Online Spatio-Temporal Matching in Stochastic and Dynamic Domains

Meghna Lowalekar, Pradeep Varakantham, Patrick Jaillet[†]
School of Information Systems, Singapore Management University
[†]Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, USA
meghnal.2015@phdis.smu.edu.sg, pradeepv@smu.edu.sg, jaillet@mit.edu

Abstract

Spatio-temporal matching of services to customers online is a problem that arises on a large scale in many domains associated with shared transportation (ex: taxis, ride sharing, super shuttles, etc.) and delivery services (ex: food, equipment, clothing, home fuel, etc.). A key characteristic of these problems is that matching of services to customers in one round has a direct impact on the matching of services to customers in the next round. For instance, in the case of taxis, in the second round taxis can only pick up customers closer to the drop off point of the customer from the first round of matching. Traditionally, greedy myopic approaches have been adopted to address such large scale online matching problems. While they provide solutions in a scalable manner, due to their myopic nature the quality of matching obtained can be improved significantly (demonstrated in our experimental results). In this paper, we present a two stage stochastic optimization formulation to consider expected future demand. We then provide multiple enhancements to solve large scale problems more effectively and efficiently. Finally, we demonstrate the significant improvement provided by our techniques over myopic approaches on two real world taxi data sets.

1 Introduction

In shared transportation (Agatz et al. 2011; Ghosh et al. 2015; Santos and Xavier 2013) which includes taxis, ride sharing, super shuttles, etc., customers have to be assigned to services in an online fashion to optimize revenue or quality of service. Also in case of emergency services (Saisubramanian, Varakantham, and Lau 2015) such as ambulances, fire trucks, etc. and delivery services (Yang, Jaillet, and Mahmassani 2004; Ritzinger, Puchinger, and Hartl 2015) involving pickup and delivery of food, equipment, clothing, home fuel etc., online matching of customers to services is essential. Specifically, in the case of taxis, the wide usage of applications such as Uber, Lyft, etc. is a testament to the importance of doing such matching. In these applications, a set of available taxis are matched to customers that are looking for taxis. An aspect that is typically not considered in such applications and most existing work is that, after taking the matched customer to their destination location, the taxi will be available for another round of assignment to customers around the destination location.

In this paper, we are specifically interested in considering such dependence (ex: assignment for a taxi in second round is dependent on the assignment in first round) in matching across multiple steps in an online fashion in domains where (a) customer demand is uncertain and time dependent, with data available about past customer demand; (b) problems are at a societal scale with thousands of customers and services with a need to make online decisions; and (c) there is a need to optimize revenue or quality of service (ex: time to pick up customers or time to delivery);

Given the challenging nature of the problems (stochasticity, dynamism, societal scale, online, multi-step), most existing work on relevant problems (described in Section 2) has focussed on myopic algorithms like bipartite matching, greedy and randomised ranking. While these approaches have good competitive ratios, they have obvious inefficiencies due to myopic nature of matching. To address these, there has been research on two-stage stochastic models that consider samples of expected demand for the next stage. However, these approaches have been limited to small scale problems and in restricted settings. In this work, we also employ two-stage stochastic models to represent future uncertain and time dependent customer demand. Specifically, we make the following key contributions:
(1) After formalizing the online spatio-temporal matching problem, we first provide a two stage stochastic optimization formulation, which considers expected future customer demand (typically obtained from historial data) for finding assignment of taxis to customers.
(2) Given the large scale problems of interest, we provide a decomposition of the above formulation to improve parallelism in handling expected future demand.
(3) Apart from considering randomly generated problems, we also evaluated our approach on datasets of two major taxi companies. We compare against myopic algorithms (typically employed by standard taxi matching applications) such as greedy and one step bipartite optimal assignment. We show that two-stage problems can be solved in times that are competitive to the myopic approaches, while providing significantly better solutions.

2 Related Work

Most existing relevant research in online matching has focussed on online bipartite matching (Karp, Vazirani, and

Vazirani 1990), where one side of vertices (services) are fixed and the other side of vertices (customers) arrive incrementally. This problem has applications in online advertising employed by Google and other companies. It is shown that myopic algorithms such as greedy (that makes the best match when nodes arrive) and randomized ranking have competitive ratios[1] of $\frac{1}{2}$ and $1 - \frac{1}{e}$ respectively. Recently a significant progress has been made on models that represent stochastic arrivals of customers (Goel and Mehta 2008; Feldman et al. 2009; Manshadi, Gharan, and Saberi 2010; Bahmani and Kapralov 2010) and competitive ratio can be shown to increase to 0.729 (Jaillet and Lu 2014). Although most work in online bipartite matching addresses the problem where one side is fixed, results in (Blum, Sandholm, and Zinkevich 2006) show that greedy algorithm achieves a competitive ratio of 1/2 when both sides of vertices appear online. Recently in Wang *et al.* (Wang and Wong 2015) an algorithm based on water-filling algorithm has been proposed which achieves a competitive ratio of 0.526 in case of both sides of vertices appearing online. Unlike online bipartite matching that assigns one service to only one customer, in this paper, we match one service to multiple customers (with one customer at any specific point) over time. Spatio-temporal aspects also impacts the computational complexity of the matching.

