A Heuristic for the Feedback Arc Set Problem

Peter Eades
Department of Computer Science
The University of Newcastle
NSW 2308
Australia
eades@cs.newcastle.edu.au

Xuemin Lin
Department of Computer Science
The University of Western Australia
Nedlands, WA 6907
Australia
lxue@cs.uwa.oz.au

ABSTRACT

Suppose that $G = (V, A)$ is a simple directed graph where $n = |V|$ and $m = |A|$. A *feedback arc set* is a set of arcs whose reversal makes G acyclic. The *feedback arc set problem* is to find a feedback arc set with the minimum cardinality. Generally, this problem is NP-hard even for a cubic directed graph. In this paper, we present a new heuristic to produce a feedback arc set with a small size. This new heuristic produces a feedback arc set whose size is at most $\frac{m}{4}$ for a cubic directed graph. This improves on all previous results.

1. Introduction

The feedback arc set (FAS) problem has been extensively investigated for the last few decades [BS90, dv83, ELS89, ELS93 J70, R88, S71, S80]. Recently, we have investigated this problem because it relates to our current research on one important aspect in information visualization - drawing a directed graph clearly [EL89]. The relationship between the feedback arc set problem and the problem of clearly drawing a directed graph may be found in [EL89, L92].

Since the FAS problem is NP-hard [K72], it is unlikely that we can solve this problem in polynomial time. A number of heuristics have been published. Some achieve the optimal solutions for special classes of graphs (such as "reducible flow graphs" and "planar graphs" [R88]), while the others are measured by performance bounds. The most recent two heuristics for solving the FAS problem may be found in [BS90, ELS93].

The heuristic provided in [BS90] produces a feedback arc set whose size is at most $\frac{m}{2} - \theta(\frac{m}{\Delta(G)^{1/2}})$ for a general directed graph and at most $\frac{5m}{18}$ for a cubic directed graph. Here, m is the number of arcs, n is the number of vertices, and $\Delta(G)$ denotes the maximum number of arcs incident to a vertex. This heuristic requires $O(mn)$ execution time.

A simple heuristic presented in [ELS93], aimed at sparse directed graphs (which frequently arise in graph drawing), outputs a feedback arc set whose size is at most $m/2 - n/6$ and has execution time $O(m)$. Over directed graphs satisfying $m \in O(n)$, this performance bound is at least as good as that of [BS90]. Also, this bound is in fact better than that of [BS90] over directed graphs with $m \in O(n)$ and $\Delta(G)$ not bounded by a constant.

In this paper, we present a new heuristic Algorithm-FASH which refines the algorithm in [ELS93] in order to potentially reduce the size of the feedback arc set produced. In [L92], it has been proven that Algorithm-FASH produces a feedback arc set whose size is at most $m/2 - n/6$ for a general directed graph. The detailed proof of the above general bound for Algorithm-FASH may be found in [L92]. Here we prove a new result given by Algorithm-FASH on cubic directed graphs:

Algorithm-FASH produces a feedback arc set whose size is at most $m/4$.

The rest of the paper is organized as follows. In Section 2, necessary preliminaries are given. In Section 3, Algorithm-FASH is presented, while in Section 4, we prove the performance of Algorithm-FASH for cubic directed graphs. This is followed by conclusions and remarks.

2. Preliminaries

We recall the following basic graph notation from [BM].

In a directed graph G, an arc with identical endpoints is a *loop*. A *directed path* with identical endpoints is a *directed cycle*. A directed cycle with length 2 is a *two-cycle*. Two vertices u and v are *strongly connected* if there are two directed paths in G, one from u to v and the other from v to u.

A directed graph is *strongly connected* if each pair of vertices is strongly connected. A subgraph G' of G is a *strongly connected component* of G if:

- G' is strongly connected, and
- for each pair of vertices u and v, where u is in G' and v is not in G', u and v are not strongly connected.

For each vertex u in G, $d_G(u)$ denotes its *total degree*, that is, the number of the arcs incident to u, $d_G^-(u)$ denotes the *indegree* of u, that is, the number of arcs into u, and $d_G^+(u)$ to denote the *outdegree* of u, that is, the number of arcs out from u. If the corresponding graph is clear from context, then $d_G(u)$, $d_G^-(u)$ and $d_G^+(G)$ are respectively abbreviated to $d(u)$, $d^-(u)$ and $d^+(u)$.

