A Class of Minimum-Time Synchronization Algorithms for 2D Rectangluar Arrays Based on L-Shaped Mapping

Hiroshi Umeo[†], Takuya Yamawaki[†], Hiroki Uchino[†], and Kaori Ishida[†]
[†]School of Information Engineering
University of Osaka Electro-Communication
Neyagawa-shi, Hastu-cho, 18-8, Osaka, Japan
corresponding email address: umeo@cyt.osakac.ac.jp
{yamawaki, uchino, ishida}@cyt.osakac.ac.jp

***Abstract*—We introduce a new class of minimum-time FSSP (Firing Squad Synchronization Problem) algorithms for two-dimensional (2D) rectangular arrays. The algorithms in the class are all based on L-shaped mapping, where the synchronized configurations on 1D arrays are mapped in an L-shaped form onto 2D arrays efficiently, yielding minimum-time FSSP algorithms. We also present a comparative study of their recent implementations. Several state-efficient implementations, new insights into 2D synchronization and multi-dimensional extensions are also discussed.**

Keywords*—*cellular automata; FSSP; synchronization.

I. Introduction

The synchronization in ultra-fine grained parallel computational model of cellular automata, known as the Firing Squad Synchronization Problem (FSSP), has been studied extensively for more than fifty years, and a rich variety of synchronization algorithms has been proposed. In the present paper, we attempt to answer the following questions:

- First, how many variations of 2D FSSP algorithms and their implementations which are based on L-shaped mapping exist?
- How can we synchronize 2D arrays using the L-shaped decomposition in minimum-time?
- What is the exact rule set for the real implementations of minimum-time FSSP algorithms on 2D arrays?
- How do the algorithms compare with each other?
- Can we extend those 2D FSSP algorithms to 3D arrays, or more generally, to multi-dimensional arrays?

In Section 2, we give a description of the 2D FSSP and review some basic results on the 2D FSSP algorithms. Section 3 introduces a new class of FSSP algorithms based on L-shaped mapping for 2D arrays. In the last section, we give a summary of the paper.

II. Firing Squad Synchronization Problem

A. FSSP on 2D Cellular Arrays

Fig. 1 shows a finite two-dimensional (2D) cellular array consisting of $m \times n$ cells. Each cell is an identical (except the border cells) finite-state automaton. The array operates in lock-step mode in such a way that the next state of each cell (except border cells) is determined by both its own present state and the present states of its north, south, east, and west neighbors. All cells (*soldiers*), except the north-west corner cell (*general*), are initially in the quiescent state at time $t = 0$ with the property that the next state of a quiescent cell with quiescent neighbors is the quiescent state again. At time $t = 0$, the north-west upper corner cell $\mathrm{C}_{1,1}$ is in the *fire-when-ready* state, which is the initiation signal for the array. The FSSP is to determine a description (state set and next-state function) for cells that ensures all cells enter the *fire* state at exactly the same time and for the first time. 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 size $m \times n$ of the array. The set of states and transition rules must be independent of m and n.

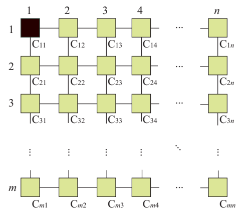

Fig. 1. A two-dimensional (2D) rectangular cellular automaton of size $m \times n$ arranged in m rows and n columns.

A formal definition of the 2D FSSP is as follows: A cellular automaton $\mathcal{M}$ is a pair $\mathcal{M} = (\mathcal{Q}, \delta)$, where

1) $\mathcal{Q}$ is a finite set of states with three distinguished states $\mathtt{G}$, $\mathtt{Q}$, and $\mathtt{F}$, each in $\mathcal{Q}$. $\mathtt{G}$ is an initial general state, $\mathtt{Q}$ is a quiescent state, and $\mathtt{F}$ is a firing state, respectively.
2) δ is a next state function such that $\delta : \mathcal{Q} \times (\mathcal{Q} \cup \{*\})^4 \rightarrow \mathcal{Q}$. The state $* \notin \mathcal{Q}$ is a pseudo state of the border of the array.
3) The quiescent state $\mathtt{Q}$ must satisfy the following conditions: $\delta(\mathtt{Q},\mathtt{Q},\mathtt{Q},\mathtt{Q},\mathtt{Q}) = \delta(\mathtt{Q},*,\mathtt{Q},\mathtt{Q},*) = \delta(\mathtt{Q},*,\mathtt{Q},\mathtt{Q},\mathtt{Q}) = \delta(\mathtt{Q},*,*,\mathtt{Q},\mathtt{Q}) = \delta(\mathtt{Q},\mathtt{Q},\mathtt{Q},*,\mathtt{Q}) = \delta(\mathtt{Q},\mathtt{Q},*,*,\mathtt{Q}) = \delta(\mathtt{Q},\mathtt{Q},*,\mathtt{Q},\mathtt{Q}) = \delta(\mathtt{Q},\mathtt{Q},*,\mathtt{Q},*) = \delta(\mathtt{Q},\mathtt{Q},\mathtt{Q},\mathtt{Q},*) = \mathtt{Q}$.

