

Some pursuit–evasion problems on grids

Robin W. Dawes

Department of Computing and Information Science, Queen's University, Kingston, Ontario, Canada K7L 3N6

Communicated by S.G. Akl

Received 29 July 1991

Revised 17 June 1992

Abstract

Dawes, R.W., Some pursuit–evasion problems on grids, Information Processing Letters 43 (1992) 241–247.

In a rectangular grid, a pursuit–evasion “game” is played according to simple rules. One player (the cop) wins if both players ever occupy either the same row or the same column of the grid. The other player (the robber) wins if the cop loses. Movement of the players is restricted to the rows and columns of the grid, and the speeds at which the players move are limited. We present a new upper bound for the minimum cop’s speed required to guarantee a win for the cop, and consider some variations on the problem.

Keywords: Algorithms; pursuit–evasion; robotics

Introduction

Pursuit–evasion processes, as in [1–4], are often described as two-player games. Given a graph G , a set of “movement rules” S , an integer k , and two sets of termination conditions T_1 and T_2 , the game commences with Player 1 (the cop) placing no more than k tokens on the vertices of G . Player 2 (the robber) then places one token on a vertex of G . The players then move their tokens according to the rules in S until one of the termination conditions is satisfied. Throughout the game, both players have complete knowledge of the location of all tokens. If the final configuration of tokens satisfies a condition in T_1 , then Player 1 wins, otherwise Player 2 wins.

Typically, T_1 contains only the condition “Player 2’s token occupies the same vertex as one of Player 1’s tokens”, and T_2 contains only the

condition “Player 2 can prevent Player 1 from winning”. However, other conditions such as “A specified number of moves have been completed”, “Vertex x is occupied by Player 2’s token”, etc. may be included to give variants of the game. S typically contains only the rules

R_1 : Players take alternate turns.

R_2 : On her turn, Player 1 may either decline to move, or move one of her tokens to a vertex adjacent to its current location.

R_3 : On her turn, Player 2 may decline to move, or move her token to an *unoccupied* vertex adjacent to its current location.

Since the game is entirely deterministic, the outcome (assuming correct play) depends only on the parameters of the game. If there exists an initial configuration of Player 1’s tokens such that for every initial position of Player 2’s token, there exists a strategy for Player 1 that guarantees a win for Player 1, the game is referred to as a *cop-win game*. The classic problem is to characterise cop-win games.

Correspondence to: Professor R.W. Dawes, Department of Computing and Information Science, Queen’s University, Kingston, Ontario, Canada K7L 3N6.

As a trivial example, if G is a tree, $k = 1$, and the rules and termination conditions are as specified above, it is clear that Player 1 has a winning strategy consisting of simply placing her token on any vertex, then always moving her token towards Player 2's token.

In [5], Sugihara and Suzuki discuss pursuit-evasion games which incorporate the concept of visibility: associated with each edge and vertex of G is a set of vertices and edges which are "visible". In this paper, we consider some variants of these games; in particular, we restrict our attention to grid graphs, for which the visibility rules are intuitive.

The *grid graph* $G_{n,m}$ has vertex set $\{v_{i,j} \mid 0 \leq i \leq n, 0 \leq j \leq m\}$ and edge set $\{(v_{i,j}, v_{k,l}) \mid (i = k \text{ and } |j - l| = 1) \text{ or } (j = l \text{ and } |i - k| = 1)\}$. It is natural to think of the vertices with first subscript equal to i as forming the i th row, and the vertices with second subscript equal to j as the j th column of $G_{n,m}$. All edges are assumed to have unit length. By convention, we will assume the graph is embedded in the plane with $v_{0,0}$ in the upper left corner.

For each vertex, all vertices in the same row are visible, as are all vertices in the same column. Further, all edges joining these vertices are visible. Similarly, for each edge joining two vertices in the same row (column), all vertices and edges in that row (column) are visible.

