Egerváry Research Group
on Combinatorial Optimization

Technical reportS

TR-2012-12. Published by the Egerváry Research Group, Pázmány P. sétány 1/C, H–1117, Budapest, Hungary. Web site: www.cs.elte.hu/egres . ISSN 1587–4451.

Equilibria in multiplayer multicommodity flow problems

Tamás Király and Júlia Pap

July 2012

Equilibria in multiplayer multicommodity flow problems

Tamás Király* and Júlia Pap**

Abstract

We investigate equilibria in multiplayer multicommodity flow problems where players have separate networks, and contracts regulate how certain players may hand over some of their traffic to others, which the latter have to route in their own network. This model was introduced by the authors with coauthors in an earlier paper, where it was shown that for fixed per-unit payments, equilibria always exist under some natural conditions. The present paper has three main results. First, we prove that finding an equilibrium is PPAD-complete. Second, we show a polynomial algorithm for the case when the digraph of contracts has a special structure: every strong component is a cycle. This result is based on an algorithm for finding approximate fixed points that may be of interest on its own. Finally, we show that it is possible to specify the prices in the contracts in such a way that the social optimum is in equilibrium, and furthermore the players are not motivated to cancel their contracts.

Keywords: Multicommodity flow; Equilibrium; PPAD

1 Introduction

The study of flow problems with selfish players dates back to the papers of Kalai and Zemel [8, 9]. The multicommodity version was studied by Derks and Tijs [5], and recent results on mechanism design have been presented in [1]. The model in these papers consists of a network where each arc is the property of a certain player, and players may cooperate, or buy some arcs or part of the capacities of arcs in order to facilitate the routing of their own demands.

In [3], the present authors, together with coauthors, introduced a slightly different model of multiplayer multicommodity flows. As in the aforementioned model, each player controls a subnetwork. The difference is that instead of buying and selling arc

*MTA-ELTE Egerváry Research Group, Eötvös Loránd University, Budapest, Hungary. Email: tkiraly@cs.elte.hu

**Department of Operations Research, Eötvös Loránd University, Budapest, Hungary. Email: papjuli@cs.elte.hu

capacities, pairs of players have contracts specifying source–destination pairs between which one player undertakes to route the traffic of the other player (up to a given maximum amount) in exchange for a specified per-unit payment.

It is important that the player who undertakes the routing can freely choose the route in his own network, so the other player is not buying specific arc capacities. Our model also includes flow value multipliers associated to the contracts; a motivation for these is that the size of data may increase during the transfer to another player's network. For example, players may wish to encrypt data before handing it over to another player.

The main part of [3] discussed the structure of solutions, the question whether a solution can be interpreted as actual routing of data in the network, and LP-based methods for computing social optimum solutions. In addition, the paper defined equilibria for fixed contract prices, and proved that in instances satisfying some natural conditions (called safe instances) an equilibrium always exists.

The current paper presents new results on equilibria for fixed contract prices, and also considers the mechanism design problem of finding suitable contract prices. In the fixed price case, we show that finding an equilibrium in safe instances is PPAD-complete, and that the difficulty of equilibrium computation depends on the structure of the contracts between players. In the case where each strong component of the digraph representing the contracts between players is a simple cycle, we are able to give a polynomial time algorithm. In contrast, the digraph for the PPAD-complete class of instances is C_4 with multiple parallel arcs. In view of these results, the problem seems particularly suited to the study of the borderline between polynomial and PPAD-complete problems.

Our polynomial algorithm can be seen as a more general result on the approximation of fixed points in a certain class of fixed point problems. Given n interval-valued mappings $\varphi_1, \ldots, \varphi_n$ on the unit interval, all with the closed graph property, Kakutani's fixed point theorem implies that there is a vector x such that $x_{i+1} \in \varphi_i(x_i)$ $(i = 1, \ldots, n-1)$ and $x_1 \in \varphi_n(x_n)$ (a *cyclically fixed vector*). We show an algorithm for finding n arbitrarily small intervals such that their Cartesian product contains a cyclically fixed vector. For the mappings obtained from the multicommodity flow problem, this approximation can be transformed into an exact equilibrium solution.

In the mechanism design problem, the contract prices are not fixed, but should be determined in a satisfying way. We show that this is possible in a strong sense. Our result gives contract prices ensuring that all social optimum solutions are in equilibrium, and in addition guaranteeing that no player is motivated to breach his contractual obligations.

The paper is organized as follows. In the rest of this section we give a precise description of the model, define social optimum and equilibrium solutions, and illustrate these concepts with examples. We also describe previous work on the topic and the results of [3]. Section 2 discusses the polyhedral aspects of the problem - these properties are used extensively later. In Section 3 we prove PPAD-completeness of finding equilibrium solutions. The polynomial algorithm for the case of simple cyclic contracts is presented in Section 4, including also the more general result on finding approximations of cyclically fixed vectors. Section 5 describes our solution for the

problem of choosing efficient and acceptable contract prices.

1.1 Problem formulation and example

An instance of the multiplayer multicommodity flow problem (MMFP) is defined by the following input. There are n players, and each player i has a digraph $D_i = (V, A_i)$ on the same node set. Every arc $a \in A_i$ has a cost $c_a \geq 0$ and a capacity $u_a \geq 0$. There is a subset of arcs $B_i \subseteq A_i$ called *contractual arcs*, with additional parameters: a *contract price* $p_a \geq 0$, and a *flow multiplier* $\gamma_a \geq 1$. Player i is the *buyer* at this arc, and there is another player associated to the arc: the *seller*. The set of contractual arcs where player i is the seller is denoted by S_i. The arcs in $A_i \setminus B_i$ are called *normal arcs*. For a contractual arc $a \in \cup_{i\in[n]} B_i$, the buyer of this edge is denoted by $\beta(a)$ and the seller by $\sigma(a)$. Let A denote the (disjoint) union of each A_i, and we define S and B analogously.

