A Trade-off in Firing Squad Synchronization Protocols for Communication-restricted Cellular Automata

Hiroshi Umeo

Univ. of Osaka Electro-Communication,

Osaka, Japan

In this talk

We study **a trade-off between internal states and inter-cell communication bits** in firing squad synchronization protocols for **communication-restricted cellular automata**.

Several state-efficient bit-transfer-based synchronization protocols are proposed.

It is shown that there exists a 1-state $\mathsf{CA}_{5-\mathsf{bit}}$ that can synchronize any n cells in $2n-2$ optimum-step.

Conventional (normal) CA vs. k-Bit CA

In the long history of the study of conventional CA,

- Number of internal states Q of each cell is finite,
- Local state transition rule δ is defined in a such way that $\delta : Q^3 \rightarrow Q$.

Thus, the amount of communication bits exchanged at one step between neighboring cells has been assumed to be **O**(1)-**bit**.

Bit-information exchanged between inter-cell has been hidden behind the definition of **conventional** automata-theoretic finite state description of CA.

k-Bit CA

A restricted class of cellular automata: $\mathbf{CA}_{k-\text{bit}}$, $k \geq 1$, whose inter-cell communication at one step is restricted to k-**bit**.

- The number of internal states of $\mathbf{CA}_{k-\text{bit}}$ is assumed to be **finite** in a usual way.

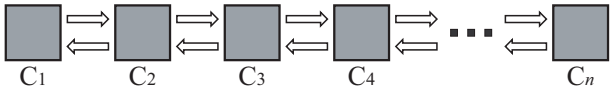
- The next state of each cell is determined by the **present state of itself and two binary k-bit inputs** from its left and right neighbor cells.

A 1-bit CA model: $\mathbf{CA}_{1-\text{bit}}$, where $k = 1$, can be thought to be one of the *most powerless and simplest models* in a variety of CAs.

1-D k-Bit CA

• Each $\mathsf{C}_i (i \geq 1)$, except C_1 and C_n, is connected with its left and right neighbor cells **via a left/right one-way communication link**.

• Each one-way communication link can transmit k-bit at each step in each direction.



1-D $\mathsf{CA}_{k-\mathsf{bit}}$

• State transition function δ: $\{0,1\}^k \times Q \times \{0,1\}^k \rightarrow \{0,1\}^k \times Q \times \{0,1\}^k$, where Q is the set of internal states.

By **encoding** each state in Q with a binary sequence of length $\lceil \log_2 |Q| \rceil$, **sending** the sequences sequentially bit by bit in each direction via each one-way communication link, **receiving** them bit by bit again, and by **decoding** them into their corresponding states in Q, the $\mathsf{CA}_{1-\mathsf{bit}}$ can simulate one step of N in $\lceil \log_2 |Q| \rceil$ steps.

[Observation] Let N be any *conventional* cellular automaton with time complexity $T(n)$. Then, there exists a $\mathsf{CA}_{1-\mathsf{bit}}$ which can simulate N in $kT(n)$ steps, where k is a positive constant integer such that $k = \lceil \log_2 |Q| \rceil$ and Q is the set of N's states.

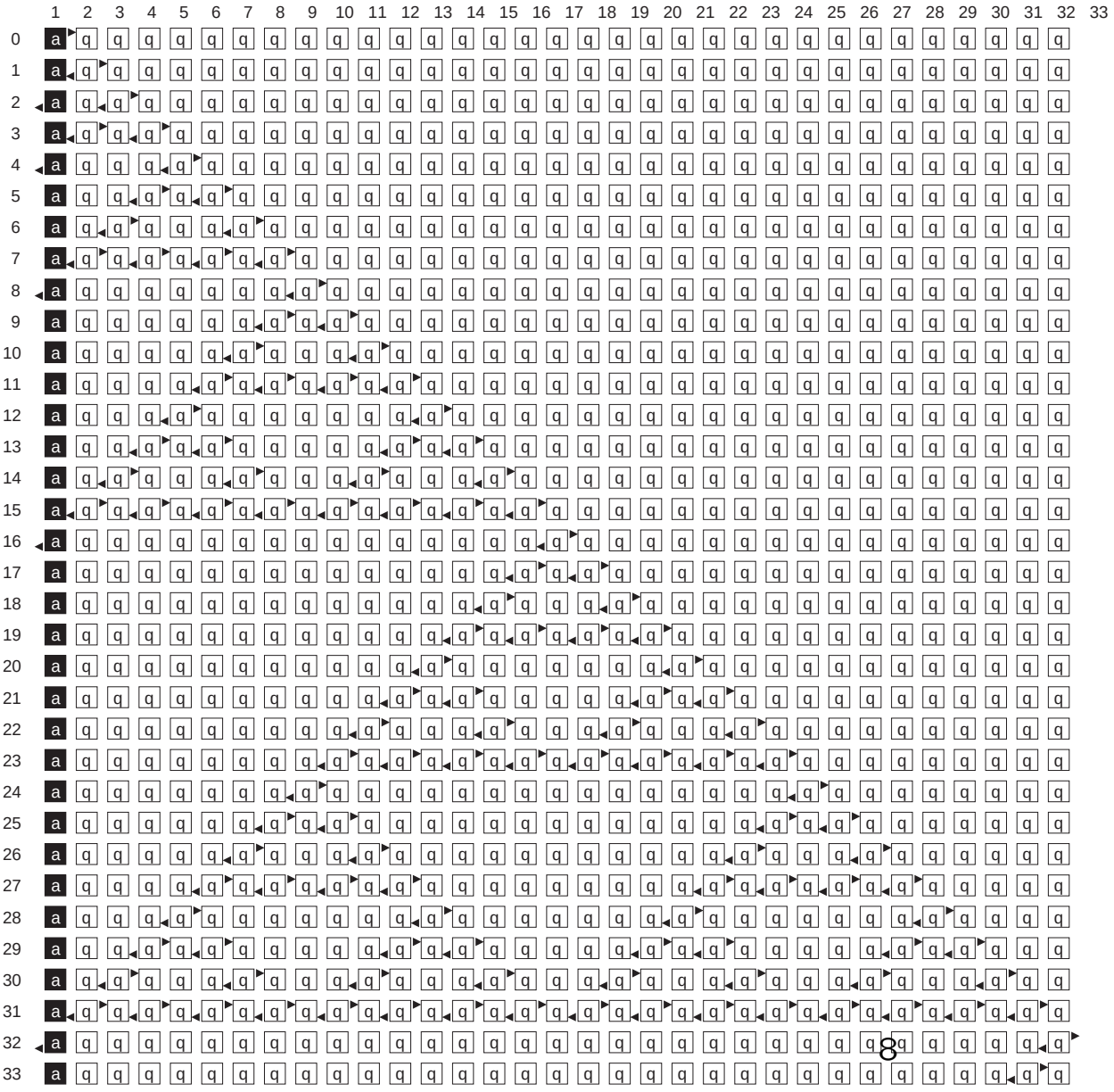
Real-time Sequence Generation Problem on $\mathrm{CA}_{1-\mathrm{bit}}$

In spite of its simplicity, the $\mathrm{CA}_{1-\mathrm{bit}}$ can generate a variety of context-sensitive sequences given below.

