E-coupons Strategy Problems in Location Based Advertisement

Keumseok Kang

kangk@purdue.edu

Krannert School of Management

Purdue University

West Lafayette, IN 47907

Kemal Altinkemer

kemal@purdue.edu

Krannert School of Management

Purdue University

West Lafayette, IN 47907

12-02-2008

Abstract

Location based advertisement (LBA) is a new contextual marketing tool which enables advertisers to attract customers more effectively than ever, by sending location-aware advertisements, which we call e-coupons, through mobile phones. LBA exploits the real-time location information of customers as well as demographic and personal traits. In LBA, due to the privacy and technology issues, an intermediate advertiser, which we call the super mobile, sends e-coupons to customers on behalf of stores. We study the e-coupons strategy problem in LBA from the point of view of the super mobile. First, we show that the full information model of the problem is reduced to a well-known optimization problem, the multidimensional 0-1 Knapsack problem (MKP). MKP belongs to the NP-Hard class of problems. Because of the total unimodularity of the problem, we show that, in a special case, significant e-coupons strategy problems can be solved in reasonable time. Second, we develop dynamic models of the e-coupons strategy problem such as the online, semi-online, and dynamic and stochastic models. In the online model, we show that our problem is reduced to the online MKP and propose threshold type algorithms which use the weighted efficiency as a criterion to select items. E-coupons strategy problems do not have to be solved purely online, where a decision is made immediately upon the arrival of an item, the decision is not revocable, and the item cannot be reconsidered; therefore, we define the semi-online model which lies between the online and full information model. We show that a semi-online solution is better than an online solution in our experiments. In the online and semi-online models, the number of customers who visit the target area is assumed to be known a priori. However, in the dynamic and stochastic model, we assume that the number of customers visit the target area is a random variable drawn from a Poisson distribution and the amount of time they stay in the area is a random variable drawn from an Exponential distribution.

Key words: Location Based Advertisement, E-coupons Delivery, Multidimensional 0-1 Knapsack Problem, Online Algorithm, Semi-online Algorithm, Continuous-time Control Theory.

1 Introduction

The mobile phone industry has recently experienced a tremendous growth in its adoption rate. According to the Mobile Marketing Association, there are 3.3 billion mobile users in the world[1]. In the united states, seventy-six percent of households now own at least one mobile phone and this adoption rate is expected to be eight-seven percent in 2011 (Forrester Research, 2006). Among mobile users, seventy-five percent always or almost always have their mobile phones when they leave the house, and forty-four percent use some form of data service including Short Message Service (SMS) on their phones (Forrester Research, 2007a). Along with the pervasive growth of mobile phones, recent advances in technology such as Global Positioning Systems (GPS) and wireless networks such as Wireless Fidelity (Wi-Fi), Worldwide Interoperability for Microwave Access (WiMAX), and 3rd Generation (3G) networks have created new valuable information, including location information, which was not available before. Location information tells the real time locations of the entities such as parcels, vehicles, and people. Location information has already been used in some areas including military, utilities, retail and whole sale trade, transportation, and rescue operations such as Enhanced 911 service[2] and are rapidly being adopted in other areas (Forrester Research, 2007b). One of the rapidly growing areas using this new information is the Location Based Advertisement (LBA). LBA is a kind of contextual marketing technique, but it is also conceived as a promising next generation advertising method. LBA enables advertisers to provide customers with attractive advertisements at the most appropriate time and location, exploiting location information of customers as well as their individual preferences. Unlike other previous contextual marketing tools such as personalized web sites, recommendation systems, search engine advertisements, and email campaigns, which do not capture the current locations of the target customers, LBA provides advertisers with an additional dimension of information to differentiate their target customers. By

[1] http://mmaglobal.com/modules/article/view.article.php/2082 assessed on 10/21/2008
[2] http://en.wikipedia.org/wiki/Enhanced_911 accessed on 10/21/2008

doing that, LBA enables advertisers to attract customers more effectively and to eventually increase their revenue.

In this paper, we study the e-coupons strategy problem in the context of LBA. In LBA, advertisements are delivered through personal mobile devices such as cellular phones and Personal Digital Assistants (PDAs), and they usually contain favorable offers such as discounts or additional gifts like traditional coupons[3]. In this regard, we call the advertisements used in LBA "e-coupons" in the sense that coupons are delivered electronically. In LBA, there are stores which should be physically located at some place. Stores want to attract potential customers who are nearby by sending e-coupons. However, in practice, due to technical difficulty and privacy issues, stores may not send e-coupons directly to the customers by themselves. Also, the location information of customers can be gathered by only a few authorities such as mobile phone companies or GPS service companies, which we call mobile network carriers in this paper. The service providers of personal mobile devices are major mobile network carriers. In addition to gathering location information, usage of that information is highly limited for privacy concerns; therefore, location information usually cannot be distributed without the permission of owners. For this reason, in LBA, an intermediary entity among the customers, stores, and mobile network carriers sends the advertisement to the customers on behalf of stores using the location data and network provided by the network carriers. Examples of this intermediary entity are Placecast[4] and Acuity Mobile[5]. Customers allow only this intermediary entity to keep and use their location information and personal information such as their preferences or demographic information, and mobile network carriers provide only the intermediate entity with the location information of their customers. We call this intermediate entity the "super mobile."

[3] It is possible that some advertisements do not provide any benefits to the customers. Instead, they may just provide some information of product, service, or brand. Actually, they may not be called coupons in practice, but for the simplicity, we also call them coupons in this paper.
[4] http://www.1020.com/ accessed on 10/21/2008
[5] http://www.acuitymobile.com/ accessed on 10/21/2008

Incentive compatibility exists among these four entities – customers, stores, mobile network carriers, and the super mobile –in LBA. As explained in the figure 1, the super mobile sends e-coupons on behalf of stores; instead, stores pay the advertising fees to the super mobile and may provide the super mobile with customer data. The super mobile uses the location information and network resources provided by mobile network carriers when they send e-coupons; instead, the super mobile pays the information and network resource fees to mobile network carriers. Customers allow only the super mobile to exploit their personal and location information and to send messages to them; instead, customers get favorable e-coupons from the super mobile.

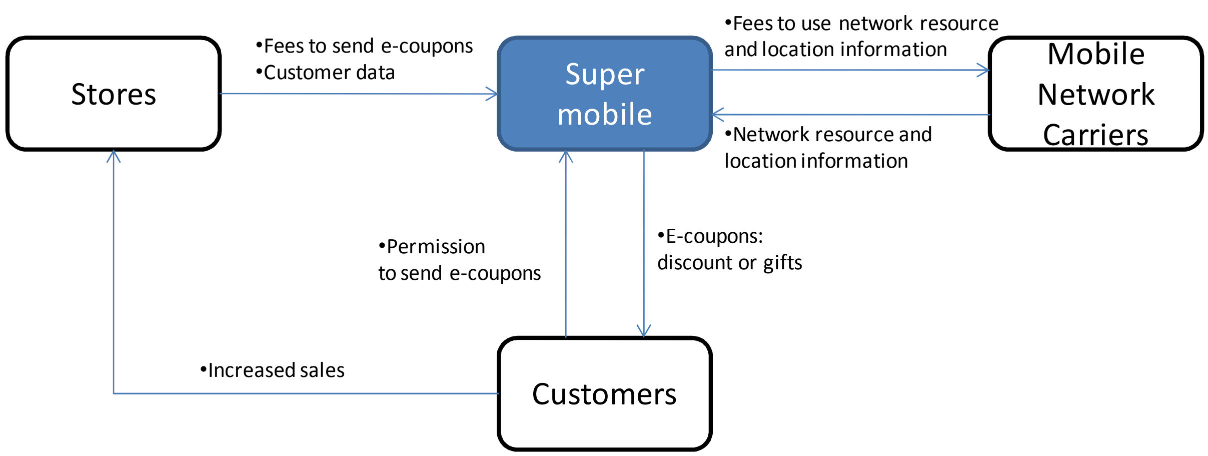


Figure 1 Incentive compatibility of location based advertisement

We study the e-coupons strategy problem from the point of view of the super mobile. The e-coupons strategy problem which we define here is an optimization problem which maximizes the profit of the super mobile. The super mobile earns revenues from stores and pays costs to mobile network carriers. We assume that prices and costs are exogenous in this problem. Prices and costs are determined in advance between the super mobile and stores, and between the super mobile and mobile network carriers, respectively. We assume that the price and cost to send an e-coupon is determined by the e-coupon, customer, location, and time; therefore, profit is determined by what e-coupons are sent to which customers, in what regions, and at what times. An advertisement campaign is usually limited by a deadline or a budget. E-coupons are the same. We assume that stores have budget constraints on sending their e-coupons. The super mobile cannot send an e-coupon if the budget of that e-coupon, which has been defined by the store which owns that e-coupon, does not remain. In addition to the budget constraints, e-coupon deliveries should be restricted by one behavioral factor of customers, which we call the "annoyance factor." Sending or exposing too many advertisements to a customer may annoy that customer, affect a negative impact, and hurt the sales of related products or services (Ducoffe, 1996; Haghirian, Madlberger, & Tanuskova, 2005; Ho & Kwok, 2002). According to Forrester Research, seventy-nine percent of customers still think that advertising messages sent through their mobile phones are annoying (Forrester Research, 2007b). Also, an empirical study reports that eighty-two percent of the respondents to a survey said that three text messages a day were about the right amount (Barwise & Strong, 2002). We call the number of e-coupons that a customer receives without getting annoyed as the "annoyance number." Finding the annoyance number is another important research topic; however, in this paper, we assume that the number is exogenous. The annoyance number can vary by customer type or customer. We discuss the annoyance number at length in section 3, where we also explain our problem assumptions.

As we have explained so far, there are constraints on sending e-coupons; therefore, the e-coupons strategy problem in LBA is to be an optimization problem with multiple constraints. Additionally, all constraints are Knapsack type capacity constraints; therefore, we define this optimization problem using a well-known problem, the multidimensional 0-1 Knapsack (MKP), which is sometimes called the 0-1 multi-Knapsack or the multi-constraints 0-1 Knapsack[6].

The remaining parts of the paper are organized as follows. In section 2, we review the previous literature on LBA and Knapsack problems. In section 3, we describe the problem and explain our major assumptions and related rationales. Besides, we show one small example of the problem. In section 4, we deal with the full information model. We show that the e-coupons strategy problem can

[6] We use the name of multidimensional Knapsack following the survey work of Kellerer et al. (2004)

be formulated as the MKP and then explain how to solve the problem in the subordinate sections. From section 5, we view the problem from a different angle. In section 4, we assume that we have full information; so, we can solve the problem at once, which is called an offline problem in computer science literature or a static and deterministic problem in operations research literature; however, from section 5, we view the problem as a dynamic problem where a sequence of decisions is made based only on what has happened before each decision epoch, which is called an online problem in computer science literature. In section 5, we deal with a time independent online model which is reduced to the online MKP problem. In section 6, we deal with a semi online model, a variant version of the online model. In section 7, we deal with a time dependent dynamic decision model. We assume stochastic properties on the arriving and staying actions of customers. They are assumed to random variables drawn from the Poisson and Exponential distributions, respectively. In section 8, we present our computational experiments, in section 9, we discuss the managerial insights, and in section 10, we conclude and discuss future directions of the research.

2 Literature Review

2.1 Advertisements strategy problem in location based advertisement

In spite of recent increasing attention on LBA, a very small number of studies have dealt with the advertisement scheduling or delivery problem in the context of LBA. To the best of our knowledge, there are only two sets of papers written by two groups of authors: One is De Reyck and Degraeve (2003, 2006), and the other is Tripathi and Nair (2006, 2007).

De Reyck and Degraeve (2003)'s work is the first paper which addresses the advertisement scheduling problem in an LBA environment. De Reyck and Degraeve (2003) are motivated by a real advertisement scheduling problem faced by a mall in London. In their model, an advertising company sends ads to customers on behalf of stores. The company sends advertisements only to the customers who have decided to receive advertisements on their mobile phones (opt-in) and only when those opt-in customers are in the mall. De Reyck and Degraeve (2003) formulate the problem as a static and deterministic Integer Programming (IP). The objective function maximizes the short-term revenue and long-term customer retention, simultaneously. Decisions include which advertisements to send to which customer segments at each given time frame. De Reyck and Degraeve (2003) propose heuristics to solve the proposed IP using relaxation and decomposition techniques and show that the problem can be solved within a reasonable time using heuristics. De Reyck and Degraeve (2006) additionally show how the system is implemented in more detail in their second paper. De Reyck and Degraeve's works are valuable because they are the first set of papers which deal with the advertisements delivery problem in LBA. In addition, their model is based on a real problem, which lends relevancy to their research and also of the subsequent research in this field. However, as an initiator, their work also leaves much room for growth.

Tripathi and Nair (2006, 2007) also conducted research on the advertisements delivery problem in LBA. They built upon the work of De Reyck and Degraeve (2003), showing that if some constraints are relaxed, De Reyck and Degraeve (2003)'s problem can be solved more effectively without altering the original assumptions of the problem (Tripathi & Nair, 2007).

One common drawback of the models used by De Reyck and Degraeve (2006) and Tripathi and Nair (2007) is that there is only one region in their models. In other words, their models do not differentiate the locations of customers in the mall. Another drawback is they assume perfect prior knowledge on the arrivals, moves, and departures of customers as well as the reactions of customers to advertisement delivered on their mobile phones. Tripathi and Nair (2006) overcome these drawbacks by assuming there are fine grained multiple regions in the target area and customers move from one region to another along time. They also assume that customer's moves have a Markov property. Using these additional assumptions, they define a stochastic and dynamic version of the advertisements delivery problem using Markov Decision Processing (MDP). Because their MDP model becomes easily unsolvable as the problem size grows, they provide heuristics to solve the problem, and in their experiments, they show their heuristic performs well in terms of solution time and objective function value.

Our model enhances the works of De Reyck and Degraeve (2003, 2006) and Tripathi and Nair (2007). Neither model considers multiple regions. In addition, their models do not account for the volume of each customer segment who visits the target area. Actual profit or revenue of the advertiser is not only dependent on the customer segment, but also the volume of customers who get the advertisements. In our full information model, we consider both of these –multiple regions and the volume of customers who visit the target area. Tripathi and Nair (2006) address the stochastic property of customers' moves among regions, but their model does not consider the stochastic properties of the arrivals and departures of customers as well as their visit length. We also study the online and semi-online models which have not been studied yet.

2.2 Knapsack problems

Multidimensional 0-1 Knapsack problems:

The MKP, which is a special case of 0-1 linear integer programming, belongs to the NP-hard class (Freville, 2004). The single constraint case of the problem, which we usually call just the Knapsack problem, is not strongly NP-hard (Freville, 2004), and the effective approximation algorithms have been developed for obtaining near-optimal solutions; however, if there are multiple constraints, it is also known that there is no fully polynomial approximation scheme (FPAS) for the MKP (Kellerer, Pferschy, & Pisinger, 2004), and finding a fully polynomial approximation algorithm for the problem with more than two constraints is also known to belong to NP-Hard class of problems (Magazine and Chern, 1984). In general, it is considered that the problem with 500 variables and 10 constraints is the maximal solvable size of the MKP (Kellerer et al, 2004). There have been many attempts to solve this well-known problem. They include exact solution, approximation, heuristics, and meta-heuristics. For those who are interested in the intricacies of MKP, we recommend Freville (2004), Freville and Hanafi (2005), and Kellerer et al. (2004) as useful references.

Online Knapsack problems:

The online Knapsack problem is also well known. A typical online Knapsack problem assumes that the candidate items to fill the Knapsack are not given at once, but given sequentially one at a time. Whenever an item is given, a decision maker evaluates and decides whether to accept the item to fill the Knapsack or not. Once the decision is made, the decision cannot be revoked and the item cannot be reconsidered.

The performance of an online algorithm on the online problem is usually measured by the gap σ, the ratio between the optimal solution of the full information model (i.e., offline model) and the online solution. An algorithm is called *k*-competitive if the σ of the online algorithm is bounded by the value k.

$$\sigma = \frac{The\ objective\ function\ value\ of\ the\ offline\ optimal\ solution}{The\ objective\ function\ vale\ of\ the\ online\ solution}$$

It is known that there is no non-trivial competitive algorithm (i.e., no k-competitive algorithm) for the online single constraint 0-1 Knapsack problem with unbounded efficiency ratios, which alternatively demonstrates that the worst-case performance ratio is not bounded (Marchettispaccamela & Vercellis, 1995). However, it was also recently found that, in the case of the online single constraint 0-1 Knapsack, if we know the upper bound (U) and lower bound (L) of the efficiencies of all items, the algorithm has the upper bound complexity *log(U/L)+1* and the lower bound complexity *log(U/L)*, respectively (Chakrabarty, Zhou, & Lukose, 2007).

The previous literature introduced so far address the online single constraint 0-1 Knapsack problem. There are a very small number of studies which extend the results of Lueker (1998) and Marchettispaccamela and Vercellis (1995) to the online MKP; however, they actually do not consider the online problems, but the probabilistic property of the static problems (Dyer & Frieze, 1989; Meanti et al., 1990). However, to the best of our knowledge, there are very few studies which deal with the online MKP. One study propose a threshold type online algorithm for the online MKP of the revenue management of air cargo, but it does not theoretically or empirically prove the performance of its algorithm (Pak & Dekker, 2004).

Some variant online Knapsack models have been studied recently (Moshe Babaioff, Hartline, & Kleinberg, 2008; Iwama & Zhang, 2007; Noga & Sarbua, 2005). They have different assumptions from the traditional online Knapsack literature such as revocable decisions, partially fractional Knapsacks, and resource augmentation; however, none of them is identical to our problem.

***K-secretary* problems:**

Another related problem is the secretary problem. If all weights and the capacity of the online single constraint 0-1 Knapsack are equal to one, the online knapsack problem becomes the secretary problem. It the capacity is integer *k* which is more than one, the problem becomes the *k*-secretary problem, which is alternatively called a multiple choice secretary problem.

An $1/(1-5/\sqrt{k})$-competitive algorithm has been found recently (M. Babaioff, Immorlica, Kempe, & Kleinberg, 2007). Babaioff et al. (2007) also extend the basic *k*-secretary model assuming different weights and provide a *10e*-competitive algorithm to find the optimal solution for that variant *k*-secretary problem. The secretary problem has been studied for a long time. For those who are interested in this topic, we recommend these two classic papers: Freeman (1983) and Ferguson (1989).

The *k*-secretary problem is different from our time-dependent online model in the sense that the *k*-secretary problem has only one constraint. Besides, in the *k*-secretary problem, the maximal number of secretaries to be selected, which is *k*, is given as an exogenous variable, while, in our problem, it needs to be found as an endogenous variable. In addition, most secretary problems including Babaioff et al. (2007) assume no prior knowledge on the distribution of the values (profits) of candidate secretaries, while our problem assumes prior knowledge on the distribution of the profits as well as the weights of candidate items. In our dynamic models, all candidates or the types of candidates are known a priori; therefore, we know the distribution of the weights and profits of candidates; however, the knowledge of who will visit among the candidates is not available as in the secretary problem.

