Linear Time Algorithm for Maximum Weighted Independent Set on Probe Interval Graphs

Pritish Chamania, Ross Mcconnell, Zhisheng Xu

Abstract

In this paper, we give a linear time algorithm for finding maximum weighted independent set on probe interval graphs.

Keywords:
Probe Interval Graph, Maximum Weighted Independent Set

1. Introduction

To be decided.

2. Preliminary

Let $G = (P, N, E)$ denote a probe interval graph. P is the set of Probe vertices, N is the set of Non-Probe vertices and we denote $V = P \cup N$. Every vertexes i has weight w_i. McConnell's recognition algorithm produces an probe interval model in linear time. In the model, each vertexes i will have a label of Non-probe or Probe, with a interval (l_i, r_i) indicating the left endpoint and right endpoint. Without loss of generosity, we may assume that all starting and ending points are unique and from 1 to $2n$.

We can separate interval $[1, 2n]$ into n segments by n left endpoints.(1 must be a left endpoint). Let $S = \{Seg(1), Seg(2), ...Seg(n)\}$ be the partition. For vertex i, Seg(i) is the segment starts at l_i, ends at next left endpoint. The vertices and segments are one to one correspondence. Let S_i be the set of all interval/vertices that intersects $Seg(i)$.

3. Algorithm

We compute non-negative values w_{ij} for every segment j in S, such that

$$\sum_{j \in S} w_{ij} = w_i \tag{1}$$

Also for each $Seg(i)$, we assign y_i such that

$$y_i = \max(\sum_{n \in N} w_{ni}, \max_{p \in P} w_{pi}) \tag{2}$$

We'll show that $\sum_i y_i$ is the solution in next section. The method for getting w_{ij} and y_i as follows:

Algorithm 1: Assigning w_{ij} and y_i values

Result: Assign w_{ij} and y_i values

Create new variables $w_i^{'}$ for each w_i, $w_i^{'}$ initialized to w_i

for $i \in V$ *from right to left by left endpoint* **do**

    Let $S_P(i)$ be the set of all Probes in S_i and has left endpoint before i

    Let $S_N(i)$ be the set of all Non-Probes in S_i and has left endpoint before i

    $y_i \leftarrow w_i^{'}$

    $w_{ii} \leftarrow w_i^{'}$

    $w_i^{'} \leftarrow 0$

    for p *in* $S_P(i)$ **do**

        $w_{pi} \leftarrow min(w_p^{'}, y_i)$

        $w_p^{'} \leftarrow w_p^{'} - w_{pi}$

    end

    if i *is a Probe* **then**

        $y_i^{'} \leftarrow y_i$

        for n *in* $S_N(i)$ *from right to left by starting point* **do**

            $w_{ni} \leftarrow min(w_k^{'}, y_i^{'})$

            $w_n^{'} \leftarrow w_n^{'} - w_{ni}$

            $y_i^{'} \leftarrow y_i^{'} - min(w_{ni}, y_i^{'})$

        end

    end

end

Lemma 3.1. $\sum_{j \in S} w_{ij} = w_i$ *after algorithm 1*

Proof The process goes from right to left by left endpoint, w_{ij} can be seen as the amount of weight interval i "unloaded" into $Seg(j)$. Once we unload some weight w_{ij}, we decrease the remaining weight $w_i^{'}$ by w_{ij}. The amount we unload each time is up to the current remaining weight $w_i^{'}$, thus $\sum_{j \in S} w_{ij} + w_i^{'} = w_i$ is invariant during the process. $w_i^{'}$ will be decreased to 0 before or at processing i, so $\sum_{j \in S} w_{ij} = w_i$ after the process.

Every time that we process an interval i, we set $y_i = w_i^{'}$, then set w_{ii} to $w_i^{'}$. No matter whether i is a probe or a non-probe, every probe j that intersects i will unload $min(w_j^{'}, y_i)$ weight on i. If i is a probe, then for every non-probe k that intersects i, we process them from right to left by starting point, unloading $min(w_k^{'}, y_i^{'})$ weight on i, $y_i^{'}$ is decreased along the process until it becomes 0.

Lemma 3.2. $y_i = \max(\sum_{n \in N} w_{ni}, \max_{p \in P} w_{pi})$ *after algorithm 1*

Proof If i is a non-probe, it's not hard to see i is the only non-probe that unloaded weight on $seg(i)$, $y_i = w_{ii}$, so $y_i = \sum_{n \in N} w_{ni}$. And every probe p that unloaded to i has weight $w_{pi} = min(w_p^{'}, y_i)$, so $y_i \geq w_{pi}$, thus $y_i = \max(\sum_{n \in N} w_{ni}, \max_{p \in P} w_{pi})$ for non-probes.

If i is a probe, $y_i = w_{ii}$. For probe p that unloaded to i, $w_{pi} = min(w_p^{'}, y_i)$, so $y_i \geq \max_{p \in P} w_{pi}$. Since $y_i = w_{ii}$ and $i \in P$, $y_i = \max_{p \in P} w_{pi}$. Also, the summation of all weights unloaded from non-probes can not be greater than y_i, then $y_i \geq \sum_{n \in N} w_{ni}$ in the above steps, so $y_i = \max(\sum_{n \in N} w_{ni}, \max_{p \in P} w_{pi})$ also holds true for probes.