[Theorem] [Umeo et al. (2004)] An infinite sequence $\{2^n | \, n = 1, 2, 3, ...\}$ can be generated in real-time by a 1-state $\mathrm{CA}_{1-\mathrm{bit}}$.

[Theorem] [Umeo and Kamikawa (2002)] **Fibonacci** and **Prime** sequences can be generated in real-time by $\mathrm{CA}_{1-\mathrm{bit}}$ with 9-/34-state, respectively.

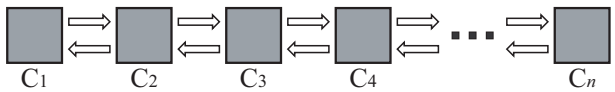
Snapshots for real-time generation of $\{2^n | n = 1, 2, 3, ...\}$ on a 1-state 6-rule CA_{1-bit}.



FSSP: Firing Squad Synchronization Problem on 1-D Arrays

Given an array of n identical cellular automata, including a *general* at the left end that is activated at time $t = 0$, we want to design the automata such that, **at some future time**, all the cells will **simultaneously** and, **for the first time**, enter a special *firing* state.

• The tricky part of the problem is that **the same kind of soldier** having **a fixed number of states** must be synchronized, regardless of the length n of the array.



- The problem itself is interesting as a mathematical puzzle and a good example of **recursive divide-and-conquer strategy** operating **in parallel**.

- The problem has been referred to as achieving **macro-synchronization** in **micro-synchronization** system and **realizing a global synchronization using only local information exchange**.

- Synchronization of general network is a **computing primitive** of parallel and distributed computations.

A Brief History of the Developments of Optimum-step Firing Squad Synchronization Algorithms

The problem known as the *firing squad synchronization problem* was devised in 1957 by J. Myhill, and first appeared in print in a paper by E. F. Moore [1964].

This problem has been widely circulated, and has attracted much attention.

The firing squad synchronization problem first arose in connection with **the need to simultaneously turn on all parts of a self-reproducing machine**.

The problem was first solved by J. McCarthy and M. Minsky who presented a **3n-step** algorithm.

In 1962, the first **optimum-time**, i.e. $(2n-2)$-step, synchronization algorithm was presented by **Goto [1962]**, with each cell having several thousands of states.

Waksman [1966] presented a **16-state** optimum-time synchronization algorithm. Afterward, **Balzer [1967]** and **Gerken [1987]** developed an **eight-state** algorithm and a **seven-state** synchronization algorithm, respectively, thus decreasing the number of states required for the synchronization.

Mazoyer [1987] developed a **six-state** synchronization algorithm which, at present, is the algorithm having the fewest states.

Complexity Measures and Properties in Synchronization Algorithms

- Time Complexity
- Number of Internal States
- Number of transition rules
- State Change Complexity
- Recursive vs. Non-Recursive Algorithms
- One-Sided vs. Two-Sided Recursive Algorithms
- O(1)-Bit vs. k-Bit Communication CA

Number of internal states in FSSP

The following **three** distinct states:

- **quiescent** state `Q` such that `Q Q Q` $\longrightarrow$ `Q`
- **general** state: `G`, and
- **firing** state: `F`

are required in order to define the firing squad synchronization problem on conventional cellular automaton.

[Theorem] [Balzer 1967],[Sanders 1994],[Berthiaume et al. 2004] There is **no four-state conventional CA** that can synchronize n cells.

State Change Complexity on Conventional CA

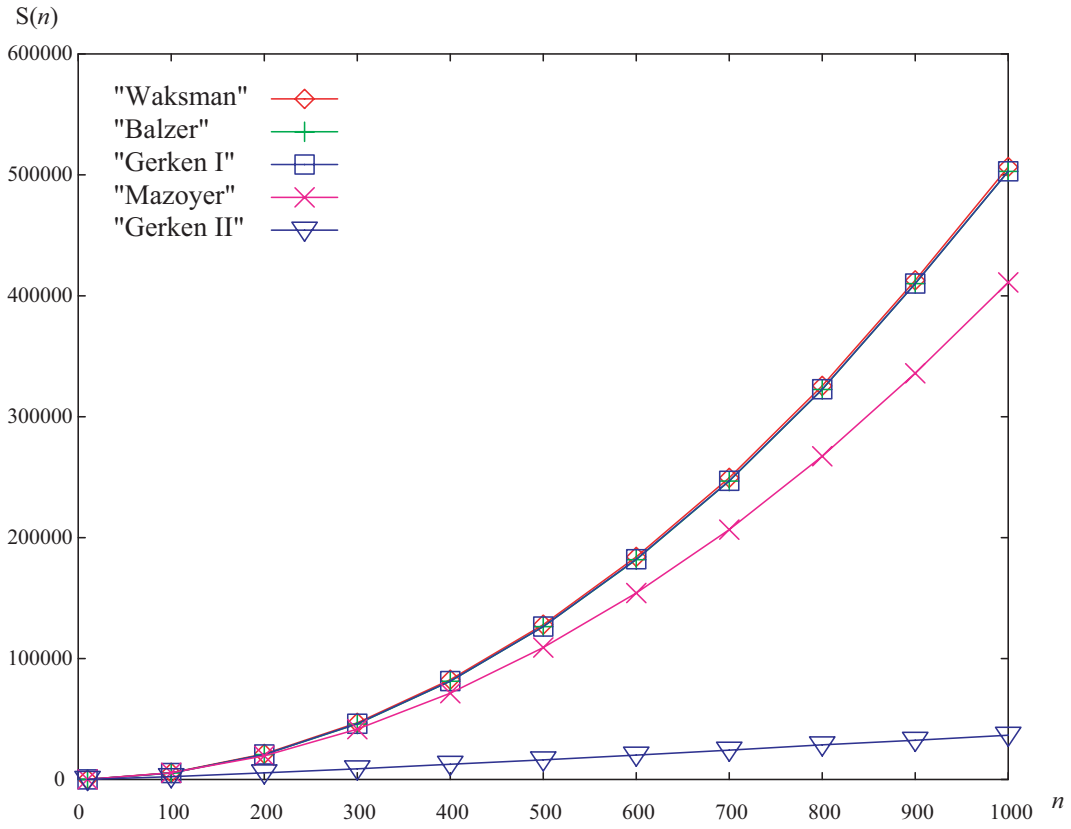
$S(n)$: Total number of state changes needed for synchronizing n cells.

Vollmar [1982] introduced a state-change complexity in order to measure the efficiency of cellular algorithms and showed that $\Omega(n \log n)$ state changes are required for the synchronization of n cells in $(2n-2)$ steps.

[Theorem] $[Vollmar(1982)]$ $\Omega(n \log n)$ state-change is necessary for synchronizing n cells in $(2n-2)$ steps.

[Theorem] [$Gerken$(1987), $Umeo\ et\ al.$ (2004)] Each optimum-time synchronization algorithm developed by Balzer [1], Gerken [2], Mazoyer [6] and Waksman [18] has an $O(n^2)$ state-change complexity, respectively.

