Connected Domination and Steiner Set on Asteroidal Triple-Free Graphs

Hari Balakrishnan Anand Rajaraman
C. Pandu Rangan*
Department of Computer Science and Engineering
Indian Institute of Technology, Madras 600 036, India.

Abstract

An ***asteroidal triple*** is a set of three independent vertices such that between any two of them there exists a path that avoids the neighbourhood of the third. Graphs that do not contain an asteroidal triple are called ***asteroidal triple-free (AT-free) graphs***. AT-free graphs strictly contain the well-known class of cocomparability graphs, and are not necessarily perfect. We present efficient polynomial-time algorithms for the minimum cardinality connected dominating set problem and the Steiner set problem on AT-free graphs. These results, in addition to solving these problems on this large class of graphs, also strengthen the conjecture of White, et. al. [9] that these two problems are algorithmically closely related.

Keywords: Design of algorithms, asteroidal-triple free graphs.

1 Introduction

An independent set of vertices x,y,z is an *asteroidal triple* if between every two of them there exists a path that avoids the neighbourhood of the third. An *asteroidal triple-free (AT-free) graph* is a graph that contains no asteroidal triples. AT-free graphs were first considered by Lekkerkerker and Boland [8] in the early 1960s.The following facts about the AT-free graphs are well-known (for the definitions of the italicized terms, see [7]).

- G is an *interval graph* if and only if it is *chordal* and AT-free [8].
- AT-free graphs are not necessarily *perfect*. Consider C_5, for example, which is AT-free but not perfect.
- Perfect AT-free graphs strictly contain the *cocomparability graphs*.

*E-mail : rangan@iitm.ernet.in

- AT-free graphs have a characterization in terms of forbidden subgraphs [3].

A *dominating set* S of a graph $G = (V, E)$ is defined as a set of vertices $S \subseteq V$ such that every vertex in V is either in S or is adjacent to some vertex in S. S is a *connected dominating set* of G if and only if S dominates G and the subgraph induced by S is connected.

For a subset R of V, we define a set S to be a *Steiner set* if

i. S is a subset of V,

ii. the subgraph induced by $(R \cup S)$ in G is connected and

iii. S is a set with least cardinality satisfying (i) and (ii).

We call the vertices of S *Steiner vertices* with respect to R. A spanning tree on the Steiner set is called a *Steiner tree*. We will henceforth denote an instance of the Steiner set problem by (G, R).

It is known that the minimum connected dominating set problem and minimum Steiner set problem are NP-complete for general graphs [6, 2]. Polynomial times have been reported in the literature for some special classes of graphs such as interval graphs, permutation graphs, strongly chordal graphs and distance hereditary graphs. [4, 1, 9, 5]. We mention here two special cases of the Steiner tree problem - if $|R| = 2$, then this reduces to the well-known *shortest path* problem; if $R = V$, then this reduces to the *minimum spanning tree* problem (in the weighted case). Polynomial algorithms are known for both these special cases on general graphs.

In this paper, we present algorithms to solve the Minimum Cardinality Connected Dominating Set (MCDS) and Steiner set problems on AT-free graphs. Throughout this paper, $P = u \cdots v$ denotes a (chordless) path P between vertices u and v. Depending on the context, P could also stand for the set of vertices on the path. $N_G(v)$ and $N_G[v]$ denote the open and closed neighbourhoods of the vertex v respectively. We often denote the graph induced by a subset S of V by S itself, instead of by $G(S)$. $dom(S)$ denotes the set of vertices dominated by an arbitrary $S \subseteq V$. Note that a set D is a dominating set of $G = (V, E)$ if and only if $dom(D) = V$.

2 Dominating Pairs in AT-free Graphs

A pair of vertices (u, v) is said to be a *dominating pair* of a graph G if every $u \cdots v$ path in G dominates all vertices of G. Corneil et. al. [3] have stated a property of AT-free graphs in terms of dominating pairs. We state their result below as a theorem :

Theorem 2.1 **([3])** *Every connected AT-free graph contains a dominating pair of vertices.*

We now present a simple algorithm to find all the dominating pairs in an arbitrary graph. This is needed as a preprocessing step in our later algorithms. It is clear that a pair of vertices (u, v) is a dominating pair of G if and only if for any vertex $x \in V$, u and v lie in different connected components of $G - N[x]$. (Otherwise, there would exist a $u \cdots v$ path that did not pass through the closed neighbourhood of x). The algorithm proceeds by removing $N[x]$ from G for every vertex x in V, and finding the connectedness status of every other pair of vertices in the resulting reduced graph. We declare all pairs (u, v) which lie in different components in every such reduced graph to be the dominating pairs of G.

