The Maximum Acyclic Subgraph Problem and Degree-3 Graphs

Alantha Newman

Laboratory for Computer Science, MIT, Cambridge, Mass 02139
E-mail: alantha@theory.lcs.mit.edu

We study the problem of finding a maximum acyclic subgraph of a given directed graph in which the maximum total degree (in plus out) is 3. For these graphs, we present a simple combinatorial algorithm that achieves an 11/12-approximation (the previous best factor was 2/3 [1]), (ii) a lower bound of $\frac{3905}{3906}$ on approximability. The problem of finding a better-than-half approximation for general graphs is open.

1. INTRODUCTION

Given a directed graph $G = (V, E)$, the maximum acyclic subgraph problem is to find a maximum cardinality subset E' of E such that $G' = (V, E')$ is acyclic. The problem is NP-hard [3] and the best-known polynomial-time computable approximation factor for general graphs is $\frac{1}{2}$.

In this paper, we focus on graphs in which every vertex has total degree (in-degree plus out-degree) at most 3. Throughout this paper, we refer to these graphs as *degree-3* graphs. The problem remains NP-hard for these graphs [3]. In Section 2, we present an algorithm that finds an $\frac{11}{12}$-approximation. This improves on the previous best guarantees of $\frac{2}{3}$ for graphs with maximum degree 3 and $\frac{13}{18}$ for 3-regular graphs [1]. The algorithm is purely combinatorial and relies heavily on exploiting the structure of degree-3 graphs. As a corollary of a Theorem in [4, 5], we obtain an approximation lower bound of $\frac{3905}{3906}$ in Section 3.

2. COMBINATORIAL APPROXIMATION ALGORITHMS

In [1], Berger and Shor present an algorithm that returns an acyclic subgraph of size at least $\frac{2|E|}{3}$ for degree-3 graphs that do not contain 2-cycles. For 3-regular graphs (note that the set of 3-regular graphs is a proper subset of the set of degree-3 graphs) with no 2-cycles, an algorithm that returns an acyclic subgraph of size $\frac{13|E|}{18}$ is given in [1]. In this section, we

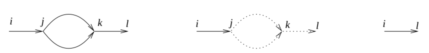

FIG. 1.

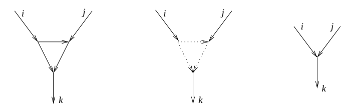

FIG. 2.

show that the problem in degree-3 graphs (with or without 2-cycles) can be approximated to within $\frac{11}{12}$ of optimal using simple combinatorial methods. First we give an $\frac{8}{9}$-approximation algorithm to illustrate some basic arguments. Then we extend these arguments to give an $\frac{11}{12}$-approximation algorithm.

2.1. An $\frac{8}{9}$-Approximation

Given a degree-3 graph $G = (V, E)$ for which we want to find an acyclic subgraph $S \subseteq E$, we can make the following assumptions.

(i) All vertices in G have in-degree and out-degree at least 1 and total degree exactly 3.

(ii) G contains no directed or undirected 2- or 3-cycles.

The explanation for assumption (i) is as follows. If G contains any vertices with in- or out-degree 0, we can immediately add all edges adjacent to these vertices to the acyclic subgraph S, since these edges are contained in any maximal acyclic subgraph. Additionally, we can contract all vertices in G that have in-degree 1 and out-degree 1. For example, say that vertex j in G has in-degree 1 and out-degree 1 and G contains edges (i, j) and (j, k). Then at least one of these two edges will be included in any maximal acyclic subgraph of G. Thus, contracting vertex j is equivalent to contracting edge (i, j) and adding it to the acyclic subgraph S.

Now we explain assumption (ii). We can contract multi-edges without adding cycles to the graph, thus removing any undirected 2-cycles. This is shown in Figure 1. The edges in the undirected 2-cycle are added to S since they are included in any maximal acyclic subgraph. In Figures 1 and 2, the dotted edges are added to S. Similarly, we can remove any undirected 3-cycle by contracting it and adding its edges to S. This results in a degree-3 vertex as shown in Figure 2. Contracting an undirected 3-cycle will not introduce any new cycles into the graph since each of the vertices in the 3-cycle has in-degree and out-degree at least 1 by (i).

