A Connectivity Rating for Vertices in Networks

Marco Abraham[1], Rolf Kötter[23], Antje Krumnack[1], and Egon Wanke[1]

[1] Institute of Computer Science, Heinrich-Heine-Universität Düsseldorf, D-40225 Düsseldorf, Germany
[2] C. & O. Vogt Brain Research Institute, Heinrich-Heine-Universität Düsseldorf, D-40225 Düsseldorf, Germany
[3] Institute of Anatomy II, Heinrich-Heine-Universität, Düsseldorf, D-40225 Düsseldorf, Germany

Abstract. We compute the influence of a vertex on the connectivity structure of a directed network by using Shapley value theory. In general, the computation of such ratings is highly inefficient. We show how the computation can be managed for many practically interesting instances by a decomposition of large networks into smaller parts. For undirected networks, we introduce an algorithm that computes all vertex ratings in linear time, if the graph is cycle composed or chordal.

1 Motivation and Introduction

This work is originally motivated by the analysis of networks that represent neural connections in a brain. The cerebral cortical sheet can be divided into many different areas according to several parcellation schemes [4, 9, 20]. The primate cortex forms a network of considerable complexity depending on the degree of resolution. Information forwarding is usually accompanied by the possibility to respond. Thus, the corresponding networks are generally strongly connected. From a systems point of view, it is a great challenge to analyze the influence of a single area to the connectivity structure of the hole system. Such information could be helpful to understand the functional consequences of a lesion.

We measure the influence of a vertex on the connectivity structure of a directed graph $G = (V_G, E_G)$ by a function ϕ based on the Shapley value theory, which was originally developed within game theory[1], see [16]. Our function ϕ is parameterized by a so-called *characteristic function* denoted by f_G. It counts for a set of vertices $V' \subseteq V_G$ the number of strongly connected components in the subgraph of G induced by the vertices of V'. In general, a characteristic function is a mapping from the subsets of a set of abstract objects N to the real numbers $\mathbb{R}$. The application of Shapley value computations to graphs was first done by Myerson in [10], who considered only undirected graphs. For a characteristic function $h : 2^{V_G} \rightarrow \mathbb{R}$ defined on vertex sets, Myerson analyzed

[1] In game theory literature the argument of ϕ is a game (usually denoted by letter v) over an abstract set of players N and the result is a vector of $\mathbb{R}^N$. Since we consider graphs, we prefer to use letter v for vertices rather for functions.

the function that computes for a given vertex set V' the sum of all $h(V'')$, where V'' is a vertex set of a connected component in the subgraph of G induced by V'. That is, for undirected graphs, our function ϕ_{f_G} is equivalent to the function defined by Myerson (called Myerson value) for the case that $h(V'') = 1$ for all $V'' \subseteq V_G$.

Several authors have already analyzed the computation of Shapley values defined for vertices in graphs. Owen shows in [12] how to compute Myerson values for trees. Gómez et al. prove in [7] a simple separation property for undirected graphs that can be used to compute some Myerson values more efficiently. Van den Brink and Borm analyze in [18] a characteristic function for vertex sets of directed graphs and show that the Shapley values for this function can be computed efficiently. However, this characteristic function covers only a local property of the vertices. Deng and Papadimitriou consider in [2] a characteristic function that sums up the weights of all edges between two vertices of V'.

The paper is organized as follows. In Section 2, we recall the definitions we need from Shapley value theory [16]. In Section 3, we introduce a binary relation on vertices called *strong separability*. If two vertices u, v are strongly separable then the rating $\phi_{f_G}(u)$ is independent of the existence of v and vice versa, that is, $\phi_{f_G}(u) = \phi_{f_{G-\{v\}}}(u)$ and $\phi_{f_G}(v) = \phi_{f_{G-\{u\}}}(v)$, where $G - \{u\}$ is graph G without vertex u and $G - \{v\}$ is graph G without vertex v. This allows us to decompose a directed graph into subgraphs such that the ratings of the vertices in the original graph are computable by the ratings of the vertices in the subgraphs (Theorem 1). We also show that deciding whether two vertices u, v are not strongly separable is NP-complete (Theorem 2) and deciding $\phi_{f_G}(u) < \phi_{f_G}(v)$ for two given vertices u, v is NP-hard. This implies that an algorithm for the computation of ϕ_{f_G} can be used to decide an NP-hard as well as a co-NP-hard decision problem.

In Section 4, we consider undirected graphs as a special case of directed graphs where undirected edges are represented by directed edges oriented against each other. Definition 1 applied to undirected graphs yields that two vertices are strongly separable if and only if there is no chordless cycle passing u and v. The extension of Theorem 1 to undirected graphs (Theorem 4) allows us to compute the rating $\phi_{f_G}(u)$ for all vertices in linear time if G is cycle composed (Theorem 5) or chordal (Theorem 6).

Although some of the results shown in this paper can be extended to a much more general case, we restrict ourself to the one characteristic function f_G. This reduces the mathematical notations and keeps the proofs as simple as possible.

2 The Shapley value

Let N be any set of *abstract objects*. A *characteristic function* f is a mapping from the subsets of N to the real numbers $\mathbb{R}$ with $f(\emptyset) = 0$. A *carrier* of f is a set $C \subseteq N$ such that $f(S) = f(S \cap C)$ for every $S \subseteq N$. Any superset of a carrier C of f is again a carrier of f. The objects outside a carrier do not contribute anything to the computations by f.

The *sum* (*superposition*) of two characteristic functions f and g, defined by $(f+g)(S) = f(S) + g(S)$, is again a characteristic function. Let π be any permutation of N, that is, π is a one to one mapping of N to itself. For a set $S \subset N$ let $\pi(S) = \{\pi(x) \mid x \in S)$ be the image of S under π. Let f_π be the characteristic function defined by $f_\pi(S) = f(\pi^{-1}(S))$.

