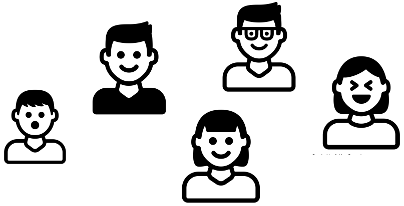
“Hey, I need something to do!”

Machine Learning in Human Resource Allocation

Rachel Brabender and Oliver Clasen

All icons from https://thenounproject.com/

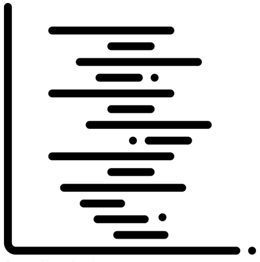
What is Resource Allocation?



Resource +

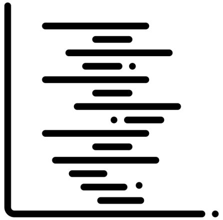
different activity instances

= **Allocation**



Activity instance *XY* will be executed by *employee 1*.

The Challenge



Log Data with Assignments (BPIC‘15)

Learning of
What activity
by who and
with which duration.

On run time allocation

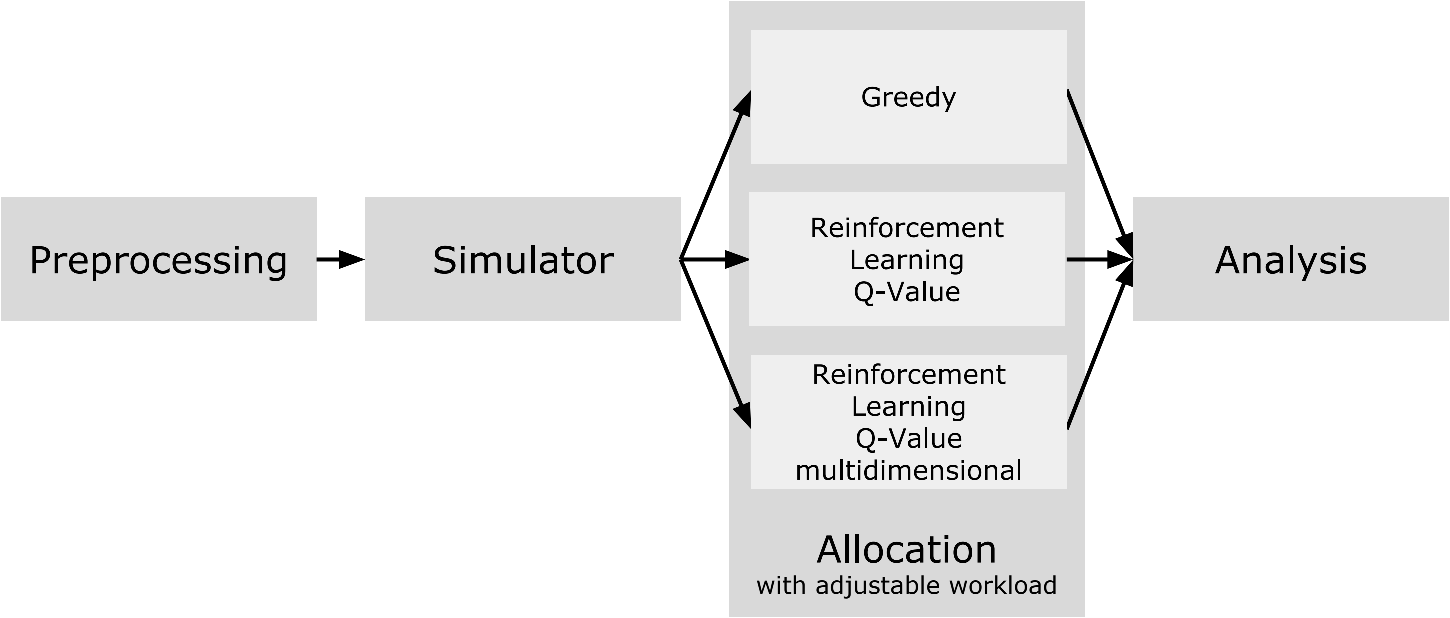
TRAINING

PREDICTION

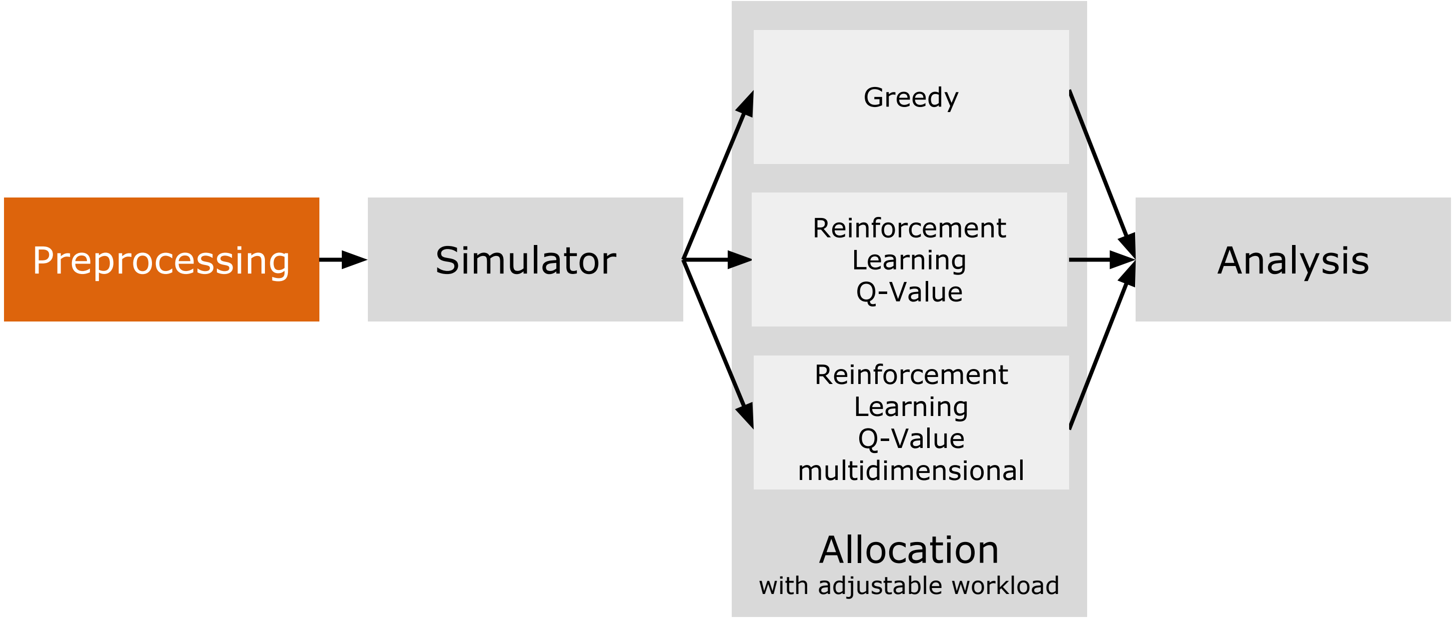
Machine Learning Algorithm

All icons from https://thenounproject.com/

Pipeline


Preprocessing
Simulator
Greedy
Reinforcement Learning Q-Value
Reinforcement Learning Q-Value multidimensional
Allocation
with adjustable workload
Analysis