Semi-online problems:

Semi-online models or problems are not widely accepted terminology with a clear definition. Instead, it is vaguely used to call any variant online model which has some additional information compared to the pure online models when making online decisions (Kellerer, Kotov, Speranza, & Tuza, 1997). In this paper, we call the model “semi-online” if the model is neither of the online and offline models but somewhere in between and its decisions do not have to be made in a real-time manner. Some semi-online models assume a buffer to store the items and makes a set of decisions related to those stored items before the buffer is exhausted, and other semi-online models assume a fixed time period during which a decision maker can stack up the incoming items and make decisions on the stacked items together (Kellerer et al., 1997). Semi-online algorithms are used in the scheduling and partitioning problem (Kellerer et al., 1997; Seiden, Sgall, & Woeginger, 2000). However, there is very little literature suggesting the semi-online algorithms for the online Knapsack problem. As far as we know, only one study (Yang, Hiroshi, Kazuo, & Naoki, 1993) investigates the online subset sum problem, which is a special case of the Knapsack problem under the assumption that items can be deleted and there is partial information on deletion requests.

Dynamic and stochastic Knapsack problems:

Most online and semi-online Knapsack problems as well as secretary problems assume that the total number of items is given. However, in reality, there are many cases where we are not able to know this number in advance. Some studies on the Knapsack problem deal with this issue (Kleywegt & Papastavrou, 1998, 2001; Lu et al., 1999; Papastavrou, Rajagopalan, & Kleywegt, 1996; Ross & Tsang, 1989; van Slyke & Young, 2000). Instead of assuming that the number of items which will arrive is given, they assume that there is a decision horizon (i.e., time deadline) and items arrive according to the Poisson distribution during the decision horizon. Kellerer et al. (2004) name these models as the time dependent Knapsack problem in the sense that the time concept is incorporated into the model. One model (Ross & Tsang, 1989) assumes not only the arrival time of an item, but also the sojourn time of the item affect the decision of whether to accept the item or not, and the sojourn times of items are also random variables drawn from follow the Exponential distribution. Most literature assumes that the decision, whether to accept the arrived item, should be decided upon the arrival of the item (Kleywegt & Papastavrou, 1998, 2001; Lu et al., 1999; Papastavrou et al., 1996; Ross & Tsang, 1989; van Slyke & Young, 2000) and does not consider multiple Knapsack constraints (Kleywegt & Papastavrou, 1998, 2001; Lu et al., 1999; Papastavrou et al., 1996; Ross & Tsang, 1989; van Slyke & Young, 2000). Some literature provides the closed form of optimal policy; however, due to the complexity of the problem, it is not easy to get the optimal policy for this significant problem (Kleywegt & Papastavrou, 1998, 2001; Lu et al., 1999; Papastavrou et al., 1996; Ross & Tsang, 1989; van Slyke & Young, 2000).

Our problem has similar assumptions to Ross and Tsang (1989); however, our model is quite different from their model. To the best of our knowledge, our model is unique and has not been addressed by any literature.

3 Problem statements and assumptions

3.1 Assumptions and issues

As we explained in section 1, there are four different entities in LBA: customers, stores, mobile network carriers, and the super mobile. In addition to these, there is one environmental entity in the context, which we call the region.

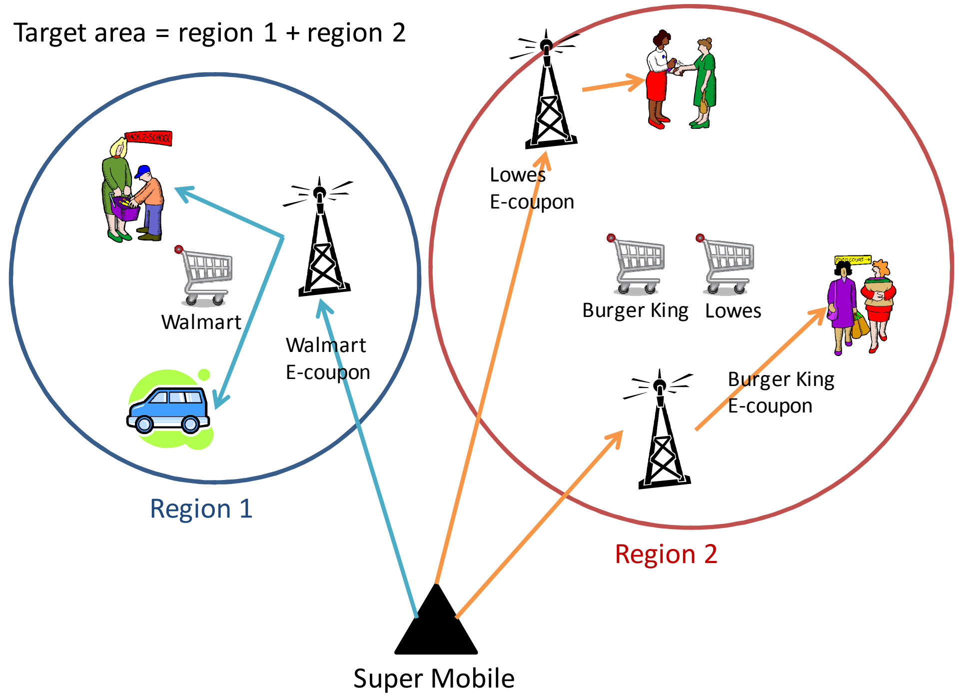


Figure 2 Regions, stores, customers, e-coupons, mobile network carriers, and the super mobile

There is an area where the super mobile and stores are interested in sending e-coupons. We call it the target area. The target area can be any area as long as the super mobile can send e-coupons to the customers who are in the target area. For instance, it can be a mall like in the case of De Reyck and Degraeve (2003, 2006), or it can be a downtown area with many stores. We assume that the target area consists of multiple regions. A region is a physical segment of the target area. However, regions do not have to be physically adjacent to each other and do not have to have the same shape or size.

Customers may visit the target area. A customer stays at only one region in the target area at a given point in time; however, he/she can move between the regions along time. A probable scenario is as follows: a customer arrives at one region in the target area, stays for a while at the region, moves to the other region and stays for a while again, repeats moving and staying, and finally leaves the target area. We define three actions that customers can take: arriving, moving, and leaving actions. Arriving and leaving actions represent the events where a customer appears in the target area and disappears from the target area, respectively. Moving represents the event that a customer, who has already visited one region, moves to another region. In reality, these actions randomly happen, and also the intervals between any two consecutive actions (i.e., length of visit in one region) are also random. We discuss this randomness at length in sections 4 and 7.

We assume that stores are physically located in some place so that the locations affect the attractiveness of the stores and e-coupons that the stores issue. Basic assumption is that a store is located in one region of the target area; however, it is possible that a store may be located outside the target area or on the border of two regions. Actually, the attractiveness of stores reflected in the price of e-coupons along with regions; therefore, the real physical location of stores does not factor in our models.

Now we consider e-coupons. We assume each e-coupon is owned by only one store and related to one or more products or services that the store provides. A store may have multiple e-coupons, but without loss of generality, we assume that each store has only one e-coupon. This simplification may affect one type of constraints, which are the budget constraints. We address this in section 4. E-coupons may have effective periods. We assume that our decision horizon is shorter than the effective period of any e-coupon so that if the e-coupon is sent, it is effective until the end of the decision horizon and that, during the decision horizon, available e-coupons do not change.

The super mobile sends e-coupons to customers who are in the target area on behalf of stores. In reality, it may be possible to have more than one super mobile in the same target area; however, in this paper, we assume there is only one super mobile who deals with a given target area. Considering multiple super mobiles and their competition is another interesting research topic, but it is outside the scope of this paper.

As we mentioned in section 1, the costs and prices for sending an e-coupon are assumed to be exogenous variables and will not change during the decision horizon. In general, these costs and prices are determined by e-coupon (i.e., what to send), customer (i.e., whom to send), time (i.e., when to send), region (i.e., where the customer is). Let us explain why the costs and prices are different. Costs are defined between the super mobile and the mobile network carriers. Customers may use different mobile network carriers for their mobile phone services; therefore, the super mobile may need to use different mobile network carriers in order to reach customers. Based on the business strategy and the infrastructure condition, each carrier may charge different fees to the super mobile for sending e-coupons to the customers. Therefore, the cost can be different by customer, which is actually by the service provider of the customer's mobile phone. E-coupons are messages and may have different types. For example, type can be a text, image, or multimedia. Multimedia or image type messages consume more network resources than ordinary text messages; therefore, they are more expensive to send. In addition, from the point of view of mobile network carriers, the network infrastructure is a sunk cost, and so they want to maximize the utilization of their networks. It naturally leads the network carriers to set a more expensive price to send a message at the peak time than at the non-peak time, and in the crowded and active region where many people are using mobile phones than in the empty and inactive region where a small number of people are using mobile phones, respectively. In this regard, we assume that costs also vary with e-coupon, time, and region. Similarly, prices are assumed to vary with customer, e-coupon, region, and time. First, naturally different costs lead the different prices. For instance, the super mobile may charge more for sending multimedia e-coupons than text e-coupons, at the peak-time than at the non-peak time, and in active regions than in inactive regions simply because they cost more. In addition, the super mobile may provide a discount to long- term contracted stores or high-volume contracted stores, which also leads heterogeneous prices.

We assume that each store has a budget for its e-coupon campaign. There are two possible scenarios. A store may have a separate budget for each e-coupon which the store owns or one budget for all owned e-coupons. Because these two scenarios do not differ in terms of the mathematical model, without loss of generality, we assume each e-coupon has its own separate budget. This assumption is consistent with our previous assumption of the one-to-one relationship between stores and e-coupons. We call these constraints "budget constraints." We explain the impact on the mathematical model, if we assume that multiple e-coupons share one budget (when a store has multiple e-coupons) in section 4.

As we previously discussed, sending too many e-coupons to the same customer within a given period can produce a negative effect. Advertisements beyond a certain number, which we call annoyance number, may irritate the customer. The annoyance number may differ between one customer and another and may change over time. Actually, in practice, it is recommended that the customers set the number or frequency of messages to be delivered in LBA (Gratton, 2002). We assume the annoyance number may vary by customer. However, without loss of generality, we assume that each number for each customer is constant during the decision horizon. We call these constraints "annoyance constraints."

In some cases, there can be network capacity constraints. Sending e-coupons consumes the network resources of mobile network carriers. Therefore, sending too many e-coupons may deteriorate the quality of the carrier's network. In order to maintain overall quality level, the mobile network carriers may set some constraints on the number of e-coupons which can be sent by the super mobile during a certain period of time. Some previous literature assumes that the network carriers allocate a portion of their network for the super mobile to use for sending e-coupons in advance and the allocated capacity may vary by time period (e.g., peak time, non-peak time, etc.) (Tripathi & Nair, 2006). However, we do not incorporate this type of constraint into our model. We think that the costs which vary with time and region influence the usage of the super mobile on the carriers' networks. In other words, in our model, the budget constraint containing the costs plays the same role as the network capacity constraints do in Tripathi & Nair (2006)'s model; actually, their model does not account for budget constraints. However, in some cases, the network capacity constraints are still needed even though the budget constraints exist. Therefore, we show that our model can be extended to have the network capacity constraint and discuss the marginal difficulty of solving the problem in section 4.

According to who decides to send or receive an e-coupon, LBA can be implemented in two different ways: the push and pull systems. In the push system, it is the advertiser, in our case, the super mobile, who decides to send an e-coupon; on the other hand, in the pull system, it is the customer who decides to request an e-coupon. For the privacy issues, even in the push system, the super mobile is required to get the consent from customers before using the location information of customers and sending the e-coupons to them. Therefore, in the push system, sending e-coupons only to the customers who have opted in to receiving e-coupons in advance--we call these customers "opt-in customers" hereafter--is a common practice (Gratton, 2002). In this paper, like De Reyck and Degraeve (2006)'s model, we assume a consent based push system where the super mobile sends e-coupons only to opt-in customers. A pull system can also be modeled using similar mathematical models, but it is out of scope of this paper. The assumption of the consent based push system enables the super mobile to know the population of the possible customers whom e-coupons can be sent to (i.e., opt-in customers). It means that the super mobile also has the distribution of the prices and costs to send e-coupons to every customer. It is very valuable information when the super mobile needs to make decisions in an online manner. This information differentiates the addressed problem from the traditional online Knapsack problems. We discuss it in sections 5, 6, and 7.

As we explained so far, the super mobile sends e-coupons based on real-time location information of customers with the budget and annoyance constraints. The e-coupons strategy problem is defined as the ideal situation to send e-coupons, which maximize the profit of the super mobile. Each decision is to decide whether to send a given e-coupon to a given customer at a given time, and at a given region. Profit is determined by the combination of decisions.

3.2 Problem example

In this example, for simplicity, we assume there are only three customers (C_p=3), four regions (R=4), and two e-coupons (A=2) which mean two stores (S=2). The decision horizon is 10 hours (T_d=10).

- *i*= 1, 2, 3 : customer
- *l*= 1, 2, 3, 4: region
- *j*=1, 2: e-coupon
- *et*: elapsed time which is continuous, $et < T_d$.

The prices which stores need to pay to the super mobile for sending e-coupons and the costs which the super mobile should pay to the mobile network carriers are determined by the combination of customer, e-coupon, time, and location and are given as constants a priori. We assume that there are two time periods defined by mobile network carriers (e.g., peak time and non-peak time). The first period lasts from 0 to 5, and the second period starts at 5 and ends at 10. Two periods have different price and cost structures.

Table 1 Prices and costs to send e-coupons

Period	0<= et <5									5<= et <=10				
i: customer	1				1				…	…	3			
j: e-coupon	1				2				…	…	2			
l: region	1	2	3	…	1	2	3	4	…	…	1	2	3	4
p: price	4	4	3	…	5	5	5	5	…	…	1	2	3	4
c:cost	2	1	2	…	3	4	4	4	…	…	1	1	1	2

The maximal number of e-coupons which can be delivered to each customer, the annoyance number, which is represented by a_i, is 1 for every customer (*i*=1, 2, and 3). The budget on each e-coupon, which are represented by b_j, is 8 for the e-coupon 1 (*j*=1) and 10 for the e-coupon 2 (*j*=2), respectively.

- a_i =1: annoyance number, for *i*=1, 2, and 3.
- b_1=8 and b_2=10: budget, for *j*=1 and 2.

So far, we have explained the exogenous variables which are provided as a priori knowledge and constants during the decision horizon. All versions of the e-coupons strategy problem addressed in this paper consistently assume these variables are exogenous and constant.

The following table represents customer actions. All six actions occur during the decision horizon.

Table 2 Customer actions along with elapsed time

t: time frame	1		2	3	4	
et: elapsed time	0.5	2.1	3.7	7.4	8	8.2
customer action	Arriving	Arriving	Moving	Moving	Moving	Leaving
i: customer	1	2	2	1	1	2
l: location	2	3	4	4	3	1

As represented in the following figure, customer 3 does not appear in the target area during the decision horizon, which is possible. It means that customer 3 is not considered to receive any e-coupons even though he/she is one of customers.

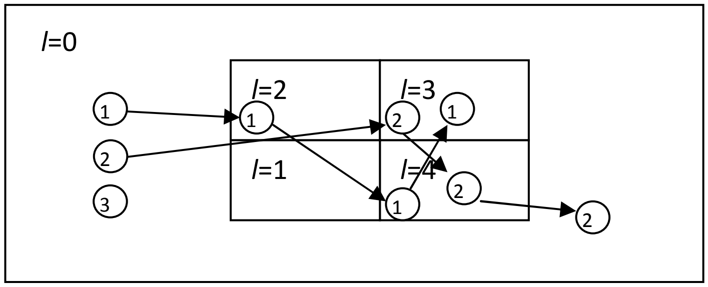


Figure 3 Customers' actions and regions

In reality, in the consent based push environment, the super mobile may know the total number of opt-in customers (C_p); however, it is very difficult for the super mobile to know how many customers will visit the target area or how many customer actions will occur during a specific period in advance (e.g., the number of customers who will arrive, move, and leave during the next ten hours). However, in some cases, those numbers can be forecasted from the historical data. We deal with different assumptions on the availability of the number of customers or customer actions. According to the assumption, the problem needs to be modeled and solved differently. We discuss the various models of the e-coupons strategy problem with different assumptions about customers' actions in the subsequent sections of the paper. In the above table 2, time frames (t's) are used to discretize the continuous time. They are explained in the full information model in section 4.

4 Full information model

In the full information model, we assume that there is an omniscient decision maker who knows exactly how many customers will arrive at each region, move between regions, and leave the region, who will come, and when they will arrive, move, and leave during the decision horizon T_d. This assumption is reasonable when there is a repeated pattern of the actions of customers so that the super mobile can predict future actions from the historical data.

In the full information model, decision horizon T_d is broken down into a finite set of small time frames, and we claim that these time frames can be defined reasonably small to include all changes in the location of customers and the prices and costs of e-coupons and also reasonably large not to oversize the problem.

One time frame cannot include more than one arrival or move of the same customer, but it may contain the arrivals or moves of more than one customer. In other words, within a time frame, a customer cannot change his location more than once, but more than one customer can change their locations simultaneously. Similarly, one time frame cannot include more than one change on the cost and price of a specific e-coupon for a specific customer, but it may contain the changes on the prices or costs of other e-coupons or for other customers. However, it is reasonable to think that prices and costs will not change as frequently as customer locations do. So, costs and prices, which are assumed to vary by time, do not have to vary by time frame. Accordingly, the intervals of time frames are not homogeneous. For instance, if a customer changes his location twice very promptly or if a customer changes his location right after the price to send an e-coupon to that customer is changes, then the time frames around those moments will be very short.

In the full information model, a set of decisions are related to each time frame. Decisions related to a time frame do not have to be confined only to the customers whose locations change and the e-coupons of which prices or costs change during that time frame; instead, all customers who are in the target area and all e-coupons are considered. We make this strategy decision intentionally because it is still possible that a common e-coupon can be sent to the same customer more than once as long as it does not annoy him/her, although everything including the location of the customer and the price and cost of the e-coupon does not change.

One might argue that we should define finer time frames so that the model includes all decisions which possibly could be made during the decision horizon. As we demonstrated in section 3.2, customers can arrive, move, or leave at any point of time; therefore, the finest model is a continuous time model where decisions can be made any time. In the full information model, which is a discrete time model, we assume that it is not reasonable to send more than one identical e-coupon to the same customer if everything in the system including the location of the other customers and the prices and costs of other e-coupons as well as his/her own location and the price and cost of the e-coupon remains the same (i.e., in one time frame by the definition of the time frame). However, in the continuous model, change based on time itself may trigger other decision making. In this case, it may be necessary to make a decision even if no change in locations, prices, and costs occurs (i.e., no change in state) because of the changes on remaining time. The continuous model is explained in section 7.