A 2D cellular automaton of size $m \times n$, $\mathcal{M}_{m \times n}$ consisting of $m \times n$ copies of $\mathcal{M}$, is a 2D array of $\mathcal{M}$. Each $\mathcal{M}$ is referred to as a cell and denoted by $\mathrm{C}_{i,j}$, where $1 \le i \le m$ and $1 \le j \le n$. We denote a state of $\mathrm{C}_{i,j}$ at time (step) t by $\mathtt{S}^t_{i,j}$, where $t \ge 0, 1 \le i \le m, 1 \le j \le n$. Each tuple in the next state function δ means that:

$$\mathtt{S}^{t+1}_{\text{itself}} = \delta(\mathtt{S}^{t}_{\text{itself}}, \mathtt{S}^{t}_{\text{north}}, \mathtt{S}^{t}_{\text{south}}, \mathtt{S}^{t}_{\text{east}}, \mathtt{S}^{t}_{\text{west}}).$$

A *configuration* of $\mathcal{M}_{m\times n}$ at time t is a function $\mathcal{C}^t : [1,m]\times[1,n] \to \mathcal{Q}$ and denoted as:

$$\begin{array}{c} \mathtt{S}^t_{1,1}\mathtt{S}^t_{1,2} \text{ .... } \mathtt{S}^t_{1,n} \\ \mathtt{S}^t_{2,1}\mathtt{S}^t_{2,2} \text{ .... } \mathtt{S}^t_{2,n} \\ \mathtt{S}^t_{3,1}\mathtt{S}^t_{3,2} \text{ .... } \mathtt{S}^t_{3,n} \\ . \\ . \\ . \\ \mathtt{S}^t_{m,1}\mathtt{S}^t_{m,2} \text{ .... } \mathtt{S}^t_{m,n}. \end{array}$$

A *computation* of $\mathcal{M}_{m\times n}$ is a sequence of configurations of $\mathcal{M}_{m\times n}$, $\mathcal{C}^0$, $\mathcal{C}^1$, $\mathcal{C}^2$, ..., $\mathcal{C}^t$, ..., where $\mathcal{C}^0$ is a given initial configuration such that:

$$\mathtt{S}^0_{i,j} = \begin{cases} \mathtt{G} & i=j=1 \\ \mathtt{Q} & \text{otherwise.} \end{cases} \qquad (1)$$

A configuration at time $t+1$, $\mathcal{C}^{t+1}$ is computed by synchronous applications of the next transition function δ to each cell of $\mathcal{M}_{m\times n}$ in $\mathcal{C}^t$ such that:

$$\mathtt{S}^{t+1}_{i,j} = \delta(\mathtt{S}^t_{i,j}, \mathtt{S}^t_{i-1,j}, \mathtt{S}^t_{i+1,j}, \mathtt{S}^t_{i,j+1}, \mathtt{S}^t_{i,j-1}).$$

A *synchronized configuration* of $\mathcal{M}_{m\times n}$ at time t is a configuration $\mathcal{C}^t$, $\mathtt{S}^t_{i,j} = \mathtt{F}$, for any $1 \le i \le m$ and $1 \le j \le n$.

The FSSP is to obtain an $\mathcal{M}$ such that, for any $m, n \ge 2$,

1) A synchronized configuration at time $t = T(m,n)$, $\mathcal{C}^{T(m,n)}$: $\mathtt{S}^{T(m,n)}_{i,j} = \mathtt{F}$, for any $1 \le i \le m$ and $1 \le j \le n$, can be computed from an initial configuration $\mathcal{C}^0$ in equation (1).
2) For any t, i such that $1 \le t \le T(m,n)-1$, $1 \le i \le m, 1 \le i \le n, \mathtt{S}^t_{i,j} \neq \mathtt{F}$.

No cells fire before time $t = T(m,n)$. We say that the array $\mathcal{M}_{m\times n}$ is synchronized at time $t = T(m,n)$ and the function $T(m,n)$ is the time complexity for the synchronization.

B. Lower-Bound and Optimality in 2D FSSP Algorithms

As for the time optimality of the 2D FSSP algorithms, Beyer [1] and Shinahr [4] gave a lower bound and minimum-time FSSP algorithm.

