Computational Analysis of Connectivity Games with Applications to the Investigation of Terrorist Networks

Tomasz P. Michalak
University of Oxford, UK
University of Warsaw, Poland

Talal Rahwan and **Nicholas R. Jennings**
University of Southampton, UK

Piotr L. Szczepański
Warsaw University of Technology, Poland

Oskar Skibski
University of Warsaw, Poland

Ramasuri Narayanam
IBM Research, India

Michael J. Wooldridge
University of Oxford, UK

Abstract

We study a recently developed centrality metric to identify key players in terrorist organisations due to Lindelauf *et al.* [2013]. This metric, which involves computation of the Shapley value for *connectivity games on graphs* proposed by Amer and Gimenez [2004], was shown to produce substantially better results than previously used standard centralities. In this paper, we present the first computational analysis of this class of coalitional games, and propose two algorithms for computing Lindelauf *et al.*'s centrality metric. Our first algorithm is exact, and runs in time linear by number of connected subgraphs in the network. As shown in the numerical simulations, our algorithm identifies key players in the WTC 9/11 terrorist network, constructed of 36 members and 125 links, in less than 40 minutes. In contrast, a general-purpose Shapley value algorithm would require weeks to solve this problem. Our second algorithm is approximate and can be used to study much larger networks.

1 Introduction

Facing, on the one hand, the increased size of terrorist groups and, on the other hand, inevitable budget cuts, security agencies urgently require efficient techniques to identify who plays the most important role within a terrorist network and, therefore, where scarce resources should predominantly be focused. In this context, a number of authors have proposed using *social network analysis* to investigate terrorist organisations (*e.g.*, [Carley *et al.*, 2003; Krebs, 2002; Farley, 2003; Lindelauf *et al.*, 2013; Ressler, 2006]). Analyst's Notebook 8 [I2, 2010]—a software package used worldwide by law enforcement and intelligence agencies—has recently included standard centrality metrics for networks (*graphs*), such as degree, closeness and betweenness centralities [Brandes and Thomas, 2005; Friedkin, 1991]. But the usefulness of these metrics for terrorist networks is limited as they are often unable to capture the complex nature of these organisations [Lindelauf *et al.*, 2013].

Recently, in an attempt to address these shortcomings, Lindelauf *et al.* [2013] developed a more advanced method specifically designed to measure centrality in terrorist networks. The new method belongs to a class of so-called *Shapley value-based centrality metrics*[1] and builds upon the notion of coalitional *connectivity games* proposed by Amer and Gimenez [2004].

Unfortunately, the computational aspects of connectivity games by Amer and Gimenez have not been studied to date. This means that the current use of Lindelauf *et al.*'s method is limited only to small terrorist networks (of 25 members or so) because general-purpose algorithms for coalitional games have to be applied, which exhaustively search the space of all possible coalitions. Thus, they are inapplicable to many real-world applications such as the terrorist networks responsible for the WTC 9/11 attack (of 35, 63 or more nodes depending on the considered type of links between terrorist).

Against this background, we provide in this paper the first computational analysis of connectivity games proposed by Amer and Gimenez:

- We prove that computing the Shapley value in connectivity games—including the centrality metrics of Lindelauf *et al.*—is #P-Hard.
- We propose a *dedicated exact algorithm* for computing these centrality metrics. While the general-purpose Shapley value algorithm requires checking all subsets of vertices in the graph, our algorithm traverses through (most often) much smaller number of connected subgraphs. It also has minimal memory requirements.
- We test our algorithm by analysing the aforementioned WTC 9/11 terrorist network with 36 members and 125 identified connections. In this setting, our algorithm returns the solution within 38 minutes, compared to weeks if a general-purpose approach was applied.
- In order to study even bigger networks, we propose a *dedicated approximate algorithm* based on Monte Carlo sampling.

[1] See the next section for more details.

2 Connectivity Games for Terrorist Networks

Terrorist networks have been recently modelled using a weighted graph, G, composed of *vertices* (or *nodes*, *i.e.*, individual terrorists) and labelled *edges* [Carley *et al.*, 2003; Krebs, 2002; Farley, 2003; Lindelauf *et al.*, 2013; Ressler, 2006]. Based on available intelligence, an edge represents, for instance, a communication link between two terrorists, and the weight of the edge represents the frequency with which that link is used. Weights can be associated not only with edges but also with vertices. This, as argued by Lindelauf *et al.*, allows for modelling additional information that intelligence agencies gather on individuals within the network.

We will denote the set of all vertices in the graph by $V(G)$ and the set of all edges by $E(G)$, respectively, where every edge in $E(G)$ connects two vertices in $V(G)$. An edge connecting vertices $v_i, v_j \in V(G)$ will be denoted (v_i, v_j), and its label (or weight) will be denoted $\omega(i,j) \in \Omega(G)$. The weight of vertex $v_i \in V(G)$ will be denoted $\gamma(i) \in \Gamma(G)$.

To address the limitations of standard centrality metrics, Lindelauf *et al.* proposed a new metric that builds upon *connectivity games* by Amer and Gimenez [2004]. In these coalitional games on graphs, all subsets of set $V(G)$ are considered to be possible coalitions of terrorists. Apart from the empty set, every single coalition is classified as belonging to either the set of *connected coalitions* (denoted $\mathcal{C}(G)$) or *disconnected coalitions* (denoted $\tilde{\mathcal{C}}(G)$). We say that C is connected if between any two nodes in C there exists at least one path of which all nodes belong to C. Otherwise C is disconnected. Importantly, any two terrorists in a connected coalition are able to communicate with each other (*via* a path), whereas in a disconnected coalition this is not the case. In many connected coalitions, from this point of view, some nodes play more important role than others, as their removal makes a coalition disconnected. We will call them *pivotal*.

Definition 1 *Given connected coalition $C \in \mathcal{C}(G)$, a node $v_i \in C$ is* ***pivotal*** *to C iff $C \setminus \{v_i\} \notin \mathcal{C}(G)$.*

To complete the definition of the connectivity game, we need to specify a so-called *characteristic function*—in coalitional games this function, $\nu : 2^V \to \mathbb{R}$, assigns to every coalition $C \subseteq V$ a numerical value representing its performance (by convention, it is assumed that $\nu(\emptyset) = 0$). In their games, Amer and Gimenez assign to every connected coalition a value of 1, and to disconnected coalitions a value of 0. Lindelauf *et al.* extend this definition by assuming that values of connected coalitions may depend on the network in a variety of ways, *i.e.*, they can be a function of adjacent edges or nodes, their weights, *etc.* More formally:

$$\nu_f(C) = \begin{cases} f(C,G) & \text{if } C \in \mathcal{C}(G) \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

The exact definition of f depends on the availability of information and analytical needs.[2] For instance, to analyse the Jemaah Islamiyah network responsible for the 2002 Bali attack in Indonesia, Lindelauf *et al.* assume:

$$f(C) = |E(C)| / \sum_{(v_i,v_j)\in E(C)} \omega_{ij}, \tag{2}$$

where $E(C) = \{(v_i, v_j) \in E \mid v_i, v_j \in C\}$ denotes the set of edges between players in C; that is f equals the number of edges in the connected coalition divided by their weight).

