Journal of Computer and System Sciences 115 (2021) 54–63

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Attack and defense in the layered cyber-security model and their $(1 \pm \epsilon)$-approximation schemes

Supachai Mukdasanit [a], Sanpawat Kantabutra [b,*]

[a] *PhD Program in Computer Engineering, Department of Computer Engineering, Faculty of Engineering, Chiang Mai University, Chiang Mai, 50200, Thailand*

[b] *Thailand Research Fund and The Theory of Computation Group, Department of Computer Engineering, Faculty of Engineering, Chiang Mai University, 50200, Thailand*

ARTICLE INFO

Article history:
Received 22 July 2019
Received in revised form 21 May 2020
Accepted 5 July 2020
Available online 21 July 2020

Keywords:
Algorithms
Complexity
Graph theory

ABSTRACT

Let $M = (T, C, P)$ be a security model, where T is a rooted tree, C is a multiset of costs and P is a multiset of prizes and let (T, c, p) be a security system, where c and p are bijections of costs and prizes. The problems of computing an optimal attack on a security system and of determining an edge $e \in E(T)$ such that the maximum sum of prizes obtained from an optimal attack in (T, c, p) is minimized when $c(e) = \infty$ are considered. An $O(G^2n)$-time algorithm to compute an optimal attack as well as an $O(G^2n^2)$-time algorithm to determine such an edge are given, in addition to a (1-ϵ) FPTAS with the time bound $O(\frac{1}{\epsilon^2}n^3 \log G)$ and a (1+ϵ) FPTAS with the time bound $O(\frac{1}{\epsilon^2}n^4 \log G)$ for the first and second problems, respectively.

1. Introduction

In today's world it is not an overstatement to say that we live in a cyber space. We do our business transactions in cyber space. We communicate in social media and basically spend most minutes in cyber space. Unfortunately, criminals also find their ways into cyber space. The cyber crimes statistics in 2017 showed that 254 benchmarked companies have experienced a total of 635 discernible cyber attacks and the average number of successful attacks per a company each week was 2.5 [1]. Moreover, there have also been some other large scale attacks such as WannaCry or NotPetya [2]. In short, cyber crimes and other cyber activities with malicious intent continue to be very costly for organizations and nations. In terms of dollars Richards et al. [1] reported that the mean annualized cost of cyber crimes for 254 benchmarked companies in 2017 was 11.7 million US dollars per year, increasing 22.7 percent from 2016 and 62 percent from the average cost over five years. Therefore, we need to understand these critical cyber security issues. Although cyber crimes have risen rapidly, cyber security has not been able to catch up. There is an urgent need to improve security and to understand the cyber environment. For this reason, Schneider suggested the need for the science of cyber security [3]. Additionally, Dunlavy et al. [4] discussed that mathematics must play a key role in improving our understanding and helping to design alternatives. Armstrong et al. also explained that complexity science is one of the tools that can be utilized to understand problems in the cyber security domain [5]. Since then, there have been some new researches in the science of cyber security.

Measuring cybersecurity is difficult but organizations need to know how effective their cybersecurity systems are. For example, chief security officers of Fortune 500 companies listed security metrics among their top three needs [6]. To respond

* Corresponding author.
E-mail addresses: supachai_m@cmu.ac.th (S. Mukdasanit), sanpawat@alumni.tufts.edu (S. Kantabutra).

https://doi.org/10.1016/j.jcss.2020.07.001
0022-0000/© 2020 Elsevier Inc. All rights reserved.

to this need, Pfleeger [6] presented an approach to cybersecurity measurement that uses existing tools to help organizations answer some basic questions about whether their networks and systems are secure. To answer these questions, the author identified key characteristics for distinguishing the security of one entity from another, where an entity can be a system, an organization, an enterprise, a business sector, or a nation. The author then selected attributes that reflect the cybersecurity aspects of interest and used appropriate ways to combine these attributes to measure the overall cybersecurity. Grid computing is a computing environment that incorporates many different computers via networking mechanism. Security in a grid environment is therefore a challenging matter, particularly at the design stage, because of the interoperability among possibly unknown entities in the system. Pagliarecci et al. [7] proposed a formal model checking-based verification methodology at the design stage for host security verification for grid systems. Their verification methodology was based on a reduction of a given grid system model to a model to which it is possible to apply a so-called cutoff theorem to make it decidable [7]. In a static and specific setting cybersecurity can be maintained easily because the environment is fixed. Indeed, there are solutions to cybersecurity problems that can usually be applied to special cases effectively. However, in a dynamically changing environment that is closer to reality, these solutions become far less effective. Shiva et al. [8] applied successfully a game theory-based holistic security approach to a dynamically changing scenario by viewing the interaction of the attacker and the defender as a game play. In terms of economics organizations often find it difficult to allocate resources to cybersecurity. This is in part due to uncertainty and severity of cybersecurity threats and vulnerabilities plus the effectiveness of mitigating measures. Several models for resource allocation in cybersecurity exist to aid decision makers but it is not clear which model should be chosen to apply to their organizations. Rue et al. [9] discussed a framework to analyze and to compare models and illustrated their framework with an analysis of three commonly-found types of models. Drugs and precursor chemicals were commonly moved through river and road networks in South America. The U.S. anti-drug efforts in South America would like to optimally allocate its limited resources to interdict *coca*, partially processed cocaine and precursor chemicals. Wood modeled this problem using a maximum flow problem in a capacitated network [10]. In the model an enemy attempted to maximize the flow through the capacitated network while an interdictor tries to minimize this maximum flow by stopping flow on network arcs with limited resources. He showed that this problem is *NP*-complete even when the interdiction of an arc requires exactly one unit of resource and developed integer programming models for the problem. This problem could also be viewed in a cyber security context. A maximum flow problem can also be used in a vulnerability attack graph to compute the best attack path [11]. Once the best attack path is determined, the key fragile links of the network system are known and become useful information for security and reinforcement by system administrators. A deterministic and static nature of a cyber system can enable adversaries to learn about the system by detections and, once enough information is gained, an attack can be launched to harm the system or for other malicious purposes. To avoid this scenario, a moving target defense was proposed to make the system more random, dynamic, and heterogeneous and therefore difficult to learn by the adversaries. Most recent work by Xiong et al. [12] proposed an effectiveness evaluation model for moving target defense based on system attack surface which is a set of ways in which an adversary can enter the system and potentially cause damages. In some situations damages caused by malicious activities on a network system can be either irrepairable or exorbitant. Hence, we want to prevent these activities from ever taking place by some prediction mechanism. In [13] the authors give a survey of all prediction and forecasting methods used in cyber security and discuss machine learning and data mining approaches in cyber security that have gained a lot of attention recently.

