Verifying Nash Equilibria in PageRank Games on Undirected Web Graphs

David Avis[1,2], Kazuo Iwama[1], and Daichi Paku[1]

[1]School of Informatics, Kyoto University, Japan
[2]School of Computer Science, McGill University, Canada

Abstract. J. Hopcroft and D. Sheldon originally introduced the PageRank game to investigate the self-interested behavior of web authors who want to boost their PageRank by using game theoretical approaches. The PageRank game is a multiplayer game where players are the nodes in a directed web graph and they place their outlinks to maximize their PageRank value. They give best response strategies for each player and characterize properties of α-insensitive Nash equilibria. In this paper we consider PageRank games for undirected web graphs, where players are free to delete any of their bidirectional links if they wish. We study the problem of determining whether the given graph represents a Nash equilibrium or not. We give an $O(n^2)$ time algorithm for a tree, and a parametric $O(2^k n^4)$ time algorithm for general graphs, where k is the maximum vertex degree in any biconnected component of the graph.

Keywords: PageRank, Game theory, Nash equilibria, Fractional optimization

1 Introduction

A PageRank value is assigned to each web page according to the stationary distribution of an α-random walk on the web graph. Here an α-random walk is a random walk modified to make a random jump with probability α at each step and a random jump is a move to a node according to a given distribution vector q. Introduced by Larry Page and Sergey Brin, it is an important basis of the Google search engine and possibly one of the most successful applications of a mathematical concept in the IT world.

Unlike rankings based on content such as keywords, tags, etc., PageRank focuses solely on the hyperlink structure of the given web graph. Web links themselves possess strategic worth and hence web authors often try to boost the PageRank of their web pages by carefully choosing links to other pages. Since these authors behave strategically in a self-interested way, this is a typical example of a non-cooperative game. In fact, Sheldon and Hopcroft recently introduced the PageRank game as a game theoretic model played over a directed graph [2]. Each player is a node and their strategy is to specify a set of outlinks for their node. The payoff for each player is the PageRank value for their node which is calculated on the resulting directed graph; the obvious goal of each player is to maximize their payoff.

In [2], the authors proved a nice property of this game, namely the best strategy of a player v is to place her outlinks to the nodes u having largest potential value ϕ_{uv}. The potential ϕ_{uv} measures the probability of returning to v before the first jump and does not depend on the outlinks from v if the other nodes do not change their outlinks. Thus, a simple greedy algorithm exists for deciding if a given graph is in Nash equilibrium and even a nice characterization of Nash equilibria graphs is possible. Interestingly, it turns out that such graphs representing Nash equilibria have very strong regularity properties as shown in Fig. 1 (a) and (b) (see Section 2 for details).

In this paper we consider undirected rather than directed graphs, where a player cannot unilaterally create a new link but may unilaterally delete an existing one. As motivation, in the web graph model described above, what v intuitively does is to cut its links to nodes u having smaller ϕ_{uv} values, assuming that u will not delete its edge to v. In the new model, if v cuts its outlink to u we assume that u also cuts its outlink to v either automatically, or as a form of revenge. Therefore all links are necessarily bidirectional and an undirected graph model is obtained. A second motivation is that PageRank has been applied directly to settings involving undirected graphs. For example one could consider international bilateral agreements between universities, with PageRank measuring how international a university is. The 'friend' relationship on Facebook defines an undirected graph, and PageRank is a possible measure of the influence of a given user, etc. It turns out that the class of equilibria graphs in the undirected model is larger. For instance, the graph in Fig. 1 (c) is now added to the class. Unfortunately, the nice property of the original model that ϕ_{uv} does not depend on the outlinks from v, no longer holds. Hence the greedy algorithm for the Nash decision problem does not work, either, and there seems to be no obvious way of checking the equilibrium condition.

Our Contribution. To our knowledge this is the first paper to consider PageRank games in undirected graphs even though, as described above, there are many natural applications of PageRank to undirected graphs.

Our first result is an algorithm for the case where the graph is a tree and runs in time $O(n^2)$. This is explained in Section 3. Our second result is a parametric algorithm for general graphs, where the parameter k is the maximum degree of any vertex in any biconnected component that contains it. In Section 4 we give a $O(2^k n^4)$ time algorithm for determining if G is a Nash equilibrium. It draws on many of the ideas introduced in Section 3. Biconnected components roughly correspond to local clusters of web pages, where one could expect the parameter k to be relatively small. Nodes linking biconnected clusters may have arbitrarily large degree without changing the time complexity.