In this game, tokens move continuously along edges of the graph (rather than discretely from vertex to vertex). The number of edges each token can traverse in a unit of time is specified as a parameter of the game.

Subsequently, we shall consider only the square grid $G_{n,n}$. The results generalise easily to $G_{n,m}$, $n \neq m$, and for brevity, we omit them here. In the form of the pursuit-evasion process we will examine, Player 1 has only one token. For notational simplicity, we will use (i, j) to indicate $v_{i,j}$ and since each player has only one token, we will use "Player i " to refer to both the player and her token. The "perfect knowledge" aspect of the classic game is partially discarded; Player 2 has complete knowledge of the position of Player 1 throughout the game, but Player 1 has no information about the location of Player 2. Further-

more, and most significantly, Player 1's movements are completely specified before the game begins, and this information is available to Player 2.

The rules and termination conditions are as follows:

Rules

- R₁: Players move simultaneously.
- R₂: Player 2's speed never exceeds 1 (the length of an edge).
- R₃: Player 1's speed never exceeds s_c , a constant.

Termination conditions

- T₁: Player 2's position is visible from Player 1's position.
- T₂: Player 2 can prevent Player 1 from winning.

This type of problem has applications in motion planning for industrial robots, where it is vital to avoid conflicts or even collisions; see [5]. As before, the outcome of this game is entirely determined by its parameters, assuming both players pursue optimal strategies. We wish to determine under what conditions Player 1 can be sure of winning this game. More formally:

For $G_{n,n}$, what is the least value S_n such that Player 1 cannot be prevented from winning whenever $s_c \geq S_n$?

For all values of $s_c \geq S_n$, we say that there is a cop-win algorithm. For any value of $s_c < S_n$, the termination condition in T₂ is satisfied immediately.

Sugihara and Suzuki [6] demonstrate that $S_n \leq 2n + 1$. With $s_c = 2n + 1$, Player 1 can start in corner $(0, 0)$ of the grid, then search up and down the columns in sequence. That is, she goes down column 0, then up column 1, etc. It is easily seen that Player 2 cannot escape being seen at some time no greater than the time it takes Player 1 to visit every vertex of the grid.

Sugihara and Suzuki also observe that $S_n > 1$. This is also easily demonstrated: if Player 1 and Player 2 move at the same speed, Player 2 can wait until Player 1 approaches, then move away from Player 1's planned direction of travel. Since

Player 2 has full knowledge of Player 1's planned movements, Player 2 can take evasive action early enough to reach another vertex and get out of sight before Player 1 arrives.

In fact, it is clear that S_n must be a function of n . If $s_c \leq k$, for any constant k , then for sufficiently large n Player 2 can again make use of her knowledge of Player 1's plan and simply move away, when Player 1 gets close, in a direction that will take Player 1 further away after she reaches Player 2's previous location. This is true even if Player 1 knows Player 2's initial location (a variant discussed in greater detail below).

A new upper bound for the minimum cop's speed

We are now able to offer a new upper bound on S_n , and present some evidence that this may be optimal. We first introduce notation for sub-grids: we shall use $[(i, j), (k, l)]$ to denote the rectangular sub-grid with its upper left corner at (i, j) and its lower right corner at (k, l) . Throughout this discussion, we shall use δ, ϵ , and γ to represent arbitrarily small positive values.

Theorem 1. $S_n \leq n + 1$.

Proof. Using the following algorithm, Player 1 will win with $s_c = n + 1$. Suppose Player 1 is at $(0, j)$ at time t_j , and either Player 2 has been seen, or Player 2 is in the sub-grid $[(0, j), (n, n)]$.

1. Player 1 moves down column j from $(0, j)$ to (n, j) then over to $(n, j + 1)$. This requires one time unit.
2. She moves up column $j + 1$ to $(0, j + 1)$, then back to $(0, j)$. This requires one time unit.
3. She repeats step 1.
4. She moves up column $j + 1$ to $(0, j + 1)$. This requires $n/(n + 1)$ time units.

