Alcom-FT Technical Report Series
ALCOMFT-TR-03-19

Combinatorial Algorithms for Feedback Problems in Directed Graphs*

Camil Demetrescu§ *Irene Finocchi*¶

Abstract

Given a weighted directed graph $G = (V, A)$, the minimum feedback arc set problem consists of finding a minimum weight set of arcs $A' \subseteq A$ such that the directed graph $(V, A \setminus A')$ is acyclic. Similarly, the minimum feedback vertex set problem consists of finding a minimum weight set of vertices containing at least one vertex for each directed cycle. Both problems are NP-complete. We present simple combinatorial algorithms for these problems that achieve an approximation ratio bounded by the length, in terms of number of arcs, of a longest simple cycle of the digraph.

Keywords: approximation algorithms, combinatorial optimization, feedback problems, directed graphs.

1 Introduction

A *feedback arc set* of a (directed) graph is a subset of its arcs whose removal makes the graph acyclic. Similarly, a *feedback vertex set* of a (directed) graph is a subset of its vertices containing at least one vertex for each (directed) cycle. The minimum feedback vertex and arc set problems consist of finding a smallest cost feedback vertex set and a smallest cost feedback arc set, respectively. The cost of the feedback set can be either its cardinality or its weight with respect to a nonnegative weight function.

Feedback problems are fundamental in combinatorial optimization and find application in many different settings: analysis of large-scale systems with feedback, constraint satisfaction problems [4], graph layout [23], and certain scheduling problems [12] represent just some examples. For this reason they have been deeply studied since the late $60's$ (see, for example, [19]).

Related work. The minimum feedback vertex set problem is NP-complete both on directed and on undirected graphs [13, 18] and remains NP-complete even on edge digraphs [14]. On the other hand, the minimum feedback arc set problem on undirected graphs can be easily solved in polynomial time by finding a maximum weight spanning tree, while its directed formulation is NP-complete [13, 18] even on digraphs with total vertex in-degree and out-degree smaller than 3 [14], but is polynomially solvable on planar digraphs [20].

*Work partially supported by the IST Programmes of the EU under contract numbers IST-1999-14186 (ALCOM-FT) and IST-2001-33555 (COSIN), by the MURST Project for Young Researchers “Algorithms and Techniques for the Visualization of Large Graphs”, and by the Italian Ministry of University and Scientific Research (Project “ALINWEB: Algorithmics for Internet and the Web”).

§Dipartimento di Informatica e Sistemistica, Università degli Studi di Roma “La Sapienza”, Via Salaria 113, 00198 Roma, Italy. E-mail: `demetres@dis.uniroma1.it`.

¶Dipartimento di Informatica, Sistemi e Produzione, Università degli Studi di Roma “Tor Vergata”, Via di Tor Vergata 110, 00133 Roma, Italy. E-mail: `finocchi@disp.uniroma2.it`.

NP-completeness results have motivated extensive research for efficient heuristics and approximation algorithms for these problems. In particular, the minimum feedback vertex set problem on undirected graphs has been deeply studied and algorithms with performance ratio equal to 2 have been presented in [1, 5].

Feedback problems on directed graphs appear significantly more difficult to be approximated. In particular, they have been proved to be equivalent from an approximability point of view and to be APX-hard [17], but no available algorithm achieves constant approximation ratio. Heuristics for the minimum feedback arc set problem are described in [9, 10, 12]. In [21] it has been shown that all minimal solutions can be enumerated with polynomial delay. The best known approximation algorithm [11, 22] achieves a performance ratio $O(\log n \log \log n)$, where n is the number of vertices of the digraph, and requires to solve a linear program. These results are in evident contrast with those obtained for the complementary problem, called maximum acyclic subgraph, that can be easily approximated by a ratio even smaller than 2 [6, 15].

Our results. In this paper we focus on feedback problems on directed graphs and we present new approximation algorithms for them built on the top of the local-ratio technique [2]. Our algorithms are combinatorial, run in $O(m \cdot n)$ worst-case time on a digraph with n vertices and m arcs, and, independently of the weight function, achieve an approximation ratio bounded by the length, in terms of number of arcs, of a longest simple cycle of the digraph. According to a preliminary experimental study in a crossing minimization application [7], they proved to be very practical on dense instances with many short cycles.

