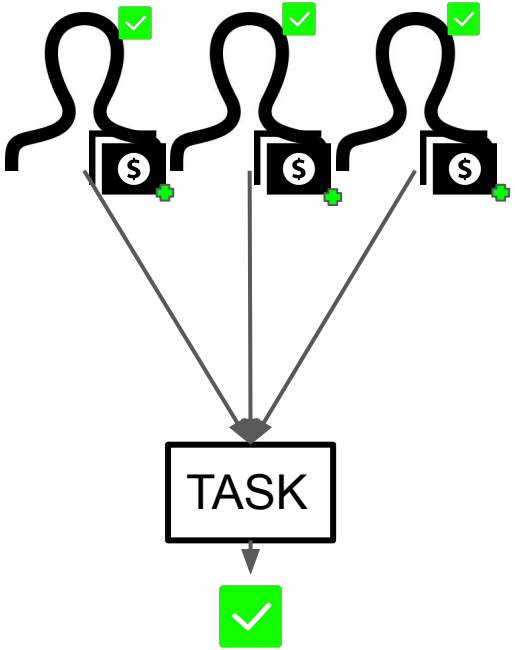
Analysis of Slashing and Reputation System Simulation

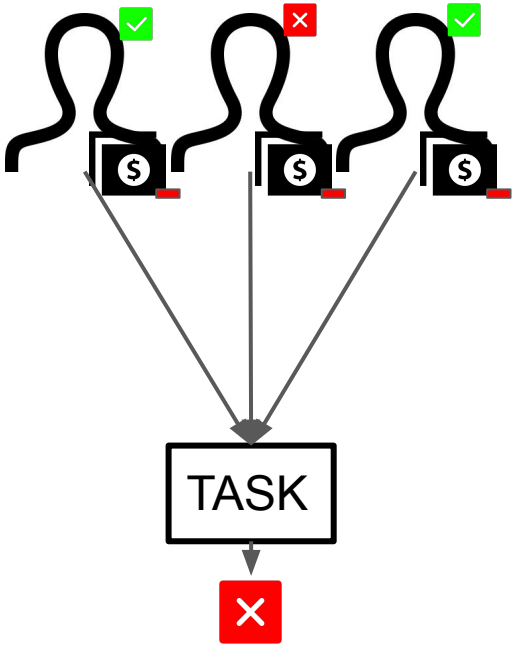
13/09/2024

Henry MONT
Intern – DRIM

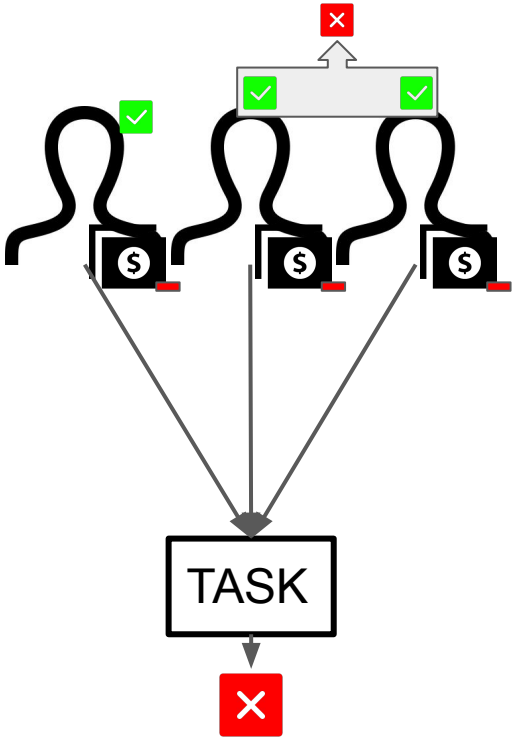
