Feedback Arc Set

Anurag Uppuluri

May 2020

1 Problem Definition

Given a digraph[1], $G = (V, E)$, a *feedback arc set* (FAS) or *feedback edge set* is a set of edges, $S \subseteq E$, such that every directed cycle in G has at least one edge in S. In other words, removing S from G would remove all the cycles in G and make it a Directed Acyclic Graph (DAG). [1]

For example, in the following graph (figure 1), all possible directed cycles are:

- $g - f - c - d - g$[2]
- $f - c - d - e - f$
- $f - d - e - f$
- $c - g - f - c$
- $c - d - e - c$
- $d - g - e - f - d$
- $d - g - e - f - c - d$
- $d - g - e - c - d$
- $d - g - f - d$

And, some possible feedback arc sets might be:

- $S = \{(b, g), (g, f), (f, c), (c, g), (e, c), (d, e), (d, g)\}$
- $S = \{(b, g), (g, f), (f, c), (c, g), (e, c), (d, e)\}$
- $S = \{(g, f), (f, c), (c, g), (e, c), (d, e), (d, g)\}$
- $S = \{(g, f), (f, c), (c, g), (e, c), (d, e), (g, e)\}$

[1]could also be edgeless
[2]where '$-$' represents an edge

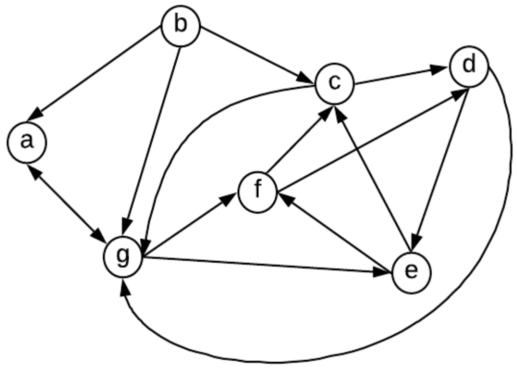


Figure 1: Finding Feedback Arc Sets

We would say that the feedback arc set $\{(b,g),(g,f),(f,c),(c,g),(e,c),(d,e),(d,g)\}$ has *size* 6, and the feedback arc set $\{(g,f),(f,c),(c,g),(e,c),(d,e),(g,e)\}$ has size 5. A trivial feedback arc set for any graph G would be obtained by including all the edges of G in it. We are more interested in finding the smallest feedback arc set:

Feedback Arc Set
Input: A graph $G = (V, E)$
Goal: Return a feedback arc set with smallest size

The above statement of Feedback Arc Set is the optimization version of the problem. We can reframe the problem as a decision problem:

Feedback Arc Set
Input: A graph $G = (V, E)$ and a positive integer k
Goal: Return a feedback arc set with size k

We will now check that the decision version of Feedback Arc Set is in fact a decision problem. In other words we now show that: Feedback Arc Set $\in$ NP

2 Feedback Arc Set $\in$ NP

We need to design an algorithm that takes as inputs an instance of Feedback Arc Set, I, and a proposed solution, S, and checks whether S is a correct solution for I or not.

Definition 1 *Topological Ordering: For a directed graph G, a topological ordering of G is an ordering of its nodes as $v_1, v_2, \ldots v_n$ so that for every edge*

(v_i, v_j), $i < j$. *In other words, all edges point "forward" in the ordering.* [4]

Lemma 2 *If G has a topological ordering, then G is a DAG.*

The usual algorithms for topological sorting have running time linear in the number of nodes plus the number of edges; asymptotically, $O(|V| + |E|)$.

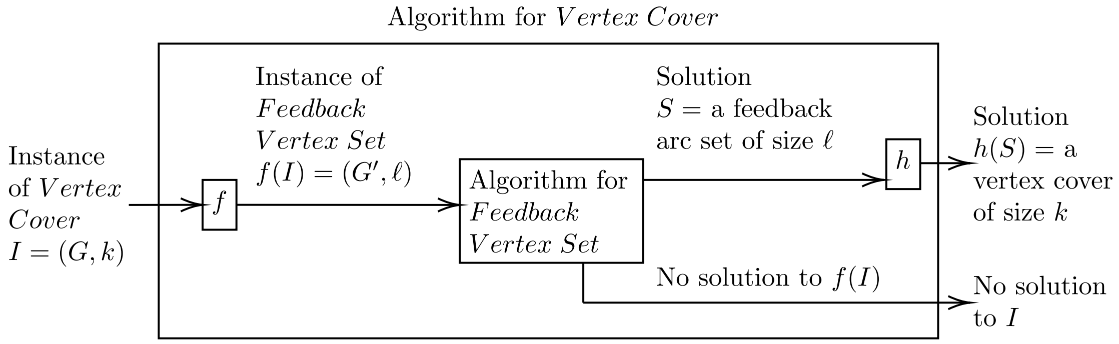
```
CheckFeedbackArcSet($I = (G = (V, E), k), S$)
Input: An instance of feedback arc set, $I$: a graph $G = (V, E)$ and a positive
    integer $k$, along with a proposed solution $S \subseteq E$.
Output: True if $S$ is a solution for $I$, False if it is not.
1: if $|S| = k$ then
2:     if $G' = (V, E \setminus S)$ can be topologically ordered then
3:         return True
4:     return False
5: return False
```

