FEEDBACK SET PROBLEMS

PAOLA FESTA, PANOS M. PARDALOS, AND MAURICIO G.C. RESENDE

ABSTRACT. This paper is a survey of feedback set problems (FSP). FSP originated from applications in combinatorial circuit design, but have found their way into numerous other applications, such as deadlock prevention in operating systems, constraint satisfaction and Bayesian inference in artificial intelligence, and graph theory. Directed and undirected feedback vertex set problems are considered, including polynomially solvable cases, approximation algorithms, exact algorithms, and practical heuristics. The relationship between the feedback vertex set and feedback arc set problems is examined and the state of the art of feedback arc set problems is surveyed. Applications of feedback set problems are described. Finally, future directions in feedback set problem research are mapped out.

1. INTRODUCTION

Not long ago, there appeared to be a consensus in the literature that feedback set problems, which originated from the area of combinational circuit design, were the least understood among all the classical combinatorial optimization problems due to the lack of positive results in efficient exact and approximating algorithms. This picture has been totally changed in recent years. Dramatic progress has occurred in developing approximation algorithms with provable performance; new bounds have been established one after the other and it is probably fair to say that feedback set problems are becoming among the most exciting frontend problems in combinatorial optimization.

The most general feedback set problem consists in finding a minimum-weight (or minimum cardinality) set of vertices (arcs) that meets all cycles in a collection C of cycles in a graph (G,w), where w is a nonnegative function defined on the set of vertices $V(G)$ (on the set of edges $E(G)$). This kind of problem is also known as the *hitting cycle problem*, since one must hit every cycle in C. It generalizes a number of problems, including the *minimum feedback vertex (arc) set problem* in both directed and undirected graphs, the *subset minimum feedback vertex (arc) set problem* and the *graph bipartization problem*, in which one must remove a minimum-weight set of vertices so that the remaining graph is bipartite. In fact, if C is the set of all cycles in G, then the hitting cycle problem is equivalent to the problem of finding the minimum feedback vertex (arc) set in a graph. If we are given a set of *special* vertices and C is the set of all cycles of an undirected graph G that contains some special vertex, then we have the *subset feedback vertex (arc) set problem* and, finally, if C contains all odd cycles of G, then we have the *graph bipartization problem*. All these problems are also special cases of *vertex (arc) deletion problems*, where one seeks a minimum-weight (or minimum cardinality) set of vertices (arcs) whose deletion gives a graph satisfying a given property.

There are different versions of *feedback set problems*, depending on whether the graph is directed or undirected and/or the vertices (arcs) are weighted or unweighted. Yannakakis [88] has given a general NP-hardness proof for almost all vertex and arc deletion problems

Date: February 18, 1999, Revised April 2, 1999.
AT&T Labs Research Technical Report: 99.2.2.

restricted to planar graphs. These results apply to the planar bipartization problem, the planar (directed, undirected, or subset) feedback vertex set problems, already proved to be NP-hard [45, 32]. Furthermore, it is NP-complete for planar graphs with no indegree or outdegree exceeding three [45], general graphs with no indegree or outdegree exceeding two [45], and edge-directed graphs [45].

2. NOTATION AND GRAPH REPRESENTATION

Throughout this chapter, we use the following notation and definitions.

A *graph* $G = (V,E)$ consists of a finite set of vertices $V(G)$, and a set of arcs $E(G) \subseteq V(G) \times V(G)$. An *arc* (or edge) $e = (v_1, v_2)$ of a directed graph (digraph) $G = (V,E)$ is an *incoming* arc to v_2 and an *outgoing* arc from v_1 and it is *incident* to both v_1 and v_2. If G is undirected, then e is said only incident to v_1 and v_2.

For each vertex $i \in V(G)$, let $in(i)$ and $out(i)$ denote the set of *incoming* and *outgoing* edges of i, respectively. They are defined only in case of a digraph G. If G is undirected, we will take into account only the *degree* $\Delta_G(i)$ of i as the number of edges that are incident to i in G. $\Delta(G)$ denotes the maximum degree among all vertices of a graph G and it is called the *graph degree*. A vertex $v \in G$ is called an *endpoint* if it has degree one, a *linkpoint* if it has degree two, while a vertex having degree higher than two is called a *branchpoint*.

A *path* P in G connecting vertex u to vertex v is a sequence of arcs $e_1, \cdots, e_r$ in $E(G)$, such that $e_i = (v_i, v_{i+1})$, $i = 1, \cdots, r$ with $v_1 = u$ and $v_{r+1} = v$. A *cycle* C in G is a path $C = (v_1, \cdots, v_r)$, with $v_1 = v_r$.

A *subgraph* $G^{'} = (V^{'}, E^{'})$ *of* $G = (V,E)$ *induced by* $V^{'}$ is a graph such that $E^{'} = E \cap (V^{'} \times V^{'})$. A graph G is said to be a *singleton*, if $|V(G)| = 1$. Any graph G can be partitioned into isolated connected components $G_1, G_2, \cdots, G_k$ and the partition is unique. Similarly, every feedback vertex set $V^{'}$ of G can be partitioned into feedback vertex sets $F_1, F_2, \cdots, F_k$ such that F_i is a feedback vertex set of G_i. Therefore, following the additive property and denoting by $\mu(G,w)$ the weight of a minimum feedback vertex (arc) set for (G,w), we have:

$$\mu(G,w) = \sum_{i=1}^{k} \mu(G_i, w).$$

3. THE FEEDBACK VERTEX SET PROBLEM

Most of the known results on vertex deletion problems deal with the feedback vertex set problem, that can be easily understood by the following deadlock prevention example in computer systems. Consider an operating system which schedules different processes with requests on different resources, which they need to use exclusively before being released by the process. A directed graph modeling these resource requirements has a node i for each process i with directed arc $e(i, j)$ implying that process i requests a resource already allocated to process j. Therefore, if there is a directed cycle in such a graph, a deadlock occurs and every process in the cycle will wait for the requested resource and will never release the resources already allocated to it. To break such cycles, one can remove some processes from the graph and put them in a waiting queue. It is clear that we want to minimize the number of processes removed.

Formally, the feedback vertex set problem can be described as follows. Let $G = (V,E)$ be a graph and let $w : V(G) \to R^+$ be a weight function defined on the vertices of G. A *feedback vertex set* of G is a subset of vertices $V^{'} \subseteq V(G)$ such that each cycle in G contains at least one vertex in $V^{'}$. In other words, a feedback vertex set $V^{'}$ is a set of vertices of G

such that by removing $V^{'}$ from G along with all the edges incident to $V^{'}$, results in a forest. The weight of a feedback vertex set is the sum of the weights of its vertices, and a *minimum feedback vertex set* of a *weighted graph (G,w)* is a feedback vertex set of G of minimum weight. The weight of a minimum feedback vertex set will be denoted by $\mu(G,w)$. The *minimum weighted feedback vertex set problem* (MWFVS) is to find a minimum feedback vertex set of a given weighted graph (G,w). The special case of identical weights is called the *unweighted feedback vertex set problem* (UFVS).

3.1. **Mathematical model of the feedback vertex set problem.** The feedback vertex set problem is a specific type of *set covering problem*, as the objective is to cover all cycles with a minimum cost collection of vertices. A simple polynomial reduction procedure from the vertex cover problem to the feedback vertex set problem is described in [2]. As a covering-type problem, it admits an integer zero-one programming formulation. Given a feedback vertex set $V^{'}$ for a graph (G,w), $G=(V,E)$, and a set of weights $w=\{w(v)\}_{v\in V(G)}$, let $x=\{x_v\}_{v\in V(G)}$ be a binary vector such that $x_v=1$ if $v\in V^{'}$, and $x_v=0$ otherwise. Let C be the set of cycles in (G,w). The problem of finding the minimum feedback vertex set of G can be formulated as an integer programming problem as follows:

$$\textbf{(MFVS)}\ \begin{cases} \min \displaystyle\sum_{v\in V(G)} w(v)\ x_v \\ \text{s.t.} \displaystyle\sum_{v\in V(\Gamma)} x_v \geq 1 \quad \forall\, \Gamma\in C \\ 0\leq x_v \leq 1 \text{ integer, } v\in V(G). \end{cases}$$

If one denotes by C_v the set of cycles passing through vertex $v\in V(G)$, then the dual of the linear programming relaxation of (MFVS), is a *packing problem*:

$$\textbf{(DMFVS)}\ \begin{cases} \max \displaystyle\sum_{\Gamma\in C} y_\Gamma \\ \text{s.t.} \displaystyle\sum_{\Gamma\in C_v} y_\Gamma \leq w(v) \quad \forall\, v\in V(G) \\ y_\Gamma \geq 0 \quad \forall\, \Gamma\in C. \end{cases}$$

3.2. **Polynomially solvable cases.** As it is unlikely that there exist polynomial time algorithms to solve NP-hard problems, to obtain polynomial time algorithms one has to restrict these problems to special classes of graphs. Perfect graphs are a class of graphs on which polynomial time algorithms have been found for a variety of problems. Because the subclasses of perfect graphs form hierarchies, one problem of interest is to determine the largest class of graphs on which such problems remain polynomially solvable. Therefore, given the NP-completeness of the feedback vertex set problem, one approach is to identify those specially structured problems which can be solved in polynomial time. Research along this line started with the pioneering work of Shamir [76], in which a linear time algorithm was proposed to find a feedback vertex set for a reducible flow graph. Wang, Lloyd, and Soffa [87] developed an $O(|E(G)|\cdot|V(G)|^2)$ algorithm for finding a feedback vertex set. The class of graphs known as *cyclically reducible graphs*, which is shown to be unrelated to the class of quasi-reducible graphs. Although the exact algorithm proposed by Smith and Walford [80] has exponential running time in general, it returns an optimal solution in polynomial time for certain types of graphs. A variant of the algorithm, called the Smith-Walford-one algorithm, selects only candidate sets F of size one and runs in $O(|E(G)|\cdot|V(G)|^2)$ time. The class of graphs for which it finds a feedback vertex set is called Smith-Walford one-reducible.

In the following, consider a set of operations called *contraction operations*. These operations contract the graph while preserving all the important properties relevant to the minimum feedback vertex set. An important property associated with these operations is the so-called Church-Rosser property which implies that the order by which a sequence of operations is performed will not affect the final graph. The basic contraction operations are as follows:

Reduction Procedures

Let (G,w) be a vertex (arc) weighted graph and let $V^{'}$ be a feedback vertex (arc) set of G, then

IN0(i) - OUT0(i):

if $|out(i)| = 0$ or $|in(i)| = 0$, then $i \in V^{'}$.
reduction: $V(G) = V(G) \setminus \{i\}$;
$E(G) = E(G) \setminus \{(x,y) |\ x = i \text{ or } y = i\}$.

LOOP(i):

if $(i,i) \in E(G)$, then $i \in V^{'}$.
reduction: $V(G) = V(G) \setminus \{i\}$;
$E(G) = E(G) \setminus \{(i,j) \text{ or } (j,i)\ \forall j \in V(G)\}$.

IN1(i):

if $|in(i)| = 1$ and $(j,i) \in E(G)$,
reduction: $V(G) = V(G) \setminus \{i\}$;
$out(j) = out(j) \cup out(i)$;
$E(G) = E(G) \cup \{(j,k) | k \in out(i)\} \setminus \{(i,k) | k \in out(i)\}$.

OUT1(i):

if $|out(i)| = 1$ and $(i,j) \in E(G)$
reduction: $V(G) = V(G) \setminus \{i\}$;
$in(i) = in(i) \cup in(j)$;
$E(G) = E(G) \cup \{(k,j) | k \in in(i)\} \setminus \{(k,i) | k \in in(i)\}$.

Levy and Lowe [55] proposed these operations and proved the following properties.

Definition 1. *Let $G = (V,E)$ be a directed graph, and let $G' = (V',E')$ be a directed graph resulting from G by repeated applications of the contraction operations until no further contraction is possible. G' is called a contracted graph of G.*

Theorem 1. *If G contains no parallel edges, then the contracted graph G' of G is unique.*

Theorem 2. *Let $G = (V,E)$ be a directed graph. (1) If G is contracted into G' by any of the operations IN0(v), OUT0(v), IN1(v), or OUT1(v), and if S' is a minimum feedback vertex set of G', then S' is a minimum feedback vertex set of G. (2) If G is contracted into G' by LOOP(v) and S' is a minimum cutset of G', then $S = S' \cup \{v\}$ is a minimum feedback vertex set of G.*

It should be noted that for a directed graph with every node having at least two in-edges and two out-edges, the set of contraction rules does not apply and all the above mentioned algorithms will fail to find an optimal feedback vertex set. In other words, this class of algorithms is only optimal for specially structured and very sparse graphs. Nevertheless, this line of work has significant impact in the study of feedback vertex set for the following two reasons. First, a class of graphs of increasing size is computed, where the feedback vertex set of each graph can be found exactly. Second, most proposed heuristics and approximation algorithms use the reduction schemes discussed above.

3.2.1. *The feedback vertex set problem on chordal, permutation, and interval graphs.* Another line of research on polynomially solvable cases focuses on other special classes, including *chordal* and *interval* graphs, *permutation* graphs, *convex bipartite* graphs, *cocomparability* graphs and on *meshes* and *toroidal meshes*, *butterflies*, and *toroidal butterflies*.