Blind Slashing: Context



Successful execution



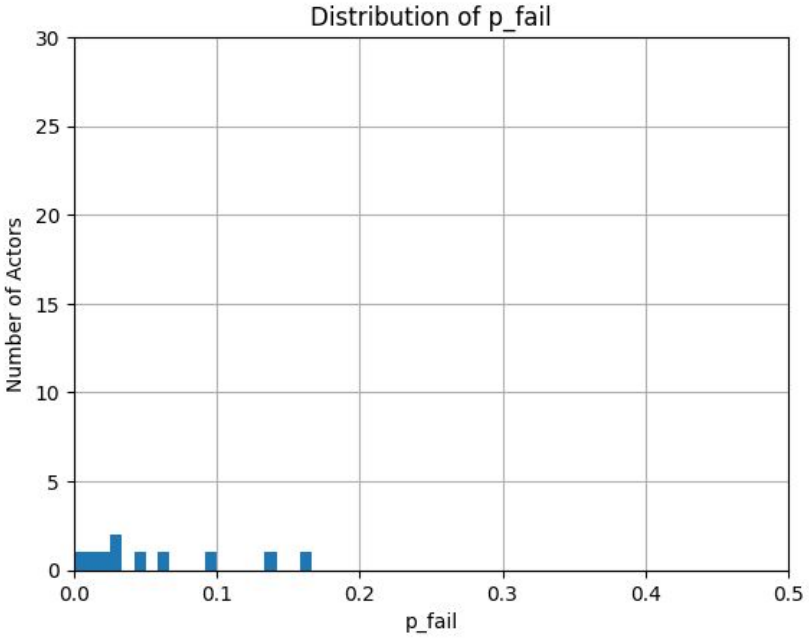
Individual failure



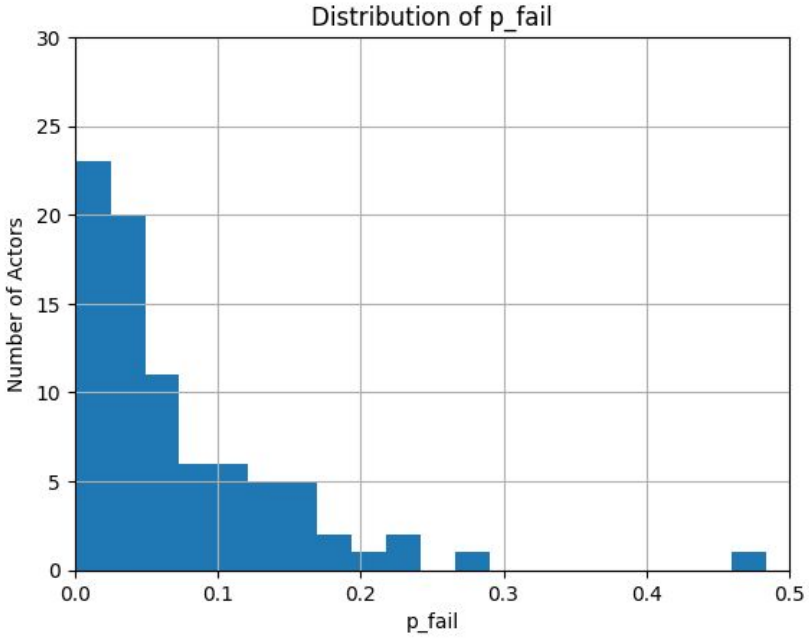
Combination failure (not covered yet)

Further analysis of slashing system simulation.

