Models and Algorithms for Distributed Order Management

Michael Berezansky[2,6], Yaron Fairstein[1,5], Luke Marshall[2,6], Ishai Menache[2,6], Joseph (Seffi) Naor[1,5], Ola Svensson[3,7], and Timur Tankayev[4,8]

[1] Technion – Israel Institute of Technology
[2] Microsoft
[3] EPFL
[4] Georgia Institute of Technology
[5] yyfairstein, naor@cs.technion.ac.il
[6] mibereza,luke.marshall,ishai@microsoft.com
[7] ola.svensson@epfl.ch
[8] timur.tankayev@gatech.edu

Abstract. The emergence of cloud computing has revolutionized various business sectors, such as transportation, health care and retail. In particular, the ability to bring together huge amounts of data with accessible compute resources, has opened the gate for solving large-scale decision problems associated with the underlying data. One such example is Distributed Order Management (DOM), a key component in modern retail applications. The goal of DOM is assigning orders of customers to stores (or warehouses) while minimizing fulfillment costs (e.g., shipping or transport costs). We introduce here a formal mathematical framework to address the underlying optimization problems. In particular, we define several variants that differ in their objective (e.g., minimize fulfillment cost, maximize satisfied orders) and constraints (e.g., unsplittable vs. splittable assignments of orders to stores). We present approximation algorithms for the different models with proven lower and upper bounds.

Keywords: Order management · Discrete optimization · Approximation Algorithms.

1 Introduction

The majority of retailers nowadays adopt the so-called *omnichannel* commerce strategy. In the omnichannel commerce world, customer orders are created through different channels such as brick and mortar stores, e-commerce websites, or call centers. Orders may be fulfilled by a network of fulfillment locations such as warehouses, fulfillment centers, third party distribution services, and retail stores. Some products may also be shipped directly from a manufacturer to the customer. Each order includes fulfillment related attributes such as fulfillment type (e.g., store pickup or shipment), expected delivery date, and delivery method (e.g., standard or expedited shipment). The basic fulfillment problem that the

order management system needs to solve is deciding which location(s) fulfill each of the orders and then release this information to the fulfillment locations.

The emergence of cloud computing allows retailers to address these fulfillment problems from a centralized global view of all relevant elements, such as inventory levels, customer orders, and fulfillment costs. Indeed, several enterprises such as IBM, Oracle and Microsoft, are offering software-as-a-service solutions for the underlying allocation problems, often called *Distributed Order Management* (DOM) [7]. In a nutshell, the main goal of DOM is to produce a fulfillment plan that maximizes the number of orders fulfilled according to customer expectations and/or minimizes the total fulfillment cost. Generally, DOM differs from traditional order managements systems in the configuration flexibility it accommodates, in particular:

- *Order splitting.* Some retailers prefer that orders are fulfilled from a single location. To increase fulfillment of orders and reduce costs, other retailers allow orders to be fulfilled from multiple locations.
- *Affine cost structure.* Fulfilling an order (or parts thereof) comes with a "cost". The cost can be a combination of a fixed component (e.g., proportional to the travel-distance of a truck from source to destination), and a linear component which depends on quantity (e.g., as in packet shipping).

We provide here a rigorous study of DOM-related optimization. Our main contribution is twofold. We first formulate different variants of the underlying optimization problems, which account for the rich configuration flexibility of retailers. Based on that, we then design approximation algorithms with guaranteed bounds on performance. While there has been work in the Operations Research literature on order fulfillment in retail contexts (see, e.g., [10, 6, 1] and references therein), to the best of our knowledge there has not been work on approximation algorithms that captures the configuration flexibility of current DOM solutions.