The feedback vertex set on chordal and interval graphs can be viewed as a special instance of the generalized clique cover problem, which is solved in polynomial time on chordal graphs [21, 89] and interval graphs [63]. For permutation graphs, an algorithm due to Brandstädt and Kratsch [7] was improved by Brandstädt [8] to run in $O(|V(G)|^6)$ time. More recently, Liang [57] presented an $O(|V(G)| \cdot |E(G)|)$ algorithm for permutation graphs that can be easily extended to *trapezoid graphs* while keeping the same time complexity. A graph $G = (V,E)$ is called *trapezoid* if and only if it is the intersection graph of some trapezoid diagram, where a *trapezoid diagram* T consists of two parallel lines called top and bottom lines and some trapezoids, each having two corner points on the top line and two on the bottom line. The intersection graph of a trapezoid diagram is the graph whose vertices have a one-to-one correspondence with the trapezoids in T and two vertices in G are adjacent if and only if the corresponding trapezoids in T intersect, i.e. their areas overlap. Both permutation and interval graphs are special trapezoid graphs.

On interval graphs, Lu and Tang [17] developed a linear-time algorithm to solve the minimum weighted feedback vertex set problem using dynamic programming. Interval graphs are special graphs that admit a so called *interval representation* F. An interval graph corresponds to a set of intervals (interval representation) in the real line. Each interval corresponds to a vertex and there is an edge between two vertices if and only if the corresponding intervals intersect. Such a representation of an interval graph can be obtained in $O(|V(G)| + |E(G)|)$ time, assuming sorted endpoints. Lu and Tang observed that a subset V' of $V(G)$ is a feedback vertex set of G if $V(G) \setminus V'$ is a *cycle-free vertex set* (CVS) of G. When G is weighted, V' is a minimum feedback vertex set of G if and only if $V(G) \setminus V'$ is a maximum weighted CVS of G. Consequently, given an interval representation I of a weighted interval graph G with sorted endpoints, and assuming without loss of generality, that no two intervals share a common endpoint, the linear-time algorithm of Lu and Tang finds the maximum weighted CVS of G to solve the minimum weighted feedback vertex set problem of G. At the same time, it also solves the maximum weighted 2-colorable subgraph problem and the maximum weighted 2-independent set problem, which are equivalent on chordal graphs.

3.2.2. *The feedback vertex set problem on cocomparability graphs.* Coorg and Rangan [20] present an $O(|V(G)|^4)$ time and $O(|V(G)|^4)$ space exact algorithm for cocomparability graphs, which are a superclass of permutation graphs. In more detail, a graph $G(V,E)$ is said to be a cocomparability graph if and only if its complement graph is *transitively orientable*, i.e. its edges can be oriented to get a directed graph $\tilde{G} = (V, \tilde{E})$ such that

$$(i,j),\ (j,k) \in \tilde{E} \implies (i,k) \in \tilde{E}.$$

The authors proved that to solve the minimum feedback vertex set problem on cocomparability graphs is equivalent to finding a minimum *cycle-free* set of the given graph. The key idea is that a *cycle-free* subgraph of G is a collection of trees, because it does not contain any cycle. Therefore, a *cycle-free* subgraph is bipartite and admits a *planar embedding*, i.e. a mapping of its n vertices to a set of n points on the plane and a mapping from its m edges to a set of m line segments on the plane such that:

- there exists a line segment $[p_1, p_2]$ if and only if $(v_1, v_2) \in E(G)$, where v_1 and v_2 are mapped to p_1 and p_2, respectively;
- any two line segments $[p_1, p_2]$ and $[p_3, p_4]$ intersect only at their endpoints.

A natural way to embed a cycle-free subgraph on the plane is the *canonical ordering* of the vertices of the transitively oriented graph $G^{''}$ obtained from G through its complement $G^{'} = (V^{'}, E^{'})$ which can be done in $O(n^2)$ time using topological sort. An *ordering* of the vertices of $G^{''}$ is *canonical* if and only if

$$(i, j) \in E(G^{''}) \Longrightarrow i < j.$$

For $G^{''}$ such an ordering always exists, because $G^{''}$ is a directed acyclic graph. Once a planar embedding is obtained, the algorithm of Coorg and Rangan conceptually triangulates the planar graph so that the cycle-free subgraph can be considered as a set of triangles, each having three vertices of G. In more detail, the cycle-free subgraph is incrementally built, starting from the empty set and adding one triangle at time. At each step it maintains the incremented subgraph cycle-free. The authors prove that the cycle-free property can be maintained by checking only the last inserted triangle. They model this process by constructing a directed graph H, whose vertices correspond to the triangles and an edge (u, v) exists whenever the algorithm can safely add triangle v to the set immediately after u has been added. A directed path of length k in H corresponds to a cycle-free subgraph of G having $k+3$ vertices and the minimum feedback vertex set problem is reduced to that of finding the longest path in a directed graph, which can be easily solved. In fact, to find the longest path in H takes time proportional to the number of the edges in H, by using, for example, depth-first search. Since there are at most $O(n)$ neighbors for each vertex in H, the number of edges in H is bounded by $O(n^4)$ and since to construct H requires $O(n^4)$ time, the complexity of the algorithm due to Coorg and Rangan remains $O(n^4)$.

More recently, Liang and Chang [13] developed a polynomial time algorithm, that by exploring the structural properties of a cocomparability graph uses dynamic programming to get a minimum feedback vertex set in $O(|V(G)^2| \; |E(G)|)$ time.

3.2.3. *The feedback vertex set problem on meshes and butterflies.* A recent line of research on polynomially solvable cases focuses on special undirected graphs having bounded degree and that are widely used as connection networks, namely *meshes* and *toroidal meshes*, *butterflies* and *toroidal butterflies*.

Definition 2. *An $m \times n$ mesh is a graph $M_{m,n} = (V, E)$, where $V = \{v_{ij} | \; 0 \leq i \leq m-1, 0 \leq j \leq n-1\}$ and $E = \{(v_{ij}, v_{i(j+1)}), (v_{ij}, v_{(i+1)j}) | i = 0, \cdots, n-2, j = 0, \cdots, m-2\}$.*

Definition 3. *A toroidal $m \times n$ mesh is a graph $TM_{mn} = (V, E)$ obtained from a mesh $M_{(m+1)(n+1)}$ identifying the vertices v_{0j} with v_{mj}, $0 \leq j \leq n$ and v_{j0} with v_{jn}, $0 \leq j \leq m$.*

In a mesh $M_{m,n}$, Luccio [61] obtained a trivial lower bound on the size of the minimum feedback vertex set $V^{'}$ equal to $((m-1)(n-1)+1)/4$, by observing that every submesh M_{22} is a cycle of 4 vertices to be broken and one vertex of $V^{'}$ breaks at most 4 such cycles. Luccio proved the stronger lower bounds $|V^{'}| \geq ((m-1)(n-1)+1)/3$ in a mesh M_{mn} and $(mn+2)/3$ in a toroidal mesh TM_{mn}. In [61], upper bounds for meshes of size $(2^r+1) \times (2^r+1)$ and toroidal meshes $TM_{2^r 2^r}$ are derived. These bounds either match the lower bounds or are very close to them. The proofs of the upper bounds are constructive, in the sense that an algorithm that derives them also finds a minimum feedback vertex set $V^{'}$ for the given graph, simply by investigating their topological characteristics. In more detail, in the case of meshes $M_{(2^r+1)(2^r+1)}$, V' is formed by patterns of vertices lying on the diagonal lines that constitute the boundaries of diamonds of varying sizes 2^j+1, $0 \leq j \leq r-2$, each having exactly one vertex not belonging to V'. By applying this algorithm to $M_{(n+1)(n-1)}$ and removing the $(n+1)^{th}$ row and the $(r+1)^{th}$ column, on which no vertex of V' lies, one can obtain a minimum feedback vertex set for $TM_{2^r2^r}$. This algorithm can be still applied to arbitrary values of m and n. Actually, Luccio showed that to obtain a minimum feedback vertex set for a mesh M_{mn}, $2^{r-1}+1 < m \leq 2^r+1$, $2^{s-1}+1 < n \leq 2^s+1$, $r,s \geq 0$, $t = \max\{r,s\}$, it is enough to apply the proposed algorithm to build V' for $M_{(2^t+1)(2^t+1)}$, and then the required minimum feedback vertex set for M_{mn} is the portion of V' lying in the upper left submesh M_{mn}. To get a minimum feedback vertex set V'' for TM_{mn} it is possible to do the same, with the only difference that now additional vertices are needed to be inserted along the external boundary of V'' in order to break possible cycles between the first and the last row and the first and the last column, thus increasing the upper bound by an additional linear term.

For *butterfly* and *toroidal butterfly* graphs, Luccio [61] found upper bounds to the size of a minimum feedback vertex set.

Definition 4. *A k-dimensional butterfly is a graph $B_k = (V,E)$ having $(k+1)2^k$ vertices organized in $k+1$ levels of 2^k vertices each. Vertex v_{ij} denotes the j^{th} vertex at level i, $0 \leq i \leq k$, $0 \leq j \leq 2^k-1$, and for $i > 0$ it is connected with the two vertices $v_{(i-1)j}$ and $v_{(i-1)j_i}$, where j_i is the integer whose binary representation differs from that of j in only the i^{th} least significant bit.*

Definition 5. *A toroidal butterfly TB_k is a k-dimensional butterfly in which the k^{th} level coincides with level* 0.

In a k-dimensional butterfly, the vertices in every two adjacent levels form 2^{k-1} cycles with two vertices in the upper level and two in the lower level. The lower bound of the size on a minimum feedback vertex set $\bar{V}$ for this graph is

$$|\bar{V}| \geq 2^{k-1}\lfloor\frac{(k+1)}{2}\rfloor,$$

because each vertex at level i, $1 \leq i \leq k-1$, belongs to exactly two cycles: one between level $i-1$ and i and one between i and $i+1$. The upper bound is

$$|\bar{V}| \leq (2^{k-2}+2^{k-4}+2^{k-5}+1)\, k,$$

found by applying an algorithm that for each vertex $v_{(i-1)j}$ adds to $\bar{V}$ either v_{ij} or v_{ij_i} connected with $v_{(i-1)j}$ at the next level.

Similar results to those obtained for butterflies B_k can also be obtained for toroidal butterflies TB_k. In fact, TB_k has $k2^k$ vertices and the vertices belonging to the $(k-1)^{th}$ level are connected to those of level 0. Therefore, such kinds of graphs contain 2^k cycles wrapped around each column. To break all of them, it is necessary to remove at least

$$\max\{(2^{k-1})\lfloor\frac{k+1}{2}\rfloor,\ 2^k\}$$

vertices, where 2^k is significant only for $k \leq 2$. To find a minimum feedback vertex set $\bar{V}$, the same algorithm as for B_k can be applied, by including in $\bar{V}$ all vertices belonging to the $(k-1)^{th}$ level.

3.2.4. *The feedback vertex set problem on cube connected cycle networks.* Luccio [61] solved the minimum feedback vertex set problem also for another important bounded degree network, the *k-dimensional cube connected cycle* (CCC_k), that has k levels of 2^k vertices each. The only difference between TB_k and CCC_k is that any vertex v_{ij} is now connected with $v_{(i-1)j}$ and $v_{i(j+1)}$, $0 \leq i \leq k-1$, $0 \leq j \leq 2^k - 1$, and the operations on i are computed modulo k. Observing that the vertices in every two adjacent levels form 2^{k-2} cycles of length 8 (4 in the upper level and 4 in the lower level) and that each vertex belongs exactly to two such cycles, at least $2^{k-2}\lfloor (k+1)2 \rfloor$ vertices have to be removed in order to break all such cycles. Moreover, because CCC_k also contains 2^k cycles wrapped around each column, a lower bound on the minimum number of vertices to be removed is given by

$$\max\{2^k,\ 2^{k-2}\lfloor \frac{(k+1)}{2} \rfloor\},$$

where 2^k is insignificant for $k \leq 6$. The upper bound found by Luccio is $2^{k-1}\lfloor \frac{(k+1)}{2} \rfloor$, obtained by building a minimum feedback vertex set using a variant of the algorithm that finds a minimum feedback vertex set for TB_k. In this case, for each level $i = 1, \cdots, k$, incrementing i by 2, 2^{k-1} vertices are inserted in the feedback set to break all cycles with a vertex at the lower level $i-1$. The remaining vertices belonging to level $k-1$ are inserted in the feedback set.

3.2.5. *The feedback vertex set problem on convex bipartite graphs.* Liang and Chang [13] solved in polynomial time the feedback vertex set problem in a special kind of bipartite graph, called the *convex bipartite graph*, whose definition is given next.

Definition 6. *A bipartite graph $G = (A, B, E)$ with two distinct sets of vertices A and B is convex if there exists a total ordering on A such that for any vertex $b \in B$ the set of vertices of A connected to b forms an interval in this ordering.*

Liang and Chang [13] proposed a polynomial time algorithm having time complexity $O(|A|^3 + |A|^2|E|)$. Their algorithm assumes a convex bipartite graph $G = (A, B, E)$ is given by specifying the total ordering on $A = \{a_1, a_2, \cdots, a_n\}$ with $a_1 < a_2 < \cdots < a_n$. The algorithm is based on dynamic programming techniques and the special structure of the graph.