To rate the objects of N with respect to a characteristic function f, we use a function ϕ that associates with every characteristic function f a *rating function* $\phi_f : N \to \mathbb{R}$ such that

(Axiom 1:) for every permutation π of N and all $x \in N$,

$$\phi_{f_\pi}(\pi(x)) = \phi_f(x),$$

(Axiom 2:) for every carrier C of f,

$$\sum_{x \in C} \phi_f(x) = f(C),$$

and
(Axiom 3:) for any two characteristic functions f and g,

$$\phi_{f+g} = \phi_f + \phi_g.$$

Shapley has shown in [16] that function ϕ is uniquely defined by the three axioms above. He has also shown that the rating of an object with respect to a characteristic function f is computable by

$$\phi_f(x) = \sum_{S \subseteq N,\ x \in S} \frac{(|S|-1)!\,(|N|-|S|)!}{|N|!}\,(f(S) - f(S - \{x\})), \qquad (1)$$

where $|S|$ and $|N|$ denote the size of S and C, respectively, or alternatively by

$$\phi_f(x) = \frac{1}{|N|!} \sum_{\pi \in \Pi_N} (f(m(\pi,x) \cup \{x\}) - f(m(\pi,x))), \qquad (2)$$

where Π_N is the set of all one to one mappings (*enumerations*) $\pi : N \to \{1, \ldots, |N|\}$ and

$$m(\pi,x) = \{y \in N \mid \pi(y) < \pi(x)\}$$

is the set of all $y \in N$ arranged on the left side of x.

3 A vertex rating for directed graphs

We now define a characteristic function f_G to rate the vertices in directed graphs. The rating will measure the influence of a vertex on the connectivity structure. The smaller the rating of a vertex the greater its importance to the connectivity.

Let $G = (V_G, E_G)$ be a finite directed graph, where V_G is a finite set of *vertices* and $E_G \subseteq V_G \times V_G$ is a finite set of *directed edges.* A *path* in G is a sequence $p = (v_1, \ldots, v_k)$, $k \geq 1$, of distinct vertices such that $(v_i, v_{i+1}) \in E_G$ for $i = 1, \ldots, k-1$. We say, p is a path of length k from v_1 to v_k. A path is called a *cycle* of G if G additionally has edge (v_k, v_1). We will consider only simple paths and cycles in which all vertices are distinct.

For a vertex set $V' \subseteq V_G$, let $G|_{V'}$ be the subgraph of G *induced* by the vertices of V', that is, $G|_{V'} = (V_{G'}, E_{G'})$ where $V_{G'} = V'$ and $E_{G'} = E \cap V' \times V'$. G is *strongly connected* if for every pair of vertices $u, v \in V_G$ there is a path from u to v in G. A *strongly connected component* of G is a maximal strongly connected subgraph of G.

Let $\mathrm{SCC}(G)$ be the set of all *strongly connected components* of G, and f_G be a function from the subsets of V_G to the real numbers $\mathbb{R}$ (here we need only the set of non-negative integers) such that for every subset $V' \subseteq V_G$,

$$f_G(V') = |\mathrm{SCC}(G|_{V'})|.$$

That is, $f_G(V')$ is the number of strongly connected components in the subgraph of G induced by the vertices of V'. Note that f_G is a characteristic function, because $f_G(\emptyset)$ is always zero. The complete vertex set V_G is always the only carrier of f_G for every directed graph G. By Axiom 2, we have

$$\sum_{v \in V_G} \phi_{f_G}(v) = f_G(V_G) = |\mathrm{SCC}(G)|.$$

Figure 1 shows an example of the vertex rating ϕ_{f_G} for a directed graph G with vertex set $V_G = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}$. Since G is strongly connected, we get $f_G(V_G) = \sum_{v \in V_G} \phi_{f_G}(v) = 1$. Following the computation of ϕ_{f_G} by Equation 2, vertex v_8 has rating $\phi_{f_G}(v_8) = \frac{1}{2}$, because $f_G(m(\pi, v_8) \cup \{v_8\}) - f_G(m(\pi, v_8)) = 0$ if and only if $\pi(v_6) < \pi(v_8)$. Otherwise, we have $f_G(m(\pi, v_8) \cup \{v_8\}) - f_G(m(\pi, v_8)) = 1$, which happens for half of all 8! enumerations π. Vertex v_1 has rating $\phi_{f_G}(v_1) = \frac{2}{3}$, because $f_G(m(\pi, v_1) \cup \{v_1\}) - f_G(m(\pi, v_1)) = 0$ if and only if $\pi(v_2) < \pi(v_1)$ and $\pi(v_3) < \pi(v_1)$. Otherwise, we have $f_G(m(\pi, v_1) \cup \{v_1\}) - f_G(m(\pi, v_1)) = 1$. Here the second case happens for two third of all 8! enumerations π.

Let $G = (V_G, E_G)$ and $G' = (V_{G'}, E_{G'})$ be two directed graphs. We call G and G' *isomorphic* if there is a one to one mapping $b : V_G \rightarrow V_{G'}$ such that for every pair of vertices $v_1, v_2 \in V_G$,

$$(v_1, v_2) \in E_G \;\Leftrightarrow\; (b(v_1), b(v_2)) \in E_{G'}.$$

Such a mapping b is called an *isomorphism* between G and G'. If G and G' are isomorphic then $f_G(V') = f_{G'}(b(V'))$ for every vertex set $V' \subseteq V_G$. Here

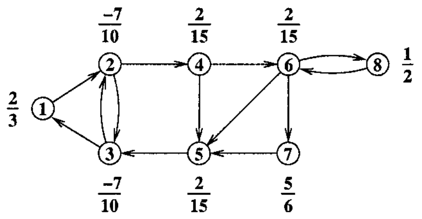


Fig. 1. The vertex rating ϕ_{f_G} for a directed graph G with 8 vertices. The smaller the rating of a vertex the greater its importance to the connectivity of the graph.

$b(V') = \{b(u) \mid u \in V'\}$ is the image of V' under b. This implies $\phi_{f_G}(v) = \phi_{f_{G'}}(b(v))$ for all vertices $v \in V$. Let $V' \subseteq V_G$ be any set of vertices of G. Graph G is called *V'-symmetric* if for every pair of vertices $v_1, v_2 \in V'$ there is an isomorphism b of G to G itself such that $b(v_1) = v_2$. In V'-symmetric graphs all vertices $v \in V'$ have the same rating. If two vertices v_1, v_2 have the same neighborhood, i.e., if $\{u \mid (u, v_1) \in E_G\} = \{u \mid (u, v_2) \in E_G\}$ and $\{u \mid (v_1, u) \in E_G\} = \{u \mid (v_2, u) \in E_G\}$, then G obviously is $\{v_1, v_2\}$-symmetric. Figure 2 shows some examples of partially symmetric graphs.