Defense-in-depth is an information assurance concept in which multiple layers of security controls are placed throughout an information technology system. This work focuses solely on defense-in-depth security. We review existing research work in defense-in-depth cyber security in this paragraph. Many cyber systems such as antivirus encryption and security firewalls contain a layered security or defense-in-depth. The defense-in-depth is a concept of protecting a computer network with a series of defense mechanism such that if one mechanism fails, another will already be in place to thwart an attack, more details of defense-in-depth have been discussed in [14,15]. In mathematical terms, the defense-in-depth can be described in the terms of a rooted weighted tree [16]. Therefore, cyber security systems or more generally layered computer systems are modeled as a fixed-weighted tree. For this reason, some researches in a weighted tree can be applied in the defense-in-depth [17–22]. Agnarsson et al. [16] were the first to propose a theoretical graph-based defense-in-depth cybersecurity model from an offensive perspective. In their article the authors considered a tree graph T with a weight function for both edges and vertices and wanted to find an attack subtree τ of T that maximizes the total weight on the tree's vertices under a given budget constraint. They showed the time complexities of the variations of this problem and gave an approximation scheme for the known *NP*-complete problem. For the defensive perspective, Agnarsson et al. [23] investigated an optimal security system by studying a way to assign a cost to an edge and a prize to a vertex of a given tree so that an attacker with a given budget obtains a minimum total prize from the model. They showed that in general it is not possible to develop an optimal security system for a given cyber security model. However, if a tree in a given model is a rooted path or a rooted star or a rooted 3-caterpillar, or a rooted 4-spider, there is an optimal security system for a certain type of the cyber security model. Additionally, they also defined a good security system and showed that it can be constructed in polynomial time.

In this article we present new results related to the model in Agnarsson et al. [16,23]. Our investigation is twofold. From the offensive perspective, we study how to approximate an optimal attack on the system with a guarantee. In a real life security system, one may want to put a strongest security measure on a part of a security system so that an attacker obtains a total prize as little as possible. In this defensive perspective, we model our security system with a security measure with cost ∞. Let $M = (T, C, P)$ be a security model, where $T = (V, E)$ is a tree rooted at r, C is a multiset of $|E(T)|$ costs and P

is a multiset of $|V(T)|-1$ prizes. Given a security system (T, c, p), where $c: E(T) \to C$ and $p: V(T) \setminus \{r\} \to P$ are bijections, and a budget $B \in \mathbb{Z}^+$, the problem is to determine an edge $e \in E(T)$, $c(e) = i$, such that the maximum total prize obtained in an optimal attack in (T, c, p) is minimized when i is replaced by ∞. For both problems, we give pseudo-polynomial time algorithms to compute the optimal solutions and fully-polynomial time approximation schemes to approximate the optimal solutions.

The outline of this article is as follows: in section 2 we describe some definitions and notations that will be used throughout this article; we discuss the complexities of the problems in section 3; a fully-polynomial time approximation scheme for the Optimal Attack Optimization Problem is given and discussed in section 4; in section 5 a fully-polynomial time approximation scheme for the Infinity Placement Optimization Problem is detailed and shown; the lack of potential drastic approximation improvement is then explained in section 6; we give a conclusion in section 7 and discuss relevance to real world and limitations of this work in section 8. In section 9 we explain our future work.

2. Definitions and notation

In this section we give the definitions of a cyber security model and its related offensive and defensive problems. The cyber security model is borrowed from [23]. Because this article is an extended version of our conference paper [24], some definitions and notations are reused here.

Definition 2.1. [Cyber Security Model] A cyber security model M is given by a three-tuple $M = (T, C, P)$, where $T = (V, E)$ is a tree rooted at r having $n \in \mathbb{N}$ non-root vertices, C is a multiset of penetration costs $c_1, \ldots, c_n \in \mathbb{Z}^+$, and P is a multiset of prizes $p_1, \ldots, p_n \in \mathbb{Z}^+$. The attack always begins at the root r and the root always has prize 0.

Observe that, although we use $\mathbb{Z}^+$ in our definition, all of our definitions can be applied to rational numbers as well because a given set of rational numbers can be multiplied by a common denominator to get an equivalent set of integers.

Definition 2.2. [Security System] A security system (T, c, p) with respect to a cyber security model $M = (T, C, P)$ is given by two bijections $c: E(T) \to C$ and $p: V(T) \setminus \{r\} \to P$. A system attack in (T, c, p) is given by a subtree τ of T that contains the root r of T. The cost of a system attack τ with respect to (T, c, p) is given by the cost $cst(c, p, \tau) = \sum_{e \in E(\tau)} c(e)$. The prize of a system attack τ with respect to (T, c, p) is given by the prize $pr(c, p, \tau) = \sum_{u \in V(\tau)} p(u)$. For a given budget $B \in \mathbb{Z}^+$ the maximum prize $pr^*(c, p, B)$ with respect to B is defined by $pr^*(c, p, B) = \max\{pr(c, p, \tau)|$ for all $\tau \subseteq T$, where $cst(c, p, \tau) \leq B\}$. A system attack τ whose prize is maximum with respect to a given budget B is called an optimal attack. Every system attack τ is maximal.

Occasionally, we use $G = \sum_{p_i \in P} p_i$ to refer to the total prize in the model. Our first problem is from an offensive perspective and is defined as follows.

Definition 2.3. [Optimal Attack Optimization Problem]
Instance: A security system $S = (T, c, p)$ associated with M and a budget $B \in \mathbb{Z}^+$.
Instruction: Compute the maximum total prize $pr^*(c, p, B)$.

The decision version of the Optimal Attack Optimization Problem is known to be *NP*-hard [16]. From the defensive perspective, given a security system and a budget, we want to place a security measure with the cost ∞ into the system to minimize the total prize from an optimal attack, assuming $\infty > B$ for any B. In real applications ∞ may represent the strongest preventive measure of security. An infinity placement operation is described. Decision and optimization versions of the problem are defined.

Definition 2.4. [Infinity Placement Operation] Given a cyber security model M, let $S = (T, c, p)$ be a security system associated with M and (T, c', p) be a security system in the model after the infinity placement operation has taken place. We say that an infinity placement operation replaces an edge cost $c(e)$ with ∞ if for all edges $e' \in E(T) \setminus \{e\}$, $c'(e') = c(e')$ and $c'(e) = \infty$.