1.1 Problem Definition

In the Distributed Order Management (DOM) problem there are: a set N of customers, a set M of stores, and a catalog K of items. There is an *order* associated with each customer, where an order is a collection of items purchased by the customer. Items are aggregated into lines, which means that each item k, ordered by customer j, defines a *line*, and d_{jk} denotes the number of items in the line, i.e., the number of items the customer ordered. Each store i has a finite stock of items of each type – the available stock of item k by store i is denoted by s_{ik}. Another important parameter throughout the paper is $\delta := \max_{j \in N} \sum_{k \in K} \mathbf{1}\{d_{jk} > 0\}$, the maximum number of lines in a single order (of a customer).

Delivering a unit of item k to customer j from store i generates revenue, while also incurring a shipping cost which is composed of a *fixed* cost component and a *variable cost* component, depending on the quantity delivered.

The goal in the DOM problem is to satisfy the orders of the customers. There are two main models that we consider. We first consider the Full Fulfillment

DOM model (FF-DOM) in which all orders must be fully satisfied. In this case the revenue is fixed (all orders are fulfilled) and the goal is to minimize the total shipping cost. There are three variations on this model: (i) Unsplittable: an order of a customer has to be fulfilled by a single store. (ii) Partially splittable: each line of a customer has to be served by a single store. (iii) Fully splittable: a line of a customer can be served by any number of stores.

The second model we consider is the Partial Fulfillment (PF-DOM) model in which orders are not necessarily fully served. The goal here is to maximize the revenue from satisfied orders minus the shipping cost. We consider two variations on this model: (i) Unsplittable, and (ii) Partially splittable. Both variations are defined as in (FF-DOM).

1.2 Our Results

Our first contribution is the introduction of a formal mathematical framework for optimization problems that arise in the setting of DOM. We define several models and variants of these optimization problems, varying with respect to the objective function and given constraints. We first formulate the problems as linear programs and compute optimal fractional solutions. We then present several LP rounding algorithms for the (FF-DOM) and (PF-DOM) models. Our rounding works via a simple two-step scheme. First, using randomized rounding, we obtain a temporary infeasible solution. Second, we fix the (infeasible) solution at a low expected cost.

For the partially splittable and fully splittable minimization problems (subject to full fulfillment) we present a bi-criteria $(O(\log \delta), O(\log \delta))$ approximation algorithms, where both the cost and the violation factor are within a factor of $O(\log \delta)$. In addition, we explain how to achieve a $O(\log \delta)$ approximation for unsplittable FF-DOM.

In the maximization problems we first consider unsplittable PF-DOM for which we present for every $\epsilon \in (0, 1)$ a $(1 - \epsilon)$ approximation algorithm, under the natural assumption that store capacities are much larger than the demand of single orders. We normalize the demands to have size at most one and so the assumption becomes $s_{ik} \geq \Omega_\epsilon(\ln |K|)$ (or $s_{ik} = 0$) for all $i \in M, k \in K$. For the partially splittable PF-DOM problem we present an approximation hardness of $\frac{1}{|M|^{1-\epsilon}}$, for any $\epsilon > 0$ even with store-capacities taking values in $\{0, \infty\}$. In order to achieve a better result we conform to a more realistic scenario in which δ is constant. Under this assumption, we present for every $\epsilon \in (0, 1)$ a $(1-\epsilon)$ approximation algorithm, under the assumption that $s_{ik} \geq \Omega_\epsilon(\ln |K|)$ (or $s_{ik} = 0$) for all $i \in M, k \in K$.

We view the models defined in this paper, together with the proven results (upper and lower bounds), as a first step in understanding the DOM optimization setting. Obvious open research directions include online variations of the problem in which stock level changes and orders arrive over time.

1.3 Related Work

Packing problems have received much attention throughout the years. For example, a classic problem that received extensive attention is the Minimum Generalized Assignment Problem (Min-GAP), in which we are given a set of machines with size constraints, and a set of jobs with a different size and cost for placement in each machine. The goal is to assign all jobs to machines such that size constraints are not violated, and the overall cost is minimized. Shmoys et al. [8] presented an LP rounding algorithm yielding a solution with optimal cost, while violating the size constraints by at most a factor of two.