3.3. **Approximation algorithms and provable bounds.** The feedback vertex (arc) set problem has found applications in many fields, including deadlock prevention [87], program verification [76], and Bayesian inference [2]. Therefore, it is natural that in the past few years there have been intensive efforts on approximation algorithms for these kinds of problems, for the cases that are not known to be polynomially solvable.

To quantify the quality of an approximation, several criteria have been established and certain classes of approximation schemes have been defined. The family of approximation algorithms that guarantees the best approximate solution is the so called *fully polynomial approximation scheme* (FAS). It is a family of algorithms $\{A_\varepsilon\}$ that for a maximization problem finds an approximate solution at least $(1-\varepsilon)$ times the optimal solution in time polynomial in the length of the input and in $1/\varepsilon$. In case of a minimization problem, the algorithm is guaranteed to find an approximate solution at least $(1+\varepsilon)$ times the value of the optimal solution. The next family of "good" approximation algorithms is the *polynomial approximation scheme* (PAS). It contains approximation algorithms that produce a solution at least $(1+\varepsilon)$ times the optimal solution in time polynomial in the length of the input for fixed ε.

When an approximation algorithm does not have any of the aforementioned characteristics, it can still be "evaluated" as well. In fact, to test the quality of the approximate solution there are special quantities called *performance ratios*, defined as follows. Suppose A is an approximation algorithm that finds a feedback vertex set V_A (arc set E_A) for any given vertex (arc) weighted graph (G,w). Denoting the sum of weights of vertices (arcs) in V_A (E_A) by $w(V_A)$ and the weight of a minimum feedback vertex (arc) set for (G,w) by $\mu(G,w)$, then the *performance ratio of A for* (G,w) is defined by

$$R_A(G,w) = w(V_A)/\mu(G,w).$$

When $\mu(G,w) = 0$, $R_A(G,w) = 1$ if $w(V_A) = 0$ and $R_A(G,w) = \infty$ if $w(V_A) > 0$.

In other words, the performance ratio of an approximation algorithm is the worst-case ratio between the weight of the algorithm's output and the weight of an optimal solution.

The *performance ratio* $r_A(n,w)$ *of A for w* is the supremum of $R_A(G,w)$ over all graphs G with n vertices and for the same weight function w. If w is the constant function ($w(i) = 1$), $r_A(n,1)$ is called the *unweighted performance ratio of A*. The *performance ratio* $r_A(n)$ *of A* is the supremum of $r_A(n,w)$ over all weight functions w.

Many approximation algorithms for feedback set problems have been proposed in the last two decades. We consider approximation algorithms for undirected and directed graphs next.

3.3.1. *Undirected graphs.* A $2\log_2|V(G)|$-approximation algorithm for the unweighted minimum feedback vertex set problem on undirected graphs is contained in a lemma due to Erdös and Posa [26]. This result was improved by Monien and Schulz [64] to obtain a performance ratio of $O(\sqrt{\log|V(G)|})$.

Bar-Yeruda, Geiger, Naor, and Roth [2] gave an approximation algorithm for the unweighted undirected case having ratio less than or equal to 4 and two approximation algorithms for the weighted undirected case having ratios $4\log_2|V(G)|$ and $2\Delta^2(G)$, respectively. A slight generalization of the unweighted case was used to reduce the complexity of a Bayesian inference procedure as will be discussed in Section 5. In their algorithm, it is assumed that the set of vertices $V(G)$ is partitioned into a nonempty set $A(G)$ of *allowed* vertices and a possibly empty set $B(G)$ of *blackout* vertices. For a *valid graph G* a feedback vertex set exists if and only if every cycle in G contains at least one allowed vertex. Note that if $B(G) = \emptyset$, then the problem reduces to the classical unweighted feedback vertex set. Before giving a description of their algorithm, the following definitions are needed.

Definition 7. *A* 2-3-*subgraph of a valid graph G is a subgraph H of G such that the degree in H of every vertex in* $V(G)$ *is either* 2 *or* 3. *A maximal* 2-3-*subgraph of G is a* 2-3-*subgraph of G that is not a subgraph of any other* 2-3-*subgraph.*

A maximal 2-3-subgraph of G always exists if G is a valid graph that is not a forest.

Definition 8. *A linkpoint v in a* 2-3-*subgraph H is said to be a critical linkpoint if it is an allowed vertex and there is a cycle* Γ *in G such that*

$$V(\Gamma) \cap V(H) \cap E(G) = \{v\}.$$

In this case Γ *is called a witness cycle of v.*

Definition 9. *A cycle in a valid graph G is branchpoint-free if it does not pass through any allowed branchpoint, i.e. a branchpoint-free cycle is made of only blackout vertices and allowed linkpoints of G.*

The algorithm in [2] takes as input a 2-3-subgraph of the given valid graph G and gives as output a feedback vertex set of G, containing the set X of all critical linkpoints, the set Y of allowed branchpoints in the maximal 2-3-subgraph of G, and the vertices of a set W covering all branchpoint-free cycles of H not covered by X. The set X can be found by applying depth-first search on G and the set Y by applying breadth-first search. The complexity of the algorithm is linear in the number of edges of G. To speedup the algorithm, Bar-Yeruda et al. show how to preprocess the input valid graph by applying the corresponding undirected versions of the reduction transformations given in Subsection 3.2, by being careful during the process in not generating any blackout vertex cycle that violates the validity property of the reduced graph.

In more detail, in the undirected case, the reduction procedures described in Subsection 3.2 degenerate to the following. A *reduction graph* $G^{'}$ of a graph G is a graph obtained from G by a sequence of the following transformations:

- Delete an endpoint and its incident edges.
- Connect two neighbors of a linkpoint v (other than a self-looped singleton) by a new edge and remove v from the graph with its incident edges.

It was proved in [2] that any valid reduction graph $G^{'}$ of a graph G is such that $\mu(G^{'},w) = \mu(G,w)$. Moreover, if the reduction graph G^* of G is *minimal*, i.e. it is valid and any proper reduction graph $G^{'}$ of G^* is not valid, it is shown that G^* does not contain any blackout linkpoints and that any of its feedback vertex sets contains all allowed vertices of G^*. In this case, if it is possible to obtain a valid graph G^* without any blackout linkpoints and any endpoints, as the reduction transformations do, then for every feedback vertex set $V^{'}$ of G^* containing all linkpoints of G, we have that

$$|V(G)| \leq (\Delta(G)+1)|V^{'}|,$$

where $\Delta(G)$ is the largest degree among all vertices in G.

In [2], the weighted feedback vertex set problem for undirected graphs was also solved by proposing two approximation algorithms having performance ratios $4\log_2|V(G)|$ and $2\Delta^2(G)$, respectively. Both of the algorithms use reduction graph transformations. The first algorithm finds at each iteration a minimal weighted reduction graph G^* of the input graph (G,w), i.e. a minimal reduction graph such that any of its reduction graphs $G^{'}$ is equal to G^*. Such a graph is necessarily *branchy*, that is, it does not contain any endpoints and its set of linkpoints induces an independent set (i.e. each linkpoint is either an isolated self-loop or it is connected to two branchpoints). Note that transforming a graph into a branchy graph takes $O(|E(G)|)$ time. The algorithm then proceeds to find a cycle Γ in the minimal weighted reduction graph with the smallest number of vertices of length less than or equal to $4\log_2|V(G)|$. Next, the algorithm subtracts from the weight of each vertex in $V(\Gamma)$ the minimum among the weights of vertices in $V(\Gamma)$. The vertices whose weights become zero are added to the building feedback vertex set $V^{'}$ and deleted from the graph. These steps are iterated until the graph is exhausted. It has been proved that this algorithm can be applied even on planar graphs achieving a performance ratio less than or equal to 10. The second algorithm for the weighted feedback vertex set problem in its undirected version is based on the property that every feedback vertex set $V^{'}$ of a branchy graph G^* contains a number of vertices less than or equal to $2\Delta^2(G)\ |V^{'}|$. This is a greedy algorithm that achieves a performance ratio less than or equal to $2\Delta^2(G)$ for any graph G. It starts each iteration i by finding the minimal weighted reduction graph (H_i, w_{H_i}) of the graph $(H_{i-1}, w_{H_{i-1}})$, where $(H_0, w_{H_0}) = (G,w)$ and by computing α_i, the minimum weight among those associated with the vertices of H_i. It then proceeds by adding to the feedback vertex set $V^{'}$ all the vertices in H_i having weight α_i and removing them from H_i. These steps are iterated until the reduced graph H_i becomes a forest.

For the feedback vertex set problem in general undirected graphs, two slightly different 2-approximation algorithms are described in Becker and Geiger [3] and by Bafna, Berman, and Fujito [1]. These algorithms improve the approximation algorithms of Bar-Yeruda et al. [2], who also give a reduction procedure from the *loop cutset problem* to the minimum weighted feedback vertex set problem. The proposed algorithms also can find a loop cutset which, under specific conditions explained in more detail in Section 5, is guaranteed in the worst case to contain less than four times the number of variables contained in a minimum loop cutset. Subsequently, Becker and Geiger [4] applied the same reduction procedure from the loop cutset problem to the minimum weighted feedback vertex set problem of Bar-Yeruda et al. [2], but their result is independent of any condition and is guaranteed in the worst case to contain less than twice the number of variables contained in a minimum loop cutset. Becker and Geiger [4] propose two approximation algorithms for finding the minimum feedback vertex set $V^{'}$ in a vertex-weighted undirected graph (G,w). The simplest of them is a *greedy algorithm* that starts with a graph $G^{'}$, after removing from the original graph G all vertices $i \in V(G)$ with degree $\Delta_G(i)$ equal to 0 or to 1 and then, it repeatedly chooses to insert a vertex v in the feedback vertex set $V^{'}$, if the ratio between v's weight $w(v)$ and its degree $\Delta_{G'}(v)$ in the current reduced graph $G^{'}$ is minimal across all vertices in $G^{'}$. When v is selected, it is removed from $G^{'}$ along with all its incident edges together with all vertices of degree equal to 0 or to 1 after those removals. This step is iterated until the current reduced graph is exhausted. It is proved that the performance ratio of this algorithm is bounded by $2\log\Delta(G)+1$, where $\Delta(G) = \max_{v\in V(G)} \Delta_G(v)$ is the degree of graph G. The second algorithm is a modified greedy algorithm having performance ratio bounded by the constant 2. It consists of two phases:

1. The first phase proceeds as in the greedy algorithm, but now, when a vertex v is chosen to be removed from the graph with all its incident edges and to be inserted in the building feedback vertex set V', the ratio $r(v)$ between its weight and its degree is saved. After the removal from the current graph of all vertices having degree 0 or 1, along with their incident edges, for every removed edge (u_1,u_2) the ratio $r(v)$ is subtracted from the weight of its endpoints u_1 and u_2.
2. The second phase, on the other hand, is completely new. It starts after exhausting the current reduced graph, i.e. when a feedback vertex set $V^{'}$ is already found, and tries to remove from $V^{'}$ redundant vertices v_i by checking if every cycle in G that passes through v_i intersects with $V^{'} \setminus \{v_i\}$.

By using a Fibonacci heap, the complexity of the algorithm is $O(m+n\log n)$, where $m = |E(G)|$ and $n = |V(G)|$. A vertex is identified and retrieved from such a heap $|V(G)|$ times, with each operation taking $O(\log|V(G)|)$ time. The vertex weights are decreased $|E(G)|$ times at a constant amortized cost each. Moreover, the complexity of the second phase does not increase the total complexity of the algorithm.

In [18], Chudak, Goemans, Hochbaum, and Williamson showed how the algorithms due to Becker and Geiger [3] and Bafna, Berman, and Fujito [1] can be explained in terms of the primal-dual method for approximation algorithms that are used to obtain approximation algorithms for network design problems. The primal-dual method starts with an integer programming formulation of the problem under consideration. It then simultaneously builds a feasible integral solution and a feasible solution to the dual of the linear programming relaxation. If it can be shown that the value of these two solutions is within a factor of α, then an α-approximation algorithm is found. The *integrality gap* of an integer program is the worst-case ratio between the optimum value of the integer program and the optimum value of its linear relaxation. Therefore, by applying the primal-dual method it is possible to proof that the integrality gap of the integer program under consideration is bounded. In fact, Chudak et al., after giving a new integer programming formulation of the feedback vertex set problem, provided a proof that its integrality gap is at most 2. They also gave the proofs of some key inequalities needed to prove the performance guarantee of their new 2-approximation algorithm, which is a simplification of the algorithm due to Bafna et al. [1].

