Exact and heuristic approaches to the budget-constrained dynamic uncapacitated facility location-network design problem

Abdolsalam Ghaderi
Iran University of Science and Technology, Tehran, Iran

Natashia Boland
University of Newcastle, Newcastle, Australia

Mohammad Saeid JabalAmeli
Iran University of Science and Technology, Tehran, Iran

January 27, 2012

Abstract

Facility location-network design problems seek to simultaneously determine the locations of facilities and the design of the network connecting the facilities so as to best serve a set of clients accessing the facilities via the network. Here we study a dynamic (multi-period) version of the problem, subject to a budget constraint limiting the investment in new facilities and network links in each period. We consider an uncapacitated setting, in which the objective is to minimize the combined travel cost for clients and operating costs of facilities and network links. We present a mixed-integer linear programming (MIP) model generalizing related models in the literature, derive some useful properties of solutions, and show how the model can be strengthened. Three heuristics based on sequential solution of MIPs, and an efficient hybrid heuristic based on variable neighborhood search (VNS), are presented. The models and algorithms are compared computationally on randomly generated instances.

Keywords: network design, dynamic (multi-period), sequential heuristic, hybrid VNS algorithm

1 Introduction

Facility location decisions are managerial strategic activities that often arise in important real-world applications. The general objective of facility location problems is to locate one or more facilities to service a set of demand points. The published literature in location theory, originating with the seminal paper of Weber in 1909 [1], has mostly concentrated on specific basic models, with single/multi-facility, covering, p-median and p-center location problems only some of the variants that have been considered. Extensions to these have been introduced more recently.

Network location models form a large and important subset of models in facility location. In these models, the underlying structure of the system in which facilities are to be located is a given network, such as a road network or a communication or transmission system. Fundamentally, the facility location-network design problem (FLNDP) is a combination of facility location and network design that involves the determination of the location of the facilities (as in facility location) required to satisfy a set of clients' demand and the determination of travelable links (as in network design) to connect clients to facilities. In these models, unlike pure facility location models, the network structure is not given, although potential links may be provided.

Generally, this class of location problem is useful for modeling real-world applications in which trade-offs between facility fixed costs, network design costs, and operating costs must be made. Such applications arise in regional planning, and in automated guided vehicles (AGVs) systems, as well as in the design of less-than-truckload (LTL) distribution systems, pipeline systems, power transmission networks, airline networks and telecommunications networks, to name just a few contexts. For example, in regional planning, the government may be simultaneously considering the construction of the new roadway system as well as the location of public and government facilities such as post offices, schools, fire stations, etc., with a limited budget [2]. Cocking [3] recently studied a specific real-world setting with the goal of improving access to health facilities for the people in the Nouna health district of Burkina Faso. In this study, remarkable improvements in accessibility to health facilities were obtained.

To date, most research on facility location-network design has focussed on static, one-period, models. Whilst such models have undoubtedly been of great benefit in practice, in many real-world situations budget constraints mean the system designed can only be built over an extended period, during which clients still need to be served. In such situations, static models do not provide the construction schedule that would best meet clients' demands during the construction period: dynamic, multi-period, models are required. Furthermore, few real-world systems are static, with changes in demographics, for example, leading to shifts in demand, motivating changes such as facility closures in some areas and new facilities opened in others. Again, dynamic models are needed to continuously re-design facility networks in the presence of parameters that change over time.

In this paper, the budget-constrained dynamic uncapacitated facility location-network design problem (DUFLNDP), is investigated. Parameters such as client demand, facility and network link costs, are allowed to vary by period. In each period, decisions about which new network links to construct, which facilities to open, and which to close, are made subject to a budget constraint. The objective is to minimize the combined client traveling cost and network (including facility) operating costs over all periods in a finite time horizon, while ensuring that in each period, all demand is fully routed through the network to an open facility. The budget-constrained DUFLNDP is clearly NP-hard, since even the single-period problem is NP-hard.

We believe this is the first paper to consider facility location and network design decisions simultaneously in conditions where the problem parameters change over time, and budget constraint effects are also seen. Whilst the particular problem setting studied here is a natural extension to those in the literature, we were motivated by an application involving the opening of healthcare facilities and road upgrades in rural areas of Iran. We discuss this further in motivating the particular assumptions of the problem definition. The key contributions in this paper are:

1. the introduction of a new dynamic variant of the combined facility location and network design problem, and a preliminary analysis of its properties;
2. a mixed integer linear programming (MIP) formulation of the problem, exploiting those properties;
3. sequential MIP heuristics for finding initial feasible solutions;
4. a variable neighbourhood search (VNS) method; and
5. the results of computational comparisons of the MIP model and the heuristics.

The paper is structured as follows. A review of the literature is given in Section 2. The problem is formally defined in Section 3, analysis deriving some useful properties of solutions carried out, and formulations proposed. In Section 4, the heuristics are described, and in Section 5, the results of computational testing are presented.

2 Literature review

There is a rich literature in each of the facility location and network design areas. The first work on facility location dates back to the Weber problem [1]. Since then, many papers have been published that provide admirable introductions, surveys and reviews of problem variants and developments in the field [4, 5, 6, 7, 8]. Network design has also received a great deal of attention, and it, too, encompasses many problem variations. The seminal work of Magnanti and Wong [9] discusses a number of these; see also [10] and the reviews in [11, 12]. It continues to be an active area of research: see for example the work of Costa et al. [13] and Hewitt et al. [14], and references therein.

Below we will focus on literature which (i) combines facility location and network design, i.e. the literature on facility location-network design problems, and (ii) addresses dynamic problems, in particular dynamic facility location. We have been able to identify no work which treats either network design, or facility location-network design, in a dynamic setting.

2.1 The facility location-network design problem

Early work that combines elements of facility location and network design is carried out by Berman [15], who investigated the relationship between facility location and network topology. However the form of facility location-network design problem closest to that studied here originated with Daskin et al. [16], who introduced the uncapacitated facility location-network design problem (UFLNDP). Later, Melkote, in his doctoral thesis [2], investigated three models for facility location-network design, including the UFLNDP, the capacitated facility location-network design problem (CFLNDP) and the maximum covering location-network design problem (MCLNDP); see also [17, 18]. In this work, a number of polynomially solvable special cases were identified, and UFLNDP and CFLNDP test problems with up to 40 nodes and 160 candidate links were solved.

Drezner and Wesolowsky [19] introduced a new network design problem with potential links where each link can be either constructed or not at a given cost. In addition, each constructed link can be constructed either as a one-way or two-way link. Four basic problems were created subject to two objective functions. Then, these problems were solved by a descent algorithm, Simulated Annealing (SA), Tabu Search (TS) and a Genetic Algorithm (GA).

Ravi and Sinha [20] and Chen and Chen [21] proposed some approximation algorithms for integrated logistics problems that combine elements of facility location and transport network design. These algorithms were introduced for the problem combining facility location and cable installation in which capacity constraints are imposed on both facilities and cables.

In a recent doctoral thesis, Cocking [3] solved the budget-constrained UFLNDP. A number of algorithms were introduced to find good upper bounds and good lower bounds on the optimal solution. Heuristics that were developed are: simple greedy heuristics, a local search heuristic, metaheuristics including SA and VNS, as well as a custom heuristic based on the problem-specific structure of UFLNDP. Also, a branch-and-cut method that uses heuristic solutions as upper bounds, and cutting planes for increasing the lower bound at each node of the problem tree were developed. This approach reduces the number of nodes needed to solve to optimality. Some of the results from this thesis were published in [22].

Bigotte et al. [23] presented a mixed-integer optimization model for integrated urban hierarchy and transportation network planning. This model simultaneously determines which urban centers and which network links should be promoted to a new level of hierarchy so as to maximize accessibility to all classes of facilities.

Other problems related to facility location-network design include location-routing, and hub location. In the former, facility location in a network is evaluated by solving a vehicle routing problem, rather than a network (flow) design problem; see, for example, the survey by Nagy and Salhi [5]. In hub location, demand must be routed to specified destinations via the (hub) facilities, rather than routed to the facility itself. In recent years, hub location variants in which network design elements, such as capacity expansion on arcs, or hub arc location, have been of increasing interest; see, for example, the survey by Alamur and Kara [6], and references therein.

2.2 Dynamic facility location

Dynamic facility location models appeared as early as the 1960's, with the work of Manne [24, 25] and Ballou [26]. Subsequent research has been vigorous, with a number of problem variants considered. As early as 1998, Owen and Daskin [27] include discussion of dynamic variants in their review of facility location. In his 2006 review, Snyder [28] focusses on facility location under uncertainty, which includes a review of work on stochastic dynamic problems. The very comprehensive review of Melo et al. [8] also discusses dynamic variants. Dynamic facility location was itself the focus of a 2012 review by Arabani and Farahani [29]. Below we give a brief selection of some of the research on dynamic facility location and closely related models, which is intended to give the reader the flavor of the field; for comprehensive treatment we refer the reader to the above-mentioned surveys and reviews.

The earliest work on dynamic facility location focussed on single facility problems. Ballou [26] provided a heuristic solution procedure based on dynamic programming to find a warehouse location-relocation plan that produces maximum aggregate incomes for a given planning period. Sweeney and Tatham [30] improved on this heuristic, applying Benders' decomposition to obtain solutions. Later papers considering dynamic single facility problems include [31, 32, 33, 34, 35].

Before long, multi-facility dynamic variants began to be addressed. Location-allocation, p-median, p-center, p-dispersion are only some of the models considered. Scott [36] introduced two major approaches to the formulation and analysis of dynamic location-allocation problem. Wesolowsky [37] proposed a dynamic multi-facility minisum problem, in which opening of new facilities and closing of new and existing facilities can occur during the planning horizon. This problem was reformulated in [38] and solved by dynamic programming. Later papers considering dynamic multi-facility location models include [39, 40, 41, 42, 43, 44, 45, 46].

The most recent of these, by Melo et al. [46], includes consideration of inventory, multiple products, and multiple echelons of facility. Dynamic multi-echelon facility location has been an important strand of research, not least because of its applications in logistics network design and distribution systems. In earlier work, Hinojosa et al. [47] combined multi-period aspects with two-echelon facility location in a multi-commodity distribution network, proposing a Lagrangian relaxation to solve the problem. This problem was later extended by Velten [48] to include inventory considerations. The approach of Melo et al. [46] combines a LP rounding heuristic with local search, which comprehensive testing shows is able to obtain solutions close to those found by solving an exact MIP model, in much less computing time. Thanh et al. [49] also treat dynamic multi-echelon, multi-commodity facility location problems with a LP rounding heuristic, but combine it with an adaptation of the Feasibility Pump heuristic ([50]) as a corrective measure.

The majority of work on dynamic facility location assumes deterministic parameters. However, there is a growing body of research on stochastic models. For example, Romauch and Hartl [51] propose an exact solution method based on stochastic dynamic programming for solving small test problems, as well as a Monte Carlo based method for solving larger instances. A two-stage stochastic programming model was developed for the dynamic reverse logistics network design in Lee and Dong [52], together with an efficient algorithm based on an integration of sample average approximation and simulated annealing.

Many algorithms and solution approaches have been developed to solve dynamic location problems. For example, exact methods such as dynamic programming [30], Benders' decomposition [53] and branch and bound algorithm [54], have been applied, as have meta-heuristics including SA [40], TS [55] and GA [56], hybrid heuristic such as GA with local search [44, 45], a hybrid GA [56] and a hybrid SA [52], as well as other heuristics such as the perfect forward algorithm [31] or the recent LP rounding based approaches of [46, 49].

Related models such as location-routing and hub location, discussed in the previous section, have also been studied in dynamic form. Chanchan et al. [57] presented an integrated optimization model for the stochastic dynamic location-routing-inventory problem. In their paper, a two-phase heuristic algorithm was also proposed to solve the model, and the validity of the model and algorithm was demonstrated by an example. Very recently, Contreras et al. [58] address a dynamic hub location problem, and note the scarcity of earlier work on such problems.

