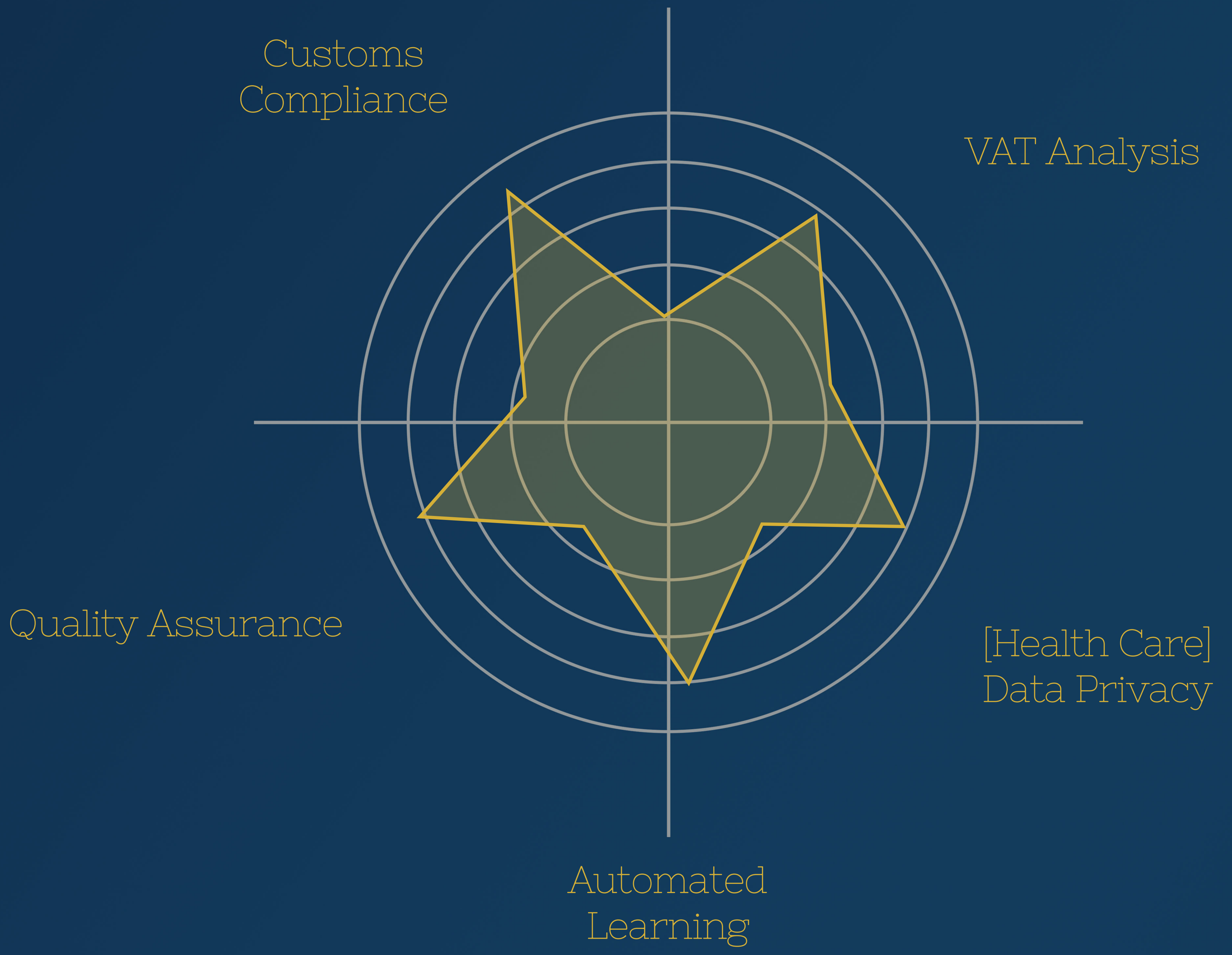
