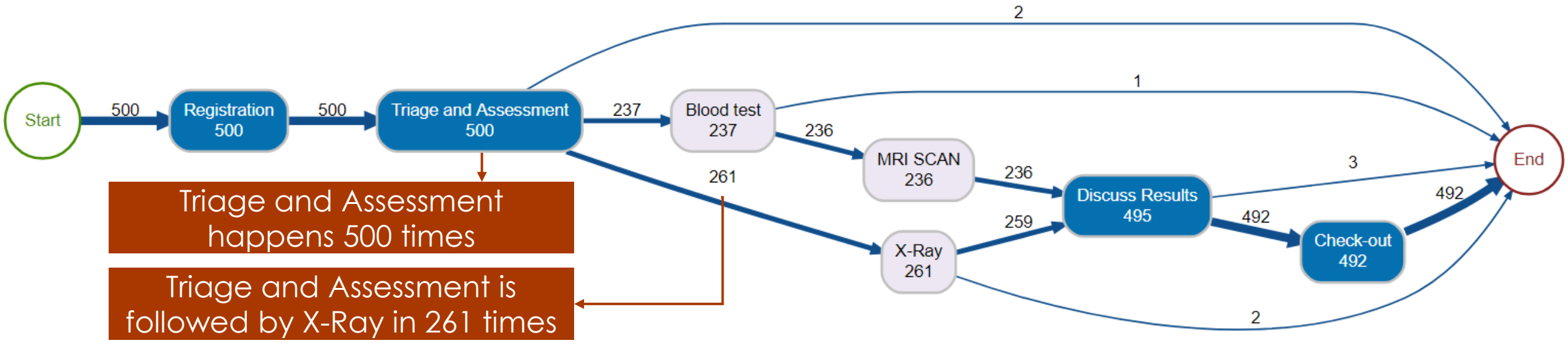
Frequency maps

Frequency type

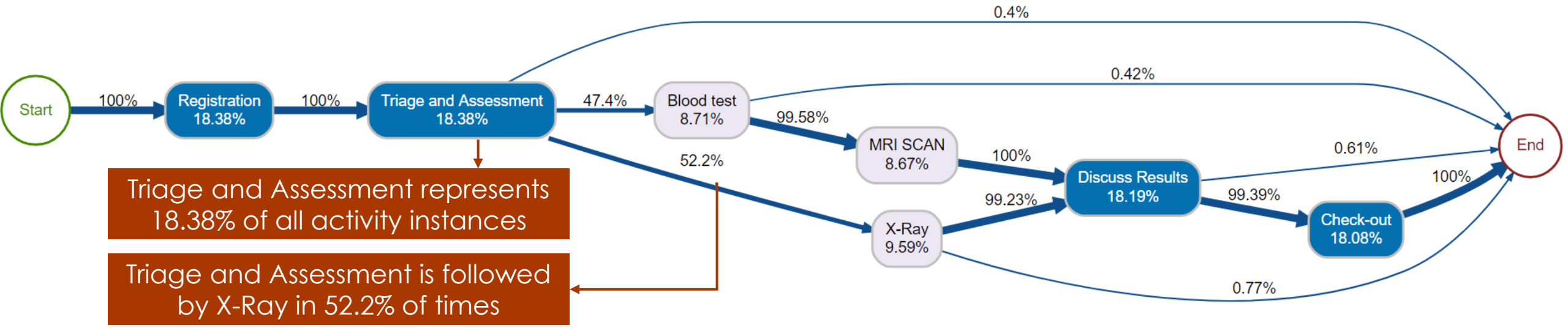


- absolute
- relative
- absolute-case
- relative-case
- relative-consequent

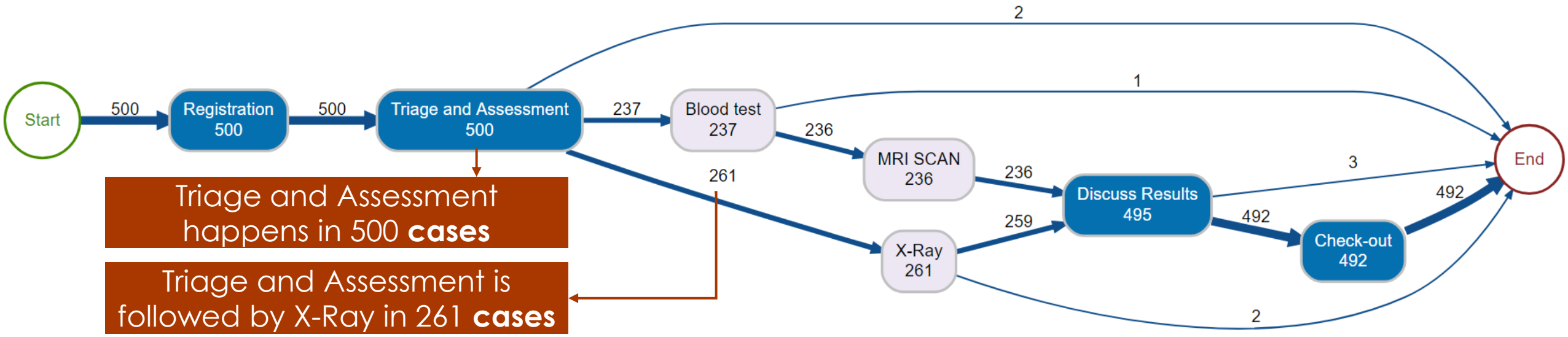
Absolute


Start
500
Registration
500
500
Triage and Assessment
500
237
Blood test
237
236
MRI SCAN
236
236
Discuss Results
495
492
Check-out
492
492
End
3
2
1
261
X-Ray
261
259
2
Triage and Assessment happens 500 times
Triage and Assessment is followed by X-Ray in 261 times

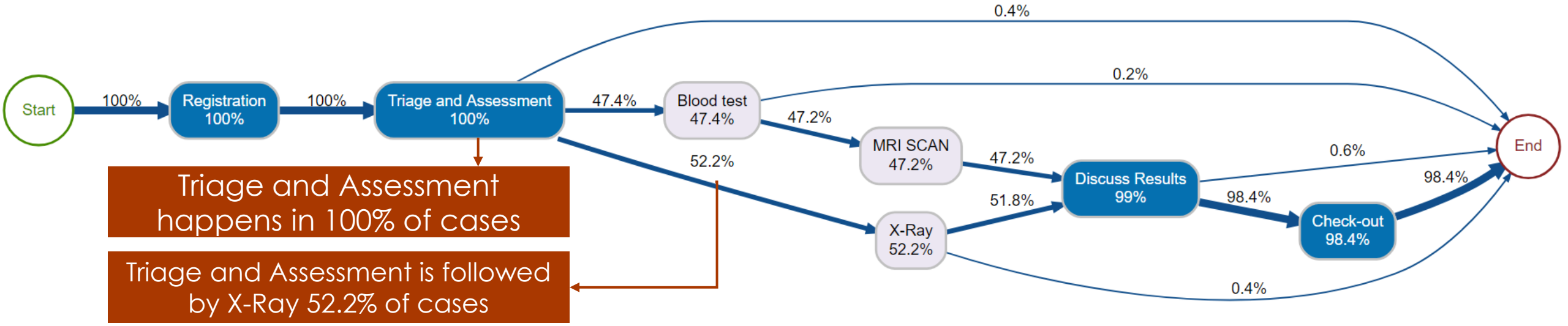
Relative


Start
100%
Registration
18.38%
100%
Triage and Assessment
18.38%
47.4%
Blood test
8.71%
99.58%
MRI SCAN
8.67%
100%
Discuss Results
18.19%
99.39%
Check-out
18.08%
100%
End
0.4%
0.42%
0.61%
52.2%
X-Ray
9.59%
99.23%
0.77%
Triage and Assessment represents 18.38% of all activity instances
Triage and Assessment is followed by X-Ray in 52.2% of times