Since the goal of a centrality metric is to create a ranking of nodes, the question now is how to evaluate the importance of an individual node given all the roles that such a node plays in the entire connectivity game. To this end, Lindelauf *et al.* proposed to apply the Shapley value—one of the fundamental concepts in cooperative game theory. Specifically, Shapley [1953] proposed to evaluate the role played by individual players in a coalitional game by comparing their marginal contributions to every possible coalition. In order to formalize this concept in the terrorist network context, let $\pi \in \Pi(V)$ denote a permutation of nodes in V, and let C_i^π denote the coalition made of all predecessors of node v_i in π. More formally, if we denote by $\pi(j)$ the location of v_j in π, then: $C_i^\pi = \{v_j \in \pi : \pi(j) < \pi(i)\}$. The Shapley value of v_i, denoted $SV_i(\nu_f)$, is then defined as the average marginal contribution of v_i to coalition C_i^π over all $\pi \in \Pi$:

$$SV_i(\nu_f) = \frac{1}{|V|!} \sum_{\pi\in\Pi} mc_i(C_i^\pi), \tag{3}$$

where $mc_i(C) = \nu_f(C \cup \{v_i\}) - \nu_f(C)$ is the marginal contribution of v_i to $C \subseteq V$. The higher an average contribution of a player to the game is, the higher the Shapley value becomes. This means that, if the coalitional game defined over a network meets certain desired properties, the Shapley value of this game can be used as a centrality metric that rates individual nodes with respect to these properties. In our context of the connectivity game, individual terrorists are rated with respect to the role they play in connecting various, possibly multiple, parts of the network.

Lindelauf *et al.* argue that the centrality ranking based on the Shapley value of the connectivity game is more effective than degree, closeness and betweenness centralities in exposing the key players in the Bali attack. For example, Azahari bin Husin—the network bomb expert who was considered the "brain" behind the entire operation—is ranked rather low by standard centralities but according to Lindelauf *et al.* metric is among top five Bali terrorists. Similarly, Feri (Isa)—again ranked low by standard centralities—was, in fact, the suicide bomber. Lindelauf *et al.* ranked him third.

The formula in (3) can also be stated in the equivalent form:

$$SV_i(\nu_f) = \sum_{C\in 2^{V\setminus\{v_i\}}} \xi_C mc_i(C), \tag{4}$$

where $\xi_C = \frac{|C|!(|V|-|C|-1)!}{|V|!}$. Although it is clearly not possible to escape $O(2^{|V|})$ complexity for a general case of a coalitional game, *i.e.*, $2^V \to \mathbb{R}$, a number of authors showed that for some games defined on networks, it is possible to take advantage of the network structure and compute. the Shapley value in polynomial time [Deng and Papadimitriou, 1994; Aadithya *et al.*, 2010; Szczepański *et al.*, 2012]. In the next section we will show that this is not the case for connectivity games defined by Amer and Gimenez.

[2] We will write V, E, $\mathcal{C}$, $f(C)$ instead of $V(G)$, $E(G)$, $\mathcal{C}(G)$, $f(C,G)$, etc., wherever G is clear from the context.

3 Computational Analysis & Algorithms

In this section, we first discuss the complexity of computing the centrality metrics of Lindelauf *et al.*. We then present our exact and approximate algorithms.

Complexity

First we show that, even for the simplest definition of the connectivity game, where $\forall_{C \in \mathcal{C}(G)} f(C, G) = 1$, computing the Shapley value in an efficient way is impossible. The main problem of interest is as follows:

Definition 2 #CG-SHAPLEY*: Given a connectivity game on graph G, where $\forall_{C \in \mathcal{C}(G)} f(C, G) = 1$, we are asked to compute the Shapley value for each node in this graph.*

In the first step, let us introduce the following problem:

Definition 3 #CONNECTED-SPANNING-SUB (#CSS)*: Given a graph G, we are asked to compute the number of connected spanning subgraphs in G.*

This problem is #P-Complete even for bipartite and planar graphs [Welsh, 1997]. We will use this hardness result to prove #P-Completeness of the following problem:

Definition 4 #CONNECTED-INDUCED-SUB (#CIS)*: Given a graph G, we are asked to compute the number of connected induced subgraphs in G.*

Theorem 1 #CONNECTED-INDUCED-SUB *is #P-Complete.*

Proof of Theorem 1[3] We note first that it is possible to check in polynomial time if a given subgraph is induced and connected. Thus, since a *witness* can be verified in polynomial time, this problem is in #P. Now, we will reduce a #CSS instance to #CIS. To this end, given a graph $G = (V, E)$, we will construct a transformed graph G' and show that determining the number of connected induced subgraphs in G' allows us to easily compute the number of connected spanning subgraphs in G.

Our transformed graph G' is constructed by adding to each edge in G an extra node. Then, to each node from original graph G we attach a clique K_A with A nodes. This reduction is shown in Figure 1. More formally we define the following graph G':

$$\begin{aligned} V(G') =& V(G) \cup \{v_i : v \in V(G) \wedge i \in \{1, \ldots, A\}\} \cup \\ & \{vu : (v, u) \in E(G)\} \\ E(G') =& \{(v, uv) : (u, v) \in E(G)\} \cup \\ & \{(v, v_i) : v \in V(G) \wedge i \in \{1, \ldots, A\}\} \cup \\ & \{(v_i, v_j) : v \in v(G) \wedge i, j \in \{1, \ldots, A\}\} \wedge i < j\} \end{aligned}$$

Now, we arbitrarily choose some connected induced subgraph F of G'. Either this subgraph intersects with the original set of vertices $V(G)$, or it does not. In the latter case, subgraph F is contained within the single copy of K_A and there are exactly $|V(G)|(2^A - 1)$ such subgraphs. In the former case, we can define some *pseudograph* F', which consists of

[3]We would like to thank Colin McQuillan from The University of Liverpool who develop the sketch of this proof.