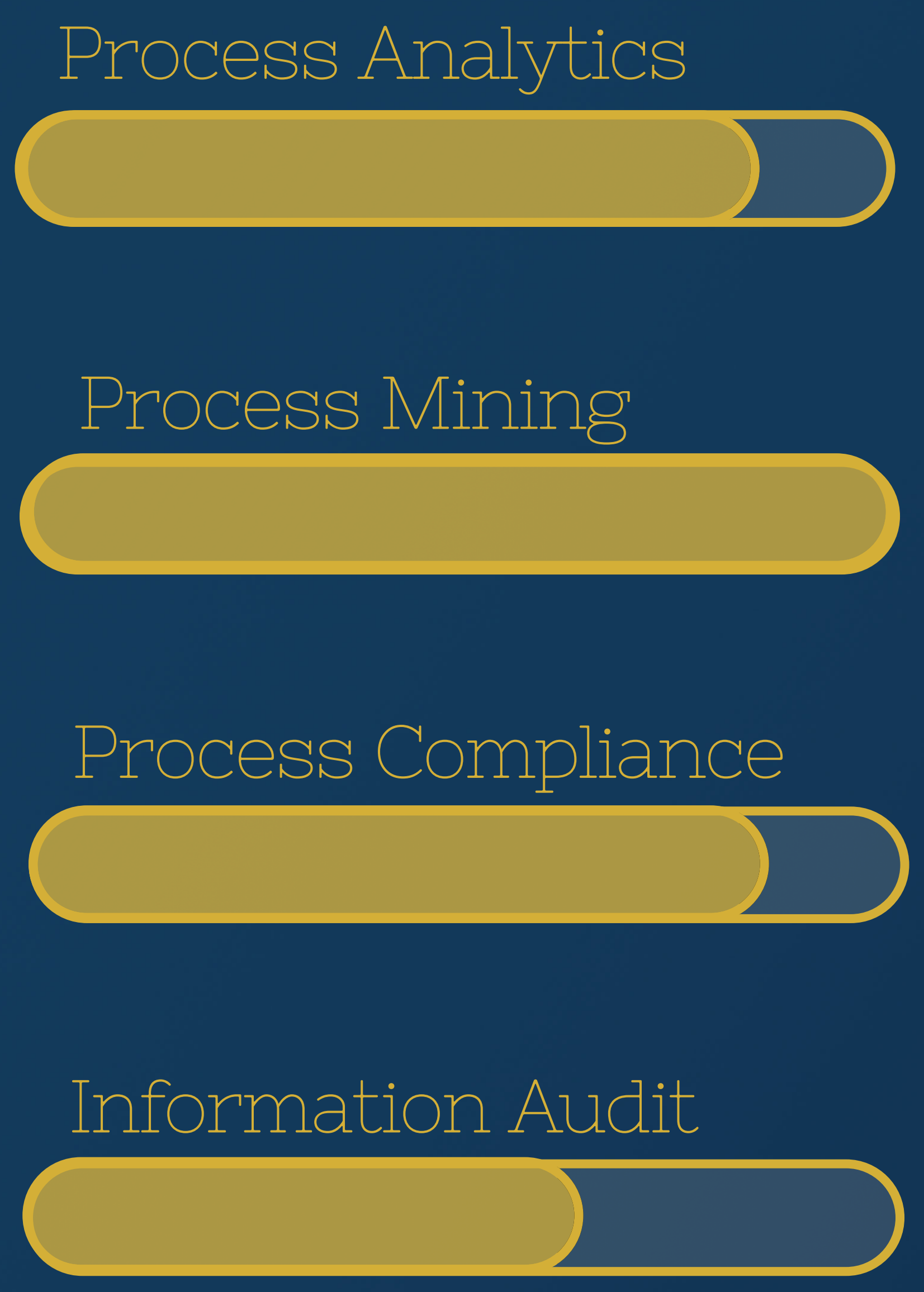
Faiza A. Bukhsh
Assistant Professor

