MAGYAR TUDOMÁNYOS AKADÉMIA
Közgazdaság- és Regionális Tudományi Kutatóközpont
MTA KRTK KTI
Centre for Economic and Regional Studies
HUNGARIAN ACADEMY OF SCIENCES

MŰHELYTANULMÁNYOK DISCUSSION PAPERS

MT-DP – 2013/21

The Nucleolus of Directed Acyclic Graph Games

BALÁZS SZIKLAI

TAMÁS SOLYMOSI

TAMÁS FLEINER

A revised version of this paper can be found at the following link:
http://econ.core.hu/file/download/mtdp/MTDP1418.pdf

INSTITUTE OF ECONOMICS, CENTRE FOR ECONOMIC AND REGIONAL STUDIES,
HUNGARIAN ACADEMY OF SCIENCES BUDAPEST, 2013

Discussion papers
MT-DP – 2013/21

Institute of Economics, Centre for Economic and Regional Studies,
Hungarian Academy of Sciences

The Nucleolus of Directed Acyclic Graph Games

Authors:

Balázs Sziklai
junior research fellow
Institute of Economics
Centre for Economic and Regional Studies
Hungarian Academy of Sciences
E-mail: sziklai.balazs@krtk.mta.hu

Tamás Fleiner
associate professor
Department of Computer Science and Information Theory
Budapest University of Technology and Economics
E-mail: fleiner@cs.bme.hu

Tamás Solymosi
associate professor
Department of Operations Research and Actuarial Sciences
E-mail: tamas.solymosi@uni-corvinus.hu

July 2013

ISBN 978-615-5243-78-3
ISSN 1785 377X

The Nucleolus of Directed Acyclic Graph Games

Balázs Sziklai - Tamás Solymosi - Tamás Fleiner

Abstract

In this paper we consider a natural generalization of standard tree games where the underlying structure is a directed acyclic graph. We analyze the properties of the game and illustrate its relation with other graph based cost games. We show that although the game is not convex its core is always non-empty. Furthermore we provide a painting algorithm for large families of directed acyclic graph games that finds the nucleolus in polynomial time.

Keywords: Cooperative game theory, Directed acyclic graphs, Nucleolus

JEL classification: C71

Acknowledgements:

The author thanks the funding of the Hungarian Academy of Sciences under its Momentum Programme (LD-004/2010).

Irányított aciklikus gráfjátékok nukleolusza

Sziklai Balázs - Solymosi Tamás - Fleiner Tamás

Összefoglaló

Cikkünkben a standard fajátékok egy általánosítását vizsgáljuk, amelyben a mögöttes hálózati struktúra egy irányított aciklikus gráf. A játék alapvető tulajdonságainak ismertetése mellett bemutatjuk más költségjátékokhoz való viszonyát. Bizonyítjuk, hogy bár a játék nem konvex, a magja nem üres. Továbbá adunk egy festő algoritmust, amely az irányított aciklikus gráfok egy nagy részcsaládja esetén hatékonyan kiszámolja a játék nukleoluszát.

Tárgyszavak: kooperatív játékok, irányított aciklikus gráfok, nukleolusz

JEL kód: C71

The Nucleolus of Directed Acyclic Graph Games

Balázs Sziklai*† Tamás Solymosi ‡ Tamás Fleiner†

July 18, 2013

Abstract

In this paper we consider a natural generalization of standard tree games where the underlying structure is a directed acyclic graph. We analyze the properties of the game and illustrate its relation with other graph based cost games. We show that although the game is not convex its core is always non-empty. Furthermore we provide a painting algorithm for large families of directed acyclic graph games that finds the nucleolus in polynomial time.

Keywords: Cooperative game theory, Directed acyclic graphs, Nucleolus

JEL-codes: C71

1 Introduction

Standard tree games are one of the most well-studied class of cost allocation games. In its most basic form, we have a tree, where nodes represents players and there is a cost function defined on the edges. There is a special node the so called root of the tree. This node can be interpreted as the service provider. The aim of every player is to get connected to the root. The question is how to allocate the costs that arise from the construction of the edges. A more general problem is when the underlying structure is considered to be a directed acyclic graph. In such a network, players can have multiple routes to the root. Naturally, not all edges will be constructed in the end, but players that have more than one possible way to reach the root have more bargaining power when it comes down to sharing the cost.

*The author thanks the funding of the Hungarian Academy of Sciences under its Momentum Programme (LD-004/2010).

†Research was funded by OTKA grant K108383.

‡Research was funded by OTKA grant K101224.

Consider for example a group of towns that would like to connect themselves to a water reserve. Clearly not every town has to build a direct pipeline to the source. A possible solution is to connect the nearest towns with each other and then one of the towns with the reserve. The towns that are already connected to the water system can force the rest to pay some of their construction cost, otherwise they can close down the outgoing water flow. On the other hand, no town can be forced to pay more than the cost of directly connecting itself to the water reserve.

Similar problems arise frequently in real world and the corresponding game theoretical literature is vast. Airport games, irrigation games and minimum cost spanning tree games (MCST) are all variations of the same cost sharing problem described above [9, 12, 7]. Shortest path games, peer group games and highway games are also similar in their concept [3, 1, 4]. Each of these games have non-empty core, which makes the nucleolus an appealing solution for such problems[1].

The nucleolus was introduced by Schmeidler in 1969 [16] and quickly became popular although it has a reputation to be much more complicated than the Shapley-value. Indeed it is very hard to axiomatize the nucleolus and even harder to compute it for a given game. However, in the past few decades there has been significant progress in this second aspect. Kuipers showed that there exists an efficient algorithm to compute the nucleolus for convex games [11]. Meanwhile, researchers developed fast algorithms for the nucleolus of important families of cooperative games like standard tree games, assignment games and some special classes of minimum cost spanning tree games [14, 18, 7].

Proceeding by its definition, it would take exponential time to compute the nucleolus. In practice, this means it is impossible to calculate it even for moderate amount of players. While this is also true for the Shapley-value usually it is easier to implement the latter due to the many existing axiomatization. On the other hand the axiomatization of the nucleolus provided by Sobolev can rarely be applied [17]. Still there are many known results that describe the general structure of this solution and these can be utilized in a wide variety of characteristic function form games. The most comprehensive work is due to Maschler, Shapley and Peleg [13]. They not only illustrate the geometric properties of the nucleolus but also give a linear program that computes it. Although this program has exponentially many inequalities (one for each coalition) it can be solved efficiently if one knows which constrains are redundant. Huberman and later Granot, Granot and Zhu provided methods to identify the coalitions that correspond to non-redundant constrains [8, 6]. Another useful tool is the Kohlberg-criteria which makes it easy to decide if a given allocation is the nucleolus or not [10]. Finally Maschler, Potters and Reijnierse developed a so called painting algorithm that can be applied in many graph related games [14]. In

[1]Since the nucleolus is always in the core if the core is non-empty.

this paper, we will employ this latter approach and define a painting algorithm to compute the nucleolus. However the structure of the proof is different as directed acyclic graph games - unlike standard tree games - are not convex.

