Statistical Analysis of the First Assignment

Michał Krepa

May 2024

1 Introduction

The purpose of this document is to provide an analysis to the first assignment based on solutions for the first assignment for Research Track 1 Course at UniGe in winter semester of 2023/2024, provided by two students - Michał Krepa and Mark Henry Dsouza. Both of these solutions had been made for a fixed solution. The goal for the assignment was to group golden tokens, that were placed in a circular pattern on the arena.

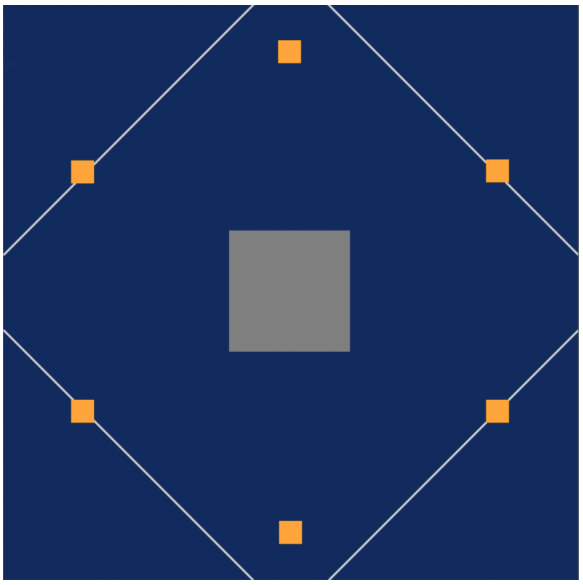

Figure 1: Circular pattern of tokens on the arena

The main idea to prepare a comparison between to algorithms is to test their universality. Both of had been prepared for a fixed problem where the tokens had been placed. To perform this the author proposes a random selection of the tokens and see what will be the success rate, and how rapidly the algorithms will be able to group the boxes together.

2 Hypotheses

The author provides a two hypotheses

Null hypothesis (H_0) There is no significant difference between the two algorithms, regarding both success rate and the average time to finish the task

Alternative Hypothesis (H_a) There is a notable discrepancy between the two algorithms, both in success rate and the average time to finish the task. One of the tasks outperforms the other and is more efficient in finalizing time within the deadline.

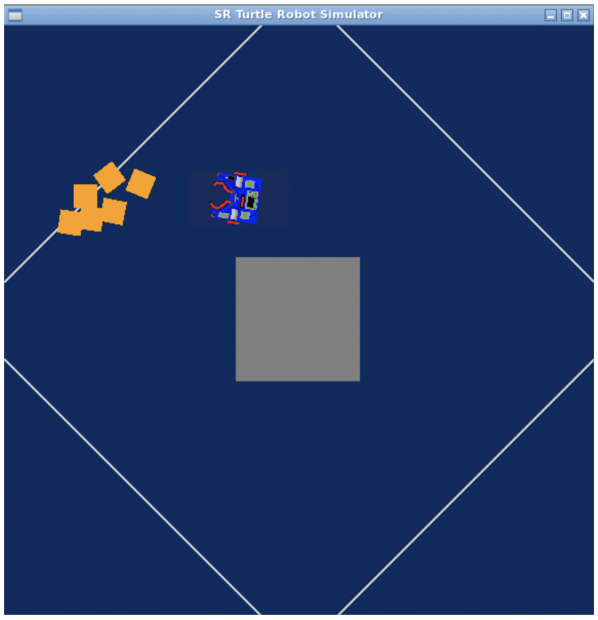

Figure 2: Solved solution for circular pattern

3 Setup

The setup provides original versions of the algorithms published on Github.com [1] [2]. The both algorithms were tested for random setups, but at the same time for having the same sessions. The simulation codes, obtained data, as well algorithms can be found online [**?**].

with a modified version of the simulation files to provide a pseudo-random generation of the tokens in with thresholds of [-1 , 1] units. Both of the algorithms had a timeout that assumed that the task execution must be done within 3 minutes and 30 seconds. If the task was still in progress - it was marked as a failure. This timeout was selected, because according to an average result rate, if the execution was usually longer than this period, robot was in a state, from which it was not able to change or finish the task

4 Results

The results were obtained by performing simulation 40 times for each set. The obtained values can be divided into two different sets.

