ELSEVIER

Theoretical Computer Science 259 (2001) 129–143

Theoretical Computer Science

www.elsevier.com/locate/tcs

On time optimal solutions of the firing squad synchronization problem for two-dimensional paths ☆

Kojiro Kobayashi [1]

Faculty of Engineering, Department of Information Systems Science, Soka University, 1-236 Tangi-cho, Hachioji-shi, Tokyo 192-8577, Japan

Received March 1999
Communicated by A. Salomma

Abstract

The firing squad synchronization problem for two-dimensional paths (2-Path FSSP, for short) is a variation of the firing squad synchronization problem where finite automata are placed along a path in the two-dimensional array space. Whether 2-Path FSSP has a time optimal solution or not is an open problem. We introduce a combinatorial problem which we call the two-dimensional path extension problem (2-PEP, for short), and show that if 2-Path FSSP has a time optimal solution then 2-PEP has a polynomial time algorithm. The computational complexity of 2-PEP is not well understood and the exhaustive search requiring exponential time is the only algorithm we know for it at present. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Firing squad synchronization problem; Network algorithms

1. Introduction

The *firing squad synchronization problem* (FSSP) is a problem on cellular automata that has a long history. The problem was posed by J. Myhill in 1957. See [7] for the origin of the problem and [5] for a survey. The problem is to construct a (deterministic) finite automaton A that satisfies some conditions.

☆ Expanded and modified version of the talk "On time optimal solutions of the two-dimensional firing squad synchronization problem" presented at MFCS'98 Satellite Workshop on Cellular Automata (Brno, Czech Republic, August 1998).

[1] Supported by Grant-in-Aid for Exploratory Research, The Ministry of Education, Science, Sports and Culture, Japan, No. 10878049.

E-mail address: kobayasi@t.soka.ac.jp (K. Kobayashi).

PII: S0304-3975(99)00332-1

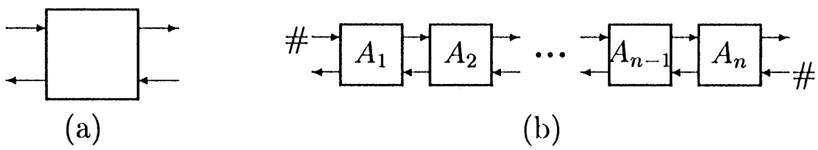


Fig. 1. The original FSSP.

The automaton A has two inputs, one from the left and one from the right, and two outputs, one to the left and one to the right (see Fig. 1(a)). The value of each output at a time t is the state of A at that time t. The state of A at time $t+1$ is completely determined by the state of A and the values of the two inputs of A at time t. We place identical copies $A_1, A_2, \ldots, A_n$ of A in a linear array and connect their inputs and outputs as shown in Fig. 1(b). The value of the left input of A_1 and the value of the right input of A_n are always a special value #.

Among the states of A are three special states Q, G, F, which we call the *quiescent state*, the *general state* and the *firing state*, respectively. If the state of A is Q and the values of the two inputs of A are either Q or # at a time t, the state of A at the next time $t+1$ must be Q. In other words, A_i in the quiescent state cannot enter a non-quiescent state unless at least one of its neighbors A_{i-1}, A_{i+1} enters a non-quiescent state.

At time 0, the state of A_1 is G and the states of all other $A_2, \ldots, A_n$ are Q. Then the states of $A_1, \ldots, A_n$ at time $0, 1, 2, \ldots$ are completely determined. The problem is to construct a finite automaton A such that, for any n $(\geqslant 2)$ there exists a time t_n such that

(1) the state of A_i at time t is not F for any $t < t_n$ and any i $(1 \leqslant i \leqslant n)$,

(2) the state of A_i at time t_n is F for any i $(1 \leqslant i \leqslant n)$.

We call a finite automaton A that satisfies the above-mentioned conditions a *solution* of FSSP. To find a solution is not difficult, and in [7] it is stated that usually it takes 2–4 h for a person to find a solution.

For each solution A, we call the time t_n the *firing time* of the solution A for an array of size n, and denote it by $\mathrm{ft}(A, n)$. For each n, we define the *minimum firing time* of an array of size n by $\mathrm{mft}(n) = \min\{\mathrm{ft}(A, n) \mid A \text{ is a solution}\}$, and call a solution A a *time optimal solution* if $\mathrm{ft}(A, n) = \mathrm{mft}(n)$ for any n $(\geqslant 2)$. We can easily show that $\mathrm{mft}(n) \geqslant 2n-2$ for any $n (\geqslant 2)$, and hence a solution A will be time optimal if $\mathrm{ft}(A, n) = 2n-2$ for any $n (\geqslant 2)$. Such a solution was first found by E. Goto in 1962. The number of the states of Goto's solution was quite large. However, since then the number has been reduced to six (see [6]).

FSSP has many variations. In the present paper, we consider the one which we call the *firing squad synchronization problem for two-dimensional paths* (2-Path FSSP, for short).

