VARIANTS OF ROMAN AND ITALIAN DOMINATION IN WEIGHTED GRAPHS

January 7, 2024

Krishna Priyatam D

Roll No. 170123028

Sujana Maithili Chindam

Roll No. 170123016

Overview

Dominating Set

- A dominating set of a graph $G = (V, E)$ is a subset $D \subseteq V$ such that every vertex $v \notin D$ is adjacent to at least one vertex $u \in D$.
- The domination problem is to find the minimum size dominating set of a given graph.
- Domination number, $\gamma(G)$.

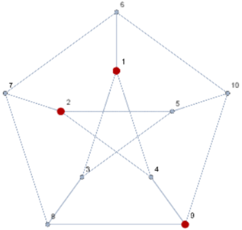


Figure: Dominating Set

Roman Domination

- A Roman dominating function(RDF) on a graph $G = (V, E)$ is a function $f : V(G) \to \{0, 1, 2\}$ s.t. every vertex $u \in V$ with $f(u) = 0$ is adjacent to at least one vertex $v \in V$ with $f(v) = 2$.
- The weight of a RDF f is the value $f(V(G)) = \Sigma_{u \in V(G)} f(u)$.
- Roman domination number $\gamma_R(G)$ of G.

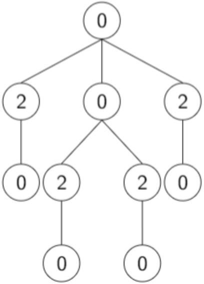


Figure: An RDF assignment

Perfect Roman Domination

- A Perfect Roman Dominating Function(PRDF) on a graph $G = (V, E)$ is a function $f : V(G) \to \{0, 1, 2\}$ s.t. every vertex $u \in V$ with $f(u) = 0$ is adjacent to exactly one vertex with $f(v) = 2$.
- Perfect Roman domination number $\gamma_R^P(G)$ of G.

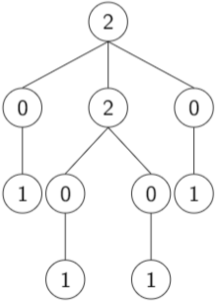


Figure: An PRDF assignment

Independent Roman Domination

- An Independent Roman Dominating Function(IRDF) on a graph $G = (V, E)$ is an RDF $f : V(G) \to \{0, 1, 2\}$ such that the set formed by the union of the vertices that are assigned the values 1 and 2, is an independent set.
- Independent Roman domination number $\gamma_R^I(G)$ of G.

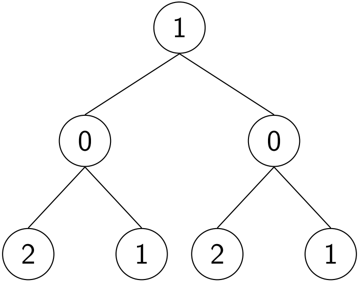


Figure: An IRDF assignment

Total Roman Domination

- A total Roman Dominating Function(TRDF) on an undirected graph $G = (V, E)$ is a function $f : V(G) \to \{0, 1, 2\}$ satisfying the following conditions:
  1. Every vertex $u \in V$ for which $f(u) = 0$ is adjacent to at least one vertex $v \in V$ for which $f(v) = 2$ and
  2. The subgraph of G induced by the set of all vertices of positive assignment has no isolated vertices.
- Perfect Roman domination number $\gamma_{tR}(G)$ of G.

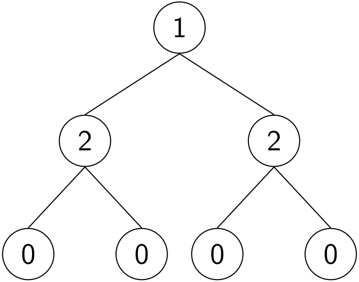


Figure: An TRDF assignment

Italian Domination

- An Italian dominating function(IDF) on an graph $G = (V, E)$ is a function $f : V(G) \to \{0, 1, 2\}$ s.t. for every vertex $u \in V$ with $f(u) = 0$, it holds that $\Sigma_{v \in N(u)} f(v) \geq 2$ where $N(u) = \{v \in V | (u, v) \in E\}$.
- Italian domination number $\gamma_I(G)$ of G.

