*AMO - Advanced Modeling and Optimization, Volume 7, Number 1, 2005*

# An Algorithm to find a Minimum Feedback Vertex Set of an Interval Graph

Anita Saha and Madhumangal Pal [1]

Department of Applied Mathematics with Oceanology and Computer Programming,
Vidyasagar University, Midnapore – 721 102, India.
e-mail: mmpal@vidyasagar.ac.in

**Abstract.** In an undirected graph, the feedback vertex set problem is to find a set of vertices of minimum cardinality whose removal makes the graph acyclic (connected or disconnected). This problem is known to be NP-hard for general graph. In this paper, we proposed two algorithms for finding the minimum feedback vertex set on interval graphs. The first algorithm is for unweighted case and takes $O(n + m)$ time while the second one is for weighted case taking $O(n\sqrt{\log C})$ time where $n$ is the number of vertices, $m$ is the number of edges and $C$ is the cost of the longest path of the graph.

**Keywords:** Design and analysis of algorithms, feedback vertex set, network flow, interval graphs,

**AMS Subject Classifications:** 68Q22, 68Q25, 68R10.

## 1 Introduction

An undirected graph $G = (V, E)$ is an *interval graph* if the vertex set $V$ can be put into one-to-one correspondence with a set $I$ of intervals on the real line such that two vertices are adjacent in $G$ iff their corresponding intervals have non-empty intersection. The set $I$ is called an interval representation of $G$ and $G$ is referred to as the intersection graph of $I$ [9].

Interval graphs arise in the process of modeling real life situations, specially involving time dependencies or other restrictions that are linear in nature. This graph and various subclass

[1] This work has been done as a part of the project sponsored to the second author by Department of Science and Technology, India, under grant No. SR/FTP/ETA-008/2002.

thereof arise in diverse areas such as archeology, molecular biology, sociology, genetics, traffic planning, VLSI design, circuit routing, psychology, scheduling, transportation and others. Recently, interval graphs have found applications in protein sequencing [11], macro substitution [7], circuit routine [14], file organization [3], job scheduling [3], routing of two points nets [10] and many others. An extensive discussion of interval graphs also appears in [9]. Thus interval graphs have been studied intensely from both the theoretical and algorithmic point of view.

In the following we give some useful definitions. Let $G = (V, E)$ be an undirected graph and let $|V| = n$ and $|E| = m$. A *cycle* $C$ of $G$ is a sequence of distinct vertices $\{v_1, v_2, \ldots, v_r\}$ such that $(v_i, v_{i+1}) \in E$ for $1 \leq i < r$ and $(v_r, v_1) \in E$. Given $S \subseteq V$, the induced subgraph $G(S)$ is the graph $G(S) = (S, E(S))$, where $E(S) = \{(v_i, v_j) : v_i, v_j \in S, (v_i, v_j) \in E\}$. A set $S$ is a *feedback vertex set* if and only if the graph $G(V - S)$ has no cycles. The *minimum feedback vertex set* (MVFS) problem is "Given $G = (V, E)$, find a feedback vertex set $S$ such that its cardinality $|S|$ is minimum among all such sets". Let $I(V)$ be the interval representation of the interval graph $G = (V, E)$. For weighted undirected graph, an independent set is called a *maximum weight independent set* if the sum of the weights of the vertices of the independent set is maximum among all the independent sets of $G$. The *maximum weight 2-independent set* (MW2IS) problem on $G$ is to find a set of two disjoint independent sets $S_1, S_2$ of $G$ such that its weight $\sum_{v \in Q_2} wt(v)$ is maximum, where $Q_2 = S_1 \cup S_2$ and each vertex $v \in V$ has a positive weight $wt(v)$.

The feedback vertex set problem is important in the study of many large scale systems with feedback such as electrical circuits. A frequently used technique in the analysis of such a system is to model the structure of the system of a graph and make the system free from feedback by removing a small set of edges/vertices to make it feedback set. It is empirically observed that the convergence is faster if the feedback vertex edge set is of small size. Thus we are interested in finding a feedback vertex/edge set which has minimum number of vertices/edges. Also, an algorithm for the minimum feedback vertex/edge set is of some importance.

The minimum feedback vertex set problem is NP-hard on general graphs [8, 12]. An $O(mn^2\log(n^2/m))$ algorithm for the weighted feedback vertex set problem on flow-reducible graphs is given in [16]. The feedback vertex set problem on chordal and interval graphs can be viewed as special instances of the generalized clique cover problem which are solved in polynomial time for chordal graphs [5, 17] and interval graph [13]. An $O(n^6)$ algorithm is there for permutation graphs [2] and an $O(n^4)$ algorithm exists for the minimum weighted feedback

vertex set problem exists on cocomparability graphs [4].

In this paper, an efficient algorithm is presented to find minimum feedback vertex set on interval graphs in $O(n+m)$ time for cardinality case. Also an algorithm is presented to find the same in $O(n\sqrt{\log C})$ time for weighted case where $C$ is the weight of the longest path of the graph.

## 2 Data Structure and Preliminaries

Let $I = \{I_1, I_2, \ldots, I_n\}$, be the interval representation of an interval graph $G$, where $a_c$ is the left endpoint and $b_c$ is the right endpoint of the interval $I_c$=$[a_c, b_c]$ for all $c = 1, 2, \ldots, n$. Without any loss of generality we assume the following:

1. the intervals in $I$ are indexed by increasing right end points i.e., $b_1 < b_2 < \cdots < b_n$,
2. the intervals are closed i.e., contains both of its endpoints and that no two intervals share a common endpoint,
3. vertices of the interval graph and the intervals on the real line are one and the same thing,
4. the interval graph $G$ is connected, and the list of sorted endpoints is given.