3 Mathematical formulations

In this section, a formulation of the budget-constrained DUFLNDP is proposed. First, the problem is formulated as a Mixed-Integer Nonlinear Programming (MINLP) model. In this model, some of the constraints have quadratic terms. We discuss two alternative linearizations of these, and then provide properties of optimal solutions that can be exploited to improve the formulation.

3.1 Problem description

The DUFLNDP is defined on a network made by a set of clients, a set of potential locations for facilities, a set of potential links for constructing the network, and a time horizon described in terms of a set of consecutive time periods. At each time period, to optimize access to facilities, an unspecified number of facilities and potential links could be opened. Some open facilities may also be closed, but links, once built (opened), may not be closed. The total cost of opening facilities, closing facilities, and opening links, is constrained by an available budget for the period, where any unspent budget in a period is lost; there is no carry-over to future periods. Note that once open, facilities and links incur operating costs whether they are used (active) or not (inactive). A client's demand is met by travel along open links in the network to open facilities. Such travel incurs a cost per unit of demand travelling on each link. Since there are no link or facility capacities in our model, it can be assumed without loss of generality that each client's demand is met by a single facility, and that it travels on a single, least cost path from the client to the facility.

An illustration of the DUFLNDP structure for a network with 10 nodes and 22 links is depicted in Figure 1.

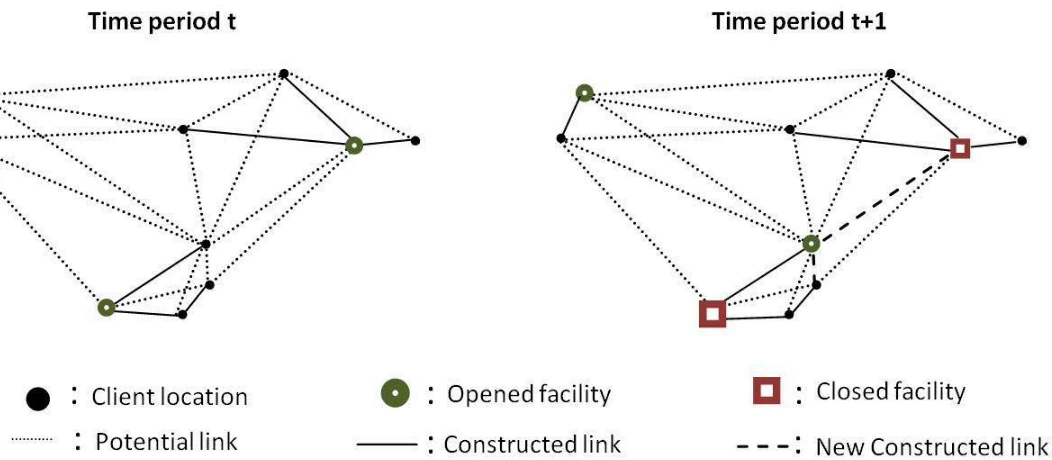

Figure 1: An illustration of the solution to a dynamic uncapacitated facility location-network design problem with 10 nodes and 22 links, over two consecutive time periods.

Our motivation for studying this particular variant came from an application in the location of healthcare services in rural Iran, where roads between population centres are sparse. The Government proposed a program of works over a number of years, to provide new healthcare facilities and build new roads so that the population could access those facilities. Roads represented a much greater capital investment than the healthcare facilities in this environment, and so whilst healthcare centres in interim locations were considered (which could subsequently be closed), it was decided that roads, once built, should be maintained. As is often the case, the Government's budget for this project did not return unspent funds in each year to the project.

It is assumed that the following problem parameters can vary over the planning horizon:

- the available budget,
- client demand,
- the opening cost of facilities,
- the closing cost of facilities,
- the operating cost of facilities,
- the opening cost of links,
- the operating cost of links, and
- the travelling cost between nodes.

In the motivating application, client demand changes, for example, reflect expected demographic shifts in the population, whilst costs might change in line with inflation.

The proposed model can readily be adapted to either a "greenfields" planning situation, in which no facilities or links are open at the start of the time horizon, or to take as input an existing set of infrastrcuture. In what follows, the latter, more general, case is modeled.

3.2 Notation and assumptions

The assumptions of the model are summarized as follows. Those proposed for the UFLNDP by Daskin et al. [16] are also adopted here: (1) each node represents a demand point, (2) facilities may be located only on the nodes of the network, (3) only one facility may be located per node, (4) the network is a customer-to-server system, in which the demands themselves travel to the facilities to be served, (5) the facilities are uncapacitated, i.e., they may serve an unlimited amount of demand, and (6) demand at a node at which an open facility is located can be met at that node without cost, and so without loss of generality is assumed so. In addition to these assumptions, we assume that: (7) unspent budget at the end of each time period is returned or reset, (8) the links are uncapacitated, (9) parameters may change over time (as discussed above), (10) travel cost per unit demand is independent of client, (11) once a link is built, it remains open throughout the time horizon but opened facilities may be closed in the subsequent periods, (12) opening and closing of facilities as well as construction of links is instantaneous, (13) opening of a facility or link must happen at the beginning of a time period, and (14) closing of a facility must happen at the end of a time period.

The notation used in the model of the problem is based on that of Melkote and Daskin [17] in their treatment of the single period UFLNDP. The index sets are described in Table 1.

Table 1: Index sets

Symbol	Indexed by	Description
N	$i, j \in \{1, 2, ..., \|N\|\}$ $k \in \{1, 2, ..., \|N\|\}$	Set of network nodes; and Set of clients
T	$t \in \{1, 2, ..., \|T\|\}$	Set of time periods
N^0	$N^0 \subseteq \{1, 2, ..., \|N\|\}$	Set of opened facilities in the existing network before the first period
L^t	$(i, j) \in L^t$	Set of potential links at time period $t = 1, \ldots, T$
L^0	$(i, j) \in L^0 \subseteq L^1$	Set of constructed links in the existing network before the first period

The infrastructure prior to the start of the planning time horizon is given by facilities in N^0 and links in L^0. At any time period, the set of possible links that could be build is given by L^t, so it is assumed that $L^0 \subseteq L^1 \subseteq \cdots \subseteq L^T$.

Input parameters are described in Table 2.

Table 2: Problem parameters

Symbol	Description
d_k^t	The demand of client k at time period t
ρ_{ij}^t	Travel cost per unit flow on link (i, j) at time period t (independent of client)
$r_{ij}^{kt} = d_k^t \rho_{ij}^t$	The cost of sending all of client k's demand on link (i, j) in time period t
f_i^t	Operating cost of opened facility on node i during time period t
c_{ij}^t	Operating cost of constructed link on (i, j) during time period t
g_i^t	Fixed cost of opening a facility on node (i, j) at time period t
p_i^t	Fixed cost of closing a facility on node (i, j) at time period t
h_{ij}^t	Fixed cost of constructing link (i, j) at time period t
B^t	Budget in time period t

3.3 Model formulation

We use the natural binary variables to indicate whether or not a facility is open:

$$Z_i^t = \begin{cases} 1 & \text{if facility } i \text{ is open at the beginning of time period } t; \\ 0 & \text{Otherwise} \end{cases}$$

for all $i = 1, \ldots, N$, $t = 1, \ldots, T$. Links are undirected, so we use binary variables

$$X_{ij}^t = \begin{cases} 1 & \text{if link } (i, j) \text{ is open at the beginning of time period } t; \\ 0 & \text{Otherwise} \end{cases}$$

for all $(i, j) \in L^t$ with $i < j$, $t = 1, \ldots, T$ to indicate whether or not a link is open. We use the obvious multi-period form of the stronger multicommodity flow variables suggested in [17]:

$$\begin{aligned} Y_{ij}^{kt} = & \quad \text{fraction of client } k\text{'s demand traveling } i \text{ to } j \text{ at time period } t, \text{ and} \\ W_i^{kt} = & \quad \text{fraction of client } k\text{'s demand served by facility } i \text{ at time period } t \end{aligned}$$

for all $k \in N$, $(i, j) \in L^t$ in the former case, and all $i, k \in N$ in the latter, and for all $t = 1, \ldots, T$.

We first give the model as a MINLP, using a nonlinear constraint to model the budget constraint. Note we abuse notation somewhat in this constraint, to allow variables with time superscript $t = 0$, and to allow link building variables for $(i,j) \in L^t \setminus L^{t-1}$ in period $t-1$. This will be corrected when we discuss linearizing these constraints. The formulation $P(X,Y,Z,W)$ is now defined as:

$$\min \quad \sum_{t\in T}\sum_{(i,j)\in L^t}\sum_{k\in N} r_{ij}^{kt} Y_{ij}^{kt} + \sum_{t\in T}\sum_{i\in N} f_i^t Z_i^t + \sum_{t\in T}\sum_{(i,j)\in L^t: i<j} c_{ij}^t X_{ij}^t \tag{1}$$

s.t.

$$Z_i^t + \sum_{j:(i,j)\in L^t} Y_{ij}^{it} = 1 \qquad \forall i \in N, \forall t \in T, \tag{2}$$

$$\sum_{j:(i,j)\in L^t} Y_{ji}^{kt} = \sum_{j:(i,j)\in L^t} Y_{ij}^{kt} + W_i^{kt} \quad \forall i,k \in N, i \neq k, \quad \forall t \in T, \tag{3}$$

$$Z_k^t + \sum_{i\in N: i\neq k} W_i^{kt} = 1 \qquad \forall k \in N, \forall t \in T, \tag{4}$$

$$Y_{ij}^{kt} + Y_{ji}^{kt} \leq X_{ij}^t \qquad \forall (i,j) \in L^t, i<j, \forall k \in N, \forall t \in T, \tag{5}$$

$$W_i^{kt} \leq Z_i^t \qquad \forall i,k \in N, i \neq k, \forall t \in T, \tag{6}$$

$$X_{ij}^t \geq X_{ij}^{t-1} \qquad \forall (i,j) \in L^t, i<j, \forall t \in T, t \leq |T|, \tag{7}$$

$$\sum_{i\in N} g_i^t Z_i^t (1 - Z_i^{t-1}) + \sum_{i\in N} p_i^t Z_i^{t-1}(1 - Z_i^t) + \sum_{(i,j)\in L^t: i<j} h_{ij}^t X_{ij}^t (1 - X_{ij}^{t-1}) \leq B^t \qquad \forall t \in T, \tag{8}$$

$$Y_{ij}^{kt} \geq 0 \qquad \forall (i,j) \in L^t, \forall k \in N, \forall t \in T, \tag{9}$$

$$X_{ij}^t \in \{0,1\} \qquad \forall (i,j) \in L^t, i<j, \forall t \in T, \tag{10}$$

$$W_i^{kt} \geq 0 \qquad \forall i,k \in N, k \neq i, \forall t \in T, \tag{11}$$

$$Z_i^t \in \{0,1\} \qquad \forall i \in N, \forall t \in T. \tag{12}$$

The objective function minimizes only the traveling and operating costs during the planning horizon. These costs are composed of the sum of transportation costs and the operating costs of the facilities and of the network. Investment costs, which include the opening cost of facilities at potential locations, the closing cost of opened facilities and the construction cost of links at potential links, are not considered in the objective function, but are subject to the maximum available budget in each time period.

Equations (2)-(4) are the flow conservation conditions, which must hold for each client, facility and period. Constraint (2) ensures that demand at i is either served by a facility at i or travels on some link out of i. Already in constraint (2) differences to the Melkote and Daskin [17] model emerge. In the static setting, it is possible to deduce that if link (i,j) is open, then client i's demand must travel on link (i,j), i.e. $Y_{ij}^i = X_{ij}$ in the model of [17], (note the lack of t superscript), so the latter variable is used in the constraint equivalent to (2). In our dynamic setting, we can make no such deduction. It is quite possible that an open link was built and used at some earlier period, but subsequent changes in facility and link structure make it unhelpful in the current period. Even if the link is first constructed in the current period, it is not necessarily used: if future budgets are tight, it may be being built in advance of when it is needed. Constraint (3) requires flow balance at node i in time period t for client k. Constraint (4) asks that in time period t client k's demand must find a destination, whether it be at node k itself ($Z_k^t = 1$) or at one of the other nodes ($\sum_{i\in N: i\neq k} W_i^{kt} = 1$).