Definition 2.5. [Infinity Placement Optimization Problem]
Instance: A security system $S = (T, c, p)$ associated with M and a budget $B \in \mathbb{Z}^+$.
Instruction: Find an edge $e \in E(T)$ such that $pr^*(c', p, B)$ is minimized after the infinity placement operation has been performed.

Definition 2.6. [Infinity Placement Decision Problem]
Instance: A security system $S = (T, c, p)$ associated with M, a budget $B \in \mathbb{Z}^+$ and a prize target $K \in \mathbb{Z}^+$.
Question: Is there an edge $e \in E(T)$ such that $pr^*(c', p, B) \leq K$ after the infinity placement operation has been performed?

This decision problem is a natural one because we could repeatedly ask for a value of K to solve the corresponding optimization version of it. Observe that the value of K is upper-bounded by the sum of all prizes and the lower bound is 0. We can apply binary search to find the optimal value of K.

3. Complexity

In this section we give results related to complexity of the Infinity Placement Decision Problem. Let us first review the class *coNP*. The class *coNP* consists of all problems whose complement are in *NP* [25,26]. A decision problem p is called *coNP*-hard if all other problems in *coNP* polynomially transform to p [27]. Moreover, p is *coNP*-complete if p is *coNP* and p is *coNP*-hard [27,26]. Additionally, p is *coNP*-complete if and only if the complement of p is *NP*-complete, unless *NP*=*coNP* [26]. We next show that Infinity Placement Decision Problem is *coNP*-hard. In order to show the hardness result, the complement of a known *NP*-complete problem called Partition is used. The Complement of the Partition Problem can be stated as follows.

Definition 3.1. [Complement of the Partition Problem]
Instance: A finite set A and a "size" $s(a) \in \mathbb{Z}^+$ for each $a \in A$.
Question: For each nonempty subset $A' \subseteq A$, is $\sum_{a\in A'} s(a)$ not equal to $\sum_{a\in A\setminus A'} s(a)$?

Theorem 3.1. *The* Infinity Placement Decision Problem *is coNP-hard.*

Proof. We show that the Infinity Placement Decision Problem is *coNP*-hard. We make a reduction from the Complement of Partition Problem. Let $s(D) = \sum_{d\in D} s(d)$ be the total size of all elements in D. We construct a security system (T, c, p), a budget $B \in \mathbb{Z}^+$, and a prize target $K \in \mathbb{Z}^+$ such that for each nonempty subset $A' \subseteq A$, $s(A') \neq s(A \setminus A')$ if and only if there is an edge $e \in E(T)$ such that $pr^*(c', p, B) \le K$ after the infinity placement operation has been performed.

Given a problem instance of the Complement of Partition Problem, we first construct a corresponding model $M = (T, C, P)$ as follows. Construct a rooted star T by letting $V(T) = A \cup \{r, x\}$ and $E(T) = \{\{r, v\}|$ for each $v \in V(T) \setminus \{r\}\}$. Furthermore, let C be $\{s(a)|$ for each $a \in A\} \cup \{1\}$ and P be $\{s(a)|$ for each $a \in A\} \cup \{\frac{s(A)}{2} + 1\}$. We construct the assignments c and p in such a way that for each $a \in V(T) \setminus \{r, x\}$, $c(\{r, a\}) = p(a) = s(a)$, $c(\{r, x\}) = 1$ and $p(x) = \frac{s(A)}{2} + 1$. Finally, we let the budget $B = \frac{s(A)}{2}$ and the prize target $K = \frac{s(A)}{2} - 1$. The whole construction of the problem instance can be computed in $O(n)$, where $n = |A|$. We will show that for each nonempty subset $A' \subseteq A$, $s(A') \neq s(A \setminus A')$ if and only if there is an edge $e \in E(T)$ such that $pr^*(c', p, \frac{s(A)}{2}) \le \frac{s(A)}{2} - 1$ after the infinity placement operation has been performed.

→ Suppose for each nonempty subset $A' \subseteq A$, $s(A') \neq s(A \setminus A')$. We will show that there is an edge $e \in E(T)$ such that $pr^*(c', p, \frac{s(A)}{2}) \le \frac{s(A)}{2} - 1$ after the infinity placement operation has been performed. Because $pr^*(c', p, \frac{s(A)}{2}) \le \frac{s(A)}{2} - 1$, the infinity must be placed on the edge $e = \{r, x\}$ where $c(\{r, x\}) = 1$ and $p(x) = \frac{s(A)}{2} + 1$. WLOG, assume that $s(A') < s(A \setminus A')$. Because $B = \frac{s(A)}{2}$ and, by construction, $c'(\{r, a\}) = p(a) = s(a)$ for each $a \in V(T) \setminus \{r, x\}$, the attack achieves exactly the sum of prizes in the largest subset A' that is at most $\frac{s(A)}{2} - 1$. Let $\tau_{A'}$ be such a corresponding attack and it follows that $\tau_{A'}$ is a maximum attack, which implies $pr^*(c', p, \frac{s(A)}{2}) \le \frac{s(A)}{2} - 1$ after the infinity placement operation has been performed.

← Suppose there is an edge $e \in E(T)$ such that $pr^*(c', p, \frac{s(A)}{2}) \le \frac{s(A)}{2} - 1$ after the infinity placement operation has been performed. We show that for each nonempty subset $A' \subseteq A$, $s(A') \neq s(A \setminus A')$. Let $A^* \subseteq V(T)$ be the set that yields $pr^*(c', p, \frac{s(A)}{2}) \le \frac{s(A)}{2} - 1 < \frac{s(A)}{2}$ and $A^* \cup A^{**} = A$. Because A^* yields the maximum prize $pr^*(c', p, \frac{s(A)}{2})$ and $pr^*(c', p, \frac{s(A)}{2}) < \frac{s(A)}{2}$, $pr(c', p, \tau) > \frac{s(A)}{2}$, where τ is the system attack that contains all prizes in A^{**}. This implies $s(A^*) \neq s(A^{**})$. Observe that, given a budget $\frac{s(A)}{2}$, all system attacks $\tau_1, \tau_2, \ldots, \tau_q$ yield at most $pr^*(c', p, \frac{s(A)}{2}) < \frac{s(A)}{2}$ and each such $V(\tau_i)$ corresponds to a subset A_i of vertices (or prizes). Therefore, $s(A_i) \neq s(A \setminus A_i)$ for all i. Because each nonempty subset A' of A is either A_i or $A \setminus A_i$, we conclude that for each nonempty subset $A' \subseteq A$, $s(A') \neq s(A \setminus A')$. Hence, the theorem holds. □

Theorem 3.1 tells us that it is highly unlikely that a reasonably fast algorithm exists to solve the Infinity Placement Decision Problem and its optimization counterpart, unless $P = NP$. In section 5 we will provide solutions to this problem.