Another line of existing research that is similar to the online multi-step matching problems is the online vehicle routing problem (Ritzinger, Puchinger, and Hartl 2015). Yang *et al.* (Yang, Jaillet, and Mahmassani 2004) present a MIP formulation for the offline problem which is applied at each decision epoch in the online strategy. Zhang *et al.* (Zhang, Smilowitz, and Erera 2011) present a sample scenario approach and a capacity reservation approach, both designed as two stage stochastic models within a rolling horizon framework for a multi resource routing problem. There has also been research on heuristic algorithms that consider two stage models to solve dynamic taxi and ridesharing problems (Agatz et al. 2011; Herbawi and Weber 2012; Santos and Xavier 2013). While, we also consider a two stage stochastic model in our work, our differentiating contributions are in (a) providing techniques (abstraction and benders decomposition) that help perform matching on real world societal scale problems online; and (b) demonstrating the utility of our techniques over the widely used myopic approaches on real world taxi data sets where there are up to 3000 requests at each time step and thousands of taxis assigned to those requests.

3 Online Spatio-Temporal Matching

An Online Spatio-Temporal Matching problem is described using the following tuple:

$$\left\langle \mathcal{Z}, \mathcal{D}, \mathcal{N}, \mathcal{C}, \xi^D, \xi^S \right\rangle$$

TSS$(\mathcal{Z}, \mathcal{D}, \mathcal{N}, \mathcal{C}, \xi^D, \xi^S)$**:**

$$\max \quad \sum_{i\in\mathcal{Z}} \sum_{j\in\mathcal{D}} \mathcal{C}_{i,o_j,d_j} * x^1_{ij} + \frac{1}{|\xi^D|} \sum_{k\le|\xi^D|} \sum_{j\in\xi^{D,k}} \sum_{i\in\mathcal{Z}} \mathcal{C}_{i,o_j,d_j} * x^{2,k}_{ij} \tag{1}$$

$$\textbf{s.t.} \quad \sum_{j\in\mathcal{D}} x^1_{ij} \le \mathcal{N}^1_i \;:::\; \forall i \in \mathcal{Z} \tag{2}$$

$$\sum_{i\in\mathcal{Z}} x^1_{ij} \le 1 \;:::\; \forall j \in \mathcal{D} \tag{3}$$

$$\sum_{j\in\xi^{D,k}} x^{2,k}_{ij} \le \mathcal{N}^1_i - \sum_{j\in\mathcal{D}} x^1_{ij} + \xi^S_i + \sum_{j\in\mathcal{D}, d_j = i} \sum_{m\in\mathcal{Z}} \delta^{1,2}_{mj} * x^1_{mj} \;:::\; \forall i \in \mathcal{Z}, k \le |\xi^D| \tag{4}$$

$$\sum_{i\in\mathcal{Z}} x^{2,k}_{ij} \le 1 \;:::\; \forall k \le |\xi^D|, j \in \xi^{D,k} \tag{5}$$

$$x^1_{ij} \in \{0,1\} \;:::\; \forall i \in \mathcal{Z}, j \in \mathcal{D} \tag{6}$$

$$x^{2,k}_{ij} \in \{0,1\} \;:::\; \forall i \in \mathcal{Z}, \forall k \le |\xi^D|, j \in \xi^{D,k} \tag{7}$$

- $\mathcal{Z}$ is the set of all zones. A zone is appropriately defined based on the specific problem domain.
- $\mathcal{D}$ represents the demand or the set of customer requests for services. Each request $j \in \mathcal{D}$ is characterized by a tuple of the origin and destination zones: $\langle o_j, d_j \rangle$.
- $\mathcal{N}^t_i$ indicates number of suppliers in zone $i \in \mathcal{Z}$ at time t.
- $\mathcal{C}$ represents the objective (ex: revenue, number of requests served, negative of waiting time), with $\mathcal{C}_{i,o_j,d_j}$ denoting the objective value obtained by matching customer request, j (having origin and destination zones $\langle o_j, d_j \rangle$) with a supplier in zone i.
- ξ^D is the set of customer request samples for the next time step, where $\xi^{D,k}$ represents customer requests in sample k.
- ξ^S is the set of suppliers that will become available in next time step, where ξ^S_i represents the number of suppliers that will be available in zone i.

Our goal is to identify the assignment of requests to suppliers so as to maximize the sum of objective values for the current time step and the expected objective values for the next time step.

Given the online nature of decision making, we now provide a two stage stochastic optimization formulation for computing the best match at this time step ($t = 1$) while considering multiple samples of potential requests and suppliers in the next time step ($t = 2$). We refer to this as **TSS**(). We use binary variables, x^t_{ij} to indicate if request j is assigned to supplier in zone i at decision epoch t. $\delta^{t,t'}_{mj}$ is a constant that is set to 1 if a zone m supplier assigned to request j at decision epoch t completes its service on or before t', $t' > t$.

