Theory Comput Syst (2018) 62:1048–1084
DOI 10.1007/s00224-017-9777-6

Exact Localisations of Feedback Sets

Michael Hecht[1,2]

Published online: 27 May 2017

Abstract The *feedback arc (vertex) set problem*, shortened FASP (FVSP), is to transform a given multi digraph $G = (V, E)$ into an acyclic graph by deleting as few arcs (vertices) as possible. Due to the results of Richard M. Karp in 1972 it is one of the classic NP-complete problems. An important contribution of this paper is that the subgraphs $G_{\mathrm{el}}(e)$, $G_{\mathrm{si}}(e)$ of all *elementary cycles* or *simple cycles* running through some arc $e \in E$, can be computed in $\mathcal{O}\left(|E|^2\right)$ and $\mathcal{O}(|E|^4)$, respectively. We use this fact and introduce the notion of the *essential minor* and *isolated cycles*, which yield a priori problem size reductions and in the special case of so called *resolvable graphs* an exact solution in $\mathcal{O}(|V||E|^3)$. We show that weighted versions of the FASP and FVSP possess a Bellman decomposition, which yields exact solutions using a dynamic programming technique in times $\mathcal{O}\left(2^m|E|^4\log(|V|)\right)$ and $\mathcal{O}\left(2^n\Delta(G)^4|V|^4\log(|E|)\right)$, where $m \leq |E|-|V|+1$, $n \leq (\Delta(G)-1)|V|-|E|+1$, respectively. The parameters m, n can be computed in $\mathcal{O}(|E|^3)$, $\mathcal{O}(\Delta(G)^3|V|^3)$, respectively and denote the maximal dimension of the cycle space of all appearing *meta graphs*, decoding the intersection behavior of the cycles. Consequently, m, n equal zero if all meta graphs are trees. Moreover, we deliver several heuristics and discuss how to control their variation from the optimum. Summarizing, the presented

This work was partially funded by the DFG project "MI439/14-1".

✉ Michael Hecht
hecht@mpi-cbg.de

[1] MOSAIC Group, Chair of Scientific Computing for Systems Biology, Faculty of Computer Science, TU Dresden and Center for Systems Biology Dresden, Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany

[2] Bioinformatics Group, Department of Computer Science, and Interdisciplinary Center for Bioinformatics, University of Leipzig, Härtelstraße 16-18, D-04107, Leipzig, Germany

results allow us to suggest a strategy for an implementation of a fast and accurate FASP/FVSP-SOLVER.

Keywords Feedback set problem · Acyclic subgraph problem · Linear ordering problem · Elementary cycle · Simple cycle

1 Introduction

The *feedback arc set* problem, shortened FASP, is to delete as less as possible arcs of a graph such that the resulting subgraph is acyclic, i.e., it contains no directed cycle. Another equivalent formulation is to find a linear ordering of the vertices of the graph such that the number of back arcs is minimized. Therefore the problem is also known as *maximum acyclic subgraph* problem or *linear ordering problem*. For directed graphs this problem is one of the classic NP-complete problems [22]. The problem of deleting a smallest subset of vertices to result in an acyclic subgraph is known as *feedback vertex set problem* (FVSP). The FASP and FVSP are linear time reducible among each other, by keeping the relevant parameters fix as we will assert in Section 2.1, alternatively see [13]. Therefore algorithmic properties of the two problems are transferable. In particular, the FVSP is also NP-complete. Analogous problems for undirected graphs can be defined. As shown in [22] the feedback vertex set problem remains NP-complete, while the feedback arc set problem can be solved efficiently by solving a maximum spanning tree problem. An excellent overview on feedback sets can be found in [4]. The problem of finding minimal transversals of directed cuts is closely related to the FASP, see [26]. The FASP stays NP-complete for graphs where every node has an in-degree and out-degree of at most three or line digraphs even when every clique has at most size three [15]. It is also NP-complete for tournament graphs [1]. However, there also exist graph classes possessing polynomial time algorithms, e.g., planar directed graphs or more general weakly acyclic digraphs [16], and reducible flow graphs [29]. The FASP or FVSP has a multitude of applications, e.g., retiming synchronous circuitry [25], circuit testing [24], computational biology and neuroscience [19], network analysis and operating systems [33].

1.1 Outline

In Section 2 we provide the graph theoretical concepts, which are fundamental for this article. In Section 3 we present our main results. The fact that the FASP/FVSP on multi-digraphs can be reduced to simple graphs is asserted in Section 4 and the first a priori problem size reduction is deduced. In Section 5 we construct an algorithm determing the induced subgraph of all cycles with one arc in common efficently. This knowledge is used in Section 6 to introduce the concept of *isolated cycles* and *resolvable graphs* and providing an efficent solution of the FASP/FVSP on resolvable graphs. Moreover, the second a priori problem size reduction is given. Afterwards we concentrate on the main result of the article. Namely, that the FASP/FVSP possesses a Bellmann decomposition and present exact solutions using this fact to

apply a dynamic programming technique in Section 7. In Section 8, we discuss how greedy approaches behave with respect to the problems and develope a strategy for a general FASP/FVSP-SOLVER. Finally, we discuss our results and other alternatives in Section 9.

2 Preliminaries

Before we can introduce the FASP (FVSP) formally, some basic concepts of graphs and cycles need to be mentioned.

2.1 Graphs and Cycles

A *multi-directed graph*, or *multi-digraph*, $G = (V, E)$ consists of a set of vertices V and a multi-set of arcs E containing elements from $V \times V$. A *directed graph* or *digraph* is a multi-digraph with a simple arc set E, i.e., $E \subseteq V \times V$ and therefore every $e \in E$ occurs exactly once. A digraph is called *simple* if there are no loops, i.e., $E \cap \mathbb{D}(V \times V) = \emptyset$, where $\mathbb{D}(V \times V) = \{(v, v) \in V \times V \mid v \in V\}$ denotes the diagonal.

If not stated otherwise throughout the article $G = (V, E)$ denotes a finite, connected, directed and loop-free multi-digraph and $G \setminus e$, $G \setminus v$ denote the graphs which occur by deleting the arc e and possibly occurring isolated vertices or the vertex v and all its adjacent arcs. For $\varepsilon \subseteq E$ and $\nu \subseteq V$ the graphs $G \setminus \varepsilon$, $G \setminus \nu$ are analogously defined. Moreover, $\mathcal{G}(\cdot)$, $\mathcal{E}(\cdot)$, $\mathcal{V}(\cdot)$ denote the induced graph, the set of all arcs, the set of all vertices which are inherited by a set or set system of graphs, arcs or vertices. With $\mathcal{P}(A)$ we denote the power set of a given set A.

For an arc $e = (u, v) \in E$ we denote $e^+ = u$ as the *tail* and $e^- = v$ as the *head* of the arc. Two arcs e and f are called *consecutive* if $e^- = f^+$ and are called *connected* if $\{e^-, e^+\} \cap \{f^-, f^+\} \neq \emptyset$. A *directed path* from a vertex u to a vertex v is a sequence of consecutive arcs where u is the tail of the first arc and v is the head of the last. A *connected path* from a vertex u to a vertex v is a sequence of connected arcs containing u and v as vertices. A digraph is *connected* if there is a connected path between every pair of its vertices. A *weighted digraph* (G, ω) or (V, E, ω), is a digraph with an additional weight function $\omega : E \longrightarrow \mathbb{R}$, which assigns a (usually positive) weight to each arc. For a given vertex $v \in V$ the sets $N_V^+(v) := \{u \in V \mid (v, u) \in E\}$, $N_V^-(v) := \{u \in V \mid (u, v) \in E\}$, $N_E^\pm(v) := \left\{e \in E \mid e^\pm = v\right\}$ shall denote the set of all outgoing or incoming vertices or arcs of v respectively. The *indegree* (respectively *out degree*) of a vertex u is given by $\deg^\pm(u) = |N_E^\pm(u)|$ and the degree of a vertex is $\deg(u) = \deg^-(u) + \deg^+(u)$. $\Delta^\pm(G)$, $\Delta(G)$ shall denote the maximal (in/out) degree, respectively.

A directed (connected) *cycle* of a digraph is a multiset of arcs $\{e_0, \ldots, e_k\}$ such that there is a permutation $\phi : \{0, \ldots, k\} \longrightarrow \{0, \ldots, k\}$ with $e_{\phi(i)}$ and $e_{\phi(i)+1 \mod k+1}$ are consecutive (connected), for $1 \leq i \leq k$. A *loop* is a cycle containing only a single arc. A cycle is *simple* if the set of contained arcs $\{e_1, \ldots, e_k\}$ is a simple set, i.e., it visits every arc, it contains, exactly once. A cycle is *elementary* if each vertex it contains is visited exactly once. We denote with $O(G)$ the set of all

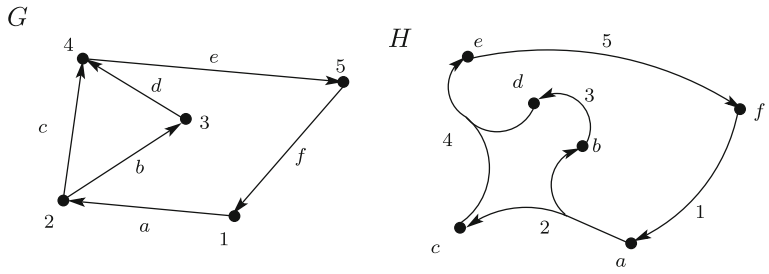

Fig. 1 The natural hyper graph $\mathcal{H}(G)$ of G

directed cycles and with $\mathrm{O}_{\mathrm{el}}(G)$, $\mathrm{O}_{\mathrm{si}}(G)$ the set of all elementary or simple cycles, respectively. Analogously, $\mathrm{O}^0_{\mathrm{el}}(G)$ and $\mathrm{O}^0_{\mathrm{si}}(G)$ shall denote the set of all *connected* (and not necessarily directed) *elementary* and *simple cycles* respectively. If not stated otherwise in the whole article a cycle is assumed to be directed and elementary. A *feedback vertex set* (FVS) of G is a set $\nu \in \mathcal{P}(V)$ such that $G \setminus \nu$ is *acyclic*, i.e., $G \setminus \nu$ contains no directed cycle. A *feedback arc set* (FAS) of G is a set $\varepsilon \in \mathcal{P}(E)$ such that $G \setminus \varepsilon$ is acyclic.

Definition 1 (line graph, natural hypergraph) The directed *line graph* $L(G) = (V_L, E_L)$ of a digraph G is a digraph where each vertex represents one of the arcs of G and two vertices are connected by an arc if and only if the corresponding arcs are consecutive. In contrast the *natural hypergraph* $\mathcal{H}(G) = (\bar{V}, \bar{E})$ of G is constructed by identifying the arcs of G with the vertices of $\mathcal{H}(G)$, i.e., $\bar{V}$ is a simple set of vertices such that $|V| = |E|$, where $|E|$ is counted with multiplicities. By fixing the identification $\bar{V} \cong E$ we introduce a directed hyperarc h_v for every vertex $v \in V$ by requiring that head and tail coincide with all outgoing and ingoing arcs respectively, i.e., $h_v = \left(N_E^-(v), N_E^+(v)\right)$. Consequently, $\bar{E} \cong V$ and therefore every hyperarc can be labeled by its corresponding vertex. See Fig. 1 for an example.

The directed, elementary cycles of the line graph $L(G)$ of G are in 1 to 1 correspondence to the directed, simple cycles of G while the directed simple cycles of $\mathcal{H}(G)$, i.e., directed cycles which run through every hyper arc exactly once are called *Berge cycles*, [5]. Note that if G is a simple digraph, i.e., there are no multi arcs, then the set of all Berge cycles of $\mathcal{H}(G)$ are in 1 to 1 correspondence to the set of all elementary cycles of G. For a set of hyperarcs $\bar{\varepsilon} \subseteq \bar{E}$ of $\mathcal{H}(G)$ we denote with $\varepsilon \subseteq V$ the corresponding vertices in G. We summarize some facts in this regard.

Proposition 1 *Let $G = (V, E)$ be a graph.*

i) *The line graph $L(G) = (V_L, E_L)$ can be constructed in $\mathcal{O}\left(|E|^2\right)$.*
ii) *The natural hypergraph $\mathcal{H}(G) = (\bar{V}, \bar{E})$ can be constructed in $\mathcal{O}\left(\Delta(G)|V|\right)$.*
iii) *$\nu \subseteq V_L$ is a FVS of $L(G)$ if and only if ν is a FAS of G.*
iv) *$\bar{\varepsilon} \subseteq \bar{E}$ is a FAS of $\mathcal{H}(G)$, with respect to the notion of Berge cycles, if and only if ε is a FVS of G.*

Proof Storing G as a adjacency list and following the definitions immediately implies $i)$ and $ii)$. Since any two vertices e, f of the line graph $L(G)$ are adjacent if and only if the corresponding arcs are consecutive in G and any two arcs h_u, h_v of $\mathcal{H}(G)$ are consecutive, i.e., $h_u^- \cap h_v^+ \neq \emptyset$, if and only if the corresponding vertices u and v are adjacent in G, $ii)$ and $iii)$ follow. □

Remark 1 Note that by introducing an additional arc h_v^* between head and tail of every hyperarc $h_v = (N_E^-(v), N_E^+(v))$ of $\mathcal{H}(G)$, the natural hypergraph becomes a directed graph $G^* = (V^*, E^*)$ with $|V^*| = |E| + 2|V|$, $|E^*| \leq \Delta(G)|V| + |V|$. The directed cycles of G^* are in 1 to 1 correspondence to the Berge cycles of $\mathcal{H}(G)$ and a FAS ε of G^* is in 1 to 1 correspondence to a FAS $\bar{\varepsilon}$ of $\mathcal{H}(G)$ by identifying $\bar{\varepsilon}$ with the additional introduced arcs belonging to the hyperarcs in $\bar{\varepsilon}$ and identifying ε with the hyperarcs corresponding to the bipartite graphs cutted by ε. If $\gamma : \bar{E} \longrightarrow \mathbb{R}+$ is an arc weight on $\mathcal{H}(G)$ then setting $\gamma^*(h) \equiv \gamma(h_v)$ for all $h \in N_E^-(v) \cup N_E^+(v) \cup \{h_v^*\}$ yields the translated weight.

To give a more algebraic notion of cycles we consider

$$X(G) := \bigoplus_{e \in E} \mathbb{Z}e$$

the free $\mathbb{Z}$-module generated by E. If we choose coordinates, i.e., a numbering for E and V then we can identify E with $\{e_1, \ldots, e_{|E|}\}$, V with $\{v_1, \ldots, v_{|V|}\}$ and X with $\mathbb{Z}^{|E|}$. In this case an element $x \in X(G)$ is a tuple $x = (x_1, \ldots, x_{|E|})$, which can be interpreted as a set of paths through G where $x_i \in \mathbb{Z}$ indicates how often we pass the arc e_i and the sign of x_i determines in which direction this is done. We denote with $\mathcal{I}(G)$ the incidence matrix of G with respect to these identifications, i.e., $\mathcal{I}(G) = (\iota_{ij})_{\substack{1 \leq i \leq |V| \\ 1 \leq j \leq |E|}}$ with

$$\iota_{ij} = \begin{cases} 0 \ , & \text{if} \quad {e_i}^+ \neq v_j \text{ and } {e_i}^- \neq v_j \\ 1 \ , & \text{if} \quad {e_i}^+ = v_j \text{ and } {e_i}^- \neq v_j \\ -1 \ , & \text{if} \quad {e_i}^+ \neq v_j \text{ and } {e_i}^- = v_j \end{cases} .$$

It is a well known fact, see for instance [8], that $x \in X(G)$ is a cycle of G if and only if $\mathcal{I}x = 0$, i.e., the submodule of all cycles of G coincides with the set of homogeneous solutions $\Lambda(G) = \ker \mathcal{I}(G)$. In particular, this implies that $\Lambda(G)$ is a free $\mathbb{Z}$-module with

$$\dim_{\mathbb{Z}} \Lambda(G) = \dim_{\mathbb{Z}}(\ker \mathcal{I}(G)) = |E| - |V| + \#G, \tag{1}$$

where $\#G$ denotes the number of connected components of G and therefore equals 1 by our assumption on G. Note that the vector space $\Lambda^0(G) := \Lambda(G)/\mathbb{Z}_2^{|E|}$ can be understood as the cycle space of connected cycles, given by the kernel of the incidence matrix $\mathcal{I}^0(G)$ defined with respect to $\mathbb{Z}_2$ coefficients. In this case

$$\dim_{\mathbb{Z}_2} \Lambda^0(G) = \dim_{\mathbb{Z}_2} \ker(\mathcal{I}^0(G)) = |E| - |V| + \#G, \tag{2}$$

still holds, see again [8].