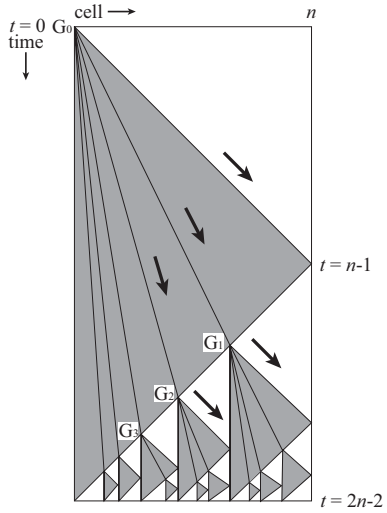
[Theorem][$Gerken$(1987), $Umeo\ et\ al.$ (2004)] Gerken's synchronization algorithm II and Goto's synchronization algorithm have a $\Theta(n \log n)$ state-change complexity, respectively.



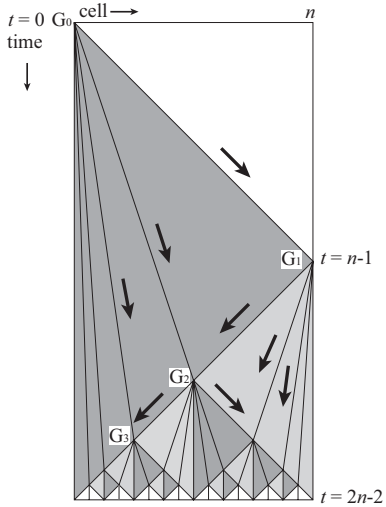
A comparison of state-change complexities in optimum-time synchronization algorithms.

One-Sided vs. Two-Sided Recursive Algorithms

• If all of the recursive calls for the synchronization are issued by *Generals* located at **one** (both **two**) end(s) of partitioned cellular spaces for which the *General* works, the synchronization algorithm is said to have **one-sided** (**two-sided**) recursive property.



Mazoyer type: One-sided recursive synchronization scheme



Waksmann-Balzer type: Two-sided recursive synchronization scheme

A Comparison of Quantitative Aspects of Optimum-Time Synchronization Algorithms

Algorithm	# of states	# of transition rules	State change complexity
Goto [1962]	many thousands	—	$\Theta(n \log n)$
Waksman [1966]	16	3216(**202***)	$O(n^2)$
Balzer [1967]	8	182 (**165** *)	$O(n^2)$
Gerken I [1987]	7	118 (**105***)	$O(n^2)$
Mazoyer [1987]	6	120 (**119***)	$O(n^2)$
Gerken II [1987]	155	2371	$\Theta(n \log n)$

Note: The "*" symbol in parenthesis shows the reduction of transition rules made in this paper.

A Comparison of Qualitative Aspects of Optimum-Time Synchronization Algorithms

Algorithm	One-/two-sided	Recursive/non-recursive	# of signals
Goto [1962]	—	non-recursive	finite
Waksman [1966]	two-sided	recursive	infinite
Balzer [1967]	two-sided	recursive	infinite
Gerken I [1987]	two-sided	recursive	infinite
Mazoyer [1987]	one-sided	recursive	infinite
Gerken II [1987]	two-sided	recursive	finite

Table 2. A qualitative comparison of optimum-time firing squad synchronization algorithms.

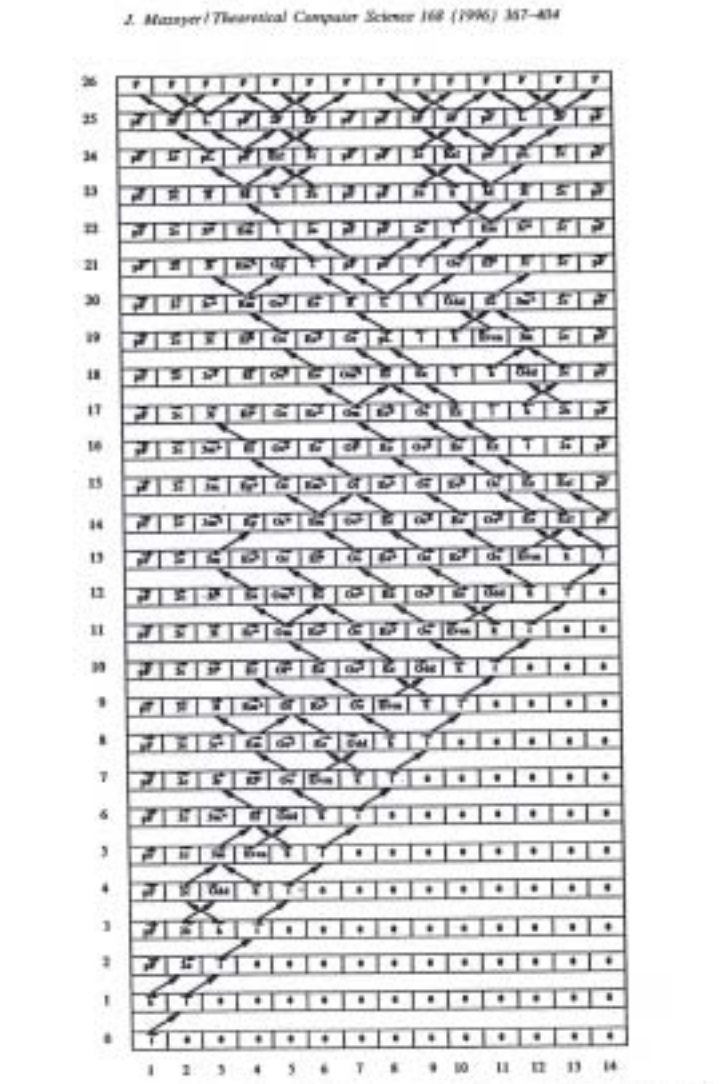
Firing Squad Synchronization Problem on $\mathbf{CA}_{1-\text{bit}}$

Mazoyer [1996] and Nishimura, Sogabe and Umeo [2000] designed an optimum-step firing squad synchronization algorithm on $\text{CA}_{1-\text{bit}}$, where $2n-2$ steps are required for synchronizing n cells on 1-D array and the general is located at the left end of the array.

[Theorem] [Mazoyer (1996)], [Nishimura, Sogabe and Umeo (2000)] There exists a $\text{CA}_{1-\text{bit}}$ which can synchronize n cells with the general on the left end in $2n-2$ steps.

