

The Benefits of Re-Evaluating Real-Time Fulfillment Decisions

Ping Josephine Xu, Russell Allgor, and Stephen Graves
MIT Amazon.com

Abstract—At the time of a customer order, the e-tailer assigns the order to one or more of its order fulfillment centers, and/or to drop shippers, so as to minimize procurement and transportation costs, based on the available current information. However this assignment is necessarily myopic as it cannot account for all future events, such as subsequent customer orders or inventory replenishments. We examine the potential benefits from periodically re-evaluating these real-time order-assignment decisions. We construct near-optimal heuristics for the re-assignment for a large set of customer orders with the objective to minimize the total number of shipments. We investigate how best to implement these heuristics for a rolling horizon, and discuss the effect of demand correlation, customer order size, and the number of customer orders on the nature of the heuristics. Finally, we present potential saving opportunities by testing the heuristics on sets of order data from a major e-tailer.

Index Terms—Fulfillment, Assignment, Network Design

I. INTRODUCTION

E-tailers pride themselves in having a universal selection of products and in providing a very customer-friendly shopping experience. In this increasingly crowded online marketplace with few barriers to entry, there is no doubt that the success or dominance of an e-tailer depends on an efficient customer fulfillment process. Because of the scale, complexity of the electronic fulfillment systems and the overwhelming amount of available data, making sound fulfillment decisions requires both good operating tactics and sophisticated tools that are based on simple ideas. We attempt to provide such tools and insights in an e-tailing setting by investigating the tactical decision of assigning each customer order to warehouse(s), so the e-tailer can ship the items to the customer.

When a customer places an order on an e-tailer's website, the e-tailer, in real time, searches for available fulfillment options from its order fulfillment centers (warehouses) or drop-shippers. The e-tailer assigns the order to one or more warehouses virtually, mainly based on the transportation cost of shipping the order from the warehouse(s) to the customer location and on the current warehouse inventory availability. Depending on the inventory availability and customer preferences, the e-tailer then quotes a promise-to-ship date to the

customer. The promise-to-ship date is the date by which the e-tailer promises to ship the order from the warehouse(s). After the e-tailer assigns the order, the order enters the picking queue at the warehouse. The order might wait six to eighteen hours before the items in the the order are picked and assembled into a shipment that is then given to a third party carrier to delivers the package(s) to the customer location.

We present Example 1 to illustrate the real-time assignment decision. Suppose a customer located at Chicago orders one

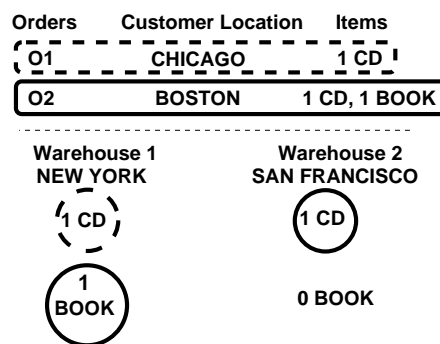


Fig. 1. Example 1 - Real-time assignments result in three shipments.

CD, as indicated in the dash box in Figure 1. In real time, the e-tailer searches for its available inventory in all of its warehouses: Warehouse 1 near New York and Warehouse 2 by San Francisco. Both warehouses have one unit of this CD available, and the e-tailer will make the assignment to minimize transportation costs. In this case, it is cheaper to ship the CD from New York, so the e-tailer assigns the CD inventory in Warehouse 1 to this order. Three seconds later, a customer from Boston orders the same CD and a book. Suppose the book is only available from Warehouse 1. The only possible assignment for the e-tailer now, without placing an inventory replenishment order, is to fulfill the second order with two shipments: Warehouse 2 can ship the CD and Warehouse 1 can ship the book to the second customer. We have a total of three shipments for the two orders.

