A BILEVEL *P*-MEDIAN PROBLEM FOR LOCATION AND PROTECTION PLANNING OF CRITICAL FACILITIES

by

Nuray Piyade

B.S., Industrial Engineering, Yıldız Technical University, 2006

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering
Boğaziçi University

2009

ACKNOWLEDGEMENTS

First and foremost, I would like to thank to my thesis supervisor Assoc. Prof. Necati Aras for his continuous guidance, support and endless patience throughout this research. He did not only serve as my supervisor but also challenged, encouraged and motivated me throughout my thesis period. This thesis would not be possible without his assistance and his tolerance especially during the last period of my study. I feel myself so lucky for having the opportunity to work with him.

I would like to express my gratitude to my co-supervisor Assist. Prof. Deniz Aksen for his support, guidance and patience throughout this thesis. Even in the hardest times, he was there to help me and made possible the constitution of this thesis through his leading comments and contributions.

I would like to express my gratitude to Assoc. Prof. Ali Tamer Ünal and Prof. Vedat Verter for taking time to examine this thesis and taking part in my thesis jury.

I am also very grateful to Prof. R. Oktay Gözü for his valuable supports throughout my education life.

Finally, and above all, I would like to thank my mother, my sisters and my brother, whose love and support I could always count on. My special appreciation goes to my mother for her endless love and patience on me. This thesis is dedicated to her whom I owe everything in my life.

I thankfully acknowledge the support of TÜBİTAK (Turkish Technological and Scientific Research Institute).

This research has been partially supported by Boğaziçi University Research Fund Grant No: 08HA301D.

ABSTRACT

A BILEVEL *P*-MEDIAN PROBLEM FOR LOCATION AND PROTECTION PLANNING OF CRITICAL FACILITIES

In this thesis, we focus on the problem of location and protection planning of critical facilities. This problem involves a Stackelberg game between a system planner (defender) and a potential attacker. The system planner aims to both find the locations of p critical service facilities and determine the ones among them that should be protected. Following this twofold action, the attacker decides which facilities to interdict having the location and protection information of the opened facilities. This problem involves strategic decisions which can be taken either sequentially or simultaneously. In this study, we consider both of these cases. In the first case, the system planner first decides on the locations and then determines the protection plan of these facilities. In the second case however, the system planner gives concurrent decisions about location and protection of the facilities. Both cases are of a bilevel nature. Therefore, we formulate this problem as a bilevel mixed-integer programming problem. We propose two solution methods. The first one is a two-phase tabu search heuristic for the case which involves concurrent decision process and a sequential solution method for the second case where the system planner prefers to give sequential decisions. Both of the methods include a binary search tree embedded into it. The efficiency of the proposed algorithms is tested on an extensive amount of randomly generated test instances each with two budget levels, namely low and high.

We also consider another case where the system planner does not have any financial resources to protect the facilities from an attack. This line of vision helps system planner to determine the critical facilities from the attacker's perspective. The results show that the protection budget plays a significant role in maintaining the service accessibility after a possible attack.

ÖZET

KRİTİK TESİSLERİN YERSEÇİMİ VE KORUMASI PLANLAMASI İÇİN ÇİFT-DÜZEYLİ *P*-MEDYAN PROBLEMİ

Bu tez çalışmasında, kritik tesisler için yerseçimi ve koruma planlaması problemi üzerinde çalışılmıştır. Bu problem, bir sistem planlamacı (koruyucu) ve potansiyel saldırgan arasında gerçekleşen statik Stackelberg oyunu içerir. Sistem planlamacı, *p* kritik hizmet tesisinin yerseçimini yapmayı ve hangilerinin korunması gerektiğini belirlemeyi amaçlar. Bu iki aşamalı eylem gerçekleştikten sonra, tesislerin yerleşimi ve koruma durumları bilgisine sahip olan saldırgan, saldıracağı tesisleri belirler. Bu karar verme problemi, ardışık veya eş-zamanlı verilebilen stratejik ve taktiksel kararlar içerir. Bu çalışmada, bahsedilen her iki durum da göz önünde bulundurulmuştur. İlk durumda, sistem planlamacı öncelikle açılacak tesislerin yerlerine karar verir ve daha sonra bu tesisler için bir koruma planı hazırlar. İkinci durumda ise, sistem planlamacı tesislerin yerseçimi ve koruma planlamasını yaparken eş-zamanlı kararlar alır. Her iki durumda da problem, çift-düzeyli bir yapıya sahiptir. Bu nedenle, problem çift-düzeyli karışık-tamsayılı matematiksel programlama olarak modellenmiştir. Değişkenlerin sürekli olduğu durumda bile çift-düzeyli problemlerin çözümü oldukça zordur ve etkili algoritmalara ihtiyaç duyulur. Bu sebeple, problemin çözümünde sistem planlamacının eş-zamanlı karar verdiği durum için iki aşamalı Tabu Arama sezgiseli, ardışık kararların alındığı durum için ise bir sıralı çözüm yöntemi önerilmiştir. Önerilen her iki metod da çift-düzeyli problemin çözümünde ikili arama ağacı algoritmasından yararlanmaktadır. Algoritmaların testi, rassal olarak üretilen problemler üzerinde değişik koruma bütçesi miktarları kullanılarak denenmiş ve sonuçlar alınmıştır.

Bu çalışmada ayrıca sistem planlamacının koruma bütçesine sahip olmadığı durum da incelenmiştir. Probleme bir de bu açıdan yaklaşılarak, saldırganın gözünden sistemdeki kritik tesislerin tespiti yapılmış olmaktadır. Elde edilen sonuçlar göstermektedir ki; koruma bütçesi saldırı sonrası hizmet verebilirliği devam ettirmede önemli bir role sahiptir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS / ABBREVIATIONS

b_j	Protection cost of the facility at site j
b_{rem}	Remaining protection budget
b_{tot}	Protection budget of the defender
c_{1j}	The unit capacity acquisition cost for the facility at site j
c_{2j}	The unit capacity expansion cost for the facility at site j
d_{ij}	Traveling cost (shortest distance cost) per unit distance between demand node i and facility site j
f_j	Fixed cost of opening a facility at site j
i	Index of the customer locations
j	Index of the facility locations
K_{Cplex}	The average number of Cplex 11.0 calls during the entire TS-BST iterations
K_{iter}	The average number of TS-BST iterations performed before any termination criterion is fulfilled
K_{sols}	The average number of neighborhood solutions explored by the TS mechanism
lb	Lower bound
q_i	Total demand of customer i
r	The number of interdicted facilities
R	Radius
ub	Upper bound
τ	Tabu tenure
$\boldsymbol{\sigma}_b$	Subset of protected candidate facility sites
$\boldsymbol{\sigma}_p$	Subset of opened candidate facility sites
$\boldsymbol{\sigma}_r$	Subset of interdicted candidate facility sites
$\boldsymbol{\sigma}_r^*$	Subset of facilities the interdiction of which maximizes the attacker's objective function

AI	After interdiction
AP	Attacker's problem
BI	Before interdiction
BIP	Bilevel integer programming
BLP	Bilevel linear programming
BP	Bilevel programming
BPCF	Bilevel *p*-median problem for the planning of critical facilities
BPPCF	Bilevel *p*-median problem for the planning and protection of critical facilities
BST	Binary search tree
CA	Closest assignment
ECL	Minimum expected coverage loss
IMF	Interdiction median problem with fortification
MCL	Minimax coverage loss
MCPC	Maximal covering problem with precedence constraints
MILP	Mixed integer linear programming
MIP	Mixed integer programming
MIPS	Million instructions per second
MPUF	Median problem with unreliable facilities
m.u.	Monetary units
OR	Operations Research
RFLP	Reliability fixed-charge location problem
RIC	*r*-interdiction covering problem
RIM	*r*-interdiction median problem
RIMF	*r*-interdiction median problem with fortification
RNS	Ratio neighborhood size
RPMP	Reliability *p*-median problem
RS-MCP	Reduced stochastic *r*-interdiction median problem with fortification
S-RIMF	Stochastic *r*-interdiction median problem with fortification
TS	Tabu search
UFLP	Uncapacitated facility location problems

1. INTRODUCTION

The increasing importance of critical infrastructure in our social life makes finding reliable service systems by planning and protection strategies very crucial. A service system should be reliable either to an event that humans do not manage, i.e. operational failure, natural disaster etc., or to an intentional attack. While the definition and realization of critical infrastructure has evolved through time, today the protection issue becomes more important than the adequacy of the infrastructure because of the growing threat of terrorism. This interesting topic attracts the attention of Operations Research (OR) professionals in our time, as well. First attempts to deal with this problem were the interdiction models that aim to identify the least reliable (in other words the most critical) elements in a system. However, it is also important to protect the most critical components in the system, rather than backing-up the least reliable components. Therefore, the option of protection is taken into consideration and it is incorporated into this problem. Once a disruption occurs in the system, there is very little recourse regarding system infrastructure since strategic decisions, i.e. the location of facilities, cannot be changed quickly. For that reason, it is important to account for disruptions or other effects during the design phase of the system so as to minimize the service provision loss after a disruption.

In the problem of location and protection planning of the critical facilities, strategic decisions can be made either simultaneously or sequentially. In both cases, the system planner determines the locations of p facilities and plans the protection status of these facilities. He/she has two options while constructing such a system. The primary option is determining facility locations to meet the customer demands and then making the protection plan of the network structure. The other option is considering the location and protection plan of the system at the same time. A system planner is responsible for giving these decisions, whereas an attacker plans to attack the system in order to give disruption in the service providence. Therefore, this problem has a bilevel nature and can be thought of as a Stackelberg game where one of the two players gives a decision according to the move of the other.

In this thesis, we consider location and protection planning of the critical infrastructure problem which we refer to as the “Bilevel p-median Problem for the Planning and Protection

of Critical Facilities", the BPPCF in short. In this problem, the system planner decides both the location of the facilities to open and the protection of these opened facilities where, on the other hand, an attacker plans to hit these opened facilities. Both of these decisions are strategic and their sequence is important. Therefore, we handle this problem from two different points of view as discussed above. The first case comprises simultaneous decision making for the location and protection planning of the facilities. The other case includes sequential planning process where the system planner first locates p facilities then decides the protection planning of this configuration. We formulate the first case as a bilevel programming problem and propose a two-phase tabu search heuristic as a solution method. We also propose a solution method to solve the sequential case. In each case, we assume that candidate facility sites have different fixed opening costs. Another assumption is that there is a capacity restriction on facility sites and accordingly they have different capacity expansion costs. We analyze the positive effect of using protection resources against intentional attacks by changing the budget level of decision maker. In addition, the problem is modeled and is solved for the case where system planner does not have any protection budget.

Bilevel programming (BP) approach is used to formulate the BPPCF problem. Being generically non-convex and non-differentiable, bilevel programs are inherently hard. Even the simplest BP where both the upper and lower level problems are formulated as linear programming models is proven to be $\mathcal{NP}$–hard by Ben-Ayed and Blair [1]. Efficient algorithms are needed in order to deal with these problems. Therefore, a two-phase Tabu Search heuristic, which is referred to as TS-BST, is proposed for the BPPCF problem. Also a Sequential Solution Method, which is referred to as SSM, is applied when the system planner gives the location and the protection decisions of the facilities sequentially.

To the best of our knowledge, this difficult problem was not tackled before considering both location and protection decisions together. Another significant contribution of this study is that we propose two different solution methods in order to solve this difficult bilevel problem.

The remainder of this thesis is organized as follows. Chapter 2 gives a brief overview of critical infrastructure planning problem including network reliability models, interdiction models and protection efforts. In Chapter 3, the mathematical programming formulation of

the problem is presented. A two-phase Tabu Search heuristic method for BPPCF problem is described in Chapter 4. Next, a sequential solution methodology is described in Chapter 5. The computational results of the proposed algorithms and comparison of SSM and TS-BST methods are given in Chapter 6. Finally, Chapter 7 presents the general conclusions and further research areas.

2. LITERATURE SURVEY

2.1. Critical Infrastructure Planning: Network Reliability Models and Interdiction Models

Facility location has been one of the interesting and spacious topics amongst the operations research (OR) society since Alfred Weber introduced the problem of locating a single warehouse so as to minimize the total distance between the warehouse and its customers [2]. Henceforth, several variations of this problem that differ in assumptions, decision making aspects, mathematical modeling and solution methods have been studied in the literature. One of the more recent research topics in facility location theory is that incorporating the protection issues in location decisions of the critical infrastructure. In this chapter, we review the literature that is directly related to the planning and protection of critical infrastructure. We first overview the reliability concept in the facility location which accounts for the failure probability of facilities during their operation. This is followed by a summary of the interdiction models in the literature. Both of these models deal with the planning of systems that are resilient to disruptions, but they do not consider the option of protection. Therefore, we conclude this chapter by discussing the models that incorporate protection efforts in critical facility planning.

The necessity for the protection of critical facilities arises from the fact that a facility can become inoperative due to some intentional factors, i.e. sabotages or terrorist attacks. A facility can also become out of order due to several other reasons, e.g. natural and systematic causes. A system of facilities is said to be reliable according to how well it performs after some parts of the system have failed. A wide range of studies is dedicated to this type of system reliability problems in the literature. Thus, we give a brief summary of reliability studies before discussing the planning and protection of critical facility problems.

The concept of facility reliability is first introduced by Drezner [3] who considers the unreliable versions of the well-known p-median and p-center problems. The unreliable p-median problem is defined by assuming that a facility has a given probability of becoming

inactive where the objective is to minimize the sum of weighted distances between demand points and their closest facilities. The (p,q)-center problem is the extension of p-center problem where p facilities need to be located in a situation where up to q of them may fail to render service at the same time. He formulates these problems and solves them with a heuristic method employing an assumption that the probability of becoming inactive for each facility is not independent.

Snyder and Daskin [4] introduce two different models called the reliability p-median problem (RPMP) and the reliability fixed-charge location problem (RFLP) which are based on the p-median and the uncapacitated fixed-charge problems (UFLP), respectively. Both models try to minimize the weighted sum of operating cost and the expected failure cost where they employ a strategy for assigning each customer. According to that strategy each customer is assigned to a primary facility as well as to a set of back-up facilities that give service when the primary facility fails. In this problem, there are two types of facilities: those that can either fail or not. Each facility of the former type has the same probability of failure, q. An optimal Lagrangian relaxation algorithm is used to solve these models. The results show that, in general, the optimal solution of the RFLP uses more facilities than that of the UFLP which can be seen as a result of the risk-diversification effect.

Berman et al. [5] consider a similar model called "median problem with unreliable facilities" (MPUF) which is an extension of Drezner's problem in [3]. The main difference between these two models is that the probabilities of failure are independent in MPUF. They come out with an observation that optimal location patterns are strongly dependent on the probability of facility failures, with facilities becoming more centralized, or even co-located, as the failure probability grows.

Snyder and Daskin formulate several optimization models for the design of reliable facility location systems under a variety of risk measures and operating strategies in [6]. Their first model is based on UFLP which is formulated in two ways: the first one is scenario based modeling which considers all of the failure scenarios and the second one captures the uncertain events explicitly. The latter model they presented allows locating two different types of facilities: facilities that are perfectly reliable and facilities that are subject to failure. This model determines how many of each type should be located in order to achieve the de-

sired objective; namely designing a reliable system vulnerable to disruptions. They provide some extensions to these two models throughout their paper. A more recent study to develop a resilient system against worst-case facility loses is addressed by O'Hanley and Church [7]. In their coverage-type model, the objective is to locate a set of facilities so as to maximize a combination of initial demand coverage and the minimum coverage level following the loss of one or more facilities. They formulate this problem as a mixed-integer program (MIP) and solve by a bilevel decomposition based algorithm.

Brown et al. [8] propose two mathematical models for the reliability problem of the critical infrastructure, which are referred as to "attacker-defender" and "defender-attacker" models, where they incorporate the concept of bilevel programming approach into modeling of the problem. Although they use the term "defender" for the follower, the "attacker-defender" model does not directly represent defensive actions; rather, it identifies a set of most critical components in the system. In their recent study, Berman et al. [9] also employ the bilevel character of leader-follower game while modeling reliable facility network from maximal coverage perspective. The objective of the leader is locating p facilities, which are subject to a terrorist attack or a natural disaster (by the follower), in such a way that the resulting design covers the most demand following a damage to a link. This problem assumes worst-case scenario. That is, the follower selects a link for removal in order to maximize the damage. In another recent research, Matisziw et al. [10] use simulation method for the problem of disruption in facility networks. They argue that simulation method makes assessment of several potential disruption scenarios possible which overcomes some shortcomings of other approaches by providing additional insight and flexibility in network management.

All of the models mentioned up to now consider the initial design of the system whose components are prone to failure. They all address the robustness in the supply side of the system as opposed to the models that aim to design a system that is sustainable under uncertain future events, which are reviewed thoroughly in [2]. Snyder et al. [11] provide a comprehensive survey of reliability models, including network design issues, based on some categories, i.e. designing from scratch versus modifying the existing system.

A different line of research in planning and protection of critical infrastructure includes interdiction models. The first interdiction model was introduced by Wollmer [12] in

order to make a sensitivity analysis on a flow network, which can be helpful in determining how sensitive a transportation system might be to having its roads closed down for repair or being tied up by traffic accidents. One other important utilization of these models is addressed by military planners in order to evaluate the impact of losses in nodes and linkages of transportation networks. In an interdiction model, the aim is to examine the vulnerabilities of a service system (e.g., product distribution system, emergency response system) from the perspective of an attacker. It is obvious that this perspective would dictate the attacker to identify the most critical elements of the service system and eliminate those so as to cause the maximal disruption in the service provision of the system. This perspective makes sense since an intelligent and belligerent attacker's attitude is towards destroying the most reliable, i.e. most critical, components in a system.

The main difference between reliability models and interdiction models is that, while reliability models consider the initial design of a reliable system which is prone to attacks, the interdiction models seek to find the most critical elements in an existing system. Church et al. [13] categorize interdiction models in terms of their objective functions and structural characteristics where they also provide a comprehensive survey of interdiction models. Accordingly, the following major types of interdiction have been studied in the literature.

- Node or arc destruction (removal) in a maximum flow network.
- Arc removal in a maximum flow network where interdiction successes are binary random variables and arc capacities are either known or uncertain.
- Arc removal in a directed shortest path network.
- Arc interdiction and its impact on shipment revenue in a multi-commodity shortest path network.
- Supply or emergency facility interdiction in a service network.

There are several arc interdiction models for network structures in the literature (more recent ones are [14], [15] and [16]); however, little emphasis is given on the facility interdiction problem. Church et al. [13] are the first who formulate two facility interdiction models: the *r*-interdiction median problem (RIM) and the *r*-interdiction covering problem (RIC). The objective in both models is to identify a subset of *r* facilities among *p* opened ones whose loss would cause the most disruption in the service delivery by the facilities. As can be seen

easily, the RIM (RIC) model is actually the antithesis of the well-known p-median (maximal covering) problem. The objective of the p-median problem is to determine the best locations of p facilities among a given number of potential sites in order to satisfy customer demands. RIM, on the contrary, tries to maximize the demand-weighted total distance in the wake of an interdiction of r existing facilities the customers of which cannot get service from them anymore, thus need to be reallocated to undamaged facilities.

No recent attempts are made to improve interdiction models for facility network designs. Instead, researchers consider the option of protecting the existing system and propose so-called fortification models as a complement to interdiction models. The motivation behind these models is that redesigning an entire system with limited resources is not always reasonable. In these fortification models, the aim is to identify the facilities whose protection upholds the post-attack functionality of the network as much as possible. Given that service facilities are prone to man-made threats, one might consider redesigning the entire service network by relocating facilities, changing suppliers, or reconfiguring the network infrastructure. However, rather than pursuing such a potentially expensive undertaking, a less costly alternative would be the enhancement of the existing infrastructure by making investments towards the protection of the facilities. This way, the damage caused by the attacks is reduced or in some cases the loss of a facility is totally prevented.

The first fortification model is due to Church and Scaparra [17] where the authors incorporate the option of protecting critical facilities against attacks into their RIM model. They call the resulting mixed-integer linear programming (MILP) problem the interdiction median problem with fortification (IMF). IMF can be described as identifying q facilities to be protected in a service network consisting of p facilities and n customer nodes such that the total demand satisfaction cost expressed as the demand-weighted shortest distance between $(p–r)$ non-interdicted facilities and customers is as small as possible. The attacks which put r facilities out of service aim at the maximization of the same total demand satisfaction cost. Clearly, $p \geq q + r$ must hold true. Moreover, it is assumed that the attacker has complete information about the protection status of the facilities. Note that the network planner (called the defender in the sequel) and the attacker have conflicting objectives. The IMF is solved using the general-purpose commercial MILP solver Cplex 7.0 embedded in the optimization software suite OPL Studio 3.5. Since the IMF formulation is based on an

explicit enumeration of all possible ways of losing r out of p facilities, its size grows exponentially as p and r increase, which results in excessive computation times. Therefore, the authors can only solve instances with up to 20 facilities, 10 fortifications, and at most four interdictions.

In order to overcome the size related restrictions in IMF, Scaparra and Church [18] develop another MILP formulation of the same problem which is called maximal covering problem with precedence constraints (MCPC). The new solution approach not only produces good approximations to the best fortification strategies, but also provides upper and lower bounds that can be used to reduce the size of the original model. The resulting reduced model can then be solved to optimality by the commercial MILP solver Cplex.

The MCPC model in [18] requires a complete enumeration of all $\binom{p}{r}$ possible ways of interdicting r out of the p facilities. In their most recent work, Scaparra and Church [19] make use of bilevel programming approach in order to handle this restriction. They propose a bilevel programming (BP) formulation of the r-interdiction median problem with fortification referred to as RIMF and solve it by an implicit enumeration method. In RIMF formulation, the lower-level problem corresponds to the RIM described in [13] where the attacker (follower) has to solve a pure interdiction problem according to the decisions of the defender (leader) that solves the upper-level fortification problem. RIMF can handle relatively larger sized problems with the help of an observation which was presented in [13]. This allows solving RIMF based on an implicit enumeration algorithm performed via tree search. The idea behind the observation is that, given the fortification scheme of the defender, at least one of the r facilities that would be the solution of the low level RIM problem should be selected to be fortified. At most $(r^{q+1}-1) / (r-1)$ RIM problems need to be solved in the search tree conditional on the protection plan of the defender. Consequently, larger problem instances and the problems that could not be solved optimally by the former methodologies can be solved to optimality.

Aksen et al. [20] extend the RIMF problem by introducing budget constraint to the defender's upper-level problem in place of the cardinality constraint that indicates the total number of fortified facilities is to be q. Moreover, they also consider that each facility has a

flexible service capacity, which can be expanded at a unit cost to accommodate the demand of customers who were serviced by some other interdicted facility before the attack.

O'Hanley et al. [21] handle this problem from the biological point of view and present two new models, namely minimum expected coverage loss (ECL) and minimax coverage loss problems (MCL), for locating and protecting critical reserve sites. The objectives of the ECL and MCL are to minimize expected species losses over all possible site loss patterns outside the reserve network and to minimize maximum species losses following the worst case loss of a limited subset of non-reserved sites under limited acquisition budget, respectively. ECL is formulated as an integer program and solved by a commercial solver, whereas MCL is modeled as a bilevel mixed-integer program which is solved by using bilevel decomposition algorithm.

The very recent model developed for a fortification problem is studied in a working paper due to Liberatore et al. [22]. They adopt the RIMF problem to a more realistic case where the number of possible losses is random. The resulting problem is referred to as the stochastic *r*-interdiction median problem with fortification (S-RIMF). They present two formulations of the problem: one is from bilevel programming perspective and the other is a max-covering type formulation which does not require either precedence constraints or ordering of interdiction patterns used in [18]. Decreasing the size of the max-covering type formulation by producing lower and upper bounds, the authors obtain a reduced formulation which is referred to as RS-MCP. The goal is to minimize the expected cost which is expressed as the probability weighted sum of the costs associated with the worst-case interdiction patterns for every feasible value of r by using increasing and decreasing probability distributions. An algorithm is proposed to solve RS-MCP along with two heuristic approaches derived from three proposed rules. They emphasize the importance of considering the stochastic nature of the problem by comparing the results with the corresponding ones in deterministic case.

2.2. Bilevel Programming Approach

BP techniques are mainly developed for solving decentralized management problems with decision makers in a hierarchical organization [23]. A BP problem is a special case of the multilevel optimization problem with two levels or two parties, one of whom takes the leader position, and the other one is the follower making his or her plan based on the leader's decision. A BP problem partitions the control over the variables between two hierarchical levels [24]. Decision makers at each level attempt to optimize their individual objectives where there is an interaction between these two hierarchical levels reflecting a decentralized decision making situation. That is, the higher level in the hierarchy can only influence rather than dictating the choices of the lower level. Applications of BP models in the literature include transportation planning, environmental systems, production systems, resource allocation in organizations.

The first appearance of bilevel problems can be traced back to the nineteen seventies where Bracken and McGill [25] introduced the first bilevel concept which they adopted into the military-related applications as well as production and marketing decision making problems. However, it was not until the early nineteen eighties that usefulness of software and practical problems aroused researchers to pay more attention to bilevel programs [26].

Being generically non-convex and non-differentiable, bilevel programs are inherently hard. Even the simplest BP where both the upper and lower level problems are formulated as linear programming models is proven to be $\mathcal{NP}$–hard by Ben-Ayed and Blair [1]. A substantial amount of effort is dedicated to the linear case of BP problem for a long period of time. The first bibliographical survey is provided by Kolstad [27]. Thereafter, some other surveys dedicated to BP problems are due to Wen and Hsu [28] and Ben-Ayed [24]. A comprehensive bibliography of bilevel and multilevel programming has been written by Vicente and Calamai [26] and by Vicente [29]. The newest review on BP inclusive of literature survey, sample applications, and existing methods is due to Colson et al. [30]. There are also two dedicated textbooks which were authored by Bard [31] and Dempe [32].

Over the years, more research is devoted to more complex BP problems. However, there are a relatively small number of attempts to solve discrete BP problems. This can be due to the higher degree of difficulty of the mixed-integer bilevel linear programming (BLP) problems as the typical concepts for fathoming in traditional branch-and-bound algorithms

for MILP cannot be directly applied to mixed integer BLP [33]. Moore and Bard [33] introduce an implicit enumeration algorithm and explained difficulties in solving integer BP problems. Wen and Yang [34] develop an exact algorithm, which works for relatively small problem instances. A tabu search algorithm is proposed by Wen and Huang [35] when there are integer variables only in the upper level problem. Dempe [36] consider the opposite case where the lower level problem is an integer programming problem and the upper level variables are continuous. He first addresses the problem of the existence of optimal solutions and uses a cutting plane approach to approximate the lower-level feasible region. The more recent attempts to solve discrete BP problems are due to Dempe and Kalashnikov [37], Gümüş and Floudas [38] and Denegre and Ralphs [39]. In the latter paper, Denegre and Ralphs extend the branch-and-bound algorithm of Moore and Bard [33] and propose a branch-and-cut algorithm for integer BP problems which uses cutting plane techniques to produce improved bounds.

