How to guard a graph against tree movements

Toshihiro Fujito[1] **Takayoshi Sakamaki**[1]

[1] Department of Computer Science and Engineering
Toyohashi University of Technology
Toyohashi 441-8580 Japan
Email: {fujito@,sakamaki@algo.}cs.tut.ac.jp

Abstract

This paper shows that the optimization problem arising from the guarding game played between the cop and robber players on an undirected graph can be approximated within a factor of $\Theta(\log n)$ when the robber region is a tree.

Keywords: Guarding game, Set cover, Greedy approximation.

1 Introduction

The *guarding game* is a combinatorial game played by two players, the robber player and the cop player, on a graph $G = (V, E)$ in which the vertex set V is partitioned into the robber region R and the cop region C. The robber player has a single pawn, a robber, while the cop player may use several pawns, cops. The players alternate their turns, starting with the robber player in which she places the robber on some vertex in R, followed by the cop player in which she places all of her cops on vertices in C. Throughout the game playing each player can see every corner of G so she knows on which vertex every pawn is currently located. Each player is allowed to move any number of her pawns in her turn, each of them from vertex v it currently sits to one of v's neighboring vertices in G. The robber, however, cannot move to a vertex occupied by a cop as the robber would be caught by a cop by doing so. On the other hand, any cop is not allowed to enter the robber region R. The goal of the robber player is that the robber eventually moves into the cop region C whereas that of the cop player is to prevent it. A *state* of the game is given by the positions of all the pawns in G. Each player moves her pawn(s) according to some *strategy*, which is a function mapping each state, i.e., (current) positions of all the pawns in G, to (next) positions of her own pawn(s) to which they can move from the given state. A robber (cop) strategy is *winning* if, when the robber (cop, resp.) player plays according to it, she wins against all the possible cop (robber, resp.) strategies.

Supported in part by a Grant in Aid for Scientific Research of the Ministry of Education, Science, Sports and Culture of Japan.

The *guarding problem*, introduced in (Fomin et al. 2008), is the problem of computing the minimum number of cops required to protect the cop region C against all the possible robber strategies. More formally, it is to compute the minimum number of cops, given $G = (V, E)$ and bipartition (R, C) of V, such that there exists a winning strategy of the cop player when the guarding game is played on G and (R, C) with that number of cops. Fomin et al. showed in (Fomin et al. 2008) that

- When R induces a path in G, the guarding problem can be solved in polynomial time by reduction to the min cost flow problem.
- When R induces a cycle in G, the guarding game can be approximated in polynomial time within a factor of 2.
- When R induces a tree in G, the guarding problem is NP-hard, the decision version is in NP, and the parameterized version is $W[2]$-hard. Moreover, the set cover problem can be reduced to the guarding problem with R being a tree in approximation preserving manner, implying that the approximation ratio guaranteed in polynomial time is $\Omega(\log n)$ for the latter problem unless NP $\subseteq$ DTIME($n^{\text{poly} \log n}$) (Lund et al. 1994).
- When G is a directed graph and R is a directed acyclic graph, the guarding problem is PSPACE-complete.

Following the work listed above, it was shown in (Thirumala Reddy et al. 2009) that

- When R induces an arbitrary graph, the decision version of the guarding problem is PSPACE-hard.
- When R induces a wheel graph the decision version is NP-hard.
- When R induces a star graph, a clique, and a wheel graph, the guarding problem can be approximated within factors of $H(|R|), 2H(|R|)$ and $H(|R|) + 3/2$, respectively, where $H(k) = 1 + 1/2 + \cdots + 1/k$.

It was Nagamochi (Nagamochi 2011) who came up with an affirmative answer to the question of polynomial solvability, left open in (Fomin et al. 2008), when R induces a cycle.

In this paper we consider the case when R induces an arbitrary tree, and show that the guarding problem can be approximated, in such a case, within a factor of $H(|R|)$. Thus, this shows that the same approximation bound of $H(|R|)$ obtained in (Thirumala Reddy et al. 2009) for the case of R being a star graph can be extended to the case of an arbitrary tree. Moreover, when our result is combined with the hardness results of (Lund et al. 1994) and (Fomin et al. 2008), the polynomial time approximation guarantee for the case of an arbitrary tree becomes $\Theta(\log n)$.