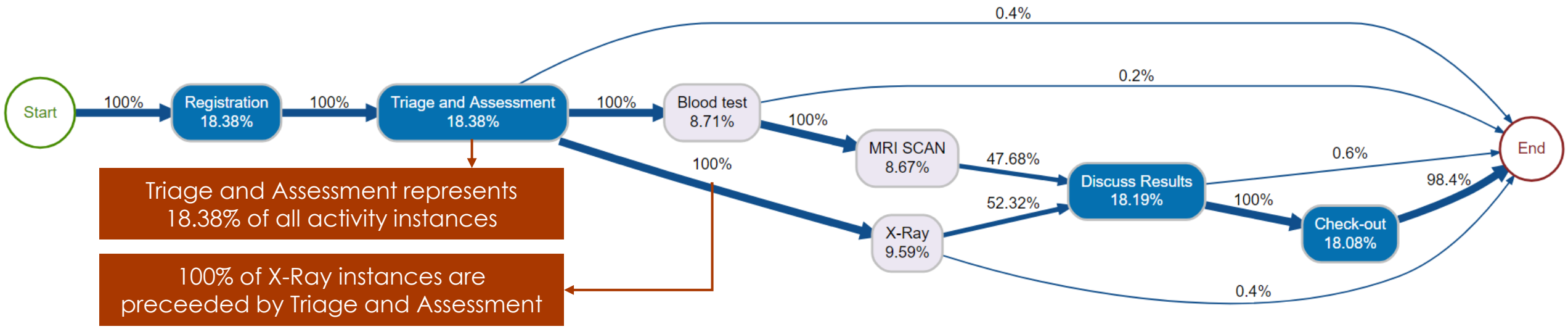
Absolute-case



Relative-case



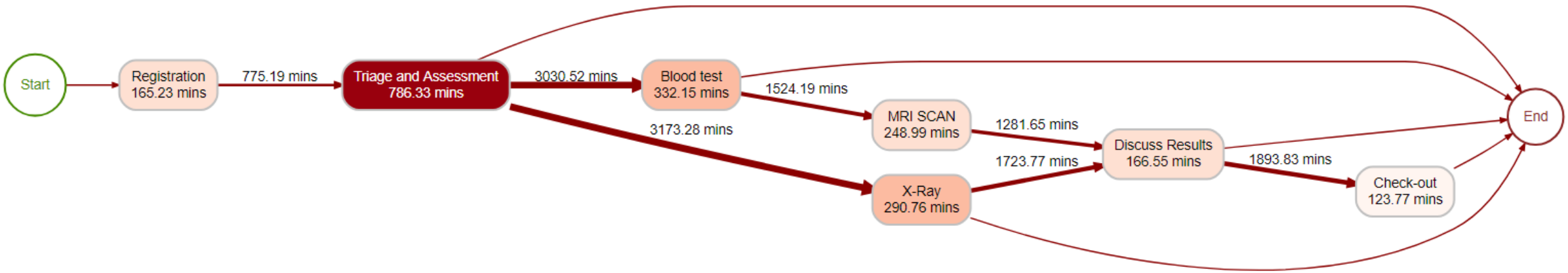
Relative-consequent



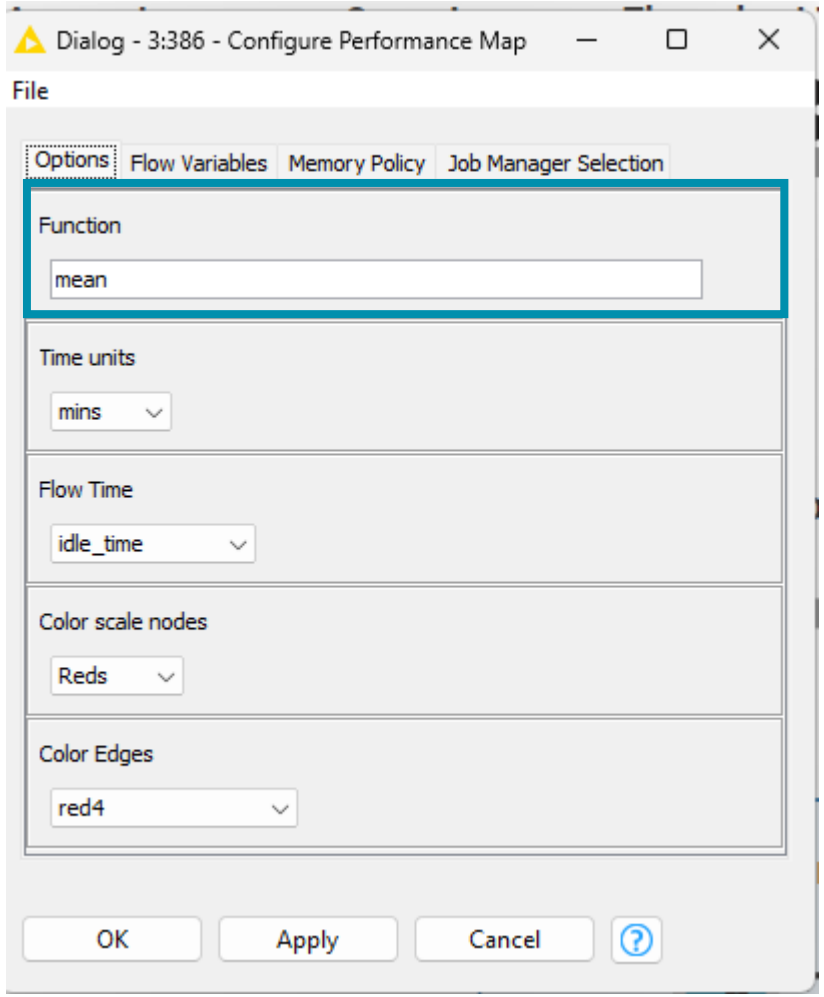
Note that for consistency **relative-antecedent** also exist, but is the same as **relative**.

BUPAR

Performance Maps

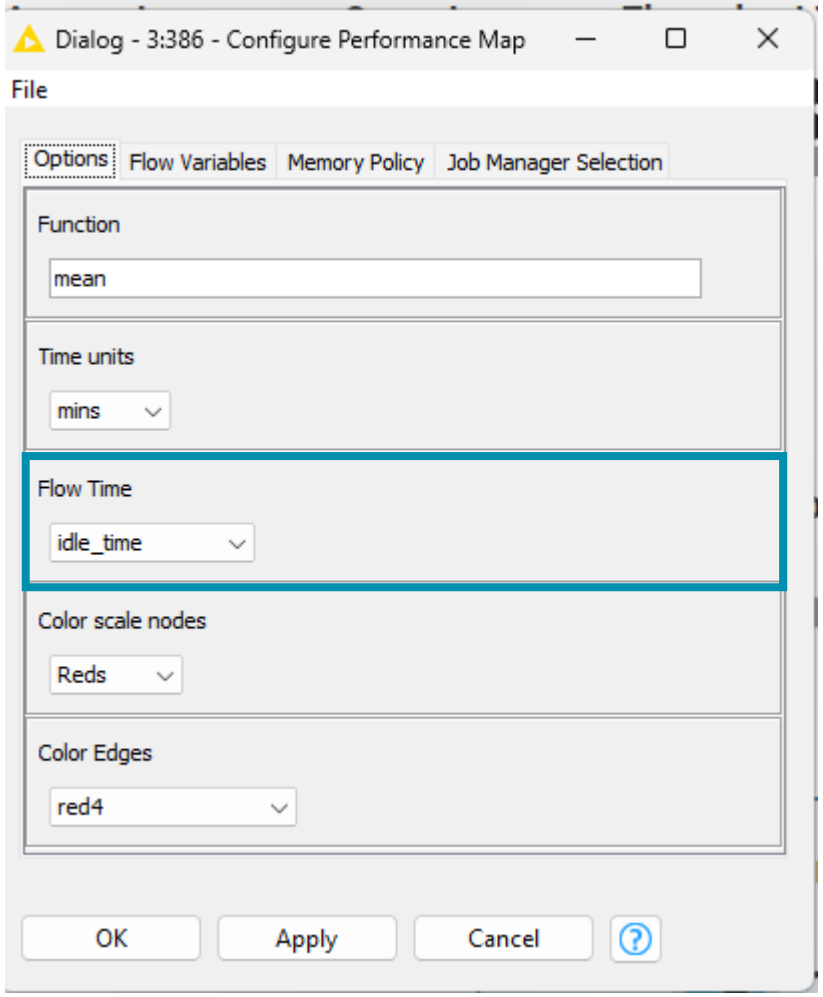

Start
Registration
165.23 mins
775.19 mins
Triage and Assessment
786.33 mins
3030.52 mins
Blood test
332.15 mins
1524.19 mins
MRI SCAN
248.99 mins
1281.65 mins
Discuss Results
166.55 mins
1893.83 mins
Check-out
123.77 mins
End
3173.28 mins
X-Ray
290.76 mins
1723.77 mins