While the first component of the objective computes objective value corresponding to the current time step, the second component computes the expected value corresponding to the future requests (provided in ξ^D). Constraints (2) and (4) ensure that at any decision epoch,

[1]Competitive ratio for an online algorithm is defined as the worst case value for the ratio of the obtained solution by the online algorithm to the optimal offline solution (assuming the arrival of nodes is known before hand).

the number of assigned suppliers from zone i is less than the number of available suppliers. In Constraint (4) the number of suppliers available at decision epoch "2" in zone i is calculated by considering the remaining suppliers in zone i after doing assignment for decision epoch "1". Constraints (3) and (5) ensure that at any decision epoch each request is assigned to only one supplier. The above program is an integer program but for a special case, Proposition 1 shows that the relaxed linear program has an integer optimal solution.

We know that for linear programs of the form $\max\ cx\ s.t.\ Ax \leq b$, the solution of the optimization program will always be integer if b contains all integer values and A is a totally unimodular matrix (Hoffman and Kruskal 2010).

In the optimization formulation of **TSS**(), the right hand side is always integer. For a special case when there is a single sample and all the suppliers finish servicing the requests between two decision epochs, we show that the constraint matrix is totally unimodular, so the linear program will have integral optima.

Proposition 1. *If* $|\xi^D| = 1$ *and* $\forall m, j, t \quad \delta_{mj}^{t,t+1} = 1$, *then the constraint matrix for* ***TSS()*** *problem is unimodular.*

Proof Sketch. From integer program theory (Bertsimas and Weismantel 2005), a matrix A is totally unimodular if every square submatrix of A has determinant in the set $\{-1, 0, 1\}$. An equivalent definition of total unimodularity (TU) and often easier to establish is captured in the following theorem by Bertsimas *et al.*:

Theorem 1. *(Bertsimas and Weismantel 2005): Let A be a matrix. Then A is TU iff for any subset of rows S of A, there exists a coloring of rows of S, with 1 or -1 such that the weighted sum of every column (while restricting the sum to rows in S) is -1, 0 or 1.*

For a single sample, we can express the constraints in **TSS**() in the form $Au \leq b$ where u is the vector

$$\text{u} = \begin{bmatrix} x_{11}^1 & .. & x_{|\mathcal{Z}||\mathcal{D}|}^1 & x_{11}^{2,1} & .. & x_{|\mathcal{Z}||\xi^{D,1}|}^{2,1} \end{bmatrix}^T$$

i.e. u will have $|\mathcal{Z}||\mathcal{D}| + |\mathcal{Z}||\xi^{D,1}|$ elements.

The block matrix A is of the form

$$\text{A} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \\ A_{31} & A_{32} \\ A_{41} & A_{42} \end{bmatrix} = \begin{bmatrix} A_{11} & 0 \\ A_{21} & 0 \\ A_{31} & A_{32} \\ 0 & A_{42} \end{bmatrix}$$

where A_1 corresponds to constraints in equation (2), A_2 corresponds to constraints in equation (3), A_3 corresponds to constraints in equation (4) and A_4 corresponds to constraints in equation (5). Matrices A_{11}, A_{21}, A_{31} and A_{41} correspond to decision epoch "1" variables (columns) and matrices A_{12}, A_{22}, A_{32} and A_{42} correspond to decision epoch "2" variables (columns). Let S be the subset of rows of A and $S = S_1 \cup S_2 \cup S_3 \cup S_4$ where $S_i \in A_i$. We can then show[2] for each of the cases (over subsets of rows), that there exists a coloring of rows such that weighted sum of columns is -1, 0, or 1. ■

[2]Details omitted due to space constraints

For a general case with multiple samples, the constraint matrix is not totally unimodular. When all the suppliers don't finish servicing between 2 consecutive decision epochs, we were able to find an example where the solution obtained by solving the relaxed version of the problem was not integral. At this juncture, we are unable to find a formal proof to demonstrate the integrality when suppliers finish servicing customers between 2 consecutive decision epochs. However, in our experiments on synthetic domains and the two real world datasets, we always (irrespective of whether suppliers finish servicing within 2 consecutive decision epochs or not) obtained integral solutions on solving the relaxed problem. Therefore, our approach is to solve the relaxed version of the problem and in case the solution is not integral, we round to an integer solution as described below.

While converting a fractional solution to integer, only the parts of the solution that are fractional are modified. From the fractional part, variables x_{ij}^1 are rounded in such a way that the number of suppliers assigned from each zone at decision epoch "1" and the number of supplier arriving in any zone at decision epoch "2" remain close to the fractional optimal solution. This ensures that suppliers which were left unassigned by the TSS, remain unassigned and the assignments at second stage are the least affected. At each iteration of the algorithm, requests which can be completed before the next decision epoch are assigned a supplier from the zone which has the maximum number of suppliers available (based on the fractional part of solution). When the number of supplier arriving in any zone i at decision epoch "2" becomes greater than or equal to the number of supplier arriving in fractional solution, no other request with destination in zone i is served. If after assigning suppliers in this manner, there is still any unassigned supplier(based on the fractional part of solution), it is greedily assigned to an unserved request.

