Open Problems in Automata Theory: An Idiosyncratic View

Jeffrey Shallit
School of Computer Science
University of Waterloo
Waterloo, Ontario N2L 3G1
Canada
shallit@cs.uwaterloo.ca
http://www.cs.uwaterloo.ca/~shallit

Outline

1. The separating words problem
2. Bucher's problem on separating context-free languages
3. Nonzero Hankel determinants
4. The Endrullis-Hendriks transducer problem
5. The Oldenburger-Kolakoski problem
6. Primes and automata
7. Divisibility and automata
8. A universality problem

The Simplest Computational Problem?

Imagine a stupid computing device with very limited powers...

What is the simplest computational problem you could ask it to solve?

The Simplest Computational Problem?

- not the addition of two numbers

- not sorting

- it's telling two inputs apart - distinguishing them

Thanks to Gavin Rymill for letting me use his Dalek image.

Our Computational Model: the Finite Automaton

Our main computational model is the **deterministic finite automaton**, or DFA.

We also consider **nondeterministic finite automata**, or NFA.

Motivation

We want to know how many states suffice to tell one length-n input from another.

On average, it's easy — but how about in the worst case?

Motivation: a classical problem from the early days of automata theory:

Given two automata, how big a word do we need to distinguish them?

Motivation

More precisely, given two DFA's M_1 and M_2, with m and n states, respectively, with $L(M_1) \neq L(M_2)$, what is a good bound on the length of the shortest word accepted by one but not the other?

- The cross-product construction gives an upper bound of $mn - 1$ (make a DFA for $L(M_1) \cap \overline{L(M_2)}$)
- But an upper bound of $m + n - 2$ follows from the usual algorithm for minimizing automata
- Furthermore, this bound is best possible.
- For NFA's the bound is exponential in m and n

Separating Words with Automata

Our problem is the inverse problem: given two distinct *words*, how big an automaton do we need to separate them?

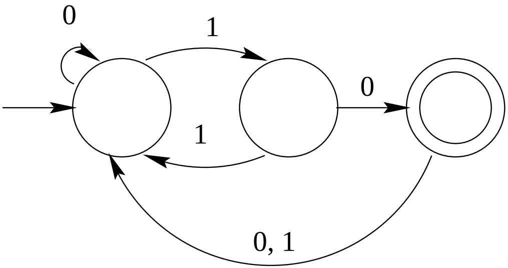
That is, given two words w and x of length $\leq n$, what is the smallest number of states in any DFA that accepts one word, but not the other?

Call this number $\text{sep}(w, x)$.

Separation

A machine M **separates** the word w from the word x if M accepts w and rejects x, or vice versa.

For example, the machine below separates 0010 from 1000.



However, no 2-state DFA can separate these two words. So $\mathrm{sep}(1000, 0010) = 3$.

Separating Words of Different Length

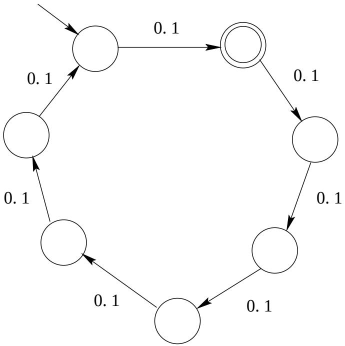
Easy case: if the two words are of different lengths, both $\leq n$, we can separate them with a DFA of size $O(\log n)$.

For by the prime number theorem, if $k \neq m$, and $k, m \leq n$ then there is a prime $p = O(\log n)$ such that $k \not\equiv m \pmod{p}$.

So we can accept one word and reject the other by using a cycle mod p, and the appropriate residue class.

Separating Words of Different Length

Example: suppose $|w| = 22$ and $|x| = 52$. Then $|w| \equiv 1 \pmod 7$ and $|x| \equiv 3 \pmod 7$. So we can accept w and reject x with a DFA that uses a cycle of size 7, as follows:



Separating Words with Automata

A similar idea works if the strings have a different number of 1's, or if the 1's are in different positions, or if the number of occurrences of a short subword is different, etc.

Separating Words With Automata

- Let

$$S(n) := \max_{\substack{|w|=|x|=n \\ w \neq x}} \mathrm{sep}(w, x),$$

the smallest number of states required to separate any two strings of length n.

- The separation problem was first studied by Goralcik and Koubek 1986, who proved $S(n) = o(n)$.