Every player has a set Q_i of *normal demands*. A demand $q \in Q_i$ has a size d_q, a source s_q, and a destination t_q. Player i has to send a flow of size d_q from s_q to t_q for every $q \in Q_i$.

The flow on a contractual arc creates a corresponding *contractual demand* at the seller: if the total flow on contractual arc $a \in S_i$ is x_a, then player i has an additional demand of $\gamma_a x_a$ from the tail of a to the head of a. The amount that the buyer of arc a pays to the seller (player i) in exchange is $p_a x_a$.

A *feasible solution* is a set of flows, one for each normal and contractual demand, that satisfies the above conditions and the capacity constraints. A *social optimum solution* is a feasible solution which is optimal for the cost function c. Note that the contract prices do not influence the social optimum.

An instance of MMFP is called a *safe instance* if each player i has a feasible solution for the conventional multicommodity flow problem on the digraph D_i (including the arcs in B_i) for all her normal demands *plus* all her contractual demands at their maximum size (for a contractual arc $a \in S_i$, the maximal contractual demand for player i is $\gamma_a u_a$).

An *equilibrium solution* is a feasible solution with the property that in each player's network it is a minimum cost multicommodity flow for the player's normal and contractual demands, where the contractual demands are fixed at the values determined by the other players' flows. The cost now includes also the contractual payments: the cost of arc $a \in B_i$ is $c_a + p_a$, while the cost of arc $a \in A_i \setminus B_i$ is c_a. Note that this notion of equilibrium is stronger than the usual Nash-equilibrium, because here players are unable to improve their objective in an equilibrium even by flow modifications that render the whole solution infeasible.

We demonstrate the various notions on a simple example that nevertheless shows the subtlety of the notion of equilibrium. The network and the arc costs are given in Figure 1 a). The capacities of all arcs are 1, and the contract prices are 0. The colour of a contractual arc represents the seller.

Player 1 has a demand of 1 unit from s to t. He can send it for free on the $suvt$ path. However, since uv is a contractual arc, this generates a unit uv demand for player 2, whose path includes a contractual arc st, resulting in an additional unit of

st demand for player 1. But if player 1 has 2 units of demand, then he cannot use arc *uv* because of the capacity limits, so he does not have 2 units of demand after all. This argument raises the question whether there is an equilibrium solution at all. The answer is that there is one (see part d) of the figure), with half a unit of flow on the contractual arc *uv*.

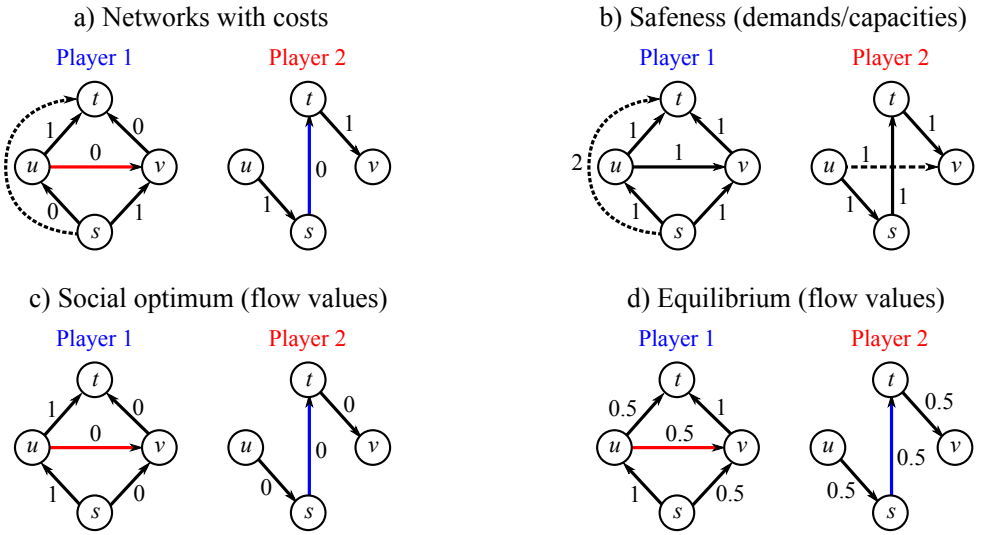

Figure 1: A simple instance of MMFP

Figure 1 b) illustrates the notion of "safe instance" on this example: In order for the instance to be safe, player 1 has to be able to route 2 units of *st* demand (his maximal possible demand) in his network including both normal and contractual arcs; likewise player 2 should be able to route 1 unit of *uv* demand in his network.

One social optimum solution is indicated on part c) of the figure, and the unique equilibrium solution is represented on part d). To check that the latter is indeed an equilibrium, we have to check that it is a minimum cost flow of size 1.5 from s to t in the network of player 1, and a minimum cost flow of size 0.5 from u to v in the network of player 2.

One may dismiss this example as unnatural because the first player is not tempted to use the contractual arc if he knows that the second player will hand back the flow to him. However, we can argue that the first player does not necessarily know that, since the data can be encrypted in a way that he does not recognize it. In fact, the definition of equilibrium is based on the assumption that the players know only their own network, and that contractual demands are not linked specifically to commodities. In the example, the first player cannot know that his contractual demand corresponds to his own commodity, since from his point of view it might as well correspond to a commodity of the second player.

Another objection against the plausibility of this example could be that a player would not sign a contract that is clearly disadvantageous for him. Although this is true, one may envisage a situation where circumstances have changed after the entry into force of a contract in such a way that is disadvantageous for one party. The

problem of finding contract prices that are beneficial to all players is addressed in Section 5.

1.2 Previous results

Flow games were introduced by Kalai and Zemel [8, 9] as cooperative games where players cooperate to achieve a maximum flow in the union of their subnetworks. They showed that the class of flow games coincides with the class of totally balanced games. Their model has been extended to a single-source single-sink multicommodity model by Derks and Tijs [5], who proved the non-emptiness of the core in this setting. A different model, where the players correspond to the nodes, was proposed by Papadimitriou [12], and the non-emptiness of the core was proved by Markakis and Saberi [10]. A game theoretic model for network formation was introduced in [2].

