Research Article

Approximate Core Allocation for Large Cooperative Security Games

Saman Zonouz¹ and Parisa Haghani²

¹ University of Miami, Coral Gables, FL, USA

² University of Illinois, Champaign, IL, USA

Correspondence should be addressed to Parisa Haghani, parisa.haghany@gmail.com

Received 2 September 2012; Accepted 24 September 2012

Academic Editors: S. Cheng, T. Erseghe, W. Jiang, and V. Tralli

Copyright © 2012 S. Zonouz and P. Haghani. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Coalition games have been recently used for modeling a variety of security problems. From securing the wireless transmissions in decentralized networks to employing effective intrusion detection systems in large organizations, cooperation among interested parties has shown to bring significant benefits. Motivating parties to abide to a solution is, however, a key problem in bridging the gap between theoretical models and practical solutions. Benefits should be distributed among players (wireless nodes in a network, different divisions of an organization in security risk management, or organizations cooperating to fight spam), such that no group of players is motivated to break off and form a new coalition. This problem, referred to as *core allocation*, grows computationally very expensive with a large number of agents. In this paper, we present a novel approximate core allocation algorithm, called the bounding boxed core (BBC), for large cooperative security games in characteristic form that rely on superadditivity. The proposed algorithm is an *anytime* (an algorithm is called *anytime* if it can be interrupted at any time point during execution to return an answer whose value, at least in certain classes of stochastic processes, improves in expectation as a function of the computation time) algorithm based on iterative state space search for better solutions. Experimental results on a 25-player game, with roughly 34 million coalitions, show that BBC shrinks the 25-dimensional bounding-box to 10^{-15} times its initial hypervolume.

1. Introduction

In the recent years, there has been a growing interest for modeling defenders in different security problems with coalition games. This is mainly due to results which confirm that many security goals can be better reached through cooperation among the interested parties. Millions of connected computers and networks of them have turned security to a problem characterized by interdependence [4]. In this interconnected world of computers, the security of a particular user is not independent of others and it heavily depends on the efforts of other users. As a prominent example in Internet security, in combating spam and unsolicited communications, the Organization for Economic Co-operation and Development recommends international cooperation and promotes cross-border enforcement cooperation on spam-related problems [18].

With regard to cyber attacks originated from any particular country, a recent study [13] shows that international

cooperation in enforcement as measured by the indicator of joining the convention on cybercrimes, can deter cyber attacks by up to 24%. Cooperative attack and defense for a range of distributed network applications, with an emphasis on phishing, has been examined in [23] and the results highlight the need for cooperative information sharing.

Building suitable models based on cooperation for a variety of security problems has been the subject of some recent studies. Coalition game theory has been used for security risk management [22] in divisions of an organization as well as secure wireless transmissions [16] in ad hoc wireless networks, and against network security attacks [5]. In detecting [12, 19] and mitigating [21] DDos attacks, cooperation has been affirmed. While many studies advocate the use of coalition games for modeling large scale security problems, the question of revenue allocation, as an important step in turning these game theoretical models to practical robust solutions, has not been studied and warrants greater attention.

Cooperative game theory studies the problem of revenue allocation for a set of n participants, called players, in a joint project where a value function v is defined for each subset of players, representing the revenue achieved by the players in that subset without assistance of other players. In the context of security, we call these games *cooperative security games*. In cooperative security games players are the defenders (e.g., organizations cooperating for intrusion detection) and revenue is the security benefit they achieve through cooperation. We will present a formal definition in the next Section. In general, each solution concept defines a set $\mathcal{X}$ of allocation vectors. Obtaining $\mathcal{X}$ is usually a nontrivial problem and has always been an important issue in the study of cooperative game theory. A particularly interesting theme to the study of such decision problems is that of the bounded rationality, which argues that decisions made by real-life agents may not spend an unbounded amount of resources to evaluate all the possibilities for optimal outcome [17]. Much effort has been made in the study of the bounded rationality in computational resource for solution concepts of cooperative games.

The definition of a cooperative game involves an exponential number (in the number of players) of values, one for each subset of players. Moreover, the definitions of many solution concepts, for example, the core [1], would involve an exponential number of constraints. Therefore, it becomes infeasible to get the set of fair allocation vectors $\mathcal{X}$ for a large cooperative game using traditional approaches, such as linear programming. Megiddo [25] observed that, for many games, the game value is calculated through succinctly defined structures and for such games, he suggested that finding a solution should be done by a good algorithm (following Edmonds [28]), that is, within time polynomial in the number of players. Deng and Papadimitriou [14] suggested computational complexities be taken into consideration as another measure of fairness for evaluating and comparing different solution concepts.

