Internet Mathematics Vol. 5, No. 4: 323–342

Complexity of Pure Nash Equilibria in Player-Specific Network Congestion Games

Heiner Ackermann and Alexander Skopalik

Abstract. Network congestion games with player-specific delay functions do not possess pure Nash equilibria in general. We therefore address the computational complexity of the corresponding decision problem and prove that it is NP-complete to decide whether a pure Nash equilibrium exists. This result is true for games with directed edges as well as for networks with undirected edges, and still holds for games with two players only. In contrast to games with networks of arbitrary size, we present a polynomial-time algorithm deciding whether there exists a Nash equilibrium in games with networks of constant size.

Additionally, we introduce a family of player-specific network congestion games that are guaranteed to possess equilibria. In these games players have identical delay functions. However, each player may use only a certain subset of the edges. For this class of games we prove that finding a pure Nash equilibrium is PLS-complete. Again, this result is true for networks with directed edges as well as for networks with undirected edges, and still holds for games with three players only. In games with networks of constant size, however, we prove that pure Nash equilibria can be computed in polynomial time.

1. Introduction

Allocating a feasible and optimal subset of a given set of resources is a fundamental problem in many applications. For example, in networks users want to choose paths, corresponding to subsets of the edges, along which they can stream data as fast as possible. In the presence of several service providers users seek the best ones. Many of these applications have in common that users are driven by selfish, economic interests rather than by social ones. In this paper we are interested in applications in which there is no central authority coordinating the assignment of users to resources and we apply game-theoretic tools in order to model the users' behavior. To be precise, we consider an extension of Rosenthal's notion of network congestion games [Rosenthal 73].

In these games a finite number of players share a network and each of them wants to select a path with minimum delay (or cost) that connects individual pairs of nodes. The delay of a path equals the sum of delays of the edges in that path, and the delay of an edge depends on the number of players currently using that edge. In recent years, network congestion games have been considered with respect to different questions, such as the existence and computational complexity of Nash equilibria,[1] the price of anarchy, network design problems, and mechanism design problems. For an introduction and formal definition of these problems we refer the reader to [Nisan et al. 07] and the references therein.

In this paper we are interested in *player-specific network congestion games*. In such games each player is equipped with a set of player-specific delay functions. This is in contrast to Rosenthal's model, in which all players allocating an edge observe the same delay. Player-specific network congestion games naturally arise when different players have different preferences regarding the edges of the network. Some players might prefer to use freeways, while others might prefer to use scenic roads. It is well known that player-specific network congestion games do not necessarily possess Nash equilibria [Milchtaich 06]. We therefore investigate the computational complexity of deciding whether such a game possesses a Nash equilibrium. We prove that this problem is NP-complete in directed and undirected networks even if there are only two players. In contrast to games with networks of arbitrary size, we present a polynomial-time algorithm deciding whether there exists a Nash equilibrium in games with networks of constant size.

In order to bypass the limitations of general player-specific congestion games, we introduce a family of games for which the existence of a Nash equilibrium is guaranteed by Rosenthal's potential function [Rosenthal 73]. We assume that all players sharing an edge observe the same delay. However, each player may use only a certain subset of the edges. Such games naturally arise when drivers are prohibited from using certain roads, e.g., trucks may be forbidden to use narrow roads, while slow-moving vehicles may be excluded from the freeway. These games—in the following called *restricted network congestion games*—are closely related to standard network congestion games in which players compute their delays with respect to common delay functions and in which each player

[1]In this paper, the term *Nash equilibrium* always refers to a pure Nash equilibrium.

may use every edge. These standard games are considered in [Fabrikant et al. 04], and its authors show that computing an equilibrium is PLS-complete, that is, computing a Nash equilibrium is "as hard to compute as any object whose existence is guaranteed by a potential function" [Fabrikant et al. 04]. Thus, computing a Nash equilibrium of a restricted network congestion game is PLS-complete, too.

However, the previously mentioned proof requires an arbitrary number of players and resources. In this paper we consider games in which one of these two parameters is constant. In the case of a constant number of players, we prove that computing a Nash equilibrium remains PLS-complete, whereas it is polynomial-time solvable in the case of constant number of resources. The latter result follows easily by a potential function argument and applies to every congestion game with common delay functions and with a constant number of resources. Again, both results apply to networks with directed and undirected edges. Unfortunately, we failed to prove PLS-completeness for computing Nash equilibria in standard network congestion games with a constant number of players. This question was our primary motivation and remains a challenging open problem.

1.1. Definitions and Notation

In the following, we introduce the notation used throughout this paper.

1.1.1. Player-Specific Network Congestion Games.

A player-specific network congestion game Γ consists of four components:

- a network $G = (V, E)$ with m directed or undirected edges;
- a finite set $\mathcal{N} = \{1, \dots n\}$ of n players;
- for every player i a source–sink pair $(s_i, t_i) \in V \times V$;
- for every player i and every edge $e \in E$ a nondecreasing delay function $d_i^e \colon \mathbb{N} \to \mathbb{N}$.

The *strategy space* of player i is the set of paths connecting source s_i with target t_i. We denote by $S = (P_1, \dots, P_n)$ a *state of the game* in which player i chooses path P_i. Furthermore, we denote by $n_e(S) = |\{i \in \mathcal{N} \mid e \in P_i\}|$ the congestion on edge e in state S, that is, $n_e(S)$ equals the number of players sharing edge e in state S. Players act selfishly and choose paths with minimum delay given fixed choices of the other players. The delay of player i in state S equals $\sum_{e \in P_i} d_i^e(n_e(S))$. Finally, we call a state S a *Nash equilibrium* if no player has an incentive to change her strategy.

We also consider network congestion games with common delay functions, i.e., all players sharing an edge observe the same delay. However, we assume that each player is restricted to a certain subset of the edges. We call such a game a *restricted network congestion game*. Such a game can easily be interpreted as a player-specific game by defining player-specific delay functions in the following way. If a player is allowed to use an edge, her delay function equals the common one, while if a player is not allowed to use an edge, she observes delay ∞ for every congestion on that edge.

1.1.2. The Complexity Class PLS. A local search problem Π is given by its set of instances $\mathcal{I}_\Pi$. For every instance $I \in \mathcal{I}_\Pi$, we are given a finite set of feasible solutions $\mathcal{F}(I) \subseteq \{0,1\}^*$, an objective function $c : \mathcal{F}(I) \to \mathbb{N}$, and for every feasible solution $S \in \mathcal{F}(I)$ a neighborhood $\mathcal{N}(S, I) \subseteq \mathcal{F}(I)$. Given an instance I of a local search problem, we seek a *locally optimal solution* S^*, i.e., a solution that does not have a strictly better neighbor with respect to the objective function c.

