Available online at www.sciencedirect.com

ScienceDirect

Theoretical Computer Science 393 (2008) 124–132

Theoretical Computer Science

www.elsevier.com/locate/tcs

The Magnus–Derek game

Z. Nedev[a,*,1], S. Muthukrishnan[b,2]

[a] *University of Victoria, Victoria, BC, Canada*
[b] *Rutgers University, USA*

Received 20 July 2006; received in revised form 18 November 2007; accepted 22 November 2007

Communicated by G. Ausiello

Abstract

We introduce a new combinatorial game between two players: Magnus and Derek. Initially, a token is placed at position 0 on a round table with n positions. In each round of the game Magnus chooses the number of positions for the token to move, and Derek decides in which direction, $+$ (clockwise) or $-$ (counterclockwise), the token will be moved. Magnus aims to maximize the total number of positions visited during the course of the game, while Derek aims to minimize this quantity.

We define $f^*(n)$ to be the eventual size of the set of visited positions when both players play optimally. We prove a closed form expression for $f^*(n)$ in terms of the prime factorization of n, and provide algorithmic strategies for Magnus and Derek to meet this bound.

We note the relevance of the game for a *mobile agent* exploring a ring network with faulty sense of direction, and we pose variants of the game for future study.

Keywords: Two-person games; Mobile agent; Network with inconsistent global sense of direction; Algorithmic strategies; Discrete mathematics; Combinatorial number theory

1. Introduction

We will introduce a new class of combinatorial games that we call the *Magnus–Derek* games, and we will study in detail the basic version of the game.

The game is played on a circular table with n positions labelled consecutively from 0 to $n-1$, and with a token initially placed at position 0. The two players are called Magnus (from *magnitude*) and Derek (from *direction*). Suppose that the token's current position is i. A round consists of Magnus calling a magnitude ℓ with $0 < \ell \leq \frac{n}{2}$ followed by Derek calling a direction $+$ (clockwise) or $-$ (counterclockwise). The token is then moved to position $i+\ell \bmod n$ or $i-\ell \bmod n$ according to Derek's choice of direction. Thus, Magnus must choose a set $N = \{i+\ell, i-\ell\}$

* Corresponding author. Tel.: +1 250 721 7437.
E-mail addresses: znedev@gmail.com (Z. Nedev), muthu@cs.rutgers.edu (S. Muthukrishnan).
1 Part of the work was done while visiting DIMACS.
2 DIMACS Permanent Member.

0304-3975/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2007.11.016

of two possibilities for the next position, and Derek chooses which of these positions is realized. (In our explanations of the algorithms, rather than stating "the token moves" we will often say "the next position will be" or even say "Magnus moves" identifying Magnus with the token.) Magnus's goal is to maximize the cardinality of the set of all positions visited by the token in the course of the game. Derek's goal is to minimize the cardinality of this set. Note that revisiting a position does not affect the cardinality.

The current token position, the size of the table (n), and the set of already visited positions are known to both players at all times. The algorithmic strategies we present for both players are *adaptive*: that is, a player's move depends both on the set of already visited positions and on the current token position. We will assume that both players know the factorization of n at the beginning of the game.

We define $f^*(n)$ to be the eventual cardinality of the set of positions visited when both players play optimally. We will show that $f^*(n)$ is well defined: Magnus has a strategy to ensure that at least $f^*(n)$ positions are visited, and Derek has a strategy to ensure that at most $f^*(n)$ positions are visited, irrespective of each others' strategies.

Thus we examine three problems:

1. What is $f^*(n)$ for a given n? Clearly $f^*(n) \leq n$.
2. What algorithm for Magnus will ensure that at least $f^*(n)$ positions are visited? We prefer an algorithm that minimizes the number of moves required to achieve this goal. Less importantly, we want to minimize the time required for Magnus to compute a magnitude in each round.
3. What algorithm for Derek will ensure that no more than $f^*(n)$ positions are visited? Again, we want to minimize the time required for Derek to compute a direction in each round. We are also interested in a lower bound on the number of moves until $f^*(n)$ positions are visited.

2. The main result

In this section we state the main result of our paper. Proofs of the result are in Sections 4–6.

Theorem 1.

- *If $n = 2^k$ for some k, then $f^*(n) = n$. In this case Magnus has a strategy for visiting all positions in exactly $n - 1$ moves; the strategy can be formulated as a sequence of $n - 1$ magnitudes at the start of the game, and calculating the sequence requires $O(n)$ time.*
- *If $n = pm$ where p is the smallest odd prime factor of n, then $f^*(n) = \frac{(p-1)n}{p}$ and there exists a strategy for Derek to ensure at most $f^*(n)$ positions are visited. Derek's calculations require $O(1)$ time in each round. Furthermore there exists a strategy for Magnus to visit at least $f^*(n)$ positions in $g(n)$ moves where:*
  - *If $m = 1$, $g(n) = O(n^2)$ moves.*
  - *If $m = 2^k$, $g(n) = O(p^2 + n)$ moves.*
  - *Otherwise, if $m = 2^k m_1$ where m_1 is bigger than two and odd, $g(n) = O(\frac{n^2}{p})$ moves.*

  Magnus's computations require $O(1)$ time in each round.