Pipeline


Preprocessing
Simulator
Greedy
Reinforcement Learning Q-Value
Reinforcement Learning Q-Value multidimensional
Allocation
with adjustable workload
Analysis

Log Data

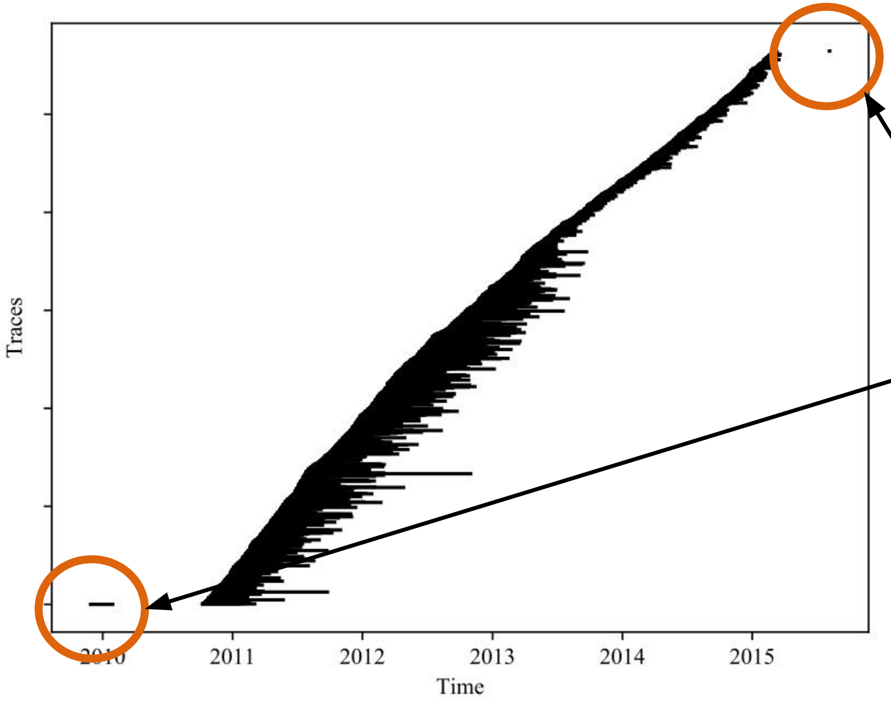

```
<event>
        <string key="question" value="EMPTY"/>
        <date key="dueDate" value="2014-01-20T09:45:30+01:00"/>
        <string key="action_code" value="01_HOOFD_010"/>
        <string key="activityNameEN" value=" register submission date request"/>
        <date key="planned" value=" 2014-01-16T09:45:30+01:00"/>
        <string key="dateFinished" value=" 2014-01-15 00:00:00"/>
        <date key="time:timestamp" value=" 2014-01-14T00:00:00+01:00"/>
        <string key="monitoringResource" value=" 560532"/>
        <string key="org:resource" value=" 560532"/>
        <string key="activityNameNL" value="registratie datum binnenkomst aanvraag"/>
        <string key="concept:name" value="01_HOOFD_010"/>
        <string key="lifecycle:transition" value="complete"/>
</event>
```


activity name

Start / endtime, for calculating the duration

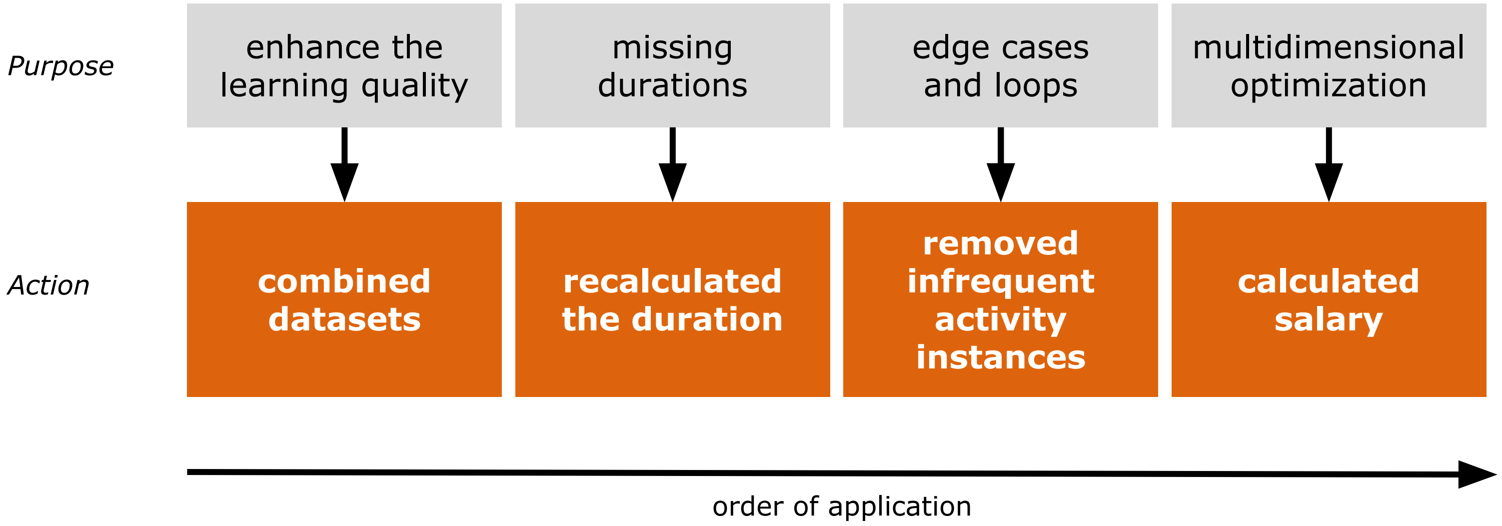
employee-id

Duration of original dataset



Removed outlines

Preprocessing



Duration calculation

Procedure:

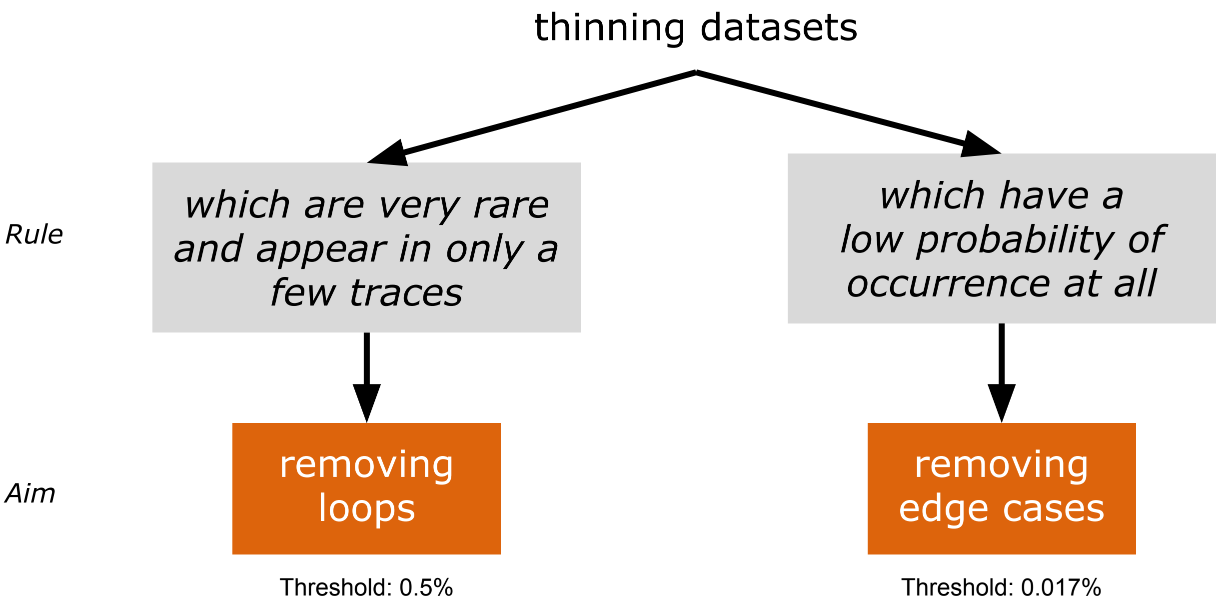
```
event['duration'] = event['end'] - event['start']
if event['duration'] < one_hour:
   event['duration'] = event['end'] - last_end
if ('planned' in event) & (event['duration'] < one_hour):
   event['duration'] = event['planned'] - event['start']
if event['duration'] < one_hour:
   event['duration'] = one_hour
if event['duration'] > thirty_days:
   event['duration'] = thirty_days
```