Run number	Execution Time [s]	success
Run 1	68.2	TRUE
Run 2	77.74	TRUE
Run 3	58.33	TRUE
Run 4	74.04	TRUE
Run 5	115.19	TRUE
Run 6	82.55	TRUE
Run 7	66.87	TRUE
Run 8	91.51	TRUE
Run 9	78.41	TRUE
Run 10	95.09	TRUE
Run 11	64.18	TRUE
Run 12	72.54	TRUE
Run 13	80.64	TRUE
Run 14	79.47	TRUE
Run 15	67.52	TRUE
Run 16	114.21	TRUE
Run 17	61.02	TRUE
Run 18	94.55	TRUE
Run 19	210.07	FALSE
Run 20	69.59	TRUE

Table 1: Mark's code - Runs 1 to 20

Run number	Execution Time [s]	success
Run 21	58.63	TRUE
Run 22	66.37	TRUE
Run 23	77.9	TRUE
Run 24	105.73	TRUE
Run 25	98.41	TRUE
Run 26	81.71	TRUE
Run 27	117.02	TRUE
Run 28	210.06	FALSE
Run 29	82.16	TRUE
Run 30	162.6	TRUE
Run 31	102.64	TRUE
Run 32	98.5	TRUE
Run 33	211.96	FALSE
Run 34	211.38	FALSE
Run 35	210.73	FALSE
Run 36	125.65	TRUE
Run 37	103.38	TRUE
Run 38	87.53	TRUE
Run 39	177.56	TRUE
Run 40	150.3	TRUE

Table 2: Mark's code - Runs 21 to 40

4.1 Success Rate

The first type corresponds to the success rate of the tested algorithms. In case of logic prepared by Mark Henry Dsouza the success rate was **87.5 percent** with a value of **35 out 40** successful executions.

When it comes to code provided by Michał Krepa, the success rate is little bit lower. Michał's code delivered **80 percent** success rate, with a number of **32 out 40 runs.**

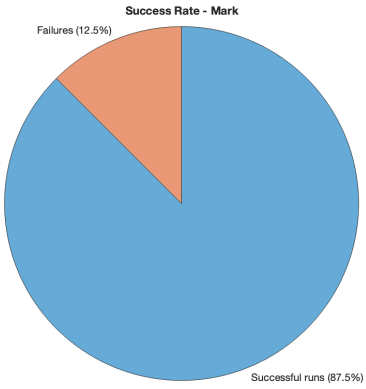

Figure 3: Success rate from Mark's code

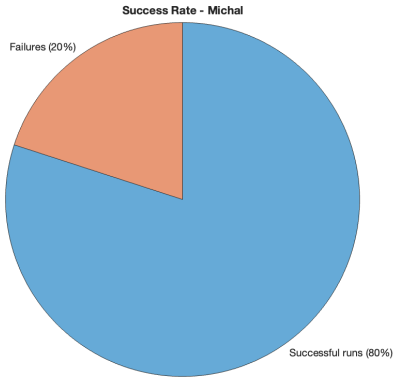

Figure 4: Success rate from Michał's code

Run number	Execution Time [s]	success
Run 1	52.09	TRUE
Run 2	66.61	TRUE
Run 3	71.11	TRUE
Run 4	210.1	FALSE
Run 5	74.95	TRUE
Run 6	103.19	TRUE
Run 7	210.1	FALSE
Run 8	126.32	TRUE
Run 9	67.18	TRUE
Run 10	210.06	FALSE
Run 11	87.26	TRUE
Run 12	76.44	TRUE
Run 13	80.48	TRUE
Run 14	210.1	FALSE
Run 15	75.28	TRUE
Run 16	69.88	TRUE
Run 17	72.66	TRUE
Run 18	77.73	TRUE
Run 19	69.65	TRUE
Run 20	70	TRUE