Theorem 3. *Let V' denote any feedback vertex set of a graph $G = (V,E)$, $E \neq \emptyset$, let τ denote the cardinality of the smallest feedback vertex set for G, and let $E(S)$ denote the subset of edges that have both endpoints in $S \subseteq V(G)$, $b(S) = |E(S)| - |S| + 1$. Then*

$$\sum_{v \in V'} [\Delta_G(v) - 1] \geq b(V(G)) \tag{1}$$

$$\sum_{v \in V'} \Delta_G(v) \geq b(V(G)) + \tau. \tag{2}$$

If every vertex in G has degree at least two, and V'_M is any minimal feedback vertex set (i.e. $\forall v \in V'_M$, $V'_M \setminus \{v\}$ is not a feedback vertex set), then

$$\sum_{v \in V'_M} \Delta_G(v) \leq 2\,(b(V(G)) + \tau) - 2. \tag{3}$$

Definition 10. *A feedback vertex set V' is almost minimal if there is at most one vertex $v \in V'$ such that $V' \setminus \{v\}$ is a feedback vertex set.*

Definition 11. *A cycle is semidisjoint if it contains at most one vertex of degree greater than 2.*

Theorem 4. *If every vertex has degree at least 2 and the graph does not contain semidisjoint cycles or is itself a cycle, and V'_{AM} is any minimal feedback vertex set, then*

$$\sum_{v \in V'_{AM}} [\Delta_G(v) - 1] \leq 2\, b(V(G)) - 1. \tag{4}$$

Note that the inequalities 2 and 3 imply

$$\sum_{v \in V'_M} \Delta_G(v) \leq 2 \sum_{v \in V'} \Delta_G(v) - 2, \tag{5}$$

while inequalities 1 and 4 imply

$$\sum_{v \in V'_{AM}} [\Delta_G(v) - 1] \leq 2 \sum_{v \in V'} [\Delta_G(v) - 1] - 1. \tag{6}$$

The condition under which the inequality 4 holds is satisfied if the graph is 2-vertex-connected. As Corollary of inequality 1, Chudak et al. proved the following result:

Theorem 5. *Let V' be any feedback vertex set. Then, for any $S \subseteq V(G)$ such that $E(S) \neq \emptyset$, we have that*

$$\sum_{v \in V' \cap S} [\Delta_S(v) - 1] \geq |E(S)| - |S| + 1 = b(S). \tag{7}$$

Inequalities 3 and 4 are needed to prove the performance guarantee of the proposed approximation algorithm, while 1 and 2 are used in the new integer programming formulation. The standard cycle formulation of the problem is given in Subsection 3.1. Even, Naor, Schieber, and Zosin [27] showed that the integrality gap of that integer program is $\Omega(\log n)$. The new integer programming formulation given by Chudak et al. [18] is as follows:

$$\textbf{(IP)} \begin{cases} \min \sum_{v \in V(G)} w(v)\, x_v \\ \text{s.t.} \sum_{v \in S} (\Delta_S(v) - 1)\, x_v \geq b(S) \quad S \subseteq V(G) : E(S) \neq \emptyset \\ x_v \in \{0,1\} \qquad v \in V(G). \end{cases}$$

The linear programming relaxation is:

$$\textbf{(LP)} \begin{cases} \min \sum_{v \in V(G)} w(v)\, x_v \\ \text{s.t.} \sum_{v \in S} (\Delta_S(v) - 1)\, x_v \geq b(S) \quad S \subseteq V(G) : E(S) \neq \emptyset \\ x_v \geq 0 \qquad v \in V(G). \end{cases}$$

and its dual is:

$$\textbf{(D)} \begin{cases} \max \sum_{S} b(S)\, y_S \\ \text{s.t.} \sum_{S : v \in S} (\Delta_S(v) - 1)\, y_S \leq w_v \quad v \in V(G) \\ y_S \geq 0 \qquad S \subseteq V(G) : E(S) \neq \emptyset. \end{cases}$$

The algorithm proposed by Chudak at al. constructs a feasible solution of the linear programming relaxation (LP) and it is essentially the algorithm proposed by Bafna et al. In [1], Bafna et al. developed an algorithm that starts with a feedback vertex set $V' = \emptyset$, the feasible dual solution $y = 0$, and the original graph $G^* = (V^*, E^*) = (V, E)$. Given a set V', if it is not a feedback vertex set, there must exist a cycle in G^*. The algorithm first recursively removes from G^* all vertices having degree equal to one and their incident edges and then chooses some set S corresponding to a violated constraint of (IP). S is called the *violated set*. To choose S the algorithm uses a subroutine called VIOLATION, that first looks for a semidisjoint cycle in G^*. If it finds such a cycle, it lets S correspond to the vertices of the cycle and returns S. Otherwise, it returns $S = V^*$. The algorithm then increases as much as possible the dual variable y_S, until some dual inequality becomes tight for some vertex in S, that is added to V' and removed from G^* together with its incident edges. The algorithm continues repeating these steps until V' is a feedback vertex set. It then goes through the vertices in V' in the reverse of the order in which they were added and removes any extraneous vertices. The simpler 2-approximation algorithm of Chudak et al. uses a different VIOLATION subroutine, that avoids searching for semidisjoint cycles by a different choice of S. Given a decomposition of G^* into its 2-vertex-connected components, their algorithm sets S to be an endblock, so that S contains at most one cutvertex.

Another 2-approximation algorithm is due to Vazirani [86], who also found a lower bound on the optimal solution for special vertex weight functions. Before describing his algorithm, the following definitions are needed.

Definition 12. *Let $GF[2]^i$ be the set of all binary vectors having i components. Then the characteristic vector of a cycle C in G is a vector in $GF[2]^m$, $m = |E(G)|$, whose components are equal to 1, if they correspond to edges in C, 0 otherwise.*

Definition 13. *The cycle space of G is defined as the subspace of $GF[2]^m$, spanned by the characteristic vectors of all cycles in G.*

Definition 14. *The cyclomatic number $cyc(G)$ is the dimension of the cycle space and it is given by*

$$cyc(G) = |E(G)| - |V(G)| + k(G),$$

where $k(G)$ denotes the number of connected components of G.

The removal of an edge i from G decreases its cyclomatic number by 1, unless i is a *bridge*, i.e. an edge whose removal increases the number of connected components. Similarly, the decrease in the cyclomatic number caused by removing a vertex v equals the maximum number of edges incident to v that can be successively removed so that none of them is a bridge at the moment of its removal.

Definition 15. *Let $\delta_G(v)$ be the decrease in cyclomatic number caused by removing the vertex v from G. A weight function is cyclomatic if it assigns to each vertex v the weight $c \cdot \delta_G(v)$, for some $c > 0$.*

Since the removal of a feedback vertex set $V^{'} = \{v_1, \cdots, v_f\}$ decreases the cyclomatic number of G down to 0, then

$$cyc(G) = \sum_{i=1}^{f} \delta_{G_{i-1}} v(i),$$

where G_0 is the original graph and, for $i > 0$, $G_i = G \setminus \{v_1, \cdots, v_i\}$.
Since a non-bridge edge in an induced subgraph cannot be a bridge in the larger graph, for each vertex v the following inequalities hold

$$\delta_{G_i}(v) \leq \delta_G(v),$$

$$cyc(G) \leq \sum_{v \in V'} \delta_G(v).$$

Therefore, if the weight function defined on the vertices of G is cyclomatic, then $c \cdot cyc(G)$ is a lower bound on the optimal solution and a straightforward factor 2 algorithm for cyclomatic weights functions is given by the following lemmas:

Lemma 1. *If $V^{'}$ is a minimal feedback vertex set of G, then*

$$\sum_{v \in V'} \delta_G(v) \leq 2 \cdot cyc(G).$$

Lemma 2. *Let w be a cyclomatic weight function defined on $V(G)$ and let $V^{'}$ be a minimum feedback vertex set of G. Then*

$$w(V^{'}) \leq \mu(G,w).$$

In [86], Vazirani solves the feedback vertex set problem also in the case of arbitrary weights. The proposed 2-approximation algorithm consists of two phases: a *decomposition* phase and an *extension* phase. During the decomposition phase, the algorithm decomposes the original graph G into a sequence of induced subgraphs, $G_k \subset G_{k-1} \subset \cdots \subset G_0 = G$, where G_k is acyclic. On the vertices of each graph G_i, $i < k$, a weight function is defined such that the sum of the weights of a vertex among these subgraphs is equal to its original weight, i.e.

$$\sum_{i:v\in V(G_i)} w_i(v) = w(v).$$

When the decomposition phase terminates, ${V'}_k$ is a minimal feedback vertex set of G_k. The algorithm proceeds then executing the extension phase. For each G_i, $i < k$, it computes a minimum feedback vertex set ${V'}_i$ by adding to ${V'}_{i+1}$ a minimal set of vertices from $V(G_i) - V(G_{i-1})$. The set ${V'}_0$ is output as a feedback vertex set for G.

Regarding the maximum size of a feedback vertex set V' in cubic graphs, Speckenmeyer [81] proved that for a connected undirected cubic graph G of order n and girth g (*girth* is the length of a shortest cycle in G)

$$|V'| \le \frac{(g+1)}{(4g-2)} + \frac{(g-1)}{(2g-1)}.$$

Bondy, Hopkins, and Staton [5] proved that for a connected cubic graph G of girth at least 4

$$|V'| \le \lceil 1/3\ |V(G)| \rceil.$$

An improvement was made by Zheng and Lu, who in [92] found that if the girth of G is at least 4 and if $|V(G)| \ne 8$, then

$$|V'| \le \lfloor 1/3\ |V(G)| \rfloor$$

and the bound is sharp. These results were improved by Liu and Zhao [58], including the result due to Zheng and Lu, resulting in

$$|V'| \le \frac{g}{4(g-1)}\, n + \frac{(g-3)}{(2g-2)}$$

for a large class of cubic graphs of order $n \ge 4$ and girth g.

For the subset feedback vertex problem, Even, Naor, Schieber, and Zosin [27] showed that it can be approximated in polynomial time by a factor of $\min\{2\ \Delta(G), 8\ \log(|V'|+1), O(\log\tau^*)\}$, where τ^* denotes the value of the optimal fractional solution. In [27] the authors also proposed a technique, called *bootstrapping*, that enhances the $O(\log|V'|)$ to a factor of $O(\log\tau^*/\beta)$, where β denotes the minimum weight of a vertex. The bootstrapping technique iteratively uses a graph partitioning algorithm. The output of each iteration is by itself a subset feedback vertex set and is used as part of the input of the next iteration. After $O(\log|V'|)$ iterations, the algorithm produces as output a subset feedback vertex set having weight at most $O(\tau^*\log\tau^*)$. Even, Nor, and Zosin [29] improved this result proposing a 8-approximation algorithm. The main tool that they used in developing their approximation algorithm and its analysis is a new version of multicommodity flow, called *relaxed multicommodity flow*, a hybrid of multicommodity flow and multiterminal flow. In multicommodity flow, the arc capacity constraints apply to the total flow of all the commodities, while in multiterminal flow, the arc capacity constraints apply to each commodity separately. A relaxed multicommodity flow is a multiterminal flow with additional constraints, called *intercommodity* constraints. For each arc, the authors considered the maximum flow, among all the commodities, which is shipped along it. They required that for each vertex $v \in V(G)$ the sum of the maximum flows shipped along its incident arcs is bounded by four times the capacity of v. By considering the multicommodity flow, the vertices for which the intercommodity constraints are tight play an important role from the point of view of the connectivity of the graph. They are called *intersatured* vertices. The main result of

Even et al. is a theorem that bounds the weight of the vertices that must be intersatured, so as to realize a given demand vector by the sum of demands.

3.3.2. *Directed graphs.* In general, problems on undirected graphs are relatively easier to handle than problems on directed graphs, since more graph theory can be utilized. Not surprisingly, the approximation results obtained so far for the undirected version are stronger than those for the directed version. In fact, none of the algorithms referred to in the previous subsections apply to the feedback vertex set problem in directed graphs and, in contrast with the undirected version, no analytical results are known for the directed case.

A very recent direction of research on approximation algorithms in the directed version focuses on the complete equivalence among all feedback set (and/or feedback subset) problems and among these and the directed minimum capacity multicut problem in circular networks. An exhaustive description of the procedures that reduce any feedback set problem to any other or any of them to the directed minimum capacity multicut problem and vice versa will be given in Subsection 4.2. These reduction procedures are formalized and used by Even, Naor, Schieber, and Sudan [28] to obtain an approximation algorithm for the subset feedback arc set problem of a weighted directed graph $G = (V, E)$, where the interesting cycles to be *hit* are contained in a set of special vertices $X \subseteq V(G)$, where $|X| = k$. The weight of the feedback arc set found by their approximation algorithm is $O(\tau^* \log^2 |X|)$, where τ^* is the weight of an optimal fractional feedback set. More detail is provided in Subsection 4.3. Nevertheless, their approach can be used to solve any other feedback set problem as well as the directed minimum capacity multicut problem.

Even et al. [28] also proposed an algorithm for approximating the minimum weighted subset vertex set problem in the weighted and directed case, leading to a result that holds for any other feedback set problem as well. Their approach is an algorithmic adaptation of a theoretical result due to Seymour [75], who proved that the integrality gap in the case of the unweighted feedback vertex set problem can be at most $O(\log \tau^* \log\log \tau^*)$, where τ^* is defined as above. Even et al. observe that all existence arguments contained in the proof of Seymour's statement can be made constructive and thus, with some additional operations, an algorithm for the unweighted feedback vertex set problem having an approximation factor of $O(\log \tau^* \log\log \tau^*)$ can be obtained. Further modifications of the algorithm lead to a polynomial time approximation scheme applicable to the weighted problem. In $O(|E(G)| \cdot |V(G)|^2)$ time the algorithm finds a feedback vertex set having weight $O(\min\{\tau^* \log \tau^* \log\log \tau^*,\ \tau^* \log |V(G)| \log\log |V(G)|\})$. All the observations contained in [28] improve the $O(\log^2 |V(G)|)$-approximation algorithm for this case due to Leighton and Rao [53].