A local search problem Π belongs to PLS if the following polynomial-time algorithms exist: an algorithm A that computes for every instance I of Π an initial feasible solution $S_0 \in \mathcal{F}(I)$, an algorithm B that computes for every instance I of Π and every feasible solution $S \in \mathcal{F}(I)$ the objective value $c(S)$, and an algorithm C that determines for every instance I of Π and every feasible solution $S \in \mathcal{F}(I)$ whether S is locally optimal and finds a better solution in the neighborhood of S in the latter case.

Johnson, Papadimitriou, and Yannakakis introduced the notion of a *PLS-reduction*. A problem Π_1 in PLS is PLS-reducible to a problem Π_2 in PLS if there exist polynomial-time computable functions f and g such that f maps instances I of Π_1 to instances $f(I)$ of Π_2; g maps pairs (S_2, I), where S_2 denotes a solution of $f(I)$ to solutions S_1 of I; and for all instances I of Π_1, if S_2 is a local optimum of instance $f(I)$, then $g(S_2, I)$ is a local optimum of I. A local search problem Π in PLS is *PLS-complete* if every problem in PLS is PLS-reducible to Π.

1.2. Related Work

Most closely related to our NP-completeness results are the results in [Milchtaich 96, Milchtaich 06, Dunkel and Schulz 06]. In [Milchtaich 96], player-specific network congestion games on parallel links are introduced, and it is shown that these games are not potential games, but that nonetheless, Nash equilibria always exist and can be computed efficiently if the player-specific delay functions are nondecreasing.

In [Milchtaich 06], the author presents some network topologies such that every player-specific network congestion game on such a topology possesses an equilibrium without further assumption on the delay functions except monotonicity.

In [Dunkel and Schulz 06], the authors consider the computational complexity of deciding whether a weighted network congestion game possesses a Nash equilibrium. In such games, players sharing an edge observe the same delay. However, the congestion on an edge depends on the sum of the players' individual weights.

The authors prove that deciding whether an equilibrium exists is NP-complete. Furthermore, they consider network congestion games on parallel links with weighted players and player-specific delay functions and prove that deciding whether such a game possesses a Nash equilibrium is NP-complete, too.

Others papers that are loosely related to our NP-completeness results are the following: [Chakrabarty et al. 05] considers player-specific network congestion games on (a constant number of) parallel links from a global optimization perspective, and investigates whether one can compute social optimal states of such games efficiently. [Gairing et al. 06] considers network congestion games on parallel links with player-specific linear latency functions without offsets with respect to the price of anarchy. [Ackermann et al. 06b] extends Milchtaich's results [Milchtaich 96] to player-specific matroid congestion games and shows that the matroid property is the maximal property on the strategy spaces guaranteeing the existence of equilibria in player-specific congestion games. Finally, [Anshelevich et al. 04] and [Meyers 06] consider several problems involving congestion games with a constant number of players.

Most closely related to our PLS-completeness result is [Fabrikant et al. 04]. The authors prove that computing a Nash equilibrium in network congestion games with common delay functions and directed edges is PLS-complete if the players have different source and sink nodes. However, an equilibrium can be computed efficiently if the players have the same source or target node. [Ackermann et al. 06a] significantly simplified their proof and extended it to networks with undirected edges and linear delay functions.

2. General Player-Specific Network Games

In this section, we consider the complexity of deciding whether a general player-specific network congestion game possesses a Nash equilibrium. We prove that this problem is NP-complete even in the case of two players. First, we consider networks of directed edges and present a fairly simple reduction from the DIRECTED-EDGE–DISJOINT-PATH problem. Unfortunately, our reduction cannot be extended to networks with undirected edges, since the UNDIRECTED-EDGE–DISJOINT-PATH problem admits a polynomial-time algorithm in the case of constant number of source–sink pairs [Robertson and Seymour 95]. We therefore present a reduction from 3-SAT in the undirected case. Finally, we consider

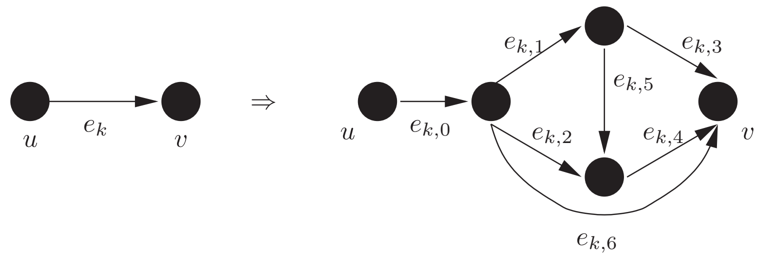

Figure 1. The gadget G_{e_k} by which an edge e_k is replaced.

games with networks of constant size and present a polynomial-time algorithm deciding whether a Nash equilibrium exists.

2.1. Networks with Directed Edges

Theorem 2.1. *It is NP-complete to decide whether a player-specific network congestion game with* directed edges *and two players possesses a Nash equilibrium.*

Proof. Obviously, the decision problem belongs to NP, since one can decide in polynomial time whether a state S of a player-specific network congestion game with directed edges and two players is a Nash equilibrium. In order to prove that the problem is complete, we present a polynomial-time reduction from the DIRECTED-EDGE–DISJOINT-PATH problem with two disjoint source–sink pairs. An instance of this problem consists of a directed graph $G = (V, E)$ and two disjoint node pairs (s_1, t_1) and (s_2, t_2). Given such an instance, we would like to decide whether there exist distinct edge-disjoint paths between the two node pairs. This problem is known to be NP-complete [Fortune et al. 80].