The total elapsed time is $t_{j+1} = t_j + 3 + n/(n + 1)$.

Assertion: At this point Player 2 has either been seen, or else must be in the sub-grid $[(0, j + 1), (n, n)]$.

We will assume that Player 2's actions are aimed at occupying the sub-grid $[(0, 0), (n, j + 1)]$

without being seen, at time t_{j+1} (i.e. falsifying the assertion).

Suppose that at time t_j , Player 2 occupies the sub-grid $[(0, j), (n, j + 1)]$. We may assume she does not occupy column j (else she is visible). During Step 1 of the algorithm, Player 2 cannot cross column j , but can approach column j along row i , for any i . Player 1 visits (i, j) after $i/(n + 1)$ time units have elapsed, so Player 2 cannot commence moving along row i until after that time. Hence at the conclusion of Step 1, Player 2 is at most $1 - i/(n + 1) - \epsilon$ of the distance from $(i, j + 1)$ to (i, j) .

Player 1 crosses row i again after a further $(n - i)/(n + 1)$ time units elapse (during Step 2). Player 2 must have moved onto column j by that time, in order to avoid detection. Thus she must move $i/(n + 1) + \delta$ in $(n - i)/(n + 1)$ time units. This yields $i < n/2$.

Between the time Player 1 crosses row i during Step 2 and the completion of Step 2, $i/(n + 1)$ time units elapse. In total, 2 time units have elapsed since t_j . Player 2, starting on column $j + 1$ and moving along row i , has not had sufficient time to move to either row $i - 1$ or row $i + 1$, since she could not start moving on row i until after Player 1 crossed that row during Step 1. Thus at the completion of Step 2, Player 2 must be on row i , at a distance of no more than $i/(n + 1) - \gamma$ from (i, j) . Since $i < n/2$,

$$\frac{i}{n + 1} - \gamma < 1 - \frac{i}{n + 1}.$$

Player 1 reaches (i, j) during Step 3 after a further $i/(n + 1)$ time units. At that time Player 2 cannot have reached column $j - 1$ (or returned to column $j + 1$) and therefore is visible.

Now suppose that at time t_j , Player 2 occupies the sub-grid $[(0, j + 1), (n, n)]$, and intends to enter $[(0, 0), (n, j + 1)]$ along row i . By the argument just given, when Player 1 crosses row i during Step 1 Player 2 cannot be on column $j + 1$.

Suppose that when Player 1 crosses row i during Step 1, Player 2 is on column $j + 2$. At the conclusion of Step 1, Player 2 has not had sufficient time to cross column $j + 1$. Hence when

Player 1 crosses row i during Step 2, Player 2 must still be on column $j + 2$. At the conclusion of Step 2, Player 2 is at most $i/(n + 1) - \varepsilon$ of the distance from $(i, j + 2)$ to $(i, j + 1)$. Player 1 crosses row i after another $i/(n + 1)$ time units elapse during Step 3. At this time Player 2 must have reached column $j + 1$. At the conclusion of Step 3, Player 2 must be on row i again, and is $(n + 1 - i)/(n + 1) - \delta$ of the distance from $(i, j + 1)$ to (i, j) .

It is apparent that when Player 1 crosses row i during Step 4, Player 2 cannot have reached (i, j) , and is thus visible.

The only remaining possibility is that when Player 1 crosses row i during Step 1, Player 2 is on column k , for some $k > j + 2$. In this case, she clearly cannot have crossed column $j + 1$ at time t_{j+1} . Since she cannot be on column $j + 1$ either, the assertion holds.

Thus in every case, at time t_{j+1} , either Player 1 has seen Player 2, or Player 2 is confined to $[(0, j + 1), (n, n)]$.

By starting this sequence at $(0, 0)$, at which point the preconditions are satisfied, Player 1 can proceed until she wins the game, which occurs no later than t_n . \square

Conjecture. $S_n = n + 1$.