In the case of directed planar graphs, Stamm [83] presented an $O(|V(G)| \log |V(G)|)$-approximation algorithm, whose performance guarantee is bounded by the maximum degree of the graph and an $O(|V(G)|^2)$ time approximation algorithm with performance guarantee no more than the number of cyclic faces in the planar embedding of the graph minus 1.

Cai, Deng, and Zang [10] obtained a 2.5-approximation algorithm for the minimum feedback vertex set problem on tournaments, improving the previously known algorithm with performance guarantee of 3 by Speckenmeyer [82]. Let H be the triangle-vertex incidence matrix of a tournament T and let e be the all-one vector. In [10], necessary and sufficient conditions are established for the linear system $\{x \mid Hx \geq e,\ x \geq 0\}$ to be a *totally dual integral system* (TDI).

Definition 16. *A rational linear system* $\{x \mid Hx \geq e,\ x \geq 0\}$ *is called totally dual integral, if the optimization problem* $\max\{y^T b \mid y^T A \leq c^T,\ y \geq 0\}$ *has an integral optimum solution* y *for every integral vector* c *for which the maximum is finite.*

It has been shown that any rational polyhedron P has a TDI System $P = \{x : \ Ax \leq b\}$ representation with A integral, and that, if P is full-dimension, there is a unique minimal TDI System $P = \{x : Ax \leq b\}$ with A and b integral if and only if P is integral. In [11] the authors have extended this approach to the feedback vertex set problems and the cycle packing problem in *bipartite tournaments*, where a bipartite tournament is an orientation of a complete bipartite graph. For the aforementioned problems they have found strongly polynomial time algorithms, which are a consequence of a min-max relaxation on packing and covering directed cycles.

3.4. **Exact algorithms.** In contrast to the numerous approximation schemes that have been studied, relatively few exact algorithms for the feedback vertex set problem have been proposed. To our knowledge, the first algorithm to find an exact minimal cardinality FVS is due to Smith and Walford [80]. Although it solves the problem in an arbitrary directed graph in exponential running time, it returns an optimal solution in polynomial time for certain types of graphs. Before giving its description, the following definitions are needed.

Definition 17. *A maximal strongly connected component of a graph is abbreviated as MSC.*

Definition 18. *A set of vertices S of graph G with the property that each loop of G contains at least one element of S is called a complete set. The number of elements in the set S is referred to as the set measure* $|S|$.

Definition 19. *A complete set S of minimal size is called an optimum set and its measure is defined as the index of G denoted by* $I(G)$.

The algorithm proposed by Smith and Walford is based on the following idea of partition implication. Given a graph G, an arbitrary subset of vertices F may imply a 2-subgraph partition (G_F, G_R) such that

1. Each vertex of G_F is contained in at least one loop in G and only in loops of G also containing an element of F. G_R contains all remaining vertices of G;
2. If G is strongly connected, then all elements of any possible set F will be in G_F;
3. By removing from G all vertices in $G_F \cap F$ all loops in G containing a vertex of G_F are eliminated from from G_F;
4. Loops in G_F and loops in G_R are independent, i.e. they do not contain any common vertex.

The authors proved the following results used by their algorithm.

Theorem 6. *If* $S_1, S_2, \cdots, S_k$ *are optimal feedback vertex set for partitions* $P_1, P_2, \cdots, P_k$ *of graph G, and if the set S given by* $S = \cup_{i=1}^{k} S_i$ *is complete for G, it is also optimal for G.*

Theorem 7. *If* $I(G_F) = |F|$*, then F is a subset of an optimal feedback vertex set of G.*

Theorem 8. *Each element of an optimum set S must belong to at least one loop of G not containing any other element of S.*

The running time of the Smith-Walford algorithm reported in the following depends on the cardinality of the partitions of the graph.

INPUT: a digraph $G(V,E)$;
OUTPUT: a minimum feedback vertex set for G.
1) Find the MSC subgraphs of the graph under test, store them on push down stack;

2) Pop top MSC from push down stack. If the stack was empty, then exit;
3) Generate next trial F set for MSC;
4) Calculate implied partitions G_F and G_R;
5) If $I(G_F) = |F|$, then go to 3);
6) Make F part of minimal feedback vertex set;
7) Break G_R subset into its MSC subgraphs;
8) Store them on the push down stack and go to 2).

Later exact algorithms of enumerative nature often used the graph reduction procedures to speed up the process. One study by Cheng [16] essentially used direct enumeration plus reduction and reported satisfactory computational results for a set of partial scan design test problems. Orenstein, Kohavi, and Pomeranz [65] proposed a somewhat more involved exact enumerative procedure based on graph reduction and efficient graph partitioning methods. In addition to the reduction procedures due to Levy et al., Orenstein et al. developed a further type of operation designed *double reduction*, which involves edges deletion:

DOUBLE(v,w):: If the vertices v and w form a 2-edged directed cycle in the graph G, then except for (v,w) and (w,v), remove from G all edges included in directed cycles going through both v and w.

If v and w satisfy the hypotheses of the DOUBLE reduction, then at least one of them must be included in the feedback vertex set and all cycles passing through both v and w will be covered. Therefore, taking into account only the smallest cycle involving v and w will have no influence on the optimum feedback vertex set solution. The algorithm proposed by Orenstein et al. has been designed for identifying a minimum feedback vertex set in a digital circuit. It is based on the following recursive steps:

1. Building the *topological* graph associated with the input sequential circuit;

   Definition 20. *A digraph $G = (V,E)$ is called a topological graph for a circuit S if each vertex in G represents a flip-flop, and a directed edge (u,v) exists if and only if there is a path from the flip-flop u to the flip-flop v through combinatorial logic. A topological graph does not contain parallel edges.*

2. Reduction operations;
3. Partition: this step is executed after no more reduction operations can be applied. The graph G is partitioned into subgraphs, which are solved separately and recursively by using this 3-step procedure.
   The authors proposed four different partition techniques:
   (a) The partition into MSC, already proposed in [80];
   (b) The partition into two independent subgraphs. The *locality* of the circuit is such that the circuit can be divided into small subcircuits with limited communication among them;
   (c) An *extended* partition which uses *loop* locality. A digital circuit generally contains many non-nested loops. This property causes the generation of two-edged loops after reduction procedures are performed on the corresponding topological graph;
   (d) Vertex elimination following a Branch-and-Bound procedure is applied when no other partition are possible.

   All partitions above described can be performed in polynomial time by using variations of depth-first search and/or breadth-first search.
4. Merging: in this step the solutions computed in step 2 are merged.

The algorithm proposed by Orenstein et al. is efficient in random graphs, even if in cliques or graphs that are *almost* cliques it has an exponential behavior, since the reduction and partition techniques cannot be applied.
Somewhat surprising, the exact algorithm for feedback vertex set based on mathematical programming formulation is quite few. Recently, Funke and Reinelt [31] considered a special variant of feedback problems, namely the problem of finding a maximum weight node induced acyclic subdigraph. They discussed valid and facet defining inequalities for the associated polytope and developed a polyhedral-based exact algorithm presenting computational results obtained by applying a branch-and-cut algorithm.

3.5. **Practical heuristics.** Although the approximation algorithms guarantee a solution of a certain quality, for many practical real world cases, heuristic methods can lead to better solutions in a reasonable amount of CPU time. As Grötschel and Lovász [38] pointed out, fast construction heuristics combined with local improvement techniques tailored for special applications have been the "workhorse" of combinatorial optimization in practice. Over the years, a number of metaheuristics, including genetic algorithms, simulated annealing, greedy randomized adaptive search procedures (GRASP), Lagrangean relaxation, and others were developed with successful computational performance on a wide range of combinatorial optimization problems. Interestingly, however, feedback vertex set problems seem to be an exception. So far, relatively few practical heuristics have been developed for this family of problems, even fewer have reported computational results. Furthermore, most of the heuristics that seem to be quite successful computationally are greedy type heuristics or generalized greedy type heuristics (e.g. GRASP).

Almost all the efficient heuristics developed so far employ the solution-preserved reduction rules studied by Levy and Lowe [55], which were presented in Subsections 3.2 and 3.3.1. It has been observed in practice that this group of heuristics greatly reduces the cardinality of the graph not only at the beginning of the algorithm, but also dynamically during the execution of node deletion type heuristics. A recent line of research on heuristic approaches is due to Pardalos, Qian, and Resende [68] where three variants of the so called *Greedy Randomized Adaptive Search Procedure* (GRASP) metaheuristic are proposed for finding approximate solutions of large instances of the feedback vertex set problem in a digraph. GRASP is a multistart method characterized by two phases: a construction phase and a local search phase, also known as a local improvement phase. During the construction phase a feasible solution is iteratively constructed. One element at time is randomly chosen from a *Restricted Candidate List* (RCL), whose elements are sorted according to some greedy criterion, and is added to the building feedback vertex set and removed from the graph with all its incident arcs. Since the computed solution, in general, may not be locally optimal with respect to the adopted neighborhood definition, the local search phase tries to improve it. These two phases are iterated and the best solution found is kept as an approximation of the optimal solution.

To improve the efficiency of the method, Pardalos et al. incorporated in each iteration of their algorithm solution-preserving graph reduction techniques in their directed version and that can be used also to check if a digraph is acyclic, returning an empty reduced graph in case of positive answer.

The authors employed the following three greedy functions used to select the node with the maximum $G(i)$ values:

1. $G_A(i) = in(i) + out(i)$;
2. $G_B(i) = in(i) * out(i)$;
3. $G_C(i) = \max\{in(i), out(i)\}$.

Greedy function G_A assigns equal weight to in- and out-degrees. G_B favors the balance between in- and out-degrees. G_C only considers the largest value of the degrees. As demonstrated in Pardalos et al., G_B produced the best computational results. GRASP was tested on two randomly generated problem sets, finding the optimal solutions to all the problems in the first set, where the optimal values are known (computed by Funke and Reinelt [31]). Furthermore, this GRASP dominates the pure greedy heuristics in all the test instances with comparable running time. Fortran subroutines for finding approximate solutions of the directed feedback vertex set problem suing GRASP are given in Festa, Pardalos, and Resende [30].

4. THE FEEDBACK ARC SET PROBLEM

Several versions of the feedback vertex set problem have been discussed in Section 3. It is possible to consider their *arc* counterparts. More specifically, given a graph $G = (V,E)$ and a nonnegative weight function $w:\ E(G) \to R^+$ defined on the arcs of G, find a minimum-weight subset of arcs $E' \subseteq E(G)$ that meets every cycle in a given collection C of cycles in (G,w). As in the vertex case, this leads to the *minimum feedback arc set problem* (MWFAS) in both directed and undirected graphs, the *minimum weighted graph bipartization problem* via arc removals, and so on.

4.1. Mathematical model of the feedback arc set problem.

Given an arc weighted graph (G,w), $G = (V,E)$ and the set C of all cycles in G, the minimum weighted feedback arc set problem can be formulated as the following integer programming problem:

$$\textbf{(MFAS)} \begin{cases} \min \displaystyle\sum_{e \in E(G)} w(e)\, x_e \\ \text{s.t.} \displaystyle\sum_{e \in \Gamma} x_e \geq 1 \quad \forall\, \Gamma \in C \\ x_e \in \{0,1\} \quad \forall\, e \in E(G). \end{cases}$$

In its relaxation, the constraints $x_e \in \{0,1\}, \forall\, e \in E(G)$ are replaced by $x_e \geq 0, \forall\, e \in E(G)$, obtaining a fractional feedback arc set.

As with the feedback vertex set problem, the feedback arc set problem is a *covering* problem and its (linear programming) dual is called a *packing* problem. In the case of the feedback arc set problem this means assigning a dual variable to all interesting cycles to be hit in the given graph, such that for each arc the sum of the variables corresponding to the interesting cycles passing through that arc is at most the weight of the arc itself.

4.2. Relation between the feedback vertex set and the feedback arc set problems.

Feedback arc set problems tend to be easier than their vertex counterparts, especially for planar graphs. In the directed case feedback vertex and feedback arc set problems are each reducible to one another. In all reductions, there is a one-to-one correspondence between feasible solutions and their corresponding costs. Therefore, an approximate solution to one problem can be translated to an approximate solution of the other problem reducible to this problem. Because most of the reduction procedures can be performed in linear time, these problems can be viewed as different representations of the same problem. Hence, as feedback vertex sets are reduced into feedback arc sets with the same weight and vice versa, all of these problems are equally hard to approximate.

Even, Naor, Schieber, and Sudan [28] showed how to perform reductions among feedback set problems and feedback subset problems and vice versa, preserving feasible solutions and their costs. In the following, we report some of these reduction procedures, where FVS and FAS denote the feedback vertex set problem and the feedback arc set problem, respectively, and BLACKOUT-FVS denotes an extension of the general feedback vertex set problem already referred to in the Subsection 3.3.1 (see also [2]). Recall that in the blackout feedback vertex set problem an additional subset of *blackout* vertices $B \subseteq V(G)$ is given and that the objective is to compute a minimum feedback vertex set that does not contain vertices of B.