The structure of the paper is as follows. In the second section we introduce the game theoretical framework used in the paper. In the third section we formally define directed acyclic graph games. In the forth we briefly demonstrate the differences and similarities of airport, standard tree and minimum cost spanning tree games and directed acyclic graph games. In the fifth and sixth section we propose a network canonization process and describe its implications. Finally in the seventh and eighth section we present the painting algorithm and prove the it results in the nucleolus of the defined game.

2 Game theoretical framework

A *cooperative cost game* is an ordered pair (N, c) consisting of the player set $N = \{1, 2, \ldots, n\}$ and a characteristic cost function $c : 2^N \to \mathbb{R}$ with $c(\emptyset) = 0$. The value $c(S)$ represent how much cost coalition S must bear if they choose to act separately from the rest of the players. Let us denote a specific cost game by Γ. A cost game is said to be *convex* if its characteristic function is submodular i.e if.

$$c(S) + c(T) \geq c(S \cup T) + c(S \cap T), \quad \forall\ S, T \subseteq N.$$

A solution for a cost allocation game is a vector $x \in \mathbb{R}^N$. For convenience, we introduce the following notation $x(S) = \sum_{i \in S} x_i$ for any $S \subseteq N$, and instead of $x(\{i\})$ we write simply $x(i)$. A solution is called efficient if $x(N) = c(N)$ and individually rational if $x(i) \leq c(i)$ for all $i \in N$. The imputation set of the game $X(\Gamma)$ contains efficient and individually rational solutions, formally

$$X(\Gamma) = \{x \in \mathbb{R}^n \mid x(N) = c(N),\ x(i) \leq c(i)\ \textit{for all}\ i \in N\}.$$

Given an allocation $x \in \mathbb{R}^N$, we define the *excess* of a coalition S as

$$exc(S, x) := c(S) - x(S).$$

The core of the cost allocation game $\mathcal{C}(\Gamma)$ is a set-valued solution where all the excesses are non-negative.

$$\mathcal{C}(\Gamma) = \{x \in \mathbb{R}^N \mid x(N) = c(N),\ x(S) \leq c(S)\ \textit{for all}\ S \subseteq N\}.$$

Let $\theta(x) \in \mathbb{R}^{2^N}$ be the excess vector that contains the 2^n excess values in a non-decreasing order. We say that a vector $x \in \mathbb{R}^m$ *lexicographically precedes* $y \in \mathbb{R}^m$ (denoted

by $x \preceq_L y$) if either $x = y$ or there exists a number $1 \leq j < m$ such that $x_i = y_i$ if $i \leq j$ and $x_{j+1} < y_{j+1}$.

Definition 1. *The nucleolus is the vector of allocations of a game Γ that lexicographically maximizes $\theta(x)$ over a set $X_0 \subseteq \mathbb{R}^N$. In other words,*

$$\mathcal{N}(\Gamma, X_0) = \{x \in X_0 \mid \theta(y) \preceq_L \theta(x) \text{ for all } y \in X_0\}$$

It is well known that if $X_0 = X(\Gamma)$, then the nucleolus of a game is a single-valued solution and it is in the core if the core is non-empty. Throughout the paper we assume X_0 to be the set of imputations and write $\mathcal{N}(\Gamma)$ instead of $\mathcal{N}(\Gamma, X(\Gamma))$.

Proceeding by its definition it would take exponential time to compute the nucleolus of a cooperative game, since we would have to compare the 2^N dimensional excess vectors with each other. In order to make this task manageable we identify the redundant coalitions which do not play any role in this process.

Definition 2. *Let $\Gamma^{\mathcal{F}} = (N, \mathcal{F}, c)$ be a cooperative game with coalition formation restrictions, where $\mathcal{F} \subseteq 2^N$ consists of all permissible coalitions. Then $\mathcal{F}$ is called a characterization set for the nucleolus of the game $\Gamma = (N, c)$ with respect to $X(\Gamma)$, if $\mathcal{N}(\Gamma^{\mathcal{F}}) = \mathcal{N}(\Gamma)$.*

Granot,Granot and Zhu constructed a sequential LP process whose input is a characterization set and the values of the cost function for coalitions contained therein, and the output is the nucleolus of the game [6]. They showed that if the size of the characterization set is polynomially bounded in the total number of players, then the nucleolus of the game can be computed in strongly polynomial time.

A collection of coalitions $\mathcal{B} \subseteq 2^N$ is said to be balanced if there exists positive weights $\lambda_S,\ S \in \mathcal{B}$ such that$\sum_{S \in \mathcal{B}} \lambda_S e_S = e_N$, where e_S denotes the indicator vector of coalition S. Applying the Kholberg/Sobolev criterion [10] to the games with coalition formation restrictions yields the following theorem.

Theorem 3. *Let $\mathcal{F}$ be a characterization set and x be an imputation of the game Γ with $\mathcal{C}(\Gamma) \neq \emptyset$. Then $x = \mathcal{N}(\Gamma)$ if and only if for all $y \in \mathbb{R}$ the collection $\{S \in \mathcal{F} \mid exc(S, x) \leq y\}$ is balanced or empty.*

For proof see [15]. Although the number of possible coalitions in an n-player game is exponential, only a fraction of them is needed to characterize the nucleolus.

Definition 4 (Essential coalitions)**.** *Let S be a coalition, such that it can be written as a disjoint union of S_1 and S_2 i.e. $S = S_1 \dot{\cup} S_2$. If for such a decomposition it is true that $c_\Gamma(S) \geq c_\Gamma(S_1) + c_\Gamma(S_2)$ then S is not an essential coalition.*

Huberman showed that if the core of the game is non-empty then the essential coalitions form a characterization set for the nucleolus [8]. This observation helps us to eliminate large inessential coalitions but not the small ones. By definition, the singleton coalitions will always be essential in every game. Applying Huberman's definition in the dual game we can sort out the small but inessential coalitions.

Definition 5 (Dually essential coalitions)**.** *For a given coalition S let S_1 and S_2 to be such coalitions, that $S_1 \cap S_2 = S$ and $S_1 \cup S_2 = N$. If for such a decomposition it is true that $c(S) \geq c(S_1) + c(S_2) - c(N)$ then S is not a dually essential coalition.*

In case of DAG-games, the set of dually essential coalitions will prove to be more effective tool to determine the nucleolus.

Theorem 6. *If $\mathcal{C}(\Gamma) \neq \emptyset$, then the dually essential coalitions form a characterization set for $\mathcal{N}(\Gamma)$.*

A formal proof of Theorem 6 can be obtained by copying the arguments of [8], but it also follows from [2] (see Theorem $1/iv$).

3 Definition of the game

A directed acyclic graph network $\mathcal{D}$ is given by the following:

- $G(V, E)$ is a directed acyclic graph, with a special node - the so called *root* of G, denoted by $\mathbf{r}$ - such that from each other node of G there leads at least one directed path to the root. G is considered to be a simple graph, i.e. it has no loops or parallel edges.

- There is a cost function $a(e) : E \to \mathbb{R}^+ \cup \{0\}$, that assigns a non-negative real number to each edge. This value is regarded as the construction cost of the edge.