In support of this conjecture, consider the class of all search algorithms which proceed by establishing that sub-grids $[(0, 0), (n, j)]$ are empty, for increasing values of j , ending each phase with a traversal of column j . It is intuitively attractive to suppose that all such algorithms must precede this final traversal of column j with a traversal of column $j - 1$. We make no such claim, but we limit our attention to algorithms which do make such a traversal. More precisely, let \mathcal{A} be the set of search algorithms which proceed from the assertion "Player 2 has been seen, or occupies the sub-grid $[(0, j), (n, n)]$ " to the assertion "Player 2 has been seen, or occupies the sub-grid $[(0, j + 1), (n, n)]$ " by traversing columns j and $j + 1$ in a cyclic fashion any number of times.

Theorem 2. *No algorithm in \mathcal{A} can lead to a guaranteed win for Player 1 unless $s_c \geq n + 1$.*

Proof. With $s_c = n + 1 - \varepsilon$, suppose that Player 1 is at $(0, j)$. Player 2 can escape detection and move into the sub-grid $[(0, 0), (n, j)]$, as follows.

By supposition, Player 1's moves before making the new assertion are to move down column j and up column $j + 1$ some number of times.

Assume Player 2 is located on column $j + 1$, within δ of $(n/2, j + 1)$.

After Player 1 passes $(n/2, j)$, $1 + \gamma$ time units elapse before she reaches $(n/2, j + 1)$. For sufficiently small δ , Player 2 has time to reach column j and move off row i . Thus Player 2 will not be visible when Player 1 reaches $(n/2, j + 1)$. An identical time will elapse before Player 1 returns to $(n/2, j)$, if her algorithm involves another traversal of column j . Again, it is sufficient time for Player 2 to reach column $j - 1$ and move off row i . Player 2 can move arbitrarily far into $[(0, 0), (n, j + 1)]$ without detection. \square

Class \mathcal{A} is not as restrictive as it may appear; most effective search strategies seem to fall into this class.

We now consider three variations of this game. Before doing so, we present a number of minor results.

Lemma 3. *With $s_c > 1$, if the distance between Player 1 and Player 2 is ever ≤ 1 (using the Manhattan metric), then Player 1 wins.*

Proof. Assuming that Player 1 only changes direction at vertices, and never stops moving, then either Player 1 is approaching a vertex from which Player 2 will be visible, or Player 1 has just passed through such a vertex. \square

Lemma 4. *Suppose the current location of Player 2 is restricted to a particular set C of edges and/or vertices. If, in one time unit, Player 1 can pass through all vertices in C , and both end-vertices of each edge in C , then Player 1 wins.*

Proof. Follows immediately from Lemma 3. \square

Lemma 5. *In any cop-win algorithm, every vertex must be visited at least once.*

Proof. If (i, j) is never visited by Player 1, then Player 2 can choose an initial position near (i, j) and move onto row i whenever Player 1 is on column j , and vice versa. Player 2 need never move more than ϵ away from the vertex. Hence no finite speed s_c will permit Player 1 to win. \square

Known starting position

A simple variation is to specify the initial position of both Player 1 and Player 2, and make this information available to both players.

Clearly the best initial position for Player 1 is the centre of the grid. (Note: we are assuming here that n is even. For odd values of n , the argument is similar.) From this position, with $s_c \geq n$, Player 1 can reach Player 2's initial position in one move. Player 2 can be at distance at most 1 from this position, so by Lemma 3, Player 1 wins.

Player 2 has the greatest opportunity for evasion (i.e. the longest possible time before Player 1 can reach Player 2's initial location) if Player 1 starts in one corner of the grid. However, as the following theorem illustrates, the knowledge of Player 2's initial location confers a significant advantage on Player 1.

Theorem 6. *If $s_c \geq n/k$, for any constant k , and Player 1 knows Player 2's initial location, then for sufficiently large n , the game is a cop-win.*

Proof. In no more than $2 * k$ time units Player 1 can reach Player 2's initial position. In this time, Player 2 can move to any of approximately $8 * k^2$ vertices (and connecting edges). If Player 1 can search this area in one time unit, Player 2 will be seen. Since Player 1 must search at a speed of n/k , it is clear that n must be at least $8 * k^3$. We now present a search algorithm to approach this bound.