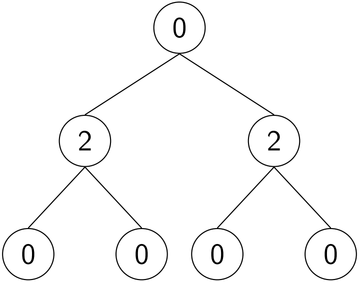


Figure: An IDF assignment

Perfect Italian Domination

- A perfect Italian dominating function on an graph $G = (V, E)$ is a function $f : V(G) \to \{0, 1, 2\}$ s.t. for every vertex $u \in V$ with $f(u) = 0$, it holds that $\sum_{v \in N(u)} f(v) = 2$.
- Perfect Italian domination number $\gamma_I^P(G)$ of G.

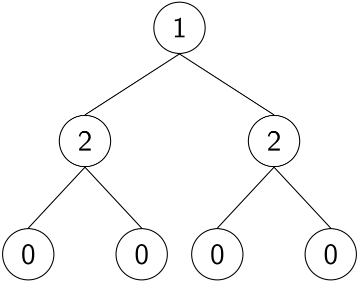


Figure: An PIDF assignment

Previous Work

- The domination problem concerns finding the minimum size dominating set for a given graph G.
- Few variants of the domination problem are Roman domination, perfect Roman domination, Italian domination and perfect Italian domination, on unweighted graphs.
- **We have extended the above variants onto a weighted graph G and given a linear time algorithm for the above variants on a weighted tree T.**

Problem Statement

- Few other variants of the domination problem are Total Roman domination and Independent Roman domination on unweighted graphs.
- **Extend the above variants onto a weighted graph G and find a polynomial time algorithm for the above variants on some classes of weighted graphs G.**

Literature Survey

- There has been a lot of work in the areas of Roman, Italian domination and their variants, in the recent years.
- The problem of Roman domination is NP-complete is found to be NP-complete for some classes of graphs(bipartite, chordal, planar and split)[1].
- Algorithms exist for some other classes of graphs(cographs, AT-free graphs)[2].

Literature Survey

- Perfect Roman domination on trees is extensively studied in [3].
- Italian domination on trees is studied in [4], and perfect Italian domination on trees is studied in [5].
- The class of block graphs is also popular in the area of domination. Algorithms for Roman and Italian domination, and some variants of Roman and Italian domination are given in [6, 7].

Variants of Roman/Italian Domination in weighted graphs

- We extend the above definition for Variants Roman/Italian domination for weighted graphs.
- Let $G = (V, E)$ be a weighted graph with the weight function $w : V(G) \to \mathbb{R}$.
- We define the weight of the variant of Roman/Italian dominating function f to be $f(V(G)) = \sum_{u \in V(G)} w(u) \times f(u)$, where f is an dominating function of that variant.

Example

Let us assign Roman(Italian) function f for every vertex of weighted trees with weight function w. Given $w(v), f(v)$ for $v \in T$.

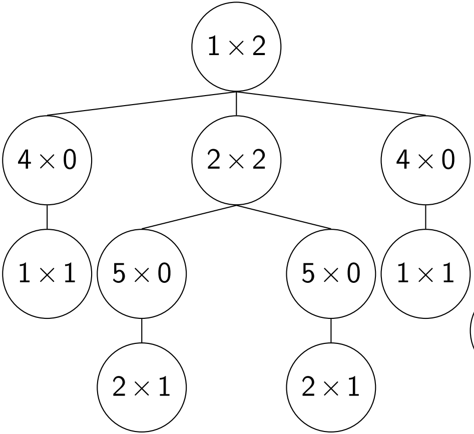


Figure: Weighted Roman Domination

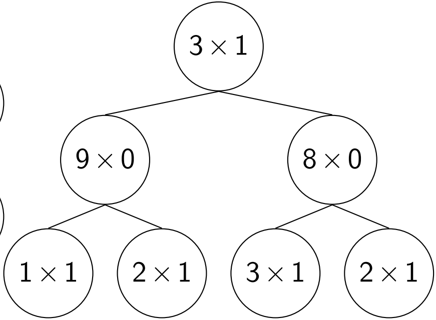


Figure: Weighted Italian Domination

Algorithm for Total Roman Domination in weighted tree

- We use the dynamic programming paradigm to solve the total Roman domination number $\gamma_{tRW}(T)$ of a weighted tree T.
- We declare 6 arrays to store optimal values for each vertex in tree. They are dp_0, dp_0^2, dp_1, $dp_1^{1,2}$, dp_2, $dp_2^{1,2}$.

