

# Algorithms for Core Stability, Core Largeness, Exactness, and Extendability of Flow Games<sup>\*</sup>

Qizhi Fang<sup>1</sup>, Rudolf Fleischer<sup>2</sup>, Jian Li<sup>2\*\*</sup>, and Xiaoxun Sun<sup>3</sup>

<sup>1</sup> Department of Mathematics, Ocean University of China, Qingdao, China  
Email: [qfang@ouc.edu.cn](mailto:qfang@ouc.edu.cn)

<sup>2</sup> Department of Computer Science and Engineering  
Shanghai Key Laboratory of Intelligent Information Processing  
Fudan University, Shanghai, China  
Email: [rudolf,lijian83@fudan.edu.cn](mailto:rudolf,lijian83@fudan.edu.cn)

<sup>3</sup> Department of Mathematics and Computing, University of Southern Queensland  
Email: [w0072830@mail.connect.usq.edu.au](mailto:w0072830@mail.connect.usq.edu.au)

**Abstract.** In this paper, we give linear time algorithms to decide core stability, core largeness, exactness, and extendability of flow games on uniform networks (all edge capacities are 1). We show that a uniform flow game has a stable core if and only if the network is a balanced DAG (for all non-terminal vertices, indegree equals outdegree), which can be decided in linear time. Then we show that uniform flow games are exact, extendable, and have a large core if and only if the network is a balanced directed series-parallel graph, which again can be decided in linear time.

## 1 Introduction

In 1944, von Neumann and Morgenstern [19] introduced the concept of *stable sets* to analyse bargaining situations in cooperative games. In 1968, Lucas [13] described a ten-person game without a stable set. Deng and Papadimitriou [5] pointed out that deciding the existence of a stable set for a given cooperative game is not known to be computable. Jain and Vohra [8] recently showed that it is decidable whether a balanced game has a stable core. When the game is convex, the core is always a stable set. In general, however, the core and the stable set are not related. Hence the question arises: when do the core and the stable set coincide, and how can we decide core stability?

Several sufficient conditions for core stability have been discussed in the literature. Sharkey [14] introduced the concepts of subconvexity of a

---

<sup>\*</sup> The work described in this paper was supported by NCET, NSFC (No. 10371114s and No. 70571040/G0105) and partially supported by NSFC (No. 60573025).

<sup>\*\*</sup> The work was partially done when this author was visiting The Hong Kong University of Science and Technology.

game and core largeness. He showed that convexity implies subconvexity which implies core largeness which implies core stability. In an unpublished paper, Kikuta and Shapley [12] studied another concept, later called extendability of the game by Gellekom *et al.* [18], and proved that it is necessary for core largeness and sufficient for core stability.

However, only few results are known about core stability and related concepts for concrete cooperative games. Solymosi and Raghavan [15] studied these concepts for assignment games, and Bietenhader and Okamoto [1] studied them for minimum coloring games on perfect graphs.

In this paper we study flow games, introduced by Kalai and Zemel [10, 11], which arise from the profit distribution problem related to the maximum flow in networks. We give the first linear time algorithms to decide core stability, core largeness, exactness, and extendability of uniform flow games (all edge capacities are 1). We obtain these efficient algorithms by characterizing structural properties of those networks that have flow games with the desired properties. These characterizations might be useful in other contexts.

We show that a uniform flow game has a stable core if and only if the network is a balanced DAG (for all non-terminal vertices, indegree equals outdegree). This can easily be tested in linear time, so we get a linear time algorithm to decide core stability, which improves a previous algorithm with runtime  $O(|V|^2 \cdot |E|^2)$  [16]. We also show that uniform flow games are exact, extendable, and have a large core if and only if the network is a balanced directed series-parallel graph. Again, this can be tested in linear time [17], so we also get a linear time algorithm to decide exactness, extendability, and core largeness of uniform flow games. In [16], Sun *et al.* established the equivalence of these three properties and proved them to be equivalent to a certain structural property of the network (see Section 2.2.4) but left it as an open problem to design an efficient algorithm to decide this property. Note that core largeness, exactness, and extendability of a flow game all imply core stability (because flow games are totally balanced).