4. Approximation scheme for the optimal attack problem

In this section we consider a fully-polynomial time approximation scheme (FPTAS) as a solution to the Optimal Attack Optimization Problem. We carefully note here that Agnarsson et al. gave a $O(nB^2)$ pseudo-polynomial time algorithm and a $O((1/\epsilon)^2 n^3)$ FPTAS to solve the Optimal Attack Problem [16] with the cost at most $(1+\epsilon)B$. Here we give another $O(nG^2)$ pseudo-polynomial time algorithm and a (1-ϵ) fully-polynomial time approximation scheme on the optimal prize with the time bound $O(\frac{1}{\epsilon^2} n^3 \log G)$. Observe that when the total prize G is less than the total budget B our pseudo-polynomial time algorithm is clearly a better alternative. In the following paragraphs we first describe the pseudo-polynomial time algorithm and then the FPTAS based on this algorithm.

4.1. Pseudo-polynomial time algorithm

Our algorithm applies the dynamic programming approach and is very similar to that of Agnarsson et al., which finds a maximum prize subtree with a given fixed cost. Our algorithm, on the other hand, finds a minimum cost subtree, given a fixed total prize. We review the central idea of their algorithm. We construct a $d(u) \times (m+1)$ matrix for each vertex u in the tree T that stores the maximum prize of a subtree rooted at u on at most k edges and that contains only the rightmost $d(u)-i+1$ branches from u, for each $k \in \{0, 1, ..., m\}$ and $i \in \{1, ..., d(u)\}$. We assume that our rooted tree T has its vertices ordered from left-to-right in some arbitrary but fixed order, that is, T is a planted plane tree. A planted plane tree can be decomposed into two rooted subtrees.

For a subtree τ of T rooted at $u \in V(T)$, denote by $\tau(v)$ the largest subtree of τ that is rooted at a vertex v (if $v \in T[V(\tau)]$). Denote by u_l the leftmost child of u in τ (if it exists). Let $\tau_l = \tau(u_l)$ denote the subtree of τ generated by u_l, that is, the largest subtree of T rooted at u_l. Finally, let $\tau'' = \tau - V(\tau_l) = T[V(\tau)\backslash V(\tau_l)]$ denote the subtree of τ generated by the vertices not in τ_l. The decomposition of τ of T gives two vertex-disjoint subtrees τ_l and τ'' whose roots are connected by a single edge $e(u_l)$.

In particular, for each vertex $u \in V(T)$, we have a partition of $T(u)$ into $T(u)_l = T(u_l)$ and $T(u)'' = T''(u)$. Note that if u is a leaf, then $T(u) = T''(u) = \{u\}$ and $u_l = T(u_l) = \emptyset$. Additionally, if u has exactly one child which is its leftmost child u_l, then $T(u)$ is the two-path between u and its only child u_l, $T''(u) = \{u\}$, and $T(u_l) = u_l$. Let $d(u)$ be the degree of u. We can recursively define the trees $T^1(u), ..., T^{d(u)}(u)$ by

$$T^1(u) = T(u),$$
$$T^{i+1}(u) = (T^i)''(u).$$

For each vertex $u \in V(T)$, we create a $d(u) \times (B+1)$ matrix as follows:

$$N(u) = \begin{bmatrix} N_0^1(u) & N_1^1(u) & \dots & N_B^1(u) \\ N_0^2(u) & N_1^2(u) & \dots & N_B^2(u) \\ & & \vdots & \\ N_0^{d(u)}(u) & N_1^{d(u)}(u) & \dots & N_B^{d(u)}(u) \end{bmatrix}$$

where $N_k^i(u)$ is the maximum prize of a subtree of $T^i(u)$ rooted at u of total cost at most k for each $i \in \{1, 2, \dots, d(u)\}$ and $k \in \{0, 1, \dots, B\}$. In particular, $N_0^i(u) = p(u)$ for each vertex u and $i \in \{1, 2, \dots, d(u)\}$. For each leaf u of T, and each i and k, we set $N_k^i(u) = p(u)$, and for each internal vertex u we have a recursion given in the following way: for a vertex u and an *arbitrary* subtree τ rooted at u, we let $N_k(u; \tau)$ be the maximum prize of a subtree of τ rooted at u having total cost at most k or 0 if vertex u does not exist. If a maximum-prize subtree of τ with total cost at most k does not contain the edge from u to its leftmost child u_l, then $N_k(u; \tau) = N_k(u; \tau'')$. Otherwise, such a maximum subtree has a total cost at most $i - c(e(u_l))$ from τ_l and a total cost at most $k-i$ from τ''. The following lemma holds.

Lemma 4.1 *(Optimal substructure). The arbitrary subtree τ rooted at u is a maximum-prize subtree of total cost at most k that contains the leftmost child u_l of u if and only if the included subtree of τ_l is a maximum-prize subtree of total cost at most $i - c(e(u_l))$ rooted at u_l and the included subtree of τ'' is a maximum-prize subtree of total cost at most $k-i$ rooted at u for some $i \in \{c(e(u_l)), \dots, k\}$.*

With the optimal substructure in mind, we have the following recurrence and the following theorem.

$$N_k(u; \tau) = \max\left(N_k(u; \tau''), \max_{c(e(u_l)) \le i \le k} (N_{i-c(e(u_l))}(u_l; \tau_l) + N_{k-i}(u; \tau''))\right) \tag{1}$$

Theorem 4.1. *If $M = (T, c, p, B, G)$ is a cyber security model, where T has n vertices and $c : E(T) \mapsto \mathbb{N}$ takes only positive-integer values, then the* Optimal Attack Optimization Problem *can be solved in $O(B^2 n)$ time.*

Our dynamic programming algorithm is similar with the following matrix, lemma, recurrence, and theorem.