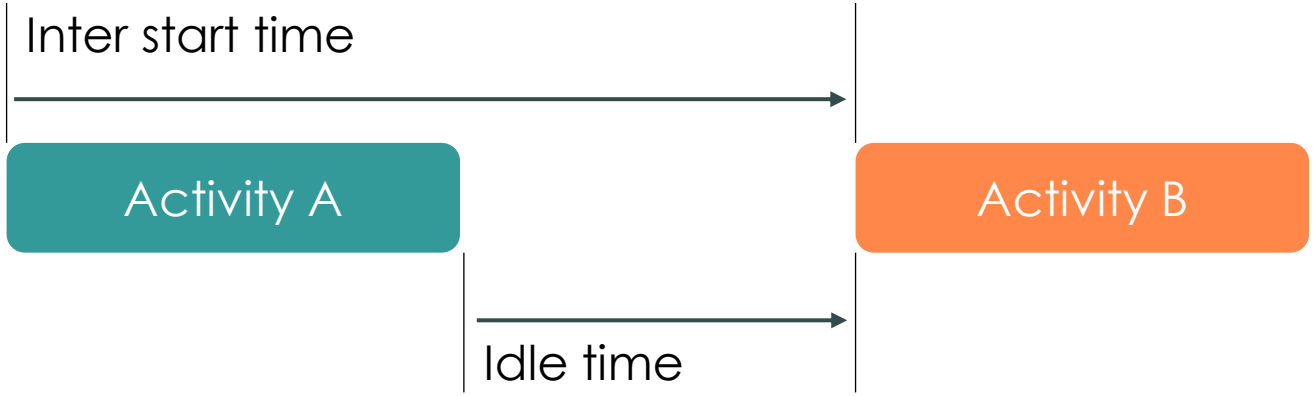
Aggregation function



- Mean
- Median
- Max
- Min
- …

Idle time vs inter start time

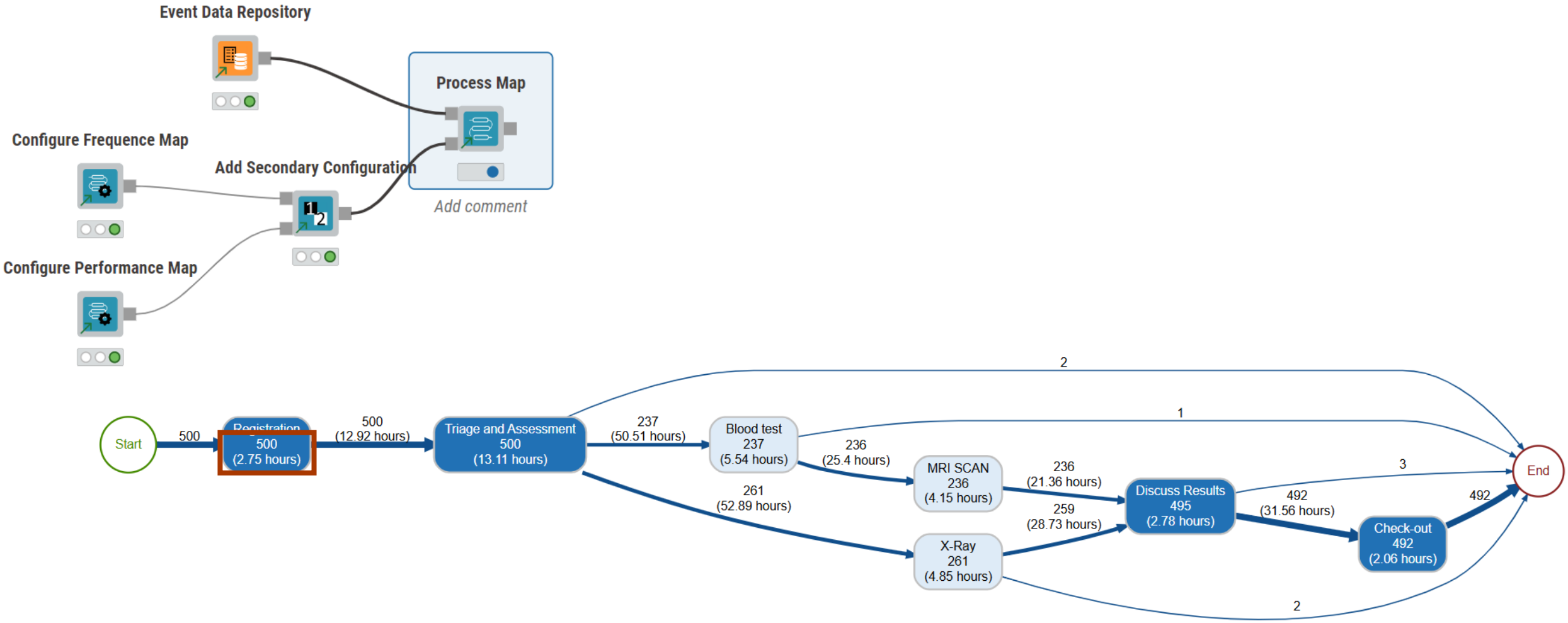




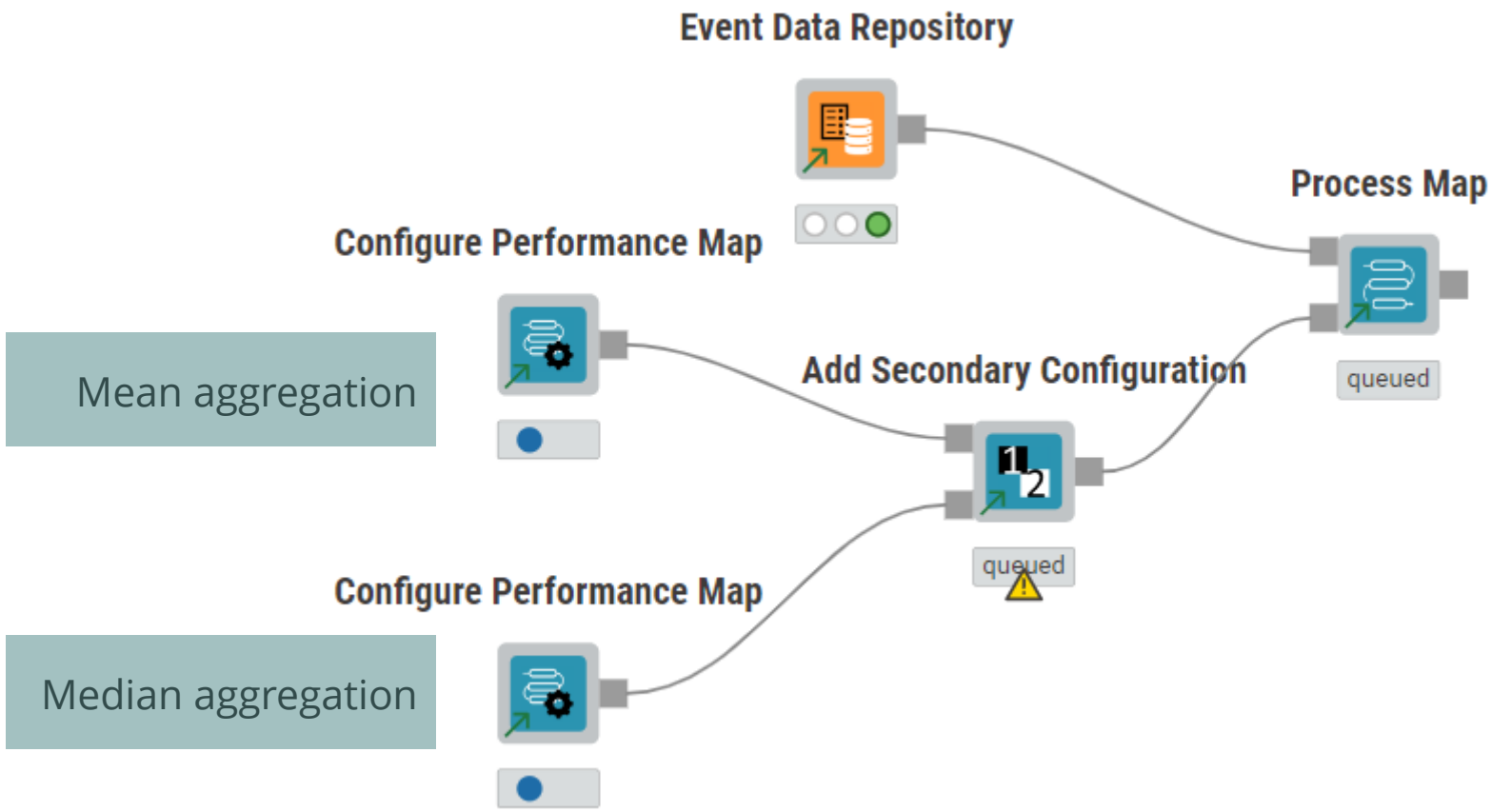
BUPAR

Hybrid map

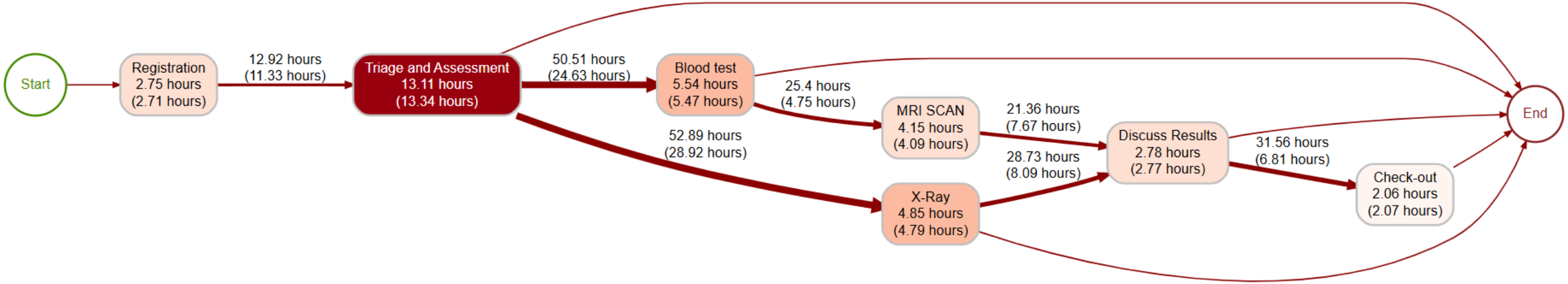
Primary + secondary configurations



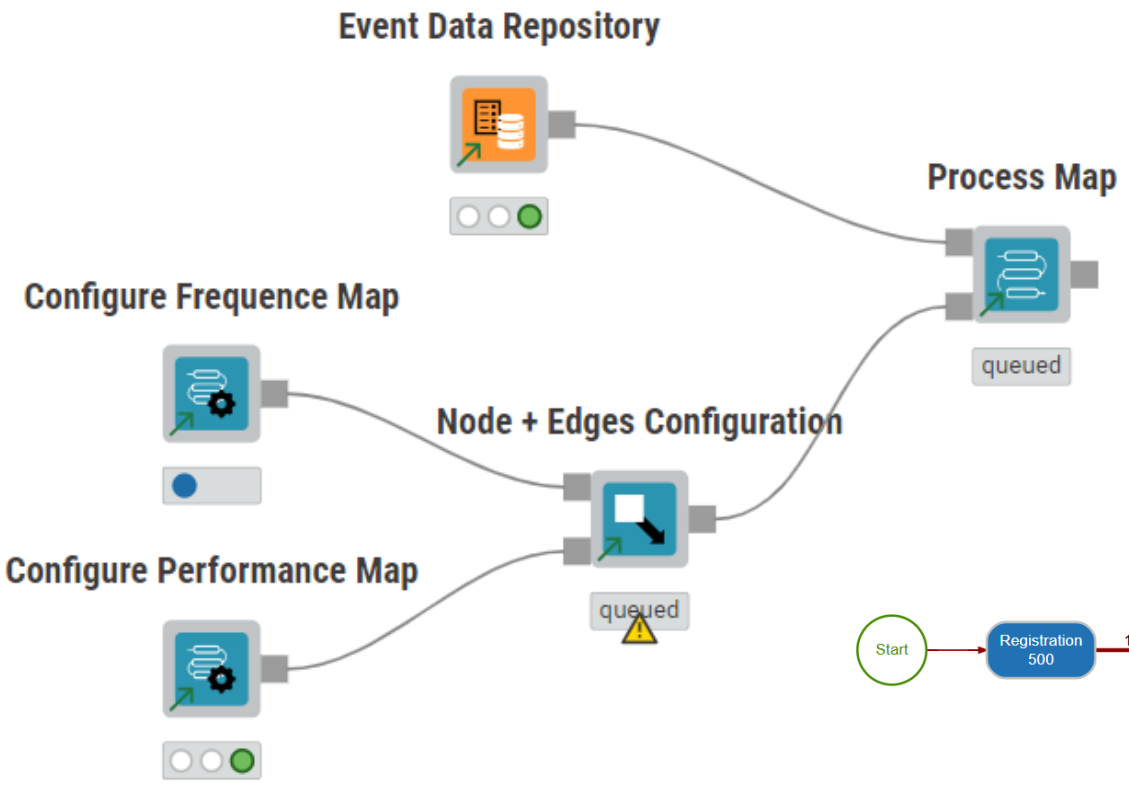
Can be of the same type

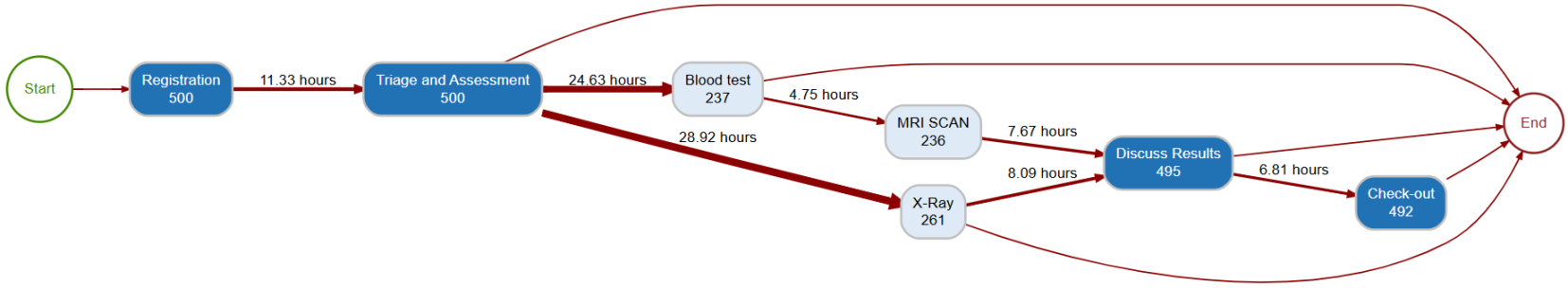


Can be of the same type



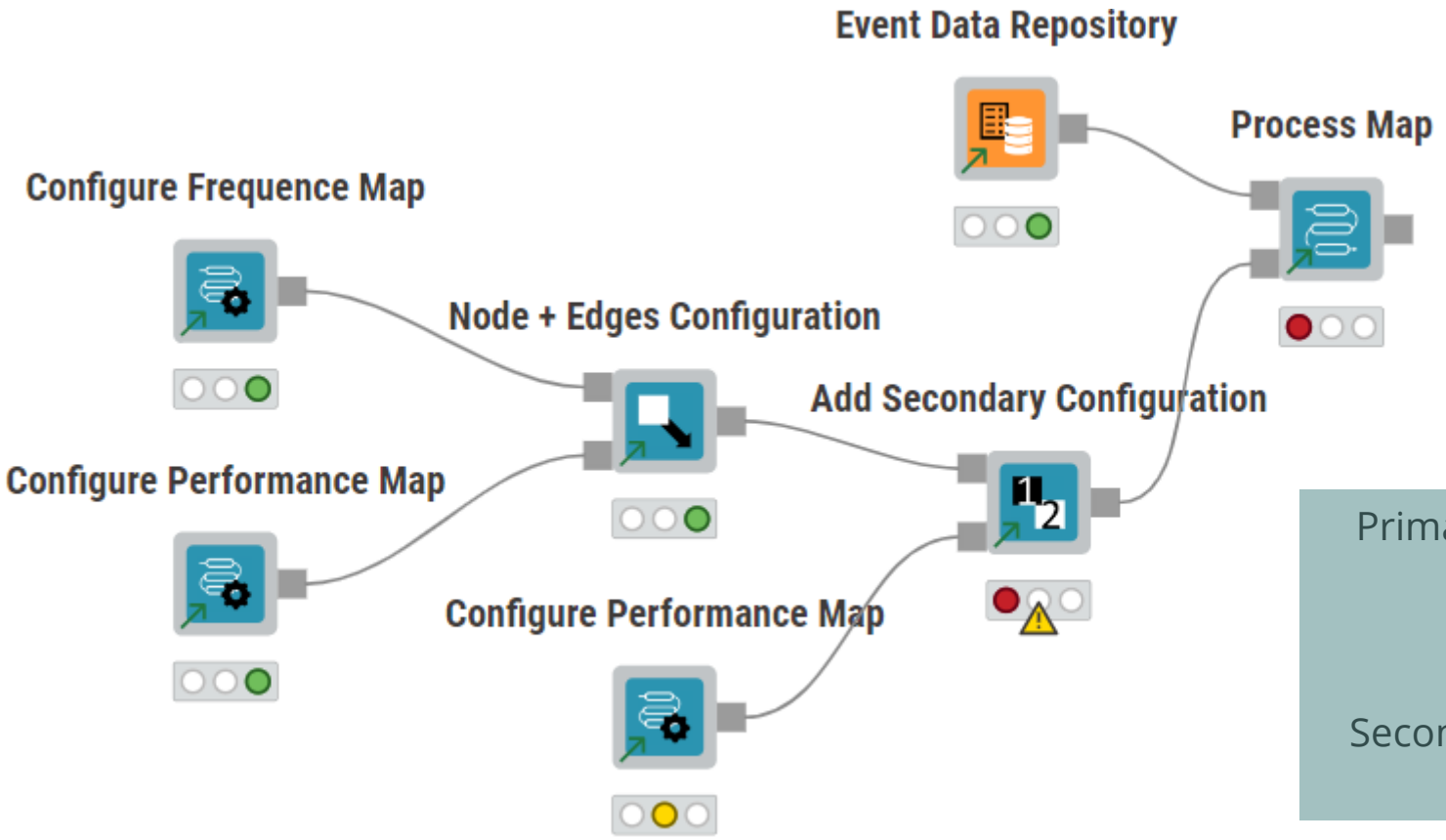
Node + edge configuration





Nodes shows frequency, flows show performance

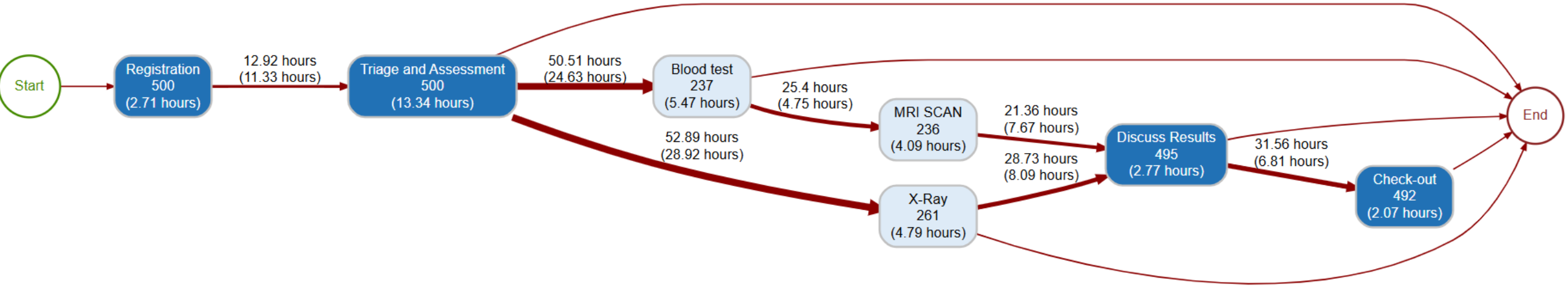
Full Hybrid



Primary: Nodes shows frequency, flows show performance (mean)

Secondary: both nodes and edges show performance (median)