3.1 Benders Decomposition

Given the scale of problems of interest in this paper (i.e.: thousands of taxis serving thousands of customers spread across hundreds of zones) , we reduce the complexity associated with increasing the number of samples by exploiting the following observation:

Observation 1. *In* ***TSS()****, once the assignment at first decision epoch,* $\{x_{ij}^1\}$ *is given, optimization models for computing assignment at second decision epoch,* $\{x_{ij}^{2,k}\}$ *for each of the samples k, are independent of each other.*

We exploit Observation 1 by using the Benders Decomposition (Benders 1962) method, a master slave decomposition technique where **the Master Problem** is responsible for obtaining solutions for the "difficult" variables; and **the Slave problem(s)** is (are) responsible for other variables and obtaining cuts (that are added to the master) based on fixed assignment of values to "difficult" variables.

Based on Observation 1, $\{x_{ij}^1\}$ are the difficult variables as they impact the values assigned to all other variables. Therefore, the master is responsible for obtaining assignments for $\{x_{ij}^1\}$ variables and the slave(s) are responsible for obtaining assignments to $\{x_{ij}^{2,k}\}$. For the master (Table 1), in the optimization provided in **TSS**() ,we replace the part of the objective dealing with future variables, $\{x_{ij}^{2,k}\}$ by the recourse function $\mathcal{Q}(\{x_{ij}^1\}_{i\in\mathcal{Z},j\in\mathcal{D}},k)$ which becomes the objective function in the slave problems. The recourse function $\mathcal{Q}()$ needs to be computed for each value of x_{ij}^1. In the slaves (Table 2), we consider fixed values of x_{ij}^1 and to avoid confusion, we refer to them using the capital letter notation, X_{ij}^1. By duality theory (Murphy 2013) we can show that by taking the dual of slave problems, we can find an upper bound on the value of the recourse function in terms of the master problem variables x_{ij}^1. These can then be added as an optimality cut to the master problem for generating better first stage assignments[3]. The dual for the slave problem is provided in Table 3. Let θ^k be the approximation of $\mathcal{Q}()$ function then the master problem with optimality cuts is provided in Table 4. It should be noted that we are using x_{ij}^1 variables in the "master with optimality cuts" and not the fixed values, X_{ij}^1. In each iteration we solve the master problem and the computed x_{ij}^1 variable values are passed to dual slave problems. After solving dual slave problems, optimality cuts are generated. If the current values of $\theta^k(\forall k)$ satisfy the optimality cut conditions then we have obtained an optimal solution else cuts are added to the master problem and the master problem is solved again. As we can see in the "Dual Slave" linear programs, the slave problems are independent of each other and are only connected by the choice of the master variables ("difficult" variables). Therefore, once the master variables are fixed, the slave problems can be solved in a parallel fashion.

Master:

$$\max \sum_{i\in\mathcal{Z}}\sum_{j\in\mathcal{D}} \mathcal{C}_{i,o_j,d_j} * x_{ij}^1 + \frac{1}{|\xi^D|}\sum_{k\le|\xi^D|}\mathcal{Q}(\{x_{ij}^1\}_{i\in\mathcal{Z},j\in\mathcal{D}},k) \quad (8)$$

$$\textbf{s.t.} \quad \sum_{j\in\mathcal{D}} x_{ij}^1 \le \mathcal{N}_i^1 ::: \forall i\in\mathcal{Z} \quad (9)$$

$$\sum_{i\in\mathcal{Z}} x_{ij}^1 \le 1 ::: \forall j\in\mathcal{D} \quad (10)$$

Table 1: Master Formulation

Slave $(\{X_{ij}^1\}_{i\in\mathcal{Z},j\in\mathcal{D}},k)$

$$\max \sum_{i\in\mathcal{Z}}\sum_{j\in\xi^{D,k}} \mathcal{C}_{i,o_j,d_j} * x_{ij}^{2,k} \quad (11)$$

$$\sum_{j\in\xi^{D,k}} x_{ij}^{2,k} \le \mathcal{N}_i^1 - \sum_{j\in\mathcal{D}} X_{ij}^1 + \xi_i^S + \sum_{j\in\mathcal{D},d_j=i}\sum_{m\in\mathcal{Z}} \delta_{mj}^{1,2} * X_{mj}^1 ::: \forall i\in\mathcal{Z} \quad (12)$$

$$\sum_{i\in\mathcal{Z}} x_{ij}^{2,k} \le 1 ::: \forall j\in\xi^{D,k} \quad (13)$$