Some may also argue that time frames should have a fixed interval (i.e., periodic decisions). However, in the case of fixed intervals, it may be possible to lose some events where decisions can be made. For instance, if we define long interval frames, some customers may arrive at and leave the target area or move around multiple regions within one frame, which is a loss of opportunity to send e-coupons; on the other hand, short interval frames may increase the frequency of decisions unnecessarily, which increases the cost to solve the problem.

The following are the definitions of the variables as well as the subscripts of the variables.

Subscripts

- $i \in I = \{1, \dots, \mathrm{C}\}$: C customers who visit the target area.
- $j \in J = \{1, \dots, \mathrm{A}\}$: A e-coupons. We assume there are A stores, and each store has only one e-coupon.
- $t \in T = \{1, \dots, \mathrm{P}\}$: P time frames. Time frames represent when decisions are made. We assume that decisions are made at the beginning of each time frame.
- $l \in L = \{1, \dots, \mathrm{R}\}$: R regions.

The region where customer i is at time frame t will be given in this full information model, and a customer cannot be located in multiple regions in one time frame. Therefore, in the full information model, l is a function of customer i and time frame t. So we define the function $l(i,t)$.

- $l(i,t) \in L = \{1, \ldots, R\}$: Region where customer i is at time frame t.

Decision variable:

- $x_{ijtl(i,t)}$: if e-coupon j is decided to be sent to customer i who is in region l at time frame t, $x_{ijtl(i,t)}$ = 1, and otherwise, $x_{ijtl(i,t)}$ =0. $x_{ijtl(i,t)} \in \{0,1\}\ \forall i, j, t$.

Exogenous variables (parameters)

- p_{ijtl}: Price that store j pays to the super mobile when the super mobile sends e-coupon j to customer i at region l and at time frame t.
- c_{ijtl}: Cost that the super mobile pays to mobile network carriers when the super mobile sends e-coupon j to customer i at region l and at time frame t.
- a_i: Annoyance number of customer i, which is the maximal number of e-coupons which can be sent to customer i.
- b_j: Budget on e-coupon j (equivalently, of store j).

All prices p_{ijtl} and costs c_{ijtl} are positive real number, and the price is always greater than the cost (i.e., $p_{ijtl} > c_{ijtl}\ \ \forall i \in I, \forall j \in J, and\ \forall t \in T$). Budgets b_j are real numbers, and annoyance numbers a_i are integers. The following is the basic model of the full information model.

Basic Model (BM):

$$Max\ \sum_{i\in I, j\in J, t\in T}\left(p_{ijtl(i,t)} - c_{ijtl(i,t)}\right)x_{ijtl(i,t)} \qquad \text{(BM-1)}$$

$s.t.$

$$\sum_{j\in J, t\in T} x_{ijtl(i,t)} \le a_i \quad \forall i \in I \qquad \text{(BM-2)}$$

$\sum_{i\in I,t\in T} p_{ijtl(i,t)} \cdot x_{ijtl(i,t)} \leq b_j \ \ \forall j \in J$ (BM-3)

$x_{ijtl(i,t)} \ \in \{0,1\} \ \ \forall i \in I, \forall j \in J, and\ \forall t \in T$ (BM-4)

(BM-2) and (BM-3) represents annoyance constraints and budget constraints, respectively.

The MKP is finding the set, $\{\boldsymbol{x} | Max\ \boldsymbol{cx}\ s.t\ \boldsymbol{Ax} \leq \boldsymbol{B}\ and\ \boldsymbol{x} \in \{0,1\}\}$. The basic model is the MKP consisting of C×P×A decision variables, C annoyance constraints with P×A variables per constraint, and A budget constraints with C×P variables per constraint; therefore, the matrix **A** is C×P×A by C+A.

In a special case, the super mobile may always charge the same price for sending any e-coupon. We can simplify the basic model adding this assumption. Without loss of generality, we assume that b_j / p for every j is integer. We call this problem the "Constant Price Model" (CPM).

Constant Price Model (CPM):

$Max\ \sum_{i\in I,j\in J,t\in T} (p - c_{ijtl(i,t)}) x_{ijtl(i,t)}$ (CPM-1)

$s.t.$

$\sum_{j\in J,t\in T} x_{ijtl(i,t)} \leq a_i \ \forall i \in I$ (CPM-2)

$\sum_{i\in I,t\in T} x_{ijtl(i,t)} \leq b_j/p \ \ \forall j \in J$ (CPM-3)

$x_{ijtl(i,t)} \ \in \{0,1\} \ \ \forall i \in I, \forall j \in J, and\ \forall t \in T$ (CPM-4)

Lemma 1. The coefficient matrix **A** of the CPM, $\{\boldsymbol{x} | Max\ \boldsymbol{cx}\ s.t\ \boldsymbol{Ax} \leq \boldsymbol{B}\ and\ all\ elements\ of\ \boldsymbol{x} \in \{0,1\}\}$, is totally unimodular.

A is a C×P×A by C+A matrix, and all elements of matrix are zero or one. Let (*i,j,t*) be the column index which represents a unique column and (*i, j*) be the row index which represents a unique row of the matrix **A**. Let row index (*i*,0) represent the annoyance constraint for customer *i* and (0,*j*) represent the budget constraint for e-coupon j. Let COL be the set of column indices and ROW be the set of row indices. Let $a_{(i,j),(i,j,t)}$ be the element of the matrix **A** where the row is (i,j) and the column is (i,j,t).

For any given column (i,j,t) of the matrix **A**, there are exactly two elements whose values are one, which are $a_{(i,0),(i,j,t)}$ and $a_{(0,j),(i,j,t)}$ elements. The other elements are all zeros. Let ROW_i be the subset of ROW consisting of $(i,0)$ rows where $i \in I$ and ROW_j be the subset of ROW consisting of $(0,j)$ rows where $j \in J$.

Then, for any given column (i,j,t) of the matrix **A**, $\sum_{k \in ROW_i} a_{k,(i,j,t)} - \sum_{k' \in ROW_j} a_{k',(i,j,t)} = 0$. Therefore, according the theorem 2.7 (page 542) of Nemhauser and Wolsey (1988), the matrix **A** is totally unimodular.■

The CPM is still the MKP, but it can be easily solved because the matrix **A** of the problem is totally unimodular. Any IP problem which has the form of $\{\boldsymbol{x} | Max\ \boldsymbol{cx}\ s.t\ \boldsymbol{Ax} \leq \boldsymbol{B}\ and\ \boldsymbol{x}\ are\ integer\}$ with totally unimodular matrix **A** can be solved by any polynomial linear programming algorithm such as interior point method in polynomial time (Nemhauser & Wolsey, 1988).

Unlike the CPM, the BM gets more difficult to solve when the problem size gets increased; therefore, we propose a Lagrangean relaxation method for large BM. The BM consists of two groups of constraints. One is annoyance constraints (BM-2), and the other is budget constraints (BM-3). We now define the Lagrangean relaxation model relaxing the budget constraints (BM-3) of BM using Lagrangean multipliers $\boldsymbol{\lambda}$.

Lagrangean Relaxation Model (LRM): *z(λ)* :=

$$Max\ \sum_{i \in I, j \in J, t \in T} \left(p_{ijtl(i,t)} - c_{ijtl(i,t)}\right) x_{ijtl(i,t)} + \sum_j \lambda_j \left(b_j - \sum_{i \in I, t \in T} p_{ijtl(i,t)} x_{ijtl(i,t)}\right) \quad \text{(LRM-1)}$$

$s.t.$

$$\sum_{j \in J, t \in T} x_{ijtl(i,t)} \leq a_i \ \forall i \in I \quad \text{(LRM-2)}$$

$$x_{ijtl(i,t)} \in \{0,1\} \ \forall i \in I, \forall j \in J, and\ \forall t \in T \quad \text{(LRM-3)}$$

$\lambda_j \geq 0 \;\; \forall j \in J$ (LRM-4)

Again, LRM becomes the MKP of which coefficient matrix **A** is totally unimodular.

Lemma 2. The coefficient matrix **A** of the LRM, $\{\boldsymbol{x} | Max\ \boldsymbol{cx}\ s.t\ \boldsymbol{Ax} \leq \boldsymbol{B}\ and\ all\ elements\ of\ \boldsymbol{x} \in \{0,1\}\}$, is totally unimodular.

A is a C×P×A by C matrix, and all elements of matrix are zero or one. Let (*i,j,t*) be the column index which represents a unique column and (*i*) be the row index which represent a unique row of the matrix **A**. Let COL be the set of column indices and ROW be the set of row indices. Let $a_{(i),(i,j,t)}$ be the element of the matrix **A** where the row is (*i*) and the column is (*i,j,t*).

For any given column (*i,j,t*) of the matrix **A**, there are exactly one element whose value is one, which are $a_{(i),(i,j,t)}$ element. The other elements are all zero. Let ROW_i be the set ROW and ROW_{Φ} be an empty set.

Then, for any given column (i,j,t) of the matrix **A**, $\sum_{k \in ROW_i} a_{k,(i,j,t)} - \sum_{k' \in ROW_{\Phi}} a_{k',(i,j,t)} = 1$. Therefore, according to the theorem 2.7 (page 542) of Nemhauser and Wolsey (1988), the matrix **A** is totally unimodular.■

Like the CPM, the LRM with given $\boldsymbol{\lambda}$ can be solved in polynomial time. We solve the Lagrangean dual problem, which is to find the set $\{\boldsymbol{\lambda} | Min\ z(\boldsymbol{\lambda})\ s.t\ \boldsymbol{\lambda} \geq 0\}$ using a subgradient optimization algorithm. Because the LRM, which our subgradient algorithm solves at each stage, can be solved in polynomial time, our subgradient algorithm solves the Lagrangean dual problem in a relatively short time.

Subgradient Optimization:

Initialize **λ**, $\lambda_j^0 = 0$, for all *j*. In each iteration *k*, calculate the subgradient $\gamma(\lambda_j^k) = b_j - \sum_{i \in I, t \in T} c_{ijtl(i,t)} x_{ijtl(i,t)}$ for all *j*. Update the multipliers, $\lambda_j^{k+1} = \max\{0, \lambda_j^k +$

$t_k \gamma(\lambda_j^k)\}$, and the step size $t_k = s_k \frac{z(\lambda^k) - z_{LB}}{\left|\gamma(\lambda_j^k)\right|^2}$, where z_{LB} is the best lower bound found so far. The parameter s_k is initially set to 2. After g number of iterations, if there is no improvement, s_k is updated, $s_{k+1} = s_k / 2$.

At any iteration k, for given $\boldsymbol{\lambda}_k$, we solve the maximization problem LRM and find the solution. Let $x_{ijtl(i,t)}$'s be the optimal solution of the LRM given $\boldsymbol{\lambda}_k$ at iteration k and $z(\boldsymbol{\lambda}_k)$ be the objective function value of the optimal solution. Let $z(\boldsymbol{\lambda}^*)$ be the minimal value of $z(\boldsymbol{\lambda}_k)$ over all iterations k, which is the upper bound. At iteration k, the gap between the upper bound and the lower bound is defined as GAP = $z(\boldsymbol{\lambda}^*)$- z_{LB}. The algorithm terminates when one of the following conditions hold: 1) A pre-specified number of iterations have been completed, or 2) GAP is close enough to zero (smaller than a precision parameter).

Because we relax some constraints, $z(\boldsymbol{\lambda}_k)$ including the best bound $z(\boldsymbol{\lambda}^*)$ are mostly infeasible solutions of the original model, BM, if the solution **x** which produce the $z(\boldsymbol{\lambda}_k)$ along with $\boldsymbol{\lambda}_k$ is feasible in the BM, then the solution **x** is the optimal solution of the BM, and the following condition, $z(\boldsymbol{\lambda}_k)$ = $z(\boldsymbol{\lambda}^*)$ = optimal objective function value of the BM, holds.

Heuristics to find the feasible lower bound:

In order to get the tight lower bound, first, we define the Linear relaxation model (LM) relaxing the integrality constraint (BM-4) of the BM and solve it using a linear programming algorithm.

Linear relaxation Model (LM):

$$Max \sum_{i \in I, j \in J, t \in T} \left(p_{ijtl(i,t)} - c_{ijtl(i,t)}\right) x_{ijtl(i,t)} \qquad \text{(LM-1)}$$

$s.t.$

$$\sum_{j \in J, t \in T} x_{ijtl(i,t)} \leq a_i \ \forall i \in I \qquad \text{(LM-2)}$$

$$\sum_{i \in I, t \in T} p_{ijtl(i,t)} \cdot x_{ijtl(i,t)} \leq b_j \ \forall j \in J \qquad \text{(LM-3)}$$

$$0 \leq x_{ijtl(i,t)} \leq 1 \ \forall i \in I, \forall j \in J, and\ \forall t \in T \qquad \text{(LM-4)}$$

The solution of the LM is mostly an infeasible solution of the BM. If the LM solution consists of only zero or one, then it is the optimal solution of the BM. Otherwise, the LM solution is infeasible. The infeasible LM solutions may have one or more fractions (i.e., the number which is greater than zero but less than one) in its solution. Let us round down all fractions in the infeasible LM solution (i.e., set them to zero). Then, this new solution is feasible in the original model, BM. We call this solution as LM feasible solution, and we start with this solution as the initial z_{LB}.

We use a greedy algorithm to find the feasible lower bound. It is known that there is no greedy algorithm to solve the general MKP optimally (Kan et al., 1993). The greedy algorithms for the MKP, $\{\boldsymbol{x} | Max\ \boldsymbol{cx}\ s.t\ \boldsymbol{Ax} \leq \boldsymbol{B}\ and\ all\ elements\ of\ \boldsymbol{x} \in \{0,1\}$, usually use the following weighted ratio, $\frac{c_j}{\sum_{i \in I} \omega_i a_{ij}}$ where ω_i (> 0) is the weight for the constraint i (i.e., the ith row of matrix **A**), to decide whether to include x_j in the solution (i.e., x_j =1) or not (i.e., x_j =0) where x_j's are selected in a decreasing order of the weights. There are many attempts to find the optimal weights of the greedy algorithm for MKP, which may not provide the optimal solution, but the best solution among the greedy solutions (Kellerer et al., 2004). When the number of constraints is fixed, irrespective of the number of the decision variables, the optimal weights of the greedy algorithm can be found in polynomial time, $O(n^m \log n)$, where m is the number of constraints, and n is the number of decision variables; however, it becomes difficult to find the optimal weights if the number of constraints gets increased when the number of variables increases (Kan et al., 1993), which is the case in our example. Finding the optimal weights of the greedy algorithm in MKP is known to belong to the NP-Hard class of problems, if the number of constraints is always greater than the number of variables when both increase (Kan et al., 1993). It is an open question whether finding optimal weight is HP-hard when the number of constraints is smaller than the number of variables like our case (Kan et al., 1993).

However, our focus is not on this open problem. Our basic model has C+A constraints and C×P×A decision variables. Usually, decision variables are more than constraints, but not always.

At every iteration k in the subgradient algorithm, we update the Lagrangean multipliers, $\boldsymbol{\lambda}_k$. We use $\boldsymbol{\lambda}_k$ for the weights of corresponding constraints in the greedy algorithm at iteration $k+1$. The greedy algorithm only costs $O(n\log n)$ which is less than the complexity of any polynomial linear programming algorithm which solves our subproblem LRM at every iteration.

Extensions of the basic assumptions:

As we explained in section 3, the full information model is based on the assumption that each store has only one e-coupon and each e-coupon has a budget. Let us explain some different scenarios and investigate their impact on our original problem. First, let us assume that there are S stores, where S is less than the number of e-coupons, which is A. Each e-coupon should belong to only one store, but each store can have one or more e-coupons. Also, budget is defined on the store not on the individual e-coupon; therefore, all e-coupons owned by a store now share one budget of the store. Now, we have a new model which consists of S budget constraints with C×P×A_s variables per constraint, where A_s is the number of e-coupons that store s has (i.e., $\sum_{s\in S} A_s = A$). The problem is still the MKP with less number of constraints, which is C+S, than the original BM. However, the problem is not different in terms of its solution approach. If we assume the same costs, it becomes totally unimodular and trivial to solve. Otherwise, we can relax these budget constraints using Lagrangean relaxation like the LRM, and the sub problem again becomes a totally unimodular case.

A second assumption is about the available network capacity. Network capacity constraints that we decided not to include in our basic model lead a different type of additional constraints. If the available network capacities is varying by time period, the problem needs additional P network capacity constraints with C×A variables per constraint, where P is the number of those time periods. Alternatively, if we assume that the network capacity is constant during the decision horizon, the model needs additional one constraint with C×P×A variables. Again, network capacity assumption does not change the structure of our problem much. If the available capacity is defined as a maximal number of messages that the super mobile can send for a given period, the constraints consists of only 0-1 coefficients, and the CPM and the subproblem of LM still holds the total unimodularity even after these new constraints are added. Otherwise, we can additionally relax these constraints in LRM and keep the properties of total unimodularity of the sub problem of LRM.

In these regards, we think that investigating the proposed original models: BM, CPM, and LRM would be sufficient to investigate the solution approach of the full information model of e-coupons strategy problem in the context of location based advertisement.

5 Time independent online model

In section 4, we assume that the super mobile has precise information about customer actions: arrivals, moves, and leaves. In this section, we see the problem from a different angle. We assume a priori knowledge on customer actions is no longer available. In reality, the super mobile in the consent based push system knows the total number of opt-in customers (C_p) who have already consented to receive e-coupons; however, all opt-in customers necessarily do not visit the target area during a given period; therefore, the super mobile is hardly able to know how many or which individuals among them will really visit the target area during the decision horizon. In order to deal with this uncertainty, we introduce dynamic decision models, where decisions are made not at once, but sequentially at stages. First, we introduce a time independent online model in this section following the conventional assumptions of the previous literature on the time independent online model. We use the term of “time independent” because those models do not include the concept of time (e.g., an elapsed time or remaining time). In section 7, we deal with a time dependent model where the remaining time plays an important role. For simplicity, the time dependent online model will be called the “online model” shortened in the remaining parts of this paper.

The online model follows the major assumptions of the traditional on-line Knapsack literature (Lueker, 1998; Marchettispaccamela & Vercellis, 1995). First, we assume that the total number of customers who visit the target area during the decision horizon, which is equivalent to the number of items in the online Knapsack literature, is assumed to be known a priori. It makes sense in some cases, where the super mobile can predict the number of customers' actions from historical data or where the target area has a spatial limit like sport game stadium or music concert hall so that every time, only a fixed number of customers can enter the target area. Second, we assume that the other information such as which individuals among all opt-in customers will come, when they will come, and which regions they will visit is assumed to be unknown a priori; instead, it is revealed only after the customer arrives. Naturally, the costs and prices of the potential e-coupons which can be sent to that customer are revealed at that time. The last major assumption of the traditional online literature is that when an item arrives to the system, the decision on the item should be made immediately and if the decision is made once, the decision is not revocable and the item cannot be revisited. The online model also follows this assumption. When a customer arrives at one region of the target area, a decision whether to send e-coupons to the customer is made immediately. Decisions made at this moment are only related to the customer who arrived most recently. Decisions are not related to other customers in the region, who were already considered to be sent e-coupons, when they arrived at the target area. Because the super mobile does not know how long each customer will stay in the region, an immediate decision upon the arrival of the customer seems reasonable. Also, every customer in the target area was already considered for e-coupons when they arrived, so reconsidering them seems unnecessary. However, this last assumption of the traditional online Knapsack model also seems too restrictive in our context – a push environment of LBA. For this reason, we define the semi-online model relaxing this last assumption. We discuss the semi-online model in section 6.