Starting with 10 actors

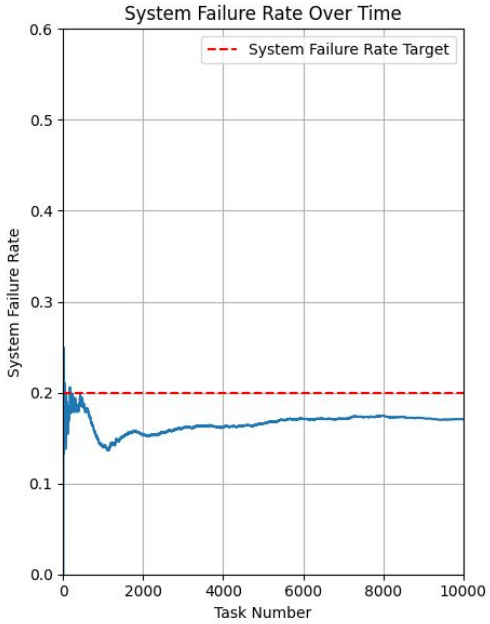


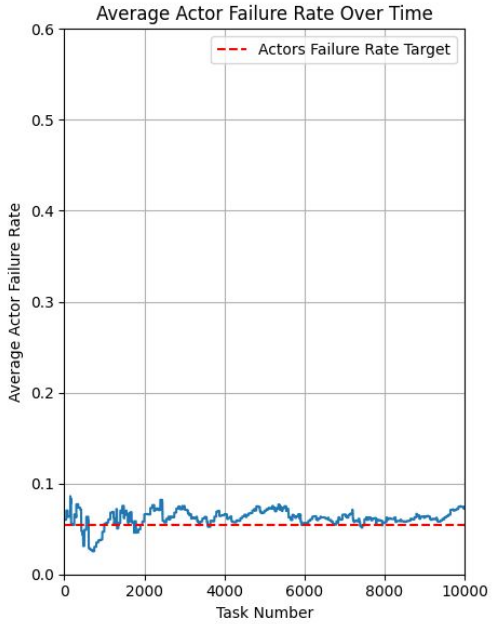
Before

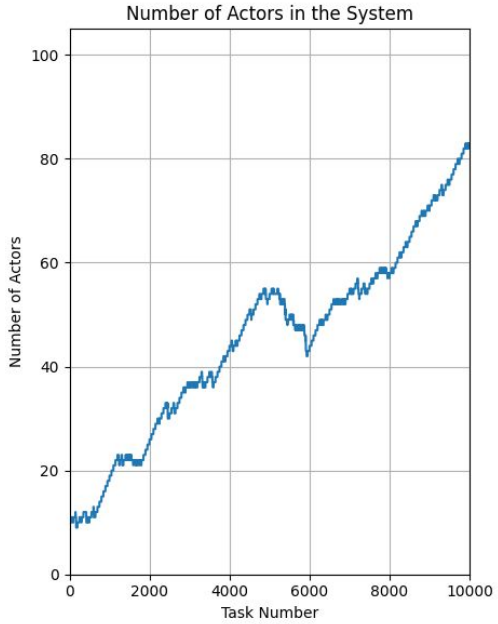


After

Starting with 10 actors

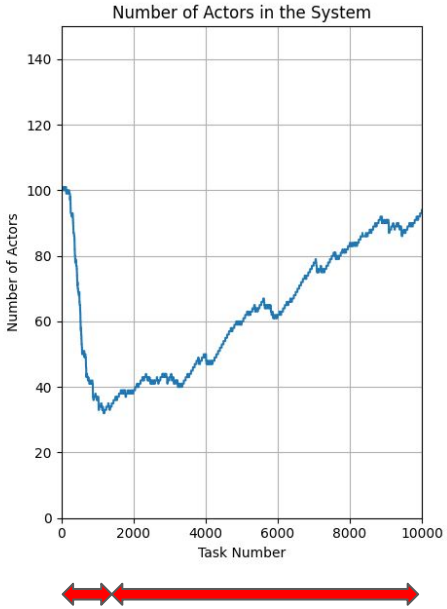






Total successful tasks: 8294	Classification Metrics:
Total failed tasks: 1706	Accuracy: 0.8
Total number of actors: 210	Precision: 0.952755905511811
Final number of actors: 83	Recall: 0.7707006369426752

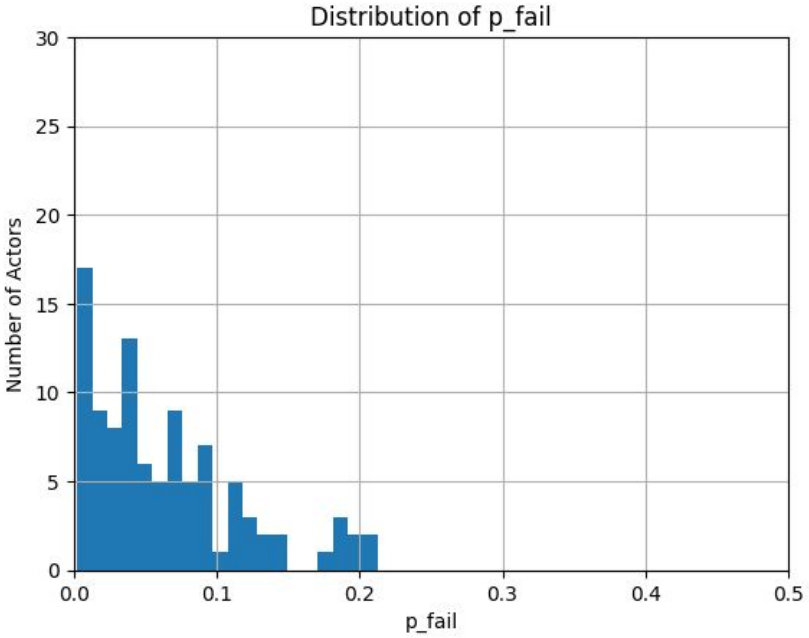
Separate Analysis of the Downward and Upward Sections