A directed graph is *cubic* if for each vertex u, $d(u) = 3$. A vertex in a directed graph is a *source* if its indegree is zero, a *sink* if its outdegree is zero.

An ordered list of the vertices of a directed graph G is a *vertex sequence* of G. If $s(G) = (v_1, v_2, \cdots, v_n)$ is a vertex sequence of a directed graph G. the an arc (v_i, v_j) is *leftward* (*rightward*) with respect to $s(G)$ if $j < i$ ($j > i$).

In the rest of the paper, n and m denote the cardinalities of the vertex set and the arc set respectively.

3. Algorithm-FASH

Algorithm-FASH, which is presented in this section, computes a vertex sequence $s(G)$ for a directed graph G; and then the set of leftward arcs will be output as the feedback arc set. It is clear that there is a vertex sequence such that the leftward arc set with respect to this sequence is a feedback arc set with the minimum cardinality.

3.1. Motivation

Algorithm-FASH, like the algorithm in [ELS93], essentially consists of the following four steps:

Step 1: Iteratively remove sinks (if any) to prepend to a vertex sequence s_2; and if the remaining graph is empty then go to Step 4 else go to Step 2.

Step 2: Iteratively remove sources (if any) to append to a vertex sequence s_1; and if the remaining graph is empty then go to Step 4 else go to Step 3.

Step 3: Choose a vertex u, such that the difference between the number of rightward arcs and the number of leftward arcs is the largest, and remove u to append to s_1; if the remaining graph is empty then goto Step 4 else goto Step 1.

Step 4: A vertex sequence s is formed by concatenating s_1 with s_2; and the leftward arc set for the vertex sequence s is reported as a feedback arc set.

Note that Step 1 and Step 2 do not produce any feedback arcs. In case there is more than one candidate at Step 3, the algorithm in [ELS93] nondeterministically chooses one. This may speed up the execution, but potentially degrade the performance. Thus, Step 3 is the key. In Algorithm-FASH, we add some additionally greedy criteria for a choice of a vertex at Step 3. Further, some manipulations of a directed graph will be added to allow Step 3 to be more effective.

To further consider the structure of a directed graph, we should first decompose the graph into strongly connected components. A standard $O(m)$ time procedure may be found in [Sed, pp. 481-483]. This procedure, called by DSC(G), returns the sequence $(G_1, G_2, \cdots, G_k)$ of the strongly connected components of a directed graph G, with the property that there are no leftward arcs between these components (that is, no arcs from G_j to G_i for $i < j$). Thus, we need only find the feedback arc sets for each of these components.

To describe the techniques to increase the greediness of Step 3, some further concepts are needed.

In a strongly connected directed graph G, a directed path $(u_1, u_2, \cdots, u_k)$ is *condensible* if $k \geq 3$, $d_G(u_1) \geq 3$, $d_G(u_k) \geq 3$, and for $2 \leq i \leq k-1$, $d_G^+(u_i) = d_G^-(u_i) = 1$. Here for $2 \leq i \leq k-1$, u_i is a *middle vertex* of the directed path, while u_1 and u_k are respectively called the *start* vertex and the *end* vertex of the directed path.

A directed graph G is *fully condensed* if it is strongly connected and there are no condensible directed paths in G. The *condensation* of strongly connected directed graph $G = (V, A)$ is formed by collapsing the condensible paths to single arcs; more precisely the condensation is the directed graph $G_c = (V_c, A_c)$ such that

- V_c is the largest subset of V such that V_c contains no middle vertices of a condensible directed path of G; and
- $A_c = \{e\colon e \in A$ or e from a start vertex of a condensible directed path in G to the end vertex of the directed path$\}$.

It is clear that the condensation of a strongly connected graph is fully condensed. Algorithm-FASH uses a function CON(G) to produce the condensation G_c of a strongly connected directed graph G such that when Algorithm-FASH removes a vertex u from G, u is chosen from G_c. By a depth first search technique, CON(G) can be implemented in

$O(m)$ time.

Next we investigate the introduction of an additional criterion at Step 3 on fully condensed directed graphs. We begin with an example. For the graph illustrated in Fig. 1, an implementation of the above four steps first nondeterministically chooses either vertex 1 or vertex 2. If vertex 1 is chosen first, then 2 feedback arcs will be produced. On the other hand, if vertex 2 is chosen first, then only 1 feedback arc will be produced.