In the transportation cost for shipping a package, the fixed cost component is very significant. We display the current Ground Commercial rates within the US continent from UPS in Figure 2. We display both the rates for shipping from Zone 1 to Zone 2 (the closest zone) and to Zone 8 (the farthest zone). We see that in both instances the shipping cost consists of a fixed cost of about \$5 per shipment, plus a variable cost that is linear in the weight of the package. Furthermore, for small shipments the fixed cost represents the

November 19, 2004. This work was supported in part by the MIT Leaders for Manufacturing Program and the Singapore-MIT Alliance.

R. Allgor is with Amazon.com at Seattle, WA 98108 USA (email: rallgor@amazon.com).

S. C. Graves is with the Sloan School of Management and the Engineering System Division at MIT, Cambridge MA 02139 USA (email: sgraves@mit.edu).

P. J. Xu is with the Operations Research Center at MIT, Cambridge MA 02139 USA (email: pingx@mit.edu).

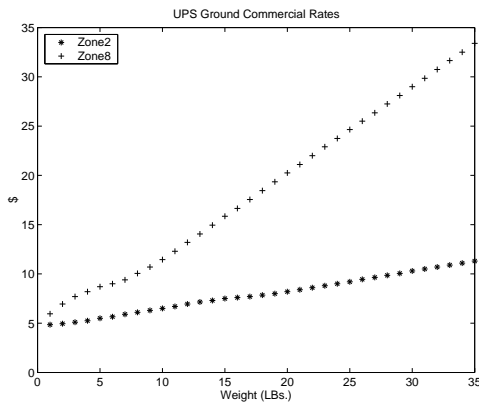


Fig. 2. UPS Ground commercial rates within the US continent

majority of the shipping costs. As a consequence, reducing the number of shipments is a very good proxy for minimizing the transportation costs in the e-tailing setting. For example, consider an order that weights about eight pounds. It is cheaper to ship a single package of eight pounds to Zone 8 than to ship two four-pound packages to Zone 2. The difference is even more pronounced at smaller weights. For example, shipping a two-pound package and a six-pound package to Zone 2 costs \$10.60, while shipping one eight-pound package to Zone 8 costs \$10.05. For items that can typically be fit into the few standard packages, their weight is at most a few pounds, e.g., books, CDs, DVDs. Therefore, the e-tailer minimizes its transportation costs by minimizing the number of shipments.

If we consider only the two orders in Example 1, we can reduce the number of shipments to two as illustrated in Figure 2 by changing the order-warehouse assignments. We assign the first customer order to Warehouse 2 and the second to Warehouse 1.

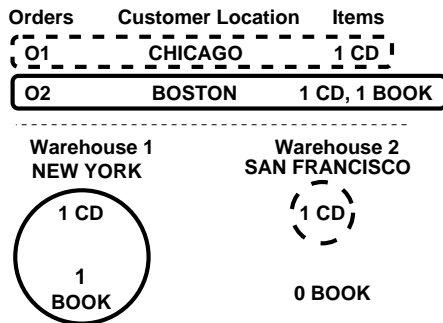


Fig. 3. Example 1 - Re-evaluating real-time assignments reduce number of shipments to two.

Example 1 is a bit extreme and the modification to the initial assignments is very straightforward. To appreciate the difficulty and the subtlety of the problem, we discuss Example 2 next. In Figure 4, we have four customer orders, labeled as O1, O2, O3, O4, and three warehouses, labeled as W1, W2, W3. The warehouses carry five SKU's, with the names CD, book, toy, camera, and DVD. . The first customer order is for the CD, and the e-tailer assigns it to W2, possibly because the first customer is nearest to W2 or W2 is the only warehouse

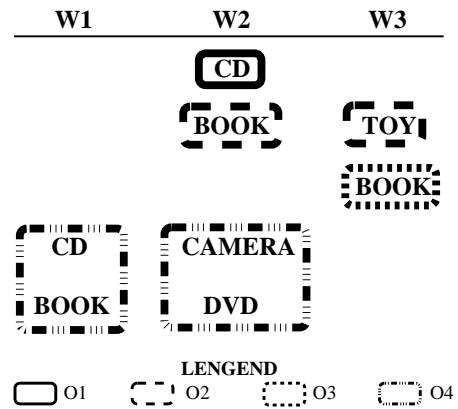


Fig. 4. Example 2 - Read-time assignments result in 6 shipments.