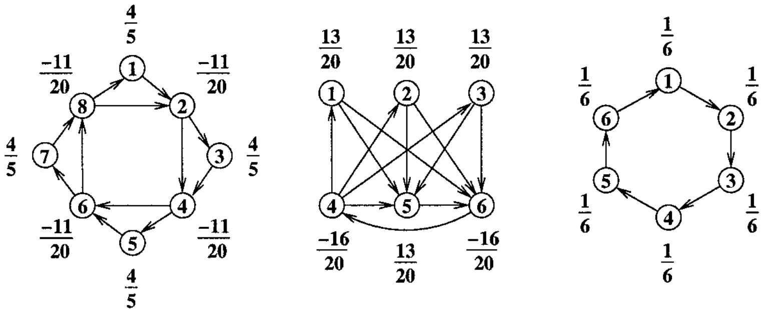


Fig. 2. The graph to the left is $\{v_1, v_3, v_5, v_7\}$-symmetric and $\{v_2, v_4, v_6, v_8\}$-symmetric, the graph in the middle is $\{v_1, v_2, v_3\}$-symmetric, and the graph to the right is $\{v_1, v_2, v_3, v_4, v_5, v_6\}$-symmetric.

The computation of a vertex rating $\phi_{f_G}(v)$ by Equation 1 or Equation 2 is highly inefficient. The number of subsets and the number of enumerations increase exponentially in the number of vertices of G. To handle the computation of ϕ_{f_G} for many practically interesting instances we will introduce a method to decompose a large graph into smaller parts. This decomposition will allow us to compute efficiently the ratings of vertices of the original graph by using the ratings of the vertices of smaller subgraphs. Our decomposition method will be introduced by the following two lemmas and Theorem 1.

The first lemma shows that the computation of a rating $\phi_{f_G}(v)$ for which the arguments of f_G are restricted to vertices of a subset $V' \subseteq V_G$ yields the computation of $\phi_{f_{G|_{V'}}}(v)$.

Lemma 1. *Let $G = (V_G, E_G)$ be a graph, $V' \subseteq V_G$, and $G' = G|_{V'}$. Let Π_{V_G} be the set of all enumerations $\pi : V_G \to \{1, \ldots, |V_G|\}$. Then for every vertex $v \in V'$*

$$\phi_{f_{G'}}(v) = \frac{1}{|V_G|!} \sum_{\pi \in \Pi_V} (f_G((m(\pi, v) \cup \{v\}) \cap V') - f_G(m(\pi, v) \cap V')).$$

Proof. Let $\Pi_{V'}$ be the set of all enumerations $\pi' : V' \to \{1, \ldots, |V'|\}$. First we show that for every enumeration $\pi' \in \Pi_{V'}$ there are $(|V'|+1) \cdot (|V'|+2) \cdot \cdots \cdot |V_G|$ unique enumerations $\pi \in \Pi_{V_G}$ such that for every pair of vertices $v_1, v_2 \in V'$, $\pi'(v_1) < \pi'(v_2)$ if and only if $\pi(v_1) < \pi(v_2)$. Let $p = (v_{i_1}, \ldots, v_{i_{|V'|}})$ be the sequence of vertices of V' in the order defined by π', that is

$$\pi'(v_{i_1}) < \pi'(v_{i_2}) < \ldots < \pi'(v_{i_{|V'|}}).$$

If we consider the vertices of $V_G - V'$ in an arbitrary order, then the first vertex of $V_G - V'$ can be placed at $|V'|+1$ positions at sequence p to get a sequence with $|V'| + 1$ vertices. After that the next vertex can be placed at $|V'| + 2$ positions in the resulting sequence to get a sequence with $|V'|+2$ vertices, and so on. The final vertex of $V_G - V'$ can be placed at $|V_G|$ positions in the sequence obtained by the preceding placement to get a sequence of all $|V_G|$ vertices of G. For all these $(|V'| + 1) \cdot (|V'| + 2) \cdot \cdots \cdot |V_G|$ enumerations π defined for enumeration π' we have

$$\begin{aligned} & f_{G'}(m(\pi', v) \cup \{v\}) && - f_{G'}(m(\pi', v)) \\ = & f_{G'}((m(\pi, v) \cup \{v\}) \cap V') && - f_{G'}(m(\pi, v) \cap V') \end{aligned}$$

for every vertex $v \in V'$, and thus

$$\begin{aligned} \phi_{f_{G'}}(v) &= \tfrac{1}{|V'|!} \textstyle\sum_{\pi' \in \Pi_{V'}} (f_{G'}(m(\pi', v) \cup \{v\}) - f_{G'}(m(\pi', v))) \\ &= \tfrac{1}{|V'|!} \textstyle\sum_{\pi \in \Pi_{V_G}} \frac{(f_{G'}((m(\pi,v) \cup \{v\}) \cap V') - f_{G'}(m(\pi,v) \cap V'))}{(|V'|+1) \cdot (|V'|+2) \cdot \cdots \cdot |V_G|} \\ &= \tfrac{1}{|V_G|!} \textstyle\sum_{\pi \in \Pi_{V_G}} (f_{G'}((m(\pi, v) \cup \{v\}) \cap V') - f_{G'}(m(\pi, v) \cap V')) \\ &= \tfrac{1}{|V_G|!} \textstyle\sum_{\pi \in \Pi_{V_G}} (f_G((m(\pi, v) \cup \{v\}) \cap V') - f_G(m(\pi, v) \cap V')). \end{aligned}$$

The last equality follows from the fact that $f_{G'}(V'' \cap V') = f_G(V'' \cap V')$ for every subset $V'' \subseteq V_G$. □

It is easy to see that the rating of a vertex in a graph G depends only on the connectivity structure of the strongly connected component the vertex belongs to, as the following observation shows. If $G' = (V_{G'}, E_{G'})$ is a strongly connected component of $G = (V_G, E_G)$ then for every vertex $v \in V_{G'}$ and every vertex set $V'' \subseteq V_G$,

$$f_G((V'' \cup \{v\}) \cap V_{G'}) - f_G(V'' \cap V_{G'}) = f_G(V'' \cup \{v\}) - f_G(V''),$$

and thus by Lemma 1, $\phi_{f_{G'}}(v) = \phi_{f_G}(v)$.

We will now define a property of a vertex pair u, v that allows us to compute independently the rating for two vertices u and v. That is, the rating of u in G will be equal to the rating of u in graph G without v.