Theorem 1 There exists no cellular automaton that can synchronize any 2D array of size $m \times n$ in less than $m+n+\max(m,n)-3$ steps, where the general is located at one corner of the array.

Theorem 2 There exists a cellular automaton that can synchronize any 2D array of size $m\times n$ at exactly $m+n+\max(m,n)-3$ steps, where the general is located at one corner of the array.

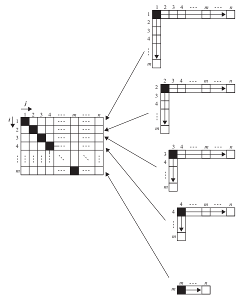
Fig. 2. A 2D array synchronization scheme based on L-shaped mapping developed in Beyer [1] and Shinahr [4].

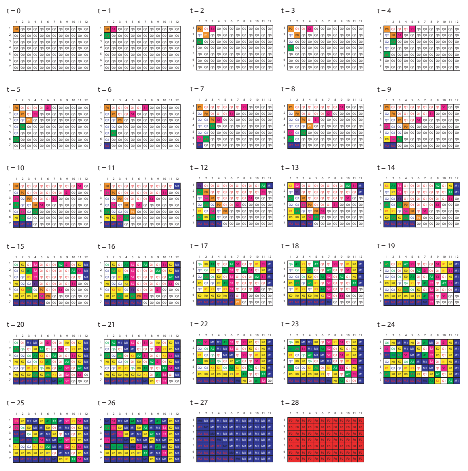
Fig. 3. Snapshots of the configurations of the Shinahr's 28-state synchronization algorithm on a rectangle array of size 7×12.

III. ALGORITHM $\mathcal{A}_1$

The first minimum-time FSSP algorithm $\mathcal{A}_1$, developed independently by Beyer [1] and Shinar [4], is based on rotated L-shaped mapping which maps configurations of generalized FSSP solutions on 1D L-shaped arrays onto 2D arrays. A rectangular array of size $m \times n$ is regarded as $\min(m,n)$ rotated L-shaped 1D arrays, where each rotated L-shaped 1D array is synchronized independently by using the Generalized FSSP (GFSSP) algorithm. The GFSSP algorithm [3], [8] is stated as follows:

Theorem 3 There exists a cellular automaton that can synchronize any 1D array of length n in optimum $n+\max(k,n-k+1)-2$ steps, where the general is located on the kth cell from left end.

We overview the algorithm $\mathcal{A}_1$ operating on an array of size $m \times n$. Configurations of the generalized synchronization processes on 1D array can be mapped on the rotated L-shaped array. We refer it as L-array. See Fig. 2. At time $t=0$, the north-west cell $\mathrm{C}_{1,1}$ is in general state and all other cells are in quiescent state. For any i such that $1 \le i \le \min(m,n)$, the cell $\mathrm{C}_{i,i}$ will be in the general state at time $t = 3i-3$. A special signal which travels towards a diagonal direction is used to generate generals on the cells $\{\mathrm{C}_{i,i} | 1 \le i \le \min(m,n)\ \}$. For each i such that $1 \le i \le \min(m,n)$, the cells $\{\mathrm{C}_{i,j} | i \le j \le n\ \}$ and $\{\mathrm{C}_{k,i} | i \le k \le m\}$ constitute the ith L-shaped array. Note that the ith general generated at time $t = 3i-3$ is on the $(m-i+1)$th cell from the left end of the ith L-array. The length of the ith L-array is $m+n-2i+1$. Thus, using Theorem 3, the ith L-array can be synchronized at exactly $t_i = 3i-3+m+n-2i+1-2+\max(m-i+1, n-i+1) = m+n+\max(m,n)-3$, which is independent of i. In this way, all of the L-arrays can be synchronized simultaneously.

Shinahr [4] presented a 28-state implementation of the algorithm, where most (97%) of the transition rules had *wild cards* which can match any state. Later, H. Umeo, K. Ishida, K. Tachibana, and N. Kamikawa [7] showed that the rule set consists of 12849 transition rules and it is valid for the synchronization for any rectangle arrays of size $m \times n$ such that $2 \leq m, n \leq 500$. Fig. 3 illustrates snapshots of the configurations on an array of size 7×12 based on our new 28-state 12849-rule implementation. Thus, we have:

Theorem 4 The algorithm $\mathcal{A}_1$ can synchronize any $m \times n$ rectangular array in optimum $m+n+\max(m,n)-3$ steps.

IV. Algorithm $\mathcal{A}_2$

In this section, we develop a new minimum-time FSSP algorithm $\mathcal{A}_2$ based on a new L-shaped mapping. First, we introduce a *freezing-thawing* technique that yields a delayed synchronization algorithm for 1D array. The technique was developed by Umeo [5] for designing several fault-tolerant FSSP algorithms for 1D arrays.