To find MFVS for unweighted interval graph, we first compute all cliques. A brief description is given below to obtain such cliques.

To find the cliques we consider fictitious vertical lines only at each '$b$' endpoints to consider the intervals that are cut by these vertical lines. Between two consecutive '$b$' endpoints, either there will be no '$a$' endpoint or there will remain some '$a$' endpoints. Thus if a vertical line is drawn at any point in between two consecutive '$b$' endpoints then the number of intervals cutting that vertical line is either same or less than the number of intervals that are cut by the vertical line drawn at the right '$b$' endpoints. Thus all maximal cliques are obtained by considering vertical lines only at the '$b$' endpoints.

For each $i\,(= 1, 2, \ldots, n)$ let $B_i$ be the set of vertices corresponding to the interval $i$ and the intervals that are cut by the fictitious line considered at $b_i$. Using algorithm [15] all maximal cliques are obtained. The minimum vertex of each such clique is noted and arranged in increasing order, say, $p_1, p_2, \ldots, p_k$ where $k$ is the total number of cliques. Let $S_j$ be the set of vertices forming the clique whose minimum element is $p_j$. Thus, the maximal cliques are $S_1, S_2, \ldots, S_k$.

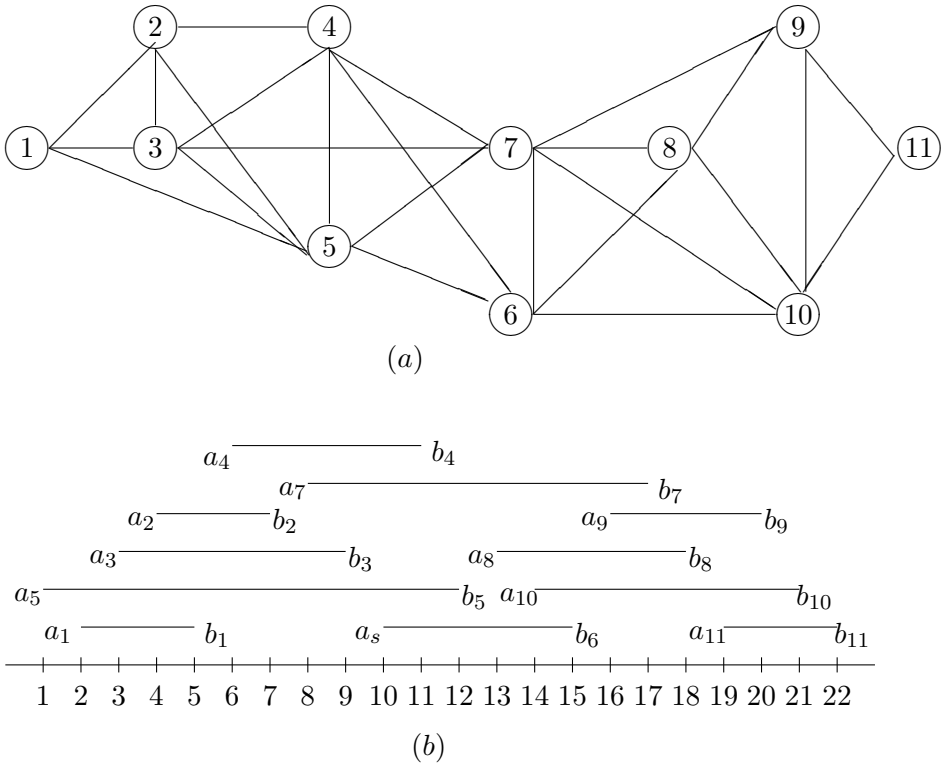


Figure 1: (a) An interval graph and (b) its interval representation

Obviously, $S_j = B_{p_j}$. All vertices in $B_{p_j}$ are arranged in increasing order for each $j$. Let the cardinality of $S_j$ be $N_j$ for each $j = 1, 2, \ldots, k$. Let $C_1$ be the set of all vertices of $B_{p_1}$ other than first two. Let $C_2$ be the set of vertices whose elements are the union of $C_1$ and $B_{p_1} - C_1$ except first two elements of $B_{p_1} - C_1$. In general, $C_i = C_{i-1} \cup \{B_{p_i} - C_{i-1} - \{x_1, x_2\}\}$, where $x_1$ and $x_2$ be the first two vertices of $B_{p_i} - C_{i-1}$. Then the feedback vertex set is $F = C_k$. For each $j = 1, 2, \ldots, k$, let $S_j(l)$ be the $l$th member of the $j$th maximal clique, where $l = 1, 2, \ldots, N_j$.

For the graph of Figure 1, the maximal cliques, the vertices of which are put in ascending order, are $B_{p_1} = S_1 = \{1, 2, 3, 5\}$, $B_{p_2} = S_2 = \{2, 3, 4, 5\}$, $B_{p_3} = S_3 = \{3, 4, 5, 7\}$, $B_{p_4} = S_4 = \{4, 5, 6, 7\}$, $B_{p_5} = S_5 = \{6, 7, 8, 10\}$, $B_{p_6} = S_6 = \{7, 8, 9, 10\}$, $B_{p_7} = S_7 = \{9, 10, 11\}$ where $p_1 = 1, p_2 = 2$, $p_3 = 3, p_4 = 4$, $p_5 = 6, p_6 = 7, p_7 = 9$. The cardinality $N_j$ of $S_j$ for this graph are $N_1 = 4, N_2 = 4, N_3 = 4$, $N_4 = 4, N_5 = 4, N_6 = 4, N_7 = 3$. Also, $S_1(3) = 3$, $S_1(4) = 5, S_2(2) = 3$ and so on. For this graph $C_1 = \{3, 5\}$, $C_2 = \{3, 5\}$, $C_3 = \{3, 5\}$, $C_4 = \{3, 5, 7\}$, $C_5 = \{3, 5, 7, 10\}$, $C_6 = \{3, 5, 7, 10\}$ and $C_7 = \{3, 5, 7, 10\}$.