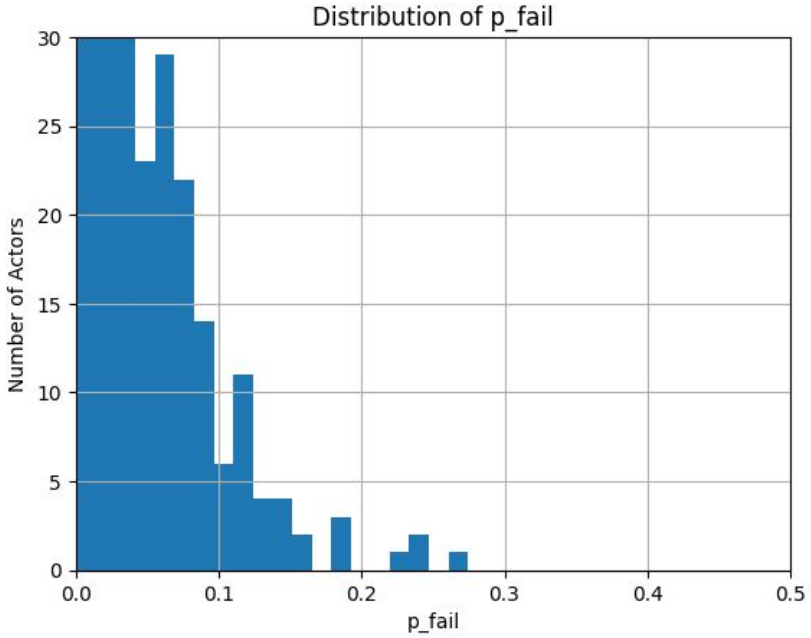
Total failed tasks: 365/1500	Classification Metrics:
Actors added: 130	Accuracy: 0.7846153846153846
Final number of actors: 37	Precision: 0.8709677419354839
Ratio of ruined actors: 71.538 %	Recall: 0.8350515463917526

Total failed tasks: 1848/10000	Classification Metrics:
Actors added: 170	Accuracy: 0.7666666666666667
Final number of actors: 94	Precision: 0.9223300970873787
Ratio of ruined actors: 66.471 %	Recall: 0.7786885245901639

