How to Guard a Graph?

Algorithmica (2011) 61:839–856
DOI 10.1007/s00453-009-9382-4

How to Guard a Graph?

Fedor V. Fomin · Petr A. Golovach · Alex Hall · Matúš Mihalák · Elias Vicari · Peter Widmayer

Received: 8 December 2008 / Accepted: 21 December 2009 / Published online: 6 January 2010

Abstract We initiate the study of the algorithmic foundations of games in which a set of cops has to guard a region in a graph (or digraph) against a robber. The robber and the cops are placed on vertices of the graph; they take turns in moving to adjacent vertices (or staying). The goal of the robber is to enter the guarded region at a vertex with no cop on it. The problem is to find the minimum number of cops needed to prevent the robber from entering the guarded region. The problem is highly non-trivial even if the robber's or the cops' regions are restricted to very simple graphs. The computational complexity of the problem depends heavily on the

F.V. Fomin and P.A. Golovach are partially supported by the Norwegian Research Council.
E. Vicari and P. Widmayer are partially supported by the National Competence Center in Research on Mobile Information and Communication Systems NCCR-MICS, a center supported by the Swiss National Science Foundation under grant number 5005-67322.

F.V. Fomin
Institute of Informatics, University of Bergen, Bergen, Norway
e-mail: fedor.fomin@ii.uib.no

P.A. Golovach
School of Engineering and Computing Sciences, Durham University, Durham, UK
e-mail: petr.golovach@durham.ac.uk

A. Hall
Google, Zurich, Switzerland
e-mail: alex.hall@gmail.com

M. Mihalák (✉) · E. Vicari · P. Widmayer
Institute of Theoretical Computer Science, ETH Zurich, Universitaetstrasse 6, 8092 Zurich, Switzerland
e-mail: matus.mihalak@inf.ethz.ch

E. Vicari
e-mail: vicariel@inf.ethz.ch

P. Widmayer
e-mail: widmayer@inf.ethz.ch

chosen restriction. In particular, if the robber's region is only a path, then the problem can be solved in polynomial time. When the robber moves in a tree (or even in a star), then the decision version of the problem is NP-complete. Furthermore, if the robber is moving in a directed acyclic graph, the problem becomes PSPACE-complete.

Keywords Game of Cops and Robbers · Guarding of vertices · Winning strategy · Complexity · Algorithms · Approximation

1 Introduction

Chases and escapes, or pursuits and evasions, are activities which are firm parts of our everyday or fantasy life. When young we are playing the game of tag or the game of hide-and-seek and watch Tom & Jerry cartoons. Pursuit-evasion problems remain fascinating for most of us even when we grow older, and it is not surprising that in half of the Hollywood movies good guys are chasing the bad guys, while in the remaining half bad guys are chasing the good ones.

The mathematical study of pursuit-evasion games has a long history, tracing back to the work of Pierre Bouguer, who in 1732 studied the problem of a pirate ship pursuing a fleeing merchant vessel. We refer to *Differential games* by Rufus Isaacs [1], the now classic book on pursuit-evasion games, published in 1965 (and reprinted by Dover in 1999), for a nice introduction to the area. One of the general pursuit-evasion problems (mentioned in Isaacs' book) is the guarding-the-target problem. There are many variations of this problem like the deadline game, patrolling a channel, and patrolling a line. In this problem, both pursuer and evader travel with the same speed. The goal of the pursuer is to guard a target C which is an area in the plane, from attacks by the evader. In Isaacs' military conception of this problem, the evader must actually reach at least the boundary of the target to be successful in her attack.

In this paper we study the *cop-robber guarding game* (*guarding game* for short), a discrete version of the guarding-the-target problem. We use the setting from the classic pursuit-evasion game on graphs called Cops and Robbers [2, 3].

The guarding game is played on a (directed or undirected) graph $G = (V, E)$ by two players, the *cop-player* and the *robber-player*, each having her *pawns* (*cops* or a *robber*, respectively) on the vertices of G. The focus of the game is on the *protected region* (called also the *cop-region*) $C \subsetneq V$—the cops guard C by preventing the robber from entering the protected region without being "caught", which happens when the robber is on a vertex of C with a cop on it. The game is played in alternating turns. In the first turn the robber-player places her single robber on a vertex of $V \setminus C$, i.e., outside the protected region C. In the second turn the cop-player places her c cops on vertices of C (more than one cop can be placed on a vertex). In each subsequent turn the respective player can *move* each of her pawns to a neighboring vertex of the pawn's position (or leave it at its current position), following the rule that the cops can move only within the protected region C, and the robber can move only on vertices with no cops (thus avoiding being "caught"). At any time of the game both players

know the positions of the cops and the robber in G. The goal of the cop-player is to *guard* (or *protect*) the region C, i.e., to position and move the cops such that, in any turn, the robber cannot move onto a vertex of C with no cop on it. We say that a cop on vertex v *guards* the vertex v (as the robber cannot move onto v). Naturally, the goal of the robber-player is to position and move the robber in such a way that at some point in time the robber can move onto a vertex of C with no cop on it. The region $V \setminus C$ where the robber moves before it eventually enters C is called the *robber-region* and is denoted by R.

The guarding game is a *robber-win* game if the robber-player can, at some turn, move the robber to a vertex in C with no cop on it. In this case we say that the robber-player *wins* the game. Otherwise (if the cop-player can forever prevent the robber-player to win) we say that it is a *cop-win* game, and that the cop-player *wins* the game, or that the cop-player can *guard* C. Thus, the game is fully specified by the number of cops c, and by the *board* of the game which is given by the graph $G = (V, E)$, the protected region C and the robber-region R. We denote the board by $[G;\ R, C]$ (obviously $R = V \setminus C$, but it is convenient for us to keep both sets R and C in our notation). In this work we will be considering cases when sets R or C induce subgraphs of G with specific properties (like being a path, a tree, or a directed acyclic graph—DAG). By slightly abusing notations, we will often be referring to robber or cop regions as induced subgraphs.

Since the game is played in alternating turns starting at the first turn, the robber-player moves her robber in odd turns, and the cop-player moves her cops in even turns. Two consecutive turns $2 \cdot i - 1$ and $2 \cdot i$ are jointly referred to as a *round* i, $i \geq 1$. A *state* of the game at *time* i is given by the positions of all cops and robbers on the board after $i - 1$ turns. A *strategy of a cop-player* (*strategy of a robber-player*) is a function $\mathcal{X}$ which, given the state of the game, determines the movements of the cops (the robber) in the current turn. If there are no cops (no robber) on the board, the function determines the initial positions of the cops (the robber).

The *guarding decision problem* is, given a board $[G;\ R, C]$, and a number c specifying the number of cops, to decide whether it is a cop-win game or a robber-win game. The *guarding problem* is, given a board $[G;\ R, C]$, to compute the minimum number of cops that can guard the protected region C against all possible strategies of the robber-player. We call this number the *guard number* of the game, and denote it by $\ell = gd(G; R, C)$. For illustration, Fig. 1 depicts a board of the guarding game. If the robber is on v_4, two cops must be on u_3 and u_4. If the robber moves to v_3 none of the two cops can reach u_1 in the next turn, hence $\ell > 2$. Later we will see that for this game $\ell = 3$.