Algorithm *Dominating Pairs*

Input : An AT-free graph $G = (V, E)$. $|V| = n$, $|E| = m$.
Output : A list of all the dominating pairs of G.
A, A' : $array[1..n, 1..n]$ of boolean entries.

begin
1. initialize $A[i, j]$ to *true* for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$;
2. for all vertices $v \in V$ do
3. construct $G' = G - N[v]$;
4. do a DFS of G' to find out its connected components;
5. for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$ do
. if vertices i and j are in the same connected component of G' then
$A'[i, j] \leftarrow false$
else
$A'[i, j] \leftarrow true$;
6. $A[i, j] \leftarrow A[i, j] \wedge A'[i, j]$;
7. for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$
if $A[i, j] = true$ then (i, j) is a dominating pair
end.

Theorem 2.2 *Algorithm* Dominating Pairs *above correctly finds all the dominating pairs of an AT-free graph G in $O(n^3)$ time.*

Proof. Correctness follows from the discussion above. We now prove the complexity bound. Step 1 takes time linear in n. Steps 3 and 4 can be also executed in linear time for every iteration. Each execution of step 5 takes $O(n^2)$ time. Hence each execution of the loop in step 2 takes $O(n^2)$ time; so step 2 requires $O(n^3)$ time for execution. Step 7 takes time linear in n. Hence the overall complexity is dominated by step 2 and is $O(n^3)$. □

3 Connected Domination on AT-free Graphs

In this section, we consider the problem of computing a minimum connected dominating set ($MCDS$) of an AT-free graph. This problem has been extensively studied on various classes of graphs [4, 2]. Arvind and Pandu Rangan [1] give an $O(m + n \log n)$ algorithm for this problem on the class of permutation graphs. No results are currently known for this problem on the class of cocomparability graphs. We present an efficient algorithm on AT-free graphs, which are a large superset of cocomparability graphs. This algorithm can be used on any AT-free graph, regardless of whether it is perfect or not.

Let (u, v) be a dominating pair of G and let $X = N(u)$ and $Y = N(v)$. For each $x \in X$, let $A_x \subseteq V$ be the set of all vertices in $V - \{u\}$ such that if $a \in A_x$, then $\{a, x\}$ dominates all vertices in X. Similarly, for each $y \in Y$ let $B_y \subseteq V$ be the set of all vertices in $V - \{v\}$ such that if $b \in B_y$, then $\{b, y\}$ dominates all vertices in Y. Define Γ as follows :

$$\begin{aligned}
\Gamma = \{P \mid\ & P = u \cdots v, && \text{or} \\
& P = u \cdots by, && \text{for } y \in Y, b \in B_y \text{ or} \\
& P = xa \cdots v, && \text{for } x \in X, a \in A_x \text{ or} \\
& P = xa \cdots by, && \text{for } x \in X, y \in Y, a \in A_x, b \in B_y\ \}
\end{aligned}$$

Theorem 3.1 *Every path $P \in \Gamma$ such that $|P|$ is minimum in Γ is an $MCDS$ of G.*

Proof. Let $P \in \Gamma$. We show first that P is a connected dominating set. Clearly P is connected. To show that $dom(P) = V$, we have to consider four cases :

Case 1: $P = u \cdots v$. In this case clearly $dom(P) = V$, since (u, v) is a dominating pair.

Case 2: $P = u \cdots by$, for $y \in Y$, $b \in B_y$. Note that $Q = Pv$ is a dominating path since (u, v) is a dominating pair. But $\{b, y\}$ dominates every vertex dominated by v, and so P is also a dominating path.

Case 3: $P = xa \cdots v$, for $x \in X$, $a \in A_x$. This case is symmetric to Case 2.