There are several solution methods used to deal with linear BP problems with continuous variables. Among them, Kuhn-Tucker approach is the most prevalent method which is achieved by replacing the lower level problem with its Karush-Kuhn-Tucker optimality conditions and attaching the resultant constraints to the leader's problem [31]. For the discrete BLP problems, only a limited number of solution algorithms are introduced. The difficulty arises from the fact that only one of the general fathoming rules used in MILP problems holds for the mixed-integer BLP problems [33]. At this point, the exploitation of heuristic methods, mainly tabu search, genetic algorithms and simulated annealing approaches, offer additional possibilities to solve a relatively large-sized problems even for nonlinear cases. Some recent references for heuristic applications in BP problems are [40], [41] and [42].

3. PROBLEM FORMULATION

In this section, the mathematical programming formulation of the bilevel *p*-median problem for the planning and protection of critical facilities (BPPCF) and the bilevel *p*-median problem for the planning of critical facilities (BPCF) are provided. Critical facility location and protection problem involves strategic decisions which can be taken either sequentially or simultaneously. In the sequential case, the system planner initially determines the optimal locations of the facilities and then considers the protection planning of the existing system. On the other hand, in the simultaneous case he/she considers different system configurations with their optimal protection plan and chooses the best alternative among them. We formulate the simultaneous case as a BP problem and propose a two-phase heuristic method. A sequential solution method is also proposed to solve the problem where the system planner gives sequential decisions. In each case, the problem has a bilevel nature where the system planner and the attacker are the upper and lower level decision makers, respectively. Furthermore, we consider the situation where the system planner has no protection budget. This time, the system planner does not give protection decisions, but he/she locates the facilities in such a way that the total cost after an attack is minimized. This second case is referred to as BPCF problem. We formulate and solve this problem in order to show the positive effect of using protection resources. The two models are explained thoroughly in the following subsections.

3.1. Problem Formulation of Bilevel *p*-Median Problem for Planning and Protection of Critical Facilities

BPPCF is modeled as a bilevel integer programming (BIP) problem. Given the interdependency between interdiction and protection, we believe this is the most suitable formulation for the nature of the problem. In the upper level (*leader's problem*), the system planner (*defender*) is the decision maker who decides about the following: which p facilities should be opened, what should be their initial capacity levels, and which of the opened facilities should be protected. After p facilities are opened, customers will get service from one of them by making their choice on the basis of distance. This implies that each customer

goes to the nearest opened facility. In addition to determining the locations of the *p* facilities, the defender decides also about their capacity. The cost of capacity acquisition at a given facility is linearly proportional to the total demand served by this facility. A further decision of the defender is about which of the *p* facilities to protect. As is the case in [19], we assume that a protected facility becomes immune to any attack, and therefore cannot be interdicted by the attacker. Instead of a cardinality constraint on the number of protected facilities used in [19], we impose a budget constraint which ensures that the total protection cost does not exceed the available budget. Since we assume that facilities have non-uniform protection costs, the defender can protect any number of facilities within the budget limit. Another assumption in our model is that when the attacker selects the facilities to hit, he/she has perfect information about which facilities are protected. If this assumption is relaxed, then the attacker may waste offensive resources by attacking fortified facilities, and his/her attack would not represent worst-case losses inflicted on the system. We assume in the model that facilities are capacitated which is more realistic in practice. Since the reassignment of the customers of the interdicted facilities incurs a cost to the defender, the capacitated case becomes more important from a practical perspective. After the attacker puts the *r* facilities out of service by hitting them, an inevitable consequence is that customers (demand nodes) need to be reassigned to the nearest non-interdicted facilities. This in turn leads to an obligatory capacity expansion in the non-interdicted facilities since no customer must be left out even after the interdiction. In our model, we account for capacity expansion at a unit cost to satisfy the demand of customers who were originally serviced by some other facility before the attack. We remark that the defender's marginal capacity acquisition cost before the attack and marginal capacity expansion cost after the attack may be different from each other.

The defender's objective is to minimize the sum of the total costs incurred before and after the interdiction attempt of the attacker. The total costs incurred before the interdiction include the following components denoted by the prefix BI:

(i) BI-1: The fixed cost of opening p facilities where the facility opening costs may be unequal.

(ii) BI-2: The variable cost of capacity acquisition to ensure that each customer's demand is met from the nearest respective facility that is opened.

Given the decisions of the defender in the upper level, the attacker in the lower level (*follower's problem*) chooses r facilities to interdict where an interdicted facility is destroyed beyond repair. As mentioned before, the attacker is assumed to never hit a protected facility.

On the other hand, the total costs incurred after an interdiction are comprised of the following components denoted by the prefix AI:

(i) AI-1: The sum of the demand-weighted traveling costs between customer locations and the nearest facilities surviving the attack.
(ii) AI-2: The capacity expansion costs at non-interdicted facilities due to the reallocation of customers whose previously assigned facilities are destroyed.

The objective of the attacker becomes the maximization of the cost component AI-1, while the defender pursues the minimization of the sum of all cost components BI-1, BI-2, AI-1, and AI-2. It is clear that BI-2 and AI-2 should both be added to the defender's objective function since capacity acquisition and expansion occur at the expense of the defender. The attacker is only interested in giving the maximum possible disruption in the service accessibility, which is measured by the total demand-weighted traveling cost between customers and the nearest facilities surviving the interdiction. This objective is confined to the cost component AI-1.

The following index sets, parameters, and decision variables are used in the BIP model of BPPCF:

Table 3.1. Index sets in the model of BPPCF

Sets	Definition
$\mathbf{I}$	Set of demand nodes (customers), $\mathbf{I} = \{1,\ldots, n\}$
$\mathbf{J}$	Set of candidate facility sites (locations), $\mathbf{J} = \{1,\ldots, m\}$

Table 3.2. Parameters in the model of BPPCF

Parameters	Definition
d_{ij}	traveling cost (shortest distance cost) per unit distance between demand node i and facility site j.
q_i	demand of customer i
f_j	fixed cost of opening a facility at site j
c_{1j}	unit capacity acquisition cost for the facility at site j
c_{2j}	unit capacity expansion cost for the facility at site j
b_j	protection cost of the facility at site j
b_{tot}	protection budget of the defender
r	the maximum number of facilities that can be interdicted by the attacker

The decision variables used in the model are as follows:

$$X_j = \begin{cases} 1 & \text{if site } j \text{ is chosen to open a facility,} \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

$$Y_j = \begin{cases} 1 & \text{if the facility at site } j \text{ is protected,} \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

$$S_j = \begin{cases} 1 & \text{if the facility at site } j \text{ is lost due to an interdiction,} \\ 0 & \text{otherwise.} \end{cases} \tag{3.3}$$

$$U_{ij} = \begin{cases} 1 & \text{if customer } i \text{ is assigned to the facility at site } j \text{ before the attack,} \\ 0 & \text{otherwise.} \end{cases} \tag{3.4}$$

$$V_{ij} = \begin{cases} 1 & \text{if customer } i \text{ is assigned to the facility at site } j \text{ after the attack ,} \\ 0 & \text{otherwise.} \end{cases} \tag{3.5}$$

We also use an auxiliary set called $\mathcal{F}_{ij}$ which is defined as the subset of candidate facility sites that are at least as close as site j to customer i, i.e., $\mathcal{F}_{ij} = \{ k \in \mathbf{J} \mid d_{ik} \leq d_{ij} \}$. Then, the mathematical model of the BPPCF is given as:

$$\min\ Z_{\text{def}} = \sum_{j\in\mathbf{J}} f_j X_j + \sum_{i\in\mathbf{I}}\sum_{j\in\mathbf{J}} c_{1j} q_i U_{ij} + \sum_{i\in\mathbf{I}}\sum_{j\in\mathbf{J}} c_{2j} q_i (1-U_{ij}) V_{ij} + \sum_{i\in\mathbf{I}}\sum_{j\in\mathbf{J}} q_i d_{ij} V_{ij} \tag{3.6}$$

Subject to

$$\sum_{j\in\mathbf{J}} X_j = p \tag{3.7}$$

$$\sum_{j\in\mathbf{J}} U_{ij} = 1 \qquad \forall i\in\mathbf{I} \tag{3.8}$$

$$\sum_{i\in\mathbf{I}} U_{ij} \le n X_j \qquad \forall j\in\mathbf{J} \tag{3.9}$$

$$\sum_{k\in\mathcal{F}_{ij}} U_{ik} \ge X_j \qquad \forall i\in\mathbf{I},\ \forall j\in\mathbf{J} \tag{3.10}$$

$$Y_j \le X_j \qquad \forall j\in\mathbf{J} \tag{3.11}$$

$$\sum_{j\in\mathbf{J}} b_j Y_j \le b_{\text{tot}} \tag{3.12}$$

$$U_{ij},\ X_j,\ Y_j \in \{0,1\} \qquad \forall i\in\mathbf{I},\ \forall j\in\mathbf{J} \tag{3.13}$$

where

$$\max\ Z_{\text{att}} = \sum_{i\in\mathbf{I}}\sum_{j\in\mathbf{J}} q_i d_{ij} V_{ij} \tag{3.14}$$

Subject to

$$\sum_{j\in\mathbf{J}} V_{ij} = 1 \qquad \forall i\in\mathbf{I} \tag{3.15}$$

$$\sum_{j\in\mathbf{J}} S_j \le r \tag{3.16}$$

$$S_j \le X_j - Y_j \qquad \forall j\in\mathbf{J} \tag{3.17}$$

$$\sum_{i\in\mathbf{I}} V_{ij} \le n X_j (1-S_j) \qquad \forall j\in\mathbf{J} \tag{3.18}$$

$$\sum_{k\notin\mathcal{F}_{ij}} V_{ik} \le 1 + S_j - X_j \qquad \forall i\in\mathbf{I},\ \forall j\in\mathbf{J} \tag{3.19}$$

$$V_{ij} \ge U_{ij}(1-S_j) \qquad \forall j\in\mathbf{J} \tag{3.20}$$

$$S_j,\ V_{ij} \in \{0,1\} \qquad \forall i\in\mathbf{I},\ \forall j\in\mathbf{J}. \tag{3.21}$$

In the above formulation, (3.6)–(3.13) represents the upper level problem and (3.14)–(3.21) corresponds to the lower level problem of BPPCF. Expression (3.6) shows the objective function Z_{def} of the defender. The first component of this objective is the fixed cost of opening p facilities. If facilities do not differ in their opening costs, this objective component will be constant and hence can be discarded. The second component indicates the cost of

initial capacity acquisition incurred by the defender at the planning phase of p facilities. It can be viewed as the variable cost of establishing p facilities such that a sufficient capacity is allocated for serving the customers assigned to them. If the same marginal capacity acquisition cost applies to all facility sites, i.e., if $c_{1j} = c_1$ for all $j \in$ **J,** this objective component can be discarded, as well. The third objective component in (3.6) represents the total cost of capacity expansions at non-interdicted facilities due to the reallocation of those customers who were serviced by some interdicted facility before the attack. Note that this objective component in Z_{def} does not apply to customers whose original facility assignment —given by the binary decision variables matrix $[U_{ij}]$— is preserved in the optimal solution of the interdiction problem in (3.14)–(3.21). In other words, if customer i was going to the closest facility opened at site j before the attack, and if this situation does not change after the attack, then the retention of customer i at facility j should not incur any capacity expansion cost. To ensure this, we multiply the post-attack assignment variable V_{ij} with (1– U_{ij}), which is then multiplied with the cost term $c_{2j}\, q_i$ and summed over all customers and facility sites. The last objective component in Z_{def} sums the demand-weighted traveling costs between customers and non-interdicted facilities after the attack.

Constraint (3.7) requires that exactly p facilities be opened in the upper level problem. Constraints (3.8) require that each customer be assigned to exactly one facility before the interdiction. Constraints (3.9) prohibit illegal assignment of customers to a facility that is not opened. The set of constraints (3.10) are closest assignment (CA) constraints, which— together with the constraints (3.9) and binary decision variables X_j —explicitly enforce the assignment of each and every customer to its closest facility put in service. We adopted these CA constraints from Teixeria and Antunes [43] where the working mechanism is explained as follows. For any pair of customer i and site j, if no facility is opened at j ($X_j = 0$), then the constraint has no effect. If a facility is opened at site j ($X_j = 1$), then customer i will go to the facility at site j or to another opened facility at the same or smaller distance than j. It is noteworthy that constraints (3.10)—backed by the constraints (3.9)—also work fine in the presence of multiple sites equidistant from a given customer. Constraints (3.11) suggest that a facility cannot be protected if it is not opened. Constraint (3.12) enforces the budget limit on protection. Binary constraints on the decision variables shown in (3.13) conclude the defender's problem.

The values of the decision variables U_{ij}, X_j, and Y_j determined in the upper level problem act as input parameters for the lower level (attacker's) problem. The attacker primarily seeks to inflict as much disruption as possible on the service network. His maximization objective Z_{att} in (3.14) is defined as the sum of demand-weighted traveling costs between customers and non-interdicted facilities that serve customers in the post-attack period. Similar to the defender's problem, customer-to-facility assignments in the attacker's problem materialize also according to the closest assignment rule, i.e., customers always choose the closest available facility to get service.

The first constraint of the lower level problem is given in (3.15) which forces each customer to be assigned to exactly one facility after the attack. The next constraint in (3.16) states that at most r facilities will be interdicted by the attacker. The reason for keeping this constraint as an inequality rather than an equality is to account for the possible situation where $p < r$ as well as for the situation where the defender's protection budget is so high that more than $(p–r)$ facilities are protected against interdiction. In either case, the trivial solution to the attacker's problem would be to interdict all of the unprotected facilities. Constraints (3.17) are logical conditions, which prevent the attacker from interdicting facilities either not opened at all or opened, but protected by the defender. These inequalities provide at the same time a linkage between the upper and lower level problems. Their right-hand side value can never be negative due to the upper level constraints (3.11). Thus, the interdiction variables S_j are never forced to become less than zero. Constraints (3.18) ensure that no customer is assigned to an interdicted facility or to a facility not opened earlier by the defender. The right-hand side of the inequality in (3.18) is a linear expression for a given value of X_j. So, there is no need to linearize the term $X_j\ (1–S_j)$.

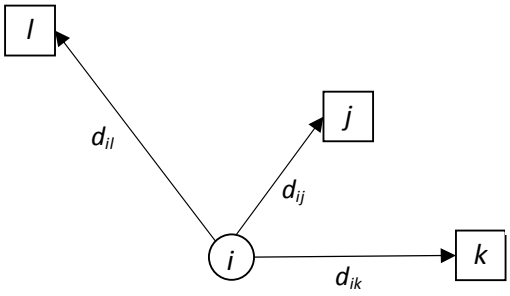
Constraints (3.19) are another expression of CA conditions which is derived from the so-called Church-Cohon (CC) constraints first proposed by Church and Cohon [44]. They enforce the post-attack assignment of customers to the closest non-interdicted facilities. In order to clarify the working mechanism of these constraints let us assume that the relationship between a particular demand node i and three facility locations that are nearest to it is as shown in Figure 3.1. First observe that given a pair of customer i and site j, the summation

on the left-hand side of (3.19) is over all other sites k that are farther from i than j. Then, the case by case analysis of the related CA constraint can be explained as follows.

(i) ***Case 1***: $X_j = 0$, $S_j = 0$. This indicates that facility j is not opened by the defender. In this situation, the attacker cannot hit facility j according to the constraint (3.17). Then the right-hand-side of the CA constraint equals to one and (3.19) becomes ineffective.

(ii) ***Case 2***: $X_j = 1$, $S_j = 1$. In this condition, facility j is opened by the defender and it is lost due to the attacker's action. Then the right-hand-side of the CA constraint again equals to one, which indicates that customer i can be assigned to any facility that is at a larger distance to customer i than facility j is. Hence (3.19) becomes ineffective.

(iii) ***Case 3***: $X_j = 1$, $S_j = 0$. This time the facility j is opened but not interdicted which makes the right-hand-side of (3.19) equal to zero. With the facility j being available in this case, (3.19) now prevents customer i from being assigned to any facility site k that is farther than j. That is, $V_k = V_l = 0$.



Distance order: $d_{ij} < d_{ik} < d_{il}$

CA constraint for $\{i,j\}$: $V_k + V_l \le 1 + S_j - X_j$

CA constraint for $\{i,k\}$: $V_l \le 1 + S_k - X_k$

CA constraint for $\{i,l\}$: No constraint since $\mathfrak{F}_{il} = \{j,k,l\}$

Figure 3.1. Visualization of the CA constraints for the customer node i and three nearest facilities

Here we have to mention the significance of the logical constraints (3.18). If there exist multiple equidistant facilities from a given customer i in the problem, the CA constraints (3.19) would still remain valid thanks to the logical constraints (3.18). Thus, the inequality

(3.18) must not be omitted in the formulation of the attacker's problem. Further discussion of the CA constraints used in the formulation of BPPCF can be found in [20].

In constraints (3.20) we use the values of the binary decision variables U_{ij} as constants imported from the upper level solution. Therefore, constraints (3.20)—like (3.18)—are actually linear constraints. They state that a customer who has been assigned to the closest of the p facilities in the upper level problem should still stay with the same facility in the lower level problem unless it is lost due to an interdiction. In other words, if facility j is not lost ($S_j = 0$) and if customer i visited j before the attack ($U_{ij} = 1$), then he/she will go again to j after the attack ($V_{ij} = 1$). Finally, binary constraints on the decision variables S_j and V_{ij} are provided in (3.21) to conclude the lower level problem.

3.2. Problem Formulation of Bilevel *p*-Median Problem for Planning of Critical Facilities

The mathematical model for the case with no protection is similar to the model shown between (3.6)–(3.21) where the system planner has budget to use for protecting facilities. In this version of the problem, only the constraints involving the protection variable Y_j is excluded from the previous model since there is no usage of protection budget in this version of the problem. Accordingly, the constraints (3.11), (3.12) and (3.17) do not appear in this formulation as well as the binary constraints on Y_j. In BPCF, the system planner cannot reduce the post-attack cost by using protection resources but considers this cost while giving decisions about the facility locations. In other words, the system planner tries to minimize the maximum total cost before and after an attack with no protection. The optimal facilities that the attacker decides to hit for each network configuration are the most critical facilities for the corresponding configuration.

The index sets and parameters are the same as used before. The bilevel formulation of BPCF, where the system planner does not use any protection budget, is given as follows:

$$\min \; Z_{\text{def}} = \sum_{j \in \mathbf{J}} f_j X_j \; + \sum_{i \in \mathbf{I}} \sum_{j \in \mathbf{J}} c_{1j} q_i U_{ij} \; + \sum_{i \in \mathbf{I}} \sum_{j \in \mathbf{J}} c_{2j} q_i (1 - U_{ij}) V_{ij} \; + \sum_{i \in \mathbf{I}} \sum_{j \in \mathbf{J}} q_i d_{ij} V_{ij} \quad (3.22)$$

Subject to $\sum_{j \in \mathbf{J}} X_j = p$ (3.23)

$\sum_{j \in \mathbf{J}} U_{ij} = 1 \quad \forall i \in \mathbf{I}$ (3.24)

$\sum_{i \in \mathbf{I}} U_{ij} \le n X_j \quad \forall j \in \mathbf{J}$ (3.25)

$\sum_{k \in \mathcal{F}_{ij}} U_{ik} \ge X_j \quad \forall i \in \mathbf{I}, \forall j \in \mathbf{J}$ (3.26)

$U_{ij},\ X_j \in \{0,1\} \quad \forall i \in \mathbf{I},\ \forall j \in \mathbf{J}$ (3.27)

where

$\max\ Z_{\text{att}} = \sum_{i \in \mathbf{I}} \sum_{j \in \mathbf{J}} q_i d_{ij} V_{ij}$ (3.28)

Subject to $\sum_{j \in \mathbf{J}} V_{ij} = 1 \quad \forall i \in \mathbf{I}$ (3.29)

$\sum_{j \in \mathbf{J}} S_j \le r$ (3.30)

$S_j \le X_j \quad \forall j \in \mathbf{J}$ (3.31)

$\sum_{i \in \mathbf{I}} V_{ij} \le n X_j (1 - S_j) \quad \forall j \in \mathbf{J}$ (3.32)

$\sum_{k \notin \mathcal{F}_{ij}} V_{ik} \le 1 + S_j - X_j \quad \forall i \in \mathbf{I},\ \forall j \in \mathbf{J}$ (3.33)

$V_{ij} \ge U_{ij} (1 - S_j) \quad \forall j \in \mathbf{J}$ (3.34)

$S_j,\ V_{ij} \in \{0,1\} \quad \forall i \in \mathbf{I},\ \forall j \in \mathbf{J}.$ (3.35)

The above mathematical model is derived from the formulation of BPPCF by removing the constraints which involve the protection variable. For this case, the main linkage between the upper and the lower levels of the bilevel formulation is provided by the help of constraint (3.31) which indicates that an attacker cannot make an interdiction plan at site *j* unless a facility is opened there by the system planner. The system planner has the same cost function as he/she has in BPPCF formulation and aims to determine facility locations so as to minimize the total cost incurred before and after an attack without any protection.

4. A TWO-PHASE TABU SEARCH HEURISTIC METHOD

Given that BPPCF has integer variables in both the upper and the lower level problems, it is clear that we need efficient heuristic methods to obtain good solutions. To this end, we propose a two-phase heuristic solution method called TS-BST. This is a Tabu Search (TS) metaheuristic which includes a Binary Search Tree (BST) algorithm embedded into it. The first phase of TS-BST involves a search for the determination of the locations for p facilities using the TS. When the locations of the p facilities to be opened are fixed, the remaining problem is still of a bilevel nature. That is, the defender has to find out which open facilities should be protected, and given the protected and unprotected facilities the attacker has to determine which unprotected facilities must be destroyed. For this subproblem, we employ the second phase of our heuristic which is based on an observation made in Church and Scaparra [17]: the defender should protect at least one of the facilities that the attacker plans to interdict. This observation can easily be verified as follows: if none of the facilities interdicted by the attacker is protected, then the defender cannot avoid the worst-case scenario. Hence, the second phase of the proposed heuristic consists of the construction of a binary search tree that provides the facilities to be protected by the defender and to be interdicted by the attacker. We start with the explanation of our TS implementation and then give the details of the search tree used.

4.1. First Phase of the Heuristic: Tabu Search

TS is a metaheuristic algorithm that guides the local search to prevent it from being trapped in premature local optima or in cycling [45]. The principal distinction of TS from other local search algorithms is that it keeps track of not only the local information, i.e. the best objective value ever met, but also some information related to the exploration process. It achieves this by labeling the moves that lead to previously visited solutions as *tabu* and prohibiting them for a certain number of iterations named as *tabu tenure*. This organized memory utilization is an essential feature of TS. In TS, one starts with an initial solution, and at each iteration of the algorithm a finite *neighborhood* of the current solution is generated using a set of move operators. The best solution from this neighborhood is chosen as

the new current solution as long as the move creating this solution is not restricted as tabu, or when the best neighborhood solution outperforms the *incumbent* (the overall best solution so far) although the move creating it was declared tabu earlier. This condition is called *aspiration criterion*. The incumbent is updated if the new current solution is better than the incumbent. A *tabu list* which is updated at the end of each iteration keeps track of the tabu attributes of the accepted moves. The iterations are continued until one or more termination criteria have been satisfied. In the literature, there are a variety of implementations of TS for different kinds of problems thanks to its flexibility feature. Besides, TS has been applied successfully for the *p*-median problem and its extension in reverse logistics [46, 47, 48].

In our TS adaptation, we implement the use of 1-, 2-, and 3-Swap moves as move operators. According to these moves, one, two, and three facilities respectively are carried in the current solution from their locations to potential sites without a facility. To put it differently, existing facilities are closed at their sites, and reopened at different sites, i.e., they are relocated. Before we provide the details of the TS procedure, we give the notation used in it.

Table 4.1. Parameters in the model of BPPCF

Parameters	Definition
num_iter	number of iterations performed so far.
Max_Iter	maximum number of iterations.
num_nonimp_iter	number of iterations throughout which the incumbent does not improve.
Max_Nonimp_Iter	maximum number of iterations throughout which the incumbent does not improve.
num_neigh	number of neighbors generated in the current iteration.
size_neight	number of neighbors generated in the current iteration using the t^{th} move, $t = 1, 2, 3$.
Obj	objective value of a newly generated neighboring solution.
Obj_Best_Neigh	objective value of the best neighboring solution.
*Obj**	objective value of the incumbent.
tabu_tenure (τ)	the number of iterations during which the current solution will be tabu-active.

4.1.1. Initial Solution

In order to generate an initial solution for the TS heuristic, we consider the cost of travelling distance, protection cost and the fixed cost of opening for each facility site. Since the objective of the system planner includes all of these components, it is significant to consider these costs while setting up the initial configuration. Note that if facility sites have identical opening costs, it is useless to consider this cost while evaluating the facility sites. This situation holds for the protection cost, as well. We evaluate each facility site by summing up its cumulative distance summed over all customer zones for each potential site *j*, the fixed cost of opening a facility and the protection cost of the corresponding facility site. This value is calculated by Equation (4.1):

$$Val_j = \sum_{i \in \mathbf{I}} d_{ij} + f_j + b_j \quad \forall j \in \mathbf{J} \tag{4.1}$$

The initial solution is obtained by choosing the first p sites with the lowest Val_j value. This choice for the initial solution may be rational since fixed costs together with travelling distance costs constitute a reasonable part of the system planner's objective function.

4.1.2. Neighborhood Structure and Tabu Restrictions

At each iteration, TS generates a neighborhood of the current solution with the help of some move operators. In our problem we utilize 1-Swap, 2-Swap, and 3-Swap moves to generate new neighboring solutions from the current solution. This choice of the neighborhood structure prevents infeasible solutions in which the number of opened facilities is not equal to p from entering the search space. The working mechanism of the move operators can be explained as follows. First of all, let us denote the index set of sites in which p facilities are opened as $\mathcal{F}_1$ and the index set of $m–p$ sites that do not have a facility opened as $\mathcal{F}_2$. Clearly, $\mathcal{F}_1 \cup \mathcal{F}_2 = \mathbf{J}$ must hold true.

1-*swap move*: Randomly select two sites, site i from $\mathcal{F}_1$ and site j from $\mathcal{F}_2$. Swap these two facilities i and j between sets. This move does not violate any constraints of the problem. There are $p(m-p)$ ways to achieve this move operation.

2-*swap move*: Randomly select two different sites (i, j) from $\mathcal{F}_1$ and (k, l) from $\mathcal{F}_2$. Remove (i, j) from $\mathcal{F}_1$ and place (k, l) into this set. Update $\mathcal{F}_2$ by adding (i, j) to its elements. This move also does not violate any constraints of the problem. There are $\binom{p}{2}\binom{m-p}{2}$ ways to achieve this move operation.

3-*swap move*: Randomly select three different sites (i, j, k) from $\mathcal{F}_1$ and (l, m, n) from $\mathcal{F}_2$. Remove (i, j, k) from $\mathcal{F}_1$ and place (l, m, n) instead. Update $\mathcal{F}_2$ by adding (i, j, k) to its elements. This move also does not violate any constraints of the problem. There are $\binom{p}{3}\binom{m-p}{3}$ ways to achieve this move operation.

