Information Processing Letters 114 (2014) 60–65

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

# The firing squad synchronization problem with sub-generals [☆]

Kazuya Yamashita [a,*], Yasuaki Nishitani [b], Sadaki Hirose [a], Satoshi Okawa [c], Nobuyasu Osato [d]

[a] *Faculty of Engineering, University of Toyama, Japan*
[b] *Faculty of Engineering, Iwate University, Japan*
[c] *Faculty of Computer Science and Engineering, University of Aizu, Japan*
[d] *Faculty of Information Science, Osaka Institute of Technology, Japan*

ARTICLE INFO

*Article history:*
Received 5 October 2010
Received in revised form 7 August 2013
Accepted 8 August 2013
Available online 16 August 2013
Communicated by M. Yamashita

*Keywords:*
Algorithms
Firing squad synchronization problem
Cellular automaton

ABSTRACT

The *Firing Squad Synchronization Problem* (FSSP), one of the most well-known problems related to cellular automata, was originally proposed by Myhill in 1957 and became famous through the work of Moore. The first solution to this problem was given by Minsky and McCarthy, and a minimal time solution was given by Goto. A considerable amount of research has also dealt with variants of this problem. In this paper, we introduce a new state called the *sub-general* to the original problem and propose the FSSP with sub-generals. In particular, we consider the case of one sub-general and determine the position of the sub-general in the array that minimizes the synchronization time. Moreover, we determine the minimal time to solve this problem and show that there exists no minimal time solution for any length of array. However, we show that the total time of our algorithm approaches arbitrarily close to the minimal time.

## 1. Introduction

The *Firing Squad Synchronization Problem* (FSSP), one of the most well-known problems related to cellular automata, was originally proposed by Myhill in 1957 and became famous through the work of Moore [1]. The first solution to the FSSP was given by Minsky and McCarthy [2], and requires $3n + O(\log n)$ steps for $n$ cells using thirteen states. It is easy to show that any solution to the FSSP requires at least $2(n-1)$ steps. Goto [3] gave a minimal time solution using several thousands of states. Inspired by his results, many researchers tried to reduce the number of states: in 1966, Waksman [4] gave a solution with sixteen states; in 1967, Balzer [5] gave a solution with eight states; in 1987, Gerken [6] reduced this to seven states; and in the same year, Mazoyer [7] gave a solution with six states, the minimal time solution with the fewest states at present. Moreover, it has been shown [5] that any solution needs at least five states.[1]

A considerable amount of research has also dealt with variants of the FSSP. The FSSP has been studied on a ring of $n$ cells [9], and on arrays of two and three dimensions [10,11]. The FSSP with multi-generals [12–14] has also been studied. The FSSPs for Cayley graphs [15] and another particular class of graphs [16] have also been studied. Some constrained variants of the FSSP have been developed to find solutions on reversible (i.e., backward deterministic) cellular automata [17] and cellular automata with a number-conserving property (i.e., a state is a tuple of non-negative integers whose sum is constant) [18]. Other kinds of constraints which have been considered involve the amount of information exchange between any pair of adjacent cells [19,20].

☆ This is an open-access article distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike License, which permits non-commercial use, distribution, and reproduction in any medium, provided the original author and source are credited.

* Corresponding author.
*E-mail address:* kazuya@eng.u-toyama.ac.jp (K. Yamashita).

[1] While Balzer's statement is correct, his proof is unfortunately wrong. See page 101 of the paper by Sanders [8].

0020-0190/$ – see front matter © 2013 The Authors. Published by Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ipl.2013.08.003

In this paper, we introduce a new state called the *sub-general* to the original problem and propose the FSSP with sub-generals. In particular, we consider the case of one sub-general and determine the position of the sub-general in the array that minimizes the synchronization time. Moreover, we clarify the minimal time to solve this problem and show that no minimal time solution exists. However, we show that the total time of our algorithm approaches arbitrarily close to the minimal time.

## 2. The firing squad synchronization problem

Since the FSSP is defined by cellular automata, we explain them briefly here.

A cellular automaton is defined by an interconnection network of finite automata. Finite automata on nodes of the network, called *cells*, are copies of a finite automaton and communicate through edges of the network. For a cell $v$, cells connected to $v$ directly are called *neighbor* cells of $v$. Each cell changes its state in discrete time by the state transition function depending on its own state and its neighbor cells' states (a set of a cell $v$ and its neighbor cells is called its *neighborhood*). The interval of the time in which cells change their states once is called a *step*.

For a one-dimensional cellular automaton, an interconnection network is a path of length $n$ for any positive integer $n$, and the neighbor cells of $v$ are the ones immediately to its right and left. Therefore, this is called a three-neighborhood one-dimensional cellular automaton.

### 2.1. Definition of the FSSP

The FSSP is defined on three-neighborhood one-dimensional cellular automata.

