DCFS 2008 (Draft)

Descriptional Complexity of Formal Systems
Deadline for submissions: April ~~7~~ 27, 2008(strict)
Final versions: June 25, 2008

Determinism without Determinization

Janusz Brzozowski[(A)] Nicolae Santean[(B)]

[(A)]David R. Cheriton School of Computer Science – University of Waterloo
Waterloo, ON – Canada N2L 3G1
brzozo@uwaterloo.ca

[(B)]Computer and Information Sciences – Indiana University South Bend
South Bend, IN 46634 – USA
nsantean@iusb.edu

Abstract. A nondeterministic semiautomaton $\mathcal{S}$ is predictable if there exists $k \geq 0$ such that, if $\mathcal{S}$ knows the current input a and the next k inputs, the transition under a is deterministic. Nondeterminism may occur only when the length of the unread input is $\leq k$. We develop a theory of predictable semiautomata. If a semiautomaton $\mathcal{S}$ with n states is k-predictable, but not $(k-1)$-predictable, then $k \leq (n^2-n)/2$, and this bound can be reached for a suitable input alphabet. We characterize k-predictable semiautomata. We introduce the predictor semiautomaton, based on a look-ahead semiautomaton. The predictor is essentially deterministic and simulates a nondeterministic semiautomaton by finding the set of states reachable by a word w, if it belongs to the language L of the semiautomaton (*i.e.*, if it defines a path from an initial state to some state), or by stopping as soon as it infers that $w \notin L$. Membership in L can be decided deterministically.

Keywords: automaton, bounds, delegator, look-ahead, nondeterminism, predictor, semiautomaton, simulation

1 Introduction

Nondeterministic finite automata (NFAs) are ubiquitous. They serve as models for nondeterministic processes, constitute design tools (arguably more convenient than deterministic ones), and are often inevitable. They also have drawbacks, such as increased simulation time and space, and inefficient minimization algorithms. Several attempts have been made to overcome the disadvantages of nondeterminism. NFAs have been used as models for service-oriented computing [1], and as tools for automated web service composition [4]. In both cases, it was imperative to enforce deterministic behavior without changing the structure of the model, *i.e.*, without determinization. For this purpose, "delegators" of NFAs were informally introduced in [4]. A delegator is an equivalent deterministic finite automaton (DFA) based on the transition graph of the NFA. It has a look-ahead buffer of a fixed length that permits it to determine which

This research was supported by the Natural Sciences and Engineering Research Council of Canada under grants no. OGP000871 and Fellowship No. PDF-32888-2006.

of several possible nondeterministic steps should be taken. Look-ahead delegation was studied systematically and in a more abstract framework in [8].

We do not solve the delegator problem, nor determinize NFAs. Instead, we provide a method in which a nondeterministic system can be used essentially deterministically. We use semiautomata, because nondeterminism involves transitions, rather than accepting states. We introduce "predictable" semiautomata, in which it is possible to replace a nondeterministic step by a deterministic one, with the aid of a bounded number of input letters from a look-ahead buffer. We compute the set of states reachable from the initial states of a semiautomaton by an input word, with as little nondeterminism as possible. We also treat nondeterminism as a local, rather than global, phenomenon. Thus our theory is substantially different from the work in [4, 8].

The remainder of the paper is organized as follows. Section 2 introduces predictable semiautomata. In Section 3 we show that, if an n-state semiautomaton is k-predictable, then $k \leq (n^2 - n)/2$. Section 4 describes the construction of an auxiliary "core semiautomaton". Semiautomata with look-ahead are defined in Section 5. Section 6 concludes the paper. Proofs that are omitted can be found in [3].

2 Predictable Semiautomata

For a set X, we denote its cardinality by $X\#$. If Σ is an alphabet, then Σ^+ and Σ^* denote the free semigroup and the free monoid, respectively, generated by Σ. The empty word is 1. For $k \geq 1$, let $\Sigma^{\leq k} = 1 \cup \Sigma \cup \cdots \cup \Sigma^k$. For $w \in \Sigma^*$, $|w|$ denotes the length of w. If $w = uv$, for some $u, v \in \Sigma^*$, then u is a prefix of w. A language L is prefix-closed if $uv \in L$ implies $u \in L$. If $u \in \Sigma^*$, $v \in \Sigma^+$, then uv is an extension of u.

A semiautomaton [5] $\mathcal{S} = (\Sigma, Q, P, E)$ consists of an alphabet Σ, a set Q of states, a set $P \subseteq Q$ of initial states, and a set E of edges of the form (q, a, r), where $q, r \in Q$ and $a \in \Sigma$. An edge (q, a, r) begins at q, ends at r, and has label a. It is also denoted as $q \xrightarrow{a} r$. A path π is a finite sequence $\pi = (q_0, a_1, q_1)(q_1, a_2, q_2) \cdots (q_{k-1}, a_k, q_k)$ of consecutive edges, $k > 0$ being its length, q_0, its beginning, q_k, its end, and word $w = a_1 \cdots a_k$, its label. We also write $q_0 \xrightarrow{w} q_k$ for π. Each state q has a null path 1_q from q to q with label 1. If $T \subseteq Q$ and $w \in \Sigma^*$, then $Tw = \{q \in Q \mid t \xrightarrow{w} q, \text{ for some } t \in T\}$. If $T = \{t\}$, we write tw for Tw; if $Tw = \{q\}$, we write $Tw = q$. A state q of a semiautomaton $\mathcal{S}$ is accessible if there exists $p \in P, w \in \Sigma^*$ such that there is a path $p \xrightarrow{w} q$. A semiautomaton is accessible if all of its states are accessible.