As n increases from 1 to ∞, $f^*(n)$ is 1, 2, 2, 4, 4, 4, 6, 8, 6, 8, 10, 8, 12, 12, 10, 16, ... This sequence has not previously appeared in the Encyclopedia of Integer Sequences [3].

3. Motivation

A *mobile agent* (MA) is an autonomous computer program that can move itself from host to host in an MA-enabled network. Mobile agents have been proposed as efficient solutions for many distributed computing and network maintenance tasks; one example is intrusion detection in computer networks [7]. A common task for an MA is to visit as many as possible of the hosts or nodes in a network.

The notion of a network with a sense of direction is defined in [4,6]. In [5], the following informal definition is given for a sense of direction in a network with a ring architecture:

> "Sense of direction refers to the capability of a processor to distinguish between its adjacent communication lines, according to some globally consistent scheme. In a ring network this property is usually referred to as *orientation*, which expresses the processor's ability to distinguish between left and right, where 'left' means the same to all processors."

A network may be without a sense of direction due to a hardware or software fault, or due to malicious activity by an intruder. We consider a ring network of n hosts in which there is *inconsistent global sense of direction*. That is, if $H_0, H_1, \ldots, H_{n-1}$ are our n hosts, H_i is connected to (and can communicate only with) $H_{i+1(\bmod n)}$ and $H_{i-1(\bmod n)}$. Furthermore, although each host can distinguish between its two adjacent neighbours, the labelling "left" and "right" is not consistent across hosts and may not be consistent over time. The game is designed to model an MA attempting to explore or visit all hosts in such a network.

In our model we assume that each time the MA moves it has no control over the initial *direction* ("left" or "right") of its movement. However, the MA may predetermine how many hosts to pass through before stopping (the *magnitude* of its movement), and is able to keep moving in the same direction on the ring until it stops. The MA can only explore hosts at which it stops; it is unable to explore the hosts through which it passes while moving.

In this paper we give an algorithmic strategy for an MA exploring such a network. The strategy ensures that the MA explores as many hosts as possible, even when its directions of movement are maliciously chosen to force the MA to visit a minimal number of hosts. Secondary goals of the MA are to (1) minimize the number of moves necessary, and (2) efficiently compute the magnitude of each move.

We formulate the problem as an adversarial game between two players: Magnus, representing the MA, attempts to visit as many hosts as possible. Derek, representing a malicious adversary, attempts to prevent Magnus from achieving this goal.

4. When $n = 2^k$

In this section, we consider the case when $n = 2^k$. We use $C(\ell, d, s)$ to denote $\{s + i \cdot d \mid 0 \leq i < \ell\}$; that is, the set of ℓ positions beginning at s that are successively d apart.

Lemma 2. *If $n = 2^k$, there is a strategy for Magnus forcing the token to visit all positions, and Magnus does not need to adapt as the game progresses. Rather, Magnus's strategy (a sequence of $n - 1$ magnitudes) can be determined at the start of the game.*

Proof. The strategy (and the proof) can be described inductively on the power k. Let $k = 1$ so that $n = 2$. Magnus calls the magnitude 1 and irrespective of Derek's chosen direction, the token moves to the unvisited position. Hence $f^*(2) = 2$ and only a single move is required.

Now let $k > 1$. We assume that $f^*(2^{k-1}) = 2^{k-1}$ and that there exists a sequence $d_1, d_2, \ldots, d_{2^{k-1}-1}$ of magnitudes to achieve visiting all positions irrespective of Derek's choice of directions.

Let $n = 2^k$. We partition the set $C(n, 1, 0)$ into $C(\frac{n}{2}, 2, 0)$ and $C(\frac{n}{2}, 2, 1)$. Magnus first uses $2d_1, 2d_2, \ldots, 2d_{\frac{n}{2}-1}$ to visit all positions in $C(\frac{n}{2}, 2, 0)$. Then, regardless of the resultant position, Magnus calls 1 forcing the token to move to some position j in $C(\frac{n}{2}, 2, 1)$. Thereafter, Magnus's choices would be $2d_1, 2d_2, \ldots, 2d_{\frac{n}{2}-1}$ to visit all positions in $C(\frac{n}{2}, 2, 1)$. Therefore, all n positions are visited and the total number of moves is $n - 1$.