Solving Packing Integer Problem (PIP) was considered by Chekuri et al. [2]. Given a matrix $A \in [0,1]^{d \times n}, b \in [1,\infty)^d$, and $c \in [0,1]^n$, where $\max_j c_j = 1$, the goal in PIP is to find a vector $x \in \{0,1\}^n$ such that $c^\intercal x$ is maximized and $Ax \leq b$. This problem captures the world of multidimensional constraints. Chekuri et al. [2] gave an approximation hardness of $\frac{1}{d^{\frac{1}{B+1}-\epsilon}}$ for $\epsilon > 0$, where $B = \max_i b_i$. This proves a connection between the hardness of the problem and the ratio between the size of the bins to the size of the items (in our terminology, the stock level and the maximum number of items in a single line).

The order fulfillment problem has been considered in the OR literature, taking into account various aspects, such as reliability, on-line order arrivals, due-dates, etc. [1, 6]. These references, however, focus on the impact of incorporating the time dimension, using heuristic approaches. Xu et al. [9, 10] focus on heuristics that 're-optimize' to improve an online generated assignment. Our work considers a more general cost structure while restricting attention to offline deterministic setting. This is a common use-case in DOM solutions, where orders are allocated in large batches (see, e.g., [7]).

2 Minimizing Cost Subject to Full Fulfillment

Here we consider the FF-DOM model in which all orders are fulfilled. In particular, this means that we can focus on minimizing costs, as the revenue is a fixed constant. We consider three different submodels that differ in the way splitting the service of a single order between several stores is allowed. Specifically, we consider the unsplittable, partially splittable, and fully splittable variants.

We mention that the unsplittable variant (where each order is satisfied by a single store) can be solved exactly using the partial re-sampling algorithm presented by [5] for the Column-sparse packing problem. The solution violates the stock constraints by at most $O(\log \delta)$.

Theorem 1. *There is an algorithm for unsplittable FF-DOM returning an exact solution violating the stock constraints by at most a factor of* $O(\log \delta)$.

2.1 Partially Splittable

In this setting each line of an order is satisfied by a single store; however, given an order with demands for several different lines, we can assign different stores

to satisfy each one. This prevents us from bundling costs together into a single cost for each store-client pair.

Let y_{ij} indicate whether store i serves *some line* for customer j, and x_{ijk} indicate that line k in an order from customer j was satisfied by store i. The fixed cost for sending a package from store i to customer j is denoted by f_{ij}. In addition, there's an additional shipping cost, c_{ijk}, paid for including line k in the package. We get the following IP:

$$\min \sum_{i\in M, j\in N, k\in K} c_{ijk}x_{ijk} + \sum_{i\in M, j\in N} f_{ij}y_{ij}$$

subject to

$$\sum_{i\in M} x_{ijk} = 1, \qquad \forall j \in N, k \in K, \tag{1a}$$

$$\sum_{j\in N} d_{jk}x_{ijk} \le s_{ik}, \qquad \forall i \in M, k \in K, \tag{1b}$$

$$y_{ij} \ge x_{ijk}, \qquad \forall i \in M, j \in N, k \in K, \tag{1c}$$

$$y_{ij}, x_{ijk} \in \{0,1\}, \qquad \forall i \in M, j \in N, k \in K.$$

Eq. (1a) ensures every line of every client is fully satisfied, (1b) limits service provided by a store, and (1c) includes shipping cost from a store only when necessary.

Theorem 2. *The Partially splittable FF-DOM cannot be approximated within a factor of $\Omega(\log \delta)$ unless $P = NP$.*

The proof can be found in Appendix A.

Algorithm 2.1:

1. Solve the LP.
2. For each customer j and store i: randomly select $\theta_{ij} \sim \text{unif}(0,1)$.
3. Set store i to serve line k to customer j if $x_{ijk} \log \delta \ge \theta_{ij}$.
4. Define a GAP instance for each item k (a job is defined for each unsatisfied customer with size d_{jk} and cost equal to the sum of shipping and additional shipping costs). Round using a rounding algorithm for GAP.

