Universal Simulation of Automata Networks

Alonso Castillo-Ramirez and Maximilien Gadouleau

Durham University

NBSAN, Manchester, 2015-11-20

Outline

Computing and transformation semigroups

Simulation of automata networks

Minimum size and time for universal networks

Complete networks and quasi-parallel simulation

Outline

Computing in terms of semigroups

I could (try to) talk about Turing Machines, automata, the syntactic monoid, etc.

Instead, let us be more practical and look at how an algorithm can be viewed as working in a semigroup.

Let B be a set, then $\mathrm{Tran}(B) = (\{f : B \to B\}, \circ)$ is the transformation semigroup of B.

INSERTION-SORT

Algorithm 1 INSERTION-SORT($x_1, \ldots, x_n$)

```
1: j ← 1
2: for j ← 2 to n do
3:     k ← x_j
4:     i ← j − 1
5:     while i > 0 and x_i > k do
6:         x_{i+1} ← x_i
7:         i ← i − 1
8:     x_{i+1} ← k
```

Everything is finite; for our example let us assume that the list is small and that every variable is only one symbol in [8].

In the transformation semigroup

Sorting is equivalent to computing $g \in \mathrm{Tran}([q]^n)$ which sorts any list of n symbols.

Let $x = (x_1, \ldots, x_n, x_{n+1} = i, x_{n+2} = j, x_{n+3} = k, x_{n+4} = l) \in [q]^{n+4}$.

Each update is the application of one of $F^{(1)}, \ldots, F^{(n+4)} \in \mathrm{Tran}([q]^{n+4})$, where e.g.

$$xF^{(2)} = (x_1, xf_2, x_3, \ldots, x_{n+4})$$

$$xf_2 = \begin{cases} x_1 & \text{if } l = 6, \\ x_{n+3} & \text{if } l = 8, \\ x_2 & \text{otherwise.} \end{cases}$$

Computing g then is done by a composition $F^{(a_1)}F^{(a_2)}\ldots F^{(a_L)}$.

Outline

Sequential ANs

An Automata Network (AN) of size m is a collection of m automata, each having a state $x_i \in [q]$ and an update function $f_i : [q]^m \to [q]$.
The state of the network is $x = (x_1, \ldots, x_m) \in [q]^m$ and the network is viewed as

$$f = (f_1, \ldots, f_m) \in \mathrm{Tran}([q]^m).$$

An AN is sequential if at each time step, only one automaton updates its state. We denote this update by

$$xF^{(i)} = (x_1, \ldots, xf_i, \ldots, x_m).$$

As such, we are interested in the semigroup

$$S_f := \langle F^{(1)}, \ldots, F^{(m)} \rangle \leq \mathrm{Tran}([q]^m).$$

(Henceforth, “network” refers to a sequential AN.)

Why study ANs?

1. Many real-life networks can be modelled as ANs (gene networks, neural networks, epidemics...).
   The dynamics of these systems are difficult to predict, especially when updates are asynchronous.
2. A lot of work has been done on distributed computational models based on similar ideas (Cellular automata, Automata networks from Tchuente or Dőmősi and Nehaniv, etc.).
   We want to understand what can be computed and how.
3. They have interesting connections to coding theory.

Computing by sequential BANs

Let $n \leq m$ and $g \in \mathrm{Tran}([q]^n)$.

The network f simulates (computes) g if there exists $h \in S_f$ s.t.

$$h\mathrm{pr}_{1..n} = \mathrm{pr}_{1..n}g,$$

or in other words, if for all $x \in [q]^m$,

$$(xh)_{1..n} = x_{1..n}g$$

The network f is n-universal if it can simulate any $g \in \mathrm{Tran}([q]^n)$.

Universal network

Theorem

For any $n \geq 1$, there exists an n-universal network.

Proof.

Construction: Take $x_{1..n}$, copy it into $x_{n+1..2n}$, and describe the transformation g.