Case 4: $P = xa \cdots by$. In this case, note that $Q = uPy$ is a dominating set; but $\{a, x, b, y\}$ dominates every vertex dominated by $\{x, y\}$ and so P is a dominating path.

Next, we show that given any MCDS S, we can convert it into an MCDS S' such that $S' \in \Gamma$. Once again, we consider four cases:

Case 1: $u, v \in S$. In this case clearly $S' \in \Gamma$.

Case 2: $u \notin S, v \in S$. In this case since S dominates u, some vertex $x \in X$ is in S. Since S is connected, it has a $x \cdots v$ path P.
Subcase 2(a): If $S \neq P$, then consider $S' = P \cup \{u\}$. Clearly, S' contains a (u, v) path and is therefore a dominating set of G with size smaller than or equal to that of S, satisfying $S' \in \Gamma$.
Subcase 2(b): In this case, $S = P$. If the vertex a adjacent to x in P belongs to A_x, clearly $S \in \Gamma$. If not, then there is some $x' \in X$ such that $\{x, a\}$ does not dominate x'. So x' must be dominated by some other vertex a' on the $x \cdots v$ path P. Replace $\{x, a\}$ in S ($= P$) by $\{u, x'\}$ to obtain S' from S. Clearly, S is connected because u is adjacent to x', x' is adjacent to a' ($\in S$), and S itself was originally connected. Now, S' is a $u \cdots v$ path, and so $S' \in \Gamma$. Also note that $|S'| = |S|$; this shows that the required transformation is possible in this case as well.

Case 3: $u \in S, v \notin S$. This case is symmetric to Case 2.

Case 4: $u \notin S, v \notin S$. In this case consider x, x', a, a' as in case 2(b) and replace $\{x, a\}$ by $\{u, x'\}$ to get S'' which satisfies the conditions of Case 3. Using the same technique as in Case 3, this can be transformed into an $MCDS$ $S' \in \Gamma$.

So in all cases we can transform S to $S' \in \Gamma$ as desired. This proves the lemma. □

We present below an algorithm to find an MCDS of an AT-free graph G which relies on the above theorem.

Algorithm *Connected Domination*
Input: An AT-free graph $G = (V, E)$. $|V| = n, |E| = m$.
Output: An MCDS for G.

begin
1. Find a dominating pair (u, v) of G.
2. $X \leftarrow N(u), Y \leftarrow N(v)$.
3. for all $x \in X$ do
   construct A_x.
   Let $A \leftarrow \bigcup_{x \in X} A_x$.
4. for all $y \in Y$ do
   construct B_y.
   Let $B \leftarrow \bigcup_{y \in Y} B_y$.
5. do an All Pairs Shortest Paths on G and store the results in array D.
   i.e. $D[i, j]$ = length of a shortest (i, j) path.
6. Let

$l_1 \leftarrow D[u, v]$
$l_2 \leftarrow \min_{a \in A} D[a, v] + 1$
$l_3 \leftarrow \min_{b \in B} D[u, b] + 1$
$l_4 \leftarrow \min_{a \in A, b \in B} D[a, b] + 2$

7. $l \leftarrow min(l_1, l_2, l_3, l_4)$.
8. l gives the size of the MCDS. We can also easily construct the MCDS with the above information.

end.

Theorem 3.2 *Algorithm* Connected domination *correctly computes the MCDS of an AT-free graph G in $O(n^3)$ time using $O(n^2)$ space.*

Proof. The correctness of the algorithm follows from the previous theorem. The only thing to note is that we add 1 or 2 to l_2, l_3 and l_4 in step 6 to take care of the fact that the actual dominating path must contain a vertex from X or Y or both, as stated in the previous theorem.

We now prove the complexity bound. Step 1 can be done in $O(n^3)$ time using Algorithm *Dominating Pairs*. Step 2 takes linear time. Each iteration of steps 3 and 4 takes $O(n^2)$ time, and so steps 3 and 4 require $O(n^3)$ time. Step 5 takes $O(n^3)$ time using the standard All Pairs Shortest Paths algorithm. Step 6 takes $O(n^2)$ time because there are totally $O(n^2)$ pairs of distances to be compared. Step 7 takes constant time. Step 8 can be executed in $O(m+n)$ time using a BFS to find the actual shortest path given the endpoints of the path. Hence the overall time complexity of the algorithm is $O(n^3)$.