Full Hybrid

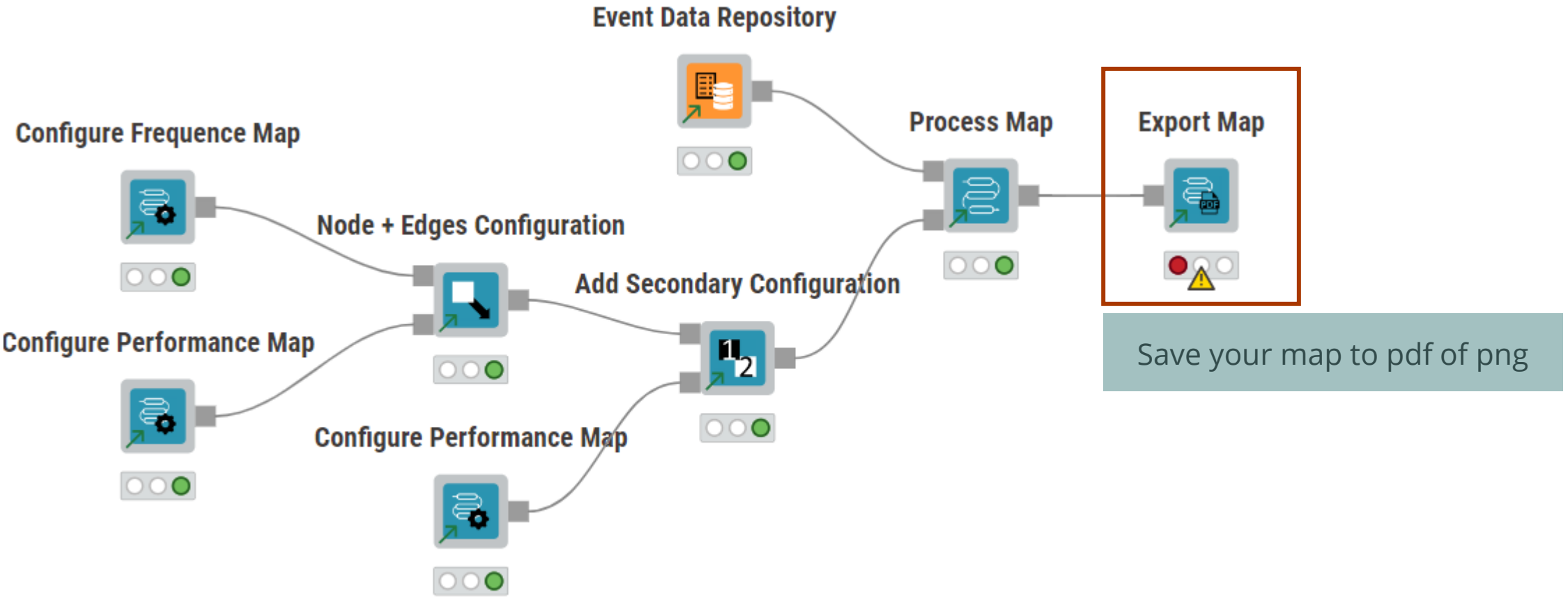


Primary: Nodes shows frequency, flows show performance (mean)

Secondary: both nodes and edges show performance (median)

BUPAR

Export map

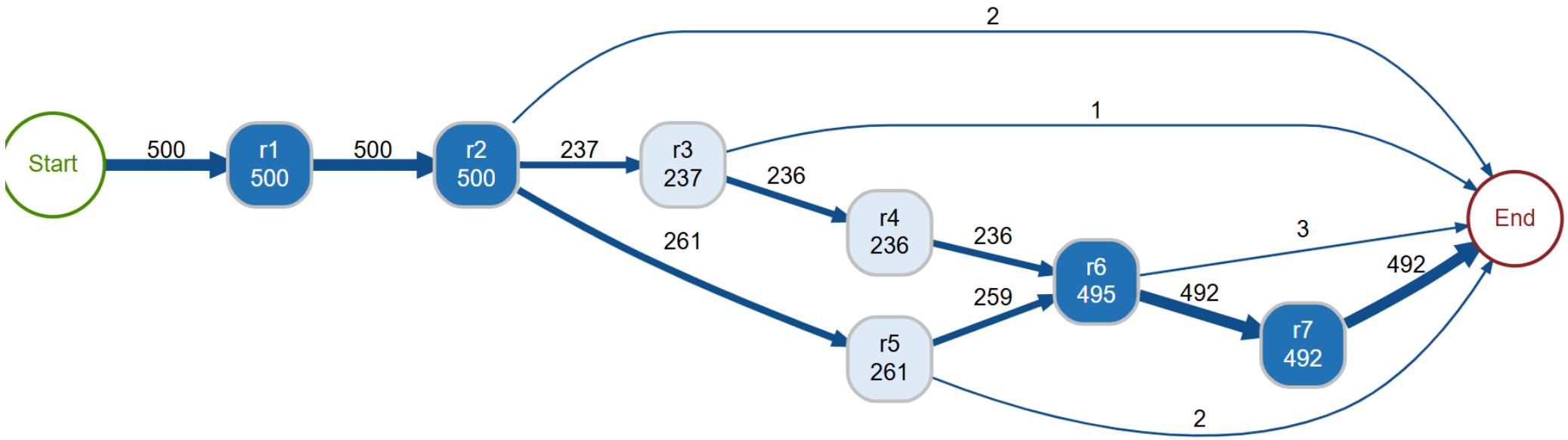

Event Data Repository
Configure Frequence Map
Process Map
Export Map
Node + Edges Configuration
Add Secondary Configuration
Configure Performance Map
Configure Performance Map
Save your map to pdf of png

Resource maps

Show hand-over of work instead of activity flow

All configuration options are identical

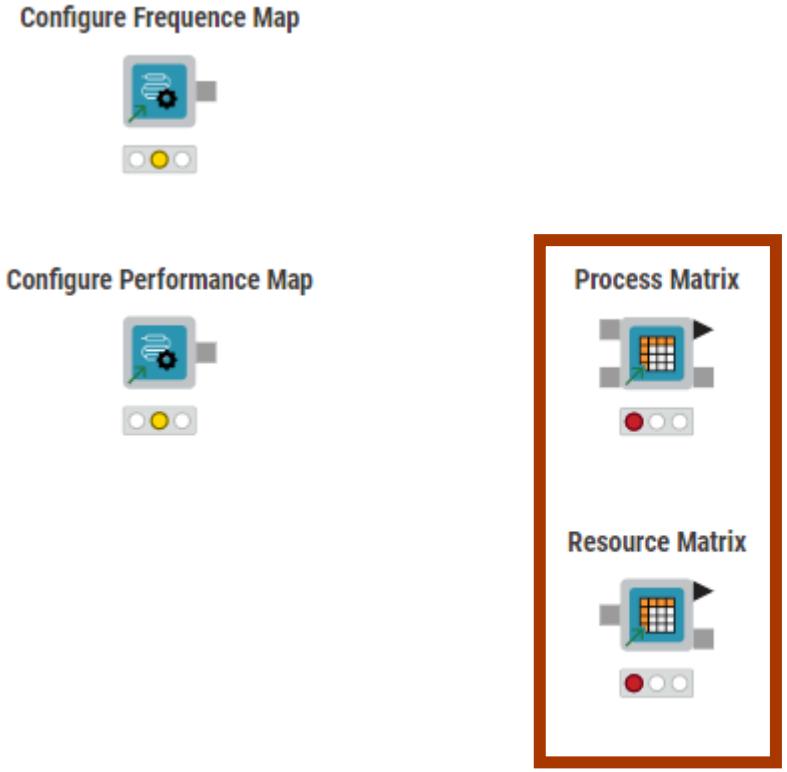
Example


Start
500
r1
500
500
r2
500
237
r3
237
2
1
236
r4
236
261
236
r6
495
3
492
259
492
r7
492
r5
261
2
End