In 2-Path FSSP, we use a finite automaton A that has four inputs, each from one of the four directions right, up, left, down, and similarly four outputs, each to one of the

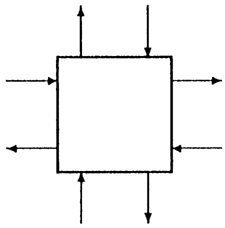


Fig. 2. Finite automata for two-dimensional arrays

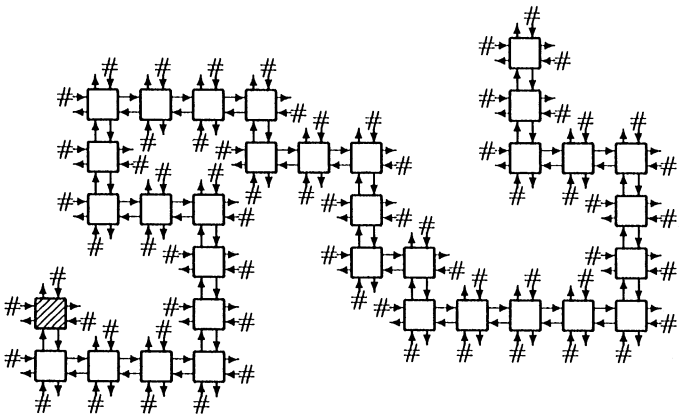


Fig. 3. An example of paths.

four directions (see Fig. 2). We place identical copies of A in the plane so that they form a path in a plane. The general (the automaton whose state at time 0 is G) is at one of the end positions of the path. (A formal definition of "paths" will be given later.)

In Fig. 3 we show an example of such paths. The automaton with shadow lines denotes the general. From now on, we will represent such a path by a simplified figure shown in Fig. 4.

The notions of a *solution* A, the *firing time* $\mathrm{ft}(A,C)$ of a solution A for a path C, the *minimum firing time* $\mathrm{mft}(C)$ of a path C, and a *time optimal solution* A are naturally defined for 2-Path FSSP.

We can easily modify a solution of the original FSSP to obtain a solution of 2-Path FSSP. Hence 2-Path FSSP has a solution. Moreover, from a time optimal solution of the original FSSP we obtain a solution A of 2-Path FSSP such that $\mathrm{ft}(A,C)=2|C|-2$, where $|C|$ denotes the length of a path C (that is, the number of automata in the path). However, this solution is not a time optimal solution of 2-Path FSSP. This follows from the following observation.

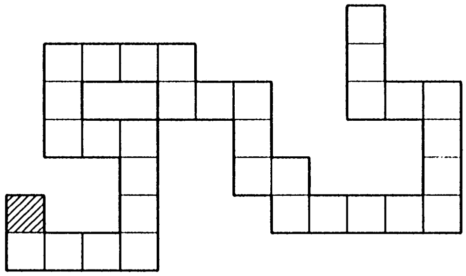


Fig. 4. A simplified representation of the path in Fig. 3.

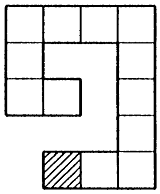


Fig. 5. A path C such that $\mathrm{mtf}(C) < 2|C| - 2$.

Let C be the path shown in Fig. 5. The length of this path C is 13. Hence we have $\mathrm{ft}(A, C) = 2 \cdot 13 - 2 = 24$. However, from what we will show later, we know that there is a solution A' such that $\mathrm{ft}(A', C) = 22$. This shows that $\mathrm{mft}(C)$ is smaller than $2|C| - 2$ for some path C and hence A is not a time optimal solution.

Whether 2-Path FSSP has a time optimal solution or not is an open problem. The present paper is an attempt to attack this open problem by a complexity theoretical approach.

We introduce a combinatorial problem which we call the *two-dimensional path extension problem* (2-PEP). We need some notions and notations to define the problem.

By a *position* p we mean a pair (x, y) of integers. Two positions $p = (x, y)$, $p' = (x', y')$ are said to be *adjacent* if either $|x - x'| = 1$ and $y = y'$ or $x = x'$ and $|y - y'| = 1$. By a *path* we mean a non-empty sequence $C = p_1 \dots p_n$ of positions such that $p_1, \dots, p_n$ are all different and p_i and p_j are adjacent if and only if $i + 1 = j$ $(1 \leqslant i < j \leqslant n)$. We call the value n the *length* of the path, and denote it by $|C|$. We call p_1 and p_n the *start position* and the *end position* of the path, respectively. Figures like the one shown in Fig. 4 are also used to represent paths. The square with shadow lines represent the start position of a path.

Now we are ready to define 2-PEP. An input to 2-PEP is a path $p_1 \dots p_n$. The problem is to decide whether we can extend the path from its end position to another path whose length is the double of the length of the original path $p_1 \dots p_n$. In other words, the problem is to decide whether there is a path of the form $p_1 \dots p_n q_1 \dots q_n$ or not. Note that, although both of 2-Path FSSP and 2-PEP use a common notion of

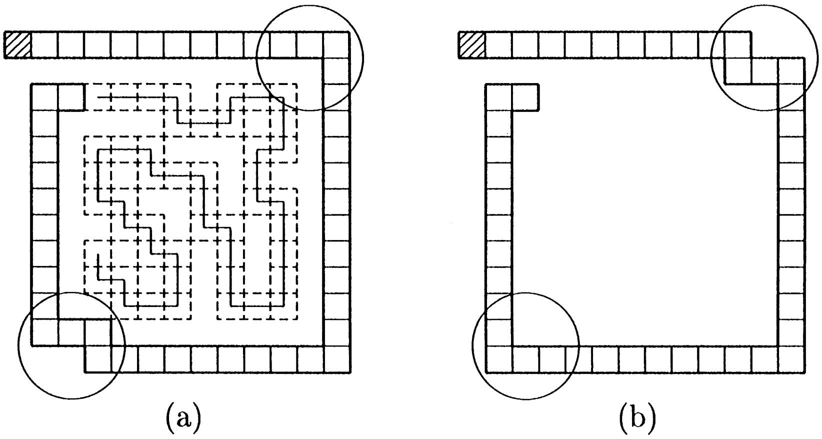


Fig. 6. Two examples of paths.

