SuzukaAgent for ANAC SCML OneShot Track

Hisakawa Soto *

March 2024

1 Introduction

Since last year's tournament, the unit price consists of only two values, with a difference of one between them. As a result, it has become paramount to fulfill the quantity. QuantityOrientedAgent prioritized fulfilling the quantity and emerged victorious in last year's tournament. In this tournament, We believe that pursuing the best unit price while ensuring the fulfillment of quantity requirements is crucial.

SuzukaAgent always proposes the best unit price. However, upon acceptance, it adjusts the degree of compromise on the unit price based on the advantages or disadvantages of the situation. We consider the following three situations as disadvantageous.

- At level 0 (the quantity required for all factories at level 0 is greater than that at level 1)
- Towards the end of the day (step > 18)
- doesn't work well with other factory agents

2 Strategy

2.1 Definitions of words

- Needs: the current quantity required
- The best unit price: $\begin{cases} \text{the highest unit price} & \text{(lavel 0)} \\ \text{the lowest unit price} & \text{(level 1)} \end{cases}$
- The worst unit price: $\begin{cases} \text{the lowest unit price} & \text{(lavel 0)} \\ \text{the highest unit price} & \text{(level 1)} \end{cases}$

2.2 Proposal Strategy

SuzukaAgent proposes the following contents to all partners.

- The proposal unit price: the best unit price
- The proposal quantity: $\begin{cases} \text{needs} & \text{(the number of partners = 1)} \\ \lceil \text{needs}/2 \rceil & \text{(else)} \end{cases}$

*Graduate School of Mathematics, Kyushu University, Japan

We base this decision on the data from reinforcement learning for searching the best proposal strategy. We used Q-learning as the reinforcement learning method that utilizes simulations with the 2023 winners: QuantityOrientedAgent, CCAgent, and KanbeAgent[2]. In the Q-learning, We define the step, needs, level, the number of partners as the state, and the profit as the reward. The acceptance strategy of the learning agent prioritizes the quantity when at level 0 or steps > 18, and prioritizes the best unit price at level 1 and step ≤ 18.

2.3 Acceptance Strategy

SuzukaAgent decides whether to accept the proposals or not as follows.

$$\begin{cases} \text{Prioritize fulfilling needs} & (\text{level } 0 \text{ and step} \leq 18) \text{ or } (\text{step} = 18) \\ \text{Prioritize the best unit price, but accept even the worst unit} & \\ \text{price until reaching ''acceptance level'' acceptances per day} & (\text{level } 1 \text{ and step} < 18) \\ \text{Maximize the profit} & (\text{step} > 18) \end{cases}$$

The degree of compromise can be adjusted to work well with other factory agents by updating acceptance level. When step > 18, the priority shifts from fulfilling needs to maximizing the profit, and the proposals exceeding the required quantity may be accepted. The initial value of acceptance level is 0, and it is updated at the end of each day as shown in Figure1.

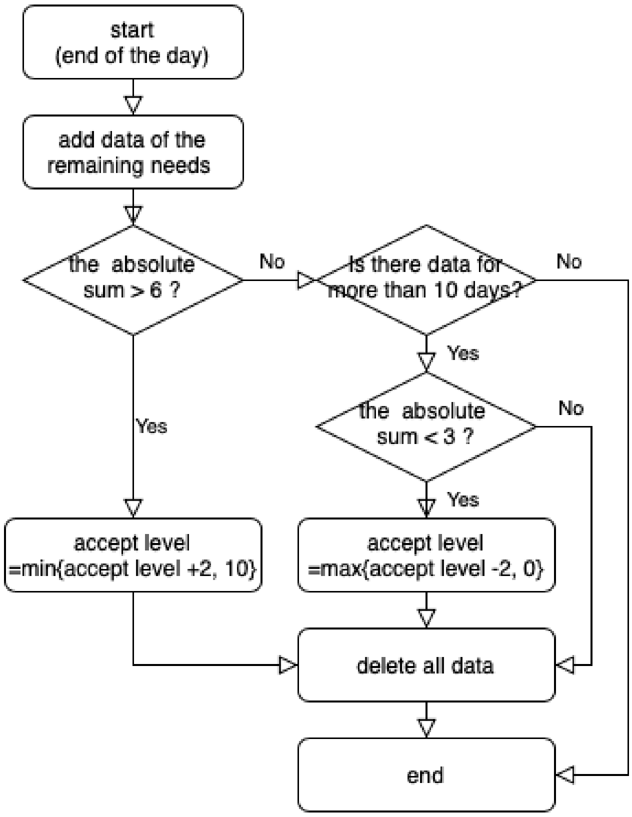
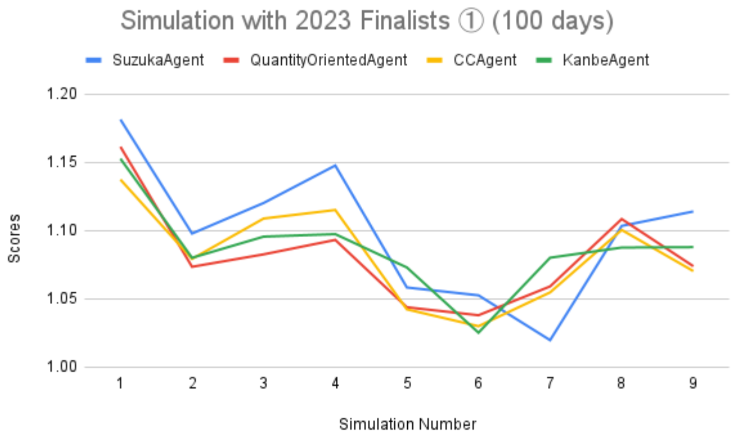