The language $|\mathcal{S}|$ of a semiautomaton $\mathcal{S}$ is the set of all labels of paths from initial states of $\mathcal{S}$, *i.e.*, $|\mathcal{S}| = \{w \in \Sigma^* \mid Pw \neq \emptyset\}$, and $|\mathcal{S}|$ is prefix-closed. Semiautomaton $\mathcal{S}$ is complete if $P \neq \emptyset$ and, for every $q \in Q$ and $a \in \Sigma$, there is an edge $(q, a, r) \in E$, for some $r \in Q$. In a complete semiautomaton $\mathcal{S}$, $qw \neq \emptyset$, for all $q \in Q, w \in \Sigma^*$, and $|\mathcal{S}| = \Sigma^*$. A semiautomaton $\mathcal{S}$ is deterministic if it has at most one initial state, and for every $q \in Q$, $a \in \Sigma$, there is at most one edge (q, a, r). If $\mathcal{S}$ is deterministic and has initial state p, we write $\mathcal{S} = (\Sigma, Q, p, E)$. If q is a state of $\mathcal{S} = (\Sigma, Q, P, E)$, the language of q is $R_q = \{w \in \Sigma^* \mid qw \neq \emptyset\}$. If $\mathcal{S}$ is complete, $R_q = \Sigma^*$, for all $q \in Q$. The language of a set $T \subseteq Q$ is $R_T = \bigcup_{t \in T} R_t$; thus $R_P = |\mathcal{S}|$.

We restrict our attention to finite semiautomata. Let $\mathcal{S} = (\Sigma, Q, P, E)$ be a semiautomaton. If $q \in Q$, $a \in \Sigma$, then a fork (with origin q and input a) is the set $\langle q, a\rangle = \{(q, a, r_1), \ldots, (q, a, r_h)\}$ of all the edges from q labeled a. The fork set of $\langle q, a\rangle$ is $\langle\langle q, a\rangle\rangle = \{r_1, \ldots, r_h\}$. We assume that $h > 0$. Note, however, that forks with single edges are permitted; they are called deterministic transitions. Allowing such forks has the advantage that a semiautomaton consists only of a set of initial states and forks. A set $T \subseteq Q$ is critical if either $T = P$ or $T = \langle\langle q, a\rangle\rangle$, for a fork $\langle q, a\rangle$ in $\mathcal{S}$.

Definition 1. Let $\mathcal{S} = (\Sigma, Q, P, E)$ be a semiautomaton, and $k \geq 0$, an integer. A set $T \subseteq Q$ is k-predictable if any two distinct states $s, t \in T$ satisfy $R_s \cap R_t \cap \Sigma^k = \emptyset$. Also $\mathcal{S}$ is k-predictable if every critical set of $\mathcal{S}$ is k-predictable, and $\mathcal{S}$ is predictable if it is k-predictable for some k.

One verifies that a semiautomaton $\mathcal{S}$ is k-predictable if and only if it is "deterministic $(k+1)$-lookahead" [7]. A set is 0-predictable if and only if it consists of a single state. Thus, $\mathcal{S}$ is 0-predictable if and only if it is deterministic, *i.e.*, its critical sets are singletons. A predictable semiautomaton is either deterministic or incomplete, because if $\mathcal{S}$ is complete, $R_s = R_t = \Sigma^*$ and $R_s \cap R_t = \Sigma^*$, for all $s, t \in Q$. If a set is k-predictable, then it is k'-predictable for all $k' > k$, since R_s and R_t are prefix-closed.

Example 2. In Fig. 1 (a), fork $\langle p, a\rangle$ is a deterministic transition, and fork set $\langle\langle q, a\rangle\rangle = \{q, r\}$ is 1-predictable. Fork set $\{q, r\}$ in Fig. 1 (b) is 1-predictable, because there are no words of length 1 in R_q or R_r. Fork set $\{p, q\}$ in Fig. 1 (c) is not k-predictable for any $k \geq 0$, because $a^k \in R_p \cap R_q \cap \Sigma^k$ for all k.

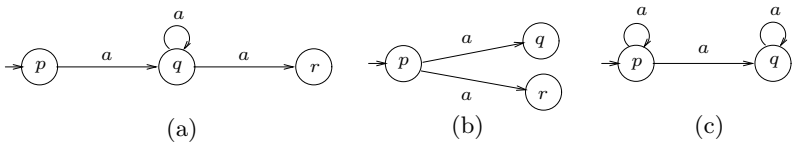

Figure 1. Illustrating predictability.

Below we classify a word w in the language R_T of a set T of states as a "t-selector in T", if it originates in t (is the label of a path from t) and in no other state of T, or as a "t-nonselector in T", if it originates in t and in at least one other state of T.

Definition 3. If $\mathcal{S} = (\Sigma, Q, P, E)$ is a semiautomaton, $T = \{t_1, \ldots, t_h\} \subseteq Q$, then a word $w \in \Sigma^*$ is a t_i-selector in T if $w \in \sigma(t_i, T) = \left(R_{t_i} \setminus \bigcup_{j \in \{1,\ldots,h\}, j \neq i} R_{t_j}\right)$. Word w is a t_i-nonselector in T if $w \in \overline{\sigma}(t_i, T) = R_{t_i} \setminus \sigma(t_i, T)$. Word w is a selector in T if it is a t_i-selector in T for some t_i, *i.e.*, if $w \in \sigma(T) = \bigcup_{i=1}^{h} \sigma(t_i, T)$. Word w is a nonselector in T if it is a t_i-nonselector in T for some t_i, *i.e.*, if $w \in \overline{\sigma}(T) = \bigcup_{i=1}^{h} \overline{\sigma}(t_i, T) = R_T \setminus \sigma(T) = \bigcup_{1 \leq i \neq j \leq h} (R_{t_i} \cap R_{t_j})$. A selector w is minimal if no prefix of w is a selector. A t_i-nonselector u is maximal if no extension of u is in R_{t_i}.

Theorem 4. *Let $\mathcal{S} = (\Sigma, Q, P, E)$ be a semiautomaton and $T \subseteq Q$. The following are equivalent:* **(1)** *T is predictable.* **(2)** *There exists $k \geq 0$ such that $\sigma(t, T) \supseteq R_t \cap \Sigma^k$, for all $t \in T$.* **(3)** *There exists $k \geq 0$ such that $\sigma(T) \supseteq R_T \cap \Sigma^k$.* **(4)** *$\overline{\sigma}(T)$ is finite.*

3 Predictability Bounds

We derive a tight upper bound on the smallest integer k for which a semiautomaton is k-predictable, in terms of the number of its states. This will then be a bound on the size of a look-ahead buffer used to reduce nondeterminism.