The mechanism design aspects of multicommodity flow games were recently studied by Agarwal and Ergun [1]. In their model, arcs are owned by players, who may buy arc capacities from other players in order to route their own demands. The paper proposes a method for determining capacity exchange costs which provide incentives for the players to route according to the optimal flow. The results were further generalized in [6].

The MMFP model studied in the present paper was introduced by the same authors with co-authors in [3]. The main results of [3] can be summarized in the following theorem.

Theorem 1.1 ([3]). *Feasible solutions of MMFP have the following properties.*

i) Given a feasible solution x of an MMFP instance, one can compute a realization of x, which corresponds to the shipment of commodities on normal arcs.

ii) A social optimum solution can be computed in polynomial time as the optimal solution of an LP of polynomial size.

iii) In a safe instance of MMFP there always exists an equilibrium solution.

2 Polyhedra

The set of multiflows in player i's network, where the contractual demands are also variables ($x_a^{i,\text{sell}}$), form a polyhedron which we denote by P_i. We also consider the direct product of these, denoted by $P_{\text{big}} \subseteq \mathbb{R}^{A \cup S}$. The feasible solutions are the vectors in P_{big} for which $x_a^{i,\text{sell}} = \gamma_a x_a^{\beta(a)}$ for each contractual arc a. That is, they form a polyhedron P_{feas} which is the intersection of P_{big} with the subspace

$$H := \{x \in \mathbb{R}^{A \cup S} : x_a^{i,\text{sell}} = \gamma_a x_a^{\beta(a)} \ \forall i \in [n], a \in S_i\}.$$

The social optimum can be obtained by minimizing the cost over the polyhedron P_{feas}, thus the social optimal solutions form a face of it. The set of equilibrium solutions on the other hand is not necessarily convex (not even connected), but the following holds.

Lemma 2.1. *The set of equilibrium solutions is the union of some faces of P_{feas}.*

Proof. Let Φ_ξ be the affine subspace of $\mathbb{R}^{A\cup S}$ where we fix the coordinates in S to some values $(\xi_a)_{a\in S}$, that is, we fix all the contractual demands. A feasible solution x is an equilibrium if and only if it is optimal in $P_{\text{big}} \cap \Phi_{x|S}$ for the objective function consisting of the costs and the prices. Here $x \mid S$ denotes the restriction of x to the coordinates in S. It is easy to see that the set of not necessarily feasible vectors x in P_{big} that minimize the objective function in $P_{\text{big}} \cap \Phi_{x|S}$ is the union of faces of P_{big}. Since the intersection of a face of P_{big} and H is a face of P_{feas}, the lemma follows. □

3 PPAD-completeness

In this section we show that the problem of finding equilibrium solutions in safe instances is PPAD-complete. Membership in PPAD follows from the fact that the computational version of Kakutani's fixed point theorem is in PPAD, see [11].

Completeness is proved by reducing two-player Nash equilibrium to our problem. To be more precise, we reduce approximate 2-Nash, so we use the following breakthrough result of Chen, Deng, and Teng [4].

Theorem 3.1 ([4])**.** *For any $c > 0$, the problem of computing an n^{-c}-approximate Nash equilibrium of a two-player game is PPAD-complete.*

We need a couple of remarks in order to use this theorem. First, it is well known that the problem of finding Nash equilibria can be reduced to finding symmetric Nash equilibria in symmetric games, so we will assume that the game is symmetric, with utility matrix A. Second, the above theorem is valid if the matrix A is normalized in the sense that its entries are bounded. For our purposes it is convenient to say that a symmetric two-player game is *normalized* if the elements of the matrix A are rationals in the interval $[1, 2]$. Third, approximate equilibria can be defined in several ways; we use a definition in [4]: x^* is an *ϵ-well supported approximate symmetric Nash equilibrium* if $x^*_k > 0$ implies that $\sum_{j=1}^n a_{kj}x^*_j > \max_{i\in[n]} \sum_{j=1}^n a_{ij}x^*_j - \epsilon$. To sum up, we use the following form of the theorem.

Corollary 3.2 ([4])**.** *For any $c > 0$, the problem of computing an n^{-c}-well supported approximate symmetric Nash equilibrium of a symmetric normalized two-player game is PPAD-complete.*

The above problem will be called n^{-c}-APPROXIMATE 2-NASH in this paper.

Theorem 3.3. *It is PPAD-complete to find an equilibrium solution in safe instances of MMFP.*

Proof. For a given c, we have to reduce n^{-c}-APPROXIMATE 2-NASH to finding an equilibrium in a safe instance of MMFP.

Given a symmetric normalized game defined by a matrix $A \in [1, 2]^{n\times n}$, we construct a safe instance of MMFP featuring 4 players. In order to make the construction more understandable, the players are named Decision Maker, Combiner, Maximizer

and Inverter. The high level view of the role of the players is that the Decision Maker decides the values of x satisfying $x \geq 0$ and $\sum_{j=1}^{n} x_j = 1$, while the other players are "gadgets" that compute $n^c(M - \sum_{j=1}^{n} a_{kj}x_j)$ for every k, where $M = \max_{i\in[n]} \sum_{j=1}^{n} a_{ij}x_j$. These values then appear in the network of the Decision Maker as contractual demands, in such a way that in an equilibrium solution x is guaranteed to be an n^{-c}-well supported approximate Nash equilibrium. The contract prices will be 0 on all contractual arcs. The details of the construction, illustrated on Figure 2, are the following.