In this paper, we present bounding-boxed core (BBC), an approximate revenue allocation algorithm for large cooperative security games that are intractable to be solved using traditional algorithms such as linear programming. Previous work in security revenue allocation investigate all coalitions and their corresponding values before computing the core. This is not feasible with large number of players, that is, organizations that attempt to protect security of their assets, because increasing number of players result in exponential growth of the coalition space. Our algorithm, to the best of our knowledge, is the first algorithm to provide an approximate solution to this problem. In addition, we utilize the special characteristics of the core, namely, its convexity, to analyze the approximation error and provide bounds on the worst case errors.

This paper is organized as follows. In Section 2, we describe the security game model and the revenue allocation problem. In Section 3 we present an overview of related background literature. The BBC algorithm is briefly explained in Section 4. Section 5 is devoted to the bounding box shrinking algorithm. Approximate core allocation approach is discussed in Section 6. Finally, experimental results for a

case study cooperative game with 25 players is presented in Section 7 and Section 8 concludes the paper.

2. Security Model and Basic Definitions

We start by describing a simplified security problem and subsequently formally define a cooperative security game. Assume a finite set of organizations denoted by $\mathcal{N} = \{1, 2, \dots, n\}$. Each organization has its own security resources and valuable assets which it protects against security attacks. Any subset $S \subseteq \mathcal{N}$ is called a coalition. The *security profit* function $v : \mathcal{P}(\mathcal{N}) \rightarrow \mathbb{R}$ is a function with $v(\emptyset) = 0$, where $\mathcal{P}(\mathcal{N})$ denotes the power set of $\mathcal{N}$. For each coalition $S \subseteq \mathcal{N}$, $v(S)$ is the value of S that is interpreted as the security profit achieved by the collective action of organizations in S without any assistance of organizations in $\mathcal{N} - S$. A security cooperative game is given by specifying the security profit function, that is, a value for every coalition. The security profit can be defined based on different measures, such as security resources of a coalition, the type and probability of successful attacks against these security measures, and the expected cost of such attacks. Instead of a security profit function, a *security cost* function can be used. In this paper we only deal with profit games, however, symmetric statement holds for cost games. Security profit functions are assumed to be superadditive, that is, the value of a union of disjoint coalitions is no less than the sum of the coalitions' separate values:

$$v(S) + v(T) \leq v(S \cup T) \quad \forall S, T \subseteq \mathcal{N} \ni S \cap T = \emptyset. \quad (1)$$

Given a pair $(\mathcal{N}, v)$, the focus is how to fairly distribute the collective income. We denote the income distributed to individual organizations by a payoff vector $x = \{x_1, x_2, \dots, x_n\}$ satisfying $\sum_{i \in \mathcal{N}} x_i = v(\mathcal{N})$, called an allocation. An allocation vector x is called an *imputation* of the game $(\mathcal{N}, v)$ if it also satisfies the individual rationality condition:

$$v(\{i\}) \leq x_i \quad \forall i \in \mathcal{N}. \quad (2)$$

As an illustrative example, consider a set of online shopping sites which are willing to cooperate against intrusion detection attacks by sharing worm and virus signatures identified by their antivirus systems. The valuable assets in this scenario consists of a database of credit-card information. We consider a dollar value as the security profit associated with an attack. Let us consider two of these shopping sites $\{1, 2\}$. Let the expected security profit of site 1 be $v(\{1\}) = c_1 - 300$, while this value is $v(\{2\}) = c_2 - 50$ for site 2. If they cooperate by sharing their signatures, the expected security profit of their coalition would be $v(\{1, 2\}) = c_1 + c_2 - 400$. The allocation problem involves assigning x_1 and x_2 in a fair manner, which also motivates the formation of a coalition.

Additional requirements for fairness, stability, and rationality lead to different sets of allocations, which are generally referred to as solution concepts. Here, we shall discuss the most important one that is the core.

Sum of the coalitions' separate values.

The concept of the core was first introduced by Gillies [1] based on the concept of subgroup rationality.