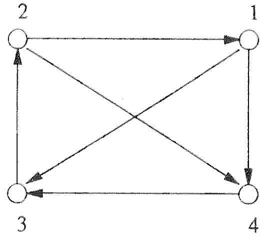

Fig 1

Consider the arc $(3,2)$: after removing it, vertex 3 becomes a sink. But the removal of any arc incident to 1 would not produce any sink. We would like to force Step 3 to choose vertex 2 in this situation. In general, we would like to choose a vertex v so that the remaining graph after v is deleted contains a vertex u which is "unbalanced", that is, the difference between the indegree and the outdegree of u is high. Thus for a strongly connected directed graph $G=(V,A)$ and a vertex set $U \subseteq V$ (a candidate set at Step 3), Algorithm-FASH applies a procedure CHS(G,U) which returns a vertex v in U such that there is an arc $(u,v) \in A$ with

$$d_G^-(u) - d_G^+(u) - 1 = \max\{d_G^-(b) - d_G^+(b) - 1 \colon (b,a) \in A,\ a \in U\}.$$

Intuitively, we hope that Step 3, as implemented with CHS, produces some extremely "unbalanced" vertices. This will potentially guarantee that the next iteration may produce a small number of feedback arcs. For instance, for the graph in Fig. 1, Step 3 will choose vertex 2 through implementing CHS. Clearly, CHS(G,U) can be implemented in $O(m)$ time.

Algorithm-FASH also applies another linear time procedure TAKEMAX(G,U). This takes a directed graph G as the input and outputs the set U of the vertices which have the maximum value of the difference between outdegree and indegree. To describe Algorithm-FASH clearly, a combination OBTAIN(G) of TAKEMAX and CHS is presented as follows:

```
OBTAIN(G: graphs): vertex
 If G has only one vertex v
 then
    return v
 else
   TAKEMAX(G,U);
   return CHS(G,U)
```

3.2. The Description of Algorithm-FASH

Suppose that G is not empty. Then Algorithm-FASH is described as follows:

Algorithm-FASH (G: directed graph) : vertex sequence
FASH1. $s \leftarrow \emptyset$;

FASH2. $(G_1, G_2, \cdots, G_k) \leftarrow$DSC($G$);

FASH3. Return the concatenation of SCFASH(G_1) , SCFASH(G_2) , ... , SCFASH(G_k)

Algorithm-SCFASH (G : strongly connected graph) : vertex sequence

SCFASH1. $s \leftarrow \emptyset$;

SCFASH2. $G_c \leftarrow$ CON(G);

SCFASH3. $v \leftarrow$ OBTAIN(G);

SCFASH4. Return the sequence formed by prepending v to FASH($G - v$)

We report the leftward arcs for s as the resulting feedback arcs.

The Theorem below follows since all of DSC, CON, CHS, and TAKEMAX use linear time.

Theorem 1: Algorithm-FASH executes in $O(mn)$ time, where n is the number of the vertices and m is the number of arcs of a graph. □

3.3. Performance Guarantee for Cubic Directed Graphs

In this section, we prove a performance guarantee of Algorithm-FASH restricted to cubic directed graphs.

Theorem 2: Suppose that $G = (V, A)$ is a cubic directed graph with no two-cycles and no loops. Then Algorithm-FASH produces at most $\frac{m}{4}$ feedback arcs, where m is the number of the arcs of G.

Note that the bound in Theorem 2 is an improvement on the bound $\frac{5m}{18}$ (that is, approximately $0.278m$) in [BS90]. The scope of Theorem 2 includes the case where there are some *multiple arcs* in G; see Fig 2, for example. To prove Theorem 2, we first prove the following Lemma.

Lemma 3: Suppose that G is a strongly connected directed graph with no two-cycles and no loops, and the total degree of each vertex in G is not greater than 3. Further suppose that there is at least one vertex in G such that its total degree in G is 3; and G_c is the condensation of G. Then there is an arc (u, v) in G_c such that $d^-_{G_c}(u) = 2$ and $d^+_{G_c}(v) = 2$.