Our Contribution In this work, we initiate the study of algorithmic and combinatorial issues of the guarding problem. We start with the case when R is a path. We show that for such robber-regions the problem can be solved in polynomial time. The solution is based on a reduction to the minimum flow problem. Furthermore, an interesting special case of the problem is when C is a path, too. In this case we establish a combinatorial result that the minimum number of cops needed to protect the path is equal to the size of a maximum independent set in a (related) comparability graph. By making use of the combinatorial result, we are able to obtain a faster solution

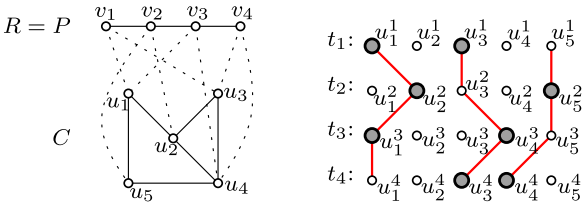

Fig. 1 An instance of the guarding game with the path R and the graph C to be guarded (*left*). The robber walks on the path R from vertex v_1 in round 1 (layer t_1) to vertex v_4 in round 4 (layer t_4). Three cops are necessary to guard C, and a strategy of the cops is depicted by the thick polygonal curves traversing the vertices which consist of 4 "copies" of C (*right*). The highlighted vertices depict vertices on which there needs to be a cop at the respective time (the neighbors of vertex v_i after round i)

for this case. The complexity of the problem increases even with small changes of the robber-region R. We do not give an answer whether the problem can be solved in polynomial time when R is a cycle, however, we are able to find a 2-approximation of the guard number in this case. When R is a tree, or even a star, the problem becomes NP-hard and the parameterized version of the problem is $W[2]$-hard. Moreover, our reduction from MINIMUM SET COVER implies that there is a constant $\rho > 0$ such that there is no $\rho \log n$-approximation algorithm, unless P = NP.[1] We also show that the decision version of the problem is in NP when R is a tree. In case R is a directed acyclic graph (DAG), the problem is PSPACE-complete. We do not solve the general case, whether the problem is in PSPACE when R is an arbitrary graph.

Related Work The guarding game can be seen as a member of a vast class of games called the pursuit-evasion games (see, e.g., [5] for an introduction). The discrete version of pursuit-evasion games, played on graphs, is the Cops-and-Robbers game. The game was defined (for one cop) by Nowakowski and Winkler [2]. Their work and the work of Quilliot [6] characterized graphs for which one cop can catch the robber. Aigner and Fromme [3] initiated the study of the problem with several cops. The minimum number of cops that are required to capture the robber is called the cop number of a graph. This problem was studied intensively and we refer to Alspach's survey [7] for references. The Cops-and-Robbers game can be seen as a special case of search games played on graphs, see the annotated bibliography [8] for further references on search and pursuit-evasion games on graphs. The computational complexity of finding the minimum number of cops in the game of Aigner and Fromme is still not settled completely. Goldstein and Reingold [9] proved that the version of the game on directed graphs is EXPTIME-complete. Also they have shown that the version of the game on undirected graphs when cops and robbers are given their initial positions is also EXPTIME-complete. They also conjectured that the problem is EXPTIME-complete on undirected graphs (without given initial positions of the players). However, until very recently even NP-hardness of the problem was open [10]. To our knowledge, cop-robber guarding games, the topic of this paper, have

[1] If NP$\not\subset$DTIME($n^{\text{poly}\log n}$), Lund and Yannakakis [4] show that MINIMUM SET COVER cannot be approximated within the ratio $\rho \log n$ for any $\rho < 1/4$.

not been studied in mathematical and algorithmic terms before. Let us note that the nature of the guarding game is very different—in the guarding game the cops move such that the robber cannot enter the protected region, while in the original Cops-and-Robbers game, the cops are "hunting" the robber so that they eventually catch the robber. The results or methods obtained for the original Cops-and-Robbers game are not applicable to the cop-robber guarding game of this paper.

2 Guarding Against a Path

In this section we consider the guarding problem with board $[G;\ R, C]$, where the protected region C induces an arbitrary graph, and the robber-region R induces a path P, i.e., the board of the game is a path P connected to C by interconnecting edges, leading to the graph G. Modeling the problem as a flow problem we can design an algorithm that computes in polynomial time the minimum number of cops that are needed to guard C when R induces a path. Moreover, our algorithm also computes a winning strategy of the cop-player.

Let n_1 and n_2 denote the number of vertices of R and C, respectively, and suppose that R is the path $P = v_1, v_2, \ldots, v_{n_1}$. The restriction on the area where a robber can move before entering the protected region C allows us to consider only one strategy of the robber-player, the *direct* strategy: the robber-player places her robber on the first vertex v_1 of the path P and moves the robber from v_1 along P to the last vertex v_{n_1} of the path P, and enters the protected region C when possible. As the following lemma shows, the cops that can guard against this strategy can also guard against any strategy of the robber-player.

Lemma 1 *If c cops can guard C against the direct strategy of the robber-player—in a guarding game where the robber-region R is a path—then c cops can guard C against any strategy of the robber-player.*

Proof Let us consider a winning strategy of the cop-player against the direct strategy using c cops, and let us observe the positions of the c cops. Let X_i be the multiset of vertices of C on which the cops are placed when the robber, following the direct strategy, is on vertex v_i, $i = 1, \ldots, n_1$, of the path P. (Hence, a vertex $u \in C$ appears k times in X_i if there are k cops on vertex u after the round in which the robber moves to vertex v_i.) Thus, $X_1, X_2, \ldots, X_{n_1}$ describe the movements of the cops when the robber moves from v_1 to v_{n_1}. Observe now that the cop-player can move the cops from X_i to X_{i-1} if the robber moves from v_i to v_{i-1}, and the cop-player can move the cops from X_i to X_{i+1} if the robber moves from v_i to v_{i+1}, and that there is no other movement for the robber at vertex v_i. Thus, for any strategy of the robber-player, when the robber moves to vertex v_i, the cop-player can always move the cops to positions X_i. Therefore, at any time, the robber cannot enter C, and the cop-player wins. □

Obviously, the number of cops in an optimal winning strategy of the cop-player against the direct strategy of the robber-player cannot be larger than the number of

cops in an optimal winning strategy of the cop-player against *every* strategy of the robber-player. Thus optimality is preserved if we restrict the robber-player to play a direct strategy. Using this remark we prove the main result of the section.

Theorem 1 *Let* $[G;\ R, C]$ *be a board such that* R *is a path. The guarding problem for board* $[G;\ R, C]$ *is solvable in time in* $O(n_1 n_2 m)$, *where* $m = |E(G)|$, $n_1 = |R|$, *and* $n_2 = |C|$.

Proof Thanks to the previous lemma, we can consider only the direct strategy of the robber-player. We present a (constructive) algorithm that computes the minimum number of cops and also a winning strategy of the cop-player.