The proof is constructive and Magnus's strategy is easy to generate in $O(n)$ time at the start of the game. If $n = 2^k$ for some k, then Magnus can declare the following sequence of moves:

$$\left(\frac{n}{2^1}; \frac{n}{2^2}; \frac{n}{2^1}\right); \frac{n}{2^3}; \left(\frac{n}{2^1}; \frac{n}{2^2}; \frac{n}{2^1}\right); \frac{n}{2^4}; (:::); \ldots; \frac{n}{2^k}; (:::);$$

Here (:::) means a repetition of all the moves starting from the beginning of the sequence until, but not including, the move before the current (:::). ■

Notice that the sequence as a whole is a palindrome, that its first and second halves are each palindromes, and so forth.

Example. For $n = 16$, Magnus's choices are (in order and with brackets for clarifications):

((8; 4; 8;)2; (8; 4; 8;))1; ((8; 4; 8;)2; (8; 4; 8;)).

Henceforth, we will consider values of n that are not exact powers of two.

5. Derek's strategy and proof of $f^*(n) \leq \frac{(p-1)n}{p}$

This is the easier part of Theorem 1.

Lemma 3. *Let $n = pm$ where p is the smallest odd prime factor of n ($p > 2$). Then $f^*(n) \leq \frac{(p-1)n}{p}$ and there exists a strategy for Derek with $O(1)$ running time in each round ensuring that at most $f^*(n)$ positions are visited.*

Proof. Consider the following strategy for Derek. At the beginning of the game, he chooses a position k where $1 \leq k \leq p - 1$. We claim that Derek can prevent any position congruent to $k \bmod p$ from being visited. The token starts at position $i = 0$, which is not congruent to $k \bmod p$. In each round Derek must choose the next position from the set $\{i + \ell, i - \ell\}$ where i denotes the current position (not congruent to $k \bmod p$) and ℓ denotes the magnitude chosen by Magnus. At least one of $i + \ell$ and $i - \ell$ is not congruent to $k \bmod p$, for, otherwise, both $i + \ell = k \bmod p$ and $i - \ell = k \bmod p$ and by adding the two equations we obtain the contradiction $2i = 2k \bmod p$, or $i = k \bmod p$.

Derek's algorithm insures that the token never visits any position in $C(\frac{n}{p}, p, k)$, the set of positions congruent to $k \bmod p$. This set has cardinality $\frac{n}{p}$ and this immediately implies that

$$f^*(n) \leq \left(n - \frac{n}{p}\right) = \frac{(p-1)n}{p}.$$

Derek's strategy does not depend on the entire set of previously visited positions, but only on the currently visited position and Magnus's choice of ℓ. In each turn, the calculations for Derek's choice run in $O(1)$ time. ■

Thus, we have proved the Derek's part in Theorem 1. In fact, we have proved a stronger result. Derek can specify in advance a set of positions ($C(\frac{n}{p}, p, k)$ for a suitable k) and ensure that the token never visits those positions. It may be of further interest to find a strategy for Derek that not only achieves at most $f^*(n)$ positions visited but also prevents the token from visiting $f^*(n)$ positions before a certain minimum number of rounds are played in the game.

6. Magnus's strategy and proof of $f^*(n) \geq \frac{(p-1)n}{p}$

For greater insight into the problem, we will show two quite different algorithmic strategies for Magnus to visit at least $f^*(n)$ different positions. The two strategies share a common idea: although in many situations Magnus cannot reach a given predetermined position, he can often reach at least one of two predetermined positions.

6.1. *First strategy for magnus*

To prove the general result for $n \neq 2^k$, we will first state the above idea as a lemma, and then use it to derive an algorithm for Magnus.

Lemma 4. *Let $n \neq 2^k$, and let the current position be K, and let I and J be two distinct positions such that $J > I$ and* $\gcd(J - I, n) = 2^f$ *for some non-negative integer f* (gcd *is the greatest common divisor). Then Magnus can find a set of magnitudes with cardinality $O(n)$ such that the token can be moved from K to one of I and J regardless of Derek's choices. These magnitudes can be found in time $O(n^2)$.*

Proof. With respect to the pair of distinct positions I and J, we partition all positions $0, \ldots, n - 1$ into classes as follows. Class C_0 contains I and J. Class C_1 contains every position $p \notin C_0$ for which there exists a magnitude d such that both $p + d \bmod n \in C_0$ and $p - d \bmod n \in C_0$. That is, if the current position is in C_1, Magnus can call a magnitude such that the token ends in a position in C_0 irrespective of Derek's choice. More generally, class C_m contains every position $p \notin C_0 \cup C_1 \cup \cdots \cup C_{m-1}$ such that there exists a magnitude d for which both $p + d \bmod n \in C_0 \cup C_1 \cup \cdots \cup C_{m-1}$ and $p - d \bmod n \in C_0 \cup C_1 \cup \cdots \cup C_{m-1}$.