The remainder of the paper is organized as follows. Section 2 introduces preliminary concepts and reminds the local-ratio technique. Section 3 presents our approximation algorithm for the minimum feedback arc set problem and shows that it can be easily adapted to deal with feedback vertex sets. The algorithm is analyzed in Section 4.

2 Definitions and Notation

Let $G = (V, A)$ be a directed graph, and let $w : A \to \Re^+$ and $z : V \to \Re^+$ be nonnegative weight functions on the arcs and on the vertices of G, respectively. The minimum feedback arc set and vertex set problems can be formally stated as follows:

FAS: Given a directed graph $G = (V, A)$ with nonnegative arc weights $w : A \to \Re^+$, find a minimum weight set of arcs $A' \subseteq A$ such that the directed graph $(V, A \setminus A')$ is acyclic.

FVS: Given a directed graph $G = (V, A)$ with nonnegative vertex weights $z : V \to \Re^+$, find a minimum weight set of vertices $V' \subseteq V$ such that V' contains at least one vertex for each directed cycle of G.

In the following we denote the weights of feedback vertex and arc sets A' and V' with $w(A') = \sum_{(x,y) \in A'} w(x, y)$ and $z(V') = \sum_{v \in V'} z(v)$, respectively.

A feedback arc set A^* is optimum if $w(A^*) \leq w(A')$ for each feedback arc set A'; moreover, a feedback arc set A' is a r-approximation, $r \geq 1$, if $w(A') \leq r \cdot w(A^*)$. Similarly, a feedback vertex set V^* is optimum if $z(V^*) \leq z(V')$ for each feedback vertex set V', and a feedback vertex set V' is a r-approximation, $r \geq 1$, if $z(V') \leq r \cdot z(V^*)$. A feedback set C is minimal if any proper subset of C is not a feedback set itself.

Feedback problems can be naturally thought as *covering problems*, i.e., as the problems of covering all cycles of a given digraph by means of a minimum cost set of vertices or arcs. Hence, the classical techniques adopted for approximating covering problems can be used. In particular,

Algorithm FAS $(G = (V, A);\ w : A \to \Re^+)$

begin

  $F \leftarrow \emptyset$ $\{F$ is the feedback arc set found by the algorithm$\}$

  while $((V, A \setminus F)$ is not acyclic$)$ $\{$Phase 1$\}$

    begin

      Let $\mathcal{C}$ be a simple cycle in $(V, A \setminus F)$

      Let (x, y) be a minimum weight arc in $\mathcal{C}$ and let ϵ be its weight

      for each $(v, w) \in \mathcal{C}$

        $w(v, w) \leftarrow w(v, w) - \epsilon$

        if $w(v, w) = 0$

        then $F \leftarrow F \cup \{(v, w)\}$

    end

  for each $(v, w) \in F$ $\{$Phase 2$\}$

    if $(V, A \setminus F \cup \{(v, w)\})$ is acyclic

    then $F \leftarrow F \setminus \{(v, w)\}$

  return F

end

Figure 1: Finding a minimal feedback arc set of a weighted directed graph.

two main approaches have been investigated in the literature [16]: the *primal-dual* approach and the *local-ratio* approach. In this paper we focus on the local-ratio technique [2, 3], that turns out to be a powerful yet simple tool for designing approximation algorithms.

With respect to covering problems, the Local Ratio theorem can be informally stated as follows: if a cover C is a r-approximation with respect to both a weight function w_1 and a weight function w_2, then C is a r-approximation with respect to the weight function $w_1 + w_2$. This suggests a general strategy followed by many local-ratio approximation algorithms: using weight reductions and solving the problem on instances with simpler weight functions. Informally speaking, if the payment at each step can be proved to cost no more than r times the optimum payment, then the total payment will be at most r times the optimum cost.

3 The Approximation Algorithms

We first consider the minimum feedback arc set problem, proposing a simple combinatorial approximation algorithm based on the use of the local-ratio technique, and then we show that the algorithm can be easily adapted to **FVS**.

3.1 Approximating FAS

According to the overall strategy of local-ratio algorithms, our approach consists of progressively reducing the weights of the arcs of the digraph and adding to the feedback arc set the arcs whose weight becomes equal to 0.