There was not always a ‘planned’ timestamp in the dataset available.

excluded details:

- start and end of a workday
- lunch break
- weekend and holiday

Thinning of the dataset



Salary calculation

Assumption

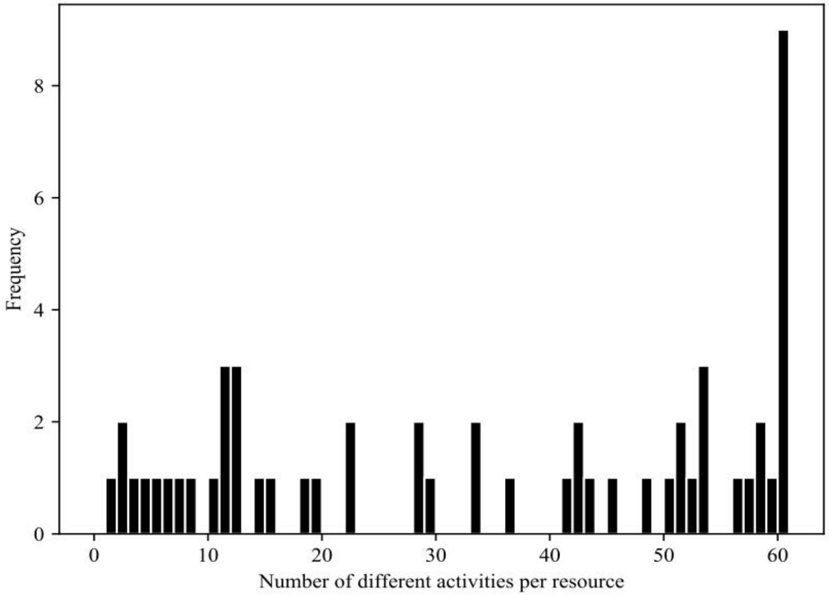
The higher the number of different activities performed by a resource, the more expertise it has.

$$s(x) = \frac{x^2}{m} + 10$$

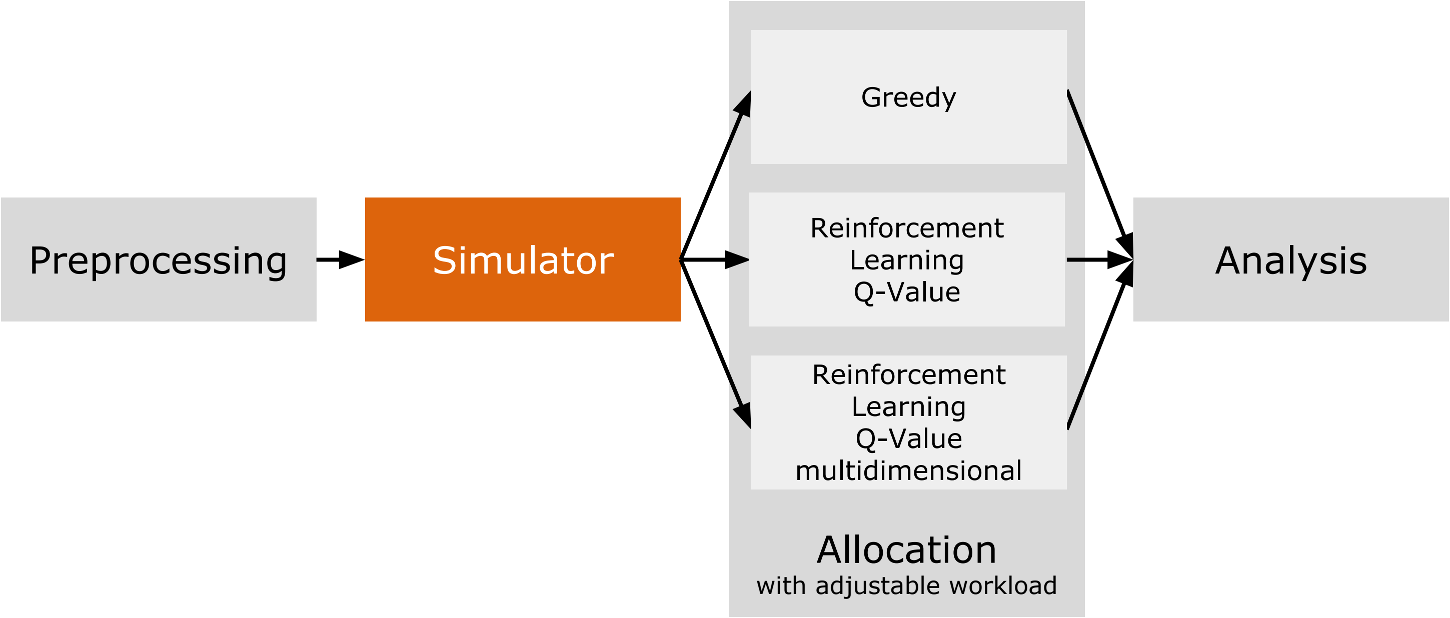
where

x = number of different activities of a resource

m = max number of different activities that a resource has executed



Pipeline



Preprocessing
Simulator
Greedy
Reinforcement Learning Q-Value
Reinforcement Learning Q-Value multidimensional
Allocation
with adjustable workload
Analysis

Assumptions

!

