Excercise 4
Implementing a centralized agent

Group №: 90 Kyle Gerard, Yann Bolliger

November 5, 2018

1 Solution Representation

1.1 Variables

Instead of representing a vehicle's journey as a sequence of tasks, we chose to represent it as a sequence of *pickup* and *delivery* actions. Each task $t \in \mathcal{T}$ has one action of both types.

$$\mathcal{P} = \{pickup(t) : t \in \mathcal{T}\},\ \mathcal{D} = \{delivery(t) : t \in \mathcal{T}\},\ \mathcal{A} = \mathcal{P} \cup \mathcal{D}$$

This accounts for the fact that a vehicle can carry multiple tasks at a time if there are two pickups in a row. The following variables define the first pickup of each vehicle (where $\mathcal{V}$ is the set vehicles).

$$\forall v \in \mathcal{V} :\ firstPickup(v) \in \mathcal{P} \cup \{\texttt{null}\}$$

If the variable is `null` this means the vehicle does not accomplish any actions. All subsequent actions of a vehicle are defined by the next set of variables:

$$\forall a \in \mathcal{A} :\ nextAction(a) \in \mathcal{A} \cup \{\texttt{null}\}$$

where again the `null` signifies that a vehicle has no further actions to perform. We will define two other sets of variables which will clarify the former:

$$\forall a \in \mathcal{A} : vehicle(a) \in \mathcal{V};\ \ \forall a \in \mathcal{A} : time(a) \in \mathbb{N}$$

The *vehicle* variables define which vehicle carries out a certain action. This can be derived from the $firstPickup$ action at the start of each action chain defined by *nextAction*. (For example if $firstPickup(v) = a, nextAction(a) = b$ then $vehicle(a) = vehicle(b) = v$.)

The second variable can also be derived from the action chains. It simply gives the rank of each action in the chain (for example if $firstPickup(v) = a, nextAction(a) = b$ then $time(a) = 1$, $time(b) = 2$).

1.2 Constraints

As explained before, the action chain for each vehicle define the *time* and the *vehicle* for each action:

$$firstPickup(v) = a \Rightarrow time(a) = 1;\ \ nextAction(b) = c \Rightarrow time(c) = time(b) + 1$$

$$firstPickup(v) = a \Rightarrow vehicle(a) = v;\ \ nextAction(b) = c \Rightarrow vehicle(c) = vehicle(b)$$

Additionally, the same vehicle must *pickup* and *deliver* a task. It has to pickup the task before it delivers it, of course, and **each task must be picked up and delivered**:

$$\forall a \in \mathcal{A} : nextAction(a) \neq a,\ nextAction(a) = \texttt{null} \Rightarrow a \in \mathcal{D}$$

$$\forall t \in \mathcal{T} : vehicle(pickup(t)) = vehicle(delivery(t)) \in \mathcal{V}$$

$$\forall t \in \mathcal{T} : time(pickup(t)) < time(delivery(t))$$

$$\forall t \in \mathcal{T}\ \exists \{pickup(t), delivery(t)\} \subset \mathcal{A} \text{ and } \forall a \in \mathcal{A}\ \exists v \in \mathcal{V} : vehicle(a) = v$$

Last but not least, at each time τ a vehicle v can never carry more weight than its capacity.

$$carriedTasks(\tau, v) = \{t \in \mathcal{T} : vehicle(pickup(t)) = v \wedge time(pickup(t)) < \tau \wedge time(delivery(t)) > \tau\};$$

$$\forall \tau \in \mathbb{N} : \sum_{t \in carriedTasks(\tau,v)} weight(t) \leq capacity(v)$$

1.3 Objective function

The goal of the company/agent is to maximise the reward. Because all tasks have to be delivered, all rewards will be earned and the overall reward sum is therefore constant. Thus the objective function we want to minimise is the cost of the overall assignment $\mathcal{S}$.

2 Stochastic optimization

2.1 Initial solution

2.2 Generating neighbours

2.3 Stochastic optimization algorithm

3 Results

3.1 Experiment 1: Model parameters

3.1.1 Setting

3.1.2 Observations

3.2 Experiment 2: Different configurations

3.2.1 Setting

3.2.2 Observations