Mazoyer's Optimum-Time Synchronization on $\mathbf{CA}_{1-\text{bit}}$

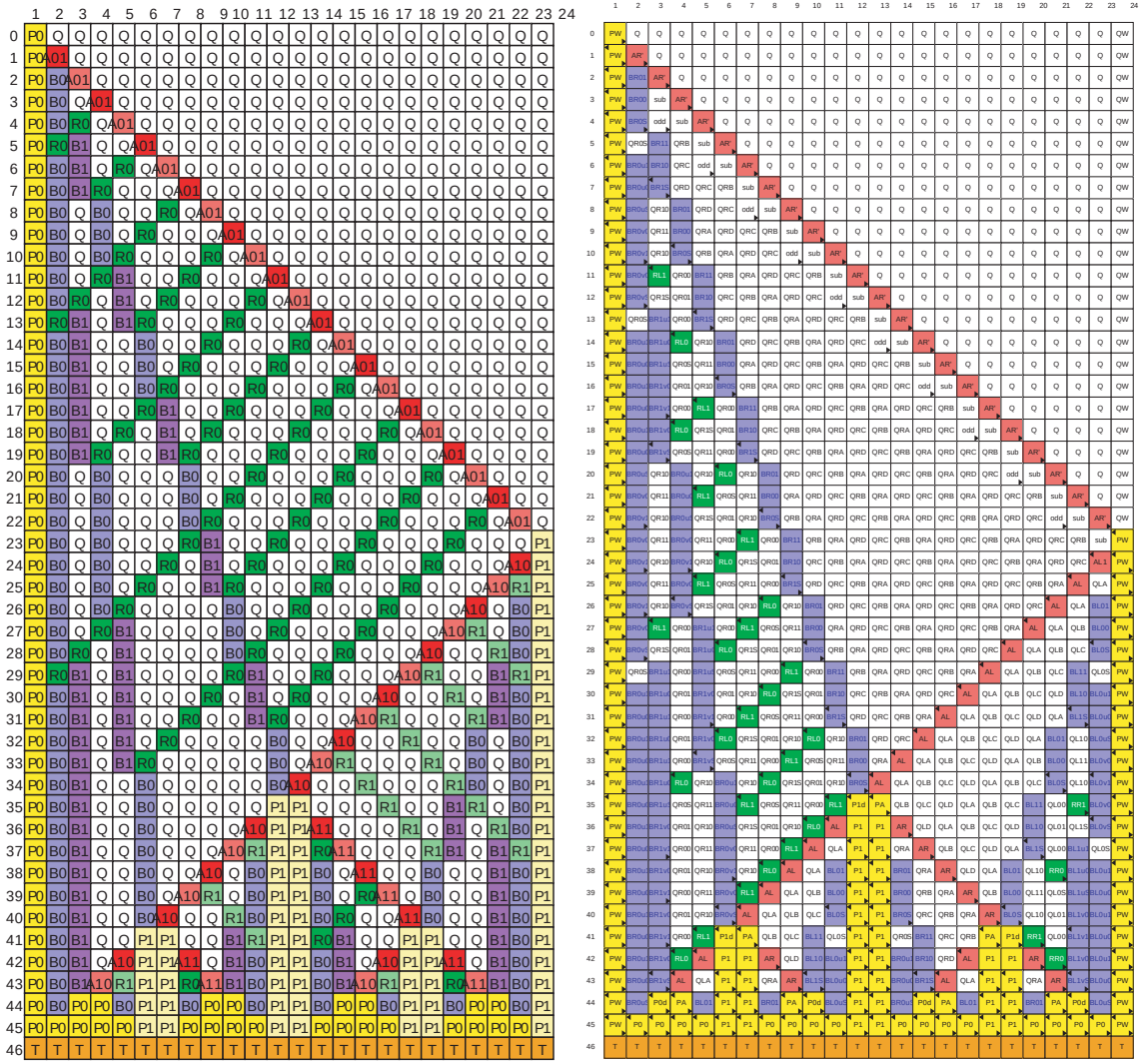
Snapshots for 58-state, $(2n-2)$-step firing squad synchronization algorithm operating on $\text{CA}_{1-\text{bit}}$.



$\mathcal{P}_0$: An Optimum-Time Synchronization Protocol on $\mathrm{CA}_{1-\mathrm{bit}}$

- Our construction: The $\mathrm{CA}_{1-\mathrm{bit}}$ constructed has 78 internal states and 208 transition rules.

Snapshots for 78-state, $(2n-2)$-step firing squad synchronization algorithm operating on $\mathrm{CA}_{1-\mathrm{bit}}$ with 24 cells.



Motivation of Our Study

[Theorem] Let N be any s-state *conventional* cellular automaton. Then, there exists an s-state $\mathsf{CA}_{k-\mathsf{bit}}$ which can simulate N in *real time*, where k is a positive integer such that $k = \lceil \log_2 s \rceil$.

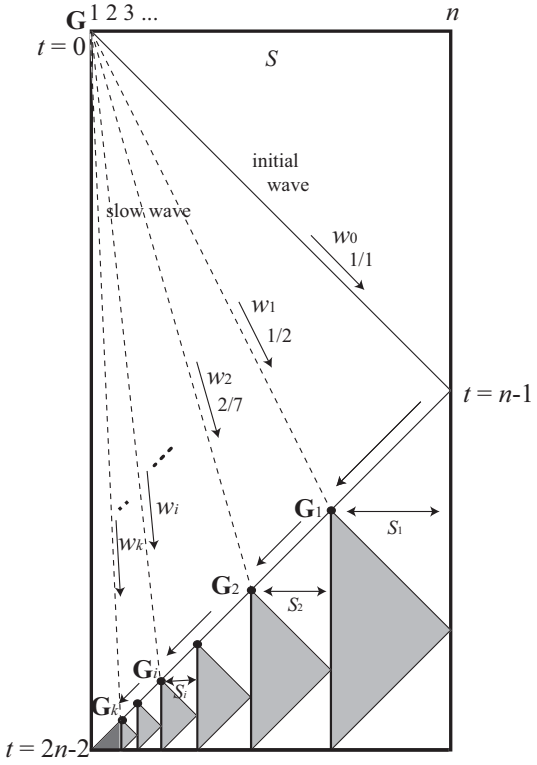
[Theorem] There exists a 6-state $\mathsf{CA}_{3-\mathsf{bit}}$ that can synchronize any n cells in $2n - 2$ steps.

Questions:

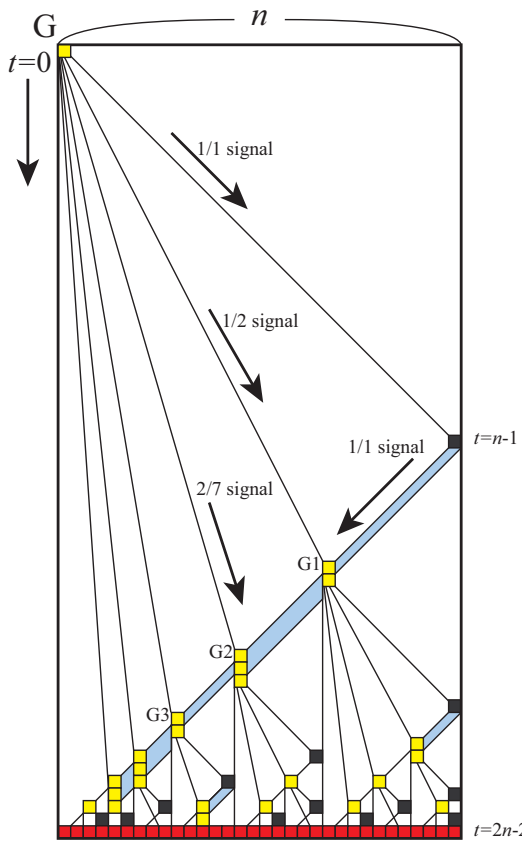
- Can we reduce the number of states?

- Is there any trade-off between internal states and inter-cell communication bits transferred?

An Overview of Mazoyer's Optimum-time Synchronization Algorithm



A time-space diagram.