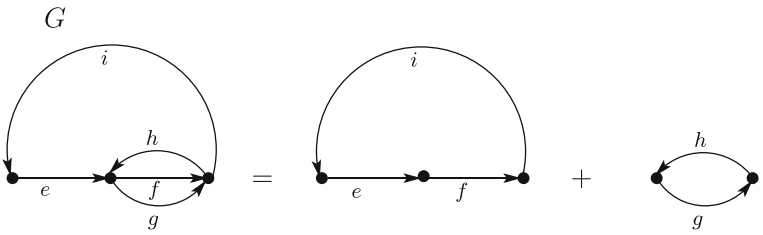

Fig. 2 Elementary and non-elementary cycles in G, see Example 1

Remark 2 If $c \in \mathrm{O}_{\mathrm{el}}(G)$ then 1 is the only non vanishing entry of c, i.e., $c \in \{0, 1\}^{|E|}$. Moreover, no elementary cycle is subset of another elementary cycle. Indeed assume the opposite and consider two elementary cycles $c, c' \in \mathrm{O}_{\mathrm{el}}(G)$ with $c \subseteq c'$, then $\mathcal{I}(G)(c' - c) = \mathcal{I}(G)c' - \mathcal{I}(G)c = 0 - 0 = 0$. Thus, $c' - c \neq 0$ is also a positive oriented cycle and therefore we have $c' = (c' - c) + c \notin \mathrm{O}_{\mathrm{el}}(G)$. A contradiction!

Not that every simple cycle is given by the union of arc disjoint elementary cycles. The following example illustrates this fact.

Example 1 Consider the graph G in Fig. 2. Then one observes that the cycle $c = \{e, f, g, h, i\}$ is a simple, non-elementary cycle while the cycles $\{e, f, i\}$, $\{f, h\}$ and $\{h, g\}$ are elementary cycles.

2.2 The Feedback Arc Set Problem (FASP)

Now we have all ingredients to give a formal definition of the FASP.

Problem 1 Let $G = (V, E)$ be a finite, connected, directed, and loop-free graph, $\omega : E \longrightarrow \mathbb{N}^+$ be a weight function. Then the *weighted FASP* is to find a set of arcs $\varepsilon \in \mathcal{P}(E)$ such that $G \setminus \varepsilon$ is acyclic, i.e., $\mathrm{O}_{\mathrm{el}}(G \setminus \varepsilon) = \emptyset$ and

$$\Omega_{G,\omega}(\varepsilon) := \sum_{e \in \varepsilon} \omega(e)$$

is minimized. We denote the set of solutions of this problem with $\mathcal{S}(G, \omega)$ and denote with $\Omega(G, \omega) := \Omega_{G,\omega}(\varepsilon)$, $\varepsilon \in \mathcal{S}(G, \omega)$ the optimal weight or *feedback length*. If ω is constant, e.g., equal to 1, then the problem coincides with the *unweighted minimal FASP*.

Remark 3 The condition on G to be loop-free is not an essential restriction. This is because every loop is contained in any solution of the minimal FASP.

Remark 4 Note that, every cycle $c \in O(G)$ can be generated by elementary cycles $c_1, \ldots, c_n \in \mathrm{O}_{\mathrm{el}}(G)$ using only non-negative coefficients. Thus, if ε is a solution of Problem 1 then certainly $O(G \setminus \varepsilon) = \emptyset$, which implies that our notion of acyclic graphs is consistent for the problem.

Remark 5 Note that for given graph $G = (V, E)$ the smallest subgraph $G_o \subseteq G$, which contains all cycles of G, i.e., $G' = G_o$ whenever $G' \subseteq G_o$ and $\mathrm{O_{el}}(G) = \mathrm{O_{el}}(G')$ can be constructed in $\mathcal{O}(|E|^2)$. Indeed the arc set of G_o is constructed from E by removing arcs (u, v) if there is no directed path from v to u. For every arc this can be done by depth-first search in linear time if G is stored in an adjacency list. Since removing arcs does not generate new cycles it suffices to check every arc once yielding the estimated runtime performance. Certainly a solution for G_o is a solution for G. We shortly denote with $\mathcal{G}_o(G) := G_o$ and with $\mathcal{G}_o^0(G) := G_o^0$ the analogous graph appearing by considering connected cycles instead of directed ones. In particular, by the argumentation above, an elementary or directed cycle $c \in \mathrm{O_{el}}(G)$, $c' \in \mathrm{O_{si}}(G)$ can be found in $\mathcal{O}(|E|^2)$ or no cycle exist.

Remark 6 If $G = (V, E)$ is a simple graph then denoting with $\lfloor \cdot \rfloor$ the Gauss bracket we observe that at most $\lfloor |E|/2 \rfloor$ arcs have to be deleted to obtain a graph where no connected path of length 2 exists anymore. In particular, the graph is acyclic in this case and therefore

$$\Omega(G, \omega) \leq \max_{e \in E} \omega(e) \cdot |E|/2.$$

See also [32].

2.3 The Feedback Vertex Problem (FVSP)

Let $G = (V, E)$ be given and $\gamma : V \longrightarrow \mathbb{R}^+$ be a vertex weight. The feedback vertex set problem (FVSP) on (G, γ) is obtained by replacing the role of arcs by vertices in Problem 1. In regard of Proposition 1, by introducing the hyperarc weight $\bar{\omega}(h_v) := \gamma(v)$, we realize that the FVSP is equivalent to the the FASP on the natural hypergraph $\mathcal{H}(G)$ of G, with respect to the notion of Berge cycles. Already in Remark 1 we mentioned how to treat this case. Vice versa the FASP on an arc weighted graph (G, ω) is equivalent to the FVSP on the line graph $L(G)$ of G by introducing the vertex weight $\gamma(v) = \omega(v)$, $v \in V_L = E$. Since the described transformations can be done efficently, see Proposition 1, an efficent solution of the FASP or FVSP for an arc and vertex weighted instance (G, ω, γ) yields an efficent solution of the FVSP or FASP for the transformed instances and vice versa. In particular, by summarizing some already known results we obtain:

Theorem 1 *The unweighted FASP and FVSP are APX complete.*

Proof Since there is an L-reduction of the Minimum Vertex Cover Problem, which is APX complete due to [12], to the FVSP, see [22], the FVSP is APX complete. Due to [21] it is known that the FASP is APX-hard. Proposition 1 shows that the FASP on (G, ω) is equivalent to the FVSP on $(L(G), \gamma)$. Thus, the feedback length of any solution remains unchanged yielding an L-reduction of the FASP to the FVSP implying the claim. □

We expect that the theorem above still holds for the weighted versions. However, due to the observations made so far, we will focus our studies on the FASP to increase the understanding of the localisation of feedback sets.

3 Main Results

Though the FASP and FVSP are equivalent problems in graph theory and computer sciences the only exact solutions of the FVSP known to us is are the algorithms of [30] with complexity $\mathcal{O}(1, 9977^{|V|})$ and [9] requiring $\mathcal{O}\left(|E|^4 4^{\Omega}\, \Omega^3 \Omega!\right)$, where Ω denotes the feedback length. A detailed comparison to our approach is given in Section 9. For now we just mention our results:

Theorem A *Let (G, ω) be a graph. Then there is an algorithm testing whether (G, ω) is resolvable and determing a solution of the weighted FASP on G in case of resolvability in $\mathcal{O}(|V||E|^3)$.*

Though there are infinitely many resolvable graphs not all graphs are resolvable. However, if the graph (G, ω) is not resolvable, we still can find an exact solution:

Theorem B *Let (G, ω) be a graph then there is a parameter $m \in \mathbb{N}$, $m \leq \dim_{\mathbb{Z}_2} \Lambda^0(G) = |E| - |V| + 1$, which can be determined in $\mathcal{O}(|E|^3)$ and an algorithm CUT with run time $\mathcal{O}\left(2^m |E|^4 \log(|V|)\right)$ solving the weighted FASP.*

Due to Proposition 1 the analogous results with respect to the FVSP hold. In particular, we call a vertex weighted graph (G, ν) resolvable iff its natural hypergraph is resolvable, see Section 2.3 again. If we replace every hyperarc of $\mathcal{H}(G) = (\bar{V}, \bar{E})$ with its corresponding bipartite graph then by following Remark 1 we have $|V^*| = |E| + 2|V|$, $|E^*| \leq (\Delta(G) + 1)|V|$ for the resulting graph $G^* = (V^*, E^*)$. The translated results therefore become:

Theorem C *Let (G, ν), $\nu : V \longrightarrow \mathbb{R}^+$ be a vertex weighted graph. Then there is an algorithm testing whether (G, ν) is resolvable and determing a solution of the weighted FVSP on G in case of resolvability in $\mathcal{O}(\Delta(G)^3 |V|^3 |E|)$.*

In case of non-resolvability we have:

Theorem D *Let (G, ν) be a graph then there is a parameter $m \in \mathbb{N}$, $m \leq (\Delta(G) - 1)|V| - |E| + 1$, which can be determined in $\mathcal{O}(\Delta(G)^3 |V|^3)$ and an algorithm CUT with run time $\mathcal{O}\left(2^m \Delta(G)^4 |V|^4 \log(|E|)\right)$ solving the weighted FVSP.*

Note that there are infinitely many instances where $m = 0$ and $m \leq |V|$ on homogenous graphs, i.e., if $\Delta(G) \leq (|E| - 1)/|V| + 2$. Thus, by computing the bound or directly m we can decide whether the algorithm of [30] or our approach will be faster for a given instance and choose the better alternative. Moreover, the feedback length Ω can not assumed to be constantly bounded. Thus, the algorithm of

[9] actually possesses a complexity of $\mathcal{O}\left(4^{|V|}|V|^3|E|^4|V|!\right)$, which is much slower than our approach. Finally, we want to mention that all theorems are based on the following crucial fact:

Theorem E *Let $G = (V, E)$ be a graph and $e \in E$. Then there exist algorithms which compute:*

i) *The subgraph $G_{\mathrm{el}}(e) \subseteq G$ induced by all elementary cycles $c \in \mathrm{O}_{\mathrm{el}}(G)$ with $e \in \mathcal{E}(c)$ in $\mathcal{O}(|E|^2)$.*
ii) *The subgraph $G_{\mathrm{si}}(e) \subseteq G$ induced by all simple cycles $c \in \mathrm{O}_{\mathrm{si}}(G)$ with $e \in \mathcal{E}(c)$ in $\mathcal{O}(|E|^4)$.*

A proof and a more precise version of the statement are given in Theorem 2.

4 The Essential Minor

In this section we introduce the notion of the essential minor (C, δ) of given graph (G, ω), which is a simple, weighted digraph that decodes the topological structure of G in a compact way and is therefore very helpful. Even though there are some crucial differences we want to mention that similar concepts were already introduced in [6].

Definition 2 (parallel arcs) Let G be a graph and $e = (u, v) \in E$ then we denote with

$$F^+(e) := \left\{ f \in E \mid f^+ = u\,,\, f^- = v \right\} \text{ and}$$
$$F^-(e) := \left\{ f \in E \mid f^+ = v\,,\, f^- = u \right\}$$

the sets of all parallel and anti-parallel arcs and set $F(e) = F^+(e) \cup F^-(e)$.

We recall that for a given set A and an equivalence relation $\sim$ on $A \times A$ the quotient $A/_\sim$ is given by the set of all equivalence classes $[a]_\sim = \{x \in A \mid x \sim a\}$.

Definition 3 (contracted graph) Let $G = (V, E)$ be a graph and $u, v \in V$. An equivalence relation $\sim_{u,v}$ on V is defined by

$$x \sim_{u,v} y \qquad \Longleftrightarrow \qquad x = y \quad \text{or} \quad x, y \in \{u, v\}.$$

The equivalence class of $x \in V$ is denoted by $[x]_{\sim_{u,v}}$ and $V/_{\sim_{u,v}}$ gives the quotient of V with respect to $\sim_{u,v}$. A multiset $E/_{\sim_{u,v}}$ is defined by

$$E/_{\sim_{u,v}} = \left\{ (p, q) \in V/_{\sim_{u,v}} \times V/_{\sim_{u,v}} \mid \exists (x, y) \in E : [x]_{\sim_{u,v}} = p\,,\, [y]_{\sim_{u,v}} = q \right\}.$$

The *contracted graph* of G with respect to $e \in E$ is defined as the topological minor

$$G/e := \left(V/_{\sim_{e^+,e^-}}, (E \setminus F(e))/_{\sim_{e^+,e^-}} \right). \tag{3}$$

If $e, f \in E$ then one can check easily that by identifying e, f with their images in G/e, G/f respectively we have $(G/e)/f = (G/f)/e$. Thus, the definition does not

depend on the order of the contracted edges. Hence, if $G' \subseteq G$ is a subgraph then $G/G' := G/\mathcal{E}(G')$ can be defined by contracting $\mathcal{E}(G')$ in arbitrary order.

Definition 4 (essential minor) Let $G = (V, E, \omega)$ be a positively weighted graph. The equivalence relation $\sim_\Gamma$ on E is defined by $e \sim_\Gamma f$ if and only if $e = f$ or there exists a directed, branch-point-free path $(w_0, w_1, \ldots, w_n, w_{n+1})$ with $\{(w_0, w_1), (w_n, w_{n+1})\} = \{e, f\}$ and $n \in \mathbb{N}^+$, i.e., for $i = 1, \ldots, n$ it holds that $\deg^-(w_i) = \deg^+(w_i) = 1$. We represent an equivalence class $[e]_{\sim\Gamma}$ by an arc (u, v), where u and v coincide with the start and endpoint of the longest directed, branch-point-free path running through e. The positively weighted graph

$$(G/_\Gamma, \omega/_\Gamma)) := (V/_\Gamma, E/_\Gamma, \omega/_\Gamma))$$

is defined by

i) $V/_\Gamma := \{v \in V \mid \deg^-(v) > 1 \vee \deg^+(v) > 1\}$,
ii) $E/_\Gamma := \left\{(u, v) \in V/_\Gamma \times V/_\Gamma \mid \exists\, [e]/_{\sim\Gamma} \in E/_{\sim\Gamma} \text{ represented by } (u, v)\right\}$
iii) $\omega/_\Gamma : E/_\Gamma \longrightarrow \mathbb{N}^+$ with $\omega/_\Gamma(e) := \min_{e' \in [e]/_{\sim\Gamma}} \omega(e')$.

Let $\sim_\Phi$ be an equivalence relation on E with $e \sim_\Phi f \iff e, f \in F^+(e)$. For (G, ω) the positively weighted graph

$$(G/_\Phi, \omega/_\Phi) := (V/_\Phi, E/_\Phi, \omega/_\Phi)$$

is defined by

i) $V/_\Phi := V$.
ii) $E/_\Phi := E/_{\sim_\Phi}$, where we identify each equivalence class $[e]_{\sim\Phi}$ with an arbitrary representative of $F^+(e)$.
iii) $\omega/_\Phi : E/_\Phi \longrightarrow \mathbb{N}^+$, $\omega/_\Phi(e) := \sum_{e' \in [e]_{\sim\Phi}} \omega(e')$.

Starting with $G_0 := G$ for $k > 0$ we define

$$(G_k, \omega_k) := (G'_{k-1}/_\Phi, \omega'_{k-1}/_\Phi) \text{ with } (G'_{k-1}, \omega'_{k-1}) := (G_{k-1}/_\Gamma, \omega_{k-1}/_\Gamma).$$

The weighted graph (G_K, ω_K) with $G_K = G_{K+1}$, $K \geq 0$, is called the *essential minor* of G and is denoted by $(C, \delta) := (V_C, E_C, \delta)$.

Example 2 In Fig. 3 the construction of the essential minor is illustrated. Furthermore, for a graph G with D diamonds connected in a cycle as in the example we obtain $\deg(G) = 3$ and $|\mathrm{O_{el}}(G)| = 2^D = 2^{|E|/5} = 2^{|V|/4}$. In contrast the essential minor C of such a graph satisfies $|\mathrm{O_{el}}(C)| = 1$. Hence, even though the number of cycles in G increases exponentially in $|E|$ and $|V|$ by adding further diamonds, the number of cycles in C remains constant equal to 1 while the weight ξ decodes the number of cycles of the original graph G.

Since the following results are quite canonically their simple but technical proofs are given in Appendix A.