Definition 1. *Let $G = (V_G, E_G)$ be a directed graph and $u, v \in V_G$ be two non-adjacent vertices, that is, neither (u, v) nor (v, u) is an edge of G. Vertex u and vertex v are* strongly separable *in G if for every strongly connected induced subgraph $H = (V_H, E_H)$ of G which contains u and v there is a strongly connected subgraph $J = (V_J, E_J)$ of H without u and v such that $H|_{V_H - V_J}$ has no path from u to v and no path from v to u.*

For the proof of the next lemma we need the notion of an undirected graph. In an *undirected graph* $G = (V_G, E_G)$ the edge set is a subset of $\{\{u, v\} \mid u, v \in V_G,\ u \neq v\}$. Analogously to the definitions for directed graphs, an *undirected path* of length k, $k \geq 1$, is a sequence $p = (v_1, \ldots, v_k)$ of k distinct vertices such that $\{v_i, v_{i+1}\} \in E_G$ for $i = 1, \ldots, k-1$. An undirected path is called an *undirected cycle* if G additionally has edge $\{v_k, v_1\}$ and the path has at least three vertices. The subgraph of G induced by a vertex set $V' \subseteq V_G$ has edge set $E_G \cap \{\{u, v\} \mid u, v \in V',\ u \neq v\}$. A graph is *connected* if there is a path between every pair of vertices, a *connected component* is a maximal connected subgraph, a *forest* is an undirected graph without cycles, and a *tree* is a connected forest.

Lemma 2. *Let $G = (V_G, E_G)$ be a directed graph and $V_H, V_J \subseteq V_G$ be two vertex sets such that $V_H \cup V_J = V_G$ and for every edge $(v_1, v_2) \in E_G$ both vertices are in V_H or in V_J, or in both sets. Let $H = G|_{V_H}$, $J = G|_{V_J}$, and $I = G|_{V_H \cap V_J}$. If every pair of vertices $u \in V_H - V_J$, $v \in V_J - V_H$ is strongly separable in G, then for every vertex set $V' \subseteq V_G$,*

$$f_G(V') = f_H(V' \cap V_H) + f_J(V' \cap V_J) - f_I(V' \cap V_I).$$

Proof. Let $V' \subseteq V_G$ be any set of vertices of G. Consider the following undirected graph $T = (V_T, E_T)$ with vertex set

$$V_T = \mathrm{SCC}(H|_{V'}) \cup \mathrm{SCC}(J|_{V'})$$

such that two vertices of V_T are connected by an undirected edge if and only if the two strongly connected components have at least one common vertex. If two distinct strongly connected components of V_T are connected by an undirected edge in T then one of them has to be from $\mathrm{SCC}(H|_{V'})$ and the other has to be from $\mathrm{SCC}(J|_{V'})$. Furthermore, for every strongly connected component C of $\mathrm{SCC}(I|_{V'})$, there is exactly one strongly connected component C_1 of $\mathrm{SCC}(H|_{V'})$ and exactly one strongly connected component C_2 of $\mathrm{SCC}(J|_{V'})$, and the common vertices of C_1 and C_2 are exactly the vertices of C.

Since every pair of vertices $u \in V_H - V_J$, $v \in V_J - V_H$ is strongly separable in G, the undirected graph T has no cycles, that is, T is a forest. The number of connected components of T (the number of trees of forest T) is equivalent to the number of strongly connected components of G. The number of connected components in a forest is always equivalent to its number of vertices minus its the number of edges. Since T has exactly one edge for every strongly connected component of $\mathrm{SCC}(I|_{V'})$ and exactly one vertex for every strongly connected component of $\mathrm{SCC}(H|_{V'})$ and $\mathrm{SCC}(J|_{V'})$, we get

$$f_G(V') = f_H(V' \cap V_H) + f_J(V' \cap V_J) - f_I(V' \cap V_I).$$

□

The following theorem states how ratings of vertices of G can be computed by the ratings of the same vertices in certain subgraphs of G.

Theorem 1. *Let $G = (V_G, E_G)$ be a directed graph and $V_H, V_J \subseteq V_G$ be two vertex sets such that $V_H \cup V_J = V_G$ and for every edge $(v_1, v_2) \in E_G$ both vertices v_i, v_2 are in V_H or in V_J, or in both sets. Let $H = G|_{V_H}$, $J = G|_{V_J}$, and $I = G|_{V_H \cap V_J}$. If every pair of vertices $u \in V_H - V_J$, $v \in V_J - V_H$ is strongly separable in G, then*

1. *for every vertex $w \in V_H \cap V_J$, $\phi_{f_G}(w) = \phi_{f_H}(w) + \phi_{f_J}(w) - \phi_{f_I}(w)$,*
2. *for every vertex $w \in V_H - V_J$, $\phi_{f_G}(w) = \phi_{f_H}(w)$, and*
3. *for every vertex $w \in V_J - V_H$, $\phi_{f_G}(w) = \phi_{f_J}(w)$.*

Proof. Let w be any vertex of V_G. By Lemma 2, for every vertex set $V' \subseteq V_G$,

$$\begin{aligned} f_G(V' \cup \{w\}) - f_G(V') = \quad & (f_H((V' \cup \{w\}) \cap V_H) - f_H(V' \cap V_H)) \\ & + (f_J((V' \cup \{w\}) \cap V_J) \quad - f_J(V' \cap V_J)) \\ & - (f_I((V' \cup \{w\}) \cap V_I) \quad - f_I(V' \cap V_I)). \end{aligned}$$

If $w \in V_H \cap V_J$, then by Lemma 1 we get

$$\phi_{f_G}(w) = \phi_{f_H}(w) + \phi_{f_J}(w) - \phi_{f_I}(w).$$

If w is a vertex of $V_H - V_J$, then $(V' \cup \{w\}) \cap V_J = V' \cap V_J$ and $(V' \cup \{w\}) \cap V_I = V' \cap V_I$, and thus

$$f_G(V' \cup \{w\}) - f_G(V') \;=\; f_H((V' \cup \{w\}) \cap V_H) - f_H(V' \cap V_H),$$

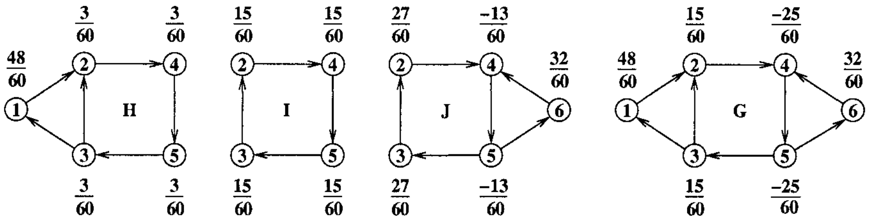
which implies by Lemma 1

$$\phi_{f_G}(w) = \phi_{f_H}(w).$$

If w is a vertex of $V_J - V_H$, then an analog argumentation yields $\phi_{f_G}(w) = \phi_{f_J}(w)$. □

Figure 3 shows an example of a directed graph G in which all ratings $\phi_{f_G}(v)$ are computed with Theorem 1 from the ratings of three subgraphs H, I, and J.