Computing the dominating pair requires $O(n^2)$ space (Theorem 2.2). X and Y take $O(n)$ space as do each A_x and B_y. Observe that we can avoid storing each A_x (B_y) explicitly by constructing A (B) directly, thus requiring only $O(n)$ space. The All Pairs algorithm requires $O(n^2)$ space, and the subsequent BFS requires $O(m+n)$ space. Hence the overall space requirement is $O(n^2)$. □

The above algorithm computes the minimum cardinality connected dominating set of an AT-free graph in $O(n^3)$ time. We have exploited the property that there exists an $MCDS$ that induces a path (or a cycle) in the algorithm. On permutation graphs, *every* minimum *weighted* connected dominating set induces a path or a cycle. This property has been exploited in [1] to solve the weighted version of the problem on permutation graphs in $O(m + n \log n)$ time. However on AT-free graphs it is not necessary that there be a minimum *weighted* connected dominating set that induces a path or a cycle. (See Figure 1.) Hence the same approach may not be extendable to the weighted case on AT-free graphs.

4 Steiner Set on AT-free Graphs

In this section we consider the problem of computing the Steiner set of an AT-free graph G, with respect to a set $R \subseteq V$ of vertices. The method used

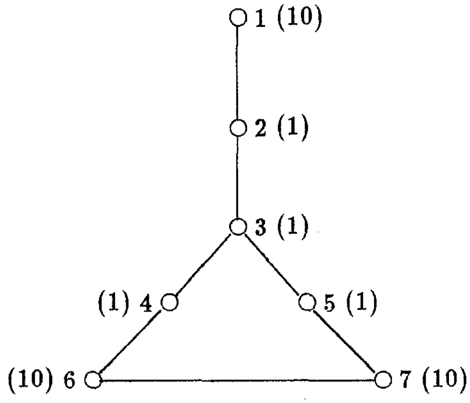

Figure 1: The weighted $MCDS = \{2, 3, 4, 5\}$ is not a path or a cycle (numbers in brackets are the weights of the vertices).

to solve this problem is similar to that used to solve the $MCDS$ problem in the previous section. It has already been observed that these two problems are algorithmically closely related [9] in the sense that they are either both polynomial or both NP-complete on all classes of graphs investigated so far. However, their precise relationship to each other is unknown. Arvind and Pandu Rangan [1] provide an $O(m + n \log n)$ algorithm for this problem on the class of permutation graphs. We present an $O(n^3)$ algorithm for the Steiner set problem on AT-free graphs.

We denote by R the set of vertices with respect to which the Steiner set is to be computed. Denote by $R_1, R_2, \ldots, R_k$ the k connected components of R. Let P be a dominating path in G between any dominating pair of vertices. Number the vertices of this path 1,2, ..., p, where $|P| = p$. Let the number assigned to any vertex v on the path be $number(v)$. Define
$min_R(P) = \{v | v \in P,\ v \in R$ or $(v, r) \in E$ for some $r \in R$, and $number(v)$ is minimum$\}$, and
$max_R(P) = \{v | v \in P,\ v \in R$ or $(v, r) \in E$ for some $r \in R$, and $number(v)$ is maximum$\}$.

We define an R-dominating pair of an AT-free graph with respect to any $R \subseteq V$ to be a pair of vertices (u, v) such that every path from u to v dominates all vertices of R. This is a generalization of the concept of a dominating pair defined previously, which is just a V-dominating pair of G.

Construct an auxiliary graph G_R from G in the following manner :
1. Replace all the vertices of each connected component R_i by a single vertex r_i, with the adjacency of r_i equal to the union of the adjacencies of all the vertices in R_i.

2. Choose any dominating path P corresponding to some dominating pair. Compute $u = min_R(P)$ and $v = max_R(P)$.
3. Delete all vertices of P that are not numbered in the range $[number(u),\ number(v)]$. Let P_R be the corresponding path obtained.
4. Delete all vertices of $G - R$ that are not adjacent to any vertex of P_R.
5. Delete all vertices of $G - R$ that are adjacent to either u or v.