Algorithm for Total Roman Domination in weighted tree

- Let u be a specific vertex in T, and let T' be the subtree rooted at u.
  1. $dp_0[u]$
  2. $dp_0^2[u]$
  3. $dp_1[u]$
  4. $dp_1^{1,2}[u]$
  5. $dp_2[u]$
  6. $dp_2^{1,2}[u]$
  7. Then $\gamma_{tRW}(T') = \min\{dp_0^2[u], dp_1^{1,2}[u], dp_2^{1,2}[u]\}$
- We use Depth First Search (DFS) to traverse the tree.

Algorithm for Total Roman Domination in weighted tree

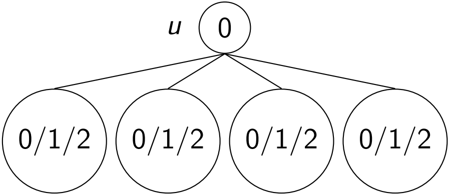


Figure: $dp_0[u]$

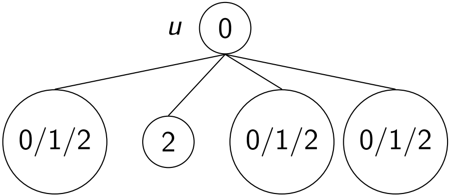


Figure: $dp_0^2[u]$

Algorithm for Total Roman Domination in weighted tree

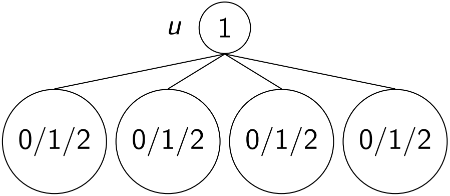


Figure: $dp_1[u]$

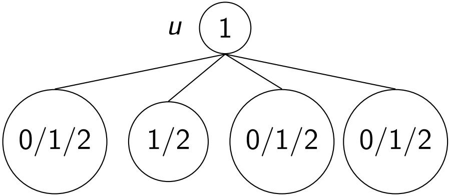


Figure: $dp_1^{1,2}[u]$

Algorithm for Total Roman Domination in weighted tree

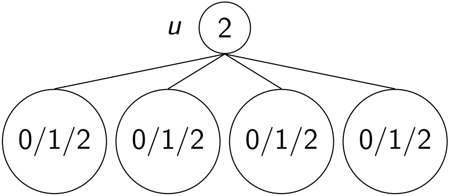


Figure: $dp_2[u]$

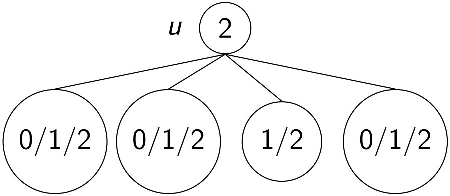


Figure: $dp_2^{1,2}[u]$

Algorithm for Total Roman Domination in weighted tree

- Let v be a leaf vertex in T, then $dp_0[v] = 0$, $dp_0^2[v] = \infty$, $dp_1[v] = 1 \times w(v)$, $dp_1^{1,2}[v] = \infty$, $dp_2[v] = 2 \times w(v)$, $dp_2^{1,2}[v] = \infty$.
- Let u be a non leaf vertex in T, and let T' be the subtree rooted at u.
  1. $dp_0[u] = \sum_{c \in C(u)} \min\{dp_0^2[c], dp_1^{1,2}[c], dp_2^{1,2}[c]\}$
  2. $dp_0^2[u] = \min_{c \in C(u)} \{dp_2^{1,2}[c] + \sum_{c' \in C(u) \setminus \{c\}} \min\{dp_0^2[c'], dp_1^{1,2}[c'], dp_2^{1,2}[c']\}\}$
  3. $dp_1[u] = 1 \times w(u) + \sum_{c \in C(u)} \min\{dp_0^2[c], dp_1[c], dp_2[c]\}$

Algorithm for Total Roman Domination in weighted tree

4. $dp_1^{1,2}[u] = 1 \times w(u) + \min_{c \in C(u)} \{\min\{dp_1[c], dp_2[c]\} + \sum_{c' \in C(u) \setminus \{c\}} \min\{dp_0^2[c'], dp_1[c'], dp_2[c']\}\}$
5. $dp_2[u] = 2 \times w(u) + \sum_{c \in C(u)} \min\{dp_0[c], dp_1[c], dp_2[c]\}$
6. $dp_2^{1,2}[u] = 2 \times w(u) + \min_{c \in C(u)} \{\min\{dp_1[c], dp_2[c]\} + \sum_{c' \in C(u) \setminus \{c\}} \min\{dp_0[c'], dp_1[c'], dp_2[c']\}\}$