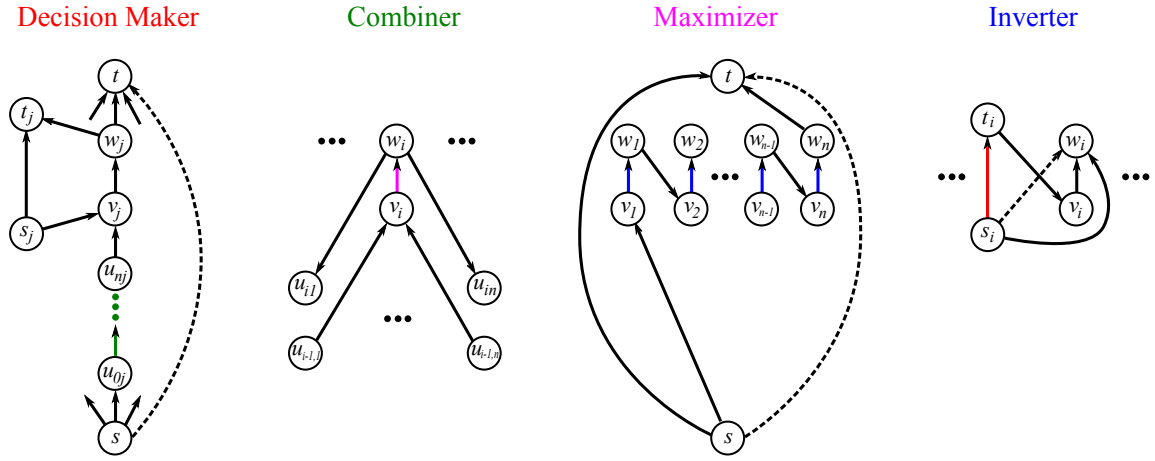

Figure 2: Illustration of the networks of various players

The Decision Maker has one normal demand: a unit commodity from s to t. His network contains n internally disjoint $s-t$-paths: $s, u_{0j}, u_{1j}, \ldots, u_{nj}, v_j, w_j, t$ $(j = 1, \ldots, n)$, each arc having cost 0 and capacity 1. The arcs $u_{i-1,j}u_{ij}$ $(i, j \in [n])$ are contractual arcs to the Combiner, and the multiplier of $u_{i-1,j}u_{ij}$ is a_{ij}.

The network of the Decision Maker also contains nodes s_j, t_j $(j = 1, \ldots, n)$, arcs s_jt_j with cost 1 and capacity $2n^c$, and arcs s_jv_j, w_jt_j with cost 0 and capacity 1.

For a feasible solution x, we will denote by x_j the flow value on the arc su_{0j} in the network of the Decision Maker.

The network of the Combiner consists of paths $u_{i-1,j}, v_i, w_i, u_{ij}$ for $i, j \in [n]$, all arcs having cost 0 and capacity 2. The arcs v_iw_i are contractual arcs to the Maximizer with multiplier 1. Note that the Combiner has a unique way to route his contractual demands, and in a feasible solution the flow value on arc v_iw_i is $\sum_{j=1}^{n} a_{ij}x_j$, so this is the contractual demand appearing at the Maximizer.

The network of the Maximizer consists of a path $s, v_1, w_1, v_2, w_2, \ldots, v_n, w_n, t$, all arcs having cost 0 and capacity 2. The arcs v_iw_i are contractual arcs to the Inverter, with multiplier 1. There is also an arc st with cost 1 and capacity 2. The Maximizer has a normal demand of 2 from s to t. The routing that the Maximizer has to choose in an equilibrium solution is the following: he must route his contractual demands on the edges v_iw_i; he must route as much of his normal demand on the long $s-t$ path as possible, and route the rest on the arc st. Let M denote $\max_{i\in[n]} \sum_{j=1}^{n} a_{ij}x_j$ (note that $M \leq 2$). Then the portion of the normal demand routed on the long path is

$2 - M$ units, so the contractual demand appearing at the Inverter between v_i and w_i is $2 - M + \sum_{j=1}^{n} a_{ij}x_j$ units.

Inverter has a path s_i, t_i, v_i, w_i for each $i \in [n]$ with arcs of cost 0 and capacity 2. The arcs s_it_i are contractual arcs to the Decision Maker, with multiplier n^c. In addition he has arcs s_iw_i ($i \in [n]$) with cost 1 and capacity 2, and normal demands of size 2 from s_i to w_i. In an equilibrium solution, he routes his contractual demands on the arcs v_iw_i, and routes as much of his normal $s_i - w_i$ demand on the path of cost zero as possible, the bottleneck being the arc v_iw_i. Thus $M - \sum_{j=1}^{n} a_{ij}x_j$ is routed on the path of cost zero, and $2 - M + \sum_{j=1}^{n} a_{ij}x_j$ on the arc s_iw_i. This means that the contractual demand appearing at the Decision Maker between s_i and t_i is $n^c(M - \sum_{j=1}^{n} a_{ij}x_j)$.

It is easy to check that this construction results in a safe instance: we added arcs of non-zero cost with enough capacity to carry the maximum demands.

Let us prove that an equilibrium solution x^* corresponds to an n^{-c}-well supported approximate Nash equilibrium. Clearly, $x^* \geq 0$ and $\sum_{j=1}^{n} x_j^* = 1$, so we have to prove that $x_i^* > 0$ implies that $\sum_{j=1}^{n} a_{ij}x_j^* > M - n^{-c}$, where $M = \max_{i\in[n]} \sum_{j=1}^{n} a_{ij}x_j^*$. In the discussion above we have showed that the Decision Maker has a contractual demand of $n^c(M - \sum_{j=1}^{n} a_{ij}x_j^*)$ from s_i to t_i. We know that there is an index k such that $\sum_{j=1}^{n} a_{kj}x_j^* = M$, so the contractual demand from s_k to t_k is 0. This means that the following is a feasible multicommodity flow in the network of the Decision Maker:

- route the normal demand on the path $s, u_{0k}, u_{1k}, \ldots, u_{nk}, v_k, w_k, t$,
- for $i \neq k$, let $\delta_i = n^c(M - \sum_{j=1}^{n} a_{ij}x_j^*)$. Route $\min\{1, \delta_i\}$ units of the $s_i - t_i$ contractual demand on the path s_i, v_i, w_i, t_i, and the rest on the arc s_it_i.