The above three assumptions enable us to reduce our online model to an online MKP; however, our model has two unique characteristics that the traditional online Knapsack models do not have. First, in traditional online Knapsack models, the decision for an arrived item is single binary decision (e.g. accept or reject the item); however, in our model, a group of binary decisions need to be made when a customer arrives. When a customer arrives, the super mobile not only decides whether to send e-coupons or not, but also decides what e-coupons to be sent among all possible e-coupons. Every possible e-coupon is considered, and more than one e-coupon can be sent to the customer at the same time. We implement this group of decisions as a batch arrival of items. In our online Knapsack model, the item is not the customer but the pair of the e-coupon and customer. When a customer arrives at the target area, every e-coupon which can be sent to the customer forms a binary decision variable (i.e., send the e-coupon to the customer or not), which represents an item of the batch arrived in the online model.

Second unique characteristics exist on the knowledge of the populations of the profits and weights of items (i.e., the efficiencies of items). Traditional single constraint online Knapsack problems assume that the total number of items which will come is known a priori, which is identical to our model, but the efficiency of items, which is the ratio between the coefficient of objective function of an item (called profit in the literature) and the coefficient of constraint of that item (called weight in the literature), is assumed not to be bounded (Lueker, 1998; Marchettispaccamela & Vercellis, 1995), while, in our online model, this efficiency is not bounded. In LBA, because of the availability of the information about the opt-in customers and e-coupons, the super mobile has perfect knowledge on the distribution of the costs and prices of all items, which are all possible pair between the opt-in customers and e-coupons. Opt-in customers and e-coupons are finite; therefore items are also fine. In this regard, the super mobile should know the upper and lower bounds on a finite set of efficiency.

In single constraint online 0-1 Knapsack problems, one of the most well-known criteria to determine whether to include the item or not is the efficiency. In the single constraint 0-1 Knapsack problem, $\{\boldsymbol{x} | Max\ \boldsymbol{cx}\ s.t\ \boldsymbol{ax} \leq b\ and\ all\ elements\ of\ \boldsymbol{x} \in \{0,1\}\}$, the efficiency of item j ,e_j, is defined as

$e_j = \frac{c_j}{a_j}$, which is same to the ratio used in the greedy algorithm. Any threshold type algorithm has a common form that if $e_j \geq threshold$, item j is accepted, and otherwise, item j is rejected. If the threshold is constant during the decision horizon, then the algorithm is called a non-adaptive online algorithm and if the threshold changes according to what has already happened, it is called an adaptive online algorithm.

The threshold and efficiency become complicated when there is more than one constraint. As we already discussed in the previous section, the efficiency can have a single value form of $e_j = \sum_i \omega_i \frac{c_j}{a_{ij}}$, where ω_i is the weight for constraint i. Then, the corresponding threshold also needs to be a single value. On the other hand, the efficiency can have a multi-valued (vector) function, $\boldsymbol{e_j} = \left\{e_{ij} \middle| e_{ij} = \frac{c_j}{a_{ij}} and\ i = 1, \dots, m\right\}$, and the threshold also takes a form of multi-valued vector $\boldsymbol{thres} = \{thres_i |\ i = 1, \dots, m\}$. Although Kan et al. (1993) show that a greedy algorithm using weighted ratios $e_j = \sum_i \omega_i \frac{c_j}{a_{ij}}$ as its ranking criteria does not necessarily find the optimal solution, we use the weighted efficiency ratio for our online threshold algorithm for its simplicity in this paper. Without loss of generality, we also assume that the sum of the weights is equal to one. Multi-valued vector thresholds will be investigated in the next iteration of this paper.

Let T_{hres}(**rb**, rm) be a threshold function, where **rb** is the vector of remaining capacities of the constraints and rm is the remaining items, Let e_j be the efficiency of the arriving item j. If e_j is greater than T_{hres}(**rb**, rm) and the remaining capacity **rb** can accommodate the item j, item j is accepted, and otherwise, item j is rejected. If the T_{hres} value varies by the values of **rb** and rm, the algorithm which use this threshold function T_{hres}(**rb**, rm) is called an adaptive online algorithm; on the other hand, if the T_{hres} value is constant, the algorithm is called an non-adaptive algorithm.

As we explained at the beginning of this section, our online model has a batch arrival. When a customer i arrives at the region l, the super mobile faces total A decisions, where A is the number of e-coupons and each decision is related to sending each e-coupon. In adaptive algorithms, because each decision affects the remaining capacities and the remaining capacity affects the subsequent decisions, the sequence of decisions among the A decision items matters. We order the items according to their efficiency in a greedy manner.

The ultimate goal of the online model is to define a randomized algorithm which uses a constant threshold drawn from a predefined distribution and to define a deterministic adaptive algorithm, respectively. In order to define good algorithms, we conduct extensive computational experiments varying the weights and the threshold values. Final algorithms and analytical analysis on the competitive ratios of these two algorithms will be reported in the next iteration of this paper. Experimental results are discussed in section 8.

6 Time independent semi-online model

Although there are some variant online Knapsack models (Babaioff et al., 2008; Iwama & Zhang, 2007; Noga & Sarbua, 2005), in a general setting of online Knapsack problems (Lueker, 1998; Marchettispaccamela & Vercellis, 1995), it is assumed that items arrive in a random order and the decision to accept or not the most recently arrived item should be made immediately upon the arrival of the item (before the next item arrives) and decisions are not revocable and the item will not be revisited after the decision is made. In this section, we relax this assumption and define a new time dependent dynamic decision model.

In LBA, does the decision really need to be made immediately upon the arrival of customer? Because the customers stay for some while in one region, it may be possible to defer the decision whether to send e-coupons to the customers as long as they are in the region. Here, we assume that sending e-coupons in a deferred manner has the same effect to the immediate decisions which send e-coupons to the customer immediately after he/she comes into a new region. Customer's behaviors may differ by the time they receive the e-coupons; however we already took that effect into our model assuming different prices sand costs. We implicitly assume that price is partially determined by the effect of e-coupons, which varies by time. For example, the super mobile may be able to charge more for the e-coupons issued by a restaurant during lunch or dinner time than other times because the expected revenue is greater during those times. As we explained in our full information model, if the price or cost of an e-coupon changes, decisions related to that e-coupon with regard to the customers already in the target area needs to be reinvestigated. However, for a short time period, during which there is no change in the price and cost of any e-coupon, we assume that the expected effects of the e-coupons stay same; therefore, the deferred decisions within that short period should have the same effects as immediate decisions.

A typical online setting assumes that if a decision for an item is made, then the item will not be revisited later. That model makes sense in some cases such as hotel or airplane reservations; however, in other cases such as graduate school admission, items can be revisited later as long as the item is still available. In the push based e-coupon delivery problem, customers even do not know that they are not selected to send e-coupons, when those decisions are made. Therefore, revisiting customers does not hurt the customers and the decision maker as long as it benefits both sides.

In our new online model, decisions to send e-coupons to the customer who just arrived or moved in the region can be made even after subsequent customers arrive in the region, as long as that customer is still in the region. Also, as long as the customer is still in the region, he/she is reconsidered to send e-coupons again and again. We call this new model a semi-online model in the sense that the decisions do not have to be made immediately in an online manner.

Revisiting the customers can be implemented easily. Like the full information model, all customers who are already in the target area are revisited in the semi-online model. The only problem in the semi-online algorithm is that the decision maker does not know precisely how long each customer will stay in the region where he/she is now; therefore, it is difficult to define the decision epochs, when decisions should be made. We propose two simple strategies to define the decision epochs. In the first strategy, decisions are made whenever an item arrives or the price or cost of an e-coupon changes. In this case, every decision epoch of the online model corresponds to that of the semi-online model; however, the semi-online model has additional epochs which the online model does not have. In addition to the additional epochs, in the semi-online model, whenever a decision is made, all customers in the target area are reconsidered, which is not the case in the online model. A second strategy is the periodical epoch strategy or random epoch strategy. In the periodic epoch strategy, at every fixed time interval, decisions to send e-coupons to the customers in the target area are made. In the random epoch strategy, decision epochs are randomly chosen, and decisions are made at each decision epoch. Regardless of the decision epoch strategy, the semi-online algorithm considers all customers and all e-coupons at every decision epoch. Therefore, a group of decisions are made at each epoch (i.e., at the same time). Like the online model, in the adaptive algorithm, all items (decisions) including those related to the newly arrived customer are ordered according to the efficiencies and are considered sequentially.

The following figure shows how the semi-online model, online model, and full information model differ in when decisions are made (i.e., decision epochs) and what decisions are made at each decision epoch. The following figure illustrates the example explained in section 3.2.

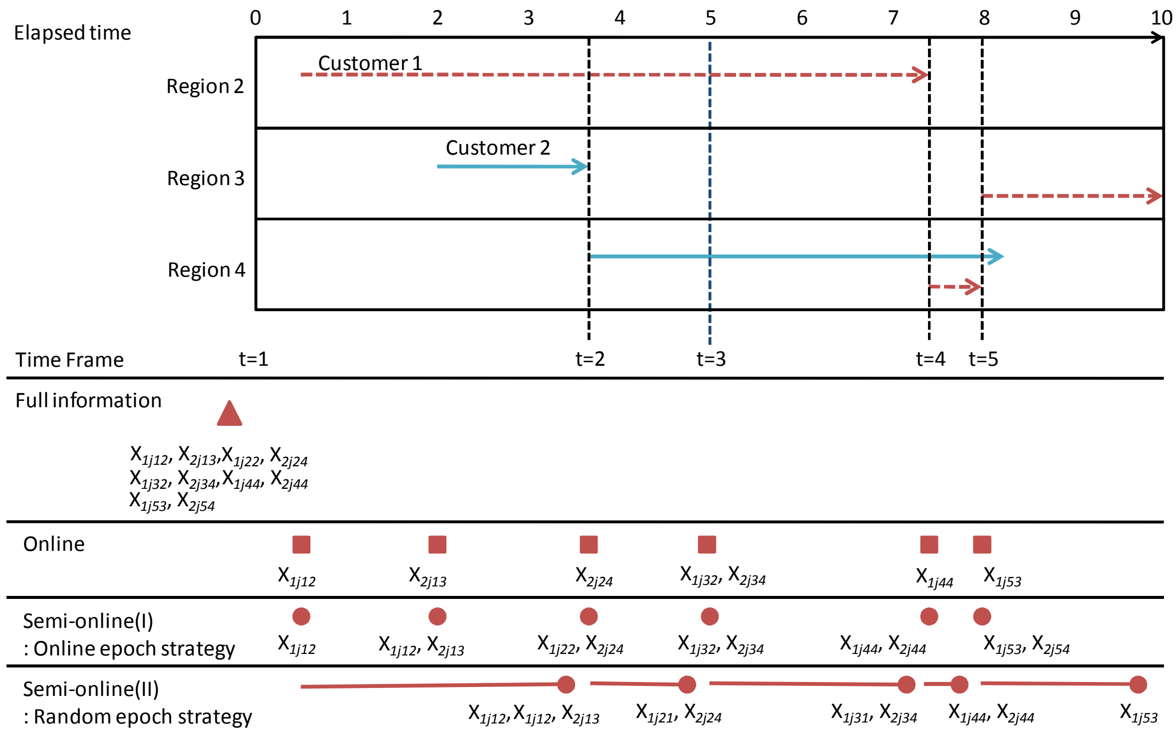


Figure 4 Comparisons among models: the full information, online and semi-online

The full information model uses time frames. One time frame contains only one change in region per each customer and one change in price or cost per e-coupon. In the full information model, all decisions are made at the beginning of the decision horizon (at et=0, notation: ▲). In the online model, a decision epoch (notation: ■) corresponds to the time when a customer arrives or moves. At each decision epoch, the decisions only related to the newly arrived customer are made, which is a group of decisions. For example, in the online model, the time (et=0.5), when customer 1 arrives at region 2, becomes the first decision epoch, and the time when customer 2 arrives at region 3 (et=2), becomes the second decision epoch. At the first decision epoch, decisions only related to customer 1 at region 2 are made and at the second decision epoch, decisions only related to customer 2 at region 3 are made. Corresponding decision variables of the full information model for both epochs are x_{1j12}'s and x_{2j13}'s for all j=1, 2,…, A, respectively. Every decision epoch in the online model corresponds to the decision epoch (notation: ●) in the semi-online model (I). Besides, like the time frame, changes in cost or price can create additional decision epochs. In the above figure, the time, et=5, is the decision epoch because of the changes in prices and costs of the e-coupons. Decisions made on each decision include every customer in the target area, which is also different from those of the online model. For example, at the second decision epoch, not only customer 2 who just arrived, but also customer 1 who is in the target area is the candidate customer to receive e-coupons. The semi-online model (II) uses time frames as its decision epochs; however, decisions are assumed to be made just before the end of each frame.

Lemma 3. Let A is a space for the problem instance and σ be an instance, $\sigma \in A$. Let $\text{opt}_{\text{online}(\mathbf{w},\text{thres})}(\sigma)$ be the objective function value of an online algorithm with the weights vector **w** for the efficiency and the threshold *thres*, $\text{opt}_{\text{semi-online}(\mathbf{w},\text{thres})}(\sigma)$ be the objective function value of an semi-online algorithm with the weights vector **w** for the efficiency and the threshold *thres*, and $\text{opt}_{\text{full}}(\sigma)$ be the objective function value of the full information optimal solution. Then, $\text{opt}_{\text{online}(\mathbf{w},\text{thres})}(\sigma) \le \text{opt}_{\text{full}}(\sigma)$ and $\text{opt}_{\text{semi-online}(\mathbf{w},\text{thres})}(\sigma) \le \text{opt}_{\text{full}}(\sigma)$ for all $\mathbf{w} > \mathbf{0}$, $t > 0$, $\sigma \in A$, and $E_A[(\text{opt}_{\text{full}}(\sigma)- \text{opt}_{\text{online}(\mathbf{w},\text{thres})}(\sigma))/ \text{opt}_{\text{online}(\mathbf{w},\text{thres})}(\sigma)] \le E_A[(\text{opt}_{\text{full}}(\sigma)- \text{opt}_{\text{semi-online}(\mathbf{w},\text{thres})}(\sigma))/ \text{opt}_{\text{semi-nline}(\mathbf{w},\text{thres})}(\sigma)]$ for all $\mathbf{w} > \mathbf{0}$ and $t > 0$[7].

We conduct extensive experiments to show the performance of our semi-algorithm. The solutions of the semi-online algorithm are compared with the deterministic optimal solutions as well as the online solutions.

7 Time dependent online version: Dynamic and stochastic model

Stochastic models assume some parameters which are constants in the full information model (i.e., deterministic version) are random variables drawn from probability distributions. Parameters can be anything such as profits, annoyance number, capacities, and customer actions in our problem. In this section, we assume that the total number of arrivals and moves of the customers, *n*, is assumed to be unknown a priori. Instead, the distributions of the arrival, moves, and leaving of customers are

[7] $E_A[\cdot]$ represents the average of a given random variable over the space A.

assumed. This stochastic assumption can be incorporated into both static and dynamic models. However, in this paper, we only consider the stochastic assumption in dynamic models, and we call this model as the dynamic and stochastic model following the naming convention of Kleywegt and Papastavrou (1996, 1998).

Like the online and semi-online models, the dynamic and stochastic model assumes there is no a priori knowledge on the occurrence of customer actions and solves the problem dynamically (i.e., at multiple stages). However, unlike the online and semi-online models, it assumes that customer actions are random variables drawn from some probability distributions. If we assume that the customers arrive according to the Poisson distributions, our problem can be reduced to a dynamic and stochastic Knapsack similar to those of van Slyke and Young (2000), Kleywegt and Papastavrou(1998, 2001), Papastavrou et al. (1996) and Lu, et al. (1999).

Like the traditional online Knapsack or secretary problem, previous literature on the dynamic and stochastic Knapsack model (Kleywegt & Papastavrou, 1998, 2001; Lu et al., 1999; Papastavrou et al., 1996; Ross & Tsang, 1989; van Slyke & Young, 2000) assumes that a decision is made right after an item arrives and the decision is only for the most recent item and is not revocable. However, as we discussed in the semi-online model, in the context of our problem, a customer may stay at the region for a while, and during that time, a decision can be deferred or a new decision is made again for that customer, if necessary. In order to support deferred and revisited decisions, we assume another stochastic parameter, the staying times of customers (or alternatively called the sojourn times in queueing model literature). In our dynamic and stochastic model, staying times are random variables drawn from the Exponential distributions. Poisson or exponential assumption on the arrivals and staying times of items is a common practice in the queueing model (Kleinrock, 1976). However, our problem is totally different from the queueing model. The objective of queueing model is to investigate or optimize the service performance of the queueing systems like expected waiting time, number of items in queues, or utilization of servers, while our model has a different objective function, which is to fill the Knapsack by selecting the items from the queueing system consisting of an infinite number of servers. Ross and Tsang (1989) investigate the online Knapsack problem with the Exponential distribution assumptions on the arrival and sojourn times; however, their objective function is different from ours. In their model, items stay in the system according to the Exponential distribution, which is similar to our model; however, while an item stays, this item occupies a portion of the Knapsack and provides the profit. When the item leaves the system, the occupied resource is released, and it can be used for next items. The objective function is to maximize the profit. In our model, any consumed resource is not recovered during the decision horizon, and the staying time is used only for deferring decisions. To the best of our knowledge, our problem is unique, and it enables us to have a unique dynamic and stochastic Knapsack model.

Poisson assumptions on the arrivals of customers to the target area from outside may make sense; however, in the case of arrivals to each region, which may include the influx from the other adjacent regions, total arrivals may not follow the Poisson distribution. Multiple adjacent regions create a complicated queueing system. Therefore, for the sake of simplicity, we assume that there is only one region in the dynamic and stochastic model. Also, the different annoyance numbers of customers creates an additional complexity to the problem; therefore, we assume the annoyance numbers for all customers are one in this section. We extend the model to have the different annoyance numbers by customer and the multiple regions in the next version of the paper.

In order to define the dynamic and stochastic model, we need following new additional assumptions.

- There are K types of customers. Opt-in customers are classified into these customer types. Customers of customer type *k* arrive according to the Poisson distribution PP(λ_k(t)) and stay according to the Exponential distribution E(μ_k(t)). Poisson and Exponential distribution are non-homogeneous; therefore, the arrival rate λ_k(t) and leaving rate μ_k(t) varies by time. The super mobile knows these rates of each customer type.