- Finally after calculating optimal values of the root r of the Tree T $\gamma_{tRW}(T) = \min\{dp_0^2[r], dp_1^{1,2}[r], dp_2^{1,2}[r]\}$

Time and Space Complexity of the Algorithm

Let n be the number of vertices in weighted tree T

- **Time Complexity:** We use DFS to traverse the tree. Hence, the time complexity is $O(n)$.
- **Space Complexity:** $O(n)$.

Correctness of the Algorithm

- The problem of finding total Roman Domination Number has two properties:
  1. Overlapping Subproblems
  2. Optimal Substructure
- We can prove the correctness of the Dynamic Programming algorithm by induction.
- **Induction Hypothesis:** Let the algorithm computes the total Roman Domination number correctly for trees of height $h'(< h)$.
- We try to show that the algorithm computes the total Roman Domination number correctly for trees of height h.

Block Graph

- A graph $G = (V, E)$ is said to be a block graph, if every biconnected component, i.e., a block in the graph G is a clique.

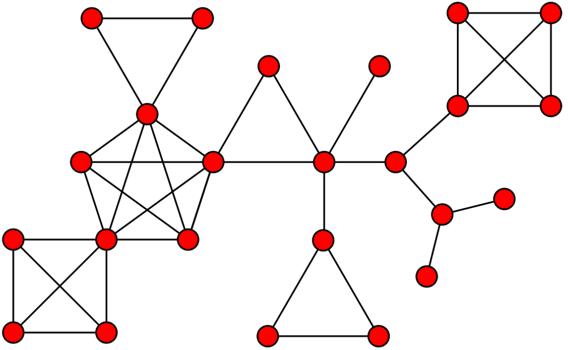


Figure: An example of block graph

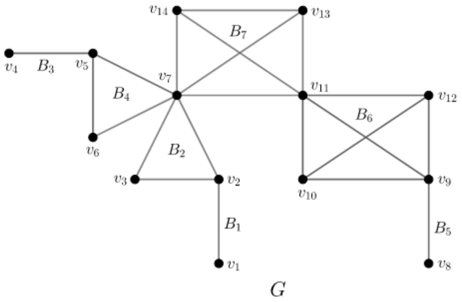
Block Graph

- A vertex v of G is said to be a cut-vertex, if removing the vertex v increases the number of connected components in the graph.
- A block in a graph G is a maximal biconnected subgraph of G.
- In a block graph G, two blocks share at most one vertex, which is a cut-vertex of G.

Block Cut-Point Tree

- A Block cut-point tree of a given block graph $G = (V, E)$ is a bipartite tree $T(G) = (V', E')$ in which one set consists of the vertices b_i corresponding to each block B_i in G, and the other set consists of the cut-vertices of G, and the set of edges E' where $vb_i \in E'$, if and only if $v \in B_i$, where v is a cut-vertex, and b_i is a block-vertex in $T(G)$ corresponding to a block in G.

Block Cut-Point Tree



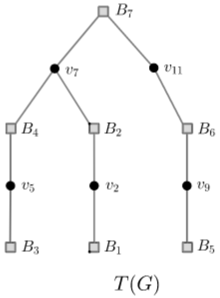


Figure: A block graph G and corresponding block cut-tree $T(G)$

Algorithm for Total Roman Domination in weighted block graph

For our algorithm, we view the block graph as a graph rooted at a specific cut-vertex v. We use the following composition to build up a block graph H, starting from the trivial graphs, $\{v_i\}$ for $v_i \in V(H)$.

- Let $H_1, H_2, \dots, H_n$, $n \geq 2$ be graphs rooted at $v_1, v_2, \dots, v_n$, such that $v_i \in H_i$, $1 \leq i \leq n$.
- Now, we call the graph H as the composition of the graphs $\{H_1, H_2, \dots, H_n\}$, if H is obtained by adding edges to $v_1, v_2, \dots, v_n$, such that $\{v_1, v_2, \dots, v_n\}$ is a clique.

