Service Delivery Planning AKA Timeslot Allocation

This note records basic facts about delivery timeslots, and how they can be manipulated to encourage customers to choose clustered delivery times. This document should evolve as my understanding of the problem and desired solutions to the problem increases.

Summary and Recommendations

Smart timeslot allocation depends on

- a marginal value calculator - estimate marginal opportunity costs arising from insertion of an additional order to an existing route.
- delivery pricing that targets each customer market segment,
- demand and choice models for each delivery product for each timeslot,
- constraint solver to hold out delivery capacity to preserve delivery capacity for high value product demand, and for popular timeslots,

1. Barebones timeslot allocation - first come first serve. *This is a good enough solution for any allocator that does not attempt to preserve timeslots for future demand.*
2. Develop approximate marginal opportunity cost model for route selection.
3. Develop demand and customer choice models for delivery products.
   a. Infrastructure prep: separate data analytic infrastructure from production realtime data,
   b. Analysis of client purchase process data from reference client => model calibration,
   c. Implement calibration service.
4. Implement yield management service
   a. hold timeslots back to maximise high value product takeup *or* preserve high-demand timeslots *or* both.
   b. does not require computing incremental marginal costs for new deliveries.
5. Dynamic pricing for delivery products, overflow management, etc.

<Reference Client>

Context for Service Delivery Planning

A delivery company receives orders from customers for the delivery of goods and services at customer locations.

The company has a collection of *delivery products*, priced delivery policies that a customer can choose from. Delivery products are defined in terms of

- *days-to-delivery*, the number of days between the order and delivery,
- *timeslot, a* working day interval in which the company undertakes to provide its service to the

customer,
- *flexibility*, the ease with which a customer can change his/her mind.

A delivery company wants
- to extract *maximum value* from its product offering, allowing it to make money.
- to maintain *efficient daily delivery* from its routes, allowing it to save money.
- to *predict load* on its delivery network over its planning horizon, allowing it to match resources to delivery requirements.

A customer of the delivery company prefers, for a given price,
- to receive their delivery sooner rather than later,
- to have more discretion over the time of delivery over less,
- to have more flexibility to change their mind, over less.

Delivery products are priced in a way that reflect customer preferences
- delivery sooner costs more than delivery later
- timeslots with fine granularity cost more than those with coarse granularity
- changeable delivery times cost more than fixed delivery times.

Requirements

Infrastructure
Requirements for data studies and model calibrations are significant and follow a different pattern to production requirement for realtime responsiveness:
- model calibration is often a process-heavy batch process,
- data studies often make heavy demands of the data layer.

Data for data studies, including sampling data from production runs, should be stored away from the production data pool. Calculations for derived model inputs should similarly be carried out outside the production application.

Time Interval Mechanics

Daily timeslots are collections of time intervals $\mathcal{T} = \{[t, t + \delta t)\}$ that tile a working day or interval of days. They are used to as inputs that contribute to a multi-vehicle route dispatch optimisation problem definition. For a time t in a working day, we can define $\tau(t)$ as the unique timeslot that contains t. For any time t up to the start of a timeslot, we can define a timeslot capacity $C_H(\tau, l, t)$, which represents what a network could deliver to an address l from a given hub, given a single network delivery run all from vehicles from $H \equiv H(V, l_H, A)$.

Timeslot capacity depends on
- hub location l_H and the delivery area A_H it covers,
- set of vehicles V available to H,
- total capacity of available vehicles at a hub at a given time t, $\{C_v(t); v \in V\}$. Note that $C_H(\tau, l, t) \leq \sum_{v \in V} C_v(t)$.

This in turn depends on the geographical area covered by the hub, A_H. Inferring the hub capacity from operational data is itself an interesting problem.

For simplicity, we'll assume that for a given region that any vehicle in A can reach any other part of the region within a timeslot.

We can dynamically manage time slot capacities to reduce the number of cases in which multiple vehicle journeys are sent to satisfy service requests to neighbourhoods, that could be satisfied by a single vehicle journey. One way:

1. tile the hub delivery area A_H into a collection $\mathcal{N}$ of *neighbourhoods.* Split the working day into a set of *delivery periods* in $\mathcal{P}$, in which a single complete delivery round is made by each vehicle in V. Each *delivery period* is divided into timeslots.
2. when a service is ordered to an address in a neighbourhood for a given delivery time, for the first time, the capacity for that timeslot is reduced by the service unit of delivery. Capacity for all other timeslots in the delivery period are reduced to 0 for that neighbourhood.
3. further orders to a neighbourhood for a given delivery time, capacity in for timeslots in the route reduce capacity to deliver for all timeslots in the delivery period. Over time, capacity of a particular timeslot might fall to 0. When this happens, no more bookings can be made within the delivery period.