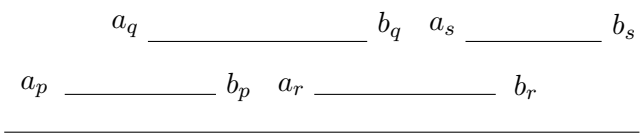


Figure 2: Proof of Lemma 1

For feedback vertex set of the interval graph $G = (V, E)$ we are to collect the set $F$ of vertices such that the interval graph $G(V - F)$ becomes cycle free and this happens when its interval representation $I(V - F)$ (I(V) is the interval representation of the interval graph $G = (V, E)$) has at most two intervals intersecting this fictitious vertical line at any position on the real line.

Following lemma gives the upper bound on the length of cycles of an interval graph.

**Lemma 1** *Let $G = (V, E)$ be an interval graph then it can have cycles only of length three.*

**Proof.** If possible let there be a cycle $[v_p, v_q, v_r, v_s]$ of length four in an interval graph $G$. Let $b_p < b_q < b_r < b_s$. As $[v_p, v_q, v_r, v_s]$ is a cycle we have $(v_p, v_q), (v_q, v_r)$,$(v_r, v_s)$ and $(v_s, v_p) \in E$. Thus

$$b_p < a_r \text{ as } (v_p, v_r) \notin E$$
$$a_r < b_q \text{ as } (v_q, v_r) \in E$$
$$b_q < a_s \text{ as } (v_q, v_s) \notin E.$$

Therefore $b_p < a_r < b_q < a_s$ *i.e.*, $b_p < a_s$. This means $(v_s, v_p) \notin E$. This is a contradiction as $(v_s, v_p) \in E$. Similar proof holds for any cycle of length greater than four. □

**Theorem 1** *The set $F$ is a feedback vertex set.*

**Proof:** From definition of feedback vertex set, the set $F$ is a feedback vertex set if and only if the graph $G(V - F)$ has no cycles. Let $I(V - F)$ be the set of intervals corresponding to the interval graph $G(V - F)$. From the construction of the set $F$, the set $I(V - F)$ has at most two intersecting intervals at any position on the real line. So $G(V - F)$ is cycle free. Hence, the set of $F$ is a feedback vertex set. □

The cardinality of minimum feedback vertex set starts from zero. In the following we show that for the interval graph the minimum feedback set may be even empty. The graph of Figure

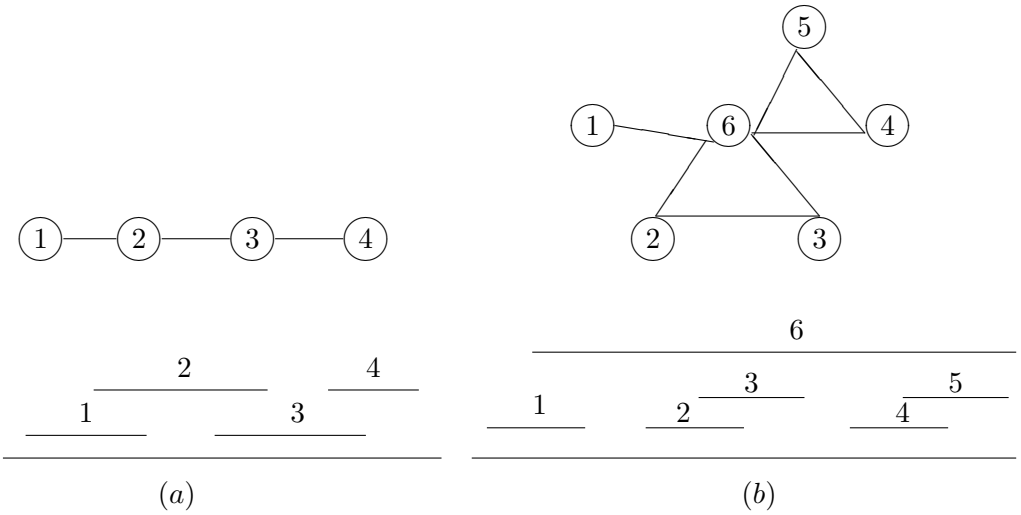


Figure 3: (a) A graph with empty MFVS (b) A graph with one MFVS

3(a) has no member in MFVS so it is an empty set and the graph Figure 3(b) has only one member in MFVS *viz.* $\{6\}$.

Thus, for the graph of Figure 3(a), $F = \phi$ and for the graph of Figure 3(b), $F = \{6\}$. It will be noted that, a graph may have multiple MFVS. Figure 4 gives this result.

Using the algorithm presented here MFVS for the graph correspond to the Figure 4 is $\{4, 5\}$. But it has other MFVS also and they are $\{2, 5\}, \{2, 3\}, \{2, 4\}$. The algorithm of finding MFVS presented here gives only one set.

The following arguments shows that feedback vertex set $F$ obtained by the method is minimum.

For the selection of members of feedback vertex set $F$ we consider the vertex corresponding to the interval whose length is extended most to the right of '$b$' and are connected to the vertex corresponding to the right end point. The selection of intervals which are extended most to the right makes the feedback vertex set minimum. This is justified in the following example of Figure 5.