“paths”, 2-PEP does not use the notion of “automaton” in its definition, and is a purely combinatorial problem.

For example, if the input $p_1 \ldots p_n$ to 2-PEP is the path shown in Fig. 4, the answer is apparently YES. However, when the end position p_n is surrounded by the path itself the problem becomes difficult. In Figs. 6(a) and (b) we show two paths. They differ only in the parts in the circles. However, the answer to 2-PEP is YES for (a) (we show how to extend the path with dotted lines) and NO for (b). This example shows that the answer to 2-PEP depends on the form of the input path in a quite subtle way.

The main result of the present paper is that if 2-Path FSSP has a time optimal solution then 2-PEP has a polynomial time algorithm. At present, the computational complexity of 2-PEP is not well understood. The problem is apparently in NP. However, we have neither a polynomial time algorithm for it nor a proof of its NP-completeness. The exhaustive search with backtracking requiring an exponential time is the only algorithm we know for it at present. The answers to the paths of Figs. 6(a) and (b) were obtained with this algorithm.

Because of this lack of our clear understanding of the computational complexity of 2-PEP, we cannot claim that out result is a convincing circumstantial evidence for non-existence of time optimal solutions of 2-Path FSSP. However, our result suggests an interesting possibility that a solution of a purely complexity theoretical problem might lead to a solution of a purely automata theoretic problem.

2. Minimum firing times of 2-Path FSSP

FSSP have many variations. In [2] we obtained a characterization of the minimum firing time $\mathrm{mft}(C)$ of problem instances C for any variation that has a solution. This characterization gave an algorithm for computing the value of $\mathrm{mft}(C)$ for any given problem instance C under the assumption that the variation satisfies a certain additional condition which all natural variations satisfy. For 2-Path FSSP however, there is a characterization of $\mathrm{mft}(C)$ of paths C that is simpler than the one obtained in [2]. We show it in Theorem 1.

By the *boundary condition* of a position p in a path C we mean the information of whether the adjacent position p' of p is in the path C or not for each of the four directions right, up, left, down.

Let $p_1 \dots p_n$ be a path and i be a value such that $1 \leqslant i \leqslant n$. By $e(p_1 \dots p_n, i)$ we denote the maximum value of m such that there exists a path of the form $p_1 \dots p_i q_1 \dots q_m$ such that the boundary condition of p_i in $p_1 \dots p_n$ and that in $p_1 \dots p_i q_1 \dots q_m$ are the same. If there is no upper bound for such m then we define $e(p_1 \dots p_n, i)$ to be ∞. Equivalently, we may define $e(p_1 \dots p_n, i)$ to be the maximum value of m such that there exists a path of the form $p_1 \dots p_i p_{i+1} q_2 \dots q_m$ for $1 \leqslant i \leqslant n-1$, and to be 0 for $i = n$. We have $e(p_1 \dots p_n, 1) = \infty$, $e(p_1 \dots p_n, n) = 0$, and $e(p_1 \dots p_n, i)$ is a strictly decreasing function of i.

For a path $p_1 \dots p_n$ let i_0 be the value defined by

$$i_0 = \min\{i \mid 1 \leqslant i \leqslant n, i \geqslant e(p_1 \dots p_n, i)\}.$$

This value is well-defined because the left-hand side value i of the inequality increases from 1 to n and the right-hand side value $e(p_1 \dots p_n, i)$ decreases from ∞ to 0. We call p_{i_0} the *critical position* of the path $p_1 \dots p_n$.

The intuitive meaning of the critical position p_{i_0} of a path $p_1 \dots p_n$ may be explained in the following way.

Suppose that a traveler is traveling along the path $p_1 \dots p_n$ starting at p_1 and that he has arrived at p_i. Let $p_1 \dots p_i q_1 \dots q_m$ be a path that is consistent with what he has observed thus far (that is, the boundary conditions of positions, or the "scenery"). Moreover, let this path $p_1 \dots p_i q_1 \dots q_m$ be such that the value m is the largest among all such paths. In other words, $p_1 \dots p_i q_1 \dots q_m$ is the worst of all the possible paths in which the traveler might be as for the length of the remaining part of the path ahead of the traveler. Then we have $m = e(p_1 \dots p_n, i)$ and

$$\begin{aligned} i \geqslant e(p_1 \dots p_n, i) &\Leftrightarrow i \geqslant m \\ &\Leftrightarrow (i+m)/2 \leqslant i \\ &\Leftrightarrow \quad i+m \text{ is even} \wedge (i+m)/2 \leqslant i \\ &\qquad \vee\ i+m \text{ is odd} \wedge (i+m+1)/2 \leqslant i. \end{aligned}$$

Moreover, note that (1) if $i+m$ is even then $(i+m)/2$ is (the index of) the last position of the first half of the path $p_1 \dots p_i q_1 \dots q_m$, and (2) if $i+m$ is odd then $(i+m+1)/2$ is (the index of) the mid-position of the path $p_1 \dots p_i q_1 \dots q_m$.