Basic ingredient for the description: the one-symbol switch. It has two automata s and t, where

$$xf_t = x_s$$
$$xf_s = x_s + 1.$$

The first line initialises the switch to OFF, while the second line flicks it ON and OFF. □

The basic universal network: in more detail

Let $T = q^{nq^n}$ and $\mathrm{Tran}([q]^n) = \{g^{(1)}, \ldots, g^{(T)}\}$.
We use T switches, one for each $g^{(s)}$.

$$xf_v = \begin{cases} x_{n+1..2n} g_v^{(s)} & \text{if } x_{2n+s} \neq x_{2n+T+s} \text{ and} \\ & x_{2n+r} = x_{2n+T+r} \ \forall r \neq s \\ x_v & \text{otherwise.} \end{cases} \qquad (1 \leq v \leq n)$$

$$xf_r = x_{r-n}, \qquad (n+1 \leq r \leq 2n)$$

$$xf_j = x_{T+j}, \qquad (2n+1 \leq j \leq 2n+T)$$

$$xf_k = x_k + 1, \qquad (2n+T+1 \leq k \leq 2n+2T).$$

Proof of universality

We simulate $g^{(s)}$, $1 \le s \le T$, as follows.

Step 1. Make a copy the first n registers: $F^{(n+1)} \circ \cdots \circ F^{(2n)}$.

Step 2. Turn all switches off: $F^{(2n+1)} \circ \cdots \circ F^{(2n+T)}$.

Step 3. Turn the right switch on: $F^{(2n+T+s)}$.

Step 4. Compute $g^{(s)}$: $F^{(1)} \circ \cdots \circ F^{(n)}$.

Or more concisely,

$$h := (F^{(n+1)} \circ \cdots \circ F^{(2n)}) \circ (F^{(2n+1)} \circ \cdots \circ F^{(2n+T)}) \circ F^{(2n+T+s)} \circ (F^{(1)} \circ \cdots \circ F^{(n)})$$

Outline

Minimum size

Theorem

For any $n \geq 1$, there is no n-universal network of size n.
Conversely, there is an n-universal network of size $n+2$.

For size n, there are two main refinements:

1. Unless $q = n = 2$, there is an network of size n which can simulate any permutation of $[q]^n$ (i.e. $\mathrm{Sym}([q]^n) \leq S_f$) [Cameron, Fairbairn, G. 14].
2. There is no network of size n which can simulate all singular transformations (i.e. $\mathrm{Sing}([q]^n) \not\leq S_f$).

The construction of size $n+2$ is based on Memoryless Computation (MC) [Burckel, Gioan, Thomé 09, G. and Riis 14].

A bit on Memoryless Computation

In MC we have n registers, each gets updated one at a time.
Main difference with networks: the update functions $f_i : [q]^n \to [q]$ can change over time.

Lemma

Any transformation can be simulated in MC, where each automaton has at most two update functions.

Corollary

There exists an n-universal network of size $n+2$.

Minimum time

Updating one automaton provides one symbol of information.
We need nq^n symbols to describe g.
Thus, the time is at least nq^n...or is it?

Theorem

There exists an n-universal network with time $q^n + O(n)$.

Idea of the proof. Encode $g_1, \ldots, g_n$ successively as an error added to a codeword in a Hamming code. The initial encoding takes time q^n, adding and removing the errors takes time $O(n)$.

Conjecture

Any n-universal network has time at least q^n.

Outline

Complete networks

An network is n-complete if it can simulate any sequence of transformations (with possible gaps in between).

Formally, if for any sequence $g^{(1)}, \ldots, g^{(\ell)} \in \mathrm{Tran}([q]^n)$, there exist $h^{(1)}, \ldots, h^{(\ell)} \in S_f$ such that for all $1 \leq i \leq \ell$,

$$\mathrm{pr}_{1..n} g^{(i)} = h^{(1)} \cdots h^{(i)} \mathrm{pr}_{1..n}.$$

The first example of an n-universal network was actually n-complete.

Minimum size and time for complete networks

Theorem

Any n-complete network has size at least $2n$.
Conversely, there exists an n-complete network of size $2n+3$.

The minimum time is defined over any ordering $g^{(1)}, \ldots, g^{(q^{nq^n})}$ of $\mathrm{Tran}([q]^n)$.

Theorem

Any n-complete network has minimum time at least q^{nq^n}.
Conversely, there exists an n-complete network with minimum time $(1+o(1))q^{nq^n}$.

Parallel simulation

Let us consider parallel ANs, where all automata update their states at the same time. This time, we are interested in

$$\langle f \rangle = \{\mathrm{id}, f, f^2, \ldots, f^k\}.$$

Theorem

For any $n \geq 1$, there is no n-universal parallel AN.

Therefore, asynchronism is compulsory for automata networks!

Theorem

Conversely, for any $n \geq 1$, there exists an n-complete quasi-parallel network, where automata 1 to $m-1$ operate in parallel and automaton m is only updated once.