Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Simulating one-reversal multicounter machines by partially blind multihead finite automata

Alan Deckelbaum [*]

Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

ARTICLE INFO

Article history:
Received 14 March 2007
Received in revised form 17 February 2008
Accepted 25 August 2008
Communicated by O.H. Ibarra

Keywords:
Blind head
Counter machine
Finite automaton

ABSTRACT

This work is concerned with simulating nondeterministic one-reversal multicounter automata (NCMs) by nondeterministic partially blind multihead finite automata (NFAs). We show that any one-reversal NCM with k counters can be simulated by a partially blind NFA with k blind heads. This provides a nearly complete categorization of the computational power of partially blind automata, showing that the power of a $(k+1)$-NFA lies between that of a k-NCM and a $(k+1)$-NCM.

© 2008 Elsevier B.V. All rights reserved.

1. Introduction

Ibarra and Ravikumar [2] introduced the partially blind NFA, a nondeterministic one-way multihead finite automaton where only a single head can differentiate between the symbols on the input tape. A partially blind k-NFA contains k total heads, of which $k-1$ are blind. These blind heads can only detect the difference between the endmarker on the input tape and the rest of the input. We will require that all blind heads as well as the read head must reach the end of the input before the machine can accept.

A nondeterministic multicounter machine is a finite automaton with the added power of being able to modify the values in a fixed number of counters. A k-NCM is such a machine containing k such counters. The machine can test whether or not a specified counter is at 0. Baker and Book [1], Ibarra and Ravikumar [2], and Ibarra [3] have studied the effect of limiting the number of times each counter is able to change between increasing and decreasing mode. In this paper, we deal with one-reversal NCM's: the machine may increment or decrement the value of each counter, but once a counter has been decremented, that counter may no longer be incremented. We will require that all counters must have value 0 and that the read head must be at the end of the input in order for an NCM to accept.

In [2] it was shown that a partially blind $(k+1)$-NFA can be simulated by a one-reversal $(k+1)$-NCM. Furthermore, they proved that every k-NCM can be simulated by a partially blind $(2k+1)$-NFA. Here we demonstrate an improvement on the latter simulation, showing that every k-NCM can be simulated by a partially blind $(k+1)$-NFA (an NFA with k blind heads).

2. Preliminaries

We will concern ourselves with partially blind NFAs where all input is augmented by an endmarker. Ibarra and Ravikumar [2] showed that adding an endmarker to the input does not change the computational power of the machine.

We will now introduce a few definitions concerning NCMs. A *state transition* of a nondeterministic one-reversal counter machine N is a command that takes the machine from one of the states of N to another state, and may additionally either

[*] Tel.: +1 484 431 6616; fax: +1 610 649 6422.
E-mail address: deckel@mit.edu.

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.08.027

move the read head to the right or instead may increment and/or decrement the values of some of the counters by 1. A transition command may specify the particular tape symbol that must currently be under the read head in order for the command to be executed. We assume that any transition that moves the read head does not change the value of any counter. (It is clear that a k-NCM can be made to comply with this restriction.)

A *counter-path* of N is a sequence of state transitions that N can follow without moving the read head. Thus, all of these transitions can be followed with the read head on top of a single input symbol. Furthermore, no counter-path may violate the one-reversal restriction of any counter. Following a counter-path results in changing the values of several of the counters of N without moving the read head.

A *counter-cycle* of a one-reversal k-NCM N is a counter-path p of state transitions of N such that the first state in p is the same as the last state of p and no other state repetitions occur. We furthermore require that a counter-cycle may not have a transition that increments a counter and another transition that decrements the same counter. We observe that N can follow a counter-cycle arbitrarily many times without violating the one-reversal rule, assuming that the cycle does not increment any counter which was decremented before entering the cycle. The *length* of a counter-cycle is the number of transitions in the cycle, and is equal to the number of distinct states in the cycle.

Lemma 1. *Given a one-reversal k-NCM N with q states, there are finitely many distinct counter-cycles, and each counter-cycle has length less than or equal to q.*

Proof. The number of transitions in each counter-cycle is less than or equal to q, since each counter-cycle has no repeated states other than the first and last. Since there are only finitely many ways to select a cycle of less than or equal to q transitions from a finite set of transitions, N contains only finitely many counter-cycles. □

In our simulation, our partially blind NFA will only be able to simulate up to some fixed number of moves of N's counters without moving the read head. The computation branch of N that we are simulating, however, has no such restriction. Therefore, the following lemma will be useful in separating long counter-paths of an NCM into a small counter-path and a collection of counter-cycles. In our final simulation, we may defer simulating some of these counter-cycles until another time, when our partially blind NFA is once again able to simulate the counter adjustments.