For each vertex $u \in V(T)$, we create a $d(u) \times (G+1)$ matrix as follows:

$$M(u) = \begin{bmatrix} M_0^1(u) & M_1^1(u) & \dots & M_G^1(u) \\ M_0^2(u) & M_1^2(u) & \dots & M_G^2(u) \\ & & \vdots & \\ M_0^{d(u)}(u) & M_1^{d(u)}(u) & \dots & M_G^{d(u)}(u) \end{bmatrix}$$

where $M_k^i(u)$ is the minimum cost of subtree of $T^i(u)$ rooted at u of total prize at least k for each $i \in \{1, 2, \dots, d(u)\}$ and $k \in \{0, 1, \dots, G\}$, where $G = \sum_{v \in V(T)} p(v)$ is the total prize.

Lemma 4.2 *(Revised optimal substructure). The arbitrary subtree τ rooted at u is a minimum-cost subtree of total prize at least k that contains the leftmost child u_l of u if and only if the included subtree of τ_l is a minimum-cost subtree of total prize at least j rooted at u_l and the included subtree of τ'' is a minimum-cost subtree of total prize at least $k - j$ rooted at u for some $j \in \{0, 1, 2, \ldots, k\}$.*

With the new optimal substructure in mind, a new recurrence can be defined as follows.

$$M_k(u;\tau) = \min\left(M_k(u;\tau''), \min_{0\le j\le k}(M_j(u_l;\tau_l) + M_{k-j}(u;\tau''))\right) \tag{2}$$

Each vertex $u \in V(T)$ is associated with a $d(u) \times (G+1)$ matrix. Therefore, the number of subproblems is $\sum_{u\in V(T)}(G+1) \times d(u)$. By the Handshaking Lemma, we obtain $(G+1) \times 2|E|$ subproblems. Because each subproblem takes $O(k)$ to compute, the total time complexity is therefore $(G+1) \times 2|E| \times O(k) \le (G+1) \times 2|E| \times O(G) = O(G^2 n)$. We have the following theorem.

Theorem 4.2. *If $M = (T, c, p, B, G)$ is a cyber security model, where T has n vertices and $c : E(T) \mapsto \mathbb{N}$ takes only positive-integer values, then the* Optimal Attack Optimization Problem *can be solved in $O(G^2 n)$ time.*

4.2. Fully-polynomial time approximation scheme

In this section we discuss a FPTAS based on the pseudo-polynomial time algorithm in the previous subsection. The main idea of the approximation scheme is to scale and round each prize and hope that the effect of scaling and rounding will give an approximate solution of at least $1 - \epsilon$ of the optimal solution.

Assume P_{opt} is an optimal prize from a set S_{opt} that can be obtained from the attack. Given a security system $S = (T, c, p)$ associated with M and a budget $B \in \mathbb{Z}^+$ and an ϵ, we scale each prize $p_i \in P$ by a factor of $\frac{n}{\epsilon P_{opt}^2}(\sum_{p_j\in S_{opt}} p_j)$. After rounding each scaled prize, we have $\lfloor p_i \frac{n}{\epsilon P_{opt}^2}(\sum_{p_j\in S_{opt}} p_j)\rfloor$. Then we run the pseudo-polynomial time algorithm with this new input.

Theorem 4.3. *Let $\epsilon > 0$ be a constant and $M = (T, c, p, B, G)$ be a cyber security model. The pseudo-polynomial time approximation scheme yields a solution S with a total prize P_S such that $P_S \ge (1-\epsilon)P_{opt}$.*

Proof. Let S_{opt} be the optimal set that gives a total prize P_{opt}. Observe that the scaling gives $\sum_{p_i\in S_{opt}} p_i \frac{n}{\epsilon P_{opt}^2}(\sum_{p_j\in S_{opt}} p_j) = \frac{n}{\epsilon}$. The rounding of each scaled prize reduces each scaled prize by at most one. Hence, the optimal solution after the rounding becomes at least $\frac{n}{\epsilon} - n$. Because the algorithm achieves an optimal solution, we have a scaled total prize at least $\frac{n}{\epsilon} - n$. To obtain the solution in terms of original prizes, we unscale $\frac{n}{\epsilon} - n$ by $\frac{\epsilon P_{opt}^2}{n\sum_{p_j\in S_{opt}} p_j}$. We have $(\frac{n}{\epsilon} - n)(\frac{\epsilon P_{opt}^2}{n\sum_{p_j\in S_{opt}} p_j}) = (1-\epsilon)P_{opt}$. Thus, P_S is at least $(1-\epsilon)P_{opt}$. The theorem holds. □

A few remarks are in order. The time complexity of the approximation scheme is $\frac{n}{\epsilon P_{opt}^2}(\sum_{p_j\in S_{opt}} p_j) \times O(P_{opt}^2 n) = O(\frac{1}{\epsilon} n^2 P_{opt})$. Observe that with the factor of P_{opt} this time bound is strictly still pseudo-polynomial. We can make this time bound fully polynomial by relating $\frac{n}{\epsilon}$ to P_{opt} and eliminating P_{opt} altogether. We need to know P_{opt} to run the exact algorithm but we do not. We use guessing. Suppose we guess P'_{opt}. If $P'_{opt} > P_{opt}$, the scaled total prize becomes less than $\frac{n}{\epsilon}$. If $P'_{opt} < P_{opt}$, the scaled total prize becomes greater than $\frac{n}{\epsilon}$. What range of total prizes should we guess for P_{opt}? We note that the scaled optimal solution $\frac{n}{\epsilon}$ becomes the original P_{opt} when unscaled and therefore $\frac{2n}{\epsilon}$ becomes the original $\frac{P_{opt}}{2}$. Our guess P'_{opt} works if it is in the range of P_{opt} and $\frac{P_{opt}}{2}$. Given the two conditions above, we can use binary search for this guess. The time for this binary search is $O(\log G)$. Because $\frac{n}{\epsilon}$ corresponds to P_{opt}, $\frac{2n}{\epsilon}$ corresponds to $\frac{P_{opt}}{2}$. Hence, we have the desired upper bound $P'_{opt} \le \frac{2n}{\epsilon}$. The total time complexity now becomes $O(\frac{1}{\epsilon^2} n^3 \log G)$, which is fully polynomial. We have the following main theorem.