Theorem 3. *Algorithm 2.1 provides a bi-criteria $(O(\log \delta), O(\log \delta))$ approximation for the partially splittable FF-DOM problem, approximating the overall cost and violating the capacities constraints by at most a factor of $O(\log \delta)$*

Proof. The rounding in the algorithm can be split into two phases, a randomized rounding phase and a correction phase. In the rounding phase we randomly select a threshold θ_{ij} for each store-customer pair. If the connection indicator x_{ijk} is larger than the threshold, we set store i to serve line k to customer j. If we denote the indicator whether customer j was not served line k by I_{jk} in the randomized rounding phase we get:

$$\Pr[I_{jk}] = \prod_{i\in M} (1 - x_{ijk}) \le \prod_{i\in M} \mathrm{e}^{-x_{ijk}} \le \mathrm{e}^{-1} \le \frac{1}{e}. \tag{2}$$

Since we also amplified the probability for a store to serve an order of a client by a factor of $O(\log \delta)$, we get that $\Pr[I_{js}] \leq \frac{1}{\delta}$. Thus we have an integral bi-criteria $(O(\log \delta), O(\log \delta))$ approximation (incurred by the variables multiplication), but all demands are satisfied with low probability.

Building on this partial solution we run the correction phase. For each item type k define an instance of GAP. A machine is defined for each store, and a job is defined for each unsatisfied customer j with size equal to d_{jk}. The cost of assigning job j to store i is the sum of shipping cost f_{ij} and additional shipping costs c_{ijk}. The size constraint of machine i equals the stock of item k that store i has. We define a fractional solution for the constructed instance of GAP. For each job j the fractional assignment of j to machine i is x_{ijk}.

We notice that for an item k the expected cost of the fractional solution is

$$\frac{\sum_{i,j} c_{ijk} x_{ijk} + \sum_{i,j} f_{ij} y_{ij}}{\delta}, \tag{3}$$

since a customer's line is not satisfied with probability of at most $\frac{1}{\delta}$. The LP rounding algorithm for GAP returns an integral solution with cost equal to the fractional cost, but the size constraints are violated by a factor of two. Let us first consider the cost. If we sum over the solutions for the GAP instances created for all items we get an expected additive cost of

$$\frac{\sum_{i,j,k} c_{ijk} x_{ijk} + \delta \cdot \sum_{i,j} f_{ij} y_{ij}}{\delta} = \frac{\sum_{i,j,k} c_{ijk} x_{ijk}}{\delta} + \sum_{i,j} f_{ij} y_{ij} \tag{4}$$

which is at most the optimal fractional cost.

We note that even though we fix the partial solution by solving a GAP instance for each item separately, there is no dependence between the items' stock. Thus rounding each instance of GAP leads to an expected additive increase to the capacities violation factor of a single item of $\frac{1}{\delta}$. Summing over the costs incurred by both phases we get a bi-criteria $(O(\log \delta), O(\log \delta))$ approximation in which all demands are satsified.

2.2 Fully Splittable

In this setting several stores can supply the same item to a single customer. The program created using notation as above, except that now x_{ijk} is an integer variable. It is easy to see that the program is not linear due to the new constraint $y_{ij} \geq \min\{x_{ijk}, 1\}$. We formulate instead a configuration IP. For a store i we say that order $q \in i$ if it can serve it. We also mark by q_k the number of units of item k in order q. Now we consider for each client the possible orders it may have from different stores. Variable x_{ijq} indicates whether store i serves customer j order q. We also mark by C_{ijq} the total cost for store i to serve customer j order q. Even though it has an exponential number of variables, the resulting LP can be solved through its dual using a separation oracle. The solution can be used to construct a solution to the original LP.

$$\min \sum_{i\in M, j\in N, q\in i} C_{ijq} x_{ijq}$$

subject to

$$\sum_{i\in M, q\in i} x_{ijq} \cdot q_k = d_{jk}, \quad \forall j \in N, k \in K, \qquad (5a)$$

$$\sum_{j\in N, q\in i} x_{ijq} \cdot q_k \leq s_{ik}, \quad \forall i \in M, k \in K, \qquad (5b)$$

$$x_{ijq} \geq 0, \qquad \forall i \in M, j \in N, q \in i.$$

Eq. (5a) ensures every order is fulfilled, and (5b) limits service provided by a store.