This paper is organized as follows. In Section 2 we define cooperative games and review some results on the core of flow games and its stability. In Section 3, we characterize those networks that have uniform flow games with these properties, leading to linear time algorithms to decide them. We conclude with some open problems in Section 4.

## 2 Definitions

### 2.1 Graphs

A *flow network* is a directed graph  $G = (V, E; \omega)$ , where  $V$  is the vertex set,  $E$  is the edge set, and  $\omega : E \rightarrow \mathbb{R}^+$  is the edge capacity function. Let  $s$  (the *source*) and  $t$  (the *sink*) be two distinct vertices of  $G$ . W.l.o.g., we assume that every edge in  $E$  lies on some simple  $(s, t)$ -path.  $G$  is a *uniform flow network* if all edge capacities are equal. By scaling the capacities we can w.l.o.g. assume that the edge capacities are equal to one in this case.

If  $S$  and  $T$  partition the vertex set into two parts such that  $s \in S$  and  $t \in T$ , then the set  $C$  of edges going from a node in  $S$  to a node in  $T$  is an  $(s, t)$ -cut. The *capacity* of  $C$  is the sum of its edge capacities.  $C$  is a *minimal*  $(s, t)$ -cut if no proper subset of  $C$  is an  $(s, t)$ -cut.  $C$  is a *minimum*  $(s, t)$ -cut if it has smallest capacity among all  $(s, t)$ -cuts. We say a uniform network  $G$  is *cut-normal* if every minimal  $(s, t)$ -cut is already a minimum  $(s, t)$ -cut.

A directed graph is a *DAG* (*directed acyclic graph*) if it does not contain a directed cycle. A *2-terminal DAG* is a DAG with two vertices  $s$  and  $t$  such that the indegree of  $s$  and the outdegree of  $t$  are zero and every other vertex appears in at least one simple  $(s, t)$ -path. A 2-terminal DAG is *balanced* if the indegree of every vertex other than  $s$  and  $t$  equals its outdegree.

A *2-terminal directed series-parallel graph* (*2-DSPG*) is a directed graph with two distinguished vertices (*terminals*)  $s$  and  $t$  that is obtained inductively as follows. A basic 2-DSPG consists of two terminals  $s$  and  $t$ , connected by a directed edge  $(s, t)$ . If  $G_1$  and  $G_2$  are 2-DSPGs with terminals  $s_i$  and  $t_i$ ,  $i = 1, 2$ , then we can combine them in series by identifying  $t_1$  with  $s_2$  to obtain a 2-DSPG with terminals  $s_1$  and  $t_2$ , or in parallel by identifying  $s_1$  with  $s_2$  and  $t_1$  with  $t_2$  to obtain a 2-DSPG with terminals the combined vertex  $s_1/s_2$  and the combined vertex  $t_1/t_2$ .

### 2.2 Cooperative Games

A *cooperative (profit) game*  $\Gamma = (N, v)$  consists of a player set  $N = \{1, 2, \dots, n\}$  and a profit function  $v : 2^N \rightarrow \mathbb{R}$  with  $v(\emptyset) = 0$ . A *coalition*  $S$  is a non-empty subset of  $N$ , and  $v(S)$  represents the profit that can be achieved by the players in  $S$  without help of other players. The central problem in a cooperative game is how to allocate the total profit  $v(N)$  among the individual players in a ‘fair’ way. An *allocation* is a vector  $x \in \mathbb{R}^n$  with  $x(N) = v(N)$ , where  $x(S) = \sum_{i \in S} x_i$  for any  $S \subseteq N$ .

Different requirements for fairness, stability and rationality lead to different optimal allocations which are generally called *solution concepts*. The core is an important solution concept.