- There is a player set N and a correspondence between N and the node set of G. If player i is assigned to node $\mathbf{p}$ we say player i *resides* at $\mathbf{p}$. The residents of a subgraph T denoted by $N(T)$.

For a subgraph T, $V(T)$ denotes the node set of T. Similarly $E(T)$ denotes its edge set, while $E_{\mathbf{p}}$ is used for the set of edges that goes out from node $\mathbf{p}$. We call nodes that have one outgoing edge as *passages* while nodes that have more outgoing edges are *junctions*.

For a subgraph T we define its construction cost $C(T)$ as the total cost of the edges in T i.e. $C(T) = \sum_{e \in E(T)} a(e)$. A path whose end point is the root is called a *rooted path*. A connected subgraph of G that is a union of rooted paths is called a *trunk*. For

each coalition S let T_S denote the set of subgraphs that have maximum number of edges among the cheapest trunks that connects all players of S to the root. Furthermore we say that subgraph T *corresponds* to the node set B if $T \in T_{N(B)}$. Throughout the paper we will identify trunks with node sets in this sense.

The characteristic function of the cost allocation game that is associated to $\mathcal{D}$ is defined as follows.

$$c_{\mathcal{D}}(S) \stackrel{def}{=} C(T) \qquad T \in T_S. \tag{1}$$

The definition is motivated by the fact that by leaving the grand coalition the players of S need not to pay more than $c_{\mathcal{D}}(S)$ to get connected to the root. As any member in T_S has the same construction cost $c_{\mathcal{D}}(S)$ is well-defined. However any DAG-network can be transformed in such way T_N contains a single element while $c_{\mathcal{D}}(N)$ remains unchanged. As we will see the canonization of a DAG-network will play an important part in the proof. Let us denote by $\Gamma_{\mathcal{D}}$ the cost game that is induced by $c_{\mathcal{D}}$ i.e. $\Gamma_{\mathcal{D}} = (N, c_{\mathcal{D}})$. For convenience sake we write simply Γ instead of $\Gamma_{\mathcal{D}}$ from now on.

4 Comparison of graph related cost games

There are many similar graph related cost games. Airport games, standard tree games, DAG-networks, and MCST games have the same setup, namely they are based on a rooted graph, where players - who are located on the nodes - would like to share the construction cost of the edges. Table 1 summarizes the differences of these games, while Figure 1 shows how they are related with each other.

	Graph	Edges	Players/node	Convexity
Airport Games	chain	(un)directed	$0-n$	convex
Standrad Tree Games	tree	(un)directed	$0-n$	convex
DAG-games	connected DAG	directed	$0-n$	not convex
MCST Games	connected	undirected	1	not convex

Table 1: Comparison of graph related cost games

In case of airport games and standard tree games the edges can be considered both directed or undirected. For both of these games the nucleolus can be computed very fast, in $\mathcal{O}(n \cdot \log n)$ time [14]. On the other hand to compute the nucleolus of a MCST game in general is NP-hard [5]. As we will see in case of large families of DAG-networks the nucleolus can be computed in $\mathcal{O}(n^3)$ time.

Figure 1: Venn-diagram of graph related cost games

5 Canonization process

We say that $\mathcal{D}$ is in canonical form if the following properties are fulfilled.

P1 Each junction has an outgoing zero cost edge,

P2 There resides a player in each passage.

P3 Each edge is used at least by one coalition.

To transform a DAG-network into a form where **P1** property is fulfilled we have to perform the following procedure for each node $\mathbf{p} \in V$ such that $|E_{\mathbf{p}}| \geq 2$ and $\min_{e \in E_{\mathbf{p}}} a(e) = \alpha_{\mathbf{p}} > 0$.

1. Introduce a new node $\mathbf{p}'$ with the same edge set as $\mathbf{p}$ but reduce the cost of the edges by $\alpha_{\mathbf{p}}$.

2. Erase all the edges that goes out from $\mathbf{p}$.

3. Finally introduce a new edge from $\mathbf{p}$ to $\mathbf{p}'$ with cost $\alpha_{\mathbf{p}}$.

This is an equivalent transformation in the sense that the construction cost of T_S unchanged for any coalition S.

If $\mathbf{p}$ is a passage where no player resides and $\mathbf{p}$ has only one ingoing edge then it can be omitted from the network. The ingoing and outgoing edge of $\mathbf{p}$ can be replaced by a single edge with an aggregated construction cost. Needless to say that this procedure does not change the costs of the T_S trunks either. If a passage has more ingoing edge then this transformation can not be applied. Therefore the **P2** property pose a restriction on the DAG-networks that can be canonized. According to **P2** there has to live at least one player in each node that has one outgoing edge and two or more ingoing edges. In the following we will assume that the network satisfies this property.

Finally edges not used in any of the T_S subgraphs can be deleted, since they do not affect the characteristic function. Figure 2 shows an example of the canonization process.

As we mentioned before the canonization of a DAG-network does not change the characteristic function of the corresponding cost game. Although canonization also ensures

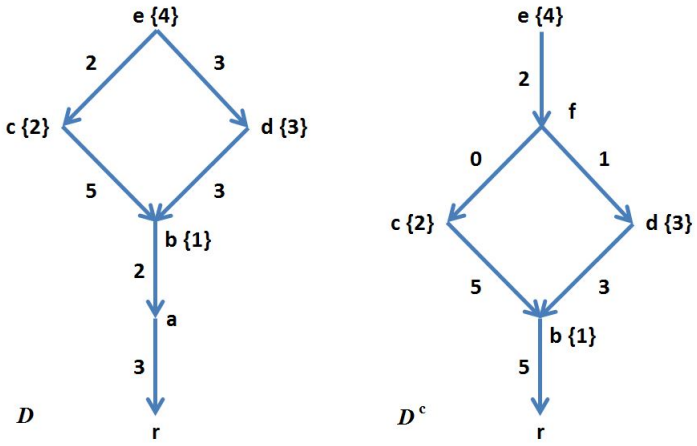

Figure 2: A DAG-network with player set $N = \{1, 2, 3, 4\}$ before and after canonization (the numbers in braces indicate the players that reside in the particular node).

that T_N contains only a single element, this cannot be said in general about other trunks. In the following we will assume that T_S contains only a single trunk for any coalition S. This can always be arranged by perturbing the positive edge costs. For convenience we will refer to T_S as this unique trunk that has maximum number of edges among the cheapest trunks that connects all members of S to the root.

6 Some consequences of canonization and further notations

For each node $\mathbf{p}$ the cheapest edges in $E_{\mathbf{p}}$ are called T_N-edges. The name comes from the fact that (if **P2** holds) an edge is a T_N-edge if and only if it is an element of $E(T_N)$. If $e, e' \in E_{\mathbf{p}}$, e is a T_N-edge and $a(e') > a(e)$, then e' is called a *shortcut*. If there exists a shortcut between $\mathbf{p}$ and $\mathbf{q}$ it is always cheaper than any alternative path between the two due to **P3**. If $e, e' \in E_{\mathbf{p}}$ are T_N-edges then the construction cost of both e and e' is zero (this is a consequence of **P1**).