At this point, we have to point out that although the size of the neighborhood varies between operators due to the above calculations, we apply these three moves in such a way that an equal number of solutions are generated from each one. The best neighbor is selected by comparing the cost of the defender, i.e., Z_{def}.

As the current solution is updated throughout the iterations of TS-BST, we employ tabu restrictions so that solutions visited earlier are not selected repeatedly. Tabu restrictions are defined for the three moves as follows. In the 1-Swap move, if facilities at sites i and j are swapped, i.e., one is added to and the other is dropped from the set of opened facilities, then the 1-Swap move cannot be applied to them during the time they are *tabu-active*. The tabu-active status of any move depends on the tabu-tenure employed. In the case of the 2-Swap, suppose facilities at sites i and j are about to be opened, i.e., to be added to the current configuration, and two centers opened previously at sites k and l are about to be closed, i.e., to be dropped from the current configuration. Suppose further that this 2-Swap move does not result in a better cost than the incumbent's cost, which means that it does not satisfy the

aspiration criterion. Thus, it cannot be executed if centers at sites *i* and *j* had been closed, and those at sites *k* and *l* had been opened during the previous τ iterations with τ denoting the then-assigned tabu tenure. That is, once a particular move has been performed upon some specific facilities, it is declared as tabu-active for the next τ iterations, during which the outcome of the move should not be reverted. The tabu attributes of the 3-Swap move are defined in a similar way. According to the aspiration criterion, a move involving facility sites that are tabu can be executed provided that it produces a solution with a lower cost than the incumbent.

Following the generation of the initial solution, the initialization of the some parameter required for the TS algorithm is achieved. The neighborhood size and tabu tenure values are calculated depending on the parameters given to the solution algorithm. The neighborhood size for 1-Swap move is calculated as the total number of solutions that can be achieved by this move (it is shown above) divided by the parameter RATIO_NEIGHBORHOOD_SIZE. The calculation of the tabu tenure which is independent of the move operator but dependent on the *p* value is given as:

$$\text{Tabu tenure} = \text{MIN}\,(p * \text{RATIO\_MAX\_TABU\_TENURE}, \text{MAX\_TABU\_TENURE}) \quad (4.2)$$

We have to point out that, in this calculation RATIO_MAX_TABU_TENURE and MAX_TABU_TENURE are parameters given to the algorithm.

4.1.3. Termination Criterion

We use two termination criteria. The first one is the total number of iterations performed controlled by the parameter Max_Iter. The second criterion is the maximum permissible number of iterations during which the best solution (incumbent) does not improve. This criterion is controlled by the parameter Max_Nonimp_Iter.

4.2. Second Phase of the Heuristic: BST

For each alternative set of opened facilities, TS-BST calls BST to compute the objective function Z_{def} of the defender. While branching on a node in the binary tree, we utilize an

observation made in [17] which indicates that the defender should protect at least one of the facilities that the attacker plans to interdict in order to avoid attacker's current best plan. By the help of this rule, there is no need to consider all possible interdiction-protection schemes, i.e. explicit enumeration of all possibilities in order to find the optimal objective value of the defender, Z_{def}. If the protection variable Y_j is set to value one on a branch leading to a node in the search tree, an optimization problem is solved at that node from the attacker's perspective in order to determine the facilities that are to be interdicted with the aim of maximizing the total demand-weighted traveling distance when the set of the protected facilities is fixed. This amounts to solving the lower level problem of BPPCF defined by expressions (3.14)–(3.21) with fixed X_j, U_{ij}, and Y_j variables. Note that in this subproblem, which is referred to as the AP in the sequel, X_j are fixed because the set of opened facilities is provided by TS, and U_{ij} are fixed because the values of these variables are obtained by assigning each customer to its nearest opened facility. The Y_j variables also become known due to the values set to these variables on the branches along the path which begins at the root node and terminates at the current node. When the AP is solved at a node, the facilities to be interdicted by the attacker, i.e., the optimal values of the S_j variables are obtained. Hence, it becomes possible to compute both the objective function Z_{def} of the defender's problem and Z_{att} of the attacker's problem defined at the node. As soon as all the nodes of the search tree are explored, the node with the minimum Z_{def} value provides the best solution attainable from the defender's viewpoint for a given set of p opened facilities. Let us elaborate on the construction of the binary search tree.

The information of the protection status of the opened facilities is kept in an array. The array, which is referred to as protection array, consists of p zero-one upper-level decision variables that are generated by fixing the protection variable Y_j to one or zero. In the root node of the tree the attacker's problem (AP) represented by (3.14)–(3.21) is solved by setting all Y_j variables equal to zero. This means that all the entries in the protection array is zero. This solution gives the optimal set of r facilities to be interdicted by the attacker when all facilities are open to attack. Let us denote the set consisting of the interdicted facilities at any node of the tree by $\Re$. It is important to mention that the cardinality of this set is equal to r at the root node whereas it is less than or equal to r at other nodes depending on the number of protected facilities. $\Re$ indicates the candidate facilities to be protected by the defender.

At the root node, any facility $j \in \Re$ can be chosen arbitrarily since none of them is protected so far and accordingly no protection resource is consumed. However, at any other node in the tree, the protection status of a facility j and the availability of the protection budget to protect $j \in \Re$ should be checked while selecting a facility to branch on. Any eligible facility can be chosen randomly. Following the selection step, two child nodes are created in the tree: in the first one, $Y_j = 0$ indicating that facility j is unprotected, and in the other one $Y_j = 1$ implying that facility j is protected. It is a binary tree, because at each step two child nodes are created from a parent node. For each child node, the parent node passes the information of the protection status of the selected facility j, which is the protection array mentioned above. According to this information, in the node with the Y_j value is set to one, let us call it as *node*1, the AP is solved with fixed values of X_j, U_{ij}. Since the only difference between the AP of a child node and that of its parent node is due to the protection array, the AP in a child node does not need to be solved from scratch. The associated AP can be modified just by changing the right-hand-side values of the constraints (3.17) according to the protection array. It yields a new set $\Re$ of interdicted facilities that is optimal from the attacker's point of view. After the generation process of the child node is completed by calculating the objective value of the defender according to (3.6), we check whether the node is leaf or active. For *node*1, the consumption in protection resources is considered in the evaluation process. If there is at least one facility whose protection cost does not exceed the remaining budget, then the node is called to be an active node. Otherwise, the total budget will be exceeded when any of the facilities in set $\Re$ is protected which means $\Re$ does not include any candidate facility to be branched on. Consequently, this node becomes a leaf node of the tree.

The process for the second child node with the Y_j value is set to zero eventuates as follows. Let us call this node as *node*2. For *node*2, there is no need to solve an AP since none of the facilities in the set $\Re$ of the parent node would be protected, which means that the protection array does not change. However, the facility j can no longer be a candidate for branching since Y_j is fixed to zero. Therefore, we update $\Re$ by just removing facility j from it. In the checking process for *node*2, there can be two cases: if $\Re$ becomes empty after removing facility j, *node*2 is said to be a leaf node. Otherwise, the set still includes at least one facility to be chosen which makes *node*2 to be an active node. The search is terminated

when all the nodes of the binary tree are explored. The node with the lowest objective value Z_{def} of the defender is stated as the optimal solution for the facility configuration given at that iteration. The optimal set of protected facilities is obtained by backtracking the path from that node to the root node. The objective value Z_{def} of the defender's problem given in (3.6) is computed by using the optimal values of the decision variables V_{ij} and S_j obtained from the AP, the optimal values of the protection variables Y_j as well as the values of assignment variables U_{ij} and location variables X_j corresponding to the set of opened facilities.

We have to point out some strategies about the implementation of the method. First, the search tree is constructed according to the depth-first search rule. The order of branching variables is not significant since all possible protections will have been evaluated by the time all nodes are fathomed. The depth of the tree depends on the protection budget b_{tot} and the protection costs b_j of the facilities. The size of the search tree is independent of the number of facilities to be opened, i.e., p. However, p has an impact on the size of the AP to be solved at each node of the tree, which in turn affects the required computation time. Recall that this problem is a mixed-integer linear programming problem consisting of $(p + np)$ variables, namely S_j and V_{ij}. We should also mention that the computational effort is reduced in solving the APs in tree by utilizing the feature of CPLEX callable library that changes the right-hand-side (rhs) of the existing problem, which prevents solving the problems from scratch. This leads to a reduction in the computational cost and memory usage. Finally, recall that the bilevel defender-attacker problem is solved to optimality by using the BST algorithm subject to the facility configuration obtained by the move operation in TS.

Let us illustrate the working procedure of the BST by a toy example with $p = 5$ and $r = 2$. Namely, at most two facilities can be interdicted by the attacker. The total protection budget of the defender is equal to 10 monetary units ($b_{tot} = 10$). The protection costs of the facilities are given as $b_1 = 5$, $b_2 = 4$, $b_3 = 4$, $b_4 = 3$, and $b_5 = 3$. The search tree corresponding to this example is depicted in Figure. The notation used in the example is shown in Table 4.2.

Table 4.2. Parameters and variables in the example

Parameters	Definition
$\Re$	The candidate facilities for protection at a node
Y_j	Protection variable (1 if facility j is protected, 0 otherwise)
Z_{def}	Defender's objective value
Z_{att}	Attacker's objective value
b_{rem}	Remaining protection budget at the current node

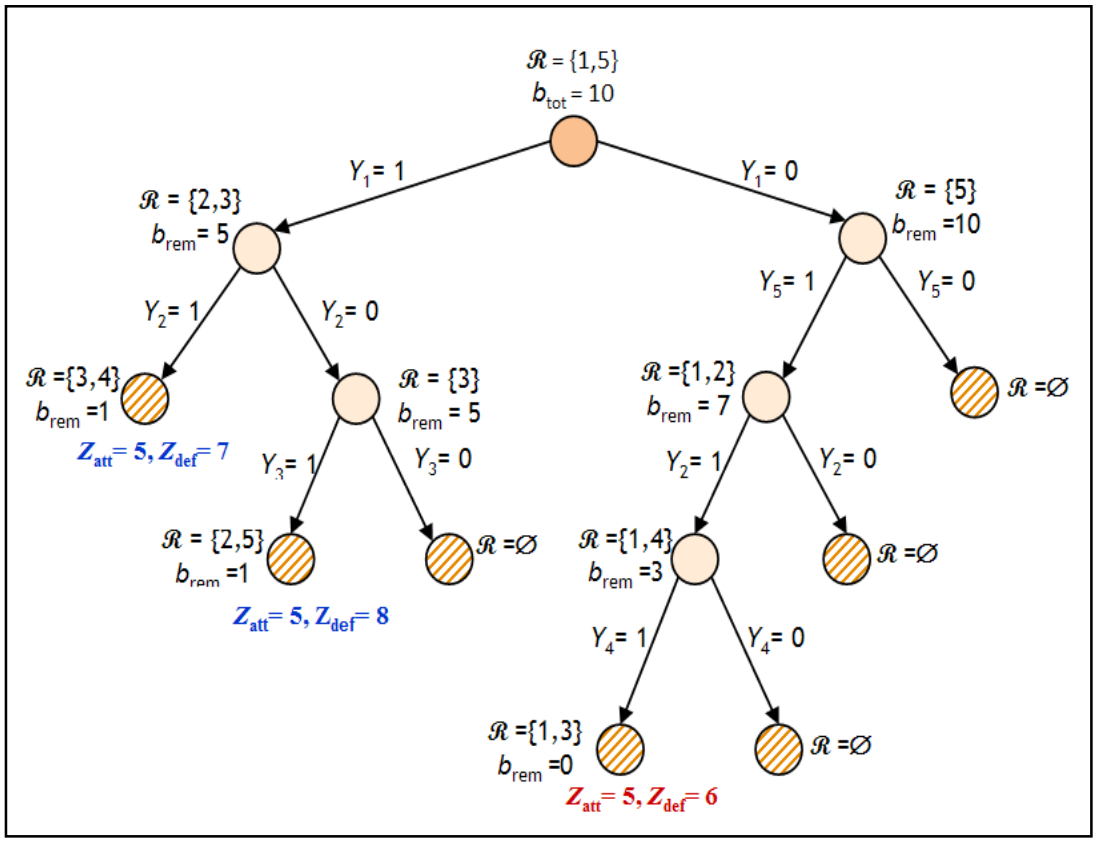


Figure 4.1. Binary search tree (BST) for a toy problem

At the root node which is shown as dark-shaded, the attacker's problem AP is solved when none of the opened facilities are protected by the defender. The facilities to be inter-

dicted by the attacker to give the maximum disruption are found to be $\Re = \{1,5\}$. The defender has 10 monetary units of protection budget initially, which lets him/her to protect either facility 1 or facility 5. Facility 1 is arbitrarily chosen for protection and branched on so that the remaining budget becomes five units ($b_{\text{rem}} = 5$). At the child node with $Y_1 = 1$, the AP is solved again with the information that facility 1 cannot be interdicted. This gives rise to the optimal interdiction plan $\Re = \{2,3\}$. The other child node obtained from the root node corresponds to the branch $Y_1 = 0$, i.e. facility 1 is not protected, which only requires updating $\Re$ by just removing facility 1 from it. Since both nodes have enough budgets for the protection of other facilities, branching is continued. Two new branches, which correspond to the cases where facility 2 is protected and not protected, are generated from the node on the left-hand side. In the former case, the AP is solved by setting $Y_1 = Y_2 = 1$ and the optimal set of interdicted facilities turn out to be $\Re = \{3,4\}$. Since the protection cost is four units, a budget of one unit remains which is not sufficient to protect any of the candidate facilities in $\Re$. Therefore, no branching can be done and this node is said to be a leaf node. At this node, the objective values of the defender and the attacker are given hypothetically as $Z_{\text{att}} = 5$, $Z_{\text{def}} = 7$. Another child node is generated when Y_1 is set to the value zero. At this node, $\Re = \{5\}$ and it is possible to generate two nodes by branching as $Y_5 = 1$ and $Y_5 = 0$. The branch with $Y_5 = 0$, which is shown as a shaded circle, leads to a node in which the set of candidate facilities becomes empty. Since no branching is accessible from there on, the node is also a leaf node. Branching process is continued until either the remaining budget is not sufficient for the protection of another facility or the set $\Re$ of facilities to be interdicted becomes empty. The optimal solution of the toy example is the leaf node with the lowest objective value Z_{def} of the defender which equals to five and the interdicted facilities are 1 and 3. By backtracking from that node to the root node, the optimal protection scheme can be found, which in this example involves the facilities 2, 4, and 5.

4.3. ESV: An Exhaustive Search and Solution Validation Method

Along with the BST algorithm embedded in TS for the BPPCF, we develop an exhaustive search and solution validation method called ESV. ESV guarantees to find an optimal solution to any given BPPCF instance since it evaluates all protection-interdiction schemes

for all possible facility configurations. The time complexity of ESV is obviously exponential since it explores in an outer loop each of the $\binom{m}{p}$ possible site combinations to open p facilities, while it may check as many as $\binom{p}{r}$ facilities in the innermost loop depending on the value of the protection budget b_{tot}.

We observe that ESV can solve instances with p ≤ 5 in reasonable time limits. Hence, it is not used for instances with six or more facilities to be opened. The pseudo-code of ESV given in Figure 4.2 explains the way an optimal solution is obtained to a given BPPCF instance. Z^*_{def} and Z^*_{att} stand for the optimal objective values of the defender and the attacker, respectively. $\left[U^*_{ij}\right]$ and $\left[V^*_{ij}\right]$ matrices indicate the optimal closest customer–facility assignments before and after the interdictions by the attacker, respectively. σp, σb, and σr are the subsets of opened, protected, and interdicted candidate facility sites, respectively, where |σp| = p, |σr| ≤ r, σb ⊆ σp, σr ⊆ σp, and σb ∩ σr = ∅. $\boldsymbol{\sigma}^*_r$ corresponds to the subset of facilities the interdiction of which maximizes the attacker's objective function Z_{att} for the given facility opening and protection plan (σp,σb).

Let $Z^*_{def} := \infty$.
For each unique subset $\boldsymbol{\sigma}_p \subset \mathbf{J}$ do
{
  Obtain $\left[U^*_{ij}\right]$ using the facilities in $\boldsymbol{\sigma}_p$.
  If $\sum_{j \in \boldsymbol{\sigma}_p} b_j \le b_{tot}$
  {
    Protect all p facilities in $\boldsymbol{\sigma}_p$, i.e., set $\boldsymbol{\sigma}_b := \boldsymbol{\sigma}_p$.
    Set $\boldsymbol{\sigma}_r := \varnothing$. **`/* Interdict none.*/`**
    Set $\left[V^*_{ij}\right] := \left[U^*_{ij}\right]$.
    Calculate Z_{def} and Z_{att} according to Eqs. (3.6) and (3.14)
    If $Z_{def} < Z^*_{def}$
      Set $Z^*_{def} := Z_{def}$ and record the current solution ($\boldsymbol{\sigma}_p$, $\boldsymbol{\sigma}_b$, ∅) as the best solution.
  } **`/*endIf*/`**
  ElseIf $b_j > b_{tot} \quad \forall j \in \boldsymbol{\sigma}_p$
  {
    Set $\boldsymbol{\sigma}_b = \varnothing$. **`/* Protect none.*/`**

Set $Z_{\text{att}}^{*} := 0$.
For each unique subset $\boldsymbol{\sigma}_r \subseteq \boldsymbol{\sigma}_p$ do
{
Obtain $\left[V_{ij}^{*}\right]$ using the facilities in $\boldsymbol{\sigma}_p$.
Calculate Z_{att} according to Eq. (3.14).
If $Z_{\text{att}} > Z_{\text{att}}^{*}$
Set $Z_{\text{att}}^{*} := Z_{\text{att}}$ and $\boldsymbol{\sigma}_r^{*} := \boldsymbol{\sigma}_r$.
} **/*endFor*/**
Calculate Z_{def} according to Eq. (3.6).
If $Z_{\text{def}} < Z_{\text{def}}^{*}$
Set $Z_{\text{def}}^{*} := Z_{\text{def}}$ and record the current solution $(\boldsymbol{\sigma}_p, \varnothing, \boldsymbol{\sigma}_r^{*})$ as the best solution.
} **/*endElseIf*/**
Else
{
For each subset $\boldsymbol{\sigma}_b \subseteq \boldsymbol{\sigma}_p$ satisfying $\sum_{j \in \boldsymbol{\sigma}_b} b_j \le b_{\text{tot}}$ do
{
Let $r :=$ MINIMUM$\{r,\ p - |\boldsymbol{\sigma}_b|\}$.
Set $Z_{\text{att}}^{*} := 0$.
For each unique subset $\boldsymbol{\sigma}_r \subseteq \boldsymbol{\sigma}_p \backslash \boldsymbol{\sigma}_b$ such that $|\boldsymbol{\sigma}_r| = r$ do
{
Obtain $\left[V_{ij}^{*}\right]$ using the facilities in $\boldsymbol{\sigma}_p \backslash \boldsymbol{\sigma}_r$.
Calculate Z_{att} according to Eq. (3.14).
If $Z_{\text{att}} > Z_{\text{att}}^{*}$
Set $Z_{\text{att}}^{*} := Z_{\text{att}}$ and $\boldsymbol{\sigma}_r^{*} := \boldsymbol{\sigma}_r$.
} **/*endFor*/**
Calculate Z_{def} according to Eq. (3.6).
If $Z_{\text{def}} < Z_{\text{def}}^{*}$
Set $Z_{\text{def}}^{*} := Z_{\text{def}}$ and record the current solution $(\boldsymbol{\sigma}_p, \boldsymbol{\sigma}_b, \boldsymbol{\sigma}_r^{*})$ as the best solution.
} **/*endFor*/**
} **/*endElse*/**
} **/*endFor*/**
Return the best solution together with the objective values Z_{def}^{*} and Z_{att}^{*}.

Figure 4.2. Pseudo-code of ESV

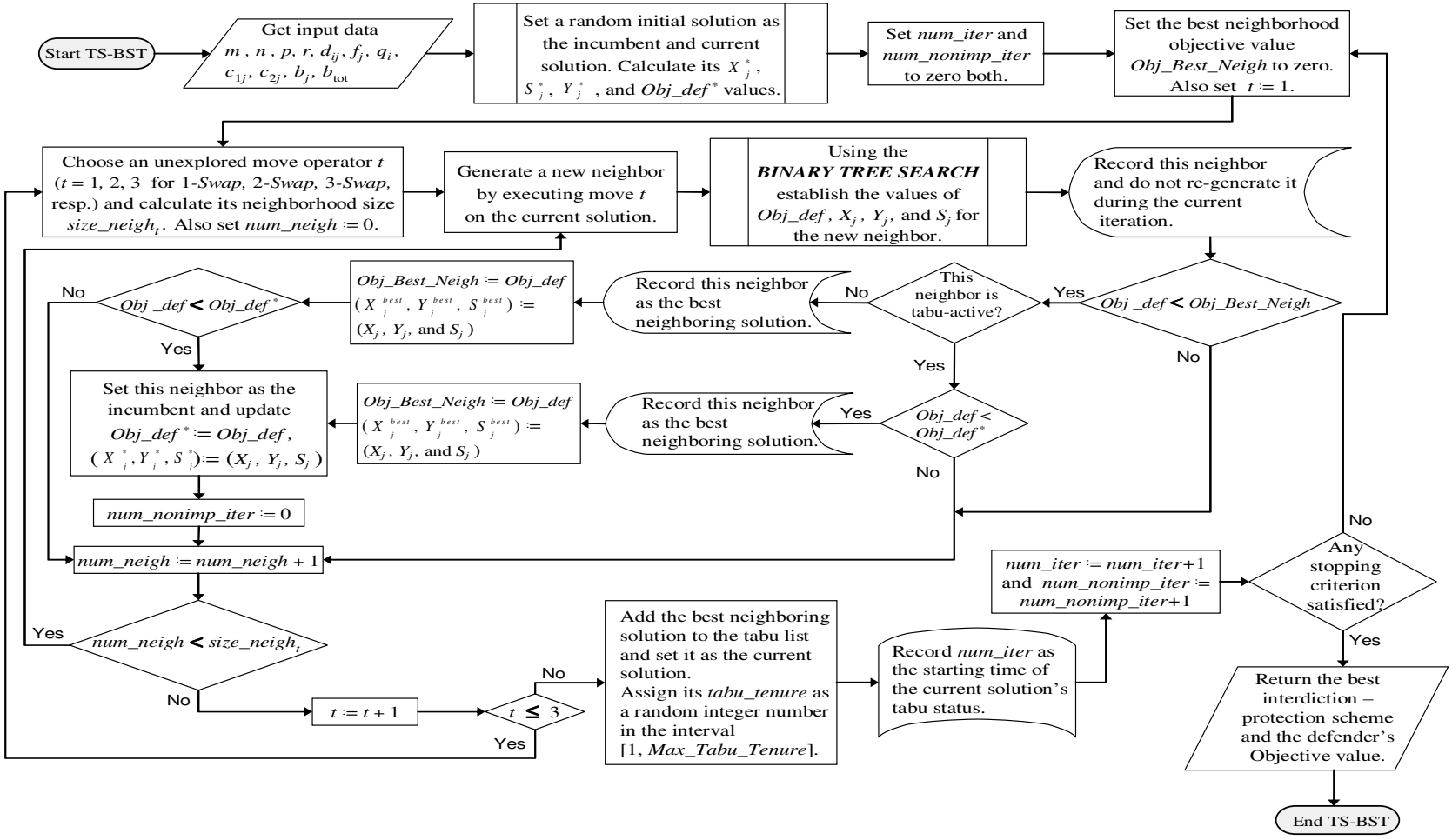


Figure 4.3. The flowchart of TS-BST for solving the BPPCF

5. A SEQUENTIAL SOLUTION METHOD

SSM is applied for solving BPPCF when the system planner makes his/her decision about facility location and protection planning in sequence. As discussed before, the system planner can give these decisions either sequentially or simultaneously. We propose TS-BST which is explained in Section 4 as a solution method for simultaneous case. Whereas SSM is proposed as a solution method for the sequential case.

SSM involves two stages in order to tackle with the sequential decisions of the system planner. In each stage different sub-problems are solved. In the first stage, the system planner wants to give location decisions in order to find the optimal facility location scheme. This is achieved by solving the well-known *p*-median problem to optimality. Once the optimal locations of the facilities are determined with respect to the sum of demand weighted travelling cost, capacity acquisition cost and fixed facility opening cost, the problem is solved by constructing a binary search tree in order to make the protection plan according to this configuration. The solution procedure of the each stage is explained below.

5.1. First Stage: *p*-median Problem

The *p*-median problem is one of the most important problems in discrete location theory which was originally designed for and widely used in facility location [49]. In a *p*-median problem, either in capacitated or uncapacitated version, the aim is to decide the locations of *p* facilities and allocate the demand points to one or more facilities. There is a huge amount of study about *p*-median problem in the literature. A study about the solution methods and annotated bibliography for *p*-median problem is provided can be found in [50]. In this sub-problem, the capacitated version of the *p*-median problem is modeled and solved. The problem is written in C and compiled with Microsoft Visual Studio 2005 using Cplex Callable Library 11.0. By using the parameter setting features of the Callable Library, it is ensured that the problem is solved to optimality.

5.2. Second Stage: The Defender - Attacker Problem

After determining the optimal facility locations by solving the *p*-median problem in the first stage, the BST algorithm is employed in order to solve the bilevel defender-attacker problem. There is only a trivial distinction while calculating the objective of the system planner in the BST algorithm. That is, the objective of the defender does not include the costs calculated in the first stage *p*-median problem, which are fixed cost of opening facilities, the cost of capacity acquisition and the cost of total demand weighted distance. This has no effect on the working mechanism of BST. Yet, these costs are included to the system planner's objective when the exploration in the search tree ends.

6. COMPUTATIONAL RESULTS

6.1. Random Problem Generation

In order to test the performance of the proposed solution methods, we randomly generate a total of 225 instances. They vary in the number of candidate facility sites (m), the number of facilities to be opened (p), the number of facilities interdicted by the attacker (r), and the level of the protection budget (b_{tot}). The number of candidate facilities (m) ranges from 10 to 50. On the other hand, the number of customers (n) depends on m and is set to $10m$ in each instance. The number of facilities to be opened (p) takes on five different values ranging between 3 and 8. For each (m, n) pair, five problem instances are created depending on p value. We do not consider an equal number of interdicted facilities (r) for each (m, n, p) instances in view of the fact that the sum of interdicted and protected facilities cannot exceed p. Therefore, when $p = \{3, 4, 5\}$, r takes on the value of 1 or 2. Alternatively, when $p = \{6, 7, 8\}$, r can be equal to 1, 2 or 3. Finally, for the low budget problems the budget amount is set to 1000 monetary units, while high budget problems have different budget amounts according to p values.

The template employed for the generation of random problem instances is described in Table 6.1. Let us explain some symbols and parameters appeared on this table. The symbol "$[\![\cdot]\!]$" signifies the round-off operation to the nearest integer, while $U(0,1)$ stands for a uniform random number between 0 and 1 and $U[lb,ub]$ symbolizes a random integer number between a lower bound lb and an upper bound ub. The coordinates of customer location i and facility site j are denoted by (cx_i, cy_i) and (fx_j, fy_j), respectively. The "*distance*(i, j)" function returns the Euclidean distance between a customer location i and a facility site j. We remark that in all random problem instances the unit capacity expansion cost (c_{2j}) is assumed to be equal to the unit capacity acquisition cost (c_{1j}) without loss of generality. As the template also reveals, we generated 75 random instances for two different budget levels, namely a low and a high protection budget, and also 75 more instances for the option of no protection yielding 225 instances in total.

