Computers & Operations Research 48 (2014) 20–30

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Efficient solution of a class of location–allocation problems with stochastic demand and congestion

Navneet Vidyarthi [a,*], Sachin Jayaswal [b]

[a] Department of Supply Chain and Business Technology Management, John Molson School of Business, Concordia University, Montreal, QC H3G 1M8, Canada
[b] Production and Quantitative Methods, Indian Institute of Management, Vastrapur, Ahmedabad, Gujarat 380 015, India

ARTICLE INFO

Available online 12 March 2014

Keywords:
Location–allocation
Service system design
Queueing
Stochastic demand
Congestion
Constraint generation method

ABSTRACT

We consider a class of location–allocation problems with immobile servers, stochastic demand and congestion that arises in several planning contexts: location of emergency medical clinics; preventive healthcare centers; refuse collection and disposal centers; stores and service centers; bank branches and automated banking machines; internet mirror sites; web service providers (servers); and distribution centers in supply chains. The problem seeks to simultaneously locate service facilities, equip them with appropriate capacities, and allocate user demand to these facilities such that the total cost, which consists of the fixed cost of opening facilities with sufficient capacities, the access cost of users' travel to facilities, and the queuing delay cost, is minimized. Under Poisson user demand arrivals and general service time distributions, the problem is set up as a network of independent M/G/1 queues, whose locations, capacities and service zones need to be determined. The resulting mathematical model is a non-linear integer program. Using simple transformation and piecewise linear approximation, the model is linearized and solved to ϵ-optimality using a constraint generation method. Computational results are presented for instances up to 400 users, 25 potential service facilities, and 5 capacity levels with different coefficients of variation of service times and average queueing delay costs per customer. The results indicate that the proposed solution method is efficient in solving a wide range of problem instances.

1. Introduction

Problems arising in several planning contexts require deciding: (i) the location of service facilities and their capacities; and (ii) allocation of service zones to the located service facilities. Examples include location of emergency service facilities such as medical clinics and preventive health care facilities [25,26,27]; stores and service centers; bank branches and automated banking machines [1,11,23]; automobile emission testing stations [11]; web service providers' facilities [3]; proxy/mirror servers in communication networks [24] and distribution centers in supply chains [17,22]. All the above examples are characterized by servers (medical clinics, bank branches, distribution centers, etc.) that are immobile in that the customers need to travel to the service facilities to avail of their services, as opposed to the servers traveling (mobile servers) to the customers' site in response to calls for their services. Such problems are generally also characterized by random nature of service calls (demand arrivals) and their service requirements (service times). These problems are commonly known in the literature as facility location problems with immobile servers, stochastic demand and congestion [8]. They are also termed as service system design problems with stochastic demand and congestion [4–6,13]. Literature review for this class of problems is provided by Berman and Krass [8] and Boffey et al. [10].

For facility location problems with stochastic demand and congestion, the following two factors are important: (i) the costs of providing service; and (ii) the quality of service, with an objective generally requiring a balance between the two. The costs of providing service are related to the fixed cost of opening/operating the service facilities and the cost of accessing these facilities by the users. The service quality, on the other hand, is often measured in terms of: (i) the average number of users waiting for service; (ii) average waiting time per user; or (iii) the probability of serving a user within a time limit [13]. Balance between service costs and service quality is commonly achieved in the literature using a combination of the total cost of opening and accessing facilities and the cost associated with waiting customers, which is minimized in the objective function [4,5,11,13,23]. Others in the literature minimize the cost of providing service subject to a minimum threshold on the service quality, where the service quality may be defined in one of the ways described above [18,19,21].

* Corresponding author. Tel.: +1 514 848 2424x2990; fax: +1 514 848 2824.
E-mail addresses: navneetv@jmsb.concordia.ca (N. Vidyarthi), sachin@iimahd.ernet.in (S. Jayaswal).

http://dx.doi.org/10.1016/j.cor.2014.02.014
0305-0548

In this paper, we use the former of the two approaches described above, i.e., we consider minimization of the total cost, which includes the cost of opening and accessing facilities and the cost associated with waiting customers. It is worth noting that due to the complexity of the underlying problem, most papers in this category make assumptions such as (i) either the number or capacity of the facilities (or both) is fixed; (ii) the demand arrival process is Poisson; and (iii) the service times follow an exponential distribution (see [1,4,13,19,23] and references therein). Despite these simplifying assumptions, the techniques proposed to date to solve the problem, with the exception of Elhedhli [13], are either approximate or heuristic based.

The contribution of this paper is twofold. First, by assuming a general distribution for the service times at facilities, as opposed to exponential distribution, we present a more generalized model of the problem than available in the extant literature. More specifically, our proposed model seeks to determine the minimum cost system-optimal configuration (location of service facilities and their capacity levels as well as the allocation of service zones to these facilities) of a service system under Poisson arrivals and general service time distribution, where the total cost consists of the costs of opening and accessing service facilities and the cost associated with waiting customers. As discussed above, the problem, even with the simplifying assumption of exponential service time distribution, is too difficult to solve using exact methods. The proposed model, with general service time distribution, is even more challenging to solve. So, our second contribution lies in the exact (ϵ-optimal) solution method that we propose to solve our model. Our proposed solution method is based on a simple transformation and piecewise linearization of our non-linear integer programming (IP) model, which is solved to optimality (or ϵ-optimality) using a constraint generation algorithm.

The remainder of the paper is organized as follows. In Section 2, we describe the problem setting, followed by its non-linear IP model. Section 3 describes the transformation and the piecewise linearization approach for the non-linear IP model. To solve the linearized model, we present a constraint generation based solution approach in Section 4. Computational results are reported in Section 5. Section 6 concludes with some directions for future research.

2. Problem formulation

Consider a set of user nodes, each indexed by $i \in I$ whose demand for service occurs continuously over time according to an independent Poisson process with rate λ_i. We consider a *directed choice* environment, where users are assigned to facilities, each indexed by $j \in J$, by a central decision maker. This is applicable, for example, in the case of a "virtual call center" consisting of geographically dispersed telephone call centers, routing of calls to which is centrally determined (see [11], and references therein). The directed choice model is also applicable in the case of medical clinics and preventive health care facilities; automobile emission testing stations; and distribution centers in supply chains, if users' choice can be influenced through imposition of tolls or differential service fees. Later, we show how our model can be adapted to the *user choice* environment where the choice of service facility is not dictated or influenced by the central authority but exercised solely by the users. Recent studies of models with directed choice settings include Aboolian et al. [2] whereas models with user choice settings can be found in Baron et al. [7,27], and references therein.

We assume that users from any node are entirely assigned to a single service facility, where each facility operates with an infinite buffer to accommodate users waiting for service. If x_{ij} is a binary variable that equals 1 if the demand for service from user node i is satisfied by facility j, and 0 otherwise, then the aggregate demand arrival rate at facility j, as a result of the superposition of Poisson processes, also follows a Poisson process with mean $\Lambda_j = \sum_{i \in I} \lambda_i x_{ij}$ [15].

There are two approaches to model the capacity of a service facility [2,7]. One is to model the given service facility as a single server with flexible service capacity μ, which can be adjusted either continuously or in discrete steps. The second approach is to assume multiple parallel servers, each with a given single capacity level μ. In this case, the decision variable is the appropriate number of servers to be installed at the given service facility. In the case of call centers, automated banking machines , automobile emission testing stations, or distribution centers, where adding capacity would imply adding a call center employee, a banking machine, a testing station, or a loading/unloading dock respectively, the multiple server model is more appropriate. However, in cases where it is not clear what a "server" represents (e.g. hospitals or emergency medical clinics), and the capacity can be increased in a variety of ways (by improving patient flow or technology; adding nurses, doctors, support staff or examination rooms, etc.), single server model would be suitable. In this paper, we adopt the former approach, and model each facility as a single server with multiple capacity levels, from which one capacity level is to be selected, if the facility is opened. We take this approach primarily for tractability of the resulting model. However, a single server model may still be a good approximation of a multi-sever facility if the utilization of the service facility is reasonably high. This is because under reasonably high system utilization, a system with s parallel servers, each with capacity μ, is known to perform similar to a single server with capacity $s\mu$.

For each service facility, we allow the option of selecting one of the several capacity levels $\mu_{jk}, k \in K$ with fixed cost f_{jk} (amortized over the planning period). Let y_{jk} be a binary variable that equals 1 if facility at site j is open and equipped with a capacity level $k \in K$, 0 otherwise. Further, assume that the service times at any facility j are independent and identically distributed with a mean $1/\mu_{jk}$ and variance σ_{jk}^2 if it is equipped with a capacity level k. Each facility j is thus modeled as an $M/G/1$ queue with a service rate $\mu_j = \sum_{k \in K} \mu_{jk} y_{jk}$ and variance in service times given by $\sigma_j^2 = \sum_{k \in K} \sigma_{jk}^2 y_{jk}$. Thus, the service system design problem is modeled as a network of independent M/G/1 queues.