Overall lower actors failure rate

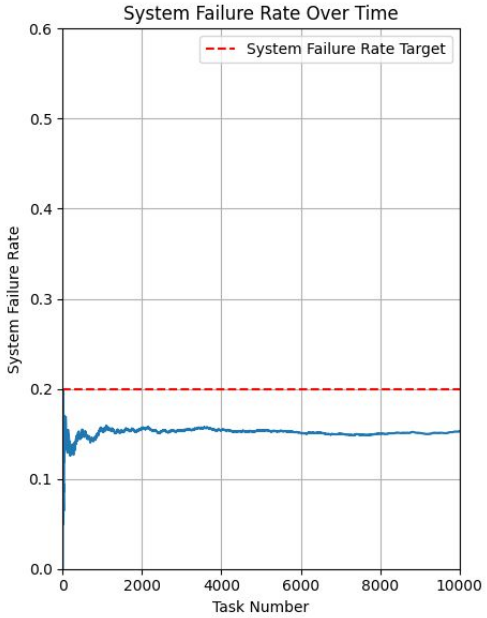


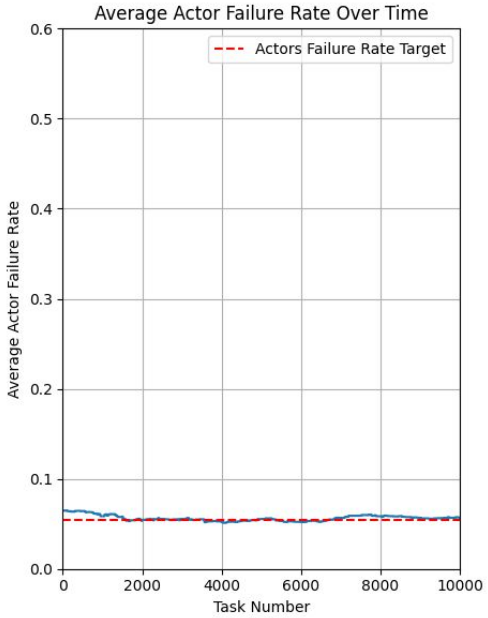
Before

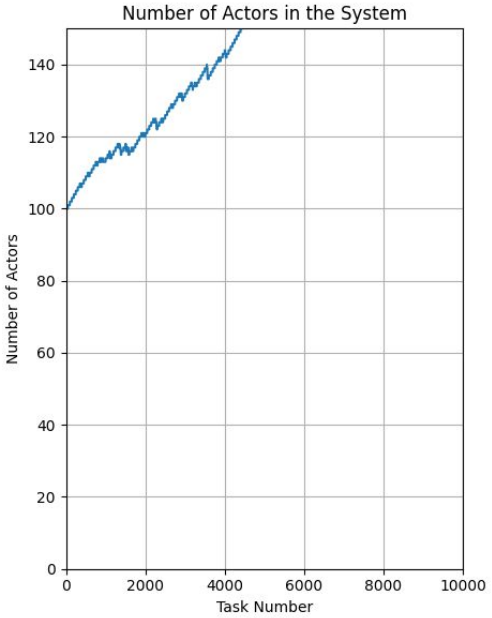


After

Overall lower actors failure rate







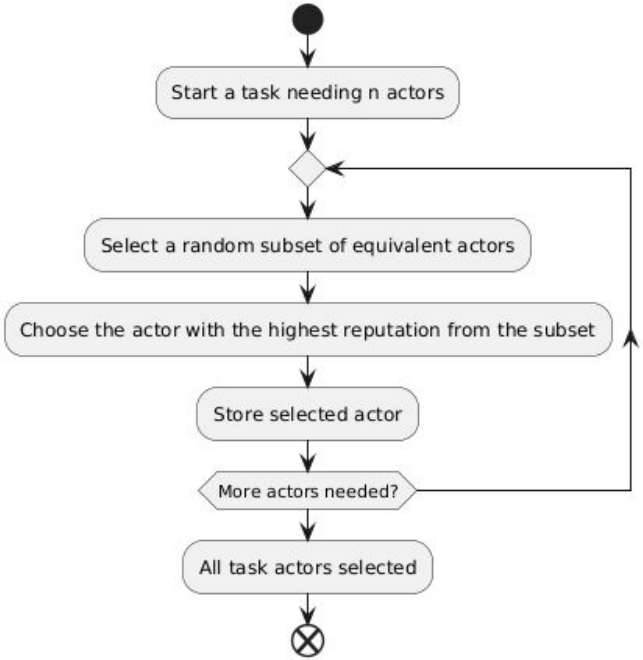
Total successful tasks: 8472	Classification Metrics:
Total failed tasks: 1528	Accuracy: 0.6466666666666666
Total number of actors: 300	Precision: 0.9305555555555556
Final number of actors: 228	Recall: 0.39880952380952384

Analysis of the Reputation System Simulation

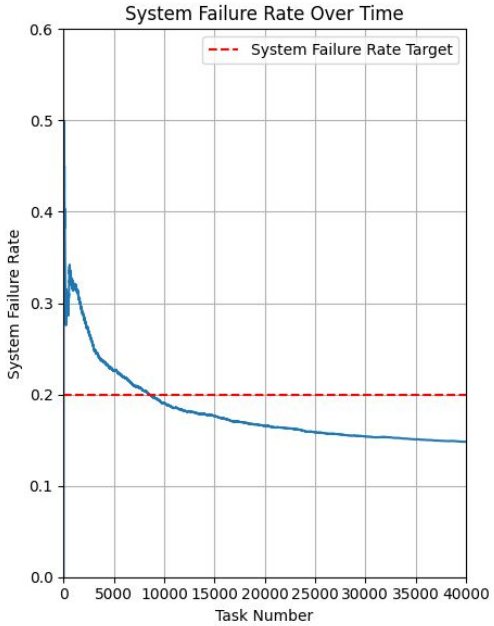
Simulation 1: Picking a random subset of competing actors

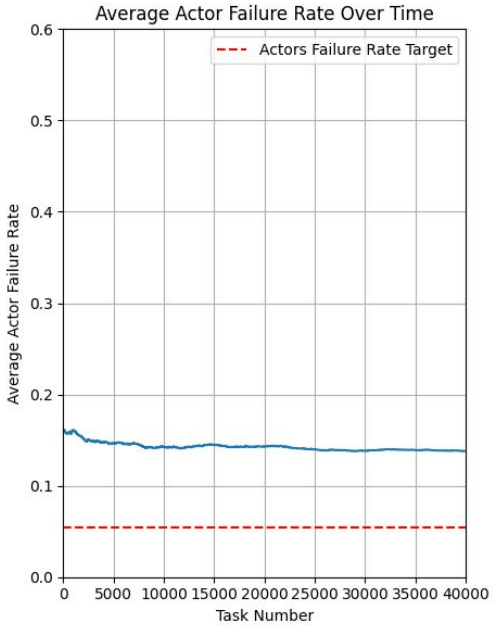
Objective: choosing one of the actors for a task.

