Lobster: An agent submitted to the ANAC 2022 SCM league

Ito Nobuhiro[1] Takanobu Otsuka[2]

[1,2]Nagoya Institute of Technology, Aichi, Japan
[1]ito.nobuhiro@otsukalab.nitech.ac.jp
[2]otsuka.takanobu@nitech.ac.jp

June 16, 2022

Abstract

The main strategy of Lobster is to reduce penalties of inventory shortages and overstocks, and to stably obtain small profits, instead of aiming for large profits. At lobster, prices are not dynamic. Based on the inventory forecast, we will conclude a contract so that no penalty will be incurred. We propose an agent that can consistently get positive score.

1 Introduction

In the actual supply chain, companies have fixed costs such as labor and maintenance. If we do nothing, we will go bankrupt one day. Therefore, we must continue to earn profits by selling goods at a higher price than cost.

However, in the SCML world, there are no fixed costs. So if we do nothing, we will have zero loss and zero profit. In other words, the final score is zero. But even then,it is difficult to get a positive score on the Standard Track. Even the third place agent in 2021 standard track (Artisan Kangaroo, score: -016) did not get a positive score.

We aim to reduce penalties of inventory shortages and overstocks, and to stably obtain small profits, instead of aiming for large profits.

2 Risk Management

There are three major risks to Standard Track. The first is surplus of input products. All excess inventory is sold to the spot market at a discount. By setting an upper limit on the amount of inventory so that it does not arrive too much Input Product. Furthermore, by setting the final date of arrival, the final stock quantity is reduced to zero.

The second is a shortage of Output products. If we run out of stock by the shipping date, we must buy products from the spot market at a high price. By accepting output contracts according to the expected inventory amount, it is possible to prevent a shortage of Output products.

Finally, the bankruptcy of another agent. In order to reduce the impact of bankruptcy of the other agent, we limit the number of days that we can negotiate and do not take contracts after last_day. In addition, by holding safety stock, the impact of inventory shortages is mitigated.

3 The Design of Lobster

Our agent was created based on the std_monolithic template. Lobster consists of four components:

- Lobster.py
  A class that defines the base behavior of our agent. It inherits SCML2020Agent.
- Myinfo.py
  A class that stores information used by our negotiators.

- ControllerA.py
  Negotiator on the buying side. It inherits SAOSyncController.

- ContorollerB.py
  Negotiator on the selling side. It inherits SAOSyncController.

3.1 Production Strategy

When the Input Product arrives, our agent schedules the production on the fastest vacant line of the factory. In order to conclude an input contract so as not to exceed the capacity of the factory line, all products will be stored in the Output Inventory. We took this strategy to simplify our trading strategy.

3.2 Trading&Negotiation Strategy

Our agent negotiates by considering the price and the quantity/time separately. When both the price and the quantity/time are satisfied, our negotiator accept the offer. Otherwise, we reject it and respond with counteroffer.

3.2.1 Price

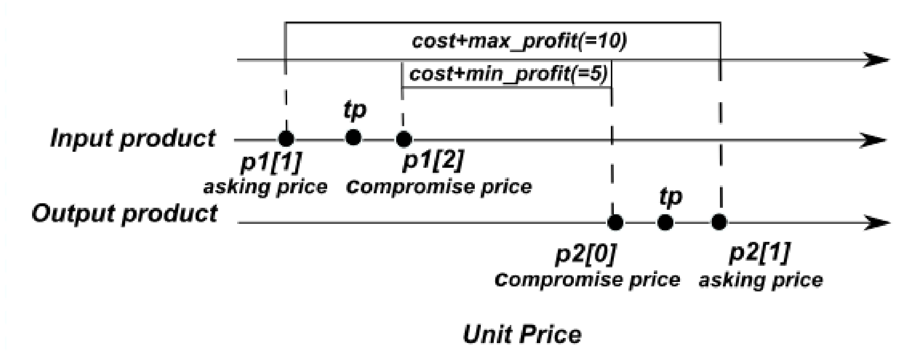

Figure 1: management of price

Figure 1 shows a diagram of pricing. We set the asking price and compromise price based on the Trading price. The asking price is set to a net profit of \$ 10, and the compromise price is set to a net profit of \$ 5.

If we make too much net profit when we trade at trading price, increase the price of Input product and decrease the price of Output product. On the contrary,if we don't have enough net profit when we trade at trading price,decrease the price of the Input product and increase the price of the Output product.

Our offer is determined based on the asking price except the last two trading negotiation steps. At the last two trading negotiation steps, our offer is determined based on the compromise price. If the price of other agents' offer is better than the price of our offer, the price is OK.

3.2.2 Time/Quantity

As explained in Production Strategy(Chapter 3.1), all Input Product are converted to Output Product as soon as possible. Therefore,we manage only the Output Inventory. We calculate the expected quantity by reflecting all transaction contracts. Figure 2 shows an overview of inventory management.

First, we set the transaction period. We make a transaction contract between first_day and last_day. First_day is current_step, and last_day is n (= 40) days after current_step. However, if the other agent is a system, there is no risk of it going bankrupt, so n is infinite.

After the "n_steps – (number of factory lines on the right side of us)" day, it is likely that other agents will not accept our shipment because the product will not be able to buy BUYER in time. Therefore, we do not trade after last_trade_day. However, if the other agent does not move rationally and accepts our shipment, the shipment transaction will be carried out.