Figure 1: the way of updata acceptance level

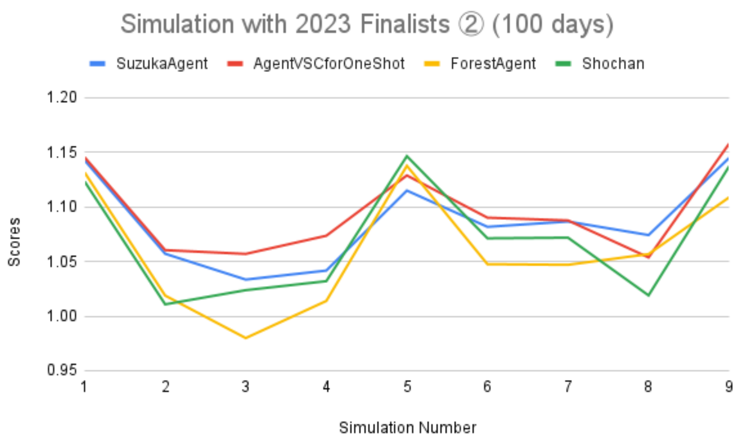
3 Evaluation

We evaluated SuzukaAgent in simulations with some of the 2023 finalists. The results of simulations with QuantityOrientedAgent, CCAgent, and KanbeAgent, which were used in the reinforcement learning, are shown in Figure 2. The results of simulations with AgentVSCforOneShot, ForestAgent, and Shochan[3], which were not used in the reinforcement learning, are shown in Figure 3. The results of simulations with varying numbers of days are shown in Figure 4.



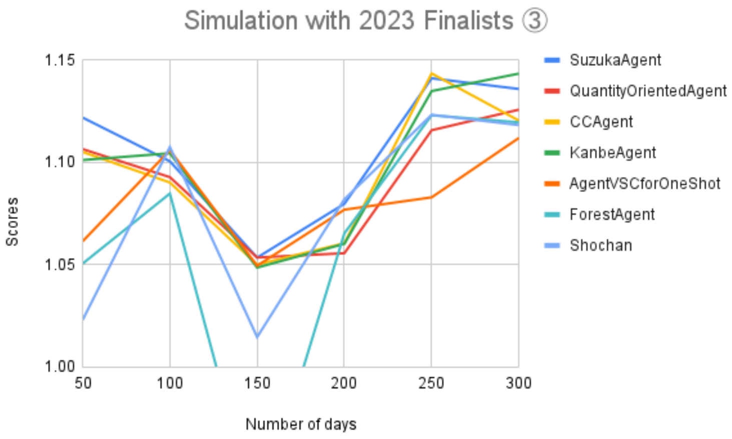
	mean	min	25%	median	75%	max
SuzukaAgent	1.09951	1.01979	1.05838	1.10348	1.12027	1.18142
QuantityOrientedAgent	1.0816	1.03805	1.05916	1.07396	1.09321	1.16145
CCAgent	1.08209	1.03002	1.05478	1.07931	1.10888	1.13744
KanbeAgent	1.08666	1.02526	1.08013	1.08762	1.09563	1.1527

Figure 2: the simulation with the 2023 Finalists in 100 days (These were used in the reinforcement learning)



	mean	min	25%	median	75%	max
SuzukaAgent	1.08631	1.03341	1.05705	1.0817	1.1149	1.14479
AgentVSCforOneShot	1.0949	1.05367	1.06035	1.08755	1.12877	1.15782
ForestAgent	1.06015	0.97987	1.01876	1.04751	1.1087	1.13758
Shochan	1.07055	1.01074	1.02372	1.07119	1.12319	1.1464

Figure 3: the simulation with the 2023 Finalists in 100 days (These weren’t used in the reinforcement learning)



	mean	min	25%	median	75%	max
SuzukaAgent	1.10550	1.05354	1.08502	1.11128	1.13245	1.14115
QuantityOrientedAgent	1.09171	1.05358	1.06498	1.09971	1.11347	1.12583
CCAgent	1.09501	1.05012	1.06799	1.0977	1.11665	1.14351
KanbeAgent	1.09883	1.04867	1.07051	1.10286	1.12731	1.14344
AgentVSCforOneShot	1.0815	1.04957	1.06534	1.07997	1.10025	1.11201
ForestAgent	1.06157	0.92584	1.0543	1.07514	1.11093	1.12309
Shochan	1.07817	1.01471	1.03781	1.09482	1.11562	1.12331

Figure 4: the simulation with the 2023 Finalists in various days

4 Conclusion

SuzukaAgent adjusts the degree of the priority for unit price flexibly in accordance with compatible with other factory agents. From the simulation results, it is evident that regardless of changes in factory agents or the number of days, high profits can be obtained. Future work involves achieving higher profits by adapting more flexibly to other factory agents.

Acknowledgment

This work was initiated in G-RIPS Sendai 2023 project[1] held at Tohoku University supported by NEC corporation. The author extends warm thanks to the project team members, Issac Brown, Kevin Zhou, and Soichiro Sato, and the mentors Dr. Masaki Ogawa of Tohoku University and Dr. Yasser Mohammad of NEC corporation, for their invaluable collaboration and support during the project.

References

[1] Issac Brown et.al. G-RIPS Sendai 2023 project report. `https://www.mccs.tohoku.ac.jp/g-rips/report/2023/pdf/nec_final_report.pdf`.

[2] Yasser Mohammad et.al. SCML 2023 League. `https://scml.cs.brown.edu/scml2023`.

[3] Yasser Mohammad et.al. scml2023 oneshot. `https://github.com/yasserfarouk/scml-agents/tree/master/scml_agents/scml2023/oneshot`.