It is easy to check that this is a minimum cost multicommodity flow for the given demands. Since x^* is an equilibrium solution, it must have the same cost as this one in the network of the Decision Maker. This is only possible if $\delta_i \leq 1 - x_i^*$ whenever $x_i^* > 0$. Thus $M - \sum_{j=1}^{n} a_{ij}x_j^* < n^{-c}$ whenever $x_i^* > 0$, which means that x^* is an n^{-c}-well supported approximate Nash equilibrium. □

4 Polynomially solvable cases

In the PPAD-completeness proof in Section 3, we reduced approximate 2-Nash to 4-player MMCF instances where the buyer-seller pairs constituted a directed 4-cycle on the set of players. One may ask if this leaves room for an interesting class of contract structures where an equilibrium can be found in polynomial time.

A natural candidate is the class of acyclic contract structures, and indeed it is not hard to show that an equilibrium can be computed efficiently in that case. The main result of this section is an efficient algorithm for a broader class of contract structures. To define it, let us consider an auxiliary directed graph $D^* = ([n], A^*)$ on the set of players. for each contractual arc a, there is an arc from the buyer of a to the seller of a. We allow parallel arcs, so $|A^*| = |B|$.

With this definition, the directed graph corresponding to the hard instances constructed in the proof of Theorem 3.3 is a 4-cycle *with many parallel arcs*. In contrast

to this, the main result of this section states that if the strongly connected components of D^* are *simple* directed cycles, then there is a polynomial time algorithm to find an equilibrium.

Theorem 4.1. *An equilibrium solution can be found in polynomial time in safe instances where every strong component of D^* is a simple directed cycle.*

The main tool of the proof is an algorithm for finding approximate fixed points of a special kind of mapping. Let $\varphi_i : [0,1] \to \mathcal{P}([0,1])$ $(i \in [n])$ be mappings such that $\varphi_i(t)$ is a non-empty interval for every $t \in [0,1]$, and the graph of φ_i is closed. We are interested in finding a fixed point of the mapping $(x_1, x_2, \ldots, x_n) \mapsto (\varphi_n(x_n), \varphi_1(x_1), \ldots, \varphi_{n-1}(x_{n-1}))$. In other words, we are looking for a vector $x = (x_1, \ldots, x_n)$ for which $x_{i+1} \in \varphi_i(x_i)$ for every $i \in [n]$. Here and later in this section, unless otherwise stated, we consider the indices modulo n, i.e. $x_{n+1} = x_1$ and $\varphi_n = \varphi_0$.

Definition. A vector $x = (x_1, \ldots, x_n)$ that satisfies $x_{i+1} \in \varphi_i(x_i)$ for every $i \in [n]$ is called a *cyclically fixed vector*.

In order to have a meaningful definition of running time, we use the following oracle model: there is an evaluation oracle which, given $t \in [0,1]$ and $i \in [n]$, returns some $z \in \varphi_i(t)$ in one step. We also consider basic arithmetic operations as one step. Of course it is hopeless to compute a cyclically fixed vector exactly in this oracle model, but we can prove the following approximation result in this setting.

Theorem 4.2. *Let $\varphi_i : [0,1] \to \mathcal{P}([0,1])$ $(i \in [n])$ be given as above, and let $0 < \epsilon < 1$. In $O(n^2 \log(\frac{1}{\epsilon}))$ steps we can find intervals $I_1, \ldots, I_n \subseteq [0,1]$ of length at most ϵ such that there is a cyclically fixed vector x with $x_i \in I_i$ $(i \in [n])$.*

Proof. The algorithm is quite simple and its time complexity is straightforward, the more involved part being the proof of its correctness. During the algorithm we always follow the rule that if at some point the oracle returns $z \in \varphi_i(t)$, then the triplet (i, t, z) is stored, and we use z in all subsequent evaluations of $\varphi_i(t)$.

The intervals are determined successively in reverse order. To determine I_n, initially let $a_n = 0$ and $b_n = 1$. Let $\psi_n = \varphi_{n-1} \circ \varphi_{n-2} \circ \cdots \circ \varphi_1 \circ \varphi_n$. The function ψ_n is an interval-valued function with a closed graph, since it is the composition of such functions.

Let $t = (a_n + b_n)/2$. We can compute a value $z \in \psi_n(t)$ by n successive oracle calls. If $z \leq t$, then let $b_n = t$. If $z \geq t$, then let $a_n = t$. These steps are repeated until $b_n - a_n \leq \epsilon$. Let $I_n = [a_n, b_n]$.

Suppose that we have already determined $I_{i+1} = [a_{i+1}, b_{i+1}]$. We modify the function φ_i as follows.

$$\varphi_i'(t) = \begin{cases} \varphi_i(t) \cap I_{i+1} & \text{if } \varphi_i(t) \cap I_{i+1} \neq \emptyset, \\ a_{i+1} & \text{if } z < a_{i+1} \text{ for every } z \in \varphi_i(t), \\ b_{i+1} & \text{if } z > b_{i+1} \text{ for every } z \in \varphi_i(t). \end{cases}$$

Note that this modification can be implemented simply by modifying the value returned by the oracle after each oracle call for φ_i: if the returned value is smaller than

a_{i+1}, then we change it to a_{i+1}, and if it is greater than b_{i+1}, then we change it to b_{i+1}.

In order to compute I_i, initially let $a_i = 0$ and $b_i = 1$, and let $\psi_i = \varphi_{i-1} \circ \varphi_{i-2} \circ \cdots \circ \varphi_0 \circ \varphi'_{n-1} \circ \cdots \circ \varphi'_{i+1} \circ \varphi'_i$.

In a general step, let $t = (a_i + b_i)/2$, and let us compute a value $z \in \psi_i(t)$ by n oracle calls.