Our techniques make a nontrivial application of the optimization scheme given in [2]. One of the novel ideas is the application of line arrangements to speed up Megiddo's [3] technique for fractional optimization.

Related Work. Although it focuses less on game theoretic aspects, there is a large literature on optimal linking strategies to maximize the PageRank of given nodes for directed graphs. On the positive side, Avrachenkov and Litvak [5]

give a polynomial-time algorithm for maximizing the PageRank of a single node by selecting its outlinks. Kerchove et. al. [7] extend this result to maximizing the sum of the PageRank of a given set of nodes. Csaji et. al. [6] give a polynomial-time algorithm for maximizing the PageRank of a single node with any given set of controllable links.

On the negative side, [6] also shows that the problem becomes NP-hard if some pairs of controllable links in the set are mutually exclusive. Olsen [8] proved that maximizing the minimum PageRank in the given set of nodes is NP-hard if we are allowed to add k new links. He also proved that the problem is still NP-hard if we restrict the node set to a single node and the k links to only incoming ones to that node [9] and gives a constant factor approximation algorithm for this problem [10]. The question of whether there are α-sensitive Nash equilibria was recently affirmatively answered by Chen et. al. [11].

2 Preliminaries

2.1 PageRank values

Initially we describe the Hopcroft-Sheldon directed graph model. Let $D = (V, E')$ be a simple directed graph on node set V and arc set E', and let q be a probability distribution on V. Throughout the paper we let $n = |V|$ denote the number of nodes. For $v \in V$, let $\Gamma(v)$ denote the set of v's neighbours, A random jump is a move to a node according to the distribution vector q instead of using one of the outlinks of the current node. An α-random walk on D is a random walk that is modified to make a random jump with fixed probability α $(0 < \alpha < 1)$ at each step. The PageRank vector π over the n vertices in V is defined as the stationary distribution of the α-random walk. We define the potential matrix $\Phi = (\phi_{uv})$ such that for vertices $u, v \in V$, ϕ_{uv} is the probability that a random walk that starting from u visits v before the first jump ($\phi_{uv} = 1$ if $u = v$), which can be written as

$$\phi_{uv} = \frac{1-\alpha}{|\Gamma(u)|} \sum_{i \in \Gamma(u)} \phi_{iv}. \tag{1}$$

In order to calculate π, we have the following equation [2]:

$$\pi_v = \alpha \frac{\sum_{u \in V} q_u \phi_{uv}}{1 - \frac{(1-\alpha)}{|\Gamma(v)|} \sum_{i \in \Gamma(v)} \phi_{iv}}. \tag{2}$$

2.2 Directed PageRank games

In the PageRank games in [2] the players are the nodes V of a directed graph D and they attempt to optimize their PageRank by strategic link placement. A *strategy* for node v is a set of outlinks. An outcome is an arc set E' for D consisting of the outlinks chosen by each player. The payoff of each player is the value of PageRank which is calculated on D.

We say a player v is in *best response*, if v takes a strategy which maximizes v's PageRank in D. A directed graph D is a *Nash equilibrium* if the set of outlinks for each node is a best response: no player can increase her PageRank value by choosing different outlinks. Several results for best response strategies and for Nash equilibria were introduced in [2]. In particular they gave a characterization of α-insensitive Nash equilibria, which are graphs being Nash equilibria for all values $0 < \alpha < 1$ of the jump parameter.

2.3 Undirected PageRank games

In this paper we study similar games for undirected graphs. Let $G = (V, E)$ be an undirected graph on vertex set V and edge set E. Define the directed graph $D = (V, E')$ on the same vertex set V, where each edge uv in E gives rise to two arcs uv and vu in E'. In our model, the payoff of each player v for the graph G is the PageRank of v in the corresponding directed graph D.