Consider a walk of the robber from vertex v_1 to vertex v_{n_1}. Assume that ℓ cops can win the game. We can describe a winning strategy of the cops by the positions of the cops in C: Let X_i, $i = 1, 2, \ldots, n_1$, be a multiset of cardinality ℓ with elements from C, with the meaning that ℓ cops are on vertices X_i when the robber is on vertex v_i. As $\{X_i : i = 1, \ldots, n_1\}$ describes a winning strategy for the ℓ cops (for the robber's direct strategy), the neighbors of v_i in C are in X_i for every $i = 1, \ldots, n_1$ (otherwise there would be a neighbor of v_i in C with no cop on it and thus the robber could move there in the subsequent move and win the game). We can visualize the walk of the robber and the cops in the following way. We consider the vertices of C in situations when the robber is on vertex v_i, $i = 1, 2, \ldots, n_1$. Thus, we consider n_1 layers of "copies" of C, where the i-th layer consists of vertices $u^i_1, u^i_2, \ldots, u^i_{n_2}$, with the obvious meaning that u^i_j corresponds to the vertex $u_j \in C$ when the robber is on vertex v_i. We depict the movements of ℓ cops by ℓ polygonal curves traversing the vertices u^i_j, where each curve depicts the movement of one cop—vertex u^i_j lies on the curve if and only if the corresponding cop visited vertex u_j in the turn when the robber was on vertex v_i. Consider Fig. 1 for an illustration. The example shows the board consisting of C on five vertices, and a path induced by R of length four (left). The dotted lines are the interconnecting lines of R and C. For the walk of the robber from v_1 to v_4 in four rounds $1, 2, 3, 4$ we consider in each round the vertices of C (right)—four layers (copies) t_1, t_2, t_3, t_4 of C. After the moves of each round i are completed, the neighbors of v_i have to be guarded by cops (the highlighted vertices). An optimal solution is depicted by the three polygonal curves between the layers, which show the position of each cop at every time, thus showing the number of necessary cops (three in this case) and their movements.

The movements of the cops (the polygonal curves) can be seen as "graph-theoretic paths" between the layers. Our algorithm computes such "paths". We construct an auxiliary directed graph A. If R induces a path on n_1 vertices, there are n_1 layers (copies) of vertices C, where layer t_i contains vertices u^i_j, $j = 1, 2, \ldots, n_2$. We add a directed edge from vertex u^i_j to vertex u^{i+1}_k if $j = k$ (a cop may stay on the same vertex), or if $u_j u_k$ is an edge in C (a cop moves from vertex u_j to vertex u_k in G). There are two additional vertices in A—source s and target t, where s is connected to every vertex of the first layer, and t is connected to every vertex of the last layer (see Fig. 2). We want that the paths from s to t (which we want to compute) reflect the guarding strategy of the cop-player. Thus, we require that at least one path traverses through every vertex u^i_j for which the corresponding vertex u_j is a neighbor of v_i in

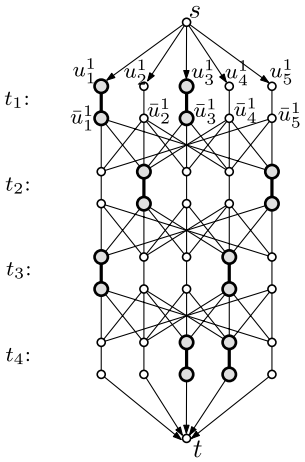

Fig. 2 The auxiliary graph A constructed from the guarding game of Fig. 1. For better readability, not every edge has its direction depicted. The duplicate of a vertex u^i_j is denoted by $\bar{u}^i_j$. The thick edges between the original and duplicated vertices denote the edges with a flow demand $b_e = 1$

graph G. We call such a vertex u^i_j an *endangered* vertex. We say that a path *covers* an endangered vertex v if the path traverses through v. It is now an easy observation that the minimum number of paths, each from s to t, that cover all endangered vertices, is the minimum number of cops for which the game is a cop-win game. Clearly, every path suggests a movement for one cop, and similarly, every cop's movement induces one path. As both the cops' movements and the paths traverse through all endangered vertices, the connection between the two values becomes obvious. Let us call a set of paths that together traverse through every endangered vertex a *path cover*.

To find an optimal guarding strategy for the cop-player, we are left to compute a path cover of minimum cardinality. This problem has been studied previously for the case where the paths should cover all vertices of a graph (i.e., the case where all vertices of G are endangered) [11]. This problem can be solved by different methods. To give an explicit solution here we show that it can be reduced to a minimum flow problem using the same approach as in [11]. To ensure that the paths traverse through endangered vertices, we modify the graph A a bit, and duplicate every vertex u^i_j. All ingoing edges to u^i_j before duplication are now the ingoing edges of the original, and all outgoing edges from u^i_j before duplication are now the outgoing edges of the copy. We connect the copy and the original with an edge. For each such newly created edge e we set the minimum flow demand b_e to be one if the vertex u^i_j is endangered (i.e., we ask the flow to traverse through u^i_j), and we set b_e to be zero otherwise (i.e., there is no need for the flow to go there—no cop is needed there). For all other edges in the graph, we likewise set b_e to be zero. It is now an easy observation that a minimum flow from s to t that satisfies the minimum demand b_e of flow for every edge corresponds to a minimum path cover: since the flow requirements are integers (and there are no cycles in A) the flow is integral (on every edge), too. Thus, every unit of flow represents a path, and clearly the paths form a path cover. As the flow is a minimum flow we further obtain that the path cover is of minimum cardinality. As the minimum flow problem can be computed in time in $O(f^* \cdot |E|)$ (an easy modification of the Ford-Fulkerson algorithm for maximum flow, see e.g. [12] for an explicit description) where f^* is the value of an initial feasible flow for the problem,

we can conclude (as for our graph A the number of edges is at most $O(n_1|E|)$; and we can take as an initial feasible flow the flow of value n_2 which uses n_2 paths where, for every i, path i goes through vertices u_i only) that the guarding problem can be solved in time in $O(n_1 n_2|E|)$. □

As we shall see in the next section, the complexity of the problem changes drastically when R is not a path. For the case where R is a cycle, Theorem 1 gives a possibility to obtain a constant factor approximation for the guard number.

Consider the case where R is a cycle $v_1, v_2, \ldots, v_{n_1}$ and let e be the edge $v_{n_1}v_1$. Using the flow approach which we have used for the case when R is a path, we can compute an optimal strategy of the cop-player against any strategy of the robber-player which walks along the cycle R for d steps in one direction: just create d layers of vertices of C, interconnect the vertices of the neighboring layers accordingly, substitute each vertex by an edge, introduce s, t and the right flow demands on edges, and the minimum flow results in an optimal guarding strategy of the cop-player against the d moves of the robber in one direction.

We now show that if we set $d = n_1$, the resulting optimal strategy against the n_1 moves of the robber can be used to design a 2-approximation of a best cop-strategy against any strategy of the robber-player. The idea is to duplicate the number of cops that are necessary to guard one walk of the robber around the cycle, and to use the two groups of cops in alternating cycle-around-walks of the robber. To be more precise, let us consider the robber-player's strategy as a walk on the cycle R (before it eventually enters C), and let us define a *cycle-walk* to be the maximal time-period in which the walk does not use the edge $v_{n_1}v_1$. We number the cycle-walks in the following way. We assume, without loss of generality, that the robber starts at vertex v_1. The cycle-walk number one is then the part of the robber's walk that starts in the first turn, and ends at the time the robber uses the edge $v_{n_1}v_1$ (in any direction). If the robber moves along the edge from vertex v_{n_1} to vertex v_1, a new cycle-walk starts with number two. If the robber moves along the edge from vertex v_1 to vertex v_{n_1} a new cycle-walk starts with number zero. Recursively we can now define any cycle-walk $i \in \mathbb{Z}$. We are now ready to prove the following corollary of Theorem 1.

