Signed Italian Domination in Bipartite Graphs

Varre Harsha Vardhan, 2018MT60796

Ramneek Singh Gambhir, 2018MT60788

Snehil Grandhi, 2018MT60795

1 Problem Statement

Dominating Set: Let $G = (V, E)$ be a simple graph. A set of vertices $S \subset V$ is called a dominating set if every $v \in V \setminus S$ is adjacent to some $u \in S$.
Closed neighbourhood: The set of all vertices adjacent to v, including v. It is denoted by $N[v]$
Italian Assignment: An assignment, $f : V \to \{-1, 1, 2\}$ on the set of vertices is called an italian assignment.
The signed Italian Domination problem: We wish to find an Italian assignment on V so that the following holds: $\sum_{u \in N[v]} f(u) \geq 1,\ \forall v \in V \quad (\star)$ and the weight of the assignment $W(f) = \sum_{v \in V} f(v)$ is minimum over all such assignments. It is denoted by $\gamma_{sI}(G)$. $(\star)$ is called the domination condition.
Note that an Italian assignment that satisfies $(\star)$ is always possible, viz assigning $f(v) = 1\, \forall v \in V$.
Also, note that under an Italian assignment the set $S = \{v\ : f(v) = 0 \text{ or } 1\}$ is a dominating set.

2 Literature review

Previously, **[1]** showed that the signed Italian domination problem is **NP** Complete for Bipartite graphs. They also derived a lower bound, $\gamma_{sI}(T) \geq \frac{-n+4}{2}$ for any tree **T** of order ≥ 2.
[2] derived the exact results for cycles, paths and complete Bipartite graphs.

3 Our Approaches

We show how to solve the Signed Italian domination problem in trees using dynamic programming.

3.1 Solution for Binary Trees

A binary tree is a rooted tree, where each node has at-most two children. The children(if they exist) are referred to as the left child and the right child.

To solve the problem on a binary tree **T** we define a dynamic programming: $\mathrm{DP}(\mathrm{node}, \mathrm{node}_{\mathrm{val}}, \mathrm{parent}_{\mathrm{val}})$, where $v \in V$, $\mathrm{node}_{\mathrm{val}} \in \{-1, 1, 2\}$ and $\mathrm{parent}_{\mathrm{val}} \in \{-1, 1, 2\}$. The value of the **DP** is equal to the minimum weight of an Italian assignment satisfying the domination condition on the sub-tree rooted at **node** with $f(\mathrm{node}) = \mathrm{node}_{\mathrm{val}}$. For writing the domination condition at node we also include it's parent node with $f(\mathrm{parent}[\mathrm{node}]) = \mathrm{parent}_{\mathrm{val}}$. However we don't care about the domination condition on the parent node; we only need it to be satisfied on the sub-tree. Thus by definition, $\gamma_{sI}(T) = \min_{\mathrm{val} \in \{-1,1,2\}} \mathrm{DP}(T_{\mathrm{root}}, \mathrm{val}, 0)$. Now we show how to calculate $\mathrm{DP}(v, x, px)$ recursively. Let C_v be the set of children of v. Since we are working on a binary-tree, $C_v = \varphi$ or $\{\mathrm{left\_child}\}$ or $\{\mathrm{right\_child}\}$ or $\{\mathrm{left\_child}, \mathrm{right\_child}\}$. Thus $|C_v| \leq 2$. Let $\mathbf{X_v}$ be the set of all possible mappings, $f : C_v \to \{-1, 1, 2\}$. We have $|X_v| = 3^{|C_v|} \leq 9$. Now we can calculate DP recursively as,

$$\mathrm{DP}(v, x, px) = x + \min_{\substack{f \in X_v: \\ \sum\limits_{u \in C_v} f(u) + x + px \geq 1}} \sum_{u \in C_v} \mathrm{DP}(u, f(u), x)$$

Essentially, we are calculating the optimal minimum value for $\mathrm{DP}(v, x, px)$ by brute-forcing over $|X_V|$. Note that, Number of DP states $= 9|T|$ and since size of $|X_v| \leq 9$, all the DP states and hence $\gamma_{sI}(T)$ can be calculated in $\mathbf{O}(|\mathbf{T}|)$ time and space complexity using memoization. For a k-ary tree, T since $|X_v| \leq 3^k$, this algorithm can be used to find $\gamma_{sI}(T)$ in $\mathbf{O}(\mathbf{3^k}|\mathbf{T}|)$ time complexity. In the next section, we present how to solve the problem in general trees by showing how to calculate $\mathrm{DP}(v, x, px)$ using another dynamic-programming instead of brute-forcing over X_v.