The subgraph associated to the grand coalition (T_N) holds special importance. First this is the graph that will be constructed in the end. All the other edges are only good for improving the bargaining positions of certain players. Note that T_N is not necessarily a tree as it may contain some additional zero-edges. Secondly T_N induces a partial order $\prec$ on the nodes. We say that $\mathbf{q}$ is a descendant of $\mathbf{p}$ if $\mathbf{p}$ can be reached from $\mathbf{q}$ by using only T_N-edges, we denote this by $\mathbf{p} \prec \mathbf{q}$. In such cases we also say that $\mathbf{p}$ is an ancestor of $\mathbf{q}$. Node $\mathbf{p}$ is a *direct ancestor* of $\mathbf{q}$ if $\mathbf{p} \prec \mathbf{q}$ and there is a T_N-edge between them. Nodes that have no descendants are called *leafs*. The node set that contains $\mathbf{p}$ together with its descendants is called a *branch* and denoted by $B_{\mathbf{p}}$. A $B_{\mathbf{q}}$ branch is called a *subbranch* of

$B_{\mathbf{p}}$ if $\mathbf{q} \in B_{\mathbf{p}}$.

Sometimes we are interested only in some of the descendants of $\mathbf{p}$ therefore we cut off some of the subbranches of $B_{\mathbf{p}}$. Since there can be more than one 'trimmed' branch originating from a given node we will denote a specific branch by $B_{\mathbf{p}}^{Q} \stackrel{def}{=} B_{\mathbf{p}} \setminus \cup_{\mathbf{q}\in Q} B_{\mathbf{q}}$ meaning we consider the node set that contains $\mathbf{p}$ and the descendants of $\mathbf{p}$ but not Q and the descendants of Q. We say that $B_{\mathbf{p}}^{Q}$ is a *proper branch* if deleting $B_{\mathbf{p}}^{Q}$ along with the in and outgoing edges from G the root can be still reached from any node on a directed path.

Let us illustrate the above introduced notations with some examples. Consider again the canonized DAG-network $\mathcal{D}^c$ depicted in Figure 2. The only shortcut in $\mathcal{D}^c$ is the one that connects node $\mathbf{f}$ with node $\mathbf{d}$. All the other edges are T_N-edges. The branch $B_{\mathbf{d}}$ contains only one node, $\mathbf{d}$ since $\mathbf{d} \not\prec \mathbf{f}$ (i.e. $\mathbf{d}$ is a leaf). Furthermore it is a proper branch as removing $\mathbf{d}$ together with the in- and outgoing edges the graph remains connected. Finally the trunk that corresponds to the node set $B_{\mathbf{r}} \setminus B_{\mathbf{c}}^{\mathbf{f}}$ is $T_{\{1,3,4\}}$ and $c_{\mathcal{D}^c}(\{1,3,4\}) = 11$.

Figure 3 shows an example when the submodularity of the characteristic function is not satisfied.

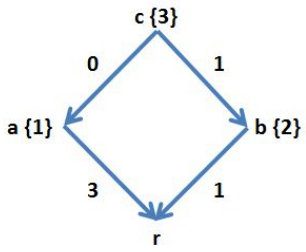

Figure 3: The supermodularity of the characteristic function

Let $S = \{1,3\}$ and $T = \{2,3\}$, then

$$3 + 2 = c(S) + c(T) < c(S \cup T) + c(S \cap T) = 4 + 2,$$

hence we conclude that DAG-games are indeed not convex.

Finally let $\mathbf{p}$ be a direct descendant of the root. Then a trunk that corresponds to $\{\mathbf{r}\} \cup B_{\mathbf{p}}$ is called a *base branch*. Note that the excess of the coalition formed by residents of any base branch is zero for any core allocation. This is not necessarily true after removing some of the subbranches of a base branch. The next lemma describes the structure of the trimmed base branches that preserve this property.

Lemma 7. *Let $B = \{\mathbf{r}\} \cup B_{\mathbf{p}}^{Q}$ be a proper base branch, such that for every $\mathbf{q} \in Q$ there leads a zero edge from $\mathbf{q}$ to $V \setminus B_{\mathbf{p}}^{Q}$. Then $exc(N(B), x) = exc(N \setminus N(B), x) = 0$ for any $x \in \mathcal{C}(\Gamma_{\mathcal{D}})$.*

Proof. As any player in $N \setminus N(B)$ can reach the root using only T_N edges the total cost can be partitioned.

$$\begin{aligned} c(N) &= c(N(B)) + c(N \setminus N(B)) \\ c(N) - x(N) &= c(N(B)) - x(N(B)) + c(N \setminus N(B)) - x(N \setminus N(B)) \\ 0 &= exc(N(B), x) + exc(N \setminus N(B), x) \end{aligned}$$

As the excesses are non-negative for any core allocation it follows that $exc(N(B), x) = exc(N \setminus N(B), x) = 0$. □

7 The painting algorithm $(PAINT)$

Now we introduce a polynomial time algorithm that computes the nucleolus. The procedure resembles to Maschler, Potters and Reijniers [14], but the general idea comes from [13], where the lexicographic center of a game is reached by 'pushing hyperplanes' with unit speed.

The following definitions will be useful. The *closest common ancestor* of a node set U is the unique node $\mathbf{p} \in V$ for which is true that $\mathbf{p} \preceq \mathbf{u}$ for any $\mathbf{u} \in U$ and there is no other $\mathbf{q} \in V$ such that $\mathbf{q} \preceq \mathbf{u}$ for any $\mathbf{u} \in U$ and either $\mathbf{p} \prec \mathbf{q}$ or $\mathbf{p}$ and $\mathbf{q}$ are not ordered. The closest common ancestor of node set U is denoted by $cca(U)$. Furthermore we denote by $Z(\mathbf{p})$ the node set that can be reached from node $\mathbf{p}$ with zero cost (therefore $\mathbf{p} \in Z(\mathbf{p})$ as well).

The painting algorithm consist of cycles. Only those players are participating in the current cycle who can not reach the root with a zero cost path. Each cycle begins with the canonization of the network. Then we repeat the following steps.

1. One player in each passage travels to its parent node.
2. Players currently located in $\mathbf{p}$ start to paint all the edges that goes out from $Z(\mathbf{p})$.
3. If a junction $\mathbf{p}$ has more than one zero-edge the players located at the junction do not paint the T_N-edges of their ancestors instead they start to paint the edges of $cca(Z(\mathbf{p}))$. They will paint however all the shortcuts that originate from any path that connects $\mathbf{p}$ to $cca(Z(\mathbf{p}))$.
4. Non-zero edges whose end point is a node $\mathbf{p}$ for which it is true that $\mathbf{r} = cca(Z(\mathbf{p}))$ are painted by $+1$ player, called the shadow player.
5. Players paint with unit speed i.e. when k players are painting an edge, they paint k unit of road upon one unit of time.