If $s_c = n$, Player 1 can reach Player 2's initial position in no more than 2 time units, at which time Player 2 can be no more than distance 2

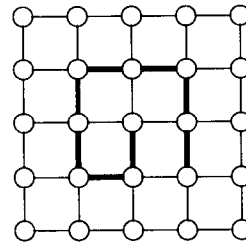


Fig. 1.

away. If $n \geq 8$, Player 1 can now perform in one time unit an exhaustive search of the significant neighbourhood of Player 2's initial position. See Fig. 1 for a simple search pattern that visits every vertex in the region of the grid that Player 2 must occupy, starting at the centre and ending at the lower right corner of the region. By Lemma 4, Player 2 cannot avoid detection.

If $s_c = n/2$, 4 time units may elapse before Player 1 reaches Player 2's initial position. The search area is correspondingly larger. Figure 2 shows a search pattern which begins with the pattern shown in Fig. 1, then continues through the remaining significant vertices and again finishes in the lower right corner. The reader will observe that this search path visits four vertices twice. However, since there exists no hamiltonian path of the area, this redundancy is unavoidable.

The reader may also observe that in both of these figures, Player 2 could in fact have reached

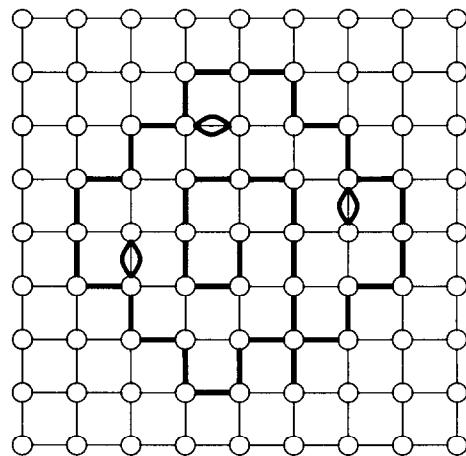


Fig. 2.

one vertex further than is searched in either direction on both the row and the column of her initial position. However, it is unnecessary to search those locations, since at those vertices Player 2 would be visible when Player 1 reached Player 2's initial location.

The pattern for $s_c = n/2$ can in turn be embedded in the obvious fashion in the larger search area required if $k = 3$, etc. Thus we arrive at a simple recurrence relation. If $f(k)$ is defined to be the length of the path Player 1 follows to exhaustively search the area that Player 2 might occupy when, with $s_c = n/k$, Player 1 reaches Player 2's initial position, then

$$f(k) = f(k-1) + 3 + 3 * (4 * k) + 4 * k - 3,$$

which yields

$$f(k) = 8 * (k^2 + k - 1).$$

If $s_c = n/k \geq f(k)$, i.e. $n \geq 8 * (k^3 + k^2 - k)$, Player 1 can follow this path in one time unit, and the game is a cop-win. \square

The interested reader may wish to verify that a similar result *cannot* be obtained for $s_c = n^x$, $x < 1$, except when $n^{2-3x} \leq r$, where r is approximately equal to $1/8$. Thus for $s_c = n^x$, $x \leq 2/3$, we believe that the grids for which cop-win algorithms exist are all very small, even when Player 1 is given Player 2's initial location.

Catching versus seeing

We may modify the game so that T_1 contains only the condition "Player 1 and Player 2 occupy the same location" (i.e. Player 2 must be "caught"). With a fixed and published search sequence for Player 1, Player 2 always wins when $n \geq 2$: she chooses an initial position within ϵ of an internal vertex of the grid (i.e. a vertex with neither coordinate = 0 or = n). Each time Player 1 approaches that vertex, Player 2 moves to a position on one of the edges incident to the vertex that Player 1 will not use on this visit to the vertex. Player 1 will see Player 2, but will not catch her.