Definition 1. The core of a game $(\mathcal{N}, v)$ is defined by

$$C(v) = \left\{ x \in \mathbb{R}^{|\mathcal{N}|} : \sum_{i \in \mathcal{N}} x_i = v(\mathcal{N}); \right. \\ \left. \sum_{i \in S} x_i \geq v(S) \quad \forall S \subseteq \mathcal{N} \right\}. \quad (3)$$

The constraints imposed on $C(v)$ ensure that no coalition would have an incentive to split from the grand coalition $\mathcal{N}$ and do better on its own. The core is a subset of the hyperplanes defined by the equation $\sum_{i \in S} x_i = v(S)$ [16], and since the inequalities $x_i \geq v(\{i\})$ are included in (3), the core is bounded. Thus, $C(v)$ is a compact convex polyhedron, and possibly empty, of dimension at most $n - 1$. The definition of the core, involves an exponential number of constraints, therefore, exact computation of the core is infeasible in large scale games. In this paper, we present an approximate algorithm for core allocation.

3. Related Work

In the security community, there has been considerable effort in introducing models based on cooperation for defenders of a security attack, see for example [12, 19, 21, 23]. Here, we review the ones which explicitly use coalition games as their models. The authors in [22] define a coalition game for modeling risk management. In their model, divisions of an organization are the players of the game. They use an influence graph to specify the positive and negative effects that each division of the organization can have on other divisions. The security benefit of each coalition is then defined as a function involving the influence graph, the security resources of each division, and threats against vulnerabilities. They further enrich their model by introducing a friction graph which denotes the costs associated with cooperation. They analyze the formation of coalitions based on this model. Coalition game theory has also been used in modeling secure transmissions in a wireless network [16]. In this problem a set of wireless transmitters try to transmit a message to a set of destinations in presence of eavesdroppers. The goal is to avoid the overhead of cryptography and self-organize the wireless transmitter such that secure wireless communication can be achieved. Coalition of transmitters can decode-and-forward (DF) or amplify-and-forward (AF) a signal, and in this way, completely null the signal at the eavesdroppers. In another recent study, [5] models defenders of a network security attack with coalition games. They analyze coalition formation in three canonical security games described previously in [26] for non-cooperative games.

The above mentioned papers mostly consider the problem of coalition formation. In this paper, we consider another important problem, namely, revenue allocation. Much of cooperative game theory is built around the question of distributing the collective income in fair and rational manners. Different philosophies result in different solution concepts that constitute the bargaining set family, that is, the various bargaining sets [15, 24], the Kernel [20]

and the Nucleolus [11]; in effect, these notions are defined separately for each coalition structure. By contrast, the Shapley value [3], Core [1], and Von Neumann-Morgenstern solutions [6] are not a priori defined with reference to a coalition structure. In this part, we review related literature on determining the core.

One of the most important problems regarding the core is *testing nonemptiness* that is determining whether a given instance of the game has a nonempty core. Generally, determining core nonemptiness is NP-complete for cooperative games both with and without transferable utility [8]. However, recently there has been increasing interest in investigating this problem in more specific games with a description polynomial in n (number of players). As a case in point, the core nonemptiness testing problem is solvable in polynomial time for weighted graph games [14]; for minimum base game with no all-negative circuits can be solved in oracle-polynomial time [10].

Since the core is usually empty, some related solution concepts arise from the core via relaxing its constraints. Shapley and Shubik [27] recommended the concepts of (strong) ε -core and weak ε -core for a cooperative game. Their main idea is to relax the requirements of subgroup rationality by $x(S) \geq v(S) - \varepsilon$ and $x(S) \geq v(S) - \varepsilon|S|$ for each proper subset S of $\mathcal{N}$, respectively.

Later, Tijs and Driesssen [7] introduced the concept of multiplicative ε -tax core by using $x(S) \geq v(S) - \varepsilon[v(S) - \sum_{i \in S} v(\{i\})]$ instead. Faigle and Kern [9] modified the requirement of Tijs and Driesssen as $x(S) \geq (1 - \varepsilon)v(S)$ to define another approximate core, called multiplicative ε -core. One explanation of these concepts is that cooperation may not be as hopeless even when the core is empty. Cooperation may be possible with the subsidies of the central authority.

All of the above mentioned algorithms need to investigate all coalitions and their corresponding values before computing the core; however, this is not feasible in cooperative games with large number of players due to the exponential growth of the coalition space.

4. The Bounding Boxed Core

The bounding-boxed Core (BBC) algorithm is a practical best-effort approach to distribute the collective security profit in an approximately fair manner among the involved organizations (players) in a given large cooperative security game $(\mathcal{N}, v)$.

More specifically, BBC consists of two main stages: tightening bounding box, and approximate core allocation. The former is an iterative global search algorithm to solve an axis-aligned smallest enclosing box problem. It finds the minimum-hypervolume n -dimensional bounding box enclosing the core, that is, closed and convex set, subject to the constraint that the edges of the box are parallel to the (Cartesian) coordinate axes. Starting from some initial large bounding box, BBC iteratively tries to tighten the bounding box, obtained at the previous iteration, by heuristically considering some useful subset of constraints by the core (3).