Hence, the critical position is the first position p_i where the traveler is certain that one of the followings is true:

(1) The length of the path which the traveler is in is even, and either he is at the last position of the first half of the path or he has already passed the position.

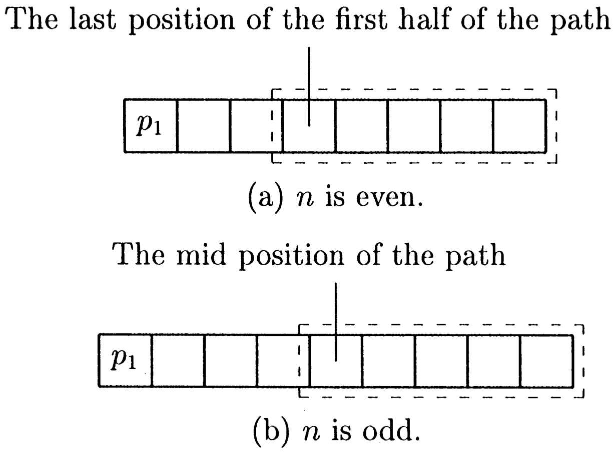


Fig. 7. Possible positions of the traveler.

(2) The length of the path is odd, and either he is at the mid-position of the path or he has already passed the position.

Fig. 7(a) and (b) shows these two cases. Here we are depicting paths schematically as if they are straight lines. The positions in dotted lines are possible positions of the traveler.

The critical position completely determines the minimum firing time $\mathrm{mft}(p_1 \dots p_n)$ of $p_1 \dots p_n$.

Theorem 1. *Let $p_1 \dots p_n$ be a path ($n \geqslant 2$) and p_{i_0} be its critical position. Then*

$$\mathrm{mft}(p_1 \dots p_n) = \begin{cases} 2i_0 - 1 & \text{if } i_0 = e(p_1 \dots p_n, i_0), \\ 2i_0 - 2 & \text{if } i_0 > e(p_1 \dots p_n, i_0). \end{cases}$$

Proof. We show the proof only for the case $i_0 = e(p_1 \dots p_n, i_0)$. The proof for the case $i_0 > e(p_1 \dots p_n, i_0)$ is similar except that we also use the additional property $e(p_1 \dots p_n, i_0 - 1) \geqslant i_0$.

First, we show $\mathrm{mft}(p_1 \dots p_n) \geqslant 2i_0 - 1$. Let A be an arbitrary solution of 2-Path FSSP. If $i_0 = n$ then we have a contradiction $1 \leqslant i_0 = e(p_1 \dots p_n, i_0) = e(p_1 \dots p_n, n) = 0$. Hence we have $i_0 \leqslant n - 1$, and there are positions $q_2, \dots, q_{i_0}$ such that $p_1 \dots p_{i_0} p_{i_0+1} q_2 \dots q_{i_0}$ is a path. Let X, Y denote the paths $p_1 \dots p_n$, $p_1 \dots p_{i_0} p_{i_0+1} q_2 \dots q_{i_0}$, respectively. Then we can show the following by the mathematical induction on t:

(1) For $0 \leqslant t \leqslant i_0 - 1$,
 (a) $\mathrm{st}(A, X, i, t) = \mathrm{st}(A, Y, i, t)$ for $1 \leqslant i \leqslant t + 1$,
 (b) $\mathrm{st}(A, X, i, t) = \mathrm{Q}$ for $t + 2 \leqslant i \leqslant n$,
 (c) $\mathrm{st}(A, Y, i, t) = \mathrm{Q}$ for $t + 2 \leqslant i \leqslant 2i_0$.

(2) For $i_0 \leqslant t \leqslant 2i_0 - 2$,

(a) $\mathrm{st}(A, X, i, t) = \mathrm{st}(A, Y, i, t)$ for $1 \leqslant i \leqslant 2i_0 - t$,

(b) $\mathrm{st}(A, Y, i, t) = \mathrm{Q}$ for $t + 2 \leqslant i \leqslant 2i_0$.

Here, for a path $C = r_1 \dots r_m$ and i $(1 \leqslant i \leqslant m)$, $\mathrm{st}(A, C, i, t)$ denotes the state of the copy of A placed at the ith position r_i of the path C at time t.

From these two statements we have $\mathrm{st}(A, X, 1, t) = \mathrm{st}(A, Y, 1, t)$ and $\mathrm{st}(A, Y, 2i_0, t) = \mathrm{Q}$ for any $t \leqslant 2i_0 - 2$. From $\mathrm{st}(A, Y, 2i_0, t) = \mathrm{Q}$ and the fact that A is a solution we have $\mathrm{st}(A, Y, 1, t) \neq \mathrm{F}$. From this and $\mathrm{st}(A, X, 1, t) = \mathrm{st}(A, Y, 1, t)$ we have $\mathrm{st}(A, X, 1, t) \neq \mathrm{F}$. Hence, we have $\mathrm{st}(A, p_1 \dots p_n, 1, t) \neq \mathrm{F}$ for any $t \leqslant 2i_0 - 2$, and hence $\mathrm{ft}(A, p_1 \dots p_n) \geqslant 2i_0 - 1$. The automation A was an arbitrary solution. Hence we have $\mathrm{mft}(p_1 \dots p_n) \geqslant 2i_0 - 1$.

Next we show $\mathrm{mft}(p_1 \dots p_n) \leqslant 2i_0 - 1$.