An allocation  $x \in \mathbb{R}^n$  is called an *imputation* if  $x_i \geq v(\{i\})$  for all players  $i \in N$ . Every player is happy in this case because he gets at least as much as he could expect from the profit function of the game. We denote by  $X(\Gamma)$  the set of imputations of  $\Gamma$ .

The *core*  $C(\Gamma)$  of  $\Gamma$  is the set of imputations where no coalition  $S$  has an incentive to split off from the grand coalition  $N$  and go their own way. Formally,  $C(\Gamma) = \{x \in \mathbb{R}^n \mid x(N) = v(N) \text{ and } x(S) \geq v(S) \text{ for all } S \subseteq N\}$ . A game is *balanced* if its core is not empty.

For a subset  $S \subseteq N$ , the *induced subgame*  $(S, v_S)$  on  $S$  has profit function  $v_S(T) = v(T)$  for each  $T \subseteq S$ . A cooperative game  $\Gamma$  is called *totally balanced* if all its subgames are balanced, i.e., all its subgames have non-empty cores.

### 2.3 Core Stability, Core Largeness, Extendability, and Exactness

In their classical work on game theory, von Neumann and Morgenstern [19] introduced the stable set which is very useful for the analysis of bargaining situations. Suppose that  $x$  and  $y$  are imputations. We say that  $x$  *dominates*  $y$  if there is a coalition  $S$  such that  $x(S) \leq v(S)$  and  $x_i > y_i$  for each  $i \in S$ . A set  $\mathcal{F}$  of imputations is *stable* if no two imputations in  $\mathcal{F}$  dominate each other, and any imputation not in  $\mathcal{F}$  is dominated by at least one imputation in  $\mathcal{F}$ . In particular, the core of a game is stable if for any non-core imputation  $y$  there is a core imputation  $x$  dominating  $y$ , i.e.,  $x(S) = v(S)$  and  $x_i > y_i$  for each  $i \in S$ .

There are three other concepts closely related to the core stability. Let  $\Gamma = (N, v)$  be an  $n$ -player cooperative game. The core of  $\Gamma$  is *large* if for every  $y \in \mathbb{R}^n$  satisfying  $y(S) \geq v(S)$ , for all  $S \subseteq N$ , there exists a core imputation  $x$  such that  $x \leq y$ .  $\Gamma$  is *extendable* if for every  $S \subseteq N$  and every core imputation  $y$  of the subgame  $(S, v_S)$  there exists a core imputation  $x \in C(\Gamma)$  such that  $x_i = y_i$  for all  $i \in S$ .  $\Gamma$  is called *exact* if for every  $S \subset N$  there exists  $x \in C(\Gamma)$  such that  $x(S) = v(S)$ .

Kikuta and Shapley [12] showed that a balanced game with a large core is extendable, and an extendable balanced game has a stable core. Sharkey [14] showed that a totally balanced game with a large core is exact. Biswas *et al.* [2] pointed out that extendability also implies exactness. Note that flow games are totally balanced.

## 2.4 Flow Games

Flow games were introduced by Kalai and Zemel [10, 11]. Consider a flow network  $G = (V, E; \omega)$ . In the corresponding *flow game*  $\Gamma = (E, v)$  on  $G$  each player controls one edge. The profit  $v(S)$  of a coalition  $S \subseteq E$  is the value of a maximum  $(s, t)$ -flow that only uses edges controlled by  $S$ . The flow game is *uniform* if the underlying network is uniform.

In this paper, we focus on uniform flow games. These games belong to the class of packing and covering games introduced by Deng *et al.* [4]. Kalai and Zemel [11] showed that flow games are totally balanced. Fang *et al.* [7] proved that the problem of testing membership in the core of a flow game is *co-NP*-complete. Deng *et al.* [4] showed that the core of a uniform flow game is exactly the convex hull of the indicator vectors of the minimum  $(s, t)$ -cuts of the network, which can be computed in polynomial time.