6. Stop when an edge is fully painted i.e. its cost becomes zero. Players return to their residence.

At time t_i the i^{th} cycle finishes. We distribute $t_i - t_{i-1}$ payoff among the players that participated in the painting of this phase (where $t_0 = 0$). The algorithm stops when the cost of all the edges are zero. Let us denote the i^{th} cycle by $\mathcal{P}_i$. Furthermore let z be the allocation that is produced by the algorithm.

Let us illustrate the algorithm with an example. Consider the canonized DAG-network $\mathcal{D}$ depicted in Figure 4. The player-set consist of four players residing in nodes $\mathbf{a}, \mathbf{b}, \mathbf{c}$ and $\mathbf{e}$. The cheapest way to connect all the players to the root is to build the edges e_1, e_3, e_4, e_5 and e_7, therefore $c_{\mathcal{D}}(N) = 13$. The only shortcuts are e_2 and e_6. As we will see these edges significantly shorten the time player 3 and 4 spends with painting.

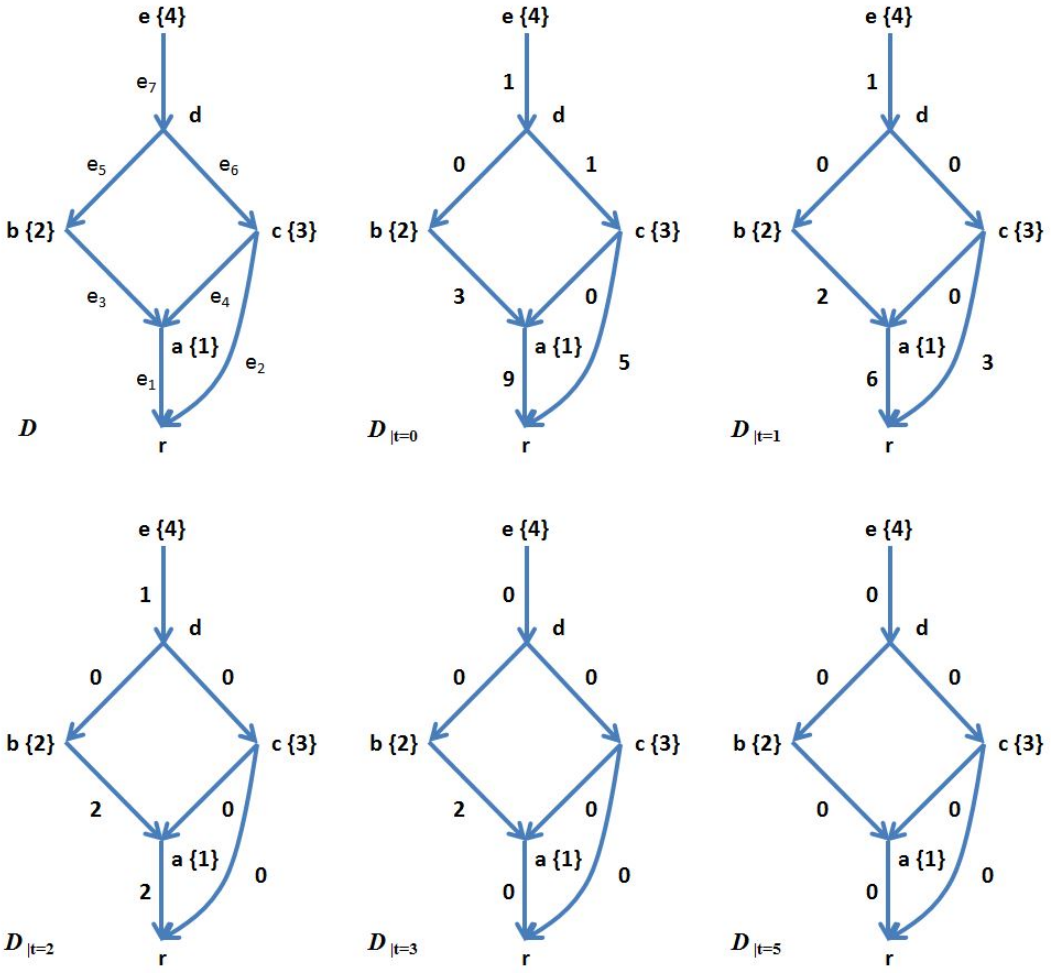

Figure 4: The painting algorithm.

At time $t = 0$ player 4 travels forward to the ancestor node of its residence and starts to paint the edges outgoing from $Z(\mathbf{d})$ i.e. e_3 and e_6. Similarly player 3 paints all the edges that goes out from $Z(\mathbf{c})$ thus e_1 and e_4. Player 2 travels forward to node $\mathbf{a}$ and paints e_1. Finally player 1 travels to the root and does nothing. However as $\mathbf{r} = cca(\mathbf{r})$

the edges e_1 and e_2 are painted by shadow players. As a result at time $t = 1$ the cost of e_3 and e_6 is reduced by one unit while the cost of e_1 and e_2 changes to 6 and 3 respectively. Since e_6 becomes a zero-edge the first cycle finishes and the players return to their home.

At time $t = 1$ player 4 travels again forth to node **d**. As **d** became a junction with more than one outgoing zero-edge, player 4 starts the painting from the closest common ancestor of $Z(\mathbf{d})$ that is node **a**. Player 4 will also paint the shortcuts originating from any path between **d** and **a**. Thus beyond e_1 he will paint e_2 as well. The other players paint the same edges as in the first cycle. As a result e_2 is painted by three players while e_1 is painted by four.

At time $t = 2$ the players finish to paint e_2. Player 3 can reach the root with a zero-cost path therefore he does not participate in the painting process anymore. The closest common ancestor of $Z(\mathbf{d})$ changes from **a** to **r** therefore player 4 travels forward to the root and e_7 is painted by a shadow player. At time $t = 3$ the cost of both e_1 and e_7 becomes zero. The only player who is participating in the remaining phase of the painting process is player 2. At $t = 5$ each player is connected to the root with a zero-cost path and the algorithm stops. The final allocation obtained this way is $x = (3, 5, 2, 3)$. It is easy to check (e.g. with the Kohlberg-criteria) that this is indeed the nucleolus of the cost allocation game corresponding to $\Gamma_{\mathcal{D}}$.

It is also easy to see that the running time of $PAINT$ is polynomial. Let $|V| = m$ and $|E| = l$. Then the number of operations needed to canonize the network or to run a cycle of $PAINT$ is a linear function of m. In every cycle at least one edge becomes a zero-edge. Therefore $PAINT$ stops in $O(m \cdot l)$ time. Which - considering that G is a directed acyclic graph - equals to $O(m^3)$.

8 Calculating the nucleolus

In our first lemma we show that the core of a DAG-network game is never empty.