At '$b_1$' the vertical line meets the intervals $I_2, I_5$ and $I_6$. Here the interval $I_6$ is extended most to the right and next right extended interval is $I_5$. So if we remove $I_5$ and $I_6$ from this interval representation then the reduced interval representation have at most two intervals at any position. So, here $\{5, 6\}$ is the feedback vertex set. But if we take the intervals $I_2$ and $I_5$ at '$b$' position instead of $I_5, I_6$ then another interval $I_6$ is necessary at '$b_3$' for the reduction

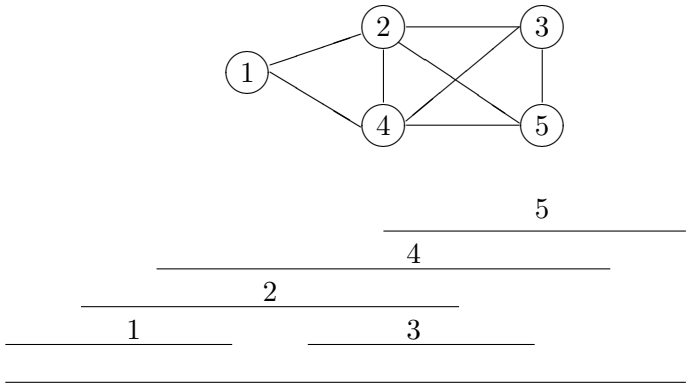


Figure 4: A graph of multiple MFVSs

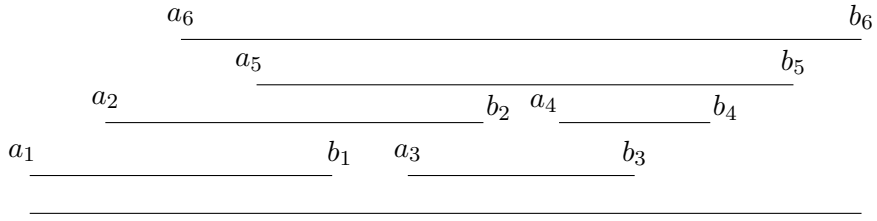


Figure 5: An illustration

to cycle free graph. Thus the feedback vertex set becomes $\{2, 5, 6\}$ which is not minimum. So for minimum feedback vertex set at each right endpoint we are to take the vertices whose corresponding intervals are extended most to the right.

## 3 Algorithm and its Complexity

For an interval graph and its corresponding interval representation we compute all maximal cliques and then arrange the vertices of the cliques in ascending order. We compute the cardinality of each maximal clique. Now to compute the sets $C_1, C_2, \ldots, C_k$, efficiently, we compute a Boolean array *mark* using the algorithm MARK.

Depending on the result of the theorem presented in Section 2 the major steps of the proposed algorithm SMFVS to form a minimum feedback vertex set of interval graph are listed below.

**Algorithm SMFVS**

**Input:** The endpoints $a_i, b_i$ of each interval $I_i, i = 1, 2, \ldots, n$.

**Output:** The minimum feedback vertex set $F$. Initially, $F = \phi$.

**Step 1:** Compute all maximal cliques $S_j$. Let $k$ be total number of maximal cliques.

**Step 2:** In each $S_j$, put the vertices in ascending order.

**Step 3:** Compute $N_j$ for each $j = 1, 2, \ldots, k$ where $N_j$ is the cardinality of $S_j$.

**Step 4:** Compute the array $mark(i)$, for $i = 1, 2, \ldots, n$, initially $mark(i) = 0$.

**Step 5:** Put the vertex $i$ to the set $F$ if $mark(i) = 1$, $i = 1, 2, \ldots, n$.

**end SMFVS**

The array $mark(\cdot)$ is computed using the following algorithm.

**Algorithm MARK**

```
For i = 1 to k do
        count= 0;
        For j = 1 to N_i do
            If (count= 2) then
                mark(j) = 1;
            else if (mark(S_i(j)) = 0) then
                count= count + 1;
            else
                mark(S_i(j)) = 1;
            endif;
        endfor;
endfor;
```

**end MARK**

**Lemma 2** *Time to compute the array $mark(\cdot)$ is $O(n + m)$.*

**Proof.** The time complexity to compute the array $mark$ is $\sum_{i=1}^{k} N_i$ which is the sum of cardinality of all maximal cliques and is equal to $O(n + m)$ [9]. □

Using algorithm SMFVS we see that for the interval graph of Figure 1, $mark(3) = 1, mark(5) = 1$, $mark(7) = 1$ and $mark(10) = 1$, so the minimum feedback vertex set for the graph is $F = \{3, 5, 7, 10\}$.

Following lemma gives the time complexity to compute the MFVS for an interval graph with $n$ vertices and $m$ edges.

**Lemma 3** *The time complexity of Algorithm SMFVS is $O(n + m)$.*

**Proof.** Pal et al. [15] have given an $O(n + m)$ time algorithm to compute all maximal cliques of an interval graph with $n$ vertices and $m$ edges. So Step 1 can be computed in $O(n+m)$ time. Each of Step 2 and Step 3 takes $O(n + m)$ time. Step 4 takes also $O(n + m)$ time to compute the array $mark(i)$ and Step 5 takes $O(n)$ time. So, overall time complexity of the algorithm SMFVS is $O(n + m)$. □

From the above lemma we have the following result.