An edge  $e \in E$  is called a *dummy edge* if  $v(E \setminus \{e\}) = v(E)$ , i.e., removal of  $e$  does not change the value of the maximum  $(s, t)$ -flow. Sun *et al.* [16] showed that a uniform flow game has a stable core if and only if the network does not contain dummy edges. Based on this structural property they designed an  $O(|V|^2 \cdot |E|^2)$  time algorithm to decide core stability of uniform flow games. In Section 3 we will see that we can recognize graphs without dummy edges much faster. Note that dummy edges also play a role in the efficient computation of the nucleolus of flow games [3].

Sun *et al.* [16] also showed that for uniform flow games the concepts of exactness, extendability, and core largeness are equivalent, and that they are equivalent to the property of the network that every  $(s, t)$ -cut contains a minimum  $(s, t)$ -cut. It is easy to see that this is equivalent to being cut-normal.

**Lemma 1.** *A 2-terminal DAG with terminals  $s$  and  $t$  is cut-normal if and only if every  $(s, t)$ -cut contains a minimum  $(s, t)$ -cut.  $\square$*

In next section we show that the cut-normal uniform flow networks are exactly the balanced directed serial-parallel graphs with two terminals. This immediately implies a linear time algorithm to decide whether a uniform flow game is exact, extendable, and has a large core.

### 3 Efficient Algorithms

#### 3.1 Core Stability

Let  $G = (V, E)$  be a uniform flow network with source  $s$  and sink  $t$ . In this section we will give a linear time algorithm to decide core stability of the flow game on  $G$ .

**Theorem 2.**  *$G$  contains no dummy edge if and only if  $G$  is a balanced DAG.*

*Proof. If:* Suppose  $G$  is a balanced DAG. Let  $f$  be a maximum integer  $(s, t)$ -flow (which is also a maximum  $(s, t)$ -flow). Then,  $f$  pushes either zero or one unit of flow along each edge. Consider the subgraph  $G'$  of  $G$  of all edges  $e \in E$  with  $f(e) = 0$ . Since  $G$  is balanced and  $f$  satisfies the flow conservation property in every vertex except  $s$  and  $t$ ,  $G'$  is also balanced. As a subgraph of  $G$  it is also acyclical. Thus, if  $G'$  is not empty, there must be a simple  $(s, t)$ -path in  $G'$ . But then we can push one more unit of flow along this path, contradicting the maximality of  $f$ . Thus,  $G'$  is empty, i.e.,  $G$  contains no dummy edge.

**Only if:** The flow conservation property and the fact that all edges have capacity one imply that for each maximum flow  $f$  at least one edge incident to an unbalanced vertex is not used by  $f$ . Thus, every unbalanced graph contains dummy edges.

Suppose  $G$  contains a directed cycle  $C$ . Since we may w.l.o.g. assume that a maximum  $(s, t)$ -flow  $f$  does not have flow flowing around in cycles, at least one edge of  $C$  will not be used by  $f$ , i.e., it is a dummy edge.  $\square$

**Corollary 3.** *We can decide core stability of uniform flow games in linear time.*

*Proof.* Sun *et al.* [16] have shown that a uniform flow game has a stable core if and only if the network does not contain dummy edges.  $\square$

#### 3.2 Extendability, Exactness, and Core Largeness

In this section we will give a linear time algorithm to decide exactness, extendability, and core largeness of uniform flow games. Note that these properties imply core stability. In view of Theorem 2 we can therefore w.l.o.g. assume in this subsection that flow networks are balanced DAGs.

Two graphs are *homeomorphic* if they can be made isomorphic by inserting new vertices of degree two into edges, i.e., substituting directed edges by directed paths (which does not change the topology of the graph). Let  $H$  denote the graph shown on the left side of Fig. 1.