We have given an algorithm, CheckFeedbackArcSet, with running time $O(|V|+|E|)$ (definitely polynomial), that correctly checks if a proposed solution to Feedback Arc Set is in fact a solution. Thus Feedback Arc Set $\in$ NP.

3 Feedback Arc Set $\in$ NP-Complete

The following reduction shows that Feedback Arc Set is NP-Complete.

3.1 Vertex Cover $\leq_p$ Feedback Arc Set


Algorithm for $Vertex\ Cover$
Instance of $Vertex\ Cover$ $I = (G, k)$
f
Instance of $Feedback\ Vertex\ Set$ $f(I) = (G', \ell)$
Algorithm for $Feedback\ Vertex\ Set$
Solution S = a feedback arc set of size ℓ
h
Solution $h(S)$ = a vertex cover of size k
No solution to $f(I)$
No solution to I

An instance of Vertex Cover is a graph $G = (V, E)$ and a positive integer k. We need to transform this into an instance for Feedback Arc Set: a graph $G' = (V', E')$[3] and a positive integer ℓ.

- $f((G = (V, E), k)) = (G' = (V', E'), \ell)$ where the peculiarities of G' are:

[3]The superscripts '*' and ''' mean different things in different sections of this paper.

  - For each vertex v in V, there are two vertices x_v and y_v in V'.
  - Moreover, each pair of vertices x_v and y_v is connected by a directed edge (x_v, y_v).
  - For each edge (u, v) in E, there are two directed edges (y_u, x_v) and (y_v, x_u).
  - Consequently, if $|V| = n$ then $|V'| = 2n$ and $|E'| = n + 2|E|$.

- Finally, $k = \ell$.

Now, suppose we have a solution, S, to FeedBack Arc Set for (G', ℓ), then we will return the set comprising of each edge $e = (x_u, y_u) \in S$ corresponding to each vertex u in $h(S) = S^*$.

- $h(S) = S^* = \{u \in V \mid (x_u, y_u) \in S\}$

This reduction will be correct if the following theorem is true:

Theorem 3 *S is a feedback arc set on $G' = (V', E')$ of size ℓ if and only if $h(S) = S^*$ is a vertex cover on $G = (V, E)$ of size ℓ.* [6] [7]

Proof:

($\Rightarrow$) Suppose S is a feedback arc set on $G' = (V', E')$ and $|S| = \ell$. Without loss of generality we can assume that the only edges that may be removed from E' to make the resulting graph $G^* = (V', E' - S)$ a DAG (i.e., all the edges in S) are of the form (x_u, y_u). Because if some other edge (y_u, x_v) were to have been removed, all cycles in which this edge participated would have included (x_v, y_v), say, and this edge could have been removed instead. The set $S^* = \{u \in V \mid (x_u, y_u) \in S\}$ would then form a vertex cover. Further, for a contradiction, suppose that S^* ($|S^*| = \ell$) is not a node cover on $G = (V, E)$. Since S is not a node cover on G, there must exist some edge, $e = (u, v) \in E$ such that it is not covered by S^*. Then, in G' the cycle (x_u, y_u), (y_u, x_v), (x_v, y_v), (y_v, x_u) is unbroken by the removal of all the edges in S. Therefore S would not be a feedback arc set on G'. $\rightarrow\leftarrow$.

