An Auctioning Agent for the Pickup & Delivery Problem

Group 16
Yoan Blanc <yoan.blanc@epfl.ch>, 213552
Tiziano Signo <tiziano.signo@epfl.ch>, 226511

November 17, 2015

You will define an auction strategy for your agent and compete against other agents. When designing your strategy, please keep in mind the following facts:

- *The minimum price you are willing to accept for delivering the task T_1 is equal to the marginal cost of delivering T_1. In other words, you have to:*
  1. *find the cost of the plan for delivering the tasks you've already won,*
  2. *estimate the cost of the plan for delivering the same tasks plus T_1,*
  3. *and compute the marginal cost of delivering T_1 as the difference between the two.*
- *Any price paid by the auction house which is superior to the marginal cost of a task will result in a profit for you to deliver that task. Therefore, you should usually bid above your own marginal cost. You may occasionally bid below your marginal cost if you believe this will reduce the cost of future tasks, but be careful not to end up with a deficit!*
- *You do not know what other tasks will be auctioned by the auction house. In fact, you do not even know* ***how many*** *more tasks will be auctioned after the current one. When you compute your current bid, you might want to speculate on the effect of taking the present task on the future auctions.*
- *You do know the probability distribution of tasks. The tasks that will be auctioned will follow this distribution, so you might want to use these probabilities in order to speculate about future tasks that will be auctioned.*
- *The winner and all bids are published immediately after each auction. This gives you the possibility to refine your strategy depending on your opponent(s).*

Implementation

We observed that the tasks are in general good to get because of their high *complementarity*. The general strategy was to grab as many tasks as possible by trying to predict the total cost of having to deliver more than 15 tasks. I won't work for few tasks, sadly. Then, we explored a learning approach by observing the others' bid, which is surprisingly efficient with regards to its simplicity.

The planning has been done using the centralized agent work with very few modifications to deal with a time constraint as well as optimizing on the newly added tasks directly. before continuing with the *simulated annealing* hill climbing algorithm.

Agent

Remark, all our agents are named after Duck Tales characters in English, French or Italian. Funny codenames to hide their strategies. And they all add a tax of 1 to their bid, for fun (and profit). Because you should still be able to win by asking 1 more than your calculation.

Dumb agents

The not so dumb agents, there is no needs to explain the very dumb ones.

Picsou

After doing a pure *greedy* one bidding 1 about its marginal cost, this one aims to gain a bit more money by doing the focusing on evicting the other to take any tasks. It will bid -1 below the cost from pickup to delivery using it's most lightweight vehicle.

It'll win versus very basic agents, relying on the marginal cost only because it forces you to go below zero if you want to take any task from it.

It relies on overlapping tasks to make profit.

Paperino

This one has a pure price prediction strategy. It computes for each segment of the map the expected number of times it will appear for a reasonable amount of rounds (7 or 8 showed to work well). Then it pre-computes based on the vehicles what cost of one usage based on the total amount it expects to obtain.

It's a pretty agressive strategy that can do well on some setup. It also aims to grab everything and will fail if the other players takes only the good ones and leaves it the costly paths. Because it has no knowledge of the other player, it won't be able to react to its situation either good or bad.

Huey

Based on *Paperino*, it weights some segments differently. If a segment leads to a city with $[1, 2]$ neighbors, then it sees it has more distant. It's the assumption that whenever you go there, you'll have to come back to continue your deliveries. So this task's price has to take into account the burden its causing to the vehicle.

This one is quite good at predicting price but it does not consider the other player because its goal is to take the good tasks, and disregards the less good ones.

Louie

This the first historical agent that bids by learning. It takes the bid of the other agents and computes an average cost per kilometer for each segment.

It's a surprisingly efficient one against the previous agents which have a cost per segment strategy.

Candidate

Dewey

This is a mix of *Huey* and *Dewey*, which is using two plannings to guess the other's marginal cost and overall rewards. It's taking the same vehicles as itself assuming that it doesn't matter much after a while.

It goes with *Huey* way of working for the first couple of rounds and then depending on its state it acts differently. If it suspects the other player to win, then it will switch to *Louie*'s strategy to mimic its opponent. In case, it thinks he wins, then he will increase its bids with a ratio of the difference between its estimation and the other's.

Efficiency

It's very hard to have an always good agent when the tasks can be very complementary with each other. And learning from the environment is limited by the assumption that the other agent is doing good. Try to combine too many variables from the environment makes the bid computation a guestimation rather then something arguable.

Conclusion

We created a third template to get another environment but when running some tournaments, none of our agents won them all. Knowing how to be properly aggressive with the starting prices, how to more accurately predict the final price or when to start acting with respect to your opponents actions are very hard.

We cannot be confident that any of our agents can bet all the other's one because they are no clear winners in our experiments either.

One major weakness of our agents is that they are going negative and can stay there, loosing to a no tasks agent. This agresive behaviour seems weak in some situations. Not being able to know what's coming next makes the bidding task hard.