Power and Stability in Connectivity Games

Yoram Bachrach Jeffrey S. Rosenschein
School of Engineering and Computer Science
The Hebrew University of Jerusalem, Israel
{yori,jeff}@cs.huji.ac.il

Ely Porat
Department of Computer Science
Bar-Ilan University, Ramat-Gan, Israel
porately@cs.biu.ac.il

ABSTRACT

We consider computational aspects of a game theoretic approach to network reliability. Consider a network where failure of one node may disrupt communication between two other nodes. We model this network as a simple coalitional game, called the vertex Connectivity Game (CG). In this game, each agent owns a vertex, and controls all the edges going to and from that vertex. A coalition of agents wins if it fully connects a certain subset of vertices in the graph, called the *primary* vertices.

We show that power indices, which express an agent's ability to affect the outcome of the vertex connectivity game, can be used to identify significant possible points of failure in the communication network, and can thus be used to increase network reliability. We show that in general graphs, calculating the Banzhaf power index is #P-complete, but suggest a polynomial algorithm for calculating this index in trees. We also show a polynomial algorithm for computing the core of a CG, which allows a stable division of payments to coalition agents.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity;
I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*;
J.4 [**Computer Applications**]: Social and Behavioral Sciences—*Economics*

General Terms

Algorithms, Theory, Economics

Keywords

Computational complexity, Network Reliability, Power indices

1. INTRODUCTION

Making joint decisions and cooperating are key features of multiagent systems. Cooperative game theory treats many aspects of these issues, and can serve as a foundation for analyzing such systems. One specific topic which has been studied extensively in game theory, as well as in other fields such as political science and economics, is that of voting. Voting is used in a variety of multiagent systems, typically to enable a team of agents to make a group decision.

An interesting measure applied to voting scenarios is that of power indices; such indices measure the control a voter has over decisions of a larger group [5]. One popular power index is the Banzhaf power index [2]. While it has mostly been used for measuring power in weighted voting systems, it can easily be adapted for other domains as well.

In this paper, we consider the use of the Banzhaf index to find key points of failure in a communication network. We model a communication network composed of servers and network links between them as a vertex connectivity game. The network is modeled as a graph, where the servers are the vertices, and the network links are the edges. A certain subset of the servers (vertices) are *primary*—a failure to send information between any two of them would constitute a major system failure. Another subset of the servers are always available (backbone servers).

In the vertex *Connectivity Game* (CG) that we introduce, each agent controls a different vertex in the graph. A coalition of agents can use any of the vertices controlled by the coalition members or the backbone vertices, and may send information between them. The coalition wins if it connects all the primary vertices, so that it can send information between any two of them. The Banzhaf index of an agent in this game reflects its criticality in maintaining this connectivity. In communication networks, this index could thus enable an administrator to identify potential critical points of failure in the network.

We consider the computational complexity of calculating the Banzhaf power index in this domain. We show that in general graphs, this problem is #P-complete. Despite this negative result, we provide a polynomial algorithm for the restricted case where the graph is a tree. Many networks, including parts of the internet's backbone, are constructed as trees when the construction of a communication line is expensive, so this algorithm can analyze important real-world domains.

We also consider another game theoretic solution concept, the core, and show that it can be computed in polynomial time in CGs. The core of the CG indicates which payoff vectors are stable. When a coalition in the CG manages to connect all the primary vertices it wins, and gains a certain profit. This profit should be divided among the members of the coalition. Choosing a payoff vector in the core guaran-

Cite as: Power and Stability in Connectivity Games, Yoram Bachrach, Jeffrey S. Rosenschein and Ely Porat, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. 999-1006.

tees that no subcoalition would choose to split from the main coalition, and attempt to establish its own network. Thus, the core can be used to allocate the gains of a coalition in a CG domain in a stable way.

The paper proceeds as follows. In Section 2, we provide background information regarding coalitional games and the Banzhaf power index, and fully define a vertex connectivity game. In Section 3, we discuss the Banzhaf power index in vertex connectivity games, and present the hardness result for the general case. Section 4 discusses vertex connectivity games in trees, and provides a polynomial algorithm for this case. Section 5 shows that the core of CGs can be computed in polynomial time. Section 6 discusses important related work, and we conclude in Section 7.

2. PRELIMINARIES

A coalitional game is composed of a set of n agents, $I = (a_1, \ldots, a_n)$, and a function mapping any subset (coalition) of the agents to a real value $v : 2^I \to \mathbb{R}$. The function v is called the *coalitional function* (or sometimes the *characteristic function*) of the game. In a *simple* coalitional game, v only gets values of 0 or 1, so $v : 2^I \to \{0, 1\}$. A coalition $C \subseteq I$ *wins* if $v(C) = 1$, and *loses* if $v(C) = 0$. The set of all winning coalitions is denoted $W(v) = \{C \subseteq 2^I | v(C) = 1\}$. An agent a_i is *critical* in a winning coalition C if the agent's removal from that coalition would make the coalition lose: $v(C) = 1$ but $v(C \setminus \{i\}) = 0$. Thus, an agent can only be critical in a coalition that contains him.

We are interested in measuring the influence a given agent has on the result of the game. A common interpretation for this is the probability that this agent would significantly affect the outcome of the game.

One common power index is the Banzhaf power index [2]. It has been widely used for measuring political power in weighted voting systems, but its definition does not rely on the specific features of a weighted voting game. The index depends on the number of coalitions in which an agent is critical. The Banzhaf index of agent a_i is the proportion of all winning coalitions where a_i is critical, out of all winning coalitions that contain a_i.