Theorem 5 Let t_0, t_1, t_2 and Δt be any integer such that $t_0 \geq 0$, $t_1 = t_0 + n - 1$, $t_1 \leq t_2$ and $\Delta t = t_2 - t_1$. We assume that a usual synchronization operation is started at time $t = t_0$ by generating a special signal at the left end of 1D array of length n. We also assume that the right end cell of the array receives another special signals from *outside* at time $t_1 = t_0 + n - 1$ and $t_2 = t_1 + \Delta t = t_0 + n - 1 + \Delta t$, respectively. Then, there exists a 1D cellular automaton that can synchronize the array at time $t = t_0 + 2n - 2 + \Delta t$.

We can freeze the entire configuration during Δt steps and delay the synchronization on the array for Δt steps. We refer the scheme as the *freezing-thawing technique*. Here, the freezing-thawing technique is employed efficiently.

A. *Segmentation of rectangular array*

First, we consider the case where $m \leq n$. We regard a 2D array of size $m \times n$ as consisting of m rotated (90° in counterclockwise direction) *L-shaped* 1D arrays. Each L-shaped array is denoted by $\mathrm{L}_i, 1 \leq i \leq m$, shown in Fig. 4. Each L_i is divided into three segments, referred to as 1st, 2nd and 3rd segments. The length of each segment of L_i is i, $n-m$, and i, respectively.

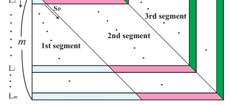

Fig. 4. A 2D array of size $m \times n$ is regarded as consisting of m rotated (90° in counterclockwise direction) L-shaped 1-D arrays. Each L-shaped array is divided into three segments.

B. *Starting the synchronization process*

At time $t = 0$, a 2D array M has a *general* at $\mathrm{C}_{1,1}$ and any other cells of the array are in quiescent state. The *general* G (denoted by ● in Fig. 4) generates three signals s_V, s_D and s_H, simultaneously, each propagating at $1/1$-speed in the vertical, diagonal and horizontal directions, respectively. See Fig. 4. The s_V- and s_H-signals work for generating wake-up signals for the 1st and 3rd segments on each L-shaped array. The s_D-signal is used for printing a delimiter between the 1st and 2nd segments. Their operations are as follows:

- **Signal $\mathbf{s}_V$**: The s_V-signal travels along the 1st column and reaches $\mathrm{C}_{m,1}$ at time $t = m-1$. Then, it returns there and begins to travel at $1/2$-speed along the 1st column towards $\mathrm{C}_{1,1}$. On the return's way, the signal initiates the synchronization process for the 1st segment of each L_i. Thus, a new *general* G_{i1} for the synchronization of the 1st segment of each L_i and its wake-up signal for the 1st segment are generated at time $t = 3m - 2i - 1$ for $1 \leq i \leq m$.
- **Signal $\mathbf{s}_H$**: The s_H-signal travels along the 1st row at $1/1$-speed and reaches $\mathrm{C}_{1,n}$ at time $t = n-1$. Then it reflects there and returns the same route at $1/2$-speed. Each time it visits a cell of the 1st row on its return way, it generates a *general* G_{i3} at time $t = 2m + n - 2i - 1$ to initiate a synchronization for the 3rd segment on each $\mathrm{L}_i, 1 \leq i \leq m$.
- **Signal $\mathbf{s}_D$**: The s_D-signal travels along a diagonal line by repeating a zig-zag movement: going one cell to the right, then going down one cell. Each time it visits a cell $\mathrm{C}_{i,i}$ on the diagonal, it marks a special symbol to denote a delimiter between the 1st and 2nd segments. The symbol on $\mathrm{C}_{i,i}$ is marked at time $t = 2i - 2$ for any $i, 1 \leq i \leq m$. Note that the wake-up signal of the 1st segment of L_m knows its right end by the arrival of s_D-signal, where they meet at $\mathrm{C}_{m,m}$ at the very time $t = 2m - 2$.

The wake-up signal generated by G_{i1} reaches its right end at time $t = 3m - i - 2$ and generates a new *general* G_{i2} for the 2nd segment. The new general G_{i2} generates a wake-up signal. The wake-up signals for the 2nd and 3rd segments of L_i meet on $\mathrm{C}_{i,n-m+i}$ at time $t = 2m + n - i - 2$. The collision of the two signals acts as a delimiter between the 2nd and 3rd segments in the case where $m \leq n$. Note that the synchronization operations on the 1st and 2nd segments are started at the left end of each segment. On the other hand, the synchronization on the 3rd segment is started at the right (upper) end of the segment.