Lemma 2. *Given a one-reversal k-NCM N, there is a constant b such that for any counter-path s of state transitions, there is a counter-path s' and a collection of counter-cycles $C_1, \ldots, C_n$ (not necessarily distinct) such that:*

(1) *s' has length at most b.*
(2) *The start and end states of s are the same as the start and end states of s'.*
(3) *Each C_i contains some state that appears in s'.*
(4) *For each transition t_i, the number of times that t_i appears in s is equal to the number of times t_i appears in s' plus the total number of times t_i appears in all of the $C_1, \ldots, C_n$.*

(Notice that setting $b = q + 1$ in Lemma 2 does not suffice. Examples can be constructed where s' must visit a set of states many times in order to have at least one state in common with each C_i.)

Proof. We construct s' from s by removing repeated counter-cycles. (The order in which we will remove the counter-cycles is irrelevant for the correctness of this proof; any order of removals will suffice.) If any counter-cycle appears two or more times in s, we remove all but the first occurrence of the counter-cycle. For example, the sequence of transitions $q_1 \xrightarrow{t_1} q_2 \xrightarrow{t_2} q_3 \xrightarrow{t_3} q_1 \xrightarrow{t_4} q_5$ would be shortened to $q_1 \xrightarrow{t_4} q_5$, if $q_1 \xrightarrow{t_1} q_2 \xrightarrow{t_2} q_3 \xrightarrow{t_3} q_1$ appeared previously in s'. Thus, for example, we would shorten

$$q_1 \xrightarrow{t_1} q_2 \xrightarrow{t_2} q_3 \xrightarrow{t_3} q_1 \xrightarrow{t_4} q_6 \xrightarrow{t_5} q_1 \xrightarrow{t_1} q_2 \xrightarrow{t_2} q_3 \xrightarrow{t_3} q_1 \xrightarrow{t_4} q_5$$

to

$$q_1 \xrightarrow{t_1} q_2 \xrightarrow{t_2} q_3 \xrightarrow{t_3} q_1 \xrightarrow{t_4} q_6 \xrightarrow{t_5} q_1 \xrightarrow{t_4} q_5.$$

We repeat this process of removing the second occurrence of repeated counter-cycles until no additional counter-cycles are removed. (It is possible that the counter-path resulting from this procedure may not be uniquely-determined, depending on the order in which cycles were removed, but any s' constructed by this method will suffice.)

We notice that any transition that appears in s must also appear in s', since the final occurrence of a transition is never removed in this construction. Therefore, s' contains at least one state from each of the removed counter-cycles. We see furthermore that the construction preserves connectivity, and thus s' is a counter-path from the first state of s to the last state of s. Therefore, condition 2 holds. It is also clear that conditions 3 and 4 are satisfied, by letting the C_i be the counter-cycles that were removed from s in this construction.

It only remains to verify condition 1. We let t be the number of distinct state-transitions in N, and let u be the number of counter-cycles of N (this value is finite by Lemma 1). We claim that the number of transitions in s' is less than $b = (t + 1)(u + 1 + k)$. Assume on the contrary that s' contains at least $(t + 1)(u + 1 + k)$ transitions. Separate the first $(t + 1)(u + 1 + k)$ transitions of s' into blocks of length $t + 1$. Each block must contain at least one repeated state, by the pigeonhole principle. As each counter is one-reversal, at least $u + 1$ of these blocks do not reverse the direction of any counter, and thus each of these $u + 1$ blocks must contain some counter-cycle. As there are only u distinct counter-cycles, there is some counter-cycle that occurs twice in the first $(t+1)(u+1+k)$ transitions of s', yielding the desired contradiction.

Therefore, letting $b = (t + 1)(u + 1 + k)$, we see that the length of s' must be less than b. Since s' was constructed from s by removing finitely many counter-cycles, the total transitions in s are equal to the transitions in s' plus the transitions in finitely many removed counter-cycles, where each of these C_i has at least one state in s'. □

3. Simulating a k-NCM by a $k + 1$-NFA

We now proceed to the main theorem of this paper.