We define $C_m^* = C_0 \cup C_1 \cup \cdots \cup C_m$. We will prove that if $|C_{m+1}| = 0$ for any m, then $C_m^* = \{0, \ldots, n - 1\}$. In other words, as we let i increase and construct C_i, if we cannot find at least one position for C_m, then we have exhausted all positions. We prove this by contradiction.

Suppose that $|C_{m+1}| = 0$ for some m and that there exists a position $\alpha \notin C_m^*$. Since C_1 contains at least 1 position (see construction, below), $(m + 1) \geq 2$ and we can always find at least three positions in C_m^*. Consider all positions in C_m^* sorted in increasing order $\alpha_1 < \alpha_2 < \ldots < \alpha_t$. In the next eleven lines, the index arithmetic is by mod t; for example, index $t + 1 \equiv 1$ (by mod t).

- If $\alpha_i - \alpha_{i-1}$ is even for some i, then there is an odd number of positions between these positions. The position equidistant from both, namely $\alpha_{i-1} + \frac{\alpha_i + \alpha_{i-1}}{2}$, belongs to C_{m+1}, giving the contradiction.
- Suppose that $\alpha_i - \alpha_{i-1}$ is odd for all i. Consider any three consecutive positions $\alpha_{\ell-1}, \alpha_\ell, \alpha_{\ell+1}$ from C_m^*. Then, $\alpha_{\ell+1} - \alpha_{\ell-1}$ must be even. If the midpoint $\alpha_{\ell-1} + \frac{\alpha_{\ell+1}+\alpha_{\ell-1}}{2}$ is not in C_m^*, it must belong to C_{m+1}, giving the contradiction. Otherwise, the midpoint equals α_ℓ. Repeating this argument for each consecutive triple from C_m^*, we obtain either a contradiction, or that $\alpha_1, \alpha_2, \ldots$ are arranged as vertices of a regular polygon of less than n vertices (since $\alpha \notin C_m^*$) with sides $\alpha_i - \alpha_{i-1} = L$ for all i, where $L \geq 3$ and L is odd. Therefore, $n = aL$ for some positive integer a. Furthermore, since $I, J \in \{\alpha_1, \alpha_2, \ldots\}$ they must be vertices of the polygon and, consequently, $J - I = bL$ for some positive integer b. This implies that $\gcd(J - I, n) = cL$ for some positive integer c. Then $\gcd(J - I, n) \neq 2^f$, contradicting the assumption in the lemma.

We have now proven that C_l^* contains all positions $0, 1, \ldots, n-1$ for some $l \leq n-1$. We now show how to construct the classes by induction and find an algorithm for Magnus to reach I or J starting from K. For efficiency, we consider each class to be stored as a list.

Trivially, $C_0 = \{I, J\}$. For convenience, we call the elements of $\{0, \frac{1}{2}, 1, 1+\frac{1}{2}, \ldots, n-2, n-1-\frac{1}{2}, n-1, n-\frac{1}{2}\}$ *points on the table*. In other words, the set of all points on the the table is the union of the set of all positions and the set of all (non-integer) midpoints between them.

To construct C_1, there are only four candidates to consider: the two points diametrically opposite I and J, and the two midpoints between I and J (namely, the two points equidistant from I and J). From these four possible points, we take only those that are integers, and therefore valid positions. If n is odd, C_1 contains either $I + \frac{J-I}{2}$ or $I + \frac{J-I+n}{2} \bmod n$, depending on which is an integer. If n is even, C_1 contains the two positions opposite to I and J, respectively $I + n/2 \bmod n$ and $J + n/2 \bmod n$. When n is even, the two midpoints are either both integers or both non-integers; if they are integers, we also include them in C_1. Notice that it is not possible for I and J to be diametrically opposite to each other. If they were, $\gcd((J - I), n) = \gcd(\frac{n}{2}, n) = \frac{n}{2}$ and cannot be 2^f since $n \neq 2^k$. Therefore, $|C_1^*| \geq 3$ regardless of whether n is even or odd.

Now suppose that we have already constructed $C_0, C_1, \ldots, C_{m-2}, C_{m-1}$ as lists. Notice that this defines an order for each element in C_{m-1}^* (and, eventually, will define an order for $\{0, 1, \ldots, n-1\}$): first come the elements from C_0, in their order, then the elements from C_1, in their order, and so forth.

For each $p \in C_{m-1}$, we consider its diametrically opposite point, $p + \frac{n}{2} \bmod n$, and all midpoints between p and any $q \in C_{m-1}^*$ with q occurring before p in the order. We add to C_m whichever of these are integers and do not already belong to C_{m-1}^* (or C_m). The construction requires $O(j)$ time for a table position in the jth place in the order; therefore, the entire process takes $O(1) + O(2) + \cdots + O(n) = O(n^2)$ time.