Basic Model Limitations

1. The model provides no opportunity to shape demand, but
   a. delivery products can be *tiered* by value,
   b. product demand is *elastic* - changes in product price affect customer demand for the product.
2. The model prioritises on arrival order without accounting for the value of demand to come, or the *relative* cost of satisfying the demand, or the possibility of losing demand via timeslot manipulation.
   a. *time slot pricing* - we can control the availability of time slots *by product* to ensure market demand for high value delivery product is satisfied,
   b. *off-peak delivery* - we can prioritise time slots that minimise traffic delays, urban tariffs, or route emissions,
   c. *sales channel protection* - we can explicitly model customer behaviour in the presence of restricted choice, to reduce checkout wastage when timeslot capacity is shut down.

Revenue and Yield Management

Revenue management is the control of product inventory and company resources to maximise returns in response to customer demand. Yield management is the component of revenue management concerned with maximising spend from customer demand.

Revenue management systems (RMS) work well in cases where

- large amount of product is sold via automation,
- customer market is clearly segmented,
- constrained ability to meet demand,
- elastic demand for product,
- market segments can be captured with product offerings tiered by price,
- perishable product with definite end-by dates.

Delivery products are clearly a good fit with RMS. In particular,

- they allow standard product tiers with clearly separated price points,
- automation permits behaviour analysis of customers in different market segments.

Elements of a Yield Management Solution

A minimal extension of the timeslot allocation algorithm is to replace the first come, first serve mechanism with a *bid pricing* mechanism using a marginal value calculator. A simple version is described here.

Product Demand estimation

Demand generation is naturally modelled using a poisson process, and we can use historical data to calibrate an intensity parameter λ. We can model a λ as a constant intensity or as a model $\lambda(t,\tau)$ that admits hourly and seasonal variation, or as a more complex learning model, as the engine matures.

We should track demand forecasting error as a model KPI. The error distribution can then determine calibration frequency or diagnose model misspecification in case of model drift.

Optimisation

We are given

- a set of products $\mathcal{P}_i$ of deliveries that are booked i days in advance,
- booking price π_p for $p \in \mathcal{P}_i$,
- an expected demand function $D_p(t,\tau)$ for a booking timeslot τ on a booking date t,
- capacity of a network hub to deliver timeslot τ on a booking date t.

Each time a customer wants to book a timeslot for a given date, we calculate

$$\arg\max_{x_p \in \{0,1\}} \ \sum_{0<i\leq I} \sum_{p\in\mathcal{P}_i} \sum_{\tau} D_p\left(t+i,\tau\right) \ \pi_p \ x_p\left(\tau\right)$$

such that for all $\tau, 0 < i \leq I$,

$$\sum_{p\in\mathcal{P}_i} D_p\left(t+i,\tau\right) \ x_p(\tau) \leq C(\tau, t+i)$$

The products $p \in \mathcal{P}_i, 0 < i \leq I$ for which $x_p = 1$ are those for which timeslot classes can be booked on a given delivery day i days out from the current booking date. The effect of this formulation is to ensure that lower-value, early booking delivery products do not crowd out higher-value, late booking products.

We can go quite a long way with the concept of preserving time slot capacity for later demand. One concern we face when closing down time slot capacity is that we deter significant amounts of inflexible demand that requires certain timeslots. With demand models that account for user cancellation, we can hold out capacity for popular timeslots in situations where a first come first serve model might shut it down.

We can add a penalty term to penalise in a rough-and-ready way attempts to travel at rush hour.

$-\lambda \sum_{0\le i\le I} \sum_{p\in\mathcal{P}_i} \sum_{\tau} r(t,\tau)\; D_p(t+i,\tau),$

where

- λ weights the relative importance of the penalty term to the revenue maximising term,
- $r(\cdot,\cdot)$ provides hourly and seasonal contributions to the penalty weight.

Alternative Demand Generation - Modelling Customer Preference for Time Slots
We can model customer preference for timeslots to prevent unpopular timeslots arriving early in a booking period, then choking off timeslot capacity for more popular timeslots. In practice, this can reduce the likelihood that a shopper will abandon a purchase process because a preferred delivery option is not available. The logit model specification comes from [2], but rationale for its usage comes from [4] - presentation of a logit model permits relaxation of a combinatorial nonlinear optimiser into a linear optimisation problem for a great gain in computational efficiency.

Let $\Omega(n)$ be the set of timeslots that have capacity to deliver to neighbourhood n. Let $\Psi(n)\subseteq\Omega(n)$ be a set of timeslots currently on offer - these are controlled by the optimisation engine - and $\Psi^c(n)$ be the complement set.

Let $v_{n,\tau}\ge 0$ be a measure of preference that a shopper in a neighbourhood n has for a timeslot delivery $\tau\in\Psi(n)$ and let $w_{n,\tau}\ge 0$ be a tendency to cancel in disappointment if $\tau\notin\Psi(n)$. Write the probability that a customer in a neighbourhood n chooses a timeslot τ on offer as $P_{n,\tau}(\Psi(n))$, and abuse notation to let $P_{0,\tau}(\Psi(n))$ represent the probability that the customer does not complete a purchase, and let $v_{n,0}>0$ represent the unconditional attraction of a no-purchase event. Then

$$P_{n,\tau}(\Psi(n)) = \frac{v_{n,\tau}}{v_{n,0}+\sum_{l\in\Psi(n)} v_{n,l}+\sum_{k\in\Psi^c(n)} w_{n,k}}$$

and

$$P_{n,\tau}(\Psi(n)) = \frac{v_{n,0}+\sum_{k\in\Psi^c(n)} w_{n,k}}{v_{n,0}+\sum_{l\in\Psi(n)} v_{n,l}+\sum_{k\in\Psi^c(n)} w_{n,k}}.$$

The likelihood function for the logit model is well-behaved (convex), and the $v_{\cdot,\cdot},\; w_{\cdot,\cdot}$ can be estimated using maximum likelihood estimation from captured purchase data.