Lemma 5. *Let $\mathcal{S} = (\Sigma, Q, P, E)$ be a predictable semiautomaton, with $Q = \{1, \ldots, n\}$, and let r_1, s_1 be two distinct states of a critical set in $\mathcal{S}$. If $w = a_1 \cdots a_{m-1}$ is a longest word in $R_{r_1} \cap R_{s_1}$, let $\pi_1 = (r_1, a_1, r_2) \cdots (r_{m-1}, a_{m-1}, r_m)$ and $\pi_2 = (s_1, a_1, s_2) \cdots (s_{m-1}, a_{m-1}, s_m)$ be two paths spelling w, originating from r_1 and s_1, respectively. Then the sequence $\mathcal{L} = (r_1, s_1), \ldots, (r_m, s_m)$ of ordered pairs of states encountered by π_1 and π_2 must satisfy for all $i, j \in \{1, \ldots, m\}, i \neq j$,* **(1)** *either $r_i \neq r_j$ or $s_i \neq s_j$,* **(2)** *either $r_i \neq s_j$ or $r_j \neq s_i$,* **(3)** *if $r_i = s_i$, then $r_j \neq r_i$ and $s_j \neq r_i$.*

Lemma 6. *Let $n > 0$, and let $\mathcal{L} = (r_1, s_1), \ldots, (r_m, s_m)$ be a sequence of ordered pairs of elements from $\{1, \ldots, n\}$. If $\mathcal{L}$ satisfies $r_1 \neq s_1$ and Conditions* **(1)**–**(3)** *of Lemma 5, then $m \leq (n^2 - n)/2$ and the bound is sharp.*

Proof: We first show the bound can be reached. Condition **(1)** is satisfied by the sequence $\mathcal{L} = (1,1), \ldots, (1,n), (2,1), \ldots, (2,n), \ldots, (n,1), \ldots, (n,n)$, which has length n^2. If we remove pairs (i,i), for all $1 \leq i \leq n$, we have a sequence of length $n^2 - n$, in which $r_1 \neq s_1$, and which satisfies **(3)** as well. Finally, for all $i \neq j$, remove either (i,j) or (j,i), but not both. Now the sequence also satisfies **(2)** and has length $(n^2 - n)/2$.

Next, we proceed by induction on n. If $n = 1$, then only the empty sequence satisfies all the conditions. Here $m = 0 = (n^2 - n)/2$. If $n = 2$, only the empty sequence, $(1,2)$ and $(2,1)$ satisfy the conditions. Thus $m \leq 1 = (n^2 - n)/2$.

For any $n > 0$, let $M(n)$ be the length of a longest sequence of pairs of elements from $\{1, \ldots, n\}$ satisfying all the conditions. Assume that $M(n-1) \leq [(n-1)^2 - (n-1)]/2$, for some $(n-1) \geq 2$. Let $\mathcal{L}$ be a sequence with $M(n)$ pairs satisfying all the conditions, and assume that $M(n) > (n^2 - n)/2$. If $M(n) > (n^2 - n)/2$ and $n \geq 3$, then $M(n) > n$. Thus $\mathcal{L}$ contains at least $n+1$ pairs. There are at most $2n-1$ pairs involving n, namely the pairs of the form (n,i) and (i,n). However, if both (n,i) and (i,n) appear in $\mathcal{L}$, then **(2)** is violated. Hence there are at most n pairs involving n, and at least one pair (i,j) not involving n. Without loss of generality we may assume that the first pair of $\mathcal{L}$ does not contain n, for if it did, we could interchange it with the pair (i,j).

Let $\mathcal{L}'$ be the sequence with m' pairs obtained from $\mathcal{L}$ by removing all the pairs containing n. Then $\mathcal{L}'$ satisfies all the conditions as well, and its elements are from the set $\{1, \ldots, n-1\}$. By the induction hypothesis, $m' \leq M(n-1) \leq [(n-1)^2 - (n-1)]/2 = (n^2 - n)/2 - (n-1)$. In addition to the pairs of $\mathcal{L}'$, $\mathcal{L}$ contains pairs from the set $\{(1,n), (2,n), \ldots, (n,n), (n,1), (n,2), \ldots, (n,n-1)\}$. If $\mathcal{L}$ contains (n,n), then it cannot contain any other pair involving n, for this would violate **(3)**. Hence,

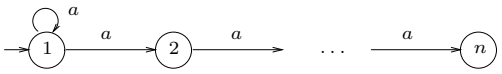

Figure 2. A n-state unary semiautomaton which is $(n-1)$-predictable.

$M(n) = m' + 1 \leq (n^2 - n)/2 - (n-2) < (n^2 - n)/2$, which contradicts our assumption. If $\mathcal{L}$ does not contain (n, n), it contains at most $(n-1)$ pairs involving n. Now $M(n) \leq m' + (n-1) \leq (n^2 - n)/2$, which is again a contradiction. Thus, $M(n) \leq (n^2 - n)/2$ and the induction step goes through. □

Theorem 7. *Let $\mathcal{S} = (\Sigma, Q, P, E)$ be a semiautomaton, $n = Q\#$, and $k \geq 0$, the smallest integer for which $\mathcal{S}$ is k-predictable. If $\Sigma\# = 1$, then $k \leq n - 1$, and the bound is reachable. If $\Sigma\# > 1$, then $k \leq (n^2 - n)/2$, and the bound is reachable for a suitable Σ.*

Proof: If $\mathcal{S}$ is deterministic, then $k = 0$, and the bound is trivially satisfied. If $\mathcal{S}$ is not deterministic, there must be at least one critical set $T = \{t_1, \ldots, t_h\}$, $h \geq 2$, which is k-predictable.

Case 1: $\Sigma\# = 1$. We omit this proof, since it is straightforward. The semiautomaton in Fig. 2 has n states and is $(n-1)$-predictable; thus the bound is reached when $|\mathcal{S}|$ is infinite. If we remove the loop and make states 1 and 2 initial, we have an example in which the bound is reached when $|\mathcal{S}|$ is finite.