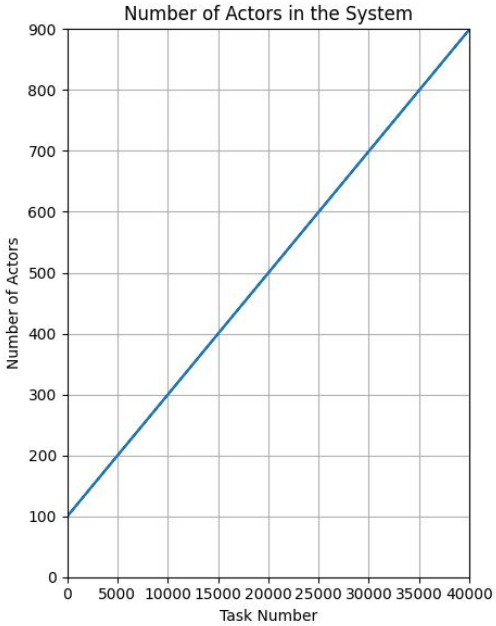
1. Select a random subset of actors considered equivalent for the task (fulfilling the same purpose).
2. From this subset, choose the actor with the highest reputation.
3. Repeat steps 1 and 2 for each of the n actors needed for the task.



Simulation 1: Picking a random subset of competing actors



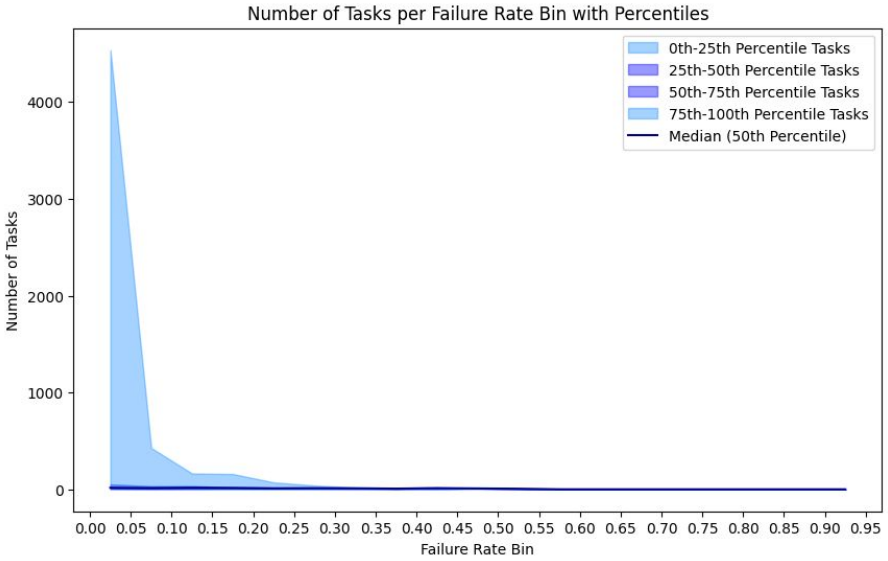


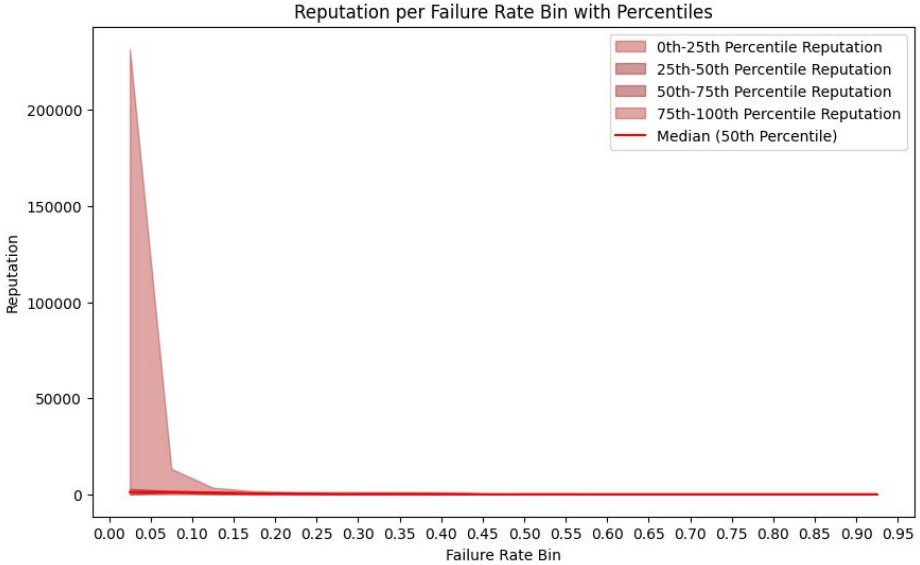


Failure rate target: 0.2	Total successful tasks: 34074
Individual failure rate target: 0.054	Total failed tasks: 5926
	Total number of actors: 900
	Final number of actors: 900