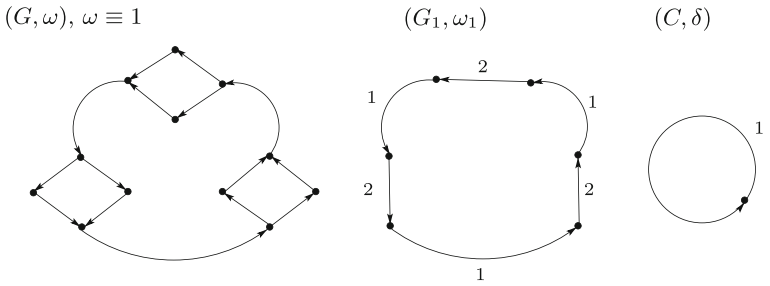

Fig. 3 The construction of (C, δ), with respect to (G, ω)

Proposition 2 *Let $G = (V, E, \omega)$ be a positively weighted graph with essential minor (C, ω_C) and let $\varepsilon \in \mathcal{P}(E)$ and ε_C be the image of ε in (C, δ). Then*

$$\varepsilon \in \mathcal{S}(G, \omega) \Longleftrightarrow \varepsilon_C \in \mathcal{S}(C, \delta)\,.$$

In particular $\Omega(G, \omega) = \Omega(C, \omega_C)$.

Proposition 2 states that solving the FASP for the essential minor is equivalent to solving the FASP on the original graph. Even though it is possible that $(C, \delta) = (G, \omega)$ the construction might yields an a priori problem size reduction in many cases as in Example 2.

Proposition 3 *Let $G = (V, E, \omega)$ be a finite, connected, directed, weighted multigraph then we can construct (C, δ) in time $\mathcal{O}(|E|^2)$. Furthermore, there is an algorithm with run time $\mathcal{O}(|E|^2)$ which constructs a solution $\varepsilon \in S(G, \omega)$ given a solution $\varepsilon_C \in \mathcal{S}(C, \delta)$.*

Remark 7 Since for given $\varepsilon_C \in \mathcal{S}(C, \delta)$ the construction of some $\varepsilon_G \in \mathcal{F}(\varepsilon_C)$ is easy to compute (Proposition 2) Proposition 2 states that it suffices to solve a the weighted FASP for the essential minor instead of the original graph. As a consequence multigraphs do not need to be considered and the number of elementary cycles of the essential minor can be drastically reduced, see for instance Example 2.

5 Subgraphs of Elementary Cycles

There are several approaches for generating the set $\mathrm{O}_{\mathrm{el}}(G)$ of all elementary cycles of a graph, see [28] for an overview. Since there can be an exponential number of cycles in a graph, generating algorithms have an exponential worst case run time. The best algorithms available today are the ones of [34] and [20] solving the problem in $\mathcal{O}\left(|\mathrm{O}_{\mathrm{el}}(G)|(|V| + |E|)\right)$. Of course counting all cycles might be less expansive than generating them. However, by reducing to the Hamiltonian cycle problem, see for instance [2], counting all cycles is a NP-hard problem. For our concerns, and supposedly in many other situations, the generation of all cycles is not necessary, but the knowledge of the arc set of all cycles including a common arc suffices. In

the following an algorithm for determing the smallest subgraph $G_{\mathrm{el}}(e) \subseteq G$ which contains all elementary cycles that include the arc e is given.

Definition 5 Let $G = (V, E)$ be a graph and $u, v \in V$ we denote with with $P(u, v)$, $P_{\mathrm{el}}(u, v)$, $P_{\mathrm{si}}(u, v)$ the set of all directed, elementary or simple paths from u to v respectively. For an arc $e \in E$ we let

$$\begin{aligned} \mathrm{O}_{\mathrm{el}}(e) &:= \{c \in \mathrm{O}_{\mathrm{el}}(G) \mid c \cap e \neq \emptyset\} = \{e\} \cup \mathrm{P}_{\mathrm{el}}(e^-, e^+) \text{ and} \\ \mathrm{O}_{\mathrm{si}}(e) &:= \{c \in \mathrm{O}_{\mathrm{si}}(G) \mid c \cap e \neq \emptyset\} = \{e\} \cup \mathrm{P}_{\mathrm{si}}(e^-, e^+) \end{aligned}$$

be the set of all elementary and simple cycles running through e. If $\varepsilon \in \mathcal{P}(E)$ then we set $\mathrm{O}_{\mathrm{el}}(\varepsilon) := \cup_{e\in\varepsilon}\mathrm{O}_{\mathrm{el}}(e)$, $\mathrm{O}_{\mathrm{si}}(\varepsilon) := \cup_{e\in\varepsilon}\mathrm{O}_{\mathrm{si}}(e)$. Moreover, we denote with $G(u, v) := \mathcal{G}(P(u, v))$, $G_{\mathrm{el}}(u, v) := \mathcal{G}\left(P_{\mathrm{el}}(u, v)\right)$, $G_{\mathrm{si}}(u, v) := \mathcal{G}\left(P_{\mathrm{si}}(u, v)\right)$ the by the corresponding paths induced graphs and with $G(e) := \mathcal{G}(P(u, v))$, $G_{\mathrm{el}}(e) := \mathcal{G}\left(P_{\mathrm{el}}(e^-, e^+) \cup \{e\}\right)$, $G_{\mathrm{si}}(e) := \mathcal{G}\left(P_{\mathrm{si}}(e^-, e^+) \cup \{e\}\right)$ by the corresponding cycles induced graphs. Moreover, $P^0(u, v)$, $\mathrm{P}^0_{\mathrm{el}}(u, v)$, $\mathrm{P}^0_{\mathrm{si}}(u, v)$, $O(e)$, $O^0_{\mathrm{el}}(e)$, $O^0_{\mathrm{si}}(u, v)$, $G^0(e)$, $G^0_{\mathrm{el}}(e)$, $G^0_{\mathrm{si}}(e)$ shall denote the connected (and not necessarily directed) analagons of the introduced sets and graphs.

Note that $P_{\mathrm{el}}(u, v) \subseteq P_{\mathrm{si}}(u, v)$ and therefore $\mathrm{O}_{\mathrm{el}}(e) \subseteq \mathrm{O}_{\mathrm{si}}(e)$. Moreover, the graphs $G(u, v)$, $G(e)$ can be determined in $\mathcal{O}(|E|^2)$ by applying a depht first search technique similar to Remark 5. In the other cases we observe:

Theorem 2 *Let $G = (V, E)$ be a graph and $u, v \in V$, $e \in E$. Then there exist algorithms which compute:*

i) *The graphs $G_{\mathrm{el}}(u, v)$, $G^0_{\mathrm{el}}(u, v)$, $G_{\mathrm{el}}(e)$, $G^0_{\mathrm{el}}(e)$ in $\mathcal{O}(|E|^2)$.*
ii) *The graphs $G_{\mathrm{si}}(u, v)$, $G^0_{\mathrm{si}}(u, v)$, $G_{\mathrm{si}}(e)$, $G^0_{\mathrm{si}}(e)$ in $\mathcal{O}(|E|^4)$.*

Proof Let $p \in P_{\mathrm{el}}(u, v)$ then no vertex $w \in \mathcal{V}(p)$ is passed twice of p. Thus, for every $f \in \mathcal{E}(p)$ there is a path $q \in P(u, f^-)$ with respect to $G \setminus \left(N_E^-(f^-) \setminus \{f\}\right)$. Vice versa if $p \in P(u, v)$ is such that $\mathcal{E}(p) \not\subseteq G_{\mathrm{el}}(u, v)$ then has to be a vertex $w \in \mathcal{V}(p)$, which is passed at least twice by p implying that there is $f \in \mathcal{E}(p)$ such that

$$P(u, f^-) = \emptyset \quad \text{with respect to} \quad G \setminus \left(N_E^-(f^-) \setminus \{f\}\right). \tag{4}$$

Thus, by setting $G' := G \setminus \{f \in E \mid f \text{ fulfills (4)}\}$, every path $p \in P(u, v)$ with $\mathcal{G}(p) \not\subseteq G_{\mathrm{el}}(u, v)$ is interrupted in G'. Hence, $G'(u, v) = \mathcal{G}(P(u, v))$ with respect to G' coincides with $G_{\mathrm{el}}(u, v)$. Algorithm 1 formalizes this procedure and runs in $\mathcal{O}(|E|^2)$ if G is stored in an adjacency list, enabling us to test whether $P(u, v) = \emptyset$ in $\mathcal{O}(|E|)$. The other cases of $i)$ can now be solved by replacing u, v with e and directed paths or cycles with connected ones.

Algorithm 1 The induced subgraph $G_{\mathrm{el}}(u, v)$

Input: $G = (V, E), e = (u, v) \in E$
Output: $G_{\mathrm{el}}(e)$
$G \leftarrow G(u, v)$;
$E^* \leftarrow \emptyset$;
for $f \in E'$ **do**
  if $P(u, f^-) = \emptyset$ *w.r.t.* $G' \setminus \left(N_E^-(f^-) \setminus \{f\}\right)$ **then**
    $E^* \leftarrow E^* \cup \{f\}$;
  end
end
$G_{\mathrm{el}}(u, v) \leftarrow G(u, v)$ w.r.t. $G \setminus E^*$;
return $G_{\mathrm{el}}(u, v)$

To show $ii)$ we add two arcs $u^* = (x, u)$, $v^* = (v, y)$, $x, y \notin V$ denote with G^* the resulting graph and consider the line graph $L(G^*) = (V_L^*, E_L^*)$. We recall that $|V_L| = |E|$, $|E_L| \leq |E^2|$ and apply the fact that the elementary paths of $L(G)$ are in 1 to 1 correspondence to the simple paths of G and therefore $\mathcal{V}(G_{\mathrm{el}}(u^*, v^*)) \setminus \{u^*, v^*\} = \mathcal{E}(G_{\mathrm{si}}(u, v))$. Hence $ii)$ follows analogue to $i)$. □

Remark 8 Note that if (C, δ) is the essential minor of (G, ω). Then the treatment of "parallel" paths is avoided by the essential minor construction. Thus, we expect that if C is significant smaller than G the run time performance will increase drastically.

6 Isolated Cycles

Of course the question arises whether a solution of the FASP on $G_{\mathrm{el}}(e)$ can be determined independently of the rest of the graph. The notion of *isolated cycles* is our starting point of investigations in this manner and as it will turn out it is a very helpful concept of answering this question.

We recall that a Min-s-t-Cut with source $s = u$ and sink $t = v$ is given by a set $\varepsilon \subseteq E$ such that $P(u, v) = \emptyset$ in $G \setminus \varepsilon$ and $\Omega_{G,\omega}(\varepsilon) = \sum_{e \in \varepsilon} \omega(e)$ is minimized.

Lemma 1 *Let (G, ω) be a weighted graph and $e \in E$. Then there is an algorithm, which determines a solution $\varepsilon \in \mathcal{S}(G_{\mathrm{el}}(e), \omega)$ of the FASP on $(G_{\mathrm{el}}(e), \omega)$ in $\mathcal{O}(|V||E|\log(|V|))$, where we slightly abused notion by still denoting ω for the the restriction of ω to $G_{\mathrm{el}}(e)$.*

Proof Observe that by interpreting ω as a capacity function on $G_{\mathrm{el}}(e)$ a solution of the FASP on ε is given by $\{e\}$ or a Min-s-t-Cut ε with source $s = e^-$ and sink $t = e^+$. The option with the smaller weight is chosen. Due to the famous Min-Cut-Max-Flow Theorem a Min-s-t-Cut can be determined by solving a Max-Flow problem with respect to ω and $s = e^-$, $t = e^+$. The algorithm of [10] solves the Max-Flow problem for arbitrary weights in time $\mathcal{O}(|V|^2|E|)$ and can be speedend up

Algorithm 2 The isolated cycles of a graph

Input: $e = (u, v) \in E$
Output: $G_I(e)$
$G \leftarrow G_{\mathrm{el}}(e)$;
$E^* \leftarrow \{f \in E_{\mathrm{el}} \,|\, \mathrm{O}_{\mathrm{el}}(f) \neq \emptyset \text{ in } G \setminus e\}$;
$G_I(e) \leftarrow G_{\mathrm{el}}(e)$ with respect to $G \setminus E^*$;
return $G_I(e)$

to $\mathcal{O}(|V||E|\log(|V|))$, [11] by using the data structure of dynamic trees. Thus the statement is proven. □

Remark 9 Note that if (C, δ) is the essential minor of (G, ω) then the absence of "parallel" paths might speeds up the time required to determine a Min-s-t-Cut drastically. Moreover, the Max-Flow-Problem is very well understood, yielding many alternatives to the algorithm of [10] and providing faster solutions in special cases, see [11] and [3] for an overview.

Definition 6 Let G be a graph and $c \in \mathrm{O}_{\mathrm{el}}(G)$ then we denote with

$$I(c) := \left\{e \in \mathcal{E}(c) \;\middle|\; c \cap c' = \emptyset\,,\ \forall c' \in \mathrm{O}_{\mathrm{el}}(G) \setminus \mathrm{O}_{\mathrm{el}}\left(F^+(e)\right)\right\}$$

the set of *isolating arcs* of c, i.e., if $e \in I(c)$ then c has empty intersection with every cycle c' that does not contain e or an parallel arc of e. For an arc $e \in E$ or set of arcs $\varepsilon \subseteq E$ we set

$$O_I(e) := \{c \in \mathrm{O}_{\mathrm{el}}(G) \;|\; e \in I(c)\}\,, \quad O_I(\varepsilon) = \bigcup_{e \in \varepsilon} O_I(e)$$

and $G_I(e) := \mathcal{G}\left(O_I(F^+(e))\right)$.

Remark 10 Note, that the sets of isolated cycles possess a *flat hierarchy* in the following sense. If $e, f \in E, e \neq f$, with $O_I(e) \neq O_I(f)$ then $O_I(e) \cap O_I(f) = \emptyset$. If vice versa $O_I(e) = O_I(f)$ then by definition we obtain $\mathrm{O}_{\mathrm{el}}(F^+(e)) = \mathrm{O}_{\mathrm{el}}(F^+(f))$.

Remark 11 Let $c \in O_I(G)$ be an isolated cycle and $I(c)$ the set of all isolating arcs of c. If we contract $I(c)$ then the resulting graph $G/I(c)$ fulfills

$$\left\langle c', c''\right\rangle = 0\,, \quad \text{for all } c' \in O_I(e)/I(c), c'' \in (O_{\mathrm{el}}(G/I(c)) \setminus (O_I(e)/I(c))\,,$$

where $\langle\cdot,\cdot\rangle$ denotes the standard scalar product on $\mathbb{R}^n$, $n = |E/I(c)|$. Thus, by detecting isolated cycles we obtain an orthogonal splitting

$$\Lambda(G/I(c)) = \mathrm{span}\,(O_I(e)/I(c)) \oplus \mathrm{span}\,(O_{\mathrm{el}}(G/I(c)) \setminus (O_I(e)/I(c))\,.$$

Such a splitting is certainly helpful whenever one wants to find a basis of $\Lambda(G)$, e.g., a minimal cycle basis of span $(O_I(e))$ can be extended to a minimal basis of $\Lambda(G)$.

Consider an isolating arc e of a graph (G, ω) or its essential minor (C, δ). The isolated cycles $O_I(e)$ running through e can be cut either by removing the arc set

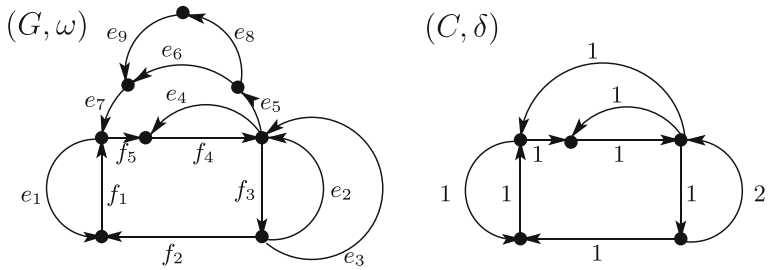

Fig. 4 A graph (G,ω) and its essential minor (C,δ). The numbers at the arcs of C indicate the value of δ. $\{e_1, f_1\}$, $\{e_2, f_3\}$, $\{e_3, f_3\}$, $\{e_4, f_4\}$, $\{e_5, e_6, e_7, f_5, f_4\}$, and $\{e_5, e_8, e_9, e_7, f_5, f_4\}$ are isolated cycles of G and $I(G) = \{f_1, f_3, f_4\}$

$\varepsilon_0 = F^+(e)$ or another feedback set ε_1 of $G_I(e)$. By definition the arc set ε_0 cuts at least the cycles in $O_I(\varepsilon_0)$. By Remark 10 the feedback set ε_1 cuts only the isolated cycles $O_I(\varepsilon_0)$ or is given by $\varepsilon_1 = F^+(f)$ of another isolating arc $f \in O_I(e)$ with $\mathrm{O_{el}}(\varepsilon_0) = \mathrm{O_{el}}(\varepsilon_1)$. Thus, if the weight of ε_0 equals the weight of a solution of the FASP on $G_I(e)$ then there is a solution ε of the FASP on G with $\varepsilon_0 \subseteq \varepsilon$. The following definition reflects this idea more formally.