Table 3: Michal's code - Runs 1 to 20

Run number	Execution Time [s]	success
Run 21	89.48	TRUE
Run 22	75.22	TRUE
Run 23	214.15	FALSE
Run 24	83.98	TRUE
Run 25	210.1	FALSE
Run 26	98.86	TRUE
Run 27	210.1	FALSE
Run 28	188.72	TRUE
Run 29	73.04	TRUE
Run 30	77.82	TRUE
Run 31	210.05	FALSE
Run 32	71.31	TRUE
Run 33	68.78	TRUE
Run 34	204.3	TRUE
Run 35	94.51	TRUE
Run 36	92.14	TRUE
Run 37	70.81	TRUE
Run 38	71.68	TRUE
Run 39	82.89	TRUE
Run 40	210.07	FALSE

Table 4: Michal's code - Runs 21 to 40

4.2 Detailed results

At the beginning of this section, it is important to note that the average on graphs 6 and 5 includes only successful entries, excluding any failed tests. When we examine the details more closely, we can see that Mark's code, despite a higher fail rate, completes the task faster with an average time of approximately 95 seconds. On the other hand, Michał's code, which has a slightly better success rate, achieves an average completion time of approximately 79 seconds.

However, if the results include the failures, the averages change significantly. For Mark's code, the mean time, including failures, increases to approximately 106.6 seconds. Similarly, for Michał's code, when including the failures, the average time extends to approximately 114.5 seconds. This inclusion of failed tests in the computation of average times reflects a more comprehensive evaluation of each code's performance under varying conditions.