Corollary 2 *Let $[G;\ R, C]$ be a board such that R is a cycle $v_1, \ldots, v_{n_1}$. Then there is a 2-approximation polynomial time algorithm for the guarding problem on board $[G;\ R, C]$.*

Proof Let us prove the following claim, which is an implicit proof of the corollary.

Claim 1 *Let $[G;\ R, C]$ be as in the corollary. Let e be the edge $v_{n_1}v_1$ of cycle R. Let G' be a graph obtained from G by the removal of e, and let $\ell = gd(G';\ R, C)$ be the guard number of the game played on $[G';\ R, C]$. Then $\ell \leq gd(G;\ R, C) \leq 2\ell$. Further, 2ℓ can be computed in time polynomial in the number of vertices of G.*

Guarding against the strategy where the robber walks within one cycle-walk can be seen as guarding in the scenario where R is a path, i.e., in the game played on $[G';\ R, C]$, and thus an optimal guarding strategy $\mathcal{X}$ for this case can be computed in

polynomial time (Theorem 1). Let the multisets X_i of elements of C, $i = 1, \ldots, n_1$, denote the positions of the cops in the optimal guarding strategy $\mathcal{X}$ of the game played on $[G'; R, C]$, when the robber is placed on vertex v_i. Double the amount ℓ of cops that are used in $\mathcal{X}$ and form two groups of cops—the first ℓ cops are in group A_1 and guard the cycle-walks with odd numbers, the next ℓ cops are in group A_2 and guard the cycle-walks with even numbers. In the beginning, when the robber sits on vertex v_1 in the first cycle-walk (cycle-walk number one), the cops from group A_1 sit on vertices in X_1 (thus guarding the protected region C), ready to move to vertices in X_2 if the robber moves to vertex v_2, and the cops from group A_2 sit on vertices in X_{n_1}, ready to guard the protected region C if the robber moves to vertex v_{n_1} and starts a new cycle-walk. In general, in an odd cycle-walk, if the robber moves from vertex v_i to v_{i+1}, $1 \leq i < n_1$, the cops from group A_1 move from X_i to X_{i+1} and the cops from group A_2 move from X_{n_1-i+1} to X_{n_1-i}. The cops move in the reverse direction if the robber moves from vertex v_{i+1} to vertex v_i, $1 \leq i < n_1$. Thus, the cops of A_1 guard C in the odd cycle-walks. The cops do not move if the robber moves using the edge $v_{n_1}v_1$, but switch their roles, i.e., for any move of the robber from v_i to v_{i+1}, the cops of A_2 follow the strategy from X_i to X_{i+1}, and the cops of A_1 are getting ready for next cycle-walk with a reverse walk from X_{n_1-i+1} to X_{n_1-i}, and so on. Thus, using this strategy with 2ℓ cops, at any time when the robber is on vertex v_i there are cops on positions in X_i and hence this strategy of a cop-player guarantees that the game is a cop-win game. Clearly, the minimum number of cops that can guard in a game played on $[G; R, C]$ is at least ℓ, and obviously, 2ℓ can be computed in polynomial time. □

Now we consider an interesting special case when both the robber-region and the protected regions are paths. We establish a combinatorial result stating that in this case the guard number is equal to the size of a maximum independent set in a related comparability graph.

The graph G of the game is a *pseudo-bipartite* graph $G = (V_1 \cup V_2, E)$, $|V_1| = n_1$ and $|V_2| = n_2$, where vertices V_1 induce a path $P_1 = v_1, v_2, \ldots, v_{n_1}$ in G, and vertices V_2 induce a path $P_2 = u_1, u_2, \ldots, u_{n_2}$ in G. We note that the paths P_1 and P_2 induce a natural linear ordering of the vertices of V_1 and V_2, respectively. The vertices of P_1 and P_2 form the robber-region R and the protected region C, respectively, i.e., the board is $[G; V_1, V_2]$.

The auxiliary graph $B = B(G)$ that reflects the structure of the game played on a pseudo-bipartite graph G is defined as follows. Every edge $e = vu$ between P_1 and P_2, $v \in V_1$, $u \in V_2$, corresponds to a vertex $B(e) = B(vu)$ of B. The vertices $B(e)$ and $B(e')$ of B are connected in B, if in G one cop can guard the endpoints of the edges e and e' in P_2 against the direct strategy. More formally, let $e = v_i u_j$ and $e' = v_{i'} u_{j'}$ be two edges interconnecting P_1 and P_2. Then the vertices $B(e)$ and $B(e')$ form an edge in B if and only if $|i - i'| \geq |j - j'|$. Intuitively, this corresponds to the situation where a single cop that guards vertex u_j when a robber is on vertex v_i can also guard vertex $u_{j'}$ when the robber moves to $v_{i'}$, because the cop can *get there on time*. Let us order the vertices of B (which correspond to edges in G) as follows: a vertex $B(e) = B(v_i u_j)$ is said to be smaller than the vertex $B(e') = B(v_{i'} u_{j'})$, if and only if ij is lexicographically smaller than $i'j'$, i.e., the vertices $B(e)$ and $B(e')$ are

sorted in the order as the edges e and e' appear on the path P_1, and in case of a tie, the position of the edge on the path P_2 decides the order. We call B the *pair-distance* graph of G. We will show that B belongs to the class of comparability graphs. The properties of comparability graphs will be further used to design an algorithm for solving the guarding problem (for this special case). Recall that a comparability graph is an undirected graph corresponding to a partial order of the vertices—there is an edge between two vertices if and only if the two vertices are comparable (in the partial order). We note that comparability graphs are a subclass of perfect graphs [13].

Theorem 3 *Let $[G; R, C]$ be a board such that $R = P_1$ and $C = P_2$ are paths, and let $B(G)$ be the auxiliary graph of $G[G; R, C]$. Then a) the guard number ℓ is equal to the size of a maximum independent set in $B(G)$, and b) graph $B(G)$ is a comparability graph.*

Proof We first prove that B is a comparability graph. Let us show that B corresponds to a partial order of the vertices of B (which correspond to edges in G). We note that the lexicographical order of the vertices of B is not the partial order we are looking for. For two vertices $B(e) = B(v_i u_j)$ and $B(e') = B(v_{i'} u_{j'})$ in B we define $B(e) \prec B(e')$ if (1) ij is lexicographically smaller than $i'j'$, and (2) $|j - j'| \leq |i - i'|$, i.e., if a single cop can guard the vertices u_j and $u_{j'}$. Observe that (2) indeed reflects the construction of the graph B from G. Thus, we are left to prove that the relation $\prec$ is transitive (and therefore, a partial order). Suppose therefore that for some three lexicographically sorted edges $v_{i_1} u_{j_1}$, $v_{i_2} u_{j_2}$, $v_{i_3} u_{j_3}$ of $E(G)$ we have $B(v_{i_1} u_{j_1}) \prec B(v_{i_2} u_{j_2})$ and $B(v_{i_2} u_{j_2}) \prec B(v_{i_3} u_{j_3})$. It follows that $|j_2 - j_1| \leq |i_2 - i_1| = i_2 - i_1$ and $|j_3 - j_2| \leq |i_3 - i_2| = i_3 - i_2$. Thus we have that $|j_3 - j_1| = |j_3 - j_2 + j_2 - j_1| \leq |j_3 - j_2| + |j_2 - j_1| \leq (i_3 - i_2) + (i_2 - i_1) = i_3 - i_1 = |i_3 - i_1|$. Consequently, $B(u_{i_1} v_{j_1}) \prec B(u_{i_3} v_{j_3})$, and thus $\prec$ is transitive.