Given an instance $(G, (s_1, t_1), (s_2, t_2))$ of the DIRECTED-EDGE–DISJOINT-PATH problem with two source–sink pairs, we construct a player-specific network congestion game Γ with two players as follows. We substitute every edge $e_k \in E$ by the gadget G_{e_k} presented in Figure 1 in order to obtain the network $G_\Gamma = (V_\Gamma, E_\Gamma)$ on which the game is played. Player $i \in \{1, 2\}$ wants to allocate a path between the nodes s_i and t_i in G_Γ. Observe that this construction ensures a one-to-one correspondence between the paths in G and in G_Γ in the natural way if we ignore the precise subpaths through every gadget. Let M be a sufficiently large number. Then the player-specific delay functions of the edges $e_{k,0}, \dots, e_{k,6}$ are defined as presented in Table 1. Observe that every gadget G_{e_k} implements a subgame that is played by the players if both want to allocate a path connecting u and v. If only one player wants to allocate such a path, then it allocates a player-specific shortest path from u to v. If we choose M sufficiently large such

	$e_{k,0}$		$e_{k,1}$		$e_{k,2}$		$e_{k,3}$		$e_{k,4}$		$e_{k,5}$		$e_{k,6}$	
cong.	1	2	1	2	1	2	1	2	1	2	1	2	1	2
player 1	0	M	1	1	M	M	M	M	1	20	1	1	5	5
player 2	0	M	1	20	1	1	5	5	5	5	M	M	M	M

Table 1. The player-specific delay functions of the edges $e_{k,0}, \ldots, e_{k,6}$.

that the first player never allocates one of the edges $e_{k,2}$ and $e_{k,3}$ and such that the second player never allocates one of the edges $e_{k,5}$ or $e_{k,6}$, then the delays of these shortest paths are 3 and 5. Suppose now that the two players play such a subgame. In this case, it is not difficult to verify that the subgame possesses no Nash equilibrium.

Suppose now that we are given two node-disjoint paths P_1 and P_2 in G connecting s_1 and t_1, and s_2 and t_2. We map these paths to paths in G_Γ in the natural way, and choose player-specific shortest paths through every gadget. Let $n(P_i)$ be the number of edges on the path P_i. Thus, player 1 has delay $3 \cdot n(P_1)$, and player 2 has delay $5 \cdot n(P_2)$. If one of the players had an incentive to change her strategy, then she will choose only a path in which she shares no gadget with the other player, since otherwise, her delay would increase to at least M. This is true because in this case, the players would share at least one edge $e_{k,0}$. This also implies that the delay of the other player does not increase due to the strategy change of the first player. Observe that this holds for any further best response. Thus, the players converge to an equilibrium after $O(n)$ best responses, since the delay of a player decreases by at least the cost of the shortest path through a gadget.

Suppose now that we are given a Nash equilibrium of Γ. In this case the players do not share a gadget, for otherwise the state is not a Nash equilibrium. Thus, we can easily construct edge-disjoint paths in G connecting (s_1, t_1) and (s_2, t_2). □

2.2. Networks with Undirected Edges

Theorem 2.2. *It is NP-complete to decide whether a player-specific network congestion game with* undirected edges *and two players possesses a Nash equilibrium.*

Proof. Obviously, the decision problem belongs to NP, since one can decide in polynomial time whether a state S of a player-specific network congestion game with undirected edges and two players is a Nash equilibrium. In order to prove that the problem is complete, we present a polynomial-time reduction from 3-SAT. Let φ be a 3-SAT formula with n variables $x_1, \ldots, x_n$ and m clauses $C_1, \ldots, C_m$.

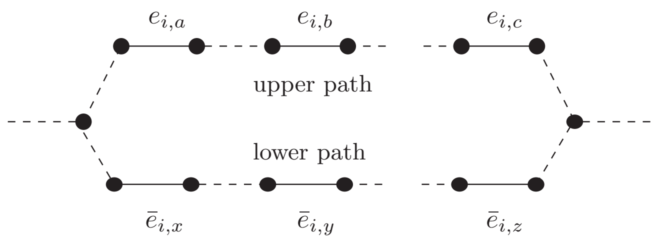

Figure 2. The bit gadget of variable x_i.

We assume that every clause in φ contains exactly three distinct literals. Given φ, we construct a player-specific network congestion game with undirected edges and two players as follows. For the sake of simplicity we refer to the players as bit player and clause player. Our construction satisfies the following three properties:

1. The bit player can choose between 2^n different paths each determining a unique assignment of the n variables.

2. The clause player can check for every clause separately whether there exists a variable satisfying that clause.

3. If and only if there exists an unsatisfied clause, then both players are forced to choose paths through a special gadget. This gadget implements a subgame that does not possess a Nash equilibrium if both players participate.

In the following we define three different kinds of gadgets called variable, clause, and subgame gadgets and describe how they are connected. The gadgets consist of bold, dashed, and dotted edges. Bold edges appear in all three kinds of gadgets, whereas dotted edges do not appear in the variable gadgets, and dashed edges do not appear in the clause gadgets. The player-specific delay functions will be chosen in such a way that the bit player never chooses one of the dotted edges and that the clause player never chooses one of the dashed edges.

For every variable x_i there is a variable gadget G_{x_i} as depicted in Figure 2. Without loss of generality let $\{C_1, \ldots, C_k\}$ be the set of clauses in which x_i appears as a positive literal. For every such clause C_j there is a bold edge $e_{i,j}$ on the upper path in G_{x_i}. Additionally, let $\{C_{k+1}, \ldots, C_l\}$ be the set of clauses in which x_i appears as a negative literal. For every such clause C_j there is a bold edge $\bar{e}_{i,j}$ on the lower path in G_{x_i}. Bold edges are connected by dashed edges as shown in the figure. The ordering of the bold edges can be chosen arbitrarily. Additionally, the gadgets G_{x_i} are arranged one after the other starting with gadget G_{x_1} and finishing with gadget G_{x_n}. They are connected by dashed edges.

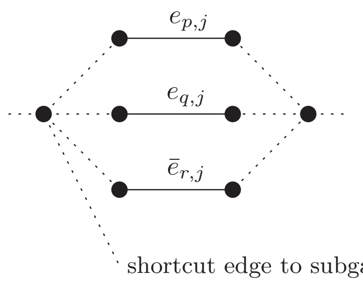

Figure 3. The clause gadget of clause $C_j = (x_p, x_q, \bar{x}_r)$.

For every clause C_j there is a clause gadget G_{C_j} as depicted in Figure 3. For every variable x_i that appears as a positive literal in C_j there is a bold edge $e_{i,j}$. For every variable x_i that appears as a negative literal in C_j there is a bold edge $\bar{e}_{i,j}$. Observe that edges $e_{i,j}$ and $\bar{e}_{i,j}$ coincide with the corresponding edges in the variable gadgets G_{x_i}. Bold edges are connected by dotted edges as shown in the figure. Additionally, there is a shortcut edge from the leftmost node from every clause gadget to the subgame gadget. The gadgets G_{C_j} are arranged one after the other starting with gadget G_{C_1} and finishing with gadget G_{C_m}. They are connected by dotted edges.