with CDs in stock. The second order consists of the book and the toy. The e-tailer assigns the book to warehouse 2 and the toy to W3, because the book is not yet in inventory at W3. Suppose an inventory replenishment of books is received at W3 before customer O3 arrives. The e-tailer then assigns customer O3 to W3. Finally, there are two shipments for customer O4: the CD and book from W1 and the camera and DVD from W2. Thus, there are six shipments for the four orders, and it may not be immediately apparent whether or not we can shuffle the assignments to reduce the number of shipments. However, in Figure 5, we show that we can reduce

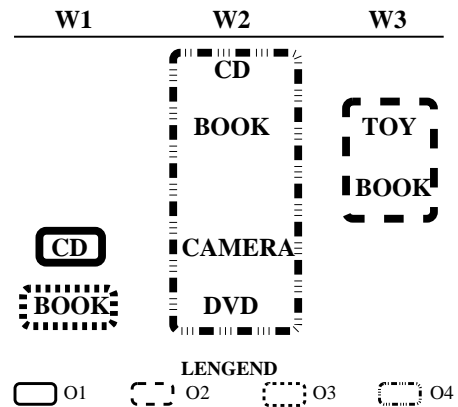


Fig. 5. Example 2 - Re-evaluating real-time assignments reduce number of shipments from 6 to 4 .

the number of shipments from six to four, which is clearly the best we can do.

We show with examples that the real-time decision is necessarily myopic because the e-tailer does not anticipate any future customer orders or inventory replenishment. The real-time assignment is myopic in practice because the e-tailer wants to reserve the inventory for the customer, then inform the customers with confidence that inventory is available and that the order can be fulfilled by the promise-to-ship date. The real-time assignment is myopic also because of two main challenges. An e-tailer might have a half a million SKUs in its warehouses; it is very difficult to develop accurate warehouse-level demand forecasts for orders that consist of from one to

twenty distinct SKUs. The second challenge is the unreliable future inventory replenishment information. We conjecture that we can reduce the total transportation cost of shipping orders from warehouses by re-evaluating the real-time assignment decisions, subject to the constraint that there is no violation of the promise-to-sip date commitment for any customer order.

This shuffling of assignments is also practically feasible. Even when all items in an order are available at the warehouse, the order may wait 8 to 16 hours until the order is release to be picked and sent for shipping. If one or more of the items in the order is not available, then the rest of the order is reserved and waits until the missing items arrive. By re-evaluating the real-time decision, the e-tailer can also afford more decision making time. We pose a problem to re-evaluate the real-time decisions. We consider the queue of *not-yet-picked* customers orders and their real-time warehouse assignments, and we re-evaluate these real-time decisions to see if we can reduce the shipping cost without violating the promise-to-ship date commitments for these orders. The not-yet-picked orders are the orders that have not yet been released to be picked at each warehouse. We take inventory availability and the real-time quoted promise-to-ship dates as given.

We will show in later sections that this snapshot optimization problem is difficulty theoretically (belong to the NP-hard complexity class) and in practice. For now, exact methods cannot solve realistically dimensioned cases. In the e-tailing setting, the problem size is also especially large. For an off-season snapshot at a large e-tailer, there are 1 million orders with 2 to 3 million units waiting to be picked. There are up to 10 warehouses. The total number of SKUs in those orders ranges from 500,000 to 800,000. In the peak season, the number of orders can reach three or five times the number in the off-season.

Therefore, we develop efficient and easy to implement sub-optimal heuristics to solve the re-evaluation problem. Given the real-time assignment decisions, we take the natural path to construct an improvement heuristic that starts with a feasible solution and iteratively finds better solutions. We also derive bounds to determine the sub-optimality of our heuristics.

In the following sections, we discuss the problem formulation and our heuristic solution approach. We also summarize some computational experiments on sets of real data from a global e-tailer.