Definition. The sequence of the returned values of these n oracle calls is called the *itinerary* of the pair (i, t).

Let $b_i = t$ if $z \leq t$, and let $a_i = t$ if $z \geq t$. The above steps are repeated until $b_i - a_i \leq \epsilon$, in which case we fix I_i to be the interval $[a_i, b_i]$. We can observe the following.

Observation 4.3. *The n-th step of the itinerary of (i, a_i) is greater than a_i, and the n-th step of the itinerary of (i, b_i) is smaller than b_i.* □

The algorithm described above computes the interval I_i in $O(n \log(\frac{1}{\epsilon}))$ steps for a given i, so the total number of steps is $O(n^2 \log(\frac{1}{\epsilon}))$. It remains to show that the intervals contain a cyclically fixed vector.

Let us define mappings $\varphi_i^* : I_i \to \mathcal{P}(I_{i+1})$ for each $i \in [n]$ by $\varphi_i^*(t) = \varphi_i(t) \cap I_{i+1}$. Note that the image for a value t is either a closed interval or the empty set. We also define for any positive integer k (this time not taken modulo n) the mapping $\psi_k^* = \varphi_k^* \circ \varphi_{k-1}^* \circ \cdots \circ \varphi_1^*$. We can observe that $\psi_{n+k}^*(I_1) \subseteq \psi_k^*(I_1)$ for any k. We define ψ_0^* to be the identity function.

Claim 4.4. *The set $\psi_k^*(I_1)$ is non-empty for every k.*

Proof. Indirectly, suppose that $in + k$ (where $1 \leq k \leq n$) is the smallest integer for which $\psi_{in+k}^*(I_1) = \emptyset$. We may assume w.l.o.g. that $\varphi_k \circ \psi_{in+k-1}^*(t) > b_{k+1}$ for every $t \in I_1$. This implies that $b_{k+1} \in \psi_{(i-1)n+k}^*(I_1)$ (provided that $(i-1)n+k \geq 0$), because $\varphi_k \circ \psi_{(i-1)n+k-1}^*(t)$ is an interval that contains b_{k+1}.

Let us examine the itinerary of $(k+1, b_{k+1})$. We claim that for any $j < n$ the j-th step of the itinerary is in $\psi_{(i-1)n+k+j}^*(I_1)$, provided that $(i-1)n+k+j \geq 0$. We show this by induction on j, first for $j < n - k$. In this case $\psi_{(i-1)n+k+j}^*(I_1)$ is non-empty and so the set $\varphi'_{k+j} \circ \psi_{(i-1)n+k+j-1}^*(I_1)$, which contains the j-th step of the itinerary, is equal to $\psi_{(i-1)n+k+j}^*(I_1)$.

Next, we show that for $n-k \leq j < n$ it also holds that the j-th step of the itinerary is in $\psi_{(i-1)n+k+j}^*(I_1)$. The difficulty here is that the itinerary proceeds according to the mapping φ_{k+j}, which, as opposed to φ'_{k+j}, may have values outside of I_{k+j+1}. Suppose that the j-th step is the first to be outside of I_{k+j+1}; we can assume w.l.o.g. that it is greater than b_{k+j+1}. This means that b_{k+j+1} is in $\psi_{(i-1)n+k+j}^*(I_1)$ (because $\psi_{(i-1)n+k+j}^*(I_1)$ is non-empty), so the itinerary of $(k+j+1, b_{k+j+1})$ leads to b_{k+1}, after which it takes the same steps as the itinerary of $(k+1, b_{k+1})$ — here we use the rule that the evaluation oracle cannot return different values for the same input. Thus the n-th step of the itinerary of $(k+j+1, b_{k+j+1})$ is greater than b_{k+j+1}, contradicting Observation 4.3.

We can conclude that the $(n-1)$-th step of the itinerary of $(k+1, b_{k+1})$ is in $\psi^*_{in+k-1}(I_1)$. Therefore the n-th step of the itinerary is greater than b_{k+1} by our assumption, again contradicting Observation 4.3. □

Since the set $\psi^*_{in}(I_1)$ is a non-empty closed interval for every i, and $\psi^*_{(i+1)n}(I_1) \subseteq \psi^*_{in}(I_1)$, we have that

$$R = \cap_{i=1}^{\infty} \psi^*_{in}(I_1) \text{ is a non-empty interval } [a, b].$$

Claim 4.5. *R contains a fixed point of ψ^*_n.*

Proof. It is easy to see that $\psi^*_n(R) = R$. For $k \in [n]$, let

$$R_k = \{t \in R : \psi^*_k(t) = \emptyset, \text{ but } \psi^*_j(t) \neq \emptyset \text{ for } j < k\},$$

and let $R^* = R \setminus \cup_{j=1}^{n} R_j$. Our aim is to find an interval $I^* = [a^*, b^*] \subseteq R^*$ such that $\psi^*_n(I^*) = R$. To do this, we show that for every $k \in [n]$, there is an interval $I^*_k = [a^*_k, b^*_k] \subseteq R \setminus \cup_{j=1}^{k} R_j$ such that $\psi^*_n(I^*_k) = R$. We can start with $I^*_0 = R$; suppose that we have already determined I^*_{k-1}. If $t \in R_k \cap I^*_{k-1}$, then either $\psi^*_k([a^*_{k-1}, t]) \subseteq \psi^*_k([t, b^*_{k-1}])$ or vice versa because both are sub-intervals of I_{k+1} containing the same endpoint of I_{k+1}. Consequently, either $\psi^*_n([a^*_{k-1}, t]) = R$ or $\psi^*_n([t, b^*_{k-1}]) = R$. We may assume w.l.o.g. that there is at least one t for which $\psi^*_n([t, b^*_{k-1}]) = R$. Let $a^*_k = \sup\{t \in I^*_{k-1} : \psi^*_n([t, b^*_{k-1}]) = R\}$.