Theorem 4.4. *Let $\epsilon > 0$ be a constant and $M = (T, c, p, B, G)$ be a cyber security model. The fully polynomial time approximation scheme yields a solution S with a total prize P_S such that $P_S \ge (1-\epsilon)P_{opt}$ and has the time bound $O(\frac{1}{\epsilon^2} n^3 \log G)$.*

5. Approximation scheme for the infinity placement problem

In this section we first consider a pseudo-polynomial time algorithm to solve the Infinity Placement Problem for an exact solution and then the fully-polynomial time approximation scheme for the problem. Our pseudo-polynomial time algorithm is based on the pseudo-polynomial time algorithm that solves the Optimal Attack Optimization Problem in the preceding section. The idea is simple. Placing infinity on a different edge at a time, we then execute this pseudo-polynomial time algorithm to obtain a corresponding maximum total prize for this particular placement. We execute the process $|E(G)|$ times to obtain $|E(G)|$ maximum total prizes for each of the $|E(G)|$ infinity placements. Among all these maximum total prizes, the smallest one indicates the optimal infinity placement.

Lemma 5.1. *Given a security system $S = (T, c, p)$ associated with M and a budget B, the* Infinity Placement Problem *can be solved in $O(G^2n^2)$ time, where $G = \sum_{p_i \in P} p_i$.*

Proof. We execute the $O(G^2n)$ pseudo-polynomial time algorithm to solve the Optimal Attack Optimization Problem $|E| = n$ times. Hence, the total time complexity is $O(G^2n^2)$. □

To obtain a PPTAS, we apply a similar scaling and rounding technique to that of the previous section. Assume P_{opt} is an optimal prize that can be obtained from an attack at the optimal infinity placement. Given a security system $S = (T, c, p)$ associated with M and a budget $B \in \mathbb{Z}^+$ and an ϵ, we scale each prize $p_i \in P$ by a factor of $\frac{n}{\epsilon P_{opt}^2}(\sum_{p_j \in S_{opt}} p_j)$. After rounding each prize p_i, we have $\lceil p_i \frac{n}{\epsilon P_{opt}^2}(\sum_{p_j \in S_{opt}} p_j)\rceil$. Then we run the pseudo-polynomial time algorithm with this new input.

Corollary 5.1. *Let $\epsilon > 0$ be a constant and $M = (T, c, p, B, G)$ be a cyber security model. The pseudo-polynomial time approximation scheme yields a solution S and infinity placement with a total prize P_S such that $P_S \le (1+\epsilon)P_{opt}$.*

Proof. Let S_{opt} be the optimal set that gives a total prize P_{opt}. Observe that the scaling gives $\sum_{p_i \in S_{opt}} p_i \frac{n}{\epsilon P_{opt}^2}(\sum_{p_j \in S_{opt}} p_j) = \frac{n}{\epsilon}$. The rounding of each scaled prize increases each scaled prize by at most one. Hence, the optimal solution after the rounding becomes at most $\frac{n}{\epsilon} + n$. Because the algorithm achieves an optimal solution, we have a scaled total prize at most $\frac{n}{\epsilon} + n$. To obtain the solution in terms of original prizes, we unscale $\frac{n}{\epsilon} + n$ by $\frac{\epsilon P_{opt}^2}{n(\sum_{p_j \in S_{opt}} p_j)}$. We have $\left(\frac{n}{\epsilon} + n\right)\left(\frac{\epsilon P_{opt}^2}{n(\sum_{p_j \in S_{opt}} p_j)}\right) = (1+\epsilon)P_{opt}$. Thus, P_S is at most $(1+\epsilon)P_{opt}$. The corollary holds. □

A few remarks are in order. First, the time complexity of the pseudo-polynomial time approximation scheme is $\frac{n}{\epsilon P_{opt}^2}(\sum_{p_j \in S_{opt}} p_j) \times O(P_{opt}^2 n^2) = O(\frac{1}{\epsilon} n^3 P_{opt})$. Similar to the running time of the pseudo-polynomial approximation scheme in the previous section, the factor P_{opt} in this time bound makes the running time strictly pseudo-polynomial. We can apply the same technique to eliminate the factor P_{opt} and make the time bound fully polynomial. We need to know P_{opt} to run the exact algorithm but we do not. We again use guessing. Suppose we guess P'_{opt}. If $P'_{opt} > P_{opt}$, the scaled total prize becomes less than $\frac{n}{\epsilon}$. If $P'_{opt} < P_{opt}$, the scaled total prize becomes greater than $\frac{n}{\epsilon}$. What range of total prizes should we guess for P_{opt}? We note that the scaled optimal solution $\frac{n}{\epsilon}$ still becomes the original P_{opt} when unscaled and therefore $\frac{2n}{\epsilon}$ becomes the original $\frac{P_{opt}}{2}$. Our guessing range is thus in the range of P_{opt} and $\frac{P_{opt}}{2}$. Given the two conditions above, we can use binary search for this guess. The time for this binary search is $O(\log G)$. Because $\frac{n}{\epsilon}$ corresponds to P_{opt}, $\frac{2n}{\epsilon}$ corresponds to $\frac{P_{opt}}{2}$. Hence, we have the desired upper bound $P'_{opt} \le \frac{2n}{\epsilon}$. The total time complexity now becomes $O(\frac{1}{\epsilon^2} n^4 \log G)$, which is fully polynomial. Our main corollary follows.

Corollary 5.2. *Let $\epsilon > 0$ be a constant and $M = (T, c, p, B, G)$ be a cyber security model. The fully-polynomial time approximation scheme yields a solution S and infinity placement with a total prize P_S such that $P_S \le (1+\epsilon)P_{opt}$ and has the time bound $O(\frac{1}{\epsilon^2} n^4 \log G)$.*

6. Improvability

At this point we have successfully shown pseudo-polynomial time algorithms to find exact solutions for both the Optimal Attack Optimization Problem and the Infinity Placement Optimization Problem and fully-polynomial time approximation schemes for both problems. The question now seems to be whether any improvement can be made to these time bounds and/or solutions. We next show that no polynomial-time approximation algorithm can solve the Optimal Attack Optimization Problem with $OPT(I) - A(I) \le k$ and no polynomial-time approximation algorithm can solve the Infinity Placement Optimization Problem with $A(I) - OPT(I) \le k$ for any fixed integer k.

6.1. Optimal attack optimization problem

Let us review the Optimal Attack Optimization Problem. Given a security system S in M and a budget B, we want to compute a total prize from an optimal attack on the system. Let $OPT(I)$ be the optimal solution of the problem instance I and $A(I)$ the solution from an approximation algorithm.