Consequently, a bounding box $B = [a_1, b_1] \times [a_2, b_2] \times \dots \times [a_n, b_n]$ is generated once a given deadline is passed or some convergences condition is met. In the approximate core allocation step, BBC computes a unique payoff vector in the generated bounding box B . The allocation values for individual players are determined by calculating the gravity center of the resulting bounding box B .

5. Bounding Box Tightening

In this section, the iterative bounding box tightening algorithm is explained in details. The goal is to find the minimum-hypervolume bounding box enclosing the core $B(C(v))$ using an iterative search algorithm. We use B_t to denote the bounding box, obtained at t -th iteration step. Initially, we know that player i gets a payoff of $x_i \in [\nu(\{i\}), \nu(\mathcal{N}) - \nu(\mathcal{N} - \{i\})]$ in the core $C(v)$. The lower bound denotes individual rationality, that is, no player receives less than what it could get on its own; and the upper bound is obtained using one of the core constraints:

$$\sum_{j \in \mathcal{N} - \{i\}} x_j \geq \nu(\mathcal{N} - \{i\}), \quad (4)$$

along with the efficiency condition:

$$\sum_{i \in \mathcal{N}} x_i = \nu(\mathcal{N}). \quad (5)$$

Therefore, we assign as shown in (13) to be the initial, that is, $t = 0$, bounding box enclosing the core. Throughout this paper, we will use notations $B_t^l(i)$ and $B_t^u(i)$ to denote the lower and upper bounds for the i th player in the t th bounding box.

The initial bounding box B_0 is further tightened during several iterations. First, we describe the single iteration t of the tightening procedure given a permutation $\pi \in \Pi$ on the set of players:

$$\pi = (\pi(1), \pi(2), \dots, \pi(n)), \quad (6)$$

where, π is in fact a one-to-one function from $\mathcal{N}$ onto $\mathcal{N}$. Moreover, as a shorthand notation, we use π_i^j ($i \leq j$) to denote $(\pi(i), \pi(i+1), \dots, \pi(j))$.

5.1. Recursive Tree Construction. At each iteration t , tightening algorithm starts with constructing a full binary tree on a given permutation of players π , in a top-down manner. The root node, which covers all players π_1^n , has two children nodes denoting two disjoint subsets of $\mathcal{N}$: $\pi_1^{i_0}$ and $\pi_{i_0+1}^n$ ($1 \leq i_0 < n$). Breaking point i_0 is *optimally* determined in a way that the bounding box is tightened the most (discussed later). Recursively, children nodes are further partitioned into two disjoint subsets of their corresponding players: $\pi_1^{i_1}, \pi_{i_1+1}^{i_0} \subseteq \pi_1^{i_0}$, and $\pi_{i_0+1}^{i_2}, \pi_{i_2+1}^n \subseteq \pi_{i_0+1}^n$. Leaf nodes of the tree represent individual players, for example, $\pi_i^i = (\pi(i))$.

To determine the optimal breaking point, let us consider an arbitrary parent node which covers a subset of players π_i^k . Suppose j ($i \leq j < k$) is the breaking point so that π_i^k is

partitioned into children nodes $\pi_i^j, \pi_{j+1}^k \subseteq \pi_i^k$. Using the core constraints (3), and the bounding box B_{t-1} , obtained at the previous iteration $t-1$, we know

$$\sum_{q \in \pi_i^k} x_q \leq \sum_{q \in \pi_i^k} x_q \leq \overline{\sum_{q \in \pi_i^k} x_q}, \quad (7)$$

where in the root node both bounds are equal to $\nu(\mathcal{N})$, but in lower level nodes

$$\sum_{q \in \pi_i^k} x_q = \max \left(\sum_{q \in \pi_i^k} B_{t-1}^l(q), \nu(\pi_i^k) \right), \quad (8)$$

$$\overline{\sum_{q \in \pi_i^k} x_q} = \min \left(\sum_{q \in \pi_i^k} B_{t-1}^u(q), \nu(\mathcal{N}) - \nu(\mathcal{N} - \pi_i^k) \right).$$