[FAS $\Rightarrow$ FVS]: Given the graph $G = (V,E)$, construct its *directed line-graph* $G' = (V', E')$ as follows:

1. Set $V'(G') = E(G)$;
2. An arc in $E'(G')$ connects the vertices $(v_1 \to v_2) \in V'(G')$ and $(v_3 \to v_4) \in V'(G')$ if and only if $v_2 = v_3$.

In the weighted version, a correspondence among the weights of the arcs of G and the weights of the "vertices" of the corresponding graph G' is required as follows:

3. The weight of the "vertex" $(v_1 \to v_2) \in V'(G')$ is equal to the weight of the arc $(v_1 \to v_2) \in E(G)$.

A subset of arcs $F \subseteq E(G)$ is a feedback arc set for G if and only if it is a feedback vertex set for G'.

[FVS $\Rightarrow$ FAS]: Given the graph $G = (V,E)$, construct a graph $G' = (V', E')$ as follows:

1. For every $v \in V(G)$ insert in $V'(G')$ two vertices v_1 and v_2;
2. For every v_1 and v_2 inserted in $V'(G')$ after splitting a vertex $v \in V(G)$ insert in $E'(G')$ an arc $(v_1 \to v_2)$, all the arcs that enter v in G connecting them to v_1 in G', and all the arcs that emanate from v in G emanating them from v_2 in G'.

In the case of weights we also have:

3. For every arc $e' = [(v_1 \to v_2) \mid v \in V(G)]$, set $w(e') = w(v)$. All other arcs in $E'(G')$ have infinite weight.

This establishes a one-to-one correspondence between the finite weighted feedback arc sets of the new graph G' and the feedback vertex sets of the original graph G.

[BLACKOUT-FVS $\Rightarrow$ FVS]: A very simple reduction procedure could consist of assigning infinite weight to each blackout vertex. However, in some cases a more formal reduction among these two similar problems could be needed. What can be done is to bypass each blackout vertex in B. For each blackout vertex $v \in B$:

1. Connect arcs between every two vertices that have a path of length two, connecting them, where v is the middle vertex;
2. Remove the blackout vertex v from the graph.

A subset of vertices $F \subseteq V(G)$ is a feedback vertex set that does not contain any blackout vertices from B if and only if it is a feedback vertex set of the graph obtained by the reduction.

Note that only the reduction **[BLACKOUT-FVS $\Rightarrow$ FVS]** requires more than linear time to be performed.

4.3. **State of the art of feedback arc set problems.** In the literature of feedback set problems most of the proposed algorithms are designed to solve the problem in vertex-weighted graphs. One of the pioneering papers on feedback arc set problems is due to to Ramachandran [73], where it is proved that finding a minimum feedback arc set in an arc-weighted reducible flow graph is as difficult as finding a minimum cut in a flow network. The proposed algorithm has complexity $O(m\, n^2 \log(\frac{n^2}{m}))$, where $m = |E(G)|$ and $n = |V(G)|$. The algorithm was adapted to solve the problem in the vertex-weighted case. Shamir's linear time algorithm [76], used for the unit-weighted case, cannot be applied to solve the arc-weighted problem, because any reduction between arc and vertex set problems does not preserve the reducibility property.

Given a directed graph $G = (V,E)$, a *dijoin* $E' \subseteq E(G)$ is a set of arcs such that the graph $G' = (V,B)$, $B = E \cup \{(v,u) \mid (u,v) \in E'\}$ is strongly connected. Given nonnegative weights w_e, $e \in E(G)$, the minimum-weight dijoin problem is to find the dijoin with minimum weight. The feedback arc set problem in planar digraphs is reducible to the problem of finding a minimum-weight dijoin in the dual graph, which is solvable in polynomial time [37]. Stamm [83] proposed a simple 2-approximation algorithm for the minimum weight dijoin problem by superposing two arborescences. It is interesting to observe that, when translated to the dual graph, all these problems lead to problems of hitting certain cutsets of the dual graph, problems which can be approximated within a ratio of 2 by the primal-dual method. Goemans and Williamson [36] proposed a primal-dual algorithm that finds a $\frac{9}{4}$-approximate solution to feedback sets problems in planar graphs.

The first approximation algorithm for the feedback arc set problem was given by Leighton and Rao [53]. Their approximation factor is $O(\log^2 n)$ in the unweighted case, where n is the number of vertices of the input graph. This bound was obtained by using a $O(\log n)$ approximation algorithm for a directed separator that splits the graph into two approximately equally-sized components. This separator can be found by approximating special cuts called *quotient cuts*. This result was improved by Seymour [75], who gave a $O(\log n \log\log n)$-approximation algorithm that solves the linear relaxation of the feedback arc set mathematical model and then interprets the optimal fractional solution x^* as a length function defined on the arcs. Systematically, in a recursive fashion, it uses this length function to delete from the graph G all arcs between S and $\bar{S}$. Note that the linear program can be solved in polynomial time by using the ellipsoid or an interior point algorithm. Hence, the quality of the bound in this approach depends on the way the graph is partitioned. Seymour in [75] proved the following lemma:

Lemma 3. *For a given strongly connected digraph $G = (V,E)$, suppose there exists a feasible solution x to the feedback arc set problem. If ϕ is the value of the optimal fractional solution x^*, then there exists a partition $(S,\bar{S})$ such that, for some ε, $0 < \varepsilon < 1$, the following conditions hold: If $\delta^+(S) = \{(u,v) \mid (u,v) \in E(G),\ u \in S,\ v \in \bar{S}\}$ and $\delta^-(S) = \{(v,u) \mid (v,u) \in E(G),\ u \in S,\ v \in \bar{S}\}$, then*

$$\sum_{e \in E(S)} w(e)\, x(e) \leq \varepsilon\phi \tag{8}$$

$$\sum_{e \in E(\bar{S})} w(e)\, x(e) \leq (1-\varepsilon)\phi \tag{9}$$

and either

$$\sum_{e \in \delta^+(S)} w(e) \leq 20\varepsilon\phi \log(\frac{1}{\varepsilon}) \log\log\phi \tag{10}$$

or

$$\sum_{e \in \delta^-(S)} w(e) \leq 20\varepsilon\phi \log(\frac{1}{\varepsilon}) \log\log\phi. \tag{11}$$

Furthermore, the partition $(S,\bar{S})$ can be found in polynomial time.

The previous lemma admits a constructive proof as has been shown by Even, Naor, Schieben, and Sudan [28]. The algorithm proposed by Even et al. finds a feedback arc set having weight $O(\tau^* \log^2 |X|)$, where X is a special set of vertices defining the cycles to be hit and τ^* is the weight of an optimal fractional feedback set. They had the idea of reducing the problem to the directed minimum capacity multicut problem in circular networks and of adapting the undirected *sphere growing* technique described in [34] to directed circular networks. They decomposed the graph in the following way. A fractional and optimal solution to the directed feedback set problem induces a distance metric on the set of arcs (on the set of vertices) $E(G)$. Their approximation algorithm arbitrarily picks a vertex $v \in X$ and solves the shortest path tree problem rooted at v with respect to the metric induced by the fractional solution. The procedure that finds the shortest path tree defines layers with respect to the source v. Each layer is a directed cut that partitions the graph into two parts. The next step of the approximation algorithm is to choose a directed cut and to add the cut to the feedback set constructed so far. The algorithm continues recursively in each part and ends when the graph does not contain any interesting cycles. The key of the algorithm is the choice of the criterion to select the directed cut that partitions the graph. Even et al. decided to relate the weight of the cut to the cost of the fractional solution.

More recently, Even, Naor, Schieber, and Zosin [27] showed that, for any weight function defined on the arcs, the subset feedback arc set problem can be approximated in polynomial time by a factor of two. The approximation algorithm consists of successive computations of minimum cuts. Its approximation factor is estimated by considering the capacities of minimum cuts as flow paths. When new minimum cuts are computed, previous flow paths are updated according to the decomposition of the graph induced by an optimal solution.

5. Applications

Feedback set problems were originally formulated in the area of combinatorial circuit design, where cycles can potentially cause a problem called a "racing condition", where some circuit node may receive new inputs before it stabilizes [43]. To avoid such a condition, a (clocked) register is placed at each cycle in the circuit. However, the delay in the circuit speed is proportional to the number of registers placed along a path. Therefore, the objective is to minimize the number of nodes (registers) to be placed so that the total delay can be minimized.

Another application of the feedback set problem is in deadlock prevention in computer systems [59, 78]. Consider an operating system which schedules different processes with requests on different resources, which need to use exclusively before being released by the process. A directed graph modeling these resource requirements is to have a node i for each process i and a directed arc $e(i, j)$ denotes process i requests a resource already allocated to process j. Therefore, if there is a directed cycle in such a graph, a deadlock occurs and every process in the cycle will wait for the requested resource and never releases the resources already allocated to it. To break such cycles, one can remove some processes from the graph and put them in a waiting queue. It is clear that the objective is to minimize the number of removed processes.

In recent years, there have been intensive research efforts in the VLSI testing community on the feedback vertex set problem due to the following application [9, 49, 52]. A circuit can be modeled by a directed graph whose vertices represent gates computing Boolean functions and the directed arcs represent wires that connect gates. In this case, finding a minimum feedback arc set helps to reduce the hardware overhead required for testing the circuit by using scan-design techniques. One problem with this architecture is the cost of additional hardware. To reduce this overhead, partial scan methods have been proposed, which only scan a subset of flip-flops in the tested circuit. This way, the hardware overhead of the scan-design can be significantly reduced.

Luccio [61] describes applications of the feedback vertex set problem in synchronous systems [69, 70]. Typically, a synchronous system can be modeled by a network, whose vertices can be colored white or black and at each step a vertex changes its color according to the majority of its neighbors implying that it is receiving the correct information. In a "monotone" synchronous system only white vertices can change their color. It can be easily seen that in a toroidal mesh a feedback vertex set of black vertices causes all vertices to become black after some steps and that a minimum feedback vertex set is an initial configuration of minimum cardinality that stabilizes the system.

Two notable applications of the unweighted feedback vertex set problem in artificial intelligence are the *constraint satisfaction problem* and *Bayesian inference*. The application in the *constraint satisfaction problem* is due to Dechter [23] and Dechter and Pearl [22]. A set of variable $\{x_1, x_2, \cdots, x_n\}$ is given, where each variable x_i belongs to a finite domain D_i. For every $i < j$ a constraint subset $R_{ij} \subseteq D_i \times D_j$ is constructed, defining pairs of values allowable for the pair of variables (x_i, x_j). The objective is to find an n-tuple of values $(v_1, v_2, \cdots, v_n)$ to be assigned to $\{x_1, x_2, \cdots, x_n\}$, such that all the constraints R_{ij} are satisfied. With each instance of the problem an undirected graph G can be associated whose vertices are the variables $\{x_1, x_2, \cdots, x_n\}$ and whose arc set contains the arc (x_i, x_j) if and only if $R_{ij} \subset D_i \times D_j$. G is called a *constraint network* and represents a *constraint satisfaction problem*. It can be easily solved by applying a backtracking method, that repeatedly assigns values to the variables in an order previously defined and backtracks whenever reaching a dead end. This exponential method can be improved, leading to another exponential technique, but in the size of a feedback vertex set of the constraint network. It first finds a feedback vertex set of the constraint network and then arranges the variables so that variables in the feedback vertex set precede all other variables. The backtracking procedure is applied, and as soon as values of the variables in the feedback vertex set are determined, a polynomial time algorithm solves the constraint satisfaction problem in the remaining forest. In case of success a solution is found. Otherwise, a new backtracking phase occurs.

The application of the feedback vertex set problem to Bayesian inference is due to Bar-Yeruda, Geiger, Naor, and Roth [2]. They reduced the weighted loop cutset problem to the weighted blackout-feedback vertex set problem in a directed graph G and gave a $2\Delta^2(G)$-approximation algorithm for solving any of these two problems, where $\Delta(G)$ is the degree of G. This algorithm reduced the computational complexity of Bayesian inference. Given a probability distribution $P(u_1, u_2, \cdots, u_n)$, where u_i belongs to a finite domain D_i, a directed acyclic graph G is called a *Bayesian network of P* if there is a one-to-one correspondence between $\{u_1, u_2, \cdots, u_n\}$ and $V(G)$ such that each u_i is associated with the vertex i for which the following holds:

$$P(u_1, u_2, \cdots, u_n) = \prod_{i=1}^{n} P(u_i | u_{i_1}, u_{i_2}, \cdots, u_{i_{j_i}}),$$

where $i_1, i_2, \cdots, i_{j_i}$ are the tail vertices of the edges in G whose head is i. The *updating problem* is that of finding for each value of i the probability $P(u_i | (v_1 = \mu_1), (v_2 = \mu_2), \cdots, (v_l = \mu_l))$ given that the values $\{\mu_1, \mu_2, \cdots, \mu_l\}$ are assigned to some variables $\{v_1, v_2, \cdots, v_l\}$ among $\{u_1, u_2, \cdots, u_n\}$. Pearl [71] solved this problem as follows. A *trail* t in a Bayesian network G is a subgraph whose underlying graph is a simple path. A vertex b is a sink with respect to a trail t if there exist two consecutive arcs $(a \rightarrow b)$ and $(b \rightarrow c)$ on t. The arc t is said to be "active" by a set of vertices Z if every sink with respect to t either belongs to Z or has a descendant in Z, and every other vertex of t does not belong to Z. Pearl's algorithm first selects a set of vertices S such that any pairs of vertices in G are connected by at most one active trail $t \in S \cup T$, where Z is any subset of vertices. Then, for each combination of value assignments to the variables belonging to S, a procedure given in [50] is applied. This solves the updating problem by viewing each vertex as a processor repeatedly sending messages to each of its adjacent vertices. When equilibrium is reached, each vertex i contains the conditional probability distribution $P(u_i|(v_1 = \mu_1),(v_2 = \mu_2),\cdots,(v_l = \mu_l))$. At the end all the obtained results are combined. This technique, called the *conditioning method*, is exponential in the size of S. Bar-Yeruda et al. showed that S is a loop cutset of the Bayesian network and hence is computable by applying their algorithms.