II. PROBLEM FORMULATION

We present two formulations of the re-evaluation problem, where one is based on the set partitioning problem, and another is a network design formulation. Both are formulated as large scale integer or mixed-integer problems. Both formulation shed light on the underlying structure and difficulty of the problem.

A. Formulation 1

For this set-partition based formulation, we first examine the real-time assignment decision for an order. For now, we assume that we have enough inventory across warehouses in the network to satisfy the order. Without this assumption, we

have a set-packing problem, where some of the items in an order may not be assigned to any warehouse. We start with some notations.

k	index for warehouses
i	index for SKU's, and $ I = m$.
N	$= \{1, \dots, n\}$, a collection of all possible subsets of the order, i.e., $C_l, l \in N$, is the l^{th} subset of the order
A	a m by n matrix such that a_{il} is the number of units of item i included in subset C_l
d_i	units of SKU i in the order
e_n	a n by 1 vector of 1's
x_l	$= 1$ if subset C_l is shipped
y_{lk}	$= 1$ if subset C_l is shipped out of warehouse k
s_{ik}	inventory units of SKU i available at warehouse k

We denote the following formulation of assigning an order to warehouses as \mathcal{P} .

$$\begin{aligned} \min \quad & \sum_{\forall l, k} y_{lk} \\ \text{s.t.} \quad & \sum_{\forall l} a_{il} x_l = d_i, \quad \forall i \end{aligned} \quad (1)$$

$$\sum_k y_{lk} = x_l, \quad \forall l \quad (2)$$

$$\begin{aligned} \sum_l a_{il} y_{lk} &\leq s_{ik}, \quad \forall i, k \\ x_l, y_{lk} &\in \{0, 1\}, \quad \forall l, k \end{aligned} \quad (3)$$

Constraint (1) guarantees that the number of units for each SKU in the order is shipped. This implies that all shipped subsets are disjoint and their union covers the entire order set. Constraint (2) guarantees subset C_l to be shipped from only one warehouse, if C_l is shipped, and zero warehouse if not shipped. Constraint (3) is a supply constraint: the amount of SKU i shipped from warehouse k cannot exceed the supply of SKU i in warehouse k .

Suppose we substitute index r for (l, k) , and restrict each SKU to have at most one unit in the order ($d_i = 1$) and allow supply to be infinite. This special case of problem \mathcal{P} is a set partitioning problem:

$$\begin{aligned} \min \quad & \sum_{\forall r} y_r \\ \text{s.t.} \quad & Ay = e_n \\ & y_r \in \{0, 1\} \end{aligned}$$

which is NP-hard in complexity. Therefore, the real-time assignment problem of an order is NP-hard by restriction.

Now we examine all orders in the not-yet-picked queue. We modify the previous notations and introduce new notations.

j	index for customer orders, J is the order set.
I	set of unique SKU's in the order set, $ I = v$.
m_j	number of SKU's in order j
N_j	$= \{1, \dots, n_j\}$, a subset collection of order j
N	$= \{N_1, \dots, N_j, \dots\}$

B_j a v by n_j matrix s. t. b_{il} is the units of SKU i in order j included in its l^{th} subset

Similarly, we define sub-matrix $A_j, [m_j, n_j]$, for the j^{th} order. Let $A, [m, n]$, be

$$A = \begin{bmatrix} A_1 & & & & \\ & A_2 & & & \\ & & \dots & & \\ & & & A_j & \\ & & & & \dots \end{bmatrix},$$

where $m = \sum_{j \in J} m_j$ and $n = \sum_{j \in J} n_j$ is the number of subsets for all orders. We also define matrix $B, [v, n]$, to be

$$B = [B_1 \quad B_2 \quad \dots \quad B_j \quad \dots].$$

Here problem \mathcal{P} still applies, but with some modification. We call the re-evaluation problem as \mathcal{Q} :

$$\min \sum_{\forall l, k} y_{lk} \quad (4)$$

$$\text{s. t. } Ax = d \quad (4)$$

$$x = Y e_K \quad (5)$$

$$BY \leq s \quad (6)$$

$$x_l, y_{lk} \in \{0, 1\}, \quad \forall l, k.$$

Clearly, problem \mathcal{P} is special case of problem \mathcal{Q} . Problem \mathcal{Q} is also NP-hard. The total number of binary decision variable is $n + nK$. The number of type (1) constraint is m , the number of type (2) constraint is n , and the number of type (3) constraint is vK . Notice n could be exponential in the input data. In problem \mathcal{Q} , we have an exponential number of binary variable and constraints.