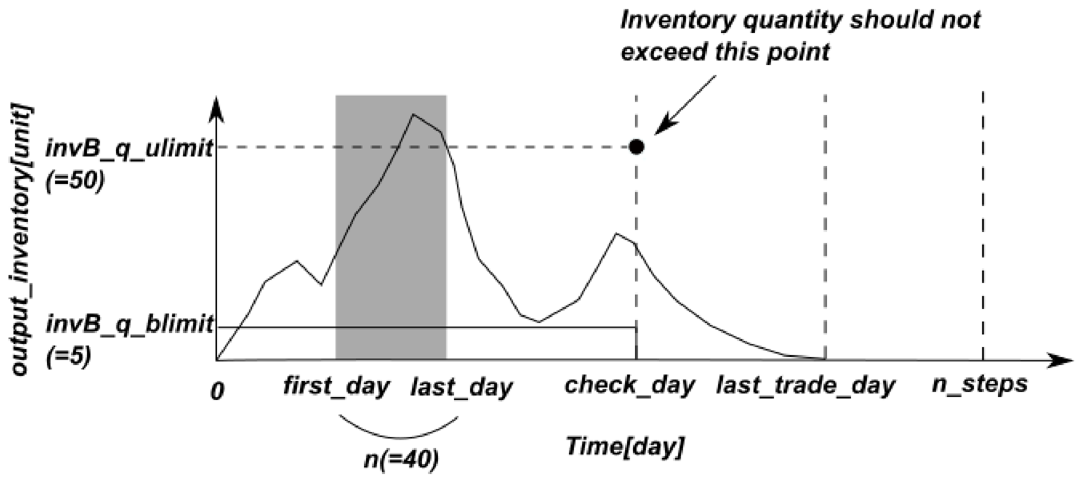

Figure 2: management of output inventory

Next, the management of the quantity of Input Product and the arrival date will be described. The upper limit of the number of stocks (invB_q_ulimit=50) is set in check_day (10 days before last_trade_day). However, the inventory quantity may exceed the invB_q_ulimit before check_day if the inventory is expected to fall below invB_q_ulimit by check_day.

We make input contracts so that the inventory amount at check_day approaches invB_q_ulimit. We don't accept input offer after check_day. In addition, we also manage the operating status of the factory's production line. If the production schedule cannot be made within the last_day, the contract will be declined.

Next, the management of the quantity of shipments and the shipment date will be described. We set the lower limit of inventory (invB_q_blimit=5). This is a mechanism to prevent inventory shortage in case the input agent goes bankrupt. There is a possibility of running out of 40[day] * 10[line] = 400 units. So this value should be higher,but since there are few factories that go bankrupt, it has been set small. We will ship as much as possible so as not to fall below the lower inventory limit. After check_day, the lower limit of inventory is set to 0 and all products are sold.

If we are on the final line in the supply chain,there is a high possibility that inventory will be left over because BUYER won't make agreement once we decline. Therefore, the lower limit of inventory is set to -30 pieces, the upper limit of inventory is set to 0 pieces. The number of shipments is decided first, and then the contract for arrival is made in time for the shipment.

3.3 Negotiation Choices

We request negotiations for all parties with following issues .

- Time: (first_day , last_day)
- Price(BUY) : (min(T P*0.5, asking price) , max(TP*1.5, compromise price))
- Price(SELL) : (min(T P*0.5, compromise price) , max(TP*1.5, asking price))
- Quantity : (1 , n_lines)

We accept negotiation requests from other agents that satisfy all of the following conditions.

- Time: The day between first_day and last_day should be included.
- Price : The compromise price and asking price are included.
- Quantity : Numbers from 1 to n_lines must be included.

4 Evaluation

We ran 5 standard track tournaments using run() method included in the template to evaluate Lobster's performance. We compared our strategy with DecentralizingAgent and BuyCheapSellExpensiveAgent. The world parameters are as follows:

- competiton:Std
- n_steps=80
- n_configs=5
- competitors = [Lobster,DecentralizingAgent,BuyCheapSellExpensiveAgent,]

The results of the tournaments are shown in Table 1.

Table 1: Score of tournaments

tournament	Lobster	DecentralizingAgent	BuyCheapSellExpensiveAgent
1st time	0.026	-0.095	-1.276
2nd time	0.041	-0.209	-1.861
3rd time	0.025	-0.541	-1.149
4th time	0.032	-0.110	-0.619
5th time	0.052	-0.136	-1.328
Average	0.035	-0.218	-1.247

With these two default strategies, profits are negative, but Lobster is consistently positive. Even when the score is the lowest, it has a positive profit, so it can be said that the score is better than the agent who does nothing.

Furthermore,checking the score of 95 bodies for each agent in each world, only 10 times were negative score. BuyCheapSellExpensiveAgent gets negative score 95 times and DecentralizingAgent get negative score 72 times. We couldn't rid of the negative score completely, but we were able to get a stable positive score compared to the default agent.

Conclusions

In this report, we explained Lobster's strategy. We aim to reduce penalties of inventory shortages and overstocks, and to stably obtain small profits, instead of aiming for large profits. As mentioned in Section 3.2.2, the current parameters are not sufficient to prevent impact of bankruptcy. This time, the price is set simply, but it can be set while considering the balance between supply and demand in the future. It is also possible to dynamically set the inventory upper limit according to the demand. Although there is still room for improvement, it has become an agent who can get a positive score.