Each position $i = 0, \ldots, n-1$ belongs to some class $C_{h(i)}$. The construction also shows how to determine for each initial position i a magnitude d_i such that Magnus can force the token to a position in class $C_{h(i)-1}$, irrespective of Derek's choice.

Now, suppose that K belongs to some class $C_{h(K)}$. Magnus can choose d_K to move to a position in $C_{h(K)-1}$. Iterating this process, the token finally moves to a position in C_0, proving the lemma. ■

Using Lemma 4, we can now prove that $f^*(n) \geq (p-1)\frac{n}{p}$.

Lemma 5. *Let $n = pm \neq 2^k$ where m, k are non-negative integers and p is the smallest odd prime factor of n. Then there exists an $O(n^3)$ time algorithm for Magnus that produces a sequence of $O(n^2)$ moves such that $f^*(n) \geq (p-1)\frac{n}{p}$.*

Proof. From any current position K, if there exist two unvisited positions I and J, $J > I$, such that $\gcd(J-I, n) = 2^f$ for some integer f, Magnus employs the algorithm in Lemma 4 to reach either I or J. After Magnus reaches either I or J, Magnus repeats this procedure until no pair of unvisited positions I and J satisfy $gcd(J - I, n) = 2^f$. We claim that at most $\frac{n}{p}$ positions remain unvisited.

Suppose otherwise. We partition all n positions into $\frac{n}{p}$ consecutive groups of p positions as follows: $(0, 1, 2, \ldots, p-1), (p, p+1, p+2, \ldots, 2p-1), \ldots, (n-p, n-p+1, \ldots, n-1)$. By the pigeonhole principle, there will be at least one group containing two unvisited positions, say I and J. We have $|J - I| \leq p - 1$ since they both belong to the same group. Since p is the smallest odd prime factor of n, $\gcd(J - I, n)$ must be a power of 2 (including $2^0 = 1$), which is a contradiction. ■

The above strategy for Magnus proves the main theorem and yields insight into the structure of the problem. We will now give an alternative, more efficient strategy for Magnus.

6.2. Second strategy for magnus

First, we will provide a strategy for the simple case when n is prime and then use it as a tool for the general case $n = pm$. Although the techniques of this strategy differ from the previous one, it is built on the same substrategy: in many situations Magnus can reach at least one position from a set of two.

Lemma 6. *If $n > 2$ is prime, then $f^*(n) \geq n - 1$ and there exists a corresponding strategy for Magnus that requires $O(n^2)$ moves. At each move Magnus's computations run in $O(1)$ time.*

Proof. First, we provide a $O(n)$ steps procedure for Magnus to reach at least one position from a set of two different positions α and β, $\beta > \alpha$. Let $d = \beta - \alpha \bmod n$ and say Magnus is at position p_0 currently. Then we can represent p_o as

$$p_0 = \alpha + (p_0 - \alpha)1 = \alpha + \left[(p_0 - \alpha)d^{-1}\right]2^0 d = \alpha + \mu_0 2^0 d$$

where d^{-1} is the inverse of d and $\mu_0 = \left[(p_0 - \alpha)d^{-1}\right]$.

Let p_i be the token's position at the end of step i for $i = 0, 1, \ldots, n-1$. A step does not always correspond to a round, so it is possible $p_i = p_{i-1}$, in which case only the formula for the current position changes. Consider the following strategy for Magnus. At each step i, if μ_{i-1} is even, let $\mu_i \leftarrow \mu_{i-1}/2$ and no move is performed; only the current position changes its representation to $p_{i-1} = p_i = \alpha + \mu_i 2^i d \bmod n$. If μ_{i-1} is odd, Magnus calls $\ell = 2^{i-1}d \bmod n$ and moves to

$$p_i = \left[\alpha + \mu_{i-1}2^{i-1}d\right] \pm 2^{i-1}d = \alpha + \left[\mu_{i-1} \pm 1\right]2^{i-1}d = \alpha + \left[\frac{\mu_{i-1} \pm 1}{2}\right]2^i d$$

depending on whether Derek's response is $+$ or $-$. Setting $\mu_i = \frac{\mu_{i-1}+1}{2}$ if Derek's response is $+$ or $\mu_i = \frac{\mu_{i-1}-1}{2}$ otherwise, the position at the end of step i can be represented again as

$$p_i = \alpha + \mu_i 2^i d \bmod n.$$

After $n-1$ iterations, $\mu_{n-1} = 0$ or 1 since $\mu_0 \leq n-1$ and $\mu_i < \mu_{i-1}$ unless μ_{i-1} was 0 or 1. Hence,

$$p_{n-1} = \alpha + 0 \times 2^{n-1}d = \alpha \quad \text{or} \quad p_{n-1} = \alpha + 1 \times 2^{n-1}d \bmod n.$$

By Fermat's little theorem [2], we know that $2^{n-1} = 1 \bmod n$ for prime n. Hence, Magnus is in either α or $\alpha + d = \beta$ after $O(n)$ steps. Therefore, the procedure takes $O(n)$ moves (since one step corresponds to zero or one round), and each step takes $O(1)$ time to calculate μ_i.

