About 4-States Solutions to the Firing Squad Synchronization Problem

Hiroshi Umeo[1], Jean-Baptiste Yunès[2], and Naoki Kamikawa[1]

[1] University of Osaka Electro-Communication
Neyagawa-shi, Hatsu-cho, 18–8. 5172–8530, Osaka, Japan
umeo@cyt.osakac.ac.jp
[2] LIAFA - Université Paris 7 Denis Diderot,
175, rue du chevaleret. 75013 Paris - France
Jean-Baptiste.Yunes@liafa.jussieu.fr

Abstract. We present some elements of a new family of time-optimal solutions to a less restrictive firing squad synchronization problem. These solutions are all built on top of some elementary algebraic cellular automata. Thus, this gives a very new insight on the problem and a more general way of computing on cellular automata.

1 The Firing Squad

As an old inverse problem, the Firing Squad Synchronization Problem (FSSP for short – see [6]) proposes to design a linear cellular automata $\mathcal{A} = (\mathcal{Q}, \delta)$, with a finite set of states $\mathcal{Q} = \{\bullet, \mathtt{A}, \mathtt{B}, \ldots, x\}$, and the transition function $\delta : \{\$\} \cup \mathcal{Q} \times \mathcal{Q} \times \{\$\} \cup \mathcal{Q} \rightarrow \mathcal{Q}$ (\$ being an external state) which is able to *synchronize* **any** chain of n cells so that there exists a distinguished state $f \in \mathcal{Q}$ so that:

- from the **starting configuration** at time 0, which is of the form $\mathtt{A}\bullet^{n-1}$;
- it reaches the **synchronized configuration** $\mathtt{f}^n$ after $\mathcal{T}(n)$ transitions;
- so that the state $\mathtt{f}$ never appears before the $\mathcal{T}(n)$-th step.

Besides the fact that the original problem has been studied for a long time (see [1,4,5,11]), it is interesting to note that recent results may give a very new insight on the problem (see [7,8,9,10,13,14,15]). First and independently, Yunès (see [13,14]), Settle & Simon (see [7]) and Umeo *et al.* (see [9]) designed non minimal-time solutions with very few states; since Balzer (see [1] in is known that there is no optimum-time solution to the original problem with 4-states, and since Mazoyer (see [4]) we know that there exists a 6-states solution and wether there exists a 5-states solution).

And a recent promising line of research, started by Umeo and Yunès (see [8,10,15]) is to look for solutions able to synchronize an infinite number but not all lines. That less restrictive problem is very surprisingly solvable with only 4-states (no gap remains as Yunès proved that there is no 3-states solution to that less restrictive problem). What is remarkable is that with 5 states (see [10]) we can use common construction as signal and collision schema, but 4-states solutions are all built using some elementary algebra.

H. Umeo et al. (Eds): ACRI 2008, LNCS 5191, pp. 108–113, 2008.

Here we present some of these solutions in only 4-states which synchronizes every line whose length is a power of 2

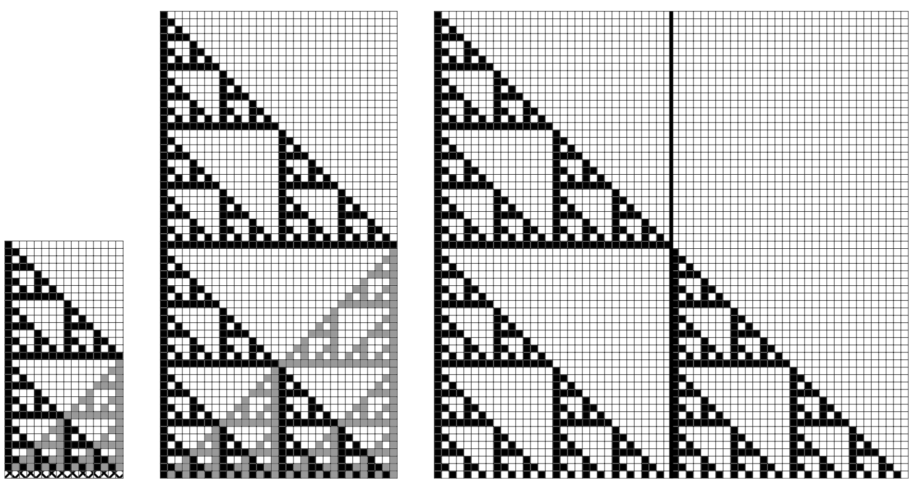
2 The Rule 60 Based Solution

A first construction is based on Wolfram's rule 60 (see [12], page 1035). Wolfram stated that running rule 60 (Pascal's triangle modulo 2) on a configuration of length a power of 2 where the left end cell is 1 leads to something that looks like a synchronization. But one can easily note that rule 60 is not a solution to the problem and we will see that we can use it to construct something able to synchronize every power of 2.

The key idea is to use a simple folding of the space-time diagram of rule 60 running on a line of length 2^{n+1}, which gives a space-time diagram of a 4-states cellular automata running on a line of length 2^n. That cellular automata exploits an interleaving property of rule 60 configurations such that a synchronization is obtained at the right time ($2n$ for a line of length n).