**Theorem 2** *The minimum cardinality feedback vertex set of an interval graph with $n$ vertices and $m$ edges can be computed in $O(n + m)$ time.*

In the following we present an algorithm to compute minimum feedback vertex set on weighted interval graphs. MFVS problem on weighted interval graph is solved by constructing an appropriate network. Before presenting the algorithm we defining some related terms such as network, network flow problem etc.

# 4 The Network Flow Problem

To solve minimum weight feedback vertex set problem, we introduce a special network to find maximum weight 2-flow problem of the weighted interval graph $G$. This network and the network flow problem is defined as follows.

**Definition 1** *A network $N$ is a finite set of nodes, where node 0 is called the source in $N$ and node $n$ is called the sink in $N$. The set of all arcs of $N$, is denoted by $E_N$.*

**Definition 2** *For each arc $(u, v) \in E_N$ there is a given number $c(u, v) > 0$ called the capacity of the arc $(u, v)$.*

**Definition 3** *A non-negative function $f(u,v)$ ranging over all arcs $(u,v) \in E_N$, is called a flow in $N$ if*
*(i) for every $(u,v) \in E_N$, $f(u,v) \leq c(u,v)$, and*
*(ii) for every node $u$,*

$$\sum_v f(u,v) - \sum_v f(v,u) = 0$$

*where each sum is over every $v$ for which the summation is meaningful.*

For a network $N$, with each arc $(u,v) \in E_N$, a non-negative weight $w_N(u,v)$ as well as a positive capacity $c(u,v)$ is associated. For each arc $(u,v)$ of $E_N$, $f(u,v)$ represents the flow in the arc $(u,v)$.

The minimum weight 2-flow problem is defined for a network $N$ as follows:

The minimum weight 2-flow problem is to obtain two disjoint paths $P_1$ and $P_2$ from the set of all possible paths from 0 to $n$ in $N$ to

$$minimize \sum_{(u,v)\in J_2} w_N(u,v) f(u,v)$$

where $J_2 = \cup_{i=1}^{2} E_N^i$ and $E_N^i$ is the set of arcs associated with the path $P_i$.

## 5 Construction of Network

Let $N = (V_N, E_N)$ be the network where $V_N = \{0, 1, \ldots, n\}$ and $E_N = E'_N \cup E''_N$ with $E'_N = (i, i+1), i = 0, 1, \ldots, n-1$, having weight zero, called as *d-arcs*. Each of these arcs is directed form $i$ to $i+1$. Now find $l(i)$ defined by $l(i) = k$, for $i = 1, 2, \ldots, n$, where $k$ is the vertex maximum among all vertices non-adjacent to $i$ $(k < i)$. Now we construct $E''_N = (l(i) - 1, i)$, $i = 1, 2, \ldots, n$ with weight $w((l(i) - 1, i))$ equal to the weight of the vertex $i$, called as *v-arcs*. Each of these arcs is directed from $l(i) - 1$ to $i$. Finally $E_N = E'_N \cup E''_N$. Figure 6(a) is weighted interval graph and Figure 6(b) is the corresponding network.

We now consider the maximum 2-flow problem $P^*$ on the network $N$.

**Problem P$^*$:** The maximum network 2-flow problem is to find two disjoint paths $P_1, P_2$ from the set of all possible paths from 0 to $n$ in $N$ to

$$maximize \sum_{(u,v)\in J_2} w_N(u,v) f(u,v)$$

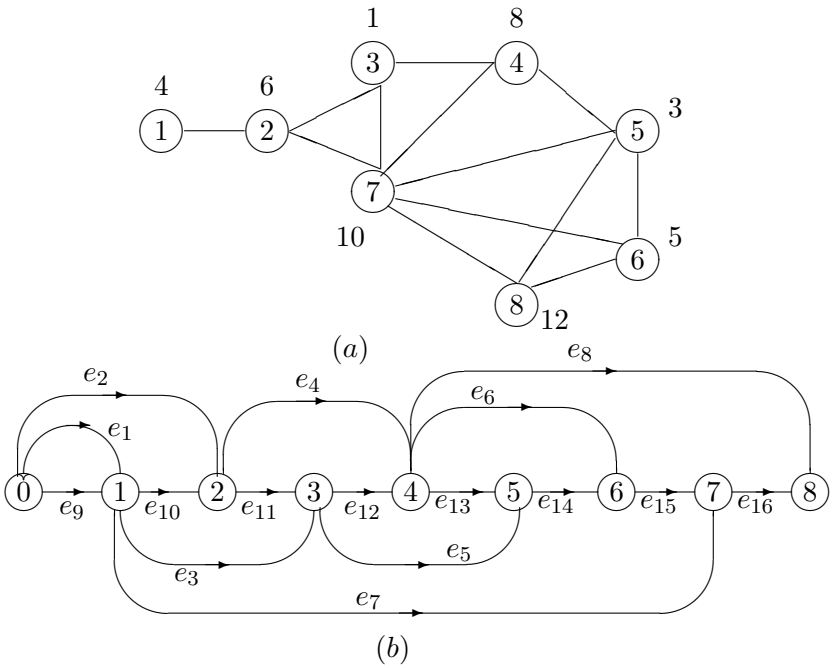


Figure 6: (a) A weighted interval graph (the number outside the circle represents the weight of the corresponding vertex), (b) Corresponding network N.

where

(i) $J_2 = \cup_{i=1}^{2} E_N^i$,

(ii) $E_N^i$ is the set of arcs associated with the path $P_i$, and

(iii) the value of $f(u, v)$ is either 0 or 1 for all $(u, v) \in E_N$.