Definition 1. *The Banzhaf index is given by $\beta(v) = (\beta_1(v), \ldots, \beta_n(v))$ where*

$$\beta_i(v) = \frac{1}{2^{n-1}} \sum_{S \subseteq N | i \in S} [v(S) - v(S \setminus \{i\})].$$

When the agents are independent in their choices, so every coalition has an equal probability of occurring, the Banzhaf index measures the probability of an agent being critical; for the winning coalitions that contain a_i, the index counts in how many of them it is critical (i.e., its removal from the winning coalition would make the coalition lose). Other power indices reflect other assumptions. For example, the Shapley-Shubik index assumes the model that agents are randomly added to a coalition, so every *ordering* of the agents is equally probable. The index then measures in how many of the orderings a_i is the first agent that makes the coalition win (i.e., the first agent that, when added to the previous agents, makes the coalition win).[1] In this paper, we consider server failures in a network. Since each server may fail independently from the others, the Banzhaf power index is more appropriate for this case.

In simple coalitional games, the coalitional function v only gets values of 0 or 1. In general coalitional games, this function may have any value in $\mathbb{R}^+$, and we say this function determines the total gains of a coalition. The coalitional function only defines the utility a coalition can achieve, but does not define how the agents distribute this utility among themselves. A payoff vector $(p_1, \ldots, p_n)$ is a division of the gains of the grand coalition I among the agents, where $p_i \in \mathbb{R}$, and $\sum_{i=1}^{n} p_i = v(I)$. We call p_i the payoff of agent a_i, and denote the payoff of a coalition C as $p(C) = \sum_{i \in \{j | a_j \in C\}} p_i$. By assumption, agents are rational and attempt to maximize their own share of the utility. Game theory offers several solution concepts, determining which payoff vectors are possible when agents act rationally.

A simple requirement for the payoff vector is *individual-rationality*: for all agents $a_i \in C$, we have $p_i \geq v(\{a_i\})$, otherwise that agent is better off on its own. Similarly, we say a coalition B *blocks* the payoff vector $(p_1, \ldots, p_n)$ if $p(B) < v(B)$, since the members of B would split off from the coalition and gain more utility without the rest of the agents. If a blocked payoff vector is chosen, the coalition is unstable. A solution concept that emphasizes such stability is the core [8].

Definition 2. *The Core of a coalitional game is the set of all payment vectors $(p_1, \ldots, p_n)$ that are not blocked by any coalition, so for any coalition C we have $p(C) \geq v(C)$.*

A value distribution in the core makes sure that no subset of the agents would split off, so the coalition is stable. In general the core can be empty, so every possible value division is blocked by some coalition. In this work, we give results regarding computing the core in vertex Connectivity Games. Another solution concept, which contains only a single possible payment vector, is the Shapley value, whose application to simple coalitional games is the Shapley-Shubik power index, discussed above. However, the core focuses on stability, and the Shapley value focuses on certain fairness conditions.

Our hardness result for calculating the Banzhaf power index in CGs considers the class #P. #P is the set of integer-valued functions that express the number of accepting computations of a nondeterministic Turing machine of polynomial time complexity. Let Σ be the finite input and output alphabet for Turing machines.

Definition 3. *#P is the class consisting of the functions $f : \Sigma^* \to \mathbb{N}$ such that there exists a non-deterministic polynomial time Turing machine M that for all inputs $x \in \Sigma^*$, $f(x)$ is the number of accepting paths of M.*

The complexity classes #P and #P-complete were introduced by Valiant [20]. These classes express the hardness of problems that "count the number of solutions".[2]

[1] Although the Banzhaf and Shapley-Shubik indices are similar, the Banzhaf index considers all possible *subsets* of agents, whereas the Shapley-Shubik index is defined over all *orderings* (or permutations) of the agents, so the two indices can be computationally quite different.

[2] Informally, NP and NP-hardness deal with checking if at least one solution to a combinatorial problem exists, while #P and #P-hardness deal with calculating the *number* of solutions to a combinatorial problem. Counting the number of solutions to a problem is at least as hard as determining

2.1 Connectivity Games

In this paper we consider a particular network reliability model. Consider a communication network connecting various servers (or, equivalently, vertices); a certain subset of the servers are designated "primary" servers. Our goal is to make sure we can send information between any two primary servers. A server in the network may malfunction, and if it does, we cannot send information through it. If all the paths between two primary servers go through a failed server, we cannot send information between these two primary servers.

Of course, a communication network may be designed with redundancy, and have no single point of failure. However, even in this case, when more than one server fails, we may cease to be able to send information between two primary servers (depending on the available paths).

In our model, we will also assume that there can be a certain subset of servers that are guaranteed never to fail (guaranteed, say, by heavy maintenance and fail-safe backup); we will call these "backbone" servers. Our question is, given our desire to ensure communication paths between "primary" vertices, which servers on the network are most critical? Given limited resources to make sure that some servers do not fail (i.e., making them backbone servers), on which vertices should we concentrate to ensure communication between primary servers?

We model the above problem as a coalitional game, called the vertex *Connectivity Game.* This game is modeled so that the Banzhaf power index in the game will give us a good indication of the criticality of vertices.

Definition 4. *A vertex Connectivity Game Domain (CGD) consists of a graph $G = \langle V, E\rangle$ where the vertices are partitioned into primary vertices $V_p \subseteq V$, backbone vertices $V_b \subseteq V$, and standard vertices $V_s \subseteq V$. We require that $V_p \cap V_b = \emptyset$, $V_b \cap V_s = \emptyset$, $V_p \cap V_s = \emptyset$, and that $V = V_p \cup V_b \cup V_s$, so this is indeed a partition.*

Given a CGD we can define the vertex Connectivity Game. In this game, each agent controls one of the standard servers. A coalition wins if it connects all pairs of primary vertices (so that it is able to send information between any two such primary servers). Let $|V_s| = n$, and consider a set of n agents $I = (a_1, \dots, a_n)$, so that agent a_i controls vertex $v_i \in V_s$. Given a coalition $C \subseteq I$ we denote the set of vertices that C controls as $V(C) = \{v_i \in V_s | a_i \in C\}$. Coalition C can use either the vertices in $V(C)$ or the always available backbone vertices V_b. In our model, we assume that the coalition can also use any of the primary vertices V_p as well.[3]

We say a set of vertices $V' \subseteq V$ fully connects V_p if for any two vertices $u, v \in V_p$ there is a path $(u, p_1, p_2, \dots, p_k, v)$ from u to v going only through vertices in V', so for all i we have $p_i \in V'$.