Proof: Note that G_c is fully condensed; and in the strongly connected directed graph G, there is no vertex whose total degree is greater than 3, and no sinks or sources. It follows that G_c is cubic. We also should note that G_c has no loops.

The above facts immediately imply that there are at least two vertices a and b in G_c such that $d^-_{G_c}(a) = 2$ and $d^+_{G_c}(b) = 2$. Since G_c is strongly connected, there is a directed path $(u_1, u_2, \cdots, u_k)$ with $a = u_1$ and $b = u_k$. If $k = 2$, that is, (a, b) is an arc of G_c, then the Lemma holds.

Otherwise $k \geq 3$. Suppose that the Lemma does not hold in this case. Since G_c is cubic and $d^-_{G_c}(u_1) = 2$, by our assumption that the Lemma does not hold, we have that $d^-_{G_c}(u_2) = 2$. Following the path with this argument, we find that $d^-_{G_c}(u_k) = 2$. Thus, $d^+_{G_c}(b) + d^-_{G_c}(b) = 2 + 2 = 4$, contradicting the fact that G_c is cubic. Hence the Lemma holds. □

For a cubic directed graph G, procedure OBTAIN(G_c) always returns vertices of indegree 1 in G_c. These vertices may have outdegree 2 (in the case where the G_c at that time has at least one vertex of total degree 3) or outdegree 1 (in the case where all vertices of G_c at that time have total degree 2). In counting the number of leftward arcs produced by Algorithm-FASH we are particularly interested in the following two subsets V^1 and V^2 of the vertex set of G:

(3.1) V^1 consists of the vertices which are returned by OBTAIN(G_c) at line (SCFASH3) and have outdegree 1 in the value of G_c at that time; and

(3.2) V^2 consists of the vertices which are returned by OBTAIN(G_c) at line (SCFASH3) and have outdegree 2 in the value of G_c at that time.

If a directed graph G is as illustrated in Fig 2 then Algorithm-FASH produces 2 feedback arcs, while there are 9 arcs in G. Thus for this graph, Theorem 2 holds. The proof of Theorem 2 explicitly excludes this case.

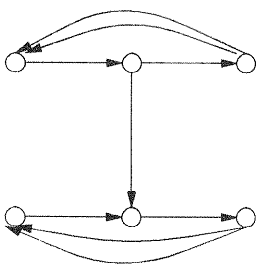

Fig 2

Next we prove Lemma 4, which is the key for the proof of Theorem 2.

Lemma 4: Suppose that $G = (V, A)$ is a cubic directed graph with no loops and no two-cycles, and that the underlying graph of G is connected; and that G is not the graph in Fig 2. Further suppose that V^1 and V^2 are defined as in (3.1) and (3.2). Then for each $u \in V^2$ and each $v \in V^1$, there are respectively two sets V^2_u and A^2_u, and two sets V^1_v and A^1_v, such that

(1) $V^2_u \subseteq V$, $V^1_v \subseteq V$, $A^2_v \subseteq A$, and $A^1_v \subseteq A$, and

(2) $|V_u^2| = 2$, $|A_u^2| = 5$, $|V_v^1| \geq 3$, and $|A_v^1| \geq 3$, and

(3) $V_u^2 \cap V_v^1 = \emptyset$ and $A_u^2 \cap A_v^1 = \emptyset$, and

(4) for each pair $\{u, u'\}$ of distinct vertices in V^2, $V_u^2 \cap V_{u'}^2 = \emptyset$ and $A_u^2 \cap A_{u'}^2 = \emptyset$, and

(5) for each pair $\{v, v'\}$ of distinct vertices in V^1, $V_v^1 \cap V_{v'}^1 = \emptyset$ and $A_v^1 \cap A_{v'}^1 = \emptyset$.

(For example in Fig 6, $V^1 = \{4\}$, $V^2 = \{1\}$, $V_1^2 = \{1,3\}$, $V_4^1 = \{2,4,6\}$, $A_1^2 = \{(1,2),(1,5),(3,1),(5,3),(4,3)\}$, and $A_4^1 = \{(2,6),(6,4),(4,2)\}$.)

Proof: Suppose that w is returned by OBTAIN(G_c). Let G_w be the value of the graph G_1 one step before the choice of w by OBTAIN(G_c), that is, at line (SCFASH2) in Algorithm-FASH. Note that G_w is strongly connected.