Similar conditions hold for children nodes, that is, π_i^j, π_{j+1}^k . Furthermore, each child node can make use of its sibling's bounds to update, that is, tighten, its own bounds. As a case in point, we update bounds on the left child, that is, π_i^j :

$$\mathcal{L}_{\pi_i^j} \leq \sum_{q \in \pi_i^j} x_q \leq \mathcal{U}_{\pi_i^j}, \quad (9)$$

where,

$$\mathcal{L}_{\pi_i^j} = \max \left(\sum_{q \in \pi_i^j} x_q, \sum_{q \in \pi_i^k} x_q - \overline{\sum_{q \in \pi_j^k} x_q} \right), \quad (10)$$

$$\mathcal{U}_{\pi_i^j} = \min \left(\overline{\sum_{q \in \pi_i^j} x_q}, \overline{\sum_{q \in \pi_i^k} x_q} - \sum_{q \in \pi_j^k} x_q \right).$$

The right child also updates her bounds using the same justification. Consequently, the optimal breaking point j^* is determined using

$$j^* = \arg \max_{x_i \leq j < k, j \in \mathbb{Z}} (\mathcal{T}_i^j + \mathcal{T}_{j+1}^k), \quad (11)$$

where,

$$\mathcal{T}_i^j = \left[\left(\overline{\sum_{q \in \pi_i^j} x_q} - \sum_{q \in \pi_i^j} x_q \right) - (\mathcal{U}_{\pi_i^j} - \mathcal{L}_{\pi_i^j}) \right], \quad (12)$$

and $\mathcal{T}_{j+1}^k$ is defined similarly. Once the optimal breaking point j^* is determined: (1) players in the parent node π_i^k are partitioned into two children coalitions $\pi_i^{j^*}, \pi_{j^*+1}^k \subseteq \pi_i^k$; (2) upper and lower bounds on children are updated. As a result, recursive tree construction procedure is terminated when it gets to leaf nodes, that is, the players' payoffs, and updates their upper and lower bounds.

Figure 1 shows how the recursive tree construction algorithm updates the initial bounding box for a sample 3-player game, that is, $\mathcal{N} = \{1, 2, 3\}$. Characteristic values for

FIGURE 1: A Sample 3-player game: tightening the initial boundingbox.

different coalitions of the game are given in the Figure 1. The initial bounding box enclosing the core

$$B_0 = ([v(\{1\}), v(N) - v(N - \{1\})], \dots, [v(\{n\}), v(N) - v(N - \{n\})]) \quad (13)$$

is obtained as $0 \leq x_i \leq 8 \forall i \in N$ that results in $0 \leq x_1 + x_3 \leq 16$ as shown in Figure 1. The updated values are illustrated in red font.

5.2. Iterative Search. In the previous section, we described how a given permutation of players is used in constructing a binary tree for tightening the bounding box. The permutation used, can be chosen at random, or as we describe in this section, it can be produced by an evolutionary algorithm which tends to generated permutations which result in larger reductions of the bounding box volume. During each iteration t of the algorithm, we have a population P , for example, $P = 100$, of permutations $\Pi_t \subseteq \Pi$, which are used for tightening the bounding box as explained in Section 5.1. Once the bounding box around feasible point in the core is updated, the fitness function is calculated as follows for each permutation in current population $\pi \in \Pi_t$ and measures its quality, that is, how helpful it was in tightness improvement:

$$lcl\psi_t(\pi) = \sum_{i \in N} \left| [B_{t,a_\pi}^u(i) - B_{t,a_\pi}^l(i)] - [B_{t,b_\pi}^u(i) - B_{t,b_\pi}^l(i)] \right|, \quad (14)$$

where B_{t,b_π} (B_{t,a_π}) denote bounds, at iteration t , just before (after) they are updated using recursive tree construction on permutation π . Once we have the population and the fitness function defined, iterative search algorithm proceeds to initialize a population of permutations randomly, then improves tightness of the bounding box through repetitive application of mutation, crossover, inversion and selection operators (discussed later).

Initialization. Initially, P permutations are randomly generated from the search space Π to form an initial population.

Selection. During each iteration t , a proportion 2α ($\alpha \in [0, 0.5]$) of the existing population Π_t is selected to breed a new generation of permutations Π_{t+1} . Individual permutations are selected through a fitness-based process, where top

$2\alpha \cdot P$ fittest permutations, as measured by $\psi_t(\cdot)$ (14), are selected.

Reproduction. The next step is to generate the next generation population of permutations Π_{t+1} from those selected. For each new permutation to be produced, a pair of *parent* permutations, that is, π_{p_1} and π_{p_2} , is selected for breeding from the pool selected previously. By producing a *child* permutation π_{child} using function composition, that is, $\pi_{\text{child}}(i) = \pi_{p_1} \circ \pi_{p_2}(i)$, a new permutation is created. New parents are selected for each child, that is, $\alpha \cdot P$ children permutations. Furthermore, $(1 - \alpha) \cdot P$ new permutations are randomly generated. These processes ultimately result in the next generation population Π_{t+1} of permutations. Generally, the average fitness will have increased by this procedure for the population, since only the most useful permutation from the previous generation are selected.