Definition 5. *A vertex Connectivity Game (CG) is a simple coalitional game, where the value of a coalition $C \subseteq I$ is defined as follows:*

$$v(C) = \begin{cases} 1 & \textit{if } V(C) \cup V_b \cup V_p \textit{ fully connects } V_p \\ 0 & \textit{otherwise} \end{cases}$$

if there is at least one solution, so #P-complete problems are at least as hard (but possibly harder) than NP-complete problems.

[3]Another possibility would be to allow some of the primary vertices we want to connect to fail. In this case a coalition would win if it manages to connect all the non-failed primary vertices. We could also disallow sending information through the primary vertices (so they can only be the final destination). Most of the results in this paper hold for these different settings as well.

We would like to locate important points of failure in the network, and we will compute the Banzhaf power index in the CG to enable us to identify possible important points of failure. Since we are interested in being able to send information between any two primary servers, we want to identify the servers which, when failing, can cause us to lose connectivity between primary servers. Suppose all the servers have an equal probability of working or failing the next day. When these failures are independent, any subset of the servers has an equal chance of surviving. Thus, we have a certain probability of having the surviving set of servers fully connect the primary servers.

Suppose we can make sure that exactly one server, owned by agent a_i, always survives. The Banzhaf power index measures the probability of having the surviving subset of vertices (which now contains v_i with probability 1) fully connect the primary servers. When attempting to maximize the probability of achieving our goal, the higher the Banzhaf index of a server is, the more we should try and make sure that server does not fail. Thus, in order to find significant points of failure, we can calculate the Banzhaf power index, and focus on the servers with the highest indices.

3. COMPUTING THE BANZHAF POWER INDEX IN CONNECTIVITY GAMES

We now consider the computational complexity of calculating the Banzhaf power index in general vertex connectivity games. We first formally define the problem.

Definition 6. *CG-BANZHAF: We are given a CG over the graph $G = \langle V, E\rangle$, with primary vertices $V_p \subseteq V$, backbone vertices $V_b \subseteq V$, and standard vertices $V_s \subseteq V$. There are $n = |V_s|$ agents, $I = (a_1, \dots, a_n)$, so agent a_i controls vertex $v_i \in V_s$. The game's coalitional function $v : 2^I \rightarrow \{0, 1\}$ is defined as in Definition 5. We are also given a specific target agent a_i, and are asked to calculate its Banzhaf power index in this game, $\beta_i(v)$.*

We now show that in general CGs, this problem is #P-complete. We first prove it is in #P, and then reduce a #SET-COVER problem to CG-BANZHAF. We begin with a few definitions.

Definition 7. *#SET-COVER (#SC): We are given a collection $C = \{S_1, \dots, S_n\}$ of subsets. We denote $\cup_{S_i \in C} S_i = S$. A set cover is a subset $C' \subseteq C$ such that $\cup_{S_i \in C'} = S$. We are asked to compute the number of covers of S.*

A slightly different version requires finding the number of set covers *of size at most k*:

Definition 8. *#SET-COVER-K (#SC-K): A set-cover with size k is a set cover C' such $|C'| = k$. As in Definition 7, we are given S and C* and a target size k, *and are asked to compute the number of covers of S* of size at most *k.*

Both #SC and #SC-K are #P-hard. [7] shows that #SC-K is #P-hard: it considers several basic NP-complete problems, and shows that their counting versions are #P-complete. The counting version of SET-COVER discussed there is #SC-K. #VERTEX-COVER is a restricted form of #SC. [19] shows that #VERTEX-COVER is #P-hard[4] so #SC is of course also #P-hard. To prove that CG-BANZHAF is #P-complete, we show a reduction from #SC to CG-BANZHAF.[5]

In order to show that CG-BANZHAF is #P-complete we need to show two things: first, that CG-BANZHAF is *in* #P, and second, a reduction of a #P-hard problem to a CG-BANZHAF instance.

LEMMA 1. *CG-BANZHAF is in #P.*

PROOF. The Banzhaf index of a_i in a CG v is $\beta_i(v)$, the proportion of all winning coalitions where a_i is critical, out of all winning coalitions that contain a_i. Given a certain coalition $C \subseteq I$, it is polynomial to check whether it wins—we only need to check whether $V(C) \cup V_b$ fully connects V_p. We can do this by creating a new graph G', dropping all edges that miss $V(C) \cup V_b$ from G (i.e., we drop any edge $(x, y) \in E$ that either $x \notin V(C) \cup V_b$ or $y \notin V(C) \cup V_b$). We then check if any two primary vertices in G' are connected (there are several polynomial algorithms to do this; a simple one is to run a depth-first search [DFS] between all pairs of primary vertices). We can thus easily test if a certain agent a_i is critical for a coalition: we perform the above test when he is in the coalition, remove him, and repeat the test. If the first test succeeds and the second fails, that agent is critical for that coalition.