Theorem 4. *The fully splittable FF-DOM problem cannot be approximated within a factor of* $\Omega(\log \delta)$ *unless* $P = NP$.

The proof of Theorem 2 proves this theorem as well, as it was proven for a partially splittable FF-DOM isntance with units demands.

Algorithm 2.2:

1. Solve the LP and fix assignment of any integral parts.
2. For each customer j and store i: randomly select $\theta_{ij} \sim unif(0, 1)$.
3. Set store i to serve line k to customer j if $x_{ijk} log(\delta) \geq \theta_{ij}$.
4. Define an instance of SGAP for each item k (a job is defined for each unsatisfied customer with size d_{jk}, assignment cost of f_{ij}, and computation cost c_{ij}. Round using a rounding algorithm for SGAP (see Appendix C).

Theorem 5. *Algorithm 2.2 provides a bi-criteria* $(O(\log \delta), O(\log \delta))$ *approximation for the fully splittable FF-DOM problem, approximating the overall cost and violating the capacities constraints by at most a factor of* $O(\log \delta)$.

The proof follows the same arguments as the proof of Theorem 3.

3 Maximizing Revenue with Partial Fulfillment

In the partial fulfillment variant we do not require every order to be fulfilled. Instead the goal is to maximize the overall revenue gained from satisfied orders, after subtracting the shipping costs. We consider two cases of the problem. First, the unsplittable PF-DOM in which all lines of an order must be satisfied by a single store. In the second case we allow different lines to be satisfied by different stores. In addition, the revenue gained is accumulated per lines satisfied, and we do not need to serve all lines in an order to earn revenue. However, as pointed out in Remark 1, our techniques also apply to the case when all lines of an order need to be satisfied.

3.1 Unsplittable

In this setting, one receives the payment of an order if and only if the whole order is fulfilled by a single store. In particular, this means that we can incorporate all payments and costs of a single order into a single value which represents the revenue earned from fulfilling the order. We denote by p_{ij} the revenue earned if store $i \in M$ served the order of client $j \in N$. Let y_{ij} indicate whether store i serves client j. Then the following IP models the problem:

$$\begin{aligned} &\max \sum_{i \in M, j \in N} p_{ij} y_{ij} \\ \text{subject to} \\ &\sum_{i \in M} y_{ij} \leq 1, && \forall j \in N, && (6a) \\ &\sum_{j \in N} d_{jk} y_{ij} \leq s_{ik}, && \forall i \in M, k \in K, && (6b) \\ &y_{ij} \in \{0,1\}, && \forall i \in M, j \in N. \end{aligned}$$

Eq. (6a) ensures every order is sent from at most one store, and (6b) limits service provided by a store.

We observe that the unsplittable PF-DOM problem generalizes the q-dimensional knapsack problem, as an instance of unsplittable PF-DOM with a single store, and $q = |K|$. For normalization, we consider the scaled problem in which for each k, $\max_j d_{jk} = 1$ and let $B = \min\{s_{ik} : s_{ik} > 0\}$. If $q = |K|$ is not a constant, it was shown by Chekuri et al. [2] that the q-dimensional knapsack problem is hard to approximate within a factor of $q^{\frac{1}{B+1}-\epsilon}$ for some $\epsilon > 0$. Thus to get good algorithmic guarantees, we need to make additional assumptions on the problem instance. While the number $|K|$ of different items may well be very large, a natural assumption is to assume that B is much larger than the demand of any single order, i.e., $B \gg \max_j d_{jk} = 1$. This is similar to the small bid assumptions often made in ad-allocation. More specifically, we analyze the following algorithm with this assumption.