We now prove the first claim. Let m_B and c_B be the size of a maximum independent set of the pair-distance graph $B = B(G)$ and the size of a smallest clique cover of B (i.e., the least number of cliques required to cover B), respectively. We show that $m_B \leq \ell \leq c_B$ holds.

To see $m_B \leq \ell$, let a maximum independent set M of B be given. Let $v_{i_1}, \ldots, v_{i_{m_B}}$ be the (not-necessarily distinct) vertices of P_1 that are the endpoints of the edges that correspond to the vertices in M, where $i_1 \leq i_2 \leq \cdots \leq i_{m_B}$. Consider the following strategy of the robber-player. The robber-player places initially the robber on v_{i_1} and a set of cops must be placed on the neighbors of v_{i_1} in P_2. In every round, the robber-player moves the robber towards $v_{i_{m_B}}$. By construction of B, whenever the robber is located on vertex v_{i_j}, $1 \leq j \leq m_B$, the cops that are preventing the robber to enter P_2 must be cops that were not previously used to prevent the robber to enter P_2 from the vertices $v_{i_1}, \ldots, v_{i_{j-1}}$. This establishes m_B as a lower bound of ℓ.

To show that $\ell \leq c_B$, we describe a strategy for the cop-player using c_B cops that prevents the robber to enter P_2. As argued before (Lemma 1), it suffices to show a winning strategy for the cop-player against a direct strategy of the robber-player. Given a clique cover of B, this is an easy task. Every clique in the cover specifies as before a set of vertices $v_{i_1}, \ldots, v_{i_{c_B}}$, $i_1 \leq i_2 \leq \cdots \leq i_{c_B}$, in P_1 and $u_{j_1}, \ldots, u_{j_{c_B}}$ in P_2, so that $B(v_{i_k} u_{i_k})$ is a vertex of the clique in B (and corresponds to an edge of

the original graph). The cop-player's strategy is to use one cop for every clique of the clique cover. When the robber is on vertex v_{i_k} during its trip to v_{n_1}, the cop is located on vertex u_{i_k}. By construction the cop reaches $u_{i_{k+1}}$ no later than in the same round as the robber reaches $v_{i_{k+1}}$. Since the clique cover spans the whole board, the robber cannot win and the inequality $\ell \leq c_B$ is proved.

In general graphs, the size of a maximum independent set is a lower bound for the size of a minimum clique cover (every vertex of the independent set needs to be covered by a distinct clique). In perfect graphs, however, the two parameters are equal [14], which proves the claim. □

Note that there are known polynomial time algorithms for computing a maximum independent set in comparability graphs [15]. However we now describe a tailor-made problem-specific dynamic programming algorithm for computing the minimum number of cops needed to guard P_2, which runs in time $O(m'^2)$, where m' is the number of interconnecting edges of the original graph G, i.e., the number of edges between V_1 and V_2. The algorithm, which we simply call Alg, efficiently computes a maximum independent set of a pair-distance graph B.

The dynamic-programming approach works in the following way. First, Alg considers the vertices of B in the following order (different from the order we considered in the previous sections): $B(e_1) = B(v_{i_1}u_{j_1})$ is said to be *smaller* than $B(e_2) = B(v_{i_2}u_{j_2})$ if j_1i_1 is lexicographically smaller than j_2i_2. For every vertex $B(e_i)$ of B the algorithm stores a biggest independent set among the vertices $B(e_1), \ldots, B(e_i)$ that contains the vertex $B(e_i)$. Let $T(e_i)$ denote such an independent set, and $|T(e_i)|$ its size. At the beginning, the algorithm sets $T(e_0) = \emptyset$ and $T(e_1) = \{B(e_1)\}$. In the generic step of computing $T(e_i)$, Alg tries to add $B(e_i)$ to every independent set $T(e_j)$, $0 \leq j < i$, resulting into candidate sets $I_j := \{B(e_i)\} \cup T(e_j)$. From I_j, the algorithm considers only independent sets $Y_i := \{I_j \mid 1 \leq j < i,\ I_j \text{ is an independent set}\}$, and defines $T(e_i)$ to be a maximum-cardinality set in Y_i. After the algorithm completes the computation of $T(e_{m'})$, a set $T(e_i)$, $1 \leq i \leq m'$, of the maximum cardinality is declared to be a maximum independent set of B.

Theorem 4 *The algorithm Alg computes a maximum independent set of a given auxiliary graph B of the guarding game played on the board* $[G;\ R, C]$ *(where R and C are paths) in time* $O(m'^2)$*, where* m' *is the number of edges in G between R and C (i.e.,* $m' = |E(G)| - (n_1 - 1) - (n_2 - 1)$*).*

Proof It is obvious that Alg computes an independent set of B, as every set $T(e_i)$, $i = 1, \ldots, m'$, is an independent set. It is also obvious that the algorithm runs in time in $O(m'^2)$. We now show that $T(e_i)$ indeed is a maximum independent set among the vertices $B(e_1), \ldots, B(e_i)$ that contains the vertex $B(e_i)$. Then, as the algorithm outputs as the result a maximum-cardinality set $T(e_{i^*})$ among sets $T(e_i)$, $i = 1, \ldots, m'$ (i.e., $i^* = \arg\max_i |T(e_i)|$), the algorithm clearly outputs a maximum independent set.

We are left to prove the claim. We prove the claim using mathematical induction. Clearly, for $i = 1$, the set $T(e_1) = \{B(e_1)\}$ is a maximum independent set

of elements $B(e_1), \ldots, B(e_1)$ (i.e., of one element only) containing vertex $B(e_1)$. Let $i > 1$, and let us assume the claim holds for every $j < i$. We show that the claim also holds for i. Let O denote a maximum independent set among vertices $B(e_1), \ldots, B(e_i)$ that contains $B(e_i)$. We show that $|T(e_i)| \geq |O|$ which proves the claim. Let $O' = O \setminus \{B(e_i)\}$. Let $B(e_l)$ be the last vertex in O' (in the order the algorithm considers the vertices). We have $l < i$. Clearly, O' is an independent set among vertices $B(e_1), \ldots, B(e_l)$ that contains $B(e_l)$. Using the induction hypothesis, we get $|T(e_l)| \geq |O'|$. We now show that $T(e_l) \cup \{e_i\}$ is an independent set (containing e_i). Using this we then obtain the desired $|T(e_i)| \geq |T(e_l) \cup \{e_i\}| \geq |O' \cup \{e_i\}| = |O|$. To show that $T(e_l) \cup \{e_i\}$ is an independent set we show that $B(e_i)$ is not adjacent to any vertex $B(e_j)$ from $T(e_l)$. We denote $e_i = v_{a_i}u_{b_i}$, $e_l = v_{a_l}u_{b_l}$ and $e_j = v_{a_j}u_{b_j}$. Observe that $b_i \geq b_l \geq b_j$. We use the fact that $B(e_i)$ is not adjacent with $B(e_l)$, and $B(e_l)$ is not adjacent with any vertex $B(e_j)$ of $T(e_l)$. Thus, according to the definition of B, $|b_i - b_l| > |a_i - a_l|$, and $|b_l - b_j| > |a_l - a_j|$. Hence $|b_i - b_j| = |b_i - b_l + b_l - b_j| = |b_i - b_l| + |b_l - b_j| > |a_i - a_l| + |a\_a_j| \geq |a_i - a_l + a_l - a_j| = |a_i - a_j|$. Consequently, there is no edge between $B(e_i)$ and $B(e_j)$ in B.