We should also give details about the way customer and facility locations are generated. Customer locations are uniformly distributed over a circular area centered at the origin (0,0) with the radius R equal to 1000. Candidate facility sites, on the other hand, are uniformly dispersed on as many as (m+1) equidistant horizontal and vertical lines which hypothetically dice a square centered at the origin (0,0) with a side length of 1500. The coordinates of both customer locations and facility sites are rounded to the nearest integer. Five different m values are used in the generation of the test problems. Figure A.1 and Figure A.2 in Appendix A visualize customer locations and candidate facility sites on the xy-plane for m = 10 and m = 50, respectively. In addition to all these, it is important to mention that there exists no customer node with more than one closest facility before an interdiction in the data test problems. This condition is tested and verified before using the generated test problems in algorithmic study phase.

Table 6.1. Random problem generation template employed in the computational study

Parameters	Values
m	{10, 20, 30, 40, 50}
n	$10m$
p	{3, 4, 5, 6, 7, 8}
r	{1, 2} when $p \in \{3, 4, 5\}$; {1, 2, 3} when $p \in \{6, 7, 8\}$
(R, L)	(1000, 1500)
(cx_i, cy_i)	Let $R_i = R \times U(0,1)$, $\theta_i = 2\pi \times U(0,1)$. Then, $cx_i = [\![R_i \cos\theta_i]\!]$ and $cy_i = [\![R_i \sin\theta_i]\!]$.
(fx_j, fy_j)	$fx_j = -0.5L + \frac{L}{m} \times U[0, m]$ and $fy_j = -0.5L + \frac{L}{m} \times U[0, m]$
q_i	$10 + 5 \times U[0, 18]$
f_j	$10000 + 1250 \times U[0, 8]$
b_j	$500 + 25 \times U[0, 20]$
$c_{1j} = c_{2j}$	$10 + 2.5 \times U[0, 4]$
d_{ij}	$0.01 \times distance(i, j)$
$(b_{tot})_{low}$	1000
$(b_{tot})_{high}$	2000 if $p \in \{3, 4\}$; 2500 if $p \in \{5, 6, 7, 8\}$. Average $(b_{tot})_{high}$ is then equal to 2367.

6.2. Computational Environment and Design of the Experiments

All codes used for developing all three of the proposed algorithms have been written in ANSI C language and compiled with Microsoft Visual Studio 2005 using CPLEX Callable Library 11.0. The runs are performed on a server equipped with two Intel Xeon X5460 3.16 GHz Quad-Core processors and 16 GB RAM. Each core on this computing platform's processors attains a speed of approximately 4800 MIPS (million instructions per second).

We split the test bed into three parts according to the budget levels, i.e. low budget, high budget and zero budget. The latter corresponds to the case of no protection. First of all, we perform some experiments in order to fine-tune some of the parameters used in TS method. We assess the effect of seven different parameter configurations on the algorithm and report the performance of them. After determining the appropriate parameter values, we conduct the experiments for the BPPCF problem instances. Following that, ESV tests are carried out in order to evaluate the performance of the TS-BST algorithm. Since the computational effort of ESV increases exponentially in p, we restrict the experiments for the problem instances up to five facilities to be opened, i.e. $p = 5$. Finally, we finish our experimental study by testing the SSM algorithm using the same test bed.

6.3. Results for BPPCF Problem

This section reports the computational results obtained by systematically testing the solution algorithm TS-BST proposed for the BPPCF problem. The computational study consists of four stages: random problem generation, fine-tuning the TS parameters, testing TS-BST with two TS parameter configurations, and investigating the effect of the problem parameters m, p and r on the solution times. We also compare the performance of the TS-BST algorithm with the results of ESV and SSM algorithms. The comparison results are reported in the following sections.

6.3.1. Fine Tuning the TS Parameters

In this section, we give the details about how we achieve setting and fine tuning of the TS-BST algorithm parameters. While doing this, we received guidance from the previous papers of Rolland et al. [46] and Aras and Aksen [47] where the *p*-median problem is solved using Tabu Search. In this problem the tabu tenure τ, the most important attribute of the TS algorithm, is set to random integer values in the interval [1, *Max_Tabu_Tenure*] with *Max_Tabu_Tenure*=⌈1.5*p*⌉. On the other hand, the first stopping condition of TS-BST, namely the maximum number of TS iterations, is defined by setting *Max_Iter* = max{2*m*, 150}.

Other TS parameter to be set is the neighborhood size for each move operator. Notice that the parameters maximum number of TS iterations and neighborhood size are the most important parameters since they have an immediate impact on the solution quality of the problem. Consequently, we aimed to reflect the trade-off between the CPU time and the objective value, which are two major criteria determining the solution quality. To this end, we selected a pilot set of 12 test problems equally divided between low and high budget instances. A fine-tuning analysis was made on this pilot set by changing the values of the TS parameters *Max_Nonimp_Iter* and $size\_neigh_t$ where $t \in \{1 = 1\text{-Swap}, 2 = 2\text{-Swap}, 3 = 3\text{-Swap}\}$. The higher *Max_Nonimp_Iter*, the longer it takes to break off the loop of the nonimproving iterations. The larger $size\_neigh_t$ values, the bigger the solution space explored before moving from the current solution to the next one. As the values of these two parameters increase, longer CPU times are expected to be spent in anticipation of better objective values.

Table 6.2. Seven configurations for the fine-tuning test

	Config-1	Config-2	Config-3	Config-4	Config-5	Config-6	Config-7
Max_Nonimp_Iter	30	15	30	15	30	15	50
RNS	5	5	7	7	9	9	9

Test instances in the pilot set have the following specifications: $m = 50$, $p \in \{3,4,5\}$, and $r \in \{1,2\}$. Since *Max_Iter* = max{2*m*, 150}, the maximum number of iterations performed before TS-BST is terminated equals 150 for each test instance. Seven configura-

tions as shown in Table 6.2 were generated and tested on the pilot set. *Max_Nonimp_Iter* changes between 15, 30, and 50 iterations. The second parameter $size\_neigh_t$ is fine-tuned indirectly by the auxiliary parameter *RNS* (*Ratio_Neighborhood_Size*). It affects the values of $size\_neigh_t$ according to the following expressions where the ceiling operator $\lceil arg \rceil$ returns the smallest integer greater than or equal to *arg*. Total number of solutions visited is determined by the summation of $size\_neigh_t$ over *t*, where $t \in \{1, 2, 3\}$.

$$size\_neigh_1 = \lceil p \times (m-p)/RNS \rceil .$$

$$size\_neigh_2 = \min\left\{ \left\lceil \binom{p}{2} \times \binom{m-p}{2} \Big/ RNS \right\rceil, size\_neigh_1 \right\}.$$

$$size\_neigh_3 = \min\left\{ \left\lceil \binom{p}{3} \times \binom{m-p}{3} \Big/ RNS \right\rceil, size\_neigh_1 \right\}.$$

Table 6.3. Average fine-tuning results of instances with low budget of the pilot test set

	Average Test Results for						
Test Criterion	Config-1	Config-2	Config-3	Config-4	Config-5	Config-6	Config-7
Z^*_{def} (m.u.)	**493415**	495178	**493415**	497426	495527	498306	495527
Dev. From Z^{opt}_{def}	**0.00%**	0.36%	**0.00%**	0.81%	0.43%	0.99%	0.43%
Z^*_{att} (m.u.)	133203	135672	133203	132651	129146	127858	129146
CPU time (s)	4032.8	2426.9	3123.1	1942.0	2114.1	**1163.8**	2815.0
K_{iter}	53	28	57	34	56	30	76
K_{sols}	5912	3159	4785	2858	3592	1907	4832
K_{Cplex}	15050	8277	11896	7153	8668	4664	11768

We obtained average fine-tuning results for each configuration with instances that has low budget, no budget and high protection budget as shown in Table 6.3, Table 6.4 and Table 6.6, respectively. Z^*_{def} and Z^*_{att} values are given in monetary units (m.u.). In addition to seven TS configurations, test instances in the pilot set were also solved with the ESV code to optimality which led to the following average results.

Low bud. instances: $Z^{opt}_{\text{def}} = 493415$, $Z^{opt}_{\text{att}} = 133203$, $(\text{CPU time})_{\text{ESV}} = 1297.3$ seconds.

High bud. instances: Z_{def}^{opt} = 450390, Z_{att}^{opt} = 108196, $(\text{CPU time})_{\text{ESV}}$ = 3387.6 seconds.

Table 6.4. Average fine-tuning results of instances zero budget of the pilot test set

Test Criterion	*Average Test Results for*						
	Config-1	Config-2	Config-3	Config-4	Config-5	Config-6	Config-7
Z_{def}^{*} (m.u.)	558446	562493	557170	561062	559715	567756	**556219**
Dev. From Z_{def}^{opt}	0.65%	1.38%	0.42%	1.12%	0.88%	2.33%	**0.25%**
Z_{att}^{*} (m.u.)	137390	142374	140422	135912	138842	139493	140135
CPU time (s)	4269.5	2220.7	4503.0	1870.3	2179.2	**1204.7**	2847.9
K_{iter}	56	28	72	29	57	27	82
K_{sols}	6401	3210	6127	2326	3625	1739	5129
K_{Cplex}	6400	3209	6126	2325	3624	1738	5128

The first column in all of the three tables contains the names of seven criteria. If a particular configuration achieves the best average Z_{def}^{*} (i.e., the smallest deviation from the average Z_{def}^{opt} value) or the shortest average CPU time result out of all seven configurations, the corresponding value is boldfaced. The denotation of the last three row titles is shown in Table 6.5.

Table 6.5. Definition of the parameters used in tests

Parameters	Definition
K_{iter}	The average number of TS-BST iterations performed before any termination criterion is fulfilled.
K_{sols}	The average number of neighborhood solutions explored by the TS mechanism.
K_{Cplex}	The average number of Cplex 11.0 calls during the entire TS-BST iterations.

Table 6.6. Average fine-tuning results of instances with high budget of the pilot test set

	Average Test Results for						
Test Criterion	Config-1	Config-2	Config-3	Config-4	Config-5	Config-6	Config-7
Z^{*}_{def} (m.u.)	**450450**	450527	450492	450552	450837	451841	450492
Dev. from Z^{opt}_{def}	**0.01%**	0.03%	0.02%	0.04%	0.10%	0.32%	0.02%
Z^{*}_{att} (m.u.)	109631	109322	108196	109631	108753	106481	108196
CPU time (s)	3862.3	2238.8	2431.6	1393.8	2331.7	**1264.1**	3862.7
K_{iter}	55	26	57	29	60	28	93
K_{sols}	6092	2927	4365	2246	3761	1758	5778
K_{Cplex}	34556	18264	23165	12479	20190	10143	31080

Note that each time a new protection plan is implied by some node of the binary search tree, one has to solve the associated problem of the attacker (AP) to optimality, which is actually a MIP problem. Clearly, the last two rows in Table 6.3, Table 6.4 and Table 6.6 are positively correlated. There appears to be a positive and fairly linear correlation between the average solution times and K_{Cplex}. The effect of keeping *Max_Nonimp_Iter* the same while increasing *RNS* from 5 to 9, thus reducing the neighborhood sizes is observed in Config-1, 3, and 5 as shortened CPU times. A similar CPU time effect is produced in Config-7, 5, and 6 by keeping *RNS* the same while decreasing *Max_Nonimp_Iter* from 50 to 15. In that case, K_{iter} drops from 93 to 28 as well.

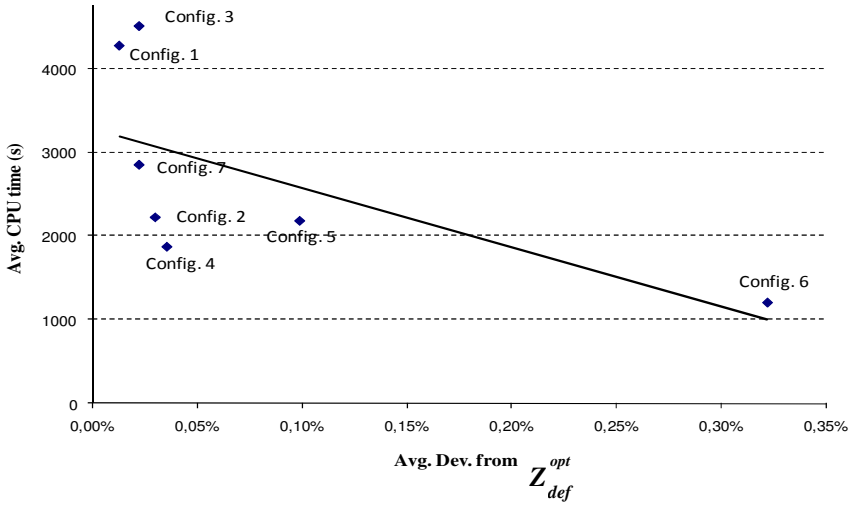


Figure 6.1. Trade-off between the solution criteria for the instances zero budget of the pilot test set

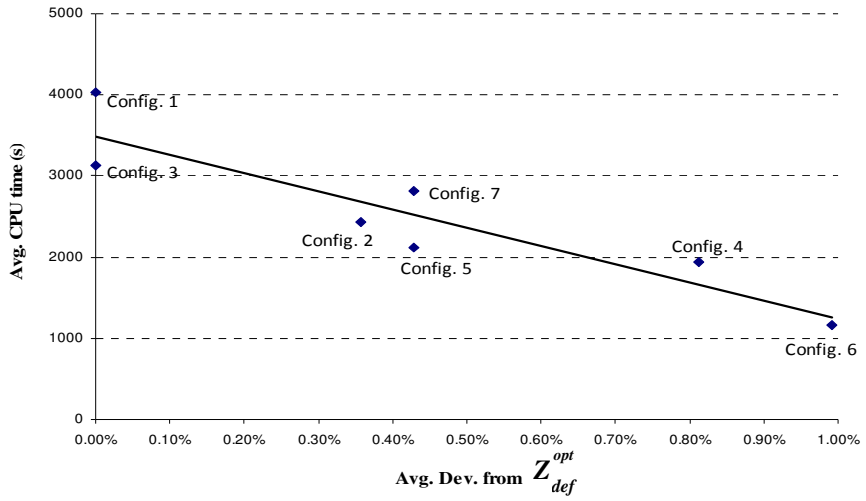


Figure 6.2. Trade-off between the solution criteria for the instances with low budget of the pilot test set

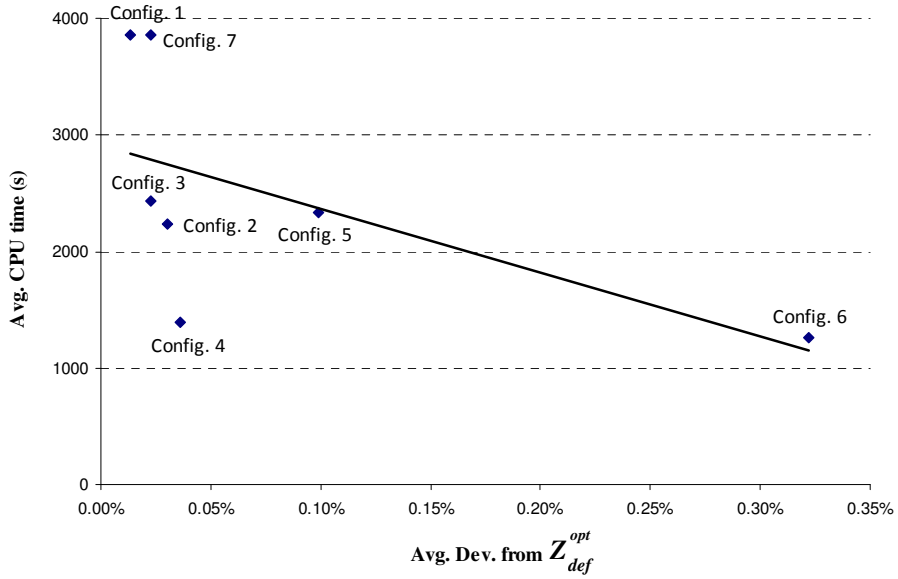


Figure 6.3. Trade-off between the solution criteria for the instances with high budget of the pilot test set.

Average deviations from Z_{def}^{opt} and average solution times for the seven configurations are plotted as a scatter diagram in Figure 6.1, Figure 6.2 and Figure 6.3 for zero budget, low budget and high budget instances, respectively. The diagrams reveal a trade-off be-

tween these two criteria. Config-6 stands out with the best solution times along with the worst deviations from the optimal solutions. Best Z^*_{def} values were obtained with Config-1. Yet, we chose to use Config-3 in our tests with BPPCF problem instances, since Config-3 achieved much shorter solution times than Config-1 without compromising the average deviation from Z^{opt}_{def}.

6.3.2. Computational Results of the Generated Problems

In this stage of the computational study, we tested the proposed solution algorithm TS-BST with Config-3 and Config-6. Since there has been no similar problem addressed in the literature before, we are not able to find benchmark problems. In order to evaluate the results of the algorithms proposed, we randomly generate problem instances as mentioned in the previous sections and use ESV algorithm to find the optimal results for a bunch of limited-sized problems. In 90 out of 225 instances of the entire test bed, p value is less than or equal to five. We optimally solved those 90 instances using ESV, thereby made it possible to assess the Z^*_{def} values obtained with TS-BST. In Appendix B we provide the detailed results of ESV (TS-BST) outputs including the opened, protected and interdicted facility indices divided into Table B.1 (Table B.4), Table B.2 (Table B.4), and Table B.3 (Table B.5) for zero budget, low budget and high budget instances, respectively. Problem names are indicative of the *m*, *p*, and *r* values. Column titles $\boldsymbol{\sigma}^*_p$, $\boldsymbol{\sigma}^*_b$, and $\boldsymbol{\sigma}^*_r$ stand for the subsets of opened, protected, and interdicted facility sites in the best TS-BST solutions, while $\boldsymbol{\sigma}^{opt}_p$, $\boldsymbol{\sigma}^{opt}_b$, and $\boldsymbol{\sigma}^{opt}_r$ stand for those in the optimal ESV solutions.

We make an interesting observation in Table B.5 for the following low budget instances: m10-p7-r2, m10-p8-r2, and m20-p8-r1. In these three instances, none of the opened facilities is protected although the budget allows to do so. For example, facility 12 in m20-p8-r1 is interdicted by the attacker since the defender did not protect it despite the fact that there were sufficient funds to protect. A closer look into the solution explains this seemingly odd behavior. Namely, the benefit that can be gained by protecting facility 12 is more than offset by the increased capacity expansion cost. When facility 12 is protected, the attacker hits facility 6. This decreases the post-attack assignment cost, i.e., the attack-

er's objective value, in favor of the defender, but also increases the capacity expansion cost. Since the magnitude of the increase in the capacity expansion cost exceeds the reduction in the post-attack assignment cost, the defender's objective value eventually deteriorates. Hence, the defender prefers to leave facility 12 unprotected.

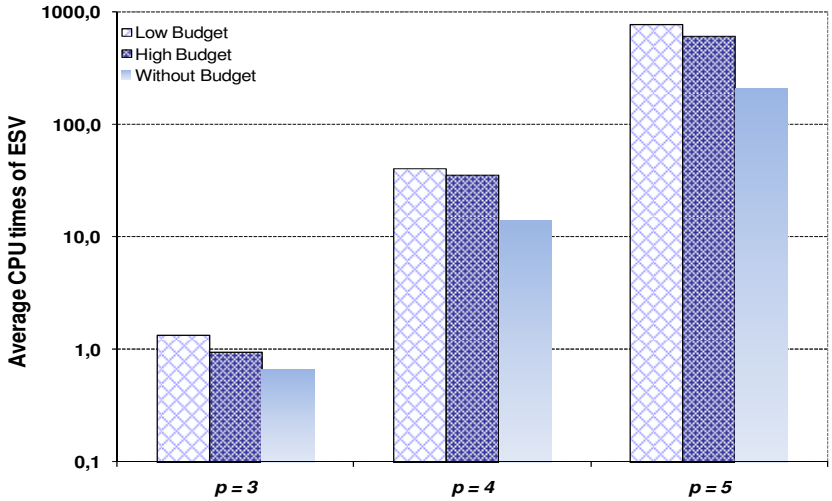


Figure 6.4. Exponential growth of the solution times of ESV.

Details of the TS-BST results obtained with Config-6 are omitted for the sake of brevity. Instead, they have been aggregated and compared to the analogous results of Config-3 for different budget levels in Table 6.7, Table 6.8 and Table 6.9. According to Table 6.8, our exhaustive search method ESV can solve low budget BPPCF instances to optimality faster than Config-6, which was actually the most efficient among all TS-BST parameter configurations tested. ESV works faster than Config-3 as well on both low and high budget instances. However, as seen in Figure 6.4, the solution times of ESV display an exponential trend as p grows; thus, ESV proves impractical for problems with $p > 5$.

Results in Table 6.7, Table 6.8 and Table 6.9 provide evidence for the trade-off between Z^*_{def} and CPU time. The average deviation from optimality with Config-6 is about six times that with Config-3, but it is barely above 1%. On the other hand, Config-3 takes about two and a half (two) times longer than Config-6 to solve low budget (high budget) test instances. Further comparison between these two configurations of the algorithm TS-BST can be found in Table 6.10, Table 6.11 and Table 6.12.

Table 6.10 where the effect of smaller neighborhood sizes and shorter *Max_Nonimp_Iter* is investigated on the entire test bed for zero budget, low budget and high budget, respectively.

Table 6.7. Comparison of the average test results of ESV, TS-BST Config-3 and Config-6 for instances with zero budget

	30 Zero Budget Instances ($p \leq 5$)			Gap (%)	
Test Criterion	Config-3	Config-6	ESV	Config-3 vs. ESV	Config-6 vs. ESV
Z^*_{def}	376646	379652	375233	0.38%	1.18%
Z^*_{att}	88829	90570	90346	-1.68%	0.25%
CPU time (s)	1205	375	73.6	1537%	409.5%
K_{iter}	52	28	—	—	—
K_{sols}	2677	1040	—	—	—
K_{Cplex}	2676	1039	—	—	—

Table 6.8. Comparison of the average test results of ESV, TS-BST Config-3 and Config-6 for low budget instances

	30 Low Budget Instances ($p \leq 5$)			Gap (%)	
Test Criterion	Config-3	Config-6	ESV	Config-3 vs. ESV	Config-6 vs. ESV
Z^*_{def}	329254	332094	328681	0.17%	1.04%
Z^*_{att}	77920	77040	78744	-1.05%	-2.16%
CPU time (s)	886.7	355.2	336.0	164%	6%
K_{iter}	44	25	—	—	—
K_{sols}	2189	942	—	—	—
K_{Cplex}	5518	2338	—	—	—

Table 6.9. Comparison of the average test results of ESV, TS-BST Config-3 and Config-6 for high budget instances

	30 High Budget Instances ($p \leq 5$)			Gap (%)	
Test Criterion	Config-3	Config-6	ESV	Config-3 vs. ESV	Config-6 vs. ESV
Z^*_{def}	303963	305668	303638	0.11%	0.67%
Z^*_{att}	66908	67540	66721	0.28%	1.23%
CPU time (s)	877.0	430.3	872.9	905%	-51%
K_{iter}	46	26	—	—	—
K_{sols}	2205	1001	—	—	—
K_{Cplex}	12661	5825	—	—	—

Table 6.10. The trade-off between solution quality of Config-3 and Config-6 for instances with zero budget

	75 Zero Budget Instances		
Test Criterion	Config-3	Config-6	Gap (%)
Z^*_{def} (m.u.)	386571	388913	0.61
CPU time (s)	3651	1402	−61.60
K_{iter}	54	30	−44.44
K_{sols}	3502	1506	−57.00
K_{Cplex}	3501	1505	−57.01

Table 6.11. The trade-off between solution quality of Config-3 and Config-6 for instances with low budget

	75 Low Budget Instances		
Test Criterion	Config-3	Config-6	Gap (%)
Z^*_{def} (m.u.)	350422	352150	0.49
CPU time (s)	3582.8	1515.0	−57.71
K_{iter}	50	29	−42.00
K_{sols}	3289	1477	−55.09
K_{Cplex}	9487	4130	−56.47

Table 6.12. The trade-off between solution quality of Config-3 and Config-6 for instances with high budget

	75 High Budget Instances		
Test Criterion	Config-3	Config-6	Gap (%)
Z^*_{def} (m.u.)	328146	329980	0.56
CPU time (s)	4020.4	1848.8	−54.01
K_{iter}	47	27	−42.55
K_{sols}	2982	1321	−55.70
K_{Cplex}	30218	13161	−56.45

Table 6.10 and Table 6.11 reveal the critical benefit of spending more on the protection of critical facilities in a service network. Increasing the average protection budget b_{tot} from 1000 (low) to 2367 (high) yields a savings of 22276 or 6.36% in the total cost of the defender obtained with Config-3. Running the algorithm with Config-6 leads to similar results, namely a savings of 22170 corresponding to 6.30%. These figures are certainly data-dependent and can fluctuate with the choice of the traveling cost per unit distance per unit demand. However, they adequately justify the significance of the defender's protection budget.

6.3.3. Effect of m, p and r on the CPU Time

In the last stage of the computational study we looked into the effect of the parameters m, p and r on the CPU times of TS-BST. To this end, we focused on Config-3 and consolidated its solutions for all low and high budget instances. The effect of the number of facilities struck by the attacker, namely r, was of primary interest, since it is an exogenous parameter largely outside the defender's control. We first calculated the CPU times averaged over five different values of m for each combination of p and r. Then, we did the same calculations by averaging over six different values of p for each combination of m and r. The resulting average times are depicted in Figure 6.5 and Figure 6.6.

In Figure 6.6 we observe that CPU times of TS-BST are linearly proportional to p, the number of facilities to be opened. Figure 6.7 suggests a higher order relationship be-

tween these and the number of candidate facility sites (m); but it is unlikely to be exponential as is the case in ESV. The effect of r on the CPU time is quite interesting. In both figures, it is scalable as r increases from 1 to 2. However, as r goes from 2 to 3, there occurs a notable jump in the CPU time. We attribute this phenomenon to the exponential growth of the binary search tree with every increment of r after the value 2. The impact of raising r from 2 to 3 is especially felt from $m = 30$ onward at all three levels of p to which $r = 3$ is applicable.