The algorithm consists of two phases. First, it looks for a simple cycle $\mathcal{C}$ in the digraph and, if such a cycle exists, identifies an arc in $\mathcal{C}$ having minimum weight, say ϵ. Then, the weight of all the arcs in $\mathcal{C}$ is decreased by ϵ and the arcs whose weight becomes equal to 0 are removed. If the digraph is now acyclic the first phase terminates, otherwise the previous steps are repeated. After Phase 1, the set of deleted arcs is certainly a feedback arc set, though not necessarily minimal. Hence, the algorithm tries to add back to the digraph some of the deleted arcs, paying attention not to re-introduce cycles. The set of removed arcs is finally returned.

The pseudocode of Algorithm **FAS** and an example of its execution are given in Figure 1 and in Figure 2, respectively. The example also shows the non-minimality of the solution after Phase 1. It

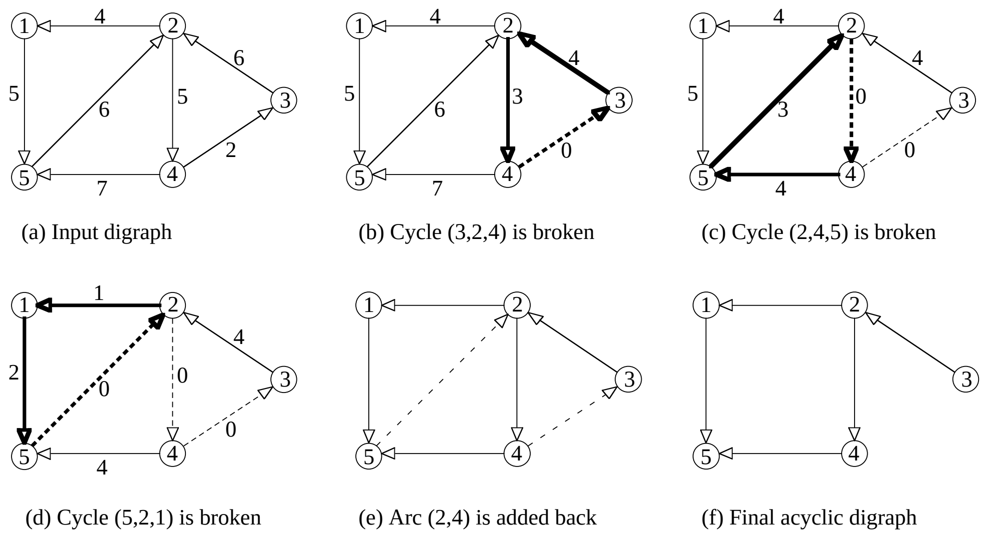


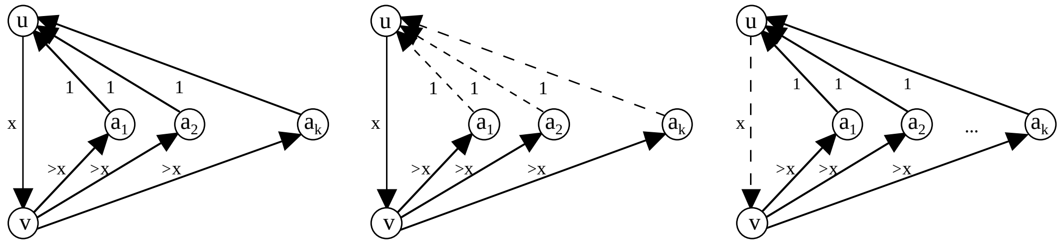
Figure 2: Algorithm execution on a digraph with three simple cycles: (a) input digraph; (b) to (d) phase 1; (e) phase 2. Arcs in the feedback set F are dotted; arcs in the cycle broken at each iteration are bold. Note the progressive reduction of arc weights.

should be clear that, if an arc is considered at a certain iteration but is not removed, it can still be considered for deletion in a successive iteration. For instance, arc $(2,4)$ in Figure 2 is not removed in the first iteration yet belonging to the cycle discovered in that step. However, it is deleted in the second iteration, since there are still cycles it belongs to. This implies that the reduction of the weights of *all* the arcs must be taken into account while computing the payed cost, even if it may happen that either an arc is successively deleted or it is not.