- A trace is ready for allocation when the **start time of the first activity instance** has occurred
- Only **one activity instance** of a trace can be allocated and executed **at the same time**
- **When** an activity instance ends, the **next activity instance** can be allocated **directly**

Simulator

```
For each interval in simulation:

     New traces available?

     For each running trace:
          If status == done of first activity instance in trace:
               Discard activity instances from trace
          If status == free of first activity instance in trace:
               Allocate activity instance

     Proceed time
```

Simulation Interval

Approach: *Reduce simulation overhead*

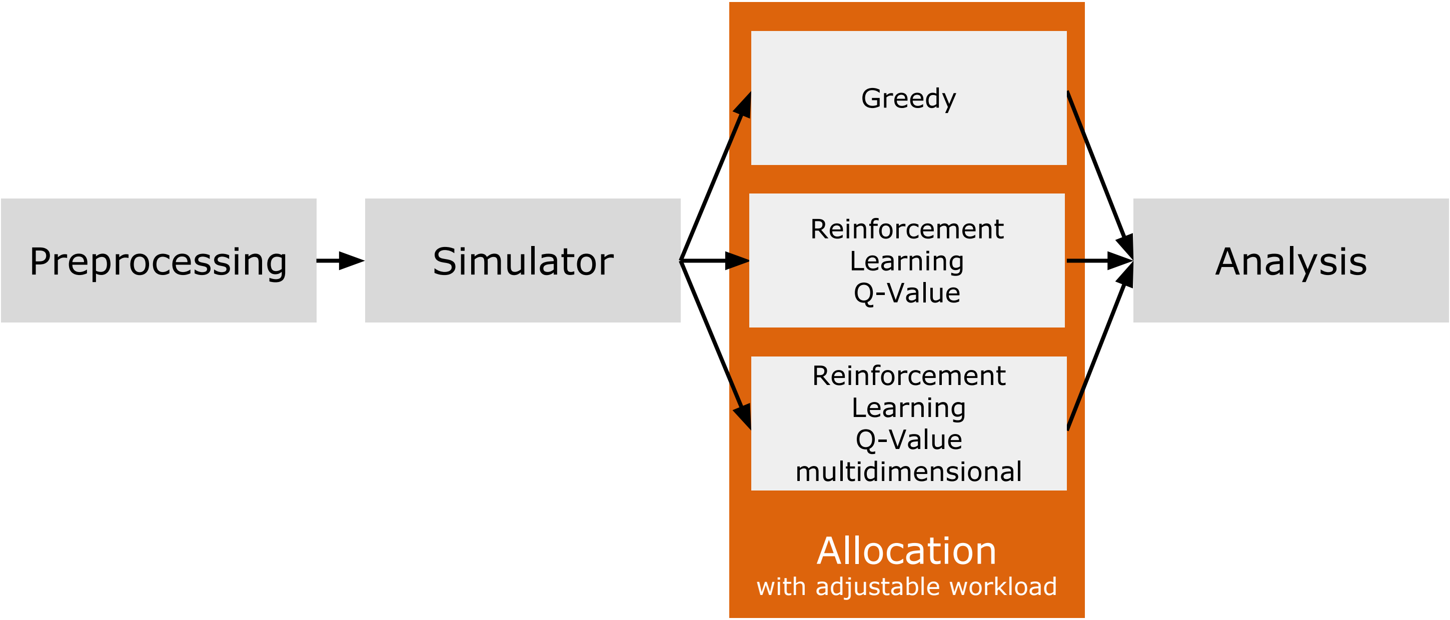
↓

Idea: *Check availability of resources and for incoming traces only in predefined time intervals.*

↓

Interval: 1 min

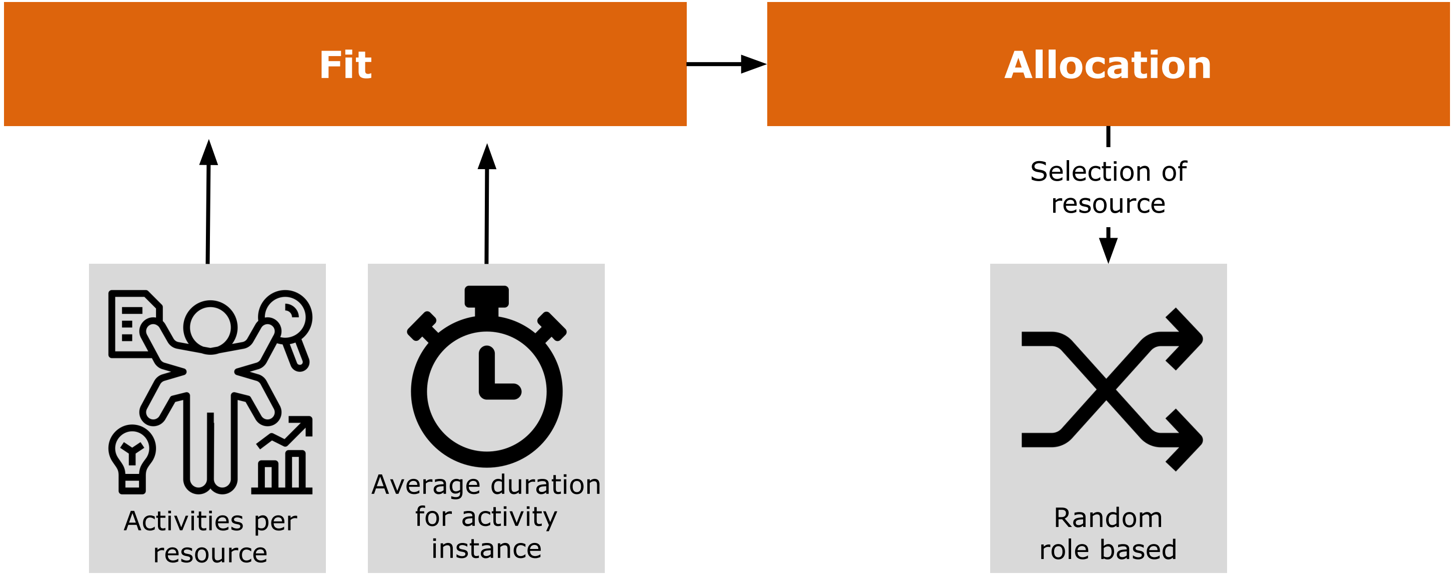
Pipeline



Preprocessing
Simulator
Greedy
Reinforcement Learning Q-Value
Reinforcement Learning Q-Value multidimensional
Allocation
with adjustable workload
Analysis