Algorithm for Total Roman Domination in weighted block graph

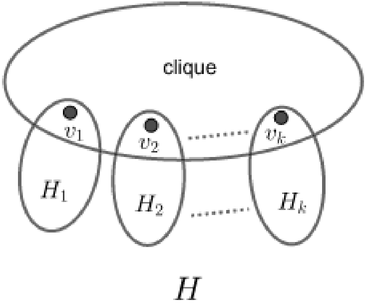


Figure: Method to construct a block graph

Algorithm for Total Roman Domination in weighted block graph

- We use the dynamic programming paradigm to find the Total Roman domination number $\gamma_{tRW}(H)$ of a weighted block graph H.
- We declare 6 arrays to store optimal values for each vertex in the Block graph. They are $dp_0^{0,1}$, dp_0^2, dp_1^0, $dp_1^{1,2}$ dp_1, $dp_2^{1,2}$.

Algorithm for Total Roman Domination in weighted block graph

For a graph H, let $F(H)$ be the set of all the functions which assign a value from $\{0,1,2\}$ to every vertex in H. Now, let us define the following sets corresponding to the graph H, and a specific vertex $v \in V(H)$:

1. $F_0^{0,1}(v,H) = \{f \in F(H)$: f_{H-v} is an TRDF and $f(v) = 0, \ \forall\, v\prime \in N_H(v) \ \ f(v\prime) \in \{0,1\}\}$
2. $F_0^{2}(v,H) = \{f \in F(H)$: f_{H} is an TRDF and $f(v) = 0$, for $v\prime \in N_H(v) \ \ f(v\prime) = 2$, and $f(x) \in \{0,1,2\} \ \forall\, x \in N_H(v) \setminus v\prime\}$
3. $F_1^{0}(v,H) = \{f \in F(H)$: f_{H-v} is an TRDF and $f(v) = 1, \ \forall\, v\prime \in N_H(v) \ \ f(v\prime) = 0\}$

Algorithm for Total Roman Domination in weighted block graph

4. $F_1^{1,2}(v,H) = \{f \in F(H)$: f_H is an TRDF and $f(v) = 1$, for $v\prime \in N_H(v)\ \ f(v\prime) \in \{1,2\}$, and $f(x) \in \{0,1,2\}\ \forall\, x \in N_H(v) \setminus v\prime\}$
5. $F_2^{0}(v,H) = \{f \in F(H)$: f_{H-v} is an TRDF and $f(v) = 2,\ \forall\, v\prime \in N_H(v)\ \ f(v\prime) = 0\}$
6. $F_2^{1,2}(v,H) = \{f \in F(H)$: f_H is an TRDF and $f(v) = 2$, for $v\prime \in N_H(v)\ \ f(v\prime) \in \{1,2\}$, and $f(x) \in \{0,1,2\}\ \forall\, x \in N_H(v) \setminus v\prime\}$

Algorithm for Total Roman Domination in weighted block graph

- Now, let us denote
  - $dp_0^{0,1}(v,H) = min\{w(f) : f \in F_0^{0,1}(v,H)\}$
  - $dp_0^{2}(v,H) = min\{w(f) : f \in F_0^{2}(v,H)\}$
  - $dp_1^{0}(v,H) = min\{w(f) : f \in F_1^{0}(v,H)\}$
  - $dp_1^{1,2}(v,H) = min\{w(f) : f \in F_1^{1,2}(v,H)\}$
  - $dp_2^{0}(v,H) = min\{w(f) : f \in F_2^{0}(v,H)\}$
  - $dp_2^{1,2}(v,H) = min\{w(f) : f \in F_2^{1,2}(v,H)\}$
- Then $\gamma_{tRW}(H) = min\{dp_0^2(v,H), dp_1^{1,2}(v,H), dp_2^{1,2}(v,H)\}$