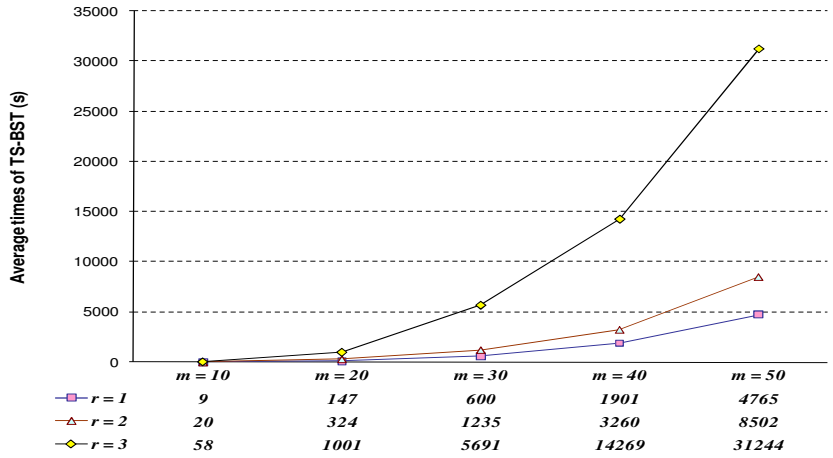


Figure 6.5. Solution times of TS-BST averaged over all m values for 225 test instances

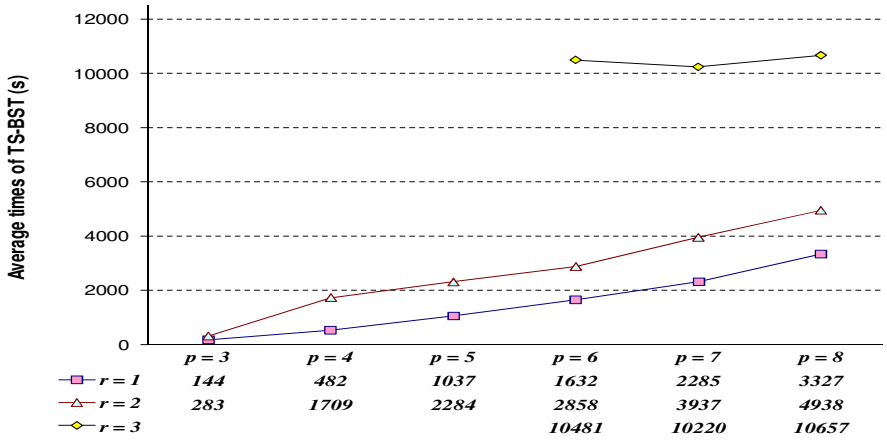


Figure 6.6. Solution times of TS-BST averaged over all p values for 225 test instances

6.4. Results of SSM

In this section, we report the computational results obtained by systematically testing the sequential solution method. We utilize test bed used to test the TS-BST algorithm. Therefore, we do not explain the characteristics of the problem instances again. We provide the results of the SSM algorithm and investigate the effect of the problem parameters *m*, *p* and *r* on the solution times. We also compare the performance of the TS-BST algorithm with the results of ESV and SSM algorithms. The results are reported in the following sections.

6.4.1. Experimental Results of the Generated Problems

In this stage of the computational study, we tested the proposed solution algorithm SSM. In order to evaluate the results of the algorithms proposed, we randomly generate problem instances as mentioned in the previous sections. We make sure that the results obtained from SSM algorithm is proven optimal by changing the value of CPLEX parameters. We also use ESV results in order to compare the optimal results of the simultaneous decision case with the optimal results of the sequential decision problem. In 90 out of 225 instances of the entire test bed, p value is less than or equal to five. We use the results of those problems for comparison with SSM results. In Appendix B we provide the detailed results of SSM outputs including the opened, protected and interdicted facility indices divided into Table B.4, Table B.5 and Table B.6 for zero budget, low budget and high budget instances, respectively. Problem names are indicative of the m, p, and r values. Column titles σ_p^{seq}, σ_b^{seq}, and σ_r^{seq} stand for the subsets of opened, protected, and interdicted facility sites in the solutions of SSM which uses sequential decision method in location and protection planning.

The objective values in Table B.4, Table B.5 and Table B.6 show that simultaneous planning strategy in BPPCF problem helps a system planner to construct better critical facility configurations under different budget levels compared to sequential planning strategy does if solution times are not taken into consideration. The average deviation in system planner's objective value obtained by SSM compared to the value obtained by TS-BST is

10%, 6.3% and 4.8% for the zero budget, low budget and high budget instances, respectively. In 32 of 225 problem instances SSM has found the same objective value for the system planner as the value found by TS-BST. Although the system planner incur a higher cost on the average if he/she uses sequential planning strategy, the results of the 4 problem instances, 2 from the instances with zero budget and other 2 from the low budget instances, show that lower costs may also be faced for this decision strategy.

When the solution methods are analyzed from the computational time point of view, sequential method turns out to be a lot cheaper. At this point, we have to point out that the algorithms for both sequential and simultaneous solution methods are performed in the same computer which, accordingly, has the same configuration properties. This reduction in CPU times is an inevitable result since the SSM solves only single tree search algorithm once p-median determines the optimal facility configuration; however, TS-BST solves one tree search algorithm for each iteration throughout the entire TS algorithm.

Table 6.13 . Comparison results between SSM and ESV

	30 Zero Budget Instances ($p \leq 5$)		30 Low Budget Instances ($p \leq 5$)		30 High Budget Instances ($p \leq 5$)	
Test Criterion	SSM	ESV	SSM	ESV	SSM	ESV
Z^*_{def}	413754	375233	355526	328.681	325474	303638
Z^*_{att}	67513	90346	78607	78744	69403	66721
CPU time (s)	2	73.6	2.1	265.2	2.1	212.5

Table 6.13 reveals the comparison results between SSM and the ESV algorithm. Since the results of SSM are also proven optimal, we can easily observe that taking concurrent decisions for facility planning and protection yields more saving in objective function. This savings are observed as 9%, 8%, 7% for zero budget, low budget and high budget instances, respectively.

Table 6.14. The trade-off between solution quality of SSM and TS-BST for instances with zero budget

	75 Zero Budget Instances		
Test Criterion	SSM	TS-BST	Gap (%)
Z_{def}	425224	386571	10%
Z_{att}	61500	76683	-20%
CPU time (s)	2.5	3651	-

Table 6.15. The trade-off between solution quality of SSM and TS-BST for instances with low budget

	75 Low Budget Instances		
Test Criterion	SSM	TS-BST	Gap (%)
Z_{def}	372598	350422	6%
Z_{att}	71716	69562	3%
CPU time (s)	2.6	3583	-

Table 6.16. The trade-off between solution quality of SSM and TS-BST for instances with high budget

	75 High Budget Instances		
Test Criterion	SSM	TS-BST	Gap (%)
Z_{def}	343815	328146	5%
Z_{att}	63286	61928	2%
CPU time (s)	2.8	4020	-

6.4.2. Effect of *m*, *p* and *r* on the CPU Time

In this section, we analyze the effect of the parameters *m*, *p* and *r* on the CPU times of SSM. We used the results of the all problem instances, namely zero budget, low budget and hi budget level problems, and consolidated these solutions as we did for TS-BST algorithm. We first calculated the CPU times averaged over five different values of *m* for each combination of *p* and *r*. Then, we did the same calculations by averaging over six different

values of p for each combination of m and r. The resulting average times are depicted in Figure 6.7 and Figure 6.8. Different from TS-BST, SSM does not show any linear relationship between CPU times and the number of facilities to be opened, p. Figure 6.7 suggests a higher order relationship between these and the number of candidate facility sites (m); but it is unlikely to be exponential as is the case in TS-BST and ESV.

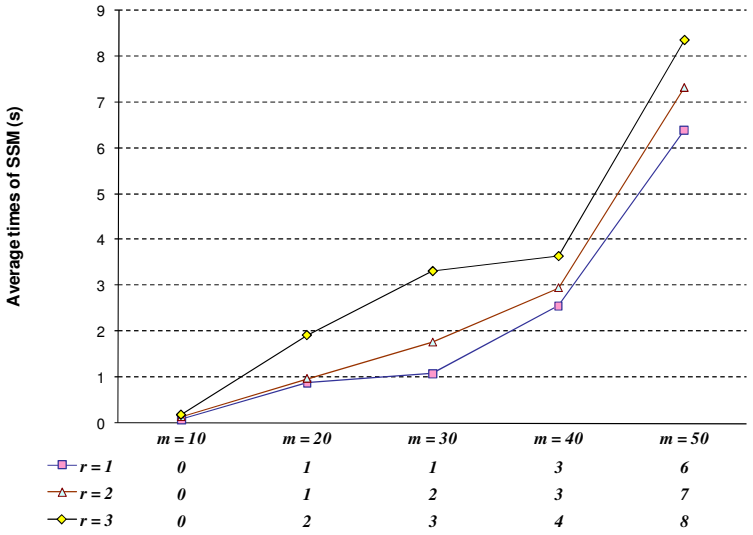


Figure 6.7. Solution times of SSM averaged over all m values for 225 test instances

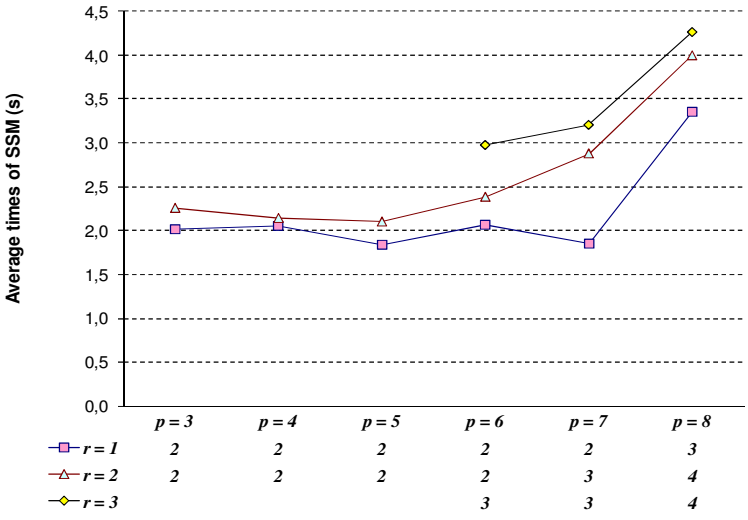


Figure 6.8. Solution times of SSM averaged over all p values for 225 test instances

6.5. Comparative Results Between TS-BST and SSM

In this section, we provide extended comparative results between TS-BST and SSM algorithms and drive some insights from these results. As explained before, in SSM the system planner does not consider the cost he/she realizes from the attacker's disruptive action while determining the locations of the p facilities. However, the results in TS-BST are derived from the concurrent decisions of the system planner.

For a system of facilities that are open to intentional attacks, the optimal configuration obtained by solving p-median problem, which basically considers the demand weighted travelling cost and–if necessary–the fixed cost of opening facilities in the cost function, may not be the best choice from the reliability point of view. This assumption is supported by SSM results. If the system planner decides to build a configuration by using sequential decision method, he/she agrees to incur a cost 10%, 6.3% and 4.8% above the cost he/she can face if decisions are made simultaneously for the zero budget, low budget and high budget, respectively. However, the experimental results show that this observation is not strictly and there is a weak probability, 1.8% for this test bed, that the reverse of this situation can be experienced. As can be observed from Table 6.17, there are only 4 instances that have lower Z_{def} values in SSM solution than they have in TS-BST solution.

Table 6.17. The distribution of problems solved with SSM according to their performance

	# of problems due to the deviation from Z_{def} compared to TS-BST			
Budget Level	> 0%	0 %	< 0%	**TOTAL**
Zero Budget	67	6	2	**75**
Low Budget	59	14	2	**75**
High Budget	63	12	0	**75**

A further issue to discuss in this section is the computational times of the proposed algorithms. It is obvious that SSM has lower CPU times since only one BST algorithm is solved in this method. On the other hand, TS-BST solves several search tree algorithms according to the number of solutions in the k-neighborhood, where $k = \{1, 2, 3\}$. Since the

critical facility planning and protection problem we addressed in this thesis is strategic and higher objective values may not be accepted, one can tolerate high computational times to find configurations that are more reliable to intentional attacks. The average CPU times of the SSM and TS-BST algorithms are provided for different budget level problems in Figure 6.9.

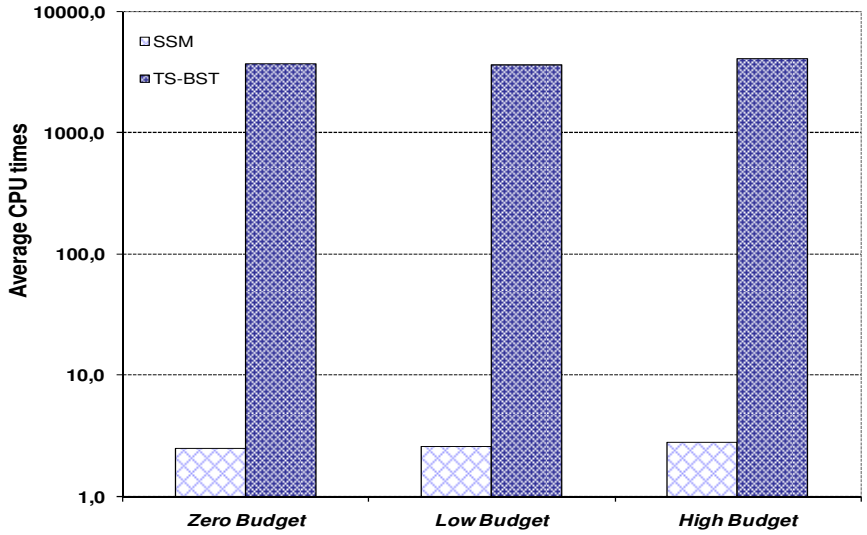


Figure 6.9. Average CPU times of TS-BST and SSM

Another discussion on the results of the proposed algorithms can be due to the effect of using protection budget. The idea of using more budget in protection of facilities always has a benefit can be a logical assumption in a protection planning problem. An attacker tries to give the most disruptive effect on the system efficiency from the customers' travelling cost point of view. However, system planner not only cares about this cost component but also cares about the capacity acquisition cost of customers who lost his/her initially assigned facility due to an attack. Of course, these cost components are only the ones that are realized after an attack. A system planner is also responsible for accounting for the costs required to create the initial facility configuration. This time, from the system planner's point of view, the aforementioned assumption may not be valid. In fact, we discussed this observation and give examples we encountered in the results of our test bed in Section 6.3.2. The examples of this condition do not go against the fact that protection efforts provide benefit in general. In Figure 6.10, the comparison of the average savings gained by

protection efforts in the TS-BST and the SSM algorithms can be seen. The marginal savings in the cost function decreases while protection budget increases which can be easily seen in the second part of the lines both for SSM and TS-BST. Higher average saving in SSM than it is in TS-BST can be due to the fact that higher cost cuts can be made from higher objective values.

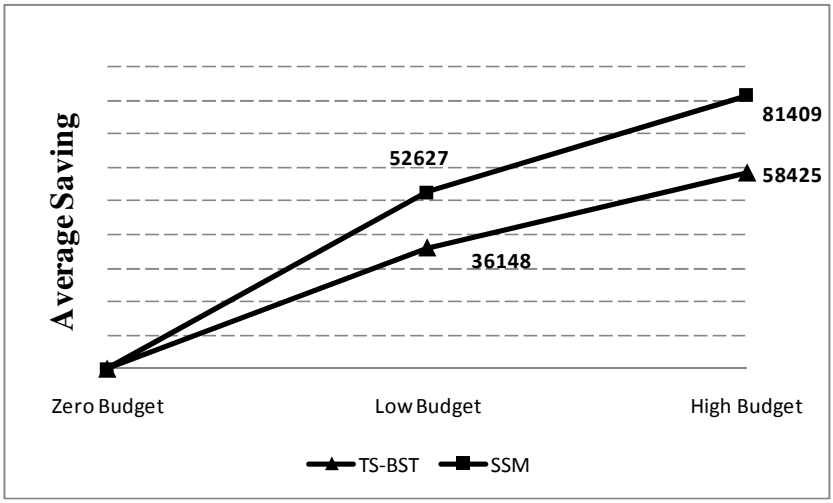


Figure 6.10. The effect of protection budget level in both TS-BST and SSM

The last discussion topic is related to the facility configurations obtained in both algorithms. It is observed that in the results of SSM, facility locations are decentralized. This observation is reasonable since the locations of the facilities are determined by solving a *p*-median problem in SSM. In a *p*-median problem, the resulting facility configuration has generally a decentralized structure in order to minimize total travelling cost. On the other hand, the facility configurations obtained by TS-BST has not a general form. In some problem instances, facilities are separately located; whereas for another bunch of problem instances, facility locations are centralized and adjacent as if they are back-up facilities. The plot diagrams of the facility locations and the customer zones of the problem instances m50-p3-R2 and m40-p6-R3 are provided for both SSM and TS-BST algorithms and different budget levels in the following figures. We choose these problem instances because their deviations in the SSM objective from the TS-BST objective are higher than the other instances'.

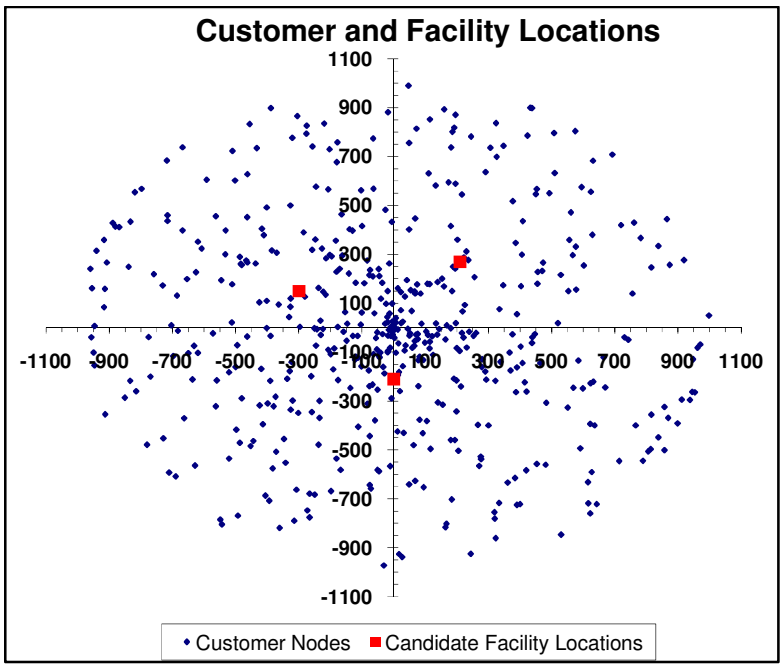


Figure 6.11. Plot of the SSM solution of the m50-p3-R2 problem with low, high, and zero budget– deviations in objective function: 7%, 20%, 20%, respectively

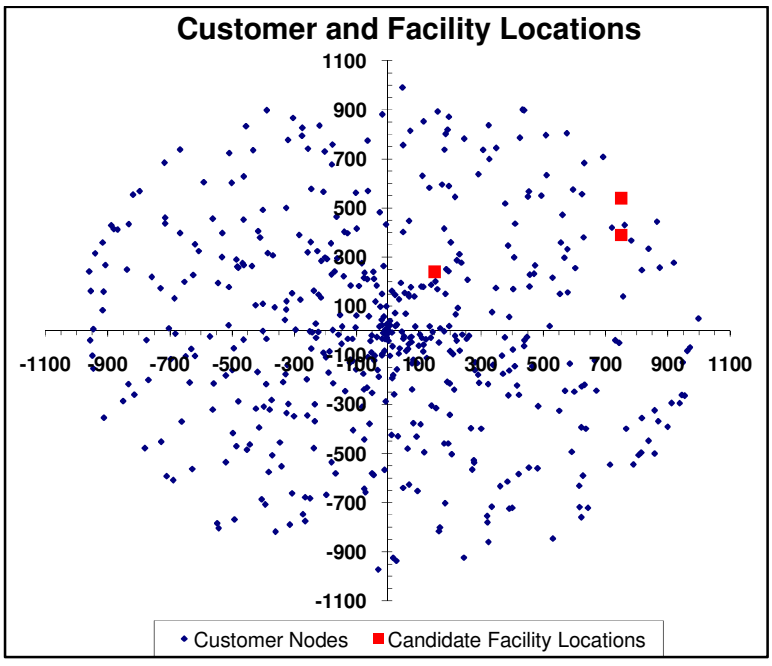


Figure 6.12. Plot of the TS-BST solution of m50-p3-R2 problem with low budget

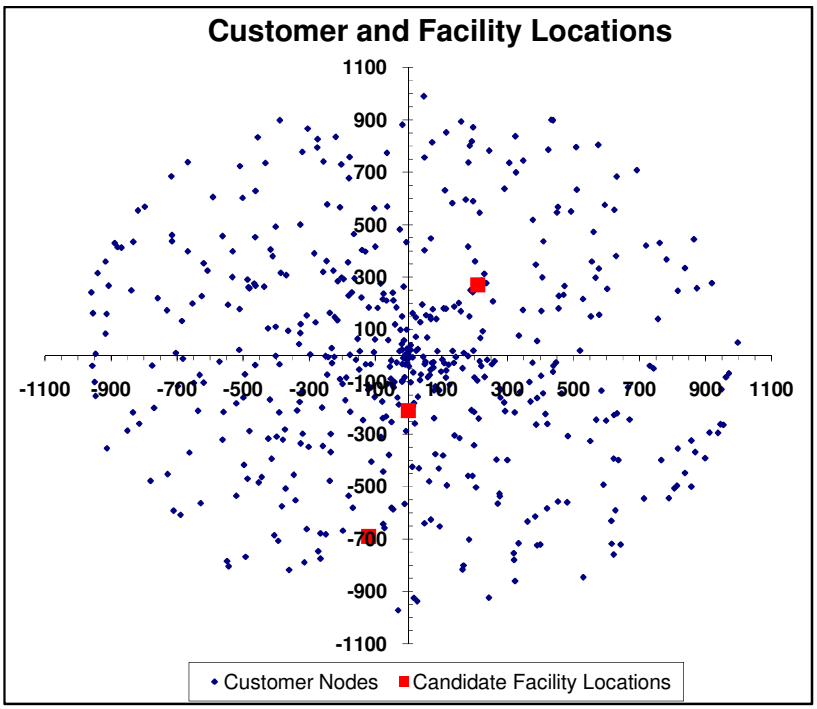


Figure 6.13. Plot of the TS-BST solution of m50-p3-R2 problem with high budget

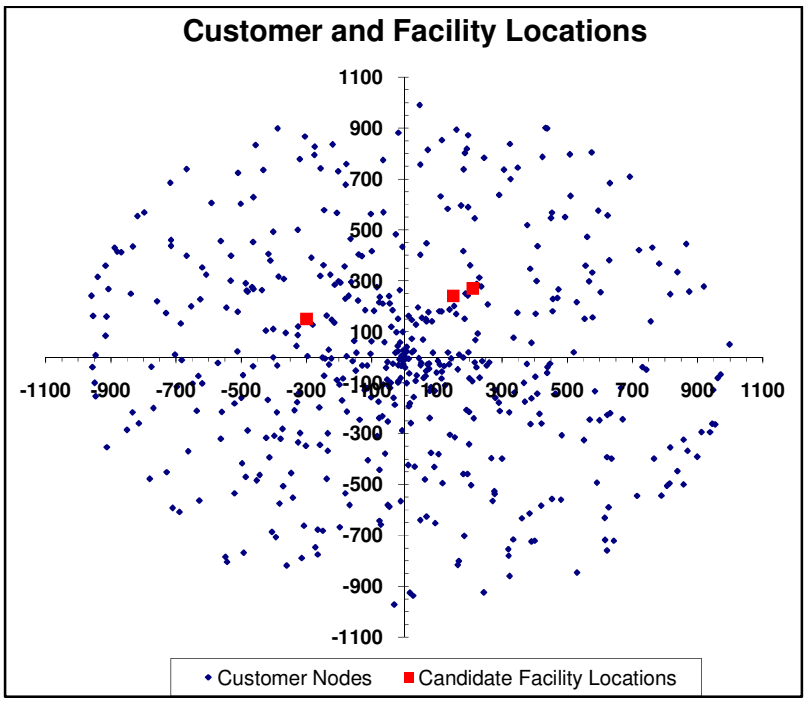


Figure 6.14. Plot of the TS-BST solution of m50-p3-R2 problem with zero budget

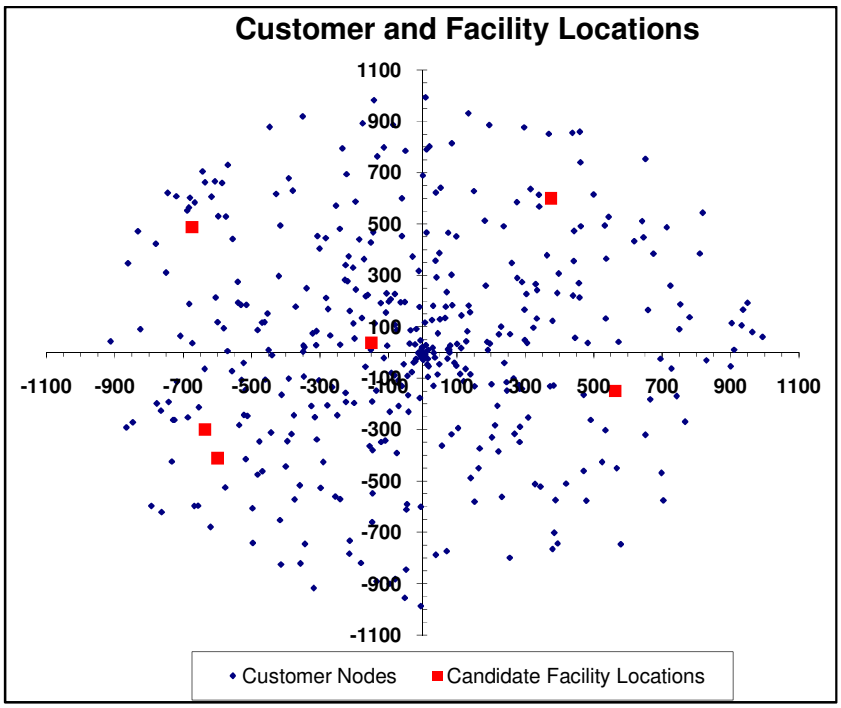


Figure 6.15. Plot of the SSM solution of m40-p6-R3 problem with low, high, and zero budget – deviations in objective function: 9.3%, 9.7%, 8.5%, respectively