For a path $q_1 \dots q_m$, the *distance* between two positions q_i, q_j in the path is $|i - j|$. This is the shortest time for a signal to travel from q_i to q_j along the path.

Let $\mathscr{D}$ be the set of all paths of the form $p_1 \dots p_{i_0} q_1 \dots q_m$ $(m \geqslant 0)$ such that the boundary condition of p_{i_0} in $p_1 \dots p_{i_0} q_1 \dots q_m$ is the same as that in $p_1 \dots p_n$. Note that $p_1 \dots p_n$ itself is in the set $\mathscr{D}$.

If $C = p_1 \dots p_{i_0} q_1 \dots q_m$ is a path in $\mathscr{D}$, then the distance between p_{i_0} and any position in C is at most i_0. This is obviously true for positions in the part $p_1 \dots p_{i_0}$. By the definition of the set $\mathscr{D}$ and the fact that the path $p_1 \dots p_{i_0} q_1 \dots q_m$ is in $\mathscr{D}$, we have $i_0 = e(p_1 \dots p_n, i_0) \geqslant m$. Moreover the distance between p_{i_0} and q_j is j $(1 \leqslant j \leqslant m)$. Hence the distance between p_{i_0} and q_j in the part $q_1 \dots q_m$ is at most i_0.

We construct a finite automaton A'. Although A' is not a solution of 2-Path FSSP, later it will be used to construct a solution A such that $\mathrm{ft}(A, p_1 \dots p_n) \leqslant 2i_0 - 1$. Suppose that copies of A' are placed in positions of a path C. We construct A' so that A' satisfies the following two conditions.

C1: If C is not in $\mathscr{D}$ then no automata A' in C enter the state F.

C2: If C is in $\mathscr{D}$ then all the automata A' in C enter the state F for the first time simultaneously at time $2i_0 - 1$.

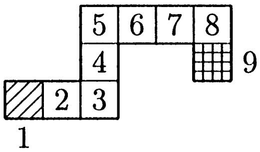
Especially, all the automata A' in the path $p_1 \dots p_n$ enter the state F for the first time simultaneously at time $2i_0 - 1$. We explain the structure of A' by explaining how signals are generated, propagate and vanish in automata A' placed in the path C.

At time 0 a signal U is generated at the position of the general. Its purpose is to see whether the path C is in $\mathscr{D}$ or not. To see this, the signal U travels from p_1 to p_{i_0} along the path $p_1 \dots p_{i_0}$ in C.

Suppose that the signal U has found some i $(1 \leqslant i \leqslant i_0)$ such that either p_i is not in C or p_i is in C but its boundary condition in C is different from that in $p_1 \dots p_n$. Then the signal U has found that C is not in $\mathscr{D}$. In this case, the signal U vanishes instantly and no automata A' in C enter the state F. This guarantees that A' satisfies the condition C1.

Suppose that the signal U has verified that, for each i $(1 \leqslant i \leqslant i_0)$, p_i is in C and its boundary condition in C is the same as that in $p_1 \dots p_n$. Then the signal U has found that C is in $\mathscr{D}$. In this case the signal U arrives at p_{i_0} at time $i_0 - 1$. As soon

Positions of both of p_s and the critical position p_{i_0}



i	$e(p_1 \ldots p_9, i)$
...	...
7	∞
8	∞
9	0

$$s = i_0 = n = 9$$
$$\mathrm{mft}(p_1 \ldots p_9) = 2i_0 - 2 = 16$$

Fig. 8. An example for the case $s = i_0 = n$.

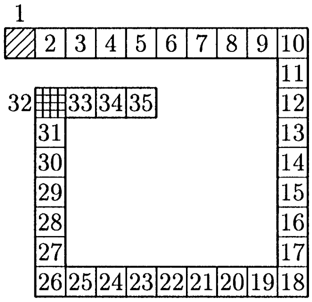
as U arrives at p_{i_0}, a new signal V_{i_0} is generated at p_{i_0} at time $i_0 - 1$, and then the signal V_{i_0} vanishes at the next time i_0. Moreover, when a signal V_i is generated at a position p in C at a time t $(1 \leqslant i \leqslant i_0)$, a signal V_{i-1} is generated at each of the adjacent positions of p in C at the time $t+1$, and then the signal V_{i-1} vanishes at the time $t+2$. Intuitively, a signal V_i is an order to "fire after i step time". Finally, when a signal V_i is generated at a position in C at a time t $(0 \leqslant i \leqslant i_0)$, the automaton A' at the position counts i step time and enters the state F at time $t+i$.

The signal V_{i_0} is generated at p_{i_0} at time $i_0 - 1$, and the distance between p_{i_0} and any position in C is at most i_0. Hence, at any position p in C a signal V_{i_0-i} is generated at time $(i_0 - 1) + i$ for some i $(0 \leqslant i \leqslant i_0)$, and the automaton A' at the position enters the state F at the time $(i_0 - 1) + i + (i_0 - i) = 2i_0 - 1$. Therefore, A' satisfies the condition C2.