Under steady state conditions ($\Lambda_j/\mu_j < 1$), first-come-first-serve (FCFS) queuing discipline, and infinite buffers to accommodate users waiting for service, the expected waiting time (including the time spent in service) of users at facility j is given, by the Pollaczek–Khintchine formula [15], as

$$E[w_j] = \left(\frac{1+Cv_j^2}{2} \right) \frac{\tau_j \rho_j}{1-\rho_j} + \tau_j = \left(\frac{1+Cv_j^2}{2} \right) \frac{\Lambda_j}{\mu_j(\mu_j - \Lambda_j)} + \frac{1}{\mu_j} \tag{1}$$

where $\tau_j = 1/\mu_j$ is the average service time at facility j, $\rho_j = \Lambda_j/\mu_j$ is the average utilization of facility j, and $Cv_j = \sigma_j \mu_j$ is the coefficient of variation of service times at facility j. $E[w_j]$ can be written in terms of location and allocation variables (y_{jk} and x_{ij}) as

$$E[w_j(\mathbf{x}, \mathbf{y})] = \frac{\left(1 + \sum_{k \in K} Cv_{jk}^2 y_{jk}\right) \sum_{i \in I} \lambda_i x_{ij}}{2 \sum_{k \in K} \mu_{jk} y_{jk} \left(\sum_{k \in K} \mu_{jk} y_{jk} - \sum_{i \in I} \lambda_i x_{ij} \right)} + \frac{1}{\sum_{k \in K} \mu_{jk} y_{jk}} \tag{2}$$

The expected number of users in service or waiting for service at facility j is given, using Little's law, as $\Lambda_j E[w_j]$. If d denotes the average waiting time cost per customer (henceforth called unit queuing delay cost), then the *total delay/congestion cost* in the network can be expressed as $d \sum_{j \in J} \Lambda_j E[w_j(\mathbf{x}, \mathbf{y})] = d \sum_{j \in J} \sum_{i \in I} \lambda_i x_{ij} E[w_j(\mathbf{x}, \mathbf{y})]$. We assume there is a variable access cost c_{ij} of providing service to users at node i from facility at site j. The problem is to simultaneously determine: (i) the locations of the service facilities and their corresponding capacity levels; (ii) the assignment of users to located service facilities, such that the total system-wide cost, consisting of cost of opening service facilities with appropriate capacities, cost of accessing service facilities by users and cost associated with customers' waiting, is minimized. To model the problem, we first summarize the notations:

Indices, Sets and Parameters:

i	index for user nodes, $i \in I$
j	index for potential facility sites, $j \in J$
k	index for capacity levels at facility sites, $k \in K$
f_{jk}	fixed set up cost of locating a facility with capacity level k at site j
c_{ij}	cost of providing service to user node i from facility at site j
λ_i	mean demand rate of service requests from user node i
μ_{jk}	mean service rate at facility site j with capacity level k
σ^2_{jk}	variance of *service times* at facility site j with capacity level k
d	unit queueing delay cost
D_{ij}	distance between user node i and service facility j
C_{ij}	set of all service facility locations that are closer to user node i than the facility located at j; $C_{ij} = \{l \mid D_{il} < D_{ij}\}$

Decision variables:

y_{jk}	1 if facility at site j is open and equipped with a capacity level $k \in K$, 0 otherwise
x_{ij}	1 if the demand for service from user node i is satisfied by facility j, 0 otherwise

Derived variables:

μ_j	mean service rate at facility site j; $\mu_j = \sum_{k \in K} \mu_{jk} y_{jk}$
Λ_j	aggregate demand arrival rate at facility j; $\Lambda_j = \sum_{i \in I} \lambda_i x_{ij}$
Cv_j	coefficient of variation of service times at facility site j; $Cv_j = \sigma_j \mu_j$
ρ_j	average utilization of service facility at site j; $\rho_j = \Lambda_j / \mu_j$
$E[w_j]$	expected waiting time (including time spent in service) at facility j

The resulting non-linear integer program (IP) model of the problem is as follows:

$$[P]: Z(\mathbf{x}, \mathbf{y}) = \min_{\mathbf{x}, \mathbf{y}} \sum_{j \in J} \sum_{k \in K} f_{jk} y_{jk} + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + d \sum_{i \in I} \sum_{j \in J} \lambda_i x_{ij} E[w_j(\mathbf{x}, \mathbf{y})] \tag{3}$$

$$\text{s.t.} \quad \sum_{i \in I} \lambda_i x_{ij} \le \sum_{k \in K} \mu_{jk} y_{jk} \quad \forall j \tag{4}$$

$$\sum_{j \in J} x_{ij} = 1 \quad \forall i \tag{5}$$

$$\sum_{k \in K} y_{jk} \le 1 \quad \forall j \tag{6}$$

$$x_{ij}, y_{jk} \in \{0, 1\} \quad \forall i, j, k \tag{7}$$

The first term in the objective function (3) is the amortized cost of opening facilities at appropriate capacity levels. The second term is the cost of accessing service facilities, while the third term captures the cost of users' waiting at facilities. The expression for $E[w_j(\mathbf{x}, \mathbf{y})]$ in the objective function is given by (2). Constraint set (4) ensures that the total demand allocated to any service facility is within its capacity. Note that constraint set (4) will be non-binding at optimality, else the term $E[w_j(\mathbf{x}, \mathbf{y})]$ in the objective function goes to infinity. This ensures the stability of the queueing system ($\rho_j = \Lambda_j / \mu_j < 1$) at each facility j. Constraint set (5) ensures that each user node is assigned to only one of the open facilities for its service. Constraint set (6) states that at most one of the multiple capacity levels is selected at a facility. Constraint set (7) imposes binary restrictions on the location and allocation variables.

(3)–(7) model a directed choice environment. This can also be adapted to the user choice environment by explicitly specifying the decision model for the users' choice of facilities. For this, the most common assumption used in the literature is that users always choose the closest service facility. Under this assumption, the users' choice of facilities is generally modeled using the following set of constraints, called the *closest assignment constraints* (originally proposed by [20]):

$$x_{ij} \ge \sum_{k \in K} y_{jk} - \sum_{l \in C_{ij}} \sum_{k \in K} y_{lk} \tag{8}$$

where $C_{ij} = \{l \mid D_{il} < D_{ij}\}$ is the set of all service facility locations that are closer to user node i than the facility located at j. Alternative ways of modeling the closest assignment constraints are discussed by Berman et al. [9] and Espejo et al. [14].

The presence of the non-linear term $\sum_{i \in I} \sum_{j \in J} \lambda_i x_{ij} E[w_j(\mathbf{x}, \mathbf{y})]$ in the objective function makes [P] challenging to solve. In the next section, we first present an approach to linearize the expression for the total waiting time spent by the users at a facility. We then present an exact solution procedure, based on a constraint generation algorithm, to solve the linearized model. We present the proposed solution method only with respect to the directed choice environment. However, the same solution method is also applicable to the user choice method with the addition of constraint (8).

3. Model linearization

The non-linear term in the objective function of [P] can be written, using (1), as

$$\Lambda_j E[w_j] = \left(\frac{1 + Cv_j^2}{2}\right) \frac{\Lambda_j^2}{\mu_j(\mu_j - \Lambda_j)} + \frac{\Lambda_j}{\mu_j} = \left(\frac{1 + Cv_j^2}{2}\right) \frac{\rho_j^2}{(1 - \rho_j)} + \rho_j \tag{9}$$

To linearize (9), it can be rewritten, upon rearranging its terms, as

$$\Lambda_j E[w_j] = \frac{1}{2}\left\{(1 + Cv_j^2)\frac{\rho_j}{1 - \rho_j} + (1 - Cv_j^2)\rho_j\right\} \tag{10}$$

Let us define a set of nonnegative auxiliary variables, U_j, such that:

$$U_j = \frac{\rho_j}{1 - \rho_j} = \frac{\Lambda_j}{\mu_j - \Lambda_j} = \frac{\sum_{i \in I} \lambda_i x_{ij}}{\sum_{k \in K} \mu_{jk} y_{jk} - \sum_{i \in I} \lambda_i x_{ij}} \tag{11}$$

which implies

$$\rho_j = \frac{U_j}{1 + U_j} \tag{12}$$

Using $\rho_j = \Lambda_j / \mu_j$, the total demand Λ_j at facility j can be expressed as

$$\Lambda_j = \sum_{i \in I} \lambda_i x_{ij} = \rho_j \mu_j = \rho_j \sum_{k \in K} \mu_{jk} y_{jk} = \sum_{k \in K} \mu_{jk} z_{jk}$$

$$\text{where } z_{jk} = \begin{cases} \rho_j & \text{if } y_{jk} = 1 \\ 0 & \text{otherwise} \end{cases}$$

Hence, the total demand at any facility j can be expressed as

$$\sum_{i \in I} \lambda_i x_{ij} = \sum_{k \in K} \mu_{jk} z_{jk} \quad \forall j$$

Given that any facility can have at most one capacity level, there exists at most one $k = k'$ such that $y_{jk'} = 1$, while $y_{jk} = 0$ $\forall k \ne k'$. Further, $\rho_j < 1$. Thus z_{jk} can alternatively be expressed using the following set of constraints:

$$z_{jk} \le y_{jk} \quad \forall j,k \qquad \sum_{k \in K} z_{jk} = \rho_j \quad \forall j \qquad z_{jk} \ge 0 \quad \forall j,k$$

With the above substitutions in (10), the expression for $\Lambda_j E[w_j]$ reduces to:

$$\Lambda_j E[w_j] = \frac{1}{2}\left\{\left(1+\sum_{k \in K} Cv_{jk}^2 y_{jk}\right)U_j + \left(1-\sum_{k \in K} Cv_{jk}^2 y_{jk}\right)\rho_j\right\}$$
$$= \frac{1}{2}\left(U_j + \sum_{k \in K} Cv_{jk}^2 w_{jk} + \rho_j - \sum_{k \in K} Cv_{jk}^2 z_{jk}\right)$$
$$\text{where } w_{jk} = \begin{cases} U_j & \text{if } y_{jk}=1 \\ 0 & \text{otherwise} \end{cases}$$

Again, using the fact that there exists at most one $k=k'$ such that $y_{jk'}=1$, while $y_{jk}=0 \ \ \forall k \ne k'$, w_{jk} can alternatively be expressed using the following set of constraints:

$$w_{jk} \le M y_{jk} \quad \forall j,k \qquad \sum_{k \in K} w_{jk} = U_j \quad \forall j$$
$$w_{jk} \ge 0 \quad \forall j,k$$

where M is a large number (Big-M). We now state a Lemma that helps us linearize the above non-linear model [P].