- The prices and costs to send e-coupons to the customers of same customer type (at the same region and at the same time) are the same.

Let **W** = (w_1, w_2, …, w_c) (c is the number of constraints of the Knapsack) be a multidimensional capacity vector. They correspond to the budget constraints in the full information model. Define f($\boldsymbol{y}$, $\boldsymbol{z}$, t) as the maximal expected benefit (the objective function) with vector $\boldsymbol{y}$ = (y_1, y_2, …, y_c), capacity and time t left as well as vector $\boldsymbol{z}$ =(z_1,z_2,…,z_K), the number of customers whom an e-coupon have sent so far by customer type. Let f($\boldsymbol{y}$, $\boldsymbol{z}$, 0) = -∞, f($\boldsymbol{y}$, $\boldsymbol{z}$, t) = -∞ if $\boldsymbol{y}<\mathbf{0}$, and f($\boldsymbol{y}$, $\boldsymbol{z}$, t) not to be bounded. Define **A**($\boldsymbol{y}$, $\boldsymbol{z}$, t) as an acceptance policy which is a function that for every value ($\boldsymbol{y}$, $\boldsymbol{z}$, t) gives the answer what items to be accepted or to be rejected. The objective is to find an acceptance policy **A**($\boldsymbol{y}$, $\boldsymbol{z}$, t) that realizes f(**W**, **0**, T). **A**($\boldsymbol{y}$, $\boldsymbol{z}$, t) returns a set of actions a_{kj}'s, where *k*=1,2,…,K and *j*=1,2,…,A and *k* and *j* represent the customer type *k* and the e-coupon *j*, respectively.

The problem holds the optimality principle of dynamic programming. In other words, the optimal policy for f($\boldsymbol{y}$, $\boldsymbol{z}$, t) does not depend on how the system got to ($\boldsymbol{y}$, $\boldsymbol{z}$, t) from (**W**, **0**, T). Therefore, we define a discrete state, continuous time, finite horizon, and dynamic programming problem.

$$f(\boldsymbol{y},\mathbf{z},\mathrm{t}) = \int_0^{t}\sum_{k=1}^{K} p_{k,\boldsymbol{z}}(\mathrm{s})\cdot \max\{\mathrm{profit}_{kjs} + \mathrm{f}(\boldsymbol{y}-\boldsymbol{w_{kjs}},\mathbf{z}+\mathbf{e}_{\boldsymbol{k}},\mathrm{s}), \mathrm{f}(\boldsymbol{y},\mathbf{z},\mathrm{s})\}\mathrm{ds} \qquad \text{(DSM-1)}$$

$p_{k,\boldsymbol{z}}(\mathrm{s})$ in (DSM-1) represent the conditional probability that, at the remaining time s, at least one customer of the customer type *k* exists in the target area, given that **z** customers have already received e-coupons.

$$p_{k,\mathbf{z}}(\mathrm{s}) = 1 - \frac{\sum_{n=z_k}^{\infty} p_{\mathrm{arrive},k,s}(n)\cdot p_{\mathrm{leave},k,s}(n-z_k)}{\sum_{n=z_k}^{\infty}\sum_{m=z_k}^{n} p_{\mathrm{arrive},k,s}(n)\cdot p_{\mathrm{leave},k,s}(m-z_k)} \qquad \text{(DSM-2)}$$

$$p_{\mathrm{arrive},k,s}(\mathrm{n}) = \frac{{\lambda_{k,s}}^{n} e^{-\lambda_{k,s}}}{n!} \qquad \text{(DSM-3)}$$

$$p_{\mathrm{leave},k,s}(\mathrm{n}) = \frac{{\mu_{k,s}}^{n} e^{-\mu_{k,s}}}{n!} \qquad \text{(DSM-4)}$$

$\lambda_{k,s} = \int_0^{T-s} \lambda_k(t)dt$ (DSM-5)

$\mu_{k,s} = \int_0^{T-s} \mu_k(t)dt$ (DSM-6)

$p_{arrive,k,s}(n)$ represents the probability that, until the remaining time s (i.e., during T-s times), n customers of the customer type *k* have arrived. Similarly, $p_{leave,k,s}(n)$ represents the probability that until the remaining time s (i.e., during T-s times), *n* customers of the customer type *k* have left without receiving nay e-coupon.

$profit_{kjs}$ in (DSM-1) represents the profit incurred by sending e-coupon *j* to one customer of the customer type *k* at remaining time s. It is calculated by

$profit_{kjs} = p_{kjs} - c_{kjs}$ (DSM-7)

p_{kjs} and c_{kjs} are the price and cost for sending e-coupon *j* to a customer of the customer type *k* at remaining time s.

$\mathbf{w}_{kjs}$ in (DSM-1) represents the resources consumed to send e-coupon *j* to a customer of customer type *k* at the remaining time s. $\mathbf{e}_k$ in (DSM-1) represents a K-elements vector of which only k^{th} element is one and the other K-1 elements are zeros.

Our ultimate goal in this section is to solve the (DSM-1) equation, get the closed form of the optimal policy, $\mathbf{A}^*(\boldsymbol{y}, \boldsymbol{z}, t)$.

7.1 Comparisons between the proposed models

The following table summarizes the main characteristics and Pros and cons of the proposed models.

Table 3 Comparison between the proposed models

	Major assumption and characteristics	Pros	Cons
Full information model	• It is assumed that every future event and parameter related to making decisions are known a priori. • We call the solution of this model as the optimal solution.	• It makes all decisions optimally at one time.	• Assumption is not realistic. • It is difficult to solve as the problem size grows.
Online model	• It is assumed that future events are unknown. • Decisions are made sequentially based on what has occurred before and what occurs now. • Decisions are made immediately upon the occurrence of a related event and are not revocable.	• It can be solved by simple decision making process; therefore, it is easy to solve and hardly affected by problem size.	• It does not provide the optimal solution.
Semi-online model	• It is assumed that future events are unknown. • Decisions are made sequentially based on what has occurred before, what occurs now and for a given time window. • Unlike the online model, decisions do not have to be made immediately, but they	• It is between the full information and online models; therefore, it has advantage of both models. • Decision making is easier than the full information model.	• It still does not provide the optimal solution. • Decision making process is more complicated than online model. • Deciding the time window is not easy and it affect the solutions. • It may perform better

	can be made within that given time window. • Decisions can be revisited and made again.		than online model on average.
Dynamic and stochastic model	• It is assumed that future events are unknown; however, stochastic properties on them are available. • Decisions are made sequentially based on what has occurred before, what occurs now, and what is expected to occur.	• It is similar to the semi online, but it provides the better solution due to the availability of the information of stochastic properties on the future events.	• It can be solved optimally.

8 Computational experiments

We generate the instances of the e-coupons strategy problem and solved the problem instances using computers. The following table shows the information of the problem parameters that we use in our experiments. There are 486 combinations of the values of the parameters; the total number of problem instances that we generate and test is 4860.

Table 4 Problem parameters

Parameter name	# of values	Values
Number of customers (C)	3 values	50;100; 200
Number of regions (R)	2 values	1; 5
Number of periods (P)	3 values	1;3;8
Max price (Max_price)	3 values	5;30;100
Max annoyance number (Max_annoyance)	3 values	1;2;6
Number of e-coupons (A)	3 values	1;10;20

Locations of customers are generated according to the uniform discrete distribution, so that customers are evenly distributed among regions. Prices vary by customer, e-coupon, region, and period and are randomly generated from the uniform distribution, $\mathrm{U}[2, \mathrm{Max\_price}]$. Costs also vary by customer, e-coupon, region, and period and are randomly generated from the uniform distribution, $\mathrm{U}[1, \mathrm{corresponding}\ price]$ so that the cost is always smaller than the corresponding price in the target area. Annoyance numbers also vary by customer and are randomly generated from the uniform distribution, $\mathrm{U}[1, \mathrm{Max\_annoyance}]$. In order to avoid the trivial solutions, we let the capacity of budget constraints be generated by the following rules (Freville, 2004; Chakrabarty et al., 2007): 1) every cost of sending an e-coupon is less than the half of the budget capacity of that e-coupon (i.e., $a_{ij} < \frac{b_i}{2}\ \forall i, j$) and 2) the total costs to send an e-coupon in all possible cases is larger than the budget of that e-coupon (i.e., $\sum_j a_{ij} > b_i\ \forall i$). These rules seem reasonable in our problem context.

All experiments are conducted on a cluster of Dell 32-bit Intel Pentium4 computers, called the Radon, at the Rosen Center for Advanced Computing of Purdue University[8]. Programs are coded by C++ and C languages and compiled by GUN GCC compilers[9]. For statistical functions, we use the Boost Library v1.36[10], and for optimization software, CPLEX v 11.0 is used[11].

8.1 Full information model

Optimal solutions

In our experiments, first, we solve the BM problems using the CPLEX MIP solver. It is known that the MKP problem gets more difficult to solve as the number of constraints or variables increases. In our experiments, CPLEX cannot solve some instances of the problems due to the limit of the computational resources. Table 5 shows how CPLEX solves the problems. It seems that the

[8] http://www.rcac.purdue.edu/index.cfm accessed on 10/30/2008.
[9] http://gcc.gnu.org/ accessed on 10/30/2008.
[10] http://www.boost.org/ accessed on 10/30/2008.
[11] http://www.ilog.com/products/cplex/ accessed on 10/30/2008.

difficulty of problem is determined by the number of constraints rather than the number of variables. The number of constraints of the problem with 50 customers and 20 e-coupons is 70, and the numbers of variables of the same combination are 1000, 3000, and 24000 according to the value of number of period, respectively. The problem with 100 customers and 10 e-coupons has 110 constraints and has the same numbers of variables as the problem with 50 customers and 20 e-coupons. Table 5 shows that almost all instances of the former problem are solved optimally (99.8%), while 4.4% of the latter problems are not solved. Similarly, the problem with 200 customers and 10 e-coupons has more constraints (210) than the problem with 100 customers and 20 e-coupons (120); they have the same number of variables. 8.9% of the former problems are not solvable, while 21.5% of the latter problems are not solved.

Table 5 also shows that among the problems with 200 customers, about 78% of instances are solvable. When it comes to the problems with 500 customers, it is found that only 35% of the problems are solved optimally[12]. 200 or 500 customers seem a relatively small sizes of customers in practice, and most practical problems are probably larger than this technical limit. This finding supports the necessity of our relaxation models. As you can see in the table, the CPU times to solve the problem are not long. The problem is the search space. It exceeds the memory and the disk space of the computing resource.

[12] We tested the 20 instances of the problems with 500 customers, 1 (and 5) regions and 8 periods; however, because of the limited availability of the optimal solutions, we omitted them in our analysis.

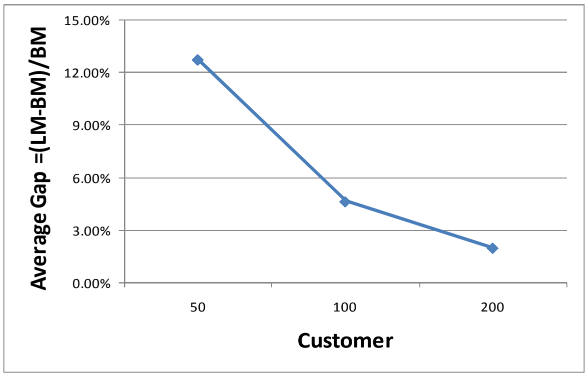
Table 5 Problem instances solved optimally

Customer	E-coupon	Problem instances solved optimally		No. of problem instances	Average CPU time[13] (sec)	Max CPU time (sec)
		No.	Percent			
50	1	420	100.0%	420	0.01	1
	10	660	100.0%	660	4.69	439
	20	539	99.8%	540	5.16	1041
50 Total		1619	99.9%	1620	3.63	535
100	1	420	100.0%	420	0.01	1
	10	631	95.6%	660	27.01	851
	20	492	91.1%	540	48.49	2369
100 Total		1543	95.2%	1620	26.51	963
200	1	420	100.0%	420	0.05	1
	10	518	78.5%	660	188.87	14939
	20	326	60.4%	540	118.31	1383
200 Total		1264	78.0%	1620	107.93	206
Grand Total		4426	91.1%	4860	41.39	206

Performance of the linear relaxation model (LM)

Next, we solve the linear relaxation model (LM) of the problems. Again, we use the CPLEX LP solver to solve them. Performance of the LM model is measured by the gap between the optimal values of the LM and the optimal value of the BM. Gap is defined as a ratio, (optimal value of the LM – optimal value of the BM) / optimal value of the BM. Experimental results show that the LM works fine overall. The following figure shows the average gaps between the optimal solutions of the BM and LM by the parameters of the model. Average gap on total instances is 6.9%.

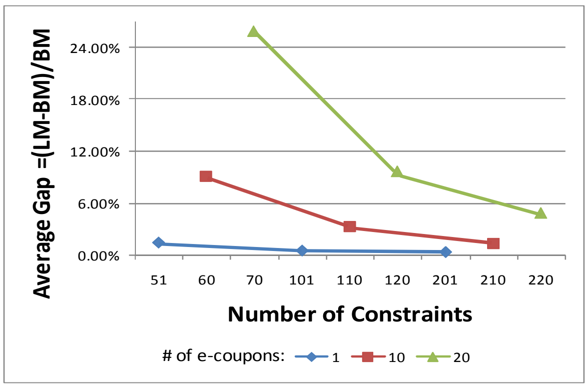
[13] Average and Max CPU times reported in the table are calculated only among the problems solved optimally.


Average Gap =(LM-BM)/BM
15.00%
12.00%
9.00%
6.00%
3.00%
0.00%
50
100
200
Customer

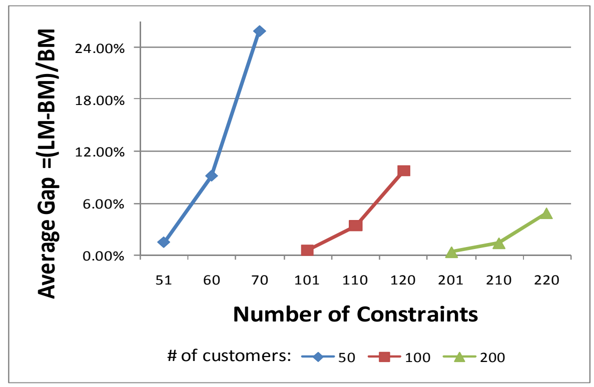
(a)


Average Gap =(LM-BM)/BM
15.00%
12.00%
9.00%
6.00%
3.00%
0.00%
1
10
20
E-coupon

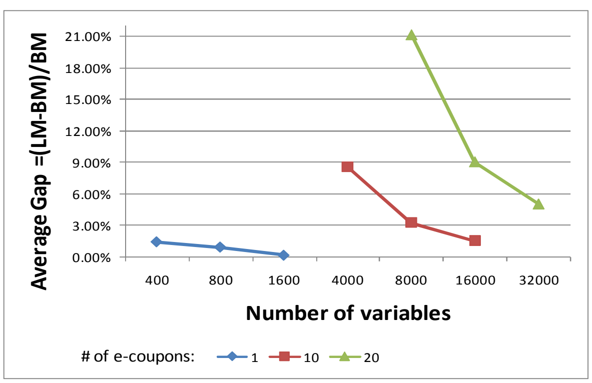
(b)


Average Gap =(LM-BM)/BM
24.00%
18.00%
12.00%
6.00%
0.00%
51
60
70
101
110
120
201
210
220
Number of Constraints
# of e-coupons:
1
10
20

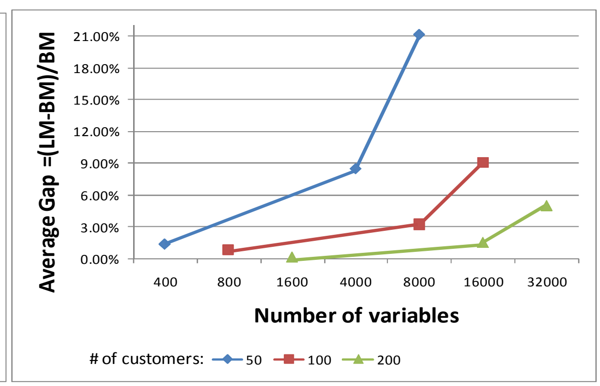
(c)


Average Gap =(LM-BM)/BM
24.00%
18.00%
12.00%
6.00%
0.00%
51
60
70
101
110
120
201
210
220
Number of Constraints
# of customers:
50
100
200

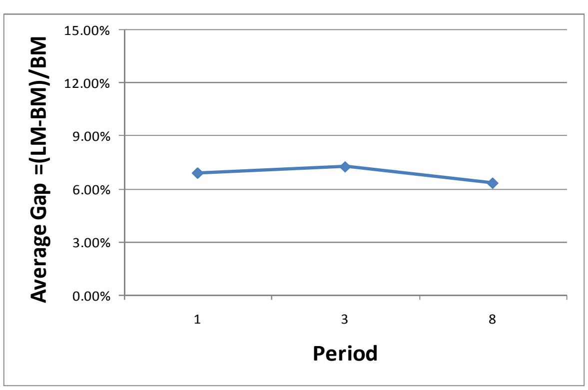
(d)


Average Gap =(LM-BM)/BM
21.00%
18.00%
15.00%
12.00%
9.00%
6.00%
3.00%
0.00%
400
800
1600
4000
8000
16000
32000
Number of variables
# of e-coupons:
1
10
20

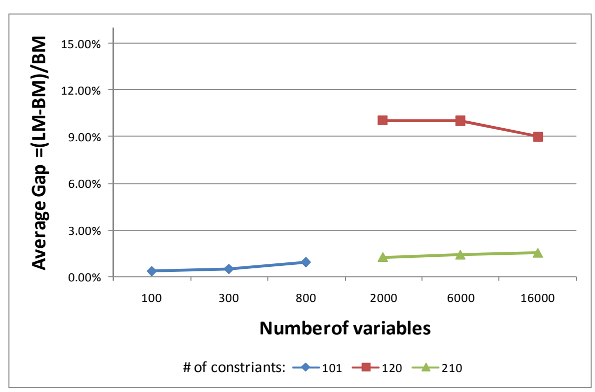
(e)


Average Gap =(LM-BM)/BM
21.00%
18.00%
15.00%
12.00%
9.00%
6.00%
3.00%
0.00%
400
800
1600
4000
8000
16000
32000
Number of variables
# of customers:
50
100
200

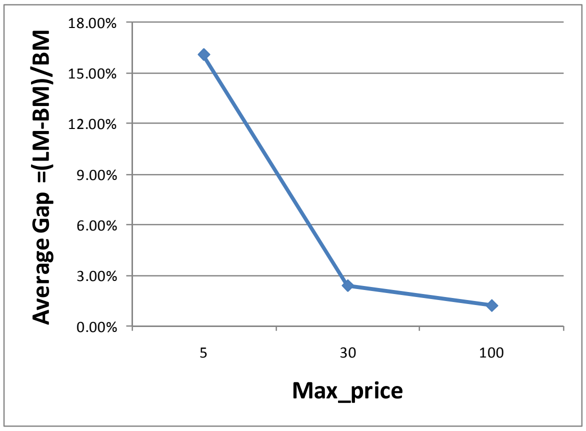
(f)


Average Gap =(LM-BM)/BM
15.00%
12.00%
9.00%
6.00%
3.00%
0.00%
1
3
8
Period

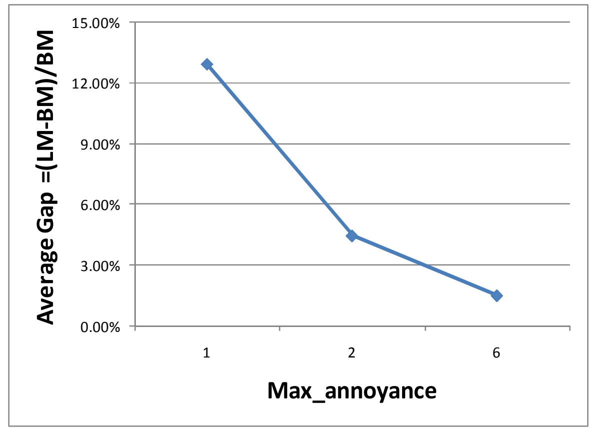
(g)


Average Gap =(LM-BM)/BM
15.00%
12.00%
9.00%
6.00%
3.00%
0.00%
100
300
800
2000
6000
16000
Numberof variables
# of constriants:
101
120
210

(h)



(i)



(j)

Figure 5 Performance of the LM

The number of customers and e-coupons are the parameters which determine the number of variables and constraints of the model; as they increase, the numbers of variables and constraints increase. However, customer and e-coupon have opposite effects on the performance of the LM. Figure 5-(a), (c), and (e) show that the gap becomes smaller as the number of customer increases. On the other hand, figure 5-(b), (d), and (f) show that the gap becomes larger as the number of e-coupons increases. The number of periods affects only the number of variables; however, it almost does not affect the gap according the figure 5-(g) and (h). It seems that it is the constraints which affect the performance and the impact on the performance of the constraints also varies by the type of the constraints. The number of customers affects the annoyance constraints and the number of e-coupons affects the budget constraints. The more the budget constraints are the larger the gap is; however, the more the annoyance constraints are the smaller the gap is.