Say $w \in V^1$. Then no vertices in G_w have total degree 3, thus G_w must be a directed cycle.

First for each $v \in V^1$, we construct V_v^1 and A_v^1 explicitly as follows. We choose the vertex set of G_v (a directed cycle) as V_v^1, and the arc set of G_v as A_v^1. Since G has no two-cycles and no loops, and G_v is a subgraph of G, we have that $|V_v^1| = |A_v^1| \geq 3$.

Next we construct V_u^2 and A_u^2 explicitly for each $u \in V^2$ as follows. From an inspection of Algorithm-FASH and Lemma 3, one may deduce that G_u has a vertex $w \neq u$ such that $d_{G_u}(u)^+ = 2$, and $d_{G_u}^-(w) = 2$; and that there is a directed path P_{wu} from w to u in G_u where P_{wu} is either an arc or a condensible directed path with w as its start vertex and u as its end vertex. We choose $V_u^2 = \{u, w\}$. There are three cases for V_u^2 with respect to G_u:

1. there are at least 5 arcs incident to either u or w in G_u; or
2. there are at most 4 arcs incident to either u or w in G_u, and P_{wu} has at least two middle vertices; or
3. there are at most 4 arcs incident to either u or w in w, and P_{wu} has at most one middle vertex.

For case 1, let A_u^2 consist of any 5 arcs incident to either u or w. For case 2, note that G_u has no loops and two-cycles, since G_u is a subgraph of G. Thus G_u is as illustrated in Fig 3, where the arc set of G_u has at least 5 arcs.

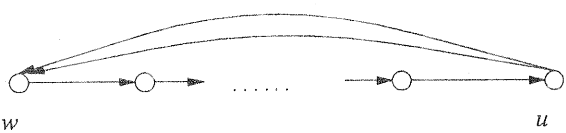

Fig 3

Let A_u consist of any 5 arcs in the arc set of G_u for case 2. Note that G_u is a subgraph of the cubic directed graph G. For case 3, P_{wu} must (since there are no 2-cycles) have a middle vertex a such that G_u is induced by the triple (w, a, u), as illustrated in Fig 4.

Since G is cubic, there is an arc e_a incident to a in G which is neither (w, a) nor (a, u). Hence for case 3, let A_u^2 consist of e_a and the 4 arcs incident to either u or w.

It is clear that the properties (1) and (2) of this Theorem hold. The following facts follow immediately from the above construction and Algorithm-FASH:

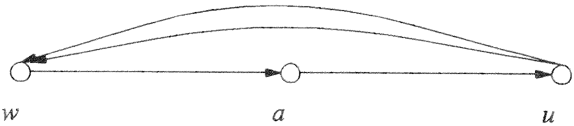

Fig 4

Fact 1: for each vertex $u \in V^2$ such that V_u^2 is covered by case 1, the subgraph containing only the vertex w in V_u^2 $(w \neq u)$ is a strongly connected component of $G_u - u$; and

Fact 2: for each vertex $u \in V^2$ such that V_u^2 is covered either by case 2 or case 3, and for each vertex z in G_u such that $z \neq u$, the subgraph containing only z is a strongly connected component of $G_u - u$; and

Fact 3: for each vertex $u \in V^2$ such that V_u^2 is covered by case 3, G_u is a strongly connected component of G; and

Fact 4: for each $v \in V^1$, and each $z \in V_v^1$ such that $z \neq v$, the subgraph, containing only z, is a strongly connected component of $G_v - v$.

From an inspection of Algorithm-FASH and the above facts, one may deduce that properties (3) and (5) hold; and for a pair $\{u, u'\}$ of distinct vertices in V^2, $V_u^2 \cap V_{u'}^2 = \emptyset$, where one of V_u^2 and $V_{u'}^2$ is not covered by case 3, then $A_u^2 \cap A_{u'}^2 = \emptyset$.

Next we verify that for every pair $\{u, u'\}$ of distinct vertices in V^2, if both V_u^2 and $V_{u'}^2$ are covered by case 3 then $A_u^2 \cap A_{u'}^2 = \emptyset$. To do this, we only need to verify that with respect to the two subgraphs G_u and $G_{u'}$ which are respectively induced by the triple (w, a, u) and triple (w', a', u') illustrated in Fig 4, there are two arcs e_a and $e_{a'}$ respectively in A_u^2 and $A_{u'}^2$ such that:

- $e_a \neq e_{a'}$, e_a incident to a, and $e_{a'}$ incident to a'; and
- e_a is neither (w, a) nor (a, u); and
- $e_{a'}$ is neither (w', a') nor (a', u').