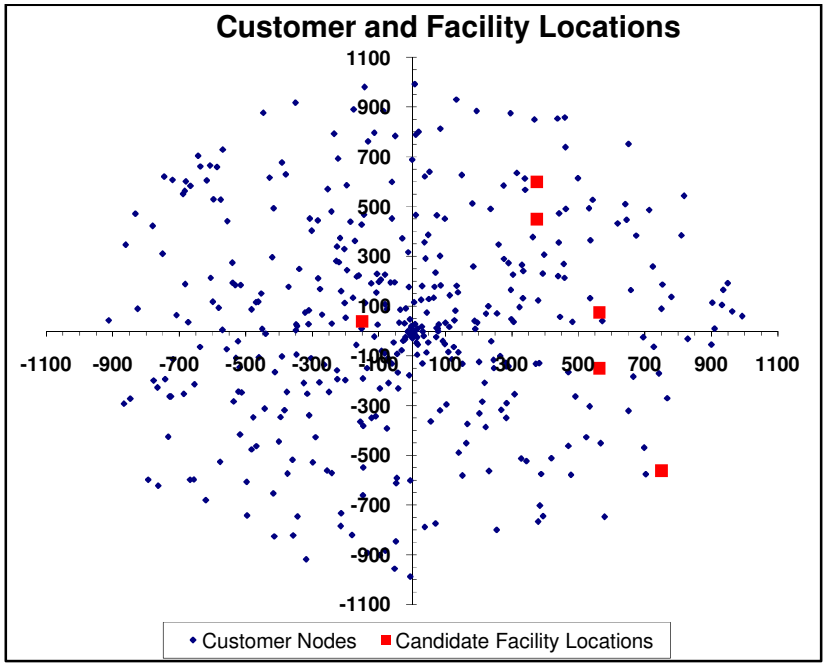


Figure 6.16. Plot of the TS-BST solution of m40-p6-R3 problem with low budget

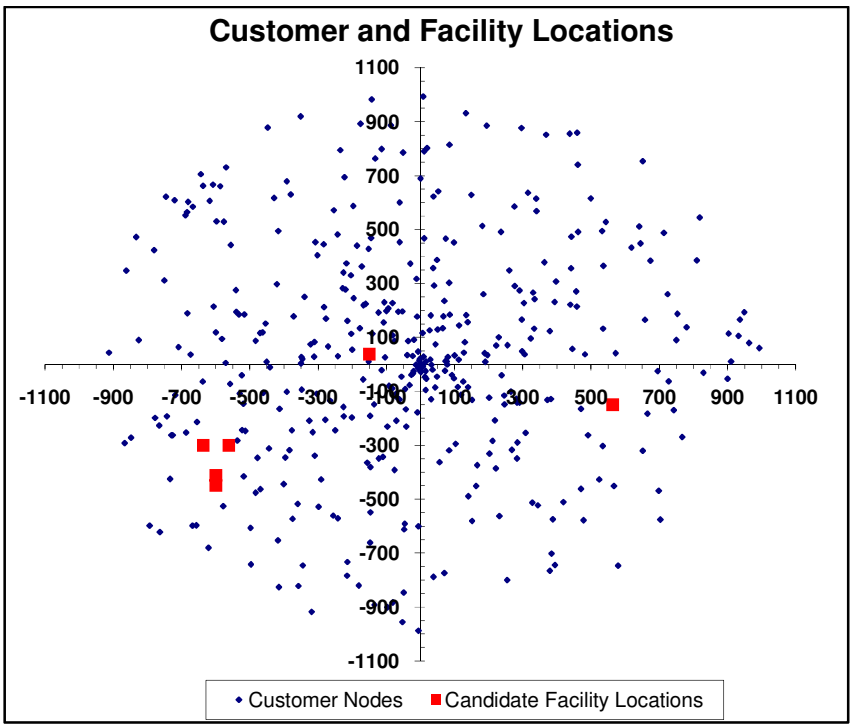


Figure 6.17. Plot of the TS-BST solution of m40-p6-R3 problem with high budget

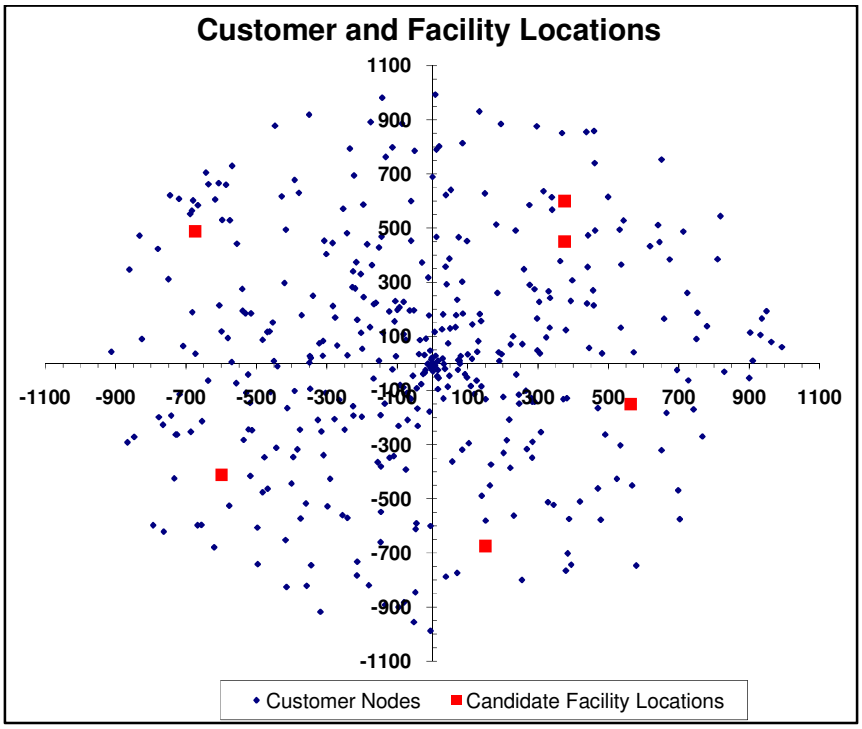


Figure 6.18. Plot of the TS-BST solution of m40-p6-R3 problem with zero budget

7. CONCLUSION AND FUTURE WORK

In this thesis, we have studied the problem of location and protection planning of the critical facilities which can be considered as a static Stackelberg game between a system planner and a potential attacker. The system planner tries to locate and operate a set of facilities to provide service to customers residing at a number of demand nodes. Since these facilities are prone to intentional attacks, the system planner is also responsible for making decisions about the protection of the opened facilities so as to minimize the disruption. We assume that the attacker is intelligent and acts rationally; that is, he/she aims to give maximum possible disruption to the service providence.

This problem includes strategic decisions which can be taken either in a sequential or simultaneous manner from the system planner's point of view. Since there is such a strategic option, we decided to handle this problem considering these two different perspectives which differ in the decision making process. First, we address the simultaneous decision making option in bilevel *p*-median problem for the planning and protection of critical facilities (BPPCF) which is modeled as a bilevel mixed-integer programming problem and solved by using a two-phase heuristic algorithm called TS-BST. The first phase of TS-BST searches a neighborhood solution according to the move operator while the second phase capitalizes on an efficient binary search tree. The sequential solution method, on the other hand, first locates the facilities according to the optimal solution of *p*-median problem and then considers protection planning decisions according to this configuration with the help of a binary search tree algorithm. In order to assess the performance of TS-BST and SSM for small problem instances, we propose a validation algorithm called ESV.

An extensive parametric study is conducted to see the effect of some parameters on the efficiency of the TS-BST such as the number of facilities to be located, the number of candidate sites, and the maximum number of interdicted facilities. We also test different variations of tabu search stopping conditions to capture the tradeoff between the objective value of the defender and the solution time spent by the algorithm.

The efficiency of the proposed algorithms is analyzed using a test bed consists of 225 randomly generated instances. From the computational time point of view, SSM is the fastest algorithm with an average CPU time of 2.63 seconds for 225 test problems. TS-BST is relatively slow but provides good objective values. Having SSM lower CPU times is a trivial result since only one BST is applied throughout the entire algorithm. On the other hand, TS-BST solves several search tree algorithms (6952 Cplex calls on the average) according to the number of solutions in the k-neighborhood where $k = \{1, 2, 3\}$. Since the critical facility planning and protection problem we addressed in this thesis is strategic and higher objective values may not be acceptable, one can tolerate high computational times to find configurations that are more reliable to intentional attacks.

When the objective values obtained from TS-BST are compared with the optimal values determined by ESV, the gap is found as 0.22% on the average which indicates that the performance of TS-BST is quite promising. Besides, the comparative results of the system planner's objective value, Z_{def}, between SSM and TS-BST explain the effect of the decision making sequence chosen by the planner. The average deviation of the objective value obtained by SSM from the value obtained by TS-BST is calculated as 13.3%. This indicates the importance of simultaneous decision making from the system planner's perspective. However, when the low computational times of SSM are taken into account, SSM outperforms TS-BST. In addition, the experimental results show that there is probability that the SSM can give a better objective value than TS-BST does for this test bed. But it is important to mention that this result can be sensitive to the characteristics of the problem instances. However, for each different budget levels it is observed that SSM can never provide better average objective values than TS-BST does.

Another discussion on the results of the proposed algorithms can be due to the effect of using protection budget. High costs associated with the facility construction and capacity acquisition make facility location decisions long-term investments. To make such undertakings cost-effective, decision makers are interested in finding the ways of making facilities remain in operation for a long time. This strategic nature of facility location problems requires that any reasonable model contain some aspect of facility reliability. The results obtained in this study reveal the importance of using protection resources.

The idea of using more budget in protection of facilities always has a benefit can be a logical assumption in a protection planning problem. However, in our problem we encountered some solutions that the system planner does not choose to protect any facility even if he/she has enough budget. This interesting situation makes sense when we question the cost components of the system planner. System planner cares about not only the post-attack travelling costs, but also the capacity acquisition cost of re-assigned customers due to an attack. Also, the objective function of the system planner includes the costs required to create the initial facility configuration. System planner may face a situation where the magnitude of the increase in the capacity expansion cost exceeds the reduction in the post-attack assignment cost. Therefore, the defender's objective value eventually deteriorates and the aforementioned assumption may not be valid from the system planner's point of view. However, the examples of this condition do not go against the fact that protection efforts provide benefit in general. The average savings in the system planner's cost are 12.2% and 15.8% for TS-BST and SSM methods, respectively.

The last observation we want to point out here is the facility configurations obtained in both algorithms. Since in SSM the locations of the facilities are determined by solving a capacitated p-median problem, the resulting facility configurations have generally decentralized structures. On the other hand, the facility configurations obtained by TS-BST does not have general pattern. In some problem instances, facilities are separately located; whereas for another bunch of problem instances, facility locations are centralized and adjacent as if they are back-up facilities.

In this study, we quantified and showed the benefit of the bilevel programming formulation which enables us to treat facility location and protection decisions in parallel. While modeling the problem, we assumed that an attack to a facility has a complete effect on the service providence of this facility. Further attempts may be to extend this assumption where partial attacks are realized on the facilities. Moreover, the expected travelling cost of the customers who do not have information about the interdiction status of the facilities can be considered rather than calculating the demand weighted travelling cost in the cost function.

APPENDIX A: VISUALIZATION OF THE CUSTOMER LOCATIONS AND CANDIDATE FACILITY SITES ON THE *XY*-PLANE

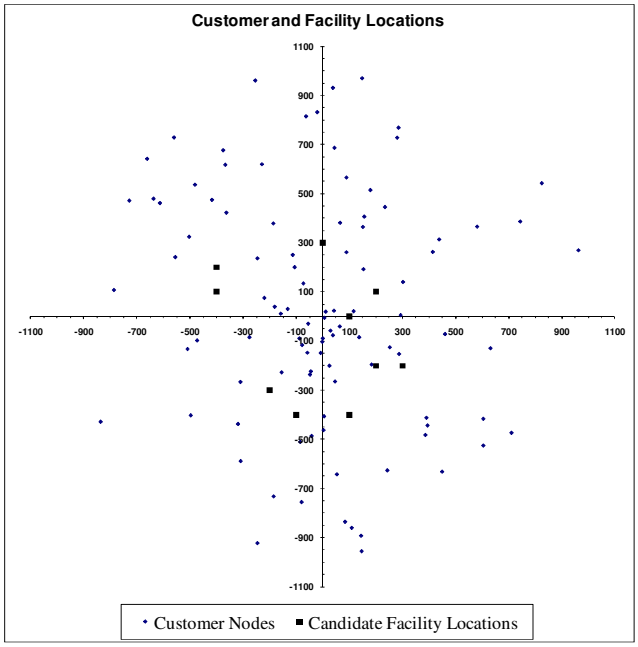


Figure A.1. Distribution of customer and facility locations for $m = 10$

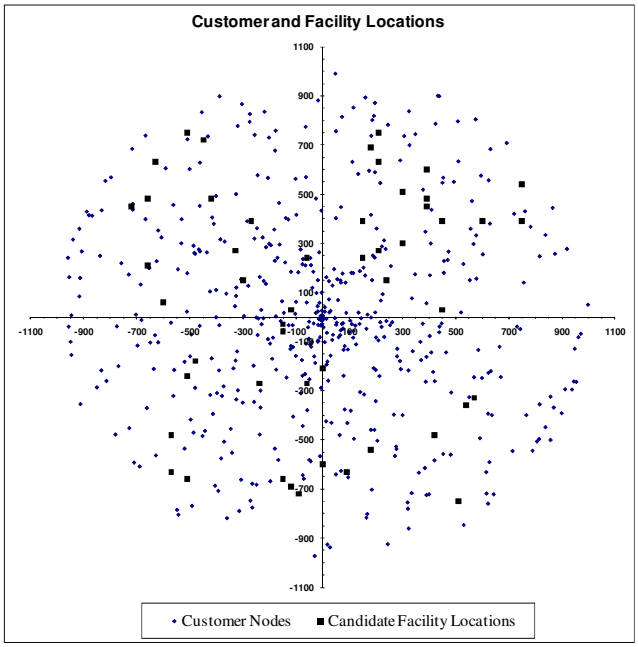


Figure A.2. Distribution of customer and facility locations for m = 50

APPENDIX B: DETAILED SOLUTIONS OF THE TEST INSTANCES

Table B.1. Detailed test results of ESV on 30 zero budget test instances

Name	Z_{def}^{opt}	Z_{att}^{opt}	time (s)	$\boldsymbol{\sigma}_p^{opt}$	$\boldsymbol{\sigma}_b^{opt}$	$\boldsymbol{\sigma}_r^{opt}$
m10-p3-r1	154561	25798	0.0	5, 8, 10	none	5
m10-p3-r2	181490	35815	0.0	1, 2, 5	none	1, 2
m10-p4-r1	160964	23064	0.01	1, 5, 7, 8	none	5
m10-p4-r2	190924	26511	0.01	2, 4, 5, 8	none	2, 5
m10-p5-r1	171325	22650	0.01	1, 5, 7, 8, 10	none	5
m10-p5-r2	189861	25311	0.01	5, 6, 7, 8, 10	none	5, 6
m20-p3-r1	253266	60516	0.06	6, 12, 15	none	12
m20-p3-r2	316424	79824	0.05	6, 15, 17	none	6, 15
m20-p4-r1	250801	47501	0.4	6,12,15,17	none	17
m20-p4-r2	305046	66146	0.5	2,6,12,15	none	6,12
m20-p5-r1	265499	45387	1.8	6,9,12,15,17	none	12
m20-p5-r2	293698	59860	2.9	6,7,12,15,20	none	12,20
m30-p3-r1	365408	107308	0.3	12,14,15	none	14
m30-p3-r2	422697	110334	0.23	12,26,30	none	26,30
m30-p4-r1	356198	75398	3.1	8,12,15,26	none	26
m30-p4-r2	412926	101326	4.1	8,15,26,27	none	15,27
m30-p5-r1	331054	66554	24.4	8,12,15,26,27	none	27
m30-p5-r2	396091	92291	40.6	12,14,15,17,27	none	14,17
m40-p3-r1	461462	101087	0.9	1,11,32	none	1
m40-p3-r2	565697	181847	0.8	1,3,5	none	3,5
m40-p4-r1	438932	110232	13.6	2,28,38,40	none	28
m40-p4-r2	526473	157373	17.9	1,2,17,28	none	2,28
m40-p5-r1	420148	111448	147.0	1,3,17,28,40	none	17
m40-p5-r2	497102	135502	247.4	1,3,28,38,40	none	3,28
m50-p3-r1	523846	158196	2.3	4,30,32	none	30
m50-p3-r2	638935	168385	1.9	4,6,32	none	4,32
m50-p4-r1	500224	120211	43.3	6,26,27,48	none	26
m50-p4-r2	608746	158196	56.4	4,6,32,47	none	6,47
m50-p5-r1	489506	107006	597.1	26,27,30,32,39	none	30
m50-p5-r2	567694	129306	1002.2	4,6,10,32,39	none	10,39

Table B.2. Detailed test results of ESV on 30 low budget test instances

Name	Z_{def}^{opt}	Z_{att}^{opt}	time (s)	$\boldsymbol{\sigma}_p^{opt}$	$\boldsymbol{\sigma}_b^{opt}$	$\boldsymbol{\sigma}_r^{opt}$
m10-p3-r1	140779	24467	0.0	1,2,8	8	1
m10-p3-r2	152571	28058	0.0	1,2,8	8	1,2
m10-p4-r1	156596	23546	0.0	1,5,6,8	8	1
m10-p4-r2	160717	24467	0.0	1,2,8,9	8	1,9
m10-p5-r1	167354	20379	0.1	1,2,5,6,8	8	2
m10-p5-r2	176296	23546	0.1	1,2,5,6,8	8	1,2
m20-p3-r1	231823	52923	0.2	12,15,17	15	12
m20-p3-r2	235908	67683	0.1	2,15,18	15	2,18
m20-p4-r1	245088	47038	1.6	6,12,15,17	17	12
m20-p4-r2	264298	52923	1.7	12,15,17,20	15	12,20
m20-p5-r1	259851	50839	8.9	2,12,15,17,18	15	12
m20-p5-r2	281613	50951	12.6	12,15,17,18,20	15	12,20
m30-p3-r1	334143	94093	1.0	14,26,28	26	28
m30-p3-r2	363017	108304	0.6	17,19,26	26	17,19
m30-p4-r1	316491	69491	13.4	8,12,26,27	12	27
m30-p4-r2	360673	78336	14.3	8,12,19,26	12	19,26
m30-p5-r1	321468	64468	125.6	8,12,14,26,27	12	27
m30-p5-r2	365398	72886	174.5	12,14,19,26,30	12	14,30
m40-p3-r1	376467	102267	3.2	3,28,32	32	28
m40-p3-r2	405494	112694	1.8	3,28,32	32	3,28
m40-p4-r1	387842	101242	58.3	3,27,28,32	32	3
m40-p4-r2	394967	102267	61.8	3,27,28,32	32	27,28
m40-p5-r1	389879	88979	756.5	3,17,19,28,32	32	19
m40-p5-r2	411192	101242	1059.7	3,5,27,28,32	32	3,5
m50-p3-r1	482788	133788	8.0	24,26,27	27	24
m50-p3-r2	512723	160623	4.6	24,26,32	32	24,26
m50-p4-r1	466003	119253	186.1	4,26,27,32	27	26
m50-p4-r2	509593	154818	195.9	15,26,32,43	32	15,43
m50-p5-r1	473880	103380	3082.3	4,6,27,30,32	27	30
m50-p5-r2	515504	127,354	4307.1	9,18,27,30,44	27	18,44

Table B.3. Detailed test results of ESV on 30 high budget test instances

Name	Z_{def}^{opt}	Z_{att}^{opt}	time (s)	$\boldsymbol{\sigma}_p^{opt}$	$\boldsymbol{\sigma}_b^{opt}$	$\boldsymbol{\sigma}_r^{opt}$
m10-p3-r1	137612	22775	0.0	8,9,10	8,9,10	none
m10-p3-r2	137612	22775	0.0	8,9,10	8,9,10	none
m10-p4-r1	152631	21206	0.1	1,2,5,8	5,8	2
m10-p4-r2	153596	24109	0.1	1,7,8,10	1,8	7,10
m10-p5-r1	164470	20270	0.2	1,2,5,6,8	2,6,8	1
m10-p5-r2	165756	21206	0.2	1,5,7,8,10	1,5,8	7,10
m20-p3-r1	222193	46818	0.3	6,15,20	6,15,20	none
m20-p3-r2	222193	46818	0.2	6,15,20	6,15,20	none
m20-p4-r1	238869	46356	3.1	9,12,15,17	12,15	9
m20-p4-r2	235236	52923	2.6	2,15,17,18	15,17	2,18
m20-p5-r1	251767	44705	30.1	2,9,12,15,17	9,12,15	2
m20-p5-r2	249351	47038	30.0	2,6,15,17,18	6,15,17	2,18
m30-p3-r1	287015	68965	1.4	15,26,27	15,26,27	none
m30-p3-r2	287015	68965	1.0	15,26,27	15,26,27	none
m30-p4-r1	309950	69612	26.4	12,13,26,27	12,26,	27
m30-p4-r2	310203	68965	21.4	7,15,26,27	15,26,27	7
m30-p5-r1	312463	62125	408.3	12,13,14,26,27	12,14,26	27
m30-p5-r2	318863	68876	408.8	7,12,13,26,27	12,26,27	7,13
m40-p3-r1	364870	93920	4.5	17,32,38	32,38	17
m40-p3-r2	364870	93920	3.1	17,32,38	32,38	17
m40-p4-r1	372397	85697	114.0	1,3,28,32	1,32	28
m40-p4-r2	386320	93920	93.4	17,19,32,38	32,38	17,19
m40-p5-r1	375438	76988	2321.4	1,3,17,28,32	1,17,32	28
m40-p5-r2	386109	83509	2390.0	3,17,19,32,38	3,32,38	17,19
m50-p3-r1	427963	113313	11.2	4,27,39	4,27,39	none
m50-p3-r2	427963	113313	7.8	4,27,39	4,27,39	none
m50-p4-r1	459706	104706	364.5	4,9,27,30	4,27	9
m50-p4-r2	460063	113313	293.9	4,26,27,39	4,27,39	26
m50-p5-r1	459071	99984	9786.4	4,9,15,26,27	4,9,27	15
m50-p5-r2	467573	104548	9861.5	4,9,18,27,44	4,9,27	18,44

Table B.4. Detailed test results of TS-BST using Config-3 on 75 test instances with zero budget

Name	σ_p^*	σ_b^*	σ_r^*	Z_{def}^*	Z_{att}^*	time (s)	K_{iter}	K_{Cplex}	K_{sols}
m10-p3-r1	5, 8, 10	none	5	154561	25798	1.8	31	280	279
m10-p3-r2	1, 2, 5	none	1, 2	181490	35815	5.4	45	406	405
m10-p4-r1	1, 5, 7, 8	none	5	160964	23064	5.4	43	517	516
m10-p4-r2	2, 4, 5, 8	none	2, 5	190924	26511	7.5	38	457	456
m10-p5-r1	1, 5, 7, 8, 10	none	5	171325	22650	5.4	31	373	372
m10-p5-r2	5, 6, 7, 8, 10	none	5, 6	189861	25311	14.8	43	517	516
m10-p6-r1	1, 5, 7, 8, 9, 10	none	5	188614	21951	7.9	33	397	396
m10-p6-r2	2, 5, 6, 7, 8, 10	none	5, 6	197506	23568	15.4	38	457	456
m10-p6-r3	5, 6, 7, 8, 9, 10	none	7, 8, 10	225723	27085	42.3	62	745	744
m10-p7-r1	1, 4, 5, 6, 8, 9, 10	none	10	204988	18763	8.6	39	352	351
m10-p7-r2	1, 5, 6, 7, 8, 9, 10	none	5, 6	205864	21951	13.6	33	298	297
m10-p7-r3	2, 3, 5, 6, 7, 8, 10	none	3, 5, 6	231606	23568	37.3	65	586	585
m10-p8-r1	1, 2, 4, 5, 6, 7, 8, 10	none	4	222945	18645	7.1	39	235	234
m10-p8-r2	1, 2, 5, 6, 7, 8, 9, 10	none	5, 6	223689	21677	10.3	31	187	186
m10-p8-r3	1, 4, 5, 6, 7, 8, 9, 10	none	4, 5, 6	239539	21951	21.4	35	211	210
m20-p3-r1	6, 12, 15	none	12	253266	60516	18.3	45	1081	1080
m20-p3-r2	6, 15, 17	none	6, 15	316424	79824	48.8	85	2041	2040
m20-p4-r1	6,12,15,17	none	17	250801	47501	38.9	48	1441	1440
m20-p4-r2	2,6,12,15	none	6,12	305046	66146	64.9	39	1171	1170
m20-p5-r1	6,9,12,15,17	none	12	265499	45387	46.4	37	1222	1221
m20-p5-r2	6,7,12,15,20	none	12,20	293698	59860	180.3	67	2212	2211
m20-p6-r1	2,6,12,15,17,18	none	17	280207	45332	73.6	39	1405	1404
m20-p6-r2	2,6,7,12,15,20	none	12,20	311257	58420	165.1	39	1405	1404
m20-p6-r3	2,6,8,12,15,17	none	8,12,17	347129	58979	496.6	55	1981	1980
m20-p7-r1	2,6,9,12,15,17,18	none	12	292133	43371	204.8	50	1951	1950
m20-p7-r2	6,7,9,12,15,17,20	none	9,17	312133	44733	273.8	42	1639	1638
m20-p7-r3	6,7,9,12,15,17,18	none	9,12,17	340294	57831	730.5	66	2575	2574
m20-p8-r1	2,6,7,9,12,15,17,18	none	12	307636	42811	259.2	45	1891	1890
m20-p8-r2	2,6,7,9,12,15,17,20	none	12,20	321001	43351	514.7	62	2605	2604
m20-p8-r3	2,6,7,12,15,17,18, 20	none	12,17,20	351874	57787	701.9	48	2017	2016
m30-p3-r1	12,14,26	none	26	368768	94268	51.4	33	1189	1188
m30-p3-r2	12,26,30	none	26,30	422697	110334	190.4	40	1441	1440
m30-p4-r1	8,12,15,26	none	26	356198	75398	120.7	34	1531	1530

(Table B.4 continued)

Name	σ_p^*	σ_b^*	σ_r^*	Z_{def}^*	Z_{att}^*	time (s)	K_{iter}	K_{Cplex}	K_{sols}
m30-p4-r2	8,15,26,27	none	15,27	412926	101326	766.1	48	2161	2160
m30-p5-r1	12,14,15,19,26	none	14	338144	69869	316.3	47	2539	2538
m30-p5-r2	8,12,14,15,27	none	8,14	400591	92291	938.2	66	3565	3564
m30-p6-r1	8,12,14,15,26,27	none	27	336297	61797	443.7	41	2584	2583
m30-p6-r2	12,14,15,19,26,27	none	19,26	393153	77391	973.3	55	3466	3465
m30-p6-r3	10,12,14,20,26,27	none	20,26,17	432969	90019	7377.9	75	4726	4725
m30-p7-r1	8,12,14,15,19,26, 27	none	27	350667	60979	1097.7	66	4555	4554
m30-p7-r2	7,12,14,15,19,26, 27	none	19,26	405524	76837	1591.4	67	4624	4623
m30-p7-r3	8,12,13,14,15,17, 27	none	8,14,17	433753	91603	5362.8	86	5935	5934
m30-p8-r1	7,8,12,14,15,19,26, 27	none	14	357883	58720	1507.0	60	4681	4680
m30-p8-r2	8,11,12,14,15,19, 26,27	none	14,27	392243	62493	1305.4	36	2809	2808
m30-p8-r3	8,11,12,14,15,19, 26,27	none	8,19,26	431894	76319	2472.1	40	3121	3120
m40-p3-r1	1,11,32	none	1	461462	101087	157.4	40	1921	1920
m40-p3-r2	36,38,40	none	36,40	567857	167094	342.5	35	1681	1680
m40-p4-r1	1,9,32,38	none	1	445643	87968	644.7	54	3403	3402
m40-p4-r2	1,2,3,17	none	2,3	530973	157373	1983.1	48	3025	3024
m40-p5-r1	1,3,17,28,40	none	17	420148	111448	1213.8	61	4576	4575
m40-p5-r2	1,3,28,38,40	none	3,28	497102	135502	2604.0	64	4801	4800
m40-p6-r1	1,11,14,28,32,38	none	1	426283	71346	3493.0	104	9361	9360
m40-p6-r2	1,2,25,28,32,38	none	1,38	486825	90975	2678.0	45	4051	4050
m40-p6-r3	1,2,22,28,38,40	none	2,22,28	548940	135502	21118.4	105	9451	9450
m40-p7-r1	1,2,3,11,22,32,38	none	1	425922	64835	4224.7	77	7624	7623
m40-p7-r2	1,11,12,22,28,32, 36	none	1,36	486860	86148	2867.5	37	3664	3663
m40-p7-r3	1,3,27,28,37,38,40	none	3,27,28	532017	135204	21834.0	128	12673	12672
m40-p8-r1	1,2,3,11,22,28,32, 38	none	1	433963	63901	3924.0	48	5329	5328
m40-p8-r2	1,2,3,16,17,25,32, 38	none	17,38	483695	73220	4642.8	42	4663	4662
m40-p8-r3	1,11,12,22,32,35, 36,40	none	1,36,40	526329	87729	8185.2	35	3886	3885
m50-p3-r1	4,30,32	none	30	523846	158196	349.1	40	2521	2520
m50-p3-r2	4,6,32	none	4,32	638935	168385	952.2	74	4663	4662
m50-p4-r1	6,26,27,48	none	26	500224	120211	963.2	35	2836	2835
m50-p4-r2	4,6,32,47	none	6,47	608746	158196	8959.6	82	6643	6642