In the case of directed 2- and 3-cycles, we can remove the minimum number of edges from the graph while breaking all such cycles. For directed 2-cycles, consider the two adjacent non-cycle edges of a 2-cycle. If they are both in edges, or both out edges, as in Figure 3A, then we can break the

2-cycle by removing an arbitrary edge. If one is out and the other is in, as in Figure 3B, then only one of the edges in the 2-cycle is consistent with the direction of a possible cycle containing both of edges that are not in the 2-cycle. For example, in Figure 3B, we would remove edge (i, j). For directed 3-cycles, consider Figure 4A. In this case, or in the analogous case where three edges point towards the 3-cycle, we can remove any edge from the 3-cycle. In the other case, we remove an edge from the 3-cycle, so that the path from the single in edge or to the single out edge is broken. For example, in Figure 4B, we would remove edge (j, k).

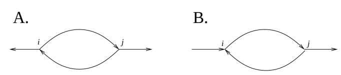

FIG. 3.

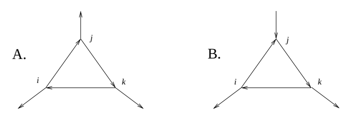

FIG. 4.

We now consider two subclasses of degree-3 graphs. We will use the following definition.

DEFINITION 2.1. An *α-edge* is an edge (i, j) such that vertex i has in-degree 2 and out-degree 1 and vertex j has in-degree 1 and out-degree 2.

For example, edge (j, k) in Figure 4B is an α-edge. First, we consider the case where G contains no α-edges. If there are no α-edges, then we can find the maximum acyclic subgraph in polynomial time. We will use the following lemma.

LEMMA 2.1. *If G is a 3-regular graph and contains no α-edges, then all cycles in G are edge disjoint.*

Proof. Assume that there are two cycles in G that have an edge (or a path) in common. First case: assume that these two cycles have a single edge (i, j) in common, i.e. edge (i, j) belongs to both cycles, but edges (a, i) and (j, b) each belong to only one of these cycles. Then vertex i must have in-degree 2 and vertex j must have out-degree 2. Thus, edge (i, j) is a α-edge, which is a contradiction. Second case: assume these two cycles have a path $\{i, \ldots, j\}$ and that this path is maximal, i.e. edge (a, i) and (j, b) each belong to only one of these cycles. Vertex i must have in-degree 2 and vertex j must have out-degree 2. Therefore, at least one of the edges on the path must be an α-edge, which is a contradiction. ▮

Since all the cycles in a graph with no α-edges are edge disjoint, we can find the maximum acyclic subgraph of such a graph in polynomial time. Given a graph G containing no α-edges, we simply find a cycle in G, throw away any edge from this cycle, and add edges to the acyclic subgraph S by contracting appropriate edges in G or removing appropriate edges from G until G satisfies properties (i) and (ii). We repeat until there are no more cycles in G.

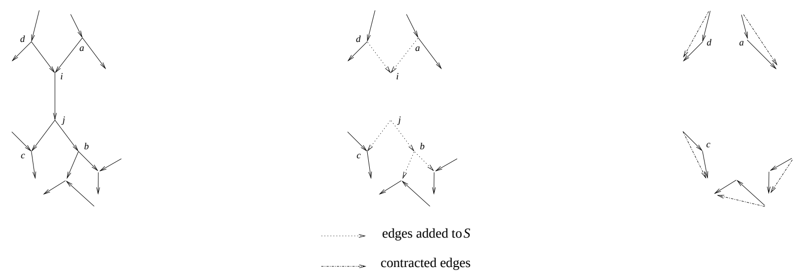

FIG. 5. An illustration of step 4.

If G contains α-edges, then the problem is NP-hard. For this case, we give the following $\frac{8}{9}$-approximation algorithm. Define $C(e)$ as the connected component containing edge e. Define $E(e)$ as the set of edges adjacent to edge e, i.e. the edges that share an endpoint with e. For example, if e is edge (i, j) in the first picture in Figure 5, then $E(e)$ contains edges $(d, i), (a, i), (j, c)$, and (j, b). S is the solution set. The first part of the algorithm is the following procedure. An illustration of step 4 is shown in Figure 5.