Theorem 6.1. *If $P \neq NP$, then no polynomial-time approximation algorithm can compute a maximum total prize with $OPT(I) - A(I) \le k$ for any fixed integer k.*

Proof. We will prove this claim by contradiction. Suppose there exists a polynomial-time approximation algorithm that can compute a maximum total prize with $OPT(I) - A(I) \le k$. We will show that this algorithm can be used to construct an optimal solution to any instance I of the problem, thereby a contradiction.

Given a security system S and a budget B, we construct a new instance I' by multiplying $k+1$ to each $p_i \in P$ of the security system S while the other parts of the instance remain the same as in the original instance I. Observe that every feasible solution for I' is exactly a feasible solution for I and vice versa. Hence, the value of the solution for I' is $k+1$ times the value of the solution for I. We run the approximation algorithm on I' to obtain the solution $A(I')$. This gives us a solution σ for I. Clearly, $OPT(I') - A(I') \le k$. Let $f : S(I) \mapsto \mathbb{Z}^+$ be a function that assigns a value to each solution in the set $S(I)$ of all feasible solutions for instance I.

$$\begin{aligned} OPT(I') - A(I') &\le k \\ (k+1)OPT(I) - (k+1)f(\sigma) &\le k \\ OPT(I) - f(\sigma) &\le \frac{k}{k+1} \end{aligned}$$

Since we are dealing with integer values, $OPT(I) - f(\sigma) \le 0$. This means the solution σ is optimal and we derive a contradiction. Hence, the theorem holds. □

Theorem 6.1 tells us that it is highly unlikely that we could come up with a polynomial time approximation algorithm with an absolute guarantee for the Optimal Attack Optimization Problem.

6.2. Infinity placement optimization problem

In this section we discuss the limit of improvement of time and/or solutions to the Infinity Placement Optimization Problem. We show that no polynomial-time approximation algorithm can solve the Infinity Placement Optimization Problem with $A(I) - OPT(I) \le k$ for any fixed integer k. Let $OPT(I)$ be the optimal solution of the problem instance I of the Infinity Placement Optimization Problem and $A(I)$ the solution from an approximation algorithm.

Theorem 6.2. *If $P \neq NP$, then no polynomial-time approximation algorithm can solve the* Infinity Placement Optimization Problem *with $A(I) - OPT(I) \le k$ for any fixed integer k.*

Proof. We will prove this by contradiction. Suppose there exists a polynomial-time approximation algorithm that can compute a minimum total prize with $A(I) - OPT(I) \le k$. We will show that this algorithm can be used to construct an optimal solution to any instance I of the problem, thereby a contradiction.

Given a security system S and a budget B, a new instance I' is constructed by multiplying $k+1$ to each $p_i \in P$ of the security system S while the other parts of the instance remain the same as in the original instance I. Observe that every feasible solution for I' is exactly a feasible solution for I and vice versa. Hence, the value of the solution for I' is $k+1$ times the value of the solution for I. We run the approximation algorithm on I' to obtain the solution $A(I')$. This gives us a solution σ for I. Clearly, $A(I') - OPT(I') \le k$. Let $f : S(I) \mapsto \mathbb{Z}^+$ be a function that assigns a value to each solution in the set $S(I)$ of all feasible solutions for instance I.

$$\begin{aligned} A(I') - OPT(I') &\le k \\ (k+1)f(\sigma) - (k+1)OPT(I) &\le k \\ f(\sigma) - OPT(I) &\le \frac{k}{k+1} \end{aligned}$$

Since we are dealing with integer values, $f(\sigma) - OPT(I) \le 0$. This means the solution σ is optimal and we derive a contradiction. Hence, the theorem holds. □

Theorem 6.2 tells us that it is highly unlikely that we could come up with a polynomial time approximation algorithm with an absolute guarantee for the Infinity Placement Optimization Problem.

7. Conclusion

In this article we defined a cyber-security model and a security system associated with the model. Given the model, we considered problems from offensive and defensive perspectives. The problem of computing an optimal attack on a security system is known to be *NP*-complete. We give a $O(G^2n)$-time algorithm to compute an optimal attack. Based on this pseudo-polynomial algorithm, a (1-ϵ) fully-polynomial time approximation scheme with the time bound $O(\frac{1}{\epsilon^2}n^3 \log G)$ is given and discussed. We also consider determining an edge $e \in E(T)$ such that the maximum sum of prizes obtained from an optimal attack in (T, c, p) is minimum when $c(e)$ is replaced by ∞. This problem is proved to be $coNP$-hard and a $O(G^2n^2)$-time algorithm is provided for computing an exact solution and a (1+ϵ) fully-polynomial time approximation scheme with the time bound $O(\frac{1}{\epsilon^2}n^4 \log G)$ is shown to solve this problem. In the end we show that no potential drastic improvement can be made to solutions and/or time to both problems and thus our fully polynomial time approximation schemes are the best possible.

8. Relevance to real world and limitations

In real world a defense-in-depth system might comprise several security layers. An intruder might want to enter the system to obtain information or things (i.e., prizes) such as blueprints of high security buildings, details of covert operations, or a name list of active intelligence agents. In the Optimal Attack Optimization Problem context, the optimal attack might be the one with all of these things and information given that the intruder has the required means (i.e., budget). Because this problem is hard to compute optimally, an intruder might want to use an approximation scheme that we provide. In terms of the real world defense, the organization might want to improve its defense by putting in place some very strong defense (i.e., cost) such as a disk encryption system. The problem of optimally putting the disk encryption system in the whole security system is equivalent to the Infinity Placement Optimization Problem. Since this problem is hard to compute, again our approximation scheme can be useful here.

When we design an abstract model that represents a real world problem, we must balance between simplicity of the model and things we want to study. If our model is too simple, we will not learn anything from the model. If our model integrates everything we may want to study, the model will be too complex and we will not see a tree from a forest. Consequently, a good model is one representing certain characteristics of the real world problems we want to investigate. Like any abstract model, our model has some limitations. First, our model is designed to study defense-in-depth security specifically and therefore is not applicable to the other types of defenses. Second, prize and cost functions need to be defined by experts. For instance, a national security database and a university database must have different prizes and the prizes should vary greatly according to the importance of each database. Likewise, a firewall and an encryption scheme should also have different costs of penetration and these costs should be assigned by experts. Third, some cyber systems contain a cycle and, therefore, are not trees. This work is not applicable in such cases.