- In 1989 Robson who obtained the best known bound: $S(n) = O(n^{2/5}(\log n)^{3/5})$.

Separating Words with Automata

For equal-length strings, $S(n)$ doesn't depend on alphabet size (provided it is at least 2).

Suppose x, y are distinct strings of length n an alphabet Σ of size > 2.

Then they must differ in some position, say

$$\begin{aligned} x &= x' \; a \; x'' \\ y &= y' \; b \; y'' \end{aligned}$$

for $a \neq b$.

Map a to 0, b to 1 and assign all other letters arbitrarily to either 0 or 1. This gives two new distinct strings X and Y of the same length. If X and Y can be separated by an m-state DFA, then so can x and y, by renaming transitions to be over Σ instead of 0 and 1.

Not the best upper bound:

Theorem (Robson, 1996). We can separate words by computing the parity of the number of 1's occurring in positions congruent to $i \pmod{j}$, for $i, j = O(\sqrt{n})$.

This gives the bound $S(n) = O(n^{1/2})$.

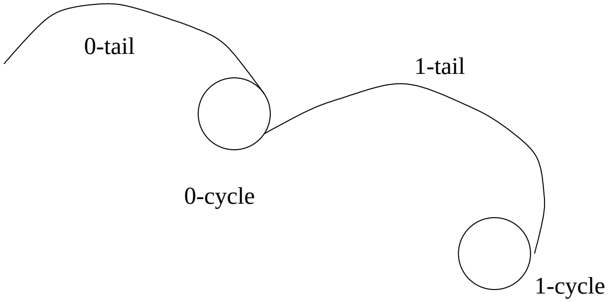
Open Problem 1 (£100): Improve Robson's upper bound of $O(n^{2/5}(\log n)^{3/5})$ on $S(n)$.

Separating Words With Automata: Lower Bound

- Claim: $S(n) = \Omega(\log n)$.
- To see this, consider the two strings

$$0^{t-1+\mathrm{lcm}(1,2,\ldots,t)}1^{t-1} \quad \text{and} \quad 0^{t-1}1^{t-1+\mathrm{lcm}(1,2,\ldots,t)}.$$

Proof in pictures:



Separating Words With Automata: Lower Bound

So no t-state machine can distinguish these strings.

Since $\text{lcm}(1, 2, \ldots, t) = e^{t+o(t)}$ by the prime number theorem, the lower bound $S(n) = \Omega(\log n)$ follows.

Variations on Separating Words

- Separation by context-free grammars; count number of productions
- Problem: right-hand sides can be arbitrarily complicated
- Solution: Use CFG's in Chomsky normal form (CNF), where all productions are of the form $A \rightarrow BC$ or $A \rightarrow a$.

Variations on Separating Words

- In 1999 Currie, Petersen, Robson and JOS proved:
  - If $|w| \neq |x|$ then there is a CFG in CNF with $O(\log \log n)$ productions separating w from x. Furthermore, this bound is optimal.
  - If $|w| = |x|$ there is a CFG in CNF with $O(\log n)$ productions separating w from x. There is a lower bound of $\Omega(\frac{\log n}{\log \log n})$.

Open Problem 2 (£10): Find matching upper and lower bounds for CFG's in the case $|w| = |x|$.

More Variations on Separating Words

- Separation by NFA. Do NFA's give more power?

Yes,

$$\mathrm{sep}(0001, 0111) = 3$$

but

$$\mathrm{nsep}(0001, 0111) = 2.$$

More Variations on Separating Words

Is

$$\mathrm{sep}(x, w)/\mathrm{nsep}(x, w)$$

unbounded?

Yes.

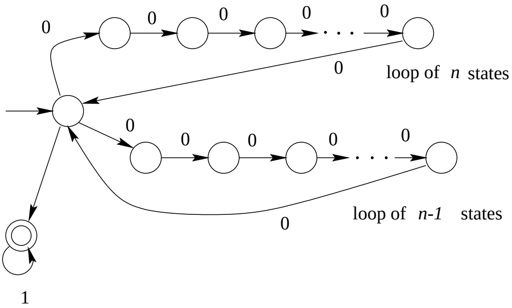
Consider once again the strings

$$w = 0^{t-1+\mathrm{lcm}(1,2,\ldots,t)}1^{t-1} \quad \text{and} \quad x = 0^{t-1}1^{t-1+\mathrm{lcm}(1,2,\ldots,t)}$$

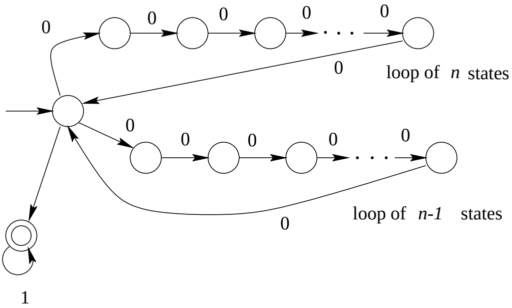
where $t = n^2 - 3n + 2$, $n \geq 4$.