The graph thus obtained is $G_R = (V_R, E_R)$, which has certain interesting properties.

Lemma 4.1 *G_R is AT-free.*

Proof. Straightforward. For if not, there exists an asteroidal triple (x, y, z) in G_R. Since G_R is constructed from G by only deleting certain vertices and edges, and adding no new edges, (x, y, z) is an asteroidal triple in G as well – a contradiction. □

Lemma 4.2 *$u = min_R(P)$ and $v = max_R(P)$ form an R-dominating pair in G_R.*

Proof. It is clear that the subpath P_R of $P = x \cdots y$ between u and v is a connected dominating set of G_R (this follows directly from the construction of G_R), and therefore dominates all vertices of R. If (u, v) is not an R-dominating pair of G_R, then there is some path P' between u and v, and some vertex $r \in R$ such that r is adjacent to no vertex on P'. Now consider the path $Q = (x \cdots u) \cup P' \cup (v \cdots y)$. Since (x, y) is a dominating pair of G, r must be adjacent to Q somewhere – moreover, this adjacency *must* be in the P'-segment of Q, since u and v are the smallest and largest numbered vertices on P to which any vertex of R is adjacent. This completes the proof. □

4.1 The Algorithm

We now present the actual algorithm for the Steiner set.

Algorithm *Steiner-Set*

Input: An AT-free graph $G = (V, E)$ and a set $R \subseteq V$ of vertices.

Output: A minimum cardinality Steiner set of G with respect to R.

begin

1. Construct the graph G_R as explained previously.
2. Let $u = min_R(P)$ and $v = max_R(P)$.
3. Let $X = N(u) \cap R$ and $Y = N(v) \cap R$.
4. As in the $MCDS$ algorithm, construct sets A_x and B_y. Use these to construct the sets A and B.
5. Construct a weighted graph $G' = (V', E')$ from G_R in the following manner: $wt(w) = 0$ if $w \in R$, and 1 otherwise.
6. As in the $MCDS$ algorithm, compute the shortest path among the four

different kinds of paths using the All Pairs Shortest Paths algorithm.
7. Let P_m be the minimum weighted path.
8. Output $S = P_m - R$
end.

4.2 Proof of correctness

Theorem 4.1 *Algorithm Steiner-Set presented above correctly computes the minimum cardinality Steiner set of an AT-free graph G with respect to a set $R \subseteq V$ of vertices.*

Proof. Let S be a Steiner set. If S is not already in a form that can be the output of the above algorithm, we show how it can be transformed to such a form, S'. Note that from the algorithm, (u, v) is an R-dominating pair of G_R, and that no two vertices of R are adjacent (since we have replaced each connected component of R by a single vertex). It is evident that this replacement does not affect the Steiner set – the same set is the output here as well. Since the weight of only those vertices in R is 0, and the others 1, it follows that a shortest weighted path will be one that minimises the number of vertices not in R, as we require. From the method of construction of the output set, S' in the above algorithm, it is apparent that $R \cup S'$ is *connected.* Define $T' = R \cup S'$. To show minimality, we distinguish the following cases :

Case 1. u and v are both in S.
In this case, the minimum weight path between u and v in $R \cup S$ sets a lower limit on the size of the Steiner set. The above algorithm attains this limit, and so is correct.

Case 2. $u \notin S$, $v \in S$.
We show that S can be transformed to a set S' of same cardinality that the above algorithm can produce. We distinguish two sub-cases:
Sub-case 2(a). Consider $T = R \cup S$. If in T, $r \in X$ is adjacent to some $a \in A_r$, we are done, because the above algorithm definitely considers a minimum weight A_r to v path.
Sub-case 2(b). $r \in X$ is adjacent to no $a \in A_r$.
Since T is connected, r is adjacent to some $a' \notin A$. In addition, $a' \notin R$, since each r is maximally connected, and so $wt(a') = 1$. The set X consists only of vertices of R (by construction); if r is the only vertex in X, we are done since this sub-case doesn't apply at all. Therefore, there is a vertex $r' \in X$, not adjacent to r, which is connected to S in some manner, say, to vertex $s \in S$. Replace $a' \in S$ by u to obtain S' from S. Now, $T' = R \cup S'$ is connected and also dominates all the vertices of R, since (u, v) is an R-dominating pair of G_R. In addition, we have replaced one vertex of unit weight (a') by another (u);

therefore $|S'| = |S|$. In our algorithm we find a minimum weight $u \cdots v$ path, which sets a lower limit on the size of the Steiner set – a limit which we actually attain. Hence the required transformation is possible in this case too.