Because of the closed graph property, $a^*_k \notin R_k$ and $\psi^*_n([a^*_k, b^*_{k-1}]) = R$ holds. If $R_k \cap [a^*_k, b^*_{k-1}] = \emptyset$, then we can set $b^*_k = b^*_{k-1}$. Otherwise we can observe that $\psi^*_n([t, b^*_{k-1}]) \neq R$ for every $t \in R_k \cap ([a^*_k, b^*_{k-1}]$. On the other hand, for such a t both $\psi^*_k([a^*_k, t])$ and $\psi^*_k([t, b^*_{k-1}])$ contain the same endpoint of I_{k+1}, so one of them contains the other. Since $\psi^*_n([a^*_k, t]) \cup \psi^*_n([t, b^*_{k-1}]) = R$, it follows that $\psi^*_n([a^*_k, t]) = R$. Let $b^*_k = \inf\{R_k \cap [a^*_k, b^*_{k-1}]\}$; then $\psi^*_n([a^*_k, b^*_k]) = R$, and $[a^*_k, b^*_k] \cap R_k = \emptyset$, as required.

We obtained an interval $I^* = [a^*, b^*] \subseteq R^*$ such that $\psi^*_n(I^*) = R$. Now it follows from the closed graph property of ψ^*_n on I^* that it has a fixed point in I^*. □

By definition, a fixed point of ψ^*_n implies the existence of a cyclically fixed vector x with $x_i \in I_i$ $(i \in [n])$. This concludes the proof of the theorem. □

Proof of Theorem 4.1. First let us consider the case when D^* is a simple directed cycle. Player i has a single contractual arc a_i, and the seller for this arc is player $i+1$ (indices are meant modulo n). Let $X_i = [0, u_i]$, the interval of possible flow values on arc a_i. The function $\varphi_i : X_i \to \mathcal{P}(X_{i+1})$ is an interval-valued function defined as follows.

Definition. If the flow on arc a_i is fixed at $t \in X_i$, then the single contractual demand of player $i+1$ is $\gamma_i t$ units, so all demands of player $i+1$ are fixed. Let F_{i+1} be the set of minimum cost multicommodity flows in the network of player $i+1$ for these demands, for cost $c+p$. We define $\varphi_i(t)$ as

$$\varphi_i(t) = \{z \in X_{i+1} \mid \exists\ f \in F_{i+1} : f_{a_{i+1}} = z\}.$$

Since we have a safe instance, the set $\varphi_i(t)$ is a non-empty interval. We can implement the oracle required for Theorem 4.2 using a minimum cost multicommodity flow computation, and by choosing a basic solution, we can guarantee that the size (in bits) of the returned value is polynomial.

The problem of finding an equilibrium solution is equivalent to finding a cyclically fixed vector of $\varphi_1, \ldots, \varphi_n$. Let N be the size of the input, and let $\epsilon = \exp(-N^2)$. According to Theorem 4.2, we can find intervals $I_i \subseteq X_i$ ($i \in [n]$) of length at most ϵ in polynomial time such that there is a cyclically fixed vector x with $x_i \in I_i$ ($i \in [n]$). Since the graph of the function φ_i is a 2-dimensional polyhedral complex obtained by projecting faces of P_{big}, the choice of ϵ guarantees that all vertices of this polyhedral complex in the interior of $I_i \times X_{i+1}$ (if there is one) have the same first coordinate t. We modify the algorithm in the proof of Theorem 4.2 the following way: after achieving $b_i - a_i \leq \epsilon$, we make an additional step with the above t in place of $t = (a_i + b_i)/2$. This way we obtain intervals I_i such that the graph of φ_i has no vertices in the interior of $I_i \times X_{i+1}$ ($i \in [n]$). This means that the mappings φ_i are "linear" on these intervals in the sense that their graph is of the form $\alpha_i x_i \leq x_{i+1} \leq \beta_i x_i$, and an equilibrium can be found by solving an LP.

Next, let us prove the general case, when every strong component of D^* is a simple directed cycle. Let $C_1, \ldots, C_k$ be the family of strong components of D^* in topological order. We compute the flows of players in $C_1, \ldots, C_k$ successively.

To determine the flows for players in C_1, we simply restrict the problem to C_1, and compute the equilibrium solution as above (if $|C_1| = 1$, then this is a single minimum cost multicommodity flow computation for cost $c + p$). Since players in C_1 are sellers only for arcs whose buyer is also in C_1, the flows satisfy the equilibrium conditions for these players.

Suppose that we have already determined the flows for players in $C_1 \cup \cdots \cup C_{j-1}$. These flows determine some of the contractual demands of players in C_j, namely those corresponding to contractual arcs with buyer not in C_j. We consider all these fixed demands as normal demands, hence the remaining contractual demands of players in C_j form a simple directed cycle, so we can use the above algorithm to compute an equilibrium for the problem restricted to C_j.

The end result is a feasible solution because all demands are satisfied, and it is in equilibrium since the construction guarantees that players in each C_j have minimum cost multicommodity flows for cost $c + p$. □

5 Efficient and fair prices

In this section we consider the problem of assigning prices to the contractual arcs. The goal is to achieve that social optimal solutions are in equilibrium. Another objective is the satisfaction of the players with the assigned prices. We prove the following.

Theorem 5.1. *It is possible to compute prices to the contractual arcs in polynomial time so that every social optimal solution is an equilibrium solution, and the players could not achieve a better solution in their own network even if they could change the amount of flow other players buy from them.*

Proof. We use the terminology and notation of Section 2. Let F_{socopt} be the set of the social optimal solutions, that is, the optimal face in P_{feas} minimizing the cost c (here the variables $x_a^{i,\text{sell}}$ have coefficient 0). Let F_{big} be the minimal face of P_{big} which contains F_{socopt}. We need the following lemma on optimal cones.