Termination. This iterative search process is repeated until a termination condition has been reached. Terminating conditions can be defined regarding various criteria, such as: (1) all possible permutations Π are investigated; (2) a predefined deadline is passed $\text{deadline} < t$; (3) tightness improvement in the last iteration t was less than some nonnegative threshold ϵ , that is, $\sum_{\pi \in \Pi_t} \psi_t(\pi) \leq \epsilon$.

6. Approximate Core Allocation

Once the bounding box around feasible points in the core is generated (see Section 5) $B = [a_1, b_1] \times [a_2, b_2] \times \dots \times [a_n, b_n]$, the candidate approximate core is computed as follows:

$$C_B(v) = \left(\frac{a_1 + b_1}{2}, \frac{a_2 + b_2}{2}, \dots, \frac{a_n + b_n}{2} \right). \quad (15)$$

However, $C_B(v)$ may not be in the actual core; therefore, here we analyze the worst case error, that is, distance between the core $C(v)$ and the approximate $C_B(v)$. First, we need to define the distance between the point $C_B(v)$ and the core $C(v)$, that is, a convex set of points:

$$d(C_B(v), C(v)) = \min_{x \in C(v)} d(x, C_B(v)). \quad (16)$$

In other words, the distance between a point and a set is the infimum of the distances between the point and those in the set.

Here, we exploit the convexity of the core in analyzing the worst case error of the the approximate core, $C_B(v)$, that is defined as follows

$$\Upsilon(B) = \max_{Q \in \mathcal{C} \text{ s.t. } B(Q)=B} d(C_B(v), Q), \quad (17)$$

where $\mathcal{C}$ is the convex set space in $\mathbb{R}^{|\mathcal{N}|}$; and $B(Q)$ denotes the minimum-volume axis-aligned bounding box around the set Q .

We first obtain Υ when $|\mathcal{N}| = 3$, and then, generalize the result to higher dimensional spaces. Given $B = [a_1, b_1] \times [a_2, b_2] \times [a_3, b_3]$, w.l.o.g. we transfer B to the origin using the linear transformation $B' = B - ((a_1 + b_1)/2, (a_2 + b_2)/2, (a_3 + b_3)/2)$; therefore, we get the bounding box B' that consists of $2 \times 3 = 6$ planes:

$$B' = \left[-\frac{\Delta_1}{2}, \frac{\Delta_1}{2}\right] \times \left[-\frac{\Delta_2}{2}, \frac{\Delta_2}{2}\right] \times \left[-\frac{\Delta_3}{2}, \frac{\Delta_3}{2}\right], \quad (18)$$

and $C_{B'}(v) = (0, 0, 0)$, where $\Delta_i = b_i - a_i$. As discussed earlier (Section 2), the core is a convex subset of the hyperplanes. It can be shown that the worst case error $\Upsilon(B')$ (see (17)) is caused by $Q \in \mathcal{C}$ whose closest hyperplane H_Q to the point $C_{B'}(v)$ includes those 3 vertices of B' , any pair of which have exactly one of the 6 bounding box planes in common:

$$\begin{aligned} p_1 &= \left(\frac{\Delta_1}{2}, -\frac{\Delta_2}{2}, \frac{\Delta_3}{2}\right), \\ p_2 &= \left(-\frac{\Delta_1}{2}, \frac{\Delta_2}{2}, \frac{\Delta_3}{2}\right), \\ p_3 &= \left(\frac{\Delta_1}{2}, \frac{\Delta_2}{2}, -\frac{\Delta_3}{2}\right); \end{aligned} \quad (19)$$

therefore,

$$H_Q : x + \frac{\Delta_1}{\Delta_2}y + \frac{\Delta_1}{\Delta_3}z - \frac{\Delta_1}{2} = 0. \quad (20)$$

Hence, $\Upsilon(B')$ is obtained as the distance between the point $C_{B'}(v)$ and H_Q :

$$\Upsilon(B') = \frac{1}{2 \times \sqrt{1/\Delta_1^2 + 1/\Delta_2^2 + 1/\Delta_3^2}}. \quad (21)$$

As a case in point, let $\Delta_i = \Delta \quad \forall i \in \mathcal{N}$. Without considering convexity of the core, we know $\Upsilon(B') = \Delta \cdot \sqrt{3}/2$ that is the distance between any of the vertices of B' and its gravity center $C_{B'}(v)$. But, exploiting the convexity of the core (21) yields a better result $\Upsilon(B') = \Delta \cdot \sqrt{3}/6 < \Delta \cdot \sqrt{3}/2$.