Algorithm for Total Roman Domination in weighted block graph

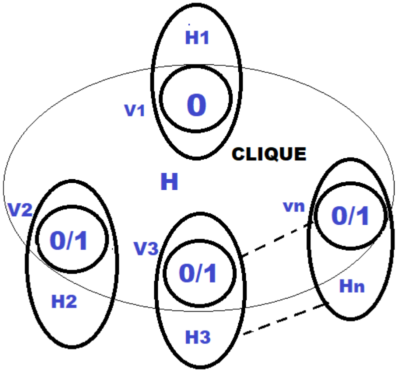


Figure: $dp_0^{0,1}(v_1, H)$

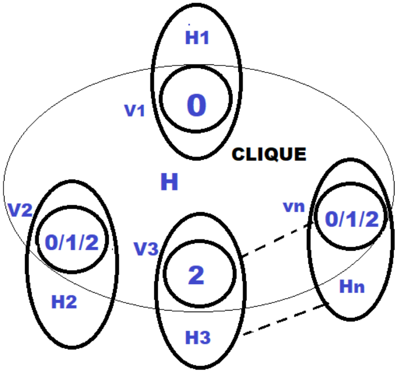


Figure: $dp_0^{2}(v_1, H)$

Algorithm for Total Roman Domination in weighted block graph

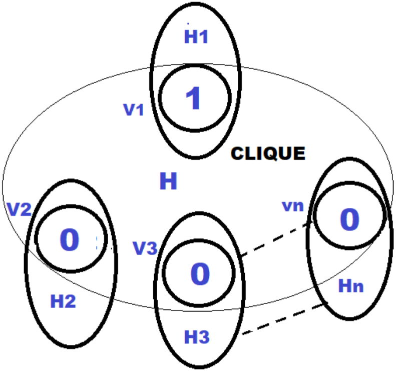


Figure: $dp_1^0(v_1, H)$

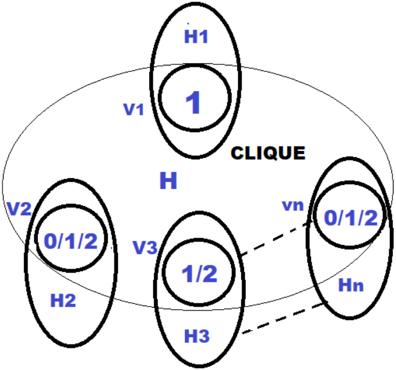


Figure: $dp_1^{1,2}(v_1, H)$

Algorithm for Total Roman Domination in weighted block graph

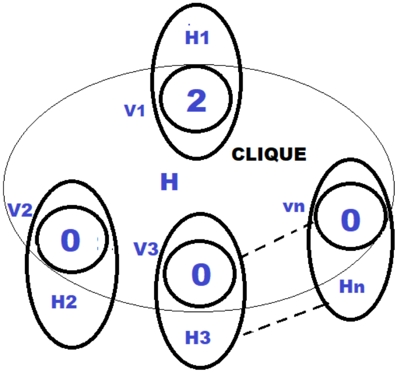


Figure: $dp_2^0(v_1, H)$

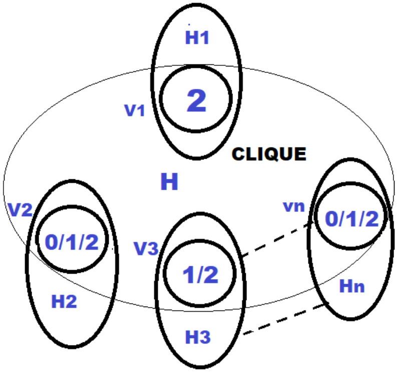


Figure: $dp_2^{1,2}(v_1, H)$

Algorithm for Total Roman Domination in weighted block graph

- Let $H_1, H_2, \ldots, H_n$, $n \geq 2$ be graphs rooted at $v_1, v_2, \ldots, v_n$, such that $v_i \in H_i$, $1 \leq i \leq n$. And, H is the graph rooted at v_1 obtained from the composition of the graphs $H_1, H_2, \ldots, H_n$, as shown above. Then, the following equalities hold:

1. $dp_0^{0,1}(v_1, H) = dp_0^{0,1}(v_1, H_1) + \sum_{i \in \{2,\ldots,n\}} \min\{dp_0^2(v_i, H_i), dp_1^{1,2}(v_i, H_i)\}$
2. $dp_0^2(v_1, H) = \min \begin{cases} \mathrm{dp}_0^2(v_1, H_1) + \sum_{i \in \{2,\ldots,n\}} \min\{dp_0^2(v_i, H_i),\ dp_1^{1,2}(v_i, H_i),\ dp_2^{1,2}(v_i, H_i)\} \\ dp_0^{0,1}(v_1, H_1) + S_0 \end{cases}$
3. $dp_1^0(v_1, H) = dp_1^0(v_1, H_1) + \sum_{i \in \{2,\ldots,n\}} dp_0^2(v_i, H_i)$