□

3 Hardness of Guarding

It is not difficult to guess that the problem of computing the guard number is a hard problem. In this section we provide two hardness results for quite restricted classes of boards.

We first show that the problem is NP-hard even when the robber-region is a *star*. A star is a tree where one vertex is adjacent to all other vertices, and the other vertices have all degree one. We present a reduction from SET COVER [16] (which is an NP-complete problem) to the guarding decision problem, which then shows that the guarding decision problem is NP-hard. SET COVER is the problem of covering a universe $U = \{u_1, \ldots, u_n\}$ by at most k sets from a given set system $\mathcal{S} = \{S_1, \ldots, S_m\}$, $S_i \subseteq U$, $i = 1, \ldots, m$, where $U \subseteq \bigcup_i S_i$.

Theorem 5 *The guarding decision problem is* NP*-hard* (*for both directed and undirected graphs*) *even if the robber-region R is a star* (*or even a directed rooted star in the case of directed graphs*). *Moreover, the parameterized version of the problem with c* (*the number of cops*) *being a parameter is* W[2]*-hard.*

Proof We prove the theorem for undirected graphs. The proof for directed graphs is a simple adaptation of what follows. We describe how to reduce SET COVER to the guarding decision problem. Consider an instance of SET COVER. We first define U as vertices which shall be effectively guarded by the cops. Thus, U will be part of the protected region C. We add to our construction one single vertex a connected to all vertices of U by paths of length two. Denote by $b_1, b_2, \ldots, b_n$ the middle vertices of these paths. The robber-region R is then $\{a, b_1, b_2, \ldots, b_n\}$—a star. We finally add vertices where the cops can be placed and from which the cops can "cover" U in the

same way as a solution to the SET COVER: we add one vertex s_i for every set S_i, and connect it to every vertex $u \in S_i \subseteq U$ by an edge. These vertices define, together with U, the protected region C. The guarding game is defined by setting the number of cops $c = k$. Suppose that c cops can win the game. Initially, if the robber is placed on a, we can assume, w.l.o.g., that cops occupy vertices from the set $\{s_1, s_2, \ldots, s_m\}$. Obviously these vertices correspond to the set cover of size no more than $c = k$. Assume now that $X \subseteq \{S_1, S_2, \ldots, S_m\}$ is a set cover of size at most k. Then for every $u_i \in U$ there is $S_{j_i} \in X$ such that $u_i \in S_{j_i}$. Therefore, in our graph construction, u_i and s_{j_i} are adjacent. Let $x \subseteq \{s_1, s_2, \ldots, s_m\}$ be the set of vertices that correspond to the set cover X. Let $x_i = (x \setminus \{s_{j_i}\}) \cup \{u_i\}$ for every $i \in \{1, 2, \ldots, n\}$. We consider the cop-player's strategy where the cops occupy vertices of x if the robber is in a, and the cops occupy vertices of x_i if the robber is in b_i. Clearly, this is a winning strategy for the cop-player.

Since our reduction is a parameterized reduction and SET COVER is a well-known W[2]-hard problem, the second claim follows immediately.[2]

The only difference for the case of directed graphs is that a is joined with $u_1, u_2, \ldots, u_n$ by directed paths, and vertices $s_1, s_2, \ldots, s_m$ are joined with $u_1, u_2, \ldots, u_n$ by arcs with the obvious orientation.

□

Let us note that the second claim of Theorem 5 is "tight" in some sense. It is well-known for Cops-and-Robbers games that determining whether c cops can capture the robber on a graph with n vertices can be done by a backtracking algorithm which runs in time $O(n^{O(c)})$ (thus polynomial for fixed c) [9, 18, 19]. A similar result holds for the guarding game. Given an integer c and a graph G on n vertices, it can be determined whether c cops can guard a given set C (and the corresponding winning strategy of c cops can be computed) by constructing the game-graph on $2\binom{|C|+c-1}{c}|R|$ vertices (every vertex of the game-graph corresponds to a possible position in G of c cops and one robber before the robber enters R, taking into account two possibilities for whose turn it is, and allowing more than one cop to be on one vertex), and then by making use of backtracking to find if some robber-winning position can be obtained from an initial position. The proof of the following proposition is standard and easy (and we omit it here).

Proposition 1 *For a given integer $c \geq 1$ and a graph G on n vertices, the question whether the c cops can guard C can be answered in time $\binom{|C|+c-1}{c}^2 \cdot |R|^2 \cdot n^{O(1)} = n^{O(c)}$.*

Thus for every fixed c, one can decide in polynomial time if c cops can guard the protected region against the robber on a given graph G. But on the other side the fact that our problem is $W[2]$-hard means that the existence of a $O(f(c) \cdot n^{O(1)})$-time algorithm deciding if c cops can win, where f is a function only of the parameter c, and G is a graph on n vertices, would imply that $FPT = \mathrm{W}[2]$, which is considered to be very unlikely in parameterized complexity.

[2] We refer to the book of Downey and Fellows [17] for an introduction to parameterized complexity.

It can be easily proved that the guarding problem is difficult to approximate. We combine the (approximation preserving) reduction from the proof of Theorem 5 and the non-approximability for MINIMUM SET COVER problem [20] and obtain the following statement.

Corollary 6 *There is a constant $\rho > 0$ such that there is no polynomial time algorithm that, for every instance, approximates the minimum number of cops which are necessary to guard the protected region within a multiplicative factor $\rho \log n$, unless* $\mathrm{P} = \mathrm{NP}$.

Theorem 5 claims only that the guarding problem is NP-hard, i.e., not NP-complete, but for some cases we can prove that the problem is in NP.

Proposition 2 *If R is a tree or a directed tree, then the guarding decision problem is* NP*-complete.*

Proof The NP-hardness was proved in Theorem 5. We only have to prove the inclusion of our problem in NP. We give the proof for undirected graphs (for directed graphs the proof is almost identical). Let R be a tree and denote it as T. Suppose that there is a winning strategy of the cop-player using c cops. Observe, as R is a tree, that there is also a winning strategy of the cop-player using c cops of a special kind: at any time the robber is on a position $v \in V(T)$, the cops are always on the same positions X_v—a multiset of C. Obviously, if a family of cops' positions $\{X_v \mid v \in V(T)\}$ is given then it is possible to test in polynomial time if this family corresponds to a winning strategy—all neighbors of v in C have to be in X_v (thus the cops guard C), and if a robber can get from vertex v to vertex w in one turn, then the cops have to be able to move from X_v to X_w in one turn, too. □

For directed graphs the problem becomes more difficult, if we allow only a slightly more general robber-region.