Lemma 8. *$\mathcal{C}(\Gamma_{\mathcal{D}}) \neq \emptyset$ for any DAG-network $\mathcal{D}$.*

Proof. Let $N(\mathbf{p})$ be a set that collects the residents of node **p**. Define an allocation x such that for each player $i \in N$ let $x(i) = \frac{a(e_{\mathbf{p}})}{|N(\mathbf{p})|}$ where $i \in N(\mathbf{p})$ and $e_{\mathbf{p}}$ is one of the outgoing T_N edges of **p**. It is easy to see that x is a core allocation. Let $V^\star \subseteq V$ denote the set of nodes where at least one player resides in G.

$$x(N) = \sum_{i \in N} x(i) = \sum_{\mathbf{p} \in V^\star} |N(\mathbf{p})| \cdot \frac{a(e_{\mathbf{p}})}{|N(\mathbf{p})|} = \sum_{\mathbf{p} \in V^\star} a(e_{\mathbf{p}})$$

Therefore x covers the construction cost of one of the cheapest outgoing edges for each node where at least one player resides. Nodes where no player resides can only be

junctions, which have an outgoing zero edge. It follows that $x(N) = c_{\mathcal{D}}(N)$. On the other $x(S) \leq \sum_{\mathbf{p} \in V(T_S)} a(e_{\mathbf{p}}) \leq C(T_S) = c_{\mathcal{D}}(S)$ for any $S \subseteq N$. □

Note that Lemma 8 holds regardless property **P2** is satisfied in a given network or not.

Now we can identify the dually essential coalitions in case of DAG-network games. A coalition S is said to be *saturated* if $i \in S$ whenever $c(S) = c(S \cup \{i\})$. The closure of S is a saturated coalition $\bar{S}$ for which $S \subseteq \bar{S}$, $T_S = T_{\bar{S}}$. Note that $\bar{S}$ is unique due to the uniqueness of $T_{\bar{S}}$, and $S = \bar{S} \iff \bar{S} = N(V(T_S))$. Granot, Granot and Zhu showed that saturated coalitions form a characterizing set for the nucleolus.

Lemma 9. *In a DAG-network game dually essential coalitions are either saturated or consist of $n-1$ players.*

Proof. Let S be a coalition with at most $n-2$ players such that $S \neq \bar{S}$. Then there exists $i \in N \setminus S$ such that $c_{\mathcal{D}}(S) = c_{\mathcal{D}}(S \cup \{i\})$. Let $S_1 := S \cup \{i\}$ and $S_2 := N \setminus \{i\}$. Then $S_1 \cup S_2 = N$ and $S_1 \cap S_2 = S$ therefore we can use Definition 5 since

$$\begin{aligned} c_{\mathcal{D}}(S) &\geq c_{\mathcal{D}}(S_1) + c_{\mathcal{D}}(S_2) - c_{\mathcal{D}}(N) \\ c_{\mathcal{D}}(S) &\geq c_{\mathcal{D}}(S) + c_{\mathcal{D}}(N \setminus \{i\}) - c_{\mathcal{D}}(N) \\ c_{\mathcal{D}}(N) &\geq c_{\mathcal{D}}(N \setminus \{i\}) \end{aligned}$$

i.e. the cost criterion of dual essentiality applies as well. □

Saturated coalitions incorporate every player of the trunk on which they reside. This leads us to the following observation.

Observation 10. *Every trunk that corresponds to a saturated coalition $S \subset N$ can be obtained by deleting some proper subbranches from G. Formally*

$$V(T_S) = B_{\mathbf{r}} \setminus \cup_{i=1}^{k} B_{p_i}^{Q_i}.$$

Furthermore the origins of the deleted branches have non-zero T_N-edges i.e. $E_{p_i} = \{e_{p_i}\}$ and $a(e_{p_i}) > 0$ for $i = 1, 2, \ldots, k$.

The first part of the observation - that each saturated trunk can be obtained by cutting some proper branches off from G - is trivial. The second part that these branches connect to the trunk by a non-negative T_N-edge follows from the definition of T_S. Let us remind the reader that T_S is the subgraph that has *maximum number of edges* among the cheapest trunks that connect S to the root. Therefore any node $\mathbf{p}$ that connects to the trunk by a zero-edge by definition is included in T_S even if no player of S resides at $\mathbf{p}$.

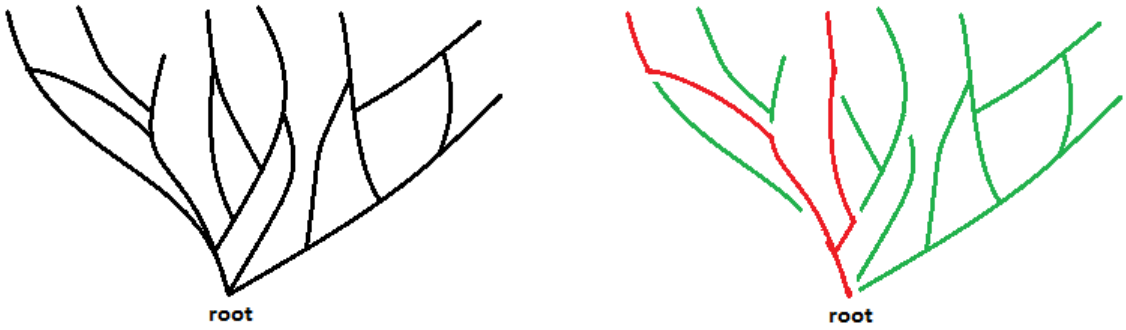

Figure 5: Any saturated trunk can be obtained by deleting some subbranches.

The following extension of the cost function will be useful. We define $\tau(Q, S)$ as the cost of the edges in T_S that goes out from node set Q i.e.

$$\tau(Q, S) \stackrel{def}{=} \sum_{e \in (\cup_{q \in Q} E_q) \cap E(T_S)} a(e).$$

Theorem 11. *The dually essential coalitions of the cost game $\Gamma_{\mathcal{D}}$ are the coalitions with $n-1$ player and saturated coalitions whose trunks correspond to node sets of the form $B_{\mathbf{r}} \setminus B^U_{\mathbf{q}}$ where $B^U_{\mathbf{q}}$ is a proper branch with $E_{\mathbf{q}} = \{e_{\mathbf{q}}\}$ and $a(e_{\mathbf{q}}) > 0$.*

Proof. We have already seen by Lemma 9 that only saturated coalitions are dually essential. By Observation 10 we know that saturated coalitions can be generated by removing subbranches from $B_{\mathbf{r}}$. The one thing we have to prove is that coalitions that correspond to trunks that have more missing subbranches are dually inessential. Let S be such a coalition for which $V(T_S) = B_{\mathbf{r}} \setminus \cup_{i=1}^{k} B^{Q_i}_{p_i}$ where $k \geq 2$. As $\mathcal{D}$ is in canonical form there resides at least one player in each of the subbranches. Furthermore we choose a representation of T_S where the node set Q_k in the branch $B^{Q_k}_{p_k}$ is either empty or a subset of $V(T_S)$. This can always be arranged since if there is a node $Q_k \ni \mathbf{u} \notin V(T_S)$ then $\mathbf{u} \in B^{Q_i}_{p_i}$ for some i. Now cut off $\mathbf{u}$ and the ancestors of $\mathbf{u}$ from $B^{Q_i}_{p_i}$ and attach them to $B^{Q_k}_{p_k}$. Repeat this process till every element in Q_k is in $V(T_S)$ or Q_k becomes empty.