Table 2: Slave Formulation

Dual Slave $(\{X_{ij}^1\}_{i\in\mathcal{Z},j\in\mathcal{D}},k)$**) :**

$$\min \sum_{i\in\mathcal{Z}} \alpha_i * (\mathcal{N}_i^1 - \sum_{j\in\mathcal{D}} X_{ij}^1 + \xi_i^S + \sum_{j\in\mathcal{D},d_j=i}\sum_{m\in\mathcal{Z}} \delta_{mj}^{1,2} * X_{mj}^1) + \sum_{j\in\xi^{D,k}} \beta_j \quad (14)$$

$$\textbf{s.t.} \quad \alpha_i + \beta_j - \mathcal{C}_{i,o_j,d_j} \ge 0 ::: \forall i\in\mathcal{Z}, j\in\xi^{D,k} \quad (15)$$

$$\alpha_i \ge 0 ::: \forall i\in\mathcal{Z} \quad (16)$$

$$\beta_j \ge 0 ::: \forall j\in\xi^{D,k} \quad (17)$$

Table 3: Dual Slave Formulation

Master Formulation with Optimality Cuts:

$$\max \sum_{i\in\mathcal{Z}}\sum_{j\in\mathcal{D}} \mathcal{C}_{i,o_j,d_j} * x_{ij}^1 + \frac{1}{|\xi^D|}\sum_{k\le|\xi^D|}\theta^k \quad (18)$$

$$\textbf{s.t.} \quad \theta^k \le \sum_{i\in\mathcal{Z}} \alpha_i * (\mathcal{N}_i^1 - \sum_{j\in\mathcal{D}} x_{ij}^1 + \xi_i^S + \sum_{j\in\mathcal{D},d_j=i}\sum_{m\in\mathcal{Z}} \delta_{mj}^{1,2} * x_{mj}^1) + \sum_{j\in\xi^{D,k}} \beta_j \quad (19)$$

$$\sum_{j\in\mathcal{D}} x_{ij}^1 \le \mathcal{N}_i^1 ::: \forall i\in\mathcal{Z} \quad (20)$$

$$\sum_{i\in\mathcal{Z}} x_{ij}^1 \le 1 ::: \forall j\in\mathcal{D} \quad (21)$$

Table 4: Master Formulation with Optimality Cuts

3.2 Competitive Ratio

Proposition 2. *In an adversarial model, when the objective of multi round matching is to maximize the number of requests satisfied, the competitive ratio of any deterministic one step algorithm*[4] *(i.e with no future information available) is bounded from above by 1/2, provided no new suppliers arrive in second stage.*

Proof: Let $\mathcal{N}^1$ is the total number of suppliers available at t=1 and D^1 and D^2 are the demand at t=1 and 2 respectively. The worst case will occur when $D^1 >> \mathcal{N}^1$ and at t=1, a one step algorithm matches $\mathcal{N}^1$ suppliers to requests such that none of the matched supplier will be available[5] at t=2 and two stage offline optimal algorithm finds a matching such that all the matched suppliers at t=1 are available at t=2.
At t=2,in that case an offline optimal algorithm will be able to serve $\min(\mathcal{N}^1, D^2)$ requests and the one step approach

[3]As the slave problems are always feasible for any value of master variables we only need to add optimality cuts to the master problem.

[4]with respect to a 2-step optimal algorithm

[5]Demand can be created at second stage in such a way that no supplier will be able to serve any request

Optimal($M, \{\mathcal{D}^t\}$):

$$\max \sum_{t=1}^{M} \sum_{i \in \mathcal{Z}} \sum_{j \in \mathcal{D}^t} \mathcal{C}_{i,o_j,d_j} * x_{ij}^t \tag{22}$$

$$\textbf{s.t.} \quad \sum_{j \in \mathcal{D}^t} x_{ij}^t \leq \mathcal{N}_i^t \text{ ::: } \forall i, t = 1 \tag{23}$$

$$\sum_{i \in \mathcal{Z}} x_{ij}^t \leq 1 \text{ ::: } \forall t, j \in \mathcal{D}^t \tag{24}$$

$$\sum_{j \in \mathcal{D}^t} x_{ij}^t \leq \mathcal{N}_i^1 - \sum_{t'=1}^{t-1} \sum_{j \in \mathcal{D}^{t'}} x_{ij}^{t'} + \sum_{t'=1}^{t-1} \sum_{\substack{j \in \mathcal{D}^{t'}, \\ d_j = i}} \sum_{m \in \mathcal{Z}} \delta_{mj}^{t',t} * x_{mj}^{t'} \text{ ::: } \forall i, t > 1 \tag{25}$$

$$x_{ij}^t \in \{0, 1\} \quad \forall i, j, t \tag{26}$$

will not be able to serve any requests. So the competitive ratio of this one step algorithm is $\frac{\mathcal{N}^1}{(\mathcal{N}^1 + min(\mathcal{N}^1, D^2))}$, which is smallest when $D^2 > \mathcal{N}^1$ resulting in a competitive ratio of $\frac{\mathcal{N}^1}{(\mathcal{N}^1 + \mathcal{N}^1)} = \frac{1}{2}$. So, in case of multi round matching, any deterministic one step algorithm can be at most 1/2 optimal.■