Max_annoyance affects the capacities of the annoyance constraints, and Max_price affect the capacities of the budget capacities. Both parameters affect the ranges of the uniform distributions to generate the capacities of constraints. As the values of parameters increase, the variance among capacities and the magnitudes of capacities should increase. The figure 5-(i) and (j) show that the gap is affected by the magnitudes and variance of the capacities of constraints. It seems that the gap of the LM gets smaller as the magnitude and variance of the capacities becomes larger. Max_price also affects the magnitude of objective function value; therefore, figure 5-(i) may contain the effects of changes on the magnitude of objective function value. However, figure 5-(j) solely tells that the correlation between the capacities and gap.

We are currently doing experiment for the LRM, and we will report the results in the next version of the paper.

8.2 Time independent models: online and semi-online algorithms

Two design mechanisms

In order to use the threshold type online or semi-online algorithm for the e-coupons strategy problem proposed in sections 5 and 6, we need to define two additional design mechanisms besides the threshold value: one is how to set up the weights of constraints to calculate the combined efficiency, which is called *weight* mechanism, and the other is how to define the sequence among decisions which should be made together; but made one by one, at each decision epoch, which is called *sequence of decisions* mechanism. In this paper, we propose and investigate six options for *weight* mechanism and two options for *sequence of the decisions* mechanism. The following table explains how we define these options. In every option for *weight* mechanism, for simplicity, we normalize the sum of all weights to be always 1.

Table 6 Two design mechanisms and its options

Mechanism	Option name	Description
Weight	*EWall* (Equal Weight all)	Set equal weights among all constraints
	EWbudget (Equal Weight Budget)	Set equal weights only among budget constraints. Weight for the annoyance constraints are all zeros.
	WCall (Weight of Capacity all)	Set weights according to the relative sizes of capacities: $w_i = \frac{b_i}{\sum_{i \in Constraints} b_i}$, where b_i is the capacity of the constraint *i*. The constraints with larger capacities have larger weights.
	WCeach (Weight of Capacity each)	Separately set weights according to the relative sizes of capacities among each constraint type: the budget constraints and annoyance constraints. Sum of weights of each constraint type is 0.5; therefore, the total sum of weights is 1.
	RWCall (Reverse Weight of Capacity all)	Set weights according to the relative sizes of the reverse of capacities, i.e., $w_i = \frac{1/b_i}{\sum_{i \in Constraints} 1/b_i}$. The constraints with larger capacities have smaller weights.
	RWCeach (Reverse Weight of Capacity each)	Separately set weights according to the relative sizes of the reveres of capacities among each constraint type: the budget constraints and annoyance constraints. Sum of weights of each constraint type is 0.5; therefore, the total sum of weights is 1.
Sequence of decisions	*adhoc*	There is a predefined order. For instance, preferred e-coupon lists or customers, etc. According to the order, decisions are made sequentially.
	eo (efficiency order)	According to the efficiencies of decision items, decisions are made sequentially. Decisions with higher efficiencies are considered earlier than those with lower efficiencies.

Online and semi-online algorithms

We solve the same problem with 30 algorithms, which are defined on the below table, and compare the performance between the options of design mechanisms. Besides two design mechanisms, the threshold value also significantly affects the objective value of an online algorithm; therefore, we also solve the same problem using the same algorithm varying the threshold value. Then, we find the best objective function value and the corresponding threshold value which produces that best value.

Performance of an online or semi-online algorithm is measured by the gap between the optimal value of the BM, which is the full information model, and the best objective function value of the algorithm. The subsequent analyses of online and semi-online algorithms are mostly based on this gap performance of the algorithm on a given problem instance, except the sensitivity analysis of threshold value.

Table 7 Online and semi-online algorithms

Weight option	Online or semi-online	*Sequence of Decisions* option	Description
EWall-, EWbudget-, WCall-, WCeach-, RWCall-, or *RWCeach-*	*online-*	*adhoc*	Decision epoch corresponds to the time when a customer arrives. Decisions are related to only the customer who arrives. There is a predefined order in e-coupons. According to the order, e-coupons are considered.
		eo	Decision epoch corresponds to the time when a customer arrives. Decisions are related to only the customer who arrives. An e-coupon with higher efficiency is considered earlier than those with lower efficiencies.
	semi-[14]	*adhoc*	Decision epoch corresponds to the time frame defined in the full information model. Decisions are related to all customers who are in the target area. There are predefined orders in customers and e-coupons. According to the orders, customers are selected first and e-coupons are considered.
		adhoc:customer eo:e-coupon	Decision epoch corresponds to the time frame defined in the full information model. Decisions are related to all customers who are in the target area. There is a predefined order in customers. According to the order, customers are selected first, and e-coupons are considered according to the efficiencies.
		eo	Decision epoch corresponds to the time frame defined in the full information model. Decisions are related to all customers who are in the target area. According to the efficiencies, customers and orders are considered.

[14] In section 6, we propose two different strategies to define the decision epochs of semi-online algorithms: strategy (I) and (II). In our experiments, we implement and test strategy (II) only. In our experiments, the non-adaptive constant threshold semi-online algorithm with strategy (I) is same to the non-adaptive constant threshold online algorithm. Strategy (I) algorithm will be tested when comparing adaptive threshold algorithms in the next version of the paper.

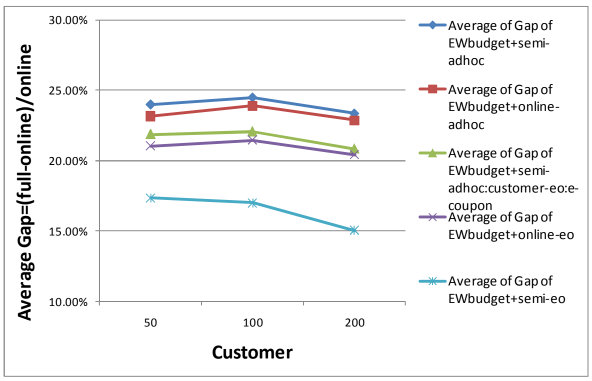
Table 8 Average gap performances of algorithms

	semi-adhoc	*online-adhoc*	*EWall+semi-adhoc: customer-eo: e-coupon*	*online-eo*	*semi-eo*	Average
EWall	27.25%	27.18%	25.32%	24.99%	18.76%	24.70%
EWBudget	23.96%	23.33%	21.63%	21.00%	16.58%	**21.30%**
WCall	24.21%	23.95%	22.17%	21.85%	16.51%	21.74%
WCeach	24.44%	24.11%	22.67%	22.19%	**15.87%**	21.85%
RWCall	71.27%	70.92%	61.17%	60.68%	53.39%	63.49%
RWCeach	37.24%	36.84%	32.59%	32.19%	22.75%	32.32%
Average	34.73%	34.39%	30.92%	30.48%	**23.98%**	30.90%

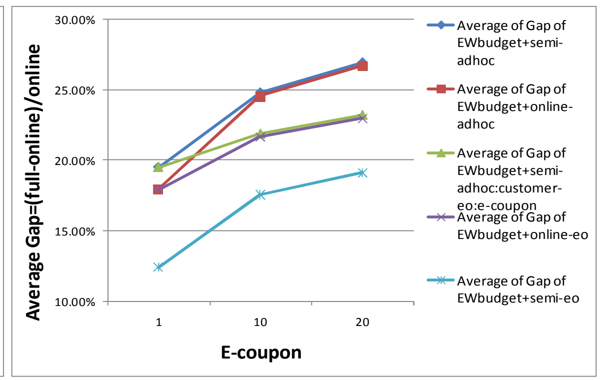
The above table shows that *WCeach+semi-eo* algorithm performs best on average in our experiments. However, among the options of weight, *EWBudget* performs better than *WCEach*, and even *WCall* is better than *WCEach.* Among the options of *sequence of decisions*, *semi-eo* performs best on average. One noticeable thing is that *semi-eo* is always better than any other options of *sequence of decisions* in every weight option; however, *WCeach* is only better than *EWBudget* in *semi-eo.* Further, *EWBudget* is better than any other options of *weight* in every *sequence of decisions* option except *semi-eo*.

Sensitivity analysis of *sequence of decisions*

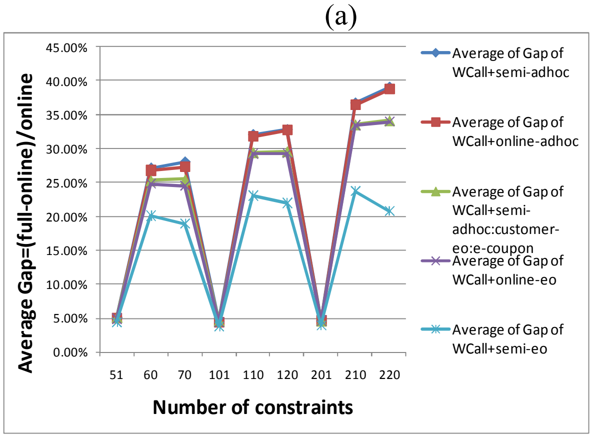
The graphs in figure 6 compare the performances of the options in *sequence of decisions* within the same option in *weight*. Due to the spatial limit of the paper, we do not show all results of every combination of options and parameters; however, we find that, on average, *eo* always performs better than *adhoc,* and *semi-eo* performs best (figure 6-(a), (b), (c), (d), (e), (f), (g), and (h)). On the other hand, we also find that, however, *semi-eo* does not always provide the best solution in every individual instance of the problem. In other words, the average performance of *eo* is better than those of the other options in *sequence of decisions* mechanism; however, *semi-eo* does not necessarily provide the best value in every case. Figure 6-(i) shows that, in most problem instances, *semi-eo* performs best; however, there are some instances where *online-adhoc* or *online-eo* performs best. These results support the Lemma 3 in section 6. Figure 6 also explains the correlation between *sequence of decision*s design mechanism and model parameters. Noticeable thing is that all five algorithms have the similar patterns. In other words, there is no correlation between the options in *sequence of decision*s and the values of parameters.



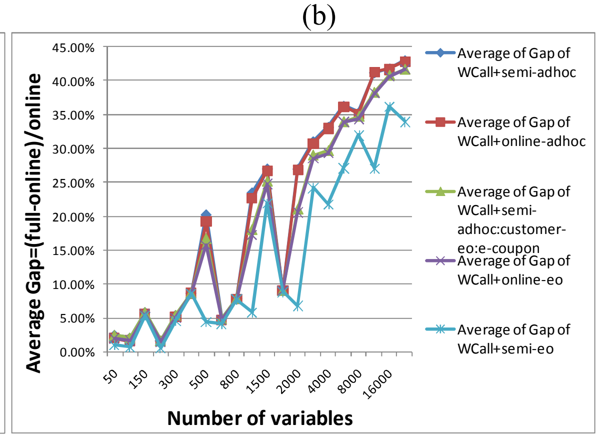
(a)



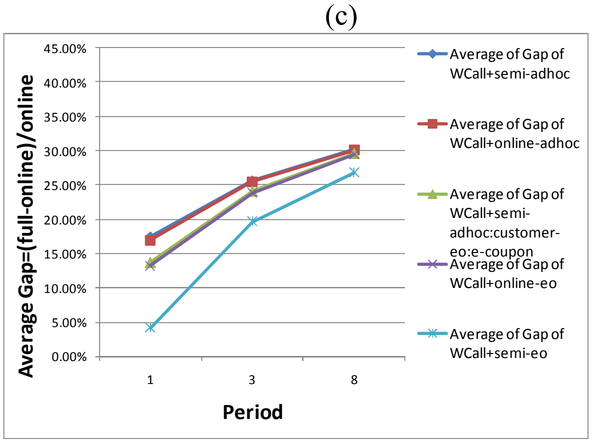
(b)



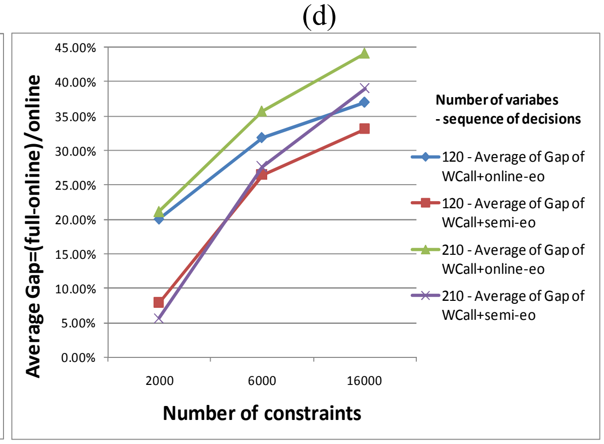
(c)



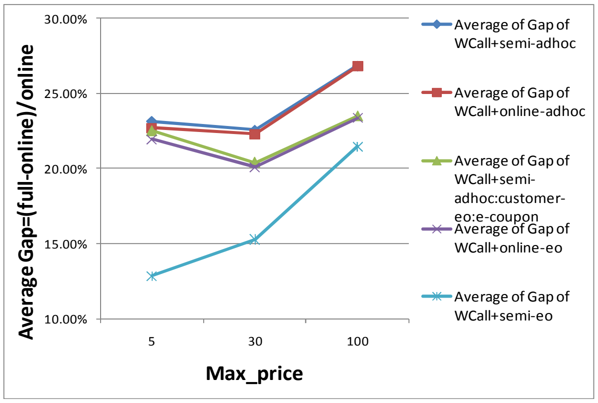
(d)



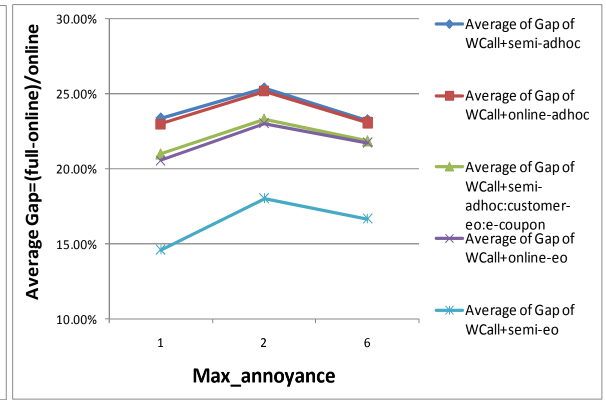
(e)



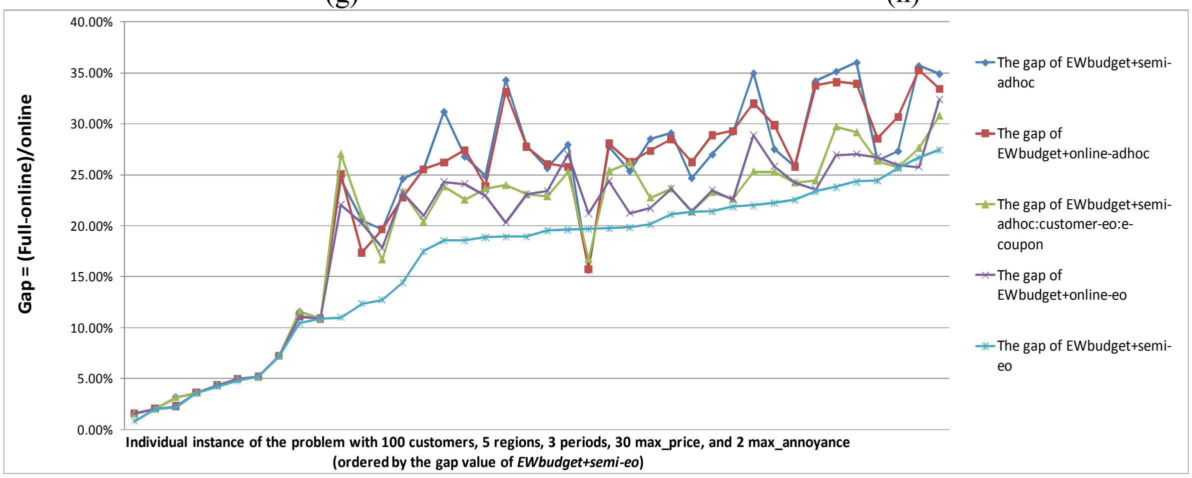
(f)



(g)



(h)

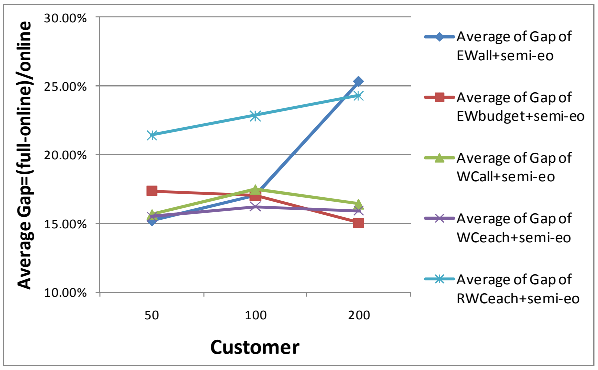


(i)

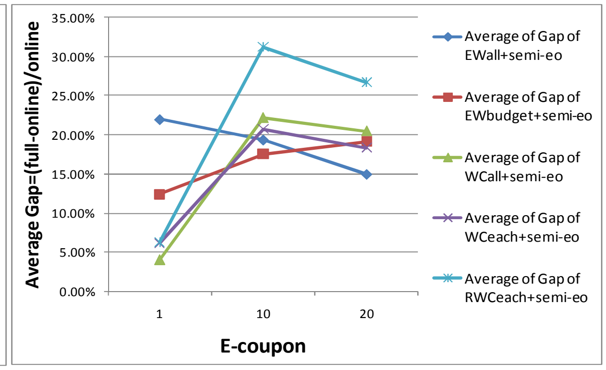
Figure 6 Performances of the online and semi-online algorithms by *sequence of decisions*

Sensitivity analysis of *weight*

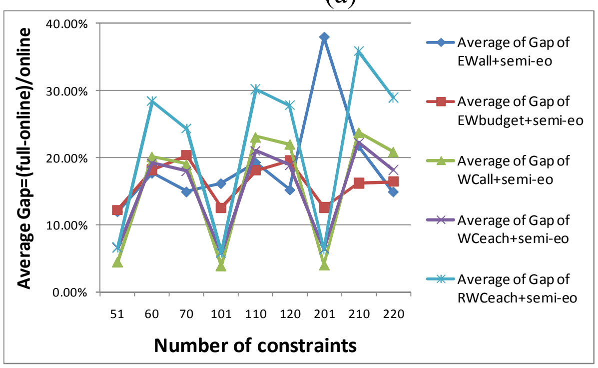
Unlike the mechanism of *sequence of decisions*, figure 7 shows that there is no dominant option for *weight* mechanism, which provides the best solution on average regardless of the values of the parameters. Model parameters do not consistently affect the online algorithms. Table 8 shows that *EWBudget* and *CWEach* perform relatively better than others on average; but, as figure 7 shows, the best option of *weight* varies by the values of the parameters. For instance, 7-(a) shows that when the number of customers is 100, *WCeach+semi-eo* performs best but when the number is 200, *EWBudget+semi-eo* performs best. *RWCall* obviously performs worse than the other options, so we excluded it from the graphs in figure 7. Figure 7-(g) also shows that best option, which performs best, varies by the instance of problem.