Theorem. *Given a one-reversal k-NCM N, there exists a partially blind k + 1-NFA P, such that* $L(N) = L(P)$.

3.1. The simulation procedure

Recall that N accepts an input only if all of the counters are at zero, the read head is at the endmarker, and the machine enters an accept state. Let q be the number of states in N. Ibarra and Ravikumar [2] observed from the proof of Theorem 5 in [1] that there is a constant c such that if N accepts an input of length n, there is an accepting branch of the computation that does so in less than cn steps.

Each blind head of P will simulate one of the counters of N. In this simulation, the zero position of the counter is represented by the cell on the tape where the read head is currently located. Each subsequent cell to the right of the read head represents a counter value of $+3cbq$, and each cell to the left of the read head represents a counter value of $-3cbq$, where b is the constant from Lemma 2. The value of each counter modulo $3cbq$ will be stored in the finite state of P.

In the simulation, whenever the read head moves forward, all blind heads move forward one space as well (unless specifically directed otherwise, as explained later). Whenever a blind head's count increases enough for the count modulo $3cbq$ to reach $3cbq$ (and thus reset to 0), the blind head moves one additional space forward on the tape. If a blind head's count decreases to bring the count modulo $3cbq$ below 0, the blind head does not move forward the next time the read head moves. (In the simulation, the value of a blind head's corresponding counter will never change by more than $3cbq$ without moving the read head, and thus a blind head will never need to lag behind the read head or move forward past the read head by more than one additional tape cell each time the read head moves.)

3.1.1. Overview of the simulation procedure

Before continuing with the details of the procedure, we will give a brief overview of the manner in which P simulates N. The simulation is divided into three repeating stages. The value of each counter will be stored as described earlier: each cell in front of the read head will represent a counter value of $+3cbq$, and each cell behind the read head will represent a counter value of $-3cbq$.

Throughout the simulation, P's read head simulates the read head of N. In the first stage (the stage reached immediately after moving the read head), P nondeterministically guesses which state N will be in when N next moves the read head. Ideally, we would like P to be able to then simulate the counter-path that N takes between these two states, and adjust the blind heads appropriately. However, this counter-path might be of arbitrarily long length, yet our method of updating changes in counter values only allows for a fixed change in the value of each counter between moves of the read head. (Since the blind heads are one-directional, they can only lag behind the read head by a single additional tape cell each time the read head moves, and thus we cannot simulate a counter-path that decreases the value of any counter by more than $3cbq$.) A difficulty is that we do not have a bound on the size of any particular counter-path that N follows. We only can bound the overall number of transitions in N's entire computation as linear in the size of the input. We resolve this problem by breaking apart a long counter-path into a short counter-path and a collection of counter-cycles (as in Lemma 2), and simulating the excess counter-cycles separately.

We resolve the difficulty of arbitrarily long counter-paths by guessing at the beginning of the simulation a set S of counter-cycles that will be reachable at some point in N's computation. Whenever we wish to simulate the counter-path that N takes between moves of the read head, we always choose a path of length less than or equal to some fixed value, b. (Notice that all counter-paths of length b or less can be stored in state memory, as there are only finitely many such paths.) We will also check if any of the counter-cycles from S have at least one state in common with this counter-path, and could have been followed by N with the read head on the current input symbol and without violating the one-reversal rule of any counter. If so, we remember in state memory that this counter-path was reachable during the computation.

After simulating the short counter-path, and before moving the read head, P goes to the second stage of the simulation. In this stage, P simulates up to $2c$ (a constant, where the choice of this constant is explained in Section 3.2.2) counter-cycles from the nondeterministically chosen set S. (P is allowed to follow any cycle from this set, even if N could not legally follow the cycle with the read-head over the current tape symbol.) At the very end of the computation (when the read-head is over the endmarker), we ensure that all of the counter-cycles from S could indeed have been followed by N at some point in the computation. (If not, we reject.) By simulating excess counter-cycles separately from the short counter-paths, and only ensuring at the end that all of these cycles could indeed have been followed by N at some point, we restrict the maximum possible change in counter value of P's blind heads between the moves of its read head. Notice that due to the separate simulation of counter-cycles, it is possible that at some point in P's computation, P stores negative counter values, which are impossible in an NCM. However, it is only necessary that at the very end of the computation all of the simulated counter values in P coincide with the counter values of N, and the counter values may disagree at intermediate steps of the overall computation. In Section 3.2.1, we show how the existence of an accepting computation branch of P implies that by rearranging the counter-cycles, we can identify a branch of N's computation which accepts without violating the one-reversal rule and where all counters remain nonnegative at all states of the computation.