Case 3. $u \in S$, $v \notin S$.
This is symmetric to *Case 2* considered above, with u and v interchanged.

Case 4. Both u and $v \notin T'$.
In this case, as in cases 2 and 3, replace a by u and b by v to get a Steiner set of the same cardinality as S'. Formally, we can replace one of them and argue as in Case 2 (or 3).

Thus, in all cases, we can transform any Steiner set into one capable of being output by the above algorithm. This completes the proof. □

4.3 Complexity Analysis

We now analyse the complexity of our *Steiner-Set* algorithm.
Step 1 takes $O(n^3)$ time since we require to determine a dominating pair. Steps 2 and 3 can be done in linear time. As in the $MCDS$ algorithm, Step 4 requires $O(n^3)$ time. Step 5 is trivially linear, while Step 6 can be done in $O(n^3)$ time using the standard algorithm. Once again, the vertices on the path can be obtained in $O(m+n)$ time using a BFS, as we did in the $MCDS$. Thus, we see that the overall time complexity of the algorithm is $O(n^3)$. The algorithm can be implemented using $O(n^2)$ space, since we only require to store all graphs in the adjacency matrix form. The precise details are similar to those explained in the $MCDS$ algorithm.

Theorem 4.2 *The minimum cardinality Steiner set problem can be solved on AT-free graphs in $O(n^3)$ time, using $O(n^2)$ space.*

Proof. Follows from the discussion above. □

5 Conclusion

In this paper, we have presented $O(n^3)$ algorithms on AT-free graphs for the problems of connected domination and Steiner set. To the best of our knowledge, no algorithms have so far appeared in the literature for solving these problems on any non-trivial class of non-perfect graphs. As already noted, AT-free graphs are not necessarily perfect. These algorithms also generalize the algorithms presented in [1] for connected domination and Steiner sets on permutation graphs. In addition, this is the largest known class of graphs that strictly contains both interval and permutation graphs on which these problems have been solved. However, we have not been able to generalize the algorithms for the weighted case, which remain open. We also note that the algorithms presented here are actually applicable to a class of graphs *larger* than the class of AT-free graphs – more specifically, to the class of graphs that contain a dominating pair of vertices. This class is a proper superset of the class of AT-free graphs (e.g., C_6 belongs to this class, but is not AT-free). Our results also strengthen the conjecture in [9] that the connected domination and Steiner set problems are algorithmically closely related. However, the precise relationship between these two problems is unknown in general. We also feel that AT-free graphs have nice structural properties which can be exploited to design algorithms for several problems which are NP-complete on general graphs.

References

[1] K. Arvind and C. Pandu Rangan. Connected domination and Steiner set on weighted permutation graphs. *Info. Proc. Letts.*, 41:215–220, 1992.

[2] C.J. Colburn and L.K. Stewart. Permutation graphs: Connected domination and Steiner trees. *Discrete Math.*, 86:145–164, 1990.

[3] D.G. Corneil, S. Olariu, and L. Stewart. Asteroidal triple-free graphs. Technical Report 262/92, University of Toronto, June 1992.

[4] D.G. Corneil and L.K. Stewart. Dominating sets in perfect graphs. *Discrete Math.*, 86:179–189, 1990.

[5] A. D'Atri and M. Mascarini. Distance hereditary graphs, Steiner trees and connected domination. *SIAM J. Computing*, 17:521–538, 1988.

[6] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* Freeman, San Fransisco, CA, 1979.

[7] M.C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs.* Academic Press, 1980.

[8] C.G. Lekkerkerker and J.C. Boland. Representation of a finite graph by a set of intervals on a real line. *Fundamenta Mathematicae*, 51:45–64, 1962.

[9] K. White, M. Farber, and W.R. Pulleybank. Steiner trees, connected domination and strongly chordal graphs. *Networks*, 15:109–124, 1985.