Finite Automata, Palindromes, Powers, and Patterns

Terry Anderson, Narad Rampersad,
Nicolae Santean, Jeffrey Shallit

School of Computer Science
University of Waterloo

19 March 2008

The Main Questions

- Let $L \subseteq \Sigma^*$ be a fixed language.
- Let M be a DFA or NFA over Σ.
- We consider the following three questions:

1. Can we efficiently decide (in terms of the size of M) if $L(M) \cap L \neq \emptyset$?
2. Can we efficiently decide if $L(M) \cap L$ is infinite?
3. What is a good upper bound on the shortest element of $L(M) \cap L$?

The Languages L Considered

- We consider these questions for the following languages L.
- The language of **palindromes**, i.e., words x such that x equals its reversal x^R.
- The language of k**-powers**, i.e., words x that can be written as $x = y^k = yy \cdots y$ (k times).
- The language of **powers**, i.e., words that are k-powers for some $k \geq 2$.
- The language of words matching a given **pattern** p, i.e., words x for which there exists a non-erasing morphism h such that $h(p) = x$.
- Let us also refer to 2-powers and 3-powers as **squares** and **cubes** respectively. We also call non-powers **primitive words**.

Testing if an NFA Accepts at Least One Palindrome

- To warm-up, let us see how to test if an NFA accepts a palindrome.
- If M is an NFA with n states and t transitions, it is easy to construct an NFA M' with $n^2 + 1$ states and $\leq 2t^2$ transitions that accepts

$$L' = \{x \in \Sigma^* : xx^R \in L(M) \text{ or there exists } a \in \Sigma \text{ such that } xax^R \in L(M)\}.$$

- Since NFA emptiness and finiteness can be tested in linear time, using M' we can determine if M accepts a palindrome (or infinitely many palindromes) in $O(n^2 + t^2)$ time.

Testing if an NFA Accepts at Least One Palindrome

- A somewhat more difficult problem is determining if an NFA accepts a **palindromic language** (i.e., accepts only palindromes).
- Horváth, Karhumäki, and Kleijn (1987) proved that the question is recursively solvable.
- They proved that if M is an n-state NFA, then $L(M)$ is palindromic if and only if $\{x \in L(M) \ : \ |x| < 3n\}$ is palindromic.
- To obtain a polynomial time algorithm for palindromicity, we intersect M with a "small" NFA M' such that M' never accepts a palindrome and accepts all non-palindromes of length less than $3n$.
- We then test if this new NFA accepts the empty language.

Testing if an NFA Accepts a Word Matching a Pattern

- A **pattern** is simply a non-empty word over some alphabet Δ.
- We say a pattern $p \in \Delta^*$ **matches** a word $w \in \Sigma^*$ if there exists a non-erasing morphism $h : \Delta^* \to \Sigma^*$ such that $h(p) = w$.
- For example, if $p = xyyx$ and $w = 02111102$, then p matches w, since we may take $h(x) = 02$ and $h(y) = 11$.
- Patterns generalize the notion of k-powers, since a k-power is a word matching the pattern x^k.

Testing if an NFA Accepts a Word Matching a Pattern

- We now consider the computational complexity of the decision problem:

  NFA PATTERN ACCEPTANCE
  INSTANCE: An NFA M over the alphabet Σ and a pattern p over some alphabet Δ.
  QUESTION: Does there exist $x \in \Sigma^+$ such that $x \in L(M)$ and x matches p?

- The solvability of this problem is implied by the following result (Restivo and Salemi (2001); Castiglione, Restivo, and Salemi (2004)): Let L be a regular language and let Δ be an alphabet. The set P_Δ of all non-empty patterns $p \in \Delta^*$ such that p matches a word in L is effectively regular.

Testing if an NFA Accepts a Word Matching a Pattern

Theorem

The **NFA PATTERN ACCEPTANCE** *problem is PSPACE-complete.*

- By Savitch's theorem it suffices to give an NPSPACE algorithm.
- For an alphabet symbol a, the transitions of an NFA M can be represented by a boolean matrix B_a.
- For a word $w = w_0 w_1 \cdots w_s$, we write B_w for the matrix product $B_{w_0} B_{w_1} \cdots B_{w_s}$.
- Suppose the pattern alphabet is $\Delta = \{1, 2, \ldots, k\}$.
- Non-deterministically guess k boolean matrices $B_1, \ldots, B_k$.
- For each i, verify that $B_i = B_w$ for some word w of length at most 2^{n^2}.