The method for getting the maximal weighted independent set as follow:

Algorithm 2: Construct the solution

Result: Produce the Solution to maximal weighted independent set problem

Solution set R initialized to $\emptyset$

All vertices initialized to unmarked.

for $i \in V$ *from left to right by left endpoint* **do**

    if $y_i > 0$ **then**

        $R \leftarrow R \cup \{i\}$

        if i *is a Probe* **then**

            Remove intervals intersects i from V

        end

        else if i *is a Non-Probe* **then**

            Mark Probes that intersects i and remove them from V

        end

    end

    else if $y_i = 0$ **then**

        if $\exists p$ *that is marked and* $w_{ip} \neq 0$ **then**

            $R \leftarrow R \cup \{i\}$

        end

    end

end

R is the maximum independent set upon termination of the algorithm.

4. Analysis

Lemma 4.1. *For any independent set I, $\sum_{i \in I} w_i \leq \sum_{j \in V} y_j$*

Proof

$$\sum_{i\in I} w_i = \sum_{i\in I}\sum_{j\in S} w_{ij} = \sum_{j\in S}\sum_{i\in I\cap Seg(j)} w_{ij} \leq \sum_{j\in S}\max(\sum_{i\in N} w_{ij}, \max_{i\in P} w_{ij}) = \sum_{j\in S} y_j = \sum_{j\in V} y_j$$

The third inequality follows from the fact that I can only intersects a segment j with a Probe or a set of Non-probes, which has less weight than sum of all weight of Non-probes or the Probe with largest weight in $Seg(j)$.

The above proof shows that any independent set must have a total weight less than all the y values summed up. Thus, if we can produce a such independent set that equals the summation of all y values, then it must be the maximum independent set. Below, we'll show that the set we get from algorithm2 is the solution.

Definition For a probe p, T_p is the set of all vertex i such that $l_p \leq l_i \leq r_p$. Note that $p \in T_P$.

Definition For a set of Non-probes N, T_N is the set of all vertex i such that $\min_{n \in N} l_n \leq l_i \leq \max_{n \in N} r_n$. Similarly, $N \subseteq T_N$.

Definition A probe p in solution set R is tight if $w_p = \sum_{i \in T_p} y_i$. For any i in T_N, we say i is covered by p.

Definition A set of Non-probes N in solution set R is tight if $\sum_{n \in N} w_n = \sum_{i \in T_N} y_i$. For any i in T_N, we say i is covered by N.

Definition Let us define the set of Non-Probes added to R continuously a Non-Probe block.

Lemma 4.2. *Any Probe p in solution set R is tight.*

Proof The premise of p being added to R is $y_p > 0$, i.e. $w_p^{'} > 0$ when we start processing p. From the definition, for any vertex in T_p, $p \in S_p(i)$. According to algorithm1, p is processed after we process every $i \in T_p$. When we process i, $w_p^{'} > y_i$, otherwise, $w_p^{'}$ will be 0 after processing i. Since $w_p^{'} > y_i$, $w_p^{'}$ is reduced by y_i afterwards. In this way, $w_p^{'}$ is initialized to w_p, and decreased by y_i for every i in T_p(including processing p itself), and become 0 after processing p. Hence $w_p = \sum_{i \in T_p} y_i$.

claim 4.3. *For any Non-Probe $n \in T_N$ for a Non-Probe block N, if $y_n > 0$, $n \in N$.*

Proof It's not hard to see that the only way to prevent it from adding to R is that it intersects some Probe $p \in R$. suppose $n \notin N$, there exists a probe $p \in R$, and p intersects n. Since N is a Non-Probe block, p is added to R before N. That indicates the Non-probe with leftmost left endpoint in N also intersects p, contradiction.

claim 4.4. *For any Probe $p \in T_N$ in a Non-Probe block N, if $y_p > 0$, processing p in Algorithm 1 decreases $\sum\limits_{n \in N} w_n^{'}$ by y_p.*

Proof Since $p \in T_N$ and $y_p > 0$, $p \notin R$ indicates p is marked by some Non-Probe $n \in R$ by algorithm2 , and $y_n > 0$. By definition of T_N, $n \in T_N$. Then all Non-Probes that unloaded weight to p will also be added to R by algorithm2 line 15-17, and belongs to T_N. So the total decrease of $w_n^{'}$ for Non-Probes in T_N is y_p by Algorithm 1.

Lemma 4.5. *Any Non-Probe block N in solution set R is tight.*

Proof It's easy to see 4.3 indicates that processing any Non-Probe $n^* \in T_N$ decrease $\sum\limits_{n \in N} w_n^{'}$ by y_{n^*}. By 4.3, 4.4, $\sum\limits_{n \in N} w_n^{'} + \sum\limits_{i \in T_n} y_i$ is a constant. $w_n^{'} = w_n$ before algorithm 1, and $w_n^{'} = 0$ after. so $\sum\limits_{n \in N} w_n = \sum\limits_{i \in T_n} y_i$

Theorem 4.6. $\sum\limits_{i \in R} w_i = \sum\limits_{i \in V} y_i$

Proof Any interval i with $y_i > 0$ is added to R otherwise contained in T_p or T_N for a probe p or a Non-Probe block N by algorithm 2. By 4.2, 4.4, $\sum\limits_{i \in R} w_i = \sum\limits_{i \in V} y_i$ if we break R into Probes and Non-probe blocks.