Lemma 5.2. *Let P_1 be a polyhedron, Π an affine subspace and $P_2 = P_1 \cap \Pi$. Let F_2 be a face of P_2 and let F_1 be the smallest face of P_1 that contains F_2. Then*

(i) $\text{opt.cone}(F_2, P_2) = \text{opt.cone}(F_1, P_1) + (\text{lin}\Pi)^\perp$,

(ii) $\text{relint}(\text{opt.cone}(F_2, P_2)) = \text{relint}(\text{opt.cone}(F_1, P_1)) + (\text{lin}\Pi)^\perp$.

Proof. To prove the "$\supseteq$" containment in part (i), let $c \in \text{opt.cone}(F_1, P_1)$ and $a \in (\text{lin}\Pi)^\perp$. Then for any $x \in F_2$ and $x' \in P_2$, $(c+a)x = cx + ax \geq cx' + ax = (c+a)x'$, so $c + a \in \text{opt.cone}(F_2, P_2)$.

For the "$\subseteq$" containment suppose that $c \in \text{opt.cone}(F_2, P_2)$ but c is not in the cone $\text{opt.cone}(F_1, P_1) + (\text{lin}\Pi)^\perp$. By Farkas' Lemma the latter implies that there is a vector y for which $cy > 0$ but $(c' + a)y \leq 0$ for every $c' \in \text{opt.cone}(F_1, P_1)$ and $a \in (\text{lin}\Pi)^\perp$. Clearly $y \in \text{lin}\Pi$. Let x^* be a vector in $\text{relint}(F_2)$. Then $\text{opt.cone}(F_1, P_1)$ is generated by the normal vectors of the inequalities of P_1 that x^* satisfies with equality. This means that y is in the tangent cone of P_1 in x^*, thus, since $y \in \text{lin}\Pi$, y is also in the tangent cone of P_2 in x^*. This contradicts to $cy > 0$.

Part (ii) follows from part (i). □

We apply the lemma with $P_1 = P_{\text{big}}$, $\Pi = H$, $P_2 = P_{\text{feas}}$, and $F_2 = F_{\text{socopt}}$. By consequence, there is a vector $h \in (\text{lin}H)^\perp$ for which $c + h$ is a member of $\text{relint}(\text{opt.cone}(F_{\text{big}}, P_{\text{big}}))$. By the definition of H, h is 0 on the normal arcs, and for a contractual arc a, $h_a^{\beta(a)} = -\gamma_a h_a^{\sigma(a),\text{sell}}$. Note that h can be computed in polynomial time by linear programming. Let the price of a contractual arc a be $p_a := h_a^{\beta(a)}$.

First we prove that the social optima are equilibria. Let x be a social optimal solution. Then by Lemma 5.2, x is optimal in P_{big} for the objective function $c + h$. Thus x is optimal in $P_{\text{big}} \cap \Phi_{x|S}$ for the objective function $c + h \mid A$. This means that x is an equilibrium solution.

It also follows that $x \mid A_i \cup S_i$ is optimal for the objective function $c + h \mid A_i \cup S_i$. Since for a contractual arc a, $x_a^{\sigma(a),\text{sell}} h_a^{\sigma(a),\text{sell}} = \gamma_a x_a^{\beta(a)} \cdot (-\frac{h_a^{\beta(a)}}{\gamma_a}) = -p_a x_a^{\beta(a)}$, the objective value of $x \mid A_i \cup S_i$ is the total cost of player i with the assigned prices, if her contractual demands are set to $x \mid S_i$ and her multiflow is $x \mid A_i$. This means that even if player i could change the amount of flow that other players buy from her, she could not achieve a multiflow with less cost. □

Acknowledgements

We would like to thank to all authors of [3] for fruitful discussions on the topic. The authors received a grant (no. CK 80124) from the National Development Agency of Hungary, based on a source from the Research and Technology Innovation Fund.

References

[1] R. Agarwal and Ö. Ergun, *Mechanism design for a multicommodity flow game in service network alliances*, Operations Research Letters 36 (2008), 520–524.

[2] E. Anshelevich, B. Shepherd, G. Wilfong, *Strategic Network Formation through Peering and Service Agreements*, Proceedings of 47th Annual IEEE Symposium on Foundations of Computer Science (2006), 77–86.

[3] A. Bernáth, T. Király, E.R. Kovács, G. Mádi-Nagy, Gy. Pap, J. Pap, J. Szabó, L. Végh, *Algorithms for multiplayer multicommodity flow problems*, to appear in Proc. 29th Hungarian Operations Research Conference, Central European Journal of Operations Research.

[4] X. Chen, X. Deng, S-H. Teng, *Settling the complexity of computing two-player Nash equilibria*, Journal of the ACM, Volume 56 Issue 3, May 2009, Article No. 14

[5] J.J.M. Derks and S.H. Tijs, *Stable outcomes for multicommodity flow games*, Methods of Operations Research, 50 (1985), 493–504.

[6] L. Guyi and Ö. Ergun, *Dual Payoffs, Core and a Collaboration Mechanism Based on Capacity Exchange Prices in Multicommodity Flow Games*, Proceedings of the 4th International Workshop on Internet and Network Economics (WINE '08), 2008, 61–69.

[7] S. Kakutani, *A generalization of Brouwer's fixed point theorem*, Duke Math. J. 8 (1941), 457–459.

[8] E. Kalai, E. Zemel, *Generalized network problems yielding totally balanced games*, Operations Research 30 (5) (1981) 998–1008.

[9] E. Kalai, E. Zemel, *Totally balanced games and games of flow*, Mathematics of Operations Research 7 (3) (1982) 476–478.

[10] E. Markakis, A. Saberi, *On the core of the multicommodity flow game*, Proceedings of the 4th ACM conference on Electronic commerce (2003), 93–97.

[11] C.H. Papadimitriou, *On the complexity of the parity argument and other inefficient proofs of existence*, Journal of Computer and System Sciences Volume 48, Issue 3 (1994), 498–532.

[12] C.H. Papadimitriou, *Algorithms, Games, and the Internet*, Proceedings of the thirty-third annual ACM symposium on Theory of computing (2001), 749–753.