The second group of constraints (5)-(6) establish the relationship between flow variables and infrastructure building variables, so that potential links and facilities are not used unless they are open. Note that again we see a difference to the static setting. In the static case, it can be deduced that a link is only ever used in one direction. Thus the link building variables can be taken to be directed, with a constraint to require that only one direction is needed, and only the relationship between flow and link variables in the same direction is required. In the dynamic case with a budget constraint with no carry forward, it is not difficult to construct examples in which a link is used in one direction in one period, and in the opposite direction in a later period. We thus model the link building variables as undirected, and ask only that if a link is used in either direction, then it is open. This could be naively modeled via $Y_{ij}^{kt} \leq X_{ij}^t$ and $Y_{ji}^{kt} \leq X_{ij}^t$, but it is not hard to see that in any optimal solution, the flow Y^{kt} must induce a tree rooted at k - in fact in the uncapacitated case it can be taken without loss of generality to be a simple path - so this logic can be expressed more strongly via constraints (5). Later we will show that we may assume without loss of generality that within a single period, flow on a link from any source can only go in one direction, and will exploit that to strengthen the formulation.

Constraint (7) implies that once a link is constructed, it remains open throughout the time horizon.

The budget constraint on investment in each time period is reflected in constraint (8). In this inequality, the two first terms model the expense of opening and closing facilities, respectively. The last term models expenditure for network link construction. The total of these investment costs should not exceed the available budget in the time period, introducing a tradeoff between investment in the network and in the new facilities. As discussed earlier, in this model any unspent budget at the end of a time periods is lost, and cannot be carried forward.

Constraints (9) and (11) enforce non-negativity of the flow variables and constraints (10) and (12) ensure the location and link opening variables are binary.

3.4 Linearizing the budget constraint and initial conditions

Two approaches are used to linearize the nonlinear terms in the budget constraint. In the first, we adopt the approach proposed by Lee and Dong [52], and determine some additional cuts to strengthen this formulation. In the second, we propose a new approach, based on a tight formulation of the facility open/close decisions, that is much more efficient than the first one.

3.4.1 Linearizing with Lee and Dong's approach

In this method, the nonlinear budget constraint is linearized in a standard way, with the introduction of new binary variables U_i^t, set to 1 to indicate that facility i is open, but not newly opened, in period t for each $i \in N$, and V_{ij}^t, set to 1 to indicate that link (i,j) is open, but not first built, in period t, for each $(i,j) \in L^t$ with $i < j$, for all $t \in T$. Equivalently,

$$\begin{aligned} U_i^t &= Z_i^t Z_i^{t-1} \\ V_{ij}^t &= X_{ij}^t X_{ij}^{t-1} \end{aligned}$$

for all i, (i,j) and t, provided the cases $t-1=0$ and $(i,j) \notin L^{t-1}$ are properly accounted for. We now show how to do this, and give the constraints needed to define a linear budget constraint. With these definitions, $Z_i^t - U_i^t$ indicates a newly opened facility, $Z_i^{t-1} - U_i^t$ a newly closed facility, and $X_{ij}^t - V_{ij}^t$ a newly built link, so can be used to model the investment costs.

First, we define "period zero" variables

$$\begin{aligned} &Z_i^0 = 1 \qquad \forall i \in N^0 \quad \text{and} \quad Z_i^0 = 0 \qquad \forall i \in N^{\setminus} N^0, \qquad \text{and} \\ &X_{ij}^0 = 1 \qquad \forall (i,j) \in L^0 \text{ with } i < j. \end{aligned}$$

Now to model the relationship between the facility open but not newly and the facility open status variables, linearly, we apply the following constraints

$$\begin{aligned} U_i^t &\le Z_i^t && \forall i \in N, \forall t \in T, \\ U_i^t &\le Z_i^{t-1} && \forall i \in N, \forall t \in T, \qquad \text{and} \\ U_i^t &\ge Z_i^t + Z_i^{t-1} - 1 && \forall i \in N, \forall t \in T. \end{aligned}$$

Similarly to relate the link opened but not newly and the link open status variables, we use

$$\begin{aligned} V_{ij}^t &\le X_{ij}^t && \forall (i,j) \in L^{t-1} \text{ with } i < j, \forall t \in T, \\ V_{ij}^t &= 0 && \forall (i,j) \in L^t \setminus L^{t-1} \text{ with } i < j, \forall t \in T, \\ V_{ij}^t &\le X_{ij}^{t-1} && \forall (i,j) \in L^{t-1} \text{ with } i < j, \forall t \in T, \qquad \text{and} \\ V_{ij}^t &\ge X_{ij}^t + X_{ij}^{t-1} - 1 && \forall (i,j) \in L^{t-1} \text{ with } i < j, \forall t \in T. \end{aligned}$$

Here we make the assumption that a link not available in period $t-1$ (not in L^{t-1}) is implicitly not open, and so V_{ij}^t cannot be 1. Recall $L^t \supseteq L^{t-1}$ for all $t \in T$, so $L^t \cap L^{t-1} = L^{t-1}$.

With these variables, the budget constraint is thus expressed linearly as

$$\sum_{i \in N} g_i^t (Z_i^t - U_i^t) + \sum_{i \in N} p_i^t (Z_i^{t-1} - U_i^t) + \sum_{(i,j) \in L^t, i<j} h_{ij}^t (X_{ij}^t - V_{ij}^t) \le B^t \qquad \forall t \in T.$$

We use the above variables and constraints to solve the budget-constrained DUFLNDP as a MILP.

To improve this formulation, we introduce some additional constraints to the model. These constraints can be viewed as encapsulating obvious preprocessing deductions that can be made by inspection of the budget constraints. In particular, if the coefficient of a term in the constraint exceeds the budget in that period, the term must be zero. This gives the knapsack preprocessing constraints:

$$\begin{array}{ll} Z_i^t - U_i^t = 0 & \text{if } g_i^t > B^t, \text{and} \\ Z_i^{t-1} - U_i^t = 0 & \text{if } p_i^t > B^t, \end{array}$$

for all $i \in N$, and all $t \in T$, and

$$X_{ij}^t - V_{ij}^t = 0 \quad \text{if } h_{ij}^t > B^t,$$

for all $(i,j) \in L^t$ with $i < j$ and all $t \in T$. As can be seen from the computational results given in Table 5, the addition of these constraints greatly improves the performance of the CPLEX solver, suggesting that it is not automatically identifying these constraints in its preprocessing steps.

3.4.2 Linearization based on a tight formulation

The benefits observed by the additional constraints above suggest an alternative approach. If the terms in the budget constraint above are replaced by single variables, then not only do the constraints simplify to become variable bounds, which are handled far more efficiently by solvers than other types of constraints, but they should be automatically identified by the solver's preprocesser. We thus suggest that instead of using the U variables as above, we instead introduce the variables $\tilde{U}$ and $\bar{U}$, defined as follows:

$$\begin{array}{ll} \tilde{U}_i^t = Z_i^t(1 - Z_i^{t-1}) = Z_i^t - U_i^t & \forall i \in N, \forall t \in T, \textit{and} \\ \bar{U}_i^t = Z_i^{t-1}(1 - Z_i^t) = Z_i^{t-1} - U_i^t & \forall i \in N, \forall t \in T. \end{array}$$

It is then sufficient to ask that the following constraint holds:

$$Z_i^{t-1} + \tilde{U}_i^t = Z_i^t + \bar{U}_i^t, \qquad \forall i \in N, \forall t \in T. \tag{13}$$

This gives a tight integer linear programming formulation of the facility open/close decisions in the following sense: for a given $i \in N$, the polytope $\{(Z_i, \tilde{U}_i, \bar{U}_i) \in [0,1]^{3|T|} : Z_i^{t-1} + \tilde{U}_i^t = Z_i^t + \bar{U}_i^t,\ \forall t \in T\}$ (where Z_i^0 is fixed appropriately) has the integrality property, since (13) can be viewed as a network flow constraint. Thus another alternative for a linearized budget constraint is

$$\sum_{i \in N} g_i^t \tilde{U}_i^t + \sum_{i \in N} p_i^t \bar{U}_i^t + \sum_{(i,j) \in L^t, i<j} h_{ij}^t (X_{ij}^t - V_{ij}^t) \leq B^t \qquad \forall t \in T.$$

Of course we could also linearize the link opening decisions in a similar fashion, but as the computational results given in Section 5 show that the additional constraints do not further improve this formulation, we leave it as is. As Table 5 shows, this gives substantial improvements over the first linearization.

3.5 Solution properties and model strengthening

The combination of the cost structure for flow in the network, and the absence of capacity constraints on either facilities or links, leads to properties of optimal solutions that can be exploited to strengthen the model.

We first show that under the following condition on the cost of flow, which we call the *product form cost* condition, there exists (for each period) an ordering of paths in the network for which path cost is non-decreasing for all clients (commodities).

Condition 1. *We say the client link flow costs a have* product form *if there exist α and b such that*

$$a_{ij}^k = b_k \alpha_{ij}$$

for all clients k and links (i,j). We call b the client part *and α the* link part *of the cost, and require that $b_k > 0$ for all clients k and $\alpha \geq 0$.*

For the problem we consider here, we have client link flow costs $r^{\cdot t}$ in product form for all $t \in T$.

For the following, we use the $\gamma(P)$ to denote the sum of γ_{ij} values over links (i,j) in P, where P is a path in the network and γ a link flow cost vector.

Lemma 1. *For any two paths P and Q, and a any client link flow costs in product form, if*

$$a^k(P) \geq a^k(Q)$$

for some client k, then

$$a^{k'}(P) \geq a^{k'}(Q)$$

for all clients k'.

Proof. Let a have (strictly positive) client part b and link part α. Let P and Q be two paths in the network, and k any client such that $a^k(P) \geq a^k(Q)$. By the product form of a, and since the cost of a path is just the sum of its link costs, we have

$$a^k(P) = b_k \alpha(P) \geq b_k \alpha(Q) = a^k(Q).$$

Since $b_k > 0$, it must be $\alpha(P) \geq \alpha(Q)$. Thus for any client k', since $b_{k'} > 0$ also, it must be that

$$b_{k'} \alpha(P) \geq b_{k'} \alpha(Q)$$

and hence

$$a^{k'}(P) = b_{k'} \alpha(P) \geq b_{k'} \alpha(Q) = a^{k'}(Q)$$

as required. ■

We can now show that we may without loss of generality assume that in each time period flow in a link is one directional, i.e. in any (undirected) link, all commodities flow either in one direction, or the other; in no case need two commodities flow in different directions in the same link. In fact, there is a more general result, from which the this follows as a special case: if the service paths for two clients traverse a common node, then we can assume without loss of generality that if they exit that node, they will do so using the same link.