In the third stage, we simulate a transition that moves the read head, and then re-enter the first stage and repeat the above procedure.

3.1.2. *The specific simulation procedure*

Let N be a one-reversal k-NCM, q be the number of states of N, b be the constant from Lemma 2, and let c be the constant such that if N accepts an input of size n then there is a computation branch which accepts in at most cn steps. We construct a $(k+1)$-NFA, P, which operates according to the following protocol:

(1) Nondeterministically initialize S to be some subset of counter-cycles of N.
(2) Initialize $S' = \emptyset$. (This will be a set of counter-cycles of N.)
(3) Initialize $n_1, \ldots, n_k = 0$. (These are variables which will always have values between 0 and $3bcq - 1$.)
(4) Initialize $r_1, \ldots, r_k = 0$. (These are variables which will always have values either 0 or 1.)
(5) Initialize q_1 to be the start-state of N.
(6) While the read-head of P is reading a symbol σ:
- (a) Nondeterministically select a state q of N, and set $q_2 \leftarrow q$.
- (b) Nondeterministically select a counter-path p of length at most b state-transitions of N such that p begins at q_1, ends at q_2, and can be followed with the read-head over σ. If no such counter-path exists, reject. (It is allowed that p has length 0 in the case $q_1 = q_2$.)
- (c) Set $a_1, \ldots, a_k = 0$. (These are variables which will always have value either -1, 0, or 1.)
- (d) For each transition t in p, taken in order:
  - (i) Suppose that t is a transition from states q_a to q_b. For each counter-cycle C of N which contains q_a as a state and which can be followed with the read-head over σ, if $C \notin S'$, then nondeterministically choose whether or not to proceed with the following three steps:
    - (A) Insert C into the set S'.
    - (B) For $i = 1, \ldots, k$, if any transition in C decreases the value of counter i, then set $r_i = 1$.
    - (C) For $i = 1, \ldots, k$, if any transition in C increases the value of counter i, and if $r_i = 1$, then reject.
  - (ii) Update the values of $n_1, \ldots, n_k$, modulo $3bcq$, according to t. (If t would increment counter i by 1, then increment n_i by 1. If t would decrement counter i by 1, then decrement n_i by 1.) If t causes n_i to reach $3bcq$ (and hence go down to 0), then set $a_i \leftarrow a_i + 1$. If t causes n_i to go below 0 (and hence go up to $3bcq - 1$), then set $a_i \leftarrow a_i - 1$.
  - (iii) For $i = 1, \ldots, k$, if t decreases counter i, then set $r_i = 1$.
  - (iv) For $i = 1, \ldots, k$, if t increases counter i and if $r_i = 1$, then reject.
- (e) Nondeterministically set $A = \{C_1, \ldots, C_n\}$ to be a collection of at most $2c$ counter-cycles from the set S, with repetitions allowed.
- (f) For each counter-cycle C in A:
  - (i) For each transition t in C, update the values of $n_1, \ldots, n_k$ modulo $3bcq$ according to t, as before. If t causes n_i to reach $3bcq$ (and hence go down to 0), then set $a_i \leftarrow a_i + 1$. If t causes n_i to go below 0 (and hence go up to $3bcq - 1$), then set $a_i \leftarrow a_i - 1$. (Note that we do not require that t can be followed with the read-head over σ, and we do not update the r_i values.)
- (g) If σ is not the endmarker, then for $i = 1, \ldots, k$:
  - (i) If $a_i = 0$, move blind head i to the right one space. If this would cause the head to move further than the end-marker on the input, then reject.
  - (ii) If $a_i = 1$, move blind head i to the right two spaces and set $a_i = 0$. If this would cause the head to move further than the end-marker on the input, then reject.
  - (iii) If $a_i = -1$, do nothing. (Observe that the only possible values of a_i are 0 and ± 1, since the value of n_i can change by at most $b + 2cq \leq 3bcq$ between steps 6d and 6f.)
- (h) If σ is not the end-marker:
  - (i) Nondeterministically select a state q of N, and set $q_3 \leftarrow q$.
  - (ii) If there is a transition in N from q_2 to q_3 which can be followed with the read-head over σ and which moves the read-head to the right one space, then move the read-head to the right. (Recall that we have required that transitions which move the read-head do not also affect the counters.) If there is no such transition, then reject.
  - (iii) Set $q_1 \leftarrow q_3$.
- (i) If σ is the end-marker:
  - (i) For $i = 1, \ldots, n$, if $a_i = 1$ set $a_i = 0$ and move blind head i to the right one space.
  - (ii) If $S = S'$, $n_1, \ldots, n_k = 0$, $a_1, \ldots, a_k = 0$ all blind heads are on the end-marker, and q_2 is an accept state of N, then accept. Otherwise, reject.