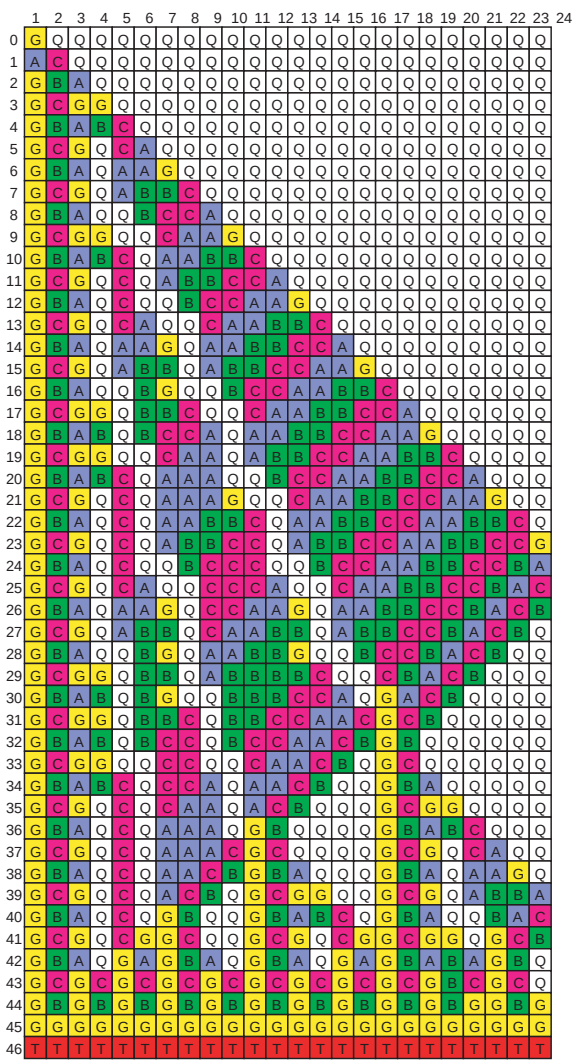


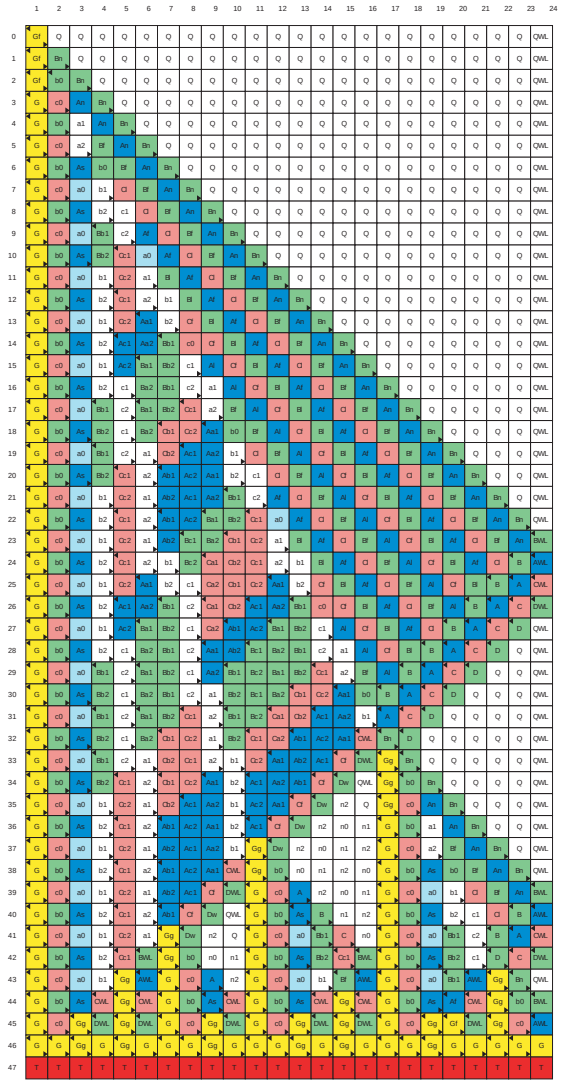
A delay scheme.

$\mathcal{P}_1$: A Non-Optimum-Time Synchronization Protocol on $\mathrm{CA}_{1-\mathrm{bit}}$

[Theorem] There exists a 54-state $\mathrm{CA}_{1-\mathrm{bit}}$ that can synchronize any n cells in $2n-1$ steps.

Snapshots for 54-state $(2n-1)$-step firing squad synchronization algorithm operating on $\mathrm{CA}_{1-\mathrm{bit}}$ with 24 cells.

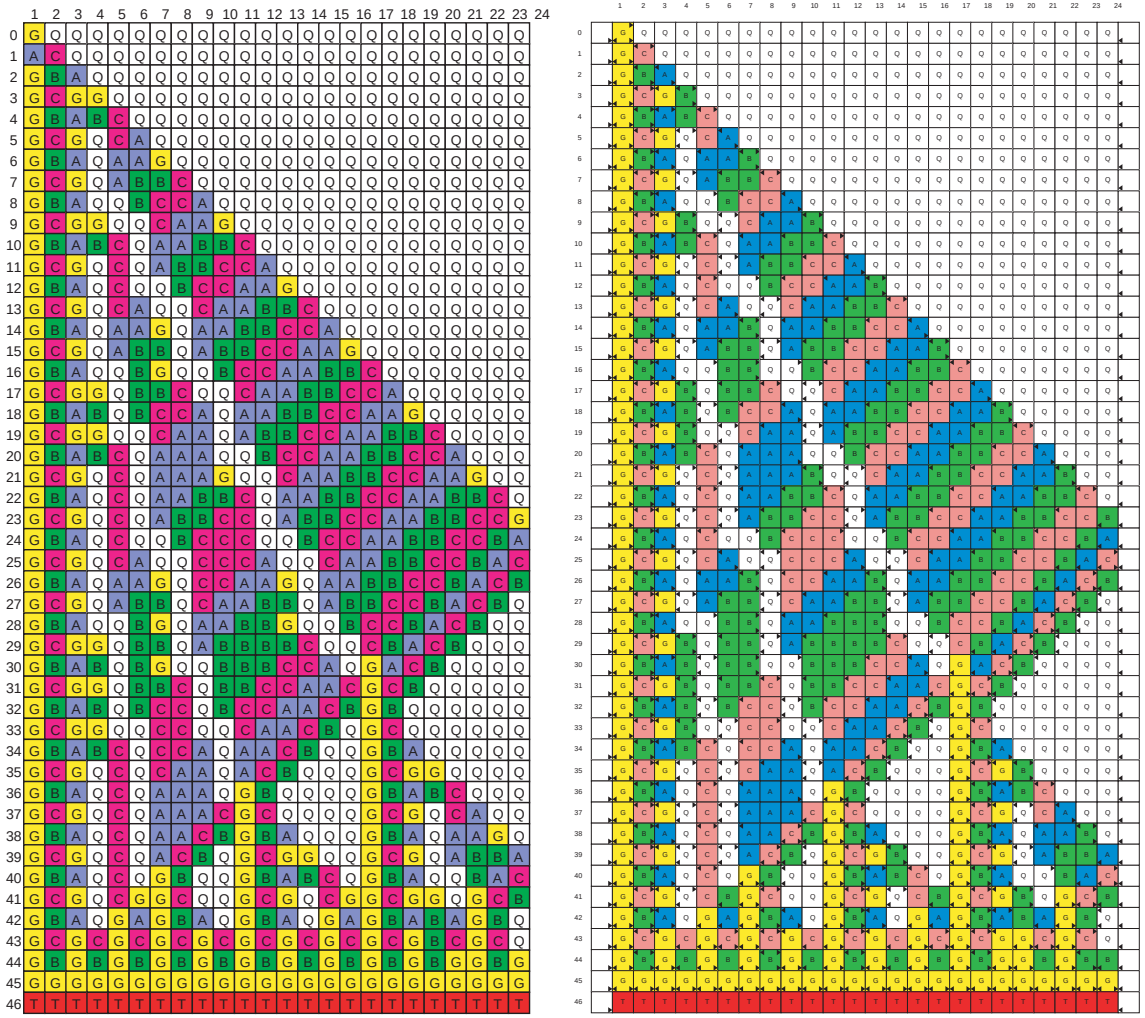




$\mathcal{P}_2$: An Optimum-Time Synchronization Protocol on $\mathrm{CA}_{2-\mathrm{bit}}$