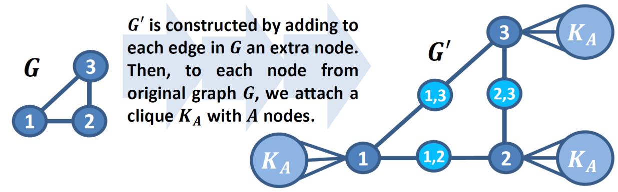

Figure 1: The reduction in the proof of Theorem 1.

the following sets of vertices and edges:[4] $V(F') = V(F) \cap V(G)$ and $E(F') = \{(u, v) : uv \in V(F) \wedge (u, v) \in E(G)\}$. Note that since F is connected, F' also has to be connected. There are exactly $2^{|V(F')|A}$ choices of F that can give us a particular pseudograph F'. Now, we can compute:

$$M = |V(G)|(2^A - 1) + \sum_{F'} 2^{|V(F')|A} \tag{5}$$

which denotes the number of induced connected subgraphs in G'. The crucial observation here is that, if $V(F') = V(G)$, then F' is a connected spanning subgraph of G. This holds because F is an induced connected subgraph. Now, let N denote the number of spanning connected subgraphs in G. Then, we can rewrite (5) as:

$$M = |V(G)|(2^A - 1) + N(2^{|V(G)|A}) + \sum_{F' : V(F') \neq V(G)} 2^{|V(F')|A} \tag{6}$$

In order to compute N we would like to bound the expression $X = M - N(2^{|V(G)|A})$. This expression is the number of induced connected subgraphs in G' that do not correspond to any connected spanning subgraph in G. We have:

$$0 \leq X \leq |V(G)|(2^A - 1) + 2^{|V(G)| + |E(G)|} 2^{(|V(G)|-1)A}$$

where the number $2^{|V(G)| + |E(G)|}$ is an upper bound on the number of all subgraphs in G. Now, we can use these bounds to transform equation (6) and to get the approximation of N:

$$\begin{aligned} N =& \frac{M}{2^{|V(G)|A}} - \frac{X}{2^{|V(G)|A}} \\ \simeq& \frac{M}{2^{|V(G)|A}} - \frac{|V(G)| + 2^{|V(G)| + |E(G)|}}{2^A} \end{aligned}$$

In order to deal with the expression $\frac{|V(G)| + 2^{|V(G)| + |E(G)|}}{2^A}$ we take $A > \log(|V(G)| + 2^{|V(G)| + |E(G)|})$ so that this fraction becomes smaller than 1. We note that A is bounded by a polynomial in the order of the input G. Then, the number of spanning connected subgraphs of G is the least integer N such that $N \geq \frac{M}{2^{A|V(G)|}}$. This is easy to compute given the number of connected induced subgraphs F of G'. □

From Theorem 1, it trivially follows that the next problem is also #P-Complete:

Definition 5 #CONNECTED-INDUCED-SUB-k (#CIS-k)*: Given a graph G, we are asked to compute the number of connected induced subgraphs of size k in G.*

[4]We note that this tuple is not necessarily a properly defined graph, since it can contain some edge (u, v) and does not contain node v.

Clearly, if we can find in polynomial time an answer for the #CIS-k problem, we could efficiently compute #CIS. Now, the #P-Hardness of #CG-SHAPLEY will be shown by the reduction from #CIS-k:

Theorem 2 #CG-SHAPLEY *is #P-Hard.*

Proof of Theorem 2: We construct a proof by reduction. In particular, we demonstrate that if there exists an algorithm for solving #CG-SHAPLEY in polynomial time, then it is possible to solve #CIS-k in polynomial time. This contradicts the fact that #CIS-k is #P-complete. Now, we will reduce #CIS-k to #CG-SHAPLEY.

Let $G = (V, E)$ be an arbitrary graph, where $|V| = n$ and $|E| = m$. We extend the set of nodes of G by a single node v, while the set of edges remains unchanged. In other words, we obtain a new graph G_0 by adding a single node not connected to any node from G. Now, from the definition of connectivity games, we note that the marginal contribution of the new node v to any coalition $C \subseteq G$ is either 0 or -1. More specifically, it is -1 if C is connected, and 0 otherwise. Based on this, the Shapley value of node v can be computed as follows, where c_s^G is the number of connected induced subgraphs in G that contain exactly s nodes:

$$SV_{v,G_0} = -\sum_{s=0}^{n} \frac{(s)!(n-s)!}{(n+1)!} c_s^G$$

Now, let us consider a new graph G_i constructed by adding to G the set of i nodes in addition to the node v, while keeping the set of edges just as in G. Analogously to G_0, the Shapley value of v in G_i is:

$$SV_{v,G_i} = -\sum_{s=0}^{n} \frac{(s)!(n+i-s)!}{(n+1+i)!} c_s^G \quad (7)$$

This equation holds because each coalition containing more than n nodes is disconnected, and so the contribution of v to every such coalition is 0. Now, we can build a system of linear equations using each equation (7) from graph G_i, where $i \in \{0, \ldots, n\}$. More precisely, we need to solve the following equation:

$$\begin{bmatrix} 0!n! & 1!(n-1)! & \cdots & n!0! \\ 0!(n+1)! & 1!n! & \cdots & n!1! \\ \vdots & \vdots & \ddots & \vdots \\ 0!(2n)! & 1!(2n-1)! & \cdots & n!n! \end{bmatrix} \begin{bmatrix} c_0^G \\ c_1^G \\ \vdots \\ c_n^G \end{bmatrix} = \begin{bmatrix} (n+1)!SV_{v,G_0} \\ (n+2)!SV_{v,G_1} \\ \vdots \\ (2n+1)!SV_{v,G_n} \end{bmatrix}$$

that can also be written as $Ax = b$.

This equation has a unique solution if and only if the determinant of the matrix A is non-zero. We can use Theorem 1.1 from [Bacher, 2002] to prove that it is non-zero. Thus, if we can compute in polynomial time the Shapley value for connectivity games, we would be able to solve this equation and determine all c_i^G values. Specifically, we can use Gaussian elimination, which works in $O(n^3)$ time complexity.

We note that the largest possible number in our matrices is $n!n!$. According to the analysis in (Proposition 2)[Aziz *et al.*, 2009] it is possible to store such a number in $km^2(\log m)^2$ bits. It is shown in (Theorem 4.10) [Korte and Vygen, 2005] that in Gaussian elimination each number occurring during the algorithm process can be stored in the number of bits quadric of the input size. □

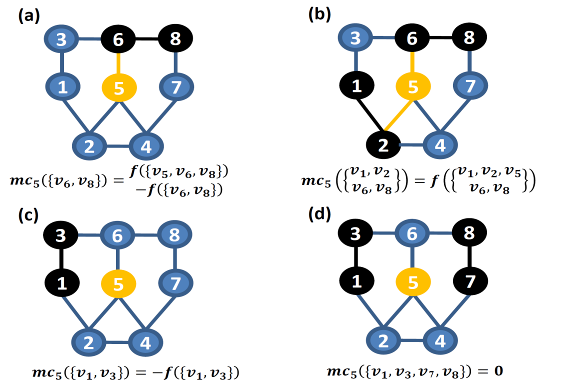

Figure 2: Four ways in which v_5 can contribute to a coalition.