Roughly speaking, our algorithm attempts to find a compromise between two (somewhat opposite) approaches, i.e., removing light arcs, that is, arcs with small weight, and removing arcs belonging to a large number of cycles. Indeed, light arcs are convenient to be deleted as they contribute to breaking cycles, yet increasing the weight of the feedback set only to a limited extent. On the other hand, if a heavy arc belongs to a large number of cycles, it may be convenient to choose it instead of a numerous set of light arcs. An example is shown in Figure 3: according to the fact that k is greater or smaller than x, it may be convenient either to remove the only arc with weight x or the whole set of arcs of weight 1 each. However, a simple greedy approach following one of the two strategies will always fail in one case. Therefore, a somehow “mixed” approach is needed in order to be able to limit the worst-case error.

Moving from the foregoing considerations, algorithm **FAS** decreases the weight of all the arcs in any cycle it finds. The bigger is the number of cycles an arc belongs to, the more likely is the reduction of its weight and the more likely is its subsequent removal. Put another way, heavy arcs belonging to a huge number of cycles become progressively more desirable as the algorithm goes on thanks to the reduction of their weight. In the bad instances of Figure 3, our algorithm chooses a 1-weight arc for $min\{x-1,k\}$ times, meanwhile decreasing the weight of arc (u,v). If $x>k$, the algorithm finally decides to remove arc (u,v). Note that the optimum solution is gained in both cases: if $x>k$ algorithm **FAS** stops with the minimum solution within k iterations; if $x<k$ the feedback arc set built during Phase 1 is not the minimum one, but Phase 2 improves it to the optimum by adding back all the 1-weight arcs.

It is worth pointing out that this approach is considerably different from both the heuristics



Parametric instance

Minimum feedback arc set
when x>k

Minimum feedback arc set
when x<k

Figure 3: Bad instances for simple-minded greedy strategies. Removing light arcs produces bad solutions when $k >> x$. Removing arc (u, v), that belongs to a large number of cycles, produces bad solutions when $x >> k$.

approaches studied in the literature and from the technique that the best approximation algorithm is based on. Indeed, this algorithm first finds an optimal solution to a relaxed integer programming formulation of **FAS** and uses it to partition the set of vertices into two disjoint sets V_1 and V_2; then, it deletes the cheapest set of arcs either from V_1 to V_2 or from V_2 to V_1; finally, it recurses both on V_1 and on V_2. From a practical point of view, one of the main advantages of our algorithm over the previous one relies in its simplicity and on the fact that it does not require any knowledge in linear programming. The analysis of its running time and approximation ratio is presented in Section 4.

3.2 Approximating FVS

The feedback arc and vertex sets problems on directed graphs are equivalent from an approximability point of view: any approximation ratio obtained for one of them can be translated into the same approximation ratio for the other one [16]. However, instead of using the reduction in [16], our algorithm can be directly adapted to solve **FVS** by means of a few straightforward changes. Actually, it is sufficient to work on vertices and vertex weights instead of arcs and arc weights.

Once a cycle $\mathcal{C}$ has been identified in Phase 1, consider a minimum weight vertex in $\mathcal{C}$, say v, and decrease the weight of each vertex in $\mathcal{C}$ by $z(v)$, adding to the feedback set the vertices whose weight becomes equal to 0. Similarly, to get a minimal feedback vertex set, in Phase 2 add back to the graph a (possibly empty) subset of the removed vertices, paying attention not to re-introduce cycles.

4 Analysis of the Algorithms

We limit here to analyze the algorithm for the feedback arc set problem: analogous considerations hold for **FVS**. In particular, we prove in Theorem 1 and in Theorem 2 that algorithm **FAS** finds a minimal feedback arc set of a digraph G in $O(m \cdot n)$ time, n and m being the numbers of vertices and arcs of G, and guarantees an approximation ratio bounded by the length λ of a longest simple cycle of G. Note that the length is in terms of number of arcs, and therefore independent of the weight function.

Theorem 1 *Let $G = (V, A, w)$ be a weighted directed graph with n vertices and m arcs. Algorithm* **FAS** *finds a minimal feedback arc set of G in $O(m \cdot n)$ worst-case running time.*

Proof: We first prove the correctness and then the running time of the algorithm.