We will show now that the problem to decide whether two vertices in a directed graph are not strongly separable is NP-complete. See [6] for an introduction to the theory of NP-completeness. This implies that there is no



$\frac{3}{60}$ $\frac{3}{60}$ $\frac{48}{60}$ H $\frac{3}{60}$ $\frac{3}{60}$
$\frac{15}{60}$ $\frac{15}{60}$ I $\frac{15}{60}$ $\frac{15}{60}$
$\frac{27}{60}$ $\frac{-13}{60}$ $\frac{32}{60}$ J $\frac{27}{60}$ $\frac{-13}{60}$
$\frac{15}{60}$ $\frac{-25}{60}$ $\frac{48}{60}$ $\frac{32}{60}$ G $\frac{15}{60}$ $\frac{-25}{60}$

Fig. 3. Four graphs $G = (V_G, E_G)$, $H = G|_{V_H}$, $J = G|_{V_J}$, and $I = G|_{V_H \cap V_J}$ such that $V_H, V_J \subseteq V_G$, $V_H \cup V_J = V_G$, and for every edge $(v_1, v_2) \in E_G$ both vertices are in V_H or in V_G, or in both sets. Vertex pair $v_1 \in V_H - V_J$, $v_6 \in V_J - V_H$ is strongly separable in G.

polynomial time algorithms which decides whether two vertices in a directed graph are not strongly separable, unless P = NP. The NP-hardness follows by a simple reduction from the satisfiability problem. The terms we use in describing this problem are the following.

Let $X = \{x_1, \ldots, x_n\}$ be a set of *Boolean variables*. A *truth assignment* for X is a function $t : X \rightarrow \{\text{true}, \text{false}\}$. If $t(x_i) = \text{true}$ we say variable x_i is *true* under t; if $t(x_i) = \text{false}$ we say variable x_i is *false* under t. If x_i is a variable of X, then x_i and $\overline{x_i}$ are *literals* over X. Literal x_i is true under t if and only if variable x_i is true under t; literal $\overline{x_i}$ is true under t if and only if variable x_i is false under t. A *clause* over X is a set of literals over X, for example $\{x_1, \overline{x_2}, x_4\}$. It represents the disjunction of literals which is *satisfied* by a truth assignment t if and only if at least one of its literals is true under t. A collection $\mathcal{C}$ of clauses over X is satisfiable if and only if there is a truth assignment t that simultaneously satisfies all clauses of $\mathcal{C}$.

The satisfiability problem, denoted by SAT, is specified as follows. Given a set X of variables and a collection $\mathcal{C}$ of clauses over X. Is there a satisfying truth assignment for $\mathcal{C}$? This problem is NP-complete even for the case that every clause of $\mathcal{C}$ has exactly three distinct literals (3-SAT, for short).

Theorem 2. *The problem to decide whether two vertices u, v of a directed graph G are not strongly separable is NP-complete.*

Proof. Let us first illustrate that the problem belongs to NP. Two vertices u and v are not strongly separable in G if and only if G has a strongly connected induced subgraph $G' = (V_{G'}, E_{G'})$ that includes u and v such that $G'|_{V_{G'} - \{u,v\}}$ has no strongly connected subgraph $G'' = (V_{G''}, E_{G''})$ such that in $G'|_{V_{G'} - V_{G''}}$ there is no path from u to v and no path from v to u. Without loss of generality we can assume that G'' is a strongly connected component of $G'|_{V_{G'} - \{u,v\}}$. So we can non-deterministically consider every strongly connected subgraph G' of G that includes u and v. Then we can verify in polynomial time for every strongly connected component $G'' = (V_{G''}, E_{G''})$ of $G'|_{V_{G'} - \{u,v\}}$ whether $G'|_{V_{G'} - V_{G''}}$ has no path from u to v and no path from v to u. Thus, the problem to decide whether two vertices u, v are not strongly separable belongs to NP.

The NP-hardness follows by a simple transformation from 3-SAT. Let $X = \{x_1, \ldots, x_n\}$ be a set of n Boolean variables and $\mathcal{C} = \{C_1, \ldots, C_m\}$ be a collection of m clauses. We define a graph $G(X, \mathcal{C})$ with two vertices u, v such that there is a truth assignment t for X that satisfies every clause of $\mathcal{C}$ if and only if u and v are not strongly separable in $G(X, \mathcal{C})$. Figure 4 shows an example of such a construction for four variables x_1, x_2, x_3, x_4 and four clauses $\{x_2, x_3, \overline{x_4}\}$, $\{x_1, \overline{x_2}, x_4\}$, $\{\overline{x_1}, \overline{x_2}, \overline{x_3}\}$, $\{\overline{x_1}, x_2, x_3\}$.

Graph $G(X, \mathcal{C})$ has six vertices u, a, b, v, c, d, two *literal vertices* x_i, $\overline{x_i}$ for every variable x_i, $1 \le i \le n$, and three *literal vertices* $c_{j,1}$, $c_{j,2}$, $c_{j,3}$ for every clause $C_j = \{c_{j,1}, c_{j,2}, c_{j,3}\}$, $1 \le j \le m$. $G(X, \mathcal{C})$ has the edges (u, a), (a, x_1), $(a, \overline{x_1})$, the edges (x_i, x_{i+1}), $(x_i, \overline{x_{i+1}})$, $(\overline{x_i}, x_{i+1})$, $(\overline{x_i}, \overline{x_{i+1}})$ for $i = 1, \ldots, n-1$, the edges (x_n, b), $(\overline{x_n}, b)$, (b, v), (v, c), $(c, c_{1,1})$, $(c, c_{1,2})$, $(c, c_{1,3})$, the edges $(c_{j,k}, c_{j+1,l})$ for $j = 1, \ldots, m-1$ and $k, l \in \{1, 2, 3\}$, and the edges $(c_{m,1}, d)$, $(c_{m,2}, d)$, $(c_{m,3}, d)$, (d, u), and (d, a). Additionally, there are a so-called *cross edges* from every literal vertex x_i ($\overline{x_i}$) for variable x_i to every literal vertex $\overline{x_i}$ (x_i, respectively) for some clauses. In Figure 4, the cross edges are drawn as dotted arcs.