Theorem 7 *The guarding decision problem is* PSPACE*-hard for directed graphs even if R is a directed acyclic graph (DAG).*

Proof We reduce the PSPACE-complete QUANTIFIED BOOLEAN FORMULA IN CONJUNCTIVE NORMAL FORM problem [16] to the guarding decision problem. For given Boolean variables $x_1, x_2, \ldots, x_n$ and a Boolean formula $F = C_1 \wedge C_2 \wedge \cdots \wedge C_m$, where C_j is a clause, this problem asks whether the expression $\phi = Q_1 x_1 Q_2 x_2 \cdots Q_n x_n F$, where either $Q_i = \forall$ or $Q_i = \exists$, has value *true*.

Given a quantified Boolean formula ϕ, we construct an instance of a guarding game in the following way. For every quantification $Q_i x_i$ we introduce a gadget graph G_i. If $Q_i = \forall$ then we define graph $G_i(\forall)$ as the graph with the vertex set $\{u_i, v_i, x_i, \overline{x}_i, y_i, \overline{y}_i, z_i\}$ and the arc set $\{u_i y_i, y_i v_i, u_i \overline{y}_i, \overline{y}_i v_i, y_i x_i, \overline{y}_i \overline{x}_i, z_i x_i, z_i \overline{x}_i\}$. Let $W_i = \{x_i, \overline{x}_i, z_i\}$. If $Q_i = \exists$ then $G_i(\exists)$ is the graph with the vertex set $\{u_i, v_i, x_i, \overline{x}_i, y_i, a_i, b_i, z_i\}$ and the arc set $\{u_i y_i, y_i v_i, y_i a_i, a_i b_i, x_i b_i, \overline{x}_i b_i, z_i x_i, z_i \overline{x}_i\}$. In this case let $W_i = \{x_i, \overline{x_i}, b_i, z_i\}$. These graphs are joined together by gluing vertices v_{i-1} and u_i for $i \in \{2, 3, \ldots, n\}$.

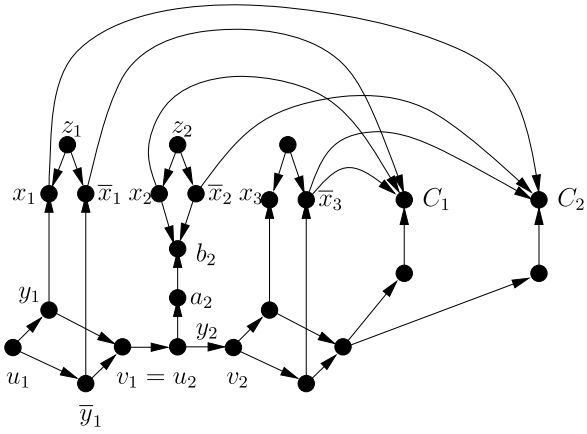

Fig. 3 Construction of G for $\phi = \forall x_1 \exists x_2 \forall x_3 (\overline{x}_1 \vee x_2 \vee \overline{x}_3) \wedge (x_1 \vee \overline{x}_2 \vee \overline{x}_3)$

In our construction we further introduce vertices $C_1, C_2, \ldots, C_m$, which correspond to clauses. A vertex x_i is joined with C_j by an arc (pointing towards C_j) if C_j contains the literal x_i, and $\overline{x}_i$ is joined with C_j if C_j contains the literal $\overline{x}_i$. The vertex v_n is connected with all vertices $C_1, C_2, \ldots, C_m$ by directed paths of length two with middle vertices $w_1, w_2, \ldots, w_m$. Denote the obtained directed graph by G, and let $C = W_1 \cup W_2 \cup \cdots \cup W_n \cup \{C_1, C_2, \ldots, C_m\}$ be the cop-region, and $R = V \setminus C$ be the robber-region. Clearly, R is a DAG. For the guarding decision problem we set $c = n$. Construction of G for $\phi = \forall x_1 \exists x_2 \forall x_3 (\overline{x}_1 \vee x_2 \vee \overline{x}_3) \wedge (x_1 \vee \overline{x}_2 \vee \overline{x}_3)$ is shown in Fig. 3.

Suppose that $\phi = \mathit{false}$. We describe a winning strategy for the robber. He chooses vertex u_1 as a starting point and then moves to vertex v_n along some directed path. If the robber comes to the vertex y_i of a graph $G_i(\forall)$ then one cop has to occupy the vertex x_i, and if the robber comes to $\overline{y}_i$ then some cop has to occupy the vertex $\overline{x}_i$. So, cops are "forced" to occupy vertices that correspond to literals. Similarly, if the robber occupies the vertex y_i in a graph $G_i(\exists)$ then at least one cop has to occupy one vertex from the set $\{x_i, \overline{x}_i, b_i\}$, and here this cop can choose between vertices x_i and $\overline{x}_i$. Since the number of cops is equal to the number of graphs $G_i(\forall)$ and $G_i(\exists)$, exactly one cop can stand on vertices of each such a gadget-graph. It follows from the condition $\phi = \mathit{false}$ that there is a directed path from u_1 to v_n such that if the robber moves along it to v_n then there is at least one vertex C_j which is not "covered" by cops, i.e. there are no cops on this vertex and on vertices x_i or $\overline{x}_i$ which are joined by arcs with it. Then the robber can move to this vertex and win.

Assume now that $\phi = \mathit{true}$ and describe a winning strategy for cops. It can be easily seen that if the robber chooses some vertex a_i or w_j as an initial position then he loses immediately. So we can assume that he occupies some vertex r on the path from u_1 to v_n. Suppose that this vertex belongs to the graph G_s ($G_s(\forall)$ or $G_s(\exists)$, it is assumed that $v_{s-1} = u_s$ belongs to G_s for $s > 1$, and v_n belongs to a virtual G_{n+1}). We place one cop on a vertex x_i or $\overline{x}_i$ for every $i < s$. The vertex is chosen arbitrarily if we consider graph $G_i(\forall)$, and if x_i is chosen then we suppose that the variable $x_i = \mathit{true}$ and $x_i = \mathit{false}$ otherwise. If $G_i(\exists)$ is considered then the choice corresponds to the value of the variable x_i: if $x_i = \mathit{true}$ then the vertex x_i is occupied

by a cop, and $\overline{x}_i = \mathit{false}$ otherwise. If $r = y_s$ in $G_s(\forall)$ then one cop is placed on the vertex x_s, and if $r = \overline{y}_s$ in $G_s(\forall)$ then one cop is placed on the vertex $\overline{x}_s$. As before, $x_i = \mathit{true}$ in the first case and $x_i = \mathit{false}$ in the second. If $r = y_s$ in $G_s(\exists)$ then a cop is placed either on x_s or $\overline{x}_s$ and the choice of the vertex corresponds to the value of the variable x_s. If $r = u_s$ then we place one cop on z_s. For all $i > s$ one cop is placed on each vertex z_i. Now the robber starts to move. If he moves to the vertex y_i of $G_i(\forall)$ then the cop moves from z_i to x_i and it is assumed that the variable $x_i = \mathit{true}$, and if the robber moves to $\overline{y}_i$ then the cop moves to $\overline{x}_i$ and $x_i = \mathit{false}$. If the robber comes to y_i of $G_i(\exists)$ then the cop moves from z_i to x_i or $\overline{x}_i$, and the choice corresponds to the value of the variable x_i. Note that if the robber moves from y_i to a_i then the cop moves to b_i and cops win. So the robber is "forced" to move along some directed path from r to v_n. Since $\phi = \mathit{true}$, cops on graphs $G_i(\exists)$ can move in such a way that when the robber occupies the vertex v_n all vertices C_j are "covered" by cops, i.e. for every vertex C_j there is an adjacent vertex x_i or $\overline{x}_i$ which is occupied by a cop. Now the robber can move only to some vertex w_j. Cops respond by moving one cop to C_j and win. □

Note that Theorem 7 (as Theorem 5) claims only "hardness", but in some cases we can prove that the problem is PSPACE-complete. The proof of PSPACE-completeness is based on the fact that when the robber region is a DAG, the number of steps in the game is polynomial.