Since we can construct a deterministic polynomial Turing machine M that tests if an agent is critical in a coalition, we can construct a non-deterministic Turing machine M', that first non-deterministically chooses a coalition (where a_i is always in the coalition), and then tests if a_i is critical in that coalition. The number of accepting paths of M' is the number of coalitions that contain a_i where a_i is critical. Denote by k the number of such accepting paths of M', and denote $|I| = |V_s| = n$. Then the Banzhaf power index of agent a_i is $\beta_i(v) = \frac{k}{2^{n-1}}$.

Calculating the numerator of $\beta_i(v)$ is thus, according to Definition 3, a problem in #P. Since the denominator is constant (given a domain with n agents), CG-BANZHAF is in #P. □

We now show that CG-BANZHAF is #P-hard. We do this by a reduction from #SC. Figure 1 shows an example of such a reduction for a specific #SC instance.

THEOREM 1. *CG-BANZHAF is #P-hard, even if there are no backbone vertices, i.e., $V_b = \emptyset$.*

PROOF. We reduce a #SC instance to a CG-BANZHAF instance. Consider the #SC instance with the collection $C = \{S_1, \ldots, S_n\}$, so that $\cup_{S_i \in C} S_i = S$. Denote the items in S as $S = \{t_1, t_2, \ldots, t_k\}$. Denote the items in S_i as $S_i = \{t_{(S_i,1)}, t_{(S_i,2)}, \ldots, t_{(S_i,k_i)}\}$. The reduced CGD is constructed with a graph $G = \langle V, E\rangle$ as follows. For each subset $S_i \in C$, the reduced CG instance has a vertex v_{S_i}. We denote the set of v_{S_i} vertices $V_{sets} = \cup_{\{i|S_i \in C\}} v_{S_i}$. For each item $t_i \in S$ the reduced CG instance also has a vertex v_{t_i}. We denote the set of v_{t_i} vertices $V_{items} = \cup_{i|t_i \in S} v_{t_i}$. The reduced CG instance also has two special vertices v_a and v_b. These are all the vertices of the reduced instance.

The vertices in the reduced CG are connected in the following way. The vertices V_{sets} are a clique: for every $v_i, v_j \in V_{sets}$, $(v_i, v_j) \in E$. The vertex v_a is also a part of that clique, so for all $v_i \in V_{sets}$ we have $(v_i, v_a) \in E$. The vertex v_a is connected to v_b, and is the only vertex connected to v_b, so $(v_a, v_b) \in E$. Each set vertex v_{S_i} is connected to all the vertices of the items in that set, $v_{t_{(S_i,1)}}, v_{t_{(S_i,2)}}, \ldots, v_{t_{(S_i,k_i)}}$, so for any $v_{S_i} \in V_{sets}$ and any $v_{t_{(S_i,j)}}$ (so that $t_{(S_i,j)} \in S_i$) we have $(v_{S_i}, v_{t_{(S_i,j)}}) \in E$.

We define the CG so that $V_i = V_{items} \cup \{v_b\}$, $V_b = \emptyset$, $V_s = V_{sets} \cup \{v_a\}$, and the CG game is defined as in Definition 5. The game has $m = |V_s| = |V_{sets}| + 1 = |C| + 1 = n + 1$ agents (where n is the number of subsets in C, the input to the #SC problem). The CG-BANZHAF query is regarding v_a. Let $\beta_i(v)$ be the answer to the CG-BANZHAF query, and k be the number of set covers in the #SC instance. We show that $k = \beta_a(v) \cdot 2^{m-1}$, by providing a one-to-one mapping between a set-cover of the original problem and a winning coalition where v_a is critical in the reduced CG.

Consider a set-cover $C' \subseteq C$ for S. C' must cover all the items t_i in S. We denote the set of vertices corresponding to the sets in this vertex cover $V_{C'} = \{v_{S_i} \in V_{sets} | S_i \in C'\}$. Since C' is a set cover for the original problem, each vertex $v_{t_j} \in V_{items}$ in the reduced graph must be connected to at least one vertex $v_i \in V_{sets}$. Since the vertices V_{sets} are a clique, in the reduced CG all the v_{t_i}'s and v_{S_j}'s are in the same connected component. However, without v_a we cannot reach v_b from any vertex. Thus, $V_{C'} \cup \{v_a\} \subseteq V_S$ is a winning coalition in the reduced CG, but $V_{C'}$ is not, so v_a is critical for that coalition. We now show the mapping in the reverse direction. Consider a coalition $V' \subseteq V_s$ where v_a is critical, and denote $C' = \{S_i \in C | v_{S_i} \in V'\}$. By definition, V' must be winning and contain v_a. Consider any vertex $v_{t_i} \in V_{items}$. Since V' wins, it must allow any vertex in V_{sets} to reach v_{t_i}, which can only happen if V' contains some v_{S_j} so that $t_i \in S_j$. Thus, C' is a set cover for the original problem.

Let k be the number of set covers in the #SC instance, and c_a be the number of winning coalitions where v_a is critical in the reduced CG. Due to the one-to-one mapping we have shown, $k = c_a$. But by the definition of the Banzhaf index, in the reduced CG we have $\beta_a(v) = \frac{c_a}{2^{m-1}}$, so $c_a = \beta_a(v) \cdot 2^{m-1}$, and then $k = \beta_a(v) \cdot 2^{m-1}$.

We have shown that given a polynomial algorithm for CG-BANZHAF, we can solve #SC in polynomial time, so CG-BANZHAF is #P-hard. □

Having demonstrated that CG-BANZHAF is in #P and that it is #P-hard, we have completed the proof that it is #P-complete. Thus, it is unlikely that a polynomial algorithm for calculating the Banzhaf power index in vertex connectivity games would be found. We can circumvent this computational problem in several ways. One is to try and find an approximation algorithm, and the other is to solve

[4]It also shows that the problem remains #P-hard even in very restricted classes of graphs.