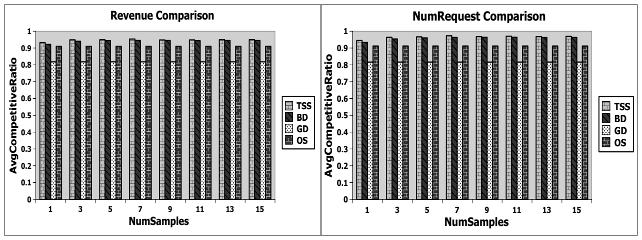
Proposition 2 provides the competitive ratio for the case when the objective function is to maximize the number of requests satisfied. However, we do not have any theoretical guarantees when the objective is to maximize revenue. So, for **TSS**() formulation and other one step approaches, we compute the competitive ratio experimentally by comparing the solution values obtained in comparison with the M-Stage optimal solution i.e., when all the requests at all the M-stages are known in advance. We use the integer program in **Optimal**() to compute the M-Stage optimal offline solution. Number of suppliers available at any stage depends on assignments at all previous stages. $\mathcal{D}^t$ is the demand or set of customer requests at decision epoch t. Constraints (24) ensure that each request is assigned to only one supplier at each decision epoch. Constraints (23) and (25) ensure that at any decision epoch the number of assigned suppliers from zone i is less than the number of available suppliers.

4 Experiments

In this section, we compare the performance of our approaches, Two Stage Stochastic optimization (TSS) and Benders Decomposition (BD) with the optimal approach (Section 3.2) and myopic approaches: Greedy (GD) algorithm, One Step optimization (OS). We employ the following metrics over M (=10) decision epochs:
(1) Total revenues of the taxis .
(2) Number of requests satisfied.
(2) Average wait time for customers.
(4) Run-time to compute a single assignment.

4.1 Setup

We conducted our experiments by taking demand distribution from real world datasets (henceforth referred to as



(a) (b)

Figure 2: Dataset 1: $|\mathcal{Z}| = 483$, $\sum_{i \in \mathcal{Z}} \mathcal{N}_i^1 = 2000$, $\frac{1}{10} \sum_{t=1}^{10} \mathcal{D}^t = 486$

Dataset1 and Dataset2) of two taxi companies[6] in a big asian city. These datasets contain data of past customer requests for taxis, revenues and times for travelling between different locations/zones at different times of the day and for different days of the week. To capture taxi movements and customer demand accurately, we divided the city into 483 zones. This division entails that in any zone, the maximum distance between any two points is less than 0.5 KM[7].

While **computing an assignment** at decision epoch t, for our approaches TSS and BD, we consider samples of customer requests at $t + 1$ from past data (at the same time on a weekday/weekend depending on whether $t + 1$ is on a weekday/weekend). Once the assignment is computed, we **evaluate the assignment** on realized requests (which are samples from past data that are not considered while computing the assignment at $t + 1$).

We consider a decision epoch every 15 minutes and we evaluate our online approach over 10 sequential decision epochs. At the start of the experiment taxis are distributed uniformly in different zones. Based on the assignment obtained by algorithms at any decision epoch availability of taxis at the next decision epoch is updated. We evaluate the approaches by running them on 10 different days and taking the average values over 10 days. The objective for all the four algorithms is to maximize revenue. To provide the right tradeoff between run-time and solution quality, we terminated BD after 3 iterations.

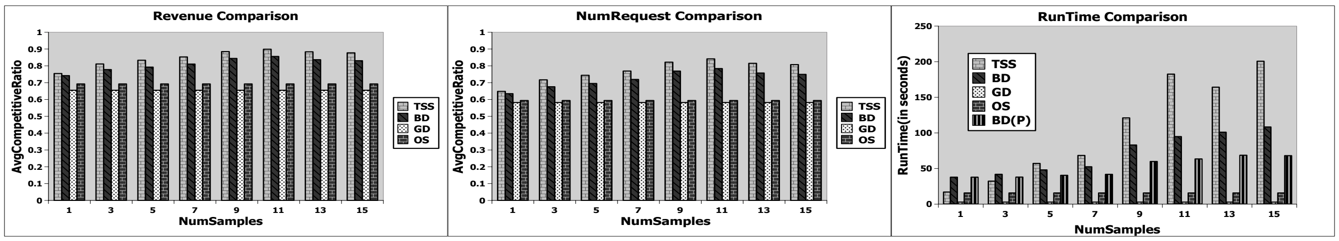
4.2 Results

We first provide results on Dataset1 by varying the number of samples[8]. We performed experiments with requests at various times of the day, 8:00 AM, 12:00 Noon, 3:00 PM, 6:00 PM and on weekday and weekend. Here, we provide results for requests at 8:00 AM, because of the high variance in requests at that time (minimum: 1514, maximum: 8946). The average number of requests over the 10 time steps was 3332 and the number of available taxis was set at 2000. Figure 1 provides the results for revenue, number of requests satisfied and runtime for computing assignment. In Figure 1a

[6]Names not revealed due to confidentiality agreements.