The subgame gadget is depicted in Figure 4. Basically it consists of three bold edges e_1, e_2, e_3 arranged as a triangle. The dotted shortcut edges from the clause gadgets are connected to vertex v_2. Additionally, there is a dotted edge from the rightmost node of the clause gadget G_{C_m} to node v_1, and there is a dashed edge from the rightmost node of the subgame gadget G_{x_n} to node v_1.

It remains to define the source and target nodes of the players and the delay functions of the edges. The bit player wants to allocate a path from the leftmost node of the variable gadget G_{x_1} to the node v_3 of the subgame gadget. The clause player, however, wants to allocate a path connecting the leftmost node from the clause gadget G_{C_1} to the node v_1 of the subgame gadget.

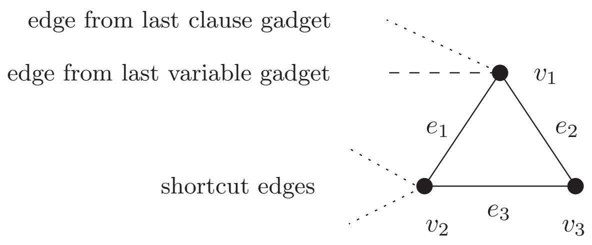

Figure 4. The subgame gadget.

	e_1		e_2		e_3	
congestion	1	2	1	2	1	2
clause player	20	21	12	15	4	10
bit player	1	21	5	15	3	10

Table 2. The player-specific delay functions of the edges e_1, e_2, e_3.

The player-specific delay functions are defined as follows. Let M be a sufficiently large number. The bit player always has delay 0 on bold and dashed edges, and delay M on every dotted edge. On bold edges the clause player has delay 0 if she does not share it with the bit player. Otherwise, it has delay M. On dotted edges the clause player always has delay 0. The player-specific delay functions of the edges $e_1, \dots, e_3$ are depicted in Table 2. If we choose M sufficiently large, then the bit player never allocates a dotted edge, and the clause player never allocates a dashed edge. For simplicity of presentation, we assume that both players always allocate cycle-free paths as best responses, that is, they never choose paths visiting a node twice. One can easily enforce this by a slight modification of the delay of the dashed and dotted edges.

Suppose now that there exists a satisfying assignment $\bar{x}$ of the given 3-sat formula. In this case, we can construct a Nash equilibrium as follows. If $x_i = 0$, then the bit player chooses the upper path in gadget G_{x_i}. Otherwise, she chooses the lower path. Intuitively, she chooses a path that corresponds to the negation of $\bar{x}$. Additionally, she chooses the player-specific shortest path with respect to congestion 1 in the subgame gadget connecting v_1 and v_3. This path is simply the edge e_3. In this case, the bit player has delay 3. Observe that this is the globally shortest path of the bit player. The clause player chooses a path through every clause gadget along which she does not share a bold edge with the bit player. This is possible, since $\bar{x}$ is a satisfying assignment and since the bit player chooses a path that corresponds to the negation of $\bar{x}$. In this case, the bit player enters the subgame gadget at its target node v_1. Observe that this path has delay 0, which is best possible. We conclude that we can construct a Nash equilibrium if we are given a satisfying assignment, since we can assign both players to globally shortest paths.

Suppose now that no satisfying assignment exists. In this case, there always exists an unsatisfied clause C_j, and thus the clause player cannot choose a path through that gadget along which both players do not share a bold edge. Due to the choice of M, the bit player always switches to the shortcut edge of G_{C_j} as best response and enters the subgame gadget at node v_2. Since the bit player always enters the subgame gadget at v_1, the players are forced to play the subgame defined by the subgame gadget. Observe now that the subgame gadget possesses no Nash equilibrium if players enter the gadget as described above. We conclude that no Nash equilibrium exists if no satisfying assignment exists. □

2.3. Networks with a Constant Number of Edges

Theorem 2.3. *One can decide in polynomial time whether a player-specific network congestion game Γ with a constant number of (un)directed edges possesses a Nash equilibrium.*

Proof. The algorithm we present generalizes a technique introduced in [Chakrabarty et al. 05] in order to compute a social optimal state of a player-specific network congestion game with a constant number of parallel links. Let $\mathcal{P} = \{P_1, \ldots, P_l\}$ be the set of paths in the network. Note that $l \leq 2^m$ is constant. Given a state S of Γ we denote by $\bar{c}(S)$ the congestion vector $(c_1(S), \ldots, c_l(S))$, where $c_i(S)$ equals the number of players choosing path P_i in S. Observe that there are at most $n^l \leq n^{2^m}$ such vectors. In the following, we describe how to decide whether there exists an equilibrium S of Γ such that $\bar{c}(S)$ equals a given congestion vector $\bar{c} = (c_1, \ldots, c_l)$.

Given a congestion vector $\bar{c} = (c_1, \ldots, c_l)$ we construct a directed graph $G_{\bar{c}} = (\{s,t\} \cup \mathcal{N} \cup \mathcal{P}, E(\bar{c}))$ with edge capacities as follows. For every player $i \in \mathcal{N}$ there is a vertex u_i that is connected to the vertex s. The capacity of such an edge is 1. For every path $P_j \in \mathcal{P}$ there is a vertex v_j that is connected to the vertex t. The capacity of such an edge is c_j. Furthermore, a vertex u_i is connected to a vertex v_j if the following conditions are satisfied:

> If the congestion on the edges is given by the vector $\bar{c}$, and player i is player P_j, then player i does not prefer a different strategy.

Now we would like to decide whether there exists an s–t flow of capacity n. Observe that such a flow exists if and only if c_j units of flow can flow from c_j different player vertices u_i to path vertices v_j. Thus, if such a flow exists, and if we assign a player to that path to which the unit of flow originating in its vertex flows, we obtain a Nash equilibrium, since the construction ensures that the player is satisfied.

Finally, since there are polynomially many distinct vectors $\bar{c}$, and since the construction and analysis of $G_{\bar{c}}$ can be done in polynomial time, we obtain a polynomial-time algorithm. □

The running time of the algorithm is $O(\text{poly}(2^m) \cdot \text{poly}(n^{2^m}))$. An interesting open problem is to prove that the problem is *fixed parameter tractable*, that is, to develop algorithms with running time $O(\text{poly}(2^m) \cdot \text{poly}(n))$.