B. Formulation 2

The re-evaluation problem can also be formulated as a network design problem, specially, a fixed-charge multi-commodity flow problem [2]. In addition to the previously defined notation, we redefine the decision variables.

x_{jk}^i units of SKU i shipped from warehouse k to customer j
 y_{jk} indicator of a shipment from k to j

We also denote set K_i to be the set of warehouses that carry SKU i inventory, J_i to be the set of customer orders that contain nonzero units of SKU i .

We denote the following formulation as \mathcal{MIP} .

$$\min \sum_{j, k} y_{jk} \quad (7)$$

$$\text{s. t. } \sum_{j \in J_i} x_{jk}^i = s_k^i, \quad \forall i \in I, k \in K_i \quad (7)$$

$$\sum_{k \in K(i)} x_{jk}^i = d_j^i, \quad \forall i \in I, j \in J_i \quad (8)$$

$$0 \leq x_{jk}^i \leq d_j^i y_{jk}, \quad \forall i \in I, j \in J_i, k \in K_i \quad (9)$$

$$y_{jk} \in \{0, 1\}, \quad \forall j, k \quad (10)$$

Notice that a commodity is a SKU. Variable x is a continuous variable here because for any given choice of y , we can

decompose the problem into a transportation problem by SKU, and there exists an optimal integer solution in transportation problems. Constraints (7) assure that the amount of each SKU shipped from each warehouse does not exceed the supply. Constraints (8) assure that the demand is met for each SKU in each order. Problem \mathcal{MIP} has JK binary variables and IJK continuous variables. It has $IK + IJ + IJK$ number of constraints. This formulation has linear number of constraints and variables in the input problem data. This fixed-charge multi-commodity flow problem is NP-hard in complexity and currently intractable empirically [3].

By examining the two formulations, we show that we need efficient and easy to implement heuristics to solve the snapshot problem.

C. Literature Review

There are two clusters of literature that are most relevant to our problem. The first is the literature on network design problems. The second is the literature on local search algorithms, a wide class of improvement algorithms.

The literature on network design problems is directly related to the second formulation of our problem. Most of the literature is on the basic fixed-charge design model. Different from our problem, the basic fixed-charge design model has a single set of source and sink for each commodity. Magnanti and Wong [5] have shown that the basic model is very flexible and contains a number of well known network optimization problems as special cases. Even many of the special cases (e.g., the uncapacitated plant location problem) are known to be difficult to solve, so is the general fixed-charge design model. In addition to the theoretical arguments, substantial empirical evidence also confirms the difficulty of the problem on large-scale instances: [4], [7], [3]. Judging by the size of the instance solved in the current literature, none are close to the scale of our problem.

There is also a rich group of literature on local search or neighborhood search. This set of literature is the inspiration of our proposed heuristics. Ahuja, Ergun, Orlin, and Punnen provide a comprehensive survey on very large-scale neighborhood search techniques [1]. Talluri [6] considers a fleet assignment problem, which can be modelled as an integer multicommodity flow problem subject to side constraints where each commodity refers to a fleet type. He considers a given solution as restricted to two fleet types only, and looks for improvements that can be obtained by swapping a number of flights between the two fleet types.

III. COMPLEX SYSTEM PROPERTIES

In solving the problem, we understand that specially tailored heuristics are more likely to out perform any general heuristics. To find any solution tailored to the problem structure, we must examine the problem data carefully. In this section, we summarize the important characteristics of the customer orders and the real-time assignments.