[5]We use #SC to prove that CG-BANZHAF is #P-hard. It is easy to show that #SC-K is #P-complete, but the fact that #SC is #P-complete is more difficult to prove (and is thus not very well known). We give the definitions of both #SC and #SC-K to avoid confusion between them, and use the result from [19] which indicates that #SC is #P-complete.

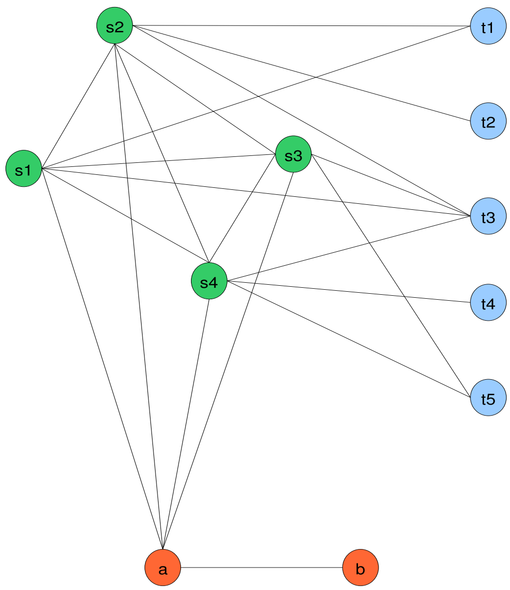

Figure 1: Example of reducing #SC to CG-BANZHAF. The items are $\{t_1, t_2, t_3, t_4\}$ and the sets are $S_1 = \{t_1, t_3\}$, $S_2 = \{t_1, t_2, t_3\}$, $S_3 = \{t_3, t_5\}$, $S_4 = \{t_3, t_4, t_5\}$.

the problem for restricted instances. In the next section, we adopt this second approach.

4. CALCULATING THE BANZHAF POWER INDEX IN TREE CGS

Although computing the Banzhaf power index in general CGs is #P-complete, restricting the structure of the graph in the domain may allow us to polynomially compute this index. We now provide a polynomial algorithm for the case where the graph is a tree. Consider a CG with graph $G = \langle V, E \rangle$ *that is a tree*, with primary vertices $V_p \subseteq V$, backbone vertices $V_b \subseteq V$, and standard vertices $V_s \subseteq V$. We call the problem of calculating the Banzhaf power index of an agent in this domain *TREE-CG-BANZHAF*. We will start with a few useful lemmas before we introduce our algorithm. We assume that there are at least 2 primary vertices $v_a, v_b \in V_p$ (otherwise, any subset of the vertices trivially fully connects the primary vertices). We first note that since the graph is a tree, some of the vertices are present in any winning coalition.

LEMMA 2. *Consider a CG where the graph G is a tree. Let $v_a, v_b \in V_p$ be two primary vertices, and a standard vertex $v_r \in V_s$ is on a simple path from v_a to v_b. Then v_r is present in* all *winning coalitions in the CG game.*

PROOF. Since G is a tree, there is only one simple path between v_a and v_b. The removal of any vertex along that simple path would make v_b unreachable from v_a. Since v_r is such a vertex, any coalition $C \subset V_s$ such that $v_r \notin C$ loses, and any winning coalition must contain v_r. □

Any standard vertex on the simple path between two primary vertices is required to be in a coalition in order for it to win. We now show that having all of these vertices allows us to connect the two primary vertices.

LEMMA 3. *Consider a CG where the graph G is a tree. Let $v_a, v_b \in V_p$ be two primary vertices. Consider a vertex coalition $C \subset V_s$ that contains all standard vertices $v_r \in V_s$ on the single simple path from v_a to v_b. Then C allows the connecting of v_a and v_b in the CG game.*

PROOF. In the CG game we are allowed to use any primary vertex $v_p \in V_p$, and any backbone vertex $v_b \in V_b$. We consider a coalition C that contains all vertices $v_r \in V$ on the single simple path from v_a to v_b. Any vertex v_x on the single simple path between v_a and v_b is either a backbone vertex (so $v_x \in V_b$) or a primary vertex (so $v_x \in V_p$) or a standard vertex (so $v_x \in V_s$). If it is a standard vertex, it is in the coalition, so $v_x \in C$. In any of these cases we are allowed to use it in the CG game, so v_a and v_b are in the same connected component for the coalition C. □

We now consider a coalition that contains any standard vertex on any simple path between two primary vertices, and show that it is a winning coalition.

LEMMA 4. *Consider a CG where the graph G is a tree. Consider a vertex coalition $C \subset V_s$ that contains all standard vertices $v_r \in V_s$ on any simple path between any two primary vertices $v_a, v_b \in V_p$ (that is, for any $v_a, v_b \in V_p$ and any vertex $v_r \in V_S$ such that v_r is on a simple path between v_a and v_b, we have $v_r \in C$). Then C is a winning coalition in the CG game, so $v(c) = 1$.*

PROOF. Consider two primary vertices v_a and v_b. Due to Lemma 3, v_a and v_b are connected in the CG game—there is a simple path composed of either backbone, primary, or coalition vertices between them. But v_a and v_b were *any* primary vertices, so there is such a simple path between any two primary vertices. Thus C fully connects the primary vertices V_p, and it is therefore a winning coalition. □

On the one hand, due to Lemma 2 any vertex which is on the path between some two primary vertices is a member of any winning coalition, so any winning coalition contains all such vertices. On the other hand, due to Lemma 4 any coalition that contains all such vertices is winning. Thus we get the following corollary.

COROLLARY 1. *The winning coalitions are exactly those coalitions that contain all the standard vertices on any simple path between two primary vertices.*

When calculating the Banzhaf power index of v_i, we are interested in counting not any winning coalition, but only those in which v_i is critical. Due to Lemma 1, if v_i is not found on any simple path between some two primary vertices, it is not critical in any coalition C: if C contains all such vertices between any two primary vertices, then C is winning, but C also wins without v_i, and if C misses even one such vertex, it is a losing coalition. Thus, we get the following corollary.