2 Point Strategies and Direct Strategies

In this section the approach shown in Fomin et al. (2008) for the case when the robber region is a path will be extended to the case when it is a tree.

A cop strategy is called a *point* strategy if the next moves of cops are determined solely by the vertex (point) where the robber currently is, and not on where the cops are. Let X_i be the multiset of vertices of C on which the cops are placed, according to some point strategy, when the robber is located at vertex r_i for $i \in \{1, \cdots, n_1\}$, where $n_1 = |R|$. (X_i is a multiset as more than one cop can sit on a vertex). Then, this point cop strategy can be fully specified by these X_i's. Conversely, let X_i be a multiset of vertices of C for $i \in \{1, \cdots, n_1\}$ such that (1) $\|X_i\| = \|X_j\|$ for any $i, j \in \{1, \cdots, n_1\}$ (where $\|X\|$ denotes the total counts of elements in X), and (2) the cops, if positioned at X_i, can change their positions to X_j in one step whenever (r_i, r_j) is an edge in $G[R]$. Then, a point cop strategy can be realized by placing the cops at X_i in response to the robber's move to $r_i \in R$, $\forall i \in \{1, \cdots, n_1\}$. For these reasons, any point strategy will be often denoted by a sequence, $\langle X_i \mid i = 1, \cdots, n_1\rangle$, of X_i's satisfying the two conditions above.

Let us assume from now on that the robber region is a tree T. A robber strategy is called *direct* if the robber-player places her robber at the root of T, and then keeps moving it in one-way direction towards a leaf of T while entering the cop region C whenever possible (thus, the robber visits vertices in T by getting there *directly* without detour).

Consider any cop strategy that can guard the cop region C against *any* direct robber strategy. Let X_i be the multiset of vertices of C on which the cops are placed when the robber moves to vertex r_i following some direct strategy. For any vertex r_i in T there exists a direct strategy that tries to move the robber to r_i, and as the robber cannot enter the protected region C (so it cannot go anywhere else but within T), the robber actually gets there. Moreover, it makes identical moves before arriving at r_i under any of such direct strategies, and hence, X_i is well-defined as well as uniquely determined by r_i.

Suppose r_i is a parent of r_j in T. Then, a team of cops positioned at X_i can move to the new position X_j in one turn. But then, it is also possible for the team to move backward, from X_j to X_i in one turn, as the graph is undirected. Thus, such X_i's give rise to the point cop strategy $\langle X_i \mid i = 1, \cdots, n_1\rangle$.

Lemma 1. *The minimum number of cops required by a point winning strategy to protect the cop region C is no larger than that of cops required by any winning strategy to protect C.*

Proof. As shown above, any cop strategy that can protect C at least against all the direct strategies implies the existence of a point strategy that can protect C against all the robber strategies and that can do so with the same number of cops. Therefore, if any cop strategy uses the number of cops less than the minimum number of cops required by a point winning strategy, such a strategy cannot protect even against all the direct strategies. □

In light of this lemma, approximation of the minimum cop strategy can be reduced to that of the minimum point cop strategy, and the latter is the one we do next.

3 Reduction to Set Cover

We use an auxiliary graph A as in Fomin et al. (2008). Let each of $C^1, C^2, \cdots, C^{n_1}$ be a copy of C, each of C^i corresponding to vertex r_i in T. For each vertex $c_k \in C$, C^i has its copy c^i_k in it. The vertices in C^i are connected to those in C^j iff (r_i, r_j) is an edge in T (thus, each C^i is an independent set in A), and between such C^i and C^j, $c^i_k \in C^i$ and $c^j_l \in C^j$ are connected by an edge iff either $k = l$ or (c_k, c_l) is an edge in $G[C]$. So, C^i and C^j are connected, if (r_i, r_j) is an edge in T, by an edge set $E^{ij} = \{(c^i_k, c^j_l) \mid \text{either } k = l \text{ or } (c_k, c_l) \text{ is an edge in } G[C]\}$. The auxiliary graph A is now specified by vertex set $\bigcup_{r_i \in V(T)} C^i$ and edge set $\bigcup_{(r_i, r_j) \in E(T)} E^{ij}$.