Testing if an NFA Accepts a Word Matching a Pattern

- We guess w symbol-by-symbol and reuse space after perfoming each matrix multiplication while computing B_w.
- If $p = p_0 p_1 \cdots p_r$, compute $B = B_{p_0} B_{p_1} \cdots B_{p_r}$ and accept if and only if B describes an accepting computation of M.
- To show hardness is a straightforward reduction from the PSPACE-complete problem

  DFA INTERSECTION
  INSTANCE: An integer $k \geq 1$ and k DFAs $A_1, A_2, \ldots, A_k$, each over the alphabet Σ.
  QUESTION: Does there exist $x \in \Sigma^*$ such that x is accepted by each A_i, $1 \leq i \leq k$?

Special Cases of Pattern Acceptance

- A special case of **NFA PATTERN ACCEPTANCE** is the **NFA ACCEPTS A** ***k*****-POWER** problem.
- When k is part of the input (i.e., k is not fixed), this is still PSPACE-complete.
- However, if k is fixed, this problem can be solved in polynomial time.

Proposition

Let M be an NFA with n states and t transitions, and set $N = n + t$, the size of M. For any fixed integer $k \geq 2$, there is an algorithm running in $O(n^{2k-1}t^k) = O(N^{2k-1})$ time to determine if M accepts a k-power.

Automata Accepting Only Powers

- Ito, Katsura, Shyr, and Yu (1988) proved the following result (stated here in a slightly stronger form than in the original).

Theorem (Ito et. al (1988))

Let L be accepted by an n-state NFA M.

1. *Every word in L is a power if and only if every word in the set $\{x \in L : |x| \leq 3n\}$ is a power.*
2. *All but finitely many words in L are powers if and only if every word in the set $\{x \in L : n \leq |x| \leq 3n\}$ is a power.*

The Idea of the Proof

- Suppose to the contrary that a shortest non-power $x \in L$ had length greater than $3n$.
- An accepting computation of M on x must repeat some state q four times.
- It follows that $x = uv_1 v_2 v_3 w$ such that $uv_1^* v_2^* v_3^* w \subseteq L$.
- Consider the words obtained by deleted one or more of the v_i's from x, e.g., $uv_1 v_3 w$, $uv_2 w$, uw, etc. These must all be powers.
- Use standard results from combinatorics on words to derive a contradiction by showing that if these words are all powers, then x must be a power, contrary to our assumption.

Slenderness

- The characterization due to Ito et al. (1988) (see also Dömösi, Horváth, and Ito (2004)) showed that any regular language consisting only of powers is slender.
- A language L is **slender** if there is a constant C such that, for all $i \geq 0$, the number of words of length i in L is less than C.
- The following characterization of slender regular languages has been independently rediscovered several times in the past (Kunze, Shyr, and Thierrin (1981); Shallit (1994); Paun and Salomaa (1995)).
- Let $L \subseteq \Sigma^*$ be a regular language. Then L is slender if and only if it can be written as a finite union of languages of the form uv^*w, where $u, v, w \in \Sigma^*$.

Bounding the Number of Words of Each Length

- Again, if a regular language L contains only powers, it contains at most C words of length i for every $i \geq 0$.
- Next we show how to bound C in terms of the number n of states of an *NFA* accepting L.
- We then use the bound to give an efficient algorithm to test if a regular language contains only powers.

Proposition

Let M be an n-state NFA and let ℓ be a non-negative integer such that every word in $L(M)$ of length $\geq \ell$ is a power. For all $r \geq \ell$, the number of words in $L(M)$ of length r is at most $7n$.

Bounding the Number of Words of Each Length

- To prove this, we use a technique from the theory of non-deterministic state complexity and a classical result from combinatorics on words.