For convenience sake let us introduce the following notation $B_1 = \cup_{i=1}^{k-1} B^{Q_i}_{p_i}$ and $B_2 = B^{Q_k}_{p_k}$. Then let $S_1 = N \setminus N(B_1)$ and $S_2 = N \setminus N(B_2)$. In this way $S_1 \cup S_2 = N$ and $S_1 \cap S_2 = S$. To prove that $c_{\mathcal{D}}(S) \geq c_{\mathcal{D}}(S_1) + c_{\mathcal{D}}(S_2) - c_{\mathcal{D}}(N)$ holds as well it is enough to show that the following two inequalities are true.

$$c_{\mathcal{D}}(S_1) \leq c_{\mathcal{D}}(S) + \tau(B_2, N) - \tau(Q_k, S) \tag{2}$$

$$c_{\mathcal{D}}(S_2) \leq c_{\mathcal{D}}(N) - \tau(B_2, N) + \tau(Q_k, S) \tag{3}$$

Note that it takes at most $\tau(B_2, N)$ to connect the players residing at B_2 to T_S. As $B^{Q_k}_{p_k}$ is a proper branch it follows that the nodes in Q_k are junctions. Since the nodes in

Q_k are direct ancestors of some nodes in B_2 they are connected with zero-edges. Therefore we can save at least $\tau(Q_k, S)$ amount of cost by connecting Q_k through the branch B_2 and not through the edges in $\cup_{\mathbf{q}\in Q_k} E_q \cap E(T_S)$. It is possible that aside from Q_k there are other nodes that can reach the root in a cheaper way using the edges of B_2, but no nodes of $V(T_S)$ is forced to take a more expensive path. Summarizing the above findings we gather that

$$c_{\mathcal{D}}(S_1) \leq c_{\mathcal{D}}(S) + \tau(B_2, N) - \tau(Q_k, S)$$

We can estimate $c_{\mathcal{D}}(S_2)$ by keeping track how the cost changes as we swift from T_N to T_{S_2}. As $N(B_2)$ are not in S_2 we can delete B_2 and subtract $\tau(B_2, N)$ amount of cost from $c_{\mathcal{D}}(N)$. Deleting B_2 from T_N only the direct descendants of B_2 get disconnected. Therefore the only nodes that are not connected to the root are Q_k and their descendants. The cost of reconnecting Q_k is at most $\tau(Q_k, S)$. The reason for this is that building the exact same edges that we deleted in case of S_1 are sufficient. Take any node in Q_k. As we builded $\cup_{\mathbf{q}\in Q_k} E_q \cap E(T_S)$ it has an outgoing edge. Lets say it points to a node $\mathbf{u_0}$. If no player resides at $\mathbf{u_0}$ then it is a junction by property **P2** hence it has an outgoing zero edge that points to a node $\mathbf{u_1}$. As there are finite number of nodes in $\mathcal{D}$ eventually we will reach a node $\mathbf{u_t}$ where a player ($j \in N$) resides. In T_N every player is connected to the root, hence j must be connected as well unless $\mathbf{u_t} \in Q_k$. The graph is acyclic therefore $\mathbf{u_t} \neq \mathbf{u_0}$. Again as we builded $\cup_{\mathbf{q}\in Q_k} E_q \cap E(T_S)$ there has to be an outgoing edge from $\mathbf{u_t}$. Since there are finite number of nodes in Q_k eventually we will reach a node where a player from $N \setminus N(B_2) \setminus N(Q_k)$ resides, that is already connected to the root. Altogether we can estimate the cost of S_2 by

$$c_{\mathcal{D}}(S_2) \leq c_{\mathcal{D}}(N) - \tau(B_2, N) + \tau(Q_k, S).$$

Now adding (2) and (3) together, then subtracting $c_{\mathcal{D}}(N)$ from both sides yield us the desired result. □

Theorem 12. *PAINT calculates the nucleolus of the game i.e.* $z = \mathcal{N}(\Gamma_{\mathcal{D}})$

Proof. Players who can reach the root with zero cost in $\mathcal{D}$ are dummy players. Their contribution to any coalition is zero. The nucleolus allocates zero cost to dummy players, but so does our algorithm. Therefore without loss of generality we can assume that in our starting network no player resides in a node where the root can be reached with zero cost (this is not necessarily true after $\mathcal{P}_1$). The proof proceeds by induction. Lets assume that the algorithm works for canonized graphs with m nodes and at most k number of non-zero edges (a canonized graph with m vertices can have at most $\frac{m(m-1)}{2}$ edges). Indeed if every edge of an m-node graph is a zero-edge then z coincides with the nucleolus. Let $\bar{\mathcal{D}}$ denote

the DAG-network that is generated after $\mathcal{P}_1$. For convenience sake let us write simply Γ instead of $\Gamma_{\mathcal{D}}$ and $\bar{\Gamma}$ instead of $\Gamma_{\bar{\mathcal{D}}}$ from now on. In $\bar{\mathcal{D}}$ the number of non-zero edges is strictly less as in $\mathcal{D}$ hence our assumption holds. Therefore $z = \mathcal{N}(\bar{\Gamma}) + \underline{t}_1$ where $\underline{t}_1$ is the $|N|$ dimensional vector whose coordinates are t_1. We need to show that z is the nucleolus of the game.

Let $\bar{\mathcal{E}}$ be the set of dually essential coalitions in $\bar{\Gamma}$. The next observation basically states that during the painting some dually essential coalitions become inessential, but this change does not occur in the opposite way. If a coalition is dually inessential it stays so even if some of the non-zero edges in the graph are replaced with zero-edges.

Observation 13. $\bar{\mathcal{E}} \subseteq \mathcal{E}$

This observation immediately follows from the structure of the dually essential coalitions and from the description of our algorithm.

As $\bar{\mathcal{E}}$ is a characterization set of the nucleolus in $\bar{\Gamma}$ by definition $\mathcal{N}(\bar{\Gamma}^{\bar{\mathcal{E}}}) = \mathcal{N}(\bar{\Gamma})$. Using Observation 13 we can also conclude that $\mathcal{N}(\bar{\Gamma}^{\bar{\mathcal{E}}}) = \mathcal{N}(\bar{\Gamma}^{\mathcal{E}})$. By Theorem 3 the set $\{S \in \mathcal{E} | exc_{\bar{\Gamma}}(S, x) \leq y\}$ is balanced or empty for any $y \in \mathbb{R}$.

Let $\mathcal{S}_0$ denote the set of coalitions whose excess is zero for any core allocation, formally

$$\mathcal{S}_0 \stackrel{def}{=} \{S \subseteq N \mid c(S) = x(S) \text{ for any } x \in \mathcal{C}(\Gamma)\}.$$

Lemma 14. *During $\mathcal{P}_1$ T_S is painted by $|S|$ players for each $S \in \mathcal{E} \cap \mathcal{S}_0$ and by $|S|+1$ players for each $S \in \mathcal{E} \setminus \mathcal{S}_0$, thus*

$$c_{\mathcal{D}}(S) = c_{\bar{\mathcal{D}}}(S) + |S|t_1 \quad \text{for any } S \in \mathcal{E} \cap \mathcal{S}_0$$
$$c_{\mathcal{D}}(S) = c_{\bar{\mathcal{D}}}(S) + (|S|+1)t_1 \quad \text{for any } S \in \mathcal{E} \setminus \mathcal{S}_0$$

Proof. Let $S \subseteq N$ be an arbitrary coalition. It follows from the description of the algorithm that each player of S paints exactly one edge in T_S. The only exceptions are those players that travelled forward to the root in the first step of the algorithm. In those cases a shadow player paints an edge of T_S instead each of them. A trunk can be painted however by more player than the number of its inhabitants if outside players contribute.