The approximation algorithms for the weighted feedback vertex set problem have applications in areas of computer science other than areas of artificial intelligence. The leading inference Bayesian algorithm is the *clique tree* algorithm [51] and Shachter et al. [77] have shown that the weight of the largest clique is bounded by the weight of the union of the loop cutset and the largest parent set of a vertex in a Bayesian network.

Even, Naor, Schieber, and Sudan [28] showed that on a special network called *circular network* any feedback vertex set problem is equivalent to another important NP-hard combinatorial optimization problem, called the *directed multicut problem*, defined by Hu [41] as follows. Given a capacitated network and a set of k source-sink pairs, find a minimum capacity set of edges whose removal disconnects all the source-sink pairs. The relation between feedback set problems and multicut problems was pointed out by Leighton and Rao [53]. Even et al. [28] described a simple procedure that reduces an instance of the feedback arc subset problem to an instance of the minimum multicut problem in circular networks, which are networks such that for each source-sink pair (s_i,t_i) an infinite capacity edge $t_i \rightarrow s_i$ is defined. The approximation algorithms for the feedback set problems proposed in [28], and discussed in Subsection 3.3.2, have been developed by the authors to improve the state of the art of ratios of approximation algorithms for the directed multicut problem.

In a very recent paper by Pachter and Kim [66], a connection between the feedback arc set problem and the *forcing problem* in square grids has been established. Given a graph G admitting a perfect matching M, the *forcing number* of M is the smallest number of arcs in a subset $S \subset M$, such that S is in no other perfect matching. A subset S having this property is said to force M. The concept of forcing arises in combinatorial chemistry [47, 48] leading to extensive study of forcing in hexagonal systems [15, 39, 56]. Pachter et al. [66] solved the forcing problem in special square grid graphs denoted by $R_n = P_{2n} \times P_{2n}$, where $\times$ is the Cartesian graph product and P_{2n} is the path on $2n$ vertices. They used a result of Lucchesi and Younger [60] stating that, for a finite planar directed graph, a minimum feedback set has cardinality equal to that of a maximum disjoint collection of directed cycles. A directed graph G has the *cycle-packing property* if the maximum size of a collection of edge disjoint cycles equals the minimum size of a feedback set. This property still holds for an undirected graph if every orientation of the edges results in a directed graph with the cycle-packing property. Starting from a perfect matching M of a bipartite graph G having the cycle-packing property, Pachter et al. constructed a directed graph $D(M)$ having the same vertex set of G partitioned into two sets A and B and whose arc set contains an arc e directed from A to B if $e \in M$, from B to A otherwise. They proved that there is a one-to-one correspondence between alternating cycles in M and directed cycles in $D(M)$ and that for every feedback set in $D(M)$ there is a forcing set in M of the same cardinality and vice versa.

Some recent papers due to Isaak [42] and Charon et al. [14] examined tournaments problems as a generalization of the feedback arc set problem for digraphs. A (round-robin) *tournament* $T = (X,U)$ is a complete asymmetric graph such that for every $x, y \in X$ there is a unique arc (x,y) or (y,x). A weighted tournament is a tournament for which a weight function w is defined on U with values from the positive set of natural numbers. A classical NP-hard tournament problem is the following: Given a weighted tournament T, find a minimum weighted set of arcs of T such that reversing these arcs makes T transitive. Consider any digraph $D = (X,V)$ as tournament $T = (X,U)$, with $V \subset U$ weighted by w. The weights w are defined on U by $w(u) = 1$ if $u \in V$ and $w(u) = 0$ if $u \in U \setminus V$. Any optimal solution of the feedback arc set problem on D gives an optimal solution to the tournament problem on D and vice versa.

6. Future directions

Despite the large body of work on feedback vertex set, many interesting problems remain to be answered.

Recent advances in approximation algorithms seem to push the boundary closer to the limit. For undirected feedback vertex set, the worst case bound 2 cannot be improved unless the vertex cover problem can approximated within a bound less than 2, which has been conjectured by Hochbaum [40] to be NP-complete. The picture is still less clear for the directed case, where despite the arduous efforts of several researchers, the best known approximation bound is still $O(\log n \log \log n)$, and no approximation algorithm with constant ratio bound has been reported. Yannakakis [88] conjectured that there is a gap between the approximation of the directed and the undirected cases for feedback vertex set. This remains the biggest open question in the approximation of feedback vertex set problems.

Traditionally, the major tool used to attack feedback vertex set has been graph theory, whereas the application of mathematical programming is relatively limited. Recently, Goemans and Williams use a primal-dual formulation to model and establish uniform worst case bounds for a wide range of node-deletion problems. They established a bound of $9/4$. However, it is still inferior to the best known bound, and Goemans and Williamson posed the open question of whether the bound by a primal-dual method can match the best bound of 2 for undirected graphs. Funke and Reinelt [31] developed a polyhedral based integer programming algorithm to exactly solve the feedback vertex set problem. Given the power of mathematical programming to other combinatorial optimization problems, it appears that much work needs to be done along this direction, both with regard to approximation and exact algorithms.

On the specially-structured polynomially solvable cases, Levy and Lowe's result [55] seems to push the study along this line to the limit, at least for flow type graphs. Recent applications in partial scan design report successful applications of the even somewhat brute force exact enumeration methods. All of these methods are used in conjunction with problem reduction techniques, which is quite unique to feedback vertex set. It would be interesting to further investigate the impact of these reduction techniques on the computational complexity of exact or approximate algorithms. In addition, it may also be possible to identify new classes of polynomially solvable feedback set problems (e.g. in VLSI design and circuit testing).

In contrast to the approximation literature, computational studies of feedback vertex set problems seem to be still in their embryonic stage. No modern metaheuristics, except the GRASP procedure recently developed by Pardalos, Qian, and Resende [68] have ever been applied to the feedback vertex set problem. The dimensions of the general problem that can be handled are still quite limited. It seems that this area of computational research has the greatest potential for progress and impact in the coming years. It is also worth noting that, since detecting cycles is a relatively expensive operation, the local search of feedback vertex set appears to be even more difficult than other notorious combinatorial problems like the traveling salesperson or set covering problems. The design of efficient local search procedures will also be a key to the effective computational procedure for feedback vertex set.

ACKNOWLEDGMENT

The research of Paola Festa was supported in part by the Italian Ministry of Scientific Research (MURST) - Research Project MOST 97. The research of Panos Pardalos was supported in part by DIMACS and by National Science Foundation Grants DMI-9622200 and BIR-9505919.

REFERENCES

[1] V. Bafna, P. Berman, and T. Fujito, Constant ratio approximations of the weighted feedback vertex set problem for undirected graphs, in *ISAAC95, Algorithms and Computation*, J. Staples, P. Eades, N. Katoh and A. Moffat Eds., Lecture Notes in Computer Science Vol.1004, Springer-Verlag (1995) pp. 142-151.

[2] R. Bar-Yehuda, D. Geiger, J. Naor, and R.M. Roth, Approximation algorithms for the vertex feedback set problem with applications to constraint satisfaction and Bayesian inference, A preliminary version of this paper appeared in the *Proc. of the* 5^{th} *Annual ACM-SIAM Symp. on Discrete Algorithms*, pp. 344-354, (1994) and subsequently published in *SIAM J. Comput.* Vol.27 No.4 (1998) pp. 942-959.

[3] A. Becker, and D. Geiger, Approximation algorithm for the loop cutset problem, in *Proceedings of the* 10^{th} *Conference on Uncertainty in Artificial Intelligence* Morgan Kaufman (1994) pp. 60-68.

[4] A. Becker, and D. Geiger, Optimization of Pearl's method of conditioning and greedy-like approximation algorithms for the vertex feedback set problem, *Artificial Intelligence* Vol.83 (1996) pp. 167-188.

[5] J. A. Bondy, G. Hopkins, and W. Staton, Lower bounds for induced forests in cubic graphs, *Canad. Math. Bull.* Vol.30 (1987) pp. 193-199.

[6] D.P. Bovet, S. de Agostino, and R. Petreschi, Parallelism and the feedback vertex set problem, *Information Processing Letters* Vol.28 (1988) pp.81-85.

[7] A. Brandstädt, and D. Kratsch, On the restriction of some NP-complete graph problems to permutation graphs, in *Proc. of Fundamentals of Computing Theory*, Lecture Notes in Comp. Sci. Vol.199 (L. Budach, Ed. Springer-Verlag, Berlin, 1985) pp. 53-62.

[8] A. Brandstädt, On improved time bounds for permutation graph problems, in *Proc. of the* 18^{th} *Workshop on Graph-theoretic concepts in computer science*, (Wiesbaden-Naurod, 1992) Springer-Verlag LNCS 657 (1993) pp. 1-10.

[9] M.A. Breuer, and R. Gupta, BALLAST: A methodology for partial scan design, in *Proc. of the* 19^{th} *Int. Symposium on Fault-Tolerant Computing* (1989) pp. 118-125.

[10] M. Cai, X. Deng, and W. Zang, A TDI system and its application to approximation algorithm, in *Proc. of the* 39^{th} *Annual Symposium on Foundations of Computer Science*, Palo Alto, California, November 8-11, (1998).

[11] M. Cai, X. Deng, and W. Zang, A min-max theorem on feedback vertex sets, to appear in *Integer Programming and Combinatorial Optimization: Proceedings* 7^{th} *International IPCO Conference, Lecture Notes in Computer Science* Springer-Verlag (1999).

[12] S. Chakradhar, A. Balakrishnan, and V. Agrawal, An exact algorithm for selecting partial scan flip-flops, Manuscript (1994).

[13] M.S. Chang, Y.D. Liang, Minimum feedback vertex sets in cocomparability graphs and convex bipartite graphs, *Acta Informatica* Vol.34 (1997) pp. 337-346.