Lemma 1. *The function $\rho_j(U_j) = U_j/(1+U_j)$ is concave in $U_j \in [0,\infty)$.*

Proof. Differentiating ρ_j w.r.t. U_j, we get the first derivative, $\delta\rho_j/\delta U_j = 1/(1+U_j)^2 > 0$, and the second derivative, $\delta^2\rho_j/\delta U_j^2 = -2/(1+U_j)^3 < 0$, which proves that the function $\rho_j(U_j)$ is concave in U_j. □

Lemma 1 implies that for a given set of points indexed by h, $h \in H$, the function $\rho_j(U_j)$ can be approximated arbitrarily close by a set of piecewise linear functions that are tangent to ρ_j at points $\{U_j^h\}_{h \in H}$, such that:

$$\rho_j = \min_{h \in H}\left\{\frac{1}{(1+U_j^h)^2}U_j + \frac{(U_j^h)^2}{(1+U_j^h)^2}\right\} \tag{13}$$

This is equivalent to the following set of constraints:

$$\rho_j \le \frac{1}{(1+U_j^h)^2}U_j + \frac{(U_j^h)^2}{(1+U_j^h)^2} \quad \forall j, h \in H \tag{14}$$

provided $\exists h \in H$ such that (14) holds with equality. Using the above substitutions results in the following linear mixed integer program (MIP) reformulation of [P]:

$$[P(H)]: \ \min \quad \sum_{j \in J}\sum_{k \in K} f_{jk}y_{jk} + \sum_{i \in I}\sum_{j \in J} c_{ij}x_{ij} + \frac{d}{2}\sum_{j \in J}\left\{U_j + \rho_j + \sum_{k \in K} Cv_{jk}^2(w_{jk}-z_{jk})\right\} \tag{15}$$

s.t. (5)–(7), (14)

$$\sum_{i \in I}\lambda_i x_{ij} - \sum_{k \in K}\mu_{jk}z_{jk} = 0 \quad \forall j \tag{16}$$

$$z_{jk} \le y_{jk} \quad \forall j,k \tag{17}$$

$$\sum_{k \in K} z_{jk} = \rho_j \quad \forall j \tag{18}$$

$$w_{jk} \le M y_{jk} \quad \forall j,k \tag{19}$$

$$\sum_{k \in K} w_{jk} = U_j \quad \forall j \tag{20}$$

$$0 \le z_{jk}, \rho_j \le 1; \quad w_{jk}, U_j \ge 0 \ \ \forall j,k \tag{21}$$

The linear MIP model [$P(H)$] has $2(|J| + |J| * |K|)$ additional continuous variables compared to the non-linear IP model [P]. Further, [$P(H)$] has $(|I| + 4*|J| + 2*|J|*|K| + |J|*|H|)$ constraints, as opposed to only $(|I| + 2*|J|)$ constraints in [P]. Hence, the non-linearity of [P] is eliminated at the expense of having to deal with a large number of additional variables and constraints in [$P(H)$].

Equivalence between [P] and [$P(H)$] requires that $\exists h \in H$ such that (14) holds with equality. Proposition 1 states that this condition will always be satisfied at optimality.

Proposition 1. *In the linearized model* [$P(H)$], *at least one of the constraints in* (14) *will be binding at optimality*.

Proof. Upon rearranging the terms, (14) can be rewritten as

$$U_j \ge ((1+U_j^h)^2\rho_j - (U_j^h)^2 \quad \forall j, h \in H \tag{22}$$

Since U_j appears in the objective function of [$P(H)$] with a positive coefficient, [$P(H)$] attains its minimum value only when U_j is minimized. This implies that $\forall j \in J, \exists h \in H$ such that (22) holds with equality if $(1+U_j^h)^2\rho_j - (U_j^h)^2 \ge 0$, else U_j=0 if $(1+U_j^h)^2\rho_j - (U_j^h)^2 < 0$. Further,

$$0 \le (1+U_j^h)^2\rho_j - (U_j^h)^2$$
$$= (\rho_j - 1)(U_j^h)^2 + 2\rho_j U_j^h + \rho_j$$
$$\Leftrightarrow U_j^h \in \left[0, \frac{\rho_j + \sqrt{\rho_j}}{1-\rho_j}\right] \quad \forall j \in J, h \in H (\text{since } \rho_j \le 1, U_j \ge 0)$$

Thus, to prove that $\forall j \in J, \exists h \in H$ such that (22) holds with equality, it is sufficient to show that

$$U_j^h \in \left[0, \frac{\rho_j + \sqrt{\rho_j}}{1-\rho_j}\right].$$

Since U_j^h is an approximation to U_j, we obtain:

$$0 \le U_j^h \approx U_j = \frac{\rho_j}{1-\rho_j}$$
$$\le \frac{\rho_j + \sqrt{\rho_j}}{1-\rho_j}$$

This proves that $\forall j \in J, \exists h \in H$ such that (22) holds with equality. □

3.1. Special cases

In many cases, the service at facilities involves repeated steps without much variation, i.e., $Cv_{jk} = 0$ (such that each service facility is modeled as an M/D/1 queuing system). For such deterministic service times, the users' expected waiting time at facility j is given by

$$\Lambda_j E[w_j] = \frac{1}{2}\left(\frac{\Lambda_j}{\mu_j - \Lambda_j} + \frac{\Lambda_j}{\mu_j}\right) = \frac{1}{2}(U_j + \rho_j)$$

and the resulting linear MIP model is as follows:

$$[P(H)_{Cv=0}]: \ \min \quad \sum_{j \in J}\sum_{k \in K} f_{jk}y_{jk} + \sum_{i \in I}\sum_{j \in J} c_{ij}x_{ij} + \frac{d}{2}\sum_{j \in J}(U_j + \rho_j)$$
$$\text{s.t.} \quad (4)\text{–}(7), (14)$$
$$0 \le \rho_j \le 1; \quad U_j \ge 0 \ \ \forall j$$

For exponentially distributed service times at the facilities, i.e., $Cv_{jk} = 1$ (M/M/1 case), the expression is given by $\Lambda_j E[w_j] = \rho_j/(1-\rho_j) = U_j$, and the linear model reduces to:

$$[P(H)_{Cv=1}]: \ \min \quad \sum_{j \in J}\sum_{k \in K} f_{jk}y_{jk} + \sum_{i \in I}\sum_{j \in J} c_{ij}x_{ij} + d\sum_{j \in J} U_j$$
$$\text{s.t.} \quad (4)\text{–}(7), (14)$$
$$0 \le \rho_j \le 1; \quad U_j \ge 0 \ \ \forall j$$

4. Solution approach

We state the following two propositions, which are used in the development of the solution algorithm for $[P(H)]$.

Proposition 2. *For any given subset of points* $\{U_j^h\}_{H^q \subset H}$, (23) *provides a lower bound on the optimal objective function value of* $[P]$, *where* $v(\bullet)$ *is the objective function value of the problem* $(\bullet)$ *and* $(\mathbf{x}^p, \mathbf{y}^q, \boldsymbol{\rho}^q, \mathbf{w}^q, \mathbf{z}^q, \mathbf{U}^q)$ *is the optimal solution to* $[P(H^q)]$.

$$LB = v(P(H^q))$$

$$= \sum_{j \in J}\sum_{k \in K} f_{jk} y_{jk}^q + \sum_{i \in I}\sum_{j \in N} c_{ij} x_{ij}^q + \frac{d}{2}\sum_{j \in J}\left\{ U_j^q + \rho_j^q + \sum_{k \in K} Cv_{jk}^2 (w_{jk}^q - z_{jk}^q) \right\} \tag{23}$$

Proof. Since $[P(H^q)]$ is a relaxation of the full problem $[P(H)]$, the objective function value of $[P(H^q)]$, given by (23), provides a lower bound on the optimal objective function value of $[P(H)]$, and hence on the optimal objective function value of $[P]$. □

Proposition 3. *For any given subset of points* $\{U_j^h\}_{H^q \subset H}$, (24) *provides an upper bound on the optimal objective function value of* $[P]$, *where* $(\mathbf{x}^p, \mathbf{y}^q)$ *is the optimal solution to* $P[(H^q)]$.