While G contains α-edges, do the following:

1. Make sure G is 3-regular and remove all 2- and 3-cycles from G (see explanation of assumptions (i) and (ii)).
2. Find an α-edge e in G.
3. If $|C(e)| = 9$, solve this component exactly.
4. Else remove e from G. Add $E(e)$ and any other edges with in- or out-degree 0 to S. Contract any vertices with in-degree and out-degree 1.

When there are no more α-edges in G, then we can solve for the maximum acyclic subgraph in polynomial time as discussed previously. Then, we uncontract every edge in S that corresponds to a path contracted in some execution of step 1 or step 4. For every edge *not* in S that corresponds to some contracted path, we throw away one edge from the path, and add the remaining edges to S. Thus, every time we contract a vertex, we guarantee that at least one more edge will be added to S.

THEOREM 2.1. *The algorithm is an $\frac{8}{9}$-approximation for the maximum acyclic subgraph problem in degree-3 graphs.*

Proof. We show that for every edge we remove, we contract or add to S a total of at least 8 edges. Consider a α-edge (i, j) in G. There must be 4 distinct vertices within distance one from i and j (since there are no 2-cycles or 3-cycles). Thus, there must be at least 9 edges in this neighborhood. If there are exactly 9 edges, then we have a component with 9 edges and the algorithm solves this component exactly. Otherwise, if there are more than 9 edges in the neighborhood of edge (i, j) (i.e. there could be as many as twelve edges) then for each of the 4 distinct vertices that are exactly one edge away from i or j, we can either contract this vertex, or we can add two more edges to S (which would let us add more than 8 edges to S in this round). Note that $E(e)$ contains 4 edges, which are added to S immediately. Therefore at least 8 edges are added to S for each edge removed. ∎

2.2. An $\frac{11}{12}$-Approximation

We now show how to extend the previous algorithm to obtain an $\frac{11}{12}$-approximation algorithm. In our $\frac{8}{9}$-approximation algorithm, we arbitrarily choose α-edges to remove. There are degree-3 graphs such that if we arbitrarily choose α-edges to remove, then we may obtain an acyclic subgraph with size only $\frac{8}{9}$ of optimal. We will show that if we choose the α-edges to remove carefully, then we can always ensure that the resulting graph contains certain α-edges whose removal allows us to add at least 11 edges (rather than 8) to the solution set.

In order to analyze the steps of the algorithm more easily, we consider a further modification of a given degree-3 graph. We contract any pair of adjacent vertices in which each vertex has in-degree 1 or each vertex has out-degree 1. An example of such a pair of adjacent vertices is shown in Figure 6. Here, j, k is a pair of vertices both with in-degree 1 and f, i is a pair of vertices both with out-degree 1, so we contract edges (f, i) and (j, k). In order to account for the contracted edges, if a vertex has d out edges or d in edges after an edge was contracted, then the *value* of these edges is $2d - 2$, since this is the number of edges they represent in the original graph. For example, in Figure 6, there are now three incoming edges to vertex i. These three edges represent 4 edges in the original degree-3 graph, so they have value 4. In other words, if the three edges coming into vertex i are added to the acyclic subgraph for the modified graph, then this is equivalent to adding all 4 edges to the acyclic subgraph for the original graph. After contracting the relevant edges, the resulting graph will no longer be a degree-3 graph, but will correspond to a degree-3 graph. However, every edge still has in- or out-degree 1 and total degree at least 3. Hence, we can still handle undirected and directed 2- and 3-cycles as described in Section 2.1 and thus property (ii) holds. We now have the additional assumption about the given graph G for which we want to find an acyclic subgraph.

(iii) G contains no adjacent vertices such that both vertices have in-degree 1 or both vertices have out-degree 1.

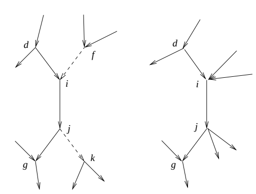

FIG. 6. Edges (f, i) and (j, k) will be contracted.

When we remove an edge e from a graph G, the graph $G - e$ will represent the graph that is obtained by removing edge e from G, removing all edges adjacent to a vertex with in- or out-degree 0 in the resulting graph, contracting all resulting vertices that have in-degree 1 and out-degree 1 and all edges (i, j) such that both i and j have in-degree or out-degree 1. We will use the following definitions.