Correctness: algorithm **FAS** progressively removes arcs from the input digraph, stopping only when the remaining arcs do not form cycles (lines 3–11). Hence, the set of arcs removed after the first phase is by definition a feedback arc set. To guarantee the minimality of the solution, a maximal subset of the previously removed arcs is added back in the second phase (lines 12–14): in this phase F remains a feedback arc set because the acyclicity condition is tested before any arc addition (line 13).

Running time: at most m iterations can be done in the first phase, since at each step at least one arc is removed from the digraph (i.e., a minimum weight arc in $\mathcal{C}$). At each iteration three basic operations are performed: a simple cycle is found (line 5), a minimum weight arc in the cycle is identified (line 6), and the weights of all arcs in the cycle are updated (lines 7–10). The second and third operations can be performed in $O(n)$ time, as n is the maximum length of any simple cycle of G. A simple-minded implementation of the first operation (by means of a visit), would yield $O(m \cdot (m+n))$ overall running time. However, this bound can be reduced to $O(m \cdot n)$ by using a dynamic algorithm for maintaining reachability information in digraphs subject to deletion of arcs.

Using the reachability data structure in [8], any sequence of arc deletions can be supported in $O(m \cdot n)$ worst-case time and any reachability query can be answered in optimal time. The dynamic algorithm maintains the Boolean transitive closure matrix M and allows it to find a path between two vertices, if any, in time proportional to the length of the path. In addition, we can easily maintain within the same time bound the list L of pairs of vertices (x, y) such that $M[x, y] = M[y, x] = 1$, i.e., x and y lie on a same cycle. To find a cycle, we pick any pair (x, y) from L and we query the reachability data structure to find the paths from x to y and back, which form a cycle. If this cycle is not simple, a simple cycle can be easily derived from it.

The same data structure can be also used to support any sequence of arc insertions [8]. Hence, similarly, the second phase of algorithm **FAS** can be implemented in $O(m \cdot n)$ time. □

In the following we focus on proving the approximation ratio guaranteed by algorithm **FAS**. We denote with w, w_1, and w_2 different nonnegative weight functions for the arcs of a digraph $G = (V, A)$. Moreover, let F^*, F_1^*, and F_2^* be the minimum feedback arc sets of the weighted digraphs (V, A, w), (V, A, w_1), and (V, A, w_2), respectively. The following lemma relates the values of the weights of the minimum feedback arc sets with respect to different weight functions w, w_1, and w_2 when these functions are linearly dependent:

Lemma 1 *Let $G = (V, A)$ be a directed graph and let w, w_1, and w_2 be three nonnegative weight functions on the arcs of G such that $w = w_1 + w_2$. Then it holds:*

$$w_1(F_1^*) + w_2(F_2^*) \leq w(F^*)$$

Proof: Since $w = w_1 + w_2$, we have that $w(F^*) = w_1(F^*) + w_2(F^*)$. Moreover, F^* is a feedback arc set for G with respect to both w_1 and w_2, but it is not necessarily a minimum feedback arc set. Hence, we have that $w_1(F^*) \geq w_1(F_1^*)$ and $w_2(F^*) \geq w_2(F_2^*)$. The claim immediately follows. □

Theorem 2 *Let $G = (V, A, w)$ be a weighted directed graph. Algorithm* **FAS** *approximates a minimum feedback arc set of G within a ratio bounded by the length λ of a longest simple cycle of G.*

Proof: The second phase of algorithm **FAS** is only required for making the previously found feedback arc set minimal. Since the weight of the feedback arc set can only decrease during this phase, it is sufficient to prove that the approximation ratio is already guaranteed after Phase 1. The proof proceeds by induction on the number of iterations of the while-loop in line 3 of Figure 1. This number is finite and strictly decreasing since at each step at least one arc is removed from the digraph.

Base step: no iteration is performed. In this case the input digraph is already acyclic and the empty feedback arc set is obviously an optimal solution.