C. *Synchronization of L_m*

Now, we consider the synchronization on L_m. Fig. 5 (left) shows a space-time diagram for synchronizing L_m. As was mentioned in the previous subsection, the synchronization of the 1st, 2nd and 3rd segments of L_m are started by the generals G_{m1}, G_{m2} and G_{m3} at time $t = m-1$, $t = 2m-2$ and $t = n-1$, respectively. Each General always generates a wake-up and a *pre-thawing* signal, each propagating at 1/1- and 1/2-speed in the same direction. The wake-up signal wakes up cells in the segment itself, however, the pre-thawing signal generates a thawing signal at its *neighboring* segment that it encounters first. Precisely, the pre-thawing signal generated by G_{m1} reaches the left end of the 2nd segments at time $t = 3m - 3$. In the case $m \leq n$, the configuration of the 2nd segment have not been frozen yet, and the signal doesn't work. The pre-thawing signal generated by G_{m2} arrives at the delimiter between the 2nd and 3rd segments at time $t = 2n-2$ and generates a thawing signal for the 3rd segment. In a similar way, the pre-thawing signal generated by G_{m3} initiates its thawing operation for the 2nd segment at time $t = 2m+n-2$.

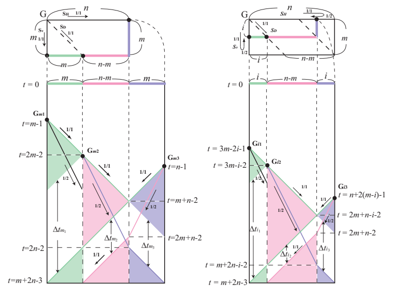

Fig. 5. Space-time diagram for synchronizing L_m (left) and L_i (right).

At time $t = 2m - 2$, the wake-up signal of the 1st segment reaches its right end and generates a freezing signal for the segment. Simultaneously, it initiates synchronization on the 2nd segments. The freezing signal for the 1st segment propagates in left direction at $1/1$-speed, and freezes the configuration on it. At time $t = 2n - 2$, a thawing signal is generated at its right end, that is initiated by the arrival of the freezing signal for the 2nd segment, which will be explained again below. The wake-up signal for the 2nd and 3rd segments meets $\mathrm{C}_{m,n}$ at time $t = m+n-2$, where $\mathrm{C}_{m,n}$ acts as an end of the both two segments. A new freezing signal is generated there for the 2nd and 3rd segments. It propagates in right and left directions at $1/1$-speed to freeze the synchronization operations on the 2nd and 3rd segments, respectively. The freezing signal for the 2nd segment reaches at its left end at time $t = 2n - 3$, which generates a thawing signal for the 1st segment at time $t = 2n - 2$. The thawing signals for the 1st, 2nd and 3rd segments are generated at time $t = 2n - 2$, $t = 2m+n-2$ and $t = 2n-2$, respectively. Synchronization operations on jth ($1 \leq j \leq 3$) segment are delayed for Δt_{m_j} steps such that:

$$\Delta t_{m_j} = \begin{cases} 2(n-m) & j = 1 \\ m & j = 2 \\ n-m & j = 3 \end{cases} \tag{2}$$

The synchronization for the 1st segment is started at time $t = m-1$ and its operations are delayed for $\Delta t = \Delta t_{m_1} = 2(n-m)$ steps. Now, letting $t_0 = m-1, \Delta t = \Delta t_{m_1} = 2(n-m)$ in Theorem 5, the 1st segment of L_m can be synchronized at time $t = t_0 + 2m - 2 + \Delta t = m + 2n - 3 = m + n + \max(m,n) - 3$. In a similar way, the 2nd and the 3rd segments can be synchronized at time $t = m + 2n - 3 = m + n + \max(m,n) - 3$. Thus, L_m can be synchronized at time $t = m + n + \max(m,n) - 3$.

D. Synchronization of L_i

Now, we discuss the synchronization for $\mathrm{L}_i, 1 \leq i \leq m$. Fig. 5 (right) shows a space-time diagram for synchronizing L_i. The wake-up signals for the three segments of L_i are generated at time $t = m + 2(m-i) - 1, 3m - i - 2$, and $n + 2(m-i) - 1$, respectively. Generation of the corresponding freezing and thawing signals are done in a similar way as employed in L_m. Synchronization operations on jth ($1 \leq j \leq 3$) segment are delayed for Δt_{i_j} steps such that:

$$\Delta t_{i_j} = \begin{cases} 2(n-m) & j = 1 \\ i & j = 2 \\ n-m & j = 3 \end{cases} \tag{3}$$

The synchronization for the 1st segment of L_i is started at time $t = m + 2(m-i) - 1$ and its operations are delayed for $\Delta t = \Delta t_{i_1} = 2(n-m)$ steps. Now, letting $t_0 = m + 2(m-i) - 1, \Delta t = \Delta t_{i_1} = 2(n-m)$ in Theorem 5, the 1st segment of L_i can be synchronized at time $t = t_0 + 2i - 2 + \Delta t = m + 2n - 3 = m + n + \max(m,n) - 3$. In a similar way, the 2nd and the 3rd segments can be synchronized at time $t = m + 2n - 3 = m + n + \max(m,n) - 3$. Thus, L_i can be synchronized at time $t = m + n + \max(m,n) - 3$.