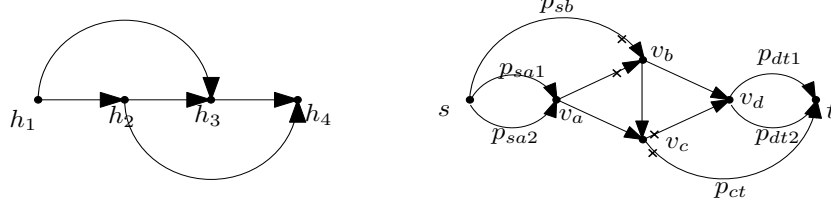


Fig. 1. Left: the graph  $H$ ; Right: the graph  $G_3$

**Theorem 4.** [6, 9, 17] *A 2-terminal DAG is a 2-DSPG if and only if it does not contain a subgraph homeomorphic to  $H$ .*  $\square$

We now give a characterization of balanced 2-DSPGs.

**Theorem 5.** *A balanced 2-terminal DAG is cut-normal if and only if it is a balanced 2-DSPG.*

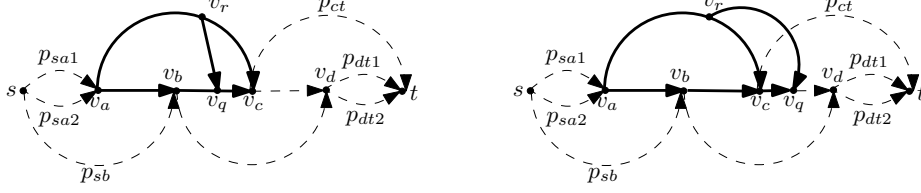
*Proof. If:* It is easy to see that balanced 2-DSPGs can be generated inductively as 2-DSPGs with the additional constraint that a combination in series step (where we identify the two terminals  $t_1$  and  $s_2$ ) requires the indegree of  $t_1$  being equal to the outdegree of  $s_2$ .

We now prove the claim by induction on  $|E|$ . If  $|E| = 1$ , the graph is  $(s, t)$ -normal. Suppose the statement holds for all balanced 2-DSPGs with fewer than  $|E|$  edges. If  $G$  was generated from  $G_1 = (V_1, E_1)$  and  $G_2 = (V_2, E_2)$  by combination in parallel, then any minimal  $(s, t)$ -cut  $C$  in  $G$  consists of a minimal  $(s_1, t_1)$ -cut  $C \cap E_1$  in  $G_1$  and a minimal  $(s_2, t_2)$ -cut  $C \cap E_2$  in  $G_2$ . By inductive hypothesis, these are minimum cuts. Thus,  $C$  is a minimum  $(s, t)$ -cut in  $G$ .

If  $G$  was generated from  $G_1$  and  $G_2$  by combination in series, then a minimal  $(s, t)$ -cut  $C$  in  $G$  is either a minimal (and thus minimum)  $(s_1, t_1)$ -cut in  $G_1$  or a minimal (and thus minimum)  $(s_2, t_2)$ -cut in  $G_2$ . Since the indegree of  $t_1$  equals the outdegree of  $s_2$  and cutting all edges incident to  $t_1$  ( $s_2$ ) defines a minimal  $(s_1, t_1)$ -cut in  $G_1$  (minimal  $(s_2, t_2)$ -cut in  $G_2$ ), a minimum  $(s_1, t_1)$ -cut in  $G_1$  has the same capacity as a minimum  $(s_2, t_2)$ -cut in  $G_2$ . Thus,  $C$  is a minimum  $(s, t)$ -cut in  $G$ .

**Only if:** Let  $G = (V, E)$  be a balanced 2-terminal DAG with terminals  $s$  and  $t$ . Let  $V = \{s = v_1, v_2, \dots, v_n = t\}$  be a topological ordering of  $G$ .