($\Leftarrow$) Suppose S^* is a vertex cover on $G = (V, E)$ and $|S^*| = \ell$. Now, suppose for each vertex $v \in S^*$ we pick the edge (x_v, y_v) from G' and build the set S out of these chosen edges, then the graph $G^* = (V', E' \setminus S)$ will be a DAG and thus S will be a feedback arc set on G'. But, for a contradiction, suppose that G^* (which will of course have $|E'| - \ell$ edges) is not a DAG. Going back to G', if a directed cycle enters a vertex x_u it can only leave by the edge (x_u, y_u). Similarly, a cycle can only enter y_u through the edge (x_u, y_u). Thus if a cycle in G' uses the edge (y_u, x_v) it must also use the edges (x_u, y_u) and (x_v, y_v). But at least one of (x_u, y_u) or (x_v, y_v) should have been removed in G^* as per our construction of G^* and if S^* were truly a vertex cover i.e., a set of vertices that includes at least one endpoint of every edge of the graph. And, since the only cycles possible in G^* are each of the cycles involving the sets of topologically adjacent vertices x_u, y_u, x_v, y_v and since there should at least be one cycle in G^* for it to not be a DAG, it should be the case that at least one of the edges $(u, v) \in E$ is not covered by S^*. Therefore S^* is not a node cover. $\rightarrow\leftarrow$. ■

4 Brute Force Algorithm

```
BruteForceFeedbackArcSet($G = (V, E), k$)
Input: A graph $G = (V, E)$ and a positive integer $k$
Output: A set of $k$ edges, $S$, that form a feedback arc set, or a statement that
    no such feedback arc set exists.
1: for all subsets $R \subseteq E$ of size $k$ do
2:     if $R$ is a feedback arc set then
3:         return $R$
4: return "no such feedback arc set"
```

The algorithm considers all possible subsets of $|E| = m$ edges of size k, there are $\binom{m}{k}$ of these sets. For each of these subsets, R, BruteForceFeedbackArcSet checks if R is a feedback arc set (this can be done using topological sort in time $O(m+n)$ where $|V| = n$). Thus, the total running time is $O((m+n)\binom{m}{k})$.

5 Approximation Algorithm

In order to use an approximation ratio, it is easier to consider approximating a weighted digraph version of FAS. Here we give a λ-approximation algorithm for the optimization version of feedback arc set where λ is the length, in terms of number of arcs, of a longest simple cycle of the digraph. [2] Note that the length is in terms of number of arcs, and therefore independent of the weight function.

The minimum FAS problem consists of finding a smallest cost feedback arc set where the cost of the feedback set can be either its cardinality or its weight with respect to a nonnegative weight function. Here is the modified optimization problem:

Weighted Feedback Arc Set
Input: A digraph $G = (V, E)$ with nonnegative arc weights $w : E \to \mathbb{R}^+$
Goal: Return a minimum weight arc set $S \subseteq E$ such that the directed graph $(V, E \setminus S)$ is acyclic

- Let the weight of a feedback arc set S be denoted by $w(S) = \sum_{(x,y)\in E} w(x, y)$.
- A feedback arc set S^* is optimum if $w(S^*) \leq w(S')$ for any feedback arc set S'.
- A feedback arc set S' is a r-approximation with $r \geq 1$, if $w(S') \leq r \cdot w(S^*)$.

The Weighted Feedback Arc Set problem is the problem of covering all cycles of a given digraph by means of a minimum cost set of arcs. In that case, the following theorem will come in handy:

Theorem 4 *Local Ratio Theorem: 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 an r-approximation with respect to the weight function $w_1 + w_2$.*

In other words, if the payment at each step can be proved to cost no more than r times the optimal payment, then the total payment will be at most r times the optimal cost. Following this approach we may progressively reduce the weights of the arcs of the digraph and add to the FAS the arcs whose weight becomes 0.

Consider the following algorithm:

```
WeightedFeedbackArcSetApproximation($G = (V, E), w : E \to \mathbb{R}^+$)
Input: A digraph $G$ and a nonnegative weight function $w$.
Output: A feedback arc set $S$ of minimal weight.
 1: $S = \emptyset$
 2: while $(V, E \setminus S)$ is not acyclic do                    ▷ Phase 1
 3:     Let $\mathcal{C}$ be a simple cycle in $(V, E \setminus S)$
 4:     Let $(x, y)$ be a minimum weight arc in $\mathcal{C}$ and let $\varepsilon$ be its weight
 5:     for all $(u, v) \in \mathcal{C}$ do
 6:         $w(u, v) = w(u, v) - \varepsilon$
 7:         if $w(u, v) = 0$ then
 8:             $S = S \cup \{(u, v)\}$
 9: for all $(u, v) \in S$ do                                       ▷ Phase 2
10:     if $(V, (E \setminus S) \cup \{(u, v)\})$ is acyclic then
11:         $S = S \setminus \{(u, v)\}$
12: Return S
```

5.1 Algorithmic Analysis

Intuition: The algorithm consists of two phases. First, it looks for a simple cycle $\mathcal{C}$ in the digraph. If such a cycle exists, it 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 weights become 0 are removed. If the digraph is now acyclic the first phase terminates, otherwise it is 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, taking care not to re-introduce cycles. The set of removed arcs is finally returned. All these steps are demonstrated in figure 2.