Definition 7 Let (G,ω) be a graph. We define a maximal list of graphs $(G_0,\omega_0),\ldots,(G_k,\omega_k)$ with $(G_0,\omega_0) = (G,\omega)$, $(G_i,\omega_i) \neq (G_{i+1},\omega_{i+1})$, $\forall\, i \in [1:k-1]$ as follows. Let (C_i,δ_i), with $C_i = (V_{C_i}, E_{C_i})$, be the essential minor of (G_i,ω_i) and $E_i^* \subseteq I(C_i)$ be a maximal subset of pairwise different isolating arcs of C_i such that $\forall e \in E_i^*$:

$$\delta(e) = \Omega\left(G_I(e), \delta_{|G_I(e)}\right), \quad G_{\mathrm{el}}(e) \neq G_{\mathrm{el}}(f), \text{ whenever } e \neq f. \tag{5}$$

Then the weighted graph (G_{i+1},ω_{i+1}) is given by

$$G_{i+1} := (V_i, E_i) = \mathcal{G}_o\left(E_{C_i} \setminus E_i^*\right), \quad \omega := \delta_{i|E_i'}$$

where $\delta_{i+1} := \delta_{i|E_{i+1}}$ denotes the restriction of δ_i to E_{i+1}. If $G_{k+1} = G_k$ for some $k \in \mathbb{N}$ then $(S,\tau) := ((V_S,E_S),\tau) = (C_k,\delta_k)$, $k \in \mathbb{N}$ is called the *resolved graph* of G, which we shortly denote with $(S,\tau) = (S(G),\tau(\omega))$. A graph (G,ω) is called *resolvable* if and only if $S = \emptyset$.

Example 3 Note that (C,δ) in Fig. 4 is resolvable, while (G,ω) in Fig. 6 is not resolvable, but becomes resolvable for uniform weight $\omega \equiv 1$.

The construction has an immediate consequence.

Theorem 3 *Let (G,ω) be a graph. Then there is an algorithm testing whether (G,ω) is resolvable and determing a solution of the weighted FASP on G in case of resolvability in $\mathcal{O}(|V||E|^3)$.*

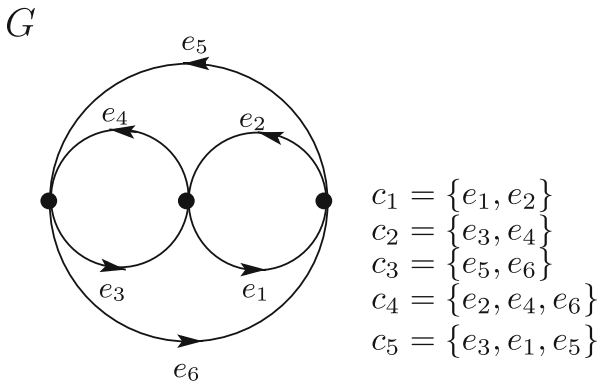

Fig. 5 The smallest graph without isolated cycles

Proof Due to Proposition 3 the construction of the essential minor (C_i, δ_i) can be achieved in $\mathcal{O}(|E|^2)$ for every $1 \leq i \leq k$. Since checking whether $\mathrm{O_{el}}(f) \neq \emptyset$ can be done in $\mathcal{O}(|E|)$ by storing G in an adjacency list and using depht first search to figure out whether $P(f^+, f^-) \neq \emptyset$ the Algorithm 2 computes the set $G_I(e)$ in $\mathcal{O}(|E|^2)$ and therefore computing $G_I(e)$ for all arcs requires at most $\mathcal{O}(|E|^3)$ computation steps. Furthermore, a solution of the FASP on $G_I(e)$ can be computed due to Lemma 1 in $\mathcal{O}(|E|^2)$. Due to the fact that during the construction of (S, τ) no parallel arcs appear, we have to recompute the isolated cycles at most $|V|$ times. Thus, (S, τ) can be determined in $\mathcal{O}(|V||E|^3)$. Furthermore, we can use the backtracking procedure of Proposition 3 to compute a solution of the FASP in $\mathcal{O}(|V||E|^2)$ once (S, τ) is known. □

Observe that Theorem 3 was already stated in Section 3 as Theorem A. However, the result leads to the question : What are fast (linear, quadratic time) checkable conditions a graph (G, ω) has to satisfy to be resolvable. Though, we can easily construct resolvable graphs as (C, δ) in Fig. 4 or modified versions of (C, δ) by adding additional isolated cycles a characterization of resolvable graphs is still open. A better understanding of the non-resolvable graphs might help to solve that problem. In order to investigate these graphs the class of graphs without isolated cycles at all, seems to be interesting. Therefore, the next result might be a good starting point for further studies.

Proposition 4 *The directed clique $D_3 = (V, E)$ in Fig. 5 is the smallest graph with $\mathrm{O_{el}}(G) \neq \emptyset$ and $O_I(G) = \emptyset$, i.e. any non-isomorphic graph $G' = (V', E')$ with $O(G') \neq \emptyset$ and $O_I(G') = \emptyset$ satisfies $|V'| + |E'| > |V| + |E|$.*

Proof If G' is a graph with $O(G') \neq \emptyset$, $O_I(G') = \emptyset$ then G' possesses at least three vertices and $\left|O(G')\right| \geq 3$ has to hold. We claim that there are at least three linear

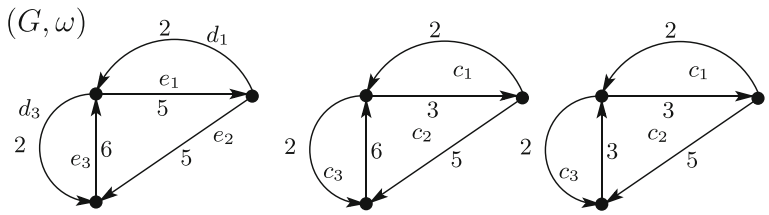

Fig. 6 The computation of the relative weights with respect to c_3

independent cycles. Indeed if $d_1, d_2, d_3 \in O(G')$ with $\lambda d_1 + \mu d_2 + \eta d_3 = 0, \lambda, \mu, \eta \in \mathbb{Z}$ then due to Remark 2 we know that $d_1, d_2, d_3 \in \{0, 1\}^{|E|}$ and no elementary cycle is subset of another. So w.l.o.g. we can assume that $d_1 = d_2 + d_3$, which contradicts that all cycles are elementary. Thus, $\dim_{\mathbb{Z}} \Lambda(G') = |E'| - |V'| + 1 \geq \dim_{\mathbb{Z}} O(G') \geq 3$. This observation implies that $|E'| \geq |E|$ whenever $|V'| > 3$. Since D_3 is an directed clique we can identify any smaller graph G' with a subgraph of D_3. It is easy to see that deletion of any arc $e \in E$ produces an isolated cycle in D_3. For instance if we delete e_4 then c_3 will be isolated, if we delete e_5 then c_2 will be isolated and so on. Hence $G' \cong D_3$. □

If G is a non-resolvable graph then one can think about different methods to solve the FASP of the resolved graph (S, τ). One possibility is discussed in the next section.

7 The Bellman Decomposition

In this section we formulate an solution of the FASP based on a dynamic programming technique. Such an approach can be applied to optimization problems whenever there is a decomposition of the problem into subproblems which satisfy the Bellman principle, i.e., every optimal solution consists only of optimal subsolutions. To motivate the following definitions we first consider an example.

Example 4 Consider the graph (G, ω) in Fig. 6. If we want to know, which arc of c_3 we have to cut for an optimal solution then this depends on the cycles c_1, c_2. The benefit of cutting e_1 instead of e_2 or e_3 is that we do not have to cut c_1 anymore which costs at least 2. Thus we introduce a new weight σ, which equals ω on $E \setminus \{e_1\}$ and is set to $\sigma(e_1) = \omega(e_1) - 2 = 3$ on e_1. Since no other cycles than c_3 are cut by d_3 the weight of d_3 remains unchanged. Now we consider $H_{e_3,d_3} = G_{\mathrm{el}}(e_3) \setminus d_3$ and $H_{d_3,e_3} = G_{\mathrm{el}}(d_3) \setminus e_3$ and compute

$$\begin{aligned} \big(\omega(e_3) - \Omega(H_{e_3,d_3}, \sigma)\big) \; - \; \big(\omega(d_3) - \Omega(H_{d_3,e_3}, \sigma)\big) \\ = (\omega(e_3) - 3) \; - \; (\omega(d_3) - 0) = 1 \end{aligned} \tag{6}$$

The best solution, which contains e_3 is $\{e_3, d_1\}$ and the best solution containing d_3 is $\{d_3, e_1\}$ and we observe that

$$(\omega(e_3) + \omega(d_1)) - (\omega(d_3) + \omega(e_1)) = 8 - 7 = 1\,.$$

Thus, the difference of the solutions coincides with the difference of the subproblems in (6) with respect to the new weight σ.

We need to introduce several concepts to show that this observation remains true in general.

Definition 8 (arc sensitivity) Let (G, ω) be a graph and $e, f \in E$, $e \neq f$ and $G_{\mathrm{el}}(e) = (V_e, E_e)$ be given. Then we say that f is *arc sensitive* to e with respect to the FASP, denoted by $f \twoheadrightarrow e$, if and only if

$$f \in E_e \quad \text{and} \quad \mathrm{O}_{\mathrm{el}}(f) \neq \emptyset \ \text{ w.r.t. } G \setminus e\,.$$

We denote with $\mathcal{N}_{\twoheadrightarrow}(e) = \{f \in E \mid f \twoheadrightarrow e\}$ the set of all arcs, which are sensitive to e.

Note that the arcs f of an isolated cycle $c \in O_I(e)$ can not be sensitive to e. Thus, arc sensitivity detects arcs, which might prevent us from solving the FASP on $G_{\mathrm{el}}(e)$ independently from the rest of the graph. An understanding of these dependencies can be reached by understanding the *meta graph* of G defined in the following.

Definition 9 (meta graph) Let (G, ω) be a graph and $c \in \mathrm{O}_{\mathrm{el}}(G)$. We set $V_0 = \mathcal{E}(c)$, $E_0 = W_0 = \emptyset$ and for $k \geq 1$ we define recursively $W_k = \cup_{i=0}^{k} V_i$ with

$$\begin{aligned} V_k &:= \bigcup_{h \in V_{k-1}} \{\mathcal{N}_{\twoheadrightarrow}(h) \ \text{ w.r.t. } \ (G \setminus (W_{k-1} \setminus \{h\}), \omega)\}\,, U_k = V_k \cup V_{k-1} \\ E_k &:= \{[h, f] \in U_k \times U_k \mid f \twoheadrightarrow h \ \text{ w.r.t. } \ (G \setminus (W_{k-1} \setminus \{h\}), \omega)\}\,, \end{aligned}$$

Stopping the recursion if $K \in \mathbb{N}$ is such that $V_K = \emptyset$ we introduce the simple, undirected graph $M_c := (V_{M_c}, E_{M_c}) = \bigcup_{k=0}^{K}(V_k, E_k)$ as the *meta graph* of G with respect to c. Furthermore, we introduce $\mathcal{C}(c) = \mathcal{G}(\mathrm{O}_{\mathrm{el}}(V_{M_c})) \subseteq G$ as the subgraph of all *arc sensitive cycles* containing c.

Algorithm 3 The relative weight for meta trees

Input: (G, ω), M, e, $f \in V_M$
Output: $\sigma_{G,M,e,f}$
$M := (V_M, E_M) \leftarrow M_{c,f}$;
$G' \leftarrow G \setminus f$;
$\mathcal{L} \leftarrow \mathcal{L}_e(M) := \{h \in V_M \setminus \{e\} \mid \deg(h) = 1\}$;
$\sigma \leftarrow \omega$;
while $\mathcal{L} \neq \emptyset$ **do**
  for $h \in \mathcal{L}$ **do**
    $G_h \leftarrow G' \setminus (\{f\} \cup (V_M \setminus \mathcal{L}))$;
    $\sigma(h) \leftarrow \sigma(h) - \Omega(G_{\mathrm{el}}(h), \sigma)$ w.r.t. G_h;
  end
  $M \leftarrow M \setminus \mathcal{L}$;
  $\mathcal{L} \leftarrow \mathcal{L}_e(M)$;
end
return σ ;

Lemma 2 *Let $G = (V, E)$ be a graph, $c \in \mathrm{O}_{\mathrm{el}}(G)$. Then we can construct the meta graph $M_c = (V_M, E_M)$ in $\mathcal{O}(|E|^3)$.*

Proof Storing G in an adjacency list enables us to test whether $\mathrm{O}_{\mathrm{el}}(e) \neq \emptyset$ in $\mathcal{O}(|E|)$ by depht first search, which we have to do at most $|E|^2$ times. Due to Theorem 2 the graph $G_{\mathrm{el}}(e)$ can be determined in $\mathcal{O}(|E|^2)$. Due to the fact that $V_k \cap V_{k-1} = \emptyset$, $1 \leq k \leq K$ we have to construct Gel(e) at most $|E|$ times, which yields the claimed complexity. □

Next we define the relative weight $\sigma_{G,c,e,f}$ with respect to some $c \in \mathrm{O}_{\mathrm{el}}(G)$ and $e, f \in \mathcal{E}(c)$. As it will turn out $\sigma_{G,c,e,f}$ decodes which arcs of c can be cutted to obtain a minimal feedback set. For a better clarity we firstly restrict ourselves to the case where the meta graph $M_c \setminus f$ is a tree. In this case, e is chosen as the root and $\sigma_{G,c,e,f}$ is given by solving the FASP for every leaf h on

$$(G_{\mathrm{el}}(h), \omega) \setminus \{\{f\} \cup \text{inner nodes of } M_c \setminus f\}$$

and subtracting this value from the weight $\omega(h)$ of the predecessor of h. Afterwards, we delete all leafs of $M_c \setminus f$ and iterate this procedure till e becomes a leaf. More precisely:

Definition 10 (relative weight for meta trees) Let (G, ω) be a weighted graph $c \in \mathrm{O}_{\mathrm{el}}(G)$, $e, f \in \mathcal{E}(c)$. Let M_c be the meta graph of G with respect to c and assume that the connected subgraph $M_{c,f}$ of $M_c \setminus f$, which contains e is a tree. Then we define the *relative weight* of G with respect to c, e, f

$$\sigma_{G,M,e,f} : E \longrightarrow \mathbb{R}$$

as the output of Algorithm 3 with input $((G, \omega), M = M_c, e, f)$.