Let  $k$  denote the outdegree of  $s$ . If  $G$  is not a 2-DSPG, then we can construct a minimal cut of size larger than  $k$ , contradicting the assumption that  $G$  is cut-normal. By Theorem 4,  $G$  contains a subgraph homeomorphic to  $H$  which we denote by  $H(a, b, c, d, P_{ab}, P_{bc}, P_{cd}, P_{ac}, P_{bd})$ , where



**Fig. 2.** Cases (1) and (2) in the proof of Theorem 5

$v_a$  is the node corresponding to  $h_1$ ,  $v_b$  the node corresponding to  $h_2$ ,  $v_c$  the node corresponding to  $h_3$ ,  $v_d$  the node corresponding to  $h_4$ , and the vertex-disjoint (except at their endpoints) paths  $P_{ab}$ ,  $P_{bc}$ ,  $P_{cd}$ ,  $P_{ac}$  and  $P_{bd}$  correspond to the edges  $(h_1, h_2)$ ,  $(h_2, h_3)$ ,  $(h_3, h_4)$ ,  $(h_1, h_3)$ , and  $(h_2, h_4)$  in  $H$ , respectively. Among all subgraphs homeomorphic to  $H$  we choose one,  $G_H$ , with largest  $b$ .

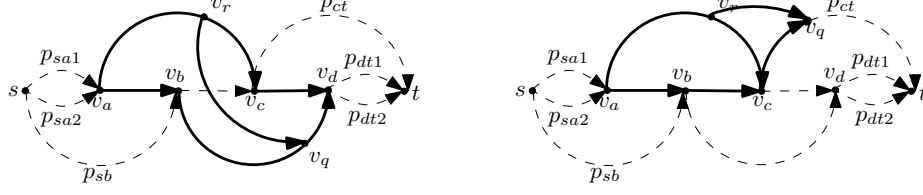
$G_H$  is not balanced, but since  $G$  is balanced we can find in  $G$  six pairwise edge-disjoint paths (they are also edge-disjoint with  $G_H$ )  $P_{sa1}$ ,  $P_{sa2}$ ,  $P_{sb}$ ,  $P_{ct}$ ,  $P_{dt1}$ , and  $P_{dt2}$  that we can add to  $G_H$  to obtain a balanced graph  $G_3$  (see Fig. 1, right side). Note that  $G_3$  is the union of three edge-disjoint  $(s, t)$ -paths  $P_{sa1} + P_{ab} + P_{bd} + P_{dt1}$ ,  $P_{sa2} + P_{ac} + P_{cd} + P_{dt2}$ , and  $P_{sb} + P_{bc} + P_{ct}$  (we use  $P_1 + P_2$  to denote the concatenation of paths  $P_1$  and  $P_2$ ).

Consider the  $(s, t)$ -cut  $C_b$  in  $G$  induced by the partition of  $V$  into  $\{v_1, \dots, v_{b-1}\}$  and  $\{v_b, \dots, v_n\}$ . Since  $G$  is a DAG and all vertices lie on some  $(s, t)$ -path, this is a minimal cut. If  $|C_b| > k$ , we have a minimal cut that is not a minimum cut. So assume  $|C_b| = k$ . We partition the edges of  $C_b$  into two classes,  $C_{b1}$  and  $C_{b2}$ . An edge  $e$  belongs to  $C_{b1}$  if every  $(s, t)$ -path containing  $e$  passes through  $v_c$ , otherwise it belongs to  $C_{b2}$ .

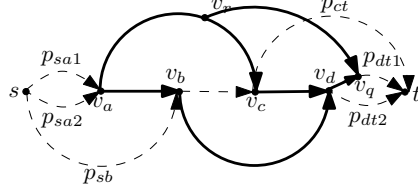
We will now show that  $C_{b1}$  is not empty. Specifically, we show it contains the edge  $e = P_{ac} \cap C_b$ . Assume there is an  $(s, t)$ -path  $P_e$  containing  $e$  but not  $v_c$ . Let  $v_r$  be the largest node in  $P_{ac} \cap P_e$ . Let  $v_q$  be the first node corresponding to a node in  $G_3$  that we encounter on  $P_e$  when starting to walk at  $v_r$  (such a node exists because  $P_e$  ends at  $t \in G_3$ ). Let  $P_{rq}$  denote the path from  $v_r$  to  $v_q$ .