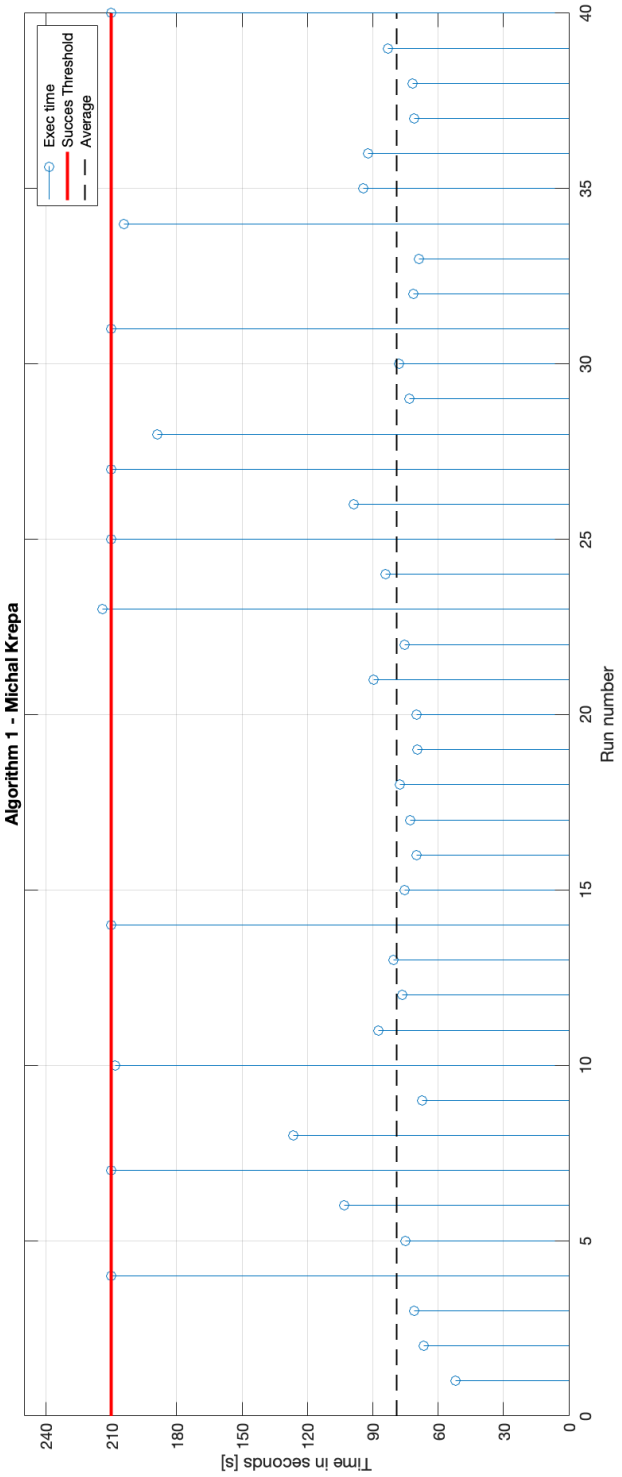

Figure 5: Detailed results - Michał

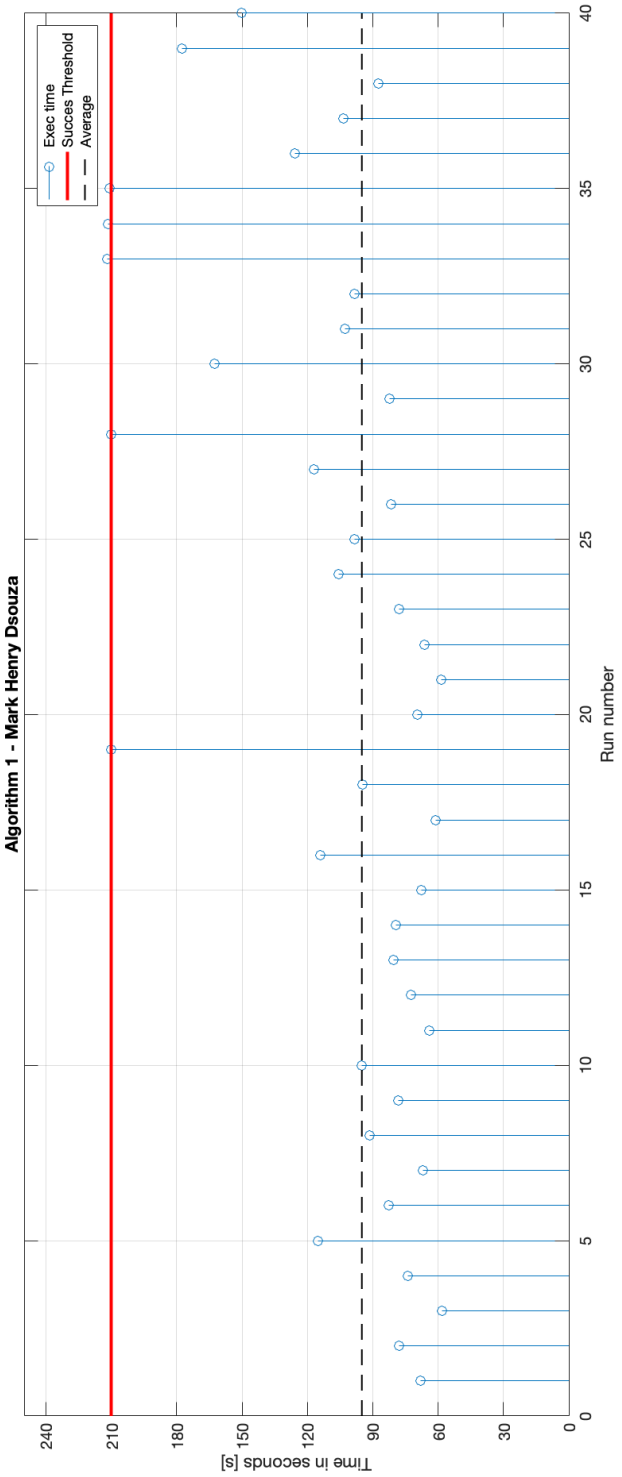

Figure 6: Detailed results - Mark

5 Statistical Analysis - Two Sample T-test

In order two compare the execution times of two algorithms a t-test was performed. Although the algorithms were tested for random values, each time they faced the same set. Therefore, in author's opinion it may be worth to perform this kind of test.

