Linear algorithms on Steiner domination of trees✩

Chenglin Gao, Chengye Zhao

Science of College, China Jiliang University, Hangzhou, China

Abstract

A set of vertices $W$ in a connected graph $G$ is called a Steiner dominating set if $W$ is both Steiner and dominating set. The Steiner domination number $\gamma_{st}(G)$ is the minimum cardinality of a Steiner dominating set of $G$. A linear algorithm is proposed in this paper for finding a minimum Steiner dominating set for a tree $T$.

Keywords: linear algorithm, Steiner dominating set, Steiner domination number

1. Introduction

In this paper, we only consider finite, connected and undirected graph $G$. We refer to the books \[1, 2\] for notation and terminology on graph theory and theory of domination.

Let $G = (V(G), E(G))$ be a graph with the order of vertex set $|V(G)|$ and the order of edge set $|E(G)|$. The open neighborhood and the closed neighborhood of a vertex $v \in V$ are denoted by $N(v) = \{u \in V(G) : vu \in E(G)\}$ and $N[v] = N(v) \cup \{v\}$, respectively. For a vertex set $S \subseteq V(G)$, $N(S) = \bigcup_{v \in S} N(v)$, and $N[S] = \bigcup_{v \in S} N[v]$. The distance $d(u, v)$ between two vertices $u$ and $v$ of a connected graph $G$ is the length of shortest $u - v$ path in $G$. For a non-empty set $W$ of vertices in connected graph $G$, the Steiner distance $d(W)$ of $W$ is the minimum size of a connected subgraph of $G$ containing $W$. Obviously, each such subgraph is a tree and is called a Steiner tree or a Steiner $W$-tree. The set of all vertices of $G$ that lie on some Steiner $W$-tree

✩The research is supported by Natural Science Foundation of China (No.61173002).

Email address: cyzhao@cjlu.edu.cn (Chengye Zhao)

Preprint submitted to Elsevier

April 16, 2019
is denoted by $S(W)$. If $S(W) = V(G)$ then $W$ is called Steiner set of $G$. The Steiner number $s(G)$ is the minimum cardinality of a Steiner set.

Chartrand and Zhang introduced the concept of Steiner number of a connected graph $G$ in [3]. Pelayo corrected main result in [4]. He proved that not all Steiner sets are geodetic sets and there are connected graphs whose Steiner number is strictly lower than their geodetic number. Hernando et al. [5] have studied the relationships between Steiner sets and geodetic sets and between Steiner sets and monophonic sets. Many results on Steiner distance were given in [6, 7].

A subset $S$ of $V(G)$ is called dominating set if every vertex $v \in V$ is either a vertex of $S$ or is adjacent to a vertex of $S$. The domination number $\gamma(G)$ is the minimum cardinality of minimal dominating set of $G$. A systematic visit of each vertex of a tree is called a tree traversal. A set of vertices $W$ in a connected graph $G$ is called a Steiner dominating set if $W$ is both Steiner and dominating set. The Steiner domination number $\gamma_{st}(G)$ is the minimum cardinality of a Steiner dominating set of $G$.

The concept of Steiner domination was introduced in [8], and Vaidya et al. have obtained various results on Steiner domination numbers in [9, 10, 11].

The most algorithmic complexity of domination and related parameters of graphs are NP-complete or NP-hard problems. But there are many linear algorithms for domination and related parameters in trees, such as domination, total domination and secure domination in trees [12, 13, 14]. In this paper, we present a linear algorithm of Steiner domination in trees. It is similar to an algorithm due to Mitchell, Cockayne and Hedetniemi [15] for computing the domination number of an arbitrary tree.

2. Lemmas

A vertex of a graph $G$ is called a leaf or end-vertex if it is adjacent to only one vertex in $G$. A vertex $v$ is an extreme vertex if the subgraph induced by its neighbors is complete. Thus, every end-vertex is an extreme vertex.

**Lemma 2.1.** [3] Each extreme vertex of a graph $G$ belongs to every Steiner set of $G$. In particular, each end-vertex of $G$ belongs to every Steiner set of $G$.