Since  $e \in C_b$ ,  $r \geq b$ . Actually,  $r > b$  because  $v_b$  is not on  $P_{ac}$ . And  $r < c$  because  $v_r \in P_{ac}$  and  $v_c \notin P_e$ . But then there exists another  $H(a, r, v_{c'}, \dots)$  in  $G$ , a contradiction because  $r > b$ . To see this, we distinguish five cases, depending on the location of  $v_q$  (see Fig. 2–4). Let  $P_{xy}(i, j)$  denote the subpath of  $P_{xy}$  from  $v_i$  to  $v_j$ , where  $x, y \in \{a, b, c, d\}$ .

1.  $v_q \in P_{bc}$ :  $H(a, r, q, c, P_{ac}(a, r), P_{rq}, P_{bc}(q, c), P_{ab} + P_{bc}(b, q), P_{ac}(r, c))$ .



**Fig. 3.** Cases (3) and (4) in the proof of Theorem 5



**Fig. 4.** Case (5) in the proof of Theorem 5

2.  $v_q \in P_{cd}$ :  $H(a, r, c, q, P_{ac}(a, r), P_{ac}(r, c), P_{cd}(c, q), P_{ab} + P_{bc}, P_{rq})$ .
3.  $v_q \in P_{bd}$ :  $H(a, r, q, d, P_{ac}(a, r), P_{rq}, P_{bd}(q, d), P_{ab} + P_{bd}(b, q), P_{ac}(r, c) + P_{cd})$ .
4.  $v_q \in P_{ct}$ :  $H(a, r, c, q, P_{ac}(a, r), P_{ac}(r, c), P_{ct}(c, q), P_{ab} + P_{bc}, P_{rq})$ .
5.  $v_q \in P_{dt1}$  or  $v_q \in P_{dt2}$ :  $H(a, r, d, q, P_{ac}(a, r), P_{ac}(r, c) + P_{cd}, P_{dq}, P_{ab} + P_{bd}, P_{rq})$ .

Next, we show that  $|C_{b1}| < c_{\text{out}}$ , where  $c_{\text{out}}$  denotes the outdegree of  $v_c$  in  $G$ . Let  $U$  be a maximal set of edge-disjoint  $(s, t)$ -paths containing  $v_c$  that includes the path  $P_{sa1} + P_{ab} + P_{bc} + P_{cd} + P_{dt1}$ . Note that  $|U| \leq c_{\text{out}}$ . Clearly,  $C_{b1}$  is a subset of  $U \cap C_b$  by the definition of  $C_{b1}$ . Since the last edge of  $P_{ab}$  belongs to  $U \cap C_b$  but not to  $C_{b1}$ , it is even a proper subset, proving the claim

Let  $C$  be the set  $C_{b2}$  plus all edges outgoing from  $v_c$ .  $C$  is an  $(s, t)$ -cut because every  $(s, t)$ -path must either contain an edge in  $C_{b2}$  or the node  $v_c$ .  $C$  is a minimal  $(s, t)$ -cut because every outgoing edge of  $v_c$  is necessary because  $C_{b1}$  is not empty, and every edge in  $C_{b2}$  is necessary by definition. The size of  $C$  is  $|C| = |C_{b2}| + c_{\text{out}} > |C_{b2}| + |C_{b1}| = |C_b| = k$ . Thus,  $C$  is not a minimum  $(s, t)$ -cut, a contradiction. Therefore, the assumption that  $G$  is not a 2-DSPG must be wrong.  $\square$

**Corollary 6.** *We can test in linear time whether a balanced uniform flow network is cut-normal. Consequently, we can decide exactness, extendability, and core largeness of uniform flow games in linear time.*

