Constant speed-up on one dimensional calculators

Olivier Heen
LITP-IBP, Université Paris 7,
2 Place Jussieu, 75251 Paris, FRANCE.

1 Recognizers and calculators

A simple cellular automaton is a finite line of cells with nearest neighborhood and a quiescent state $\sharp$. Language *recognizers* stop when the leftmost cell enters a special state (either accepting or rejecting) which is the result of the computation. Function *calculators* stop when the leftmost cell enters a terminating state, and the result is the whole configuration. In this case, the output can be stable or synchronized.

2 Elementary complexity results

There is a large panel of complexity results for recognizers such as constant or linear speed-up theorems.

Theorem 1 (speed-up for recognizers [IKM85, MR92, Hee95]) *Any recognizer with s states working in time $u \mapsto |u| + t(u)$ can be simulated by a recognizer with $\mathcal{O}(s^{k+1})$ states working in time $u \mapsto |u| + \sup\{t(u) - k, 0\}$ and by a recognizer with $\mathcal{O}(s^k)$ states working in time $u \mapsto |u| + \lceil t(u)/k \rceil$ (where k is a positive integer).*

For calculators, however, there is no general linear speed-up technique (although there might be some for "slow enough" synchronized calculators). Constant speed-up is possible [CC84] but requires an exponential growth of the set of states.

3 New speed-up methods for calculators

We discuss new techniques to achieve the speed-up with a polynomial increasing of the set of states (as it is the case for recognizers).

From a calculator with s states working in time $u \mapsto |u| + 1 + t(u)$ we build a new calculator with $\mathcal{O}(s^{2k+1})$ states working in time $u \mapsto |u| + 1 + \sup\{t(u) - k, 0\}$, by using the so called *redunding* method and a minimal time synchronization from both ends of the line.

Then we present a better method leading to $\mathcal{O}(s^{k+1})$ states, without synchronization. The figure below illustrates this method with the conventions that ct is the state of the c^{th} cell at time t in the initial computation and that $a = 10$.

♯	00	10	20	30	40	50	60	70	♯
♯	• ♯ 01	11	21	31	41	51	61	70 ♯ •	♯
♯	♯ 02	• 02 12	22	32	42	52	62 72 •	72 ♯	♯
♯	♯ ♯	03 13	• 13 23	33	43	53 63 •	63 73	♯ ♯	♯
♯	$	05 04	14 24	• 24 34	44 54 •	54 64	74 75	$	♯
♯	$	06	16 15	+ 25 35	45 55 +	65 66	76	$	♯
♯	$	07	+ 17 26	27 37	47 57	56 67 +	77	$	♯
♯	$	+ 08 17	18 28	38	48	58 68	67 67 +	$	♯
♯	+ ♯ 08	09 19	29	39	49	59	69	78 ♯ +	♯
♯	+ ♯ 0a	1a	2a	3a	4a	5a	6a	7a ♯ +	♯

From an information theoretical point of view, there seems to be no obstacle to constant speed-up with less than a polynomial growth of the set of states. Althougth our method uses product states, they appear very sparsely in the space-time diagram (that is with density $\mathcal{O}(|u|)$).

References

[CC84] C. Choffrut and K. Culik. On real-time cellular automata and treillis automata. *Acta Informatica*, 21:393–407, 1984.

[Hee95] O. Heen. Linear speed-up without synchronization. Rapport LITP 57, Institut Blaise Pascal, 1995.

[IKM85] O. H. Ibarra, S.M. Kim, and S. Moran. Sequential machine caracterization of treillis and cellular automata and application. *SIAM J. Comput.*, 14:426–427, 1985.

[MR92] J. Mazoyer and N. Reimen. A linear speed-up theorem for cellular automata. *Theoretical Computer Science*, 101:59–98, 1992.