Enthusiastic assistant profession
Interested in using intelligent event-driven techniques such as process mining to improve the efficiency and effectiveness of information compliance while keeping in view the normative and empirical privacy perspective.

Domains of Expertise





Optimal Scenario Mining for Business Strategy Decision-making through Process Mining

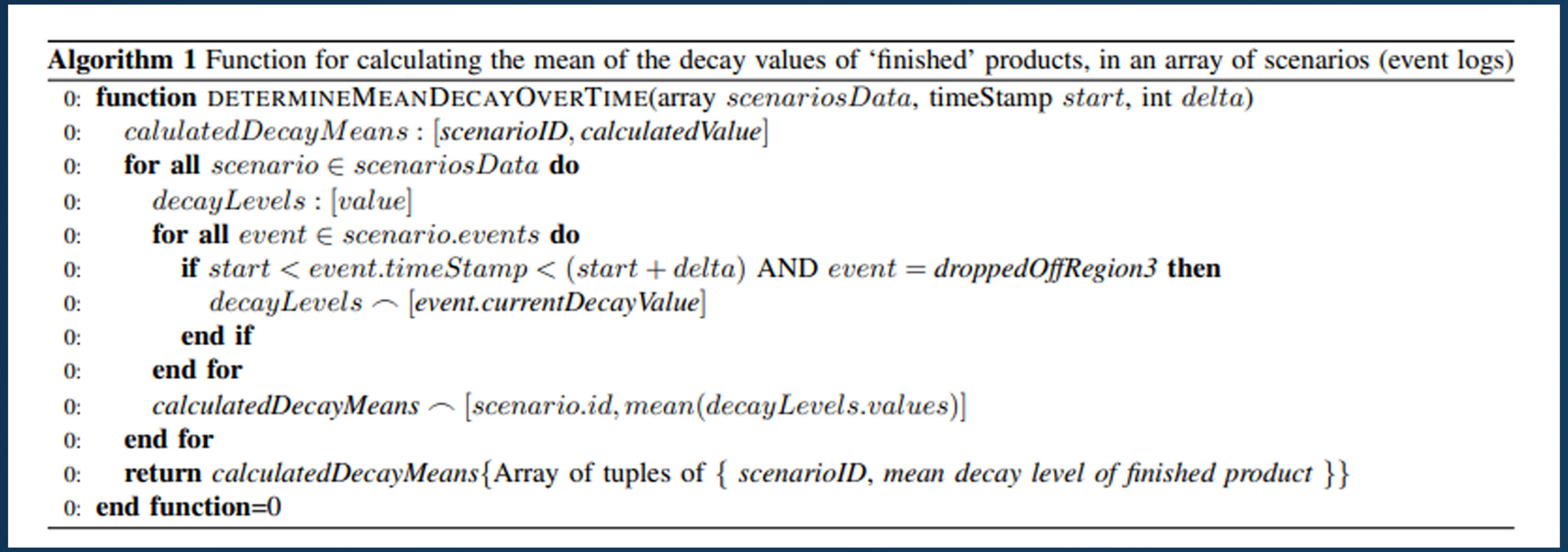
1) What are suitable performance indicators in a business in a particular domain?
2) How can the event logs be used to derive quality measures?
3) How can event log analysis be used to narrow down possible business scenarios to comply with the requirements of the performance indicators?

Recommendations

- The sample size in the time intervals needs to be considered.

- Statistical analysis should be taken into account when processing the results. For example, a combination of performance indicators instead of just the mean values in these experiments

Experiment 1



Algorithm 1 Function for calculating the mean of the decay values of 'finished' products, in an array of scenarios (event logs)

```
function DETERMINEMEANDECAYOVERTIME(array $scenariosData$, timeStamp $start$, int $delta$)
  $calulatedDecayMeans : [scenarioID, calculatedValue]$
  for all $scenario \in scenariosData$ do
    $decayLevels : [value]$
    for all $event \in scenario.events$ do
      if $start < event.timeStamp < (start + delta)$ AND $event = droppedOffRegion3$ then
        $decayLevels \frown [event.currentDecayValue]$
      end if
    end for
    $calculatedDecayMeans \frown [scenario.id, mean(decayLevels.values)]$
  end for
  return $calculatedDecayMeans${Array of tuples of { $scenarioID$, $mean decay level of finished product$ }}
end function=0
```

