A polynomial time algorithm for determining if a transducer's associated semigroup is Abelian

Tsutomu Okano & Evan Bergeron

September 14, 2016

We have the following characterization from Okano: $S(A)$ is Abelian iff there exists a unique $\Theta_A \in S(A)$ such that $\partial_0 \underline{t} = \partial_1 \underline{t} \Theta_A$ for all toggle states t. So it suffices to verify that each toggle state obeys this.

Since these transducers are synchronous (output exactly one character per transition), we can build a DFA that recognizes the input-output relation of some fixed starting state. Take some s this start state. Then build a DFA over the alphabet $\Sigma \times \Sigma$ with transitions simulating the behavior of the transducer starting at s. This DFA is called the *acceptor of A at s*.

An automaton semigroup $S(A)$ has a presentation with generators corresponding to the states of A. Thinking of elements of $S(A)$ as words over this generator alphabet, we see that the state set of A is precisely the words of length 1. We can build an automaton whose state set is the set of all length 2 words as follows:

If $A = (Q, \Sigma, \delta)$, the product automaton $A \times A$ has state set $Q \times Q$ with transition function $\partial_a(s_1, s_2) = (\partial_a s_1, \partial_{a s_1} s_2)$. We can see by induction that each state (s_1, s_2) corresponds to the word $s_1 s_2 \in S(A)$.

Remember that $(\partial_1 \underline{t}) = (\partial_0 \underline{t}) \Theta_A$. So then we have $(\partial_1 \underline{t})(\partial_0 \underline{t})^{-1} = \Theta_A$. (This is straight from Okano's thesis, I think this expression is better written as $(\partial_0 \underline{t})^{-1}(\partial_1 \underline{t}) = \Theta_A$ - it seems more consistent with function-application-from-the-right).

Anyway, $\Theta_A = (\partial_1 \underline{t})(\partial_0 \underline{t})^{-1} = (\partial_1 \underline{t})(\partial_1 \underline{t^{-1}})$, where t^{-1} lies in the inverse automaton for A. So then build for each toggle state the following: $(A \times A^{-1})(\partial_1 \underline{t}, \partial_1 \underline{t^{-1}})$. Note that the language of this automaton is $\{x : y \mid y = x(\partial_1 \underline{t})(\partial_0 \underline{t})^{-1}\}$. It suffices

to then verify that all of these constructed DFAs are equivalent, which can be done via Hopcroft's minimization algorithm.

So to summarize algorithmically: take the input automaton A. Build it's inverse automaton A^{-1}. Construct the product automaton $A \times A^{-1}$. Then for each toggle state t_i of A, take the state $s_i = (\partial_1 t_i, \partial_1 t_i^{-1})$ in $A \times A^{-1}$ and construct the DFA $(A \times A^{-1})(s_i)$. Verify all the constructed DFAs are equivalent.

This product automaton construction also gives us that the word problem for automaton semigroups is decidable.