(Table B.4 continued)

Name	σ_p^*	σ_b^*	σ_r^*	Z_{def}^*	Z_{att}^*	time (s)	K_{iter}	K_{Cplex}	K_{sols}
m50-p5-r1	26,27,30,32,39	none	30	489506	107006	5289.3	107	10594	10593
m50-p5-r2	5,15,26,27,48	none	15,26	581761	130536	9884.7	96	9505	9504
m50-p6-r1	26,27,30,32,39,47	none	30	489872	101222	5186.9	65	7411	7410
m50-p6-r2	4,6,15,27,47,48	none	4,15	553280	112880	8568.7	61	6955	6954
m50-p6-r3	6,26,27,32,39,48	none	27,39,48	630466	125928	41851.1	86	9805	9804
m50-p7-r1	4,5,6,27,30,43,47	none	30	498293	90830	6391.3	54	6967	6966
m50-p7-r2	5,6,19,26,27,39,50	none	26,50	557397	108672	10488.8	62	7999	7998
m50-p7-r3	4,6,30,32,39,43,47	none	6,39,47	625090	133590	20798.4	46	5935	5934
m50-p8-r1	4,5,6,27,30,39,43, 47	none	30	504396	86621	7933.3	41	5905	5904
m50-p8-r2	17,26,27,30,32,36, 39,50	none	30,50	556209	98084	9536.4	42	6049	6048
m50-p8-r3	4,5,6,26,27,32,47, 48	none	4,26,32	589048	111748	28260.0	50	7201	7200

Table B.5. Detailed test results of TS-BST using Config-3 on 75 low budget test instances

Name	σ_p^*	σ_b^*	σ_r^*	Z_{def}^*	Z_{att}^*	time (s)	K_{iter}	K_{Cplex}	K_{sols}
m10-p3-r1	1,2,8	8	1	140779	24467	3.3	38	686	342
m10-p3-r2	1,2,8	8	1,2	152571	28058	6.0	35	948	315
m10-p4-r1	1,5,6,8	8	1	156596	23546	6.1	33	794	396
m10-p4-r2	1,5,8,9	8	1,9	162499	24374	11.7	33	1191	396
m10-p5-r1	1,2,5,6,8	8	2	167354	20379	8.9	34	818	408
m10-p5-r2	1,2,5,6,8	8	1,2	176296	23546	25.0	43	1551	516
m10-p6-r1	1,5,6,8,9,10	8	10	184110	20060	13.8	39	938	468
m10-p6-r2	1,5,6,7,8,10	8	5,6	188575	22650	27.4	36	1299	432
m10-p6-r3	1,2,5,6,8,9	8	1,2,9	196921	23546	66.0	47	2260	564
m10-p7-r1	1,2,5,6,7,8,10	8	6	200248	19473	12.8	38	686	342
m10-p7-r2	1,5,6,7,8,9,10	none	5,6	205864	21951	25.3	33	894	297
m10-p7-r3	1,2,5,6,7,8,10	8	1,5,6	213443	23568	45.4	33	1192	297
m10-p8-r1	1,2,5,6,7,8,9,10	8	6	218855	19155	8.5	31	374	186
m10-p8-r2	1,2,5,6,7,8,9,10	none	5,6	223689	21677	19.3	31	561	186
m10-p8-r3	1,2,5,6,7,8,9,10	8	5,6,9	231400	22375	37.1	31	748	186
m20-p3-r1	12,15,17	15	12	231823	52923	19.3	35	1682	840
m20-p3-r2	2,15,18	15	2,18	235908	67683	36.4	49	3531	1176
m20-p4-r1	6,12,15,17	17	12	245088	47038	45.5	41	2462	1230
m20-p4-r2	12,15,17,20	15	12,20	264298	52923	145.8	65	5853	1950
m20-p5-r1	2,12,15,17,18,	15	12	259851	50839	71.5	41	2708	1353
m20-p5-r2	12,18,20,17,15,	15	12,20	281613	50951	167.3	41	4062	1353
m20-p6-r1	2,6,12,15,17,18	17	12	273116	44954	195.6	76	5474	2736
m20-p6-r2	6,7,12,15,17,20	17	12,20	285195	46383	441.1	67	7239	2412
m20-p6-r3	9,11,12,15,17,20	15	11,12,20	307710	51272	582.8	41	5908	1476
m20-p7-r1	2,6,7,12,15,17,18	17	12	288619	44394	289.5	59	4604	2301
m20-p7-r2	6,7,12,15,17,18, 20	17	12,20	301938	44438	820.8	86	10065	3354
m20-p7-r3	9,11,12,15,17,18, 20	15	11,12,20	323630	49368	1121.9	57	8896	2223
m20-p8-r1	2,6,7,9,12,15,17, 18	none	12	307636	42811	454.9	68	5714	2856
m20-p8-r2	3,6,9,12,15,17,18, 20	15	12,20	322057	42969	519.6	43	5421	1806
m20-p8-r3	2,3,9,10,14,15,17, 18	10	3,9,17	337462	44187	1262.4	52	8740	2184
m30-p3-r1	8,12,26	12	26	341636	78336	91.8	47	3386	1692
m30-p3-r2	17,19,26	26	17,19	363017	108304	227.4	52	5619	1872

(Table B.5 continued)

Name	σ_p^*	σ_b^*	σ_r^*	Z_{def}^*	Z_{att}^*	time (s)	K_{iter}	K_{Cplex}	K_{sols}
m30-p4-r1	8,12,26,27	12	27	316491	69491	167.4	40	3602	1800
m30-p4-r2	8,12,19,26	12	19,26	360673	78336	703.0	42	5673	1890
m30-p5-r1	8,12,14,26,27	12	27	321468	64468	318.6	39	4214	2106
m30-p5-r2	12,14,19,26,30	12	14,30	365398	72886	745.3	40	6483	2160
m30-p6-r1	8,12,14,19,26,27	12	27	335838	63651	754.4	58	7310	3654
m30-p6-r2	7,8,12,19,26,27	12	7,27	345686	68673	1730.6	69	13044	4347
m30-p6-r3	8,12,14,19,20,26	12	19,20,26	389176	73313	3271.7	34	8572	2142
m30-p7-r1	7,8,12,14,15,26, 27	26	14	343513	59538	680.2	35	4832	2415
m30-p7-r2	7,8,12,17,19,26, 27	12	7,27	362791	67841	1353.9	41	8490	2829
m30-p7-r3	7,8,12,19,26,27, 30	12	7,8,27	401233	70533	4509.0	55	15184	3795
m30-p8-r1	7,8,12,13,14,15, 26,27	26	14	356123	59186	1431.5	49	7646	3822
m30-p8-r2	8,12,13,14,19,26, 27,28	12	13,27	370336	63236	2399.0	51	11937	3978
m30-p8-r3	1,7,8,12,13,26,27, 30	12	7,13,27	390404	68779	6516.3	65	20284	5070
m40-p3-r1	3,28,32	32	28	376467	102267	158.1	34	3266	1632
m40-p3-r2	3,28,32	32	3,28	405494	112694	269.6	38	5475	1824
m40-p4-r1	3,27,28,32	32	3	387842	101242	387.8	33	4160	2079
m40-p4-r2	3,27,28,32	32	27,28	394967	102267	1224.1	36	6807	2268
m40-p5-r1	1,2,17,32,36	32	2	391605	81068	817.7	38	5702	2850
m40-p5-r2	3,17,19,28,32	32	17,19	417402	100302	2369.6	60	13503	4500
m40-p6-r1	1,3,17,28,32,40	32	17	396331	80281	2026.4	57	10262	5130
m40-p6-r2	2,3,17,19,28,32	32	17,19	422492	91642	4959.1	87	23493	7830
m40-p6-r3	1,17,32,36,38,40	32	17,36,38	443774	92674	6849.6	37	13324	3330
m40-p7-r1	1,2,3,19,28,32,36	32	2	401740	67240	2002.3	38	7526	3762
m40-p7-r2	1,3,17,28,32,38, 40	32	17,38	429081	80281	5639.0	63	18714	6237
m40-p7-r3	3,14,17,19,22,28, 32	32	17,19,22	457811	93124	11951.3	59	23368	5841
m40-p8-r1	1,2,3,17,28,32,36, 38	32	2	409799	67424	5260.1	70	15542	7770
m40-p8-r2	1,3,17,22,28,32, 38,40	32	1,40	439784	74559	6815.1	60	19983	6660
m40-p8-r3	3,5,17,19,22,28, 29,32	32	3,5,28	467360	91822	9859.6	46	20428	5106
m50-p3-r1	24,26,27	27	24	482788	133788	421.2	38	4790	2394
m50-p3-r2	24,26,32	32	24,26	512723	160623	589.5	37	6996	2331

(Table B.5 continued)

Name	σ_p^*	σ_b^*	σ_r^*	Z_{def}^*	Z_{att}^*	time (s)	K_{iter}	K_{Cplex}	K_{sols}
m50-p4-r1	4,26,27,32	27	26	466003	119253	2178.6	77	12476	6237
m50-p4-r2	15,26,32,43	32	15,43	509593	154818	4943.0	59	14340	4779
m50-p5-r1	4,6,27,30,32	27	30	473880	103380	3373.6	62	12278	6138
m50-p5-r2	9,18,27,30,44	27	18,44	515504	127354	7067.1	69	20496	6831
m50-p6-r1	4,6,15,27,30,47	27	47	475878	92515	3160.3	35	7982	3990
m50-p6-r2	4,9,26,27,30,32	27	9,30	513339	114689	9740.6	69	23601	7866
m50-p6-r3	9,15,26,27,30,43	27	15,26,43	551779	127354	30946.9	84	38308	9576
m50-p7-r1	4,6,15,26,27,30, 47	27	47	482678	90366	9371.1	69	17804	8901
m50-p7-r2	4,9,26,27,30,32, 47	27	9,30	509746	104946	14123.6	71	27480	9159
m50-p7-r3	4,9,15,26,27,32, 50	27	9,15,50	545489	114689	22626.9	44	22708	5676
m50-p8-r1	4,6,9,15,26,27,39, 47	27	9	492744	84331	8450.2	42	12098	6048
m50-p8-r2	4,9,26,27,30,32, 39,47	27	9,30	518733	100183	22943.5	83	35859	11952
m50-p8-r3	4,15,26,27,30,32, 44,50	27	30,44,50	539921	111433	36790.8	68	39172	9792

Table B.6. Detailed test results of TS-BST using Config-3 on 75 high budget test instances

Name	σ_p^*	σ_b^*	σ_r^*	Z_{def}^*	Z_{att}^*	time (s)	K_{iter}	K_{Cplex}	K_{sols}
m10-p3-r1	8,9,10	8,9,10		137612	22775	3.8	37	1015	333
m10-p3-r2	8,9,10	8,9,10		137612	22775	7.5	37	2017	333
m10-p4-r1	1,2,5,8	5,8	2	152631	21206	7.4	33	1194	396
m10-p4-r2	1,7,8,10	1,8	7,10	153596	24109	15.2	32	2531	384
m10-p5-r1	1,2,5,6,8	2,6,8	1	164470	20270	18.9	54	2526	648
m10-p5-r2	1,5,7,8,10	1,5,8	7,10	165756	21206	27.4	31	3902	372
m10-p6-r1	1,2,5,7,8,10	5,8	1	179849	20199	16.9	36	1707	432
m10-p6-r2	1,5,6,7,8,10	1,5,8	7,10	180479	20379	45.3	36	4736	432
m10-p6-r3	1,5,6,7,8,10	1,6,8	5,7,10	187035	20610	104.8	43	11283	516
m10-p7-r1	1,2,5,6,7,8,10	6,8	1	194571	19371	19.5	44	1577	396
m10-p7-r2	1,2,5,6,7,8,10	1,6,8	7,10	196069	19544	54.9	43	4348	387
m10-p7-r3	1,2,5,6,7,8,10	1,6,8	2,7,10	203704	20379	96.8	38	8013	342
m10-p8-r1	1,2,5,6,7,8,9,10	6,8,10	9	212720	18645	11.6	31	748	186
m10-p8-r2	1,2,5,6,7,8,9,10	1,6,8	7,10	214676	19226	32.4	31	2054	186
m10-p8-r3	1,2,5,6,7,8,9,10	2,6,8	1,5,9	220632	19757	73.2	31	4537	186
m20-p3-r1	6,15,20	6,15, 20		222193	46818	30.7	48	3847	1152
m20-p3-r2	6,15,20	6,15, 20		222193	46818	47.2	48	7306	1152
m20-p4-r1	9,12,15,17	12,15	9	238869	46356	52.5	41	4018	1230
m20-p4-r2	2,15,17,18	15,17	2,18	235236	52923	99.9	35	8000	1050
m20-p5-r1	2,9,12,15,17	9,12, 15	2	251768	44705	105.5	49	6446	1617
m20-p5-r2	2,6,15,17,18	6,15, 17	2,18	249351	47038	290.2	48	17097	1584
m20-p6-r1	2,6,7,15,17,18	6,15, 17	7	259629	44954	114.9	36	5174	1296
m20-p6-r2	2,6,7,15,17,18	6,15, 17	2,18	264908	46383	588.3	67	26239	2412
m20-p6-r3	2,6,7,15,17,18	6,15, 17	2,7,18	270313	47038	1343.5	60	48584	2160
m20-p7-r1	2,6,9,12,15,17,18	6,12, 15	9	278712	39500	224.3	40	6181	1560
m20-p7-r2	2,9,12,15,17,18,20	9,15, 20	2,18	287009	42159	673.5	53	22555	2067
m20-p7-r3	2,6,10,12,14,15,18	6,10, 12	2,14,18	291986	35411	1017.7	35	30128	1365
m20-p8-r1	2,6,7,9,12,15,17,18	6,12, 15	9	294215	38940	402.3	52	8673	2184
m20-p8-r2	2,6,9,12,15,17,18, 20	6,9,15,20	2,18	299460	37835	752.7	44	21364	1848
m20-p8-r3	2,6,7,10,12,14,15, 18	6,10, 12	2,14,18	307543	34756	1748.3	46	44532	1932
m30-p3-r1	15,26,27	15,26,27		287015	68965	85.7	41	4687	1476
m30-p3-r2	15,26,27	15,26,27		287015	68965	226.4	41	9118	1476
m30-p4-r1	12,13,26,27	12,26	27	309950	69612	259.5	56	7914	2520

(Table B.6 continued)

Name	σ_p^*	σ_b^*	σ_r^*	Z_{def}^*	Z_{att}^*	time (s)	K_{iter}	K_{Cplex}	K_{sols}
m30-p4-r2	7,15,26,27	15,26,27	7	310203	68965	731.8	37	12099	1665
m30-p5-r1	12,13,14,26,27	12,14,26	27	312463	62125	571.0	63	13173	3402
m30-p5-r2	7,12,13,26,27	12,26,27	7,13	318863	68876	1289.2	51	28748	2754
m30-p6-r1	7,12,13,14,26,27	12,14,26	27	322646	61696	516.8	35	8521	2205
m30-p6-r2	4,7,12,13,26,27	12,26,27	4,13	332422	68322	2007.2	55	36694	3465
m30-p6-r3	4,7,12,13,26,27	12,26,27	4,7,13	337301	68876	7713.7	72	97178	4536
m30-p7-r1	7,12,13,14,26,27, 28	12,26	27	339226	60776	866.2	40	10440	2760
m30-p7-r2	4,7,12,13,14,26,27	12,14,26	7,27	348223	61998	2256.3	45	32747	3105
m30-p7-r3	7,8,12,13,14,26,27	12,14,26	7,13,27	351406	64468	6440.8	45	69196	3105
m30-p8-r1	7,8,12,13,14,26,27, 28	8,12, 26	27	351348	57898	1518.0	46	13440	3588
m30-p8-r2	7,8,12,13,14,26,27, 28	8,12, 26	14,28	360961	62123	2798.6	42	34395	3276
m30-p8-r3	7,8,12,13,14,19,26, 27	12,14,26	7,13,27	365776	63651	7555.0	40	72057	3120
m40-p3-r1	17,32,38	32,38	17	364870	93920	266.3	55	8115	2640
m40-p3-r2	17,32,38	32,38	17	364870	93920	450.2	55	16038	2640
m40-p4-r1	1,3,28,32	1,32	28	372397	85697	432.1	35	6744	2205
m40-p4-r2	5,17,19,40	5,19, 40	17	391938	100038	2452.5	61	25218	3843
m40-p5-r1	1,3,17,28,32	1,17, 32	28	375438	76988	823.7	35	9813	2625
m40-p5-r2	17,19,27,32,38	27,32,38	17,19	389634	82984	3793.1	78	55115	5850
m40-p6-r1	2,5,17,28,32,38	2,32, 38	17	382788	72738	1944.7	48	16099	4320
m40-p6-r2	1,3,17,27,28,32	1,17, 32	27,28	392938	76988	3749.7	48	41277	4320
m40-p6-r3	3,5,27,28,32,38	27,32,38	3,5,28	409434	82984	10769.1	53	90883	4770
m40-p7-r1	1,2,3,14,17,28,32	1,17, 32	2	392301	68251	2864.4	50	18063	4950
m40-p7-r2	2,3,17,27,28,32,38	2,32, 38	3,17	408147	73547	6321.8	60	56848	5940
m40-p7-r3	1,3,5,17,27,28,32	1,17, 32	5,27,28	420438	76988	14527.3	60	111356	5940
m40-p8-r1	1,2,3,19,28,32,38, 40	2,32, 38	19	403759	61609	3575.6	42	17372	4662
m40-p8-r2	3,17,19,27,28,32, 38,40	32,38,40	17,19	415856	70456	5510.4	37	39216	4107
m40-p8-r3	2,3,5,17,27,28,32, 38	2,32, 38	3,5,17	431497	73547	23323.4	69	148120	7659
m50-p3-r1	4,27,39	4,27, 39		427963	113313	495.9	52	10410	3276
m50-p3-r2	4,27,39	4,27, 39		427963	113313	842.1	52	20241	3276
m50-p4-r1	4,9,27,30	4,27	9	459706	104706	1920.3	62	15712	5022
m50-p4-r2	4,26,27,39	4,27, 39	26	460063	113313	3523.7	37	21134	2997
m50-p5-r1	4,9,15,26,27	4,9,27	15	459071	99984	2580.0	47	18060	4653

(Table B.6 continued)

Name	σ_p^*	σ_b^*	σ_r^*	Z_{def}^*	Z_{att}^*	time (s)	K_{iter}	K_{Cplex}	K_{sols}
m50-p5-r2	4,9,27,44,50	4,9,27	44,50	468185	104548	4860.4	36	37589	3564
m50-p6-r1	4,9,18,27,32,47	9,27, 47	18	466394	94494	6525.7	68	30536	7752
m50-p6-r2	4,6,15,26,27,43	4,6,27	15,43	478440	99965	7181.8	35	40330	3990
m50-p6-r3	4,9,18,27,44,50	4,9,27	18,44,50	485685	104548	24673.8	58	141508	6612
m50-p7-r1	4,9,15,26,27,32,47	9,27, 47	15	472768	89930	6022.5	42	21418	5418
m50-p7-r2	2,15,26,27,30,32, 50	27,30,32	2,15	488460	95872	12557.3	48	63607	6192
m50-p7-r3	2,4,9,15,26,27,43	4,9,27	2,15,43	498409	99984	42201.4	78	204207	10062
m50-p8-r1	4,15,26,27,30,32, 47,50	27,30,47	50	481593	86956	15164.7	68	38955	9792
m50-p8-r2	2,4,9,15,26,27,32, 50	4,9,27	2,15	499970	95445	16269.9	45	68520	6480
m50-p8-r3	2,4,9,15,26,27,32, 43	9,27, 32	2,15,43	510327	98152	33046.0	38	111084	5472

Table B.7. Detailed test results of SSM on 75 test instances with zero budget

Name	σ_p^{seq}	σ_b^{seq}	σ_r^{seq}	Z_{def}^{seq}	Z_{att}^{seq}	time (s)
m10-p3-r1	1,5,8	none	8	166232	28350	0.06
m10-p3-r2	1,5,8	none	1, 8	191290	36722	0.06
m10-p4-r1	1,5,6,8	none	8	180380	26380	0.06
m10-p4-r2	1,5,6,8	none	1, 8	205803	34603	0.06
m10-p5-r1	1,5,7,8,10	none	5	171325	23116	0.08
m10-p5-r2	1,5,7,8,10	none	5, 8	224039	27789	0.08
m10-p6-r1	1,5,6,7,8,10	none	8	205332	21182	0.08
m10-p6-r2	1,5,6,7,8,10	none	1, 8	227453	23116	0.11
m10-p6-r3	1,5,6,7,8,10	none	5, 6, 8	239589	27789	0.11
m10-p7-r1	1,2,5,6,7,8,10	none	8	223157	19696	0.08
m10-p7-r2	1,2,5,6,7,8,10	none	5, 6	206400	21254	0.14
m10-p7-r3	1,2,5,6,7,8,10	none	5, 6, 8	257414	23915	0.08
m10-p8-r1	1,2,5,6,7,8,9,10	none	8	239528	18601	0.08
m10-p8-r2	1,2,5,6,7,8,9,10	none	5, 6	223689	20740	0.16
m10-p8-r3	1,2,5,6,7,8,9,10	none	5, 6, 8	268034	21951	0.09
m20-p3-r1	12,15,17	none	15	294226	57571	0.31
m20-p3-r2	12,15,17	none	15, 17	336064	27569	0.28
m20-p4-r1	6,12,15,17	none	17	250801	35815	0.80
m20-p4-r2	6,12,15,17	none	6, 15	315076	26380	0.78
m20-p5-r1	2,12,15,17,18	none	15	325883	34603	0.36
m20-p5-r2	2,12,15,17,18	none	12, 17	301589	22650	0.61
m20-p6-r1	2,6,12,15,17,18	none	17	280207	27789	0.47
m20-p6-r2	2,6,12,15,17,18	none	12, 17	313572	21182	0.52
m20-p6-r3	2,6,12,15,17,18	none	6, 12, 17	346839	23116	1.70
m20-p7-r1	2,6,7,12,15,17,18	none	17	295710	27789	0.42
m20-p7-r2	2,6,7,12,15,17,18	none	12, 17	329074	20907	0.47
m20-p7-r3	2,6,7,12,15,17,18	none	12, 15, 17	375042	22375	0.72
m20-p8-r1	2,6,7,9,12,15,17,18	none	12	307636	27514	2.73
m20-p8-r2	2,6,7,9,12,15,17,18	none	12, 17	341938	20466	2.70
m20-p8-r3	2,6,7,9,12,15,17,18	none	9, 12, 17	350624	21677	2.52
m30-p3-r1	12,14,26	none	26	368768	25922	0.97
m30-p3-r2	12,14,26	none	12, 26	507975	70626	0.98
m30-p4-r1	12,14,26,27	none	26	358212	86864	0.98
m30-p4-r2	12,14,26,27	none	12, 26	451844	47501	1.42

(Table B.7 continued)

Name	σ_p^{seq}	σ_b^{seq}	σ_r^{seq}	Z_{def}^{seq}	Z_{att}^{seq}	time (s)
m30-p5-r1	8,12,14,26,27	none	12	382324	70626	0.98
m30-p5-r2	8,12,14,26,27	none	12, 27	418560	54196	1.08
m30-p6-r1	7,8,12,14,26,27	none	12	396686	65514	1.20
m30-p6-r2	7,8,12,14,26,27	none	12, 26	447919	45332	2.58
m30-p6-r3	7,8,12,14,26,27	none	8, 12, 26	488646	58347	3.00
m30-p7-r1	7,8,12,13,14,26,27	none	12	407494	65514	1.03
m30-p7-r2	7,8,12,13,14,26,27	none	12, 14	437134	44772	2.19
m30-p7-r3	7,8,12,13,14,26,27	none	8, 12, 26	503712	57787	3.03
m30-p8-r1	7,8,12,13,14,19,26,27	none	12	421864	63030	1.13
m30-p8-r2	7,8,12,13,14,19,26,27	none	12, 14	451504	42811	1.31
m30-p8-r3	7,8,12,13,14,19,26,27	none	8, 12, 14	480916	46925	2.41
m40-p3-r1	1,28,32	none	32	506238	57787	2.80
m40-p3-r2	1,28,32	none	1, 32	613520	94268	2.53
m40-p4-r1	1,28,32,38	none	32	473146	150025	2.36
m40-p4-r2	1,28,32,38	none	28, 32	549332	80062	3.03
m40-p5-r1	1,2,28,32,38	none	32	454724	102844	2.33
m40-p5-r2	1,2,28,32,38	none	32, 38	520015	75924	2.50
m40-p6-r1	1,2,3,28,32,38	none	32	464293	90710	2.56
m40-p6-r2	1,2,3,28,32,38	none	32, 38	529377	72136	2.55
m40-p6-r3	1,2,3,28,32,38	none	1, 32, 38	599806	81094	2.69
m40-p7-r1	1,2,3,17,28,32,38	none	32	475270	98971	2.47
m40-p7-r2	1,2,3,17,28,32,38	none	1, 32	517534	70694	3.00
m40-p7-r3	1,2,3,17,28,32,38	none	1, 32, 38	574502	79234	2.83
m40-p8-r1	1,2,3,17,22,28,32,38	none	32	487230	95612	2.81
m40-p8-r2	1,2,3,17,22,28,32,38	none	1, 32	529494	69877	2.95
m40-p8-r3	1,2,3,17,22,28,32,38	none	1, 32, 38	587729	78416	3.72
m50-p3-r1	4,6,27	none	27	574621	83591	5.39
m50-p3-r2	4,6,27	none	4, 27	683885	125688	6.81
m50-p4-r1	4,27,30,47	none	27	569092	183570	6.06
m50-p4-r2	4,27,30,47	none	4, 27	748312	103196	4.94
m50-p5-r1	4,26,27,30,47	none	27	575768	140832	5.13
m50-p5-r2	4,26,27,30,47	none	4, 27	693583	90324	5.84
m50-p6-r1	4,6,26,27,30,47	none	27	566475	112815	5.94
m50-p6-r2	4,6,26,27,30,47	none	4, 27	636905	88643	5.30