DEFINITION 2.2. Edge (i, j) is a *profitable α-edge* if either i or j has in-degree or out-degree at least 3.

DEFINITION 2.3. A *super-profitable graph* is a graph that contains either a 4-cycle or an α-edge (i, j) in which the in-degree of i plus the out-degree of j is at least 6.

Our algorithm will use the following lemmas.

LEMMA 2.2. *If e is a profitable α-edge, then removing e from G allows us to add 11 edges to the solution set S.*

Proof. Consider a profitable α-edge $e = (i, j)$. When we remove edge e, we can immediately add at least 6 edges to S since the total value of the edges incoming to i and outgoing from j is at least 6. Since $E(e)$ contains at least 5 distinct vertices and since there are no 2- or 3-cycles, we can make at least 5 contractions. Otherwise it is an isolated component and we can solve it exactly. ∎

LEMMA 2.3. *If G is not super-profitable and G does not contain any profitable α-edges, then G contains an edge e such that the graph $G - e$ contains a profitable α-edge.*

Proof. For some α-edge $e = (i, j)$, we let $V(e)$ denote the set of vertices adjacent to i and j. For example, in Figure 7, $V(i, j)$ is the set $\{a, b, c, d\}$. This is the set of vertices that would be contracted if we removed edge (i, j) from G. The first case we consider is when there is at least one vertex in $V(e)$–wlog say it is vertex a–such that there is no edge with one endpoint a and the other endpoint in $V(e)$. Vertex a must have in-degree 2 and out-degree 1, as shown in Figure 7. Then besides edge (j, a), there are also edges (f, a) and (a, g) for some vertices f and g. Vertex f must have out-degree 2; if it had in-degree 2, then $\{f, a\}$ would have been contracted. For the same reason, vertex g must have out-degree 2. When we remove edge (i, j), vertex a is contracted, but neither vertex f nor vertex g is affected, since neither vertex is in the neighbor set of (i, j). Thus, the graph $G - e$ contains the edge $\{f, g\}$ which will be contracted. If G contains one edge with out degree greater than 2, then it contains a profitable α-edge.

The second case we consider is when for each vertex v in the set $V(e)$, G contains an edge with one endpoint v and the other endpoint in the set $V(e)$. Note that we cannot have an edge from a or b to c or d because G contains no 4-cycles. G also contains no undirected 3-cycles. Therefore, in this case, the only possible situation is the one depicted in Figure 8. Note that vertices l and f must have out-degree 2 and vertices g and h must have in-degree 2. Thus, if we remove the α-edge (a, f), we would contract vertices c and j, and the graph $G - (a, f)$ would contain the profitable α-edge (b, ℓ). Note that vertex h is unaffected by the removal of edge (a, f). ∎

LEMMA 2.4. *If G is not a super-profitable graph and G contains a profitable α-edge, then there is some set $\{e_1, \ldots e_k\}$ of edges for some $k \in \{1, 2, 3\}$ such that $G - \{e_1, \ldots e_k\}$ contains a profitable α-edge and removing these k edges from G allows us to add at least $11k$ edges to the solution set S.*

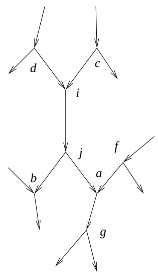

FIG. 7.

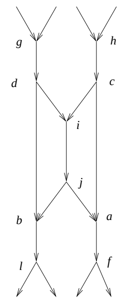

FIG. 8.

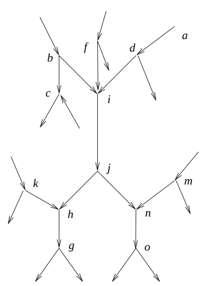

FIG. 9.

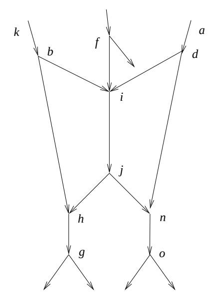

FIG. 10.