Analysis of Marginal Contributions

In this section, we analyse how node $v_i \in V$ can marginally contribute to coalition $C \subseteq V \setminus \{v_i\}$. Four general cases, depicted in Figure 2, can be distinguished:

(a) Node v_i can join a *connected* coalition $C \in \mathcal{C}$ and the resulting coalition is also *connected*, *i.e.*, $C \cup \{v_i\} \in \mathcal{C}$. Here, the marginal contribution is equal to the difference in the value of $C \in \mathcal{C}$ caused by the addition of v_i:
$mc_i(C) = \nu_f(C \cup \{v_i\}) - \nu_f(C) = f(C \cup \{v_i\}) - f(C)$

(b) Node v_i can join a *disconnected* coalition $C \in \tilde{\mathcal{C}}$ and the resulting coalition becomes *connected*, *i.e.*, $C \cup \{v_i\} \in \mathcal{C}$. Here, v_i's contribution is the whole value of $C \cup \{v_i\}$:
$mc_i(C) = \nu_f(C \cup \{v_i\}) - \nu_f(C) = f(C \cup \{v_i\})$

(c) Node v_i can join a *connected* coalition $C \in \tilde{\mathcal{C}}$ and the resulting coalition becomes *disconnected*, *i.e.*, $C \cup \{v_i\} \in \tilde{\mathcal{C}}$. This means that v_i brings down the value of C to 0:
$mc_i(C) = \nu_f(C \cup \{v_i\}) - \nu_f(C) = -f(C)$

(d) Node v_i can join a *disconnected* coalition $C \in \tilde{\mathcal{C}}$ and the resulting coalition remains *disconnected*, *i.e.*,$C \cup \{v_i\} \in \tilde{\mathcal{C}}$:
$mc_i(C) = \nu_f(C \cup \{v_i\}) - \nu_f(C_1)) = 0$

The key conclusion to be drawn from the above analysis is that both connected and disconnected coalitions play a role when computing the Shapley value. This is because a node can contribute not only to a connected coalition, but also to a disconnected one (by making it connected). However, in the further sections, we will show that it is possible to develop an exact algorithm that only cycles through connected coalitions.

GeneralSV Algorithm

Based on the observation that both connected and disconnected coalitions play a role when computing the SV, we construct the general-purpose Shapley value algorithm (**GeneralSV**) for computing the SV in connectivity games defined by Amer and Gimenez. It is presented in Algorithm 1, where for each $C \in 2^V$, we denote by $\mathcal{N}(C)$ the set of all neighbours of C. Essentially, for each coalition $C \in 2^V$ this algorithm considers cases (a), (b) and (c) as outlined in Figure 2 in our paper.[5] In all of these three cases, a node

[5] As for (d), it can be disregarded since the marginal contribution in this case equals 0.

Algorithm 1: GeneralSV Algorithm for the SV

```
Input: Graph G = (V, E) and characteristic function ν_f
Output: Shapley Value, SV_i(ν_f), for each node v_i ∈ V
foreach v_i ∈ V do
    SV_i(ν_f) ← 0;
foreach C ∈ 2^V do
    CheckConnectedness(C);
    if C ∈ 𝒞 then
        foreach v_i ∈ 𝒩(C) do
            SV_i(ν_f) ←
            SV_i(ν_f) + ξ_C(ν_f(C ∪ {v_i}) − ν_f(C))
        foreach v_i ∉ 𝒩(C) do
            SV_i(ν_f) ← SV_i(ν_f) − ξ_C ν_f(C)
    else
        foreach v_i ∈ 𝒩(C) do
            CheckConnectedness(C ∪ {v_i})
            if C ∪ {v_i} ∈ 𝒞 then
                SV_i(ν_f) ← SV_i(ν_f) + ξ_C ν_f(C ∪ {v_i})
```

v_i can make a non-zero contribution by joining a coalition. We note that function $CheckConnectedness(C)$ runs in $O(|C| + |E(C)|)$.

Unfortunately, the use of this algorithm in practice is limited by the number of nodes in the network. It runs in $O((|V| + |E|)2^{|V|})$ and already for $|V| = 50$, the Algorithm 1A has to cycle through more than 10^{15} coalitions. In our paper, we developed an algorithm that for many networks is able to compute the SV substantially faster.

FasterSVCG Algorithm

Many real-world terrorist networks are sparse, *i.e.*, $|\mathcal{C}| \ll |\tilde{\mathcal{C}}|$ [Krebs, 2002]; thus, if the Shapley value could be computed only by considering coalitions in $\mathcal{C}$, it would be possible to analyse much larger terrorist networks. To this end, for each $v_i \in V$, let us define the following disjoint sets of coalitions:

$$\mathcal{C}_i^{\#} = \{C \subseteq V \setminus \{v_i\} : \ C \in \mathcal{C} \ \wedge \ C \cup \{v_i\} \in \mathcal{C}\}$$
$$\mathcal{C}_i^{+} = \{C \subseteq V \setminus \{v_i\} : \ C \in \tilde{\mathcal{C}} \ \wedge \ C \cup \{v_i\} \in \mathcal{C}\}$$
$$\mathcal{C}_i^{-} = \{C \subseteq V \setminus \{v_i\} : \ C \in \mathcal{C} \ \wedge \ C \cup \{v_i\} \in \tilde{\mathcal{C}}\}$$

which correspond to cases (a), (b) and (c) from the previous section. Based on this, the Shapley value can be computed as:

$$SV_i(\nu_f) = \sum_{C \in 2^{V \setminus \{v_i\}}} \xi_C mc_i(C) = \sum_{C \in \{\mathcal{C}_i^{+} \cup \mathcal{C}_i^{\#} \cup \mathcal{C}_i^{-}\}} \xi_C mc_i(C) \quad (8)$$

where case (d), when $mc_i(C) = 0$, is simply omitted. The key idea behind our exact algorithm to compute the Shapley value in connectivity game (abbreviated FasterSVCG) is to represent the sets $\mathcal{C}_i^{+}$ and $\mathcal{C}_i^{-}$ differently, such that $\tilde{\mathcal{C}}$ does not appear in the new representation.[6] In particular, we represent $\mathcal{C}_i^{+}$ and $\mathcal{C}_i^{-}$ as follows, where $\mathcal{P}(C)$ is the set of agents that are pivotal to C, and $\mathcal{N}(C)$ is the set of neighbours of C:

$$\mathcal{C}_i^{+} = \{C \subseteq V \setminus \{v_i\} : \ C \cup \{v_i\} \in \mathcal{C} \ \wedge \ v_i \in \mathcal{P}(C \cup \{v_i\})\}$$
$$\mathcal{C}_i^{-} = \{C \subseteq V \setminus \{v_i\} : \ C \in \mathcal{C} \ \wedge \ v_i \notin \mathcal{N}(C)\}$$

Now since $\tilde{\mathcal{C}}$ no longer appears in the definitions of $\mathcal{C}_i^{\#}$, $\mathcal{C}_i^{+}$ and $\mathcal{C}_i^{-}$, it is possible to compute the Shapley value as in equation (8) *without enumerating any of the coalitions in* $\tilde{\mathcal{C}}$. Based on this, our algorithm enumerates every connected coalition, $C \in \mathcal{C}$, and determines for each agent $v_i \in C$ whether $C \setminus \{v_i\} \in \mathcal{C}_i^{\#}$ or $C \setminus \{v_i\} \in \mathcal{C}_i^{+}$ and for $v_i \notin C$ if $C \in \mathcal{C}_i^{-}$.[7] The enumeration is carried out using Moerkotte and Neumann [2006]'s method—the fastest such enumeration method in the literature. Its basic idea is that, for each connected coalition $C \in \mathcal{C}$, it expands C by adding to it certain subsets of its neighbours. These subsets are chosen so as to ensure that no connected coalition is enumerated more than once (see Moerkotte an Neumann [2006] for more details).