## 6 Properties of the Network

The size of $V_N$ is $|V_N| = |V| + 1 = n + 1$. Again for each node associated with $V_N$ except '0' node two arcs end at $i$. Obviously the total number of such arcs is of $O(n)$. As there are $n$ nodes in $N$ except '0' node, the size of set $E'_N$ must be $n$ *i.e.*, $O(n)$ and size of $E''_N$ is $O(n)$. Therefore, the total number of arcs in the network $N$ is of $O(n)$. Hence we have the following results.

**Lemma 4** *The total number of nodes and arcs of $N$ are respectively $n + 1$ and $O(n)$.*

The following lemma is obvious from the construction of the network $N$ which gives the relationship between the arcs of the network $N$ and the vertices of the interval graph $G$.

**Lemma 5** *For any path in $N$, the two vertices at which any two $v$-arcs ends are disjoint.*

**Lemma 6** *The total weight of any 2-independent set of $G$ and the total weight of the arcs of the corresponding two-paths of $N$ are same.*

**Proof:** The construction of the network $N$ from the weighted graph $G$ shows that there is one to one correspondence between the set of vertices of $G$ and the set of $v$-arcs of $N$. According to the construction of $N$ each $v$-arc $e_i$ corresponds to the vertex $i \in V$ of $G$. The arc $e_i$ ends at $i$ and starts from $j$ where $j$ is the vertex which is maximum among all non-adjacent vertices to $i$ $(j < i)$. Hence each path in $N$ corresponds to an independent set of $G$ and conversely, each independent set of $G$ corresponds to a path from 0 to $n$ in $N$. Further, we note that the weight of any $d$-arc is zero and the weight of any $v$-arc is same as the weight of the corresponding vertex of $G$.

Let $S_1$ and $S_2$ be any two independent sets of $G$ they correspond respectively to the set of paths $P_1$ and $P_2$ of Problem P$^*$. We take $f(u,v) = 1$ for each arc associated with these two paths and $f(u,v) = 0$ otherwise.

Therefore,

$$\begin{aligned}
\sum_{x \in Q_2} wt(x) &= \sum_{i=1}^{2} \sum_{x \in S_i} wt(x) = \sum_{i=1}^{2} \sum_{(u,v) \in E_N^i} w_N(u,v) \\
&\quad [\text{as } w_N(u,v) = wt(x) \text{ for } v - arc \text{ and} \\
&\quad w_N(u,v) = 0 \text{ for } d - arc] \\
&= \sum_{i=1}^{2} \sum_{(u,v) \in E_N^i} w_N(u,v) f(u,v) \\
&\quad [\text{since } f(u,v) = 1, \text{ for all } (u,v) \in E_N^i, \text{ for all } i \text{ and} \\
&\quad f(u,v) = 0 \text{ otherwise}] \\
&= \sum_{(u,v) \in J_2} w_N(u,v) f(u,v)
\end{aligned}$$

Hence the lemma. □

From this result the following lemma is obvious.

**Lemma 7** *The vertices associated with the solutions of maximum weight 2-flow problem and MW2IS problem of $G$ are same.*

From the lemmas 6 and 7 we have the following result.

**Theorem 3** *MW2IS for $G$ and Problem $P^*$ for $N$ are equivalent.*

The minimum weight 2-flow for $N$ can be solved using the algorithm of Edmonds and Karp [6]. Thus we could possibly modify $N$ by taking negative of the weight of each arc and then finding a minimum weight 2-flow. But this apparently easy conversion is not possible here because in their algorithm weights of all arcs in the network are required to be non-negative.

In order to convert a maximum weight flow problem (Problem P$^*$) to a minimum weight flow problem with non-negative arc weights, we define an array $d$ as follows.

$d(i)$ = largest weight of a path from $i$ to $n$ in $N$, $i \in V_N$.

Using the following algorithm the array $d$ is computed for all $i \in V_N$.

**Algorithm DISTANCE**
**Input:** The network $N$.
**Output:** The array $d(i), i \in V_N$.
**Initialization:** $d(i) = 0, i \in V_N$.
**for** $i = n + 1$ **to** 0 step $-1$ **do**
    **for** each arc $(i, j) \in E_N$ **do**
        $d(i) = max\{d(i),\ w_N(i, j) + d(j)\}$;
    **endfor;**
**endfor;**
**end DISTANCE**

The following lemma gives the time complexity of the Algorithm **DISTANCE**.

**Lemma 8** *The array $d$ is correctly computed in $O(n)$ time.*

**Proof:** Let $m_i$ be the total number of arcs adjacent to $i$. From Algorithm **DISTANCE** it follows that the time complexity of this algorithm is $\sum_{i=1}^{n} m_i$ = total number of arcs of $N = O(n)$ (Lemma 4). As the network $N$ is directed and acyclic, the correctness follows from the algorithm directly. Hence the lemma follows. □

The array $d$ for the network $N$ of Figure 6(b) are $d(0) = 26$, $d(1) = 20$, $d(2) = 20$, $d(3) = 12$, $d(4) = 12$, $d(5) = 0$, $d(6) = 0$, $d(7) = 0$, $d(8) = 0$.

## 7 Construction of a Minimum Weight Flow Network

Now, we convert the problem $P^*$ to a minimum weight flow problem $P^{**}$ as follows:
A network $N^U$ is constructed from $N$ using the same set of arcs $E_N$, the same set of nodes $V_N$ and identical capacities, but different weights. The weight on the arc $(i,j)$, $i<j$, is taken as

$$w_{N^U}(i,j) = d(i) - d(j) - w_N(i,j),$$

for all $(i,j) \in E_N$.

