ON FEEDBACK PROBLEMS IN DIGRAPHS

Ewald Speckenmeyer
Abteilung Informatik, Universität Dortmund
Postfach 500 500, D-4600 Dortmund 50

Abstract: A subset $F \subseteq V$ of vertices of a digraph $G=(V,A)$ with n vertices and m arcs, is called a feedback vertex set (fvs), if G-F is an acyclic digraph (dag). The main results in this paper are:
(1) An $O(n^2 m)$ simplification procedure is developed and it is shown that the class of digraphs, for which a minimum fvs is determined by this procedure alone, properly contains two classes for which minimum fvs's are known to be computable in polynomial time, see [13,15].
(2) A new $O(n^3 \cdot |F|)$ approximation algorithm MFVS for the fvs-problem is proposed, which iteratively deletes in each step a vertex with smallest mean return time during a random walk in the digraph. It is shown that MFVS applied to symmetric digraphs determines a solution of worst case ratio bounded by $O(\log n)$. The quality of fvs's produced by MFVS is compared to those produced by Rosen's fvs-algorithm, see [11], for series of randomly generated digraphs.

INTRODUCTION

Let $G:=(V,A)$ be a digraph with vertex set $V=\{v_1,\dots, v_n\}$ and arc set $A \subseteq V \times V$. By n we always denote the number of vertices and by m the number of arcs of G. By definition G may contain selfloops $(v,v) \in A$, but no parallel arcs. Two arcs (u,v), (v,u) $\in A$, $u \neq v$, are called antiparallel. For a subset $V' \subseteq V$ we define G-V' to be the digraph, which is induced by the vertex set V-V' in G. A subset $F \subseteq V$ is called a **feedback vertex set (fvs)** of G, if G-F is an acyclic digraph (dag). A minimum fvs F of G is a fvs of G of minimal cardinality and we define $\mathbf{f(G)}:=|F|$. The problem of determining a fvs of cardinality f(G) of G is called the **fvs-problem.**
Related to the fvs-problem is the **feedback arc set-problem (fas-problem)** where we look for a minimum arc set $B \subseteq A$ such that the digraph G-B, resulting from G by deleting all the arcs of B from G, is a dag. It is well known that the fas-problem can be reduced to the fvs-problem simply by solving the fvs-problem for the corresponding arc-digraph of G. The fvs-problem can be reduced to the fas-problem as well, simply by splitting each vertex v of G into vertices iv and ov, where an arc runs from iv to ov and all incoming arcs into v are going into iv and all arcs leaving v are leaving ov. Then a minimum fas in the resulting digraph G' can be transformed into a minimum fvs of G. Because of this analogy we concentrate on the fvs-problem in this paper.
Solving the fvs-/ fas-problem often occurs as an algorithmic problem in computer science. We mention here only three applications. In the area of operating systems the problem of breaking deadlocks is formulated as fvs-problem, see [11], e.g..
Proving partial correctness of programs by means of Floyd's method requires to determine first a fvs, as small as possible, of the corresponding flowchart digraph, see [3]. Checking whether a VLSI-circuit C satisfies certain design rules, relies

on first deleting as few links as possible from C s.t. the resulting circuit C' is a combinational circuit, i.e. C' is a dag. Here we meet the fas-problem, see [1].
Unfortunately the fvs- and the fas-problem, both, belong to the class of NP-hard problems, see [4]. So one is often content finding sufficiently small, but not necessarily optimal, fvs's/ fas's for digraphs in "reasonable" time, or one is interested in finding large "interesting" subclasses of digraphs, for which the fvs-/ fas-problem can be solved in polynomial time. Concerning these aims there are only few general results, see [5,6,11,13,15]. For this reason Wang, et al. denote the fvs-problem to be "the least understood of the classic (NP-complete, E.Sp.) problems", see [15].

In this paper we show the following results.
In chapter 2 we develop a $O(n^2(n+m))$ simplification procedure, which first determines all vertices, which can "easily" be seen to belong to a minimum fvs of G, and then all parts from the remaining digraph are deleted, which are "irrelevant" for the fvs-problem. We denote the class of digraphs, for which a minimum fvs is found by applying this simplification procedure only, the class of **fvs-simple** digraphs. Then we show that the class of fvs-simple digraphs properly contains the class of **reducible flow graphs**, which has been considered by Shamir in[13], and the class of **cyclically reducible graphs**, introduced and studied by Wang, et al. in [15]. The algorithm by Wang, et al. for solving the fvs-problem for cyclically reducible graphs has the same running time as our simplification procedure, where Shamir's fvs-algorithm for reducible flow graphs based upon depth first search (dfs) runs in time $O(n+m)$, only.
In chapter 3 a Markovian approximation algorithm for the fvs-problem in digraphs, MFVS, is proposed, and it is shown that MFVS always determines a minimum fvs of G, if $f(G)=1$. Then we show that MFVS applied to symmetric digraphs G determines a fvs F of worst case ratio $|F|/f(G)=O(\log n)$, and MFVS may produce fvs's F for certain classes of symmetric digraphs G satisfying $|F|/f(G)=c\cdot\log n$, for some $c>0$.
We conjecture that MFVS applied to arbitrary digraphs always determines fvs's of worst case ratio $O(\log n)$.
The idea behind MFVS is as follows. In order to determine a (minimum) fvs F of a digraph G it is sufficient to decompose G into its strongly connected components, which can be treated independently. Each strongly connected component H with at least two vertices is processed as follows. For each vertex $v\in V(H)$ the mean return time $r(v)$ to vertex v during a random walk in H is determined. Then a vertex v with smallest mean return time $r(v)$ is added to F and deleted from H. Then $H-\{v\}$ recursively is processed like G.
The most time consuming part of MFVS is to determine the mean return times $r(v)$,