Algorithm 4 The relative weight for arbitrary meta graphs

Input: (G, ω), $M = (V_M, E_M)$, $e \in V_M$
Output: $\sigma_{G,M,e}$
$M := (V_M, E_M) \leftarrow M_{c,f}$;
$W \leftarrow \text{FILO}\ \{e\}$, $Q \leftarrow \text{FILO}\ \{e\}$, $K \leftarrow K(M, e)$, $U_{M_W} \leftarrow U(M, e)$;
while $W \neq \emptyset$ **do**
  if M_W *is a tree* **then**
    $W \mapsto h$, $Q \mapsto q$;
    $k \leftarrow d(q, p_h(M, q))$;
    $N \leftarrow M_W \setminus D_{k-1}(M_W, q)$;
    $\sigma(h) \leftarrow \sigma_{N, p_h(M_W, e), f}(h)$;
    $U(M_W, q) \leftarrow U(M_W, q) \setminus \{h\}$;
    $W \leftarrow W \setminus h$;
    if $U(M_W, q) \neq \emptyset$ **then**
      Choose $h \in U(M_W, q)$;
      Push h to W ;
    else
      $Q \mapsto q$, $Q \mapsto\mapsto o$;
      $W \mapsto h$, $W \leftarrow W \setminus h$;
      $M_W \leftarrow M_W^{\leq U(M_W, q))}$;
      $W \mapsto h$;
      $\sigma(h) \leftarrow \sigma_{G, M_W, p_h(M_W, o), f}(h)$;
      $Q \leftarrow Q \setminus q$;
    end
  else
    $Q \mapsto q$;
    $U \leftarrow U(M_W, q)$;
    Choose $h \in U$;
    Push h to W;
    Push $p_h(M_W, q)$ to Q;
  end
end
return σ

To define the relative weight in general, we have to consider all spanning trees of $M_{c,f}$ generated by deleting edges, which cut cycles for the first time, seen from e. In Example 5 we assert the definition for a special meta graph. The precise definition can be found below, using the following notions:

For any tree $M = (V_M, E_M)$ and any vertices $h, e \in V_M$ we denote with $p_h(M, e)$ the predecessor of h with respect to root e. If $M = (V_M, E_M)$ is an arbitrary simple, undirected graph and $q \in V_M$ then we consider the set $D_k(M, q) := \{w \in V_M \mid d(q, w) = k\}$ of all vertices possessing shortest path distance k with respect to q in M. Furthermore, we consider

$$U_k(M, q) := \Big\{ w \in D_k(M, q) \mid \exists\, x \in D_k(M, q) \setminus \{w\} \text{ such that } P(w, x) \neq \emptyset \text{ with respect to } M \setminus \Big(\cup_{l=0}^{k-1} D_l \Big) \Big\},$$

$K(M,q) := \min\{k \in \mathbb{N} \mid U_k(M,q) \neq \emptyset\}$ and $U(M,q) := U_{K(M,q)}(M,q)$. In other words: $U(M,q)$ denotes the set of vertices w which cut cycles for the fist time, seen from starting point q.

We set

$$\widetilde{M}_h := M \setminus \{[p_h(M,q),k] \in E_M \mid k \in U(M,q) \setminus \{h\}\} \tag{7}$$

and denote with M_h the connected component of $\widetilde{M}_h$ containing h. Recursively for $n \geq 1$ and an ordered set $F = \{h_0, \ldots, h_n\}$ with $h_n \in U(M_{h_0,\ldots,h_{n-1}}, q)$ we define $M_{h_0,\ldots,h_n} := (M_{h_0,\ldots,h_{n-1}})_{h_n}$. For $h \in U(M,q)$ we consider

$$\widetilde{M}^{\leq U(M,q)} := M \setminus \{[h,k] \in E_M \mid k \in U(M,q)\,, d(q,k) \geq d(q,h)\}$$

and set $M^{\leq U(M,q)}$ to be the connected component of $\widetilde{M}^{\leq U(M,q)}$ containing q, which is therefore a tree.

Definition 11 (relative weight in general) Let (G,ω) be a graph, $c \in \mathrm{O_{el}}(G)$, $e, f \in \mathcal{E}(c)$ and M_c be the meta graph of G with respect to c. Let $\sigma_{G,M,e}$ be the output of Algorithm 4 with input $((G,\omega), M_c, e, f)$, $M = M_{c,e}$. Then, we define

$$\sigma_{G,c,e,f} : E \longrightarrow \mathbb{R}\,, \quad \sigma_{G,c,e,f}(h) := \begin{cases} \sigma_{G,M,e,f}(h) & \text{, if } h \in \mathcal{N}_{\twoheadrightarrow}(e) \text{ w.r.t. } G \setminus f \\ \omega(h) & \text{, else} \end{cases}$$

as the *relative weight* of G with respect to c, e, f.

Example 5 Let (G,ω) be a graph, $c \in \mathrm{O_{el}}(G)$, $e_0, e_1 \in \mathcal{E}(c)$ and assume that M_{c,e_1} coincides with M from Example 7. We follow Algorithm 4 to compute $\sigma_{G,M,e_0,e_1} : E \longrightarrow \mathbb{R}$. Observe that $U(M,e_0) = \{f_0, f_1\}$ and $U(M_{f_1}, e_0) = \{h_0, h_1\}$. The graph M_{f_1,h_0} is sketched in the next picture and turns out to be a tree. Now we delete all vertices which are closer to e_0 as $p_{h_1}(M,e_0)$ and obtain the graph N. Next we compute the relative weight $\sigma(h_1) := \sigma_{N,p_{h_1}(M,p_{h_1}),e_1}(h_1)$ of h_1 with respect to N, p_{h_1}. Analogously, we compute $\sigma(h_0) := \sigma_{N,p_{h_0}(M,p_{h_0}),e_1}(h_0)$ and consider the graph $M^{\leq} = M_{f_1}^{\leq U(M_{f_1},e_0)}$, which is sketched in the last picture. Now $M^{\leq}$ is a tree and the predecessor of f_1 is e_0. Thus, we can compute

$$\sigma_{M_{f_1}^{\leq U(M_{f_1},e_0)},e_0,e_1}(f_1) \quad \text{and analogously} \quad \sigma_{M_{f_0}^{\leq U(M_{f_0},e_0)},e_0,e_1}(f_0)\,,$$

which finishes the computation of $\sigma_{G,M,e_0,e_1} : E \longrightarrow \mathbb{R}$ by replacing $\omega(f_0), \omega(f_1)$ with these weights, respectively (Fig. 7).

Proposition 5 *Let $G = (V,E)$ be a graph $c \in \mathrm{O_{el}}(G)$, $e, f \in \mathcal{E}(c)$ and let the meta graph M_c of G with respect to c be given. Denote with*

$$m(c,f) := \dim_{\mathbb{Z}_2}\left(\Lambda^0(M_{c,f})\right) = |E_{M_{c,f}}| - |V_{M_{c,f}}| + 1$$

the $\mathbb{Z}_2$-dimension of the cycle space $\Lambda^0(M_{c,f})$ of $M_{c,f}$. Then

i) *The computation of $\sigma_{G,c,e,f}$ can be realized in $\mathcal{O}\left(2^{m(c,f)}|V||E|^2\log(|V|)\right)$.*
ii) $m(c,f) \leq \dim_{\mathbb{Z}_2}\Lambda^0(G) = |E| - |V| + \#G$

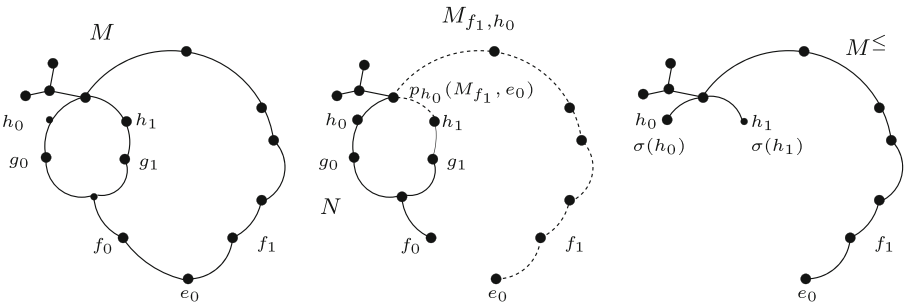

Fig. 7 Computation of $\sigma_{G,M,e_0}(f_1)$

Proof Assume that $M_{c,f}$ is a tree. As already mentioned, due to [10] and [11] the feedback length $\Omega(G_f, \sigma)$ can be determined in $\mathcal{O}(|V||E|\log(|V|))$, and has to be computed at most $|V_{M_{c,f}}| \leq |E|$ times. If $M_{c,f}$ is not a tree then we observe that the most expansive computation step in Algorithm 4 is again the computation of the relative weight with respect to a certain subtree of $M_{c,f}$ (lines 8 and 19 in Algorithm 4). This computation step has to be computed for every pair $f, h \in U(M, q)$ twice, for some M, q. In worst case the combination of the pairs is independent, i.e., every other pair still appears once M_f and M_h are considered. In this case the set of remaining cycles running through the remaining pairs (h', f') do not contain the edges h and f and can therefore not be generated by the cycles running through (h, f) with respect to $\mathbb{Z}_2$-coefficients. Consequently, there are at most $2^{m(c,f)}$ iterations. Together with the argumentation above this yields $i)$.

To show $ii)$ we write $d = \{h_0, \ldots, h_k\} \subseteq V_{M_{c,f}}$ as a list of connected meta vertices. Now we choose connected paths $p_i \subseteq G$, $0 = 1 \ldots, k+1$ connecting h_i^- with $h_{i+1 \mod k+1}^+$. Then $c = \{\{h_0\} \cup p_0 \cup \ldots, \{h_k\} \cup p_k \in \Lambda^0(G)$ is a connected cycle in G. If $d_1, \ldots, d_m \in \Lambda^0(M_{c,f})$ is a set of $\mathbb{Z}_2$-linear independent meta cycles then regardless of choices for the paths p_i representing an meta edge the corresponding cycles $c_1, \ldots, c_m \in \Lambda^0(G)$ are $\mathbb{Z}_2$-linear independent in $\Lambda^0(G)$. Thus, $m(c, f)$ is bounded by the $\mathbb{Z}_2$-dimension of $\Lambda^0(G)$, proving $ii)$. □

Remark 12 Note that a graph $G = (V, E)$ with $c \in \mathrm{O}_{\mathrm{el}}(G)$ such that M_c coincides with M in Fig. 11 can be easily constructed by choosing a starting cycle c and additional cycles c_1, c_2 intersecting with c in f_0, f_1, respectively. Then we continue this process by follwing M for the ramining cycles. Hence, the set of graphs G with cycle c and meta graphs M_c such that $\dim_{\mathbb{Z}_2 \Lambda^0}(M)$ is small, is actually a huge set.

Indeed the relative weights satisfy a Bellman condition, which can be formulated as follows. We recall that $\mathcal{G}_o(G) \subseteq G$ denotes the subgraph induced by all cycles of G and state:

Theorem 4 *Let (G,ω) be a weighted graph $c \in \mathrm{O_{el}}(G)$, $e, f \in \mathcal{E}(c)$ and $H_{e,f} = \mathcal{G}_o\left(G_{\mathrm{el}}(e) \setminus f\right)$, $H_{f,e} = \mathcal{G}_o\left(G_{\mathrm{el}}(f) \setminus e\right)$ then*

$$\begin{aligned} &\left(\omega(e) - \Omega(H_{e,f}, \sigma_e)\right) - \left(\omega(f) - \Omega(H_{f,e}, \sigma_f)\right) \\ &= \left(\omega(e) + \Omega(G \setminus e, \omega)\right) - \left(\omega(f) + \Omega(G \setminus f, \omega)\right), \end{aligned} \tag{8}$$

where we shorten $\sigma_e = \sigma_{G,c,e,f}, \sigma_f = \sigma_{G,c,f,e}$ and slightly abuse notation by still denoting $\sigma_e, \sigma_f, \omega$ for the restriction of the arc weights to the corresponding subgraphs, respectively.

Remark 13 Note that (8) is a quite comfortable way of formulating the Bellman principle, i.e., though we do not know the values of $\Omega(G \setminus e, \omega)$ and $\Omega(G \setminus f, \omega)$ we know that if e maximizes

$$\omega(e) - \Omega(H_{e,h}, \sigma_{G,c,e,h}) - \left(\omega(h) - \Omega(H_{h,e}, \sigma_{G,c,h,e})\right),$$

for all on $\mathcal{E}(c)$. Then $\omega(e) + \Omega(G \setminus e, \omega) = \Omega(G, \omega)$. Thus, $\{e\}$ can be extended to a global optimal solution. Maybe this relative formulation can be applied also to other problems for which one wants to use a dynamic programming technique. The described observation is also used in the proof of Theorem 4.

In addition to the observation above the following statement is needed to prove Theorem 4.

Lemma 3 *Let $G = (V, E)$ be a graph, $c \in \mathrm{O_{el}}(G)$ and $e, f \in \mathcal{E}(c)$ such that there is $c' \in \mathrm{O_{el}}(e) \setminus \mathrm{O_{el}}(f)$ and $p \in \mathcal{E}(c')$. Then*

i) $\sigma_{G,c,e,f}(h) = \sigma_{G\setminus f,c',e,p}(h)$ *for all* $h \in \mathcal{E}\left(H'_{e,p}\right)$.

ii) $\sigma_{G,c,e,f}(p) = \omega(p) - \Omega\left(H'_{p,e}, \sigma_{G\setminus f,c',p,e}\right)$, *where $H'_{p,e}$ is understood with respect to G'.*

Proof To verify *i)* and *ii)* one has to follow directly Definitions 9, 10 and 11, which is left to the reader. □

Proof of Theorem 4 If $\mathrm{O_{el}}(e) = \mathrm{O_{el}}(f)$ with respect to G then $\sigma_{G,c,e,f} = \sigma_{G,c,f,e}$ and $\Omega(G \setminus e, \omega)) = \Omega(G \setminus f, \omega))$ and therefore the claim follows. Now we argue by induction on $|\mathrm{O_{el}}(G)|$. If $|\mathrm{O_{el}}(G)| = 1$ then there is only one totally isolated cycle and therefore $\mathrm{O_{el}}(e) = \mathrm{O_{el}}(f) = \{c\}$. Thus, we are in a special case of the situation above and obtain the claim. Now assume that $|\mathrm{O_{el}}(G)| > 1$ and $\mathrm{O_{el}}(f) \subsetneq \mathrm{O_{el}}(e)$. We consider $G' := G \setminus f$ and observe that $|\mathrm{O_{el}}(G')| < |\mathrm{O_{el}}(G)|$. We choose $c' \in \mathrm{O_{el}}(G')$ with $e \in \mathcal{E}(c')$ and choose $p \in \mathcal{E}(c')$ such that

$$\left(\omega(e) - \Omega(H'_{e,p}, \sigma_{G',c',e,p})\right) - \left(\omega(p) - \Omega(H'_{p,e}, \sigma_{G',c',p,e})\right) \tag{9}$$

is maximized on c', where $H'_{e,p}$,$H'_{p,e}$ are understood with respect to G'. Thus, following Remark 8 there holds

$$\omega(p) + \Omega(G' \setminus p, \omega) = \Omega(G', \omega)\,. \tag{10}$$

We set $\sigma'_e := \sigma_{G',c',e,p}$, $\sigma'_p := \sigma_{G',c',p,e}$ then by induction and (10) we compute

$$\begin{aligned}\Big(\omega(e) - \Omega(H'_{e,p}, \sigma'_e)\Big) - \Big(\omega(p) - \Omega(H'_{p,e}, \sigma'_p)\Big) &= \big(\omega(e) + \Omega(G' \setminus e, \omega)\big) \\ &\quad - \big(\omega(p) + \Omega(G' \setminus p, \omega)\big) \\ &= \omega(e) + \Omega(G \setminus \{e, f\}, \omega) \\ &\quad - \Omega(G \setminus f, \omega)\,. \end{aligned} \tag{11}$$

On the other side we consider $G'' = (V'', E'') := H_{e,f}$ with the arc weight

$$\gamma : E'' \longrightarrow \mathbb{R}^+\,, \quad \gamma(h) := \sigma_{G,c,e,f}(h)\,.$$

Now observe that $H''_{p,e} = \emptyset$ with respect to G'' and by Lemma 3 i) we have $\sigma''_e := \sigma_{G'',c',e,p}(h) = \sigma_{G',c,e,p}(h) = \gamma(h)$ for all $h \in \mathcal{E}(H''_{e,p})$, where $H''_{e,p}$ is understood with respect to G''. Moreover, $\gamma(e) = \omega(e)$ and therefore

$$\begin{aligned}\Big(\gamma(e) - \Omega(H''_{e,p}, \sigma''_e)\Big) - \Big(\gamma(p) - \Omega(H''_{p,e}, \sigma''_p)\Big) &= \gamma(e) - \Omega(H''_{e,p}, \gamma) - \gamma(p) \\ &= \gamma(e) - \Omega(G'' \setminus p, \gamma) - \gamma(p) \\ &= \omega(e) - \Omega(H'_{e,p}, \sigma_{G',c,e}) \\ &\quad - \sigma_{G,c,e}(p) \end{aligned} \tag{12}$$

Due to Lemma 3 ii) we have that $\sigma_{G,c,e,f}(p) = \omega(p) - \Omega(H'_{p,e}, \sigma'_p)$, $\sigma'_p = \sigma_{G',c',p,e}$. Inserting this fact in (12) gives

$$\begin{aligned}\Big(\gamma(e) - \Omega(H''_{e,p}, \sigma''_e)\Big) - \Big(\gamma(p) - \Omega(H''_{p,e}, \sigma''_p)\Big) &= \Big(\omega(e) - \Omega(H'_{e,p}, \sigma'_e)\Big) \\ &\quad - \Big(\omega(p) - \Omega(H'_{e,p}, \sigma'_p)\Big) \end{aligned} \tag{13}$$