Proof. G must contain a profitable α-edge $e = (i, j)$ that has in-degree 2 and out-degree 3 or vice versa. The first case is when $V(i, j)$ contains at least one vertex adjacent to i and at least one vertex adjacent to j such that neither of these vertices is adjacent to another vertex in $V(i, j)$. See Figure 9 for an example of this. Here, n and d form such a pair of vertices. In this case, if we remove edge e, $G - e$ will contain a profitable α-edge. For example, in Figure 9, suppose n has in-degree 2. Then if we remove edge (i, j), the edge incoming to vertex m will become a profitable α-edge. If n has in-degree 3, note that we can remove the profitable α-edge (n, o) and vertex i will still have in-degree 3 in the resulting graph (since G is not super-profitable, it does not contain 4-cycles, so there is no edge from o to b, f or e).

The second case is when $V(i, j)$ contains two pairs of adjacent vertices as shown in Figure 10. In this case, we can remove both α-edges from below, which in this case would be edges (h, g) and (n, o). If either vertex d or h

has in-degree 3, then we can remove one of the profitable α-edges adjacent to one of these vertices and the edge (i, j) will still be a profitable α-edge in the resulting graph since vertex j will have out-degree 3. Otherwise if both d and h have in-degree 2, then we have two subcases to consider. The first is that there is an edge from o to k, i.e. edge (i, j) is contained in a 6-cycle. However, this is not a problematic case because if we remove edges (h, g) and (n, o), all of the 5 other edges in the 6-cycle will be added to S. Thus, in this case we can make much more than a profit of 11 edges per discarded edge, since the optimal solution can also only get 5 edges from a 6-cycle.

In the last case, assume (i, j) is not in a 6-cycle. In this case, if we remove edges (h, g) and (n, o) we can immediately add at least 18 edges to S (i.e. we can add all of the edges shown in Figure 10 except for two of the edges adjacent to f and in addition, we can add the two edges coming into a and two edges coming into k, which are not shown, for a total of 18) and we can make at least 9 contractions. In the next move we can remove an α-edge such that the resulting graph contains a profitable α-edge by Lemma 2.3. Thus, we can remove 3 edges, add at least 35 edges to S which is at least 11 edges per discarded edge. ▮

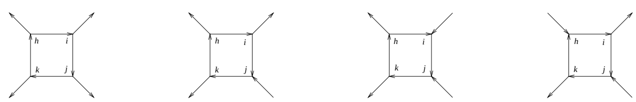

FIG. 11.

LEMMA 2.5. *If G is a super-profitable graph, then G contains some α-edge whose removal allows us to add at least 14 edges to the solution set S.*

Proof. If G contains an α-edge with total in- and out-degree at least 6, then we can add at least 14 edges to S (8 immediately plus 6 contractions). If G contains a 4-cycle, then there are the following 4 cases shown in Figure 11. Each case is easy to handle optimally except for the last one. In the first case, it doesn't matter which edge we remove from the 4-cycle–any one is optimal. In the second and third, we remove edge (j, k).

Now consider the last case. Note that edge (h, i) is an α-edge. Assume the total in-degree of h plus out-degree of i is less than 6 (otherwise, we have the case above). Without loss of generality, assume h has in-degree only 2 in the modified graph, as shown in Figure 12. In this case, the incoming edge to vertex m is an α-edge. If we remove this α-edge, we add at least 8 edges to S. If vertices n and p are unique, then note that the resulting graph contains a 3-cycle since edge (m, h) will be removed and vertex h contracted as a result. Thus, on the next move we will get 8 extra edges. Thus, by removing one edge, we can add at least 14 edges to S. If n and p are not unique (i.e. there are two adjacent 4-cycles), and we remove edge (p, m) (or (n, m)), we will contract vertices h and i, which results in a 2-cycle. However, we can only add 5 edges to S by handling a 2-cycle optimally (we add 3 immediately and make 2 contractions), which is not enough to establish our lemma.