⇒ Seems to be working, but very unpredictable behaviour depending on the parameters.

Simulation 2: Picking actors more often depending on their reputation

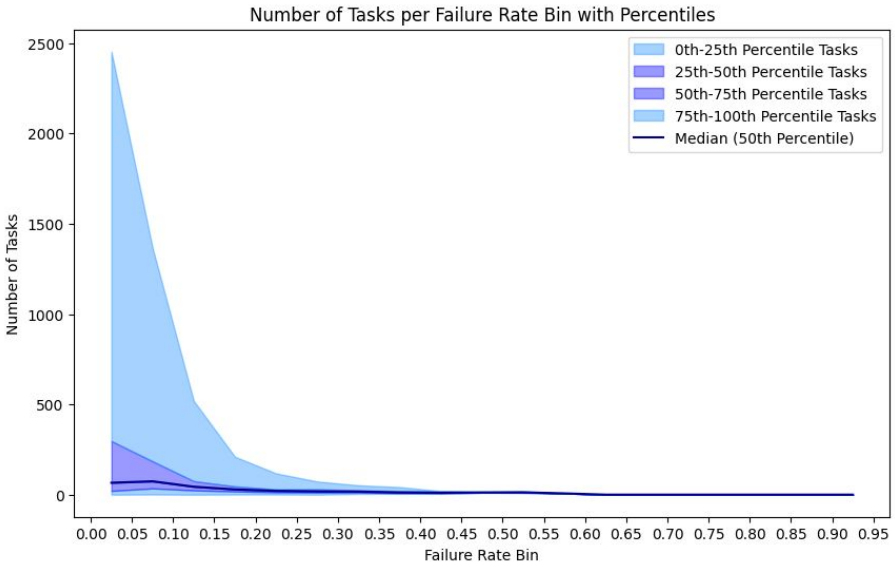


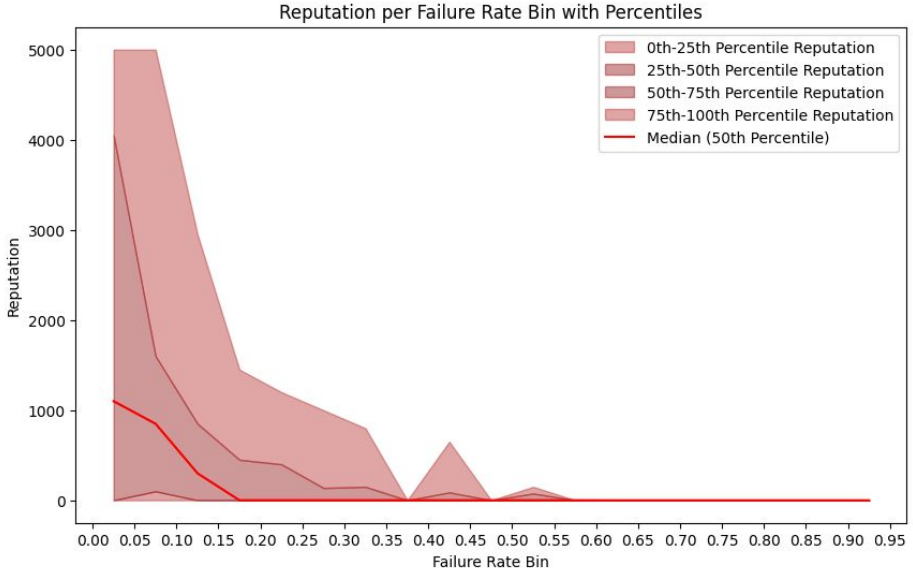


Failure rate target: 0.2	Total successful tasks: 34257
Individual failure rate target: 0.054	Total failed tasks: 5743
	Total number of actors: 900
	Final number of actors: 900

⇒ Seems to be working. Is more stable and consistent than previous simulation, but all tasks are concentrated around a small pool of outliers having very high reputation. This is not desirable nor realistic.