On the other, by (10) we have that (13) is maximized on c'. Thus, again by induction

$$\begin{aligned}\Big(\gamma(e) - \Omega(H''_{e,p}, \sigma''_e)\Big) - \Big(\gamma(p) - \Omega(H''_{p,e}, \sigma''_p)\Big) &= \big(\gamma(e) + \Omega(G'' \setminus e, \gamma)\big) \\ &\quad - \big(\gamma(p) + \Omega(G'' \setminus p, \gamma)\big) \\ &= \omega(e) - \Omega(G'', \gamma) \\ &= \omega(e) - \Omega(H_{e,f}, \sigma_{G,c,e,f}) \end{aligned} \tag{14}$$

Thus, by combining (13) with (11) and again (13) with (14) we obtain

$$\omega(e) - \Omega(H_{e,f}, \sigma_{G',c,e}) = \omega(e) + \Omega(G \setminus \{e, f\}, \omega) - \Omega(G \setminus f, \omega) \tag{15}$$

If $\mathrm{O_{el}}(f) \subseteq \mathrm{O_{el}}(e)$ then $\Omega(H_{f,e}, \sigma_{G,c,f,e}) = 0$ and $\Omega(G \setminus \{e, f\}, \omega) = \Omega(G \setminus e, \omega)$. Thus, by (15) this yields the claim. If $\mathrm{O_{el}}(f) \subsetneq \mathrm{O_{el}}(e)$ then the analogous of (15) with respect to f yields

$$\begin{aligned}\big(\omega(e) - \Omega(H_{e,f}, \sigma_e)\big) - \big(\omega(f) - \Omega(H_{f,e}, \sigma_f)\big) &= \omega(e) + \Omega(G \setminus e, \omega) \\ &\quad + \Omega(G \setminus \{e, f\}, \omega) - \Omega(G \setminus \{f, e\}, \omega) - \omega(f) - \Omega(G \setminus f, \omega)\,. \end{aligned}$$

Algorithm 5 CUT

Input: $G = (V, E, \omega)$
Output: $\varepsilon \in \mathcal{S}(G, \omega)$
$\varepsilon = \emptyset$;
while $\exists f \in E$ *with* $P(f^-, f^+) \neq \emptyset$ **do**
  $F \leftarrow \mathcal{E}(G_{\mathrm{el}}(f))$;
  while $\exists e \in F$ *with* $P(e^-, e^+) \neq \emptyset$ **do**
    Choose $p \in \mathrm{P}_{\mathrm{el}}(e^-, e^+)$;
    $c \leftarrow \{e\} \cup p$;
    $k \leftarrow \mathrm{argmax}^*_{h \in \mathcal{E}(c)} \left(\left(\omega(e) - \Omega(H_{e,h}, \sigma_{G,c,e,h})\right) - \left(\omega(h) - \Omega(H_{h,e}, \sigma_{G,c,h,e})\right)\right)$;
    $\varepsilon \leftarrow \varepsilon \cup \{k\}$;
    $(G, \omega) \leftarrow (G \setminus k, \omega)$;
    $F \leftarrow F \setminus \{k\}$;
  end
end
return ε

Since $\Omega(G \setminus \{e, f\}, \omega) - \Omega(G \setminus \{f, e\}, \omega) = 0$ this finishes the proof. □

We consider the Algorithms 5 and 6, denote with output(A) the set of all possible outputs an algorithm A can produce and conclude:

Corollary 1 *Let (G, ω) be a graph then the algorithm CUT is exact and complete with respect to the FASP, i.e,*

$$\mathrm{output(CUT)} = \mathcal{S}(G, \omega)\,,$$

while the algorithm CUT & RESOLVE is exact, i.e.,

$$\mathrm{output(CUT\ \&\ RESOLVE)} \subseteq \mathcal{S}(G, \omega)\,.$$

Moreover, there is $m \in \mathbb{N}$ such that CUT and CUT & RESOLVE possess run times $\mathcal{O}\left(2^m |E|^4 \log(|V|)\right)$, where the parameter $m \leq |E| - |V| + 1$ can be determined in $\mathcal{O}\left(|E|^3\right)$.

Indeed the corollary proves Theorem B and by Remark 12 the set of graphs with small $m \in N, m << |V|$ is a huge set.

Proof If $\varepsilon \in \mathcal{S}(G, \omega)$ and $e \in \varepsilon$ then $\varepsilon \setminus \{e\}$ solves the minimal FASP on $G(E \setminus \{e\})$. Thus, the exactness and completeness statements follow directly from Theorems 4,3. For $c \in \mathrm{O}_{\mathrm{el}}(G)$ we set $m(c) := \dim_{\mathbb{Z}_2} \Lambda^0(M_c)$. Then $m(c) \geq m(c, f)$ for all $f \in \mathcal{E}(c)$ with $m(c, e)$ and Proposition 5 implies that $m(c) \leq |E| - |V| + 1$ holds. Let $c' \in \mathcal{C}(c)$ with $c' \cap c \neq 0$, be any cycle in the component of all arc connected cycles containing c, introduced in Definition 9. Then $m(c') \leq m(c)$ on $G \setminus e$ for every $e \in \mathcal{E}(c)$. Thus, as long as at least one arc $e \in \mathcal{E}(c)$ was deleted the maximal number of appearing $\mathbb{Z}_2$-linear independent meta cycles appearing for the computation of $\sigma_{G \setminus e, c', h, k}$, $h, k \in \mathcal{E}(c')$ are bounded by $m(c)$. Thus, by setting $G_0 = G$, choosing a

Algorithm 6 CUT & RESOLVE

Input: $G = (V, E, \omega)$
Output: $\varepsilon \in \mathcal{S}(G, \omega)$
$\varepsilon = \emptyset$;
while $\exists f \in E$ *with* $P(f^-, f^+) \neq \emptyset$ **do**
  $F \leftarrow \mathcal{E}(G_{\mathrm{el}}(f))$;
  while $\exists e \in F$ *with* $P(e^-, e^+) \neq \emptyset$ **do**
    Choose $p \in \mathrm{P}_{\mathrm{el}}(e^-, e^+)$;
    $c \leftarrow \{e\} \cup p$;
    $k \leftarrow \mathrm{argmax}^*_{h \in \mathcal{E}(c)}(\omega(e) - \Omega(H_{e,h}, \sigma_{G,c,e,h}) - \omega(h) - \Omega(H_{h,e}, \sigma_{G,c,h,e}))$;
    $\varepsilon \leftarrow \varepsilon \cup \{k\}$;
    $(G, \omega) \leftarrow (S(G \setminus k), \tau(\omega))$;
    $F \leftarrow \mathcal{E}$ $((G_{\mathrm{el}}(e))$ w.r.t. G;
  end
end
return ε

cycle $c_0 \in \mathrm{O}_{\mathrm{el}}(G_0)$, determing $M_{c_0} = (V_{M_{c_0}}, E_{M_{c_0}})$ and considering $c_k \in \mathrm{O}_{\mathrm{el}}(G_k)$, $G_k = G_{k-1} \setminus V_{M_{c_{k-1}}}$, $k \geq 1$ we obtain cycles $c_0, \ldots, c_n$, $n \leq |E|$ with $\mathcal{C}(c_i) \cap \mathcal{C}(c_j) = \emptyset$ and $\cup_{i=0}^{n} \mathcal{C}(c_i) = \mathrm{O}_{\mathrm{el}}(G)$. Since $\mathcal{E}(\mathcal{C}(c_i)) \cap \mathcal{E}\left(\mathcal{C}(c_j)\right) = \emptyset$ the parameter $m := \max_{i=0,\ldots,n} m(c_i)$ can be determined in $\mathcal{O}\left(|E|^3\right)$ due to Lemma 2.

Since the algorithm *CUT* computes $\sigma_{G,c,e,h}, \sigma_{G,c,h,e}$ for fixed $e \in \mathcal{E}(c)$ and all $h \in \mathcal{E}(c)$ and $|\mathcal{E}(c)| \leq |V|$, cuts the right arc and repeats the computation at most $|E|$ times by observing that $\mathcal{O}(|V|^2) = \mathcal{O}(|E|)$ the run time of the algorithm *CUT* can be estimated as claimed. Recall, that due to Theorem 3 the resolved graph can be computed in $\mathcal{O}\left(|V||E|^3\right)$. Therefore, the analogous argumentation yields the claimed run time for the algorithm *CUT & RESOLVE*. □

Due to the fact that the FASP is NP complete, as expected our approach depends exponentially on some parameter, which in our approach is the number m of liner independent meta cycles. In cases where m is large we have to use another method to solve the FASP or use a heuristic.

8 Valid Greedy Solutions

As for instance shown in [19] a greedy solution for the FASP needs not to be optimal. We give a criterium on solutions which guarantees optimility. Moreover, we can estimate the failure of every sub optimal solution. Finally, we suggest a heuristic given by a hybrid technique of the already presented approaches.

For given graph (G, ω) we consider the functions

$$\theta_G, \varphi_G : E \longrightarrow \mathbb{N}, \quad \theta(e) := |\mathrm{O}_{\mathrm{el}}(e)|, \quad \varphi_G(e) := |\mathcal{E}(G_{\mathrm{el}}(e))| .$$

Recall, that due to [2] determing $\theta_G(e)$ is a NP-hard problem and the results of [34] and [20] solving the problem in $\mathcal{O}(|\theta_G(e)|(|V| + |E|))$, where $\theta_G(e)$ can depend exponentially on G. However, [31] could establish efficent and close estimations of the number of $s - t$ paths. Since $|\mathrm{O}_{\mathrm{el}}(e)| = |P_{\mathrm{el}}(e^-, e^+)|$ the result enables us to

determine $\theta_G(e)$ efficently, with small failure. In contrast, φ_G can be determined in $\mathcal{O}(|E|^2)$ due to Theorem 2.

Definition 12 Let (G, ω) be a given weighted graph with $G = G_o$. We introduce the efficent weights

$$\xi_{G,\omega}, \eta_{G,\omega} : E \longrightarrow \mathbb{Q}^+, \quad \xi_{G,\omega}(e) := \frac{\theta_G(e)}{\omega(e)}, \quad \eta_{G,\omega}(e) := \frac{\varphi_G(e)}{\omega(e)}$$

and set $\omega_{\max}(G, \omega) := \max_{e\in E} \omega(e)$, $\theta_{\max}(G) = \max_{e\in E} \theta_G(e)$, $\varphi_{\max}(G) := \max_{e\in E} \varphi_G(e)$, $\xi_{\max}(G, \omega) := \max_{e\in E} \xi(e)$, $\eta_{\max}(G, \omega) := \max_{e\in E} \eta(e)$, and $\mu(G, \omega) = \left\lceil \frac{|\mathrm{O_{el}}(G)|}{\xi_{\max}(G,\omega)} \right\rceil$, $\upsilon(G, \omega) = \left\lceil \frac{|E|}{\eta_{\max}(G,\omega)} \right\rceil$, where $\lceil\cdot\rceil$ denotes the Gauss-bracket.

Theorem 5 *Let (G, ω) be a given graph. Then*

$$\Omega(G, \omega) \geq \max\{\mu(G, \omega), \upsilon(G, \omega)\} . \tag{16}$$

Moreover, there are infinitely many weighted graphs (G, ω) with $\mu(G, \omega) = \Omega(G, \omega)$ or $\upsilon(G, \omega) = \Omega(G, \omega)$.

Proof Let $\varepsilon = \{e_1, \ldots, e_n\} \in \mathcal{S}(G, \omega)$ be an arbitrarily ordered solution of the weighted FASP. We set $G_0 = G$ and $G_i = (V_i, E_i) := G(E \setminus \{e_1, \ldots, e_{i-1}\})$, for $i \geq 1$ and denote with ω_{G_i} the corresponding restriction of ω to E_i. Now due to the fact that $\xi_{G_i,\omega_i}(e) \leq \xi_{G,\omega}(e)$ for all $e \in E_i$, $1 \leq i \leq n$ we obtain

$$\Omega(G, \omega) = \sum_{i=1}^{n} \omega_{G_i}(e_i) = \sum_{i=1}^{n} \frac{\theta_{G_i}(e_i)}{\xi_{G_i}(e_i)} \geq \frac{1}{\xi_{\max}(G, \omega)} \sum_{i=1}^{n} \theta_{G_i}(e_i) = \frac{|\mathrm{O_{el}}(G)|}{\xi_{\max}(G, \omega)} .$$

Since $\Omega(G, \omega) \in \mathbb{N}$ this proves (16). Now let (G, ω) be a graph with $\omega = 1$ and $\mathrm{O_{el}}(G) = \{c_0, \ldots, c_n\}$, which are arranged path like, i.e., $|\mathcal{E}(c_i \cap c_{i+1})| = 1$ and $|\mathcal{E}(c_i \cap c_j)| = 0$ if $j \neq i$. Then one verifies easily that $\mu(G, \omega) = \Omega(G, \omega)$. By replacing θ_G with φ_G and $\mu(G, \omega)$ with $\upsilon(G, \omega)$ the exact same argumentaion yields the remaining claim. □

Remark 14 Note, that of course there are many more graphs with $\mu(G, \omega) = \Omega(G, \omega)$ or $\upsilon(G, \omega) = \Omega(G, \omega)$ then those used in the proof above. Nevertheless, it is hard to give a good condition on a graph such that $\mu(G, \omega) = \Omega(G, \omega)$ or $\upsilon(G, \omega) = \Omega(G, \omega)$ holds. For instance in [19] an example of a planar graph is given, where this is not the case. Certainly, the lower bounds can be used to improve the performance of a variety of algorithms solving the FASP or to control the quality of a heuristic as the Algorithms 7 and 8.

We consider the heuristics *GREEDY-CUT* and *GREEDY-CUT & RESOLVE* and discuss their properties.

Algorithm 7 GREEDY-CUT

Input: $G = (V, E, \omega)$
Output: Feedback set $\varepsilon \subseteq E$
$\varepsilon = \emptyset$;
while $\xi_{\max}(G) \neq 0$ **do**
  $\varepsilon \leftarrow \varepsilon \cup \operatorname{argmax}^*_{f \in E} \xi(f)$;
  $(G, \omega) \leftarrow (G \setminus \varepsilon, \omega)$;
end
return ε

Algorithm 8 GREEDY-CUT & RESOLVE

Input: $G = (V, E, \omega)$
Output: Feedback set $\varepsilon \subseteq E$
$\varepsilon = \emptyset$;
while $\xi_{\max}(G) \neq 0$ **do**
  $\varepsilon \leftarrow \varepsilon \cup \operatorname{argmax}^*_{f \in E} \xi(f)$;
  $(G, \omega) \leftarrow (S(G \setminus \varepsilon), \tau(\omega))$;
end
return ε

Proposition 6 *Let (G, ω) and ε be a solution of GREEDY-CUT or GREEDY-CUT & RESOLVE with respect to the effective weigth ξ. Then*

$$\frac{\Omega(G, \omega)}{\Omega_{G,\omega}(\varepsilon)} \geq \frac{\omega_{min}(G)}{\omega_{\max}(G) \cdot \theta_{\max}(G)} \geq \frac{\omega_{min}(G)}{\omega_{\max}(G) \cdot |\mathrm{O_{el}}(G)|}\,. \tag{17}$$

If in particular $\omega \equiv 1$ then $|\varepsilon| \leq |E|/2$.