Greedy Allocator

Main approach:
Role Allocation



Q-Value Allocator

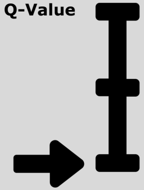
Main approach:
Reinforcement Learning

Fit → **Allocation**

Q-Value
calculation

Duration for each activity instance

Selection of resource



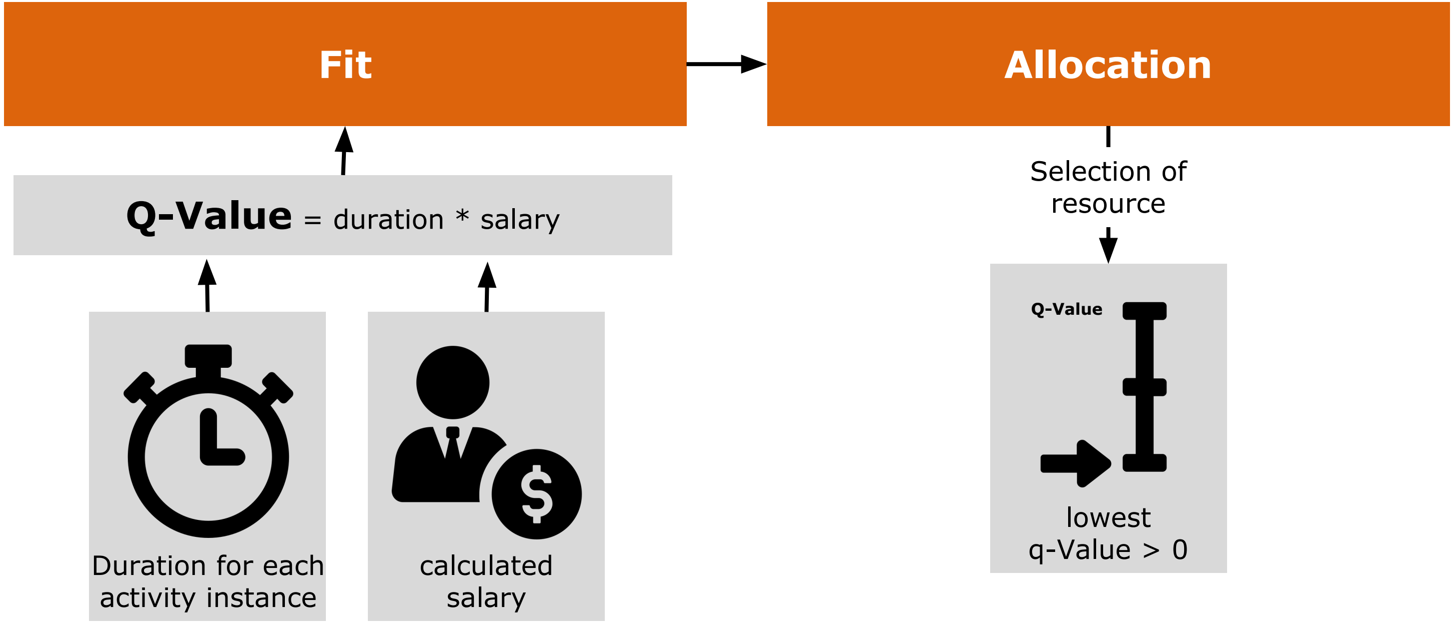
lowest
q-Value > 0

All icons from https://thenounproject.com/

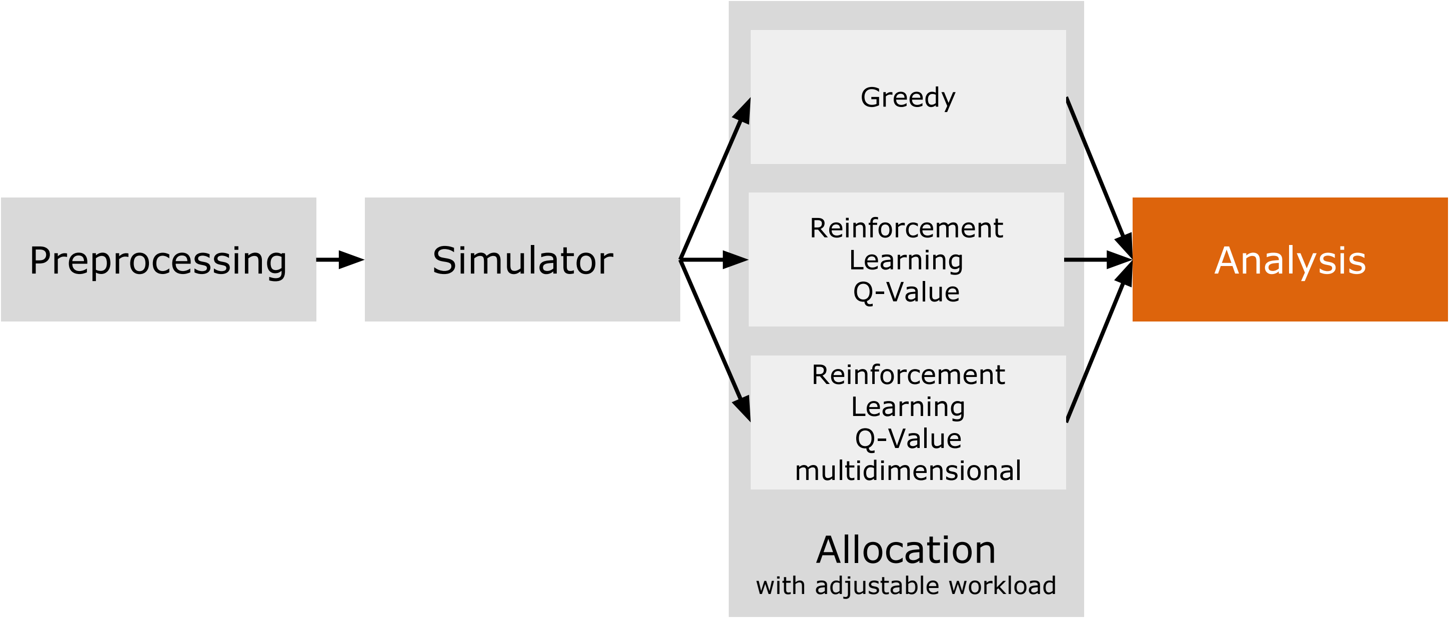
Q-Value Allocator optimizing multiple dimensions

Main approach:

Reinforcement Learning

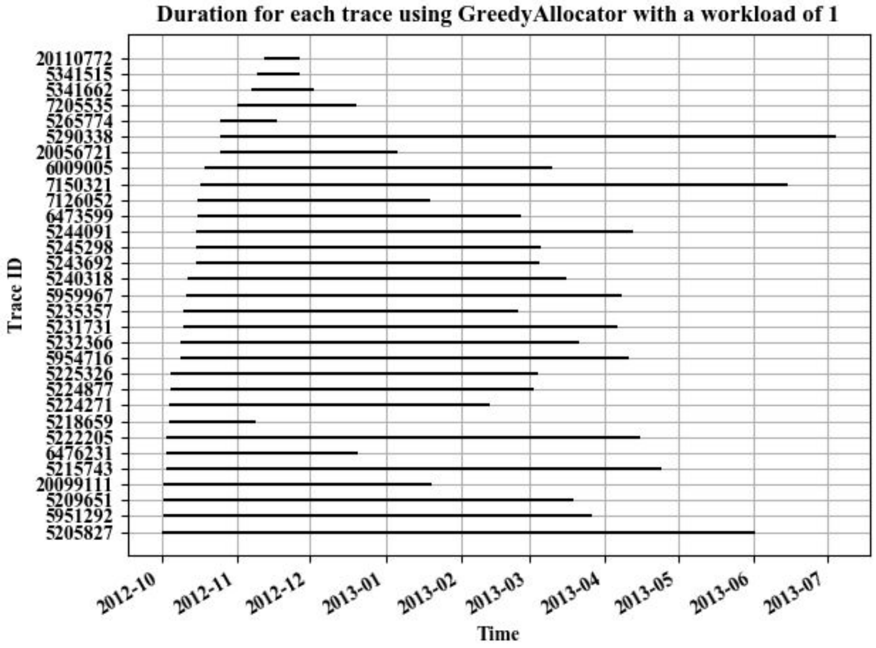


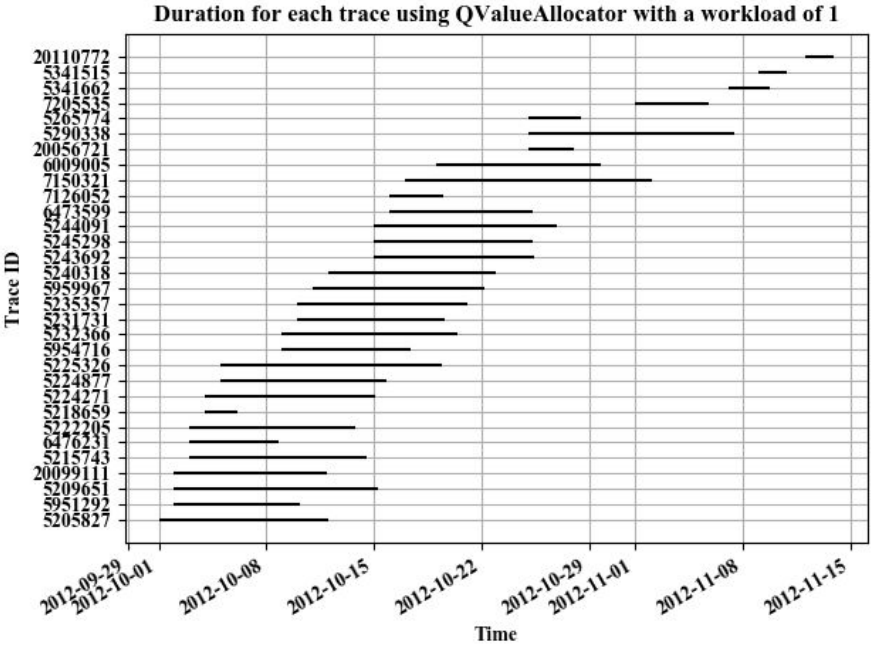
Pipeline


Preprocessing
Simulator
Greedy
Reinforcement Learning Q-Value
Reinforcement Learning Q-Value multidimensional
Allocation
with adjustable workload
Analysis