Case 2: $\Sigma\# > 1$. Let r_1, s_1 be two distinct states of T. By Lemma 5, every sequence $\mathcal{L} = (r_1, s_1), \ldots, (r_m, s_m)$ must satisfy all the conditions of the lemma, and by Lemma 6, the length of a longest word $w \in R_{r_1} \cap R_{s_1}$ is $|w| = m - 1 \leq (n^2 - n)/2 - 1$. This implies that, if T is k-predictable, then necessarily $k \leq |w| + 1 \leq (n^2 - n)/2$. Since this holds for all critical sets, it holds for $\mathcal{S}$.

Next we prove that this bound is achievable for a suitable alphabet. Let $n \geq 1$, and let $\mathcal{L} = (r_1, s_1), \ldots, (r_k, s_k)$ be a sequence satisfying the conditions of Lemma 6, with $k = (n^2 - n)/2$. Consider the semiautomaton $\mathcal{S} = (\Sigma, Q, P, E)$, with $\Sigma = \{a_1, \ldots, a_k\}$, $Q = \{1, \ldots, n\}$, $P = \{r_1\}$ and $E = E_1 \cup E_r \cup E_s$, where $E_1 = \{(r_1, a_1, r_1), (r_1, a_1, s_1)\}$, $E_r = \{(r_i, a_{i+1}, r_{i+1}) \mid 1 \leq i < k\}$, and $E_s = \{(s_i, a_{i+1}, s_{i+1}) \mid 1 \leq i < k\}$. We show that $\mathcal{S}$ is k-predictable, but not $(k-1)$-predictable.

Observe that $E_1 = \langle r_1, a_1 \rangle$ is a fork in $\mathcal{S}$ and its fork set $\langle\langle r_1, a_1 \rangle\rangle = \{r_1, s_1\}$ is critical. Consider $w = a_2 \ldots a_k$; clearly, $w \in R_{r_1} \cap R_{s_1}$, implying that $\mathcal{S}$ is not $(k-1)$-predictable, since $|w| = k - 1$. Also $R_{r_k} \cap R_{s_k} = \emptyset$; for if both (r_k, a_j, r) and (s_k, a_j, s) were in E for some $r, s \in Q$ and $j \in \{1, \ldots, m\}$, then either $(r_k, s_k) = (r_{j-1}, s_{j-1})$ for $j > 1$, or $(r_k, s_k) = (r_1, s_1)$ for $j = 1$. In both cases, this would violate **(1)** of Lemma 5. Thus w is a longest word in $R_{r_1} \cap R_{s_1}$, implying that $\langle\langle r_1, a_1 \rangle\rangle$ is k-predictable.

Since $P\# = 1$, P is 0-predictable. If there exists a fork other than $\langle r_1, a_1 \rangle$ in $\mathcal{S}$, then there must be a pair (r_i, s_i) in $\mathcal{L}$ with $r_i = s_i$, since only such states have outgoing edges with a same label, by the construction of $\mathcal{S}$. By the argument in the proof of Lemma 6, $\mathcal{L}$ cannot be of maximal length. Hence there are no other forks, and $\mathcal{S}$ is k-predictable, where $k = (n^2 - n)/2$ is the smallest such integer. □

In [3] we provide a family of n-state semiautomata over a binary alphabet, that are $n(n+1)/4$-predictable. Although we don't know yet whether the bound in Theorem 7 can be reached for a binary or a larger fixed alphabet, this example shows that the alphabet size doesn't significantly impact the worst-case lower bound for predictability.

Testing for predictability can be done in polynomial time. Note first that we can test for k-predictability by using a "nondeterministic direct product" similar to the "square" of a semiautomaton in [2] (also used in [6] under the name "state-pair graph"). By Theorem 7, $\mathcal{S}$ is predictable if and only if it is $d = (n^2 - n)/2$-predictable. Thus, to test for predictability, it suffices to test for d-predictability. If we want to find the precise k for which $\mathcal{S}$ is k-predictable, but not $(k-1)$-predictable, we can test for $(d-1)$-predictability, $(d-2)$-predictability, *etc.*, until we find the minimum k.

4 Core semiautomata

We now introduce deterministic "core semiautomata", whose definition involves subset constructions. These semiautomata are not needed for finding minimal selectors and maximal nonselectors; rather, they are of theoretical interest. They explain the nature and the role of selectors and nonselectors, and provide another test for predictability.

For $T = \{t_1, \ldots, t_h\} \subseteq Q$ in $\mathcal{S} = (\Sigma, Q, P, E)$, we need to find the intersections of the languages R_{t_i} to determine predictability. We could make the semiautomata $\mathcal{S}_i = (\Sigma, Q, \{t_i\}, E)$, $t_i \in T$, deterministic and find their direct product. Instead, we obtain a deterministic direct product by using the subset construction in each step of the direct product construction. For a set Q, let 2^Q be the set of all subsets of Q. The direct product of h copies of 2^Q is denoted $(2^Q)^h$.

Definition 8. Suppose $\mathcal{S} = (\Sigma, Q, P, E)$ is a semiautomaton and $T = \{t_1, \ldots, t_h\} \subseteq Q$. Define the deterministic semiautomaton $\widehat{\mathcal{D}}(T) = (\Sigma, (2^Q)^h, \gamma_0, \widehat{E}_{\mathcal{D}})$, where $\gamma_0 = (\{t_1\}, \ldots, \{t_h\})$, and, for every h-tuple $(S_1, \ldots, S_h)$ of sets of states of $\mathcal{S}$ and every $a \in \Sigma$, there is an edge $((S_1, \ldots, S_h), a, (S_1a, \ldots, S_ha)) \in \widehat{E}_{\mathcal{D}}$, where S_ia is the set of successor states of the set S_i under input a in the semiautomaton $\mathcal{S}$. The product semiautomaton for T is the accessible subsemiautomaton of $\widehat{\mathcal{D}}(T)$ denoted by $\mathcal{D}(T) = (\Sigma, \Gamma, \gamma_0, E_{\mathcal{D}})$. Note that $\mathcal{D}(T)$ is complete.

Our next proposition characterizes the selectors and nonselectors in a set T of states of a semiautomaton $\mathcal{S}$ in terms of the types of states reached by these words in the product automaton $\mathcal{D}(T)$. Let $\gamma_\emptyset = (\emptyset, \ldots, \emptyset) \in (2^Q)^h$.