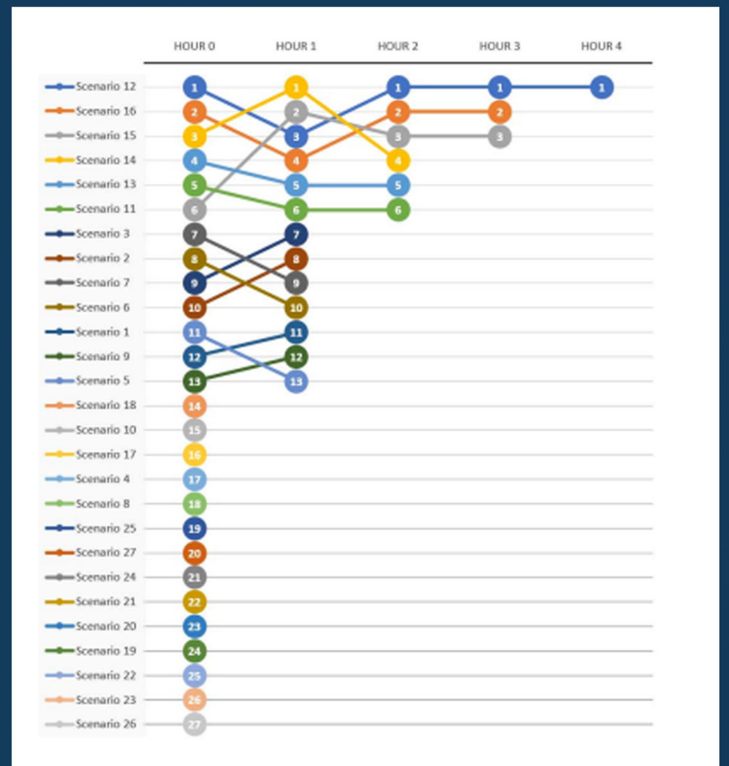
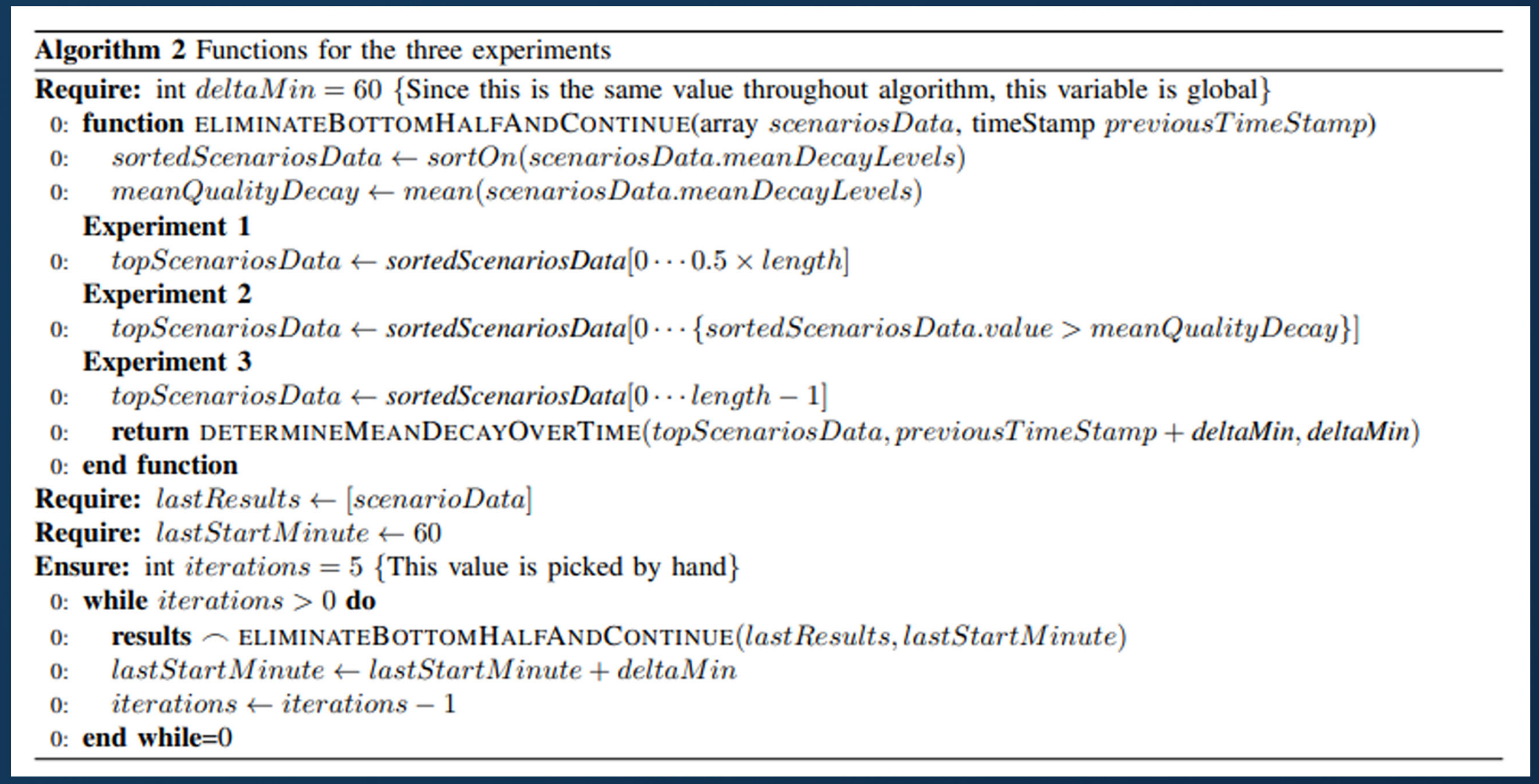