COROLLARY 2. *If v_i is not found on any simple path between some two primary vertices, the Banzhaf power index of v_i in the CG is $\beta_i(v) = 0$.*

PROOF. v_i is not critical in any coalition. Since the Banzhaf index is the proportion of coalitions where v_i is critical out of all coalitions that contain v_i, $\beta_i(v) = 0$. □

A vertex that is found on a simple path between two primary vertices is critical in all winning coalitions, so we get the following corollary.

COROLLARY 3. *If v_i is found on any simple path between some two primary vertices, the Banzhaf power index of v_i in the CG is $\beta_i(v) = 1$.*

PROOF. Due to Lemma 2, v_i is critical in *all* winning coalitions. Since the Banzhaf index is the proportion of coalitions where v_i is critical out of all coalitions that contain v_i, $\beta_i(v) = 1$. $\square$

We now show that TREE-CG-BANZHAF is in P, by providing a polynomial algorithm for this problem.

THEOREM 2. *TREE-CG-BANZHAF is in P.*

PROOF. The above corollaries allow us to polynomially calculate the Banzhaf power index of any agent in the CG. We simply mark all the standard vertices that are found between any two primary vertices. Given two primary vertices $v_a, v_b \in V_p$ we can do the marking as follows. We perform a DFS, which defines a parent-son relation for the vertices in the tree.[6] We then start at v_b, continue to its parent $p(b)$ and so on until we reach v_a, and mark any standard vertex $v_x \in V_s$ along the way. After performing this process for all pairs of primary vertices, we can easily calculate the Banzhaf index of every vertex. Due to Corollary 2 and Corollary 3, marked vertices have a Banzhaf index of 1, and the others have a Banzhaf index of 0. Since DFS takes polynomial time, and we perform a DFS $|V_p|^2$ times, this is a polynomial algorithm. $\square$

We have thus shown that despite the high complexity result for the general case derived above in Section 3, in the restricted domain where the graph is a tree, we can polynomially calculate the Banzhaf power index. This restricted case can be of value for analyzing reliability in real-world networks. As an example, consider the situation where internet connectivity is established between companies, where one company is the supplier and another company is the client. An example of a cycle in this relationship would be if company A buys an internet connection from company B, which in turn buys an internet connection from company C, which eventually buys an internet connection from company A. This would mean that, in a sense, company A would have become a client of itself, and would be paying money for its own connection. This scenario, with a cycle, is unlikely; many interesting domains where we may want to use network reliability models are, in fact, trees.

5. THE CORE OF CGS

Consider a CG domain where a certain reward is promised to any coalition that manages to connect all the primary vertices. Agents who form a winning coalition get this reward as a group, and must then decide how to divide the reward among the coalition members. In this case, even when a winning coalition is formed, it may not be stable—agents who are given a small share of the reward may attempt to form another coalition, where they may get a higher share of the reward, and thus increase their own utility. One prominent game theoretic solution concept that takes such considerations into account is the core, which can thus be used to allocate the gains of a coalition in a CG domain in a stable way.

We now consider the problem of computing the core of general CGs. When the core is non-empty, it contains payoff vectors that are stable. When it is empty, the coalition would be unstable no matter how we divide the utility among the agents. We first note that it is not always possible to concisely represent the core, since it may contain an infinite number of payoff vectors. However, in the case of CGs, it can be done.

We first note that the core is a very demanding concept in simple coalitional games (where the coalitional function can only take 0 or 1 as values). An agent a_i is a *veto player* if it is present in all winning coalitions, so if $a_i \notin C$ we have $v(C) = 0$. It is a well-known fact that in simple coalitional games, the core is non-empty if and only if there is at least one veto player in the game. Consider a simple coalitional game that has no veto players. For every agent a_i we have a winning coalition that does not contain a_i. Take a payoff vector $p = (p_1, \ldots, p_n)$ where $p_i > 0$. Since $\sum_{i=0}^{n} p_i = 1$ and since $p_i > 0$ we know that $p(C) \leq \sum_{p_j \in I_{-a_i}} p_j < 1$, so $p(C) < v(C) = 1$, which makes C a blocking coalition. On the other hand, we can see that any payoff vector p where non-veto players get nothing is in the core: any coalition C that can potentially block p must have $v(C) = 1$ (if $v(C) = 0$ it cannot block), and must contain all the veto players, so $\sum_{p_j \in C} p_j = 1$, and thus cannot block p. Due to this fact, calculating the core of simple games simply requires obtaining a list of veto players in that game.

We now consider computing the core in CGs. We first show a monotonicity property of CGs.

LEMMA 5. *Let $W \subseteq I$ be a winning coalition in a CG (so $v(W) = 1$), and let $C \subseteq I$ be any coalition in that game. Then $W \cup C$ is also a winning coalition (so $v(W \cup C) = 1$). This can be restated as, for all coalitions $A, B \subseteq I$ in a CG we have $v(A \cup B) \geq v(A)$.*

PROOF. If C fully connects V_p then $W \cup C$ also fully connects V_p, since we are now allowed to use even more vertices in our paths (so all the paths that had existed before between some $v_a, v_b \in V_p$ also remain). $\square$

We now denote the set of all the agents except a_i as $I_{-i} = I \setminus \{a_i\}$. Let G be the CG graph. We denote by G_{-i} the same graph when we drop the vertex v_i owned by a_i, so $G_{-i} = \langle V_{-i}, E_{-i} \rangle$ where $V_{-i} = V \setminus \{v_i\}$ and $E_{-i} = \{(u, v) \in E | u \neq v_i \wedge v \neq v_i\}$. We now show a polynomial algorithm for testing if a player is a veto agent in CGs.