For higher dimensional spaces, that is, games with more than 3 players, discussions remain exactly the same and the worst case error is obtained as:

$$\begin{aligned} \Upsilon(B') &= \frac{1}{2 \times \sqrt{1/(b_1 - a_1)^2 + 1/(b_2 - a_2)^2 + \dots + 1/(b_n - a_n)^2}}. \end{aligned} \quad (22)$$

FIGURE 2: Number of coalitions in each coalition class with particular no. of players.

FIGURE 3: Maximum characteristic value distribution on coalition classes.

7. Evaluations

In this section, we present the results from our prototype implementation of BBC on a case study large cooperative game. The system we used for implementation was a 2.20 GHz AMD Athlon 64 processor 3700+ with 2.00 GB of memory and Windows XP SP3 operating system. Because of large number of coalitions the bottleneck is usually the memory that caused our first prototype to crash; hence, we reimplemented the whole algorithm in C++ from scratch using more sophisticated data structures to speed up the iterative search and reduce memory consumption.

Here, we present the evaluation results of the BBC algorithm for a cooperative game with 25 players in which there are about 34 million coalitions of players.

Figure 2 shows the number of coalitions in each coalition class that is defined based on its size, that is, number of players. As discussed earlier, every cooperative game is determined by its characteristic function for each coalition. Figure 3 shows the maximum characteristic value for each coalition class. As shown in the figure, value for grand coalition is 1604. Furthermore, coalition values satisfy superadditivity property.

Given the characteristic values for the cooperative game, we start iterative tightening of the 25-dimensional bounding

FIGURE 4: Bounding box tightening result for each dimension in $\mathbb{R}^{25}$.

FIGURE 5: Bounding box volume improvement during iterative search.

box. Figure 4 illustrates the initial lower and upper bounds for individual dimensions, that is, players' payoffs, of the initial bounding box, as in (13), and their corresponding updated bounds. In this experiment, we assigned $P = 100$ and $\alpha = 0.3$.

Finally, bounding box volume reduction, during the iterative tightening algorithm, is shown in Figure 5 with 10^{20} scale. As illustrated in the logarithmic-scale graph, although the bounding box volume has shrank more than 10^{12} units during only a few seconds, tightening speed significantly decreases afterwards. Consequently, after about 25 minutes, bounding-box volume is 10^{-15} times its initial value.

8. Conclusion

In this paper, we present an approximate core allocation algorithm, called the bounding-boxed core (BBC), for cooperative security games with large number of organizations (players). Since the definition of core involves an

exponential number of constraints in terms of the number of organizations, it is infeasible to solve this problem for exact solutions. We present an analysis of the maximum approximation error incurred by BBC. We also proposed a heuristic search algorithm based on Genetic algorithms to search the input space. Experimental results show that BBC significantly speeds up core allocation. In modeling cooperation among large number of organizations, this is an important step in achieving a practical solution in limited time.