(Table B.7 continued)

Name	σ_p^{seq}	σ_b^{seq}	σ_r^{seq}	Z_{def}^{seq}	Z_{att}^{seq}	**time (s)**
m50-p6-r3	4,6,26,27,30,47	none	4, 6, 27	722133	110927	5.91
m50-p7-r1	4,6,26,27,30,39,47	none	27	550223	151556	5.03
m50-p7-r2	4,6,26,27,30,39,47	none	4, 27	617956	87120	7.91
m50-p7-r3	4,6,26,27,30,39,47	none	4, 6, 27	694206	99584	7.17
m50-p8-r1	4,6,15,26,27,30,39,47	none	27	561382	123802	10.50
m50-p8-r2	4,6,15,26,27,30,39,47	none	27, 39	599184	81180	11.50
m50-p8-r3	4,6,15,26,27,30,39,47	none	4, 6, 27	698378	93644	9.64

Table B.8. Detailed test results of SSM on 75 test instances with low budget

Name	σ_p^{seq}	σ_b^{seq}	σ_r^{seq}	Z_{def}^{seq}	Z_{att}^{seq}	time (s)
m10-p3-r1	1,5,8	8	1	141874	24374	0.06
m10-p3-r2	1,5,8	8	1,5	158183	28058	0.08
m10-p4-r1	1,5,6,8	8	1	156596	23546	0.06
m10-p4-r2	1,5,6,8	8	1,6	170124	24374	0.06
m10-p5-r1	1,5,7,8,10	none	5	171325	22650	0.08
m10-p5-r2	1,5,7,8,10	8	1,5	192486	25311	0.08
m10-p6-r1	1,5,6,7,8,10	8	1	189879	20842	0.08
m10-p6-r2	1,5,6,7,8,10	1	5,6	188575	22650	0.16
m10-p6-r3	1,5,6,7,8,10	8	1,5,6	209736	25311	0.16
m10-p7-r1	1,2,5,6,7,8,10	8	6	200248	19473	0.08
m10-p7-r2	1,2,5,6,7,8,10	none	5,6	206400	22375	0.22
m10-p7-r3	1,2,5,6,7,8,10	8	1,5,6	213443	23568	0.14
m10-p8-r1	1,2,5,6,7,8,9,10	8	6	218855	19155	0.09
m10-p8-r2	1,2,5,6,7,8,9,10	none	5,6	223689	21677	0.22
m10-p8-r3	1,2,5,6,7,8,9,10	8	5,6,9	231400	22375	0.16
m20-p3-r1	12,15,17	15	12	231823	52923	0.30
m20-p3-r2	12,15,17	15	12,17	272183	67683	0.33
m20-p4-r1	6,12,15,17	17	12	245088	47038	0.78
m20-p4-r2	6,12,15,17	15	12,17	284166	60516	0.80
m20-p5-r1	2,12,15,17,18	15	12	259851	50839	0.38
m20-p5-r2	2,12,15,17,18	none	12,17	301589	65514	0.67
m20-p6-r1	2,6,12,15,17,18	17	12	273116	44954	0.48
m20-p6-r2	2,6,12,15,17,18	none	12,17	313572	58347	0.58
m20-p6-r3	2,6,12,15,17,18	none	6,12,17	346839	65514	1.80
m20-p7-r1	2,6,7,12,15,17,18	17	12	288619	44394	0.56
m20-p7-r2	2,6,7,12,15,17,18	none	12,17	329074	57787	0.61
m20-p7-r3	2,6,7,12,15,17,18	12	6,7,15	363696	54196	1.02
m20-p8-r1	2,6,7,9,12,15,17,18	none	12	307636	42811	2.63
m20-p8-r2	2,6,7,9,12,15,17,18	12	9,17	317260	44772	2.98
m20-p8-r3	2,6,7,9,12,15,17,18	none	9,12,17	350624	57787	3.20
m30-p3-r1	12,14,26	none	26	368768	94268	0.88
m30-p3-r2	12,14,26	12	14,26	410484	110334	1.14
m30-p4-r1	12,14,26,27	none	26	358212	80062	1.03
m30-p4-r2	12,14,26,27	12	14,26	399118	95318	1.47

(Table B.8 continued)

Name	σ_p^{seq}	σ_b^{seq}	σ_r^{seq}	Z_{def}^{seq}	Z_{att}^{seq}	time (s)
m30-p5-r1	8,12,14,26,27	12	27	321468	64468	1.09
m30-p5-r2	8,12,14,26,27	12	8,26	379412	80062	1.17
m30-p6-r1	7,8,12,14,26,27	12	26	353493	63618	1.20
m30-p6-r2	7,8,12,14,26,27	12	8,26	391783	79508	2.92
m30-p6-r3	7,8,12,14,26,27	12	8,14,26	426489	94764	3.31
m30-p7-r1	7,8,12,13,14,26,27	12	26	365559	63246	1.11
m30-p7-r2	7,8,12,13,14,26,27	12	8,26	404154	79129	2.64
m30-p7-r3	7,8,12,13,14,26,27	12	8,14,26	438859	94384	3.25
m30-p8-r1	7,8,12,13,14,19,26,27	12	14	357381	61306	1.25
m30-p8-r2	7,8,12,13,14,19,26,27	12	8,14	385380	65518	1.42
m30-p8-r3	7,8,12,13,14,19,26,27	12	8,19,26	419341	79129	3.42
m40-p3-r1	1,28,32	32	1	402831	101681	2.49
m40-p3-r2	1,28,32	32	1,28	452394	112694	2.92
m40-p4-r1	1,28,32,38	32	38	394660	85110	2.73
m40-p4-r2	1,28,32,38	32	1,38	443431	101681	2.92
m40-p5-r1	1,2,28,32,38	32	38	400230	76930	2.48
m40-p5-r2	1,2,28,32,38	32	1,38	445801	92701	2.56
m40-p6-r1	1,2,3,28,32,38	32	38	410421	75871	2.59
m40-p6-r2	1,2,3,28,32,38	32	1,38	455992	91642	2.72
m40-p6-r3	1,2,3,28,32,38	32	1,2,38	486652	100302	2.88
m40-p7-r1	1,2,3,17,28,32,38	32	1	405446	71396	2.14
m40-p7-r2	1,2,3,17,28,32,38	32	1,38	447119	80319	3.33
m40-p7-r3	1,2,3,17,28,32,38	32	1,17,38	469242	91642	3.47
m40-p8-r1	1,2,3,17,22,28,32,38	32	1	419424	65899	2.59
m40-p8-r2	1,2,3,17,22,28,32,38	32	1,2	450084	74559	3.25
m40-p8-r3	1,2,3,17,22,28,32,38	32	1,2,38	487915	82840	4.27
m50-p3-r1	4,6,27	27	4	512986	123986	6.67
m50-p3-r2	4,6,27	27	4,6	615464	151414	6.78
m50-p4-r1	4,27,30,47	27	4	536079	121229	5.78
m50-p4-r2	4,27,30,47	27	4,30	592669	141669	5.59
m50-p5-r1	4,26,27,30,47	27	30	481025	107725	5.17
m50-p5-r2	4,26,27,30,47	27	4,30	565443	124043	6.14
m50-p6-r1	4,6,26,27,30,47	27	4	499496	95096	5.98
m50-p6-r2	4,6,26,27,30,47	27	4,26	535248	109998	5.77

(Table B.8 continued)

Name	$\boldsymbol{\sigma}_p^{seq}$	$\boldsymbol{\sigma}_b^{seq}$	$\boldsymbol{\sigma}_r^{seq}$	Z_{def}^{seq}	Z_{att}^{seq}	**time (s)**
m50-p6-r3	4,6,26,27,30,47	27	4,6,30	596493	124043	6.13
m50-p7-r1	4,6,26,27,30,39,47	27	4	508483	90333	5.30
m50-p7-r2	4,6,26,27,30,39,47	27	4,26	544235	105235	7.64
m50-p7-r3	4,6,26,27,30,39,47	27	4,6,30	605480	119280	8.83
m50-p8-r1	4,6,15,26,27,30,39,47	27	30	489994	84331	9.66
m50-p8-r2	4,6,15,26,27,30,39,47	27	6,30	547824	99662	12.61
m50-p8-r3	4,6,15,26,27,30,39,47	27	4,6,30	604404	109429	9.39

Table B.9. Detailed test results of SSM on 75 test instances with high budget

Name	$\boldsymbol{\sigma}_p^{seq}$	$\boldsymbol{\sigma}_b^{seq}$	$\boldsymbol{\sigma}_r^{seq}$	Z_{def}^{seq}	Z_{att}^{seq}	time (s)
m10-p3-r1	1,5,8	1,8	5	141584	24109	0.05
m10-p3-r2	1,5,8	1,8	5	141584	24109	0.06
m10-p4-r1	1,5,6,8	1,8	6	152681	21206	0.08
m10-p4-r2	1,5,6,8	1,8	5,6	158834	24109	0.08
m10-p5-r1	1,5,7,8,10	1,5,8	10	169036	20161	0.08
m10-p5-r2	1,5,7,8,10	1,5,8	7,10	165756	21206	0.11
m10-p6-r1	1,5,6,7,8,10	1,8	6	182422	19747	0.09
m10-p6-r2	1,5,6,7,8,10	1,5,8	7,10	180479	20379	0.20
m10-p6-r3	1,5,6,7,8,10	1,6,8	5,7,10	187035	20610	0.27
m10-p7-r1	1,2,5,6,7,8,10	6,8	1	194571	19371	0.09
m10-p7-r2	1,2,5,6,7,8,10	1,6,8	7,10	196069	19544	0.27
m10-p7-r3	1,2,5,6,7,8,10	1,5,8	2,7,10	203704	20379	0.28
m10-p8-r1	1,2,5,6,7,8,9,10	6,8,10	9	212720	18645	0.13
m10-p8-r2	1,2,5,6,7,8,9,10	1,6,8	7,10	214676	19226	0.33
m10-p8-r3	1,2,5,6,7,8,9,10	2,6,8	1,5,9	220632	19757	0.41
m20-p3-r1	12,15,17	15	12	231823	52923	0.30
m20-p3-r2	12,15,17	15,17	12	231823	52923	0.31
m20-p4-r1	6,12,15,17	17	12	245088	47038	0.80
m20-p4-r2	6,12,15,17	15,17	6,12	277073	52923	0.80
m20-p5-r1	2,12,15,17,18	15	12	259851	50839	0.38
m20-p5-r2	2,12,15,17,18	15,17,18	2,12	263038	50951	0.72
m20-p6-r1	2,6,12,15,17,18	17	12	273116	44954	0.53
m20-p6-r2	2,6,12,15,17,18	6,15,17	12,18	288774	45512	0.63
m20-p6-r3	2,6,12,15,17,18	6,15,17	2,12,18	284701	47038	1.95
m20-p7-r1	2,6,7,12,15,17,18	17	12	288619	44394	0.58
m20-p7-r2	2,6,7,12,15,17,18	6,15,17	7,12	294979	44954	0.66
m20-p7-r3	2,6,7,12,15,17,18	6,15,17	2,12,18	300258	46383	1.14
m20-p8-r1	2,6,7,9,12,15,17,18	6,12,15	9	294215	38940	2.88
m20-p8-r2	2,6,7,9,12,15,17,18	9,12,15	6,7	316814	42689	3.03
m20-p8-r3	2,6,7,9,12,15,17,18	6,15,17	7,9,12	326379	44954	3.27
m30-p3-r1	12,14,26	12,26	14	309038	75338	1.02
m30-p3-r2	12,14,26	12,26	14	309038	75338	1.00
m30-p4-r1	12,14,26,27	12,26	14	315076	68876	0.84
m30-p4-r2	12,14,26,27	12,26	14,27	342988	75338	1.42

(Table B.9 continued)

Name	σ_p^{seq}	σ_b^{seq}	σ_r^{seq}	Z_{def}^{seq}	Z_{att}^{seq}	time (s)
m30-p5-r1	8,12,14,26,27	12,26,27	14	318029	63029	1.14
m30-p5-r2	8,12,14,26,27	8,12,26	14,27	345941	69491	1.41
m30-p6-r1	7,8,12,14,26,27	12,26	14	330400	62475	1.19
m30-p6-r2	7,8,12,14,26,27	8,12,26	7,27	336293	64468	2.86
m30-p6-r3	7,8,12,14,26,27	8,12,26	7,14,27	360766	69491	3.63
m30-p7-r1	7,8,12,13,14,26,27	12,26	14	343011	62123	1.19
m30-p7-r2	7,8,12,13,14,26,27	8,12,26	14,27	358736	63336	2.83
m30-p7-r3	7,8,12,13,14,26,27	8,12,26	7,13,27	351406	64468	3.86
m30-p8-r1	7,8,12,13,14,19,26,27	12,14	8	356698	58498	1.36
m30-p8-r2	7,8,12,13,14,19,26,27	12,14,26	8,19	366408	60483	1.91
m30-p8-r3	7,8,12,13,14,19,26,27	8,12,26	7,13,27	365776	63651	3.97
m40-p3-r1	1,28,32	1,32	28	386424	96124	2.64
m40-p3-r2	1,28,32	1,32	28	386424	96124	2.98
m40-p4-r1	1,28,32,38	32,38	28	388623	83323	2.45
m40-p4-r2	1,28,32,38	32,38	1,28	431420	93920	3.27
m40-p5-r1	1,2,28,32,38	32,38	28	391776	74826	2.84
m40-p5-r2	1,2,28,32,38	32,38	1,28	431373	84623	2.97
m40-p6-r1	1,2,3,28,32,38	32,38	1	394469	72919	2.58
m40-p6-r2	1,2,3,28,32,38	2,32,38	1,28	420099	74849	2.81
m40-p6-r3	1,2,3,28,32,38	2,32,38	1,3,28	444423	84623	3.92
m40-p7-r1	1,2,3,17,28,32,38	1,32	2	395709	69459	2.41
m40-p7-r2	1,2,3,17,28,32,38	1,32	17,38	423671	75871	3.34
m40-p7-r3	1,2,3,17,28,32,38	1,17,32	2,3,28	445272	81522	4.06
m40-p8-r1	1,2,3,17,22,28,32,38	1,32	2	409687	63962	2.84
m40-p8-r2	1,2,3,17,22,28,32,38	1,32	2,22	438084	69459	3.48
m40-p8-r3	1,2,3,17,22,28,32,38	1,32	17,22,38	454246	75871	4.98
m50-p3-r1	4,6,27	4,27	6	512984	122034	6.38
m50-p3-r2	4,6,27	4,27	6	512984	122034	7.61
m50-p4-r1	4,27,30,47	4,27	30	474339	112289	6.08
m50-p4-r2	4,27,30,47	4,27	30,47	512584	122034	5.50
m50-p5-r1	4,26,27,30,47	4,27,30	47	465792	100142	5.08
m50-p5-r2	4,26,27,30,47	4,27,30	26,47	491206	104706	5.59
m50-p6-r1	4,6,26,27,30,47	4,27,47	30	469762	92312	6.05
m50-p6-r2	4,6,26,27,30,47	4,6,27	30,47	503215	99965	6.06

(Table B.9 continued)

Name	$\boldsymbol{\sigma}_p^{seq}$	$\boldsymbol{\sigma}_b^{seq}$	$\boldsymbol{\sigma}_r^{seq}$	Z_{def}^{seq}	Z_{att}^{seq}	**time (s)**
m50-p6-r3	4,6,26,27,30,47	4,6,27	26,30,47	528629	104529	7.28
m50-p7-r1	4,6,26,27,30,39,47	4,27	30	478749	87549	5.38
m50-p7-r2	4,6,26,27,30,39,47	4,27,30	39,47,	502619	93569	7.97
m50-p7-r3	4,6,26,27,30,39,47	4,6,27	30,39,47	534315	99965	8.30
m50-p8-r1	4,6,15,26,27,30,39,47	27	30	489994	84331	9.70
m50-p8-r2	4,6,15,26,27,30,39,47	4,6,27	39,47	513778	90366	12.23
m50-p8-r3	4,6,15,26,27,30,39,47	4,6,27	30,39,47	545560	96747	12.59

REFERENCES

1. Ben-Ayed, O. and C. E. Blair, "Computational Difficulties of Bilevel Linear Programming", *Operations Research*, Vol. 38, No. 3, pp. 556-560, May-June 1990.

2. Owen, S. H. and M. S. Daskin, "Strategic Facility Location: A Review", *European Journal of Operational Research*, Vol. 111, No. 3, pp. 423–447, December 1998.

3. Drezner, Z., "Heuristic Solution Methods for Two Location Problems With Unreliable Facilities", *Journal of Operational Research Society*, Vol. 38, No. 6, pp. 509–514, June 1987.

4. Snyder, L. V. and M. S. Daskin, "Reliability Models for Facility Location: The Expected Failure Cost Case", *Transportation Science*, Vol. 39, No. 3, pp. 400–416, August 2005.

5. Berman, O., Krass D. and Menezes M. B. C., "Facility Reliability Issues in Network p-Median Problems: Strategic Centralization and Co-Location Effects", *Operations Research*, Vol. 55, No. 2, pp. 332–350, March–April 2007.

6. Snyder, L. V. and Daskin M. S., "Models for Reliable Supply Chain Network Design" in A. T. Murray and T. H. Grubesic (eds.), *Advances in Spatial Science, Critical Infrastructure Reliability and Vulnerability*, pp. 257–289, Springer, Berlin Heidelberg, 2007.

7. O'Hanley, J. and R. L. Church, "Designing Robust Coverage Networks to Hedge Against Worst-Case Facility Losses", Working Paper No. 173, ISSN 1748-7595, 2008.

8. Brown, G. G., W. M. Carlyle, J. Salmeron and K. Wood, "Analyzing The Vulnerability of Critical Infrastructure to Attack and Planning Defenses" in H. J. Greenberg and J. C. Smith (eds.), *Tutorials in Operations Research: Emerging Theory, Methods, and Applications*, INFORMS, Hanover, 2005.

9. Berman, O., T. Drezner, Z. Drezner, G. O. Wesolowsky, "A Defensive Maximal Covering Problem on a Network", *International Transactions in Operational Research*, Vol. 16., No. 1, pp. 69–86, January 2009.

10. Matisziw, T., A. Murray and T.Grubesic, "Exploring The Vulnerability of Network Infrastructure to Disruption", *The Annals of Regional Science*, Vol. 43, No. 2, pp. 307-321, June 2009.

11. Snyder, L. V., M. P. Scaparra, M. S. Daskin and R. L. Church, "Planning for Disruptions in Supply Chain Networks" in H. K. Greenberg (ed.), *Tutorials in Operations Research*, pp. 234–257, INFORMS, Baltimore, 2006.

12. Wollmer, R., "Removing Arcs From a Network", *Operations Research*, Vol. 12, No. 6, pp. 934–940, Nov/Dec 1964.

13. Church, R. L., M. P. Scaparra, R. S. Middleton, "Identifying Critical Infrastructure: The Median and Covering Facility Interdiction Problems", *Annals of the Association of American Geographers*, Vol. 94, No. 3, pp. 491–502, 2004.

14. Lim, C. and J. C. Smith, "Algortihms for Discrete and Continuous Multicommodity Flow Network Interdiction Problems",*IIE Transactions*, Vol. 39, No. 1, pp.15-26, January 2007.

15. Smith, J. C., C. Lim, F. Sudargho, "Survivable Network Design Under Optimal and Heuristic Interdiction Scenarios", *Journal of Global Optimization*, Vol. 38, No.2, pp. 181–199, June 2007.

16. Matisziw, T. C. and A. T. Murray, "Modeling s–t Path Availability to Support Disaster Vulnerability Assessment of Network Infrastructure" *Computers and Operations Research*, Vol. 36, No.1, pp. 16-26, 2009.

17. Church, R. L. and M. P. Scaparra, "Protecting Critical Assets: The *r*-interdiction Median Problem With Fortification", *Geographical Analysis*, Vol. 39, No. 2, pp. 129–146, 2007.

18. Scaparra, M. P. and R. L. Church, "An Exact Solution Approach for The Interdiction Median Problem With Fortification", *European Journal of Operational Research*, Vol. 189, No. 1, pp. 76–92, August 2008.

19. Scaparra, M. P. and R. L. Church, "A Bilevel Mixed-integer Program for Critical Infrastructure Protection Planning", *Computers and Operations Research*, Vol. 35, No. 6, pp. 1905–1923, June 2008.

20. Aksen, D., N. Piyade, N. Aras, "A Defender-attacker Game for the Budget Constrained *r*-interdiction Median Problem With Capacity Expansion", Technical Report, 2008.

21. O'Hanley, J. and R. L. Church, "Designing Robust Coverage Networks to Hedge Against Worst-Case Facility Losses", Working Paper No. 173, ISSN 1748-7595, 2008.

22. Liberatore, F., M. P. Scaparra and M.S. Daskin, "Analysis of Facility Protection Strategies Against Uncertain Numbers of Attacks: The Stochastic r-interdiction Median Problem With Fortification", Working Paper No. 176, ISSN 1748-7595, 2008.

23. Lu, J., C. Shi and G. Zhang, "On Bilevel Multi-follower Decision Making: General Framework and Solutions", *Information Sciences*, Vol. 176, No. 11, pp. 1607-1627, June 2006.

24. Ben-Ayed, O., "Bilevel Linear Programming", *Computers and Operations Research*, Vol. 20, No. 5, pp. 485–501, June 1993.

25. Bracken, J. and J. T. McGill, “Mathematical programs with optimization problems in the constraints”, *Operations Research*, Vol. 21, No. 1, pp. 37-44, January–February 1973.

26. Vicente, L. N. and P. Calamai, “Bilevel and Multilevel Programming: A Bibliography Review”, *Journal of Global Optimization*, Vol. 5, No. 3, pp. 291–306, October 1994.

27. Kolstad, C. D., “A Review of The Literature on Bi-level Mathematical Programming”, Technical Report, LA-10284MS, US-32, Los Alamos National Laboratory, 1985.

28. Wen, U. and S. Hsu, “Linear Bi-level Programming Problems - A Review”, *Journal of the Operational Research Society*, Vol. 42, pp. 125–133, 1991.

29. Vicente, L. N., “Bilevel Programming: Introduction, History and Overview”, in C. A. Floudas and P. M. Pardalos (eds.), *Encyclopedia of Optimization*, Vol. 1, pp. 178-180, Kluwer Academic Publishers, Dordrecht, 2001.

30. Colson, B., P. Marcotte and G. Savard, “An Overview of Bilevel Optimization”, *Annals of Operations Research*, Vol. 153, No. 1, pp. 235-256, September 2007.

31. Bard, J. F., *Practical Bilevel Optimization: Algorithms and Applications. Nonconvex Optimization and Its Applications*, Vol. 30, Kluwer Academic, Dordrecht, 1999.

32. Dempe, S., *Foundations of Bilevel Programming. Nonconvex optimization and Its Applications*, Vol. 61, Kluwer Academic, Dordrecht, 2002.

33. Moore, J. T. and J. F. Bard, “The Mixed-integer Linear Bilevel Programming Problem”, *Operations Research*, Vol. 38, No. 5, pp. 911–921, September-October 1990.

34. Wen, U. P., Y. H. Yang, “Algorithms For Solving The Mixed-integer Two-level Linear Programming Problem”, *Computers and Operations Research*, Vol. 17, No. 2, pp. 133–142, 1990.

35. Wen, U. P., A. D. Huang, “A Simple Tabu Search Method to Solve The Mixed-integer Linear Bilevel Programming Problem”, *European Journal of Operational Research*, Vol. 88, No. 3, pp. 563–571, February 1996.

36. Dempe, S., “Discrete Bilevel Optimization Problems”, Technical Report, D-04109 Leipzig, Institut fur Wirtschaftsinformatik, Universitat Leipzig, Germany, 2001.

37. Dempe, S. and V. Kalashnikov, “Discrete Bilevel Programming: Application to a Gas Shipper's Problem”, Preprint Nr. 2002-02, TU Bergakademie Freiberg, Fakultät für Mathematik und Informatik, 2002.

38. Gümüş, Z. H. and C. A. Floudas, “Global Optimization of Mixed-integer Bilevel Programming Problems”, *Computational Management Science*, Vol. 2, No. 3, pp. 181–212, July 2005.

39. DeNegre S. T. and T. K. Ralphs, “Branch-and-cut Algorithm for Integer Bilevel Linear Programs” in J. W. Chinneck, B. Kristjansson and M. Saltzman (eds.), *Operations Research and Cyber-Infrastructure. Operations Research/Computer Science Interfaces Series*, Vol. 47, No. 2, Springer, 2009.

40. Hejazi, S. R., A. Memariani, G. Jahanshahloo and M. M. Sepehri, “Linear Bilevel Programming Solution by Genetic Algorithm”, *Computers and Operations Research*, Vol. 29, No. 13, pp. 1913-1925, November 2002.

41. Nishizaki, I. and M. Sakawa, “Computational Methods through Genetic Algorithms for Obtaining Stackelberg Solutions to Two-Level Integer Programming Problems”, *Cybernetics and Systems: An International Journal*, Vol. 36, No. 6, pp. 565–579, September 2005.

42. Calvete, H. I., C. Gale and P. M. Mateo, “A New Approach for Solving Linear Bilevel Problems Using Genetic Algorithms”, *European Journal of Operational Research*, Vol. 188, No. 1, pp. 14–28, July 2008.

43. Teixeira, J.C. and A. P. Antunes, "A Hierarchical Location Model for Public Facility Planning", *European Journal of Operational Research*, Vol. 185, No. 1, pp. 92–104, 2008.

44. Church, R. L. and J. L. Cohon, "Multiobjective Location Analysis of Regional Energy Facility Sitting Problems", Report prepared for the US Energy Research and Development Administration (BNL 50567), 1976.

45. Glover, F. and M. Laguna, *Tabu Search*, Kluwer Academic Publishers, Dordrecht, 1997.

46. Rolland, E., D. Schilling and J. R. Current, "An Efficient Tabu Search Procedure for the *p*-median Problem", *European Journal of Operational Research*, Vol. 96, No. 2, pp. 329–342, January 1997.

47. Aras, N. and D. Aksen, "Locating Collection Centers for Distance and-Incentive-dependent Returns", *International Journal of Production Economics*, Vol. 111, No. 2, pp. 316–333, February 2008.

48. Aras, N., D. Aksen and A. G. Tanuğur, "Locating Collection Centers for Incentive-dependent Returns Under a Pick-up Policy With Capacitated Vehicles", *European Journal of Operational Research*, Vol. 191, No. 3, pp. 1223–1240, December 2008.

49. Hakimi, S. L., "Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph", *Operations Research*, Vol. 12, No. 3, pp. 450– 459, May – June 1964.

50. Reese, J., "Solution Methods for the *p*-median Problem: An Annotated Bibliography", *Networks*, Vol. 48, No. 3, pp. 125–142, 2006.