Problem definition

Riccardo Orizio

13 October 2015

1 Problem definition

Our problem concerns the need of a company to accomodate client requests to specific time slots where the company will have more profit, given the actual requests status and the probable future requests. This problem treats a small part of a bigger problem which is the problem of the attended home delivery services for furnitures and appliances.

First of all, the company needs to choose if the new incoming request will be accepted or not: remember that every refused request is lost and cannot be recovered in the future, so every declined request will cost to the company because there will be no profit from it.

apparently we can accept EVERY new customer because the company could rent as many vehicle as it wants. Every new vehicle has a fixed rent cost.

If a request is accepted, then it has to fit in the already known request calendar, knowing that every new request has a limited set of possible time windows in which it could be served (usually the time windows are given due to geographical request studies) and that every request must respect its time window.

The goal of this problem is to bias the customers to make them choose what is more suitable for the company, so it could reach the maximum profit or it could serve the highest number of requests that it could do or get the minimum costs through clusterization of customers. The company will try to influence its customers through small price incentives[1] over the possible time windows listed to them, so there is a high probability that customers choose the time window that the company wants them to pick. The problem is to place these incentives, that is to choose in which time window they have to be placed and how big every one of them has to be.

Every customer has to pay a fixed fee for the delivery service. JF suggested something like 50$, independent to the distance traveled.

The problem is dynamic due to the fact that we keep receiving new requests even after our planning is started; it's stochastic because we try to predict future requests from the history and because we don't know when the request will come

[1] There is no need to place big incentives, small are good enough, knowledge given by home grocery delivery providers, Campbell and Savelsbergh 2006.

neither how big the demand of each one will be. We use this information with the intention to keep the best incoming requests and discarding the worsts.

2 Model

ADDED

The pappappero routing with TWs and stocaz customers is defined as follows. We have a set of known customers C, each one with a service time s_i and a demand to be served q_i, spread into a limited geographical area, usually divided in sub-zones (e.g. ZIP-code division, districts), stored in set Z. The company could count on almost an infinite number of vehicles because every vehicle could be rented at a fixed cost f, each of them has a capacity q and from now on they will be referenced as $k \in K$, set of all vehicles available to the company. The set of time windows is known as T and for every one of them there could be an incentive I_t which, paired with every customer i, has a probability p_i^t to be chosen by the customer i as his delivery time window. We will refer to new upcoming customers with $c \in U$. To reach our goal we need to take some decisions based on at least one possible future scenario ω, taken from the set of all scenarios Ω. A single scenario ω is nothing more then the union of the known customers C and a subset of the future customers U: $\omega = (U \cup C)$.

3 Objective function

The objective function could vary depending on the priorities of the company, a couple examples are listed here:

- $MAX\ profit = MAX\ \sum revenue - \sum delivery\_costs - \sum incentives$
- $MAX\ orders\_delivered$
- $MIN\ costs$ ADDED

4 Variables

Rewritten in the model section, after JK suggestion

C known customers spread into a limited geographical area, usually divided in sub-zones (e.g. ZIP-code division, districts). Every $i \in C$ customer has a service time s_i and a demand q_i

K company vehicles, each one having Q^s storage and Q^f fuel capacity[2]

vehicles that the company could rent, at a fixed cost F, each one having capacity Q

[2]Every vehicle could have different capacities.

T delivery time window[3] options, each being L[4] wide (time windows are chosen by the company itself)

I_t possible incentives, one per time window

p_i^t the probability that the customer i chooses the time window t to get his items delivered

U set of incoming customers, one will be addressed as c

Z set of zones

Ω set of scenarios, $\omega \in \Omega = (U \cup C)$

5 Constraints

- Don't exceed vehicles capacity q , both storage and fuel (or delivery costs)
- Make sure that every delivery will stay in the defined time-window, remembering also that every delivery has a service time[5]
- $\forall t \in T : I_t \geq 0$

6 Assumptions

- Real-time delivery are not available, we want at least 1-day notice to create the best plan that we can
- Delivered products are not subject to any particular restriction: they are unbreakable and without expiry date (e.g. no food)
- We keep a low number of C customers because the problem is hard to solve even with small sets of input
- $\forall i \in V : Q_i^s = a, Q_i^f = b, a\, b \in \mathbb{N}$

7 Future extensions

- Everything that could vary from the assumptions
- Different time-windows: different L, overlap between windows

[3] Time windows could be overlapped with each other or not.
[4] The larger the better: we have more flexibility. (Campbell and Savelsbergh, 2006).
[5] Time between vehicle arrival to the customer and its departure.

– Predict customers: keep track of every request done by every customer so we could predict them even better, knowing their particular time window preferences and using those for placing incentives only when it's really necessary

– Negative incentives: make the delivery cost more high for those customers that demand a specific delivery that is unprofitable for the company, without a negative incentive