WeightedFeedbackArcSetApproximation roughly tries 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. 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.

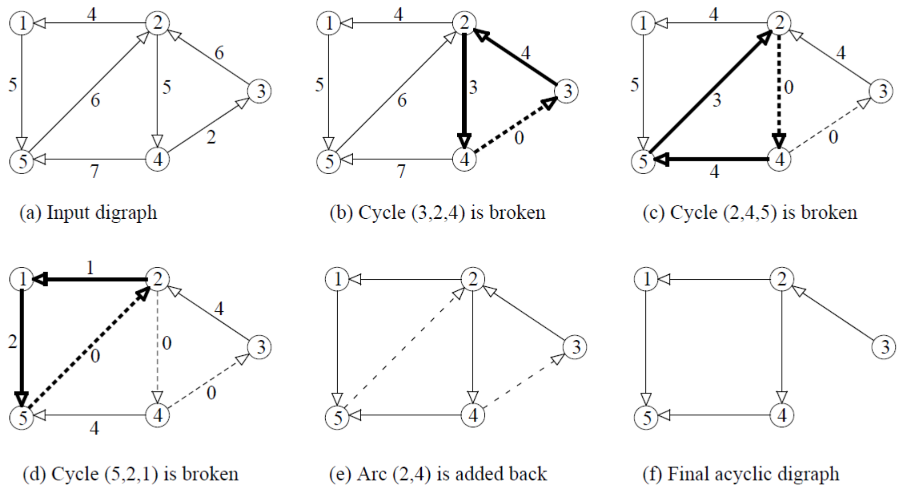

(a) Input digraph
(b) Cycle (3,2,4) is broken
(c) Cycle (2,4,5) is broken
(d) Cycle (5,2,1) is broken
(e) Arc (2,4) is added back
(f) Final acyclic digraph

Figure 2: An illustration of WeightedFeedbackArcSetApproximation

- Is the algorithm *feasible*? Does it return a feedback arc set?
- How close is the size of the feedback arc set that the algorithm gives to the size of the optimal FAS?

We'll answer the question of feasibility first:

Theorem 5 *Let $G = (V, E, w)$ be a weighted directed graph with n vertices and m arcs.* WeightedFeedbackArcSetApproximation *finds a minimal feedback arc set of G in $O(m \cdot (m + n))$ worst-case running time.*

Proof:

Correctness: WeightedFeedbackArcSetApproximation progressively removes arcs from the input digraph, stopping only when the remaining arcs do not form cycles (lines 2-8). 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 9-11): in this phase S remains a feedback arc set because the acyclicity condition is tested before any arc addition (line 10).

Runing 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 3), a minimum weight arc in the cycle is identified (line 4), and the weights of all arcs in the cycle are updated (lines 5-8). 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 straight-forward implementation of the first operation (by means of a visit), would yield $O(m \cdot (m+n))$ overall running time. ■

Now we will show that WeightedFeedbackArcSetApproximation gives a λ-approximation for feedback arc set.

We denote with w, w_1, w_2 different nonnegative weight functions for the arcs of a digraph $G = (V, E)$. Moreover, let S^*, S_1^*, S_2^* be the minimum feedback arc sets of the weighted digraphs $(V, E, w), (V, E, w_1), (V, E, w_2)$ 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, w_2 when these functions are linearly dependent:

Lemma 6 *Let $G = (V, E)$ be a directed graph and let w, w_1, w_2 be three non-negative weight functions on the arcs of G such that $w = w_1 + w_2$. Then it holds:*

$$w_1(S_1^*) + w_2(S_2^*) \leq w(S^*)$$

Proof: Since $w = w_1 + w_2$, we have that $w(S^*) = w_1(S^*) + w_2(S^*)$. Moreover, S^* 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 $w_1(S^*) \geq w_1(S_1^*)$ and $w_2(S^*) \geq w_2(S_2^*)$. The claim immediately follows. ■

Theorem 7 *Let $G = (V, E, w)$ be a weighted directed graph.* WeightedFeedbackArcSetApproximation *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 WeightedFeedbackArcSetApproximation 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 2 of the algorithm. 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 3), 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 $G = (V, E)$ as follows:

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

Observe that the cost of the minimum feedback arc set of the digraph $G = (V, E)$ with respect to w_1 is equal to ε, i.e., $w_1(S_1^*) = \varepsilon$: 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 S w.r.t w_1 cannot be greater than $k \cdot \varepsilon \leq \lambda \cdot \varepsilon$, because all the arcs not belonging to $\mathcal{C}$ cost 0 and at most all the arcs in $\mathcal{C}$ can participate to S. Therefore, for any S:

$$w_1(S) \leq \lambda \cdot w_1(S_1^*)$$

In the following we denote with S_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, $S_1 = \{(u,v) \in \mathcal{C}\ such\ that\ w(u,v) = \varepsilon\}$. Since any arc $(u,v) \in \mathcal{C}$ has weight $w_1(u,v)$ equal to ε, it holds:

$$w(S_1) = w_1(S_1)$$

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 $(V, E \setminus S_1, w_2)$, WeightedFeedbackArcSetApproximation returns a feedback arc set S_2 of this digraph such that:

$$w_2(S_2) \leq \lambda \cdot w_2(S_2^*)$$

.

Let S be the feedback arc set returned by WeightedFeedbackArcSetApproximation on the weighted digraph $G = (V, E, w)$. As far as the algorithm is concerned, S consists both of the arcs in S_1 and of the arcs in S_2, i.e., $S = S_1 \cup S_2$. It is also worth pointing out that $S_1 \cap S_2 = \emptyset$, due to the fact that once an arc has been removed, it will be no longer be considered by the algorithm. By linearity of w, w_1, w_2, it holds $w(S) = w_1(S) + w_2(S)$. In conclusion:

$$w(S) =$$

$$\triangleright\ w = w_1 + w_2$$

$$w_1(S) + w_2(S) =$$

$$\triangleright\ S = S_1 \cup S_2$$

$$w_1(S) + w_2(S_1) + w_2(S_2) - w_2(S1 \cap S2) =$$

$$\triangleright\ S_1 \cap S_2 = \emptyset$$

$$w_1(S) + w_2(S_1) + w_2(S_2) =$$

$$\triangleright\ w_2 = w - w_1 \text{ and } w(S_1) = w_1(S_1)$$

$$w_1(S) + w_2(S_2) \leq$$

$$\triangleright\ w_1(S) \leq \lambda \cdot w_1(S_1^*)$$

$$\lambda \cdot w_1(S_1^*) + w_2(S_2) \leq$$

$$\triangleright\ \text{Inductive Hypothesis}$$

$$\lambda \cdot w_1(S_1^*) + \lambda \cdot w_2(S_2^*) \leq$$

$$\triangleright\ \text{Lemma 6}$$

$$\lambda \cdot w(S^*)$$

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

■

6 Applications/Other Interesting Things

- There are several practical applications of FAS including the analysis of large-scale systems with feedback, graph layout [8], and certain scheduling problems [3].
- The minimum feedback arc set problem on undirected graphs can be easily solved in polynomial time by finding a maximum weight spanning tree. Moreover, although its directed version is NP-complete even on digraphs with total vertex in-degree and out-degree smaller than 3, yet it is polynomial-time solvable on planar digraphs [5].

References

[1] Feedback arc set. *Wikipedia*, May 2020. Page Version ID: 954214673.

[2] Camil Demetrescu and Irene Finocchi. Combinatorial algorithms for feedback problems in directed graphs. *Information Processing Letters*, 86(3):129–136, May 2003.

[3] Merrill 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):1–23, 1990. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.3230200102.

[4] Jon Kleinberg and Éva Tardos. *Algorithm Design*. Pearson/Addison-Wesley, 2006. Google-Books-ID: OiGhQgAACAAJ.

[5] C. L. Lucchesi and D. H. Younger. A Minimax Theorem for Directed Graphs. *Journal of the London Mathematical Society*, s2-17(3):369–374, 1978. _eprint: https://londmathsoc.onlinelibrary.wiley.com/doi/pdf/10.1112/jlms/s2-17.3.369.

[6] University of Pennsylvania. Final Exam Solutions. *CIS 502 - Algorithms*, page 1, 2003.

[7] University of Toronto. Assignment 4: Due Friday August 9, Noon - Solutions. *CSC 373 - Algorithm Design, Analysis, and Complexity*, pages 1–3, 2014.

[8] Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for Visual Understanding of Hierarchical System Structures. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(2):109–125, February 1981. Conference Name: IEEE Transactions on Systems, Man, and Cybernetics.