Induction step: let us consider a generic iteration of the algorithm and let us denote with w the weight of the arcs of G in that iteration. Let $\mathcal{C}$ be the simple cycle identified by the algorithm (line 5), let k be its length, and let ϵ be the weight of the smallest cost arc in $\mathcal{C}$. We define a new weight function w_1 for the digraph (V, A) as follows:

$$\forall (u,v) \in A \qquad w_1(u,v) = \begin{cases} \epsilon & \text{if } (u,v) \in \mathcal{C} \\ 0 & \text{otherwise} \end{cases}$$

Observe that the cost of the minimum feedback arc set of the digraph (V, A) with respect to w_1 is equal to ϵ, i.e., $w_1(F_1^*) = \epsilon$: this is because cycle $\mathcal{C}$ is simple, and removing only one arc is sufficient to break it.

Moreover, the weight of any feedback arc set F w.r.t w_1 cannot be greater than $k \cdot \epsilon \leq \lambda \cdot \epsilon$, because all the arcs not belonging to $\mathcal{C}$ cost 0 and at most all the arcs in $\mathcal{C}$ can participate to F. Therefore, for any F:

$$w_1(F) \leq \lambda \cdot w_1(F_1^*) \tag{1}$$

In the following we denote with F_1 the set of arcs of cycle $\mathcal{C}$ removed by the algorithm in order to break it. As far as the algorithm is concerned, $F_1 = \{(u,v) \in \mathcal{C} \text{ such that } w(u,v) = \epsilon\}$. Since any arc $(u,v) \in \mathcal{C}$ has weight $w_1(u,v)$ equal to ϵ, it holds:

$$w(F_1) = w_1(F_1) \tag{2}$$

Let us now define a new weight function $w_2 = w - w_1$. We have that $0 \leq w_2(\cdot) \leq w(\cdot)$ since $w(\cdot) \geq w_1(\cdot) \geq 0$. In addition, by the inductive hypothesis on the digraph $(V, A \setminus F_1, w_2)$, algorithm **FAS** returns a feedback arc set F_2 of this digraph such that $w_2(F_2) \leq \lambda \cdot w_2(F_2^*)$.

Now, let F be the feedback arc set returned by algorithm **FAS** on the weighted digraph (V, A, w). As far as the algorithm is concerned, F consists both of the arcs in F_1 and of the arcs in F_2, i.e., $F = F_1 \cup F_2$. It is also worth pointing out that $F_1 \cap F_2 = \emptyset$, due to the fact that once an arc has been removed, it will be no longer considered by the algorithm.

By linearity of w, w_1, and w_2, it holds $w(F) = w_1(F) + w_2(F)$. In conclusion:

$$\begin{array}{ll}
w(F) = & (w = w_1 + w_2) \\
w_1(F) + w_2(F) = & (F = F_1 \cup F_2) \\
w_1(F) + w_2(F_1) + w_2(F_2) - w_2(F_1 \cap F_2) = & (F_1 \cap F_2 = \emptyset) \\
w_1(F) + w_2(F_1) + w_2(F_2) = & (w_2 = w - w_1 \text{ and Equation 2}) \\
w_1(F) + w_2(F_2) \leq & (\text{Equation 1}) \\
\lambda \cdot w_1(F_1^*) + w_2(F_2) \leq & (\text{Inductive hypothesis}) \\
\lambda \cdot w_1(F_1^*) + \lambda \cdot w_2(F_2^*) \leq & (\text{Lemma 1}) \\
\lambda \cdot w(F^*) &
\end{array}$$

The inequality $w(F) \leq \lambda \cdot w(F^*)$ proves that algorithm **FAS** is a λ-approximation algorithm. □

We remark that various heuristics may be used to improve the performance of the algorithm. In particular, choosing the shortest available cycle in Phase 1, or ordering arcs by decreasing weight in Phase 2 might be helpful to improve the quality of the solution.

5 Concluding Remarks

We have presented simple approximation algorithms for feedback problems in directed graphs. Our algorithms are combinatorial, run in $O(m \cdot n)$ worst-case time on digraphs with n vertices and m arcs, and, independently of the weight function, guarantee an approximation ratio bounded by the length of a longest simple cycle of the digraph. It would be interesting to carry out an experimental study of our algorithms, addressing both running time and quality of the obtained solution, boosting them with different heuristics, and comparing their behaviour to the performances of the best approximation algorithm (based on linear programming) and of the most popular heuristics for the same problems. According to a preliminary investigation in a crossing minimization application [7], the feedback arc set algorithm proved to be very practical on dense instances with many short cycles.