E. Synchronization of rectangle longer than wide

In the case where $m > n$, a two-dimensional array of size $m \times n$ is regarded as consisting of n rotated L-shaped arrays. Segmentation and synchronization operations on each L-shaped array can be done almost in a similar way. It is noted that the right end cell of the 2nd segment works as a *General* for the 2nd segment. Any rectangle of size $m \times n$ can be synchronized at time $t = 2m+n-3 = m+n+\max(m,n)-3$.

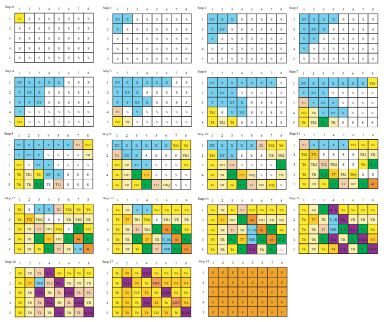

Fig. 6. Snapshots of the synchronization process on 5×8 array.

Fig. 6 shows some snapshots of the synchronization process operating in optimum-steps on a 5×8 array. Now, we can establish the next theorem.

Theorem 6 The algorithm $\mathcal{A}_2$ can synchronize any $m \times n$ rectangular array in optimum $m + n + \max(m,n) - 3$ steps.

V. ALGORITHM $\mathcal{A}_3$

In this section, we develop a new minimum-time firing squad synchronization algorithm $\mathcal{A}_3$ based on a new L-shaped mapping. The overview of the algorithm $\mathcal{A}_3$ is as follows:

1) A 2D array of size $m \times n$ is regarded as $\min(m, n)$ rotated or mirrored *L-shaped* 1-D arrays, each consisting of a horizontal and a vertical segment.
2) The shorter segment is synchronized by the freezing-thawing technique with $\Delta t =\mid m - n \mid$ steps delay. The longer one is synchronized in the usual way without using the freezing-thawing technique.
3) All of the *L-shaped* arrays fall into a special firing (synchronization) state simultaneously and for the first time.

A. Segmentation of rectangular array of size $m \times n$

First, we consider the case where $m \leq n$. We assume that the initial general **G** is on the north-west corner denoted by ● in Fig. 7. We regard a two-dimensional array of size $m \times n$ as consisting of m rotated ($90°$ in counterclockwise direction) *L-shaped* one-dimensional arrays. Each bending point of the *L-shaped* array is the farthest one from the general. Each *L-shaped* array is denoted by $\mathrm{L}_i, 1 \leq i \leq m$, shown in Fig. 7. Each L_i is divided into two segments, that is, one horizontal and one vertical segment, each referred to as 1st and 2nd segments. The length of each segment of L_i is $n - m + i$ and i, respectively.

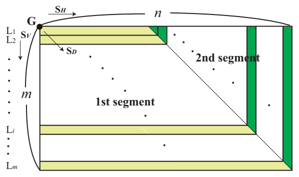

Fig. 7. A 2D array of size $m \times n$ ($m \leq n$) is regarded as consisting of m rotated ($90°$ in counterclockwise direction) L-shaped 1-D arrays. Each L-shaped array is divided into two segments.

B. Starting the synchronization process

At time $t = 0$, a two-dimensional array M has a *general* at $\mathrm{C}_{1,1}$ and any other cells of the array are in quiescent state. The *general* G (denoted by ● in Fig. 7) generates three signals s_V, s_D and s_H, simultaneously, each propagating at $1/1$-speed in the vertical, diagonal and horizontal directions, respectively. See Fig. 3. The s_V- and s_H-signals work for generating wake-up signals for the 1st and 2nd segments on each *L-shaped* array. The s_D-signal is used for printing a special marker "■" for generating a thawing signal that thaws frozen configurations on shorter segment. Their operations are as follows:

- **Signal $\mathbf{s}_V$:** The s_V-signal travels along the 1st column and reaches $\mathrm{C}_{m,1}$ at time $t = m - 1$. Then, it returns there and begins to travel again at $1/2$-speed along the 1st column towards $\mathrm{C}_{1,1}$. On the return's way, the signal initiates the synchronization process for the 1st segment of each L_i. Thus, a new *general* G_{i1} for the synchronization of the 1st segment of each L_i is generated, together with its wake-up signal, at time $t = 3m - 2i - 1$ for $1 \leq i \leq m$.
- **Signal $\mathbf{s}_D$:** The s_D-signal travels along a principal diagonal line by repeating a zigzag movement: going one cell to the right, then going down one cell. Each time it visits cell $\mathrm{C}_{i,i}$ on the diagonal, it marks a special symbol "■" to inform the wake-up signal on the segment of the position where a thawing signal is generated for the neighboring shorter segment. The symbol on C_{ii} is marked at time $t = 2i - 2$ for any $i, 1 \leq i \leq m$. Note that the wake-up signal of the 1st segment of L_m knows the right position by the arrival of the s_D-signal, where they meet at $\mathrm{C}_{m,m}$ at the very time $t = 2m - 2$.
- **Signal $\mathbf{s}_H$:** The s_H-signal travels along the 1st row at $1/1$-speed and reaches $\mathrm{C}_{1,n}$ at time $t = n - 1$. Then it reflects there and returns the same route at $1/2$-speed. Each time it visits a cell of the 1st row on its return way, it generates a *general* G_{i2} at time $t = 2m + n - 2i - 1$ to initiate a synchronization for the 2nd segment on each $\mathrm{L}_i, 1 \leq i \leq m$.

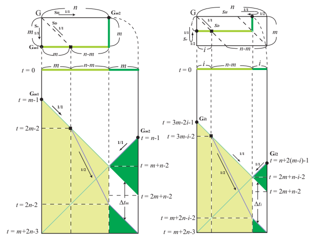

Fig. 8. Space-time diagram for synchronizing L_m (left) and L_i (right).

The wake-up signals for the 1st and 2nd segments of L_i meet on $\mathrm{C}_{i,n-m+i}$ at time $t = 2m + n - i - 2$. The collision of the two signals acts as a delimiter between the 1st and 2nd segments. Note that the synchronization operations on the 1st segment are started at the left end of the segment. On the other hand, the synchronization on the 2nd segment is started at the right (upper) end of the segment. The wake-up signal generated by G_{i1} reaches a cell where the special mark is printed at time $t = 3m - i - 2$ and generates a thawing signal which travels at $1/2$-speed along the 1st segment to thaw the configuration on the 2nd segment.

C. Synchronization of L_m

Now, we consider the synchronization on L_m. Fig. 8 (left) shows a space-time diagram for synchronizing L_m. As was mentioned in the previous subsection, the synchronization of the 1st and 2nd segments of L_m are started by the generals G_{m1} and G_{m2} at time $t = m - 1$ and $t = n - 1$, respectively. Each general generates a wake-up signal propagating at 1/1-speed. The wake-up signal for the 1st and 2nd segments meets C_{mn} at time $t = m + n - 2$, where $\mathrm{C}_{m,n}$ acts as an end of the both two segments. A freezing signal is generated simultaneously there for the 2nd segment at time $t = m+n-2$. It propagates in upper (right in Fig. 8 (left)) direction at $1/1$-speed to freeze the synchronization operations on the 2nd segment. At time $t = 2m - 2$, the wake-up signal of the 1st segment reaches the symbol "■" and generates a thawing signal for the 2nd segment. The thawing signal starts to propagate from the cell at 1/2-speed in the same direction. Those two signals reach at the left end of the 2nd segment with time difference $n - m$ which is equal to the delay for the 2nd segment. The synchronization for the 1st segment is started at time $t = m-1$ and it can be synchronized at time $t = m+2n-3 = m+n+\max(m,n)-3$ by the usual way. On the other hand, the synchronization for the 2nd segment is started at time $t = n-1$ and its operations are delayed for $\Delta t = \Delta t_m = n-m$ steps. Now, letting $t_0 = n-1, \Delta t = n-m$ in Theorem 1, the 2nd segment of length m on L_m can be synchronized at time $t = t_0 + 2m - 2 + \Delta t = m + 2n - 3 = m+n+\max(m,n)-3$. Thus, L_m can be synchronized at time $t = m+n+\max(m,n)-3$.

D. Synchronization of L_i

Now, we discuss the synchronization for $\mathrm{L}_i, 1 \leq i \leq m$. Fig. 8 (right) shows a space-time diagram for synchronizing L_i. The wake-up signals for the two segments of L_i are generated at time $t = 3m-2i-1$ and $n+2(m-i)-1$, respectively. Generation of freezing and thawing signals is done in a similar way as employed in L_m. Synchronization operations on the 2nd segment are delayed for $\Delta t_i = n-m$ steps. The synchronization for the 1st segment of length $n-m+i$ is started at time $t = 3m-2i-1$ and it can be synchronized by a usual method at time $t = m+2n-3 = m+n+\max(m,n)-3$. On the other hand, the synchronization for the 2nd segment is started at time $t = n+2(m-i)-1$ and its operations are delayed for $\Delta t = \Delta t_i = n-m$ steps. Now, letting $t_0 = n+2(m-i)-1, \Delta t = n-m$ in Theorem 5, the 2nd segment of length i of L_i can be synchronized at time $t = t_0+2i-2+\Delta t = m+2n-3 = m+n+\max(m,n)-3$. Thus, L_i can be synchronized at time $t = m+n+\max(m,n)-3$.