The problem $P^{**}$ may be stated as:

**Problem P$^{**}$:** For the network $N^U$ find the set of two disjoint paths $P_1$ and $P_2$ from the set of all possible paths from 0 to $n$ in $N^U$ to

$$minimize \sum_{(u,v)\in J_2} w_N(u,v) f(u,v)$$

where

(i) $J_2 = \cup_{i=1}^{2} E_N^i$,

(ii)$E_N^i$ is the set of arcs associated with the path $P_i$,

(iii) the value of $f(u,v)$ is either 0 or 1 for all $(u,v) \in N^U$.

Table 1 shows the weights of each arc of the networks $N$ and $N^U$.

| arcs | $e_1$ | $e_2$ | $e_3$ | $e_4$ | $e_5$ | $e_6$ | $e_7$ | $e_8$ | $e_9$ | $e_{10}$ | $e_{11}$ | $e_{12}$ | $e_{13}$ | $e_{14}$ | $e_{15}$ | $e_{16}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| weights in $N$ | 4 | 6 | 1 | 8 | 3 | 5 | 10 | 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| weights in $N^U$ | 2 | 0 | 7 | 0 | 9 | 7 | 10 | 0 | 6 | 0 | 8 | 0 | 12 | 0 | 0 | 0 |

Table 1: The arc weights of the networks $N$ and $N^U$

From Table 1, it may be observed that the weights of all arcs in $N^U$ are non–negative integers. This result is proved in general in the following lemma.

**Lemma 9** *The weight $w_{N^U}(i,j), i<j$ for all $(i,j) \in E_N$ are non-negative.*

**Proof:** We prove this by the method of contradiction. If possible let $w_{N^U}(i,j) < 0$. From the definition of $N$ it follows that, in the ascending ordering of the nodes, the node $i$ appears before the node *i.e.*, $j$, $i<j$. Since, $w_{N^U}(i,j) < 0$ , therefore

$$d(i) - d(j) - w_N(i,j) < 0, \ or \ d(i) < d(j) + w_N(i,j).$$

But, $d(i)$ is the largest weight of a path from $i$ to $n$ in $N$ and similar is the interpretation for $d(j)$, so

$$d(i) \geq d(j) + w_N(i,j),\ i < j.$$

This is a contradiction. Hence $w_{N^U}(i,j) \geq 0$, for all $(i,j) \in E_N$. □

From the definition of $d$ it is clear that $d(0)$ is the largest weight of a path from 0 to $n$ and $d(n) = 0$. Thus, for any arc $(i,j) \in E_N$, the upper bound of $w_{N^U}(i,j)$ is $d(0)$, which is proved in the following lemma.

**Lemma 10** *The maximum value of $w_{N^U}(i,j)$, for all $(i,j) \in E_N$ is $d(0)$.*

**Proof:** For $i < j$,

$$\begin{aligned} \max_{(i,j) \in E_N} w_{N^U}(i,j) &= \max_{(i,j) \in E_N} \{d(i) - [d(j) + w_N(i,j)]\} \\ &< \max_{(i,j) \in E_N} d(i) = d(0). \end{aligned}$$

since $d(j) + w_N(i,j) \geq 0$. Hence the lemma. □

By Lemma 10 the upper bound of the weights of any arcs of the set $E_N$ is the largest weight of a path from 0 to $n$ in $N$.

It can be shown that the maximum flow of $N$ is equivalent to minimum flow of $N^U$.

**Lemma 11** *For the same set of arcs the maximum weight flow from 0 to $n$ in $N$ is equal to the minimum weight flow in $N^U$.*

**Proof:** Let $W_N$ and $W_{N^U}$ be the weights corresponding to a flow in the network $N$ and $N^U$ respectively.

Therefore,

$$\begin{aligned} W_{N^U} &= \sum_{(i,j) \in E'_N} w_{N^U}(i,j) \\ &= \sum_{(i,j) \in E'_N} \{d(i) - d(j) - w_N(i,j)\} \\ &= d(0) - d(n) - \sum_{(i,j) \in E'_N} w_N(i,j) \\ &= d(0) - \sum_{(i,j) \in E'_N} w_N(i,j)\ \ (as\ (d(n) = 0) \\ &= d(0) - W_N \end{aligned}$$

or, $W_N + W_{N^U} = d(0)$, which is constant, that is, independent of any flow. Therefore, if $W_{N^U}$ is minimum then $W_N$ is maximum. Hence the lemma follows. □

From the above lemma the following theorem is obvious.

**Theorem 4** *The problem $P^*$ for $N$ is equivalent to the problem $P^{**}$ for $N^U$.*

If $O_{P^*}$ and $O_{P^{**}}$ denote the sets of $v$-arcs associated with the output of Problem P* and Problem P** respectively, it can be concluded from the above theorem that both the outputs are identical. Thus, the vertices associated with the $v$-arcs of the output of Problem P* are nothing but the required vertices of Problem P for $G$. Hence we have the following theorem.

**Theorem 5** *The vertices corresponding to the $v$-arcs of the output of the problem $O_{P^{**}}$ are the vertices of maximum weight 2-independent set $Q_2$ of $G$.*

## 8 The Algorithm for Weighted Graph

This section describes all steps of the algorithm for minimum weight feedback vertex set problem and gives the complexity of the proposed algorithm.

**Algorithm MWFVS**

**Input:** Set of intervals $I$, $I_i = [a_i, b_i]$ for $i = 1, 2, \ldots, n$.

**Output:** A minimum weight feedback vertex set $S$.

**Step 1:** Construct a network $N$.

**Step 2:** Compute the array $d$ using Algorithm **DISTANCE**.

**Step 3:** Convert the network $N$ to the network $N^U$ by changing the weights to $w_{N^U}(i,j) = d(i) - d(j) - w_N(i,j), (i,j) \in E_N$.