Theorem (Birget (1992))

Let $L \subseteq \Sigma^*$ be a regular language. Suppose there exists a set of pairs $S = \{(x_i, y_i) \in \Sigma^* \times \Sigma^* : 1 \leq i \leq n\}$ such that: (a) $x_i y_i \in L$ for $1 \leq i \leq n$; and (b) either $x_i y_j \notin L$ or $x_j y_i \notin L$ for $1 \leq i, j \leq n$, $i \neq j$. Then any NFA accepting L has at least n states.

Theorem (Lyndon and Schützenberger (1962))

If x, y, and z are words satisfying an equation $x^i y^j = z^k$, where $i, j, k \geq 2$, then they are all powers of a common word.

Bounding the Number of Words of Each Length

- Let $r \geq \ell$ be an arbitrary integer.
- Consider the set A of words w in $L(M)$ such that $|w| = r$ and w is a k-power for some $k \geq 4$.
- For each such w, write $w = x^i$, where x is a primitive word, and define a pair (x^2, x^{i-2}). Let S_A denote the set of such pairs.
- Consider two pairs in S_A: (x^2, x^{i-2}) and (y^2, y^{j-2}).
- The word $x^2 y^{j-2}$ is primitive by the Lyndon–Schützenberger theorem and hence is not in $L(M)$. The set S_A thus satisfies the conditions of Birget's theorem. Since $L(M)$ is accepted by an n-state NFA, we must have $|S_A| \leq n$ and thus $|A| \leq n$.
- Similar considerations (which we omit) allow us to bound the number of cubes and squares in $L(M)$, and result in the claimed bound of $7n$.

Testing if an Automaton Only Accepts Powers

Theorem

Given an NFA M with n states, it is possible to determine if every word in $L(M)$ is a power in $O(n^5)$ time.

- Checking if a word is a power can be done in linear time using the Knuth-Morris-Pratt algorithm.
- By the results previously mentioned it suffices to enumerate the words in $L(M)$ of lengths $1, 2, \ldots, 3n$, stopping if the number of such words in any length exceeds $7n$.
- If all these words are powers, then every word is a power.
- Otherwise, if we find a non-power, or if the number of words in any length exceeds $7n$, then not every word is a power.
- By the work of Mäkinen (1997) or Ackerman & Shallit (2007), we can enumerate these words in $O(n^5)$ time.

Testing if An NFA Only Accepts k-powers

- How can we efficiently test if an NFA only accepts k-powers?
- First we establish a result for k-powers analogous to that of Ito et. al for powers.

Theorem

Let L be accepted by an n-state NFA M and let $k \geq 2$ be an integer.

1. *Every word in L is a k-power if and only if every word in the set $\{x \in L : |x| \leq 3n\}$ is a k-power.*
2. *All but finitely many words in L are k-powers if and only if every word in the set $\{x \in L : n \leq |x| \leq 3n\}$ is a k-power.*

Testing if An NFA Only Accepts k-powers

- Next we use this result to deduce the following algorithmic result.

Theorem

Let $k \geq 2$ be an integer. Given an NFA M with n states and t transitions, it is possible to determine if every word in $L(M)$ is a k-power in $O(n^3 + tn^2)$ time.

- The idea is to create a "small" NFA M'_r for $r = 3n$ such that no word in $L(M'_r)$ is a k-power, and M'_r accepts all non-k-powers of length $\leq r$ (and perhaps some other non-k-powers).
- We now form a new NFA A as the cross product of M'_r with M. It follows that $L(A) = \emptyset$ iff every word in $L(M)$ is a k-power.
- Again, we can determine if $L(A) = \emptyset$ in linear time.

Summary of Results for Various L

L	decide if $L(M) \cap L = \emptyset$	decide if $L(M) \cap L$ infinite
palindromes	$O(n^2 + t^2)$	$O(n^2 + t^2)$
non-palindromes	$O(n^2 + tn)$	$O(n^2 + t^2)$
k-powers (k fixed)	$O(n^{2k-1} t^k)$	$O(n^{2k-1} t^k)$
k-powers (k part of input)	PSPACE-complete	PSPACE-complete
non-k-powers	$O(n^3 + tn^2)$	$O(n^3 + tn^2)$
powers	PSPACE-complete	PSPACE-complete
non-powers	$O(n^5)$	$O(n^5)$

Thank you!