It is not difficult to see that a finite number of states is sufficient to simulate this behavior of A'. Hence we can design A' as a finite automaton. (Of course the structure of A' essentially depends on the path $p_1 \ldots p_n$.) Let A'' be an arbitrary solution of 2-Path FSSP and let A be the finite automaton that simulates both of A', A'' and enters F if at least one of A', A'' enters F. Then A is a solution of 2-Path FSSP such that $\mathrm{ft}(A, p_1 \ldots p_n) \leqslant 2i_0 - 1$. This shows that $\mathrm{mft}(p_1 \ldots p_n) \leqslant 2i_0 - 1$. □

In Figs. 8–10 we show three examples of the computation of the value $\mathrm{mft}(p_1 \ldots p_n)$. Let s denote the value such that $e(p_1 \ldots p_n, s-1) = \infty$, $e(p_1 \ldots p_n, s) < \infty$. Figs. 8–10 are examples such that $s = i_0 = n$, $s = i_0 < n$, $s < i_0$, respectively. For the path $C = p_1 \ldots p_{13}$ shown in Fig. 5, we have $i_0 = 12$, $e(p_1 \ldots p_{13}, 12) = 1$, and hence $\mathrm{mft}(C) = 2i_0 - 2 = 22$.

Positions of both of p_s and the critical position p_{i_0}



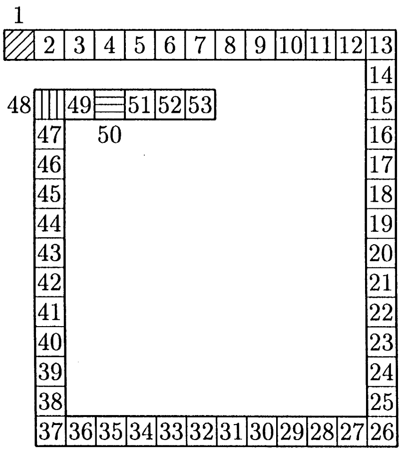
i	$e(p_1 \dots p_{35}, i)$
...	...
30	∞
31	∞
32	18
33	17
34	16
35	0

$$s = i_0 = 32, n = 35$$
$$\text{mft}(p_1 \dots p_{35}) = 2i_0 - 2 = 62$$

Fig. 9. An example for the case $s = i_0 < n$.

Position of p_s

Critical position p_{i_0}



i	$e(p_1 \dots p_{53}, i)$
...	...
46	∞
47	∞
48	52
49	51
50	50
51	49
52	48
53	0

$$s = 48, i_0 = 50, n = 53$$
$$\text{mft}(p_1 \dots p_{53}) = 2i_0 - 1 = 99$$

Fig. 10. An example for the case $s < i_0$.

3. The main result

Our new characterization of $\mathrm{mft}(C)$ of paths C for 2-Path FSSP gives a close relation between 2-Path FSSP and 2-PEP (Corollary 3). Using this relation we can show our main result (Corollary 4).

Theorem 2. *Suppose that* $p_1 \dots p_n r$ *is a path* $(n \geqslant 1)$. *Then* $\mathrm{mft}(p_1 \dots p_n r) \geqslant 2n-1$ *or* $\mathrm{mft}(p_1 \dots p_n r) \leqslant 2n-2$ *according as there exists a path of the form* $p_1 \dots p_n r q_2 \dots q_n$ *or not.*

Proof. Suppose that there is a path of the form $p_1 \dots p_n r q_2 \dots q_n$, and hence $e(p_1 \dots p_n r, n) \geqslant n$. We have $e(p_1 \dots p_n r, n-1) \geqslant e(p_1 \dots p_n r, n) + 1 \geqslant n+1 > n-1$, and hence $n \leqslant i_0$, where i_0 is the value $\min\{i \mid 1 \leqslant i \leqslant n+1, i \geqslant e(p_1 \dots p_n r, i)\}$. If $e(p_1 \dots p_n r, n) > n$ then we have $i_0 = n+1$, $e(p_1 \dots p_n r, i_0) = e(p_1 \dots p_n r, n+1) = 0$, and hence $\mathrm{mft}(p_1 \dots p_n r) = 2i_0 - 2 = 2n$. If $e(p_1 \dots p_n r, n) = n$ then we have $i_0 = n$, $e(p_1 \dots p_n r, i_0) = i_0$, and hence $\mathrm{mft}(p_1 \dots p_n r) = 2i_0 - 1 = 2n-1$. In any case we have $\mathrm{mft}(p_1 \dots p_n r) \geqslant 2n-1$.

Suppose that there is no path of the form $p_1 \dots p_n r q_2 \dots q_n$, and hence $e(p_1 \dots p_n r, n) < n$. If $e(p_1 \dots p_n r, n-1) \leqslant n-1$ we have $i_0 \leqslant n-1$ and hence $\mathrm{mft}(p_1 \dots p_n r) \leqslant 2i_0 - 1 \leqslant 2n-3$. If $e(p_1 \dots p_n r, n-1) > n-1$ we have $i_0 = n$, $i_0 > e(p_1 \dots p_n r, i_0)$ and hence $\mathrm{mft}(p_1 \dots p_n r) = 2i_0 - 2 = 2n-2$. In any case we have $\mathrm{mft}(p_1 \dots p_n r) \leqslant 2n-2$. □

Corollary 3. *The answer to* 2-*PEP for a path* $p_1 \dots p_n$ *is YES if and only if there exists a position* r *such that* $p_1 \dots p_n r$ *is a path and* $\mathrm{mft}(p_1 \dots p_n r) \geqslant 2n-1$.