3. Restricted Network Congestion Games

In this section, we analyze the complexity of computing Nash equilibria of restricted network congestion games with a constant number of players or resources.

3.1. Networks with Directed Edges

Theorem 3.1. *Computing a Nash equilibrium of a restricted network congestion game with directed edges and k players is PLS-complete for any $k \geq 3$.*

Proof. We prove the theorem by a reduction from the local search problem *positive not-all-equal 2-satisfiability* PosNae2Sat, which is known to be PLS-complete [Schäffer and Yannakakis 91]. Let $x_1, \ldots, x_n$ be Boolean variables. An instance φ of PosNae2Sat consists of a set of m weighted clauses C_j over the variables x_i that contain two positive literals each. We denote by w_j the (nonnegative integer) weight of clause C_j. A clause is satisfied if and only if the two variables it contains have different values. By $\bar{X} = (X_1, \ldots, X_n) \in \{0,1\}^n$ we denote a bit assignment to the variables $x_1, \ldots, x_n$. The weight $w(\bar{X})$ of a bit assignment $\bar{X}$ is defined as the sum of the weights of all satisfied clauses. We denote the maximal weight by $W = \sum_{j=1}^{m} w_j$. By $\bar{X}_{X_i=b}$ we denote the bit vector $(X_1, \ldots, X_{i-1}, b, X_{i+1}, \ldots, X_n)$. A local optimum of φ is a bit assignment $\bar{X}$ whose weight cannot be increased by flipping a single variable x_i, i.e., $w(\bar{X}) \geq w(\bar{X}_{x_i=b})$ for all $1 \leq i \leq n$ and $b \in \{0,1\}$. Therefore, the neighborhood of an assignment is defined as the set of assignments with Hamming distance one.

Given an instance φ, we construct a restricted network congestion game Γ_φ such that one can easily construct a local optimum of φ given a Nash equilibrium of Γ_φ; Γ_φ simulates in parallel two copies of φ, which we call φ_A and φ_B. Furthermore, the game consists of three players, a *bit player* and two *clause players.*

Every path the bit player can choose determines assignments $\bar{X}_A$ and $\bar{X}_B$ for φ_A and φ_B, respectively. The set of paths the bit player can choose from can be divided into two disjoint sets $\mathcal{P}_1$ and $\mathcal{P}_2$. If she chooses a path from $\mathcal{P}_1$, $\bar{X}_A$ is the actual assignment for φ and $\bar{X}_B$ is a (probably better) neighboring assignment. For every path in $\mathcal{P}_2$ it is the other way round. The bit player switches between paths in $\mathcal{P}_1$ and $\mathcal{P}_2$ as long as she can switch to a better neighboring assignment.

The paths of the clause players lead through $2m$ gadgets. For both copies of φ there is one gadget for every clause. The two clause players simulate a clause by choosing from four paths through the corresponding gadget. For each of the two variables, there are two paths. There is one path for each bit assignment.

We ensure that they always have an incentive to correctly simulate the clauses according to the assignments determined by the bit player. Through every gadget they choose those two paths that correspond to this bit assignment.

To implement this we introduce four levels of delays: large, medium, small, and tiny. If the bit player is on a path in $\mathcal{P}_1$ ($\mathcal{P}_2$) and the clause players do not correctly simulate the clauses of φ_A (φ_B) according to the assignment $\bar{X}_A$ ($\bar{X}_B$), at least one of them has a large delay. If the bit player is on a path in $\mathcal{P}_1$ ($\mathcal{P}_2$) and the clause players simulate φ_A (φ_B) correctly, the bit player observes a medium delay proportional to the weight of the unsatisfied clauses according to the actual assignment $\bar{X}_A$ ($\bar{X}_B$). Furthermore, she has additionally a small delay that is proportional to the weight of the unsatisfied clauses of the neighboring assignment $\bar{X}_B$ ($\bar{X}_A$). If the bit player is on a path in $\mathcal{P}_1$ ($\mathcal{P}_2$) and the clause players do not correctly simulate φ_B (φ_A), they additionally have tiny delays. This ensures that the clause players have an incentive to correctly simulate the clauses and that the bit player has an incentive to choose the best neighboring assignment.

As long as there is a better neighboring assignment, the bit player can change from a path from $\mathcal{P}_1$ ($\mathcal{P}_2$) to a path from $\mathcal{P}_2$ ($\mathcal{P}_1$) by adopting the neighboring assignment as the actual assignment and by choosing a new neighboring assignment.

We are now ready to describe our construction in detail. We present the network of Γ_φ as two subnetworks. One subnetwork contains the edges the bit player is allowed to choose; the other subnetwork contains the edges the two clause players are allowed to choose. The edges that are contained in both networks are called *connection edges*. The connection edges are almost the only edges that cause delay to the players. Almost all other edges have delay 0 regardless of the number of players using it. To further simplify the presentation we merge path segments into sets of edges and use dashed edges to indicate these path segments in Figures 5 and 6. The complete network can be constructed by concatenating the edges from a set in arbitrary order while adding an edge that is not contained in the other subnetwork between every pair of consecutive edges with constant delay 0. Note that the order of these edges along the paths is not important.

The subnetwork of the bit player is depicted in Figure 5. We now define the corresponding sets of edges and the delays on the edges. Let $M \gg \alpha W \gg \alpha \gg \beta W \gg \beta \geq 4m$.

- $\mathcal{P}^A_{x_i=b} := \{u^A_{j,x_i=b}, t^B_{j,x_i=b} |$ for all clauses C_j with $x_i \in C_j\}$. Such a path segment corresponds to the fact that bit x_i equals b in the assignment $\bar{X}_A$. It also corresponds to the fact that $x_i = b$ in the assignment $\bar{X}_B$,

