15-354: Computational Discrete Math	**K. Sutner**
Assignment 7	Due: November 3, 2011

Problem 1: Representations of Regular Languages (40 pts.)

Background

Task

We have seen that a regular language (i.e., a language that is accepted by a finite state machine) can also be described by a regular expression or by a formula in $\mathsf{MSO}[<]$.

Recall that u is a factor of v if $\exists\, x, y \in \Sigma^\star\, (v = xuy)$ and $u = u_1 u_2 \ldots u_n$ is a subword of v if $\exists\, x_0, \ldots, x_n \in \Sigma^\star\, (v = x_0 u_1 x_1 u_2 x_3 \ldots u_n x_n)$.

Let K be the language of all words over alphabet $\Sigma = \{a, b, c\}$ containing exactly two factors ab. Likewise, let L be the language of all words over alphabet $\Sigma = \{a, b, c\}$ containing exactly two subwords ab.

A. Construct the minimal automaton for K and L.

B. Give a regular expression for K and L.

C. Give a formula in $\mathsf{MSO}[<]$ for K and L.

Comment

For the regular expressions and formulae try to find short, elegant answers. Make sure to explain what you are doing.

Problem 2: Acceptance for Büchi Automata (30 pts.)

Background

As mentioned in class, running a finite state machine on a particular input is one of the two "killer apps." The other is to use the machines to describe certain systems and/or properties thereof and solving the Emptyness or Universality problem for these machines.

Büchi automata are clearly in the second category; in fact, it is not even clear how the input would be specified in this case. If we consider input words given by a computable function then acceptance is undecidable. However, for very simple infinite words acceptance is still decidable.

Let $\mathcal{A} = \langle Q, \Sigma, \tau; I, F \rangle$ be a Büchi automaton (so, by default, $\mathcal{A}$ is nondeterministic).

Task

A. Show how to decide whether $\mathcal{A}$ accepts a "constant" input word $U = a^\omega$ where $a \in \Sigma$.

B. Generalize to periodic inputs $U = u^\omega$ where $u \in \Sigma^\star$.

C. Generalize to ultimately periodic inputs $U = vu^\omega$ where $u, v \in \Sigma^\star$.

D. What is the running time of your three algorithms?

Comment

One might try to push things a bit further and consider inputs that are not periodic but have some reasonably simple description. For example, we can think of the characteristic function of the primes as an infinite word defined by

$$U(k) = \begin{cases} 1 & \text{if } k \text{ is prime,} \\ 0 & \text{otherwise.} \end{cases}$$

So

$$U = 00110101000101000101000100000101000001000101000100000100000010\ldots$$

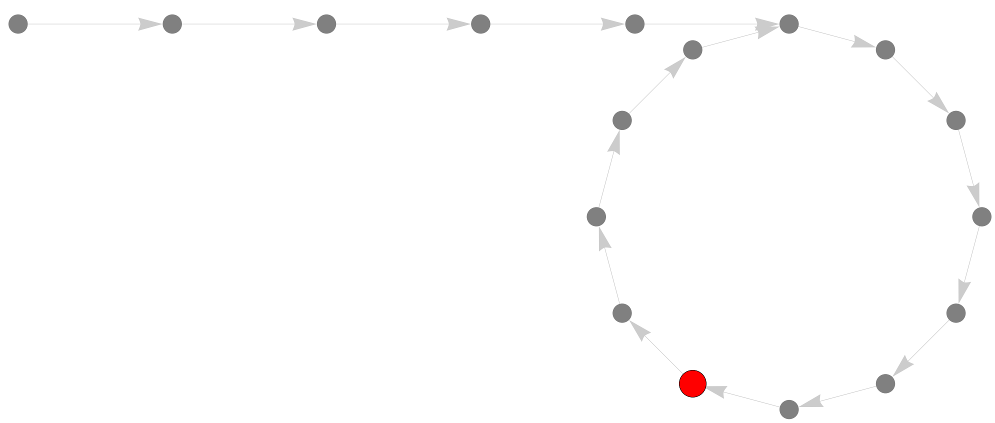
Do you think it is decidable whether some arbitrary Büchi automaton accepts U?

Problem 3: Floyd goes Algebraic (30 pts.)