From our assumption that the underlying graph of G is connected and that G is cubic and that G is not the graph as illustrated in Fig 2, the above claim follows immediately. Hence $A_u^2 \cap A_{u'}^2 = \emptyset$ in the case that both V_u^2 and $V_{u'}^2$ are in case 3.

Hence the Lemma holds. □

Next we prove Theorem 2.

Proof of Theorem 2:

Without loss of generality, we may assume that the underlying undirected graph of G is connected. Note that for the graph illustrated in Fig 2, the Theorem holds. Thus next we prove that if G is not covered by the case in Fig 2 then the Theorem also holds.

Suppose that V^2 and V^1 are defined as in (3.2) and (3.1). Let $|V^1| = n_1$ and $|V^2| = n_2$. From Algorithm-FASH, it follows that the removal of a vertex (see line (SCFASH2) in Algorithm-FASH) which is either in V^2 or in V^1 from the value of G at that time causes one leftward arc. Note that Algorithm-FASH produces leftward arcs due only to the removal of the vertices in either V^2 or V^1. Thus Algorithm-FASH produces at most r feedback arcs (leftward arcs) where

(3.3) $r = n_1 + n_2$.

By Lemma 4, we have that $2n_2 + 3n_1 \le n$ and $5n_2 + 3n_1 \le m$. Then we can rewrite these inequalities as:

(3.4) $2n_2 + 3n_1 + x = n$, and

(3.5) $5n_2 + 3n_1 + y = m$,

where $x \ge 0$ and $y \ge 0$. From (3.3), (3.4) and (3.5), it follows immediately that $r \le \frac{7m}{27}$. To further reduce the bound, we next prove that $x + 2y \ge n_1$.

Note that for all $u \in V^1$, the directed graph (V_u^1, A_u^1) is a directed cycle (see the proof of Lemma 4); this cycle is obtained by $DSC(G)$, at the line (FASH2), as a strong connected component in the value of G at that time. We partition V^1 into two sets V_1^1 and V_2^1, defined by:

(1) For each $u \in V_1^1$, $|V_u^1| \ge 4$, and

(2) For each $u \in V_2^1$, $|V_u^1| = 3$.

We may immediately verify that for each $u \in V_2^1$, the cycle (V_u^1, A_u^1) is obtained from G by either

(a) The deletion of an edge e_u, which is incident to a vertex in V_u^1, not in A_v^2 for any $v \in V^2$, and not in A_v^1 for any $v \in V^1$ and $v \ne u$. (In the worst case, e_u may join two directed cycles.)

or

(b) The deletion of a vertex, which is not in V_v^2 for any $v \in V^2$, and not in V_v^1 for any $v \in V^1$ and $v \ne u$. (See the case illustrated by Fig 4; vertex a is an example.)

Hence, $x + 2y \ge |V^1| = n_1$. This can be rewritten as

(3.6) $y = \frac{n_1 + z - x}{2}$,

for some $z \ge 0$

Since G is cubic, $n = \frac{2m}{3}$. Replacing y in (3.5) by (3.6), and solving (3.4) and (3.5) for n_1 and n_2, we have that

$$n_1 = \frac{m}{6} + \frac{z}{8} - \frac{3x}{4}, \text{ and}$$

$$n_2 = \frac{m}{12} - \frac{3z}{16} + \frac{5x}{8}.$$

Thus

$$r = \frac{m}{6} + \frac{z}{8} - \frac{3x}{4} + \frac{m}{12} - \frac{3z}{16} + \frac{5x}{8}$$

$$= \frac{m}{4} - \frac{z}{16} - \frac{x}{8}.$$

Hence $r \le \frac{m}{4}$.

This completes the proof of Theorem 2. □

The bound of Theorem 2 for Algorithm FASH is tight, since for the graph in Fig 5, in the worst case the algorithm returns 3 feedback arcs out of a total of 12 arcs.