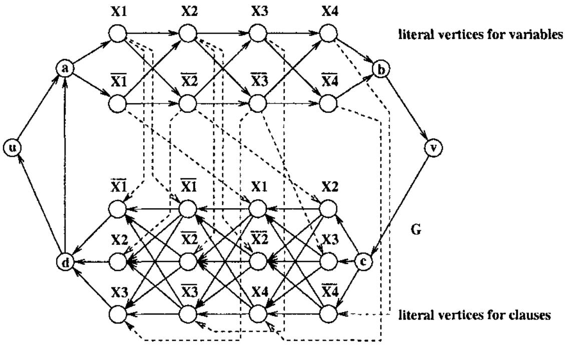


Fig. 4. The graph $G(X, \mathcal{C})$ for $X = x_1, x_2, x_3, x_4$ and $\mathcal{C} = \{x_2, x_3, \overline{x_4}\}$, $\{x_1, \overline{x_2}, x_4\}$, $\{\overline{x_1}, \overline{x_2}, \overline{x_3}\}$, $\{\overline{x_1}, x_2, x_3\}$.

Every cycle of $G(X, \mathcal{C})$ that includes vertex u and v consists of two vertex disjoint path $p_1 = (u, a, \ldots, b, v)$ and $p_2 = (v, c, \ldots, d, u)$. Path p_1 passes exactly one literal vertex for every variable, and defines in this way an assignment t for the variables, where path p_2 passes exactly one literal vertex for every clause.

Assume there is a truth assignment t for X that satisfies every clause. Then there is an induced subgraph G' of $G(X, \mathcal{C})$ that includes vertex u and v but no cross edge, for example the subgraph of $G(X, \mathcal{C})$ induced by u, v, a, b, c, d and all true literal vertices. In this case, it is not possible to destroy all paths between u and v and all paths between v and u by removing a strongly connected subgraph of G'. Thus u and v are not strongly separable.

Assume there is no truth assignment for X that satisfies every clause. Then every strongly connected induced subgraph G' of G that includes u and v has at least one cross edge (u', v'). In this case it is easy to destroy all paths from u to v and all path from v to u by removing a cycle that includes the edge (d, a) and the cross edge (u', v'). Thus u and v are strongly separable. □

Theorem 2 can be used to prove that deciding whether two vertices have a different rating is NP-hard. Consider again the graph $G(X, \mathcal{C})$ with the two vertices u and v constructed for an instance $(X, \mathcal{C})$ of 3-SAT as in the proof of Theorem 2. Let $G'(X, \mathcal{C})$ be the graph $G(X, \mathcal{C})$ without the vertex v and its incident edges. Then $\phi_{f_{G(X,\mathcal{C})}}(u) = \phi_{f_{G'(X,\mathcal{C})}}(u)$ if u and v are strongly separable in G, and $\phi_{f_{G(X,\mathcal{C})}}(u) < \phi_{f_{G'(X,\mathcal{C})}}(u)$ if u and v are not strongly separable in G.

Theorem 3. *The problem to decide whether $\phi_{f_G}(u) < \phi_{f_G}(v)$ for two vertices u, v of a directed graph G is NP-hard.*

Thus, an algorithm for the computation of ϕ_{f_G} can be used to decide an NP-hard as well as a co-NP-hard decision problem.

4 A vertex rating for undirected graphs

The vertex rating ϕ_{f_G} for directed graphs can simply be extended to undirected graphs. For an undirected graph G let $\mathrm{dir}(G)$ be the directed graph we get if we replace every undirected edge $\{u, v\}$ by two directed edges (u, v) and (v, u). Let f_G now be the function from the subsets of V_G to the real numbers $\mathbb{R}$ such that for every $V' \subseteq V_G$, $f_G(V')$ is the number of connected components in the subgraph of G induced by the vertices of V'. That is, the rating of a vertex v in an undirected graph G is equal to the rating of v in the directed graph $\mathrm{dir}(G)$.

Figure 5 shows an example of the vertex rating ϕ_{f_G} for an undirected graph G with vertex set $V_G = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}$.

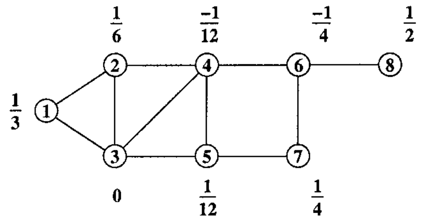


Fig. 5. The vertex rating ϕ_{f_G} for an undirected graph G with 8 vertices.

It is easy to verify that two vertices u, v of $\mathrm{dir}(G)$ are not strongly separable if and only if G has a chordless cycle that includes u and v. A *chord* for a cycle $c = (u_1, \ldots, u_k)$ is an edge $\{u_i, u_j\}$ such that $2 \leq |i - j| \leq k - 2$. The problem of determining whether an undirected graph G contains a chordless cycle can be solved in linear time [3, 15, 17]. This is the well-known chordal graph recognition problem. A graph G is a *chordal graph* if any cycle of G of length at least four has at least one chord, or alternatively, if G has no chordless cycle, see [8]. The problem of determining whether G contains a chordless cycle of length $k \geq 5$ can be solved in $O(|V_G| + |E_G|^2)$ time on $O(|V_G| \cdot |E_G|)$ space, see [11].

Theorem 1 applied to undirected graphs yields the following theorem which is a more general version of Proposition 2 of [7].

Theorem 4. *Let $G = (V_G, E_G)$ be an undirected graph and $V_H, V_J \subseteq V_G$ be two vertex sets such that $V_H \cup V_J = V_G$ and for every edge $\{v_1, v_2\} \in E_G$ both vertices u_1, u_2 are in V_H or in V_J, or in both sets. Let $H = G|_{V_H}$, $J = G|_{V_J}$, and $I = G|_{V_H \cap V_J}$. If G has no chordless cycle with a vertex of $V_H - V_J$ and a vertex of $V_J - V_H$, then*

1. *for every vertex $w \in V_H \cap V_J$, $\phi_{f_G}(w) = \phi_{f_H}(w) + \phi_{f_J}(w) - \phi_{f_I}(w)$,*
2. *for every vertex $w \in V_H - V_J$, $\phi_{f_G}(w) = \phi_{f_H}(w)$, and*
3. *for every vertex $w \in V_J - V_H$, $\phi_{f_G}(w) = \phi_{f_J}(w)$.*

Assume a connected graph G can be separated into $k > 1$ connected components $G_1, \ldots, G_k$ by removing a complete subgraph I of G. Let $G_i' = G|_{V_{G_i} \cup V_I}$ for $i = 1, \ldots, k$ be the subgraphs of G induced by the vertices of connected component G_i and the vertices of the removed complete subgraph I. Then by Theorem 4 for every $i = 1, \ldots, k$ the vertex rating for a vertex w of G_i' is $\phi_{f_G}(w) = \phi_{f_{G_i'}}(w)$ and the vertex rating for a vertex w of I is

$$\phi_{f_G}(w) = \left(\sum_{j=1}^{k} \phi_{f_{G_j'}}(w) \right) - (k-1) \cdot \phi_{f_I}(w).$$

