Optimal Sequence in Which to Service Orders

Ken Cor* Gaurav Sood†

July 15, 2021

Given the cost of a set of tickets on any given day is defined by matrix c_ij where i denotes ticket and j denotes day. On which day should each ticket be purchased in order to minimize the total cost to purchase the tickets. The maximum number of hours that can be purchased on any given day is given by a vector v_j. Each ticket can only be purchased once and all tickets must be purchased. The cost of buying a ticket is the product of time it takes to buy a ticket in hours (h) and cost per hour of labor (e).

Decision Variables:

$$\mathrm{x}_{ij} = \begin{cases} 1 & \text{if ticket t is sold on day j} \\ 0 & \text{otherwise} \end{cases}$$

Minimize the objective Function:

$$\Sigma_j \Sigma_i (c_{ij} + h * e) x_{ij}$$

subject to the following constraints:

$$\Sigma_j x_{ij} == 1$$
$$\Sigma_i h * x_{ij} \leq v_j$$

What is the optimal order in which to service orders assuming a fixed budget?

Let's assume that we have to service orders $o_1, ..., ...o_n$, with the n orders iterated by i. Let's also assume that for each service order, we know how the costs change over time. If we receive service order o_i at time t, we expect the cost to be c_{it} at t, c_{it+1} at $t+1$, etc. Each service order also has an expiration time, j, after which the order cannot be serviced. The cost at expiration time, j, is the cost of failure and denoted by c_{ij}. Expectedly, $j \geq t$.

*Ken can be reached at mcor@ualberta.ca
†Gaurav can be reached at: gsood07@gmail.com

For simplicity, let's also assume that time is discrete and portioned in days. It means that the cost of servicing an item on the same day—c_{it}—don't change depending on when the item was serviced during the day.

New service items arrive each day. And we re-prioritize the queue every day.

Let's denote the time it takes to finish a work item by w_i and our budget (total person hours) for each day, W_t. For the moment, let's assume w_i to be the same for all i—it takes the same amount of time to clear an item.

We work till $\Sigma_{it} w_{it} = W_t$ or till the queue for the day runs out.

The first question is if we need to punt any items to tomorrow. If $\Sigma_{it} w_{it} \leq W_t$, there is nothing more to do. If not, we need to make some choices. The optimal sequence of servicing orders is determined by expected losses—first service the order where the expected loss is the greatest. This leaves us with the question of how to estimate expected loss at time t. We need to answer two questions: 1. if we don't service the order today, what are our chances of getting to it the next day, the day after, etc., till j, or p_{it}, and 2. what is the cost we will have to bear if we postpone servicing the order to $t+1$ to j, or c_{it}. To get the total cost of deferring, we just need to first estimate what we expect to pay if we deferred: multiply p_{it} with c_{ij} over all t from $t+1$ to j. And then subtract what we expect to pay if we deferred from what we would pay at time t. So framed, the expected loss for order i at time t = $c_{it} - \Sigma_{t+1}^{j} p_{it} * c_{it}$.

However, determining p_{it} is not straightforward. New items are added to the queue every day. On the flip side, we also get to re-prioritize every day. The question then is if we will get to the item o_i at $t+1$? (It means p_{it} is 0 or 1.) For that, we need to forecast the length of the queue tomorrow. We can do this with a model. Let's assume, for now, that the model is perfect and we can forecast with perfect accuracy.

If tomorrow's queue length including items from today is less than W_{t+1}, then the cost of deferral is simply c_{it+1}. But if the queue will take more than W_{t+1}, then we need to calculate expected losses for all the items tomorrow to figure out which make the cut. To do that, we need to again start afresh. Hence, this becomes an infinite process.

To make it tractable, we need to make some assumptions. In some cases, one reasonable assumption is to assume that c_{ij} is very high. This effectively would mean that the items are always cleared by $j-1$. This kind of queue clearing in cases where we don't have enough days where we can pick the slack can eventually become a system where the only items that are cleared are ones that are about to expire.

Another reasonable assumption in some cases would be to assume that the cost of deferring for a day give us the right rank order of costs. Then the optimal thing to do is to clear items based on one day out costs.

A more rational approach to queuing would be to budget in a profitable way. Forecast the size of the queue and the items in the queue. Forecast the cost of waiting one day for all items. Sequence the costs in a way that we clear the most costly. And then buy as many person-hours as make sense to allow for clearing the remaining items the same day.