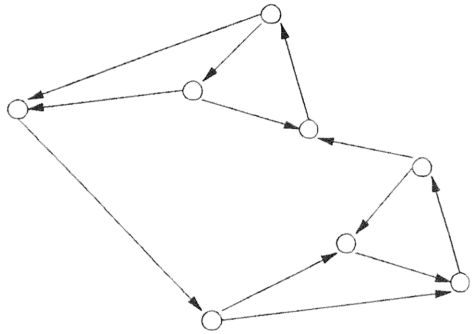

Fig 5

We can also show a lower bound for any algorithm on cubic directed graphs.

Lemma 5 For each n divisible by 6, there is a simple cubic directed graph (with no loops or two-cycles) for which every feedback arc set has at least $\frac{2m}{9}$ arcs.

Proof: Note that if a directed graph G is k copies of the directed graph illustrated in Fig 6 then G has $6k$ vertices and $9k$ arcs and at least $2k$ feedback arcs. Thus the Lemma holds. $\square$

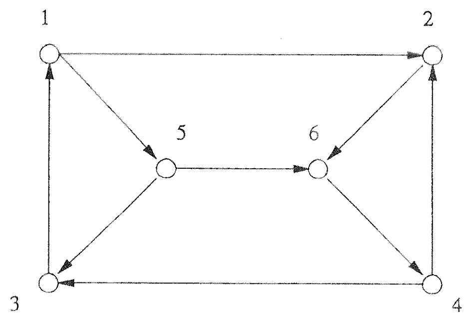

Fig 6

4. Conclusions and Remarks

In Section 3, we describe a new and simple heuristic with a good performance bound for cubic graphs and $O(mn)$ execution time. This bound is better than the bound in [BS90], while the execution time is the same. For a general simple directed graph, we can only show the performance bound $m/2 - n/6$ [L92]. This is at least as good as that of [BS90] over sparse directed graphs. In future, we would like to find a precise estimate for the performance bound of Algorithm-FASH on dense directed graphs, since the estimation employed in [L92] is quite loose.

Further remarks on the complexity of approximating the feedback arc set can be found in [BS90].

References

[BM] J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*, Macmillan, 1976.

[BS90] B. Berger & P. W. Shor, "Approximation algorithms for the maximum acyclic subgraph problem", *Proc. First ACM - SIAM Symposium on Discrete Algorithms* (1990) 236-243.

[dV83] W. F. de la Vega, "On the maximum cardinality of a consistent set of arcs in a random tournament", *J. Combin. Theory, Series B 35* (1983) 328-332.

[EL89] P. Eades & X. Lin, "How to Draw a Directed Graph", Proc. 1989 IEEE Workshop in Visual Languages, IEEE Computer Society Press pp. 13-17, 1989.

[ELS89] P. Eades, X. Lin & W. F. Smyth, "Heuristics for the feedback arc set problem", Technical Report No. 1, Curtin University of Technology, School of Computing Science (1989).

[ELS93] P. Eades, X. Lin & W. F. Smyth, "A fast and effective heuristic for the feedback arc set problem", *Information Processing Letters*, (1993) 319-323.

[EM65] P. Erdos & J. W. Moon, "On sets of consistent arcs in tournaments", *Canadian Mathematical Bulletin* (1965) 269-271.

[J70] H. A. Jung, "On subgraphs without cycles in tournaments", *Combinatorial Theory & Its Application II*, North-Holland (1970) 675-677.

[K72] R. M. Karp, "Reducibility among combinatorial problems, *Complexity of Computation*, Plenum Press (1972) 85-103.

[L92] X. Lin, "Analysis of Algorithms for Drawing Graphs", Ph.D thesis, University of Queensland, Department of Computer Science (1992).

[R88] V. Ramachandran, "Finding a Minimum Feedback Arc Set in Reducible Flow Graphs", *Journal of Algorithms* 9, 299-313, 1988.

[Sed] R. Sedgewick, *Algorithms*, Second Edition, Addison-Wesley Publishing Company, 1988.

[S61] P. Slater, "Inconsistencies in a schedule of paired comparisons", *Biometrika 48* (1961) 303-312.

[S71] J. Spencer, "Optimal ranking of tournaments", *Networks* 1 (1971) 135-138.

[S80] J. Spencer, "Optimally ranking unrankable tournaments", *Period. Math. Hungar.* 11-2 (1980) 131-144.

(Received 11/2/94)