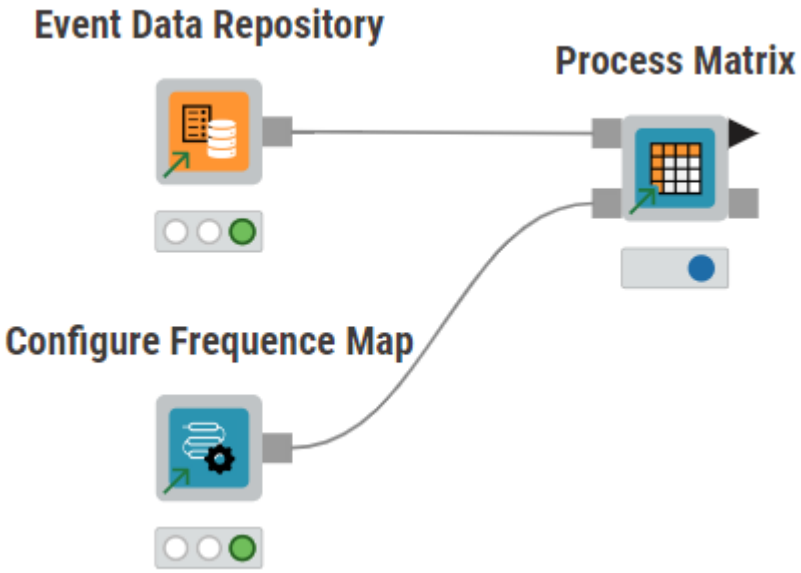
Process / Resource Matrix

Show directly-follows relationships in matrix format

Use the same configurations, but only one at a time



Example



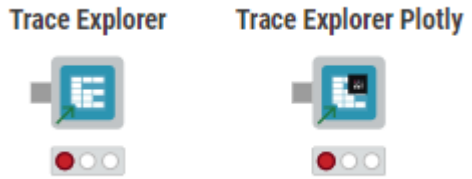
You can examine the result in table format

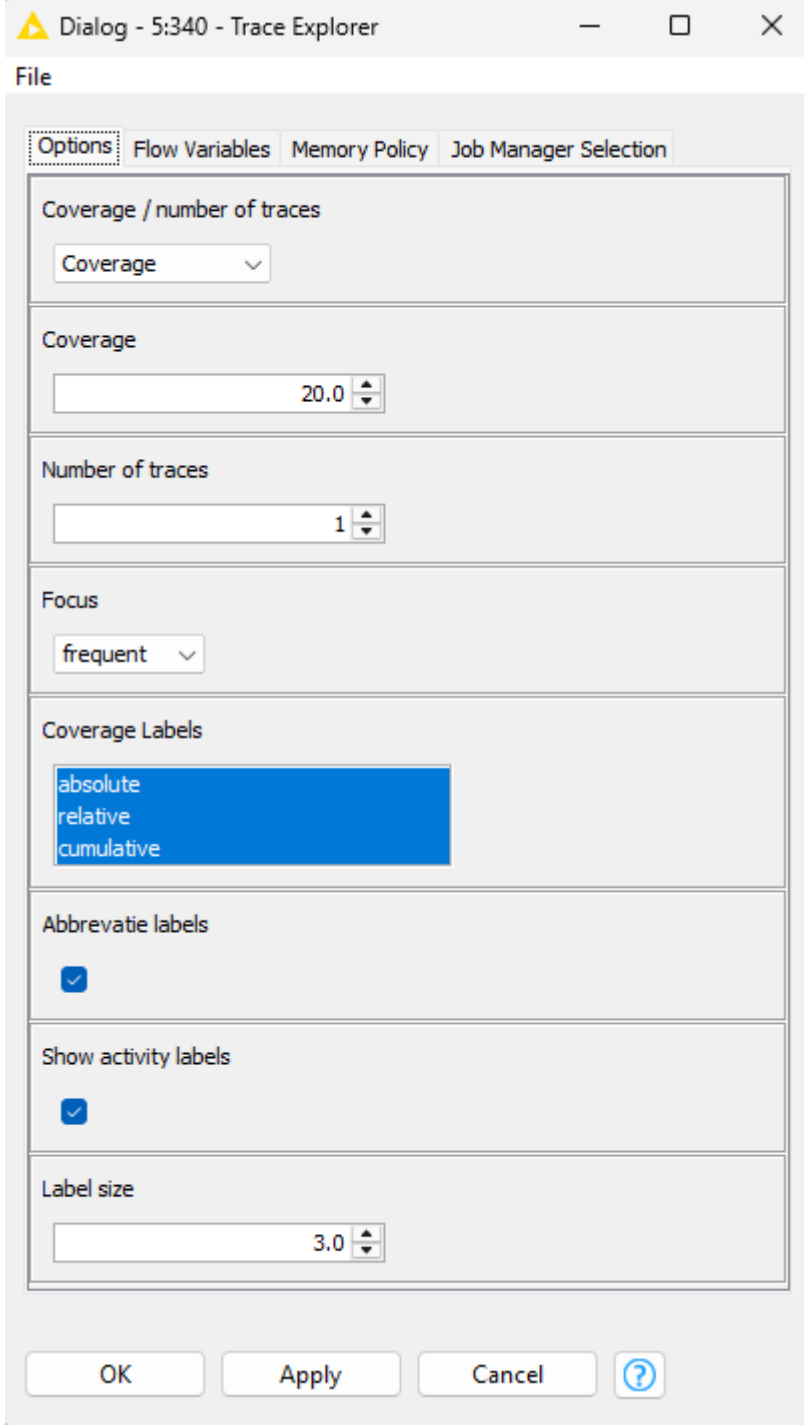
antecedent *String*	consequent *String*	n *Number (Float)*
Blood test	End	1
Blood test	MRI SCAN	236
Check-out	End	492
Discuss Results	Check-out	492
Discuss Results	End	3
MRI SCAN	Discuss Results	236
Registration	Triage and Assessment	500
Start	Registration	500
Triage and Assessment	Blood test	237
Triage and Assessment	End	2
Triage and Assessment	X-Ray	261
X-Ray	Discuss Results	259
X-Ray	End	2

As well as visually

Trace explorer

Explore (in) frequent traces





Define number of traces in relative terms

Or absolute terms

Frequent or infrequent

Frequency statistics to show

Configure layout options

Default: three metrics

1. Relative coverage of the trace (% of cases)

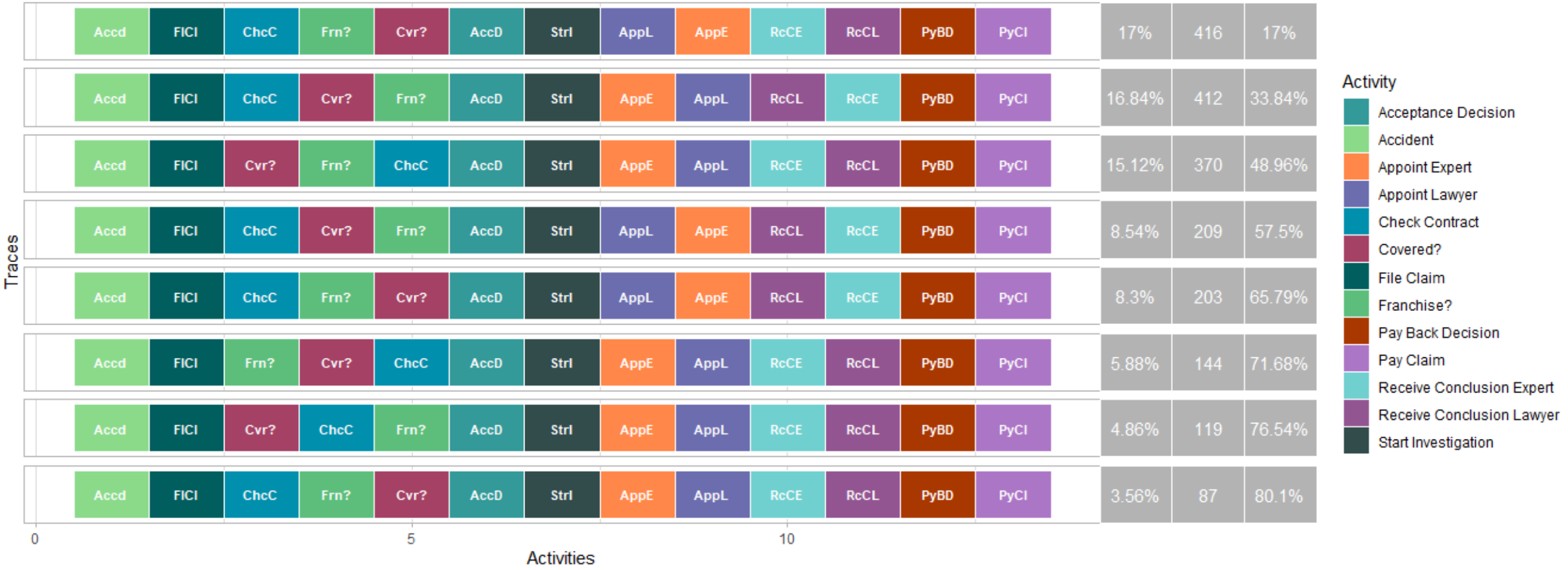
2. Absolute coverage of the trace (# of cases)