In the undirected model, a player cannot unilaterally create a bidirected link to another node, but it can delete a bidirectional link. Therefore, we will say that a player v is in best response if v cannot increase her PageRank by any deletion of her (bidirectional) links. A Nash equilibrium is a graph for which every player is in best response. We consider the following problem:

Input: An undirected graph G, α, q.
Output: Is the input a Nash equilibrium? (yes/no)

An equivalent formulation is to decide whether no player can increase her PageRank for the given input, where she is only allowed to delete edges to her neighbours. As for directed graphs, we let $\Gamma(v)$ denote the neighbours of vertex v in G. A strategy for v is to retain a subset $E_v \subseteq \Gamma(v)$ of neighbours and delete edges to her other neighbours. Let x be a 0/1 vector of length $d_v = |\Gamma(v)|$, which indicates v's strategy. Formally, if $i \in E_v$ then $x_i = 1$, otherwise $x_i = 0$, for $i = 1, ..., d_v$. Let $\phi_{uv}(x)$ denote the potential function (1) for the subgraph of G formed by deleting edges $vi, i \in \Gamma(v) - E_v$. By (2) applied to the corresponding directed graph D the PageRank of v can be written as

$$\pi_v(x) = \alpha \frac{\sum_{u \in V} q_u \phi_{uv}(x)}{1 - (1 - \alpha)\frac{\sum_{i \in \Gamma(v)} \phi_{iv}(x) x_i}{\mathbf{1}^T x}}, \tag{3}$$

where $x \neq 0$. Let $\mathbf{1}_m$ denote a vector of ones of length m. Usually the length is clear by the context so for simplicity we may drop the subscript m. If the input is a Nash equilibrium then v is using a best response and no edge deletions for v will raise her PageRank. Therefore $\pi_v(\mathbf{1}_{d_v}) \geq \pi_v(x)$ for any 0/1 vector x.

The approach we will use to solve the problem described in this section is to compute the maximum of $\pi_v(x)$ over all 0/1 vectors x of length d_v, for each vertex v. In the next section we give a quadratic algorithm for the special case when G is a tree, and in Section 4 we give a FPT-algorithm for general graphs.

We give some examples in Fig. 1. Graphs (a), (b) are α-insensitive Nash equilibria in directed PageRank games, and are also a Nash equilibrium in our

model for any given α and q. Graph (c) is an example which is not a Nash equilibrium in directed games, since v can increase its PageRank if we delete the arc from v to 2 which has less potential than 1. However (c) is a Nash equilibrium in our model for $\alpha = 0.15$ and uniform distribution q, since if v cuts its edge to 2 then it decreases v's PageRank. Graph (d) is not a Nash equilibrium in both directed and undirected games for $\alpha = 0.15$ and uniform distribution q, where K_m is a m-clique, $m = 8$ for example. In this graph the potentials from 2, 3 to v are much less than 1, so v may try to cut the edge to 2 or the edge to 3, but a single edge deletion decreases v's PageRank. Interestingly, the deletion of both edges leads to a greater PageRank for v.

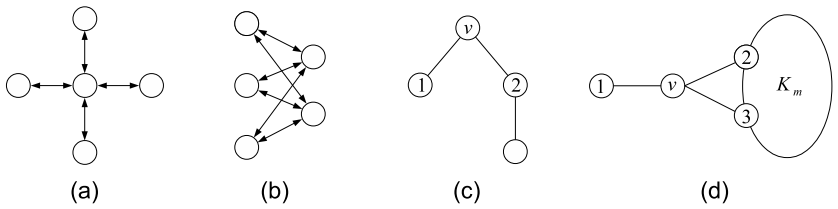

Fig. 1. Examples

3 Trees

In this section, we study the problem described in the previous section where the graph G is a tree. We prove the following theorem.

Theorem 1. *Given a tree G, jump probability α and distribution q, we can determine in $O(n^2)$ time whether this is a Nash equilibrium and if not give an improving strategy for at least one player.*

The remainder of this section is devoted to the proof of this theorem.

Let v be a node in G, let $\Gamma(v)$ be the set of neighbours of v, and let $d_v = |\Gamma(v)|$. Consider any strategy for v as described in Section 2.3 and let x be the 0/1 vector that represents it. For $i \in \Gamma(v)$, let N_i be the set of nodes which are descendants of i (including i itself) in the subtree of G rooted at v. For a node $u \in N_i$,

$$\phi_{uv}(x) = \phi_{uv} x_i, \tag{4}$$

since potentials of all nodes in N_i depend on only link (v, i) and the other links of v do not affect these potentials. This is because if v cuts link (v, i), all nodes in N_i are disconnected from the other nodes in G.