A state is t_i-singular if only its ith component is nonempty; it is t_i-plural if at least two of its components, including the ith, are nonempty. A state γ_i is t_i-primary if there exists a path $(\gamma_0, a_1, \gamma_1) \cdots (\gamma_{i-1}, a_i, \gamma_i)$, where $\gamma_0, \ldots, \gamma_{i-1}$ are t_i-plural and γ_i is t_i-singular. A state γ is t_i-ultimate if it is t_i-plural and the ith component of γa is empty, for each $a \in \Sigma$. A state is singular, plural, primary, or ultimate, if it is t_i-singular, t_i-plural, t_i-primary, or t_i-ultimate for some $t_i \in T$, respectively. Let Γ_{pl} (Γ_{pr}) be the set of all plural (primary) states of Γ. A state is cyclic if it appears in a cycle.

A word w is singular (primary) if $\gamma_0 w$ is singular (primary); $w = a_1 \cdots a_i$ is nullary if its path is $(\gamma_0, a_1, \gamma_1) \cdots (\gamma_{i-1}, a_i, \gamma_i)$, where $\gamma_0, \ldots, \gamma_{i-1}$ are plural and $\gamma_i = \gamma_\emptyset$.

Proposition 9. *Let $\mathcal{D}(T)$ be the product semiautomaton of a set T in $\mathcal{S}$. We have:* **(1)** *w is a t_i-selector in T if and only if $\gamma_0 w$ is t_i-singular.* **(2)** *w is a minimal t_i-selector if and only if $\gamma_0 w$ is t_i-primary.* **(3)** *w is a t_i-nonselector if and only if $\gamma_0 w$ is t_i-plural.* **(4)** *w is a maximal t_i-nonselector if and only if $\gamma_0 w$ is t_i-ultimate.*

Theorem 10. *A set $T = \{t_1, \ldots, t_h\} \subseteq Q$ of a semiautomaton $\mathcal{S}$ is predictable if and only if the product semiautomaton $\mathcal{D}(T)$ does not have cyclic plural states. If $\mathcal{S}$ is predictable and the length of a longest primary or nullary word in product semiautomata $\mathcal{D}(T)$ over all critical sets T is k, then $\mathcal{S}$ is k-predictable, but not $(k-1)$-predictable.*

Corollary 11. **(1)** *If T is a predictable set of a semiautomaton $\mathcal{S}$, then the set of minimal selectors in T is finite.* **(2)** *If T is k-predictable, then every word w in $R_T \cap \Sigma^k \Sigma^*$ has a prefix which is a minimal selector.*

Proposition 12. *Let $\mathcal{S} = (\Sigma, Q, P, E)$, and $T = \{t_1, \ldots, t_h\} \subseteq Q$. If T is k-predictable, then:* **(1)** *If t_i has no selectors, then $R_{t_i} \subseteq \Sigma^{\leq k-1}$.* **(2)** *Every $t_i \in T$ has either a minimal t_i-selector or a maximal t_i-nonselector.* **(3)** *Every nonselector is of length less than k.*

We now show that, for a predictable semiautomaton $\mathcal{S}$, a part of the product semiautomaton suffices for finding selectors and nonselectors.

Definition 13. The core semiautomaton of a product semiautomaton $\mathcal{D}(T) = (\Sigma, \Gamma, \gamma_0, E_\mathcal{D})$ is an incomplete deterministic semiautomaton $\mathcal{C}(T) = (\Sigma, \Omega, \gamma_0, E_\mathcal{C})$, where

$$\Omega = \begin{cases} \Gamma_{pl} \cup \Gamma_{pr} \cup \{\gamma_\emptyset\} & \text{if there is an edge from a plural state to } \gamma_\emptyset, \\ \Gamma_{pl} \cup \Gamma_{pr} & \text{otherwise,} \end{cases}$$

and $E_\mathcal{C}$ consists of edges that join a plural state to a plural state, primary state, or $\gamma_\emptyset$.

Since minimal selectors are primary words and maximal nonselectors lead to ultimate states, both can be found from the core semiautomaton. Thus, it is not necessary to construct the full product semiautomaton.

Example 14. Semiautomaton $\mathcal{S}$ of Fig. 3 (a) has one nontrivial critical set $T = \{p, q\}$, of fork $\langle p, a \rangle$. The product semiautomaton $\mathcal{D}(T)$ is shown in Fig. 3 (b), where we write p for $\{p\}$, pq for $\{p, q\}$, *etc.* Since no plural state is cyclic, $\mathcal{S}$ is predictable. The core semiautomaton $\mathcal{C}(T)$ is in Fig. 4. The length of a longest primary or nullary word is 3; hence the set $\{p, q\}$ and $\mathcal{S}$ are 3-predictable by Theorem 10. The minimal p-selectors are a, c, ba, bb and bcb, and the only minimal q-selector is bcc. The nonselectors are 1, b, and bc and none is maximal. There is one nullary word bca. In each deterministic transition in Fig. 3 (a), 1 is the minimal selector.

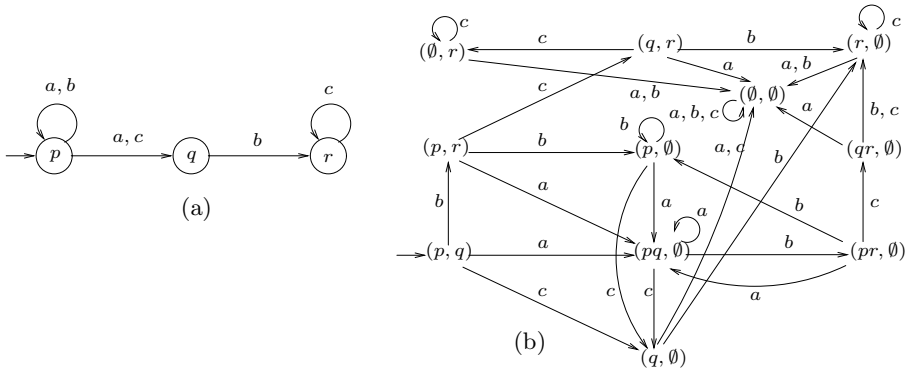

Figure 3. A predictable semiautomaton and its product semiautomaton.