Let $\langle X_i \mid i = 1, \cdots, n_1\rangle$ be a point cop strategy. Recall that each X_i is a multiset of vertices of C, and from now on, we set X_i to the corresponding multiset of vertices of C^i for each $i \in \{1, \cdots, n_1\}$. Recall again that, when (r_i, r_j) is an edge in T, all the cops positioned at X_i can get in one step to positions X_j (and vice versa), that is, there is a matching within E^{ij} between vertices of X_i and X_j. Pick any cop to be used in $\langle X_i \mid i = 1, \cdots, n_1\rangle$. It is straightforward to observe that 1) the cop occupies some unique vertex within each C^i, and 2) for each edge (r_i, r_j) of T the vertex occupied by the cop in C^i and the one in C^j are connected by an edge in A. Therefore, those vertices occupied by a single cop induce a tree in A having exactly one vertex in each C^i, and thus, any point cop strategy induces a collection of such trees in A, where the number of trees is that of cops used in the strategy. Let us call such a tree in A *proper*, i.e., the one having exactly one vertex in C^i for all $i \in \{1, \cdots, n_1\}$. The other direction is also easy to observe; any collection of k proper trees in A specifies a point cop strategy using k cops, where each proper tree specifies the move of a single cop entirely corresponding to the robber's move on T.

Now that the equivalence between a point cop strategy and a collection of proper trees is established, let us consider next the property to be satisfied by a point strategy so that it becomes winning, and the corresponding property to be satisfied by a collection of proper trees. Let Γ_i denote the set of all the vertices in C adjacent to r_i in T. Whenever the robber moves to r_i in T in any robber player's turn, unless all the vertices in Γ_i become occupied by cops in the following turn of the cop player, the robber can enter C in the next turn, and thus the cop player loses. Conversely, if the cop player can immediately position cops at all the vertices in Γ_i whenever the robber moves to r_i for any $r_i \in T$, then the cop player never loses, which is a win for her. Therefore, a point cop strategy $\langle X_i \mid i = 1, \cdots, n_1\rangle$ is winning iff $\Gamma_i \subseteq X_i$ for all $r_i \in T$, and this corresponds to the condition for a collection of proper trees in A such that every vertex in $\Gamma_i \subseteq C^i$ is contained in at least one of the proper trees. The guarding problem thus can be reduced to the problem of computing the minimum collection of proper trees in A, given A and $\Gamma_i \subseteq C^i$ for each $i \in \{1, \cdots, n_1\}$, such that every vertex in $\bigcup_{i \in \{1, \cdots, n_1\}} \Gamma_i$ is contained in at least one of proper trees in the collection. Since this is the problem of "covering" all the vertices in $\bigcup_{i\in\{1,\cdots,n_1\}}\Gamma_i$ by the smallest number of proper trees in A, it can be considered to be the set cover problem:

Lemma 2. *The guarding problem can be reduced to the set cover problem in which the family of all the proper trees in A (or their vertex sets to be more precise) is the family of subsets, and the vertices in $\bigcup_{i\in\{1,\cdots,n_1\}}\Gamma_i$ are those elements to be covered by subsets.*

(Note: there is no need to cover any vertex not in any of Γ_i's, and although the standard set cover does not include such elements not needed to be covered, we may simply ignore the existence of such elements).

4 Greedy Approximation

It is well known that the best performance guarantee in approximating the set cover problem is attained by the greedy algorithm, in which the subset containing the largest number of uncovered elements in it is repeatedly found and chosen into the solution, until all the elements become covered. The approximation ratio guaranteed by the greedy algorithm is $H(s_{\max})$, where $s_{\max}$ is the size of the largest subset in a given instance (Johnson 1974, Lovasz 1975).

To run the greedy algorithm on A and $\Gamma_i \subseteq C^i$ for each $i \in \{1,\cdots,n_1\}$, one needs to find a proper tree containing the maximum number of vertices, not yet covered by already chosen proper trees, in $\bigcup_{i\in\{1,\cdots,n_1\}}\Gamma_i$. Notice that one cannot go over all the proper trees existing in A to find such a proper tree simply because there are too many of them. It is possible though to efficiently compute the best proper tree by a rather simple algorithm, which we will describe below.