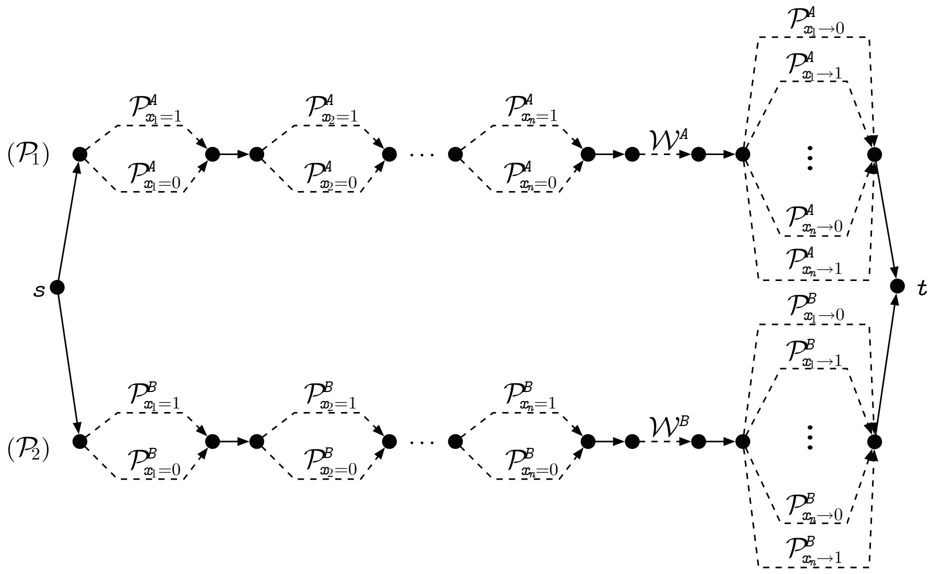

Figure 5. The subnetwork of the bit player. The dashed edges correspond to sets of edges.

unless the bit player chooses to flip this bit (see below). The u-edges have delay 0 for one player and delay M for two or more players. They induce a large delay to clause players if they do not correctly simulate this bit assignment φ_A. The t-edges have delay 0 for one player and delay 1 for two or more players. They induce tiny delays to the clause players if they do not correctly simulate the bit assignment φ_B.

- $\mathcal{W}^A := \{w^A_{j,0}, w^A_{j,1} |$ for all $1 \leq j \leq m\}$. If the clause players correctly simulate φ_A, this path segment induces medium delay proportional to the weight of the unsatisfied clauses of $\bar{X}_A$ to the bit player. The edges $w^A_{j,0}$ and $w^A_{j,1}$ have delay 0 for one or two players and delay αw_j for three players.

- $\mathcal{P}^A_{x_i\to b} := \{w^A_{j,0,x_i\to b}, w^A_{j,1,x_i\to b} |$ for all $1 \leq j \leq m$ with $x_i \notin C_j\} \cup \{t^B_{j,x_i\to b}, w^A_{j,x_i\to b} |$ for all $1 \leq j \leq m$ with $x_i \in C_j\}$. If the bit player chooses such a path segment, then she determines the neighboring assignment $\bar{X}_B$ to be obtained from $\bar{X}_A$ by flipping bit x_i to b. If the clause players correctly simulate φ_A, this path segment induces a small delay proportional to the weight of the unsatisfied clauses of that neighboring assignment. For each $1 \leq j \leq m$ with $x_i \notin C_j$, the edges $w^A_{j,0,x_i\to b}$ and $w^A_{j,1,x_i\to b}$ have delay 0 for at most two players and delay βw_j for three. For each $1 \leq j \leq m$ with

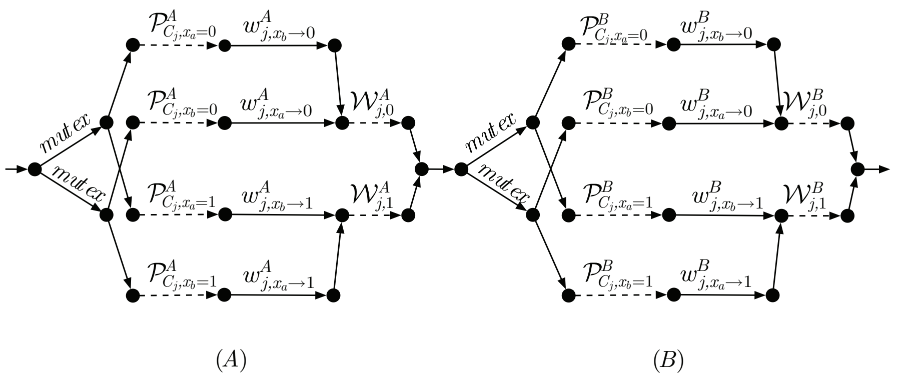

Figure 6. This figure shows an A-gadget and a B-gadget for a clause $C_j = \{x_a, x_b\}$. There are four paths through each gadget. From top to bottom, we denote the paths by $x_b = 0$, $x_a = 0$, $x_b = 1$, and $x_a = 1$. The subnetwork of the two clause players is a concatenation of the A- and B-gadgets for all clauses.

$x_i \in C_j$, the edge $w^A_{j,x_i\to b}$ has delay 0 for one player and delay βw_j for two or more players. The t-edges have delay 0 for one player and delay 2 for two or more players. They induce tiny delays to the clause players if they do not simulate this bit flip in φ_B.

Additionally, there are sets $\mathcal{P}^B_{x_i=b}$, $\mathcal{W}^B$, and $\mathcal{P}^B_{x_i\to b}$ that are defined in the same manner.

The two clause players are symmetric in the sense that they play on the same subnetwork and have the same source and target nodes. Their subnetwork is a concatenation of m A-gadgets and m B-gadgets. Figure 6 depicts such a pair of gadgets. Their source–sink paths lead through all $2m$ gadgets. The edges labeled with *mutex* have delay 0 for one player and delay M^2 for two or more players. The dashed edges correspond to the following sets of connection edges:

- $\mathcal{P}^A_{C_j,x_i=b} := \{u^A_{j,x_i=1-b}, t^A_{j,x_i=1-b}, t^A_{j,x_i\to 1-b}\}$. A clause player using such a path segment simulates the assignment of b to x_i of $\bar{X}_A$ in the clause C_j of φ_A. In the following, we say that she sets $x_i = b$ in this gadget. If this is not a correct simulation and the bit player is on a path from $\mathcal{P}_1$, then a u-edge induces large delay. If this is not a correct simulation and the bit player is on a path from $\mathcal{P}_2$, then a t-edge induces a tiny delay.

- For each $d \in \{0,1\}$, $\mathcal{W}^A_{j,d} := \{w^A_{j,d}\} \cup \{w^A_{j,d,x_i\to b} |$ for all $b \in \{0,1\}$ and $1 \le i \le n$ with $x_i \notin C_j\}$. If and only if both players use the same $\mathcal{W}^A_{j,d}$ path segment they simulate an unsatisfying assignment for C_j. If, additionally, the bit player chooses a path from $\mathcal{P}_1$, the edge $w^A_{j,d}$ has medium delay proportional to w_j. Furthermore, one of the edges $w^A_{j,d,x_i \to b}$ induces a small delay if x_i is not in clause C_j. Note that in the case that x_i is in the clause C_j there are extra edges in the gadget.