References

[1] V. Bafna, P. Berman, and T. Fujito. Constant ratio approximations of the weighted feedback vertex set problem. In *Proceedings of the 6th International Symposium on Algorithms and Computation (ISAAC'95)*, pages 142–151, 1995. LNCS 1004.

[2] R. Bar-Yehuda. One for the price of two: A unified approach for approximating covering problems. In *Proceedings of the 1st Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'98)*, pages 49–62, 1998.

[3] R. Bar-Yehuda and S. Even. A local-ratio theorem for approximating the weighted vertex cover problem. *Annals of Discrete Mathematics*, 25:27–46, 1985.

[4] R. Bar-Yehuda, D. Geiger, J.S. Naor, and R.M. Roth. Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and bayesian inference. *SIAM Journal on Computing*, 27(4):942–959, 1998.

[5] A. Becker and D. Geiger. Optimization of Pearl's Method of Conditioning and Greedy-Like Approximation Algorithms for the Feedback Vertex Set Problem. *Artificial Intelligence*, 83:167–188, 1996.

[6] B. Berger and P.W. Shor. Approximation Algorithms for the Maximum Acyclic Subgraph Problem. In *Proceedings of the 1st Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 236–243, 1990.

[7] C. Demetrescu and I. Finocchi. Break the "right" cycles and get the "best" drawing. In B.E. Moret and A.V. Goldberg, editors, *Proceedings of the 2nd International Workshop on Algorithm Engineering and Experiments (ALENEX'00)*, pages 171–182, 2000.

[8] C. Demetrescu and G.F. Italiano. Fully dynamic transitive closure: Breaking through the $O(n^2)$ barrier. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science (FOCS'00)*, pages 381–389, 2000.

[9] P. Eades and X. Lin. A new heuristic for the feedback arc set problem. *Australian Journal of Combinatorics*, 12:15–26, 1995.

[10] P. Eades, X. Lin, and W.F. Smyth. A Fast and Effective Heuristic for the Feedback Arc Set Problem. *Information Processing Letters*, 47(6):319–323, 1993.

[11] G. Even, J. Naor, S. Rao, and B. Schieber. Divide-and-conquer approximation algorithms via spreading metrics. In *Proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science*, pages 62–71, 1995.

[12] M.M. Flood. Exact and Heuristic Algorithms for the Weighted Feedback Arc Set Problem: A Special Case of the Skew-Symmetric Quadratic Assignment Problem. *Networks*, 20:1–23, 1990.

[13] M.R. Garey and D.S. Johnson. *Computers and Intractability: a Guide to Theory of NP-completeness.* W.H.Freeman, 1979.

[14] F. Gavril. Some NP-complete problems on graphs. In *Proceedings of the 11th Conference on Information Sciences and Systems*, pages 91–95, 1977.

[15] R. Hassin and S. Rubinstein. Approximations for the Maximum Acyclic Subgraph Problem. *Information Processing Letters*, 51:133–140, 1994.

[16] D.S. Hochbaum, editor. *Approximations Algorithms for NP-Hard Problems.* Thomson Publishing Company, 1997.

[17] V. Kann. *On the Approximability of NP-Complete Optimization Problems.* PhD thesis, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, 1992.

[18] R.M. Karp. Reducibility among Combinatorial Problems. In R.E. Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[19] A. Lempel and I. Cederbaum. Minimum feedback arc and vertex sets of a directed graph. *IEEE Trans. Circuit Theory*, 13(4):399–403, 1966.

[20] C.L. Lucchesi. *A Minimax Equality for Directed Graphs.* PhD thesis, University of Waterloo, 1966.

[21] B. Schwikowski and E. Speckenmeyer. On Computing All Minimal Solutions for Feedback Problems. Technical report, Universität zu Köln, 1997.

[22] P.D. Seymour. Packing directed circuits fractionally. *Combinatorica*, 15:281–288, 1995.

[23] K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical system structures. *IEEE Trans. Syst. Man Cybern.*, 11(2):109–125, 1981.