Algorithm 3.1 with parameter $\epsilon \in (0,1)$:

1. Solve the LP to obtain optimal solution y.
2. For each order $j \in N$, assign it to store i with probability $(1-\epsilon)y_{ij}$ (independent of other orders).
3. Throw away all orders assigned to stores that cannot satisfy all orders assigned to them.

Theorem 6. *For every $\epsilon \in (0,1)$, Algorithm 3.1 returns (in expectation) a $(1-\epsilon)$-approximate solution for the unsplittable PF-DOM problem, assuming that $s_{ik} \geq 3\epsilon^{-2}(1-\epsilon)^{-1}\ln(|K|/\epsilon)$ (or $s_{ik} = 0$) for all $i \in M$ and $k \in K$.*

The proof follows the same arguments as the proof of Theorem 8.

3.2 Partially splittable

In the partially splittable variant an order can be satisfied by several stores; however, each line is satisfied by a single store. The presence of orders with many lines makes this variant very hard to approximate.

Theorem 7. *Unless $ZPP = NP$, for any $\epsilon > 0$ the partially splittable problem cannot be approximated within a factor of $\frac{1}{|M|^{1-\epsilon}}$, where M is the set of stores. Moreover, this hardness holds for instances with a single order and store capacities taking values in $\{0, \infty\}$.*

The proof can be found in Appendix B. The above hardness shows that a general algorithm with no assumptions on the input cannot give any reasonable guarantees for the considered problem. However, based on the data that inspired the problem we noticed that in realistic scenarios, the instances have several nice structural properties. First, either a store has a large stock compared to the client requirements, or it does not have any stock (as in the previous section). This assumption alone is not sufficient here (as the above hardness says). We therefore also make a second assumption motivated by the observation that although there are many different items, in practice, each order contains only a small (constant) number of different lines. This leads us to consider instances of the partially splittable PF-DOM problem where (i) $B = \min\{s_{ik} : s_{ik} > 0\} \gg 1$ is large[9] and (ii) the maximum number of lines in an order, denoted by δ, is a constant.

In order to solve the problem we define a new IP. Let F_j be the set of all possible assignment of order j to the stores. We say that in an assignment $\sigma \in F_j$ line k is served by store i if $\sigma(i,k) = 1$; otherwise $\sigma(i,k) = 0$. We define for order j an indicator $y_{j\sigma}$, where $\sigma \in F_j$. The revenue of assignment σ is denoted by $p_{j\sigma}$. We get the following IP for the problem:

$$\max \sum_{j \in N, \sigma \in F_j} p_{j\sigma} y_{j\sigma}$$

subject to

$$\sum_{\sigma \in F_j} y_{j\sigma} \leq 1, \qquad \forall j \in N, \tag{7a}$$

$$\sum_{j \in N, \sigma \in F_j} d_{jk} \sigma(i,k) y_{j\sigma} \leq s_{ik}, \qquad \forall i \in M, k \in K, \tag{7b}$$

$$y_{j\sigma} \in \{0,1\}, \ \forall j \in N, \sigma \in F_j.$$

Eq. (7a) ensures each order is assigned once, and (7b) limits service provided by a store.

We remark that the linear programming relaxation of the above IP can be solved in polynomial time assuming the number of lines δ per order is a constant. This is true since the number of constraints is polynomial in $|N|, |M|, |K|$, and the number of variables is polynomial in $|N|$ and the size of F_j's, which is upper bounded by $|M|^\delta$. We now give an algorithm for rounding a fractional solution.

[9] As in the previous section, we normalize demands so that, $\forall\ k \in K$, $\max_j d_{jk} = 1$.

Algorithm 3.2 with parameter $\epsilon \in (0,1)$:

1. Solve the linear relaxation of the IP to obtain optimal solution y.
2. For each order j: with probability $(1-\epsilon)y_{j\sigma}$ assign j as in σ.
3. Throw away all lines assigned to stores with violated stock-capacity constraints.