We may define a non-trivial form of this game by permitting Player 1 to modify her strategy whenever she sees Player 2.

Theorem 7. *If Player 1 is allowed to revise her planned movements upon sighting Player 2, then for $s_c \geq n/k$, $k \geq 1$, catching Player 2 is no harder than seeing Player 2, for sufficiently large n .*

Proof. Suppose there exists a cop-win algorithm for $s_c \geq n/k$, $k \geq 1$. If $k = 1$, Player 1's strategy to catch Player 2 is simple. When Player 2 is seen, Player 1 moves immediately towards Player 2. Player 2 may subsequently disappear, but only at a vertex. The maximum distance between Player 1 and Player 2 at the time of disappearance is n . Thus Player 2 must be visible again when Player 1 reaches the vertex at which Player 2 disappeared. Furthermore, Player 1 will now be closer to Player 2. Capture is inevitable, regardless of the value of n .

If $k \geq 2$, Player 2 may not be visible when Player 1 reaches the last vertex at which Player 2 was visible. In this case Player 1 may institute an exhaustive search of the area, as described in the previous section. Again, for sufficiently large values of n , at some point during this search Player 2 must be visible to Player 1, and must be closer than at the last sighting. \square

Speed versus search time

Using the algorithm described in Theorem 1, Player 1 takes about $4 * n$ time units to traverse the entire grid. This can be reduced to approximately $3 * n$ by observing that Step 4 of the algorithm not only concludes the search of $[(0, 0), (n, j + 1)]$ but can also be considered as the first step of the next pass of the algorithm. That is, if Player 1 is at $(n, j + 1)$ at the end of Step 3, the following traversal of column $j + 1$ will serve as Step 4 of the current phase of the algorithm, and also as part of Step 1 of the next phase (starting at the bottom rather than the top of the grid).

This raises the question of a speed versus time trade-off. Using T_x to represent the time re-

quired to search the grid with $s_c = x$, we may define $c(x) = x * T_x$ to be the cost of $s_c = x$. Since by Lemma 5 each of $(n + 1)^2$ vertices must be visited, it is clear that $c(x) > n^2$ for all x . It is also easy to observe that for $x \geq 2n + 1$, $c(x) = n^2 + O(n)$.

However, with $s_c = n + 1$, using the algorithm given in this paper, each vertex is visited several times, and thus $c(n + 1) = 3 * n^2 + O(n)$.

This leads to another variation of the original problem: what is the least speed L_n such that there exists a cop-win algorithm in which each vertex is visited only once? It is very easy to show that for $s_c = 2n + 1 - \varepsilon$, the simple traversal described for $s_c = 2n + 1$ (from [6]) will not suffice, but a general proof that $L_n = 2n + 1$ seems to be difficult.

Acknowledgment

This work was supported by a grant from the Natural Sciences and Engineering Research Council of Canada.

References

- [1] M. Aigner and M. Fromme, A game of cops and robbers, *Discrete Appl. Math.* **8** (1984) 1–12.
- [2] P. Frankl, Cops and robbers in graphs with large girth and Cayley graphs, *Discrete Appl. Math.* **17** (1987) 301–305.
- [3] M. Maamoun and H. Meyniel, On a game of policemen and robber, *Discrete Appl. Math.* **17** (1987) 307–309.
- [4] R. Nowakowski and P. Winkler, Vertex-to-vertex pursuit in a graph, *Discrete Math.* **43** (1983) 235–239.
- [5] K. Sugihara and I. Suzuki, On a pursuit-evasion problem related to motion coordination of mobile robots, in: *Proc. 21st Hawaii Internat. Conf. on System Sciences*, Kailua-Kona, Hawaii (1988) 218–226.
- [6] K. Sugihara and I. Suzuki, Optimal algorithms for a pursuit-evasion problem in grids, *SIAM J. Discrete Math.* **2** (1989) 126–143.