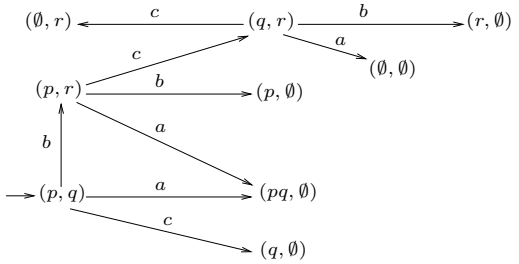

Figure 4. The core of the product semiautomaton in Fig. 3 (b).

The number of states of the core semiautomaton is $\leq (n+1)(2^n)^n$, since there are $n+1$ critical sets, each of which may have n states, and each state is a subset of the set of all n states. However, we can test in polynomial time whether a word is a minimal selector or a maximal nonselector, with the aid of nondeterministic direct products mentioned at the end of Section 3. Also, we can construct in polynomial time a semiautomaton that accepts the language of all nonselectors of a state or of a critical set. The number of selectors and nonselectors may be exponentially large, since there are m^k words of length k over an m-letter alphabet, and k may be as large as $n(n-1)/2$. It remains an open problem, however, to find a tight upper bound for the number of minimal selectors and maximal nonselectors.

When the semiautomaton has a small predictability bound, or only a few forks, or small critical sets, its predictor may be constructed efficiently. In examples we show minimal selectors in square brackets on the arrows leading to the selected states of a semiautomaton, and maximal nonselectors in "floor" brackets.

Example 15. In Fig. 5 there are two initial states q_1 and q_6 and two forks. Here $[ba]$ is a minimal selector of state q_3, and $\lfloor b \rfloor$ is a maximal nonselector of q_6. The core semiautomata are shown in Fig. 6. The semiautomaton is 2-predictable.

Minimal selectors permit us to determine precisely which state t must be chosen from a set T in a computation step. Sometimes it is also possible to reduce nondeterminism further by using maximal nonselectors. A maximal nonselector restricts the

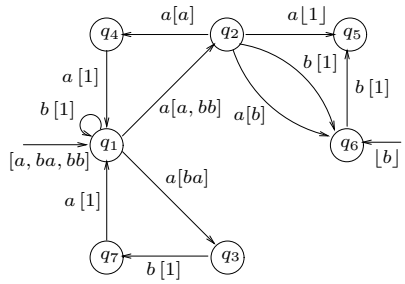

Figure 5. Illustrating selectors and nonselectors.

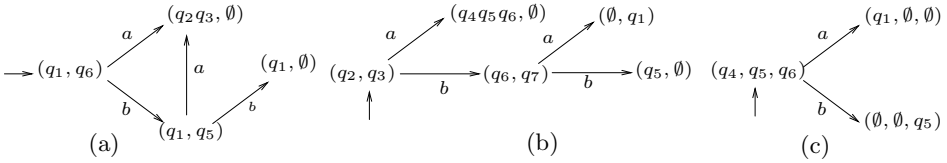

Figure 6. Core semiautomata for Example 15.

choice of states from T to a subset $S \subseteq T$ containing at least two states. In the worst case, when $S = T$, no reduction is possible. These ideas are applied in the next section.

5 Predictors

The concepts of the previous sections are now used to simulate a predictable semiautomaton almost deterministically. Let $\Sigma_s^{\leq k} = \{\lceil w \rceil \mid w \in \Sigma^{\leq k}\}$ be the set of all possible minimal selectors, and $\Sigma_n^{\leq k} = \{\lfloor w \rfloor \mid w \in \Sigma^{\leq k}\}$, the set of all possible maximal nonselectors. Let $\Sigma_{s\cup n}^{\leq k} = \Sigma_s^{\leq k} \cup \Sigma_n^{\leq k}$. Starting with a semiautomaton $\mathcal{S}$, we define a semiautomaton $\mathcal{P}$ that has $\Sigma \times \Sigma_{s\cup n}^{\leq k}$ as input alphabet; the new input consists of the current input letter a and up to k letters of look-ahead information.

Definition 16. Let $\mathcal{S} = (\Sigma, Q, P, E)$ be a k-predictable semiautomaton, $k \geq 0$. The predictor of $\mathcal{S}$ is a semiautomaton $\mathcal{P} = (\Sigma \times \Sigma_{s\cup n}^{\leq k}, Q, P, E_{\mathcal{P}})$, where we have: **(1)** The set of initial states is P. The sets of minimal p-selectors and maximal p-nonselectors in P are associated with each state $p \in P$. **(2)** If $\langle q, a\rangle$ is a fork, and (q, a, r), an edge in $\mathcal{S}$, then $(q, (a, \lceil u \rceil), r) \in E_{\mathcal{P}}$, if u is a minimal r-selector, and $(q, (a, \lfloor u \rfloor), r) \in E_{\mathcal{P}}$, if u is a maximal r-nonselector.

By Proposition 12 **(2)**, each state in P and in $\langle\langle q, a\rangle\rangle$, for each $q \in Q$, $a \in \Sigma$, has a minimal selector or a maximal nonselector. Thus it is easily verified that there is a one-to-one correspondence between predictable semiautomata and predictors.

We now describe how a predictor can be used to simulate a nondeterministic semiautomaton. The objective is to compute the set of states that can be reached by any $w \in |\mathcal{S}|$; if w is not in $|\mathcal{S}|$, then the set of states reached is empty. The simulation decides as soon as possible whether the input is accepted or rejected.

Lemma 17. *Let $\mathcal{S} = (\Sigma, Q, P, E)$ be a semiautomaton, and let $T \subseteq Q$ be k-predictable. If $w \in R_t$, for some state $t \in T$, then one of the following conditions holds:* **(1)** *A prefix u of w is a minimal t-selector.* **(2)** *$|w| < k$, and w is a prefix of a minimal t-selector.* **(3)** *$|w| < k$, and w is a prefix of a maximal t-nonselector.*

Definition 18. In a predictor $\mathcal{P}$, for a word $w \in \Sigma^*$ and $T \subseteq Q$, we define the handle of w in T. If a minimal selector x in T is a prefix of w, then x is the handle. If w is a prefix of a minimal selector or a maximal nonselector in T, then w itself is the handle. Otherwise, w does not have a handle. If w has a handle in T, the handle applies to a state $t \in T$ if either the handle is a minimal t-selector, or w itself is the handle, and is a prefix of a minimal t-selector or of a maximal t-nonselector.