Now starting with any position, Magnus has a strategy to achieve $f^*(n) \geq n-1$. Whenever there are two or more unvisited positions, he applies the above substrategy of $O(n)$ moves for a total of $O(n^2)$ moves, giving the lemma. ■

The lemma holds regardless of the starting position. Also, the easiest way to calculate the new $2^i d$ is to multiply the old $2^{i-1}d$ by 2 mod n. Combining Lemma 6 with Lemma 3 gives the value for $f^*(n)$ when n is prime. As an aside, the study of the length of the μ_i sequence with adversarial sequence of $+$ and $-$ responses from Derek may be of independent interest. Now we turn to the general case of n.

Lemma 7. *Let $n = 2^k p_1 p_2 \ldots p_s$ where $p_1, p_2, \ldots, p_s$ are prime numbers, $3 \leq p_1 \leq p_2 \leq \ldots \leq p_s$ and $s \geq 1$ is an integer. Then there exists a strategy for Magnus such that $f^*(n) \geq \frac{(p-1)n}{p}$ and the number of moves and the running time for calculating the strategy is bounded as follows. If $s = 1$, that is, $n = 2^k p_1$, so $(n/p_1) = 2^k$, then the upper bound is $O(p_1^2 + n)$. Otherwise, the upper bound is $O(n^2/p_1)$.*

Proof. For convenience let $p = p_1$, that is, we denote with p the smallest odd prime factor of n. We partition the set of all positions, $C(n, 1, 0)$, into p sets, $C(n/p = m, p, i)$, for $i = 0, 1, \ldots, p-1$.

As a preliminary, we show that with $O(p)$ moves Magnus can visit a position from any two sets $C(n/p, p, j_1)$ and $C(n/p, p, j_2)$, where $j_1 \neq j_2$. Magnus uses an adaptation of the technique from Lemma 6. He calculates the magnitudes to call as if the table size was p; if his current position is k, he assumes that the current position is $k \bmod p$; and he plays as if he wanted to visit one of the two positions j_1 or j_2 on that imaginary table. The positions he visits will not be the same as if he were playing on a table with size p, but since the table size $n = pm$, they will be congruent by (mod p) to the positions he would have visited if the table size was p. After $O(p)$ moves, he will be in a real position that is either congruent to j_1 or j_2, and, therefore, the position will belong to $C(n/p, p, j_1)$ or $C(n/p, p, j_2)$.

Magnus proceeds in two phases and inductively uses the same two phase strategy on smaller problem instances.

Phase 1. Phase 1 consists of the repetition of two main steps: A and B. Magnus applies step A after the token comes for the first time on a position from any set $C(n/p = m, p, i)$. In step A, Magnus plays as if the table size were n/p:

- if n/p is a power of 2, he applies the strategy from Section 4. On each round, he calculates an intermediate magnitude ℓ, playing as if only the positions in $C(n/p, p, i)$ existed, but then he calls $p\ell$.
- if n/p is not a power of 2, he applies recursively this two phase algorithm. On each round, he calculates an intermediate magnitude ℓ, playing as if only the positions in $C(n/p, p, i)$ existed, but then he calls $p\ell$.

When he has visited at least $f^*(n/p)$ positions from $C(n/p, p, i)$, we say that Magnus has explored the set $C(n/p, p, i)$ and then he performs step B. There may be some unvisited positions left from $C(n/p, p, i)$.

In step B, Magnus chooses any two sets $C(n/p, p, j_1)$ and $C(n/p, p, j_2)$ never explored before and using the adaptation of the technique from Lemma 6, Magnus comes to one of these sets in $O(p)$ moves. Then he applies step A again.

Phase 1 ends after $p-1$ of the $C(n/p, p, i)$ sets have been thus explored and in each navigated set at least $f^*(n/p)$ positions have been visited.

Let at the end of this phase, the unexplored set be $C(n/p, p, \gamma)$. There may also exist positions not in $C(n/p, p, \gamma)$ that Magnus has not visited; if no such positions exist, Phase 2 is not needed.