$$UB = Z(\mathbf{x}^q, \mathbf{y}^q) = \sum_{j \in J}\sum_{k \in K} f_{jk} y_{jk}^q + \sum_{i \in I}\sum_{j \in J} c_{ij}\lambda_i x_{ij}^q$$

$$+ d\sum_{j \in J}\left\{ \frac{\left(1+\sum_{k \in K} Cv_{jk}^2 y_{jk}^q\right)\left(\sum_{i \in I}\lambda_i x_{ij}^q\right)^2}{2\sum_{k \in K}\mu_{jk} y_{jk}^q \left(\sum_{k \in K}\mu_{jk} y_{jk}^q - \sum_{i \in I}\lambda_i x_{ij}^q\right)} + \frac{\sum_{i \in I}\lambda_i x_{ij}^q}{\sum_{k \in K}\mu_{jk} y_{jk}^q} \right\} \tag{24}$$

Proof. For any subset of points $\{U_j^h\}_{H^q \subset H}$, the optimal solution $(\mathbf{x}^p, \mathbf{y}^q)$ to $[P(H^q)]$ is also a feasible solution to $[P]$ as all the constraints of $[P]$ are also contained in $[P(H^q)]$. Hence, the objective function of $[P]$ evaluated at $(\mathbf{x}^p, \mathbf{y}^q)$, which is given by (24), provides an upper bound on the optimal objective of $[P]$. □

4.1. Solution algorithm

Although there are a large number of constraints/cuts (14) in the linear MIP model $[P(H)]$, it is not necessary to generate all of them. Instead, it suffices to start with a subset $H^1 \subset H$ of these cuts, where H^1 may be empty or chosen a priori, and generate the rest as needed. Our preliminary computational experiments, presented in Table 1, suggest a much faster convergence of the algorithm when H^1 is non-empty. We, therefore, use a carefully chosen subset H^1 of initial cuts in all our subsequent experiments. The subset of points $\{U_j^h\}_{H^1 \subset H}$, required to obtain the initial subset of cuts, is generated to approximate the function $\rho_j(U_j) = U_j/(1+U_j)$ using its tangents $\widehat{\rho_j}(U_j)$ at these points such that the approximation error, measured as $\widehat{\rho_j}(U_j) - \rho_j(U_j)$, is at most 0.001 [12]. The resulting $[P(H^1)]$ is solved, giving a solution $(\mathbf{x}^1, \mathbf{y}^1, \boldsymbol{\rho}^1, \mathbf{w}^1, \mathbf{z}^1, \mathbf{U}^1)$. The lower bound ($LB^1$q) and upper bound ($UB^1$) are computed using (23) and (24) respectively. If UB^1 equals LB^1 within some accepted tolerance (ϵ), then $(\mathbf{x}^1, \mathbf{y}^1)$ is an optimal solution to $[P]$, and the algorithm stops. Otherwise, a new set of points $\{U_j^h\}$ is generated using the current solution as

$$U_j^{h_{new}} = \frac{\sum_{i \in I}\lambda_i x_{ij}^1}{\sum_{k \in K}\mu_{jk} y_{jk}^1 - \sum_{i \in I}\lambda_i x_{ij}^1} \quad \forall j.$$

A new set of constraints/cuts of the form (14) is generated at these new points, which are appended to $[P(H^1)]$ to give $[P(H^2)]$. Then, $[P(H^2)]$ is solved to yield a solution $(\mathbf{x}^2, \mathbf{y}^2, \boldsymbol{\rho}^2, \mathbf{w}^2, \mathbf{z}^2, \mathbf{U}^2)$ and a lower bound LB^2. Since the upper bound, as given by (24), changes non-monotonically, the new upper bound UB^2 is retained as min $\{UB^1, Z(\mathbf{x}^2, \mathbf{y}^2)\}$. If UB^2 equals LB^2 within the accepted tolerance (ϵ), then the algorithm terminates with $(\mathbf{x}^2, \mathbf{y}^2)$ as an optimal solution. Otherwise, the above process is repeated until UB^q equals LB^q within (ϵ) for some iterations q. The steps of the algorithm are outlined below:

Table 1
Effect of adding a priori cuts on the performance of the solution method: instances with 100 user nodes and 10 potential facilities.

Cv	d	Without a priori cuts		With a priori cuts		% Reduction
		# Iter	CPU	# Iter	CPU	in CPU times
0	1	3	2.24	2	1.58	29.45
	10	6	7.97	2	1.80	77.47
	25	6	5.36	3	2.97	44.64
	50	7	12.14	2	3.75	69.14
	100	7	13.54	2	3.42	74.73
	250	8	10.97	2	2.04	81.44
	500	8	12.06	2	1.59	86.85
	1000	12	21.67	3	2.53	88.32
	5000	20	24.78	2	2.48	90.01
0.5	1	3	2.07	2	1.57	24.21
	10	6	5.89	3	3.65	38.13
	25	6	6.14	2	2.40	60.95
	50	7	10.54	2	3.48	66.97
	100	7	12.34	2	4.02	67.42
	250	11	13.83	2	2.33	83.19
	500	12	19.13	3	2.85	85.10
	1000	14	22.14	3	2.26	89.81
	5000	18	28.03	2	1.81	93.54
1	1	4	3.16	2	1.31	58.54
	10	6	7.07	2	2.23	68.45
	25	6	8.67	2	3.80	56.18
	50	8	15.98	2	4.09	74.43
	100	7	12.06	2	3.33	72.39
	250	9	16.68	2	1.35	91.93
	500	16	22.20	3	2.38	89.30
	1000	13	24.42	2	2.26	90.77
	5000	15	38.65	3	8.25	78.65
1.5	1	4	2.63	2	1.61	39.00
	10	6	6.42	2	2.33	63.79
	25	8	14.73	2	3.18	78.43
	50	7	14.83	2	4.16	71.99
	100	8	17.07	2	1.98	88.39
	250	13	34.21	2	1.75	94.88
	500	13	25.71	2	2.16	91.60
	1000	11	29.30	2	3.20	89.07
	5000	12	60.59	3	16.79	72.29
2	1	4	2.80	2	1.42	49.45
	10	6	9.12	2	3.38	62.98
	25	6	11.73	2	4.42	62.30
	50	8	15.75	3	3.92	75.10
	100	9	19.78	2	1.43	92.76
	250	12	25.27	3	2.29	90.96
	500	12	22.45	2	2.86	87.26
	1000	12	38.50	3	4.33	88.75
	5000	12	162.50	3	15.60	90.40
2.5	1	6	5.19	2	1.73	66.63
	10	7	12.16	3	4.68	61.51
	25	7	12.23	2	3.91	68.05
	50	10	23.52	2	2.53	89.27
	100	12	31.28	2	2.05	93.46
	250	10	25.39	2	2.32	90.88
	500	11	26.78	3	4.90	81.72
	1000	11	49.96	3	4.78	90.44
	5000	14	50,154.70	3	13.14	99.97
	Min.	3	2.07	2	1.31	24.21
	Avg.	9	948.78	2	3.56	75.25
	Max.	20	50,154.70	3	16.79	99.97

Algorithm 1. Constraint generation algorithm for $[P(H)]$.

1: $q \leftarrow 1$; $UB^{q-1} \leftarrow +\infty$; $LB^{q-1} \leftarrow -\infty$.
2: Choose an initial set of points $\{U^h\}_{h \in H^q}$ to approximate the function $\rho_j(U_j) = U_j/1+U_j$.

3: **while** $(UB^{q-1}-LB^{q-1})/UB^{q-1} > \epsilon$ **do**
4: Solve $P(H^q)$, and obtain its optimal solution $(\mathbf{x}^q, \mathbf{y}^q, \rho^q, \mathbf{w}^q, \mathbf{z}^q, \mathbf{U}^q)$.
5: Update the lower bound: $LB^q \leftarrow v(P(H^q))$ using (23).
6: Update the upper bound: $UB^q \leftarrow \min\{UB^{q-1}, Z(\mathbf{x}^q, \mathbf{y}^q)\}$ using (24).
7: Generate a new set of points $U_j^{h_{new}} = \frac{\sum_{i \in I} \lambda_i x_{ij}^q}{\sum_{k \in K} \mu_{jk} y_{jk}^q - \sum_{i \in I} \lambda_i x_{ij}^q}$ $\forall j$ using (11).
8: $H^{q+1} \leftarrow H^q \cup \{h_{new}\}$.
9: $q \leftarrow q+1$
10: **end while**

Proposition 4. *The constraint generation algorithm to solve* $[P(H)]$ *is finite.*

Proof. Since $x_{ij}, y_{jk} \in \{0,1\}$ $\forall i, j, k$ and

$$U_j = \frac{\sum_{i \in I} \lambda_i x_{ij}}{\sum_{k \in K} \mu_{jk} y_{jk} - \sum_{i \in I} \lambda_i x_{ij}},$$

U_j can take only a finite set of values. Therefore, in order to prove the finiteness of Algorithm 1, it is sufficient to prove that the generated values of U_j^h are not repeated.