3. Cumulative coverage of the trace (% of cases)

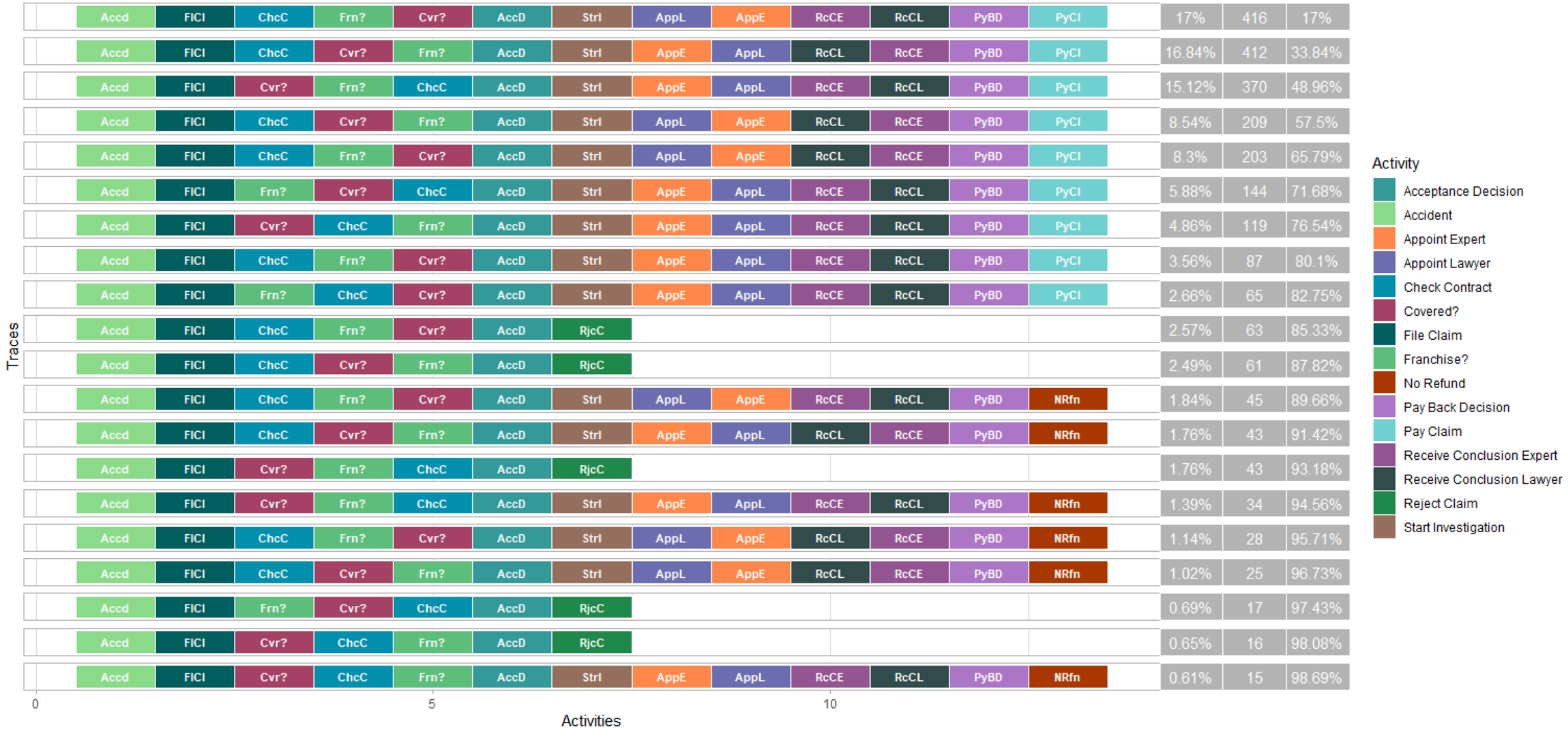
Default: 20% coverage

Only most frequent traces are shown until cumulative coverage exceeds 20%

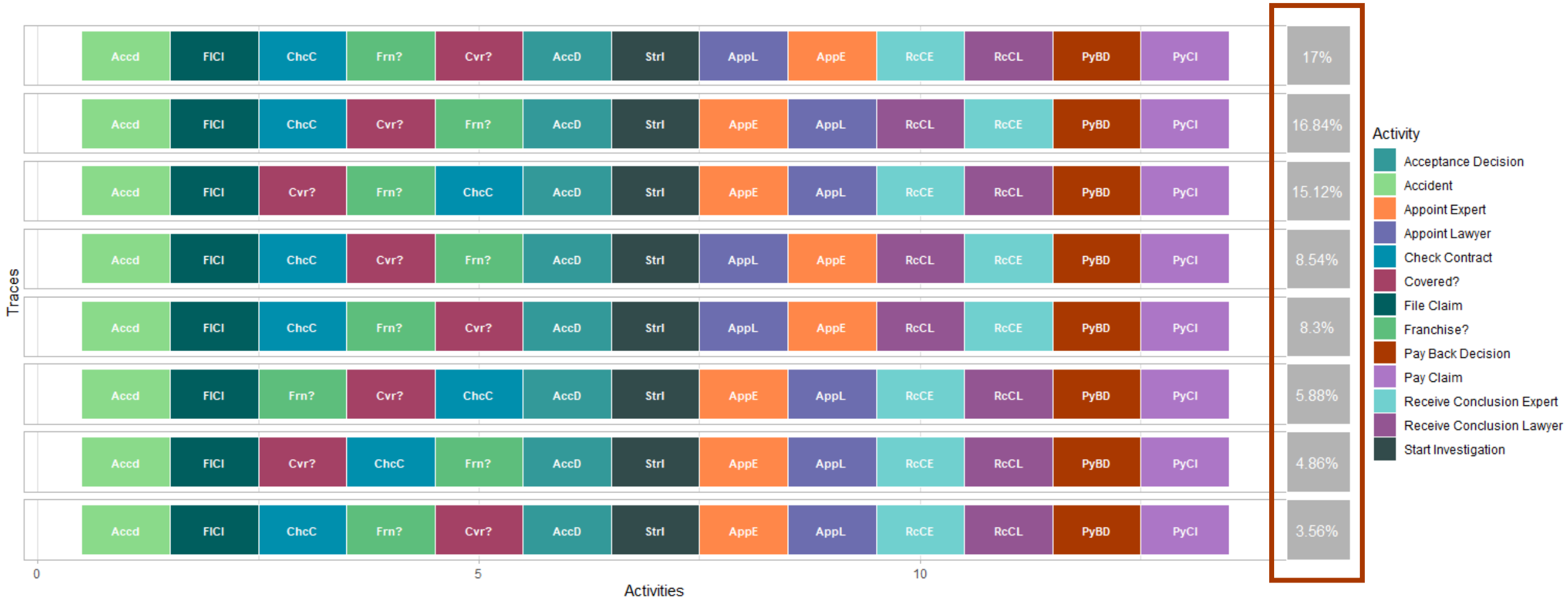
Coverage = 80%


Accd FICI ChcC Frn? Cvr? AccD Strl AppL AppE RcCE RcCL PyBD PyCl 17% 416 17%
Accd FICI ChcC Cvr? Frn? AccD Strl AppE AppL RcCL RcCE PyBD PyCl 16.84% 412 33.84%
Accd FICI Cvr? Frn? ChcC AccD Strl AppE AppL RcCE RcCL PyBD PyCl 15.12% 370 48.96%
Accd FICI ChcC Cvr? Frn? AccD Strl AppL AppE RcCL RcCE PyBD PyCl 8.54% 209 57.5%
Accd FICI ChcC Frn? Cvr? AccD Strl AppL AppE RcCL RcCE PyBD PyCl 8.3% 203 65.79%
Accd FICI Frn? Cvr? ChcC AccD Strl AppE AppL RcCE RcCL PyBD PyCl 5.88% 144 71.68%
Accd FICI Cvr? ChcC Frn? AccD Strl AppE AppL RcCE RcCL PyBD PyCl 4.86% 119 76.54%
Accd FICI ChcC Frn? Cvr? AccD Strl AppE AppL RcCE RcCL PyBD PyCl 3.56% 87 80.1%
Traces
0
5
10
Activities
Activity
Acceptance Decision
Accident
Appoint Expert
Appoint Lawyer
Check Contract
Covered?
File Claim
Franchise?
Pay Back Decision
Pay Claim
Receive Conclusion Expert
Receive Conclusion Lawyer
Start Investigation