Phase 2. Consider any position $\alpha \notin C(n/p, p, \gamma)$. Then $\alpha \in C(n/p, p, \beta)$, with $\beta \neq \gamma$. Magnus can use the following strategy to get with $O(p)$ moves into either α or a position from $C(n/p, p, \gamma)$. First, using the adaptation of the technique from Lemma 6 described above, in $O(p)$ moves, Magnus comes into a position in either $C(n/p, p, \gamma)$ or $C(n/p, p, (\beta + \gamma)/2)$ where the division / is performed on prime field $[p]$. Suppose that Magnus gets onto a position δ in $C(n/p, p, (\beta + \gamma)/2)$. Then Magnus's next move is $\alpha - \delta$. If Derek responds $+$, Magnus reaches $\delta + (\alpha - \delta)$ which is α. Else, if Derek responds $-$, then Magnus reaches $\delta + (\delta - \alpha) = 2\delta - \alpha$ which is a position in $C(n/p, p, \gamma)$. This is because we have $\delta = (\beta + \gamma)/2 + p\ell_1$ and $\alpha = \beta + p\ell_2$ for some non-negative integers ℓ_1 and ℓ_2. Thus $2\delta - \alpha = \gamma + p(2\ell_1 - \ell_2)$.

Using the above strategy repeatedly with unvisited positions outside of the set $C(n/p, p, \gamma)$ Magnus can either visit them all in which case the phases are complete or come onto a position in $C(n/p, p, \gamma)$. In this second case, Magnus explores $C(n/p, p, \gamma)$ by using this two phase algorithm recursively as in step A of Phase 1. Thus, Magnus visits $f^*(n/p)$ positions in $C(n/p, p, \gamma)$ and the phases are complete. ■

We can now count the number of visited positions. There are two possibilities: either all positions in each set $C(n/p, p, i)$, where $i \neq \gamma$ are visited, or all $C(n/p, p, i)$'s are visited and explored using the two phase strategy. Hence, we have

$$f^*(n) \geq \min\{(p-1)n/p, pf^*(n/p)\}.$$

Observe that inductively we can set $f^*(n/p) \geq \frac{(p-1)(n/p)}{p}$ since either n/p is exact power of two in which case $f^*(n/p) = n/p$ or otherwise because p is no larger than the smallest odd prime factor of n/p. That proves $f^*(n) \geq \frac{(p-1)n}{p}$.

We can upper bound the number of moves, say $g(n)$.

Case 1. If $s = 1$, that is, $n = 2^k p$ so $(n/p) = 2^k$.

Then once Magnus reaches for the first time a set $C(n/p, p, i \neq \gamma)$, he can use the algorithm from Section 4 to visit all positions in it for $(n/p) - 1$ moves. So $g(n) \leq p^2 + (p-1)(2^k - 1) \leq p^2 + n$.

Case 2. If n has at least two odd prime factors, that is, $s \geq 2$ and $n = 2^k p_1 p_2 \ldots p_s$ where $p_1, p_2, \ldots, p_s$ are the odd prime factors of n sorted in increasing order.

Then Magnus needs at most p_1^2 moves in total to get for the first time into each of the sets $C(n/p_1, p_1, i \neq \gamma)$'s. Together for Phase 1 and Phase 2, the total number of moves for navigating within the various $C(n/p_1, p_1, i)$'s is upper bounded by

$$(p_1 - 1)g(n/p_1) + g(n/p_1) + (p_1 - 1)(n/p_1)(1/p_1)(p_1)$$

where the first summand is for navigating within the $p_1 - 1$ sets $C(n/p_1, p_1, i \neq \gamma)$'s in Phase 1, and the second summand is for navigating $C(n/p_1, p_1, \gamma)$ if Magnus reaches positions in it in Phase 2. The third summand is for visiting all unvisited nodes left after Phase 1 in the $p_1 - 1$ number of sets $C(n/p_1, p_1, i \neq \gamma)$. To visit each of these unvisited nodes takes at most p_1 moves and in each such set there are at most $(1/p_1)(n/p_1)$ nodes left unvisited after Phase 1. Hence,

$$g(n) \leq p_1^2 + p_1 * g(n/p_1) + n. \tag{1}$$

Using the above inequality (1), we get that for $j = 1, 2, \ldots, s - 2$ we have

$$g\left(\frac{n}{p_1 p_2 \ldots p_j}\right) \leq p_{j+1}^2 + \frac{n}{p_1 p_2 \ldots p_j} + p_{j+1} * g\left(\frac{n}{p_1 p_2 \ldots p_{j+1}}\right). \tag{2}$$

From Case 1 it follows that

$$g\left(\frac{n}{p_1 p_2 \ldots p_{s-1}}\right) \leq p_s^2 + \frac{n}{p_1 p_2 \ldots p_{s-1}}. \tag{3}$$

Applying recursively the inequalities from (2) and from (3) to the right-hand side of inequality (1), we get

$$\begin{aligned} g(n) &\leq n + p_1^2 + p_1 * g(n/p_1) \leq 2n + p_1 p_2^2 + n + p_1 p_2 g\left(\frac{n}{p_1 p_2}\right) \\ &\leq 4n + p_1 p_2 p_3^2 + n + p_1 p_2 p_3 g\left(\frac{n}{p_1 p_2 p_3}\right) \leq \ldots \leq 2n(s-1) + p_1 p_2 \ldots p_{s-1} g\left(\frac{n}{p_1 p_2 \ldots p_{s-1}}\right) \\ &\leq 2n(s-1) + p_1 p_2 \ldots p_{s-1}\left(p_s^2 + \frac{n}{p_1 p_2 \ldots p_{s-1}}\right) \leq 2n(s-1) + np_s + n \leq np_s + n(2s-1). \end{aligned} \tag{4}$$

Since obviously $(2s - 1) \leq p_1 p_2 \ldots p_{s-1}$, we have the simpler $g(n) \leq np_s + n^2/p_s \leq 2n^2/p_1 = O(n^2/p_1)$.