LEMMA 6. *Testing if agent a_i is a veto agent in a CG is in P.*

PROOF. We first show that I_{-i} is a losing coalition if and only if a_i is a veto agent. If I_{-i} is a losing coalition then due to Lemma 5 any sub-coalition of it, $C \subseteq I_{-i}$, is also losing. Thus, any coalition without a_i is losing, so a_i is a veto player. On the other hand, if I_{-i} is a winning coalition, it is a winning coalition where a_i is not present, so by definition a_i is not a veto player. Thus, to test if a_i is a veto agent we only need to test if I_{-i} is losing or winning. According to Definition 5 of CSG, to check if I_{-i} wins we need to check if I_{-i} fully connects the primary vertices. This test can be

[6] Vertex v is the parent of vertex u, denoted $v = p(u)$, if we found u when we popped v from the DFS stack.

performed in polynomial time by trying all pairs $v_a, v_b \in V_p$, and performing a DFS between v_a and v_b in the graph G_{-i}. Thus, we can check if an agent is a veto agent in a CG in polynomial time. □

Since computing the core in simple coalitional games simply requires returning a list of all the veto agents, we get the following corollary.

COROLLARY 4. *It is possible to compute the core of a CG in polynomial time.*[7]

PROOF. Computing the core of CG requires returning a list of the veto players in the game. Due to Lemma 6, we can check each agent and see if it is a veto player. Thus, testing all the agents and finding all the veto players can also be done in polynomial time. If there are no veto players, the core is empty. Otherwise, any payoff vector that distributes 1 (the total utility $v(I) = 1$) among the veto players and gives none to the non-veto players is in the core. □

Due to the above proofs, if we adopt the core requirement for stability, there is a stable way of dividing the total utility among the agents only if there is a least one vertex which is always required to fully connect the primary vertices.

6. RELATED WORK

Power indices originated in work in game theory and political science, attempting to measure the power that players have in weighted voting games. In these games, each player has a certain weight, and a coalition's weight is the sum of the weights of its participants. A coalition wins if its weight passes a certain threshold. This is a common situation in legislative bodies. Power indices have been suggested as a way of measuring the influence that players in such a game have on choosing outcomes. The most popular indices suggested for such measurement are the Banzhaf index [2] and the Shapley-Shubik index [17].

The Shapley-Shubik index [17] is the direct application of the Shapley value [16] to simple coalitional games. The Banzhaf index emerged directly from the study of voting in decision-making bodies, where a certain *normalized* form of the index was introduced [2]. The Banzhaf index was later mathematically analyzed in [4], where it was shown that this normalization has certain undesirable qualities, and the standard Banzhaf index is introduced. These indices were applied in an analysis of the voting structures of the IMF and the European Union Council of Ministers (as well as many other bodies) [10, 11]. The Shapley value can also be used to fairly allocate costs in various domains. One such example is [6], where the Shapley value is used for the fair allocation of the costs of multicast transmissions.[8] Our work considers a different application of power indices, using them to locate reliability problems in a network. We believe power indices and other game theoretic notions can be very helpful in analyzing computer networks, and [6] provides a good example of this.

[7] In fact, the same can be done for any simple monotone coalitional game where the value of a coalition can be computed in polynomial time: due to the same proof of Lemma 6, in such games we can test whether an agent is a veto player, and in simple games computing the core simply requires finding out who the veto players are.

[8] We thank an anonymous reviewer for pointing out this work to us.

The differences between the Banzhaf and Shapley-Shubik indices were analyzed in [18], where it was shown that each index reflects specific conditions in a voting body. An axiomatization of these indices (as well as several others) was given in [9]. The Banzhaf index reflects the assumption that agents are independent in their choices, and is thus more appropriate for our network reliability problem.

It is possible to calculate the Banzhaf power index for any simple coalitional game. However, the complexity of such a procedure depends on the representation of the game. When the game is defined only by the value of each coalition, in the form of an oracle which tests a certain coalition and answers whether it wins or loses, calculating the Banzhaf power index is problematic. A naive implementation of an algorithm for calculating the Banzhaf index of an agent a_i enumerates over all coalitions containing a_i. Since there are 2^{n-1} such coalitions, the naive algorithm is exponential in the number of agents.

[13] surveys algorithms for calculating power indices in weighted majority games, and [14] shows that calculating both the Banzhaf and Shapley-Shubik indices in weighted voting games is NP-complete. Since weighted voting games are a restricted case of simple coalitional games, the problem of calculating either index in a general coalitional game is of course NP-hard. In fact, in certain cases, calculating power indices is not just NP-hard but also #P-hard. [3] shows that computing the Shapley-Shubik index in weighted voting games is #P-complete.

The solution concept of the core originated in [8]; it focuses on stability of the coalition. The Shapley value is another well-known solution concept, and has been used to fairly allocate the gains of a coalition. While it is known that the Shapley value is the only value division where certain fairness axioms hold, it is susceptible to some forms of strategic behavior, as noted in [21].

Fair allocation of the gains of the coalition using the Shapley value has been explored in many papers. However, using it to measure power in weighted voting systems, as in [17], has been much less studied. The use of power indices for game types other than weighted voting games is indeed rare. One example of such use is [1], which considered a network reliability problem. In that scenario, agents control edges in a network flow graph, where a coalition wins if it can maintain a certain flow between a source and a target. [1] shows that finding the Banzhaf index of an edge in this domain is #P-complete, and gives an algorithm for a restricted case. Our research in this paper handles a scenario very different from [1]—our agents are required to maintain *connectivity*, rather than a certain flow. Also, we are interested in maintaining this connectivity between *every* two primary vertices, rather than two specific vertices (we can simulate the case of two specific servers by having only two primary servers). Also, in this work, the agents are the servers in the communication network, rather than the links.