- The sets $\mathcal{P}^B_{C_j,x_i=b}$ and $\mathcal{W}^B_{j,d}$ are defined analogously.

We now prove that every Nash equilibrium of Γ_φ corresponds to a locally optimal assignment of φ. Consider a Nash equilibrium of Γ_φ and assume that the bit player chooses a path from the set $\mathcal{P}_1$. Let $\mathcal{P}^A_{x_1=X_1}, \dots, \mathcal{P}^A_{x_n=X_n}$, $\mathcal{W}^A$, and let $\mathcal{P}^A_{x_{i^*} \to b}$ be the path segments she chooses. Then the following properties hold.

Lemma 3.2.

(a) *In every A-gadget for every clause $C_j = \{x_a, x_b\}$ one clause player sets $x_a = X_a$ and the other player sets $x_b = X_b$.*

(b) *In every B-gadget for every clause $C_j = \{x_a, x_b\}$ with $a, b \neq i^*$ one clause player sets $x_a = X_a$ and the other player sets $x_b = X_b$.*

(c) *In every B-gadget for every clause $C_j = \{x_{i^*}, x_c\}$ one clause player sets $x_c = X_c$ and the other player sets $x_{i^*} = b$.*

Proof. Observe that in any gadget for any clause $C_j = \{x_a, x_b\}$ one of the clause players chooses $x_a = 0$ or $x_a = 1$, whereas the other player chooses $x_b = 0$ or $x_b = 1$. Otherwise, both have delay M^2 and thus an incentive to change.

(a) Consider the A-gadget of a clause $C_j = \{x_a, x_b\}$. Due to our assumptions, all edges of the path segment $\mathcal{P}^A_{C_j,x_a=X_a}$ are not used by the bit player and therefore have delay 0 for a single clause player, whereas the edge $u^A_{j,x_a=X_a}$ that is contained in the path segment $\mathcal{P}^A_{C_j,x_a=(1-X_a)}$ is used by the bit player and therefore causes delay M to a clause player. The same is true for the path segments $\mathcal{P}^A_{C_j,x_b=X_b}$ and $\mathcal{P}^A_{C_j,x_b=(1-X_b)}$, respectively. The delay of all other edges in the gadget sums to less than M. Thus, in every Nash equilibrium, one of the clause players chooses $x_a = X_a$ and the other player chooses $x_b = X_b$.

(b) In the B-gadgets all w^B-edges and all edges in the $\mathcal{W}^B$-sets are not used by the bit player and therefore have delay 0. Consider the B-gadget for a clause $C_j = \{x_a, x_b\}$ with $a, b \neq i^*$. All edges of the path segment $\mathcal{P}^B_{C_j,x_a=X_a}$ are not used by the bit player and therefore have delay 0 for a single clause player, whereas the edge $t^B_{j,x_a=X_a}$ that is contained in the path segment $\mathcal{P}^B_{C_j,x_a=(1-X_a)}$ is used by the bit player and therefore has delay 1 for a clause player. The same is true for the path segments $\mathcal{P}^B_{C_j,x_b=X_b}$ and $\mathcal{P}^B_{C_j,x_b=(1-X_b)}$, respectively.

(c) Let $C_j = \{x_{i^*}, x_c\}$ be a clause that contains x_{i^*}. In the B-gadgets of clause C_j, one clause player sets $x_c = X_c$, which has delay 0. The other clause player sets $x_{i^*} = b$, which has delay of at most 1. The path $x_{i^*} = 1 - b$ has delay of at least 2 due to the edge $t^B_{j,x_i \to b}$ that is currently used by the bit player. □

Note that an equivalent version of Lemma 3.2 holds for Nash equilibria in which the bit player chooses a path from the set $\mathcal{P}_2$. The following observation follows directly from Lemma 3.2.

Remark 3.3. In every Nash equilibrium the path segment $\mathcal{W}^A$ has delay $\alpha(W - w(\bar{X}))$ for the bit player. Furthermore, the delay on the path segment $\mathcal{P}^A_{x_{i^*} \to b}$ equals $\beta(W - w(\bar{X}_{x_{i^*}=b}))$ plus an additive term of at most $2m$ for the bit player.

Lemma 3.4. *Every Nash equilibrium of Γ_φ corresponds to a local optimum of φ.*

Proof. For the purpose of contradiction, consider a Nash equilibrium that does not correspond to a local optimum of φ. Let $\mathcal{P}^A_{x_1=X_1}, \ldots, \mathcal{P}^A_{x_n=X_n}$, $\mathcal{W}^A$, and let $\mathcal{P}^A_{x_{i^*} \to b}$ be the path segments used by the bit player. By Remark 3.3, we can conclude that $\bar{X}_{X_{i^*}=b}$ is the best neighboring assignment; otherwise, the path segment $\mathcal{P}^A_{x_{i^*} \to b}$ has more delay than another path segment $\mathcal{P}^A_{x_{i^{**}} \to b^{**}}$ for the bit player. We show that this implies that the bit player can improve her delay by choosing another path. The delays of all edges in the set $\mathcal{W}^A$ sum to $\alpha(W - w(\bar{X}))$. Thus, the bit player has at least this amount of delay.

Now observe that each path segment $\mathcal{P}^B_{x_i=X_1}$ with $i \neq i^*$ has delay 0 for the bit player, since the clause players correctly simulate φ_B with the assignment $\bar{X}_{X_{i^*}=b}$. The path segment $\mathcal{P}^B_{x_{i^*}=b}$ has delay of at most m. The delays of all edges in the set $\mathcal{W}^B$ sum to $\alpha(W - w(\bar{X}_{x_{i^*}=b}))$. The delay of any path $\mathcal{P}^B_{x_{i'} \to b'}$ is at most $\beta W + 2m$. Note that $\beta W + 3m < \alpha$. Thus, the bit player could decrease her delay by changing to such a path. This is a contradiction to the assumption that this is a Nash equilibrium. □

We conclude that every Nash equilibrium of G_φ corresponds to a locally optimal assignment of φ. Obviously, the construction of G_φ and the mapping of an equilibrium to an assignment of φ can be done in polynomial time. □

It is an interesting open problem whether computing Nash equilibria for restricted network congestion games with two players remains PLS-complete. Moreover, it is a challenging open problem to prove any results in standard congestion games with a constant number of players.