Theorem 8. *Algorithm 3.2 runs in time polynomial in $|N|, |M|^{\delta}, |K|$ and returns (in expectation) a $(1-\epsilon)$-approximate solution for the partially splittable PF-DOM problem under the assumption that $s_{ik} \geq 3\epsilon^{-2}(1-\epsilon)^{-1}\ln(|K|/\epsilon)$ (or $s_{ik} = 0$) for all $i \in M$ and $k \in K$.*

Proof. The running time of the algorithm is dominated by solving the linear programming relaxation. The running time thus follows from the discussion after the definition of the IP. Let $Y_{j\sigma}$ be a random variable indicating whether the assignment of order j to the stores is σ. Thus, $\Pr[Y_{j\sigma}] = (1-\epsilon)y_{i\sigma}$. We would like to bound the probability that store i has a shortage in stock of item k, given that order j' was partially assigned to it in assignment σ'. Using standard Chernoff bound we get that:

$$\Pr\left[\sum_{j,\sigma} d_{jk}\sigma(i,k)y_{j\sigma} > s_{ik} | Y_{j'\sigma'}\right] \leq \epsilon/|K|\,.$$

By union bounding over all $|K|$ items we get that the probability that store i is thrown, given that order j' is partially assigned to it in σ' is at most ϵ.

It is important to notice that if a store is thrown away, and an order is partially assigned to it, only lines assigned to this store are thrown away as well. Building on this observation and our previous calculations we may sum over all orders to get that the expected revenue is at least $(1-\epsilon)\sum_{j\in N,\sigma\in F_j} p_{j\sigma}y_{j\sigma}$, which concludes the proof.

Remark 1. The above techniques also imply that we can get an algorithm that finds $(1-\epsilon)$-approximate solution even in the case when an order is either fully satisfied or not sent at all. The only changes are (i) Step 3 of Algorithm 3.2 now becomes to throw away whole orders assigned to stores with violated stock-capacity constraints, and (ii) to make the slightly stronger assumption $s_{ik} \geq 3\epsilon^{-2}(1-\epsilon)^{-1}\ln(|K|\cdot\delta/\epsilon)$ (or $s_{ik} = 0$).

A Partially Splittable FF-DOM Hardness

Proof. We recall that there is no algorithm for the set cover problem, with approximation ratio better than $\log(|K|)$, where $|K|$ is the number of elements, unless $P = NP$ [3]. We will show a reduction from set cover to an instance of our problem with $|K|$ items, which concludes that an approximation ratio better than $\log(|K|)$ is not achievable.

We are given an instance of set cover in which there are $|K|$ elements and a collection $S = S_1, ..., S_{|M|}$ of subsets of the elements. We are also given a cost c_{S_i} for each subset S_i. We now define an instance of the partially splittable FF-DOM problem where we have a single customer, $|M|$ stores, one for each subset S_i, and $|K|$ items, one for each element. The shipping cost of store i is c_{S_i}, the cost of set i. For each store i, the stock of items corresponding to elements in S_i is one, and for other items it is zero. The additional shipping cost is set to zero. The customer demands one of each item. One can notice that in the constructed instance $\delta = |K|$, and that a solution to the partially splittable FF-DOM instance provides a solution to the set cover problem (select the sets corresponding to the chosen stores), which concludes the proof.

B Partially Splittable FF-DOM Hardness

Proof. We will prove this hardness result using a reduction from the Maximum Independent Set (MIS) problem. In the MIS problem we are given a graph $G = (V, E)$ and the goal is to find a maximum cardinality subset $F \subset V$ such that for each pair of nodes $u, v \in F, (u, v) \notin E$.