We claim that P accepts exactly the same input strings that N accepts.

3.2. Correctness of the simulation

3.2.1. $x \in L(P) \Rightarrow x \in L(N)$

We will show that if P accepts an input x, then we can identify an accepting computation of N. We will focus on a particular accepting computation branch of P. (Any branch which accepts will suffice.) We now observe that when N computes on x, one branch of N's nondeterministic computation will match the transitions identified in the following description:

(1) Let $S = \{C_1, \ldots, C_m\}$ be the set of counter-cycles from step 1 of P's computation.
(2) Let d_j be the total number of times that counter-cycle C_j is simulated in step 6f of P's computation, with the total taken over the entire computation of P.
(3) Begin in the start state of N.
(4) While the read-head is over a symbol σ:
  (a) Let q_1 be the current state of N.
  (b) Let q_2 be the state chosen in step 6a of P's computation.
  (c) Let p be the path from q_1 to q_2 selected in step 6b of P's computation.
  (d) Begin following all of the transitions in p, in order. When we reach the particular transition t that caused counter-cycle C_j to be added into S' (in step 6diA of P's accepting computation), follow counter-cycle C_j a total of d_j times before following transition t.
  (e) After all of the transitions in p have been followed, if σ is not the end-symbol, follow the transition identified in step 6hi which moves the read-head, and set q_1 to be the resulting state.

We claim that the above description does indeed describe a computation branch of N. We must show that all of the transitions can be followed without violating the one-reversal restriction of any counter, and that no counter ever becomes negative. The fact that the transitions in step 4d and 4e do not violate the one-reversal rule of the counter is clear from the fact that P's protocol uses the variables $r_1, \ldots, r_k$ to keep track of whether a counter has ever been decremented. Thus, at the time when a transition is used in step 6dii of P's computation, or when a counter-cycle is added to S in step 6diA of P's computation, we only use a transition that increases counter i if $r_i = 0$, and we set $r_i = 1$ if the counter is in decreasing-mode.

Now that we have verified that all counters obey the one-reversal rule, we notice that at the end of the computation all of the counters have value 0, since in the end of P's computation we have all $n_i = 0$ and all blind-heads are at the endmarker, and the computation branch of N described above follows exactly those state-transitions which were simulated in P's accepting computation branch. Since we know that the one-reversal rule of each counter is obeyed in the computation of N described above, and that all counters have final value of 0 at the end of the computation branch, it is obvious that no counter stores a negative value at any point in the computation. Thus, the description above does indeed describe a computation branch of N on x which obeys the one-reversal rules and in which the final state is an accept state with all counters having value 0. We have therefore used the fact that P accepts x to identify a particular computation branch of N which accepts x, and hence $x \in L(P) \Rightarrow x \in L(N)$.

3.2.2. $x \in L(N) \Rightarrow x \in L(P)$

If N accepts x, we will show that there is a computation branch of P that accepts. By [1], there is some computation branch of N that accepts x in less than cn steps, where n is the length of x. Pick the subsequence of states of N $r_0, q_1, r_1, q_2, r_2, \ldots, q_n, r_n, q_{n+1}$ from this computation branch such that r_0 is the start state, and such that the read head moves to the right in each transition between q_j and r_j. (The transitions between q_j and r_j are the transitions that P will simulate in step 6hii.) Let t_j be this transition from q_j to r_j, and let s_j be the counter-path in the computation that N followed to get from r_j to q_{j+1}, as shown:

$$r_0 \xrightarrow{s_0} q_1 \xrightarrow{t_1} r_1 \xrightarrow{s_1} q_2 \xrightarrow{t_2} r_2 \rightarrow \cdots \xrightarrow{s_n} q_{n+1}.$$

By Lemma 2, there is a counter-path s'_j of length less than b from r_j to q_{j+1} such that the transitions in counter-path s_j are equal to the transitions in s'_j plus the transitions in counter-cycles $C_1, \ldots, C_m$, each of which has at least one state in s'_j.