Therefore (3) can be rewritten:

$$\pi_v(x) = \alpha \frac{\sum_{i \in \Gamma(v)} \sum_{u \in N_i} q_u \phi_{uv} x_i}{1 - (1 - \alpha) \dfrac{\sum_{i \in \Gamma(v)} \phi_{iv} x_i}{\mathbf{1}^T x}}. \tag{5}$$

Note that the potential matrix Φ on G can be computed in $O(n^2)$ time, by using Gaussian elimination methods for each column vector $(\Phi)_v$ defined by equation (1). Since G is a tree we can apply elimination steps in post-order, where we consider v to be the root of G. There are at most n forward eliminations and backward substitutions because every node except v has only one parent. Therefore it costs $O(n^2)$ time to compute Φ.

Let $a_i = \alpha \sum_{u \in N_i} q_u \phi_{uv}$ and let $b_i = 1 - (1-\alpha)\phi_{iv}$ for $i \in \Gamma(v)$. Consider the fractional integer programming problem,

$$P : \text{maximize} \quad \pi_v(x) = \mathbf{1}^T x \, \frac{a^T x}{b^T x}, \quad x \in \{0,1\}^n,$$

where $a_i \geq 0$ and $b_i \geq 0$ for $i \in \Gamma(v)$ are known constants.

In order to solve problem P, we fix the Hamming weight l of x and we solve the following problem for each $l = 1, ..., d_v$:

$$Q : \text{maximize} \quad f(x) = \frac{a^T x}{b^T x} \qquad \text{subject to} \quad \mathbf{1}^T x = l.$$

Problem Q can be solved directly by Megiddo's method in $O(n^2 \log^2 n)$ time [3], and it can be also solved by Newton's Method in time $O(n^2 \log n)$ [4]. However we are able to specialize Megiddo's method for our problem to obtain an $O(n^2)$ time algorithm. Our approach initially follows the technique describe in [3].

Since $\max f(x) = \max_l l \; g(x)$, we can solve problem P by solving problem Q for each $l = 1, ..., d_v$. Consider the following maximization problem for some fixed δ.

$$R : \text{maximize} \quad g(x) = (a - b\delta)^T x \qquad \text{subject to} \quad \mathbf{1}^T x = l.$$

Let $c_i = a_i - b_i\delta, i \in \Gamma(v)$ and let $S(\delta)$ be the decreasing sequence of indices ordered by the values of c_i. Problem R is easily solved by choosing the first l indices in $S(\delta)$. Let $h(\delta)$ be the optimal value of problem R for a given δ, that is, $h(\delta) = \max\{g(x) : \mathbf{1}^T x = l\}$. When $h(\delta) = 0$, then δ is equal to δ^* which is the optimal value of problem Q. On the other hand, if $h(\delta) > 0$ then $\delta < \delta^*$, and if $h(\delta) < 0$ then $\delta > \delta^*$, ie. a root of h (see Fig. 2). The task in solving problem Q is to find the value δ^* for which $h(\delta^*) = 0$ by some tests on δ. The key point is how many values of δ have to be tested for finding δ^*. Since the optimal solution for R can change only when the order of $S(\delta)$ changes, we only have to test δ at intersection values of lines $\{c_i = a_i - b_i\delta : i \in \Gamma(v)\}$. Using some results from computational geometry, we are able to do this efficiently. First we compute the line arrangement of lines $\{c_i = a_i - b_i\delta : i \in \Gamma(v)\}$. Namely we define the planar graph H which is formed by subdivision of the plane induced by these lines. Then we look all edges in H as directed according to the positive direction of δ.

For each l we test values of δ at the change point of the lth entry in $S(\delta)$, which are the nodes in H lying on the lth layer of the arrangement. An example with $l = 3$ is shown in Fig. 3.