Proposition 1. *Provided $r^{\cdot t}$ satisfies the product form cost condition for all $t \in T$, there exists an optimal solution to the DUFLNDP in which the flow vector $(Y_{ij}^{kt})_{\{k,t,(i,j)\}}$ has the property that for each $t \in T$ and each node $i \in N$, there exists a node $j(i) \in N$ with $(i, j(i)) \in L^t$ such that $Y_{ij}^{kt} = 0$ for all j with $(i,j) \in L^t$ but $j \neq j(i)$.*

Proof. Let $(Y_{ij}^{kt})_{\{k,t,(i,j)\}}$ be an optimal solution which serves each client along the lexicographically smallest minimum flow cost path. We may also safely assume all clients k are served by a unique facility on a unique path P_k, called its *service path* (amongst all paths from the client to a facility, choose one with minimal flow cost that is lexicographically smallest). Now suppose the property doesn't hold, i.e. there exists $t \in T$, $i, j, j' \in N$ with $\{(i,j),(i,j')\} \subseteq L^t$ and $j \neq j'$, and pair $k, k' \in N$, with $k \neq k'$, for which both $Y_{ij}^k > 0$ and $Y_{ij'}^{k'} > 0$. Let P denote the service path for k after node i and P' denote the service path for k' after node i. It must be that

$$r^k(P) \leq r^k(P') \qquad \text{and} \qquad r^{k'}(P') \leq r^{k'}(P),$$

by the optimality of P_k and $P_{k'}$ respectively. Applying Lemma 1 to the latter inequality gives $r^k(P') \leq r^k(P)$ which combines with the former to imply $r^k(P') = r^k(P)$. Similar steps yield $r^{k'}(P') = r^{k'}(P)$. So serving k with P' after i would also be optimal, as would serving k' with P, so either could be swapped without losing optimality of the solution. Suppose without loss of generality that $j < j'$. Then replacing the part of $P_{k'}$ after i with P gives a lexicographically smaller service path with identical cost, and we get a contradiction. The result follows. ■

Corollary 1. *Provided $r^{\cdot t}$ satisfies the product form cost condition for all $t \in T$, there exists an optimal solution to the DUFLNDP in which the flow vector $(Y_{ij}^{kt})_{\{k,t,(i,j)\}}$ has the property that for each $t \in T$ and each $\{i,j\}$ such that $\{(i,j),(j,i)\} \subseteq L^t$, either $Y_{ij}^{kt} = 0$ for all $k \in N$, or $Y_{ji}^{kt} = 0$ for all $k \in N$.*

Proof. Consider $(Y_{ij}^{kt})_{\{k,t,(i,j)\}}$ an optimal solution in which there exists $t \in T$, $\{i,j\}$ with $\{(i,j),(j,i)\} \subseteq L^t$, and pair $k, k' \in N$, with $k \neq k'$, for which both $Y_{ij}^{k} > 0$ and $Y_{ji}^{k'} > 0$. We may obviously assume without loss of generality that all service paths in this solution are acyclic, and that no service path exits a node with a facility. Then the service paths for k and k' have node j in common. Since the path for k is acyclic, it cannot use link (j,i), so either there is a facility at j, or there must exist $h \neq i$ such that $(j,h) \in L^t$ lies on the service path for k. Thus we have $Y_{jh}^{kt} > 0$ and $Y_{ji}^{k't} > 0$ where $i \neq h$, contradicting Proposition 1. The result follows. ■

This corollary could be used to strengthen the formulation directly via the addition of the constraints

$$Y_{ij}^{kt} + Y_{ji}^{k't} \leq X_{ij}^t, \qquad \forall t \in T, \forall k, k' \in N, k \neq k', \forall (i,j) \in L^t, i < j.$$

However this is a very large class of constraints and will very substantially increase the size of the LP. Instead, we propose introducing a new binary variable

$$\bar{Y}_{ij}^t = \begin{cases} 1 & \text{if } \sum_{k \in N} Y_{ij}^{kt} > 0, \text{ and} \\ 0 & \text{otherwise,} \end{cases}$$

for all $(i,j) \in L^t$ and all $t \in T$. This can be modelled with the constraints

$$Y_{ij}^{kt} \leq \bar{Y}_{ij}^t \qquad \forall (i,j) \in L^t, \forall k \in N, \forall t \in T.$$

The $\bar{Y}$ variables can be used to enforce the property observed in Proposition 1, by applying the constraint

$$Z_i^t + \sum_{j:(i,j) \in L^t} \bar{Y}_{ij}^t = 1 \qquad \forall i \in N, \forall t \in T.$$

This also captures the observation that without loss of generality, we may assume no service path will exit a node with a facility. The property observed in Corollary 1 can be enforced readily via

$$\bar{Y}_{ij}^t + \bar{Y}_{ji}^t \leq X_{ij}^t \qquad \forall (i,j) \in L^t \text{ with } i < j, \forall t \in T,$$

which makes (5) redundant.

It is not hard to see that in fact – as for the Y and W variables – $\bar{Y} \geq 0$ will suffice, and an explicit integrality requirement is not needed. There will always exist an optimal solution in which all of Y, W and $\bar{Y}$ are binary. However it is possible that asking for $\bar{Y}$ to be binary will give the MILP solver access to useful branching variables, and may be helpful for generating cutting planes. In our computational study, we thus experiment with alternative combinations of non-negativity versus binary constraints for these variables. We discuss this further in Section 5.

4 Heuristic methods

Even with the tighter formulations proposed above, solving DUFLNDP problems to optimality for realistically sized instances is still challenging. Thus we consider heuristic approaches. The approaches we propose exploit two observations we have made about the computational behaviour of MIP models for these problems: (1) single-period problems solve quickly, and (2) if the Z variables (indicating facility open status) are fixed, the rest of the problem solves quickly. We thus consider

- sequential, period-by-period (PBP) heuristics, and
- a hybrid variable neighborhood search algorithm (HVNS) with the solution space determined by the facility status decisions.

In the former case, we present three variants: a standard one-period-at-a-time approach, a variant in which a relaxation is solved in each period, and a two-period-at-a-time approach. In the latter, a variable neighborhood search method is used to explore the space of facility status decisions, with each solution evaluated by solving the MIP formulation with the Z variables fixed accordingly. We discuss these in more detail below.

4.1 Sequential heuristics

The main idea behind these heuristics is to solve a sequence of MIP models for single-period problems, fixing the decisions in the previous period before moving onto the next. Note that it is only facility and link open status from the previous period which impacts the problem in a given period, since changing the status of a facility or link incurs a cost in the budget constraint. Otherwise the problems in each period are independent, and facility and link status prior to the previous period has no impact on the current period's problem.

In the basic variant, Variant I, we solve the MIP induced by the variables in period t only, taking the decisions in period $t-1$ to be fixed, for each $t = 1, \ldots, T$. (In period $t = 1$, the decisions in the previous period are determined by the sets N^0 and L^0.) At the completion of this process, we take all facility status variables (the Z variables) to be fixed, and solve a MIP over all the other variables (in all time periods).

In Variant II, we seek to accelerate the MIP solve at each period by relaxing the budget constraint, in a way that we hope will minimize the impact on the quality of the resulting solution. Since we know we can "repair" the final solution by doing a final solve over the whole problem with the Z variables fixed, it is only to these variables that the budget constraint must apply in order to guarantee a feasible solution at the end. Thus we remove the link status variables (the X variables) from the budget constraint, and do not impose the requirement that a link once opened must stay open. This accelerates the MIP solve. It also allows more flexibility in the budget to seek the best facility status decisions in the period-by-period process, leaving the best link choices to be made *a posteriori* in the final solve.

Variant III investigates the incremental impact of making less myopic decisions, by solving the MIP induced by the variables in periods t and $t+1$ together, taking the facility and link status decisions in period $t-1$ to be fixed, for each $t = 2, \ldots, T$. (We still solve for the first period individually, since in our test instances all facilities and links are initially closed, and the first period problem plays the role of an initialization step, installing sufficient infrastructure to feasibly service the demand; this tends to require much longer MIP solution time than problems in later periods.) Again a final solve over all time periods with the Z variables fixed is used to obtain the final solution.

4.2 Hybrid Variable Neighborhood Search

Variable neighborhood search (VNS) is a relatively new meta-heuristic aimed at solving combinatorial and global optimization problems. This algorithm systematically exploits the idea of neighborhood change, both in descent to local minima and in escape from the valleys which contain them. The VNS algorithm first appears in the paper by Mladenovic and Hansen [59] and since then rapidly developed both in methodology and application. Hansen et al. [60] provide a thorough survey and an extensive bibliography about VNS, discussing both these aspects.

There are many successful applications of VNS, or of hybrids of VNS with other algorithms, in numerous areas. Hansen et al. [60] categorized a list of different application of VNS. Industrial applications, design problems in communication, data mining and facility location are only some examples of these applications. Facility location problems in discrete and continuous space have attracted much attention from VNS researchers and practitioners. Among discrete models, the p-median has been the most studied and has played a central role in the development of the VNS metaheuristic. Furthermore, the multi-source Weber problem was the first model in continuous space solved by VNS [61]. A hybrid VNS was proposed by Jabalameli and Ghaderi [62] to solve the uncapacitated continuous location-allocation problem.

The hybrid algorithm we propose here was developed from the basic VNS principle. The algorithm is given in Algorithm 1. A *solution* consists of the facility open status of each facility in each period, i.e. is an instantiation of the Z variables. A solution S is *evaluated* by solving the MIP model with the Z variables fixed as specified in the solution, to obtain the objective function value, denoted by $OF(S)$, which is taken to be $+\infty$ if the MIP is infeasible. We are able to accelerate solution of the MIP by using the fact that we are only interested in solutions with objective better than that of the current solution: we give Cplex's "upper cut off" function the value of the current solution, $OF(S)$, to eliminate parts of the search tree with lower bound greater than the objective of the best solution found so far.

We define K_{max} *neighborhoods* (described later), and search each neighborhood in turn, in order of increasing size of the neighborhood, generating up to N^K_{max} neighbors in neighborhood K. The algorithm cycles through the neighborhoods, returning to the first ($K = 1$) if no improving solution is found in all K_{max} neighborhoods. The algorithm is quite aggressive in returning to the first neighborhood, doing so at any time an improving solution is found. This strategy was adopted after initial computational tests showed the majority of improving neighbors were found in the first neighborhood.

Algorithm 1: The Hybrid VNS Algorithm

Input: Instance parameters, an initial solution S, K_{max} the number of neighborhoods, and N^K_{max} the maximum number of solutions in the Kth neighborhood generated in each round, for each $K = 1, \ldots, K_{max}$

Output: An improved solution S

1.1 Evaluate S by solving a MIP model with the corresponding Z variables fixed to obtain $OF(S)$;
1.2 **while** stopping criteria not met **do**
1.3   Initialize neighborhood counter $K := 1$ and solution counter for the neighborhood $\eta_1 := 1$;
1.4   **while** $K \leq K_{max}$ **do**
1.5 $\hat{S} := GenerateNeighbor(S, K)$;
1.6   Evaluate $\hat{S}$ by solving a MIP model with the corresponding Z variables fixed to obtain $OF(\hat{S})$;
1.7   **if** $OF(\hat{S}) < OF(S)$ **then** replace $S := \hat{S}$, and go to 1.3;
1.8   **else if** stopping criteria met **then** go to 1.13;
1.9   **else if** $\eta_K \leq N^K_{max}$ **then** increment solution counter $\eta_K := \eta_K + 1$ and go to 1.5;
1.10   **else** increment neighborhood counter $K := K + 1$, and initialize solution counter for the neighborhood $\eta_K := 1$;
1.11   **end**
1.12 **end**
1.13 **return** S;

In what follows, we describe the neighborhood structure, and the neighbor generation procedure.

4.2.1 Solution representation and neighborhood operations

Solution representation is a key issue for a successful VNS implementation. The approach we propose exploits the facility open status decisions as the key drivers of the overall solution quality, and the fact that MIPs with these decisions fixed can be solved efficiently. Thus the variables associated with opening of the facilities constitute the solution. Other variables are obtained by solving the MIP with the facility open status variables fixed using the IBM ILOG CPLEX solver. The representation of a solution is illustrated in Figure 2, by a matrix with N rows, one for each facility, and T columns, one for each time period. The cell in row i, column t, corresponds to the value of Z^t_i. Each cell will have a 1 indicating that the facility is open during the corresponding time period, and 0 if it is closed during that time period.