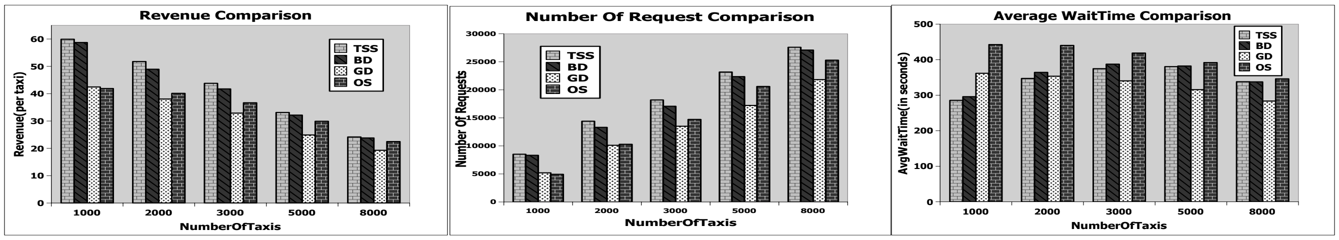
[7]We had a similar set of results for a division of 927 zones with a maximum of 0.2 KM between any two points in any zone.

[8]We obtained similar results on Dataset2.



(a) (b) (c)

Figure 1: Dataset 1: $|\mathcal{Z}|$ = 483, $\sum_{i\in\mathcal{Z}} \mathcal{N}_i^1 = 2000$, $\frac{1}{10}\sum_{t=1}^{10} \mathcal{D}^t = 3332$



(a) (b) (c)

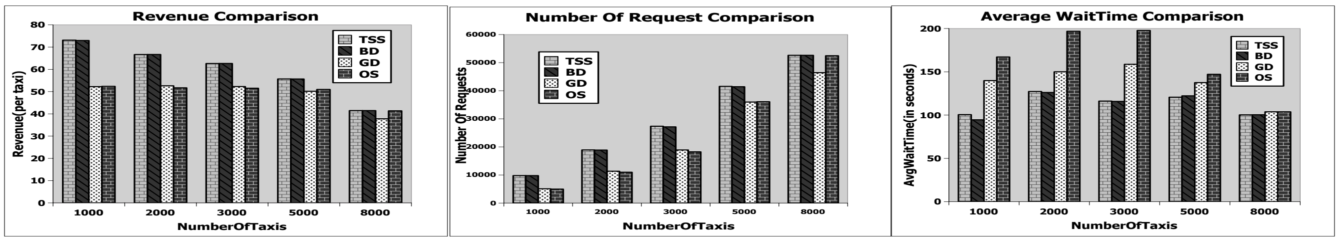
Figure 3: Dataset 1: In (a),(b) and (c) $|\mathcal{Z}|$ = 483,$|\xi^D|$ = 10, $\frac{1}{10}\sum_{t=1}^{10} \mathcal{D}^t = 3332$

and Figure 1b, X-axis denotes the number of samples considered while computing the assignment and Y-axis denotes the competitive ratio (from Section 3.2). The key observations are as follows:
(1) The most significant result is that TSS and BD are able to achieve more than 85% of optimal revenue at 11 samples. (2) Even with 1 sample, there is a clear improvement provided by TSS and BD over the myopic approaches (OS and GD). (3) TSS provides 5% additional improvement with respect to revenue and 10% improvement with respect to requests over BD. (4) BD provides the right tradeoff between runtime and solution quality. While TSS provides high quality solutions, it takes significantly more time (up to 100 seconds when making online decisions) and myopic approaches provide solutions very quickly but solution quality is significantly lower. (5) Since, the reported run-time for BD is based on sequential solving of slaves, the actual run-time when slaves are run in parallel on multiple cores will be lower. In figure 1c, we show the expected time which will be taken by BD, if slaves are solved in parallel(denoted by BD(P) in the figure). We calculated this time by considering the time taken to solve slaves in parallel as the maximum time taken by any slave in each iteration. Moreover, we performed our experiments on academic systems,on commercial systems, runtime should be much lower.

Figure 2 provides the results for revenue and number of requests satisfied on a weekend day at 8:00 AM. In this case, the key difference in set up is that the average number of requests (486) is lower than the number of taxis (2000). Both our approaches were able to complete within 10-15 seconds. Even in this example, where there are so many additional taxis, adding one sample improves the competitive ratio by 3% and at 5 samples, the difference is close to 5% with respect to both revenue and number of requests served.

Next we compare the algorithms by experimenting with different number of taxis for a fixed number of samples (10). We experimented by considering demand distributions from both datasets. Figure 3 provides results for Dataset1 and here are the key observations:
(1) With respect to revenue and number of requests served, when number of taxis is lower, the gap between TSS, BD and myopic approaches is significant. With respect to revenue, TSS and BD in the best case provided close to 20$ improvement per taxi and with respect to number of requests, our approaches are able to serve an additional 3500 requests. This is a significant result, because in most cities at rush hours, the number of taxis is almost always lower than the actual demand available. (2) While it is not the optimization criterion for our approaches, we are still able to perform better than myopic approaches with respect to average wait time. (3) As we increase the number of taxis gap between TSS, BD and myopic approaches (GD and OS) decreases. This is because when more taxis are available, taxis will be free even after assigning taxis at current decision epoch, so future demands can be met irrespective of current assignment. Furthermore, when there are significantly more taxis than demand, sophisticated matching approaches are not required.