Next, we explain our algorithm. To enhance clarity, for every connected coalition $C \in \mathcal{C}$, we will define three disjoint sets of agents: $V_C^{\#} = \{v_i \in C : C \setminus \{v_i\} \in \mathcal{C}_i^{\#}\}$, $V_C^{+} = \{v_i \in C : C \setminus \{v_i\} \in \mathcal{C}_i^{+}\}$, and $V_C^{-} = \{v_i \in V \setminus C : C \in \mathcal{C}_i^{-}\}$. If we compute the above sets for every $C \in \mathcal{C}$, then we can compute the Shapley value. Let us take a closer look at the difficulty of computing those sets for a given C.

- Computing V_C^{-} can be done in $O(|V|)$ time. This is because the agents in V_C^{-} are basically all those that are not members, nor neighbours, of C.
- Now, to compute V_C^{+}, we need to find the pivotal agents in C. This can be computed using a method *"findPivotals"* that runs in $O(|V| + |E|)$ [Alsuwaiyel, 1999].
- Having computed V_C^{+}, it becomes easy to compute $V_C^{\#}$. This is because $V_C^{\#} = C \setminus V_C^{+}$.

From the above analysis, it is clear that the main difficulty lies in *findPivotals*. Therefore, whenever possible, we would like to compute V_C^{+} using some other, easier, technique. In particular, when we expand a connected coalition, C, into another connected coalition $C' = C \cup S$, we try to *update* the set of pivotal agents, rather than compute it from scratch with *findPivotals*. Here, we distinguish between three conditions:

- Condition 1: The cycles in C' are exactly like those in C. In this case, the set $V_{C'}^{+}$ consists of elements of V_C^{+}, expanded by the nodes in C that are connected to S.
- Condition 2: C' contains a cycle that is not in C. Here, we need to call *findPivotals*.
- Condition 3: $|C| = 2$. In this case, since we assumed that a singleton is a connected coalition, none of the two agents in C is pivotal.

The pseudocode of FasterSVCG is presented in Algorithm 1. It is easy to see that it runs in $O((|V| + |E|)|\mathcal{C}|)$.

In the next section we propose an approximation algorithm that is able to provide a ranking for larger networks.

[6]As for $\mathcal{C}_i^{\#}$, it does not depend on $\tilde{\mathcal{C}}$, and so there is no need to represent it differently.

[7]Note, that we do not consider the impact of agent $v_i \notin C$ that do not disconnect C because the contribution of this agent will be calculated for connected coalition $C \cup \{v_i\}$ as $C \in \mathcal{C}_i^{\#}$.

Algorithm 2: FasterSVCG for $SV_i(\nu_f), v_i \in V$

Input: Graph $G=(V,E)$ and characteristic function ν_f
Output: Shapley value $SV_i(\nu_f)$ of each node $v_i \in V$

$X \leftarrow V$; `// initialize X, which is only used for Moerkotte & Neumann's enumeration`
foreach $v_i \in V$ **do** $SV_i(\nu_f) \leftarrow 0$;
for $i \leftarrow |V|$ **to** 1 **do**
  $computeSV(\{v_i\}, \mathcal{N}(v_i), X, X \setminus (\mathcal{N}(v_i) \cup \{v_i\}), \emptyset)$;
  $X \leftarrow X \setminus \{v_i\}$
`// ----------- Next, we define computeSV -----------`
$computeSV(C, NC, X, V_C^-, V_C^+)$ **begin**
  $X' \leftarrow X \cup NC$; `// where NC consists of the neighbours of C`
  foreach $S \subseteq (NC \setminus X) \wedge S \neq \emptyset$ **do**
    $C' \leftarrow C$; `// a new coalition that will be constructed from the old coalition C`
    $NC' \leftarrow NC \setminus S$; `// the neighbours of C'`
    $isCycle \leftarrow false$; `// to indicate whether a new cycle has appeared while constructing C'`
    foreach $v \in S$ **do**
      $TEMP \leftarrow \emptyset$; `// a temporary set used to compute neighbours of: C' ∪ {v}`
      $TEMP2 \leftarrow \emptyset$; `// a temporary set used to compute the pivotal agents in: C' ∪ {v}`
      foreach $u \in \mathcal{N}(S)$ **do**
        if $u \notin (C \cup NC)$ **then**
          $TEMP \leftarrow TEMP \cup \{u\}$;
        else if $(isCycle = false) \wedge (u \in C')$ **then** `// condition 1`
          $TEMP2 \leftarrow TEMP2 \cup \{u\}$;
    if $|TEMP2| > 1$ **then** `// condition 2`
      $isCycle \leftarrow true$;
    $C' \leftarrow C' \cup \{v\}$; $NC' \leftarrow NC' \cup TEMP$;
    $V_{C'}^- \leftarrow V_{C' \setminus \{v\}}^- \setminus TEMP$;
    if $isCycle = false$ **then** `// condition 1`
      $V_{C'}^+ \leftarrow V_{C' \setminus \{v\}}^+ \cup TEMP2$;
  if $|C'| = 2$ **then** `// condition 3`
    $V_{C'}^+ \leftarrow \emptyset$;
  else if $isCycle = true$ **then** `// condition 2`
    $V_{C'}^+ \leftarrow FindPivotals(C')$;
  foreach $v_i \in C'$ **do** `// update Shapley value`
    if $v_i \in V_{C'}^+$ **then**
      $SV_i(\nu_f) \leftarrow SV_i(\nu_f) + \xi_{C' \setminus \{v_i\}} f(C')$
    else `// deal with the set V#_{C'}`
      $SV_i(\nu_f) \leftarrow SV_i(\nu_f) + \xi_{C' \setminus \{v_i\}} (f(C') - f(C' \setminus \{v_i\}))$
  foreach $v_i \in V_{C'}^-$ **do** `// update Shapley value`
    $SV_i(\nu_f) \leftarrow SV_i(\nu_f) - \xi_{C'} f(C')$
  $computeSV(C', NC', X', V_{C'}^-, V_{C'}^+)$;