To facilitate the presentation, we introduce the following definitions.

- A *single order* is a customer order that consists of exactly one unit of one SKU.
- A *multi order* is a customer order that consists of more than one SKU or multiple units of one SKU.
- A *split order* is a customer order split over warehouses in the real-time assignment, i.e., orders with more than one shipment.
- A *single shipment* is a one-unit shipment of a split order.
- A *double shipment* is a two-unit shipment of a split order.

We examine a number of snapshot data sets in the off season from a large e-tailer. There are close to 1 million orders with 2 to 3 million units in the not-yet-picked queue for the snapshot data. Typically, 30% to 40% of the orders are multi orders. The size of multi orders tends to follow a geometric distribution, with the average size being around 3 to 4 units in each multi order.

The real-time assignments splits about 15% of the multi orders. The number of shipments in each split order is two or three shipments with few exceptions. There is at least one single shipment in more than 80% of the split orders. Over 90% of the split orders have at least one single shipment or one double shipment.

To investigate whether the problem can be decomposed into a number of smaller problems, we examined the connectivity of the order-SKU graph constructed for the snapshot of the not-yet-picked queue. There is one node for each SKU and we connect two SKU nodes when there exists an order that includes both SKUs. We found that there exists one very large component in the graph, containing the majority of the SKU's. Furthermore, any removal of small subsets of SKU's does not change the connectivity of the graph. Therefore, we do not see a clear way to decompose the problem by considering a limited number of orders or SKUs.

IV. HEURISTIC APPROACH

In solving the optimization problem, we start with a feasible solution, i.e., the real-time assignments. It seems natural to focus on an improvement algorithm, by which we iteratively create better solutions. The focus on improvement algorithms is also driven by practical concerns. Improvement algorithms generate a feasible solution at every iteration. After each iteration, we can implement the recommended changes to the current (incumbent) assignments to get an improved order assignment. This facilitates greatly the implementation of this solution approach, since we always have a feasible solution, even if there were a sudden termination of the algorithm.

One key idea for our heuristic is to consider how to use the single orders to fix the split orders. The motivation for this is twofold. First, single orders always entail a single shipment but are very flexible in their assignment. Second, the vast majority of split orders include a single shipment. By re-assigning a single order from warehouse A to warehouse B, we free up a unit of inventory at warehouse A that might be used to avoid a split order. Our first example illustrates such an instance. A second key idea is to consider one SKU at a time. For each SKU we can construct and solve a transportation problem that attempts to reduce the number of split orders.

The transportation problem allocates the supply of the SKU at the supply nodes (the warehouses) to the demand nodes (orders that include the SKU). We illustrate this problem later in the section. Thus, with these ideas, we state the heuristic as the following:

For SKU $i: i = 1 \rightarrow N$

1. Construct transportation problem for SKU i
2. Solve transportation problem i
3. Update all affected orders

where N is the number of SKU's. We only consider SKUs that have single orders or uncommitted inventory, as well as split orders with single shipments that consist of SKU i . Starting with a sequence of SKU's, we construct a maximization transportation problem for each SKU. After solving a transportation problem, we update the affected orders, and continue with the next SKU. We terminate at the end of the SKU sequence.

We start with an example to describe the transportation problem. We consider a batch of orders listed in Figure 6. We construct the corresponding maximization transportation

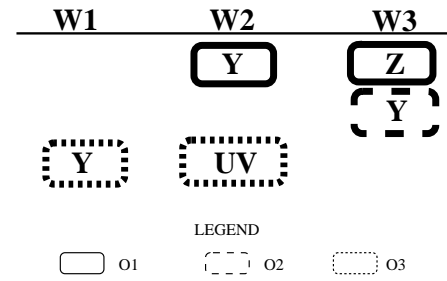


Fig. 6. Real-time assignments

problem for SKU Y in Figure 7. Each warehouse represents a

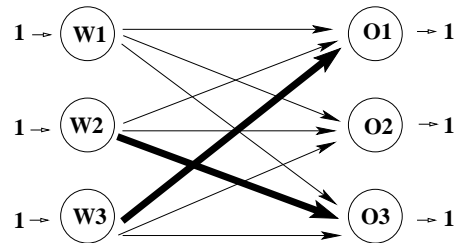


Fig. 7. Transportation problem for SKU Y.