Therefore, we argue the following. If there is an edge from n to m *and* from r to q, then we have 3 adjacent 2-cycles. It may be the case as shown in the second picture in Figure 12, that these edges form a connected component with 9 edges. However, in this case we can solve exactly and so we do not discard any possibly unecessary edges. So we consider the case shown in the last drawing in Figure 12. In this case, if there is not an edge from t to s, then we can use the original argument and obtain a graph with a 3-cycle after removing the α-edge adjacent to t. If there is an edge from t to s, then notice that after we remove edge (m, n) and obtain the 2-cycle $\{j, k\}$, we will obtain *another* 2-cycle after we handle the $\{j, k\}$ 2-cycle optimally. Thus, we will add at least 10 extra edges to S bringing the total to at least 14. ▮

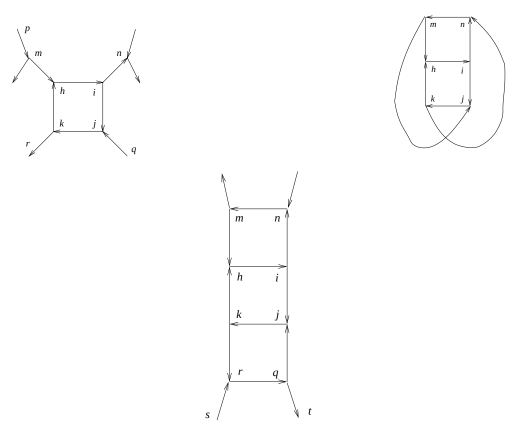

FIG. 12.

We have now stated all the lemmas that we will use to show that we can approximate our problem to within $\frac{11}{12}$. The algorithm is similar to the previous algorithm, except that the while loop is more complex. We will give a high-level description of the new while loop.

The main idea is that during each iteration of the while loop, we want to remove a profitable α-edge from the graph and simultaneously ensure that the resulting graph also contains a profitable α-edge or is super-profitable. We can assume that the given degree-3 graph G contains a profitable α-edge. If it does not, we can use Lemma 2.3 to obtain a graph that does. We will only discard one edge in the process and since G contains at least one cycle (otherwise it is already acyclic), the number of edges in the new graph is no less than the maximum acyclic subgraph of the original graph. Then we have two cases. In the first case, if this graph is super-profitable, by Lemma 2.5, we can remove an α-edge and add 14 edges to S. If after removing this edge, we are not left with a graph that is super-profitable or contains a profitable α-edge, then we can use Lemma 2.3 again to obtain a graph that contains a profitable α-edge. Thus we will discard two edges and add at least 22 edges to S. In the second case, if the graph is not super-profitable, we can use Lemma 2.4 remove a set of $k \in \{1, 2, 3\}$ edges and add a set of $11k$ edges to S so that the resulting graph contains a profitable α-edge.

THEOREM 2.2. *The algorithm is an $\frac{11}{12}$-approximation algorithm for the maximum acyclic subgraph problem in degree-3 graphs.*

Proof. If G is a super-profitable graph, then by Lemma 2.5, there is some α-edge whose removal allows us to add 14 edges to S. If we are not left with a super-profitable graph or a graph containing a profitable α-edge, then by Lemma 2.3 we can find an α-edge whose removal leaves us with a graph containing a profitable α-edge. Thus, if we would discard at most two edges and add at least 14+8=22 edges to S.

If G is not a super-profitable graph, then by Lemma 2.4, we add at least 11 edges to S for each discarded edge and are left with a graph containing a profitable α-edge. ▮

3. A LOWER BOUND

We can make a modification of the gadgets in [4, 5] to obtain the following lower bound for degree-3 graphs. Specifically, we can add edges to the gadgets so that the graphs obtained in the reduction are degree-3 graphs. The original reduction in [4, 5] was from the problem of linear equations mod 2 with exactly 3 variables per clause. In this paper, we use Theorem 1 from [2], which shows that the problem of linear equations mod 2 with exactly 3 variables per clause *and each variable occuring in at most 3 clauses, i.e. (3-OCC-E3-LIN-2),* is NP-hard to approximate to within better than $61/62 + \epsilon$ for any $\epsilon > 0$.

THEOREM 3.1. *It is NP-hard to approximate the maximum acyclic subgraph of a 3-regular graph to within* $\frac{3905}{3906} + \epsilon$ *for any* $\epsilon > 0$.