Algorithm 3: ApproximateSVCG for $SV_i(\nu_f), v_i \in V$

Input: Graph $G=(V,E)$ and characteristic function ν_f
Output: Shapley value $SV_i(\nu_f)$ of each node $v_i \in V$

foreach $v_i \in V$ **do** $SV_i(\nu_f) \leftarrow 0$;
for $it = 1$ **to** $maxIter$ **do**
  $k \leftarrow$ *random number from* $\{0, \ldots, |V|\}$;
  $C \leftarrow$ *random coalition of size* k;
  if *!CheckConnectedness(C)* **then** *continue*;
  $P \leftarrow FindPivotals(C)$;
  foreach $v_i \in C \setminus P$ **do** `// case (a)`
    $SV_i(\nu_f) \leftarrow SV_i(\nu_f) + \frac{|V|+1}{|C|} \cdot (f(C) - f(C \setminus \{v_i\}))$
  foreach $v_i \in P$ **do** `// case (b)`
    $SV_i(\nu_f) \leftarrow SV_i(\nu_f) + \frac{|V|+1}{|C|} \cdot f(C)$
  foreach $v_i \in (V \setminus C) \setminus \mathcal{N}(C)$ **do** `// case (c)`
    $SV_i(\nu_f) \leftarrow SV_i(\nu_f) - \frac{|V|+1}{|V|-|C|} \cdot f(C)$
foreach $v_i \in V$ **do** $SV_i(\nu_f) \leftarrow SV_i(\nu_f)/maxIter$;

ApproximateSVCG Algorithm

ApproximateSVCG is our dedicated application of Monte Carlo sampling to connectivity games. Unlike the existing algorithm to approximate the Shapley value for characteristic function games [Castro *et al.*, 2009], in our algorithm we do not sample permutations, but coalitions. Since any marginal contribution of an agent, v_i, links two coalitions—one with this agent, i.e., $C \cup \{v_i\}$, and one without him, i.e., C—sampling of coalition C can be viewed as *sampling of v_i's marginal contribution*. Generally speaking, in our algorithm, we will randomly select a number of marginal contributions of v_i and approximate the SV_i using the resulting average. Due to the fact that, in formula (4) for the Shapley value, marginal contributions are calculated with different weights, to obtain an unbiased estimator we have to sample marginal contributions with appropriate probabilities. To this end, we propose the following general process. In **Step 1**, we uniformly select $k \in \{0, \ldots, |V|\}$. In **Step 2**, we choose a random coalition C of size k,[8] and in **Step 3**, for every agent, compute the marginal contribution of this agent obtained by leaving/entering C. To better understand our motivation, let us transform the formula for the Shapley value as follows:

$$SV_i(\nu_f) = \frac{1}{|V|} \sum_{0 \le k < |V|} \cdot \frac{1}{\binom{|V|-1}{k}} \sum_{C \subseteq V \setminus \{i\}, |C|=k} mc_i(C)$$

From this formula it is clear that, to obtain an unbiased estimator, the sampling method should satisfy two conditions:

(i) the probability that a randomly chosen marginal contribution is obtained from entering a coalition of size k is equal for every k: $\frac{1}{n+1} \cdot \frac{n-k}{n} + \frac{1}{n+1} \cdot \frac{k+1}{n} = \frac{1}{n}$ $(n = |V|)$;[9]

[8] To this end, we take the first k elements of a random permutation (generated uniformly with Knuth shuffle). This method is unbiased, as every coalition appears in the same $k!(n-k)!$ number of permutations.

[9] Note that a given marginal contribution appears twice in our

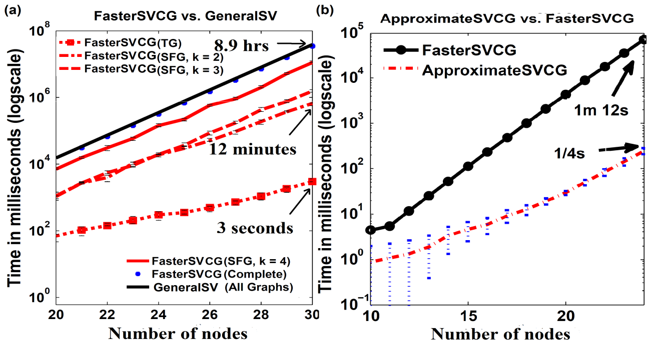

Figure 3: Time performance of both algorithms.

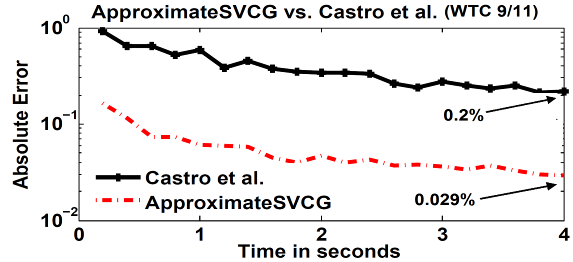

Figure 4: Error performance of ApproximateSVCG.

(ii) marginal contributions to all coalitions of size k are chosen with the same probability.

This technique allows us to compute the marginal contributions of all agents for a randomly selected coalition, which in the connectivity games (and, potentially, in many more classes of games) can be performed much faster than estimating the Shapley value for each player separately [Mann and Shapley, 1962] or by sampling of a random permutation [Castro *et al.*, 2009], where we have to calculate the marginal contributions for a sequence of coalitions growing in size.

To this end, in the ApproximateSVCG algorithm we merge our Monte Carlo technique with the analysis of marginal contribution presented before and used in FasterSVCG. The pseudo code is presented in Algorithm 2. Line 3 corresponds to **Step 1**, where we sample the size of a coalition. Now, we modify **Step 2** in order to select only *connected coalitions* (lines 4 and 5).[10] We also modify **Step 3**, when we consider cases (a), (b), and (c) from Figure 2 (lines 7-12). The modification of **Step 3** has to be done due to the following reason: since we no longer consider disconnected coalitions, any non-zero marginal contribution made to a disconnected coalition have to be transferred to a corresponding connected coalition (lines 8, 10, and 12). Furthermore, this should be done in a way that preserves appropriate probabilities (thus, in lines 8, 10, and 12 we multiply marginal contributions by adequate weights). Finally, we divide the sum of the contributions by the number of iterations (lines 13). For every sample, algorithm runs in time $O(|V| + |E|)$.

4 Performance Evaluation

We generate random graphs focusing on two topologies commonly found in social networks, and in terrorist organisations in particular [Krebs, 2002; Sageman, 2004]: (i) *scale-free graphs* (SFGs), where the network is generated according to a power law; and (ii) *random trees* (TG), which model hierarchical organisations. To construct SFGs, we use the preferential attachment generation model [Albert and Barabási, 2002], with parameters $k = 1, 2, 3$ (*cfr.* [Voice *et al.*, 2012]).[11] To construct TGs, every new node is attached to a randomly picked incumbent. Finally, we include in our analysis (iii) *complete graphs*, where all coalitions are connected. Although complete graphs are unlikely to arise in a terrorist network context, they constitute a suitable benchmark for our simulations. For all the games on random graphs we assume that f is defined as in (2).

FasterSVCG: In Figure 3(a), we compare the performance of FasterSVCG to the general-purpose Shapley value algorithm (abbreviated GeneralSV).[12] Clearly, this latter algorithm runtime is the same for any type of graph. Unlike GeneralSV, FasterSVCG takes advantage of the sparsity of the network and, consequently, significantly outperforms the benchmark. For instance, for SFG($k = 2$) and $|V| = 30$, FasterSVCG needs only 0.39% of GeneralSV runtime! Naturally, the best performance is obtained for TGs, where the game over a network of 30 nodes can be computed in about 3 seconds.