Proof. We have

$$
\begin{aligned}
&\text{The answer is YES} \\
&\Leftrightarrow \exists r[p_1 \dots p_n r \text{ is a path } \wedge\ \exists q_2 \dots q_n\ [p_1 \dots p_n r q_2 \dots q_n \text{ is a path}]] \\
&\Leftrightarrow \exists r[p_1 \dots p_n r \text{ is a path } \wedge\ \mathrm{mft}(p_1 \dots p_n r) \geqslant 2n-1]. \qquad \square
\end{aligned}
$$

We formulate 2-PEP as a decision problem of a set of words. We represent a path $p_1 \dots p_m$ of length m by a word $s_1 \dots s_{m-1}$ of length $m-1$ such that s_i is the symbol R, U, L or D according as we move right, up, left or down respectively, to go from p_i to p_{i+1} $(1 \leqslant i \leqslant m-1)$. Let a set $L_{\text{2-PEP}}$ be defined by

$$
\begin{aligned}
L_{\text{2-PEP}} = \{s_1 \dots s_{m-1} \mid\ & s_1 \dots s_{m-1} \text{ represents a path } p_1 \dots p_m \text{ such that} \\
& \text{there is a path of the form } p_1 \dots p_m q_1 \dots q_m\}.
\end{aligned}
$$

Corollary 4. *If* 2-*Path FSSP has a time optimal solution then the set* $L_{\text{2-PEP}}$ *is decidable within* $\mathrm{O}(n^2)$ *time by a deterministic multitape Turing machine.*

Proof. Assume that 2-Path FSSP has a time optimal solution A. Suppose that we want to decide whether a sequence $x_1 \dots x_n$ of four symbols R, U, L, D is in $L_{\text{2-PEP}}$ or not. Let $p_1, \dots, p_{n+1}$ be the sequence of positions determined by the sequence $x_1 \dots x_n$.

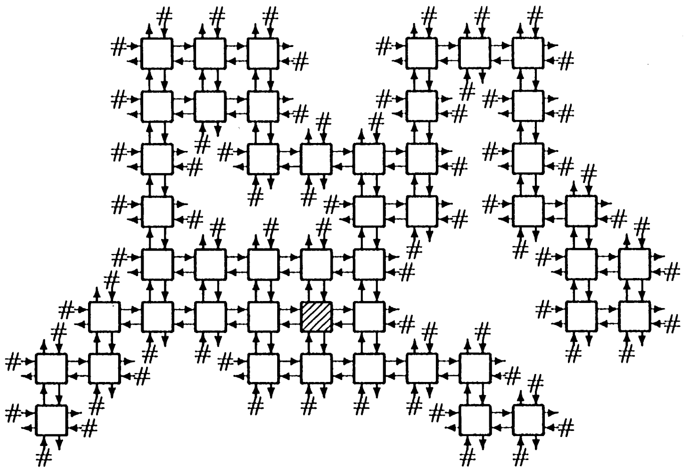


Fig. 11. FSSP for two-dimensional arrays.

First we must test whether $p_1 \dots p_{n+1}$ is a path or not, or equivalently whether or not there exist no i, j $(1 \leqslant i, i+2 \leqslant j \leqslant n+1)$ such that either $p_i = p_j$ or p_i and p_j are adjacent. This can be checked within $O(n^2)$ time.

Suppose that $p_1 \dots p_{n+1}$ is a path. Then it suffices to check the condition:

$$\exists r[p_1 \dots p_{n+1} r \text{ is a path} \wedge \mathrm{mft}(p_1 \dots p_{n+1} r) \geqslant 2n+1]$$
$$\Leftrightarrow \exists r[p_1 \dots p_{n+1} r \text{ is a path} \wedge \mathrm{ft}(A, p_1 \dots p_{n+1} r) \geqslant 2n+1].$$

There exists at most three r such that $p_1 \dots p_{n+1} r$ is a path. To test the condition $\mathrm{ft}(A, p_1 \dots p_{n+1} r) \geqslant 2n+1$ for a specific r, we simulate the behavior of $n+2$ copies of A placed at the $n+2$ positions $p_1, \dots, p_{n+1}, r$ until time $2n$. The simulation of one time step can be carried out within $O(n)$ time. Hence the simulation of up to time $2n$ can be carried out within $O(n^2)$ time. Hence, all the computation can be carried out within $O(n^2) + 3O(n^2) = O(n^2)$ time. □

4. Applications to other variations of FSSP

There are several variations of FSSP for which we do not know the time optimal solutions. We will list some of them below.

(1) *FSSP for two-dimensional arrays* [2]. Each automaton is of the type shown in Fig. 2 (that is, an automaton for two-dimensional arrays) and automata constitute a connected two-dimensional array of an arbitrary shape. The general may be at any position. Fig. 11 shows an example of such two-dimensional arrays.

(2) *FSSP for bilateral networks* [8]. Each automaton has a inputs and a outputs ($a(\geqslant 1)$ is a parameter of the variation). Automata constitute a connected network. The connection is bilateral in that if the rth input of an automaton A_i is connected to the

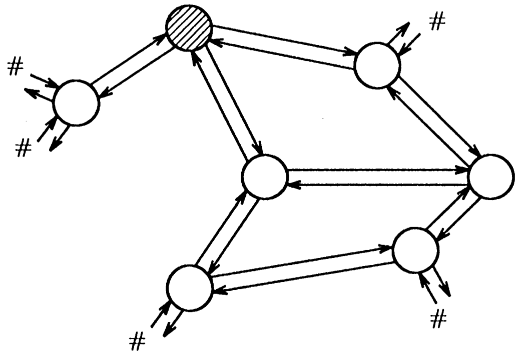


Fig. 12. FSSP for bilateral networks.