We summarize the algorithm for solving P. First compute constants a_i, b_i, for $i \in \Gamma(v)$, and line arrangement of lines $\{c_i = a_i - b_i\delta : i \in \Gamma(v)\}$. Compute

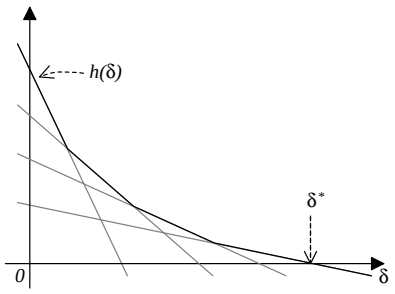

Fig. 2. Solving $h(\delta) = 0$

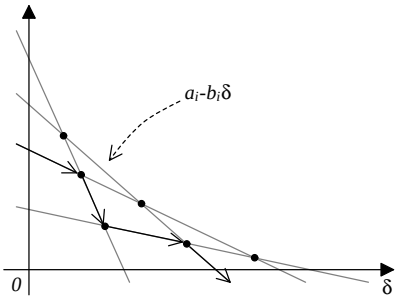

Fig. 3. Path for $l = 3$ on H

$S(0)$ by sorting the values of a_i. For $l = 1, ..., d_v$, do following steps: let $x_i = 1$ if i is within lth entry of $S(0)$, and otherwise $x_i = 0$. Compute $A = a^T x$ and $B = b^T x$. From the starting edge, which is lth edge from the top, follow the l-th layer as follows: When we follow the edge on the line $c_i = a_i - b_i\delta$, and visit the node which is intersection of $c_i = a_i - b_i\delta$ and $c_j = a_j - b_j\delta$, let $A \leftarrow A - a_i + a_j$ and $B \leftarrow B - b_i + b_j$. If $A - B\delta < 0$ then output x as the solution, otherwise let $x_i \leftarrow 0$ and $x_j \leftarrow 1$, and go to next node by following the edge on the line $c_j = a_j - b_j\delta$.

Finally, we analyze the running time of the algorithm. Computing the line arrangement takes $O(d_v^2)$ time, which is done by the incremental method or topological sort algorithms (see Edelsbrunner [1]). Since the number of nodes in H is $O(d_v^2)$ and each of them is visited twice, we can find δ^* for each l in at most $O(d_v^2)$ time. Therefore this algorithm solves problem P in $O(d_v^2)$ time. Note that if v is not in best response the solution to P gives an improving strategy for v.

As we have seen we can test whether a node v is in best response in $O(d_v^2)$ time. Because G is a tree we have $\sum_{v \in V} d_v = 2(n-1)$, and we have $\sum_{v \in V} d_v^2 < \left(\sum_{v \in V} d_v\right)^2 = 4(n-1)^2$. Therefore we can test whether all nodes are in best response in $O(n^2)$ time. This concludes the the proof of Theorem 1.

4 General Graphs

In this section we give a parametric algorithm for general connected graphs based on a parameter $k = k(G)$ defined as follows. If $G = (V, E)$ is a tree we set $k = 1$ else k is the maximum vertex degree in any biconnected component of G. Note that $k(G)$ can be computed in $O(|E|)$ time by decomposing G into its biconnected components and by finding the maximum vertex degree in every such component. Note that graphs can have a large maximum vertex degree but small parameter k. This would occur whenever the large degree vertices were cut vertices. In a network setting the biconnected components could represent small groups of well connected web pages with relatively few links per page. These groups would be linked together by a few pages containing many more links. We prove the following theorem.

Theorem 2. *Given a graph G with $k = k(G)$, jump probability α and distribution q, in $O(2^k n^4)$ time we can determine if this is a Nash Equilibrium and if not give an improving strategy for at least one player.*

The remainder of this section is devoted to the proof of this theorem.

Let v be a node in G and let $\{C_1, C_2, ..., C_d\}$ be the set of connected components in the subgraph that is induced by deletion of v from G. It follows from the definition of $k = k(G)$ that v has at most k links to C_i for $i = 1, ..., d$. Let $U_i = \{u : u \in C_i, u \in \Gamma(v)\}$ indicate the set of v's neighbours in C_i, for $i = 1, ..., d$. We have $|U_i| \leq k$ by the definition of k. Consider any strategy for v, as described in Section 2.3 and let x be the 0/1 vector of length $d_v = |\Gamma(v)|$ that represents it. We write $x = (x^1, x^2, ..., x^d)$ as the concatenation of the the 0/1 vectors x^i representing the strategy restricted to the component C_i, $1 = 1, ..., d$ Then, for $u \in V$, if $u \in C_i$ for $i = 1, ..., d$, the potential of u is written as follows:

$$\phi_{uv}(x) = \sum_{S \subseteq U_i} \phi_{uv}^S \prod_{s \in S} x_s^i \prod_{s \in U_v - S} (1 - x_s^i) \tag{6}$$

where ϕ_{uv}^S are the potentials from u for the subgraph of G formed by deleting edges $vu, u \in \Gamma(v) - S$. This is because potentials to v from all nodes in C_i depend only on links to U_i and never depend on other links. To compute each column vector $(\Phi)_v^S$ for $S \subseteq U_i$ for $i = 1, ..., d$, we solve the linear systems defined on (1) by Gaussian elimination method in $O(2^k n^3)$ time.