The figure 1 illustrates how the construction is made. The figure 1(a) shows a full run of the solution on a line of length 16, and figures 1(b)and 1(c) exhibits how the *folding* is used. Table 1(a) contains the transition function of our solution.

For a proof of correctness the reader must refer to [15].

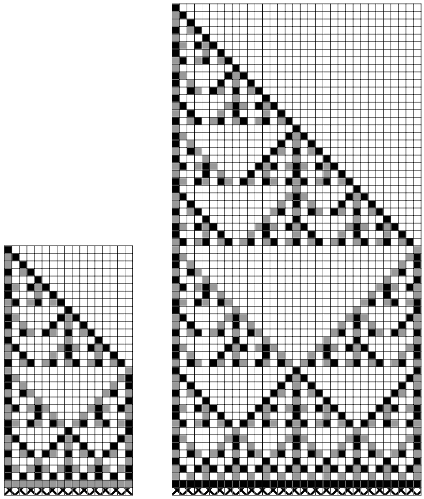


(a) $n = 16$ (b) $n = 32$ (c) unfolded $\rightarrow n = 64$

Fig. 1. Folding of a Pascal's triangle modulo 2

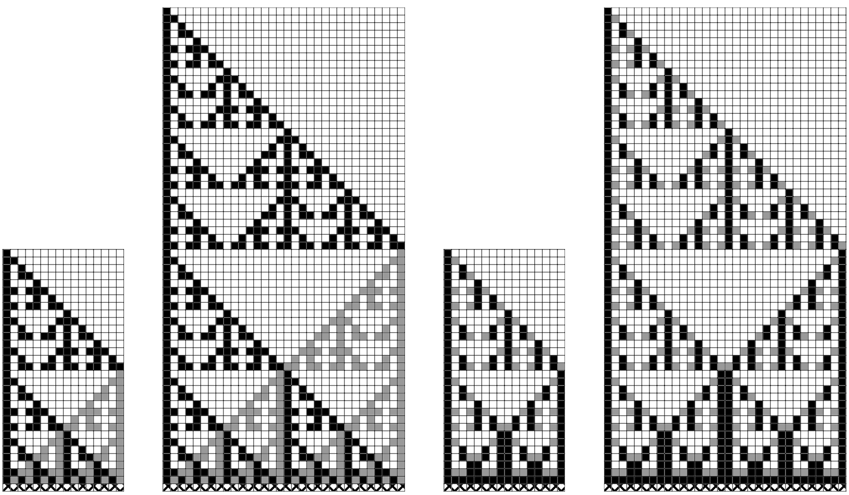
3 The Rule 150 Based Solutions

All the following constructions use Wolfram's rule 150 as a base for their design.



(a) $n = 17$ (b) $n = 33$

Fig. 2. Strict optimum-time solution



(a) $n = 16$ (b) $n = 32$ (c) $n = 16$ (d) $n = 32$

Fig. 3. Rule 150 based solution

3.1 Strict Optimum-Time Solution

The figure 2 illustrates how that solution works and the table 1(b) gives the transition function of it. The reader must note that this construction synchronizes every line of length $N = 2^n + 1$ at time $2N - 1$ ($2N - 2$ steps), so this is a strict

Table 1. The transition functions

(a) Rule 60. Synchronizes lengths 2^n

•	$	•	A	B
$		•		
•	•	•	•	B
A	A	A	A	
B		•		B

A	$	•	A	B
$		A	A	C
•		A	A	
A	B	•	•	B
B		A	A	C

B	$	•	A	B
$				
•	B	B	B	•
A	C		C	A
B	B	B	B	•

(b) Rule 150. Synchronizes lengths $2^n + 1$

•	$	•	A	B
$				
•	•	•	A	B
A	B	A	B	•
B		B	•	A

A	$	•	A	B
$		B	C	A
•	B	B		•
A	C		C	
B	A	•		A

B	$	•	A	B
$		A	B	C
•	A	A	•	
A	B	•	B	
B	C			C

(c) Rule 150. Synchronizes lengths 2^n

•	$	•	A	B
$		•		
•	•	•	A	B
A	A	A	•	
B		B		•

A	$	•	A	B
$		A	A	C
•		A	•	
A	B	•	A	B
B		A	A	C

B	$	•	A	B
$				
•	B	B	B	•
A	C		C	A
B	B	•	B	B

(d) Rule 150. Synchronizes lengths 2^n

•	$	•	A	B
$		•		
•	•	•	B	A
A	B	B	•	
B		A		•

A	$	•	A	B
$		A	C	A
•	A	A	A	•
A	C	A	C	A
B	A	•	A	A

B	$	•	A	B
$				
•			•	
A	A	•		A
B			A	

optimum-time solution, as can be opposed to the previous construction which synchronizes in $2N - 1$ steps - one more extra step.

Here the key idea is also to use a folding of some space-time diagram, but such that states A and B used in the unfolded part are respectively projected to B and A in the folded part.