Nummber of traces = 20

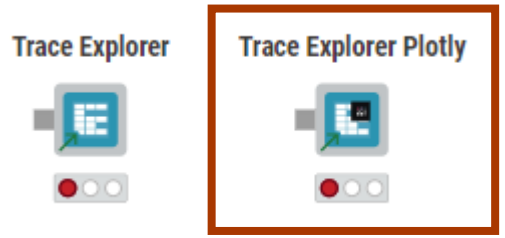


Coverage labels = Relative



Trace explorer

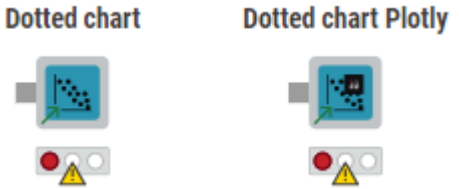
Explore (in) frequent traces

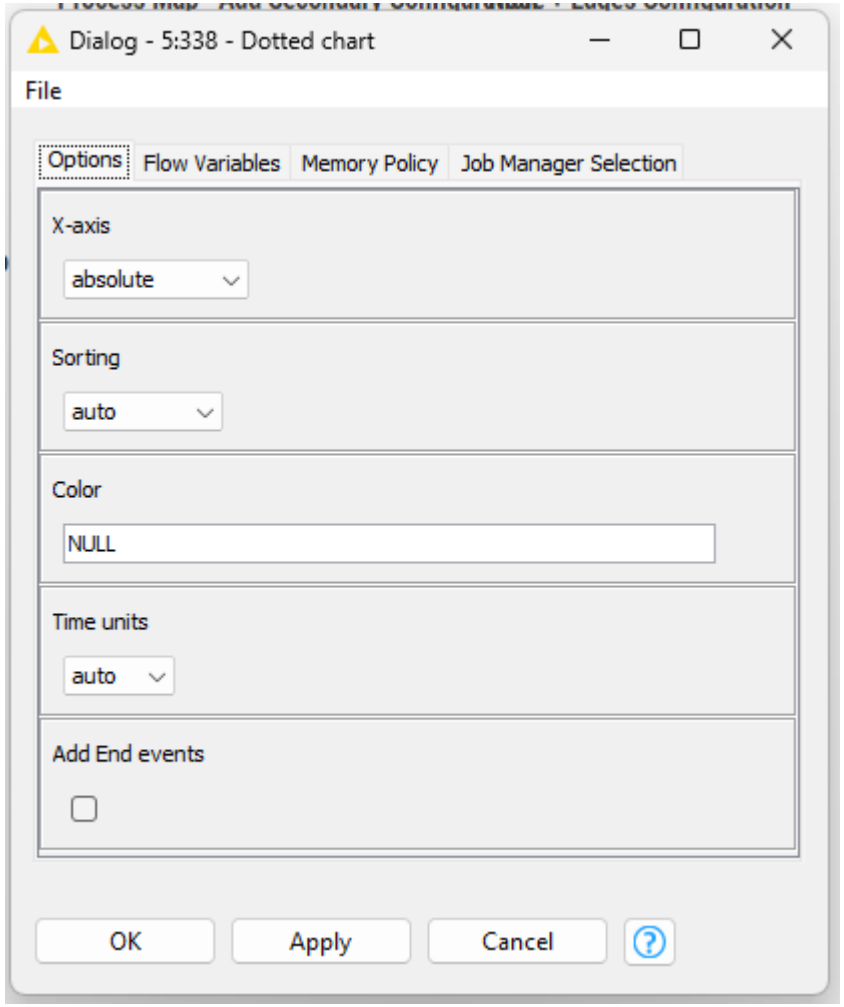


Plotly version for added interativity

Dotted chart

Show all events at a glance

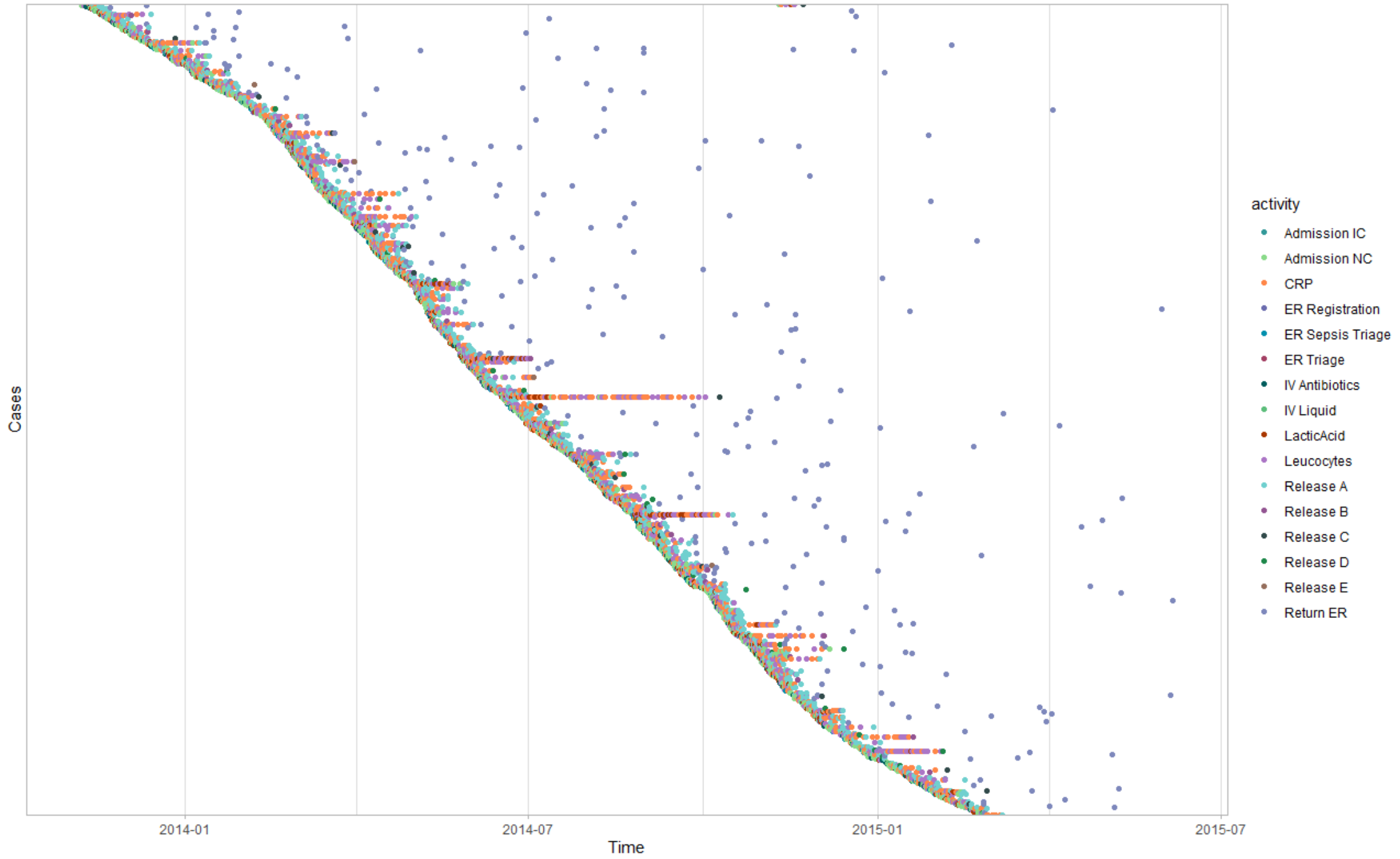




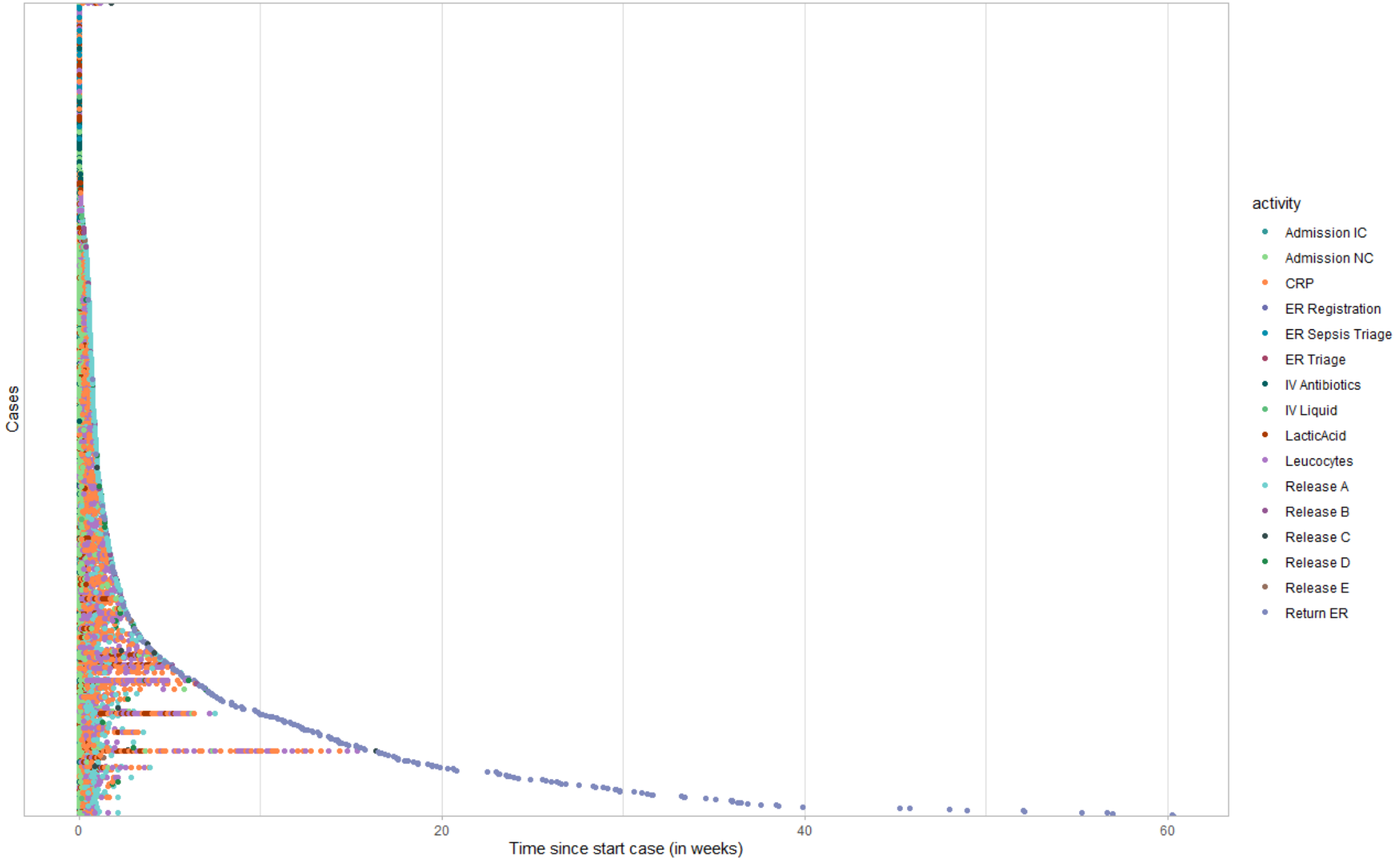
Representation of time

Sorting of cases

Layout configurations



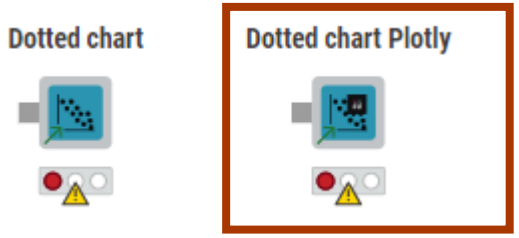
x = “absolute”, sort = “start” (or “auto”)



x = “relative”, sort = “duration” (or “auto”)

Dotted chart

Show all events at a glance



Plotly version for added interativity

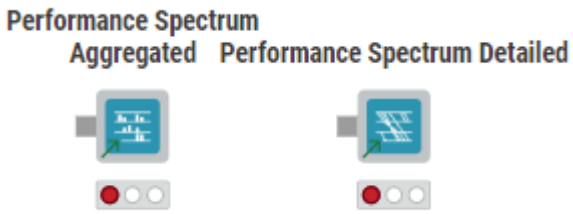
psmineR

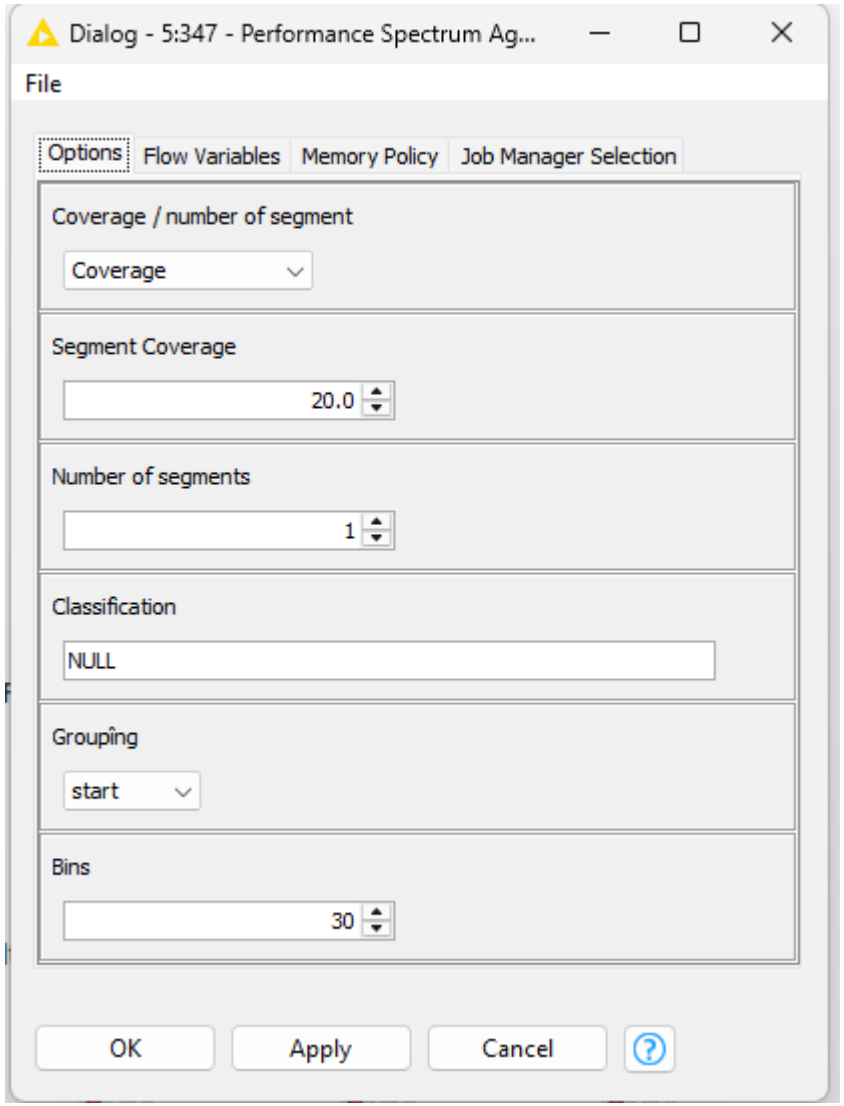
BUPAR

BUPAR

psmineR

Performance spectrum visualisations





Select segments based on coverage or amount

Coverage 20% > only segments that occur in 20% of cases

Absolute number of segments

Color classification. NULL = performance quartiles

Binning (only for aggregated spectrum)