The performance of a VNS algorithm also depends critically on the neighborhood structure, and the operations used to generate neighbors. Using an efficient procedure for neighborhood production is very important for a successful algorithm. To this end, various strategies were examined to generate neighborhoods, and a combination of the best of them was used. How to generate the neighborhood for this problem is described as follows with the aid of some examples. Figure 2 shows a solution for a problem with 6 nodes and 5 time periods. In this figure, the values in the top left, and top right, corner in each cell are the opening and closing cost of the facility, respectively (recall that these costs are period-dependent). These costs apply in the period in which the facility is first opened, or first closed. In the two last rows, the maximum budget for investing in facilities (BF) and the investment cost of this solution in terms of facility open/close costs (bf) are shown. For example, in period 4, facilities 4 and 6 are closed, and the status of all other facilities is unchanged. Each of facilities 4 and 6 costs 2 units to close in this period, so the total cost given by bf is 4. In period 3, by contrast, no facilities are closed (after being open), but facilities 2 and 6 are opened (after being closed). The opening costs for these facilities are 8 and 5 respectively, giving total cost in period 3 of $bf = 13$.

We propose three mechanisms for perturbing a solution, and build up our neighborhoods using

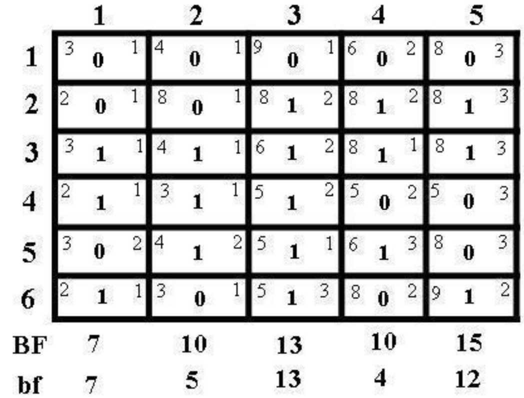

Figure 2: A feasible solution for a small problem with 6 nodes and 5 time periods

operations based on these mechanisms. The three mechanisms are:

1. Opening a closed facility;
2. Closing an open facility; and
3. Exchanging a sequence of facility status decisions between two nodes.

Our aim with these operations is to perturb the budget impost of the modified solution in a minimal way. So in the first two mechanisms, once a new opening or closing decision is made, that status is maintained for future periods until the next change in status already in the original solution. This corresponds to moving an opening or closing decision earlier in time. For example, if the facility open status vector for a node is given by $(0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0)$, the first mechanism could change this to any of $(0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0)$, (move the second opening decision forward by two periods), $(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0)$, (move it forward by three periods), $(1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0)$, (move the first opening decision forward by two periods), or $(0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1)$, (open from the last period to the end), to give just four options out of the seven possible. For a completely closed facility, e.g. with status vector $(0, 0, 0, 0, 0)$, this mechanism would open it at some point in time, and maintain it open until the end of the time horizon, e.g. would give $(0, 0, 1, 1, 1)$, to give just one of the five options. The closing-earlier mechanism is illustrated by taking the complement of the vectors used in the opening-earlier mechanism examples.

In the third mechanism, the sequence to exchange is defined to start in a period t_1 in which the two nodes have different status, and for which they have the same status in the previous period. The sequence ends in period t_2, the latest period in which both nodes have maintained the same status as they do in period t_1. The status of facilities at the two nodes is thus different, but constant, over the periods t_1 to t_2, inclusive, and the status vectors over these time periods can be exchanged without changing the number of opening and closing decisions in any time period. To illustrate, consider the two vectors $(0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1)$ and $(0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0)$, corresponding to facility open status decisions for nodes 1 and 2 respectively, say. One exchange between these two nodes could be carried out by taking $t_1 = 3$, in which case we must take $t_2 = 4$. Exchanging the status of the two vectors between periods 3 and 4 inclusive yields the vector $(0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1)$ for node 1, and the vector $(0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0)$ for node 2. Note that in the original solution the facility at node 1 was first opened in period 5, and that at node 2 first opened in period 3; these are now swapped, with the facility at node 1 now first opened in period 3 and that at node 2 in period 5. Another possible exchange would be to choose $t_1 = 12$, in which case it must be $t_2 = 15$, and the new status vectors are $(0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0)$ and $(0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1)$, with node 1's facility now closed in period 12 rather than node 2's. These illustrate two of the four options for exchange between the two original vectors.

Of course, applying these operations may change the budget consumed in the affected periods, leading to an infeasible solution. We do not permit such cases to be considered, and restrict our choice of operation to only those that would result in a solution satisfying the budget constraints. We illustrate the mechanisms in the context of the budget constraints using the solution given in Figure 2. Consider node 2. Its facility is opened at time period 3 and remains open until the end of the time horizon. The first mechanism cannot be applied at all, because opening it in either periods 1 or 2 instead would violate the budget constraint. However the second mechanism can be applied to close the facility in any of periods 3, 4 and 5 without violating the budget. Note that in period 3, this is only possible because the opening cost will no longer be incurred, as this is the period in which the facility is newly opened. Thus there are three valid operations that could be applied to node 2 using either of the first two mechanisms. Now consider the third mechanism, and possible exchanges between node 2 and some other node. The only possible exchange with node 1 has $t_1 = 3$ and $t_2 = 5$, but this is not valid as the cost of opening a facility at node 1 costs one more than at node 2 in period 3, and the budget constraint is already tight in this period. Nodes 2 and 3 can be exchanged between $t_1 = 1$ and $t_2 = 2$, (in this example N^0 is the empty set), which would decrease the budget by 1 ($= 3 - 2$) in period 1, and by 2 ($= 8 - 6$) in period 3, and thus is budget-feasible. For nodes 2 and 4, two exchange operations are possible: from periods 1 to 2, and from periods 4 to 5. Both are feasible. Nodes 2 and 5 can have their status exchanged in period 2, and also in period 5. Again both operations would be feasible. Nodes 2 and 6 can have their status exchanged in period 1, and also in period 4. Again both operations would be feasible. There are thus a total of 7 possible valid exchange operations involving node 2, giving in total 10 valid operations (including the three using the second mechanism). The results of the two possible exchange operations between nodes 2 and 6 are shown in Figure 3.

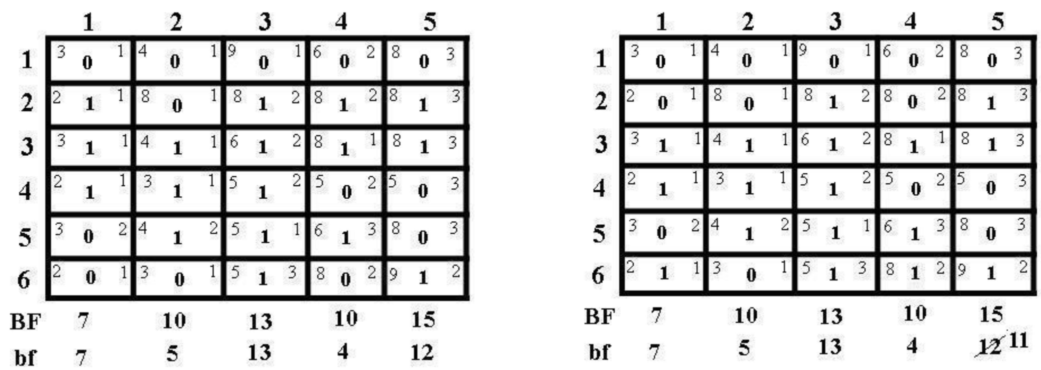

Figure 3: The two feasible solutions generated by exchanging parts of the facility status vectors for nodes 2 and 6.

4.2.2 Neighbor generation procedure

The neighborhood operations discussed above are used to generate a neighbor of a current solution S, in the following way. If the neighbor is to be generated from neighborhood K (via the $GenerateNeighbor(S, K)$ procedure), then a neighborhood operation is selected at random, applied to S, and the process repeated K times. To generate a neighbor at random, a node is selected at random (with all nodes having equal probability of being selected). Then all possible valid neighbor operations involving that node are identified, and one of these selected at random, with each having equal probability. For example, in the case discussed at the end of the previous section, if node 2 had been selected at random, then one of the 10 valid operations identified would have been selected at random, with each having probability one-tenth of being selected.

5 Computational experience

In this section, we discuss the computational study we undertook, and the performance of the approaches proposed.

5.1 Data generation process

Because the DUFLNDP we study here has not previously been considered in the literature, and there are no published benchmark instances for the problem, we generate test problems using a random procedure, based on those proposed in [2] and [3]. The generation process takes as input the number of clients (nodes), $|N|$, the number of time periods, $|T|$, and the number of potential links, $|L|$, for the instance to be generated. Note that in these instances we, for simplicity, take the potential links in all periods to be identical, i.e. define $L \triangleq L^0 = L^1 = \ldots = L^{|T|}$. The process then generates data with the following features.

- The locations of the clients (network nodes) are generated randomly and uniformly distributed over a 100×100 area of Euclidean space.
- The candidate links are randomly selected and added to the network, with a bias towards shorter links to emulate transportation networks.
- The operating cost of a facility in each period is taken to be a fixed percentage of its opening cost in that period.
- The construction cost of a link is proportional to its Euclidean length.
- The link travel cost is proportional to its Euclidean length.
- The operating cost of a link is taken to be a fixed percentage of the construction cost of the link.
- The service network is initially empty (initially no facilities are open and no links are built).

Table 3 gives details of how demand, costs and other input data are generated. In this table, $U[a, b]$ denotes the random generation of numbers in the interval $[a, b]$ according to a discrete uniform distribution. If this is followed by a percentage sign, the value drawn from the distribution is to be treated as a percentage (divided by 100). For example, in the first row of the table, we see that for each client, their demand in the first period is an integer drawn uniformly from between 10 and 100. Their demand in each subsequent period represents an increase of 0%, 1%, 2%, 3%, 4% or 5% over that in the previous period, with equal probability (i.e. the increase is drawn from $U[0, 5]\%$). All of client demand, facility opening cost, travel cost per unit flow and construction cost undergo similar evolution over time. Facility closing cost follows a similar pattern, but is allowed to decrease (have a negative percentage increase), reflecting, for example, increased salvage values. Facility closing costs are considerably lower than opening costs and may even take negative values. This reflects the likelihood that revenue could be earned in the process of decommissioning a facility, for example, by selling property, whereas opening a facility is anticipated to be a purely cost-generating enterprize. Note that the Euclidean distance between nodes i and j (the length of link (i, j)) is denoted m_{ij} in Table 3.

Table 3: Parameters used in the random generation of instances

Symbol	Description	Value generated
d_k^t	Demand of client k during time period t	$d_k^t = d_k^{t-1}(1 + U[0, 5]\%)$ $t \geq 2$, $d_k^1 = U[10, 100]$
g_i^t	Opening cost of facility at i in time period t	$g_i^t = g_i^{t-1}(1 + U[2, 10]\%)$ $t \geq 2$, $g_i^1 = U[1200, 1500]$
p_i^t	Closing cost of facility at i in time period t	$p_i^t = p_i^{t-1}(1 + U[-2, 10]\%)$ $t \geq 2$, $p_i^1 = U[0, 150]$
f_i^t	Operating cost of facility at i in time period t	$\delta g_i^t, \delta = 0.06$
ρ_{ij}^t	Travel cost per unit flow on link (i, j) at time period t	$\rho_{ij}^t = \rho_{ij}^{t-1}(1 + U[0, 5]\%)$ $t \geq 2$, $\rho_{ij}^1 = \epsilon m_{ij}, \epsilon = 0.1$
r_{ij}^{kt}	Travel cost of client k on link (i, j) at time period t	$r_{ij}^{kt} = \rho_{ij}^t d_k^t$
h_{ij}^t	Construction cost of link (i, j) at time period t	$h_{ij}^t = h_{ij}^{t-1}.(1 + U[0, 5]\%)$ $t \geq 2$, $h_{ij}^1 = \sigma m_{ij}, \sigma = 10$
c_{ij}^t	Operating cost of link (i, j) at time period t	$c_{ij}^t = \tau h_{ij}^t, \tau = 0.04$

The following process is used to determine the budget parameter in each time period. Melkote [2] in his thesis showed that for the static case if α facilities are opened in a feasible solution, then $(|N| - \alpha)$ links should be constructed. We use this property to generate the budget parameter for the first time period. First, we take α to be a fixed percentage of the number of nodes; we found $\alpha = 0.05|N|$ worked well. We then calculate the average cost of opening α facilities and $N - \alpha$ links in the first period:

$$\gamma_1 = \alpha \bar{g}_1 + (N - \alpha)\bar{h}_1$$

where

$\bar{g}_t$ = the average facility opening cost in period t, over all nodes, and

$\bar{h}_t$ = the average link construction cost in period t, over all links.