3.2 Simple Folding Variants of Rule 150 Based Solution

The figure 3 illustrates two variants of the preceding solution and the tables 1(c) and 1(d) contain their respective transition functions. These solutions are also based on Wolfram's rule 150 as one can observe that the *shadow* of their space-time diagram is exactly the rule 150. What we call the shadow of a space-time diagram is what is obtained identifying, in a given space-time diagram, all non quiescent states into a single one. The picture obtained is something which uses only two states (the original quiescent and the shadow state). Of course, in the general case, the shadow diagram is not a 2-states cellular automata space-time diagram, but it can be in some particular cases (as the cases illustrated here). By example, identifying black and grey states in the upper part of figures 2(a), 2(b), 3(c), and 3(d) give the space-time diagram of rule 150 (upper part of figures 3(a) and 3(b)).

The first solution is built as the first solution of this paper (rule 60 based solution). The space-time diagram for a line of length 2^n is obtained by a simple folding of the original space-time diagram of Wolfram's rule 150 running on a line of length 2^{n+1}.

The second solution is also built using our simple folding but combined with a parity cell position coloring. That coloring doesn't perturb the property used to obtain the synchronization.

4 Conclusion

We were able to construct different 4-states solution to the firing squad which synchronizes every line of length 2^n or $2^n + 1$. As their are all based on some simple algebraic computation, we think that it would be possible to find a simple formal proof of their correctness as it has already been done to the first of them (see [15]).

All theses solutions have a very low Kolmogorov complexity, the three first presented solutions use only 33 transitions and the last uses only 30 transitions.

One can note that the number of non quiescent cells is not in the order of $n.log(n)$ for a line of length n, neither it is in the order of n^2, but something in between. That number is of course directly related to the Hausdorff dimension of the fractal built.

These solutions all belong to a new family of solutions based on algebraic computation, may be a new way of **programming on cellular automata** (algebraic cellular automata have already been studied but in the field of discrete dynamical systems).

Moreover, if these constructions doesn't synchronize every line, it is worth notice that a recent reconstruction of Goto's first FSSP solution (see [3,16]) can naturally be based on.

References

1. Balzer, R.: An 8-State Minimal Time Solution to the Firing Squad Synchronization Problem. Information and Control 10, 22–42 (1967)
2. Gerken, H.-D.: Über Synchronisations - Probleme bei Zellularautomaten. Diplomarbeit, Institut für Theoretische Informatik. Technische Universität Braunschweig (1987)
3. Goto, E.: A minimum time solution of the firing squad synchronization problem. Courses Notes for Applied Mathematics 298, 52–59 (1962)
4. Mazoyer, J.: A Six-State Minimal-Time Solution to the Firing Squad Synchronization Problem. Theoretical Computer Science 50, 183–238 (1987)
5. Minsky, M.: Computation: Finite and Infinite Machines. Prentice-Hall, Englewood Cliffs (1967)

6. Moore, E.F.: The Firing Squad Synchronization Problem. In: Moore, E.F. (ed.) Sequential Machines. Selected Papers, pp. 213–214. Addison-Wesley, Reading (1964)
7. Settle, A., Simon, J.: Non-Minimal Time Solutions for the Firing Squad Synchronization Problem. Technical Report 97-08, University of Chicago (1997)
8. Umeo, H., Kamikawa, N.: A 4-states solution to the firing squad based on Wolfram's rule 150. Private communication (2007)
9. Umeo, H., Maeda, M., Hongyo, K.: A Design of Symmetrical Six-State 3n-Steps Firing Squad Synchronization Algorithms and Their Implementations. In: El Yacoubi, S., Chopard, B., Bandini, S. (eds.) ACRI 2006. LNCS, vol. 4173, pp. 157–168. Springer, Heidelberg (2006)
10. Umeo, H., Yanagihara, T.: A smallest five-state solution to the firing squad synchronization problem. In: Durand-Lose, J., Margenstern, M. (eds.) MCU 2007. LNCS, vol. 4664, pp. 291–302. Springer, Heidelberg (2007)
11. Waksman, A.: An Optimum Solution to the Firing Squad Synchronization Problem. Information and Control 9, 66–78 (1966)
12. Wolfram, S.: Cellular Automata and Complexity: Collected Papers. Westview Press (2002)
13. Yunès, J.-B.: Seven States Solutions to the Firing Squad Synchronization Problem. Theoretical Computer Science 127(2), 313–332 (1994)
14. Yunès, J.-B.: An Intrinsically non Minimal-Time Minsky-like 6-States Solution to the Firing Squad Synchronization Problem. RAIRO ITA/TIA 42(1), 55–68 (2008)
15. Yunès, J.-B.: A 4-states Algebraic Solution to Linear Cellular Automata Synchronization. International Processing Letters (2008), doi: 10.1016/j.ipl.2008.01.009
16. Yunès, J.-B.: Goto's construction and Pascal's triangle: new insights into cellular automata synchronization. In: Durand, B. (ed.) JAC 2008, Uzès, France, April 21–25, pp. 195–203. MCCME Publishing House Moscow (2008)