We have the formula for PageRank of v as

$$\pi_v(x) = \mathbf{1}^T x \frac{\sum_i \sum_{S \subseteq U_i} a_S \prod_{s \in S} x_s^i}{\sum_i \sum_{S \subseteq U_i} b_S \prod_{s \in S} x_s^i}. \tag{7}$$

where a_S and b_S be constants such that

$$a_S = \sum_{u \in C_i : S \subseteq C_i} \alpha q_u \sum_{T \subseteq S} (-1)^{|S|-|T|} \phi_{uv}^T$$

$$b_S = \begin{cases} 1 - (1-\alpha)\phi_{uv}^S & S = \{u\} \\ -(1-\alpha) \sum_{u \in S} \sum_{T \subseteq S} (-1)^{|S|-|T|} \phi_{uv}^T & \text{otherwise.} \end{cases}$$

Note that $a_S \geq 0$ for all S, and the denominator is always positive.

We can determine whether v maximizes its PageRank by solving the following fractional integer programming problem:

$$P : \text{maximize} \quad \pi_v(x), \quad x_s^i \in \{0, 1\} \quad \text{for } s \in U_i, \ i = 0, ..., d.$$

The method we use to solve P is the similar to that used in Section 3. We fix the Hamming weight of x, $\mathbf{1}^T x = l$, and consider the following linear fractional integer programming problem:

$$Q : \text{maximize} \quad f(x) = \frac{\sum_i \sum_{S \subseteq U_i} a_S \prod_{s \in S} x_s^i}{\sum_i \sum_{S \subseteq U_i} b_S \prod_{s \in S} x_s^i} \quad \text{subject to} \quad \mathbf{1}^T x = l.$$

Since $\max f(x) = \max_l lg(x)$, we can solve problem P by solving problem Q for each $l = l, ..., d_v$. Let δ be a positive real number, and let $c_S = a_S - b_S\delta$ for all $S \subseteq U_i$ for $i = 1, ..., d$.

$$R : \text{maximize} \quad g(x) = \sum_i \sum_{S \subseteq U_i} c_S \prod_{s \in S} x_s^i \quad \text{subject to} \quad \mathbf{1}^T x = l.$$

Let $h(\delta)$ be the optimal value of problem R for some δ, i.e., $h(\delta) = \max_x\{g(x) : \mathbf{1}^T x = l\}$. To solve problem Q, we can repeatedly solve R for various δ to find the root of $h(\delta)$. However, the original problem is to determine whether v maximizes its value. Namely, we do not have to maximize $f(x)$ subject to $\mathbf{1}^T x = l$, but only determine $l\delta^* > d_v f(\mathbf{1})$. So, the task is to solve R for $\delta = \frac{d_v}{l}\delta'$, where $\delta' = f(\mathbf{1})$, and only check if $h(\delta) < 0$ or not.

For converting $g(x)$ in problem R to linear function, let $y_{i,t}$ denote the 0/1 variable and let $e_{i,t}$ be the constant for $1 \le i \le d$ and $1 \le t \le |U_i|$ such that,

$$y_{i,t} = \begin{cases} 1 & \sum_{s \in U_i} x_s^i = t \\ 0 & \text{otherwise} \end{cases}, \qquad e_{i,t} = \max_S \left\{ \sum_{T \subseteq S} c_T : S \subseteq U_i, |S| = t \right\}.$$

$y_{i,t}$ indicates whether the number of edges used in the strategy x going from v to U_i is equal to t or not. If $y_{i,t} = 1$, let $S \subseteq U_i$ be the t edges chosen. Then we consider that $g(x)$ earns $\sum_{T \subseteq S} c_T$, with cost t. In the optimal strategy x, if $y_{i,t} = 1$ for some i, t then it must be that S maximizes $\sum_{T \subseteq S} c_T$, that is $e_{i,t}$, as any other assignment can be improved to it. We then have the equivalent integer linear program to R:

$$R' : \text{maximize} \quad \sum_{i=1}^{d} \sum_{t=1}^{|U_i|} e_{i,t}\, y_{i,t} \quad \text{subject to} \quad \sum_{i=1}^{d} \sum_{t=1}^{|U_i|} t\, y_{i,t} = l \tag{8}$$

$$\sum_{t=1}^{|U_i|} y_{i,t} \le 1 \quad \text{for } i = 1, ..., d. \tag{9}$$

Problem R' is similar to a knapsack problem where each item has positive integer weight t and value $e_{i,t}$, and the total weight must be l. The only difference is the constraint (9).