Simulation 3: Setting maximum and minimum reputation

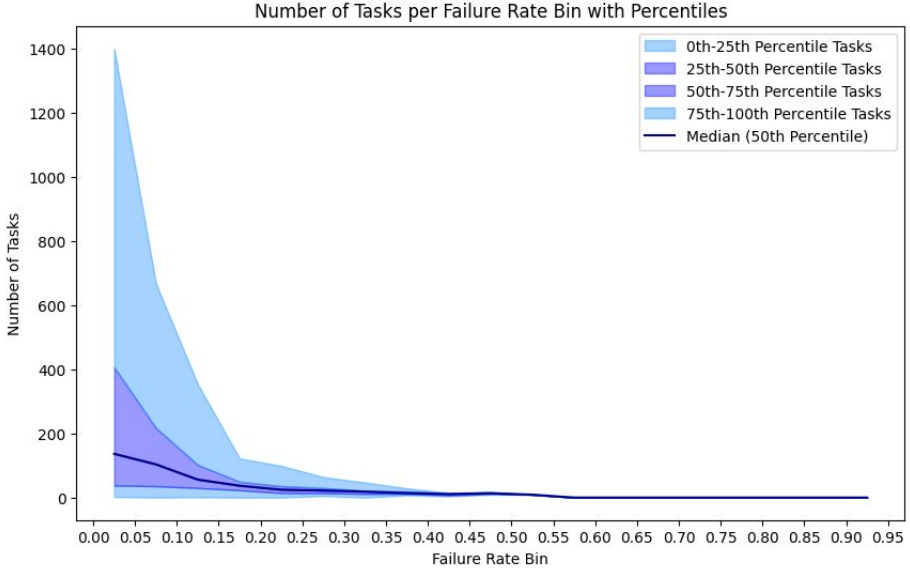


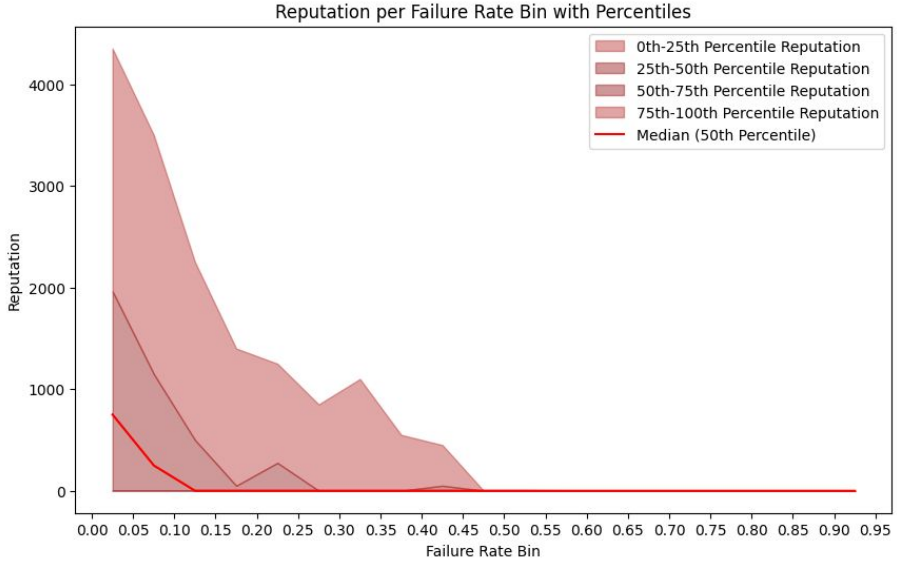


Failure rate target: 0.2	Total successful tasks: 33128
Individual failure rate target: 0.054	Total failed tasks: 6872
	Total number of actors: 900
	Final number of actors: 900

⇒ It is behaving in a more desirable way, with actors over 10% failure rate having the lowest reputation and actors below that having higher reputation. However, higher failure rate actors still seem to get plenty of tasks.

Simulation 4: Only accounting for recent reputation





Failure rate target: 0.2	Total successful tasks: 33132
Individual failure rate target: 0.054	Total failed tasks: 6868
	Total number of actors: 900
	Final number of actors: 900

⇒ We are getting similar results to the simulation 3. We should notice that the sliding window size can change the behaviour dramatically.