E. Synchronization of rectangle longer than wide

In the case where $m > n$, a two-dimensional array of size $m \times n$ is regarded as consisting of n mirrored *L-shaped* arrays. Segmentation and synchronization operations on each *L-shaped* array can be done almost in a similar way. It is noted that the thawing signal is generated on the 2nd segment to thaw frozen configurations on the 1st segment. Any rectangle of size $m \times n$ can be synchronized at time $t = 2m+n-3 = m+n+\max(m,n)-3$.

One notes that the algorithm needs no a priori knowledge on side length of a given rectangle, that is, whether wider than long or longer than wide. Fig. 9 shows some snapshots of the synchronization process operating in optimum-steps on a 6×9 arrays. Now, we can establish the next theorem.

Theorem 7 The synchronization algorithm $\mathcal{A}_3$ can synchronize any $m \times n$ rectangular array in optimum $m+n+\max(m,n)-3$ steps.

VI. Conclusion and Future Work

In the present paper, we gave a survey on recent developments in FSSP algorithms for 2D cellular arrays. We focused our attention on a new class of the 2D minimum-time FSSP algorithms based on L-shaped mapping. It is shown that the L-shaped mapping presents a rich variety of 2D minimum-time FSSP algorithms. As a future work, the mapping could be applied to the design of 3D minimum-time FSSP algorithms. The procedure would be as follows: First, a 2D version of the freezing-thawing technique has to be developed. Secondary, decompose a given 3D array into many thin layers, each consisting of three 2D faces. Last, apply the minimum-time 2D FSSP algorithm developed in this paper to each face.

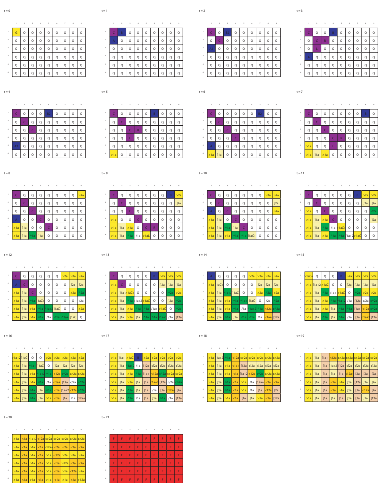
Fig. 9. Snapshots of the synchronization process on 6×9 array.

References

[1] W. T. Beyer, "Recognition of topological invariants by iterative arrays." Ph.D. Thesis, MIT, (1969), pp. 144.

[2] E. F. Moore, "The firing squad synchronization problem." In *Sequential Machines, Selected Papers* (E. F. Moore, ed.), Addison-Wesley, Reading MA., pp. 213-214(1964).

[3] F. R. Moore and G. G. Langdon, "A generalized firing squad problem." *Information and Control*, 12, pp.212-220(1968).

[4] I. Shinahr, "Two- and three-dimensional firing squad synchronization problems." *Information and Control*, vol. 24, pp. 163-180(1974).

[5] H. Umeo, "A simple design of time-efficient firing squad synchronization algorithms with fault-tolerance." *IEICE Trans. on Information and Systems*, Vol. E87-D, No.3, pp.733-739(2004).

[6] H. Umeo, "Firing squad synchronization problem in cellular automata." In *Encyclopedia of Complexity and System Science*, R. A. Meyers (Ed.), Springer, Vol.4, pp. 3537-3574(2009).

[7] H. Umeo, K. Ishida, K. Tachibana, and N. Kamikawa, "A transition rule set for the first 2-D optimum-time synchronization algorithm." *Proc. of the 4th International Workshop on Natural Computing*, PICT 2, Springer, pp.333-341(2009).

[8] H. Umeo, N. Kamikawa, K. Nishioka, and S. Akiguchi, "Generalized firing squad synchronization protocols for one-dimensional cellular automata - a survey." *Acta Physica Polonica B, Proceedings Supplement*. Vol.3, pp.267-289(2010).

[9] H. Umeo and H. Uchino, "A new time-optimum synchronization algorithm for rectangle arrays." *Fundamenta Informaticae*, Vol.87, No.2, pp.155-164(2008).