The budget for the first time period, B^1, is selected at random from the interval $[0.8\gamma_1, 1.2\gamma_1]$. For the other time periods, the budget selection is parameterized by the first period budget, weighted by the average increase in facility opening costs for that period over the average costs in the first period, i.e. by

$$\gamma_t = \frac{\bar{g}_t}{\bar{g}_1} B^1.$$

The budget at time period t, B^t, is then selected at random from the interval $[0, 0.5\gamma_t]$.

5.2 Test instances

We use a suite of thirty test instances, generated using the process described above for each combination of the following input parameter settings.

- The number of nodes, $|N|$, is set to each value in $\{20, 40, 60, 80, 100\}$.
- The number of time periods, $|T|$, is set to each value in $\{5, 10, 20\}$.
- The number of potential links, $|L|$, is set to two possible levels for each value of $|N|$ (see Table 4).

Key parameters of the thirty instances are summarized in Table 4. For each instance, we also compute a time limit setting for solution of the MIP with IBM ILOG CPLEX. The dimensions of the test problems was chosen to display a range of behaviors when the MIP is solved with CPLEX: its performance on these instances varies from solving to optimality in short times for the smaller problems, to being unable to find any feasible solutions even in very long run times for the largest problems. In the results we report, we run CPLEX with default options, but with CPU time limited to $50|N||T|$ seconds. For example, the CPU time for a test problem with 80 nodes and 20 time periods was $80,000$ seconds, which is equivalent to 22.22 hours.

Table 4: the dimensions of instances with CPLEX CPU time limit

Inst.	N	L	T	CPU(Sec.)	Inst.	N	L	T	CPU(Sec.)
1	20	46	5	5000	16	60	205	5	15000
2	20	46	10	10000	17	60	205	10	30000
3	20	46	20	20000	18	60	205	20	60000
4	20	61	5	5000	19	80	171	5	20000
5	20	61	10	10000	20	80	171	10	40000
6	20	61	20	20000	21	80	171	20	80000
7	40	137	5	10000	22	80	280	5	20000
8	40	137	10	20000	23	80	280	10	40000
9	40	137	20	40000	24	80	280	20	80000
10	40	162	5	10000	25	100	283	5	25000
11	40	162	10	20000	26	100	283	10	50000
12	40	162	20	40000	27	100	283	20	100000
13	60	180	5	15000	28	100	453	5	25000
14	60	180	10	30000	29	100	453	10	50000
15	60	180	20	60000	30	100	453	20	100000

5.3 Computational results

The proposed algorithms were all coded in Python 2.6, and all programs were run on a dual quad core 2.66GHz Intel Xeon X5550 processor with 32GB of RAM. All MIPs were solved with the IBM ILOG

CPLEX 12.1 solver. All optimality gaps reported, denoted by *Gap* in the results tables, are computed by

$$Gap = 100(Obj - LB)/LB$$

where *Obj* is the value of the feasible solution found, and *LB* is the best lower bound reported by CPLEX within its given time limit.

5.3.1 Comparison of budget constraint linearizations

The computational results comparing the alternatives for linearization of the budget constraint are given in Table 5. This shows the results of solving the alternative models with CPLEX, on the 12 smallest test instances. For each model, lower bound (LB) and best integer solution (Obj.) found, are shown. If CPLEX is not able to prove optimality within the time limit, we show the gap between lower bound and upper bound; in this case the CPU time equals the time limit given in Table 4. Otherwise, we report the CPU time required by CPLEX solve to optimality, given in brackets. For example, for the first model and first instance, CPLEX is not able to prove optimality within the 5000 second time limit, but finds an integer solution (in fact finds an optimal solution), and is able to prove its gap is at most 9.20%. On the same instance, CPLEX is able to prove optimality with other three models within 157, 26 and 37 seconds, respectively.

Despite the small size of the instances solved, the first approach cannot prove optimality in any case, and in several cases cannot find any feasible solution within the time limit. This situation is improved significantly with the addition of the knapsack preprocessing constraints, but still optimality is not proved in all but one case, and in one case no feasible solution is found. The approach based on the tight formulation is clearly much better. It proves optimality in most cases and finds a good quality feasible solution in all. With the reformulation, the additional constraints are no longer helpful. In what follows, we proceed with the tight formulation, without additional cuts (the third approach reported in the table).

Table 5: Computational comparison of alternatives for linearization of the budget constraints

	Lee and Dong's [52] approach			Lee and Dong [52] + cuts			Tight formulation			Tight formulation + cuts		
Inst.	LB	Obj.	Gap(Time)	LB	Obj.	Gap(Time)	LB	Obj.	Gap(Time)	LB	Obj.	Gap(Time)
1	19235.50	21004.36	9.20	21004.36	21004.36	(157)	21004.36	21004.36	(26)	21004.36	21004.36	(37)
2	32187.8	44244.33	37.46	40917.01	41092.00	0.43	41091.49	41091.49	(38)	41091.49	41091.49	(90)
3	64161.27	81015.07	26.27	75063.83	80722.56	7.54	80397.13	80397.13	(360)	80397.13	80397.13	(373)
4	32069.02	39052.39	21.78	33773.58	37073.19	9.77	37073.19	37073.19	(187)	37073.19	37073.19	(320)
5	62459.61	76078.65	21.80	70753.5	74808.28	5.73	73252.05	73252.05	837	73252.05	73252.05	(938)
6	140222.97	NA	–	144790.52	159160.18	9.92	158829.81	158829.81	(2449)	158829.81	158829.81	(2628)
7	21846.76	22811.05	4.41	22350.17	22618.09	1.20	22605.48	22605.48	(706)	22605.48	22605.48	(475)
8	42441.47	NA	–	44505.92	45595.09	2.45	45582.70	45582.70	(9782)	45539.28	45586.89	0.10
9	98345.83	NA	–	102312.25	105755.24	3.37	105024.61	105024.61	(10142)	105024.61	105024.61	(18969)
10	17991.78	20556.68	14.26	18597.93	19815.97	6.55	19365.21	19877.83	2.65	19256.29	19994.02	3.83
11	34296.50	NA	–	35408.5569	NA	–	37697.4	39073.57	3.65	37647.19	39205.35	4.14
12	85267.81	NA	–	87059.76	96042.89	10.32	91116	92603.74	1.63	91073.04	93247.36	2.39
Av.	54210.52	–	(17500)	58044.78	–	(17096)	61086.61	61367.99	0.66(7877)	61066.16	61442.64	0.87(9485)

5.3.2 Model strengthening

The behavior of the model in two cases with and without defining $\bar{Y}$ is studied via computational experiment. To do this, the formulations with and without the $\bar{Y}$ variables (and associated constraints) are solved for the 12 smallest test instances. We also test to see if requiring the Y variables to be binary is helpful or not. The results are shown in Table 6. The first two sets of results give a mixed impression: it is hard to tell if requiring the Y variables to be binary is a better or worse strategy. More problems are solved to optimality with the requirement, but on the other hand some problems no longer return a feasible solution within the time limit. However the third set of results shows unequivocally that it is better to use $\bar{Y}$ variables and related constraints: all problems return a feasible solution and for most, optimality is proved.

Table 6: Computational results of the model with and without $\bar{Y}$ and the associated constraints

	$Y \geq 0$ and without $\bar{Y}$			Y binary and without $\bar{Y}$			Y binary and with $\bar{Y} \geq 0$		
Inst.	LB	Obj.	Gap(Time)	LB	Obj.	Gap(Time)	LB	Obj.	Gap(Time)
1	21004.36	21004.36	(239)	21004.36	21004.36	(185)	21004.36	21004.36	(26)
2	38495.74	41100.70	6.77	41091.49	41091.49	(480)	41091.49	41091.49	(38)
3	74570.28	80610.48	8.10	80397.13	80397.13	(1800)	80397.13	80397.13	(360)
4	26888.65	37171.37	38.24	27313.82	39674.04	45.25	37073.19	37073.19	(187)
5	52621.60	77668.10	47.60	53734.45	88124.49	64.00	73252.05	73252.05	(837)
6	118716.45	165105.50	39.08	124644.13	170903.97	37.11	158829.81	158829.81	(2449)
7	21401.20	26475.38	23.71	21756.24	NA	–	22605.48	22605.48	(706)
8	43113.66	65838.86	52.71	45582.70	45582.70	(9772)	45582.70	45582.70	(9782)
9	101794.48	116040.44	13.99	102378.54	NA	–	105024.61	105024.61	(10142)
10	18090.75	19963.28	10.35	18386.92	19812.28	7.75	19365.21	19877.83	2.65
11	36080.4	39655.93	9.91	36463.79	40320.12	10.58	37697.4	39073.57	3.65
12	88841.72	95189.16	7.14	89477.40	94465.02	5.57	91116.00	92603.74	1.63
AV.	53468.27	65485.29	21.47(17103)	55185.91	–	(13936)	61086.61	61367.99	0.66(7877)

We note this third set of results is based on the requirement that the Y variables are binary, but the $\bar{Y}$ variables simply non-negative. We tested all combinations of Y, $\bar{Y}$ binary versus non-negative in Table 7, on a larger set of test instances (the 22 smallest). From this, we see that when used in conjunction with the $\bar{Y}$ variables, it is better to require Y binary. It is not entirely obvious whether requiring $\bar{Y}$ to be binary or not is helpful. Without $\bar{Y}$ binary, one more instance is proved optimal, and the gap is smaller in more cases otherwise, but there is one large gap for Instance 23. With $\bar{Y}$ binary, this large gap is avoided, but across the board optimality gaps are worse. We thus decided that the alternative without requiring $\bar{Y}$ to be binary will be used as the final version of the model for comparisons with other methods.

Table 7: Computational comparison of alternative integrality requirements on the Y and $\bar{Y}$ variables.