Because, at each move, Magnus's computations in Lemma 6 run in $O(1)$ time, here, in the general case of n, Magnus's computations run also in $O(1)$ time. ■

Both the number of moves and the running time of the algorithm for Magnus are non-monotonic functions over n; similarly, $f^*(n)$ is also non-monotonic with increasing n. These three factors depend on the factorization of n, and especially on its smallest odd prime factor p.

7. Concluding remarks

We have initiated the study of two-person games that we call the Magnus–Derek games. We have proved tight existential bounds and shown matching algorithmic strategies on the basic version of the problem. An immediate open problem is if we can improve the number of moves (or the running time) of the algorithms.

The following variants for the game may be of further interest.

- *Dual problems.* Say the game stops after no more than m rounds, and let $f_m^*(n)$ be the size of visited set if both players play optimally. What is $f_m^*(n)$ and what are associated strategies?
- *Off-line strategy construction.* Given a sequence $d_1, d_2, \ldots, d_k$ of Magnus's moves, design corresponding **+**'s or **–**'s for Derek so that $f^*(n)$ is minimized.
- *Non-adaptive strategies.* Limit Magnus and/or Derek to use only *non-adaptive* strategies, that is, Magnus (resp. Derek) prespecifies a sequence of moves without knowing Derek's (resp. Magnus's) responses, and evaluate the sequence when Derek (resp. Magnus) later specifies responses in order to optimize their goal. What is $f^*(n)$ and what are associated non-adaptive strategies?
- *Limit revisits.* Suppose that Magnus may not visit a position more than say k times. Then, what is $f_k^*(n)$ and what are associated strategies?

- *Random strategies.* Suppose that Magnus or Derek used random strategies. The problem is to estimate the expected $f^*(n)$. This is akin to random walks with random steps, but differs in having either one of the player being deterministic or adversarial.
- *Limiting information.* Suppose that Magnus and/or Derek do not know n or they know n but do not have a way to factorize it. This is similar to graph exploration problems studied in theoretical computer science with limited number of tokens or memory [1].
- *Making each player nag as well as move.* The two players have asymmetrical roles. For each move, first, Magnus chooses its magnitude, then Derek determines its direction. While Magnus wants to maximize the cardinality of the set of visited positions, Derek wants to maximize the cardinality of the set of unvisited positions. In other two-person games too, this type of asymmetry is seen (for example, the Pusher–Chooser games of [8]). We propose a modified game in which Magnus and Derek alternate their roles each round. Positions visited by Magnus are colored red, and those by Derek are colored blue. Derek is restricted to non-red positions and Magnus to non-blue. What are suitable strategies and how many positions are visited by each player?

Acknowledgements

We are grateful to the following people and institutions for their advice, discussion, and support: Jeffrey O. Shallit, Noam Sturmwind for editing assistance, Valerie King, Uriel Feige, Nicola Santoro, Graham Cormode, Richard Guy, Susan Hope and Peter Winkler; the School of Computer Science at University of Waterloo and the DIMACS Center.

References

[1] M.A. Bender, A. Fernandez, D. Ron, A. Sahai, S. Vadhan, The power of a pebble: Exploring and mapping directed graphs, in: Proc. of the 30th Annual ACM Symposium on Theory of Computing, STOC, 1998, pp. 269–278.
[2] I. Niven, H. Zuckerman, An Introduction to the Theory of Numbers, John Wiley and Sons Inc, 1960.
[3] N. Sloane, On line encyclopedia of sequences. http://www.research.att.com/~njas/sequences/.
[4] N. Santoro, Sense of direction, topological awareness and communication complexity, SIGACT News 16 (2) (1984) 50–56.
[5] H. Attiya, J. van Leeuwen, N. Santoro, S. Zaks, Efficient elections in chordal ring networks, Algorithmica 4 (1989) 437–446.
[6] P. Flocchini, B. Mans, N. Santoro, Sense of direction: Definitions, properties, and classes, Networks 32 (3) (1998) 165–180.
[7] Wayne A. Jansen, Intrusion detection with mobile agents, Computer Communications 25 (15) (2002) 1392–1401.
[8] J. Spencer, P. Winkler, Three thresholds for a liar, Combinatorics, Probability and Computing 1 (1) (1992) 81–93.