MTL760 Term paper

Varre Harsha Vardhan 2018MT60796
Ramneek Singh Gambhir 2018MT60788
Snehil Grandhi 2018MT60795

Problem Statement

Signed Italian Domination Problem in Bipartite Graphs

Let G = (V, E) be a graph we wish to make an assignment:

f : V -> {-1, 1 , 2} so that $\sum_{u \in N[v]} f(u) \quad >= 1 \qquad \forall v \in V$, N[v] is the closed neighbourhood of v.

(* We call this domination condition from now)

Note that S = {v : f(v) = 1 or 2} forms a dominating set for G under such an assignment.

We wish to minimize Σ f(v) for v in V, over all such possible assignments.

The optimal value is denoted by $\gamma sI(G)$

Note that it is always possible to do such an assignment by assigning f(v) = 1 to all vertices.

Previous Work

Karamzadeh, A., Maimani, H.R. & Zaeembashi, A. Further results on the signed Italian domination. J. Appl. Math. Compute. (2020).

Derived a lower bound for any tree T of order n >= 2

$$\gamma_{sI}(T) \geq \frac{-n+4}{2}$$

They have also showed that the signed italian domination problem is NP-Complete for bi-partite graphs.

Complete Bi-partite Graphs

Let $K_{m,n}$ be a complete bi-partite graph then, the signed Italian domination number of a graph G($\gamma_{sI}(G)$), [Karamzadeh_A_et_al.dvi (math.md)]

For $n \geq 2$,

$$\gamma_{sI}(K_{1,n-1}) = \begin{cases} 1 & \textit{if } n \textit{ is even}, \\ 2 & \textit{if } n \textit{ is odd}. \end{cases}$$

For $2 \leq m \leq n$,

$$\gamma_{sI}(K_{m,n}) = \begin{cases} 2 & \textit{if } m = 2 \textit{ and } n \geq 2, \\ 3 & \textit{if } m = 3 \textit{ and } n \geq 3, \\ 4 & \textit{if } n, m \geq 4. \end{cases}$$

Our Approach

We present a polynomial time algorithm for solving the signed italian domination problem in trees using dynamic programming.

Tree

Let G = G(V, E) be a rooted tree.

Define a dp(v, val, parent_val)

for v belong to V, val belong to {-1, 1, 2} and parent_val belong to {-1, 1, 2}

to be the minimum weight of an italian assignment to the subtree rooted at v

with val = f(v) and parent_val = f(parent[v]) ,

satisfying the domination condition for all v in the subtree

If such an assignment is not possible then it returns +inf.

Note: for the domination condition at v(the root of the sub-tree) we also include its parent into the neighbours.

Then the answer for our optimization problem is given by:

$$\min_{x \in \{-1,1,2\}} dp(tree\_root, x, 0)$$

Now we see how we calculate dp(v, x, px):-

Base case: If there are no children of v, then we can trivially assign the value to the dp state

Let c1, c2, ... ck be the child nodes of v.

First we recursively calculate all the dp states:

dp(c, cx, x) for all c in {c1, c2, .. ck} and cx in {-1, 1, 2}.

Define F(c, cx) := dp(c, cx, x)

Define another dp,

$$dp1(i, s) = \min_{cx_j \in \{-1,1,2\}} \sum_{j=1}^{i} F(c_j, cx_j) \text{ such that :}$$

$$\sum_{j=1}^{i} cx_j >= s$$

Note that now the answer for dp(v, val, parent_val) = dp1(k, 1 - val - parent_val), (k = # of child nodes)

We calculate the dp1 in a knapsack fashion, using the recurrence

$$dp1(i, s) = \min_{cv \in \{-1,1,2\}} dp1(i-1, s-cv) + F(c_i, cv)$$

Base case :-

$dp1(1, s)$	$= \min[\, F(c1, -1), F(c1, 1), F(c1, 2) \,]$,	for $s \le -1$
	$= \min[\, F(c1, 1), F(c1, 2) \,]$,	for $0 \le s \le 1$
	$= F(c1, 2)$	for $s = 2$
	$= +\infty$	otherwise.

Time Complexity Analysis

For each node we compute dp in a bottom up fashion. These 9 cases will correspond to the permutations of value of current node and the value of the parent node. (For each node it is 3 i.e. (-1, 1, 2)).

We memoize the *dp*s for future use.

Dp1 is created like a knapsack problem with number of values = k and sum of values ranging from -k to 2k. Hence, it can be calculated in {k*(3k) =} O(k^2) steps.

Number of steps for each node = O(k^2)

Final time complexity = **Σ** k^2 for all k = degree of each node - 1,

<= (**Σ** k)^2 = O(n^2)

Space Complexity

Space Complexity for dp = O(n)

Space complexity for dp1 = O(k^2)

Total Space complexity = **Σ** k^2{dp1 is stored for each node} + O(n)

= O(n^2) + O(n)

= O(n^2)

THANK YOU