Theorem 8 *If R is a DAG then the guarding decision problem is* PSPACE*-complete for directed graphs.*

Proof PSPACE-hardness is proved above, so we have to prove that our problem can be solved by an algorithm which uses polynomial space. We need the following observation the proof of which is easy.

Observation 1 *Let X be the positions (a multiset) of the k cops in C. Then it is possible to generate all possible positions of k cops that can be obtained from X in one step of the cops (note that the amount of such positions can be exponential) in polynomial space.*

We assume that at every step of the game the robber moves to an adjacent vertex (if the robber does not move then the cops can either also stay or improve their positions).

We describe now a recursive procedure $W(r, X, l)$ which returns *true* if k cops can win starting from a position X against the robber which starts from a vertex $r \in R$ with the additional restriction that the robber can make at most l moves, and the procedure returns *false* otherwise.

If $l = 1$ then the procedure returns *false* if r is joined by an arc with some vertex $u \in C$ which is not occupied by a cop, and it returns *true* otherwise. Suppose that $l > 1$. If there is a vertex $u \in C$ to which the robber can move and which is not occupied by a cop then our procedure returns *false*. Otherwise we consider all vertices $u \in R$ to which the robber can move before entering C. For every such vertex all

positions of the cops X' such that the cops can move from X to X' are listed, and then we call $W(u, X', l-1)$. The procedure W returns *true* for r, X and l if and only if for every vertex u there is a position X' such that $W(u, X', l-1) = true$.

Since the depth of the recursion is at most l, by Observation 1, the procedure $W(r, X, l)$ uses $O(l \cdot n^{O(1)})$ space for a graph on n vertices.

Since R is a DAG, we have that the robber can make at most $|R| < n$ moves, and we can use our procedure to solve the guarding problem in polynomial space. We check all possible initial positions of the robber and for every such position we test all possible positions (Observation 1 is also used here) of cops and run our procedure for $l = |R|$. Obviously k cops can win if and only if for every initial position of the robber there is an initial position for the cops such that they can win starting from this position. □

4 Conclusions and Future Research

We have shown that when the robber territory R is a DAG, the decision version of the problem is in PSPACE. We do not know if this is the case when R is a more complicated graph. An interesting related question, which remains open, concerns the number of the rounds in the game. Is it true that for a given number k and a (di)graph G, we can always decide if a subgraph of G can be guarded by k cops by playing a game with only a polynomial number of rounds? If the answer to this question is "yes", then similarly to the proof of Theorem 8 one can show that the problem is in PSPACE. Another consequence of this would be that the case when R is a cycle can be solved in polynomial time by making use of an approach similar to the path case. If the answer is "no", what is the complexity of the problem? In this paper we have not answered the question if on (undirected) graphs the guarding problem is PSPACE-hard. Since the first public appearance of the results presented in this paper, these questions have been answered: The guarding problem for a guarding game where R is a cycle can be solved in polynomial time, as was shown by Nagamochi [21]; Reddy et al. have shown that the guarding problem is PSPACE-hard for undirected graphs [22].

To conclude, we feel that with this paper we barely scratched the tip of the iceberg of the universe of guarding games, but should provide insights and incentives to motivate further research in the area.

References

1. Isaacs, R.: Differential Games. A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization. Wiley, New York (1965)
2. Nowakowski, R., Winkler, P.: Vertex-to-vertex pursuit in a graph. Discrete Math. **43**(2–3), 235–239 (1983)
3. Aigner, M., Fromme, M.: A game of cops and robbers. Discrete Appl. Math. **8**(1), 1–11 (1984)
4. Lund, C., Yannakakis, M.: On the hardness of approximating minimization problems. J. ACM **41**(5), 960–981 (1994)
5. Nahin, P.J.: Chases and Escapes. The Mathematics of Pursuit and Evasion. Princeton University Press, Princeton (2007)

6. Quilliot, A.: Some results about pursuit games on metric spaces obtained through graph theory techniques. Eur. J. Comb. **7**(1), 55–66 (1986)
7. Alspach, B.: Searching and sweeping graphs: a brief survey. Matematiche (Catania) **59**(1–2), 5–37 (2006)
8. Fomin, F.V., Thilikos, D.M.: An annotated bibliography on guaranteed graph searching. Theor. Comput. Sci. **399**, 236–245 (2008)
9. Goldstein, A.S., Reingold, E.M.: The complexity of pursuit on a graph. Theor. Comput. Sci. **143**(1), 93–112 (1995)
10. Fomin, F.V., Golovach, P., Kratochvíl, J.: On tractability of Cops and Robbers game. In: 5th IFIP International Conference on Theoretical Computer Science, vol. 273, pp. 171–185. Springer, Berlin (2008)
11. Ntafos, S.C., Hakimi, S.L.: On path cover problems in digraphs and applications to program testing. IEEE Trans. Softw. Eng. **5**(5), 520–529 (1979)
12. Even, S.: Algorithmic Combinatorics. MacMillan, New York (1973)
13. Brandstädt, A., Le, V.B., Spinrad, J.P.: Graph Classes: A Survey. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia (1999)
14. Lovász, L.: Normal hypergraphs and the perfect graph conjecture. J. Discrete Math. **2**, 253–267 (1972)
15. Čenek, E., Stewart, L.: Maximum independent set and maximum clique algorithms for overlap graphs. Discrete Appl. Math. **131**(1), 77–91 (2003)
16. Garey, M.R., Johnson, D.S.: Computers and Intractability. A Guide to the Theory of NP-completeness. A Series of Books in the Mathematical Sciences. Freeman, San Francisco (1979)
17. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Monographs in Computer Science. Springer, New York (1999)
18. Berarducci, A., Intrigila, B.: On the cop number of a graph. Adv. Appl. Math. **14**(4), 389–403 (1993)
19. Hahn, G., MacGillivray, G.: A note on k-cop, l-robber games on graphs. Discrete Math. **306**(19–20), 2492–2497 (2006)
20. Raz, R., Safra, S.: A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In: Proceedings of the 29th STOC, pp. 475–484 (1997)
21. Nagamochi, H.: Cop-robber guarding game with cycle robber region. In: Proceedings of the Third International Frontiers of Algorithmics Workshop (FAW), pp. 74–84 (2009)
22. Reddy, T.V.T., Krishna, D.S., Rangan, C.P.: The guarding problem—complexity and approximation. In: Proceedings of the 20th International Workshop on Combinatorial Algorithms (IWOCA), pp. 460–470 (2009)