[Theorem] There exists a 6-state $\mathrm{CA}_{2-\mathrm{bit}}$ that can synchronize any n cells in $2n-2$ optimum-step.

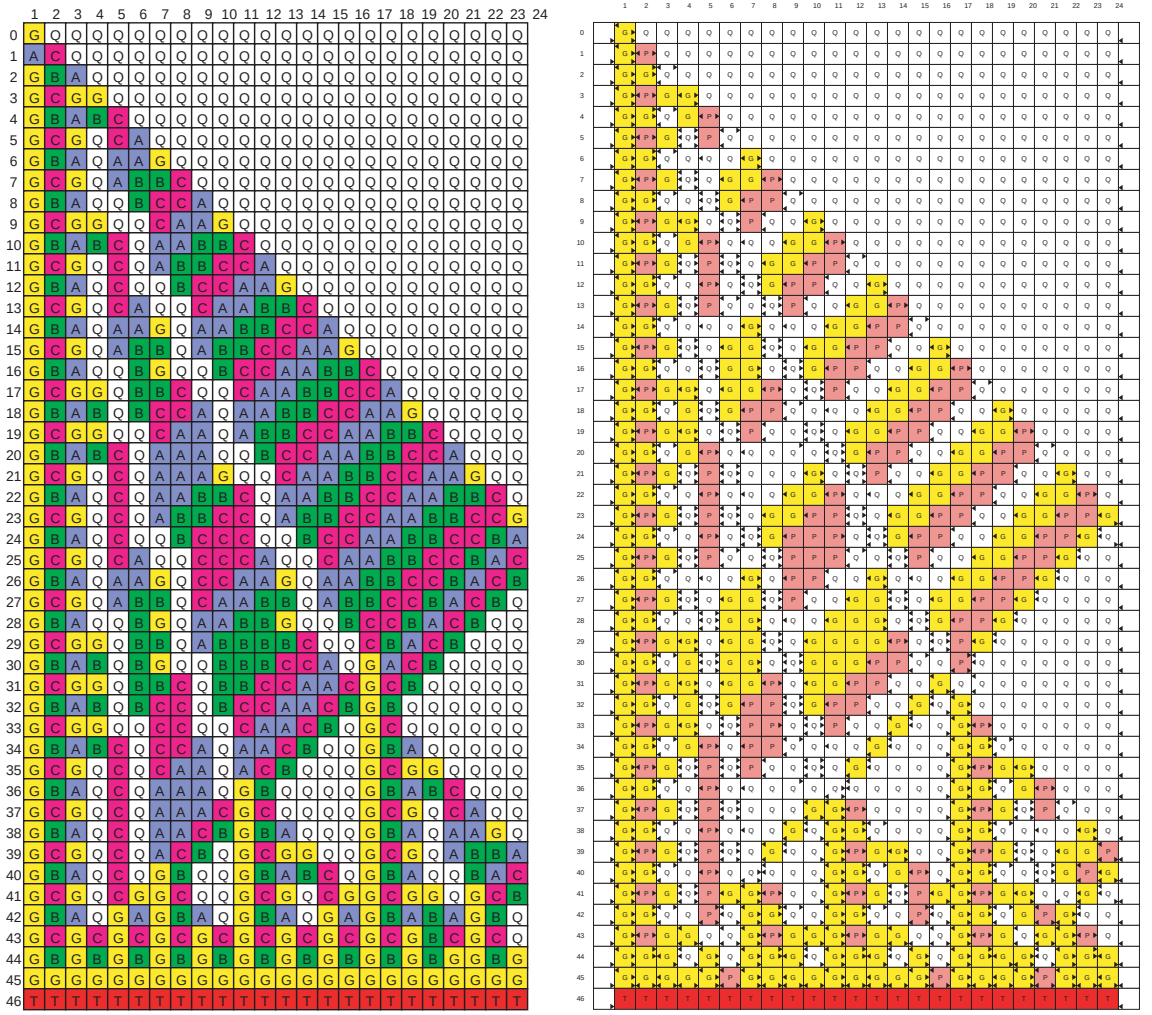
Snapshots for 6-state $(2n-2)$-step firing squad synchronization algorithm operating on $\mathrm{CA}_{2-\mathrm{bit}}$ with 24 cells.



$\mathcal{P}_3$: An Optimum-Time Synchronization Protocol on $\mathrm{CA}_{3-\mathrm{bit}}$

[Theorem] There exists a 4-state $\mathrm{CA}_{3-\mathrm{bit}}$ that can synchronize any n cells in $2n-2$ optimum-step.