Consider an iteration q, wherein Algorithm 1 has not yet converged, that is $UB^q > LB^q$. Further, suppose $(\mathbf{x}^q, \mathbf{y}^q, \rho^q, \mathbf{w}^q, \mathbf{z}^q, \mathbf{U}^q)$ is a solution to $[P(H^q)]$. The new points $U_j^{h_{new}}$ generated at iteration q are given by

$$U_j^{h_{new}} = \frac{\sum_{i \in I} \lambda_i x_{ij}^q}{\sum_{k \in K} \mu_{jk} y_{jk}^q - \sum_{i \in I} \lambda_i x_{ij}^q} \quad \forall j$$

Suppose the values of $U_j^{h_{new}}$ are already generated at iteration $q^0 < q$ $\forall j \in J$. Then,

$$(14) \Rightarrow \frac{U_j^{h_{new}}}{1+U_j^{h_{new}}} \le \frac{1}{\left(1+U_j^{h_{new}}\right)^2} U_j^q + \left(\frac{U_j^{h_{new}}}{1+U_j^{h_{new}}}\right)^2 \quad \forall j$$

$$\Rightarrow U_j^{h_{new}} \le U_j^q \quad \forall j$$

We now have:

$$\begin{aligned}
LB^q &= \sum_{j \in J} \sum_{k \in K} f_{jk} y_{jk}^q + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}^q + \frac{d}{2} \sum_{j \in J} \left\{ U_j^q + \rho_j^q + \sum_{k \in K} C v_{jk}^2 (w_{jk}^q - z_{jk}^q) \right\} \\
&\ge \sum_{j \in J} \sum_{k \in K} f_{jk} y_{jk}^q + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}^q + \frac{d}{2} \sum_{j \in J} \left\{ U_j^{h_{new}} + \rho_j^q + \sum_{k \in K} C v_{jk}^2 (w_{jk}^q - z_{jk}^q) \right\} \\
&= \sum_{j \in J} \sum_{k \in K} f_{jk} y_{jk}^q + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}^q + \frac{d}{2} \sum_{j \in J} \left\{ \frac{\sum_{i \in I} \lambda_i x_{ij}^q}{\sum_{k \in K} \mu_{jk} y_{jk}^q - \sum_{i \in I} \lambda_i x_{ij}^q} \right. \\
&\quad \left. + \rho_j^q + \sum_{k \in K} C v_{jk}^2 (w_{jk}^q - z_{jk}^q) \right\} \\
&= \sum_{k \in K} f_{jk} y_{jk}^q + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}^q + d \sum_{j \in J} \left\{ \frac{\left(1+\sum_{k \in K} C v_{jk}^2 y_{jk}^q\right)\left(\sum_{i \in I} \lambda_i x_{ij}^q\right)^2}{2 \sum_{k \in K} \mu_{jk} y_{jk}^q \left(\sum_{k \in K} \mu_{jk} y_{jk}^q - \sum_{i \in I} \lambda_i x_{ij}^q\right)} \right. \\
&\quad \left. + \frac{\sum_{i \in I} \lambda_i x_{ij}^q}{\sum_{k \in K} \mu_{jk} y_{jk}^q} \right\} = Z(\mathbf{x}^q, \mathbf{y}^q) \\
&\ge \min\{UB^{q-1}, Z(\mathbf{x}^q, \mathbf{y}^q)\} = UB^q
\end{aligned}$$

This contradicts the initial assumption $UB^q > LB^q$. Therefore, at any given iteration, $U_j^{h_{new}}$ will be different from the previously generated points at least for any one j. Furthermore, the number of values that U_j^h can take is finite. Hence, the algorithm should terminate in a finite number of iterations. □

5. Computational results

We present our computational experiments with the solution approach proposed in Section 4. The solution procedures are coded in C++ (Visual Studio 2010), while $[P(H^q)]$ at every iteration q is solved using IBM ILOG CPLEX 12.4 on a personal laptop with Intel Core i5-3230 M, 2.60 GHz CPU; 4 GB RAM; and Windows 64-bit operating system. The test instances are generated using a combination of the schemes used by Amiri [5] and Holmberg et al. [16], described in detail in Section 5.1. We generate four sets of test problems by varying the combination of the number of user nodes and the number of potential facility locations $(|I|, |J|)$ as (100, 10), (200, 15), (300, 20), and (400, 25). The number of capacity levels $(|K|)$ is set at 5, and the tolerance level for optimality ϵ is set at 10^{-5} in all the experiments.

5.1. Data generation

The co-ordinates of the nodes representing user nodes are randomly generated using a uniform distribution $U \sim (10, 300)$. The mean demand rate at any user node i is randomly generated as $\lambda_i \sim U(10, 50)$. The locations of the potential facilities are obtained from the solution to a p-median facility location problem. The other parameters are generated as follows:

- *Capacity levels* (μ_{jk}) are set as $\mu_{j1} = 0.50 * \mu_{j3}$; $\mu_{j2} = 0.75 * \mu_{j3}$; $\mu_{j4} = 1.25 * \mu_{j3}$; $\mu_{j5} = 1.50 * \mu_{j3}$, where $\mu_{j3} = (1.25 \sum_{i \in I} \lambda_i)/(|J| * LR)$. The Load Ratio (LR), which is defined as the average facility utilization if all the potential facilities are assigned capacity level 3 $(k=3)$ to satisfy 125% of the total demand in the network, is set at 0.60.
- *Fixed costs* (f_{jk}) are generated as $f_{j3} = FCR * E_{jj_0}$, where E_{jj_0} is the Euclidean distance between the facility site j and a fixed point $j_0 = (155, 155)$, which is the center of the box within which the coordinates of user nodes are generated. FCR, called the Fixed Cost Ratio, is a constant set at 40. $f_{j1} = 0.60 * f_{j3}$, $f_{j2} = 0.85 * f_{j3}$, $f_{j4} = 1.15 * f_{j3}$, $f_{j5} = 1.35 * f_{j3}$. The chosen values of fixed cost for different capacity levels exhibit both an underlying economy as well as diseconomy of scale. For example, for a 25% increase in capacity (corresponding to μ_{j4} over μ_{j3}), the capacity cost increases only 15%. However, for the a 50% increase in capacity (corresponding to μ_{j5} over μ_{j3}), the capacity cost increases 35%.
- *Service costs* (c_{ij}) are generated as $c_{ij} = 5 * E_{ij}$, where E_{ij} is the Euclidean distance between user node i and the facility site j.
- *Unit queueing delay cost* (d) is assumed to be one of the values from the set {1, 10, 25, 50, 100, 250, 500, 1000, 5000}.

5.2. Analysis of computational results

Table 1 compares the performance of the proposed solution approach with a priori cuts versus without a priori cuts for test instances with 100 user nodes and 10 facilities. The table reports the number of iterations required (#Iter) and the computation time in seconds (CPU) for different values of coefficient of variation of service times (Cv) and the unit queuing delay cost (d). The table also reports the percentage reduction in computation time as a result of adding a priori cuts, which is computed as (CPU time without a priori cuts – CPU time with a priori cuts) $\times$ 100/(CPU time without a priori cuts). As described in Section 4.1, a priori cuts for the function $\rho(U) = U/(1+U)$ are generated based on the piecewise linear approximation $\widehat{\rho}(U)$ of the function $\rho(U)$ such that the approximation error (measured by $\widehat{\rho}(U) - \rho(U)$) is at most 0.001 [12]. Fig. 1 shows for ($|I| = 100, |J| = 10$) the percentage reduction in the number of iterations and the CPU times as a result of addition of a priori cuts for different values of d and Cv.

Following observations can be made from Table 1 and Fig. 1. The results in Table 1 show that without a priori cuts, the problem takes, on an average, 949 s and 9 iterations to solve. The addition of a priori cuts reduces these numbers to less than 4 s of CPU time and less than three iterations. The percentage reduction in CPU time due to the addition of a priori cuts varies between 24.21% and 99.97%, with an average reduction of 75.25%. Plots in Fig. 1 further show that the benefits of a priori cuts, in terms of % reduction in CPU time and number of iterations, increase significantly with an increase in the unit queuing delay cost (d). This is because, as Table 1 suggests, without a priori cuts, the computation time and the number of iterations required by the proposed solution approach increase significantly with an increase in d. However, with the addition of a priori cuts, the computation time and the number of iterations required are almost *insensitive* to d.

Tables 2–5 summarize the computational results for four sets of instances ($|I| = 100, |J| = 10$; $|I| = 200, |J| = 15$; $|I| = 300, |J| = 20$; $|I| = 400, |J| = 25$), for different values of the unit queueing delay cost (d=1, 10, 25, 50, 100, 250, 500, 1000, 5000) and coefficient of variation of service times (Cv=0, 1, 0.5, 1.5, 2, 2.5). In solving each of the problem instances, we use a priori set of 32 cuts, which corresponds to a maximum approximation error of 0.001 in the linear approximation of $\rho(U)$. The tables report for each problem instance the total cost (TC); fixed cost (FC), access cost (AC), and delay cost (DC), expressed as a percentage of the total cost; computation time in seconds (CPU); number of iterations for convergence (#Iter); number of facilities opened (#Facility); and the minimum, maximum, and average facility utilizations among the open facilities. Fig. 2 shows the effect of varying d on FC, AC, DC, and TC for different values of Cv. Fig. 3 shows the effect of varying d on the maximum and average facility utilizations. Following observations can be made from the figures:

- An increase in d or Cv increases TC, as expected. An increase in d also results in a higher DC, which is expected, provided the expected total number of users waiting at different service facilities in the network ($\sum_j \Lambda_j E[w_j]$) remains unchanged with an increase in d. However, an increase in d implies a larger penalty for congestion (waiting customers), which forces the system to either attain more uniform utilization ($\rho_j = \Lambda_j/\mu_j$) among the open facilities or to install a larger total service capacity in the network (either by opening more facilities or by installing larger capacities at fewer, more or the same number of opened facilities). In either case, the maximum facility utilization decreases and the average facility utilization either decreases or remains constant, as evident from Fig. 3. This results in a decrease in the expected total number of users waiting in the network ($\sum_j \Lambda_j E[w_j]$). We observe from our results that the percentage decrease in the expected total number of waiting users in the network is smaller compared to the percentage increase in d, such that the net effect is always an increase in the total delay cost (DC).
- For a fixed network configuration (location and allocation of service facilities), an increase in Cv is expected to increase the expected total number of waiting users in the network, and hence increases DC. However, when the location and allocation of service facilities are also decision variables, as they are in the current problem, Fig. 2 suggests that an increase in Cv may sometimes result in a decrease in DC, which initially appears to be counter intuitive. However, this can be justified as follows. To counter the increase in congestion at a higher value of Cv, the system chooses to either attain more uniform utilization among the open facilities or to install a larger total service capacity in the system, thereby resulting in a decrease in the expected total number of waiting users in the network, and hence a decrease in DC.
- The fixed cost (FC) changes non-monotonically with an increase in d or Cv, which also seems counter intuitive since to counter the effect of increase in d or Cv, the system is expected to either attain more uniform utilization among the open facilities or to install a larger total service capacity in the system, neither of which should decrease FC. In case the system chooses to redistribute the total service capacity more uniformly among the open facilities to uniformize their utilizations, some facilities may exhibit economies of scale while others may exhibit diseconomies in scale in fixed costs (as described in Section 5.1), which may result in either an increase or decrease in FC depending on the net effect of economies and diseconomies of scale. Moreover, since the fixed cost of opening a service facility with a given capacity level (f_{jk}) depends on its distance from a fixed point (j_0), an increase in d or Cv may result in an increase or decrease in FC depending on whether the system chooses to open service facilities closer to or farther from j_0 at the higher value of d or Cv. AC may also change non-monotonically with an increase in d or Cv for similar reason.
- Comparison of CPU times across Tables 2–5 shows, as expected, that the computation time increases as the number of user nodes and potential facilities ($|I|, |J|$) increases.

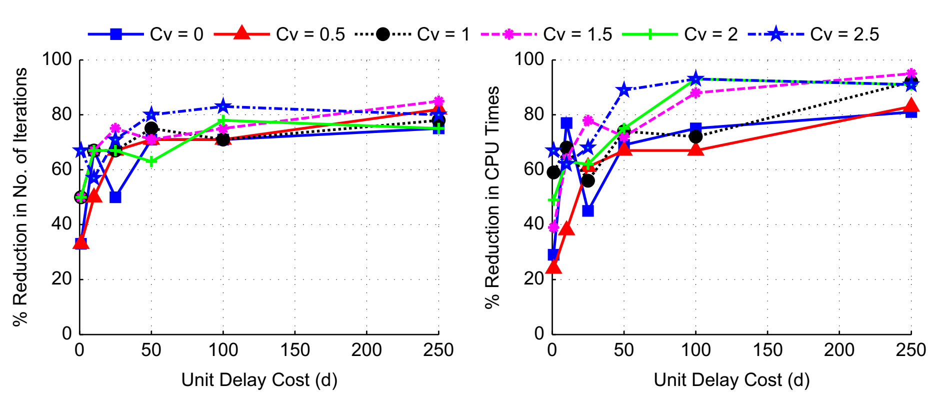

Fig. 1. Effect of adding a priori cuts on the number of iterations and the CPU times of the algorithm.

Table 2
Computational performance: instances with 100 user nodes and 10 potential facilities.

Cv	d	TC	FC (%)	AC (%)	DC (%)	# Facility	% Utilization Min.	Max.	Avg.	# Iter	CPU
0	1	42,405	47	53	0	7	89	98	67	2	2
	10	43,248	46	52	2	7	95	97	67	2	2
	25	43,962	46	51	2	7	89	94	64	3	3
	50	44,955	47	50	3	7	83	93	61	2	4
	100	46,308	47	48	5	7	81	89	59	2	3
	250	48,741	40	53	7	6	74	85	47	2	2
	500	51,438	34	56	10	5	72	81	38	2	2
	1000	55,569	36	51	13	5	65	72	34	3	3
	5000	79,578	36	32	32	6	48	56	32	2	2
0.5	1	42,429	47	53	0	7	89	98	67	2	2
	10	43,392	47	52	1	7	89	97	64	3	4
	25	44,186	46	51	3	7	89	94	64	2	2
	50	45,251	46	50	4	7	83	93	61	2	3
	100	46,720	46	48	6	7	81	89	59	2	4
	250	49,255	35	58	7	5	75	82	40	2	2
	500	52,233	37	54	9	5	69	73	35	3	3
	1000	56,507	35	50	14	5	65	72	34	3	2
	5000	81,897	35	31	34	6	48	56	32	2	2
1	1	42,501	47	53	0	7	89	98	67	2	1
	10	43,684	47	51	2	7	89	94	64	2	2
	25	44,802	47	50	3	7	83	93	61	2	4
	50	46,013	47	49	4	7	81	89	59	2	4
	100	47,555	41	54	5	6	74	85	47	2	3
	250	50,479	35	57	8	5	72	81	38	2	1
	500	53,859	37	53	10	5	65	70	34	3	2
	1000	59,040	35	48	17	5	61	70	33	2	2
	5000	88,546	41	25	34	7	42	52	32	3	8
1.5	1	42,615	47	52	1	7	95	98	67	2	2
	10	44,135	46	51	3	7	89	94	64	2	2
	25	45,525	46	50	4	7	83	92	61	2	3
	50	46,920	47	48	5	7	74	85	56	2	4
	100	48,671	41	53	6	6	72	81	46	2	2
	250	52,162	37	54	8	5	69	73	35	2	2
	500	56,163	37	50	12	5	61	70	33	2	2
	1000	62,778	35	45	20	5	62	66	32	2	3
	5000	97,044	38	23	39	7	42	48	32	3	17
2	1	42,763	47	52	1	7	95	97	67	2	1
	10	44,711	47	50	3	7	83	93	61	2	3
	25	46,278	47	48	5	7	81	89	59	2	4
	50	47,791	41	54	5	6	74	85	47	3	4
	100	49,903	35	57	7	5	72	81	38	2	1
	250	53,941	37	53	10	5	65	70	34	3	2
	500	59,023	35	48	17	5	61	70	33	2	3
	1000	66,506	42	38	20	6	49	59	33	3	4
	5000	107,202	41	19	40	8	36	44	32	3	16
2.5	1	42,954	47	52	1	7	95	97	67	2	2
	10	45,238	46	50	4	7	83	93	61	3	5
	25	46,993	47	48	5	7	74	85	56	2	4
	50	48,735	41	53	6	6	72	81	46	2	3
	100	51,249	38	55	7	5	69	73	35	2	2
	250	55,950	37	51	12	5	61	70	33	2	2
	500	61,983	43	41	16	6	48	60	34	3	5
	1000	70,952	40	36	23	6	48	56	32	3	5
	5000	118,106	44	15	41	9	34	39	32	3	13
	Min.	42,405	34	15	0	5	34	39	32	2	1
	Avg.	54,645	42	48	10	6	72	80	48	2	4
	Max.	118,106	47	58	41	9	95	98	67	3	17

Table 3
Computational performance: instances with 200 user nodes and 15 potential facilities.

Cv	d	TC	FC (%)	AC (%)	DC (%)	# Facility	% Utilization Min.	Max.	Avg.	# Iter	CPU
0	1	64,897	55	44	0	14	91	99	90	2	18
	10	66,409	55	43	2	14	76	98	87	2	18
	25	67,976	54	43	3	14	78	95	84	2	23
	50	69,594	55	41	4	14	71	93	80	2	20
	100	72,116	49	46	5	12	67	90	66	3	60
	250	76,189	42	51	7	9	62	82	47	2	26
	500	80,387	42	49	9	9	56	76	42	2	17
	1000	86,593	43	45	12	9	45	72	37	2	14
	5000	120,794	36	33	32	9	48	60	32	3	28
0.5	1	64,961	55	44	1	14	91	99	90	2	16
	10	66,685	55	43	2	14	88	96	87	2	22
	25	68,372	55	42	3	14	76	94	82	2	23
	50	70,108	55	41	5	14	71	93	80	2	19
	100	72,765	49	45	6	12	70	87	66	2	31
	250	76,982	43	51	7	9	62	82	45	3	40
	500	81,364	43	48	9	9	50	76	40	2	15
	1000	87,860	42	44	13	9	50	72	37	3	10
	5000	124,123	40	29	31	10	41	55	32	5	57
1	1	65,136	55	44	1	14	91	99	90	2	19
	10	67,401	55	42	3	14	88	96	87	4	54
	25	69,295	55	41	3	14	71	93	80	2	19
	50	71,555	54	41	5	14	75	89	77	3	48
	100	74,309	45	50	5	10	67	83	52	2	32
	250	78,810	43	50	7	9	56	76	42	2	23
	500	83,774	44	46	10	9	46	72	38	2	17
	1000	91,294	41	43	15	9	46	68	36	3	17
	5000	132,533	38	27	35	10	43	51	32	4	35
1.5	1	65,396	55	44	1	14	92	98	90	2	22
	10	68,258	55	42	3	14	76	94	82	2	23
	25	70,557	54	41	5	14	75	93	79	2	24
	50	73,197	49	46	5	12	67	87	64	3	59
	100	76,017	45	48	6	10	64	82	51	3	35
	250	81,115	43	48	9	9	51	76	40	2	20
	500	86,919	43	45	12	9	50	70	37	3	17
	1000	95,844	42	41	17	9	45	65	34	3	23
	5000	144,737	39	24	38	11	38	48	32	4	53
2	1	65,675	56	43	1	14	76	98	87	2	16
	10	69,116	56	41	3	14	71	93	80	2	21
	25	72,011	54	41	6	14	75	89	77	2	36
	50	74,649	45	50	5	10	67	83	52	3	42
	100	77,852	43	50	7	9	56	79	43	2	23
	250	83,654	44	46	9	9	50	72	37	3	15
	500	90,632	43	44	13	9	45	67	35	3	22
	1000	101,278	46	36	18	10	41	56	34	5	52
	5000	159,170	39	20	40	12	37	44	32	4	43
2.5	1	65,957	56	43	1	14	76	98	87	2	20
	10	70,031	55	41	4	14	73	93	80	2	20
	25	73,347	49	45	5	12	70	86	64	3	53
	50	76,108	45	48	6	10	64	80	51	3	40
	100	79,628	44	49	7	9	50	76	40	2	24
	250	86,404	44	45	11	9	50	68	37	2	16
	500	94,597	42	42	16	9	49	65	34	4	32
	1000	106,997	45	34	22	10	44	55	33	3	34
	5000	175,392	40	17	43	13	33	41	32	6	64
	Min.	64,897	36	17	0	9	33	41	32	2	10
	Avg.	84,015	48	42	10	11	62	80	57	3	29
	Max.	175,392	56	51	43	14	92	99	90	6	64