An example of a class of graphs for which the rating ϕ_{f_G} is efficiently computable is the class of *cycle composed* graphs which can recursively be defined as follows. The cycle C_n with $n \geq 3$ vertices is *cycle composed.* Let $G = (V_G, E_G)$ be a cycle composed graph and $e_1 = \{u_1, v_1\}$ be an edge of G. Let $C_n = (V_{C_n}, E_{C_n})$ be a cycle with $n \geq 3$ vertices and $e_2 = \{u_2, v_2\}$ be edge of C_n. Then the graph obtained by the vertex disjoint union of G and C_n and the identification of u_2 with u_1 and v_2 with v_1 is *cycle composed.* That is, the composed graph has vertex set $V_G \cup V_{C_n} - \{u_2, v_2\}$ and edge set $\{\{h(u), h(v)\} \mid \{u, v\} \in E_G \cup E_{C_n}\})$ where $h(u) = u$ for every $u \in V_G \cup V_{C_n} - \{u_2, v_2\}$, and $h(u_2) = u_1$ and $h(v_2) = v_1$.

Cycle composed graphs are biconnected and have tree-width at most 2, see [13, 1] for a definition of tree-width. Graphs of tree-width at most 2 can be recognized in linear time by removing vertices of degree at most 2. When a vertex u of degree 2 is removed then the two neighbors of u will be connected by an edge if they are not adjacent. A graph has tree-width 2 if and only if it can completely be reduced by removing vertices of degree at most 2 in the way described above, see for example [19].

Let $\mathcal{C}$ be the set of vertex sets of the cycles used to compose a cycle composed graph G, that is, $\mathcal{C}$ has a vertex set C for every cycle used to compose G. The vertex rating $\phi(u)$ for all vertices u of G is computable in linear time by the following simple procedure.

```
1. for every $u \in V$ do {
2.     let $\phi_{f_G}(u) := (\deg(u) - 2) * (-\frac{1}{2})$; }
3. for every $C \in \mathcal{C}$ do {
4.     for every $u \in C$ do {
5.         let $\phi_{f_G}(u) := \phi_{f_G}(u) + \frac{1}{|C|}$; } }
```

Since a vertex u is involved in $\text{degr}(u) - 2$ vertex identifications, the rating for u can be initialized by $(\deg(u) - 2) * (-\frac{1}{2})$. After that the algorithm adds for every cycle C the fraction $\frac{1}{|C|}$ to the ration of every vertex of C. Since the number of vertices in the sets of $\mathcal{C}$ is $\sum_{C \in \mathcal{C}} |C| = 2|E_G| - |V_G|$, the rating for all vertices in cycle composed graphs can be computed in linear time, if $\mathcal{C}$ is given.

The vertex sets of the cycles can be computed by the following algorithm. We assume that an empty vertex list is initially assigned to every edge. That is, every edge $\{u, v\}$ is initially represented as a pair $(\{u, v\}, \emptyset)$. An edge $e = (\{u, v\}, L)$ with a non-empty vertex list L represents a path between u and v passing the vertices of L. If G has a vertex u of degree 2 such that the two neighbors v, w of u are not adjacent, we remove vertex u and its two incident edges $(\{u, v\}, L_1)$, $(\{u, w\}, L_2)$ and insert a new edge $(\{v, w\}, L_1 \cup L_2 \cup \{u\})$ between u and v.

If G has a vertex u of degree 2 such that the two neighbors v, w of u are adjacent, the vertices of a cycle can be reported. Let $(\{u, v\}, L_1)$, $(\{u, w\}, L_2)$, $(\{v, w\}, L_3)$ be the three edges between the vertices u, v and w. The algorithm then reports vertex set $L_1 \cup L_2 \cup L_3 \cup \{u, v, w\}$. If graph G has no further edges than the three edges above, then all cycles are reported and the algorithm finishes. If graph G has some further edges and L_3 is non-empty, then the graph is not cycle composed. In any other case the algorithm removes the two edges $(\{u, v\}, L_1)$, $(\{u, w\}, L_2)$ and so forth. If this processing ends because there are no further vertices of degree 2, then the graph is also not cycle composed.

This algorithm computes the vertex sets of all cycles used to compose a cycle composed graph. The running time of this algorithm is $O(|V_G|^2)$ because we have to check for every vertex whether its two neighbors are adjacent. However, this problem can be eliminated by a simple trick which is also used in [19] for the recognition of outerplanar graphs. The trick is to check whether the two neighbors v, w of u are adjacent at the time when one of these two vertices v, w gets a degree of 2 or less. At that point the test can be done in a fixed number of steps and either a new edge is inserted or a cycle is reported. This modification yields a linear time algorithm for the computation of all cycles of a cycle composed graph. The following example shows a possible implementation.

```
create-new-edge (vertex $u$)
{
    let $e_1 = (\{u,v\}, L_1), e_2 = (\{u,w\}, L_2) \in E_G$ be the two edges incident to $u$;
    insert $(\{v,w\}, L_1 \cup L_2 \cup \{u\})$ into $E_{\text{new}}$;
    remove $e_1$ and $e_2$ from $E_G$;
    if ($\deg_{E_G}(v) = 2$) then
        insert $v$ into $M$;
    if ($\deg_{E_G}(w) = 2$) then
        insert $w$ into $M$;
}

move-new-edge (edge $e_{\text{new}} = (\{u,v\}, L_{\text{new}})$)
{
    if there is an edge $e = (\{u,v\}, L) \in E_G$ then {
        output $L \cup L_{\text{new}} \cup \{u,v\}$;
        remove $e_{\text{new}}$ from $E_{\text{new}}$;
        if ($|E_G| = 1$) and ($|E_{\text{new}}| = 0$) then
            halt "all cycles reported";
        else if ($L \neq \emptyset$) then
            halt "$G$ is not cycle composed";
    else {
        remove $e_{\text{new}}$ from $E_{\text{new}}$;
        insert $e_{\text{new}}$ into $E_G$;
        if ($\deg_{E_G}(u) = 3$) then
            remove $u$ from $M$;
        if ($\deg_{E_G}(v) = 3$) then
            remove $v$ from $M$; }
}

compute-cycles (graph $G = (V_G, E_G)$)
{
    let $M := \emptyset$;
    for every $u \in V_G$ do {
        if ($\deg_{E_G}(u) = 2$) then {
            insert $u$ into $M$; } }
    while ($M \neq \emptyset$) {
        let $u \in M$;
        if there is an edge $e_{\text{new}} \in E_{\text{new}}$ incident to $u$ then
            move-new-edge ($e_{\text{new}}$);
        else {
            if ($\deg_{E_G}(u) = 2$) then
                create-new-edge ($u$);
            remove $u$ from $M$; } }
    halt "$G$ is not cycle composed";
}
```