	Y binary and with $\bar{Y} \geq 0$			Y and $\bar{Y}$ binary			$Y \geq 0$ and $\bar{Y} \geq 0$			$Y \geq 0$ and $\bar{Y}$ binary		
Inst.	LB	Obj.	Gap(Time)	LB	Obj.	Gap(Time)	LB	Obj.	Gap(Time)	LB	Obj.	Gap(Time)
1	21004.36	21004.36	(26)	21004.36	21004.36	(29)	21004.36	21004.36	(29)	21004.36	21004.36	(29)
2	41091.49	41091.49	(38)	41091.49	41091.49	(78)	41091.49	41091.49	(79)	41091.49	41091.49	(63)
3	80397.13	80397.13	(360)	80397.13	80397.13	(372)	80397.13	80397.13	(597)	80397.13	80397.13	(306)
4	37073.19	37073.19	(187)	37073.19	37073.19	(254)	37073.19	37073.19	(331)	37073.19	37073.19	(63)
5	73252.05	73252.05	(837)	73252.05	73252.05	(1053)	73252.05	73252.05	(1213)	73252.05	73252.05	(926)
6	158829.81	158829.81	(2449)	158829.81	158829.81	(3032)	158829.81	158829.81	(3082)	158829.81	158829.81	(3199)
7	22605.48	22605.48	(706)	22605.48	22605.48	(1917)	22605.48	22605.48	(761)	22605.48	22605.48	(951)
8	45582.70	45582.70	(9782)	45518.79	46356.01	1.84	45522.98	45582.70	0.13	45582.70	45582.70	(17169)
9	105024.61	105024.61	(10142)	105024.61	105024.61	(34163)	105024.61	105024.61	(8205)	105024.61	105024.61	(14334)
10	19365.21	19877.83	2.65	19216.83	20114.94	4.67	19149	20103.6	4.99	19411.11	19782.8	1.91
11	37697.4	39073.57	3.65	37740.82	39247.46	3.99	37766.56	38917.25	3.05	37811.51	39115.08	3.45
12	91116	92603.74	1.63	91392.94	92451.35	1.16	91050.03	93291.86	2.46	91418.92	92533.18	1.22
13	29520.27	29520.27	(3997)	29520.27	29520.27	(6773)	29520.27	29520.27	(4250)	29520.27	29520.27	(5814)
14	61479.91	61479.91	(8945)	61479.91	61479.91	(8915)	61479.91	61479.91	(22545)	61479.91	61479.91	(28009)
15	149232.15	149232.15	(29098)	149232.15	149232.15	(25913)	149232.15	149232.15	(29593)	149232.15	149232.15	(31870)
16	26173.73	27982.48	6.91	27178.6	27856.18	2.49	26238.44	28089.59	7.06	26210.21	27913.43	6.50
17	53276.29	60810.45	14.14	53415.7	65683.2	22.97	53271.35	NA	–	53683.91	57990.26	80.2
18	130445.51	145804.9	11.77	130417.35	161281.75	23.67	130458.27	142215.91	9.01	130686.54	157287.02	20.35
19	36283.66	36878.57	1.64	36215.01	36916.74	1.94	36275.74	37083.01	2.23	36320.59	37030.69	1.96
20	73125.38	NA	–	73204.47	75614.85	3.29	73101.27	NA	–	73135.82	NA	–
22	32656.77	32765.88	0.33	32589.32	NA	–	32639.9	32824.59	0.57	32568.65	NA	–
23	68460.63	99865.61	45.87	68454.73	72411.08	5.78	68408.23	NA	–	68418.60	NA	–
AV.	63349.71	–	(16434)	63402.50	–	(18068)	63336.01	–	(17531)	63398.13	–	(18078)

5.3.3 Heuristic methods

The performance of all heuristic methods, and the CPLEX solver using the best MIP model variant, on all thirty test instances, is reported in Table 8. The first column indicates the instance identifier, corresponding to the parameters given in Table 4. The next four columns, under the "CPLEX" heading, present the lower bound (LB), the value of the best solution found (Obj.), its gap, and the CPU time requirement in seconds, respectively. Columns 6 through 14 show the results for the three period-by-period (PBP) heuristic variants. For each, we report the objective value of the solution found (Obj.), the CPU time requirement in seconds (Time), and the gap between the solution found by the heuristic and the lower bound found by CPLEX. Columns 15 through 17 report similar statistics for the Hybrid Variable Neighborhood Search (HVNS) method, with the exception that the column labeled "Bcpu" gives the CPU time when the best solution is obtained by the algorithm. The last column indicates the number of times an improved solution was found in the course of the algorithm.

From Table 8, we see that CPLEX solved the MIP model well, obtaining either optimal solutions or solutions within a few percent of optimality, for all small instances (numbers 1 to 15), and for instances 19, 22 and 25. These instances all have the largest number of nodes (80, 80 and 100 respectively), but smallest number of time periods or potential links. In particular, instances 19 and 22 are the two 80-node instances having the smallest number of time periods, and Instance 25 is the 100-node instance with the smallest number of both potential links and time periods. On all 12 other, *harder*, instances, CPLEX struggled to solve the MIP model, able to find only poor quality feasible solutions for four of them, and not able to find a feasible solution at all for the remaining 8 instances.

The benefits of the PBP heuristics can clearly be seen on the 12 harder instances. All heuristics found a feasible solution, within relatively modest computing times. The solution quality was generally good, mostly within 4% of optimality (and perhaps less, given the lower bound found by CPLEX may not be optimal for these instances). Even on the easier instances, in many cases the PBP heuristics offer a reasonable trade-off between computing time and solution quality, finding solutions within a few percent of optimality taking, in some cases, no more than a fraction of the computing time.

It is difficult to "pick a winner" between the three PBP variants. It would seem that on average, Variant II is able to give slightly better quality solutions, in somewhat less computing time, than Variant I. What is clear is that the extra computational effort required to solve 2-period MIPs in Variant III is not outweighed by the benefits of a less myopic approach. In terms of solution quality, in 7 instances the two methods are tied, in 12 Variant III gives a better solution, and in 11 instances Variant II's solution is better. Tellingly, Variant II is better or no worse than Variant III on the 5 largest instances, and the average gap (and time) is slightly less. Thus we would suggest Variant II is the preferred approach.

We note that we also tried other variants, for example, fix-and-relax approaches, as well as solving 3-period-at-a-time, and 4-period-at-a-time MIPs. However we found in all cases these did not offer any advantages. In particular, solving for more than 2 periods at a time led to prohibitively slow MIP solve times. To illustrate, for Instance 24, Variant III finds a solution with 5.21% gap in 17,027 seconds, where the 3-period-at-a-time variant could only generate a 5.28% gap solution in well over 20,000 seconds, and the 4-period-at-a-time variant unable to finish within 80,000 seconds.

Before discussing the HVNS results, we make some notes regarding our implementation of the method. The initial solution used for VNS algorithms is usually generated randomly. We found in this case that randomly generated initial solutions were of very low quality, which negatively affected the quality of the final solution returned by the algorithm. Thus we used the solution of the PBP heuristics (Variant II, discussed further below), to initialize HVNS; we found this scheme gave significantly better performance than the random case. As mentioned earlier, we use the value of the current best feasible solution as a cut-off value to accelerate the MIP for evaluating a neighbor. The stopping criteria used was either a solution with value equal to the lower bound reported by Cplex has been found, or the time limit was reached, where the time limit was set as given in Table 4. Furthermore, to accelerate the runtime for the HVNS, we use the observation that for our test instances, there was not sufficient incentive to close facilities once opened, i.e. all facilities were either opened once and never closed, or not opened at all. Thus we did not apply the facility closing mechanism in the neighborhood generation process. Finally, we note our parameter settings for the HVNS: the maximum number of neighborhoods, K_{max}, was set to 3, with N_{max}^K for $K = 1, 2, 3$ set to 3, 2,and 1, respectively. These values were chosen after extensive experimentation. Whilst taking more neighborhoods and higher N_{max}^K values can lead to better solutions in some cases, the benefits seem to be marginal, and not uniformly better. For example, taking $K_{max} = 4$, with N_{max}^K for $K = 1, 2, 3$ set to 12, 6, 4,and 3, respectively was slightly better in 8 instances, worse in 10, had an average gap of 2.34% versus 2.33%, and took about 4% longer to run.

It is apparent from Table 8 that HVNS is able to make noticeable improvements in solution quality, albeit at some computational cost. It finds an improved solution in all but one instance, and gives better or no worse solutions than the MIP model on 22 of the 30 instances. Its benefits compared to the MIP model are particularly noticeable on the harder instances, for which it finds much better solutions in less CPU time.

Finally, we note that the heuristics would appear to be much less sensitive to problem parameters, and instance variation, than is the MIP model. The latter clearly struggles as problem size grows. Whilst the PBP heuristics naturally take longer with increasing number of time periods, since they require a MIP solve per time period, the solve time per time period does not seem to be particularly sensitive to network size, and the solution quality is markedly consistent. The HVNS method also seems to offer consistent improvements.

Table 8: The computational results of CPLEX and the proposed algorithms

	CPLEX				PBP(VariantI)			PBP(VariantII)			PBP(VariantIII)			HVNS			
Inst.	LB	Obj.	Gap	Time	Obj.	Time	Gap	Obj.	Time	Gap	Obj.	Time	Gap	Obj.	Bcpu	Gap	Imp.
1	21004.36	21004.36	0.00	26	21118.66	11	0.54	21118.66	11	0.54	21118.66	13	0.54	21004.36	16	0.00	1
2	41091.49	41091.49	0.00	38	42459.62	34	3.33	42409.06	30	3.21	42459.62	39	3.33	41091.49	174	0.00	7
3	80397.13	80397.13	0.00	360	82399.55	110	2.49	82249.54	100	2.30	82399.55	122	2.49	80397.13	262	0.00	1
4	37073.19	37073.19	0.00	187	38092.52	15	2.75	38276.26	16	3.25	38092.52	18	2.75	37073.19	410	0.00	6
5	73252.05	73252.05	0.00	837	76300.90	44	4.16	76858.59	38	4.92	76300.90	58	4.16	73252.05	225	0.00	2
6	158829.81	158829.81	0.00	2449	164711.48	124	3.70	165082.62	113	3.94	164711.48	153	3.70	158829.81	202	0.00	1
7	22605.48	22605.48	0.00	706	22608.65	129	0.01	22631.11	103	0.11	22608.65	164	0.01	22605.48	1918	0.00	2
8	45541.65	45588.33	0.10	9782	46251.83	345	1.56	46219.32	330	1.49	46251.83	463	1.56	45582.70	8102	0.09	5
9	105024.61	105024.61	0.00	10142	107013.31	1181	1.89	106891.10	1221	1.78	107013.31	1528	1.89	105024.61	19710	0.00	7
10	19365.21	19877.83	2.65	10000	20308.69	157	4.87	20062.04	141	3.60	20061.73	205	3.60	19891.85	400	2.72	2
11	37697.40	39073.57	3.65	20000	40527.48	378	7.51	40527.48	368	7.51	40499.92	478	7.43	40118.85	3500	6.42	2
12	91116.00	92603.74	1.63	40000	97215.83	1342	6.69	97215.83	1051	6.69	98030.75	1490	7.59	94440.30	38753	3.65	8
13	29520.27	29520.27	0.00	3997	29627.63	454	0.36	29628.27	473	0.37	29627.63	564	0.36	29562.66	3527	0.14	1
14	61479.91	61479.91	0.00	8945	62942.07	1165	2.38	62970.16	1479	2.42	62903.28	1950	2.32	62308.18	19796	1.35	5
15	149232.15	149232.15	0.00	29098	153852.18	4822	3.10	154978.43	4662	3.85	154760.17	5530	3.70	153054.79	51544	2.56	10
16	26173.73	27982.48	6.91	15000	28144.81	493	7.53	28144.81	479	7.53	28144.81	665	7.53	27901.46	10567	6.60	3
17	53276.29	60810.45	14.14	30000	58209.01	1608	9.26	58151.74	1442	9.15	58151.74	1717	9.15	57316.47	26411	7.58	7
18	130445.51	145804.90	11.77	60000	138930.32	4958	6.50	138814.48	4933	6.42	138858.81	6311	6.45	138118.05	48127	5.88	4
19	36283.66	36878.57	1.64	20000	38886.77	3273	7.17	37252.04	3105	2.67	37252.04	3289	2.67	36843.54	9842	1.54	4
20	73125.38	NA	–	40000	78810.43	5655	7.77	75984.38	6214	3.91	75952.23	7861	3.87	75952.23	33698	3.87	2
21	177635.35	NA	–	80000	193846.27	17152	9.13	187264.10	14528	5.42	186958.12	18314	5.25	186961.47	60519	5.25	2
22	32656.77	32765.88	0.33	20000	32908.06	1464	0.77	32804.72	2074	0.45	32804.72	2686	0.45	32789.76	12069	0.41	1
23	68460.63	99865.61	45.87	40000	71013.67	4236	3.73	70661.84	4687	3.22	70661.89	6139	3.22	70616.47	27344	3.15	1
24	167752.18	NA	–	80000	176473.81	15027	5.20	176623.03	13186	5.29	176492.07	17027	5.21	176051.34	73553	4.95	3
25	36196.34	36240.72	0.12	25000	36963.58	2783	2.12	36951.86	2564	2.09	36934.97	3909	2.04	36894.58	22767	1.93	2
26	77792.22	NA	–	50000	80266.72	10097	3.18	80010.25	8660	2.85	80260.38	11640	3.17	79765.71	38025	2.54	3
27	202493.63	NA	–	100000	209246.99	35084	3.34	207730.34	31559	2.59	208381.45	32729	2.91	207223.80	31559	2.34	2
28	35350.97	NA	–	25000	36224.15	3748	2.47	36141.71	2886	2.24	36141.51	4361	2.24	36045.97	21085	1.97	2
29	76342.58	NA	–	50000	79156.11	27621	3.69	78510.86	13255	2.84	78863.31	13927	3.30	78508.00	15677	2.84	1
30	199560.89	NA	–	100000	206797.38	36346	3.63	203769.39	37325	2.11	205667.50	34268	3.06	203769.39	37325	2.11	0
AV.	78892.56	–	–	29052	82376.95	5995	4.03	81864.47	5234	3.49	81945.52	5921	3.53	80966.52	20654	2.33	

6 Concluding remarks and future directions

Dynamic, integrated facility location and network design problems, whilst having obvious attractions for practical application, would appear to offer significant computational challenges. Whilst the work presented here shows the importance of careful MIP formulation for exact solution, and the effectiveness of heuristic methods in finding good-quality feasible solutions in reasonable time, there is much that could still be done. For example, the use of MIPs to define and search neighborhoods is finding success in vehicle routing and other application areas; its use could be considered here. Additional strengthening of MIP models may be possible, and an investigation of strong cutting planes may be worthwhile, as might reformulations such as could be solved by column generation. Other natural problem variations, such as capacitated versions, may also be of interest.