We know from before that any DFA separating these strings must have at least $t + 1 = n^2 - 3n + 3$ states.

Now consider the following NFA M:



The language accepted by this NFA is $\{0^a \ : \ a \in A\}1^*$, where A is the set of all integers representable by a non-negative integer linear combination of n and $n - 1$.



But $t - 1 = n^2 - 3n + 1 \notin A$.

On the other hand, every integer $\geq t$ is in A. Hence
$w = 0^{t-1+\mathrm{lcm}(1,2,\ldots,t)}1^{t-1}$ is accepted by M but
$x = 0^{t-1}1^{t-1+\mathrm{lcm}(1,2,\ldots,t)}$ is not.

M has $2n = \Theta(\sqrt{t})$ states, so
$\mathrm{sep}(x, w)/\mathrm{nsep}(x, w) \geq \sqrt{t} = \Omega(\sqrt{\log |x|})$, which is unbounded.

More Variations on Separating Words

Open Problem 3 (£50): Find good bounds on $\mathrm{nsep}(w, x)$ for $|w| = |x| = n$, as a function of n.

Open Problem 4 (£50): Find good bounds on $\mathrm{sep}(w, x)/\mathrm{nsep}(w, x)$.

More Variations on Separating Words

- Must $\mathrm{sep}(w^R, x^R) = \mathrm{sep}(w, x)$?

  No, for $w = 1000$, $x = 0010$, we have
  $$\mathrm{sep}(w, x) = 3$$
  but
  $$\mathrm{sep}(w^R, x^R) = 2.$$

Open Problem 5 (£10):
Is
$$\left|\mathrm{sep}(x, w) - \mathrm{sep}(x^R, w^R)\right|$$
unbounded?

More Variations on Separating Words

- Two words are *conjugates* if one is a cyclic shift of the other.
- Is the separating words problem any easier if restricted to pairs of conjugates?

Another Kind of Separation

Suppose you have regular languages R_1, R_2 with $R_1 \subseteq R_2$ and $R_2 - R_1$ infinite.

Then it is easy to see that there is a regular language R_3 such that $R_1 \subseteq R_3 \subseteq R_2$ such that $R_2 - R_3$ and $R_3 - R_1$ are both infinite.

This is a kind of topological separation property.

Another Kind of Separation

In 1980, Bucher asked:

Open Problem 6 (£100): Is the same true for context-free languages?

That is, given context-free languages L_1, L_2 with $L_1 \subseteq L_2$ and $L_2 - L_1$ infinite, need there be a context-free language L_3 such that $L_1 \subseteq L_3 \subseteq L_2$ such that $L_2 - L_3$ and $L_3 - L_1$ are both infinite?

Not even known in the case where $L_2 = \Sigma^*$.

The primitive words problem

Speaking of context-free languages, it's still not known if the primitive words over $\{0, 1\}$ are context-free.

A word is *primitive* if it is a non-power.

This is **Open Problem 7** (£200).

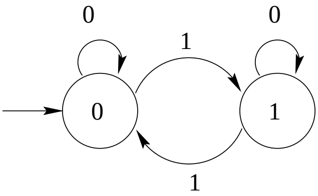
A forthcoming 500-page book by Dömösi, Horváth, and Ito called *Context-Free Languages and Primitive Words* will appear in June 2014.