Algorithm for Total Roman Domination in weighted block graph

- $dp_1^{1,2}(v_1,H) = \min \begin{cases} \mathrm{dp}_1^{1,2}(v_1,H_1) + \sum\limits_{i\in\{2,\ldots,n\}} \min\{dp_0^2(v_i,H_i),\ A_i,\ B_i\} \\ dp_1^0(v_1,H_1) + S_1 \end{cases}$
- $dp_2^0(v_1,H) = dp_2^0(v_1,H_1) + \sum\limits_{i\in\{2,\ldots,n\}} \min\{dp_0^2(v_i,H_i), dp_0^{0,1}(v_i,H_i)\}$
- $dp_2^{1,2}(v_1,H) = \min \begin{cases} \mathrm{dp}_2^{1,2}(v_1,H_1) + \sum\limits_{i\in\{2,\ldots,n\}} \min\{A_i,\ B_i,\ C_i\} \\ dp_2^0(v_1,H_1) + S_2 \end{cases}$

- Where,
  - $S_0 = \min\limits_{i\in\{2,\ldots,n\}} \{dp_2^{1,2}(v_i,H_i) + \sum\limits_{j\in\{2,\ldots,n\}\setminus\{i\}} \min\{dp_0^2(v_j,H_j), dp_1^{1,2}(v_j,H_j), dp_2^{1,2}(v_j,H_j)\}\}$

Algorithm for Total Roman Domination in weighted block graph

- $S_1 = \min_{i \in \{2,\ldots,n\}} \{\min\{A_i,\ B_i\} + \sum_{j \in \{2,\ldots,n\} \setminus \{i\}} \min\{dp_0^2(v_j, H_j), A_j, B_j\}\}$,
- $S_2 = \min_{i \in \{2,\ldots,n\}} \{\min\{A_i,\ B_i\} + \sum_{j \in \{2,\ldots,n\} \setminus \{i\}} \min\{A_j, B_j, C_j\}\}$,
- $A_r = \min\{dp_1^0(v_r, H_r), dp_1^{1,2}(v_r, H_r)\}$,
- $B_r = \min\{dp_2^0(v_r, H_r), dp_2^{1,2}(v_r, H_r)\}$,
- $C_r = \min\{dp_0^{0,1}(v_r, H_r), dp_0^2(v_r, H_r)\}$

- Finally after calculating the optimal values $dp_0^{0,1}(v, H)$, $dp_0^2(v, H)$, $dp_1^0(v, H)$, $dp_1^{1,2}(v, H)$, $dp_2^0(v, H)$, $dp_2^{1,2}(v, H)$ for a cut vertex v of the block graph H, we can write,

$$\gamma_{tRW}(H) = \min\{dp_0^2(v, H), dp_1^{1,2}(v, H), dp_2^{1,2}(v, H)\} \tag{1}$$

An Example

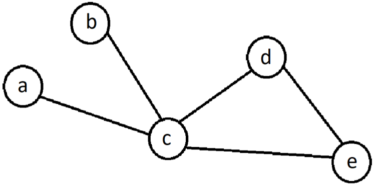


Figure: A block graph H

An Example

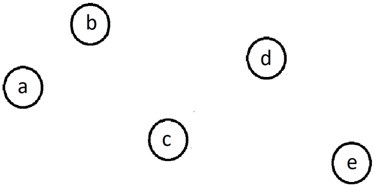


Figure: Initial trivial graphs

	$dp_0^{0,1}$	dp_0^2	dp_1^0	$dp_1^{1,2}$	dp_2^0	$dp_2^{1,2}$
(a, {a})	0	∞	1	∞	2	∞
(b, {b})	0	∞	1	∞	2	∞
(c, {c})	0	∞	1	∞	2	∞
(d, {d})	0	∞	1	∞	2	∞
(e, {e})	0	∞	1	∞	2	∞

An Example

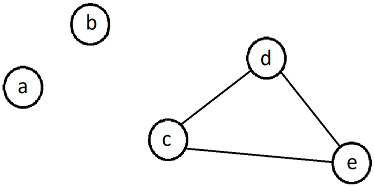


Figure: Step 1

	$dp_0^{0,1}$	dp_0^2	dp_1^0	$dp_1^{1,2}$	dp_2^0	$dp_2^{1,2}$
(a, {a})	0	∞	1	∞	2	∞
(b, {b})	0	∞	1	∞	2	∞
(c, {c, d, e})	∞	3	∞	3	∞	3
(d, {d})	0	∞	1	∞	2	∞
(e, {e})	0	∞	1	∞	2	∞