Proof Certainly it suffices to prove the first estimate in (17). We show the claim for a solution $\varepsilon = \{e_1, \ldots, e_n\}$ of *GREEDY-CUT*. Assume that ε is ordered with respect to appearing arcs, set $G_1 = G$ and $G_i = (V_i, E_i) := G\,(E \setminus \{e_1, \ldots, e_{i-1}\})$, for $i \geq 2$ and denote with ω_{G_i}, θ_{G_i} and ξ_{G_i} the corresponding restrictions of ω, θ_G, ξ_G to E_i. Then we compute

$$\Omega_{G,\omega}(\varepsilon) = \sum_{i=1}^{n} \omega_i(e_i) = \sum_{i=1}^{n} \frac{\theta_{G_i}(e_i)}{\xi_{G_i}(e_i)} \leq \frac{1}{\xi_{G_n}(e_n)} |\mathrm{O_{el}}(G)|\,.$$

Since $\xi_n(e_n) = \xi_{\min}(\varepsilon)$ we use Theorem 5 to compute

$$\frac{\Omega(G, \omega)}{\Omega_{G,\omega}(\varepsilon)} \geq \frac{\xi_{G_n}(e_n)|\mathrm{O_{el}}(G)|}{\xi_{\max}(G)|\mathrm{O_{el}}(G)|} \geq \frac{\omega_{min}(G) \cdot \theta_{G_n}(e_n)}{\omega_{\max}(G) \cdot \theta_{\max}(G)} \geq \frac{\omega_{min}(G)}{\omega_{\max}(G) \cdot \theta_{\max}(G)}$$

and the claim follows. A proof of the statement for *GREEDY-CUT & RESOLVE* can be given by an easy adaption of the argument above and is left to the reader. Now let $\omega \equiv 1$ then we argue by induction on $|\varepsilon|$ to show that $|\varepsilon| \leq |E|/2$ for both algorithms. If $|\varepsilon| = 1$ then due to the fact that G possesses no loops the claim follows. Now let $|\varepsilon| > 1$ we order $\varepsilon = \{e_1, \ldots, e_n\}$ with respect to appearance and consider $\varepsilon_1 := \{e_1\}$, $\varepsilon_2 = \varepsilon \setminus \{e_1\}$ and $G_1 := (V_1, E_1) = G(\mathrm{O_{el}}(e_1))$, $G_2 := (V_2, E_2) = G(\mathrm{O_{el}}(\varepsilon_2))$.

By induction we have $|\varepsilon_2| \leq |E_2|/2$. Consider $G/(E_1 \cap E_2)$, delete all appearing loops and denote the resulting graph with G_1^*. If $\mathrm{O_{el}}(G_1^*) = \{c\}$ then all cycles of $\mathrm{O_{el}}(G)$ are totally isolated and the claim follows by triviality. If $|\mathrm{O_{el}}(G_1^*)| > 1$ then $|\varepsilon_1| \leq |E_1^*|/2$. Since $E_1^* \cap E_2 = \emptyset$ this implies that

$$|\varepsilon| = |\varepsilon_1| + |\varepsilon_2| \leq |E_1^*|/2 + |E_2|/2 \leq |E|/2$$

as claimed. □

Be replacing θ_G, with φ_G and $\xi_{G,\omega}$ with $\eta_{G,\omega}$ in *GREEDY-CUT* or *GREEDY-CUT & RESOLVE* the analouge argumentaion yields.

Proposition 7 *Let $(G, \omega)$with $G = G_o$ and ε be a solution of GREEDY-CUT or GREEDY-CUT & RESOLVE with respect to the effective weigth η. Then*

$$\frac{\Omega(G,\omega)}{\Omega_{G,\omega}(\varepsilon)} \geq \frac{\omega_{min}(G)}{\omega_{\max}(G)\cdot\varphi_{\max}(G)} \geq \frac{\omega_{min}(G)}{\omega_{\max}(G)\cdot|E|}\,.$$

If in particular $\omega \equiv 1$ then $|\varepsilon| \leq |E|/2$.

Example 6 Consider the directed clique D_3 from Fig. 5 with constant weight $w \equiv 1$. Then D_3 coincides with its resolved graph and regardless of possible choices every candidate ε the algorithm *GREEDY-CUT* or *GREEDY-CUT & RESOLVE* proposes, satisfies $|\varepsilon| = 3$. Since $\mu(D_3) = 3$ every candidate is optimal.

Summarizing our results so far the heuristics *GREEDY-CUT* or *GREEDY-CUT & RESOLVE* solve the FASP with controlled variance in $\mathcal{O}(|E|^4)$, due to Theorem 2, in case of effective weight η and in $\mathcal{O}(f_\theta|E|^2)$ in case of effective weight ξ, where f_θ shall control the computation steps of $\theta_G(e)$, $\forall e \in E$. Even if we approximate $\theta(e)$ by the method of [31] the resulting algorithm remains an efficient heuristic. However, possibly there is a more accurate method available, given by a hybrid algorithm of the methods introduced in this article. We expect that an implementation of this strategy yields a fast and precise general FASP-SOLVER, which due to Section 2 is therfore also a FVSP-SOLVER.

Strategy 1 *For given weighted graph (G, ω)*

1. *Compute the resolved graph (S, τ).*
2. *Choose a cycle $c \in \mathrm{O_{el}}(S)$ and compute the meta graph M_c.*

3a. *If the number of meta cycles $m(c)$ is large determine a "good" feedback vertex set v_{M_c} of M_c, with respect to the vertex weight*

$$\omega_{M_c} : V_{M_c} \longrightarrow \mathbb{R}^+\,, \quad \omega_{V_c}(v) = \frac{1}{\omega(v)}\,,$$

using one of the known or presented methods.

3b. *Alternatively, compute a maximal spanning tree* $T_{M_c} = (V_{m_c}, E_{T_c})$ *with respect to the arc weight weight*

$$\omega_{M_c} : E_{M_c} \longrightarrow \mathbb{R}^+, \quad \omega_{M_c}(h) = \frac{1}{\omega(e) + \omega(f)},$$

where $h = [e, f]$, $e, f, \in E$ *and set* $\varepsilon_{M_c} = E_{M_c} \setminus E_{T_c}$.

4. *Set* $G^* = G/\nu_{M_c}$ *and use CUT or CUT & RESOLVE to solve the FASP on on the component* $\mathcal{C}(c^*)$ *of arc connected cycles containing* $c^* = c/\nu_{M_c}$.
5. *Choose a new cycle* c' *of the resulting graph and repeat 1.-4. until no such cycle exists.*
6. *Use the backtracking procedure of Proposition 3 to compute a feedback arc set* $\varepsilon \subseteq E$ *of* G.

The union ν_M *of the meta feedback vertex sets of the meta graphs can be interpreted as arcs, which are forbidden to cut in* G. *The resulting feedback arc set* ε *will be optimal up to this obstruction, i.e., we have*

$$M^*_{c^*} = M_c \setminus \nu_M,$$

where $M^*_{c^*}$ *denotes the meta graph of* G^* *with respect to* $c^* = c/\nu_M$. *Hence, the quality of this heuristic can be evaluated by measuring how good* G^* *approximates* G. *Thus, if* $|\nu_M| << |E|$ *and the weight of the forbidden arcs is very high, i.e.,*

$$\frac{\sum_{f \in \nu_M} \omega(f)}{|\nu_M|} >> \frac{\sum_{e \in E} \omega(e)}{|E|}$$

the arcs of ν_M *will probably not be contained in any optimal solution, yielding the correctness of Strategy 1. Additionally, the lower bounds* $\mu(G, \omega)$, $\upsilon(G, \omega)$ *from Section 8 can be used to validate correctness. Analogous controls can be thought of, if we choose the alternative 3b.*

9 Discussion

An implementation of the described algorithms is planned to be realized. Certainly, a comparison of real run times with other approaches would be of great interest. So far we compare our results with other theoretical approaches. Due to the immense amount of results during the last decades we restrict our discussion to publications, which do not restrict themselves to very tight graph classes as *tournaments* [23] or *reducible flow graphs* [29]. Finally, we suggest how the approaches of this article might be adapted to related problems.

9.1 The algorithms from I. Razgon and J. Chen et al.

Note that for given graph (G, ω, γ) with arc weight ω and vertex weight γ a *brute force* method of solving the FASP/FVSP is given by considering every subset $\varepsilon \subseteq E$ or $\nu \subseteq V$ and check whether the graphs $G \setminus \varepsilon$, $G \setminus \nu$ are acyclic, respectively. Since due to Remark 5, checking for acyclicity requires $\mathcal{O}(|E|^2)$ operations, we can generate a list of all FAS's or FVS's possessing length $l_A \leq |\mathcal{P}(E)| = 2^{|E|}$, $l_V \leq$

$|\mathcal{P}(V)| = 2^{|V|}$. Choosing the cheapest FAS or FVS yields therefore a brute force algorithm solving the FASP/FVSP in $\mathcal{O}(|2^{|E|}|E|^2|)$, $\mathcal{O}(2^{|V|}|E|^2)$, respectively.

The algorithm of [30] solves the unweighted FVSP on simple graphs in $\mathcal{O}(1.9977^{|V|}|V|^{\mathcal{O}(1)})$. Compared to the brute force algorithm this yields almost no improvement. Therefore, the question occured whether the parametrised version of the FVSP could be solved by an *fixed parameter tractable algorithm*. Every NP-complete problem can be solved by a fixed parameter tractable algorithm, i.e., by choosing p as the problem size there is an algorithm with complexity $\mathcal{O}(f(p))$, where f is an on the parameter p exponentially depending function. Thus, the term fixed parameter tractable could be misleading. The precise question is whether there exists an algorithm with run time $\mathcal{O}(f(k)|V|^{\mathcal{O}(1)})$ computing a FVS of length less than k or determing that no such set exists. Since the FVSP is NP-complete the function f will be exponentially dependent on k unless $P = NP$. Indeed, the algorithm of [9] solves the parametrised version of the FVSP in $\mathcal{O}\left(|E|^4 4^k k^3 k!\right)$. Thus, $f(k) = k^3 4^k k!$, increases even worse than exponentially in k. Since a small feedback length almost always correlates to small graphs or very special graphs, e.g. tree-like graphs, even improvements of the algorithm won't be usefull in many applications. Therefore, the article might be seen as an purely theoretical approach answering this question. Indeed, to the best of our knowledge none of the algorithms were used for an implementation of a general FVSP/FASP-SOLVER.

In contrast, the algorithms *CUT* and *CUT & RESOLVE* solve the FASP or FVSP on weighted multi-digraphs in $\mathcal{O}\left(2^m|E|^4 \log(|V|)\right)$ and $\mathcal{O}\left(2^n \Delta(G)^4|V|^4 \log(|E|)\right)$. The parameters m and n fulfill $m \leq |E| - |V| + 1$, $n \leq (\Delta(G) - 1)|V| - |E| + 1$ and can be computed in $\mathcal{O}(|E|^3)$, $\mathcal{O}(\Delta(G)^3|V|^3)$, respectively. Thus, in both cases we can efficently control the run time of the exact solutions, which enables us to a priori decide whether the given instance shall be solved exactly or by an heuristic, e.g., Strategy 1. This crucial difference to the other approaches and the fact that Strategy 1 is an heuristic on the meta level and not on the instance itsself, makes us confident that an implementation generates a fast and accurate FASP/FVSP-SOLVER yielding a deep impact on computational an applied sciences.

9.2 The polytop approach from C. Lucchesi et al.

In [26] and [16] a polytope of arc sets is assigned to a given graph. The FASP translates to solve a certain linear optimization over this polytope. In the case of *planar* or more general *weakly acyclic graphs* the polytope is integral, i.e., it possesses integral corners. Since the optimum will be obtained in at least one of the corners, one can apply the so called *ellipsoid method for submodular functions* [17] to find the right corner in polynomial time, see also [27] for further details. The approach is certainly remarkable though it contains some weaknesses.

The first problem is that though the algorithm runs in polynomial time the degree of the polynomial depends on a variety of parameters and cannot be estimated by hand a priori. Therefore, there are planar or weakly acyclic graphs, which can be solved efficently from a theoretical view point but actually a computer based implementation of the approach cannot ensure to meet a performance behavior applications require.

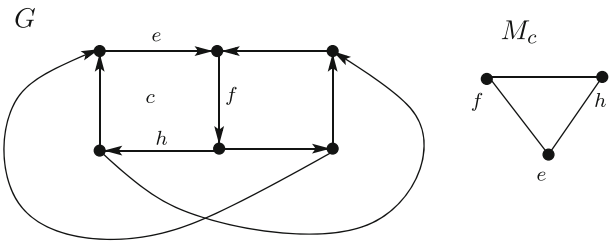

Fig. 8 A directed version of $K_{3,3}$

Secondly, the class of weakly acyclic graphs is not well classified yet. Thus, if we leave the class of planar graphs it is hard to say whether a given graph is weakly acyclic or not. For instance, consider the directed version of $K_{3,3}$ in Fig. 8, which is known to be a weakly acyclic graph. Then it is not hard to see that the meta graph M_c contains only one cycle. Thus, though G is not resolvable we can efficently solve the FASP on G by applying the algorithm *CUT*. In fact all examples of weakly acyclic graphs given in [17] turn out to be efficently solvable by *CUT*. Since the techniques of this article are not sensitive to topological obstructions as planarity we expect that there are instances of the FASP, which are neither planar nor weakly acyclic and even though can be solved efficently by *CUT* or *CUT & RESOLVE*. On the other hand, by arranging cycles along several meta cycles it is quite easy to construct a planar graph G with a number $m \in \mathcal{O}(|E|)$ of linear independent meta cycles. Thus, so far none of the approaches can state to solve the "larger" instance class efficently.

However, a deeper understanding of the meta graphs and their topology seem to be the most relevant tasks for further research, which might enable us to classify weakly acyclic graphs and yield a completely new perspective to other questions in graph theory.

9.3 Heuristics

In [32] a summarization of heuristic approaches is given and several new ones are introduced. The weak point in all these approaches is that they do not provide an non-empirical control of the variance of the heuristical solution from the optimum. Therefore, it is impossible to guarantee whether a solution is tight to the optimum. In [18] a good lower bound of the feedback length for Eulerian graphs is given and therefore it would be interesting how our bound behaves on this graph class. In general, if ε denotes a feedback set the heuristic *GREEDY-CUT* or *GREEDY-CUT & RESOLVE* proposes, then by Proposition 6 we have shown that if $\omega \equiv 1$ then

$$\max\{\mu(G), \upsilon(G)\} \leq \Omega(G, \omega) \leq \Omega_G(\varepsilon) \leq |E|/2\,, \tag{18}$$

yielding a controlled variance, as long as $\mu(G)$ can be determined or estimated from below, see [31]. We conjecture that (18) improves the known estimates given by [7]. Furthermore, other heuristics can be improved by the results of this article. For instance, the counter example for the Greedy approach introduced in [19] is resolvable and therefore *RESOLVE & CUT* closes this gap. A comparison

of Strategy 1, with the approximation of [13], where an approximation ratio in $\mathcal{O}(\log(\Omega)\log(\log(\Omega)))$ was established, might be usefull as well.

9.4 New Approach to the Subgraph Homeomorphism Problem

The NP-complete *directed subgraph homeomorphism problem* studied by [14] is to consider two given graphs $G = (V_G, E_G)$ and $P = (V_P, E_P)$ together with an injective mapping $m : V_P \longrightarrow V_G$ of vertices of P into the vertices of G. Now the problem is given by deciding whether there exists a injective mapping from arcs of P into pairwise node disjoint elementary paths of G such that an arc f with head h and tail t is mapped on an elementary path from $m(t)$ to $m(h)$. For a given graph $G^* = (V^*, E^*)$, $u, v \in V^*$, $f = (p, q) \in E^*$ with $\deg(p), \deg(q) \geq 2$ we consider a special instance of the directed subgraph homeomorphism problem by setting $G = G^* \setminus f$, $P = (V_P, E_P)$ with $V_P = \{a, b, c, d\}$, $E_P = \{(a, b), (c, d)\}$ and $m(a) = u, m(b) = p, m(c) = q, m(d) = v$. Thus, solving the subgraph homeomorphism problem with respect to these special instances is equivalent to decide whether f is an arc of $G^*_{\mathrm{el}}(u, v)$. Hence, in addition to the polynomial time solvable instance classes known from [14], e.g. stars, also problem instances as defined above are polynomial time solvable due to Theorem 2. Potentially, our observations can be generalized in regard of this problem.

Acknowledgements Open Access Funding provided by Max Planck Society. I want to thank Matthias Bernt for many fruitful discussions and his support for formalizing some algorithms. Moreover, a heartful thank you goes to Peter F. Stadler for the nice time at his bioinformatics institute in Leipzig.

Appendix A: The Essential Minor

This section is used to prove Propositions 2 and 3. To do so we recall that $\mathrm{O}_{\mathrm{el}}(e) = \{c \in \mathrm{O}_{\mathrm{el}}(G) \mid e \in \mathcal{E}(c)\}$ and state the following Lemmas.

Lemma A1 *Let G be a graph, $\omega : E \longrightarrow \mathbb{N}^+$ be an arc weight, $\varepsilon \in \mathcal{S}(G, \omega)$, and $e \in E$. Then either*

$$\varepsilon \cap [e]_{\sim_\Gamma} = \emptyset \quad or \quad |\varepsilon \cap [e]_{\sim_\Gamma}| = 1\,. \tag{19}$$

If in particular, $\varepsilon \cap [e]_{\sim_\Gamma} \neq \emptyset$ then $\varepsilon \cap [e]_{\sim_\Gamma}$ minimizes ω on $[e]_{\sim_\Gamma}$.