We did a similar comparison of revenue, number of requests and waittime on Dataset2 and obtained similar results, with TSS and BD outperforming GD and OS on revenue and number of requests. Unlike in Dataset1, the quality



(a) (b) (c)

Figure 4: Dataset 2: In (a),(b) and (c)$|\mathcal{Z}| = 483$,$|\xi^D| = 10$, $\frac{1}{10}\sum_{t=1}^{10}\mathcal{D}^t = 5261$

of BD solutions was equivalent to TSS solutions even with early termination.

5 Conclusion

In this paper, we presented a two stage stochastic formulation for online assignment of suppliers to customers. We also presented a Benders decomposition of the formulation to deal with large number of future scenarios. We compared the assignment computed by our our algorithms against greedy and one step algorithms and found that even with one sample of future demand, we can get a large gain in competitive ratio which will in turn help in increasing revenue and customer satisfaction. In the future, we plan on implementing this for operations of a real taxi company and technically, extend this work to consider multiple stages of sampling instead of only one future sampling stage.

6 Acknowledgements

This work was partially supported by the Singapore National Research Foundation through the Singapore-MIT Alliance for Research and Technology (SMART) Centre for Future Urban Mobility (FM).

References

Agatz, N.; Erera, A. L.; Savelsbergh, M. W.; and Wang, X. 2011. Dynamic ride-sharing: A simulation study in metro atlanta. *Procedia-Social and Behavioral Sciences* 17:532–550.

Bahmani, B., and Kapralov, M. 2010. Improved bounds for online stochastic matching. In *Algorithms - ESA 2010, 18th Annual European Symposium, Liverpool, UK, September 6-8, 2010. Proceedings, Part I*, 170–181.

Benders, J. F. 1962. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik* 4(1):238–252.

Bertsimas, D., and Weismantel, R. 2005. *Optimization Over Integers*. Dynamic Ideas.

Blum, A.; Sandholm, T.; and Zinkevich, M. 2006. Online algorithms for market clearing. *J. ACM* 53(5):845–879.

Feldman, J.; Mehta, A.; Mirrokni, V. S.; and Muthukrishnan, S. 2009. Online stochastic matching: Beating 1-1/e. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*, 117–126.

Ghosh, S.; Varakantham, P.; Adulyasak, Y.; and Jaillet, P. 2015. Dynamic redeployment to counter congestion or starvation in vehicle sharing systems. In *Eighth Annual Symposium on Combinatorial Search*.

Goel, G., and Mehta, A. 2008. Online budgeted matching in random input models with applications to adwords. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*, 982–991.

Herbawi, W. M., and Weber, M. 2012. A genetic and insertion heuristic algorithm for solving the dynamic ridematching problem with time windows. In *Genetic and Evolutionary Computation Conference (GECCO)*, 385–392. ACM.

Hoffman, A. J., and Kruskal, J. B. 2010. Integral boundary points of convex polyhedra. In *50 Years of Integer Programming 1958-2008*. Springer. 49–76.

Jaillet, P., and Lu, X. 2014. Online stochastic matching: New algorithms with better bounds. *Math. Oper. Res.* 39(3):624–646.

Karp, R. M.; Vazirani, U. V.; and Vazirani, V. V. 1990. An optimal algorithm for on-line bipartite matching. In Ortiz, H., ed., *STOC*, 352–358. ACM.

Manshadi, V. H.; Gharan, S. O.; and Saberi, A. 2010. Online stochastic matching: Online actions based on offline statistics. *CoRR* abs/1007.1673.

Murphy, J. 2013. Benders, nested benders and stochastic programming: An intuitive introduction. *CoRR* abs/1312.3158.

Ritzinger, U.; Puchinger, J.; and Hartl, R. F. 2015. A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research* (ahead-of-print):1–17.

Saisubramanian, S.; Varakantham, P.; and Lau, H. C. 2015. Risk based optimization for improving emergency medical systems. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Santos, D. O., and Xavier, E. C. 2013. Dynamic taxi and ridesharing: A framework and heuristics for the optimization problem. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*.

Wang, Y., and Wong, S. C. 2015. Two-sided online bipartite matching and vertex cover: Beating the greedy algorithm. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, 1070–1081.

Yang, J.; Jaillet, P.; and Mahmassani, H. S. 2004. Real-time multivehicle truckload pickup and delivery problems. *Transportation Science* 38(2):135–148.

Zhang, G.; Smilowitz, K.; and Erera, A. 2011. Dynamic planning for urban drayage operations. *Transportation Research Part E: Logistics and Transportation Review* 47(5):764–777.