supply node, and each order with a single shipment of SKU Y represents a demand node. The supply at each supply node is the number of units of SKU Y that are available at the warehouse for re-assignment. The demand at each demand node is the number of units of SKU Y in the order. A unit of flow from supply node k to demand node j signifies that warehouse k ships a unit of SKU Y to fill order j 's requirement.. Let $P(j)$ be the set of warehouses such that, $\forall k \in P(j)$, shipping the SKU Y from warehouse k reduces a split in order j . That is, there will be one less shipment if warehouse k supplies the SKU Y or order j . The arc cost for arc $(k, j), \forall k \in P(j)$ is 1, signifying that a unit flow on this

arc results in one less shipment. The arc cost is zero for all other arcs. In Figure 7, $P(1) = 3, P(2) = \emptyset, P(3) = 2$, and only arcs (2, 3) and (3, 1) (the dark arcs) have a cost of 1.

By inspection, we see that the optimal solution is to send one unit of flow along arcs, (1, 2), (2, 3) and (3, 1). The optimal solution corresponds to the results in Figure 8. We

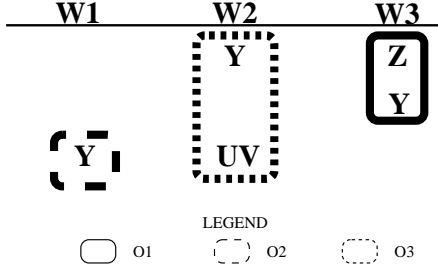


Fig. 8. Re-evaluation reduces number of shipments from 5 to 3.

reduce the number of shipments in the three orders from 5 to 3. We can also see the cyclic exchanges that are required to implement the solution: we need to re-assign the SKU Y inventory at W1, which had been committed to O3 in real time, to O2; we then assign the inventory from W3, which had been for O2, to O1; and finally we assign the inventory from W2 to O3. That is, by implementing the cyclic exchange of SKU Y according to $O3 \rightarrow O2 \rightarrow O1 \rightarrow O3$, we arrive at the solution in Figure 8.

A. Generalized transportation problem

The actual transportation problem that we need to solve is a bit more complex. First, we need to take into consideration the promise-to-ship dates quoted by the e-tailer at the time of the order placement. Second, we also need to differentiate whether the unit of inventory assigned to the customer order is physically in the warehouse or on order. To account for the time dimension, we create T time buckets. In the context of the transportation problem for each SKU, we need to create supply and demand nodes for each time bucket. We have a supply node for each warehouse for each time bucket. We have a demand node for each order, grouped according to its promise-to-ship date. All promise-to-ship dates greater than or equal to T days away from the current date are grouped in the T^{th} category. Since we may be solving the re-evaluation problem periodically, those orders will eventually be in the specific day category.

We formulate a transportation problem for each SKU as in Figure 9.

The details of the transportation problems are as the following.

Supply: We have $T + 1$ supply blocks, where each block contains a supply node for each warehouse. The supply available at each warehouse for the current time block, s_0 , reflects the on-hand inventory, whereas the supply for future time blocks, $s_t, t > 0$, is the on-order inventory that will arrive during the time block.