By Lemma 7 and Observation 10 for any $S \in \mathcal{E} \cap \mathcal{S}_0$ the trunk T_S corresponds either to a base branch $\{\mathbf{r}\} \cup B^Q_{\mathbf{p}}$ or to its complement $V \setminus B^Q_{\mathbf{p}}$, where for every $\mathbf{q} \in Q$ there leads a zero edge from $\mathbf{q}$ to $V \setminus B^Q_{\mathbf{p}}$. The closest common ancestor of any $\mathbf{q} \in Q$ is therefore the root. Hence in the first case players that reside in $\mathbf{q}$ or one of its ancestor nodes will not help to paint any T_N edges of $\{\mathbf{r}\} \cup B^Q_{\mathbf{p}}$. In the second case $V \setminus B^Q_{\mathbf{p}}$ incorporates every descendant of its own nodes except for the root, but no player resides at the root. It follows that T_S is painted by $|S|$ players for each $S \in \mathcal{E} \cap \mathcal{S}_0$.

In case of $S \in \mathcal{E} \setminus \mathcal{S}_0$ the trunk T_S corresponds to a branch $B_{\mathbf{r}} \setminus B^Q_{\mathbf{p}}$. By property **P2** there resides at least one player in $\mathbf{p}$. One player of $N(\mathbf{p})$ travels forward to its ancestor

at the first step of the algorithm. All the other players of $N(B^Q_{\mathbf{p}})$ paint only edges that goes out from $B^Q_{\mathbf{p}}$ i.e. no edges of T_S. It follows that T_S is painted by $|S|+1$ players for each $S \in \mathcal{E} \setminus \mathcal{S}_0$.

It can be shown in a similar manner that the trunk of any $n-1$ player coalition is painted by $n-1$ or n player depending on whether the coalition is a member $\mathcal{S}_0$ or not.

□

As a direct consequence of Lemma 14 for any $S \in \mathcal{E} \setminus \mathcal{S}_0$

$$c_{\mathcal{D}}(S) - z(S) = c_{\bar{\mathcal{D}}}(S) + (|S|+1)t_1 - z(S)$$
$$exc_{\Gamma}(S,z) = exc_{\bar{\Gamma}}(S, \mathcal{N}(\bar{\Gamma})) + (|S|+1)t_1 - |S|t_1$$
$$exc_{\Gamma}(S,z) = exc_{\bar{\Gamma}}(S, \mathcal{N}(\bar{\Gamma})) + t_1$$
$$exc_{\Gamma}(S,z) = exc_{\bar{\Gamma}}(S, \mathcal{N}(\bar{\Gamma}^{\mathcal{E}})) + t_1$$

That means the non-zero excesses in $\Gamma^{\mathcal{E}}$ with respect to z differ only by a constant from the excesses of $\bar{\Gamma}^{\mathcal{E}}$ with respect to $\mathcal{N}(\bar{\Gamma}^{\mathcal{E}})$. By the same argument the coalitions whose excess are zero for any core allocation is the same in $\Gamma^{\mathcal{E}}$ and $\bar{\Gamma}^{\mathcal{E}}$. By Theorem 3 it follows that z is the nucleolus of $\Gamma^{\mathcal{E}}$. But $\mathcal{E}$ is a characterization set for the nucleolus in Γ therefore

$$z = \mathcal{N}(\Gamma^{\mathcal{E}}) = \mathcal{N}(\Gamma)$$

□

References

[1] R. Brânzei, V. Fragnelli, and S.H. Tijs. Tree-connected peer group situations and peer group games. *Mathematical Methods of Operations Research*, 55:93–106, 2002.

[2] R. Brânzei, T. Solymosi, and S.H. Tijs. Strongly essential coalitions and the nucleolus of peer group games. *International Journal of Game Theory*, 33:447–460, 2005.

[3] Rosenthal E. C. Shortest path games. *European Journal of Operation Research*, 224:132–140, 2013.

[4] B. Çiftçi, P. Borm, and H. Hamers. Highway games on weakly cyclic graphs. *European Journal of Operational Research*, 204(1):117–124, 2010.

[5] U. Faigle, W. Kern, and Kuipers J. Computing the nucleolus of min-cost spanning tree games is np-hard. *International Journal of Game Theory*, 27, 1998.

[6] D. Granot, F. Granot, and W. R. Zhu. Characterization sets for the nucleolus. *International Journal of Game Theory*, 27(3):359–374, November 1998.

[7] D. Granot and G. Huberman. On the core and nucleolus of minimum cost spanning tree games. *Mathematical Programming*, 29(3):323–347, July 1984.

[8] G. Huberman. The nucleolus and essential coalitions. In A. Bensoussan and J. L. Lions, editors, *Analysis and Optimization of Systems*, volume 28 of *Lecture Notes in Control and Information Sciences*, pages 416–422. Elsevier B.V., 1980.

[9] Potters J. and Sudhölter P. Airport problems and consistent allocation rules. *Mathematical Social Sciences*, 38:83–102, 1999.

[10] E. Kohlberg. On the nucleolus of a characteristic function game. *SIAM Journal on Applied Mathematics*, 20:62–65, 1971.

[11] J. Kuipers. A polynomial time algorithm for computing the nucleolus of convex games. Report m 96-12, University of Maastricht, 1996.

[12] J. Márkus, M. Pintér, and A. Radványi. The shapley value for airport and irrigation games. *Mathematical Programming*, 2011.

[13] M. Maschler, B. Peleg, and L.S. Shapley. Geometric properties of the kernel, nucleolus and related solution concepts. *Math. Oper. Res.*, 4:303–338, 1979.

[14] M. Maschler, J. Potters, and H. Reijnierse. The nucleolus of a standard tree game revisited: a study of its monotonicity and computational properties. *International Journal of Game Theory*, 39(1-2):89–104, September 2009.

[15] Michael Maschler. The bargaining set, kernel, and nucleolus. In R.J. Aumann and S. Hart, editors, *Handbook of Game Theory with Economic Applications*, volume 1 of *Handbook of Game Theory with Economic Applications*, chapter 18, pages 591–667. Elsevier, 1992.

[16] D. Schmeidler. The nucleolus of a characteristic function game. *SIAM Journal on Applied Mathematics*, 17:1163–1170, 1969.

[17] A. Sobolev. A characterization of optimality principles in cooperative games by functional equations (russian). *Math. Methods Soc. Sci.*, 6:94–151, 1975.

[18] T. Solymosi and T. E. S. Raghavan. An algorithm for finding the nucleolus of assignment games. *International Journal of Game Theory*, 23:119–143, 1994.