Algorithm	Mean (with failures)	standard deviation	95% Confidence Interval
Mark's	106.55	48.038	91.661 - 121.44
Michal's	114.48	59.639	95.998 - 132.96

Table 5: Detailed data between two algorithms

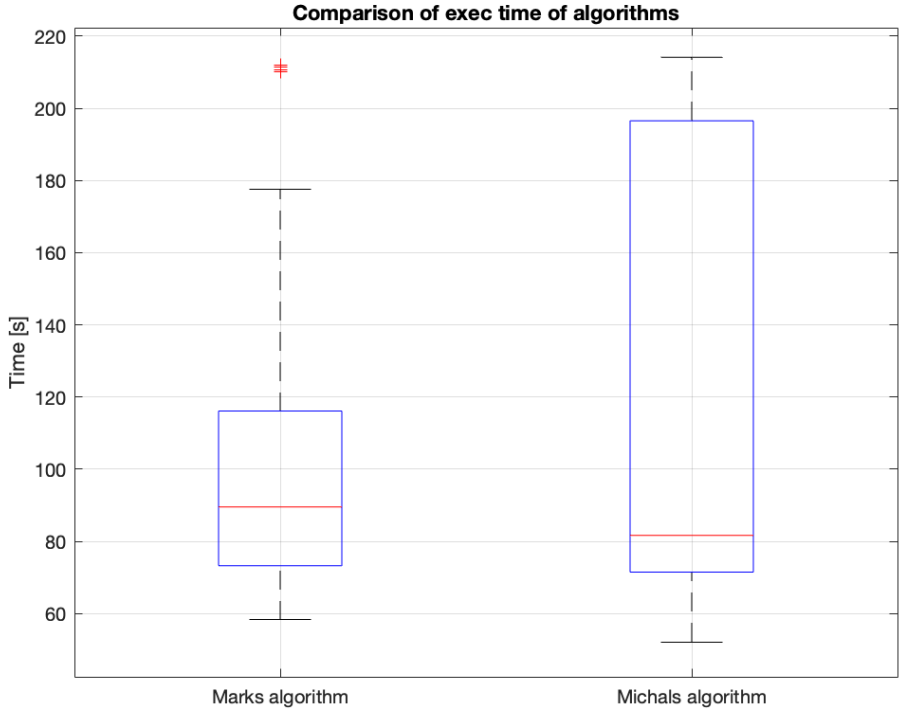

Figure 7: Comparison between two exec times

Graph data Mark's algorithm displays a median execution time of equals to 89.52 seconds, as indicated by the red line within the box on the plot. The interquartile range (IQR) for this algorithm is relatively narrow, ranging from 73.28 to 116 seconds, which suggests a lower variability in execution times around the median. Notably, there are outliers, marked by a red cross above the upper whisker, indicating occasional failures, with 5 instances.

In contrast, Michał's algorithm shows a median execution time with 81.69 seconds, despite having much broader IQR, stretching from about 71.5 to 196.51 seconds. The overall execution times for Michał's algorithm range from about 70 seconds to 200 seconds, suggesting that it can occasionally achieve very fast execution times and also swing to much slower executions.

6 Discussion

By performing analysis using *Two sample T-Test* using the also the failed samples from both sides follows the null hypothesis of this report indicating that the there are no significant differences between the two algorithms.

Nevertheless one should consider not only the performed test as also the success rate between two algorithms. In Author's opinion Mark's code, having smaller failure rate, provides more universal solution to the general problem explained in this document.

References

[1] Mark Henry Dsouza. Research track assignment - mark dsouza, 2023. If you use this software, please cite it using the metadata from this file.

[2] Michał M Krepa. Research track assignments, 2023. If you use this software, please cite it using the metadata from this file.