Fix any vertex in T, say r_1, as its root, and order the vertices in C^i's accordingly. That is, if r_i is the parent of r_j in T, $c^i_k \in C^i$ is the parent of $c^j_l \in C^j$ when $(c^i_k, c^j_l) \in E^{ij}$. The descendant/ascendant relation is also defined accordingly. Let us color all the vertices in $\bigcup_{i\in\{1,\cdots,n_1\}}\Gamma_i$ red, and all the others white.

1. For each vertex c^i_k in A we compute the maximum number, denoted s^i_k, of red vertices that can be covered by a proper subtree rooted at c^i_k and the subtree itself, denoted T^i_k, containing that many red vertices. Suppose that the computation of all s^j_l's and all T^j_l's is completed within C^j for each child r_j of r_i (Initially, r_i is such a vertex only if it is a leaf in T). Then, we set

$$s^i_k = \sum_{j:r_j \text{ is a child of } r_i \text{ in } T} \max\{s^j_l \mid (k,l) \in E^{ij}\} + \delta$$

for each c^i_k in C^i, where $\delta = 1$ if c^i_k is red and $\delta = 0$ otherwise. At the same time T^i_k is composed by connecting, via c^i_k, all of T^j_l's with the corresponding s^j_l's each attaining the maximum value in the summation.

2. Eventually, s^i_k's and T^i_k's will be computed for all c^i_k's in A, the last ones being for those vertices in C^1. Then, T^1_k with s^1_k being the largest among all the s^1 values in C^1 is a proper tree containing the maximum number of red vertices.

3. Choose T^1_k into our solution, and before going for the next search, recolor all the red vertices in T^1_k to white.

4. Repeat all the above as long as there remains a red vertex in A.

The correctness of this algorithm can be easily verified by the mathematical induction on the number of vertices in T.

To find one proper tree containing the maximum number of red vertices (i.e., those still uncovered in $\bigcup_{i\in\{1,\cdots,n_1\}}\Gamma_i$), besides initializing $s^i_k = 0$ and T^i_k = NULL for all the "leaves" C^i's and all c^i_k's in C^i, the algorithm traverses every edge in A exactly once in the computation of s^i_k's and T^i_k's. Hence, it takes $O(|R|(|C| + |E(G[C])|))$ to find such a tree, where $O(|R||C|)$ time for initialization and $O(\sum_{(r_i,r_j)\in E(T)} |E^{ij}|) = O(|R|(|C| + |E(G[C])|))$ for the others. As there exist at most $|R||C|$ red vertices to be covered at the outset, the overall running time of the greedy algorithm is $O(|R|^2|C|(|C| + |E(G[C])|))$.

In each iteration the algorithm above chooses a proper tree containing the maximum number of red vertices, and as any proper tree cannot contain more than $|R|$ red vertices in it, we have

Theorem 3. *The guarding problem can be approximated, by the greedy algorithm presented above, with approximation ratio of $H(|R|) \leq \log|R| + 1$ when R induces a tree.*

When this theorem is combined with reduction of the set cover problem to the guarding problem with tree R (Fomin et al. 2008) and the lower bound for set cover approximation (Lund et al. 1994), we have

Corollary 4. *When R induces a tree the approximation ratio that can be guaranteed in polynomial time for the guarding problem is $\Theta(\log n)$ unless NP $\subseteq$ DTIME($n^{poly \log n}$).*

References

Fomin, F.V., Golovach, P.A., Hall, A., Mihalák, M., Vicari, E. & Widmayer, P. (2008), How to guard a graph ?, *in* 'ISAAC', LNCS Vol. 5369, pp. 318–329.

Thirumala Reddy, T., Sai Krishna, D. & Pandu Rangan, C. (2009), The Guarding Problem — Complexity and Approximation, *in* 'IWOCA', LNCS Vol. 5874, pp. 460–470.

Nagamochi, H. (2011), 'Cop-robber guarding game with cycle robber-region', *Theoretical Computer Science* **412**(4-5), 383–390.

Lund, C. & Yannakakis, M. (1994), 'On the hardness of approximating minimization problems', *J. of the ACM* **41**(5), 960–981.

Johnson, D.S. (1974), 'Approximation algorithms for combinatorial problems', *J. of Computer and System Sciences*, **9**, 256–278.

Lovász, L. (1975), 'On the ratio of optimal integral and fractional covers', *Discrete Mathematics*, **13**, 383–390.