The application of GreedyNE algorithm [1] on a new task allocation problem

1 Introduction

This report summarizes and explains the application of GreedyNE algorithm on the new task allocation problem. This is the programming practices of Python libraries in the application of task allocation in Multi-agent Systems.

2 GreedyNE program

The Fig. 1 represents the UML Diagram of the GreedyNE algorithm application in Python. In general, the yellow boxes are to generate agents and tasks depending on their constraints. Blue boxes are the $GreedyNE$ and $GreedyNE\_LS$ algorithms. Green boxes are the calculation function to calculate the Leader distance (for the communication cost), the task utility based on resource satisfaction and communication cost, and the pay-off of moving agents between different coalitions.

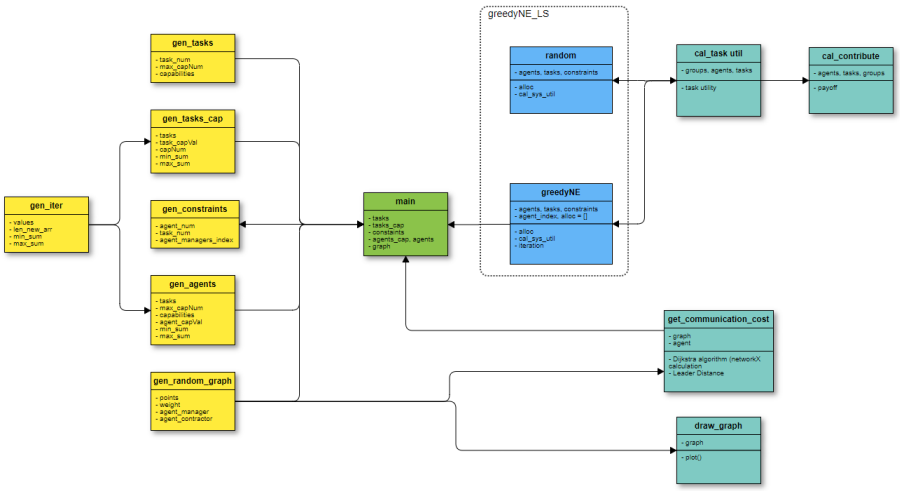

Figure 1: UML Diagram

3 Data Analysis

The Fig. 2 represents the total task utility by task number. As can be seen, the *random* Algorithm shows unreliable results, while the $GreedyNE$ and $GreedyNE\_LS$ algorithms could improve the total task utility. Moreover, the $GreedyNE\_LS$ outputs show a slightly higher in total task utility, compared to the $GreedyNE$.

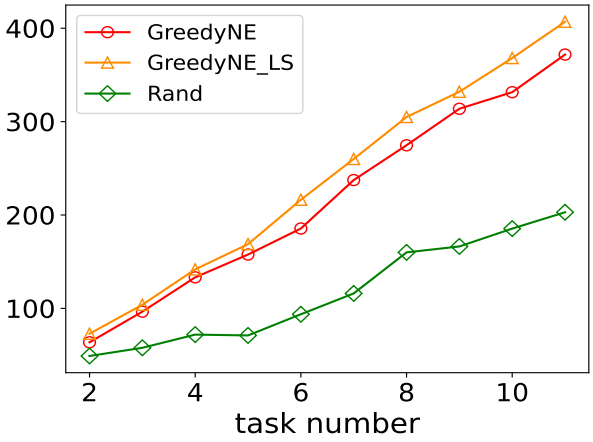

Figure 2: Total task utility

The Fig. 3 represents the computation time by task number. the $GreedyNE\_LS$ takes a slightly higher computation time, compared to the $GreedyNE$, as it also requires the *random* Algorithm to randomize the coalition at the first iteration.

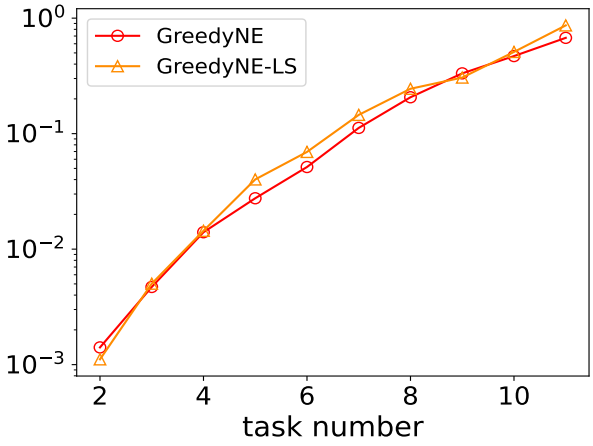

Figure 3: Computation time

The Fig. 4 represents the number of iterations by task number. This only

represents the iteration numbers provided by the $GreedyNE$ algorithm.

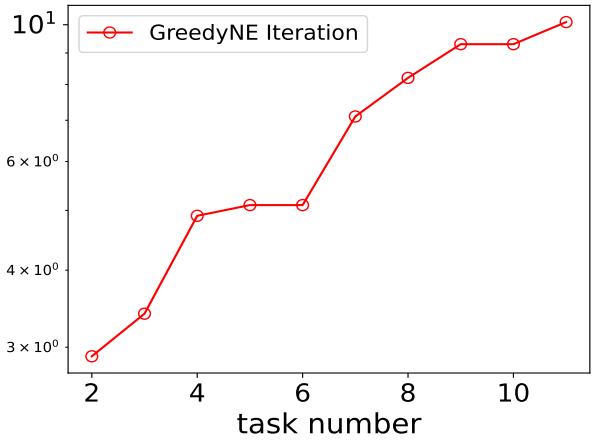

Figure 4: Iteration

4 Conclusion

This practice is inspired by applying the GreedyNE algorithm in the [1]. However, due to the complexity of calculation in task utility (based on resource satisfaction and communication cost), the program does not run large-scale experiments. A future practice direction would be running a larger-scale experiment and improving the data analysis.

References

[1] Q. Li, M. Li, B. Q. Vo, and R. Kowalczyk, "An anytime algorithm for large-scale heterogeneous task allocation," in *2020 25th International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE, 2020, pp. 206–215.