The following corollary is an immediate consequence of Lemma 2.1.
Corollary 2.2. Every nontrivial tree with exactly k end-vertices has Steiner number k.

By Corollary 2.2 and Lemma 2.1 we have

Corollary 2.3. Let L(T) include all end-vertices of a tree T, then L(T) is a Steiner set of T.

Let $H = T[V - N[L(T)]]$ be the induced subgraph of $T$ from the set $V - N[L(T)]$. We have,

Theorem 2.4. For any nontrivial tree $T$, $\gamma_{st}(T) = |L(T)| + \gamma(H)$.

Proof. Let $S$ be a minimum dominating set of $H$ and $\gamma(H) = |S|$. By Corollary 2.3, L(T) is a Steiner set of $T$. Hence the set $S \cup L(T)$ is a Steiner dominating set of $T$ and $\gamma_{st}(T) \leq |L(T)| + \gamma(H)$.

Nextly, we prove $\gamma_{st}(T) \geq |L(T)| + \gamma(H)$. By contradiction, let $\gamma_{st}(T) < |L(T)| + \gamma(H)$ and there is a $\gamma_{st}$-set $S'$ such that $\gamma_{st}(T) = |S'|$. By Lemma 2.1, L(T) is a subset of each minimum Steiner set of $T$. Let $S'' = S' - L(T)$. By the definition of $H$, $S''$ is a minimum dominating set of $H$ such $|S''| = \gamma_{st}(T) - |L(T)| < \gamma(H)$, it is a contradiction. □

3. Linear algorithms for Forest Domination

In this section, we construct a linear algorithms for domination in forest. The algorithms is based on the algorithm for computing the domination number of an arbitrary tree by Mitchell, Cockayne and Hedetniemi [15].

By Theorem 2.1 the minimum Steinier dominating set of a tree is divided into two subsets: $L(T)$ and the $\gamma$-set of subgraph $H$ of $T$.

By the definition of $H$, $H$ is a tree or a forest. So the algorithm in [15] has to be changed for computing the domination number of a forest. Algorithm 1 for domination of a forest $F$, and each tree $T$ in $F$ is rooted. Two linear arrays are maintained during this traversal process:

Parent[i]: contains the index of the parent of vertex $i$ in a forest $F$; in the Parent array, that the Parent of a vertex labelled $i$ is given by Parent[i], and Parent[j]=0 if vertex $j$ is the root of a tree in $F$; for any vertex labelled $i$ in $F$, Parent[i]<i.

Label[i]: contains three states: 'Bound', 'Required' and 'Free'; the usage of Label array is similar to the algorithm in [15].
Compared with the algorithm in [15], we add the condition that $\text{Parent}[i] \neq 0$. This condition ensures that we construct the dominating set of each tree in $F$ by Algorithm 1 and get the minimum dominating set of a forest $F$.

**Algorithm 1 Forest Domination**

| Input: | input parameters a forest $F$ represented by an array $\text{Parent}[1..N]$ |
|--------|----------------------------------------------------------------------------------|
| Output: | output a minimum dominating set $D$ of $F$                                         |

1: \( D \leftarrow \emptyset \)
2: for \(i=1\) to \(N\) do
3: \( \text{Label}[i] = \text{'Bound'} \)
4: for \(i=N\) to 1 by -1 do
5: if \(\text{Label}[i] = \text{'Bound'} \) and \(\text{Parent}[i] \neq 0\) then
6: \(\text{Label}[\text{Parent}[i]] = \text{'Required'}\)
7: else
8: if \(\text{Label}[i] = \text{'Required'}\) then
9: \(D \leftarrow D \cup \{i\}\)
10: if \(\text{Label}[\text{Parent}[i]] = \text{'Bound'}\) then
11: \(\text{Label}[\text{Parent}[i]] = \text{'Free'}\)
12: for \(i=1\) to \(N\) do
13: if \(\text{Parent}[i]=0\) and (\(\text{Label}[i]=\text{'Bound'}\) or \(\text{Label}[i]=\text{'Required'}\)) then
14: \(D \leftarrow D \cup \{i\}\)

**Theorem 3.1.** *(Complexity of Algorithm 1).* If the input forest to Algorithm 1 has order $n$, then both the space complexity and the worst-case time complexity of Algorithm 1 are $O(n)$.