ApproximateSVCG: In Figure 3(b), we evaluate the time performance of ApproximateSVCG. Importantly, for our purpose of identifying key terrorists, we are mostly interested in the approximation of the correct (Shapley value) ranking of top nodes, and less in the approximation of their actual Shapley values. To this end, we evaluate the time required by ApproximateSVCG to obtain the ranking of the top $\lceil\sqrt{n}\rceil$ nodes, with at most one error (i.e., one inversion compared to the exact ranking). We present an average time calculated over 500 iterations, each for a randomly selected SFG($k = 4$), with 75% confidence level. ApproximateSVCG returns top nodes much faster than FasterSVCG—for the graph of 24 nodes the former algorithm runs in 0.25 sec. and the exact one in 72 sec.

As far as other than $\lceil\sqrt{n}\rceil$ thresholds are concerned, the lower the threshold, the faster our Monte Carlo method becomes (and vice versa). Also ApproximateSVCG becomes faster, the higher the allowable error is (measured with the

process—the first term represents the probability that we select a coalition of size k without player v_i, while the second one—that we select the corresponding coalition of size $k+1$, with player v_i.

[10]It should be noted, that generating a random *connected coalition* uniformly will create a biased algorithm.

[11]In this model, while gradually constructing a graph, every new node v_i is linked to k incumbents such that the probability that v_i is linked to incumbent v_j is $\frac{degree(v_i)}{\sum_j degree(v_j)}$.

[12]All experiments are carried out on a 64 bit, Intell Zeon E5-2643 with 2 CPU (8 cores each) 3.3 Ghz, 128GB RAM. For $|V| \leq 25$, we repeated each experimental run 50, for $26 \leq |V| \leq 30$—30 times. In the experiments involving random graphs no parallel computations were performed. Krebs' 9/11 network was analyzed with a parallelised code on 16 cores (each individual core was assigned with computing a tree for one node).

Rank	Lindelauf *et al.* $|V| = 19, |E| = 32$	Our analysis $|V| = 36, |E| = 125$
1.	A. Aziz Al-Omari	Z. Moussaoui
2.	H. Alghamdi	A. Aziz Al-Omari
3.	W. Alsheri	M. Atta
4.	H. Hanjour	W. Alshehri
5.	M. Al-Shehhi	N. Alhazmi

Table 1: FasterSVCG allows us to consider the bigger network of WTC 9/11 attack than Lindelauf *et al.* which may deliver new insights into the leadership structure of this network

number of allowable inversions). Finally, we note that ApproximateSVCG is faster than our exact algorithm in generating an error-free rankings (on average, approximately 6, 26, and 174 times faster for 15, 20, and 25 agents, respectively).

Figure 4 presents the error convergence of the ApproximateSVCG and compares the results to the random permutation sampling studied by Castro *et al.* [2009]. Here, we focus on the maximum absolute error of the Shapley value, computed as a percentage of the value of the grand coalition. The results are calculated for Krebs' 9/11 WTC terrorist network with 36 nodes (as an average from over 30 iterations). ApproximateSVCG outperforms Castro's method—after 4 second, the error of ApproximateSVCG equals 0.029%, while for Castro *et al.* exceeds 0.2%. The fact that the absolute error ultimately converges to zero indicated that our sampling method is not biased.

New insight on Krebs' 9/11 WTC network: In the already classic work, Krebs [2002] constructed the 9/11 network from publicly available sources and computed standard centrality metrics to determine the key players in this network. Lindelauf *et al.* instead used their new centrality metric and, similarly to the case of the Jemaah Islamiyah's network, argued that the Shapley value-based approach delivers qualitatively better insights (as it captures the network structure in a more sophisticated manner). However, Lindelauf *et al.*'s analysis focuses only on the network of 19 hijackers with 32 relationships, whereas Krebs reported also a bigger network of 36 nodes and 125 edges (mentioned above) that included accomplices. With FasterSVCG, we were able to analyse this bigger network as well. The computations took 38 minutes during which FasterSVCG traversed 401.963.129 connected coalitions (0.59% of all).

Interestingly, the five top terrorists (out of 36) identified by FasterSVCG differ from the top five terrorists reported by Lindelauf *et al.* when only the network of 19 terrorist was taken into account—see Table 1. One of the differences is with respect to **M. Atta**, who was widely believed to be one of the ring-leaders of the conspiracy [Krebs, 2002] and was positioned the third place in our computations, but was classified in Lindelauf *et al.*'s work on the 6th place.

5 Related Work

A rapidly growing body of work is directed to the analysis of terrorist organisations using the methods of social network analysis. A very good introduction to this line of research can be found in Ressler [2006]. Also worthy of note is the work of Farley [2003], Carley [2003], and Husslage *et al.* [2012], who conduct quantitative analysis of the terrorist networks.

Since the work of Grofman and Owen [1982], a number of game-theoretic centrality measures have been developed either to enrich the existing well-known centralities or as completely new ones (*e.g.*, [Van den Brink *et al.*, 2007; Gómez *et al.*, 2003; Szczepański *et al.*, 2012]). In the terrorist network context, Lindelauf *et al.* [2009a; 2009b] employed the game-theoretic approach to analyze covert networks.

In the first computational analysis of game-theoretic centrality, Aadithya *et al.* [2010] present polynomial time algorithms for computing Shapley value-based centrality measures built upon four coalitional games that modelled possible influence of coalitions on their neighbours. In subsequent work, Szczepanski *et al.* [2012] presented an algorithm for the Shapley value-based betweenness centrality.

We also mention works on the Shaple value approximation algorithms ([Fatima *et al.*, 2007; Bachrach *et al.*, 2008a]).

The hardness result presented in this paper is consistent with other studies of the complexity of the Shapley value in various settings. For instance, computing the Shapley value was shown to be #P-Complete for weighted majority games [Deng and Papadimitriou, 1994] and in minimum spanning tree games [Nagamochi *et al.*, 1997]. Aziz *et al.* 2009 obtained negative results for a related problem of computing the Shapley-Shubik power index for the spanning connectivity games that are based on undirected, unweighted multigraphs. Also, Bachrach *et al.* [Bachrach *et al.*, 2008b] showed that the computation of the Banzhaf index for connectivity games, in which agents own vertices and control adjacent edges and aim to become connected to the certain set of primary edges, is #P-Complete. A comprehensive review of these issues, including some positive results for certain settings, can be found in [Chalkiadakis *et al.*, 2011].