Proof. We can convert the clause and variable gadgets depicted in Figure 3 of [5] to clause and variable gadgets in which each vertex has degree 3. In Figure 3 of [5], each clause gadget has 36 edges. For each vertex labeled $x_2, \dots x_5, y_2, \dots y_5, z_2, \dots z_5$, we can add two edges so that these 12 vertices are now each degree-3. An example of this is shown in Figure 13. This adds 24 edges per clause gadget. Note that none of these new edges are α-edges and only α-edges belong to a minimum feedback arc set.

Now we can connect these clause gadgets so that the resulting graph is degree-3. First, we can connect the clause gadgets to the vertices x_0, x_1, y_0, y_1, z_0, z_1 as shown in Figure 3 in [5]. Since each variable appears at most 3 times, these vertices have in-degree at most 3 and out-degree at most 3. We can assume each variable appears exactly 3 times, since otherwise the reduction graph will have fewer edges. We can transform these degree-6 vertices to degree-3 vertices as shown in Figure 14. Note that every edge labeled $x = 1$ in a clause gadget is in a cycle with every other edge labeled $x = 0$ from this clause gadget or from other clause gadgets. This is how consistency in an assignment is maintained.

Suppose we have an assignment of the variables for an instance of *3-OCC-E3-Lin-2*. We say an assignment corresponds to an acyclic subgraph if all edges labeled $x = 0$ are removed if x is *true* in the assignment and if all edges labeled $x = 1$ are removed if x is *false* in the assignment. If a clause is satisfied, we only need to remove 3 edges from the respective clause gadget and if the assignment for all clauses is consistent, then there are no cycles between clauses. Note that since a variable can occur at most 3 times, at most one of the clauses it appears in can be false. Otherwise, we would flip the value of that variable and obtain an assignment that satisfies more clauses.

Thus, an optimal assignment that satisfies s clauses and does not satisfy u clauses corresponds to an acyclic subgraph with $57s + 56u + 6m$ edges. It is NP-hard to distinguish between a set of clauses in which all m clauses can be satisfied and at most $61/62m$ clauses can be satisfied. Thus, if we can approximate the problem to within more than $\frac{3905}{3906}$, we can distinguish between the case in which we have $57m+6m$ (which corresponds to all clauses being satisfied) and the case in which we have $57(61/62)m + 56(1/62)m + 6m$. ▮

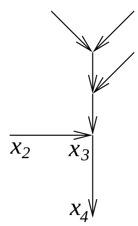

FIG. 13.

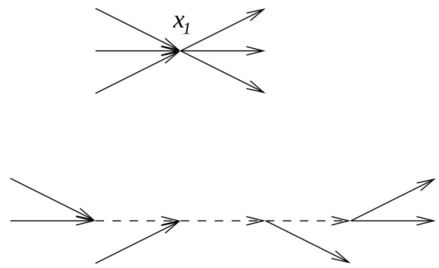

FIG. 14.

4. COMMENTS

A preliminary version of this paper appeared in the proceedings of APPROX 2001. The proof of Theorem 4.2 in the preliminary version is incorrect. This is due to an error in the proof of Lemma 4.1: the construction used may not actually preserve the size of the feedback arc set, i.e. there is a counter example. Additionally, the same error was made in Theorem 3.1 of the preliminary version, which has been amended and appears as Theorem 3.1 in this version.

ACKNOWLEDGEMENTS

I thank Santosh Vempala for many discussions on the maximum acyclic subgraph problem.

REFERENCES

1. Bonnie Berger and Peter W. Shor. Tight Bounds on the Maximum Acyclic Subgraph Problem, *Journal of Algorithms*, vol. 25, pp 1–18, 1997.
2. Piotr Berman and Marek Karpinski. On Some Tighter Inapproximability Results, *Proceedings of ICALP*, pp 200-109, 1999.
3. Richard M Karp. Reducibility Among Combinatorial Problems, in R. Miller and J. Thatcher eds., *Complexity of Computer Computations*, Plenum Press, pp 85-103, 1972.
4. Alantha Newman. Approximating the Maximum Acyclic Subgraph, M.S. Thesis, MIT, June 2000.
5. Alantha Newman and Santosh Vempala. Fences Are Futile: On Relaxations for the Linear Ordering Problem, *Proceedings of IPCO 2001*, Springer-Verlag.