Proof Let $\varepsilon \in \mathcal{S}(G, \omega)$ and $e \in \varepsilon$. Since every arc $f \in E$ with with $e \sim_\Gamma f$ is connected by a branch point free path with e we have that $\mathrm{O}_{\mathrm{el}}(e) = \mathrm{O}_{\mathrm{el}}(f)$. Thus, at most one arc in $[e]_{\sim_\Gamma}$ will be cutted and this arc has to minimize ω on $[e]_{\sim_\Gamma}$. □

Lemma A2 *Let G be a positively weighted graph, $\varepsilon \in \mathcal{S}(G, \omega)$, and $e \in E$. Then either*

$$\varepsilon \cap [e]_{\sim_\Phi} = \emptyset \quad or \quad \varepsilon \cap [e]_{\sim_\Phi} = [e]_{\sim_\Phi}\,. \tag{20}$$

Proof Let $\varepsilon \in \mathcal{S}(G,\omega)$ and $e \in \varepsilon$. Assume there is $f \in F^+(e) \setminus \varepsilon$ then certainly $\varepsilon \cap F^-(e) = F^-(e)$ otherwise there would be a two-cycle that is not cutted. Now, let $e_1,\dots,e_k \in \varepsilon \setminus F(e)$, $k \in \mathbb{N}$, be such that $\mathrm{O_{el}}(f) \cap \mathrm{O_{el}}(e_i) \neq \emptyset$ and $\mathrm{O_{el}}(\{e_1,\dots,e_k\}) \supseteq \mathrm{O_{el}}(f)$. Since the cycles in $\mathrm{O_{el}}(e)$ and $\mathrm{O_{el}}(f)$ differ only in a single arc, i.e., e and f, it suffices to cut the arcs $F^-(e) \cup \{e_1,\dots,e_k\}$ to cut all cycles in $\mathrm{O_{el}}(e)$, i.e.,

$$\mathrm{O_{el}}\left(F^-(e)\right) \cup \mathrm{O_{el}}\left(\{e_1,\dots,e_k\}\right) \supseteq \mathrm{O_{el}}(e)\,.$$

Since $F^-(e) \cup \{e_1,\dots,e_k\}$ is therefore a cheaper possibility than ε cutting $\mathrm{O_{el}}(F^+(e))$, this contradicts that $\varepsilon \in \mathcal{S}(G,\omega)$ and yields the claim. □

Now we state again Proposition 2 and deliver its proof.

Proposition A1 *Let $G = (V,E,\omega)$ be a positively weighted graph with essential minor (C,ω_C) and let $\varepsilon \in \mathcal{P}(E)$ and ε_C be the image of ε in (C,δ). Then*

$$\varepsilon \in \mathcal{S}(G,\omega) \Longleftrightarrow \varepsilon_C \in \mathcal{S}(C,\delta)\,.$$

In particular $\Omega(G,\omega) = \Omega(C,\omega_C)$.

Proof Let $\varepsilon \in \mathcal{P}(E)$ and $\varepsilon_1 = (\varepsilon/_\Gamma)/_\Phi \subseteq E_1$ be the image of ε in $G_1 = (G/_\Gamma)/_\Phi$. We recall that $\omega_1 = (\omega/_\Gamma)/_\Phi$ was defined in Definition 4 and show that

$$\varepsilon \in \mathcal{S}(G,\omega) \Longleftrightarrow \varepsilon_1 \in \mathcal{S}(G_1,\omega_1) \text{ and } \Omega(G,\omega) = \Omega(G_1,\omega_1)\,.$$

Assume that $\varepsilon \in \mathcal{S}(G,\omega)$ then by Lemmas A1, A2 and the construction of (G_1,ω_1) we obtain $\Omega_{G,\omega}(\varepsilon) = \Omega_{G_1,\omega_1}(\varepsilon_1)$. Thus, if $\varepsilon_1 \notin \mathcal{S}(G_1,\omega_1)$ then we choose $\alpha \in \mathcal{S}(G_1,\omega_1)$ and a FAS $\varepsilon' \subseteq E$ of G such that the (19), (20) hold and $(\varepsilon'/_\Gamma)/_\Phi = \alpha$. Consequently $\Omega_{G,\omega}(\varepsilon') = \Omega_{G_1,\omega_1}(\alpha)$ and therefore due to the construction of (G_1,ω_1) we get

$$\Omega_{G,\omega}(\varepsilon') = \Omega_{G_1,\omega_1}(\alpha) < \Omega_{G_1,\omega_1}(\varepsilon_1) = \Omega_{G,\omega}(\varepsilon)\,,$$

which contradicts that $\varepsilon \in \mathcal{S}(G,\omega)$. Thus, $\varepsilon_1 \in \mathcal{S}(G_1,\omega_1)$.

Vice versa assume that $\varepsilon \subseteq E$ is such that $\varepsilon_1 \in \mathcal{S}(G_1,\omega_1)$. We claim that (19), (20) are satisfied by ε. Assume the opposite then due to Lemmas A2 and A1 we can delete an arc $e \in \varepsilon$ or replace an arc $e \in \varepsilon$ by an arc $f \in [e]_{\sim_\Gamma}$ with $\omega(e) > \omega(f)$. If this is not the case then we can delete all arcs $f \in F^+(e) \cap \varepsilon$ whenever e is such that $\emptyset \neq F^+(e) \cap \varepsilon \neq F^+(e)$. If ε' denotes this modified set, then ε' is FAS of G and in all cases

$$\Omega_{G_1,\omega_1}(\varepsilon_1) > \Omega_{G_1,\omega_1}(\varepsilon_1')\,.$$

A contradiction! Hence, the (19), (20) hold for ε and therefore the construction of (G_1,ω_1) yields

$$\Omega_{G,\omega}(\varepsilon) = \Omega_{G_1,\omega_1}(\varepsilon_1) = \Omega(G_1,\omega_1)\,.$$

Thus, if $\varepsilon \notin \mathcal{S}(G,\omega)$ then we choose $\beta \in \mathcal{S}(G,\omega)$ and obtain that β_1 is a FAS of G_1 with $\Omega_{G,\omega}(\beta) = \Omega_{G_1,\omega_1}(\beta) < \Omega(G_1,\omega_1)$, which is impossible. Hence $\varepsilon \in \mathcal{S}(G,\omega)$ and the claim follows by iteration of these arguments. □

Algorithm 9 $G/_\Gamma$

Input: $G = (V, E, \omega)$
Output: $G/_\Gamma, \kappa$
$\kappa(e) \leftarrow \{e\}, \forall e \in E$;
for $v \in V$ **do**
  if $\deg^-(v) = \deg^+(v) = 1$ **then**
    Let $u, w \in V : (u, v), (v, w) \in E$;
    $E \leftarrow E \cup \{(u, w)\} \setminus \{(u, v), (v, w)\}$;
    $V \leftarrow V \setminus \{v\}$;
    $\omega((u, w)) \leftarrow \min\{\omega((u, v)), \omega((v, w))\}$;
    $\kappa((u, w)) \leftarrow \operatorname{argmin}^*_{\{(u,v),(v,w)\}} \omega(\cdot)$;
  end
end
return $(G, \omega), \kappa$

Algorithm 10 $G/_\Phi, \kappa$

Input: $G = (V, E, \omega), \kappa$
Output: $G/_\Phi, \kappa$
for $e = (u, v) \in E$ **do**
  for $f \in F^+(e) \setminus \{e\}$ **do**
    $\omega(e) \leftarrow \omega(e) + \omega(f)$;
    $E \leftarrow E \setminus \{f\}$;
    $\kappa(e) \leftarrow \kappa(e) \cup \{f\}$;
  end
end
return G, κ

Proposition A2 *Let $G = (V, E, \omega)$ be a finite, connected, directed, weighted multigraph then we can construct (C, δ) in time $\mathcal{O}(|V||E|^2)$. Furthermore, there is an algorithm with run time $\mathcal{O}(|E|^2)$ which constructs a solution $\varepsilon \in S(G, \omega)$ given a solution $\varepsilon_C \in \mathcal{S}(C, \delta)$.*

Proof The graph $G/_\Gamma$ can be computed in a single iteration over V, see Algorithm 9, where κ is explained later. Each non branching node v is removed and its two incident arcs $e = (u, v)$ and $f = (v, w)$ are replaced by an arc (u, w) with weight $\min\{\omega(e), \omega(f)\}$. This is possible in time $\mathcal{O}(|V|)$ if the graph is represented as adjacency list where the targets of the outgoing arcs and the origins of the ingoing arcs are stored separately. Thus, the iteration over V yields a runtime of $\mathcal{O}(|V|^2)$.

To construct $G/_\sim\Phi$ we iterate over E_Γ yielding $G_1 = (G/_\Gamma)/_\Phi$, see Algorithm 10. For each arc e the weight is updated to $\sum_{e' \in F^+(e)} \omega(e')$ and the parallel arcs $F^+(e) \setminus \{e\}$ are purged from the graph. This can be realized in time $\mathcal{O}(|E| + |V|) = \mathcal{O}(|E|)$ with a counting sort prepossessing step if the target nodes in the adjacency list are stored such that equal targets are stored consecutively. Because the construction of (C, δ) requires at most $|V|$ iteration steps, i.e., if $(G_K, \omega_K) = (C, \delta)$ then $K \in$

$\mathcal{O}(|V|)$, the essential minor (C, δ) can be computed in time $\mathcal{O}(|V|^3 + |V||E|^2) \subseteq \mathcal{O}(|V||E|^2)$.

A simple extension of the algorithms allows to compute the information that is necessary to compute a solution $\varepsilon \in \mathcal{S}(G, \omega)$ once $\varepsilon_C \in \mathcal{S}(C, \delta)$ is given. During the application of Γ and Φ we store the set of arcs of G that are part of a solution if the corresponding arc from $G/_\Gamma$ and $G/_\Phi$, respectively, are in a FAS. That is, an arc that gave the minimum weight of the two arcs in a non branching path or all parallel arcs, respectively, see Lemma A1 and Lemma A2 . In Algorithms 9,10 this is realized by κ which can be considered as $\kappa : E_C \longrightarrow \mathcal{P}(E)$. The mapping is initialized as $\kappa(e) \leftarrow \{e\}$. Storing the arcs as linked list allows to update κ in linear time, i.e., the asymptotic run time of Algorithms 10,9 remains unchanged. Note that, argmin* returns only one arc in the case of equality. Now, replacing each arc $e \in \varepsilon_C$ by $\kappa(e)$ yields ε, which due to Proposition 2 is a solution for the FASP on (G, ω). Thus, the replacement can be realized in time $\mathcal{O}(|E|^2)$. □

Remark 15 Algorithm 9 may be extended to generate all solutions of the FASP for G given all solutions for the FASP on C. Therefore the equal weight alternatives in a non branching path need to be stored. The generation of the combinations of the alternatives of different paths yields all solutions. Certainly, then the run time depends exponentially on the number of possible combinations of alternatives.

References

1. Alon, N.: Ranking tournaments. SIAM J. Discret. Math. **20**(1), 137–142 (2006)
2. Arora, S., Barak, B.: Computational Complexity: A Modern Approach, 1st edn. Cambridge University Press, New York (2009)
3. Korte, B., Lovász, L.: Mathematical structures underlying greedy algorithms. Fundamentals of Computation Theory, Lecture Notes in Comp. Sci. pages 205–209 (1981)
4. Bang-Jensen, J., Gutin, G.Z.: Digraphs: Theory, Algorithms and Applications, 2nd edn. Springer Publishing Company, Incorporated (2008)
5. Berge, C.: Hypergraphs, volume 45 of North-Holland Mathematical Library. Combinatorics of Finite Sets, North-Holland (1989)
6. Berge, C.: The Theory of Graphs. Dover books on mathematics, Dover (2001)
7. Berger, B., Shor, P.W.: Approximation algorithms for the maximum acyclic subgraph problem. In SODA, pp. 236–243. SIAM (1990)
8. Biggs, N.: Algebraic Graph Theory, 2nd edn. Cambridge University Press, Cambridge (1993)
9. Chen, J., Liu, Y., Lu, S., O'sullivan, B., Razgon, I.: A fixed-parameter algorithm for the directed feedback vertex set problem. J. ACM **55**(5), 21:1–21:19 (2008)
10. Dinic, E.: Algortithm for solution of a problem of maximum flow in network with power estimates. Dokl. Akad. Nauk SSSR **194**(4), 1277–1280 (1970)
11. Dinitz, Y.: Dinitz' algorithm: The original version and even's version. In: Theoretical Computer Science, Essays in Memory of Shimon Even, pp. 218–240 (2006)
12. Dinur, I., Safra, S.: On the hardness of approximating minimum vertex cover. Ann. Math., 162 (2005)
13. Even, G., (Seffi) Naor, J., Schieber, B., Sudan, M.: Approximating minimum feedback sets and multicuts in directed graphs. Algorithmica **20**(2), 151–174 (1998)
14. Fortune, S., Hopcroft, J., Wyllie, J.: The directed subgraph homeomorphism problem. Theor. Comput. Sci. **10**(2), 111–121 (1980)
15. Gavril, F.: Some NP-complete problems on graphs. In: 11th Conference on Information Sciences and Systems, pp. 91–95 (1977)

16. Grötschel, M., Jünger, M., Reinelt, G.: On the acyclic subgraph polytope. Math. Program. **33**(1), 28–42 (1985)
17. Grötschel, M., Lovasz, L., Schrijver, A.: The ellipsoid method and its consequences in combinatorial optimization. Combinatorica **1**(2), 169–197 (1981)
18. Huang, H., Ma, J., Shapira, A., Sudakov, B., Yuster, R.: Large feedback arc sets, high minimum degree subgraphs, and long cycles in eulerian digraphs. Comb. Probab. Comput. **22**, 859–873 (2013)
19. Ispolatov, I., Maslov, S.: Detection of the dominant direction of information flow and feedback links in densely interconnected regulatory networks. BMC Bioinforma. **9**(1), 424 (2008)
20. Johnson, D.B.: Finding all the elementary circuits of a directed graph. SIAM J. Comput. **4**(1), 77–84 (1975)
21. Kann, V.: On the approximability of the maximum common subgraph problem. In: STACS 92, 9th Annual Symposium on Theoretical Aspects of Computer Science, Cachan, France, February 13-15, 1992, Proceedings, pp. 377–388 (1992)
22. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W. (eds.) Complexity of Computer Computations, pp. 85–103 (1972)
23. Karpinski, M., Schudy, W.: Faster Algorithms for Feedback Arc Set Tournament, Kemeny Rank Aggregation and Betweenness Tournament 3–14. Springer Berlin Heidelberg, Berlin Heidelberg (2010)
24. Kunzmann, A., Wunderlich, H.-J.: An analytical approach to the partial scan problem. J. Electron. Test. **1**(2), 163–174 (1990)
25. Leiserson, C.E., Saxe, J.B.: Retiming synchronous circuitry. Algorithmica **6**(1-6), 5–35 (1991)
26. Lucchesi, C., Younger, D.: A minimax theorem for directed graphs. J. Lond. Math. Soc. **2, 17**(3), 369–374 (1978)
27. Martí, R.: The Linear Ordering Problem: Exact and Heuristic Methods in Combinatorial Optimization. Applied Mathematical Sciences. Springer (2011)
28. Mateti, P., Deo, N.: On algorithms for enumerating all circuits of a graph. SIAM J. Comput. **5**(1), 90–99 (1976)
29. Ramachandran, V.: Finding a minimum feedback arc set in reducible flow graphs. J. Algorithm. **9**(3), 299–313 (1988)
30. Razgon, I.: Computing minimum directed feedback vertex set in $o(1.9977^n)$. In: Theoretical Computer Science, 10th Italian Conference, ICTCS 2007, Rome, Italy, October 3-5, 2007, Proceedings, pp. 70–81 (2007)
31. Roberts, B., Kroese, D.P.: Estimating the number of s-t paths in a graph. J. Graph Algorithm. Appl. (2007)
32. Saab, Y.: A fast and effective algorithm for the feedback arc set problem. J. Heuristics **7**(3), 235–250 (2001)
33. Silberschatz, A., Galvin, P.B., Gagne, G.: Operating System Concepts, 8th edn. Wiley Publishing (2008)
34. Tarjan, R.E.: Enumeration of the elementary circuits of a directed graph. SIAM J. Comput. **2**(3), 211–216 (1973)