6 Conclusions

In this paper, we proposed an algorithm to compute the Shapley value for the connectivity games of Amer and Gimenez defined over arbitrary graphs. Although our method is general, it has been created with an aim to study centrality metrics proposed by Lindelauf *et al.* to identify key individuals in terrorist networks. We proved that the problem is #P-Hard. Nevertheless, using our exact algorithm we were able to analyse in moderate time the 36 node version of the network responsible for the 9/11 WTC terrorist attacks. We also presented the approximate algorithm that allows for an efficient study of bigger networks and for further validation of the Lindelauf et al's metrics.

Acknowledgments

We would like to thank the anonymous IJCAI reviewers for valuable feedback. Tomasz Michalak and Michael Wooldridge were supported by the European Research Council under Advanced Grant 291528 ("RACE"). Talal Rahwan and Nicholas R. Jennings were supported by the ORCHID Project, funded by EPSRC (Engineering and Physical Research Council) under the grant EP/I011587/1.

References

[Aadithya *et al.*, 2010] K.V. Aadithya, B. Ravindran, T.P. Michalak, and N.R. Jennings. Efficient Computation of the Shapley Value for Centrality in Networks. In *WINE*, 2010.

[Albert and Barabási, 2002] R. Albert and A.L. Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74:47–97, 2002.

[Alsuwaiyel, 1999] H. Alsuwaiyel. *Algorithms: Design Techniques and Analysis*. Lecture Notes Series on Computing. World Scientific, 1999.

[Amer and Gimnez, 2004] R. Amer and J. M. Gimnez. A connectivity game for graphs. *Math. Methods of OR*, 60:453–470, 2004.

[Aziz *et al.*, 2009] H. Aziz, O. Lachish, M. Paterson, and R. Savani. Power Indices in Spanning Connectivity Games. In *AAIM*, pages 55–67, 2009.

[Bacher, 2002] R. Bacher. Determinants of matrices related to the pascal triangle. *JTNB*, 14(1):19–41, 2002.

[Bachrach *et al.*, 2008a] Y. Bachrach, E. Markakis, A. Procaccia, J. Rosenschein, and A. Saberi. Approximating power indices. In *AAMAS '08: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 943–950, 2008.

[Bachrach *et al.*, 2008b] Y. Bachrach, J.S. Rosenschein, and E. Porat. Power and stability in connectivity games. In *AAMAS*, 2008.

[Brandes and Thomas, 2005] U. Brandes and E. Thomas. *Network Analysis: Methodological Foundations*. LNCS, 2005.

[Carley *et al.*, 2003] K.M. Carley, J. Reminga, and N. Kamneva. Destabilizing terrorist networks. In *NAACSOS*, 2003.

[Castro *et al.*, 2009] J. Castro, D. Gomez, and J. Tejada. Polynomial calculation of the shapley value based on sampling. *C&OR*, 36(5), 2009.

[Chalkiadakis *et al.*, 2011] G. Chalkiadakis, E. Elkind, and M. Wooldridge. *Computational Aspects of Cooperative Game Theory*. Synthesis Lectures on Artificial Intelligence and Machine Learning. 2011.

[Deng and Papadimitriou, 1994] X. Deng and C. Papadimitriou. On the complexity of cooperative solution concepts. *MOR*, 19:257–266, 1994.

[Farley, 2003] J.D. Farley. Breaking al qaeda cells: a mathematical analysis of counterterrorism operations. *Studies in Conflict and Terrorism*, 31:126–137, 2003.

[Fatima *et al.*, 2007] S. Fatima, M. Wooldridge, and N. R. Jennings. A randomized method for the shapley value for the voting game. In *AAMAS*, pages 955–962, 2007.

[Friedkin, 1991] N. Friedkin. Theoretical foundations for centrality measures. *Am. J. of Sociology*, 96(6):1478–1504, 1991.

[Gómez *et al.*, 2003] D. Gómez, E. González, C. Manuel, G. Owen, M. Del Pozo, and J. Tejada. Centrality and power in social networks: A game theoretic approach. *Math. Soc. Sci.*, 46:27–54, 2003.

[Grofman and Owen, 1982] B. Grofman and G. Owen. A game theoretic approach to measuring degree of centrality in social networks. *Social Networks*, 4:213–224, 1982.

[Husslage *et al.*, 2012] B. Husslage, R. H. A. Lindelauf, and H. Hamers. Leaderless Covert Networks: A Quantitative Approach, 2012.

[I2, 2010] I2. Analyst's Notebook 8, Social Network Analysis Whitepaper. `www.i2group.com`, 2010.

[Korte and Vygen, 2005] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Algorithms and Combinatorics. Springer, 2005.

[Krebs, 2002] V.E. Krebs. Mapping networks of terrorist cells. *Connections*, 24:43–52, 2002.

[Lindelauf *et al.*, 2009a] R.H.A. Lindelauf, P. Borm, and H.J.M. Hamers. The influence of secrecy on the communication structure of covert networks. *Social Networks*, 31:126–137, 2009.

[Lindelauf *et al.*, 2009b] R.H.A. Lindelauf, P. Borm, and H.J.M. Hamers. *On Heterogeneous Covert Networks*. Springer, 2009.

[Lindelauf *et al.*, 2013] R.H.A. Lindelauf, H.J.M. Hamers, and B.G.M. Husslage. Cooperative game theoretic centrality analysis of terrorist networks: The cases of jemaah islamiyah and al qaeda. *European Journal of Operational Research*, 229(1):230 – 238, 2013.

[Mann and Shapley, 1962] I. Mann and L.S. Shapley. Values of large games. vi: Evaluating the electoral college exactly. Technical report, DTIC Document, 1962.

[Moerkotte and Neumann, 2006] G. Moerkotte and T. Neumann. Analysis of two existing and one new dynamic programming algorithm for generation of optimal bushy join trees without cross products. In *VLDB*, 2006.

[Nagamochi *et al.*, 1997] H. Nagamochi, D. Zeng, N. Kabutoya, and T. Ibaraki. Complexity of the minimum base game on matroids. *MOR*, 22(1):146–164, 1997.

[Ressler, 2006] S. Ressler. Social network analysis as an approach to combat terrorism: past, present and future research. *Homeland Security Affairs*, 2:1–10, 2006.

[Sageman, 2004] M. Sageman. *Understanding Terror Networks*. University of Pennsylvania Press, 2004.

[Shapley, 1953] L. S. Shapley. A value for n-person games. In *In Contributions to the Theory of Games*, pages 307–317. PUP, 1953.

[Szczepański *et al.*, 2012] P.L. Szczepański, T.P. Michalak, and T. Rahwan. A new approach to betweenness centrality based on the shapley value. In *AAMAS*, 2012.

[Van den Brink *et al.*, 2007] R. Van den Brink, P. Borm, R. Hendrickx, and G. Owen. Characterization of the β and the degree network power measure. *Theory and Decision*, 64:519–536, 2007.

[Voice *et al.*, 2012] T. Voice, S. D. Ramchurn, and N. R. Jennings. On coalition formation with sparse synergies. In *AAMAS*, 2012.

[Welsh, 1997] D.J.A. Welsh. *Approximate counting*, volume 241. 1997.