*Proof.* We can test in linear time whether a DAG is balanced and whether it is a 2-DSPG [17]. By Theorem 5 this is equivalent to being cut-normal. By Lemma 1, a uniform flow network is cut-normal if and only if every  $(s, t)$ -cut contains a minimum  $(s, t)$ -cut. Sun *et al.* [16] have shown that this is equivalent to the flow game being exact, extendable, and having a large core.  $\square$

## 4 Open Problems

In this paper, we gave structural characterizations of exact, extendable, large-core and stable-core uniform flow games that can be tested in linear time. Currently, little is known about core stability of flow games on networks with arbitrary capacities. Although it is  $co\text{-}\mathcal{NP}$ -complete to decide whether an imputation belongs to the core this does not rule out the possibility that core stability can be decided efficiently. We leave it as an open problem.

## Acknowledgements

We would like to thank Mordecai Golin for his helpful discussions.

## References

1. T. Bietenhader and Y. Okamoto. Core stability of minimum coloring games. In *Proceedings of 30th International Workshop on Graph-Theoretic Concept in Computer Science (WG'04)*. Springer LNCS 3353, pages 389–401, 2004.
2. A.K. Biswas, T. Parthasarathy, J.A.M. Potters, and M. Voorneveld. Large cores and exactness. *Game and Economic Behavior*, 28:1–12, 1999.
3. X. Deng, Q. Fang, and X. Sun. Finding nucleolus of flow game. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'06)*, pages 124–131, 2006.
4. X. Deng, T. Ibaraki, and H. Nagamochi. Algorithmic aspects of the core of combinatorial optimization games. *Mathematics of Operations Research*, 24:751–766, 1999.
5. X. Deng and C.H. Papadimitriou. On the complexity of cooperative solution concepts. *Mathematics of Operations Research*, 19:257–266, 1994.
6. R. Duffin. Topology of series-parallel networks. *Journal of Mathematical Analysis and Applications*, 10:303–318, 1965.
7. Q. Fang, S. Zhu, M. Cai, and X. Deng. Membership for core of LP games and other games. In *Proceedings of the 7th Annual International Computing and Combinatorics Conference (COCOON'01)*. Springer LNCS 2108, pages 247–256, 2001.
8. K. Jain and R.V. Vohra. On stability of the core. Manuscript, 2006. <http://www.kellogg.northwestern.edu/faculty/vohra/ftp/newcore.pdf>

9. A. Jakoby, M. Liśkiewicz, and R. Reischuk. Space efficient algorithms for directed series-parallel graphs. *Journal of Algorithms* 60(2):85–114, 2006.
10. E. Kalai and E. Zemel. Totally balanced games and games of flow. *Mathematics of Operations Research*, 7:476–478, 1982.
11. E. Kalai and E. Zemel. Generalized network problems yielding totally balanced games. *Operations Research*, 30:998–1008, 1982.
12. K. Kikuta and L.S. Shapley. Core stability in  $n$ -person games. Manuscript, 1986.
13. W.F. Lucas. A game with no solution. *Bulletin of the American Mathematical Society*, 74:237–239, 1968.
14. W.W. Sharkey. Cooperative games with large cores. *International Journal of Game Theory*, 11:175–182, 1982.
15. T. Solymosi and T.E.S. Raghavan. Assignment games with stable cores. *International Journal of Game Theory*, 30:177–185, 2001.
16. X. Sun and Q. Fang. Core Stability of Flow Games. In *Proceedings of the 2005 China-Japan Conference on Discrete Geometry, Combinatorics and Graph Theory*. Springer LNCS 4381, pages 189–199, 2007.
17. J. Valdes, R.E. Tarjan, and E.L. Lawler. The recognition of series-parallel digraphs. In *Proceedings of the 11th Annual ACM Symposium on Theory of Computing (STOC'79)*, pages 1–12, 1979.
18. J.R.G. van Gellekom, J.A.M. Potters, and J.H. Reijnierse. Prosperity properties of TU-games. *International Journal of Game Theory*, 28:211–227, 1999.
19. J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behaviour*. Princeton University Press, Princeton, 1944.