9. Future work

One can approach problems in cyber security in many ways. For example, the authors [13] discussed machine learning and data mining approaches in cyber security. Our approach is, however, based on mathematics and complexity theory, which responds to the call for the science of cyber security by Schneider [3]. In this context our future work can be listed as follows.

- From an attacker's perspective, we know how to compute an optimal attack on a security system using the dynamic programming technique. From a defender's perspective, we do not want an attacker to compute an optimal attack easily. Can we somehow defeat the dynamic programming technique? We know that the dynamic programming paradigm depends on the optimal substructure property and repeated subproblems. Can we characterize a security system without one of the two properties so that dynamic programming cannot be applied?
- In some cases an attacker does not necessarily want to obtain a maximum prize from a security system. Suppose an attacker wants to obtain a fraction f of the maximum prize. Is this problem *NP*-complete? How does one compute such a fraction f of the maximum prize? From a defensive perspective, what can we do to prevent such an attack, given we know f?
- In the Infinity Placement Optimization Problem we considered placing infinity cost on a single edge to minimize an optimal attack. What happen if we have k infinity costs to place on k edges? Is this problem *NP*-complete? Is it harder or easier to compute as k grows in value? What is the value of k such that the problem becomes polynomial-time computable?
- Similar to the first open problem, a defender may not necessarily want to place infinity to minimize the sum of prizes. He might only want to place infinity so that an attacker only obtains at most c times the minimized sum of prizes. Is this problem *NP*-complete? We can also consider this problem with many infinity edge costs.

CRediT authorship contribution statement

Supachai Mukdasanit: Investigation, Validation. **Sanpawat Kantabutra:** Funding acquisition, Investigation, Validation, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research is funded by the Thailand Research Fund grant number RSA6080029. The second author is thankful for this support.

References

[1] K. Richards, R. Lasalle, F. van den Dool, 2017 cost of cyber crime study, https://www.accenture.com/us-en/insight-cost-of-cybercrime-2017, may 2018.
[2] P. Passeri, 2017 cyber attacks statistics, https://www.hackmageddon.com/2018/01/17/2017-cyber-attacks-statistics/, jan 2018.
[3] F.B. Schneider, Blueprint for a science of cybersecurity, Tech. Rep., 2011.
[4] D.M. Dunlavy, B. Hendrickson, T.G. Kolda, Mathematical challenges in cybersecurity, Sandia Report, February 2009.
[5] R. Armstrong, J. Mayo, F. Siebenlist, Complexity science challenges in cybersecurity, Sandia National Laboratories SAND Report, 2009.
[6] S.L. Pfleeger, Useful cybersecurity metrics, IT Prof. Mag. 11 (3) (2009) 38.
[7] F. Pagliarecci, L. Spalazzi, F. Spegni, Model checking grid security, Future Gener. Comput. Syst. 29 (3) (2013) 811–827.
[8] S. Shiva, S. Roy, D. Dasgupta, Game theory for cyber security, in: Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research, ACM, 2010, p. 34.
[9] R. Rue, S.L. Pfleeger, D. Ortiz, A framework for classifying and comparing models of cyber security investment to support policy and decision-making, in: WEIS, 2007.
[10] R.K. Wood, Deterministic network interdiction, Math. Comput. Model. 17 (2) (1993) 1–18.
[11] H. Wang, Z. Chen, J. Zhao, X. Di, D. Liu, A vulnerability assessment method in industrial Internet of things based on attack graph and maximum flow, IEEE Access 6 (2018) 8599–8609.
[12] X.-L. Xiong, L. Yang, G.-S. Zhao, Effectiveness evaluation model of moving target defense based on system attack surface, IEEE Access 7 (2019) 9998–10014.
[13] M. Husák, J. Komárková, E. Bou-Harb, P. Čeleda, Survey of attack projection, prediction, and forecasting in cyber security, IEEE Commun. Surv. Tutor. 21 (1) (2019) 640–660.
[14] T. Bass, R. Robichaux, Defense-in-depth revisited: qualitative risk analysis methodology for complex network-centric operations, IEEE MILCOM 2001 (2001) 28–31.
[15] A. Ahmad, S.B. Maynard, S. Park, Information security strategies: towards an organizational multi-strategy perspective, J. Intell. Manuf. 25 (2) (2014) 357–370.
[16] G. Agnarsson, R. Greenlaw, S. Kantabutra, On cyber attacks and the maximum-weight rooted-subtree problem, Acta Cybern. 22 (3) (2016) 591–612.
[17] S.-Y. Hsieh, T.-Y. Chou, Finding a weight-constrained maximum-density subtree in a tree, in: International Symposium on Algorithms and Computation, Springer, 2005, pp. 944–953.
[18] H.C. Lau, T.H. Ngo, B.N. Nguyen, Finding a length-constrained maximum-sum or maximum-density subtree and its application to logistics, Discrete Optim. 3 (4) (2006) 385–391.
[19] H.-H. Su, C.L. Lu, C.Y. Tang, An improved algorithm for finding a length-constrained maximum-density subtree in a tree, Inf. Process. Lett. 109 (2) (2008) 161–164.
[20] S. Coene, C. Filippi, F.C. Spieksma, E. Stevanato, Balancing profits and costs on trees, Networks 61 (3) (2013) 200–211.
[21] D.S. Johnson, K. Niemi, On knapsacks, partitions, and a new dynamic programming technique for trees, Math. Oper. Res. 8 (1) (1983) 1–14.
[22] A. Hamacher, W. Hochstättler, C. Moll, Tree partitioning under constraints — clustering for vehicle routing problems, Discrete Appl. Math. 99 (1–3) (2000) 55–69.
[23] G. Agnarsson, R. Greenlaw, S. Kantabutra, The structure of rooted weighted trees modeling layered cyber-security systems, Acta Cybern. 22 (4) (2016) 735–769.
[24] S. Mukdasanit, S. Kantabutra, The complexity of the infinity replacement problem in the cyber security model, in: 2017 21st International Computer Science and Engineering Conference (ICSEC), IEEE, 2017, pp. 1–5.
[25] R.G. Michael, S.J. David, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, 1979.
[26] B. Korte, J. Vygen, Combinatorial Optimization: Theory and Algorithms, Algorithms and Combinatorics, Springer Berlin Heidelberg, 2018, https://books.google.co.th/books?id=EjtRDwAAQBAJ.
[27] L. Cummings, Combinatorics on Words: Progress and Perspectives, Elsevier Science, 2014, https://books.google.co.th/books?id=CLPiBQAAQBAJ.