**Proof.** Step 1 can be performed in $O(1)$ time. Steps 2-3, 4-11, 12-14 are three for-loops, and each operation in these loops can be performed in $O(1)$ time. So the total operation time is $3n + 1 = O(n)$.

A total of $3n$ memory units are required to store the array $\text{Label}$, $\text{Parent}$ and the set $D$. Two memory units are required to store the values of the variables $i$ and $N$. The space complexity of Algorithm 1 is therefore $3n + 2 = O(n)$. □
4. Linear algorithms for Tree Steiner Domination

In this section, we construct a linear algorithms for Steiner domination in a tree. By Theorem 2.4, the definition of $H$ and Algorithm 1, we only consider the structures of $L(T)$ and $H$. Five linear arrays are maintained during this traversal process:

- **Parent**: contains the index of the parent of vertex $i$ in tree $T$; in the Parent array, that the Parent of a vertex labelled $i$ is given by Parent[$i$], and Parent[$i$]=0 if vertex $i$ is the root of $T$; for any vertex labelled $i$ in $T$, Parent[$i$]<$i$.
- **Flag**: Flag[$i$]=0 if the vertex $i$ is a end-vertex of $T$, else Flag[$i$]=1.
- **PFlag**: PFlag[$i$]=1 if the vertex $i$ is adjacent to a end-vertex of $T$, else PFlag[$i$]=0.
- **Index**: contains the index in $T$ of the vertex $i$ in $H$.
- **NParent**: contains the index of the parent of vertex $i$ in a forest $H$; in the Parent array, that the Parent of a vertex labelled $i$ is given by Parent[$i$], and Parent[$j$]=0 if vertex $j$ is the root of a tree in $H$; for any vertex labelled $i$ in $H$, Parent[$i$]<$i$.

By the steps 1-23 in Algorithm 2, we get $L(T)$ (the end-vertex set of $T$) and NParent array of $H = G[V − N[L(T)]]$. We obtain the $\gamma$-set of $H$ by the step 24 in Algorithm 2 (Nparent array as a input of Algorithm 1). Finally, we have a minimum Steiner dominating set of tree $T$ by the step 25 in Algorithm 2.

We conclude this section with a result on the space and time complexities of Algorithm 2.

**Theorem 4.1. (Complexity of Algorithm 2).** If the input tree to Algorithm 2 has order $n$, then both the space complexity and the worst-case time complexity of Algorithm 2 are $O(n)$.

**Proof.** Steps 1 and 25 can be performed in $O(1)$ time. Steps 2-4, 5-7, 8-10, 11-18, 19-23 are five for-loops, and each operation in these loops can be performed in $O(1)$ time. So the total operation time of four loops is $4n + m$. The operation time in step 24 is $O(m)$ by Theorem 3.1. So the total operation time is $4n + m + 2 + O(m) = O(n)$.

A total of $8n$ memory units are required to store the array Label, Parent, NParent, Flag, PFlag, Index, the set $D$ and $SD$. Three memory units are required to store the values of the variables $i$, $N$ and $m$. The space complexity of Algorithm 1 is therefore $8n + 3 = O(n)$. □
Algorithm 2 Tree Steiner Domination

Input: input parameters a tree $T$ represented by an array $\text{Parent}[1..N]$
Output: output a minimum Steiner dominating set $SD$ of $T$