The algorithm above stores in a set M all vertices of degree 2. Note that the degree of a vertex is always determined by the edges of E_G. For every vertex u adjacent with exactly two vertices v, w a new edge is inserted into a set denoted by E_{new} but not yet into edge set E_G of graph G. Whenever a vertex u of M is considered for processing it is first checked whether there are edges incident to u in set E_{new}. If E_{new} has an edge e incident to u then e will either be inserted into E_G (if the two vertices of e are not adjacent by some edge of E_G), or a cycle is reported (if the two vertices of e are adjacent by some edge of E_G). The test whether the two vertices of e are adjacent by some edge of E_G can be done in time $O(1)$ because u is one of the end vertices of e and has vertex degree 2. This proves the following theorem.

Theorem 5. *The vertex rating $\phi_{f_G}(u)$ for all vertices u of a cycle composed graph G is computable in linear time.*

The vertex rating ϕ_{f_G} is also computable in linear time for *chordal graphs.* An interesting characterization of chordal graphs is the existence of a *perfect elimination order.* Let $p = (u_1, \dots, u_n)$ be an order of the $|V_G| = n$ vertices of $G = (V_G, E_G)$, and let $N(G, p, i)$ for $i = 1, \dots, n$ be the set of neighbors u_j of vertex u_i with $i < j$,

$$N(G,p,i) := \{u_j \mid \{u_i, u_j\} \in E_G \ \wedge \ i < j\}.$$

The vertex order $p = (u_1, \dots, u_n)$ is called a *perfect elimination order* (PEO) if the vertices of $N(G,p,i)$ for $i = 1, \dots, n-1$ induce a complete subgraph of G.

Dirac [3], Fulkerson and Gross [5], and Rose [14] have shown that a graph G is chordal if and only if it has a perfect elimination order. Rose, Tarjan, and Lueker have shown in [15], that a perfect elimination order can be found in linear time if one exists. If a perfect elimination order $p = (v_1, \dots, v_n)$ of the vertices of $G = (V_G, E_G)$ is given, then the vertex rating ϕ_{f_G} can be computed with Theorem 4 by the following algorithm. Note that, in a complete graph G with n vertices, $\phi_{f_G}(v) = \frac{1}{n}$ for every vertex of G, because G is V_G-symmetric.

```
1. let $\phi_{f_G}(v_n) := 1$;
2. for $i = n-1, \dots, 1$ do {
3.     let $\phi_{f_G}(v_i) := \frac{1}{|N(G,p,i)|+1}$;
4.     for all $v \in N(G,p,i)$ do {
5.         let $\phi_{f_G}(v) := \phi_{f_G}(v) + \frac{1}{|N(G,p,i)|+1} - \frac{1}{|N(G,p,i)|}$; } }
```

The running time of this algorithm is linear in the size of G, because the assignment of Line 3 is done exactly $|V_G| - 1$ times and the assignment of line 5 is done exactly $|E_G|$ times. Since the perfect elimination order can be found in linear time, we get the following theorem.

Theorem 6. *The vertex rating $\phi_{f_G}(v)$ for all vertices v of a chordal graph G is computable in linear time.*

References

1. H.L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209:1–45, 1998.
2. X. Deng and C.H. Papadimitriou. On the complexity of cooperative solution concepts. *Methods of Operations Research*, 19(2):257–266, 1994.
3. G. Dirac. On rigid circuit graphs. *Abh. Math. Sem. Univ. Hamburg*, 25:71–76, 1961.
4. D.J. Felleman and D.C. Van Essen. Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex*, 1:1–47, 1991.
5. D.R. Fulkerson and O.A. Gross. Incidence matrices and interval graphs. *Pacific J. Math.*, 15:835–855, 1965.
6. M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, San Francisco, 1979.
7. D. Gómez, E. González-Arangüena, C. Manuel, G. Owen, M. del Pozo, and J. Tejada. Splitting graphs when calculating Myerson value for pure overhead games. *Mathematical Methods of Operations Research*, 59:479–489, 2004.
8. A. Hajnal and J. Surányi. Über die Auflösung von Graphen in vollständige Teilgraphen. *Ann. Univ. Sci. Budapest, Eötvös Sect. Math.*, 1:113–121, 1958.
9. R. Kötter and E. Wanke. Mapping brains without coordinates. *Philosophical Transactions of the Royal Society London, Biological Sciences*, 360(1456):751–766, 2000.
10. R.B. Myerson. Graphs an cooperations in games. *Methods of Operations Research*, 2:255–229, 1977.
11. S.D: Nikolopoulos and L. Palios. Hole and antihole detection in graphs. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pages 850–859. ACM-SIAM, 2004.
12. G. Owen. Values of graph-restricted games. *SIAM Journal on Algebraic and Discrete Methods*, 7(2):210–220, 1986.
13. N. Robertson and P.D. Seymour. Graph minors II. Algorithmic aspects of tree width. *Journal of Algorithms*, 7:309–322, 1986.
14. D.J. Rose. Triangulated graphs and elimination process. *J. Math. Analys. Appl.*, 32:597–609, 1970.
15. D.J. Rose, R.E. Tarjan, and G.S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing*, 5:266–283, 1976.
16. L.S. Shapley. A value for n-person games. In H.W. Kuhn and A.W. Tucker, editors, *Contributions to the Theory of Games II*, pages 307–317, Princeton, 1953. Princeton University Press.
17. R.E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 13:566–579, 1984.
18. R. van den Brink and P. Borm. Digraph competitions and cooperative games. *Theory and Decision*, 53:327–342, 2002.
19. M. Wiegers. Recognizing outerplanar graphs in linear time. In *Proceedings of Graph-Theoretical Concepts in Computer Science*, volume 246 of *LNCS*, pages 165–176. Springer-Verlag, 1987.
20. K. Zilles. Architecture of the Human Cerebral Cortex. Regional and Laminar Oganization. In G. Paxinos and J.K. Mai, editors, *The Human Nervous System*, pages 997–1055, San Diego, CA, 2004. Elsevier. 2nd edition.