- The proposed solution approach succeeds in finding exact (within an optimality gap of 10^{-5}) solutions to the four sets of problem instances ($|I|=100, |J|=10; |I|=200, |J|=15; |I|=300, |J|=20; |I|=400, |J|=25$) within an average CPU time of 4, 29, 100 and 323 s, respectively, while the maximum CPU times for these sets are 17, 64, 668, and 1585 s, respectively. The average number of iterations are 2, 3, 3, and 3 whereas the maximum number of iterations are 3, 6, 6, and 6, respectively for the four sets of problem instances. This reveals the efficiency of the solution approach where the number of iterations/cuts implies that only a fraction of the exponential number constraints is required.

Table 4
Computational performance: instances with 300 user nodes and 20 potential facilities.

Cv	*d*	TC	FC (%)	AC (%)	DC (%)	# Facility	% Utilization Min.	% Utilization Max.	% Utilization Avg.	# Iter	CPU
0	1	87,130	57	43	1	18	84	99	87	2	668
	10	89,241	56	42	2	18	91	97	84	3	52
	25	91,380	55	41	3	18	83	96	83	3	56
	50	93,915	54	41	5	18	85	92	81	3	71
	100	97,458	55	39	6	18	76	90	77	3	220
	250	103,111	47	45	8	14	61	84	54	2	75
	500	109,103	45	45	10	13	47	76	46	4	119
	1000	117,835	42	46	12	11	41	72	36	3	72
	5000	163,701	36	33	31	12	42	61	32	3	93
0.5	1	87,242	57	43	1	18	86	99	87	2	262
	10	89,583	56	42	2	18	91	97	84	2	38
	25	91,976	55	41	4	18	83	95	82	2	35
	50	94,704	55	41	5	18	76	92	79	2	48
	100	98,312	49	45	5	15	76	88	62	2	91
	250	104,193	46	47	7	13	56	79	48	3	91
	500	110,477	45	45	10	13	47	75	44	2	50
	1000	119,613	41	46	13	11	46	71	36	4	93
	5000	167,997	39	29	32	13	40	57	32	3	66
1	1	87,523	56	43	1	18	91	99	87	3	114
	10	90,593	55	42	3	18	91	96	84	2	59
	25	93,513	55	41	4	18	85	92	81	3	70
	50	96,690	55	39	6	18	76	90	77	3	119
	100	100,348	48	46	6	14	61	85	56	4	128
	250	106,782	45	46	8	13	51	78	47	4	98
	500	113,941	43	46	11	12	46	74	40	3	76
	1000	124,301	40	45	14	11	41	67	34	2	47
	5000	179,921	37	28	35	13	44	52	32	5	110
1.5	1	87,845	56	43	1	18	92	98	87	2	56
	10	91,847	55	41	4	18	83	95	82	2	38
	25	95,403	55	40	5	18	76	90	77	2	71
	50	98,800	50	45	5	15	61	84	60	3	115
	100	102,896	48	46	6	14	56	81	52	3	108
	250	110,165	45	45	9	13	51	74	44	3	71
	500	118,582	42	46	12	11	46	69	35	3	78
	1000	130,498	42	41	17	12	42	64	34	4	120
	5000	197,199	41	22	36	15	38	49	32	4	112
2	1	88,279	56	43	1	18	92	98	86	2	123
	10	93,271	55	41	4	18	85	92	81	3	74
	25	97,338	56	39	5	18	68	87	74	4	202
	50	100,919	49	44	6	15	68	83	59	3	107
	100	105,374	46	47	7	13	51	78	47	3	79
	250	113,993	44	46	10	12	43	70	38	2	45
	500	123,494	44	43	13	12	44	64	35	3	74
	1000	137,761	46	36	18	13	40	59	33	3	77
	5000	216,365	44	19	37	17	34	42	32	4	88
2.5	1	88,695	56	42	1	18	86	97	84	2	38
	10	94,674	55	41	5	18	76	92	79	3	83
	25	98,978	50	45	5	15	61	84	60	2	73
	50	102,958	48	46	6	14	56	80	51	2	71
	100	107,960	46	46	8	13	47	74	44	3	76
	250	117,909	46	43	12	13	44	68	39	2	51
	500	128,813	44	41	15	12	42	64	33	3	87
	1000	145,395	45	34	21	13	42	57	32	5	116
	5000	238,223	44	16	40	18	33	38	32	6	150
	Min.	87,130	36	16	1	11	33	38	32	2	35
	Avg.	113,782	49	41	10	15	62	79	58	3	100
	Max.	238,223	57	47	40	18	92	99	87	6	668

Table 5
Computational performance: instances with 400 user nodes and 25 potential facilities.

Cv	*d*	TC	FC (%)	AC (%)	DC (%)	# Facility	% Utilization Min.	% Utilization Max.	% Utilization Avg.	# Iter	CPU
0	1	108,302	53	46	0	19	97	99	74	3	807
	10	110,755	53	45	2	19	90	97	72	2	70
	25	113,050	52	45	3	19	87	97	70	2	110
	50	115,745	52	44	4	19	78	94	67	3	241
	100	119,588	53	42	6	19	73	92	65	2	396
	250	126,660	50	43	7	17	66	85	52	2	332
	500	134,432	45	45	10	15	53	80	43	3	206
	1000	145,174	43	44	13	14	48	76	37	3	106
	5000	203,766	38	30	32	15	45	64	32	4	242
0.5	1	108,430	53	46	1	19	97	99	74	3	988
	10	111,135	53	45	2	19	92	97	71	3	98
	25	113,694	53	44	3	19	82	96	69	3	150
	50	116,659	53	43	4	19	78	92	66	4	1033
	100	120,705	52	43	6	18	78	90	60	2	511
	250	128,061	50	42	7	17	62	83	51	3	237
	500	136,165	45	45	10	15	54	79	42	4	192
	1000	147,499	42	44	14	14	48	76	36	2	65
	5000	209,065	41	27	31	16	40	56	32	4	149
1	1	108,768	53	46	1	19	86	98	73	2	558
	10	112,176	52	45	2	19	81	97	70	2	75
	25	115,324	52	44	4	19	78	94	67	3	229
	50	118,776	53	42	5	19	73	92	65	2	396
	100	123,345	50	44	6	17	68	87	54	4	945
	250	131,521	50	41	9	17	58	81	49	3	183
	500	140,513	46	43	11	15	48	75	39	3	112
	1000	153,517	43	42	15	14	46	71	34	3	131
	5000	224,302	41	25	34	17	39	53	32	5	335
1.5	1	109,115	53	46	1	19	86	98	73	2	93
	10	113,548	53	44	3	19	84	94	69	3	151
	25	117,369	54	42	4	19	73	92	65	4	707
	50	121,414	51	43	6	18	73	88	59	3	734
	100	126,409	50	43	7	17	62	83	51	4	335
	250	135,945	46	45	9	15	54	78	41	3	181
	500	146,324	44	44	12	14	48	74	35	5	208
	1000	161,516	42	40	17	14	45	68	33	3	113
	5000	245,741	43	21	36	19	37	51	32	4	175
2	1	109,592	53	46	1	19	88	98	73	2	100
	10	115,071	52	45	3	19	78	94	68	3	218
	25	119,552	53	42	5	19	78	88	64	3	771
	50	124,030	51	44	6	17	69	85	53	2	350
	100	129,675	50	42	8	17	58	80	49	3	162
	250	140,567	46	43	10	15	48	74	39	3	114
	500	152,752	44	42	13	14	46	69	33	5	240
	1000	170,954	48	33	19	16	43	58	33	4	177
	5000	269,628	42	18	40	20	35	45	32	4	122
2.5	1	110,130	53	45	1	19	82	98	72	2	181
	10	116,603	54	43	4	19	78	93	66	2	373
	25	121,739	51	43	5	18	73	88	58	5	1585
	50	126,527	51	43	6	17	62	83	51	2	97
	100	133,103	50	41	9	17	57	78	47	2	82
	250	145,560	46	42	12	15	47	71	37	3	133
	500	159,695	46	39	15	15	45	66	34	6	369
	1000	181,019	46	32	22	16	43	56	33	3	146
	5000	298,560	43	16	41	22	32	40	32	5	621
	Min.	108,302	38	16	0	14	32	40	32	2	65
	Avg.	140,727	49	41	10	17	64	81	52	3	323
	Max.	298,560	54	46	41	22	97	99	74	6	1585