**Step 4:** Solve the minimum weight 2-flow problem for the network $N^U$, using the algorithm of Edmonds and Karp [6].

**Step 5:** For any $v$-arc $(u,v) \in E_N$, if $f(u,v) > 0$ then put the corresponding vertex to the set $Q_2$.

**Step 6:** Compute the minimum weight feedback vertex set $S = V - Q_2$.

**end MWFVS**

Using Algorithm MWFVS, we obtain the following from Figure 6. The arcs of the minimum weight 2-flow of the network $N^U$ are $e_1$, $e_2$, $e_4$, $e_7$, $e_8$, $e_{16}$. The vertices corresponding to the

arcs $e_1, e_2, e_4, e_7, e_8$ are $\{1, 2, 4, 7, 8\}$. The set $Q_2$ (*i.e.*, the vertices of MW2IS) is {1, 2, 4, 7, 8}. Finally, the MWFVS is {3, 5, 6}. The total weight of feedback vertex set is 9.

The following theorem gives the time complexity of the algorithm MWFVS.

**Theorem 6** *The running time of Algorithm MWFVS is $O(n\sqrt{\log C})$ where $C$ is the maximum weight of a longest path of the graph.*

**Proof.** The array $l(i)$ for all $i \in V$, can be computed for any interval graph in $O(n)$ time. Construction of the network $N$ can be done $O(n)$ time. The $d$-array and the weight of each arc of the network $N^U$ can be computed $O(n)$ time. Using the algorithm of Edmonds and Karp the minimum weight 2-flow problem of $N^U$ can be solved in $O(2 \times$ complexity of shortest path) time. Ahuja et al. [1] have given an $O(m + n\sqrt{\log C})$ time algorithm to compute shortest paths for a general graph with $n$ vertices and $m$ edges, where cost of each arc is non-negative integer number bounded by $C$. In $N^U$, the weight of each arc is bounded by $d(0)$ $(\leq C)$, if $C$ is the longest weight of a path in the given interval graph. Thus, since there are $O(n)$ arcs in $N^U$, the algorithm requires $O(n+n\sqrt{\log C})$ time to solve the flow problem. The Step 5 and Step 6 can be computed using $O(n)$ time. Therefore, the total time complexity is of $O(n + n\sqrt{\log C}) = O(n\sqrt{\log C})$. □

**Corollary 1** *In worst case, if $C = n^n$ then the time complexity becomes $O(n^{1.5} \log n)$.*

## References


[1] Ahuja, R. K., Mehlhorn, K., Orlin, J. B. and Tarjan, R. E., Faster algorithms for the shortest path problem, *J. ACM*, 37 (1990) 213- 223.

[2] Brandstadt, A. and Kratsch, D., On the restriction of some NP-complete graph problems to permutation graphs, *Proceedings of Fundamentals of Computing Theory, LNCS*, 199, Springer - Verlag, Berlin, (1985) 53- 62.

[3] Carlisle. M. C., and Loyd. E. L., On the $k$-coloring of intervals, *LNCS, 497, ICCI'91*, (1991) 90-101.

[4] Coorg, R. and Pandu Rangan, C., Feedback vertex set on co-comparability graphs, *Networks*, 26 (1995) 101- 111.

[5] Corneil, D. G. and Fonlupt, J., The complexity of generalized clique covering, *Discrete Applied Mathematics*, 22 (1988) 109- 118.

[6] Edmonds, J. and Karp, R. M., Theoretical improvements in algorithmic efficiency for net work flow problem, *J. ACM*, 19 (1972) 248- 264.

[7] Fabri. J., *Automatic storage optimization*, UMI Press Ann Arbor, MI, 1982.

[8] Garey, M. R. and Johnson, D. S., *Computers and intractability: A guide to theory of NP completeness*, W. H. Freeman, San Francisco, 1979.

[9] Golumbic. M. C., *Algorithmic graph theory and perfect graphs*, Academic Press, New York, 1980.

[10] Hashimoto. A. and Stevens. J., Wire routing by optimizing channel assignment within large apertures, *Proc., 8th IEEE Design Automation Workshop*, (1971) 155-169.

[11] Jungck. J. R., Dick. O. and Dick. A. G., Computer assisted sequencing, interval graphs and molecular evolution, *Biosystem*, 15 (1982) 259-273.

[12] Karp, R. M., Reducibilities among combinatorial problems, complexity of computer computations, R. E. Miller and J. W. Thatcher. Eds., Plenum press, New York, (1972) 85-103.

[13] Maratha, M. V., Pandu Rangan, C. and Ravi, R., Efficient algorithms for generalized clique covering on interval graphs, *Discrete Applied Mathematics*, 39 (1992) 87- 93.

[14] Ohtsuki. T., Mori. H., Khu. E. S., Kashiwabara. T., Fujisawa. T., One dimensional logic gate assignment and interval graph, *IEEE Trans. Circuits and Systems*, 26 (1979) 675-684.

[15] Pal, M. and Bhattacharjee, G. P., An optimal parallel algorithm for computing all maximal cliques of an interval graph and its applications, *J. Institution of Engineers (India)*, 76 (1995) 22-33.

[16] Ramachandran, V., Finding a minimum feedback arc set in reducible flow graphs, *J. Algorithms*, 9 (1988) 299- 313.

[17] Yannakakis, M. and Gavril, F., The minimum $k$-colourble subgraph problem for chordal graphs, *Information Processing Letters*, 24 (1987) 133- 137.