An Example

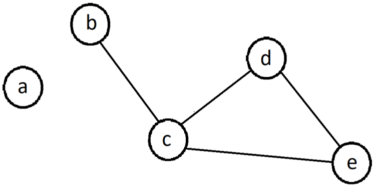


Figure: Step 2

	$dp_0^{0,1}$	dp_0^2	dp_1^0	$dp_1^{1,2}$	dp_2^0	$dp_2^{1,2}$
(a, {a})	0	∞	1	∞	2	∞
(b, {b})	0	∞	1	∞	2	∞
(c, {b, c, d, e})	∞	∞	∞	4	∞	3
(d, {d})	0	∞	1	∞	2	∞
(e, {e})	0	∞	1	∞	2	∞

An Example

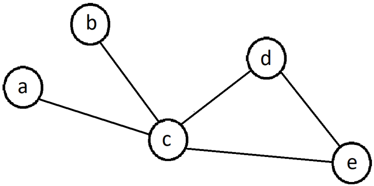


Figure: Step 3

	$dp_0^{0,1}$	dp_0^2	dp_1^0	$dp_1^{1,2}$	dp_2^0	$dp_2^{1,2}$
(a, {a})	0	∞	1	∞	2	∞
(b, {b})	0	∞	1	∞	2	∞
(c, {a, b, c, d, e})	∞	∞	∞	5	∞	3
(d, {d})	0	∞	1	∞	2	∞
(e, {e})	0	∞	1	∞	2	∞

An Example

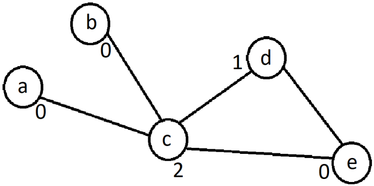


Figure: The solution to the assignment problem

Now,

$$\gamma_{tRW}(H) = min\{dp_0^2(c,H), dp_1^{1,2}(c,H), dp_2^{1,2}(c,H)\}$$
$$= min\{\infty, 5, 3\} = 3$$

Time Complexity of the Algorithm

For a graph G with V vertices and E edges

- **Time Complexity:** Construction of cut-point tree from a given block graph takes $O(|V| + |E|)$ time, which is given in [8]. We use DFS to traverse the Cut-point tree which takes $O(|V| + |E|)$. Hence, for the time complexity is $O(|V| + |E|)$.

Correctness of the Algorithm

- The problem of finding perfect Roman/Italian has two properties:
  1. Overlapping Subproblems
  2. Optimal Substructure
- We can prove the correctness of the Dynamic Programming algorithm by induction.
- **Induction Hypothesis:** Let the algorithm computes the Total Roman Domination number correctly for the block graphs $H_1, H_2, \ldots, H_n$.
- We try to show that the algorithm computes the Total Roman domination number correctly for the block graph H, where H is constructed from $H_1, H_2, \ldots, H_n$ using the method shown above.

Conclusion

- We have given linear time algorithms for Roman/Italian domination and Perfect Roman/Italian domination on weighted trees.
- We have given linear time algorithm for total Roman Domination on weighted trees.
- We have given linear time algorithms for Independent Roman Domination, Total Roman Domination and Perfect Italian Domination on weighted block graphs.

References I

[1] Ernie J. Cockayne, and Paul A. Dreyer
Roman domination in graphs.
Journal of Discrete Mathematics, 278:11–22, 2004.

[2] Mathieu Liedloff, and Ton Kloks
Efficient algorithms for Roman domination on some classes of graphs
Journal of Discrete Applied Mathematics, 156:3400–3415, 2000

[3] Michael A. Henning and William F. Klostermeyer and Gary MacGillivray
Perfect Roman Domination in trees.
Journal of Discrete Applied Mathematics, 236:235–245, 2018.

[4] Michael A. Henning, and William F. Klostermeyer
Italian domination in trees.
Journal of Discrete Applied Mathematics, 217:557–564, 2017.

References II

[5] Michael A. Henning, and Teresa W. Haynes
Perfect Italian domination in trees.
Journal of Discrete Applied Mathematics, 260:164–177, 2019.

[6] S. Banerjee, J. Mark Keil, D. Pradhan
Perfect Roman domination in graphs.
Theoretical Computer Science, 796:1–21, 2019.

[7] Decheng Wei, Changhong Lu
Independent Italian Domination on Block Graphs.

[8] D. B. West
Introduction to graph theory.
Upper Saddle River, NJ: Prentice hall, 1996.

Thank You