Lemma 19. *Let $\mathcal{S} = (\Sigma, Q, P, E)$ be a semiautomaton, let $T \subseteq Q$ be k-predictable and let $w \in \Sigma^*$. If $w \in R_T$ and $t \in T$, then $w \in R_t$ if and only if the handle of w in T applies to t.*

Definition 20. Given a predictor $\mathcal{P}(\mathcal{S}) = \mathcal{P} = (\Sigma \times \Sigma^{\leq k}_{s \cup p}, Q, P, E_{\mathcal{P}})$ of a k-predictable $\mathcal{S}$ and an input word w, a prefix y of w yields a state $s \in Q$, written $y \Rightarrow s$, as follows:

- *Basis Step* (Step 0): $1 \Rightarrow s$ if $s \in P$ and the handle of w in P applies to s.
- *Induction Step* (Step $m+1$, $m \geq 0$): Assume now that $w = yaz$, for some $a \in \Sigma$, $y, z \in \Sigma^*$. Then $ya \Rightarrow s$ if $y \Rightarrow r$, for some $r \in Q$, $s \in \langle\langle r, a \rangle\rangle$ and the handle of z in $\langle\langle r, a \rangle\rangle$ applies to s.

Theorem 21. *Let $\mathcal{S} = (\Sigma, Q, P, E)$ be a k-predictable semiautomaton, $\mathcal{P} = (\Sigma \times \Sigma^{\leq k}_{s \cup n}, Q, P, E_{\mathcal{P}})$, its predictor, and $w = yv \in \Sigma^*$, an input word of $\mathcal{S}$. Then the predictor operation is correct in the following sense:*

1. *If $w \in |\mathcal{S}|$, then $y \Rightarrow q$ in $\mathcal{P}$ if and only if $q \in Py$ and $v \in R_q$ in $\mathcal{S}$.*
2. *The simulation stops with the remaining input v if and only if one of the following holds: (a) It is step 0 and w has no handle in P; this implies $w \notin |\mathcal{S}|$. (b) It is a step > 0 and $v = 1$; this implies that $w \in |\mathcal{S}|$. (c) It is a step > 0 and $v = az$, for some $a \in \Sigma$, $z \in \Sigma^*$ and there is no fork $\langle q, a \rangle$ in $\mathcal{S}$; then $w \notin |\mathcal{S}|$. (d) It is a step > 0 and $v = az$, for some $a \in \Sigma$, $z \in \Sigma^*$ there is a fork $\langle q, a \rangle$ in $\mathcal{S}$, but z has no handle in $\langle\langle q, a \rangle\rangle$; this implies $w \notin |\mathcal{S}|$.*

Proof: First, we show that, if $y \Rightarrow q$, then $q \in Py$ and $v \in R_q$, by induction on the length of the prefix y. If $1 \Rightarrow s$, then $s \in P$ by Definition 20. Thus $s \in P1 = P$. Since $w \in R_P$ by assumption, and the handle of w in P applies to s, we have $w \in R_s$, by Lemma 19, and the basis holds. Now assume that, for an arbitrary prefix y of $w = yaz$, if $y \Rightarrow r$, then $r \in Py$ and $v \in R_r$. Suppose that $ya \Rightarrow s$. Then $y \Rightarrow r$, for some $r \in Q$, $s \in \langle\langle r, a \rangle\rangle$, and the handle x of z in $\langle\langle r, a \rangle\rangle$ applies to s. By the induction hypothesis, $r \in Py$ and $az \in R_r$. Since $s \in \langle\langle r, a \rangle\rangle$, there is an edge $(r, a, s) \in E$; hence $s \in Pya$. Since $z \in R_{\langle\langle r,a \rangle\rangle}$ because $az \in R_r$, and the handle of z in $\langle\langle r, a \rangle\rangle$ applies to s, by Lemma 19 we have $z \in R_s$. Thus the induction goes through.

Second, assume that $q \in Py$ and $v \in R_q$; we show that $y \Rightarrow q$, by induction on the length of y. Consider first the factorization $w = yv = 1w$. By Lemma 19, if $p \in P$ and $w \in R_p$, then the handle of w in P applies to p. By Definition 20, the empty prefix 1 of w derives p. Now assume that, for an arbitrary prefix y of $w = yv$, if $r \in Py$ and $v \in R_r$, then $y \Rightarrow r$. Suppose now that $w = yaz$. Since $w \in R_P$, there exists $r \in Py$ such that $az \in R_r$, and hence a fork $\langle r, a\rangle$. Let s be any state in $T = \langle\langle r, a\rangle\rangle$ such that $z \in R_s$. By the induction hypothesis, $y \Rightarrow r$. Since $s \in T$ and $z \in R_s$, the handle of z in T applies to s by Lemma 19. Therefore $ya \Rightarrow s$, and the claim holds.

For (a), if w has no handle in P, then $1 \Rightarrow p$ is false for all $p \in P$, by Definition 20, and the simulation stops. Now if $w \in |\mathcal{S}|$, then $w \in R_P$, and hence $w \in R_p$, for some $p \in P$. Since also $p \in P1$, the first part of the theorem applies, and $1 \Rightarrow p$, which is a contradiction; thus $w \notin |\mathcal{S}|$.

For (b), if the simulation has consumed y, $y \Rightarrow q$, and $v = 1$, then $w = y$, and the entire input has been processed. Since v does not have the form az, the simulation stops. Since $w = y \Rightarrow q$, we have $w \in |\mathcal{S}|$.

For (c), assume that $w = yv = yaz$, the simulation has consumed y, $y \Rightarrow q$, but there is no fork $\langle q, a\rangle$ in $\mathcal{S}$. Clearly, the induction step cannot be carried out, and $az \notin R_q$. Suppose now that $w \in |\mathcal{S}|$. Since $y \Rightarrow q$, we have $q \in Py$ and $az \in R_q$, by the first part of the theorem. This is a contradiction; thus $w \notin |\mathcal{S}|$.

