

GreedyFactoryManager2:

An Autonomous Agent for ANAC 2020 SCM league

Yuta Hosokawa

Tokyo University of Agriculture and Technology
yuta@st.go.tuat.ac.jp

1 Introduction

GreedyFactoryManager2 is an agent submitted to the ANAC 2020 supply chain management league. In this report, we describe the strategy of GreedyFactoryManager2. The main idea of our agent is that GreedyFactoryManager2 tries to earn profits as much as possible with the care of not making any breach.

2 The Design of GreedyFactoryManager2

2.1 Negotiation Choices

GreedyFactoryManager2 requests negotiations with all suppliers and consumers in each step. Our agent chooses the issue range so that our agent will not make a huge loss, however, the ranges are relatively large so the negotiators can search the optimal agreement.

2.2 Utility Function

The negotiators employ linear additive utility functions. When the negotiator is a buyer, a lower unit price and a faster delivery time make the higher utility value. On the other hand, when the negotiator is a seller, a higher unit price and a later delivery time bring the higher utility value. The quantity does not affect the utility value, as long as the quantity does not exceed the number of manufacturing lines.

2.3 Concurrent Negotiation

GreedyFactoryManager2 does not apply any concurrent strategy. All the negotiators negotiate without communicating with each other so that the negotiators perform each negotiation efficiently and get the best agreements. We expect the negotiators to make a variety of agreements, and then in the signing phase, GreedyFactoryManager2 decides which subset of contracts to be signed. Because our signing strategy is the core of our risk management, we describe the signing phase in the next section.

2.4 Risk Management

We focused most of our attention on risk management in the design of our agent. As mentioned in the previous section, the negotiators make various agreements without communication. Greedy-

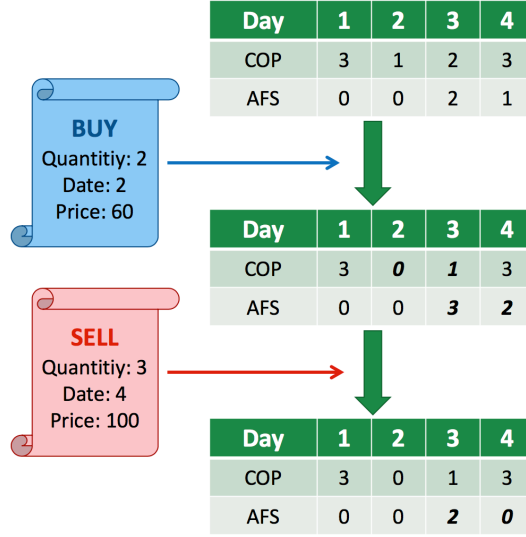


Figure 1: Example of the change in COP and AFS tables

FactoryManager2 takes care of not making a breach as much as possible when it selects contracts to sign.

GreedyFactoryManager2 uses two types of tables to avoid breaches:

- *Capacity of Production (COP)*: A table showing the number of available production lines to use on each day. The table is initialized by the number of production lines. When a buying contract is signed, the number on the delivery date will be subtracted by the quantity.
- *Availability for Sale (AFS)*: A table indicating the quantity of output products produced each day. The sum of the numbers before the selling date means the estimated stock quantity on that day. The table is initialized by zeros. The numbers will increase when a buying contract is signed, as well as they will decrease for a selling contract.

Figure 1 shows an example of the change in the two tables. GreedyFactoryManager2 tests combinations of contracts so that to avoid breaches. For each day, GreedyFactoryManager2 selects an optimal combination of contracts and sign them.

2.5 Collusive Strategy

GreedyFactoryManager2 does not employ any special collusion strategy. We believe that even if multiple instances of the agent in the chain collude with each other, they have to share the total profit with each other. On the other hand, if each instance profits independently, it may maximize its profit by earning from the other agents. For these reasons, GreedyFactoryManager2 does not make any contract with their team member.

3 Evaluation

We ran several simulations between our agent and built-in default strategies. From these results, we confirmed that GreedyFactoryManager2 does not make many breaches because of the risk management strategy. However, some breaches of contracts occurred when some of the contracts are not signed by the opponent. Further expansion to handle such risks is required.