3.2 Solution for General Trees

Let $T = (V, E)$ be a rooted tree. We will define $\mathrm{DP}(\mathrm{node}, \mathrm{node}_{\mathrm{val}}, \mathrm{parent}_{\mathrm{val}})$ as before. The crux of the problem is, earlier we were calculating $\mathrm{DP}(v, x, px)$ by looking over all possible assignments to C_v, which is exponential in general, however for the binary-tree case it is bounded by a small constant. Can we do better than this brute-force approach? It turns out, that our problem is similar to the weighted knapsack problem **[3]**. We show how to calculate $\mathrm{DP}(v, x, px)$ efficiently using another DP. If $|C_v| = 0$ we can assign the value to the DP trivially. Otherwise let $|C_v| = k > 0$ and we index the children of v so that $C_v = \{c_1, c_2, \cdots, c_k\}$. For a fixed v, define $\mathrm{DP}_{\mathrm{knap}}(n, \mathrm{sum})$ for $n \in \{1, 2, \cdots, k\}$ and $\mathrm{sum} \in \mathbb{Z}$. It's value is equal the minimum of $\sum\limits_{i=1}^{n} \mathrm{DP}(c_i, f(c_i), x)$ over all possible assignments $f : \{c_1, c_2, \cdots, c_n\} \to \{-1, 1, 2\}$,satisfying, $\sum\limits_{i=1}^{n} f(c_i) \geq \mathrm{sum}$. Thus by definition $\mathrm{DP}(v, x, px) = \mathrm{DP}_{\mathrm{knap}}(k, 1 - x - px)$.

We can calculate these $\mathrm{DP_{knap}}$ states in a knapsack fashion as follows:

$$\mathrm{DP_{knap}(n, sum)} = \min_{\mathrm{val} \in \{-1,1,2\}} (\mathrm{DP_{knap}(n-1, s-val) + DP(c_n, val, x)}), \text{ for } \mathrm{n} > 1$$

For $\mathrm{n} = 1$, we arrive at the base case and we can assign the values to the $\mathrm{DP_{knap}}$ trivially as follows:

$$\mathrm{DP_{knap}(1, sum)} = \begin{cases} \min\{\mathrm{DP(c_1, -1, x), DP(c_1, 1, x), DP(c_1, 2, x)}\} & ; \mathrm{sum} \leq -1 \\ \min\{\mathrm{DP(c_1, 1, x), DP(c_1, 2, x)}\} & ; 0 \leq \mathrm{sum} \leq 1 \\ \mathrm{DP(c_1, 2, x)} & ; \mathrm{sum} = 2 \\ \infty & ; \mathrm{sum} > 2 \end{cases}$$

Note that -$\mathrm{k} \leq \sum\limits_{\mathrm{i}=1}^{\mathrm{k}} \mathrm{f(c_i)} \leq 2\mathrm{k}$. Hence we will need to calculate $\mathrm{DP_{knap}}$ only for -$\mathrm{k} \leq \mathrm{sum} \leq 2\mathrm{k}$. Thus total states for $\mathrm{DP_{knap}(n, sum)} = 3\mathrm{k}^2$ and each state can be calculated in constant time(minimum over 3 values), thus we can compute $\mathrm{DP(v, x, px)}$ using $\mathrm{DP_{knap}}$ in $\mathbf{O}(\mathbf{k^2})$ time and space complexity. To recall, $\mathrm{DP(v, x, px)} = \mathrm{DP_{knap}(|C_v|, 1 - x - px)}$ and we have $9|\mathrm{T}|$ DP states and each can be calculated in $\mathbf{O}(|\mathrm{C_v}|^2)$ time and space. Thus overall time and space complexity is
$\sum_{\mathrm{v} \in \mathrm{V}} \mathrm{O}(|\mathrm{C_v}|^2) = \mathrm{O}((\sum_{\mathrm{v} \in \mathrm{V}} |\mathrm{C_v}|)^2) = \mathbf{O}(|\mathbf{T}|^{\mathbf{2}})$.

4 References

[1] Karamzadeh, A. Maimani, Hamid Zaeembashi, A.. (2020). Further results on the signed Italian domination. Journal of Applied Mathematics and Computing. 1-12. 10.1007/s12190-020-01465-x.

[2] http://www.math.md/files/csjm/v27-n2/v27-n2-(pp204-229).pdf

[3] https://cses.fi/problemset/task/1158