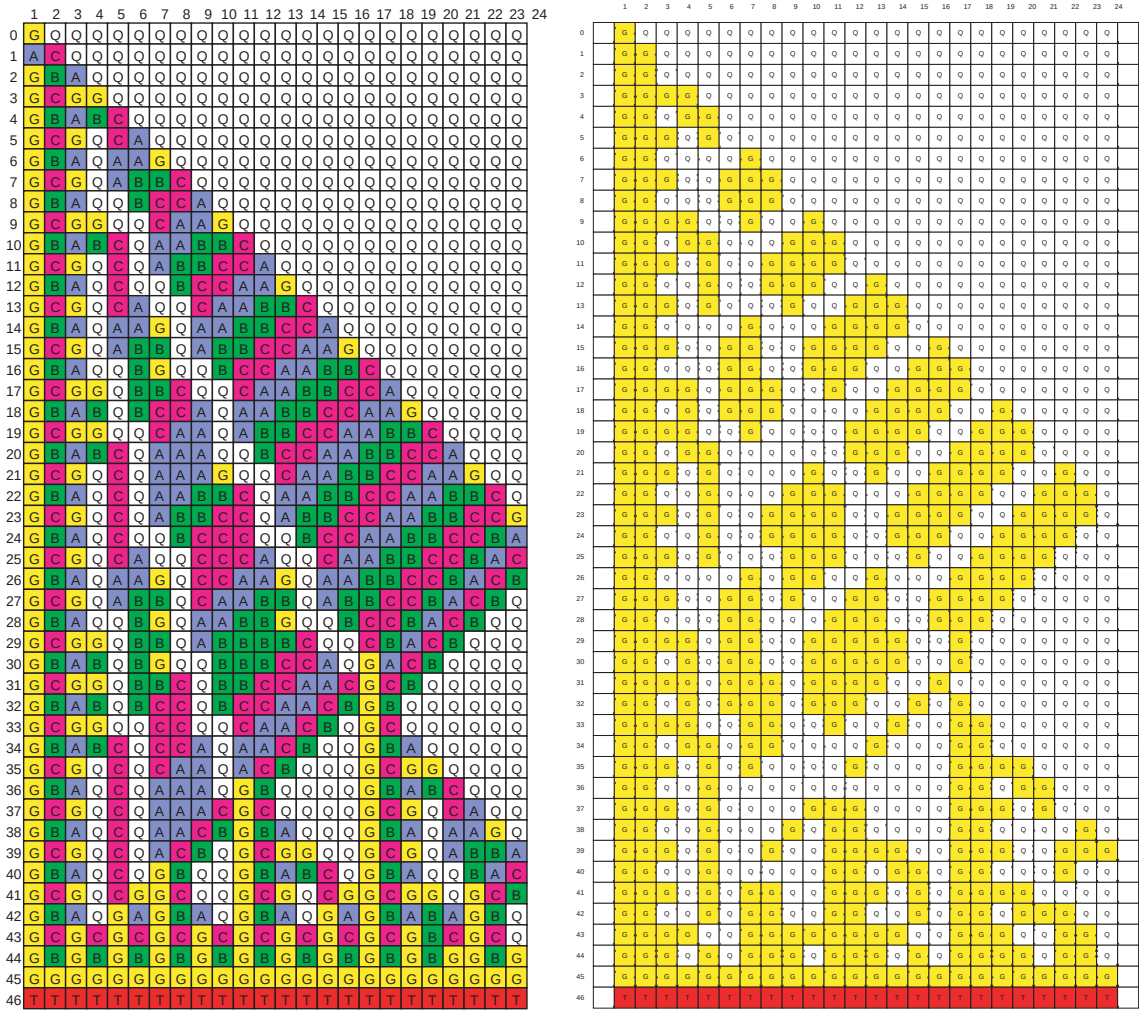
Snapshots for 4-state $(2n-2)$-step firing squad synchronization algorithm operating on $\mathrm{CA}_{3-\mathrm{bit}}$ with 24 cells.



$\mathcal{P}_4$: An Optimum-Time Synchronization Protocol on $\mathrm{CA}_{4-\mathrm{bit}}$

[Theorem] There exists a 3-state $\mathrm{CA}_{4-\mathrm{bit}}$ that can synchronize any n cells in $2n-2$ optimum-step.

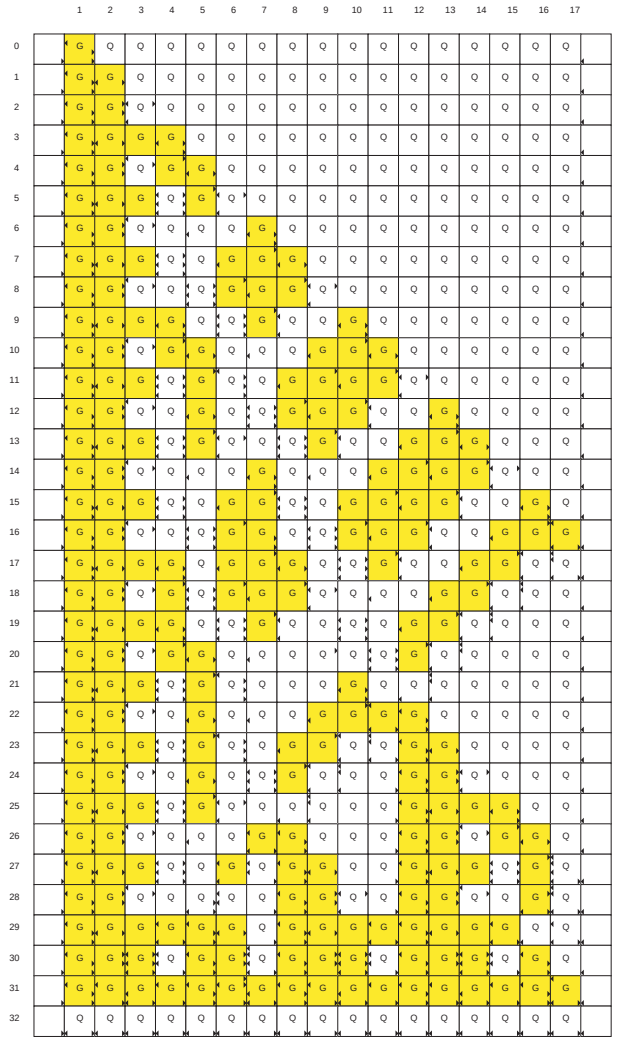
Snapshots for 3-state $(2n-2)$-step firing squad synchronization algorithm operating on $\mathrm{CA}_{3-\mathrm{bit}}$ with 24 cells.



$\mathcal{P}_4^*$: An Optimum-Time Synchronization Protocol on $\mathsf{CA}_{4-\mathsf{bit}}$

[Theorem] There exists a 2-state $\mathsf{CA}_{4-\mathsf{bit}}$ that can synchronize any n cells in $2n-2$ optimum-step.

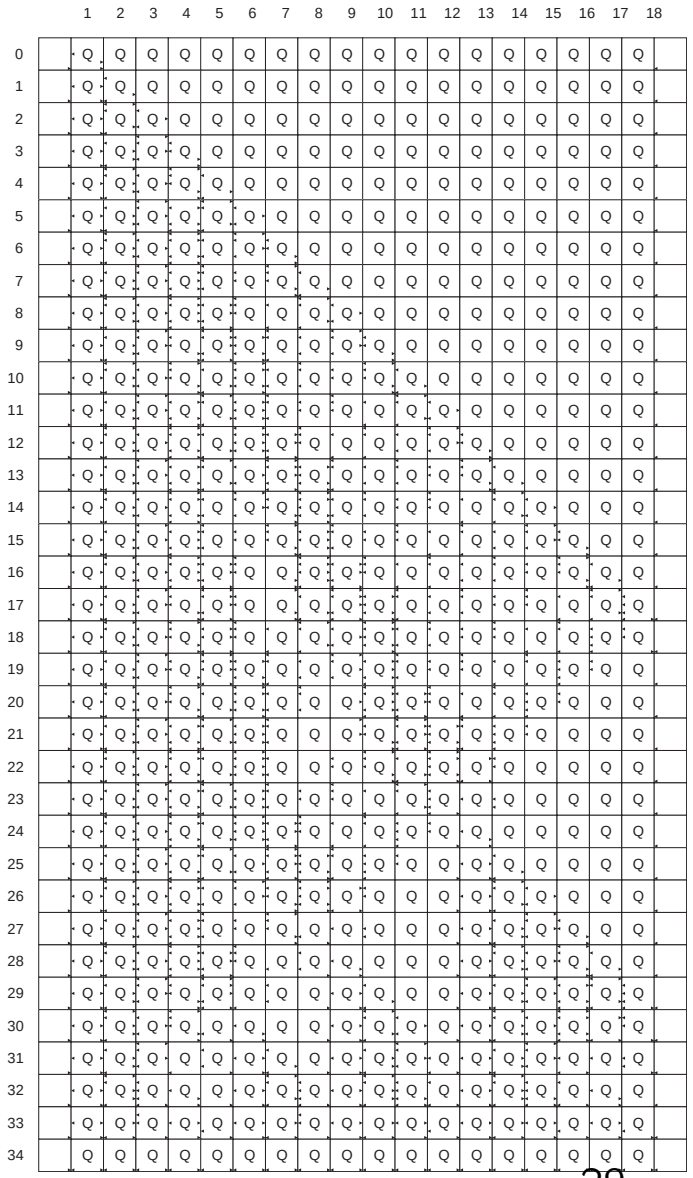
Snapshots for 2-state $(2n-2)$-step firing squad synchronization algorithm operating on $\mathsf{CA}_{4-\mathsf{bit}}$ with 17 cells.