The Thue-Morse sequence

$$\mathbf{t} = t_0 t_1 t_2 \cdots = 0110100110010110100101100\cdots$$

can be described in many ways

- by the recurrence $t_{2n} = t_n$ and $t_{2n+1} = 1 - t_n$
- as the fixed point of the map $0 \to 01$ and $1 \to 10$
- as the sequence generated by the following DFA, where the input is n expressed in base 2



Avoidability

- One of the beautiful properties of the Thue-Morse word is that it avoids overlaps
- An **overlap** is a subword (factor) of the form $axaxa$, where a is a single letter, and x is a word
- We'd like to generalize this
- One possible generalization involves Hankel determinants

Avoidability

- An $n \times n$ **Hankel determinant** of a sequence $(a_i)_{i \geq 0}$ is a determinant of a matrix of the form

$$\begin{bmatrix} a_k & a_{k+1} & \cdots & a_{k+n-1} \\ a_{k+1} & a_{k+2} & \cdots & a_{k+n} \\ a_{k+2} & a_{k+3} & \cdots & a_{k+n+1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k+n-1} & a_{k+n} & \cdots & a_{k+2n-2} \end{bmatrix}.$$

- If a sequence of real numbers has an overlap

$$\overbrace{a_k}^{a}\ \overbrace{a_{k+1} \cdots a_{k+n-2}}^{x}\ \overbrace{a_{k+n-1}}^{a}\ \overbrace{a_{k+n} \cdots a_{k+2n-3}}^{x}\ \overbrace{a_{k+2n-2}}^{a}$$

then the corresponding $n \times n$ Hankel matrix has the first and last rows both equal to axa, and so the determinant is 0.

Avoidability

- Is there a sequence on **two** real numbers for which all the Hankel determinants (of all orders) are nonzero?
- No - a simple backtracking argument proves that the longest such sequence is of length 14. For example,

$$1, 1, 2, 1, 1, 2, 2, 1, 2, 1, 1, 2, 1, 2.$$

- How about sequences over **three** numbers?

Open Problem 8 ($\pounds 100$): Is there a sequence on three real numbers for which all the Hankel determinants are nonzero?

Avoidability

Indeed, a backtracking algorithm easily finds such a sequence on $\{1, 2, 3\}$ with 200 terms.

What is interesting is that this algorithm never had to backtrack!

The sequence begins

$$1, 1, 2, 1, 1, 2, 2, 1, 2, 1, 1, 2, 1, 2, 3, 1, 1, 2, 1, \ldots$$

Avoidability

Furthermore, the following morphism seems to generate such a sequence on 4 symbols:

$1 \to 12$, $2 \to 23$, $3 \to 14$, $4 \to 32$.

This generates the sequence

$$1223231423141232\cdots.$$

Open Problem 9 (£50): Does this sequence have all nonzero Hankel determinants? We have checked up to length 800.

The Endrullis-Hendriks Problem on Transducers

- For two infinite words $\mathbf{w}$ and $\mathbf{x}$, we write $\mathbf{w} \leq \mathbf{x}$ if we can transform $\mathbf{x}$ into $\mathbf{w}$ by a finite-state transducer.
- If $\mathbf{w} \leq \mathbf{x}$ and $\mathbf{x} \leq \mathbf{w}$ then we write $\mathbf{w} \equiv \mathbf{x}$.
- We call an infinite word $\mathbf{w}$ *prime* if whenever $\mathbf{x} \leq \mathbf{w}$ either $\mathbf{x} \equiv \mathbf{w}$ or $\mathbf{x}$ is ultimately periodic.

The Endrullis-Hendriks Problem on Transducers

- Example: the infinite word

$$0\,1\,\overbrace{00}^{2}\,1\,\overbrace{000}^{3}\,1\,\overbrace{0000}^{4}\,\ldots$$

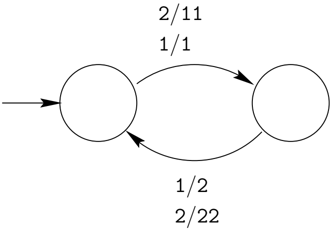
  is prime.

Open Problem 10 (£20): are there any other (inequivalent) prime words over two letters?

Open Problem 11 (£20): is the Thue-Morse word prime?

The Oldenburger-Kolakoski problem

Speaking of transducers, consider the following transducer:



The fixed point of this transducer is

$$\mathbf{k} = 12211212212211211221211 \cdots ,$$

the Oldenburger-Kolakoski word.

Open Problem 12 (£100): Do the frequencies of letters exist? Are they $\frac{1}{2}$?

Open Problem 13 (£50): Is the following question recursively solvable?

Given a DFA M over the alphabet $\Sigma_k = \{0, 1, \ldots, k-1\}$, does M accept the base-k representation of at least one prime number?

If it were, we could decide if there are any more Fermat primes after the largest known one $(2^{16}+1)$.