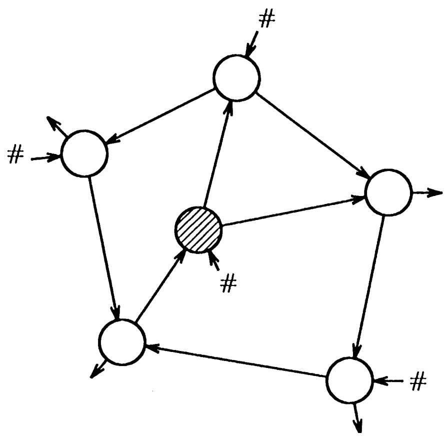


Fig. 13. FSSP for unilateral networks with at most one fan-out.

sth output of an automaton A_j then the sth input of A_j must be connected to the rth output of A_i, and if the rth input of A_i is open then the rth output of A_i is also open. Fig. 12 shows an example of such networks for the parameter $a = 3$.

(3) *FSSP for unilateral networks with at most one fan-out* [3]. Each automaton has a inputs and b outputs ($a(\geqslant 1)$ and $b(\geqslant 1)$ are parameters of the variation). Automata constitute a network. An output of an automaton is either open or is connected to an input of an automaton. In other words, the "fan-out" number of an output is at most one. The network must be strongly connected in that for each pair A_i, A_j of automata there is a directed path of connections from an output of A_i to an input of A_j. Fig. 13 shows an example of such networks for the parameters $a = 2, b = 2$.

Note that in variation (3) we do not allow an output of an automaton to be connected with more than one input of automata, and hence we do not allow, for example, the network shown in Fig. 14. If we allow such networks then the variation has no solution

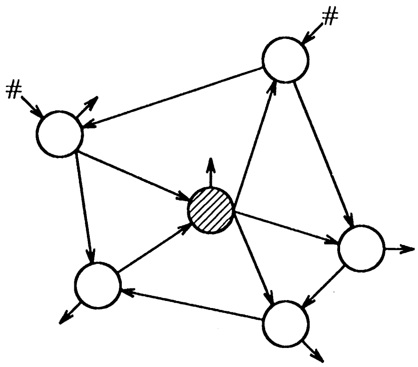


Fig. 14. FSSP for unilateral networks with unbounded fan-out.

[2], that is, FSSP for unilateral networks with unbounded fan-out has no solution. A solution to variation (2) was shown in [8]. A solution to variation (3) was shown in [3] (a solution with an exponential firing time) and in [1] (a solution with a polynomial firing time).

Our main result (Corollary 4) holds true for variation (1) too. The proof is the same except that in the proof of Theorem 1 the set $\mathscr{D}$ should be the set of all configurations C of variation (1) (not necessarily paths) that satisfy the following conditions:

(1) The path $p_1 \ldots p_{i_0}$ is included in C and p_1 is the general.

(2) For each i such that $1 \leqslant i \leqslant i_0$, the boundary condition of p_i in C is the same as that in $p_1 \ldots p_n$.

For the details of the proof, see [4], the preliminary version of this paper where we presented out main result for variation (1) instead of 2-Path FSSP.

As for the two variations for networks (2), (3), our problem 2-PEP cannot be used. It is an interesting problem to find another combinatorial problem X concerning graphs similar to 2-PEP such that

(1) the problem X seems to be intractable,

(2) the problem X is efficiently reducible to the problem of computing the minimum firing time mft(C) of a problem instance C (a network) of variation (2) or (3) (and hence, if the variation has a time optimal solution then problem X can be solved efficiently using the solution).

References

[1] S. Even, A. Litman, P. Winkler, Computing with snakes in directed networks of automata, 31st FOCS, 1990, pp. 740–745.

[2] K. Kobayashi, On the minimal firing time of the firing squad synchronization problem for polyautomata networks, Theoret. Comput. Sci. 7 (1978) 149–167.

[3] K. Kobayashi, The firing squad synchronization problem for a class of polyautomata networks, J. Comput. System Sci. 17 (1978) 300–318.

[4] K. Kobayashi, On time optimal solutions of the two-dimensional firing squad synchronization problem, in: T. Worsch, R. Vollmar (Ed.), Proc. MFCS'98 Satellite Workshop on Cellular Automata, August 25–27, Brno, Czech Republic, Universität Karlsruhe, Falultat für Informatik, Interner Bericht 19/98, 1998, pp. 17–26.
[5] J. Mazoyer, An overview of the firing squad synchronization problem, in: C. Choffrut (Ed.), Automata Networks, Lecture Notes in Computer Science, Vol. 316, Springer, Berlin, 1986, pp. 82–94.
[6] J. Mazoyer, A six-state minimal time solution to the firing squad synchronization problem, Theoret. Comput. Sci. 50 (1987) 183–238.
[7] E.F. Moore, The firing squad synchronization problem, in: E.F. Moore (Ed.), Sequential Machines, Selected Papers, Addison-Wesley, Reading, MA, 1964, pp. 213–214.
[8] P. Rosenstiehl, J.R. Fiksel, A. Holliger, Intelligent graphs: networks of finite automata capable of solving graph problems, in: R.C. Read (Ed.), Graph Theory and Computing, Academic Press, New York, 1972, pp. 219–265.