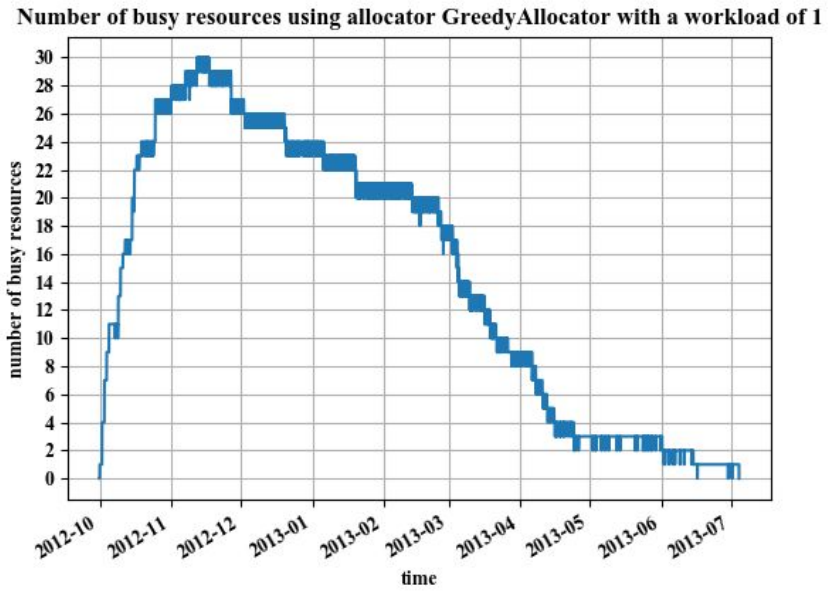
Comparison

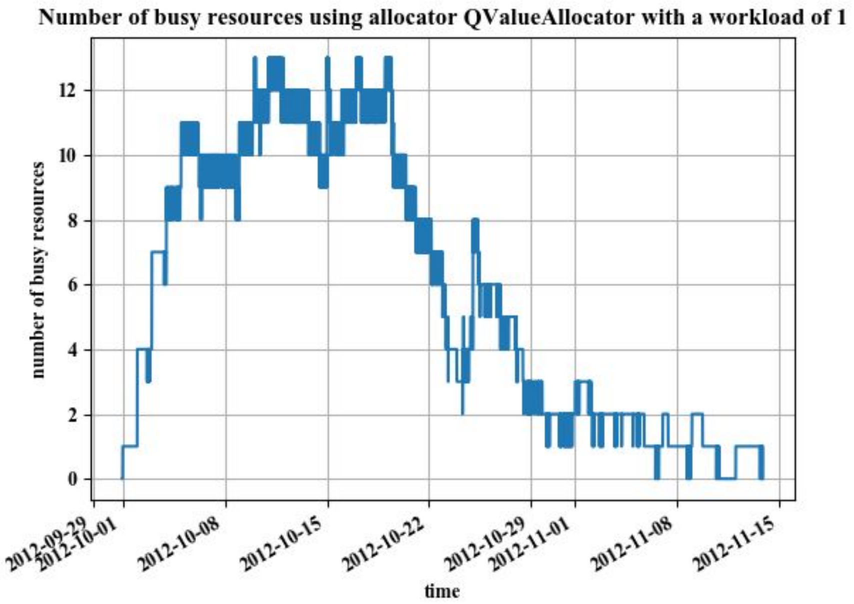
HPI Hasso Plattner Institut



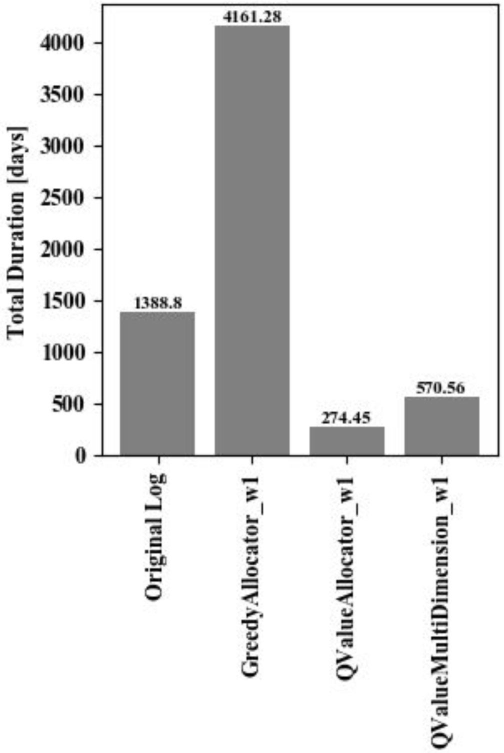


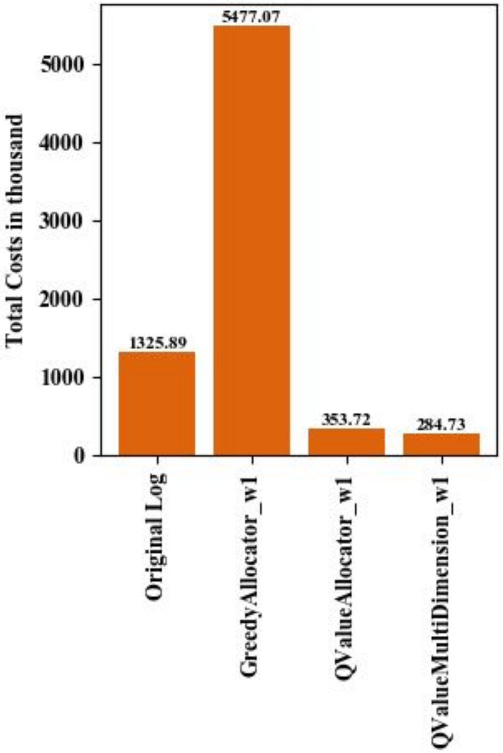
Comparison of busy resources





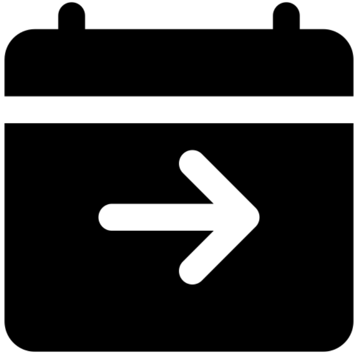
Comparison of time and cost of the three allocators





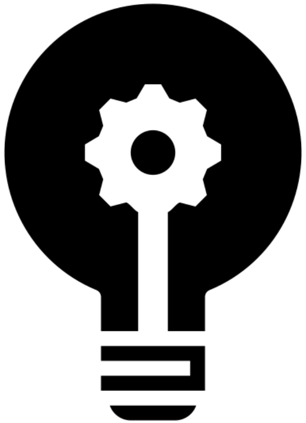
Further work

- Continual update of the Q-Value with actual needed duration
- New additional approaches:
  - Introduce priority for different activities
  - Forward looking allocation
  - Other optimization dimensions e.g. quality



Summary

- Log Quality is crucial for simulation
  -> a lot of preprocessing necessary
- realistic simulation is a challenging task (illness, worktime, holiday)
- By improving allocation with Q-Value based approaches the duration and costs are decreased
- Through adaptations different optimization goals can be set



Thank you for your attention!