Consider a one-dimensional array of $n$ cells, $C_0, C_1, \dots, C_{n-1}$. Assume that $n$ is arbitrary, but finite. The cell $C_0$ at the left end is called a *general*, and the remaining cells $C_1, C_2, \dots, C_{n-1}$ are called *soldiers*. The problem is to design a set of states and a state transition function so that the general can cause all cells to transit to a particular terminal state, called the *firing* state, for the first time and at exactly the same time.

A more precise definition is as follows. $\mathcal{A} = (\mathcal{S}, \delta)$ is the finite automaton of each cell $C_i$ $(0 \leqslant i \leqslant n-1)$. $\mathcal{S}$ is a finite set of states: a *general* state $G \in \mathcal{S}$, a *quiescent* state $Q \in \mathcal{S}$, and a *firing* state $F \in \mathcal{S}$, each differing from the others. The state transition function is $\delta : (\mathcal{S} \cup \{B\}) \times \mathcal{S} \times (\mathcal{S} \cup \{B\}) \to \mathcal{S}$, where $B \notin \mathcal{S}$ is a signal that indicates the border of the array. For the quiescent state $Q$, we define $\delta(s_Q, Q, s_Q) = Q$, where $s_Q$ indicates any element of $\{Q, B\}$.

Initially, all soldiers are assumed to be in the quiescent state, and only the general is assumed to be in the general state. The state of $C_i$ $(0 \leqslant i \leqslant n-1)$ at time $t$ is denoted by $s_i^t$. The state $s_i^{t+1}$, which is the state of $C_i$ at time $t+1$, is determined by the state transition function and the states of its neighborhood at time $t$:

$$s_i^{t+1} = \delta\left(s_{i-1}^t, s_i^t, s_{i+1}^t\right).$$

If all cells first transit to the firing state $F$ at time $t_F$, we say that the array of $n$ cells is *fired* by the finite automaton $\mathcal{A}$. So, the sequence of states of each cell $C_i$ $(0 \leqslant i \leqslant n-1)$ is

$$s_i^{t_F} = F \quad \text{and} \quad s_i^t \neq F \quad \text{for all } 0 \leqslant t < t_F.$$

A finite automaton $\mathcal{A}$ solving the FSSP for all $n$ is called a *solution*. The time $t_F$ required to solve the FSSP by a finite automaton $\mathcal{A}$ for an array of $n$ cells is written as $t_F(n, \mathcal{A})$. The *minimal time* of the solutions for an array of $n$ cells is written as $t_{F\,\min}(n)$, where

$$t_{F\,\min}(n) = \min\left\{t_F(n, \mathcal{A}) \mid \mathcal{A} \text{ is a solution for the FSSP}\right\}.$$

If $t_F(n, \mathcal{A}) = t_{F\,\min}(n)$ holds for all $n$, the solution $\mathcal{A}$ is called a *minimal time solution*. In this case the solution $\mathcal{A}$ must achieve synchronization in minimal time for all $n$.

### 2.2. Outline of a minimal time solution of the FSSP

An outline of a minimal time solution (Waksman's algorithm [4]) of the FSSP is as follows.

At time $t = 0$, the general $C_0$ simultaneously sends signals,[2] at propagation speeds of $\frac{1}{1}, \frac{1}{3}, \frac{1}{7}, \dots, \frac{1}{2^k-1}, \dots,$ where $k$ is any positive integer[3] to the right. These signals are called the *firing signals* and play an important role in dividing the array into two, four, eight, …, equal parts synchronously. When a signal with a propagation speed of $\frac{1}{1}$ reaches the right end of the array at time $t = n-1$, the rightmost cell $C_{n-1}$ transits to the general state and behaves like the original general, except that it sends the firing signal to the left instead of the right. The signal with propagation speed $\frac{1}{3}$ which $C_0$ sent at time $t = 0$ and the signal with propagation speed $\frac{1}{1}$ which $C_{n-1}$ sent at time $t = n-1$ meet at the center of the array, midway between $C_0$ and $C_{n-1}$. Then, the middle cell transits to the general state and sends firing signals to the left and the right. As a result of this process, the array is divided into two equal parts. This process is repeated until all cells in the array are in the general state. Every cell transits to the firing state at the next step. Note that the above process ensures synchronization of the array and also permits the determination of the firing time if a cell as well as the cells on either side of it are in the general state.

It is easily understood that the time required for the above-mentioned process is $2(n-1)$ steps. This equals the time required for the signal with propagation speed $\frac{1}{1}$, sent by $C_0$, to reach $C_{n-1}$ and return.

Fig. 1 shows a time–space diagram of the minimal time algorithm. The horizontal axis is the cell space and the vertical axis is the time. The fractions in the figure are the propagation speeds of the firing signals and the dots indicate cells that are in the general state.

[2] A signal with a propagation speed of $\frac{1}{t}$ moves at a rate of one cell every $t$ steps.

[3] This unbounded number of different signals can be realized with a finite number of states. See for example [7] for details.