References

- [1] D. B. Gillies, "Solutions to general non-zero-sum games," in *Contributions to the Theory of Games 4*, R. D. Luce and A. W. Tucker, Eds., Annals of Mathematics Studies, pp. 47–85, Princeton, 1959.
- [2] L. S. Shapley, "Cores of convex games," *International Journal of Game Theory*, vol. 1, no. 1, pp. 11–26, 1971.
- [3] L. S. Shapley, "A value for n-person games," in *Contributions to the Theory of Games 2*, H. W. Kuhn and A. W. Tucker, Eds., Annals of Mathematics Studies, pp. 307–317, Princeton, 1953.
- [4] H. Kunreuther and G. Heal, "Interdependent Security," *Journal of Risk and Uncertainty*, vol. 26, no. 2-3, pp. 231–249, 2003.
- [5] H. Narasimhan, V. Varadarajan, and U. Rangan, "Towards a cooperative defense model against network security attacks," in *Proceeding of the 9th Workshop on the Economics of Information Security (WEIS '10)*.
- [6] J. V. Neumann and O. Morgenstern, *Theory of Games and Economic Behaviour*, Princeton, 1944.
- [7] S. H. Tijs and T. S. H. Driessen, "Extensions of solution concepts by means of multiplicative ϵ -tax games," *Mathematical Social Sciences*, vol. 12, no. 1, pp. 9–20, 1986.
- [8] V. Conitzer and T. Sandholm, "Complexity of determining nonemptiness of the core," in *Proceedings of the 4th ACM Conference on Electronic Commerce*, pp. 230–231, June 2003.
- [9] U. Faigle and W. Kern, "On some approximately balanced combinatorial cooperative games," *Methods and Models of Operations Research*, vol. 38, no. 2, pp. 141–152, 1993.
- [10] H. Nagamochi, D. Z. Zeng, N. Kabutoya, and T. Ibaraki, "Complexity of the minimum base game on matroids," *Mathematics of Operations Research*, vol. 22, no. 1, pp. 146–164, 1997.
- [11] D. Schmeidler, "The nucleolus of a characteristic function game," *SIAM Journal of Applied Mathematics*, vol. 17, no. 1163, 1170 pages, 1969.
- [12] D. Frincke, D. Tobin, J. Mcconnell, J. Marconi, and D. Polla, "A framework for cooperative intrusion detection," in *Proceedings of the 21st NIST-NCSC National Information Systems Security Conference*, pp. 361–373, 1998.
- [13] Q. Hong Wang and S. Hyun Kim, "Cyber attacks: Cross-country interdependence and enforcement," in *Proceeding of the 8th Workshop on the Economics of Information Security (WEIS 2009)*.
- [14] X. Deng and C. Papadimitriou, "On the complexity of cooperative game solution concepts," *Mathematics of Operations Research*, vol. 19, pp. 257–266, 1994.
- [15] M. Davis and M. Maschler, "Existence of stable payoff configurations for cooperative games," in *Essays in Mathematical Economics in Honor of Oskar Morgenstern*, M. Shubik, Ed., pp. 39–52, Princeton, 1967.

- [16] W. Saad, Z. Han, T. Başar, M. Debbah, and A. Hjørungnes, "Distributed coalition formation games for secure wireless transmission," *Mobile Networks and Applications*, vol. 16, no. 2, pp. 231–245, 2011.
- [17] H. A. Simon, "Theories of bounded rationality," in *Decision and Organization*, C. Radner and R. Radner, Eds., pp. 161–176, 1972.
- [18] Report of the OECD task force on spam, "Antispam toolkit of recommended policies and measures," Tech. Rep., Directorate for Science, Technology and Industry, Committee on Consumer Policy Committee for Information, Computer and Communications Policy, 2006.
- [19] F. Cuppens and A. Miège, "Alert correlation in a cooperative intrusion detection framework," in *Proceedings of the Symposium on Security and Privacy*, pp. 202–215, IEEE Computer Society, Washington, DC, USA, May 2002.
- [20] M. Davis and M. Maschler, "The kernel of a cooperative game," *Naval Research Logistics*, vol. 12, pp. 223–259, 1965.
- [21] D. Nojiri, J. Rowe, and K. Levitt, "Cooperative response strategies for large scale attack mitigation," in *Proceedings of the 3rd DARPA Information Survivability Conference and Exposition (DISCEX '03)*, pp. 293–302, 2003.
- [22] W. Saad, T. Alpcan, T. Başar, and A. Hjørungnes, "Coalitional game theory for security risk management," in *Proceedings of the 5th International Conference on Internet Monitoring and Protection (ICIMP '10)*, pp. 35–40, Washington, DC, USA, May 2010.
- [23] T. Moore, "Cooperative attack and defense in distributed networks," Tech. Rep. UCAM-CL-TR-718, Computer Laboratory, University of Cambridge, 2008.
- [24] R. J. Aumann and M. Maschler, "The bargaining set for cooperative games," in *Advances in Game Theory*, M. Dresher, L. S. Shapley, and A. W. Tucker, Eds., Annals of Mathematics Studies, pp. 443–476, Princeton, 1964.
- [25] N. Megiddo, "Computational complexity and the game theory approach to cost allocation for a tree," *Mathematics of Operations Research*, vol. 3, pp. 189–196, 1978.
- [26] J. Grossklags, N. Christin, and J. Chuang, "Secure or insure? a game-theoretic analysis of information security games," in *Proceedings of the 17th International Conference on World Wide Web (WWW '08)*, pp. 209–218, New York, NY, USA, April 2008.
- [27] L. Shapley and M. Shubik, "Quasi-cores in a monetary economy with nonconvex preference," *Econometrica*, vol. 34, pp. 805–827, 1966.
- [28] J. Edmonds, "Path, tree, and flowers," *Canadian Journal of Mathematics*, vol. 17, pp. 449–469, 1965.