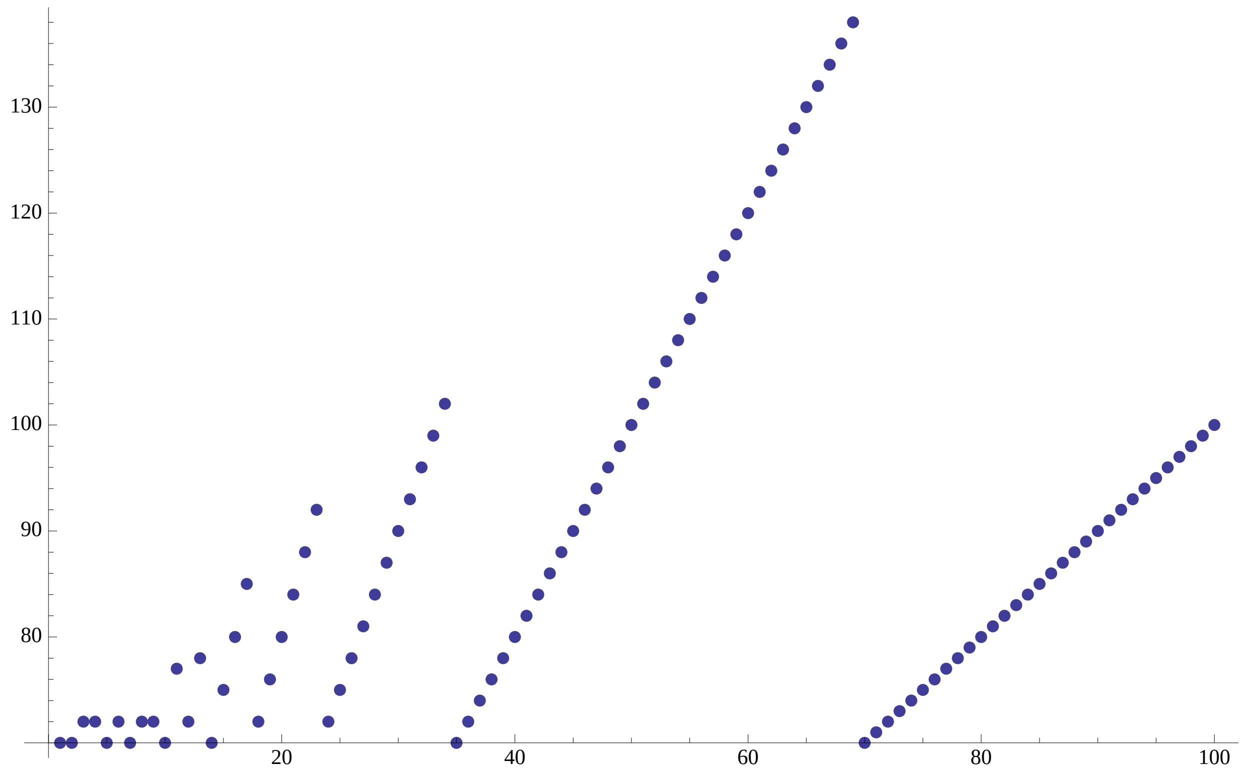
Background

One of the most elementary results about finite semigroups is that every element in the semigroup has an idempotent power. Recall that x is idempotent if $x^2 = x$. Then the claim is that for S any finite semigroup and $a \in S$, there exists some integer $r \geq 1$ such that a^r is idempotent.

Ignoring algebraic aspects for a moment, note that the powers of a (the points a^i, $i \geq 0$) must form a lasso since S is finite. Hence we can associate a with a transient $t = t(a) \geq 0$ and a period $p = p(a) \geq 1$.



More importantly, we can apply Floyd's trick to find a point on the loop (the big, red dot above). Let's call the time when the algorithm finds this point the *Floyd time*, an integer in the range 0 to $t + p - 1$. The next picture shows the Floyd times for $t = 70$ and $p = 1, \ldots, 100$.



Inquisitive minds will wonder if there is any connection between the Floyd point and the idempotent from above.

Task

A. Prove the claim about idempotent powers. Hint: think about the Floyd point.

B. Explain the plot of the Floyd times above.

C. Describe the exponent r in the claim in terms of the transient and period of a.

D. Show that the powers of a that lie on the loop form a group with the idempotent a^r as identity.

Comment

For extra credit you might (re-)consider the question of what happens in Floyd's algorithm when one changes the velocities of the particles to $1 \leq u < v$.