6. Conclusions

In this paper, we presented a class of location–allocation problems with immobile servers, stochastic demand and congestion. The model captures the trade-off among the fixed cost of opening service facilities and equipping them with sufficient capacities, the access cost associated with users' travel to service facilities, and the queueing delay cost associated with customers waiting for service. Under the assumption that the customer demands follow a Poisson process and service times follow a general distribution, the facilities were modeled as a network of independent M/G/1 queues, whose locations, capacity levels and workload allocations are decision variables. We presented a non-linear IP formulation and a constraint generation based exact method to solve its linear MIP reformulation. The computational results indicate that the proposed approach provides optimal solution for a wide range of problem instances

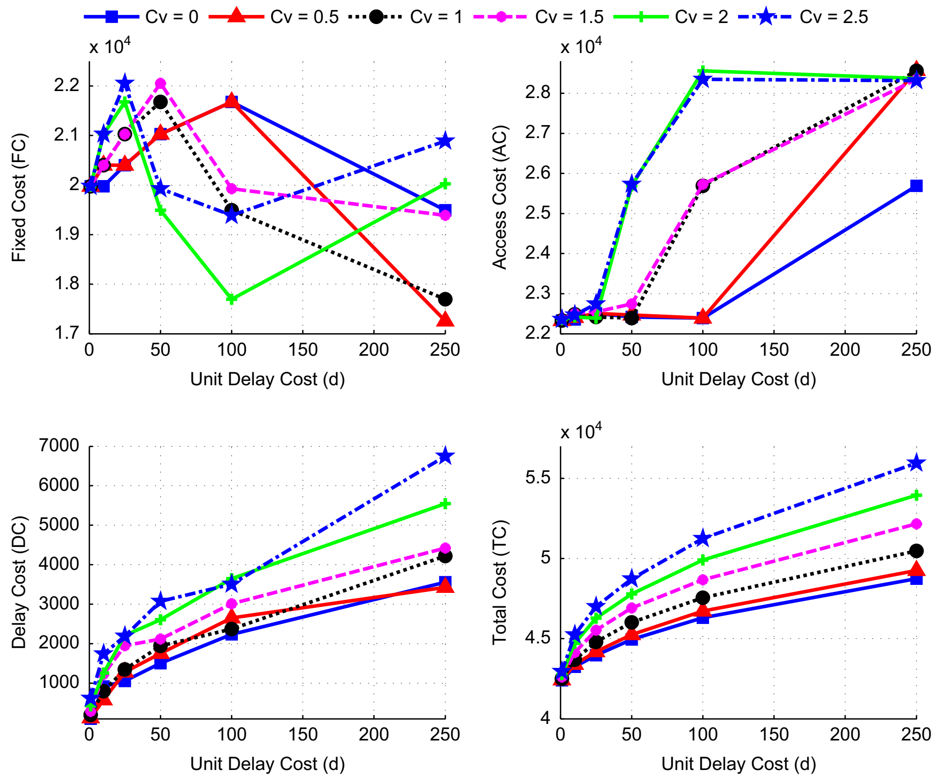

Fig. 2. Effect of varying unit delay cost on the fixed cost, access cost, delay cost, and the total cost.

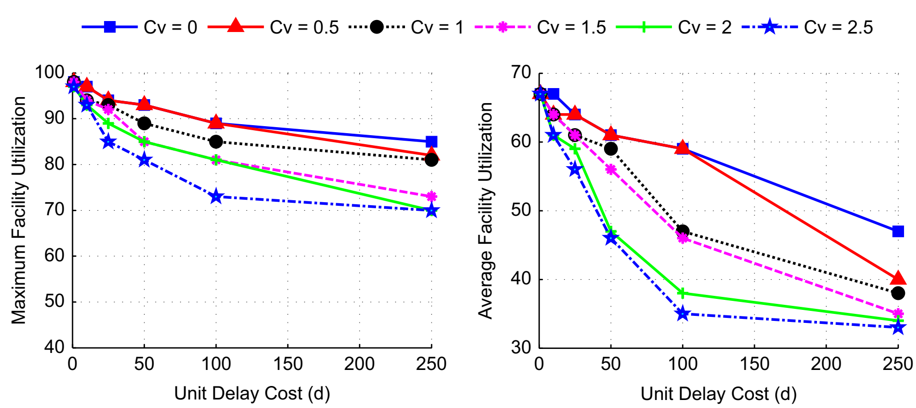

Fig. 3. Effect of varying unit delay cost on maximum and average facility utilization.

within reasonable computation times. Future research directions may include extending the proposed solution procedures to deal with systems with multiple servers (M/G/s) and general demand processes (G/G/s).

Acknowledgments

This research was supported by the Discovery grant from National Science and Engineering Research Council of Canada, provided to the first author, and by the Research and Publication Grant, Indian Institute of Management Ahmedabad, provided to the second author. The authors also acknowledge the assistance provided by Ankit Bhagat and Vikranth Babu in the computational study. The paper has also benefited from the anonymous referees' comments.

References

[1] Aboolian R, Berman O, Drezner Z. Location and allocation of service units on a congested network. IIE Trans 2008;40:422–33.

[2] Aboolian R, Berman O, Krass D. Profit maximizing distributed service system design with congestion and elastic demand. Transp Sci 2012;46:247–61.

[3] Aboolian R, Sun Y, Koehler GJ. A location–allocation problem for a web services provider in a competitive market. Eur J Oper Res 2009;194:64–77.

[4] Amiri A. Solution procedures for the service system design problem. Comput Oper Res 1997;24:49–60.

[5] Amiri A. The design of service systems with queuing time cost, workload capacities, and backup service. Eur J Oper Res 1998;104:201–17.

[6] Amiri A. The multi-hour service system design problem. Eur J Oper Res 2001;128:625–38.

[7] Baron O, Berman O, Krass D. Facility location with stochastic demand and constraints on waiting time. Manuf Serv Oper Manag 2008;10:484–505.
[8] Berman O, Krass D. Facility location problems with stochastic demands and congestion. In: Drezner Z, Hamacher H, editors. Facility location: applications and theory. Berlin-Heidelberg: Springer; 2004. p. 329–71.
[9] Berman O, Krass D, Wang J. Locating service facilities to reduce lost demand. IIE Trans 2006;38:933–46.
[10] Boffey B, Galvao R, Espejo L. A review of congestion models in the location of facilities with immobile servers. Eur J Oper Res 2007;178:643–62.
[11] Castillo I, Ingolfsson A, Sim T. Socially optimal location of facilities with fixed servers, stochastic demand, and congestion. Prod Oper Manag 2009;18: 721–36.
[12] Elhedhli S. Exact solution of class of nonlinear knapsack problems. Oper Res Lett 2005;33:615–24.
[13] Elhedhli S. Service system design with immobile servers, stochastic demand and congestion. Manuf Serv Oper Manag 2006;8:92–7.
[14] Espejo I, Marin A, Rodriguez-Chia A. Closest assignment constraints in discrete location problems. Eur J Oper Res 2012;219:49–58.
[15] Gross D, Harris CM. Fundamentals of queueing theory. 3 ed. New York: John Wiley and Sons; 1998.
[16] Holmberg K, Rönnqvist M, Yuan D. An exact algorithm for the capacitated facility location problems with single sourcing. Eur J Oper Res 1999;113:544–59.
[17] Huang S, Batta R, Nagi R. Distribution network design: selection and sizing of congested connections. Naval Res Logist 2005;52:701–12.
[18] Marianov V, Serra D. Probabilistic, maximal covering location–allocation models for congested systems. J Reg Sci 1998;38:401–24.
[19] Marianov V, Serra D. Location–allocation of multiple-server service centers with constrained queues or waiting times. Ann Oper Res 2002;111:35–50.
[20] Rojeski P, ReVelle C. Central facilities location under an investment constraint. Geogr Anal 1970:343–60.
[21] Silva F, Serra D. Locating emergency services with different priorities: the priority queuing covering location problem. J Oper Res Soc 2008;59:1229–38.
[22] Vidyarthi N, Elhedhli S, Jewkes E. Response time reduction in make-to-order and assemble-to-order supply chain design. IIE Trans 2009;41:448–66.
[23] Wang Q, Batta R, Rump CM. Algorithms for a facility location problem with stochastic customer demand and immobile servers. Ann Oper Res 2002;111: 17–34.
[24] Wang Q, Batta R, Rump CM. Facility location models for immobile servers with stochastic demand. Naval Res Logist 2004;51:138–52.
[25] Zhang Y, Berman O, Macotte P, Verter V. A bilevel model for preventive healthcare facility network design with congestion. IIE Trans 2010;42:865–80.
[26] Zhang Y, Berman O, Verter V. Incorporating congestion in preventive healthcare facility network design. Eur J Oper Res 2009;198:922–35.
[27] Zhang Y, Berman O, Verter V. The impact of client choice on preventive healthcare facility network design. OR Spectr 2012;34:349–70.