Exercise 5: An Auctioning Agent for the Pickup and Delivery Problem

Group №24: Lucas Burget, Lucas Waelti

November 25, 2018

1 Bidding strategy

1.1 Speculation on the cost of the new task

In order to make a realistic bid, the first step is to approximate the marginal cost of the new task. To do so, we used the Stochastic Local Search planner from the previous exercise to find the cost of delivering the previously won tasks and the newly auctioned task. We then compare the cost of including the task in the plan compared to the cost of the plan without it. This gives us the minimal bid acceptable that avoids any loss by taking the cost difference between the two plans.

The SLS algorithm is initialised with the best path previously found. The newly auctioned task is added to the plan of the largest vehicle by default and then the SLS algorithm tries to place the new task in an efficient way. This is particularly useful if the computation times are rather short, as it avoids to start the search from a very inefficient solution. The algorithm may therefore find a good solution in less time than a random initialisation would have required.

1.2 Speculation on current path

We also measure the *Potential* (equation 4) of our plan with the auctioned task. To do so, we calculate two things: the probability P_{onPath} (equation 1) of having a future task on the plan and the expected cost C_{nearby} (equations 2 and 3) of nearby tasks. Both are normalised to have a value between 0 and 1.

P_{onPath} is computed by averaging the probability of finding a task in city c_1, $p_t(c_1)$, times the probability of having a task from one city of the path c_1 to another c_2, $p_t(c_1, c_2)$, and which can be carried without overload. It means that this hypothetical future task will not request any change of plan and thus any additional costs. A high value would imply a high probability of having a task on the path. Since the probabilities tend to be less than 0.1, a factor 10 is applied if possible to make it comparable with the nearby cost:

$$P_{onPath} = \left(\sum_{vehicles} \frac{\sum_{c_1=0}^{N_p} \sum_{c_2=c_1}^{N_p} p_t(c_1) \cdot p_t(c_1, c_2)}{n} \right) \cdot \frac{1}{N_{vehicles}} \tag{1}$$

with N_p the number of cities on the path (one city can be included multiple times) and n the number of tasks evaluated which don't produce any overload (they are otherwise discarded from the sum).

C_{nearby} is computed by averaging the relative cost increase of picking up a task from a neighbouring city of the current path to any city of the path. The cost is weighted by the probability of having a task in that city. Here, a low value indicates there should be close tasks available nearby the current path.

$$C_{nearby} = \left(\sum_{v \in vehicles} \frac{\sum_{c_1=0}^{N_p} \sum_{c_2=0}^{N_{c_1}} \sum_{c_3=c_2}^{N_p} p_t(c_2) \cdot p_t(c_2, c_3) \cdot cost(v) \cdot addedDistance(c_1, c_2)}{2 \cdot maximalDistance \cdot cost(v) \cdot \sum_{c_1} \sum_{c_2} \sum_{c_3} p_t(c_2) \cdot p_t(c_2, c_3)} \right) \cdot \frac{1}{N_{vehicles}} \tag{2}$$

with N_{c_1} the number of neighbours of city c_1 that can rejoin the path at c_1 or c_1+1, $addedDistance()$ the extra distance needed to go through the neighbouring city c_2 and $maximalDistance$ the longest intercity distance found in the topology. Any configuration that produces an overload is as well discarded from the calculation.

We make it so that a value close to 1 is good:

$$C'_{nearby} = 1 - C_{nearby} \tag{3}$$

We finally obtain the final *potential* of a task:

$$Potential = (P_{onPath} + C'_{nearby})/2 \quad \in [0, 1] \tag{4}$$

1.3 Speculation on other agents' behaviour

We also take into account the opponents' behaviour. We want to maximise our profit so we should try to bid as high as possible. To do so, we want to iteratively adjust the margin our agent can apply on its bids. We compute the weighted mean of the opponents' bids and consider the lowest mean. The weights are $\frac{1}{\sqrt{n}}$ with n the number of auctions that took place since the considered bid was emitted. We can therefore focus our calculation on the most recent results. Then, we adjust our margin according to this mean:

$$margin_{t+1} = margin_t \cdot \left(1 + \nu \cdot (1 - \frac{\overline{b}_a}{\overline{b}_o}) \cdot (1 - \frac{1}{N_a})\right)$$

with $\overline{b}_a$ and $\overline{b}_o$ respectively our agent's average bid and the lowest opponent's average bid, ν a factor (typically between 0.5 and 0.8, depending if we are bidding respectively to high or to low) that slows down the fluctuations of the margin and $N_a \geq 1$ the number of tasks auctioned until now. N_a allows to disregard means with a small number of tasks since they are not really reliable.

1.4 Final bid

It is a good strategy to try to win a few tasks before considering making profits. Indeed, with a bunch of tasks, the marginal cost of a new task would be decreased and so we can earn more on it. So, as long as we have less than 5 tasks, we will only bid the marginal cost without margin. Furthermore, always with the idea that having a few tasks is mandatory, if the opponents win tasks and we still have not won any, we decrease our margin by 10% increments, each time the opponent wins a task and we still have none, with a capped loss of 50%. This is a risky but very aggressive strategy that might give us a chance to come back and gain a few tasks, and at least limit the opponent's profit.

The fact that we really want to avoid having no task at all is that it becomes harder and harder to win an auction if the opponent is accumulating tasks. The opponent ends up winning all auctions. In any other case, the bid is computed as follows:

$$bid = marginalCost \times (1 + margin \times (1 - Potential/2)) \quad \in [0.01, 1]$$

2 Results

2.1 Experiment 1: Comparisons with dummy agents

2.1.1 Setting

We tested our agent in two different configurations. Auction is based on the England topology, with all vehicles having the same capacity and the weight of the tasks are constant. Auction 2 is based on the Netherlands topology, the capacity of the vehicles are not symmetric and the weight of the tasks can vary.

The agents we used for testing are:

- **Final**, our agent.
- **DummySafe** just bids the marginal cost with a very small margin.
- **DummyRandom** is the one provided at the beginning.
- **V1** is an agent that has the same settings as Final except it has not implemented the functions encouraging to take on early tasks. It will not make any loss or zero profit bids.

2.1.2 Observations

Our Final agent performs approximately equally well in low constrained environment. But, as soon as it gets more difficult, our agent performs better.

A very important observation is about the effect of the topology. Typically the England topology has isolated cities, which will yield high marginal costs for the first tasks of vehicles starting in those cities. The agent with those vehicles has barely no chance of competing against an agent with better placed vehicles.

Furthermore, an agent that starts accumulating tasks while other agents have still none or just a few will tend to be able to make much lower bids and therefore keep winning the auctions. This is why our aggressive strategy can help cope with this effect if our agent finds itself in a difficult situation. The issue here is that it might end up with losses if it could not then gain enough tasks with sufficient margin to compensate the initial aggressive bid.

The SLS algorithm produces rather stochastic results too, therefore the marginal cost might fluctuate and not be perfectly consistent between two cost computations, which adds to the uncertainty of making adequate bids.

2.2 Experiment 2: Effect of the topology

2.2.1 Setting

We use here the Switzerland topology. We consider 3 companies with 3 vehicles each of respective capacity 45, 100 and 200. Twenty tasks are auctioned with weights uniformly ranging between 40 and 60 kg. The first company starts in Geneva, Lausanne and Sion (south-west), the second in Fribourg, Bern (center) and Neuchatel and the third in Basel, Aarau and Zurich (north-east).

2.2.2 Observations

By letting compete our agent against other versions of itself, we can clearly show the effect of the topology.

First company	0	0	202	0	$mean = 50.5$
Second company	2437	1568	296	1169	$mean = 1367.5$
Third company	37	1682	0	-400	$mean = 329.75$

Table 1: Profit of each companies using our agent for different seeds

We clearly see that the company whose vehicles are located in the center of the topology has an advantage over its opponents.