Demand: We have T demand blocks, one for each shipping date category. Each order is allocated to a demand block according to the *promise-to-ship date* for the order. As illustrated

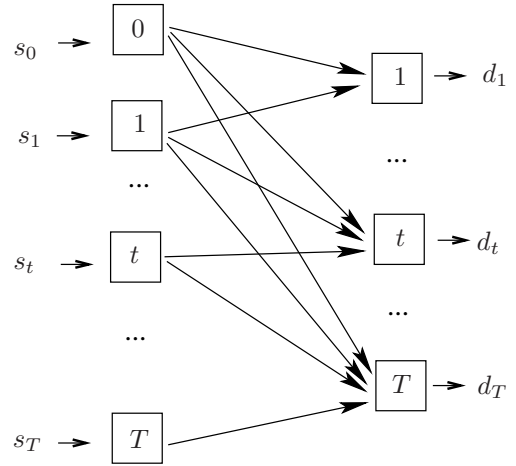


Fig. 9. Transportation problem for each SKU

in Figure 10, for each demand block t , we have one node for each single shipment order, $B_j, j = 1, \dots, m_t$, and we can group all of the single orders into one node A_t .

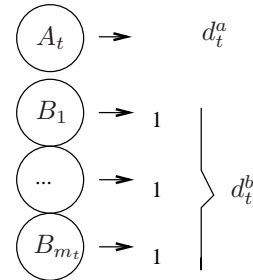


Fig. 10. Demand block t in the transportation problem for a SKU

Arcs: We permit arcs from the nodes in supply block t_2 to the demand nodes in demand block $t_1, \forall t_1 \geq t_2$.

Costs: The cost of all arcs to a single order demand node are all zero, since there is no reduction in shipments from any re-assignment of a single order. The cost of an arcs from the "profitable" warehouse k to a single shipment node j is one, for element of $P(j)$ for SKU i , since this will result in a reduction of one shipment. Otherwise the arc cost is zero for all arcs to a single shipment node j .

V. RESULTS

We have implemented the heuristics on several real data sets from a large e-tailer. The data sets consistently have 800-900K orders and 400K SKU's. In each test set there has been on the order of 16,000 split orders for consideration. For practical reasons, we examine only a subset of all split orders. We exclude some split orders because any re-assignment would make the customer worse off. We exclude some other split orders since the items are too large to be fit into the standard packages.

As one illustration, the 16000 split orders require 33,200 shipments; that is these split orders entail 17,200 splits or extra shipments.

By application of the heuristic, we are able to reduce about 40% of the splits. That is, we can reduce the number of

shipments by approximately 8,000, from 33,200 shipments to 25,200 shipments. The total transportation cost savings can be in the range of \$20, 000 if we save \$2 to \$3 for every split we reduce. As the not-yet-picked queue corresponds to orders for one or two days, we expect that we can repeat this saving by re-solving the problem every one or two days. Thus, we conjecture that there is a significant opportunity for cost reduction.

Our heuristic is relatively easy to implement, as each iteration translates into a series of cyclic exchanges among a limited set of orders. We can feed these exchanges into the e-tailer's existing order-management systems, and as such, are optimistic that implementation is possible.

We conclude that there is an opportunity to reduce the transportation costs for an e-tailer by means of a re-evaluation of its real-time fulfillment decisions. We have developed a heuristic to do this re-evaluation and shown with preliminary testing that it results in better decisions by utilizing more resources and on more information.

REFERENCES

- [1] Ahuja, R. K., Ergun, O. , Orlin, J. B., and A. P. Punnen 2002. A Survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics* 123 75-102.
- [2] R. Allgor and D. Stratila, *Personal Communications*, 2004.
- [3] Balakrishnan, A., T. L. Magnanti, and R. T. Wong 1989. A Dual-Ascent Procedure For Large-Scale Uncapacitated Network Design. *Operations Research*. Vol. 37, No. 5, 716-740.
- [4] Billheimer, J. and P. Gray 1973. Network Design with Fixed and Variable Cost Elements. *Transportation Science* 7, 49-74.
- [5] Magnanti, T. L. and R. T. Wong 1984. Network Design and Transportation Planning: Models and Algorithms. *Transportation Science* 18, 1-55.
- [6] Talluri, K. T. 1996. Swapping applications in a daily airline fleet assignment. *Transportation Science* 30 237-248.
- [7] Wong, R. T. 1985. Probabilistic Analysis of an Optimal Network Problem Heuristic. *Networks* 15, 347-363.
- [8] <http://www.ups.com>