Average Gap=(full-online)/online
30.00%
25.00%
20.00%
15.00%
10.00%
50
100
200
Customer
Average of Gap of EWall+semi-eo
Average of Gap of EWbudget+semi-eo
Average of Gap of WCall+semi-eo
Average of Gap of WCeach+semi-eo
Average of Gap of RWCeach+semi-eo

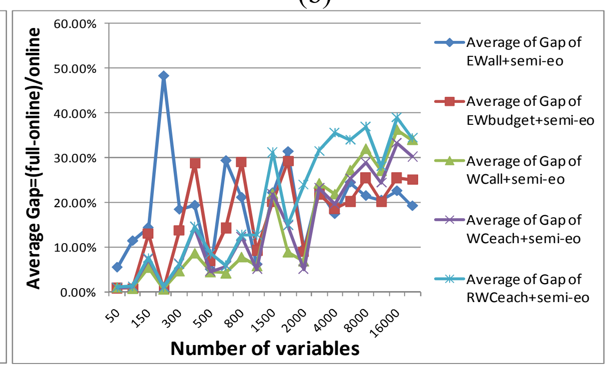
(a)


Average Gap=(full-online)/online
35.00%
30.00%
25.00%
20.00%
15.00%
10.00%
5.00%
0.00%
1
10
20
E-coupon
Average of Gap of EWall+semi-eo
Average of Gap of EWbudget+semi-eo
Average of Gap of WCall+semi-eo
Average of Gap of WCeach+semi-eo
Average of Gap of RWCeach+semi-eo

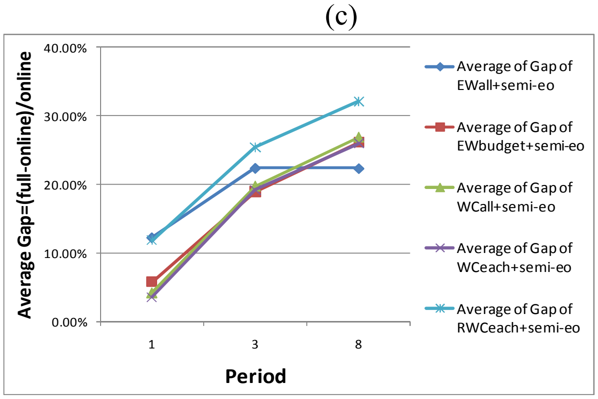
(b)


Average Gap=(full-online)/online
40.00%
30.00%
20.00%
10.00%
0.00%
51
60
70
101
110
120
201
210
220
Number of constraints
Average of Gap of EWall+semi-eo
Average of Gap of EWbudget+semi-eo
Average of Gap of WCall+semi-eo
Average of Gap of WCeach+semi-eo
Average of Gap of RWCeach+semi-eo

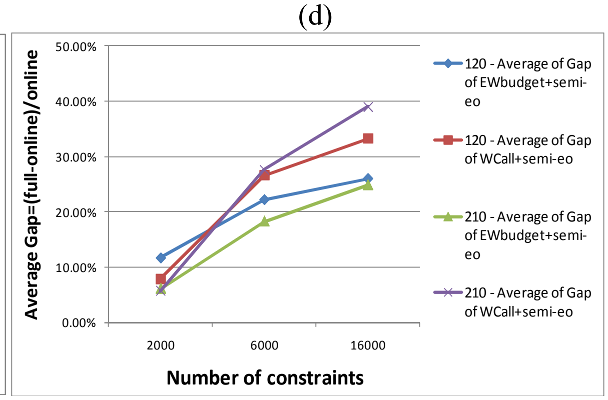
(c)


Average Gap=(full-online)/online
60.00%
50.00%
40.00%
30.00%
20.00%
10.00%
0.00%
50
150
300
500
800
1500
2000
4000
8000
16000
Number of variables
Average of Gap of EWall+semi-eo
Average of Gap of EWbudget+semi-eo
Average of Gap of WCall+semi-eo
Average of Gap of WCeach+semi-eo
Average of Gap of RWCeach+semi-eo

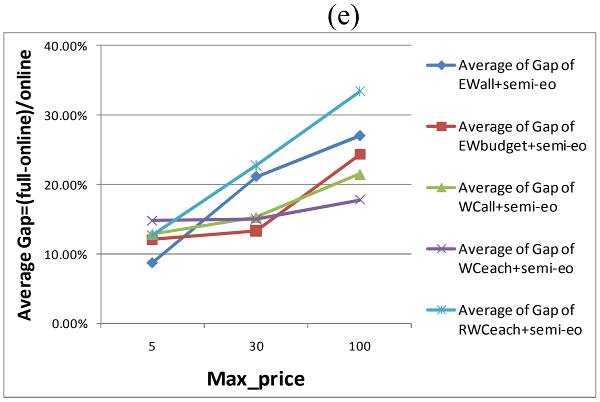
(d)


Average Gap=(full-online)/online
40.00%
30.00%
20.00%
10.00%
0.00%
1
3
8
Period
Average of Gap of EWall+semi-eo
Average of Gap of EWbudget+semi-eo
Average of Gap of WCall+semi-eo
Average of Gap of WCeach+semi-eo
Average of Gap of RWCeach+semi-eo

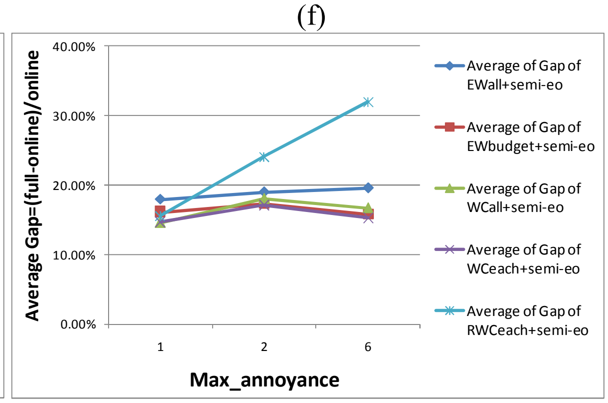
(e)


Average Gap=(full-online)/online
50.00%
40.00%
30.00%
20.00%
10.00%
0.00%
2000
6000
16000
Number of constraints
120 - Average of Gap of EWbudget+semi-eo
120 - Average of Gap of WCall+semi-eo
210 - Average of Gap of EWbudget+semi-eo
210 - Average of Gap of WCall+semi-eo

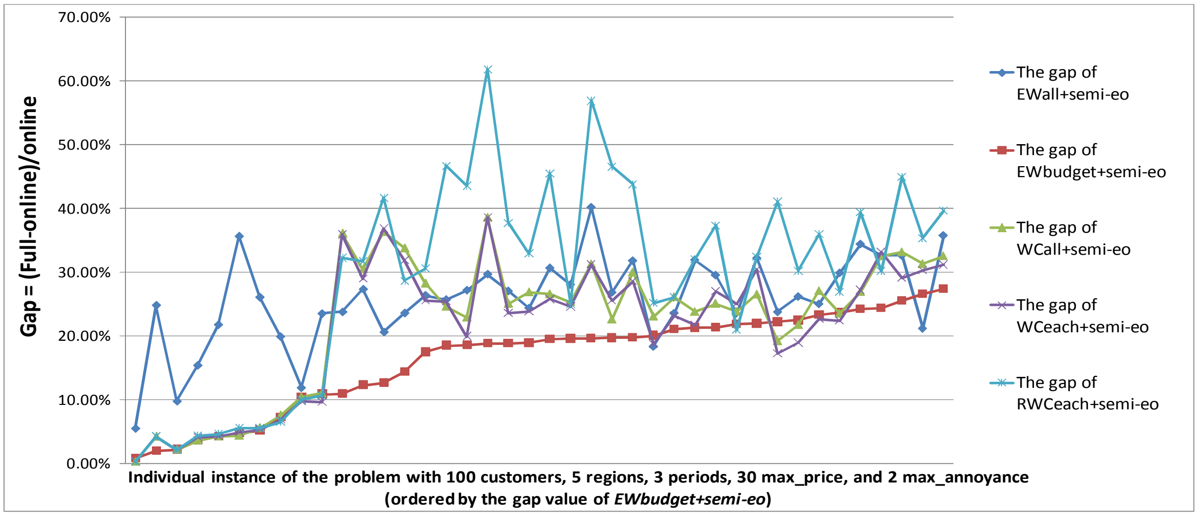
(f)


Average Gap=(full-online)/online
40.00%
30.00%
20.00%
10.00%
0.00%
5
30
100
Max_price
Average of Gap of EWall+semi-eo
Average of Gap of EWbudget+semi-eo
Average of Gap of WCall+semi-eo
Average of Gap of WCeach+semi-eo
Average of Gap of RWCeach+semi-eo

(g)


Average Gap=(full-online)/online
40.00%
30.00%
20.00%
10.00%
0.00%
1
2
6
Max_annoyance
Average of Gap of EWall+semi-eo
Average of Gap of EWbudget+semi-eo
Average of Gap of WCall+semi-eo
Average of Gap of WCeach+semi-eo
Average of Gap of RWCeach+semi-eo

(h)



(i)

Figure 7 Performances of the online and semi-online algorithms by *weight*

Sensitivity analysis of model parameters

In order to investigate the impact of the model parameters on the performances of each online and semi-online algorithm, we build the regression model as flows.

$$Performace\ of\ algorithm\ (Gap) = \beta_0 + \beta_0 Customer + \beta_1 E\_coupon + \beta_2 Period + \beta_3 Max\_price + \beta_4 Max\_annoaynce + \beta_5 Customer * E\_coupon + \beta_5 Customer * E\_coupon * Period + \beta_5 Max\_price * Max\_annoyance + \varepsilon_i$$

The following table reports the coefficient of the regression model (***β***). As figure 7-(a) represents, the number of customers affects the performance of *EW* and *RWCall* algorithms negatively (i.e., positive slopes), while it does not influence WC and RWCeach algorithms. On the other hand, the performances of most algorithms increase with the number of e-coupons and periods. The number of customers, e-coupons, and periods affect the number of variables and constraints; therefore, the number of variables and constraints should affect the performances of algorithms negatively, overall. Figure 6-(c) and (d) and figure 7-(c) and (d) show this overall relationship.

The impacts of max_annoyance and max_price on performance vary by algorithm. In general, the gap increases with max_price and max_annoyance except that of *EWbudget* and *CWeach* algorithms. Experimental results show that *EWbuget* seems to work better as max_annoyance increases and *WCeach* seems to work better as max_price increases. As we explained earlier, the bigger max_annoyance means the larger variance among the annoyance numbers of customers. Max_price plays a similar role to max_annoyance, which increases the variance among the capacities of budget constraints. The results imply that *EWbudget* and *CWeach* algorithms work better as capacities gets non-homogeneous.

The location of a customer should be one at a given point of time. Therefore, the location of a customer is a dependent variable of time and customer. Therefore, algorithms are not affected by the number of regions, because it does not increase the problem size (i.e., number of variables or constraints). If we incorporate the network capacity constraints in our models and assume that capacities vary by region, then the number of regions will affect the problem complexity and will negatively affect the performance of online and semi-online algorithms as the number of constraints does.

Table 9 Results of Linear regressions

	EWall					EWbudget				
	SA	OA	SAE	OE	SE	SA	OA	SAE	OE	SE
I	-.0517**	-.0561**	-.0500**	-.0546**	-.1011**	.0806**	.0489**	.0854**	.0532**	-.0156
C	.0017**	.0017**	.0017**	.0017**	.0016**	.0002*	.0002**	.0002*	.0002**	.0000
E	.0098**	.0095**	.0085**	.0082**	.0063**	.0050**	.0057**	.0034**	.0041**	.0041**
P	.0008	.0011	.0026**	.0031**	.0120**	.0275**	.0279**	.0294**	.0298**	.0316**
MP	.0024**	.0025**	.0021**	.0021**	.0018**	.0001	.0003*	-.0004**	-.0002	.0004**
MA	.0096**	.0103**	.0098**	.0103**	.0090**	-.0181**	-.0151**	-.0189**	-.0156**	-.0133**
C×E	-.0001**	-.0001**	-.0001**	-.0001**	-.0001**	.0000**	.0000**	.0000**	.0000**	.0000
MP×MA	-.0001**	-.0001**	-.0001*	-.0001*	.0000	.0004**	.0003**	.0004**	.0004**	.0004**
C×E×P	.0000**	.0000**	.0000**	.0000**	.0000	.0000**	.0000**	.0000**	.0000**	.0000**
	WCall					WCeach				
	SA	OA	SAE	OE	SE	SA	OA	SAE	OE	SE
I	.0200**	.0115	.0203**	.0117	-.0598**	.0659**	.0528**	.0671**	.0519**	-.0228**
C	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
E	.0086**	.0084**	.0075**	.0071**	.0058**	.0078**	.0079**	.0068**	.0065**	.0046**
P	.0143**	.0150**	.0160**	.0166**	.0217**	.0160**	.0162**	.0172**	.0179**	.0236**
MP	.0006**	.0006**	.0003**	.0003**	.0010**	-.0002**	-.0001	-.0005**	-.0004**	.0004**
MA	.0014	.0020	.0027*	.0039**	.0078**	-.0029*	-.0018	-.0015	.0004	.0051**
C×E	.0000**	.0000**	.0000**	.0000**	.0000	.0000**	.0000**	.0000**	.0000**	.0000*
MP×MA	-.0001**	-.0001**	.0000*	.0000*	-.0001**	.0000	.0000	.0000*	.0000	.0000**
C×E×P	.0000*	.0000	.0000**	.0000**	.0000**	.0000**	.0000**	.0000	.0000	.0000**
	RWCall					RWCeach				
	SA	OA	SAE	OE	SE	SA	OA	SAE	OE	SE
I	-.3921**	-.4046**	-.2931**	-.3193**	-.3698**	-.1101**	-.1260**	-.0790**	-.0969**	-.1317**
C	.0043**	.0043**	.0042**	.0044**	.0043**	.0000	.0001	.0000	.0001	.0000
E	.0084**	.0087**	.0055**	.0058**	.0026	.0119**	.0118**	.0069**	.0071**	.0046**
P	.0437**	.0452**	.0401**	.0416**	.0480**	.0294**	.0300**	.0301**	.0305**	.0346**
MP	.0036**	.0035**	.0029**	.0029**	.0026**	.0011**	.0012**	.0008**	.0009**	.0009**
MA	.1144**	.1145**	.0835**	.0848**	.0899**	.0323**	.0343**	.0241**	.0260**	.0148**
C×E	-.0001**	-.0001**	-.0001**	-.0001**	-.0001**	.0001**	.0001**	.0001**	.0001**	.0001**
MP×MA	.0001	.0001	.0001	.0001	.0001*	.0004**	.0004**	.0003**	.0003**	.0004**
C×E×P	.0000	.0000*	.0000	.0000	.0000	.0000**	.0000**	.0000**	.0000**	.0000**

SA = semi-adhoc, OA= online-adhoc, SAE = semi-adhoc:customer-eo:e-coupon, OE=online-eo, SE=semi-eo

I= Intercept, C= Customer, E= E-coupon, P= Period, MA= Max_price, MA= Max_annoyance

** p-value < .01, * p-value <.05

Sensitivity analysis of threshold value

Because we use normalized weights in every *weight* option, possible threshold values for a given problem instance are bounded. In our problem, the required capacity of every item (i.e., the coefficient of constraints) is between 1 and max_price, and its profit (i.e., the coefficient of objective function) is also between 1 and max_price. Therefore, the range of possible efficiencies is [1/max_price, max_price], and the range of threshold values is also [1/max_price, max_price] because of the normalized weights. However, the probability of an item satisfying the threshold value gets much smaller as the threshold value gets closer to the upper bound. Therefore, based on the result on sample test on randomly selected instances of different problems, without loss of generality, we reduce the range of threshold values as [1/max_price, max_price/10] in our experiment. Then, we divide this range into 200 constant intervals and solve the same instance of the problem for 200 times, using these 200 threshold values, for every algorithm.