While our treatment of the model is game theoretic, such problems can be formulated as network reliability problems. The computational complexity of such problems has been studied in several papers. Classical network reliability problems consider an undirected graph $G = \langle V, E \rangle$, when each edge $e \in E$ has a probability assigned to it, p_e. This is the probability that edge e remains in the surviving graph.

One prominent problem is s-t connectivity probability (STC-P): given the above domain, compute the probability of having a path between $s, t \in V$ in the surviving graph. Another prominent problem is full connectivity probability (FC-P): given the above domain, compute the probability that the surviving graph is connected (so that there is a path between any two vertices). One seminal paper by Valiant [20] proves that STC-P is #P-hard. Provan and Ball [15] show that FC-P is also #P-hard.

The problem we study is similar to FC-P, but since we use the Banzhaf power index, we deal with a *very specific* case of the general problem, where the probability of every vertex subset is equal. Since this is a restricted case, we cannot use the hardness result of [15], and have to prove that even the restricted case is #P-complete (we did this in Section 3).

The complexity results shown in this paper (and in several papers dealing with similar domains) demonstrate the difficulty of using certain techniques in real-world applications; for example, in order to use the Banzhaf index to find network reliability bottlenecks, one must be able to calculate it. There are several ways to circumvent this problem. One way is to find algorithms that work for restricted cases of the problem (we used this approach in Section 4). Another approach is approximating these power indices. Several such approximation methods have been suggested. [12] suggests approximating the Shapley-Shubik power index using a Monte-Carlo technique; [13] shows a similar method for the Banzhaf power index. Such results make it more tractable to use power indices in real-world applications.

7. CONCLUSIONS

We have considered some computational aspects of a game theoretic approach to network reliability. We modeled a communication network as a simple coalitional game, and have shown that power indices can be used to find significant possible points of failure. We have shown that in this domain, for general graphs, computing the Banzhaf power index is #P-complete. Despite this high complexity result for the general domain, we also gave a polynomial result for the restricted domain where the graph is a tree. We have also shown that computing the core can be done in polynomial time in any CG, and gave a simple characterization of the instances when the core is non-empty in CGs.

It remains a topic of future research to tractably compute or approximate the Banzhaf index in general graph CGs, and to find other interesting restricted domains where this index can be calculated polynomially. We also note that the Banzhaf power index is just one of several such game theoretic power indices that could be used to analyze such domains; another open question is that of computing other game theoretic solution concepts in this domain.

8. ACKNOWLEDGMENT

This work was partially supported by Israel Science Foundation grant #898/05.

9. REFERENCES

[1] Y. Bachrach and J. S. Rosenschein. Computing the Banzhaf power index in network flow games. In *The 6th Int. Joint Conference on Autonomous Agents and Multiagent Systems*, pages 323–329, May 2007.

[2] J. F. Banzhaf. Weighted voting doesn't work: a mathematical analysis. *Rutgers Law Review*, 19:317–343, 1965.

[3] X. Deng and C. H. Papadimitriou. On the complexity of cooperative solution concepts. *Math. Oper. Res.*, 19(2):257–266, 1994.

[4] P. Dubey and L. Shapley. Mathematical properties of the Banzhaf power index. *Mathematics of Operations Research*, 4(2):99–131, 1979.

[5] E. Elkind, L. A. Goldberg, P. W. Goldberg, and M. Wooldridge. Computational complexity of weighted threshold games. In *The National Conference on Artificial Intelligence*, pages 718–723, 2007.

[6] J. Feigenbaum, C. H. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences*, 63(1):21–41, 2001.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., 1979.

[8] D. B. Gillies. *Some theorems on n-person games*. PhD thesis, Princeton University, 1953.

[9] A. Laruelle. On the choice of a power index. Papers 99-10, Valencia — Instituto de Investigaciones Economicas, 1999.

[10] D. Leech. Voting power in the governance of the International Monetary Fund. *Annals of Operations Research*, 109(1-4):375–397, 2002.

[11] M. Machover and D. S. Felsenthal. The treaty of Nice and qualified majority voting. *Social Choice and Welfare*, 18(3):431–464, 2001.

[12] I. Mann and L. S. Shapley. Values of large games, IV: Evaluating the electoral college by Monte-Carlo techniques. Technical report, The Rand Corporation, Santa Monica, CA, 1960.

[13] Y. Matsui and T. Matsui. A survey of algorithms for calculating power indices of weighted majority games. *Journal of the Operations Research Society of Japan*, 43, 2000.

[14] Y. Matsui and T. Matsui. NP-completeness for calculating power indices of weighted majority games. *Theoretical Computer Science*, 263(1–2):305–310, 2001.

[15] J. S. Provan and M. O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing*, 12(4):777–788, 1983.

[16] L. S. Shapley. A value for n-person games. *Contributions to the Theory of Games*, pages 31–40, 1953.

[17] L. S. Shapley and M. Shubik. A method for evaluating the distribution of power in a committee system. *American Political Science Review*, 48:787–792, 1954.

[18] P. Straffin. Homogeneity, independence and power indices. *Public Choice*, 30:107–118, 1977.

[19] S. P. Vadhan. The complexity of counting in sparse, regular, and planar graphs. *SIAM Journal on Computing*, 31(2):398–427, 2002.

[20] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8:410–421, 1979.

[21] M. Yokoo, V. Conitzer, T. Sandholm, N. Ohta, and A. Iwasaki. Coalitional games in open anonymous environments. In *The 20th National Conference on Artificial Intelligence*, pages 509–514, 2005.