Fig. 1. Visualisation of experiment 1. After every hour, half of the scenarios are discarded. Numbers in the data points represent the current ranking.

Experiment 2



Algorithm 2 Functions for the three experiments

```
Require: int $deltaMin = 60$ {Since this is the same value throughout algorithm, this variable is global}
function ELIMINATEBOTTOMHALFANDCONTINUE(array $scenariosData$, timeStamp $previousTimeStamp$)
  $sortedScenariosData \leftarrow sortOn(scenariosData.meanDecayLevels)$
  $meanQualityDecay \leftarrow mean(scenariosData.meanDecayLevels)$
 Experiment 1
  $topScenariosData \leftarrow sortedScenariosData[0 \cdots 0.5 \times length]$
 Experiment 2
  $topScenariosData \leftarrow sortedScenariosData[0 \cdots \{sortedScenariosData.value > meanQualityDecay\}]$
 Experiment 3
  $topScenariosData \leftarrow sortedScenariosData[0 \cdots length - 1]$
  return DETERMINEMEANDECAYOVERTIME($topScenariosData, previousTimeStamp + deltaMin, deltaMin$)
end function
Require: $lastResults \leftarrow [scenarioData]$
Require: $lastStartMinute \leftarrow 60$
Ensure: int $iterations = 5$ {This value is picked by hand}
while $iterations > 0$ do
  results $\frown$ ELIMINATEBOTTOMHALFANDCONTINUE($lastResults, lastStartMinute$)
  $lastStartMinute \leftarrow lastStartMinute + deltaMin$
  $iterations \leftarrow iterations - 1$
end while=0
```

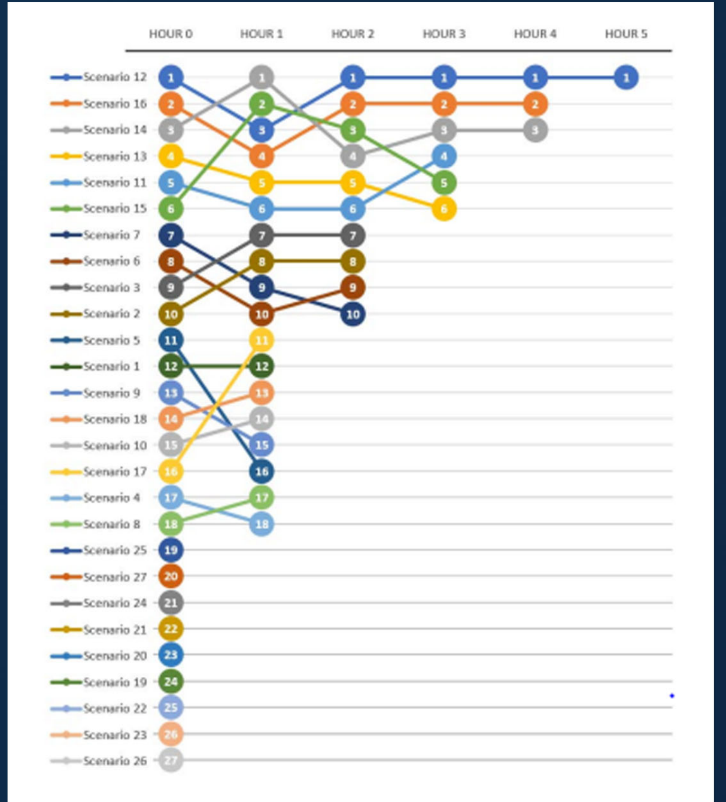


Fig. 2. Visualisation of experiment 2. After ever hour, scenarios performing worse than average are discarded. Numbers in the data points represent the current ranking.