1: $SD \leftarrow \emptyset$
2: for $i=1$ to $N$ do
3: \hspace{1em} Flag[$i$]=0
4: \hspace{1em} PFlag[$i$]=0
5: for $i=1$ to $N$ do
6: \hspace{1em} if $\text{Parent}[i] \neq 0$ then
7: \hspace{2em} Flag[$\text{Parent}[i]$]=1
8: for $i=1$ to $N$ do
9: \hspace{1em} if Flag[$i$] = 0 then
10: \hspace{2em} PFlag[$\text{Parent}[i]$]=1
11: for $i=1$ to $N$ do
12: \hspace{1em} $m = 0$
13: \hspace{1em} if Flag[$i$] = 0 then
14: \hspace{2em} $SD \leftarrow SD \cup \{i\}$
15: \hspace{1em} else
16: \hspace{2em} if PFlag[$i$] $\neq 1$ then
17: \hspace{3em} $m = m + 1$
18: \hspace{3em} Index[$m$]=$i$
19: \hspace{1em} for $i=1$ to $m$ do
20: \hspace{2em} if PFlag[$\text{Parent}[\text{Index}[i]]$] = 0 then
21: \hspace{3em} $\text{NParent}[i]$=$\text{Parent}[\text{Index}[i]]$
22: \hspace{2em} else
23: \hspace{3em} $\text{NParent}[i]$=0
24: \hspace{1em} Input $\text{NParent}$ as $\text{Parent}$ into Algorithm 1, and get the result $D$
25: $SD \leftarrow SD \cup D$
References

[1] D. B. West, Introduction to Graph Theory, 2e, Prentice - Hall of India, New Delhi, (2003).

[2] T. W. Haynes, S. T. Hedetniemi and P. J. Slater , Fundamentals of Domination in Graphs, Marcel Dekker, New York, (1998).

[3] G. Chartrand and P. Zhang, The Steiner Number of a Graph, Discrete Mathematics. 242(2002)41-54.

[4] I. M. Pelayo, Comment on The Steiner Number of a Graph by G. Chartrand and P. Zhang Discrete Mathematics 242, (2002), 41 - 54; Discrete Mathematics. 280(2004) 259 - 263.

[5] C. Hernando, T. Jiang, M. Mora, I. M. Pelayo and C. Seara, On the Steiner, Geodetic and Hull Number of Graphs, Discrete Mathematics. 293(2005)139 - 154.

[6] G. Chartrand, O. R. Oellermann, S. Tian and H. B. Zou, Steiner Distance in Graphs, Casopis Pro. Pest. Mat. 114(1989)399 - 410.

[7] A. P. Santhakumaran and J. John, The Forcing Steiner Number of a Graph, Discussion Mathematicae Graph Theory. 31(2011)171 - 181.

[8] J. John, G. Edwin and P. Arul Paul Sudhahar, The Steiner Domination Number of a Graph, International Journal of Mathematics and Computer Application Research. 3(3)(2013)37 - 42.

[9] S. K. Vaidya and S. H. Karkar, Steiner Domination Number of Some Graphs, International Journal of Mathematics and Scientific Computing. 5(1)(2015)1 - 3.

[10] S. K. Vaidya and R. N. Mehta, Steiner Domination Number of Some Wheel Related Graphs, International Journal of Mathematics and Scientific Computing. 5(2)(2015) 15-19.

[11] S. K. Vaidya and R. N. Mehta, On Steiner domination in graphs, Malaya Journal of Matematik. 6(2)(2018)381-384.
[12] E. J. Cockayne, S. E. Goodman and S. T. Hedetniemi, A linear algorithm for the domination number of a tree, Information Processing Letter. (4)(1975)41-44.

[13] R. C. Laskar, J. Pfaff, S. M. Hedetniemi and S. T. Hedetniemi, On the algorithmic complexity of total domination, SIAM J. Algebraic Discrete Methods. 5(1984)420-425.

[14] A. P. Burger, A.P. de Villiers and J. H. van Vuuren, A linear algorithm for secure domination in trees, Discrete Applied Mathematics. 171(2014)15-27.

[15] S. L. Mitchell, E. J. Cockayne and S. T. Hedetniemi, Linear algorithms on recursive representations of trees, J. Comput. Syst. Sci. 18(1979)76-85.