Full Revenue Management Implementation
We formulate an online revenue management problem. For simplicity, we consider only one delivery product type

Demand for a delivery day books up over the booking period $[0,I]$. Consider the collection of small intervals $\{\delta i, 2\,\delta i,\cdots,I-\delta i, I\}$ in which at most one booking can be made with probability λ. Let the probability that a customer live in a neighbourhood n be μ_n. Then if $V_i(\mathbf{x})$ is expected revenue from the current booking state $\mathbf{x}=\{x\}_{n,\tau}$ of each neighbourhood n and booking slot τ,

$$V_i(\mathbf{x}) = \max_{\Psi(n)}\Big\{\sum_{n\in\mathcal{N}}\lambda\mu_n\sum_{\tau\in\Psi_{n,\tau}(n)} P_{n,\tau}(\Psi_{n,\tau}(n))(r+g(\tau)+V_{i+1}(\mathbf{x}+\delta x_{n,\tau}))$$
$$+\,(1-\sum_{n\in\mathcal{N}}\lambda\mu_n\sum_{\tau\in\Psi_{n,\tau}(n)} P_{n,\tau}(\Psi_{n,\tau}(n))\;V_{i+1}(\mathbf{x}))\Big\}$$

$$= \max_{\Psi(n)} \left\{ \sum_{n\in\mathcal{N}} \lambda\mu_n \sum_{\tau\in\Psi_{n,\tau}(n)} P_{n,\tau}(\Psi_{n,\tau}(n))(r + g(\tau) - (V_{i+1}(\mathbf{x}) - V_{i+1}(\mathbf{x} + \delta\mathbf{x}_{n,\tau}))) + V_{i+1}(\mathbf{x}) \right\}$$

The timeslot policy to optimise this problem is
$\Phi^*(n) = \arg\max_{\Psi_{n,\tau}(n)} \sum_{\tau\in\Psi_{n,\tau}(n)} P_{n,\tau}(\Psi_{n,\tau}(n))(r + g(\tau) - (V_{i+1}(\mathbf{x}) - V_{i+1}(\mathbf{x} + \delta\mathbf{x}_{n,\tau})))$.

Approximating a Route Optimization Calculation
Computing this solution requires us to estimate the contribution made by$V_{i+1}(\mathbf{x}) - V_{i+1}(\mathbf{x} + \delta\mathbf{x}_{n,\tau})$,

the cost change of delivery when an additional delivery is made to a neighbourhood n in a timeslot τ. In principle, this requires a solution to a full VRP, but this is not practicable. [2] and [4] propose an approximation scheme that uses a seeding scheme to estimate intra-neighbourhood route movement, added to movement costs between calculated neighbourhood centroids. Initially, it is appropriate to adopt a coarser approximation, taking into account only the routes taken between neighbourhood centroids. If the number of neighbourhoods gets high, we repeat the trick, we embed collections of neighbourhoods in super-neighbourhoods. We can either coarsen the approximation by calculating centroids at this level, or recurse within the tree structure for greater route distance precision.

Decision Diagrams
We can use *Decision Diagrams* (DDs) to calculate $V_{i+1}(\mathbf{x}) - V_{i+1}(\mathbf{x} + \delta\mathbf{x}_{n,\tau})$ in place of our heuristic. DDs also represent optimization problems as dynamic programming problems. They are great for

- early termination of an optimiser - we can always expect a solution to within a tolerance that we can query the model for. This allows smooth tradeoff between accuracy and speed,
- simulation - there is a more natural fit for decision diagrams within a simulation environment.

DDs are performant, achieving orders of magnitude improvements in timed performance for classes of combinatorial optimisation problems relating to routing [5].

Examples (WIP)

References (WIP)

1. *Optimization of Multiple Vehicle Routing Problems using Approximation Methods*, R. Nallusamy et al., International Journal of Science and Technology Vol 1 (3), 2009, 129-135.
2. *Choice-based dynamic time slot management in attended home delivery*, Jochen Makert, Computers and Industrial Engineering 129 (2019), 333-345.
3. *Assortment Planning under the Multinomial Logit Model with Totally Unimodular Constraint Structures*, Guillermo Gallego et al., Preprint, April 10, 2013.
4. *Choice-Based Demand Management and Vehicle Routing in E-Fulfillment*, Xinan Yan et al., Transportation Science, Published online in Articles in Advance 07 Aug 2014.
5. *Decision Diagrams for Solving Traveling Salesman Problems with Pickup and Delivery in Real Time*, Ryan J. O'Neil, Karla Hoffmann, Operations Research Letters (Preprint), 2019.