References

[1] A. Weber. Uber den standort der industrian,translated as (in 1929): Alferd weber's theory of the location of industries, university of chicago press,. 1909.

[2] S. Melkote. *Integrated models of facility location and network design,*. PhD thesis, Northwestern University, Doctor of Philosophy, 1996.

[3] C. Cocking. *Solutions to Facility Location-Network Design Problems*. PhD thesis, University of Heidelberg, Doctor of Philosophy Thesis, 2008.

[4] A. Klose and A. Drexl. Facility location models for distribution system design. *European Journal of Operational Research*, 162(1):4–29, 2005.

[5] G. Nagy and S. Salhi. Location-routing: Issues, models and methods. *European Journal of Operational Research*, 177(2):649–672, 2007.

[6] S. Alumur and B.Y. Kara. Network hub location problems: The state of the art. *European Journal of Operational Research*, 190(1):1–21, 2008.

[7] C.S. ReVelle, H.A. Eiselt, and M.S. Daskin. A bibliography for some fundamental problem categories in discrete location science. *European Journal of Operational Research*, 184(3):817–848, 2008.

[8] M.T. Melo, S. Nickel, and F. Saldanha-da Gama. Facility location and supply chain management - a review. *European Journal of Operational Research*, 196(2):401–412, 2009.

[9] T.L. Magnanti and R.T. Wong. Network design and transportation planning: Models and algorithms. *Transportation Science*, 18(1):1–55, 1984.

[10] A. Balakrishnan, T.L. Magnanti, and R.T. Wong. A dual-ascent procedure for large-scale uncapacitated network design. *Operations Research*, 37(5):716–740, 1989.

[11] A. Balakrishnan, T. Magnanti, and P. Mirchandani. Network design. *Annotated Bibliographies in Combinatorial Optimization*, pages 311–334, 1997.

[12] B. Gendron, T.G. Crainic, and A. Frangioni. Multicommodity capacitated network design. *Telecommunications Network Planning*, pages 1–19, 1999.

[13] A. Costa, J-F. Cordeau, and B. Gendron. Benders, metric and cutset inequalities for multicommodity capacitated network design. *Computational Optimization and Applications*, 42:371–392, 2009. 10.1007/s10589-007-9122-0.

[14] Nemhauser G.L. Hewitt, M. and M.W.P. Savelsbergh. Combining exact and heuristic approaches for the capacitated fixed charge network flow problem. *INFORMS Journal of Computing*, 22:314–325, 2010.

[15] O. Berman, D.I. Ingco, and A.R. Odoni. Improving the location of minisum facilities through network modification. *Annals of Operations Research*, 40(1):1–16, 1992.

[16] M.S. Daskin, A.P. Hurter, and M.G. VanBuer. Toward an integrated model of facility location and transportation network design, 1993.

[17] S. Melkote and M.S. Daskin. An integrated model of facility location and transportation network design. *Transportation Research Part A*, 35(6):515–538, 2001.

[18] S. Melkote and M.S. Daskin. Capacitated facility location-network design problems. *European Journal of Operational Research*, 129(3):481–495, 2001.

[19] Z. Drezner and G.O. Wesolowsky. Network design: Selection and design of links and facility location. *Transportation Research Part A*, 37(3):241–256, 2003.

[20] R. Ravi and A. Sinha. Approximation algorithms for problems combining facility location and network design. *Operations Research*, 54:73–81, 2006.

[21] X. Chen and B. Chen. Approximation algorithms for soft-capacitated facility location in capacitated network design. *Algorithmica*, 53(3):263–297, 2009.

[22] C. Cocking and G. Reinelt. Heuristics for budget facility location-network design problems with minisum objective. In *Annual International Conference of German Research Society, University of Augsburg*, volume 12, pages 563–568. Operations Research Proceedings, Springer Berlin Heidelberg, 2008.

[23] J.F. Bigotte, D. Krass, A.P. Antunes, and O. Berman. Integrated modeling of urban hierarchy and transportation network planning. *Transportation Research Part A*, 44(7):506–522, 2010.

[24] A.S. Manne. Capacity expansion and probabilistic growth. *Econometrica*, 29(4):632–649, 1961.

[25] A.S. Manne. *Investment for capacity expansion: size, location, and time phasing*. MIT press, Cambridge, 1967.

[26] R. H. Ballou. Dynamic warehouse location analysis. *Journal of Marketing Research*, 5:271–276, 1968.

[27] S.H. Owen and M.S. Daskin. Strategic facility location: A review. *European Journal of Operational Research*, 111:423–447, 1998.

[28] L.V. Snyder. Facility location under uncertainty: A review. *IIE Transactions*, 38:537554, 2006.

[29] A.B. Arabani and R.Z. Farahani. Facility location dynamics: An overview of classifications and applications. *Computers and Industrial Engineering*, 62:408420, 2012.

[30] D.J. Sweeney and R.L. Tatham. An improved long-run model for multiple warehouse location. *Management Science*, 22(7):748–758, 1976.

[31] S. Chand. Decision/forecast horizon results for a single facility dynamic location/relocation problem. *Operation Research Letters*, 7(5):247–251, 1988.

[32] Z. Drezner and G.O. Wesolowsky. Facility location when demand is time dependent. *Naval Research Logistics*, 38(5):763–771, 1991.

[33] M. Bastian and M. Volkmer. A perfect forward procedure for a single facility dynamic location/ relocation problem. *Operations Research Letters*, 12(2):11–16, 1992.

[34] G. Andretta and F. M. Mason. A note on a perfect forward procedure for a single facility dynamic location/ relocation problem. *Operation Research Letters*, 15(2):81–83, 1994.

[35] R.Z. Farahani, Z. Drezner, and N. Asgari. Single facility location and relocation problem with time dependent weights and discrete planning horizon. *Annals of Operations Research*, 167(1):353–368, 2009.

[36] A.J. Scott. Dynamic location-allocation systems: some basic planning strategies. *Environment and Planning*, 3(1):73–82, 1971.

[37] G.O. Wesolowsky. Dynamic facility location. *Management Science*, 19(11):1241–1248, 1973.

[38] G.O. Wesolowsky and W.G. Truscott. The multi-period location-allocation problem with relocation of facilities. *Management Science*, 22(1):57–65, 1975.

[39] A. Shulman. An algorithm for solving dynamic capacitated plant location problems with discrete expansion sizes. *Operations Research*, 39:423–36, 1991.

[40] P. Chardaire, A. Sutter, and M-C. Costa. Solving the dynamic facility location problem. *Networks*, 28(2):117–24, 1996.

[41] J. Current, S. Ratick, and C. ReVelle. Dynamic facility location when the total number of facilities is uncertain: A decision analysis approach. *European Journal of Operational Research*, 110(3):597–609, 1998.

[42] CO. Fong and V. Srinivasan. The multiregion dynamic capacity expansion problem- an improved heuristic. *Management Science*, 32(9):1140–1152, 1986.

[43] M.T. Melo, S. Nickel, and F. Saldanha-da Gama. Dynamic multi-commodity capacitated facility location: A mathematical modeling framework for strategic supply chain planning. *Computers and Operations Research*, 33:181–208, 2006.

[44] J. Dias, M.E. Captivo, and J. Clmaco. Capacitated dynamic location problems with opening, closure and reopening of facilities. *IMA Journal of Management Mathematics*, 17(4):317–348, 2006.

[45] J. Dias, M. Eugnia Captivo, and J. Clmaco. Efficient primal-dual heuristic for a dynamic location problem. *Computers and Operations Research*, 34:1800–1823, 2007.

[46] T. Melo, S. Nickel, and F. Saldanha-da Gama. An lp-rounding heuristic to solve a multi-period facility relocation problem, 2009.

[47] Y. Hinojosa, J. Puerto, and FR. Fernndez. A multi-period two-echelon multicommodity capacitated plant location problem. *European Journal of Operational Research*, 123(2):271–91, 2000.

[48] S. Velten. *Heuristic solution methods for a dynamic location model with inventory*. PhD thesis, University of Kaiserslautern, Master thesis, 2002.

[49] Phuong Nga Thanh, Olivier Pton, and Nathalie Bostel. A linear relaxation-based heuristic approach for logistics network design. *Computers and Industrial Engineering*, 59(4):964 – 975, 2010.

[50] Glover F. Lodi A. Fischetti, M. The feasibility pump. *Mathematical Programming*, 104(1):91–104, 2005. cited By (since 1996) 33.

[51] M. Romauch and R. Hartl. Dynamic facility location with stochastic demands. In *Stochastic algorithms: foundations and applications*, pages 180–189. Springer, 2005.

[52] D. H. Lee and M. Dong. Dynamic network design for reverse logistics operations under uncertainty. *Transportation Research Part E: Logistics and Transportation Review*, 45(1):61–71, 2009.

[53] D.L. Kelly and A.J. Marucheck. Planning horizon results for the dynamic warehouse location problem. *Journal of Operations Management*, 4(3):279–294, 1984.

[54] A. M. Hormozi and B. M. Khumawala. An improved algorithm for solving a multi-period facility location problem. *IIE Transactions*, 28(2):105–114, 1996.

[55] M. Gendreau, G. Laporte, and F. Semet. A dynamic and parallel tabu search heuristic for realtime ambulance relocation. *Parallel computing*, 27(12):1641–1643, 2001.

[56] T. ChingJung, H. MingHong, and C. ChiaHo. On the use of genetic algorithms to solve the dynamic location problem. *Journal of the Eastern Asia Society for Transportation Studies*, 5:2247–60, 2003.

[57] W. Chanchan, M. Zujun, and L. Huajun. Stochastic dynamic location-routing-inventory problem in closed-loop logistics system for reusing end-of-use products, 2008.

[58] Cordeau J.-F. Contreras, I. and G. Laporte. The dynamic uncapacitated hub location problem. *Transportation Science*, 45:18–32, 2011.

[59] N. Mladenovic and P. Hansen. Variable neighborhood search. *Computers and Operations Research*, 24(11):1097–1100, 1997.

[60] P. Hansen, N. Mladenovic, and J.A.M. Prez. Variable neighborhood search: methods and applications. *Annals of Operations Research*, 175(1):367–407, 2010.

[61] J. Brimberg, P. Hansen, N. Mladenovic, and . Taillard. Improvements and comparison of heuristics for solving the multisource weber problem. *Operations Research*, 48(3):444–460, 2000.

[62] M. S. Jabalameli and A. Ghaderi. Hybrid algorithms for the uncapacitated continuous location-allocation problem. *The International Journal of Advanced Manufacturing Technology*, 37(1-2):202–209, 2008.