$\mathcal{P}_5$: An Optimum-Time Synchronization Protocol on $\mathbf{CA}_{4-\text{bit}}$

[Theorem] There exists a 1-state $\text{CA}_{5-\text{bit}}$ that can synchronize any n cells in $2n-2$ optimum-step.

Snapshots for 1-state $(2n-2)$-step firing squad synchronization algorithm operating on $\text{CA}_{5-\text{bit}}$ with 18 cells.



Bit-Transfer Complexity on $\mathsf{CA}_{k-\mathsf{bit}}$

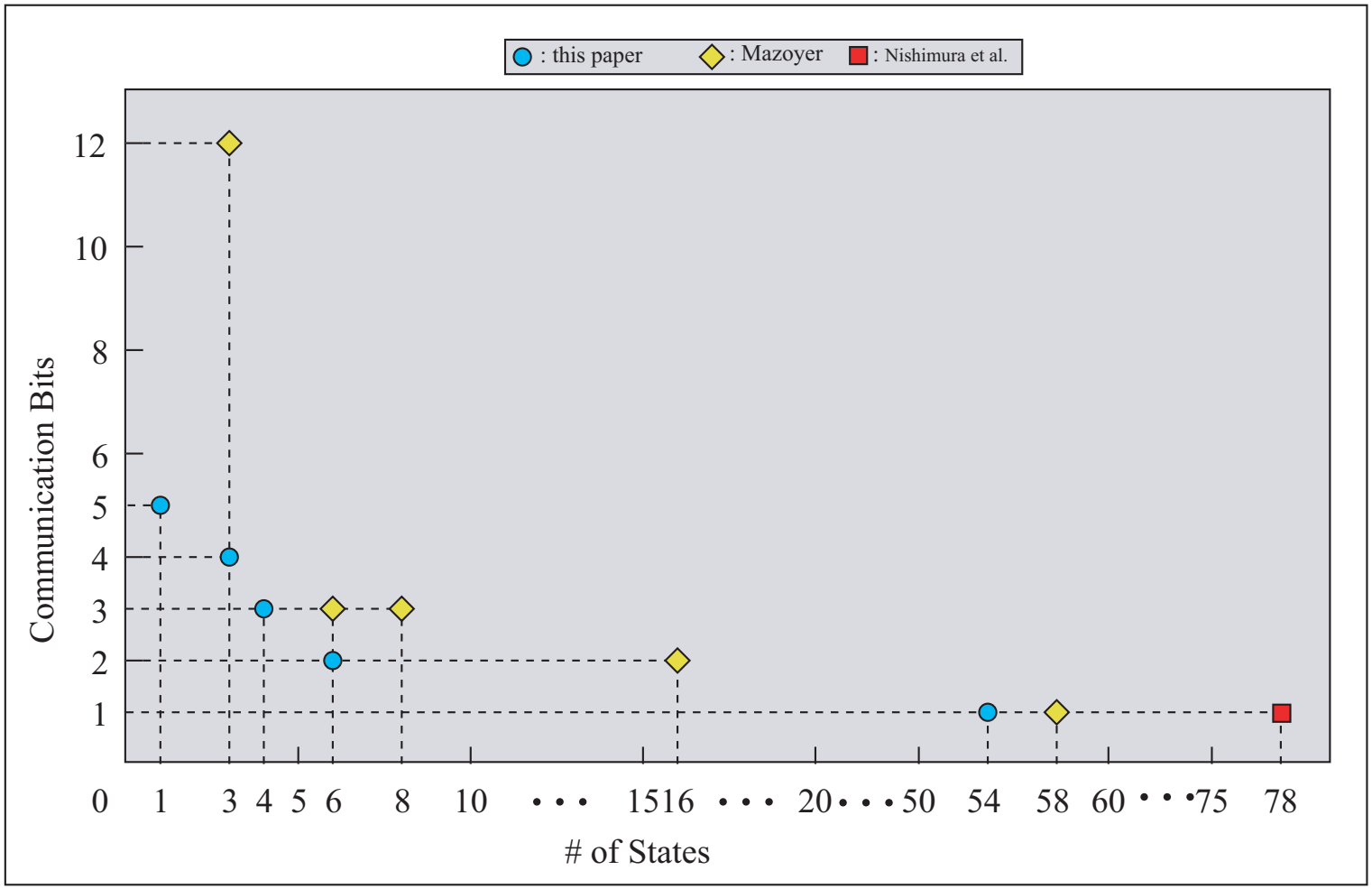
$BT(n)$: Total number of bits transferred needed for synchronizing n cells on $\mathsf{CA}_{k-\mathsf{bit}}$.

[Theorem] $\Omega(n \log n)$ bit-transfer is necessary for synchronizing n cells in $(2n-2)$ steps.

[Theorem] Each optimum-time/non-optimum-time synchronization protocols $\mathcal{P}_i, 0 \leq i \leq 5$, given in this talk, has an $O(n^2)$ bit-transfer complexity, respectively.

Synchronization Protocol	Communication Bits Transferred	# of States	# of Transition Rules	Time Complexity	One-/Two-sided Recursiveness
$\mathcal{P}_1$	1	54	207	$2n-1$	One-Sided
$\mathcal{P}_2$	2	6	60	$2n-2$	One-Sided
$\mathcal{P}_3$	3	4	76	$2n-2$	One-Sided
$\mathcal{P}_4$	4	3	87	$2n-2$	One-Sided
$\mathcal{P}_4^*$	4	2	88	$2n-2$	One-Sided
$\mathcal{P}_5$	5	1	114	$2n-2$	One-Sided
Mazoyer [1996]	1	58	-	$2n-2$	Two-Sided
Mazoyer [1996]	12	3	-	$2n-2$	-
Nishimura, Sobabe and Umeo [2000]	1	78	208	$2n-2$	Two-Sided

Table 3. Quantitative and qualitative comparison of optimum-time/non-optimum-time bit-communication synchronization protocols.


: this paper
: Mazoyer
: Nishimura et al.
Communication Bits
12
10
8
6
5
4
3
2
1
0
1
3
4
5
6
8
10
• • •
15
16
• • •
20
• • •
50
54
58
60
• • •
75
78
# of States

Conclusions

- Several state-efficient bit-transfer-based synchronization protocols are proposed.

- A bit-transfer complexity is introduced.

- What is the minimum number of states for protocols on each $\mathsf{CA}_{k-\mathsf{bit}}$, $k = 1, 2, 3, 4, ..$?