[14] I. Charon, A. Guenoche, O. Hudry, and F. Wairgard, New results on the computation of median orders, *Discr. Math.* Vol.165/166 (1997) pp. 139-153.
[15] R. Chen, X. Guo, and F. Zhang, The z-transformation graphs of perfect matchings of hexagonal system, *Discr. Math.* Vol.72 (1988) pp. 405-415.
[16] K.T. Cheng, and V.D. Agrawal, A partial scan method for sequential circuits with feedback, *IEEE Transactions on Computers* Vol.39 No.4 (1990) pp. 544-548.
[17] Chin Lung Lu, and Chuan Yi Tang, A linear-time algorithm for the weighted feedback vertex problem on interval graphs, *Information Processing Letters* Vol.61 (1997) pp. 107-111.
[18] F.A. Chudak, M.X. Goemans, D. Hochbaum, and D.P. Williamson, A primal-dual interpretation of two 2-approximation algorithms for the feedback vertex set problem in undirected graphs, *Operations Research Letters* Vol.22 (1998) pp.111-118. Vol.4 (1979) pp. 233-235.
[19] V. Chvátal, A greedy heuristic for the set covering problem, *Mathematics Of Operations Research* Vol.4 (1979) pp. 233-235.
[20] S.R. Coorg, and C.P. Rangan, Feedback vertex set on cocomparability graphs, *Networks* Vol.26 (1995) pp. 101-111.
[21] D.G. Corneil, and J. Fonlupt, The complexity of generalized clique covering, *Discr. Appl. Math.* Vol.22 (1988) pp. 109-118.
[22] R. Dechter, and J. Pearl, The cycle cutset method for improving search performance in AI, in *Proc. of the* 3^{th} *IEEE on AI Applications* Orlando FL (1987).
[23] R. Dechter, Enhancement schemes for constraint processing: Backjumping, learning, and cutset decomposition, *Artif. Intell.* Vol.41 (1990) pp. 273-312.
[24] J. Donald, J. Elwin, R. Hager, and P. Salamon, A bad example for the minimum feedback vertex set problem, *IEEE Transactions on Circuits and Systems* Vol.32 (1995) pp. 491-493.
[25] R.G. Downey, and M.R. Fellows, Fixed-parameter tractability and completeness I: Basic results, *SIAM Journal on Computing* Vol.24 (1995) pp. 873-921.
[26] P. Erdös, and L. Posa, On the maximal number of disjoint circuits of a graph, *Pubbl. Math. Debrecen* Vol.9 (1962) pp. 3-12.
[27] G. Even, S. Naor, B. Schieber, and L. Zosin, Approximating minimum subset feedback sets in undirected graphs, with applications, in the *Proc. of the* 4^{th} *Israel Symposium on Theory of Computing and Systems* (1996) pp. 78-88.
[28] G. Even, S. Naor, B. Schieber, and M. Sudan, Approximating minimum feedback sets and multicuts in directed graphs, *Algorithmica* Vol.20 (1998) pp. 151-174.
[29] G. Even, J.S. Naor, and L. Zosin An 8-Approximation Algorithm for the Subset Feedback Vertex Problem, 37th Symp. on Foundations of Comp. Sci. (FOCS) (1996) pp. 310-319.
[30] P. Festa, P.M. Pardalos, and M.G.C. Resende, Fortran subroutines for approximate solution of feedback set problems using GRASP, Manuscript, AT&T Labs Research, Florham Park, NJ (1999).
[31] M. Funke, and G. Reinelt, A polyhedral approach to the feedback vertex set problem, Manuscript (1996).
[32] M.R. Garey, and D.S. Johnson, Computers And Intractability –A Guide to the Theory of NP-Completeness, *W. H. Freeman*, San Francisco, (1979).
[33] M.R. Garey, and R.E. Tarjan, A linear-time algorithm for finding all feedback vertices, *Information Processing Letters* Vol.7 (1978) pp. 274-276.
[34] N. Garg, V.V. Vazirani, and M. Yannakakis, Approximate max-flow min-(multi) cut theorems and their applications, *SIAM Journal on Computing* Vol.25 No.2 (1996) pp. 235-251.
[35] F. Gavril, Some NP-complete problems on graphs, in *Proceedings of the* 11^{th} *Conference on Information Science and Systems*, John Hopkins Univ. Baltimore, Md., (1977) pp. 91-95.
[36] M.X. Goemans, and D.P. Williamson, Primal-dual approximation algorithms for feedback problems in planar graphs, in *Proceedings of the* 5^{th} *MPS Conference on Integer Programming and Combinatorial Optimization (IPCO)* (1996) pp.147-161.
[37] M. Grötschel, L. Lovász, and A. Schrijver, Geometric algorithms and combinatorial optimization, Springer-Verlag, Berlin (1988) pp. 253-254.
[38] M. Grötschel, and L. Lovász, Combinatorial optimization: A survey, *DIMACS Technical Report 93-29* DIMACS Rutgers University (1993).
[39] F. Harary, D.J. Klein, and T.P. Zivkovic, Graphical properties of polyhexes: Perfect matching vector and forcing, *J. Math. Chem.* Vol.6 (1991) pp. 295-306.
[40] D. Hochbaum, Approximation algorithms for set covering and vertex cover problem, *SIAM Journal on Computing* Vol.11 No.3 (1982) pp. 555-556.
[41] T.C. Hu, Multi-commodity network flows, *Operations Research* Vol.11 (1963) pp. 344-360.

[42] G. Isaak, Tournaments as feedback arc sets, *Electronic Journal of Combinatorics* Vol.20 No.2 (1995) pp. 1-19.

[43] D.B. Johnson, Finding all the elementary circuits of a directed graph, *SIAM J. Computing*, Vol. 4, No.1 (1975) pp. 77-84.

[44] D.S. Johnson, Approximation algorithms for combinatorial problems, *Journal of Computer and System Science* Vol.9 (1974) pp. 256-278.

[45] R.M. Karp, Reducibility among combinatorial problems, *Complexity Of Computer Computations* R.E. Miller and J.W. Thatcher, Eds., New York: Plenum Press (1972) pp. 85-103.

[46] A.K. Kevorkian, General topological results on the construction of a minimum essential set of a directed graph, *IEEE Trans. Circuits and Systems* Vol.27 (1980) pp. 293-304.

[47] D.J. Klein, and M. Randić, Innate degree of freedom of a graph, *J. Computat. Chem.* Vol.8 (1987) pp. 516-521.

[48] D.J. Klein, T.P. Zivković, and R. Valenti, Topological long-range order for resonating-valance-bond structures, *Phys. Rev. B* Vol.43A (1991) pp. 723-727.

[49] A. Kunzmann, and H.J. Wunderlich, An analytical approach to the partial scan problem, *J. of Electronic Testing: Theory and Applications* Vol.1 (1990) pp. 163-174.

[50] H. Kim, and J. Perl, A computational model for combined causal and diagnostic reasoning in inference systems, in *Proc. of the* 8^{th} *IJCAI*, Morgan-Kaufmann, San Mateo, CA, (1983) pp. 190-193.

[51] S.L. Lauritzen, and D.J. Spiegelhalter, Local computations with probabilities on graphical structures and their application to expert systems (with discussion), *J. Roy. Stat. Soc. Ser.B* Vol.50 (1988) pp. 157-224.

[52] D. Lee, and S. Reedy, On determining scan flip-flops in partial scan designs, in *Proceedings Of International Conference on Computer Aided Design* (1990) pp. 322-325.

[53] T. Leighton, and S. Rao, An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms, in *Proceedings of the* 29^{th} *Annual Symposium on Fundations of Computer Science*, (1988) pp. 422-431.

[54] A. Lempel, and I. Cederbaum, Minimum feedback arc and vertex sets of a directed graph, *IEEE Transactions on circuit theory* CT-13 (1966) pp. 399-403.

[55] H. Levy, and L. Lowe, A contraction algorithm for finding small cycle cutsets, *Journal Of Algorithm* Vol.9 (1988) pp. 470-493.

[56] X. Li, and F. Zhang, Hexagonal systems with forcing edges, *Discr. Math.* Vol.140 (1995) pp. 253-263.

[57] Y.D. Liang, On the feedback vertex set problem in permutation graphs, *Information Processing Letters*, Vol. 52 (1994) pp. 123-129.

[58] J. Liu, and C. Zhao, A new bound on the feedback vertex sets in cubic graphs, *Discrete Mathematics* Vol.148 (1996) pp. 119-131.

[59] E.L. Lloyd, M.L. Soffa, and C.C. Wang, On locating minimum feedback vertex sets, *Journal of Computer and System Sciences* Vol.37 (1988) pp. 292-311.

[60] C.L. Lucchesi, and D.H. Younger, A minimax theorem for directed graphs, *J. London Math. Soc.* Vol.17 (1978) pp. 369-374.

[61] F.L. Luccio, Almost exact minimum feedback vertex set in meshes and butterflies, *Information Processing Letters* Vol.66 (1998, pp. 59-64.

[62] C. Lund, and M. Yannakakis, On the hardness of approximating minimization problems, *Proceedings Of the 25th ACM Symp. On Theory Of Computing* (1993) pp. 286-293.

[63] M.V. Marathe, C.Pandu Rangan, and R. Ravi, Efficient algorithms for generalized clique covering on interval graphs, *Discr. Appl. Math.* Vol.39 (1992) pp. 87-93.

[64] B. Monien, and R. Schultz, Four approximation algorithms for the feedback vertex set problems, in *Proc. of the* 7^{th} *Conference on Graph Theoretic Concepts of Computer Science*, Hanser-Verlag, Münich (1981) pp. 315-326.

[65] T. Orenstein, Z. Kohavi, and I. Pomeranz, An optimal algorithm for cycle breaking in directed graphs, *J. of Electronic Testing: Theory and Applications* Vol.7 (1995) pp. 71-81.

[66] L. Pachter, and P. Kim, Forcing matchings on square grids, *Discr. Math* Vol.190 (1998) pp. 287-294.

[67] C. Papadimitriou, and M. Yannakakis, Optimization, approximation and complexity classes, in *Proc. of the 20th Annual ACM Symp. on Theory of Computing* (1988) pp. 251-277.

[68] P.M. Pardalos, T. Qian, and M.G.C. Resende, A greedy randomized adaptive search procedure for feedback vertex set, *J. Comb. Opt.* Vol.2 (1999) pp.399-412.

[69] D. Peleg, Local majority voting, small coalitions, and controlling monopolies in graphs: A review, in *Proc. of the* 3^{th} *Colloquium on Structural Information and Communication Complexity* (1996) pp. 152-169.

[70] D. Peleg, Size bounds for dynamic monopolies, in *Proc. of the* 4^{th} *Colloquium on Structural Information and Communication Complexity* (1997) Carleton Univ. Press, Ottawa, pp. 165-175.

[71] J. Pearl, Fusion, propagation and structuring in belief networks, *Artif. Intell.* Vol.29 (1986) pp. 241-288.
[72] T. Qian, Y. Ye, and P.M. Pardalos, A Pseudo-ε approximation algorithm for feedback vertex set, *Recent Advances in Global Optimization*, Floudas, C.A. and Pardalos, P.M., Eds., Kluwer Academic Publishing (1995) pp. 341-351.
[73] V. Ramachandran, Finding a minimum feedback arc set in reducible flow graphs, *Journal of Algorithms* Vol.9 (1988) pp. 299-313.
[74] B. Rosen, Robust linear algorithms for cutsets, *Journal of Algorithms* Vol.3 (1982) pp. 205-217.
[75] P.D. Seymour, Packing directed circuits fractionally, *Combinatorica* Vol.15 (1995) pp. 281-288.
[76] A. Shamir, A linear time algorithm for finding minimum cutsets in reduced graphs, *SIAM Journal On Computing* Vol.8 No.4 (1979) pp. 645-655.
[77] R.D. Shatcher, S.K. Andersen, and P. Szolovits, Global conditioning for probabilistic inference in belief networks, in *Proc of the* 10^{th} *Conferences on Uncertainty in AI*, Seattle, WA (1994) pp. 514-522.
[78] A.C. Shaw, The logical design of operating systems, Prentice-Hall, Englewood Cliffs, NJ, (1974).
[79] D.A. Simovici, and G. Grigoras, Even initial feedback vertex set problem is NP-complete, *Information Processing Letters* Vol.8 (1979) pp. 64-66.
[80] G.W. Smith, and R.B. Walford, The identification of a minimal feedback vertex set of a directed graph, *IEEE Transactions on Circuits and Systems* Vol.CAS-22 No.1, (1975) pp. 9-14.
[81] E. Speckenmeyer, On feedback vertex sets and nonseparating independent sets in cubic graphs, *Journal of Graph Theory* Vol.12 (1988) pp. 405-412.
[82] E. Speckenmeyer, On feedback problems in digraphs, in *Lecture Notes in Computer Science*, Springer-Verlag Vol.411 (1989) pp. 218-231.
[83] H. Stamm, On feedback problems in a planar digraph, in R. Möhring ed. *Graph-Theoretic Concepts in Computer Science, Lecture Notes in Computer Science*, Springer-Verlag (1990) Vol. 484 pp. 79-89.
[84] R.E. Tarjan, Depth first search and linear graph algorithms, *SIAM Journal on Computing* Vol.1 (1972) pp. 146-160.
[85] S. Ueno, Y. Kajitani, and S. Gotoh, On the nonseparating independent set problem and feedback set problem for graphs with no vertex degree exceeding three, *Discrete Mathematics* Vol.72 (1988) pp. 355-360.
[86] V. Vazirani, Approximation Algorithms, Manuscript, College of Computing, Georgia Institute of Technology.
[87] C. Wang, E. Lloyd, and M. Soffa, Feedback vertex sets and cyclically reducible graphs, *Journal of the Association for Computing Machinery* Vol.32 No.2 (1985) pp. 296-313.
[88] M. Yannakakis, Node and edge-deletion NP-complete problems, in *Proceedings of the* 10^{th} *Annual ACM Symposium on Theory of Computing* (1978) pp. 253-264.
[89] M. Yannakakis, and F. Gavril, The maximum k-colorable subgraph problem for chordal graphs, *Info. Process. Lett.* Vol.24 (1987) pp. 133-137.
[90] M. Yannakakis, Some open problems in approximation, in *Proc. of the second Italian Conference on Algorithm and Complexity, CIAC'94* Italy, Feb. (1994) pp. 33-39.
[91] D.H. Younger, Minimum feedback arc set for a directed graph, *IEEE Transactions on Circuit Theory* Vol. CT-10 (1963) pp. 238-245.
[92] M. Zheng, and X. Lu, On the maximum induced forests of a connected cubic graph without triangles, *Discr. Math.* Vol.85 (1990) pp. 89-96.

(P. Festa) MATHEMATICS AND COMPUTER SCIENCE DEPARTMENT, UNIVERSITY OF SALERNO, 84081 BARONISSI (SA), ITALY.
E-mail address, P. Festa: `paofes@udsab.dia.unisa.it`

(P. M. Pardalos) CENTER FOR APPLIED OPTIMIZATION, DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, GAINESVILLE, FL 32611 USA.
E-mail address, P. M. Pardalos: `pardalos@ufl.edu`

(M. G. C. Resende) INFORMATION SCIENCES RESEARCH, AT&T LABS RESEARCH, FLORHAM PARK, NJ 07932 USA.
E-mail address, M. G. C. Resende: `mgcr@research.att.com`