Given an instance of MIS on an input graph $G = (V, E)$ we define an instance of the partially splittable PF-DOM problem. In this instance we define an item for each edge, i.e., for each edge (u, v), there is an item k_{uv}. We also define a single client with a single order with a line for each of the $|E|$ items, for each the demand is a single unit. A store is defined for each vertex. Store u only has (infinitely many) items corresponding to adjacent edges in its stock. The revenue earned for serving an item is 1 (at all stores), and the packing cost in store u is $\delta(u) - 1$ (where $\delta(u)$ is the number of neighbors of u).

Given a solution to the partially splittable PF-DOM instance we construct a solution to the MIS instance by selecting all vertices corresponding to the stores chosen to serve the order. A revenue is gained from a store if and only if it serves all the items it can serve, i.e., all items corresponding to the adjacent edges of the store's vertex. Since only a single unit of each item is required, for each item k_{uv} a single store is chosen to serve and pay for the packing. Thus, for each edge in the graph only one of its vertices is chosen in the constructed solution. Moreover, by the same arguments, one can see that any solution to MIS defines a solution of the same value to the partially splittable PF-DOM instance.

This proves that given an α-approximation algorithm for the partially splittable problem, an α-approximation algorithm can be constructed for MIS. Since for any $\epsilon > 0$ MIS cannot be approximated within a factor of $\frac{1}{|V|^{1-\epsilon}}$ assuming

$ZPP \neq NP$ [4], the partially splittable PF-DOM problem cannot be approximated within a factor of $\frac{1}{|M|^{1-\epsilon}}$.

C Splittable Generalized Assignment Problem

There is a set N of jobs and a set M of machines. Machine i is associated with a size constraint W_i. Each job j has an assignment size w_j and two different costs, a computation cost c_{ij} paid for each unit of size job j occupies on machine i, and an assignment cost f_{ij} paid if job j is assigned to machine i. Job j is said to be satisfied if it is assigned to several machines in which it occupies a total size of w_j. The objective is to satisfy all jobs such that the size constraints are not violated, and the total assignment cost is minimized. Given a fractional solution to the problem it can be rounded by fixing the itnegral parts of the solution, and rounding the remainder using the rounding algorithm for GAP [8].

Theorem 9. *Given a fractional solution to the SGAP LP, there exists an algorithm that returns a feasible solution to the problem with the same cost.*

References

1. Chan, F.T., Chung, S.H., Choy, K.L.: Optimization of order fulfillment in distribution network problems. Journal of Intelligent Manufacturing **17**(3), 307–319 (2006)
2. Chekuri, C., Khanna, S.: On multidimensional packing problems. SIAM journal on computing **33**(4), 837–851 (2004)
3. Dinur, I., Steurer, D.: Analytical approach to parallel repetition. In: Proceedings of the forty-sixth annual ACM symposium on Theory of computing. pp. 624–633. ACM (2014)
4. Hå stad, J.: Clique is hard to approximate within $n^{1-\epsilon}$. Acta Math. **182**(1), 105–142 (1999)
5. Harris, D.G., Srinivasan, A.: The moser-tardos framework with partial resampling. In: Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on. pp. 469–478. IEEE (2013)
6. Jasin, S., Sinha, A.: An lp-based correlated rounding scheme for multi-item ecommerce order fulfillment. Operations Research **63**(6), 1336–1351 (2015)
7. Purushotham, A.: Distributed order management (DOM). Tech. rep. (2018), https://docs.microsoft.com/en-us/dynamics365/unified-operations/retail/dom
8. Shmoys, D.B., Tardos, É.: Scheduling unrelated machines with costs. In: SODA. vol. 93, pp. 448–454 (1993)
9. Xu, P.J.: Order Fulfillment in Online Retailing : What Goes Where. Thesis, Massachusetts Institute of Technology (2005)
10. Xu, P.J., Allgor, R., Graves, S.C.: Benefits of Reevaluating Real-Time Order Fulfillment Decisions. Manufacturing & Service Operations Management **11**(2), 340–355 (Dec 2008). https://doi.org/10.1287/msom.1080.0222