Rachel Brabender and Oliver Clasen

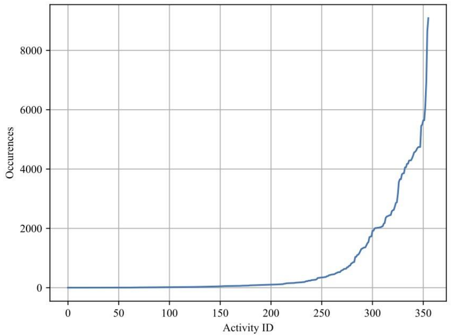
All icons from https://thenounproject.com/

Parameter

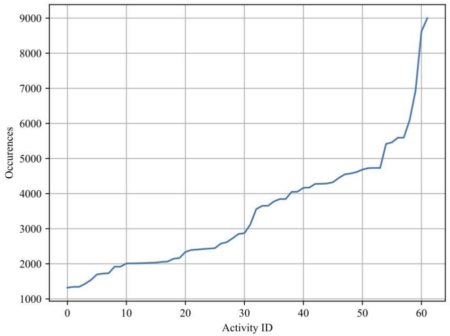
Parameter	Values
Removing loops threshold	0.5 %
Removing total occurence	0.17%
Min duration	1 h
Max duration	30 days
Simulation Interval	1 min
Q-Value Learn Rate	0.5
Q-Value γ	0.9
Workload	`<definable>`

Thinning of the dataset (visualization)

unprocessed data



filtered data



Threshold for min occurrence of an activity

- in traces = 0.5%
- in the whole dataset = 1.7%.

Removed ~ 82 % of the unique activities
Removed ~ 19 % of all activities

Workload

Each resource has a **FIFO-queue**:

- where the length is the number of activity instances a resource placed in the **discard pile**
- can be defined for each allocator **individually**
- multiple activity instances in the discard pile have **no impact on the working speed**
- if an activity is finished within an interval, the resource starts **with beginning of the next interval with the next activity**

Q-Learning reinforcement learning algorithm

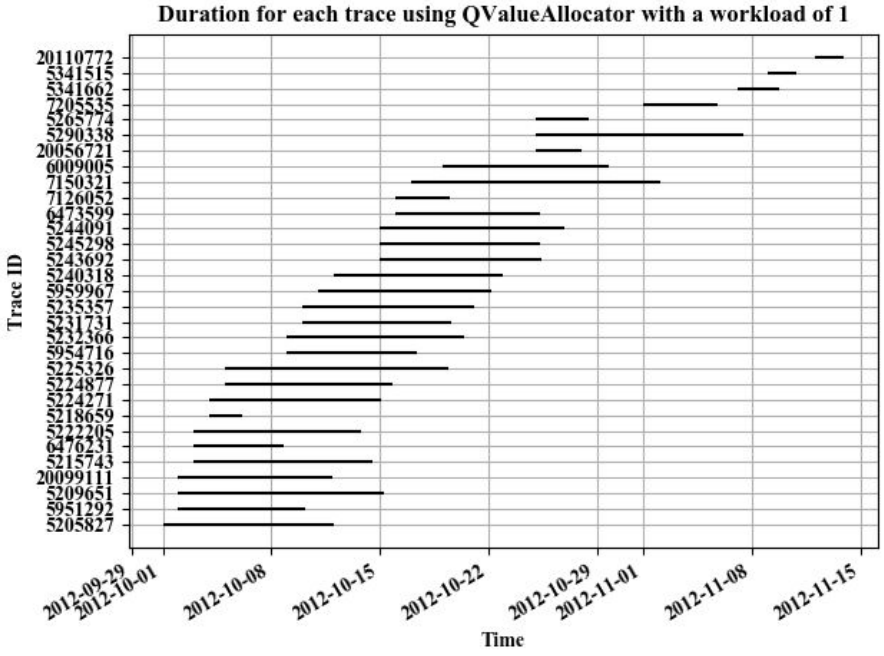
Hasso Plattner Institut

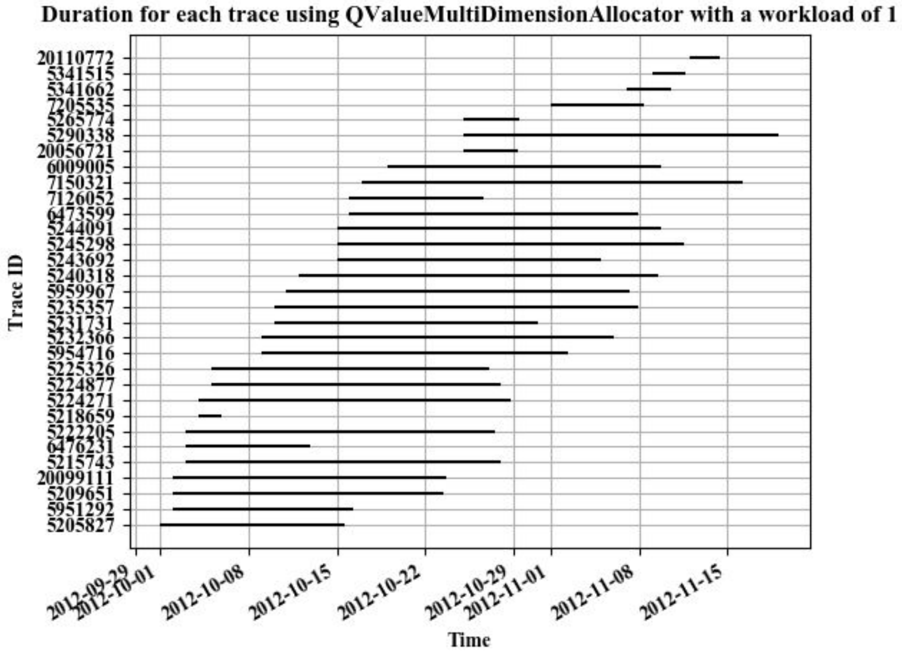
$$\underbrace{Q'(s,a)}_{\text{new Q-Value}} = \underbrace{(1-\alpha) * Q(s,a)}_{\text{current Q-Value}} + \alpha * \underbrace{(r(a)}_{\text{Reward}} + \underbrace{\gamma \min(Q(s',a'))}_{\text{Q-Value of next state}}$$