3.2. Networks with Undirected Edges

Theorem 3.5. *Computing a Nash equilibrium of a restricted network congestion game with undirected edges and k players is PLS-complete for any $k \geq 3$.*

Proof. We extend the previous proof (Section 3.1) and interpret the network as one with undirected edges. Obviously, any best response path of the bit player in the undirected network is also a best response in the directed case.

The same is true for the clause players. Every best response path of a clause player visits each A- and B-gadget exactly once, because otherwise, one of the *mutex* edges has delay M. Additionally, a path with a gadget that includes an edge in the reverse direction of the directed case cannot be a best response either. This follows from the fact that such a path includes at least three u-edges. One of these has delay M. □

3.3. Games with a Constant Number of Resources

Theorem 3.6. *One can compute a Nash equilibrium of a restricted network congestion game Γ with a constant number of resources in polynomial time.*

Proof. Rosenthal's potential function $\phi(S) = \sum_{e \in E} \sum_{i=0}^{n_e(S)} d_e(i)$ already establishes a pseudo-polynomial-time upper bound on the convergence time of better response dynamics in congestion games [Rosenthal 73]. In the case of better response dynamics, players iteratively deviate to better strategies. Now observe that in games with constant number m of resources, there are at most n^m different congestion vectors $\bar{c} = (n_{e_1}, \ldots, n_{e_m})$. Thus, every state has one out of n^m possible potential values, and therefore the better responses dynamics terminate after at most n^m steps. □

Note that the previous proof applies to every congestion game with constant number of resources. Again, it is an open problem whether the problem is *fixed-parameter tractable.*

Acknowledgments. The authors wish to thank Matthias Englert and Berthold Vöcking for helpful discussions.

This work was supported in part by the EU within the 6th Framework Program under contract 001907 (DELIS) and the German Israeli Foundation (GIF) under contract 877/05.

References

[Ackermann et al. 06a] Heiner Ackermann, Heiko Röglin, and Berthold Vöcking. "On the Impact of Combinatorial Structure on Congestion Games." In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pp. 613–622. Los Alamitos, CA: IEEE Press, 2006.

[Ackermann et al. 06b] Heiner Ackermann, Heiko Röglin, and Berthold Vöcking. "Pure Nash Equilibria in Player-Specific and Weighted Congestion Games." In *Internet and Network Economics: Internet and Network Economics, Second International Workshop, WINE 2006, Patras, Greece, December 15–17, 2006, Proceedings*, Lecture Notes in Computer Science 4286, pp. 50–61. Berlin: Springer, 2006.

[Anshelevich et al. 04] Elliot Anshelevich, Anirban Dasgupta, Jon M. Kleinberg, Éva Tardos, Tom Wexler, and Tim Roughgarden. "The Price of Stability for Network Design with Fair Cost Allocation." In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pp. 295–304. Los Alamitos, CA: IEEE Press, 2004.

[Chakrabarty et al. 05] Deeparnab Chakrabarty, Aranyak Mehta, and Viswanath Nagarajan. "Fairness and Optimality in Congestion Games." In *Proceedings of the 6th ACM conference on Electronic Commerce*, pp. 52–57, New York: ACM Press, 2005.

[Dunkel and Schulz 06] Juliane Dunkel and Andreas S. Schulz. "On the Complexity of Pure-Strategy Nash Equilibria in Congestion and Local-Effect Games." In *Internet and Network Economics: Internet and Network Economics, Second International Workshop, WINE 2006, Patras, Greece, December 15–17, 2006, Proceedings*, Lecture Notes in Computer Science 4286, pp. 62–73. Berlin: Springer, 2006.

[Fabrikant et al. 04] Alex Fabrikant, Christos Papadimitriou, and Kunal Talwar. "The Complexity of Pure Nash Equilibria." In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pp. 604–612. New York: ACM Press, 2004.

[Fortune et al. 80] Steven Fortune, John E. Hopcroft, and James Wyllie. "The Directed Subgraph Homeomorphism Problem." *Theoretical Computer Science* 10 (1980), 111–121.

[Gairing et al. 06] Martin Gairing, Burkhard Monien, and Karsten Tiemann. "Routing (Un-)splittable Flow in Games with Player-Specific Linear Latency Functions." In *Automata, Languages and Programming: 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10–14, 2006, Proceedings, Part I*, Lecture Notes in Computer Science 4051, pp. 501–512. Berlin: Springer, 2006.

[Meyers 06] Carol Meyers. "Network Flow Problems and Congestion Games: Complexitiy and Approximation." PhD thesis, Massachusetts Institute of Technology, 2006.

[Milchtaich 96] Igal Milchtaich. "Congestion Games with Player-Specific Payoff Functions." *Games and Economic Behavior* 13:1 (1996), 111–124.

[Milchtaich 06] Igal Milchtaich. "The Equilibrium Existence Problem in Finite Network Congestion Games." In *Internet and Network Economics: Internet and Network Economics, Second International Workshop, WINE 2006, Patras, Greece, December 15–17, 2006, Proceedings*, Lecture Notes in Computer Science 4286, pp. 87–98. Berlin: Springer, 2006.

[Nisan et al. 07] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay Vazirani. *Algorithmic Game Theory.* Cambridge, UK: Cambridge University Press, 2007.

[Robertson and Seymour 95] Neil Robertson and Paul D. Seymour. "Graph Minors xiii: The Disjoint Paths Problem." *Journal of Combinatorial Theory, Series B*, 63:1 (1995), 65–110.

[Rosenthal 73] Robert W. Rosenthal. "A Class of Games Possessing Pure-Strategy Nash Equilibria." *Int. Journal of Game Theory* 2 (1973, 65–67.

[Schäffer and Yannakakis 91] Alejandro A. Schäffer and Mihalis Yannakakis. "Simple Local Search Problems That Are Hard to Solve." *SIAM Journal on Computing* 20:1 (1991), 56–87.

Heiner Ackermann, Fraunhofer-Institut für Techno- und Wirtschaftsmathematik, Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany (heiner.ackermann@itwm.fraunhofer.de)

Alexander Skopalik, Department of Computer Science, RWTH Aachen University, Ahornstr. 55, D-52056 Aachen, Germany (skopalik@cs.rwth-aachen.de)

Received February 18, 2008; accepted January 27, 2009.