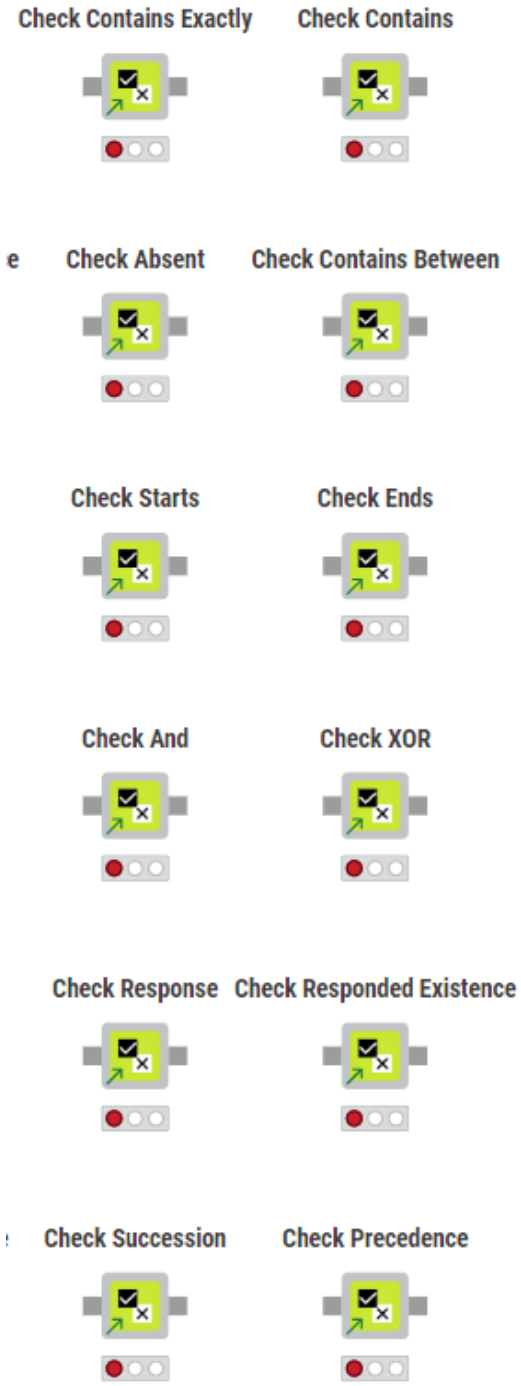
processcheckR

BUPAR

BUPAR

processcheckR

Check declarative rules



Cardinality

- **contains**: activity occurs n times or more
- **contains_exactly**: activity occurs exactly n times
- **contains_between**: activity occurs between min and max number of times
- **absent**: activity occurs less than n times

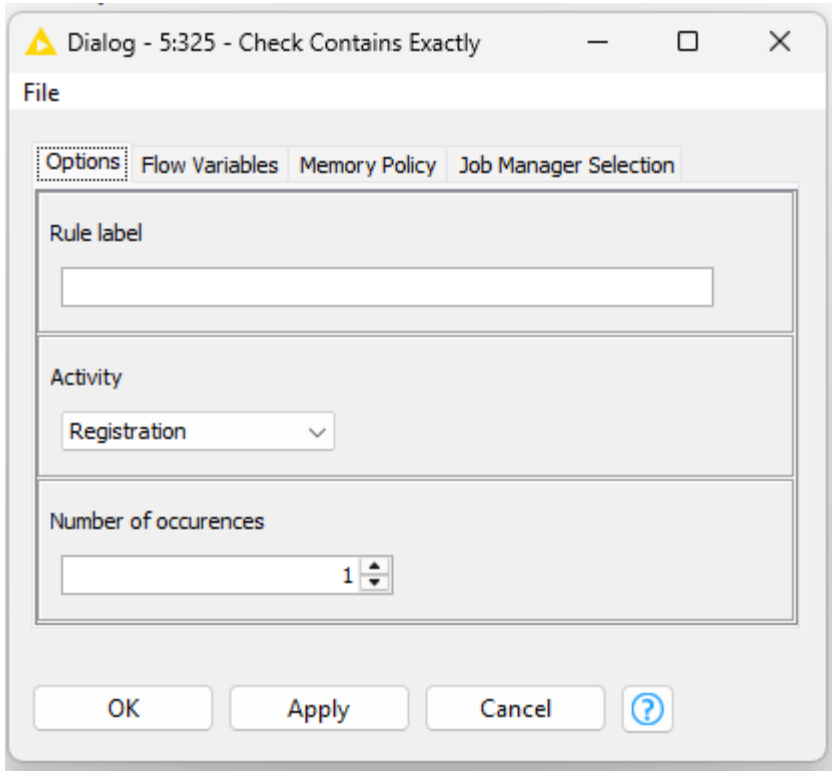
Order

- **starts**: case starts with activity
- **ends**: case ends with activity
- **succession**: if activity A occurs, then activity B must succeed it. If B occurs, then A must have occurred before it.
- **response**: if activity A occurs, then B must occur later
- **precedence**: if activity B occurs, then A must have occurred before it
- **responded_existence**: if activity A occurs, B must also occur (before or after A)

Exclusivity

- **and**: two activities must occur together
- **xor**: two activities cannot occur together (exclusive or)

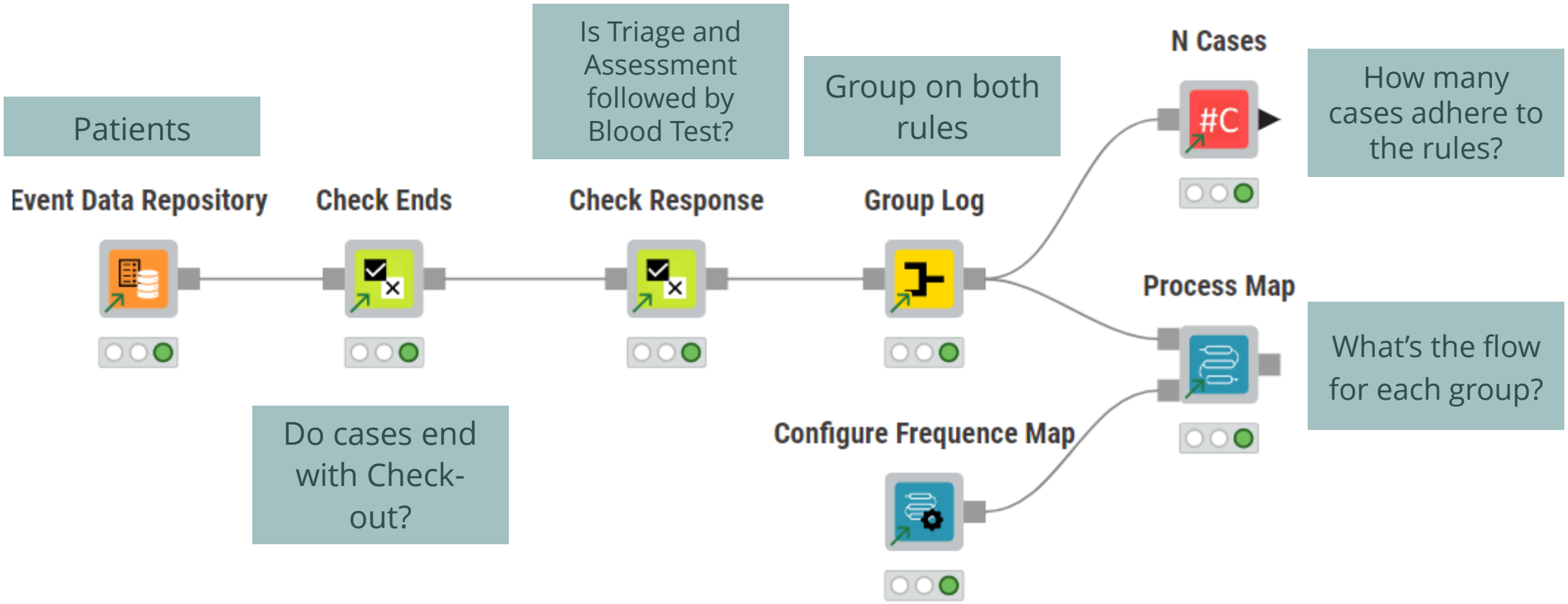
Rule template



Rule label. Variable with this name will include results

Rule specific parameters

Example

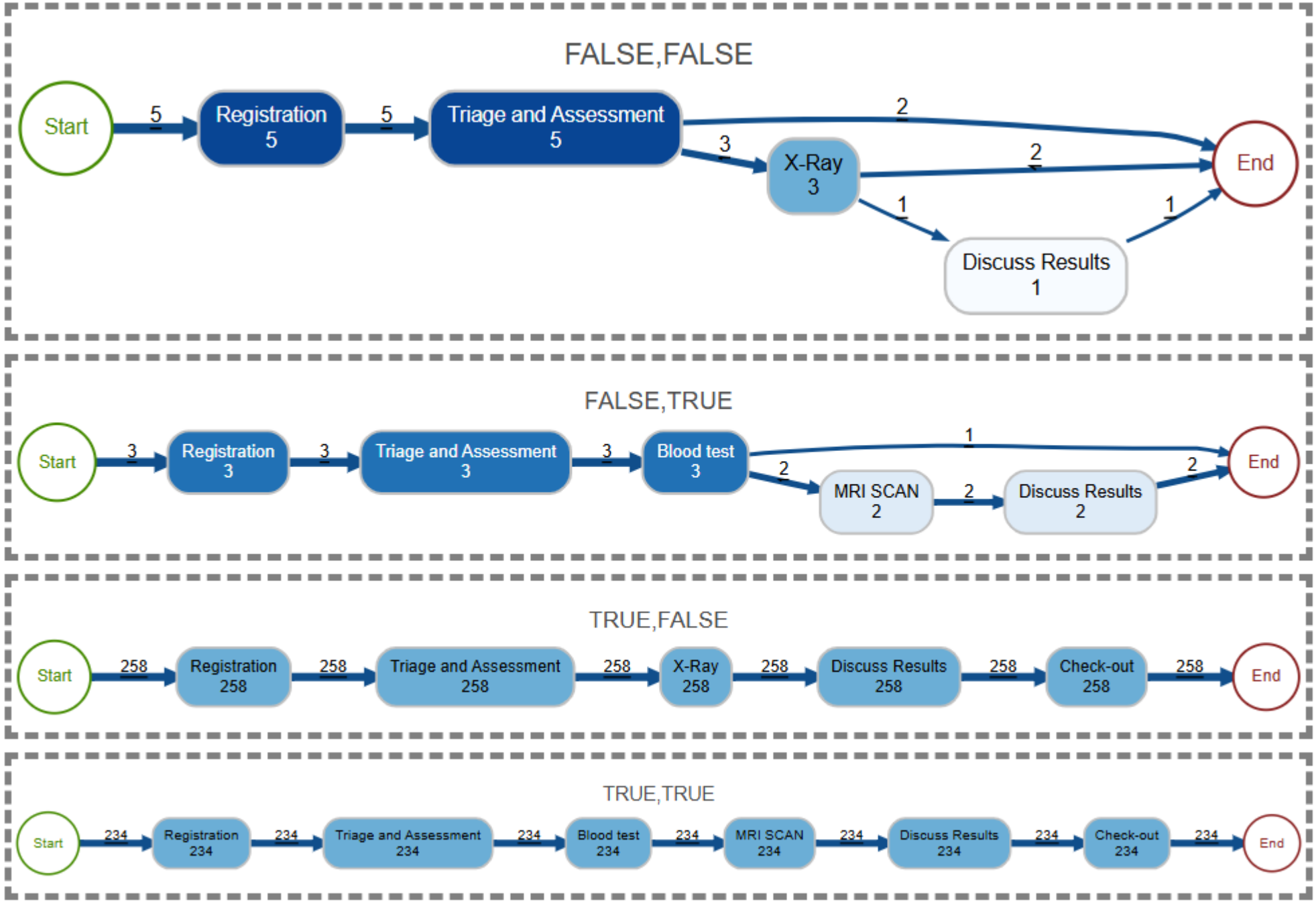

Patients
Event Data Repository
Check Ends
Do cases end with Check-out?
Is Triage and Assessment followed by Blood Test?
Check Response
Group on both rules
Group Log
N Cases
#C
How many cases adhere to the rules?
Process Map
What's the flow for each group?
Configure Frequence Map

ending_checked_out *Boolean*	triage_then_blood_test *Boolean*	n_cases *Number (Integer)*
false	false	5
false	true	3
true	false	258
true	true	234

N Cases

#C

Process Map



BUPAR

bupar.net
bupaverse.github.io/docs/
github.com/bupaverse