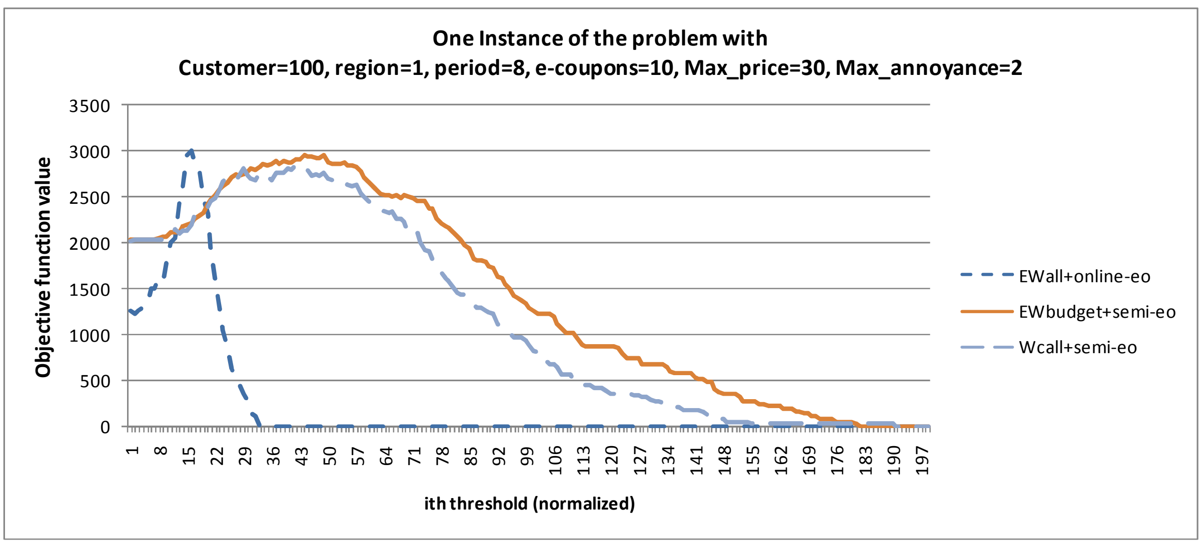
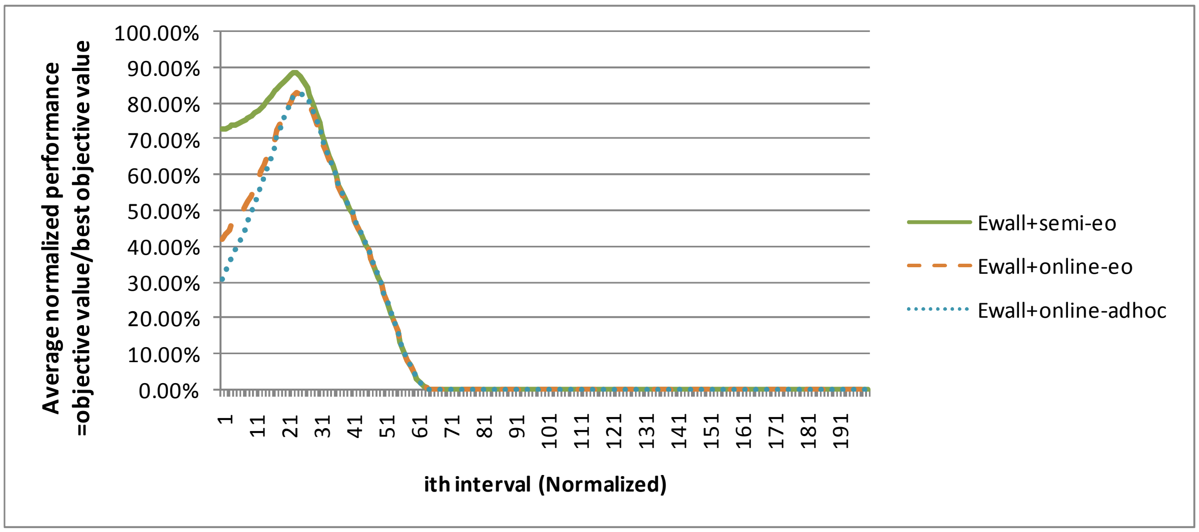
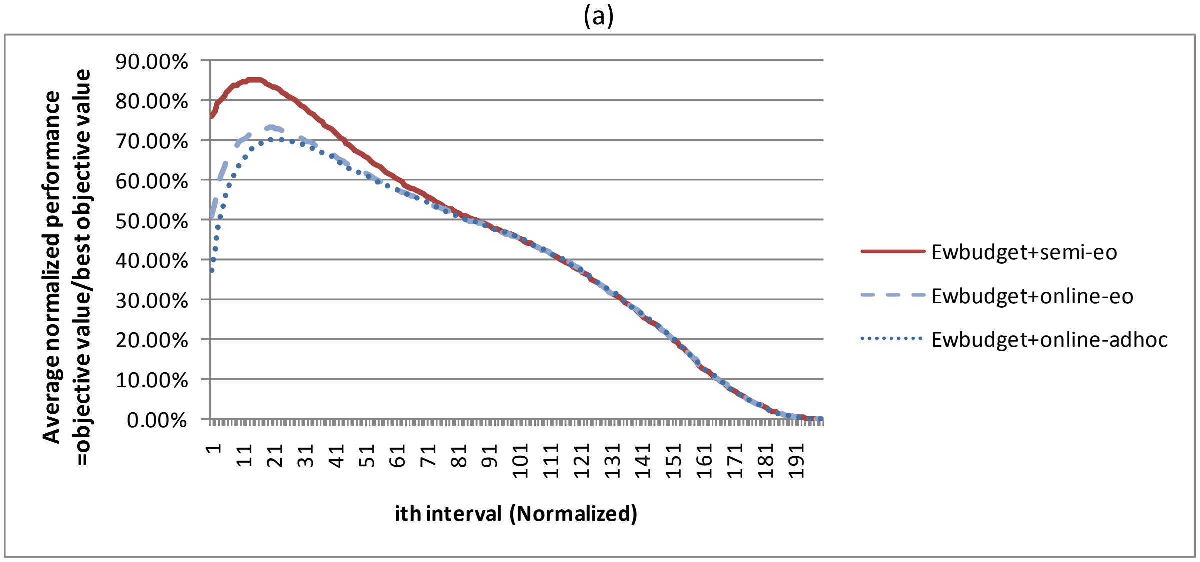


Figure 8 Performance of the online and semi-online algorithms by threshold value

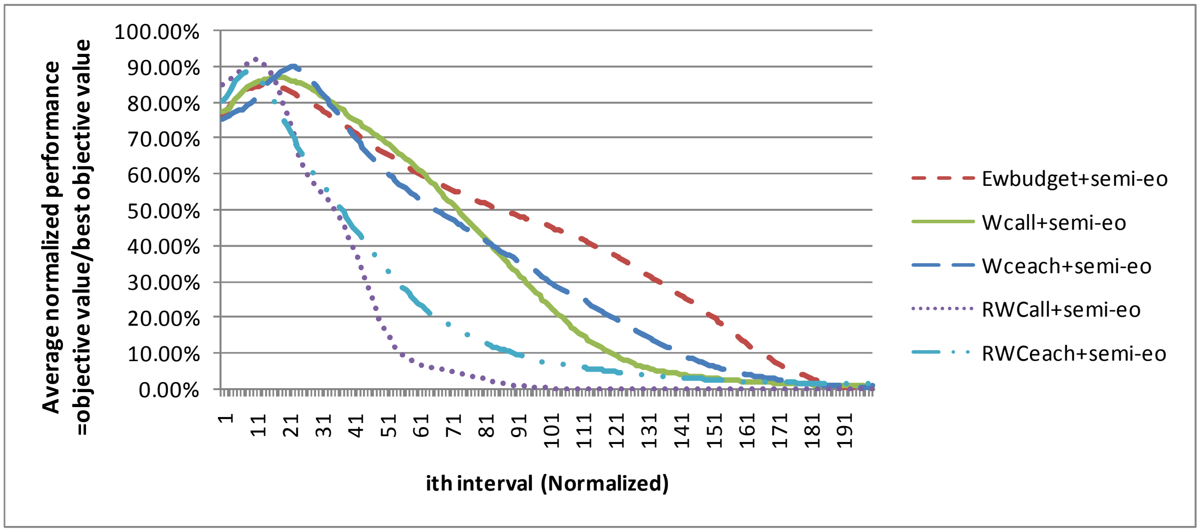
Figure 8 shows the example of one instance solved by several online and semi-online algorithms with varying the threshold value. X axis represents the ordinal intervals rather than real threshold values. In the example of figure 9, the range of threshold values is [1/30,3], and the constant interval is (3-1/30)/200=.01483. According to the options of algorithm, the best objective function value varies, and the threshold value which produces that best objective function value also differs. In addition, according to the options of the algorithm, the shapes of the curvatures are also different. It means that the sensitivity of the threshold value differs by the options of algorithm. For example, in figure 8, *EWall+online-eo* have steeper slopes than *EWbudget+semi-eo* and *CWall+semi-eo.* It means that, in *EWall+online-eo* algorithm, if we mistakenly choose a threshold value far away from the best threshold value, then we will get a much worse performance than *EWbudget+semi-eo* and *CWall+semi-eo* algorithms for this instance.



(a)



(b)



(c)

Figure 9 Sensitivity of threshold value

Figure 9 shows that the sensitivity of threshold value on the average performances of algorithms. According to the max_price, the range of threshold value varies; therefore, we normalize the range using 200 intervals to compare the algorithms with different threshold value ranges. Similarly, the magnitude of objective function value varies by problem parameter and instance, and the best objective function value of a given online algorithm is also different instance by instance. Therefore, for the same reason to normalize the threshold values, we also normalize the performance. Normalized performance of an algorithm with a given threshold value for a given problem instance is defined as $\frac{\text{Objective function value of the algorithm with the threshold value}}{\text{Best objective function value that the algorithm can get using differet threshold values}}$. Because the online and semi-online algorithms inherently have a limit to find the optimal solution, in order to extract the effects of threshold values, we use the best objective function value which can be obtained with a given algorithm instead of the optimal value which may be not obtainable by the given algorithm. We solve the same problem instance with the same algorithm but varying the threshold value; therefore, we can obtain this ratio in every combination of threshold values, options of algorithm, and problem instances. In figure 9, Y axis represents the average of normalized performance at each normalized threshold value.

The sensitivity of threshold value seems not to differ by *sequence of decisions*. Figure 9-(a) and (b) show that algorithms in the same *weight* option have similar curvatures. However, on the other hand, sensitivity seems to vary by *weight* (figure 9-(c)). Among options in *weight*, *EWbudget* has the smoothest slope (figure 9-(c)); therefore, in terms of the sensitivity of threshold value, *EWbudget* is the most preferable option in *weight*.

Figure 9 also provides the guideline to set the threshold value for a given algorithm. It may not provide the threshold value which performs best in every case, but it provides a better threshold value which performs better than any other values on average.

Based on the findings from computational experiments, *EWbudget+semi-eo* or *WCeach+semi-eo* algorithm looks better than others in terms of the performance and robustness. If the threshold values are not sure, *EWbudget+semi-eo* seems slightly more preferable, while the average performance of *WCeach+semi-eo* is better.

9 Managerial insights

Due to the recent advances of technologies, the real time location information of customers now becomes available to the advertisers and plays an essential role in the contextual marketing. At the same time, the advertisers in LBA are now facing a new decision problem finding the most profitable way to send location-aware advertisements to customers, which we call e-coupons strategy problem. We study the several analytical models of e-coupons strategy problem. When the full information can be used, the advertisers are encouraged to use a single-price strategy where all prices are same. We show that the problem with the same prices can be solved optimally with relative ease. Otherwise, the linear relaxation approximation can be alternative solution because it finds the solution which is very close to the optimal solution. When the full information is not available and the decisions are to be made dynamically, the advertisers are strongly encouraged to use the semi-online algorithms, where decisions can be deferred within a given time window, and *EWbudget* or *WCeach* for its *weight* mechanism of the algorithm. Our study finds the advantages of semi-online algorithms in e-coupons strategy problem for the first time, and the semi-algorithms can be applied in other dynamic Knapsack problems such as revenue management.

10 Conclusions and future research

We study the e-coupons strategy problem from the point of view of the super mobile. The problem is reduced to a well known optimization problem, the MKP. We propose several models including the full information, online, semi-online, and dynamic and stochastic model to address the different situations which may occur in practice. Even though MKP belongs to the NP-hard class of problems, as we show, the linear relaxation performs well in our problems; therefore, the full information model seems to have merits in the practical sense. However, in reality, it is also very difficult to predict the mobility of customers; so, the full information model may not be appropriate in some cases. In order to cope with this difficulty in using the full information model in reality, we propose several dynamic models. As the full information MKP is very hard to solve optimally, the corresponding online MKP is also difficult to solve well. In addition, because the previous literature mostly deals with the single constraint Knapsack problem, there is little known about how to solve the online MKP. Therefore, we propose several simple online algorithms and test their performance through computational experiments. Through the performance and sensitivity analysis, we find that the simple equal weights algorithms such as *EWall or EWbudget* perform better than the other sophisticated algorithms using non-equal weights algorithms such as WC and Reverse Weight of Capacity (RWC) algorithm; however, we also find that WC and RWC algorithms perform well in the case of non-homogeneous capacities. Experimental results also provide the guideline on how to select the most appropriate algorithm and corresponding threshold value which may produce better solution on average.

One unique characteristic of the e-coupons strategy problem compared to the traditional online Knapsack problems, where the decision is made upon the arrival of item, is that decisions may be given within a time window. Exploiting this unique characteristic, we propose several semi-online algorithms and show that they perform better than other pure online algorithms on average.

Future directions of this research are as follows. First, in the full information model, more investigation on the ways to solve the large problems optimally needs to be conducted. In the online and semi-online models, analytical models to support the experimental findings need to be developed. In the dynamic and stochastic model, the model needs to be solved analytically, and the way to solve the large size problems needs to be proposed and tested through experiments.

References

Babaioff, M., Hartline, J. D., & Kleinberg, R. D. (2008). *Selling banner ads: online algorithms with buyback*. Paper presented at the Workshop on Ad Auctions.

Babaioff, M., Immorlica, N., Kempe, D., & Kleinberg, R. (2007). A knapsack secretary problem with applications. *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques, 4627*, 16-28.

Barwise, P., & Strong, C. (2002). Permission-based Mobile Advertiseing. *Journal of Interactive Marketing, 16*(1), 14-24.

Chakrabarty, D., Zhou, Y., & Lukose, R. (2007). *Budget Constrained Bidding in Keyword Auctions and Online Knapsack Problems*. Paper presented at the WWW.

Ducoffe, R. H. (1996). Advertising value and advertising on the web. *Journal of Advertising Research, 36*(5), 21-35.

Dyer, M. E., & Frieze, A. M. (1989). Probabilistic Analysis of the Multidimensional Knapsack-Problem. *Mathematics of Operations Research, 14*(1), 162-176.

Forrester Research (2006). *Is The US Ready For Mobile Marketing?*

Forrester Research (2007a). *Best Practices: US Mobile Marketing*.

Forrester Research (2007b). *Telcos Should Look To Location-Based Services For Growth*: Forrester Research.

Freville, A. (2004). The multidimensional 0-1 knapsack problem: An overview. *European Journal of Operational Research, 155*(1), 1-21.

Gratton, E. (2002). M-commerce: The Notion of Consumer Consent in Receiving Location-Based Advertising. *Canadian Journal of Law and Technology, 1*(3), 59-77.

Haghirian, P., Madlberger, M., & Tanuskova, A. (2005). *Increasing Advertising Value of Mobile Marketing– An Empirical Study of Antecedents.* Paper presented at the 38th Hawaii International Conference on System Sciences, Hawaii.

Ho, S. Y., & Kwok, S. H. (2002). The attraction of personalized service for users in mobile commerce: an empirical study. *ACM SIGecom Exchanges, 3*(4), 10-18.

Iwama, K., & Zhang, G. (2007). Optimal resource augmentations for online knapsack. *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques, 4627*, 180-188.

Kan, A., Stougie, L., & Vercellis, C. (1993). A Class of Generalized Greedy Algorithms for the Multi-Knapsack Problem. *Discrete Applied Mathematics, 42*(2-3), 279-29.

Kellerer, H., Kotov, V., Speranza, M. C., & Tuza, Z. (1997). Semi on-line algorithms for the partition problem. *Operations Research Letters, 21*(5), 235-242.

Kellerer, H., Pferschy, U., & Pisinger, D. (2004). *Knapsack Problems*: Springer.

Kleinrock, L. (1976). *Queueing Systems* (Vol. I): John Wiley & Sons.

Kleywegt, A. J., & Papastavrou, J. D. (1998). The dynamic and stochastic knapsack problem. *Operations Research, 46*(1), 17-35.

Kleywegt, A. J., & Papastavrou, J. D. (2001). The dynamic and stochastic knapsack problem with random sized items. *Operations Research, 49*(1), 26-41.

Lu, L. L., Chiu, S. Y., & Cox, L. A. (1999). Optimal project selection: Stochastic knapsack with finite time horizon. *Journal of the Operational Research Society, 50*(6), 645-65.

Lueker, G. S. (1998). Average-case analysis of off-line and on-line Knapsack problems. *Journal of Algorithms, 29*(2), 277-305.

Marchettispaccamela, A., & Vercellis, C. (1995). Stochastic Online Knapsack-Problems. *Mathematical Programming, 68*(1), 73-104.

Meanti, M., Kan, A., Stougie, L., & Vercellis, C. (1990). A Probabilistic Analysis of the Knapsack Value Function. *Mathematical Programming, 46*(2), 237-247.

Nemhauser, G. L., & Wolsey, L. A. (1988). *Integer and Combinatorial Optimization*.

Noga, J., & Sarbua, V. (2005). An online partially fractional knapsack problem. *8th International Symposium on Parallel Architectures, Algorithms and Networks, Proceedings*, 108-112.

Pak, K., & Dekker, R. (2004). Cargo Revenue Management: Bid Price for a 0-1 Multi Knapsack Problem.

Papastavrou, J. D., Rajagopalan, S., & Kleywegt, A. J. (1996). The dynamic and stochastic knapsack problem with deadlines. *Management Science, 42*(12), 1706-1718.

Ross, K. W., & Tsang, D. H. K. (1989). The Stochastic Knapsack-Problem. *IEEE Transactions on Communications, 37*(7), 740-747.

Seiden, S., Sgall, J., & Woeginger, G. (2000). Semi-online scheduling with decreasing job sizes. *Operations Research Letters, 27*(5), 215-221.

Tripathi, A. K., & Nair, S. K. (2006). Mobile advertising in capacitated wireless networks. *IEEE Transactions on Knowledge and Data Engineering, 18*(9), 1284-1296.

Tripathi, A. K., & Nair, S. K. (2007). Narrowcasting of wireless advertising in malls. *European Journal of Operational Research, 182*(3), 1023-1038.

van Slyke, R., & Young, Y. (2000). Finite horizon stochastic knapsacks with applications to yield management. *Operations Research, 48*(1), 155-172.

Yang, D., Hiroshi, I., Kazuo, I., & Naoki, K. (1993). How to Treat Delete Requests in Semi-Online Problems. *Lecture Notes in Computer Science, 762*, 48-57.