Dynamic programming can be used to solve R' in $O(ld_v)$ time. Let $w(i, t)$, for $1 \le i \le d$ and $1 \le t \le l$, denote the maximum value which has total weight t and uses only the i first items. Let $e_{i,0} = 0$, then,

$$w(i, t) = \max_{0 \le s \le |U_i|} \{w(i - 1, t - s) + e_{i,s}\}.$$

For each $i = 1, ..., d$, we can compute $w(i, t)$ for $1 \le t \le l$ in $l|U_i|$ time. Since $d_v = \sum_{1 \le i \le d} |U_i|$, the computation time for solving R' is $O(ld_v)$.

In order to determine whether v is in best response, we test $g(x)$ for $\delta = \frac{d_v}{1}\delta', \frac{d_v}{2}\delta', ..., \frac{d_v}{d_v}\delta'$. Since $d_v \le kd$ the running time per vertex is $O(2^k n^3 + k^2 d^3 + 2^k d)$. Moreover $\sum_{v \in V} d_v < 2kn$ from the assumption. Therefore, in order to determine whether every node is in best response, it takes $O(2^k n^4 + k^2 n^3 + 2^k n) = O(2^k n^4)$ time. This completes the proof of Theorem 2.

5 Conclusion

We have constructed a new undirected model for PageRank games and studied the problem of verifying Nash equilibria for these games. Our first result is an algorithm for trees that runs in time $O(n^2)$. Our second result is an extension of this method for general graphs. We defined a parameter k to be the maximum vertex degree in any biconnected component of the graph and gave a $O(2^k n^4)$ time algorithm for testing if it is a Nash equilibrium.

Acknowledgments

We thank Yuichi Yoshida and Junichi Teruyama for many helpful discussions.

References

1. H. Edelsbrunner. Algorithms in Combinatorial Geometry. *EATCS Monographs in Theoretical Computer Science 10*, Springer-Verlag, 1987.
2. J. Hopcroft, D. Sheldon. Network reputation games. *Manuscript*, eCommons@Cornell, `http://hdl.handle.net/1813/11579`, 2008.
3. N. Megiddo. Combinatorial Optimization with rational objective functions. *Math. of Oper. Res.*, 4:414-424, 1979.
4. T. Radzik. Newton's method for fractional combinatorial optimization. Proceedings, 33rd Annual Symposium on Foundations of Computer Science, 659-669, 1992.
5. K. Avrachenkov and N. Litvak. The effect of new links on Google PageRank. *Stochastic Models*, 319-331, 2006.
6. B. C. Csaji, R. M. Jungers, and V. D. Blondel. PageRank optimization in polynomial time by stochastic shortest path reformulation. Proceedings of the 21st international conference on Algorithmic Learning Theory, 130-140, 2010.
7. C. De Kerchove, L. Ninove, and P. Van Dooren. Maximizing pagerank via outlinks. *Linear Algebra and its Applications*, 429:1254-1276, 2008.
8. M. Olsen. The computational complexity of link building. Proceedings of the 14th annual international conference on Computing and Combinatorics, 119-129, 2008.
9. M. Olsen. Maximizing pagerank with new backlinks. Proceedings of the 7th international conference on Algorithms and complexity, 37-48, 2010.
10. M. Olsen, et. al. A constant-factor approximation algorithm for the link building problem. Proceedings of the 4th international conference on Combinatorial optimization and applications, Volume Part II, 87-96, 2010.
11. W. Chen, S. H. Teng, Y. Wang, Y. Zhou. On the α-Sensitivity of Nash equilibria in pagerank-based network reputation games. Proceedings of the 3rd International Workshop on Frontiers in Algorithmics, 63-73, 2009.