Open Problem 14 (£50): Is the following question recursively solvable?

Given a DFA M over the alphabet $\Sigma_k \times \Sigma_k$, does it accept the base-k representation of a pair of integers (x, y) with $x \mid y$?

A universality problem

Classical problem: given M, a machine of some type (e.g., DFA, NFA, PDA), decide if $L(M) = \Sigma^*$.

- Unsolvable, if M is a PDA;
- PSPACE-complete, if M is an NFA;
- Solvable in polynomial time if M is a DFA.

A universality problem

A simple variation:

Given M, does there exist an integer $n \geq 0$ such that $\Sigma^n \subseteq L(M)$?

- Still unsolvable for PDA's
- NP-complete for DFA's
- PSPACE-hard for NFA's - but is it in PSPACE? This is **Open Problem 15** (£25).

A universality problem

It would follow that this problem is in PSPACE if we knew that if $\Sigma^r \subseteq L(M)$ for some n, then there always exists a "small" such r (e.g., $r \leq exp(p(n))$ for some polynomial p).

To see this, on input the DFA, we just examine every length l up to the bound r and nondeterministically guess a string of length l (symbol-by-symbol) that *fails* to be accepted. All this can be done in NPSPACE and hence PSPACE by Savitch's theorem.

A related problem

Here is a related problem.

Open Problem 16 (£25): what is the complexity of the following problem: given a finite language L, is $\overline{L^*}$ infinite?

If L is represented by a regular expression, this problem is NP-hard.

Another related problem

Open Problem 17 (£25): What is the complexity of the following problem: given a finite list of words L over an alphabet Σ, is $\mathrm{Fact}(L^*) = \Sigma^*$?

Here by "Fact" we mean the set of all factors (contiguous subwords).

Similarly, we'd like to find good bounds on the length of the shortest word in $\Sigma^* - \mathrm{Fact}(S^*)$, given that $\mathrm{Fact}(L^*) \neq \Sigma^*$. This is **Open Problem 18** (£200). Currently examples of quadratic length are known, but the best upper bound is doubly-exponential in the length of the longest word in S.

One More for Dessert: Pierce Expansions

Let $a > b > 0$ be integers. Define $b_0 = b$ and $b_{i+1} = a \bmod b_i$ for $i \geq 0$.

Let $P(a, b)$ be the least index n such that $b_n = 0$.

One More for Dessert: Pierce Expansions

An example with $a = 35$, $b = 22$:

$$\begin{aligned}
b_0 &= 22 \\
b_1 &= 35 \bmod 22 = 13 \\
b_2 &= 35 \bmod 13 = 9 \\
b_3 &= 35 \bmod 9 = 8 \\
b_4 &= 35 \bmod 8 = 3 \\
b_5 &= 35 \bmod 3 = 2 \\
b_6 &= 35 \bmod 2 = 1 \\
b_7 &= 35 \bmod 1 = 0.
\end{aligned}$$

So $P(35, 22) = 7$.

Open Problem 19 (£200): Find good estimates for how big $P(a, b)$ can be, as a function of a.

The problem is interesting because it is related to the so-called "Pierce Expansion" of b/a:

$$\frac{22}{35} = \frac{1}{1}(1 - \frac{1}{2}(1 - \frac{1}{3}(1 - \frac{1}{4}(1 - \frac{1}{11}(1 - \frac{1}{17}(1 - \frac{1}{35})))))).$$

This is called a *Pierce expansion*.

One More for Dessert: Pierce Expansions

It is known that $P(a, b) = O(a^{1/3})$. However, the true behavior is probably $O((\log a)^2)$. There is a lower bound of $\Omega(\log a)$, which can be obtained by choosing

$$a = \operatorname{lcm}(1, 2, \ldots, n) - 1$$

$$b = n.$$

I offer £200 for a significant improvement to either the known upper or lower bound.

For Further Reading

- J. M. Robson, Separating words with machines and groups, *RAIRO Info. Theor. Appl.* **30** (1996), 81–86.
- J. Currie, H. Petersen, J. M. Robson, and J. Shallit, Separating words with small grammars, *J. Autom. Lang. Combin.* **4** (1999), 101–110.
- N. Rampersad, J. Shallit, Z. Xu, The computational complexity of universality problems for prefixes, suffixes, factors, and subwords of regular languages, *Fundam. Inform.* **116** (1-4) (2012), 223–236.