For (d), assume that $w = yv = yaz$, the simulation has consumed y, $y \Rightarrow q$, and z has no handle in $\langle\langle q, a\rangle\rangle$. The derivation stops because the induction step cannot be carried out. If $w \in |\mathcal{S}|$, since $y \Rightarrow q$, we know by the first part of the theorem that $q \in Py$ and $v \in R_q$. Also, there exists $r \in \langle\langle q, a\rangle\rangle$ such that (q, a, r) is an edge in $\mathcal{S}$ and $z \in R_r$. But now $r \in Pya$ and $z \in R_r$ implies that $ya \Rightarrow r$, again by the first part. Thus z must have a handle in $\langle\langle q, a\rangle\rangle$, which is a contradiction, and $w \notin |\mathcal{S}|$.

One verifies that, if none of the conditions (a)–(d) holds, then the derivation continues. This concludes the proof of the second claim. □

Corollary 22. *In a predictor $\mathcal{P}$, $w \in |\mathcal{S}|$ if and only if $w \Rightarrow q$, for some $q \in Q$.*

Example 23. Figure 5 without the minimal selectors and maximal nonselectors is a semiautomaton $\mathcal{S}$. With the minimal selectors and maximal nonselectors, it becomes a predictor. The incoming edge of q_1 is labeled with minimal selectors $[a]$, $[ba]$, and $[bb]$, and that of q_6, with maximal nonselector $\lfloor b \rfloor$. The edge (q_1, a, q_2) is replaced by edges $(q_1, (a, [a]), q_2)$ and $(q_1, (a, [bb]), q_2)$, *etc.* Suppose $w = aaababaab$. We write $a(z)$ for az to make it easier to identify the handle of z. Let q be the current state, and v, the remaining input. In Step 0, $v = w = aaababaab$, and the handle of w in $\{q_1, q_6\}$ is a. We have $1 \Rightarrow q_1$, since a applies only to q_1. The next seven steps are deterministic: $\overset{[a]}{\Rightarrow} q_1 \overset{a[a]}{\Rightarrow} q_2 \overset{a[a]}{\Rightarrow} q_4 \overset{a[1]}{\Rightarrow} q_1 \overset{b[1]}{\Rightarrow} q_1 \overset{a[ba]}{\Rightarrow} q_3 \overset{b[1]}{\Rightarrow} q_7 \overset{a[1]}{\Rightarrow} q_1$. In particular, if $q = q_1$, $v = a(aababaab)$, the handle of $aababaab$ in $\{q_2, q_3\}$ is a. Hence $a \Rightarrow q_2$, since a applies only to q_2, and a is consumed from the input. Next $q = q_2$, $v = a(ababaab)$, and the handle of $ababaab$ in $\{q_4, q_5, q_6\}$ is a. Then $aa \Rightarrow q_4$, since a applies only to q_4, and a is consumed, *etc.* This deterministic computation continues until $q = q_1$ and $v = a(b)$. Now the handle of b in $\{q_2, q_3\}$ is b itself. Since the handle is a prefix of minimal

selectors ba and bb, it applies to both q_2 and q_3. Finally, for $v = b(1)$, q_2 moves to q_6, and q_3, to q_7. Thus w yields $\{q_6, q_7\}$. For $w = abba$, we have: $\stackrel{[a]}{\Rightarrow} q_1 \stackrel{a[a]}{\Rightarrow} q_2 \stackrel{b[1]}{\Rightarrow} q_6 \stackrel{b[1]}{\Rightarrow} q_5$. Then the computation stops, yielding the empty set of reachable states.

The simulation operates "almost deterministically" in finding next states. By Corollary 22, we can achieve total determinism concerning the membership of a word in the language of the semiautomaton: Run the simulation until more than one choice appears, and then choose an arbitrary branch in every step (since all choices lead to the same conclusion). If the size of the alphabet Σ, the number of nontrivial forks and the integer k are not prohibitively large, one can pre-compute the set of states reachable from any fork by each word of length $\leq k$, and use a look-up table for the last part of the computation, thus making the entire computation completely deterministic.

6 Conclusions

We have introduced a class of semiautomata in which it is possible to remove most of the nondeterminism by using a finite amount of look-ahead information from the input tape. In the worst case, the length k of the look-ahead buffer is quadratic in the number n of states of $\mathcal{S}$. As long as the input word has length $> k$, the computation is deterministic, and nondeterminism is limited to the last k letters of the input word. The application to nondeterministic automata (semiautomata with accepting states) is straightforward. To determine whether a word w is accepted by an automaton, find the set of states derived by w and check whether any of these states are accepting.

References

[1] D. Berardi, D. Calvanese, G. D. Giacomo, M. Lenzerini, M. Mecella. Automatic composition of e-services that export their behavior. In E. Orłowska, M. Papazoglou, S. Weerawarana, J. Yang (eds.), Proc. ICSOC 2003, LNCS, 2910, 43–58. Springer-Verlag, 2003.

[2] J. Berstel, D. Perrin. Theory of Codes, Academic Press, New York, 1985.

[3] J. Brzozowski, N. Santean. Predictable semiautomata. Technical Report CS-2007-03, D. R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada, 30 pp., 2007. `http://www.cs.uwaterloo.ca/research/tr/2007/`

[4] Z. Dang, O. H. Ibarra, J. Su. Composability of infinite-state activity automata. Proc. ISAAC 2004, R. Fleischer, G. Trippen (eds.), LNCS, 3341, 377–388, 2004.

[5] A. Ginzburg. Algebraic Theory of Automata, Academic Press, New York, 1968.

[6] Y.-S. Han, Y. Wang, D. Wood. Infix-free regular expressions and languages. Int. J. of Foundations of Computer Science, 17(2), 379–393, 2006.

[7] Y.-S. Han, D. Wood. Generalizations of one-deterministic regular languages. LATA 2007, Tarragona, Spain, March 29 - April 4, 2007.

[8] B. Ravikumar, N. Santean. Deterministic simulation of NFA with k-symbol lookahead. Int. J. of Foundations of Computer Science, 18(5), 949–973, 2007.