Q-Learning reinforcement learning algorithm

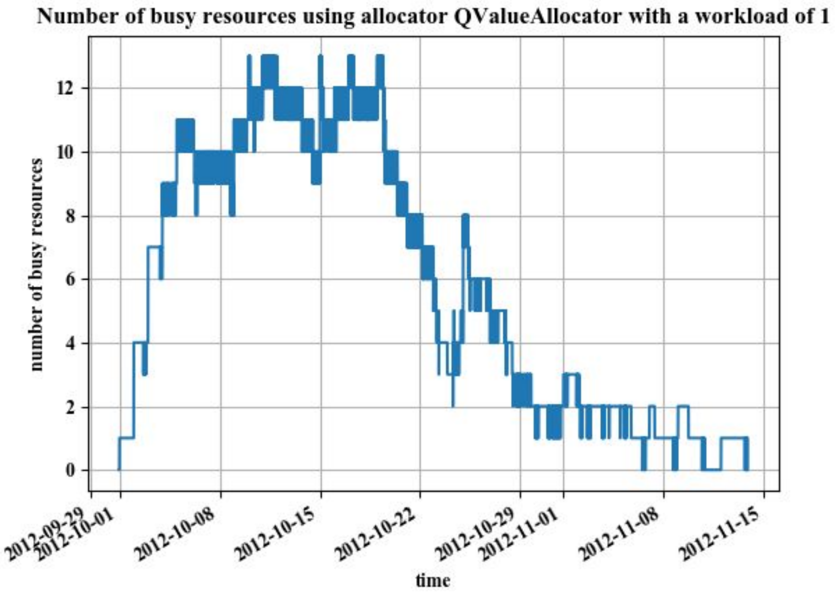
Q(s,a)	Q-Value for state s if you take action a
Q'(s,a)	new Q-Value for state s if you take action a
α	Adjust the importance of the old Q-Value in calculation of the new one
r(a)	Reward the agent gets for executing action a
argmin(Q(s',a'))	Smallest Q-Value of next state s'

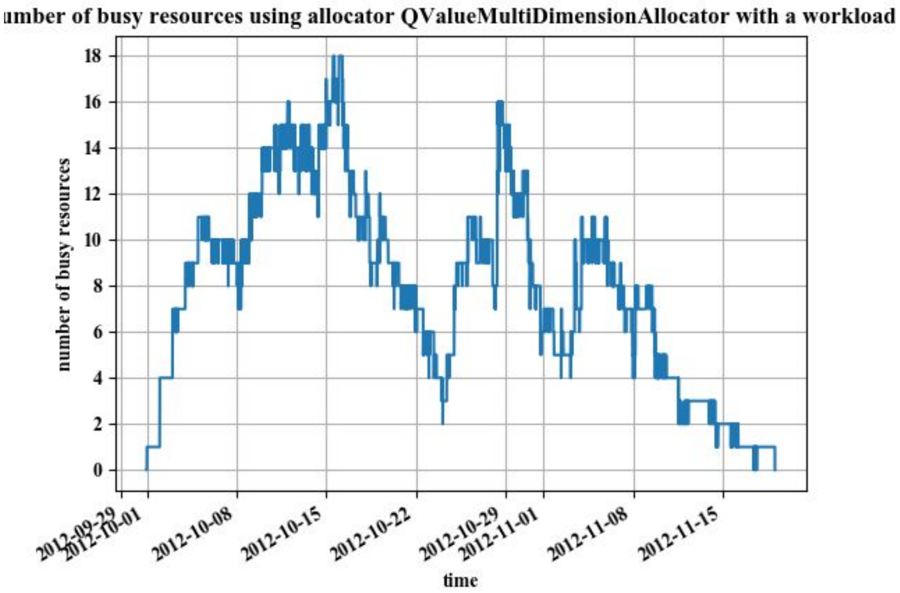
Comparison of results multidimensional Q-Value



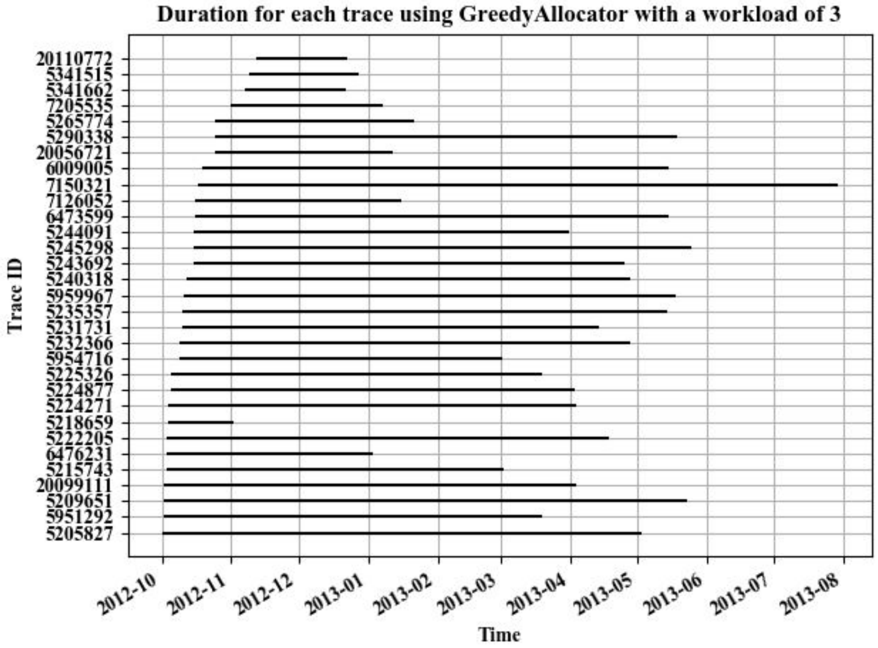


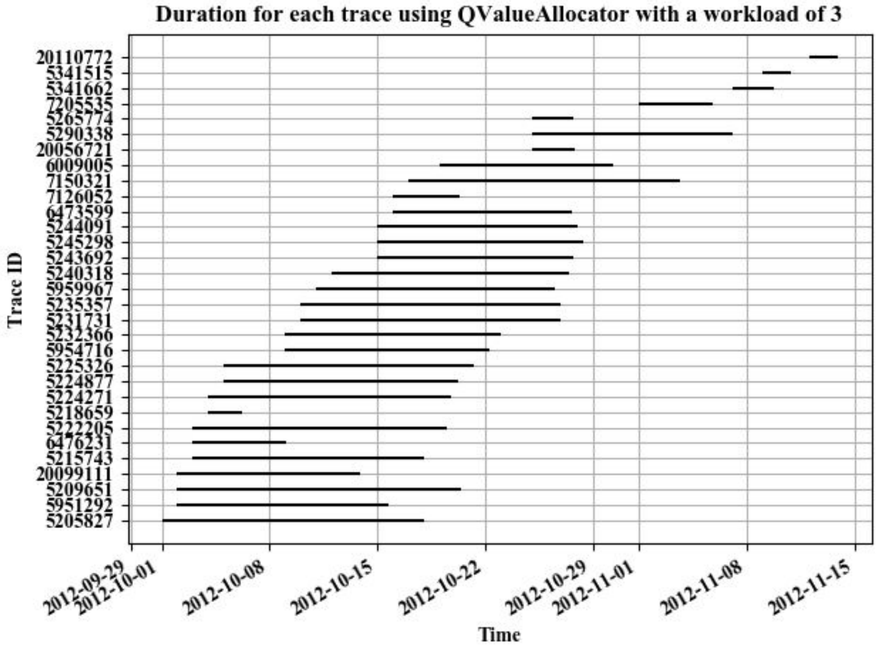
Comparison of busy resources





Comparison of results with workload = 3





Comparison of busy resources