We now identify an accepting computation branch of P: in step 1 of P's computation, S should contain a counter-cycle of each type removed in the Lemma 2 constructions of the s''s. When P simulates the instance of N being in state q_j from the list above (i.e. when $q_2 = q_j$ at step 6hi), P should simulate transition t_j to get to state r_j in step 6hii. When a counter-path of length less than b is chosen between r_i and q_{i+1} in step 6b, this path should be s'_i.

When P simulates counter-cycles in steps 6e and 6f, the counter-cycles should be selected from S so that by the time the read head reaches half-way across the input, each counter-cycle has been simulated precisely the number of times it was removed in the construction of all of the s''s. (As N's computation has less than cn transitions, at most cn counter-cycles were followed in the entire computation. Since P can simulate $2c$ cycles between moves of the read head, it can simulate all of the necessary counter-cycles by the time the read head gets half-way across the input tape.)

It is clear that all of the counter-cycles in S can eventually be added to S' in step 6diA, as each of these counter-cycles shares a state with some s' and the transitions were indeed followed at some point in N's computation. Thus, at the time when the transitions were followed by N, they can be followed without violating the one-reversal rules of the counters.

(This fact is immediately obvious if we realize that the Lemma 2 constructions never remove the first instance of a state transition in any counter-path.)

Since P simulates all of the necessary counter-cycles from S before the read head passes the halfway point of the input tape, it is clear that no blind head will reach the end of the input before all of the necessary counter-cycles in S have been simulated the appropriate number of times, since a blind head will move at most one cell in front of the read head each time the read head moves.

Since N accepts only if all counters are 0, and since every transition of N's accepting computation branch is simulated by P the same number of times as the transition was followed by N, it is clear that no blind head will ever reach the endmarker before the read head reaches the endmarker: when a blind head reaches the endmarker, the read head will be more than half-way across the input tape, and thus all of the necessary counter-cycles will already have been simulated. Thus, for each tape cell that the read head moves, the count in the blind head can decrease by no more than b. If the blind head is m cells ahead of the read head when it reaches the end, its associated count value must be at least $3bcqm$, and this count can decrease by no more than b for each of the m cells that the read head moves. The count decreases by no more than an additional b once the read head reaches the endmarker. As $(m+1)b < 3bcqm$, if a blind head reaches the endmarker before the read head it is impossible that the corresponding count can reach 0 by the time the read head reaches the end of the input. Since all of the simulated counters must be 0 when the read head reaches the end of the input, it is therefore the case that no blind head reaches the endmarker before the read head. Therefore, at the end of this computation branch of P, all of the n_i will be 0, all the blind-heads will be at the endmarker, S will equal S', and P will be simulating an accept state of N, and hence P will accept.

Therefore, $L(N) = L(P)$. □

This theorem immediately implies the following corollary, which answers an open question posed at the end of [2].

Corollary. *If a language L is recognized by a one-reversal* 1*-NCM, then L is also recognized by some partially blind* 2*-NFA.*

Proof. This is an immediate result of the above theorem when restricted to the case $k = 1$. □

4. Conclusion

It was shown in [2] that a partially blind $(k+1)$-NFA can be simulated by a $(k+1)$-NCM. Our result therefore provides a nearly complete categorization of the computational power of nondeterministic partially blind finite automata: any language recognized by a k-NCM is also recognized by some $(k+1)$-NFA, and any language recognized by a $(k+1)$-NFA is recognized by some $(k+1)$-NCM. The power of k blind heads therefore lies between the power of k and $k+1$ one-reversal counters.

Acknowledgments

I would like to thank Michael Sipser for his suggestions and guidance throughout all stages of the research process. I would also like to thank the anonymous reviewers for their comments which greatly improved the quality of the paper.

References

[1] B. Baker, R. Book, Reversal-bounded multipushdown machines, J. Comput. System Sci. 8 (1974) 315–332.
[2] O. Ibarra, B. Ravikumar, On partially blind multihead finite automata, Theoret. Comput. Sci. 356 (2006) 190–199.
[3] O. Ibarra, Reversal-bounded counter machines and their decision problems, J. ACM 25 (1) (1978).