Fast Synchronization in P Systems

Artiom Alhazov[1], Maurice Margenstern[2], and Sergey Verlan[1,3]

[1] Institute of Mathematics and Computer Science
Academy of Sciences of Moldova
str. Academiei 5, MD-2028, Chişinău, Moldova
artiom@math.md
[2] Université Paul Verlaine - Metz, LITA, EA 3097, IUT de Metz
Ile du Saulcy, 57045 Metz Cédex, France
margens@univ-metz.fr
[3] LACL, Département Informatique, Université Paris Est
61 av. Général de Gaulle, 94010 Créteil, France
verlan@univ-paris12.fr

Abstract. We consider the problem of synchronizing the activity of all membranes of a P system. After pointing the connection with a similar problem dealt with in the field of cellular automata, where the problem is called the *firing squad synchronization problem*, $FSSP$ for short, we provide two algorithms to solve this problem for P systems. One algorithm is non-deterministic and works in $2h+3$ steps, the other is deterministic and works in $3h+3$ steps, where h is the height of the tree describing the membrane structure.

1 Introduction

The synchronization problem can be formulated in general terms with a wide scope of application. We consider a system constituted of explicitly identified elements and we require that starting from an initial configuration where one element is distinguished, after a finite time, all the elements which constitute the system reach a common feature, which we call **state**, all at the same time and the state was never reached before by any element.

This problem is well known for cellular automata, where it was intensively studied under the name of the *firing squad synchronization problem* (FSSP): a line of soldiers have to fire at the same time after the appropriate order of a general who stands in one end of the line, see [2,5,4,9,10,11]. The first solution of the problem was found by Goto, see [2]. It works on any cellular automaton on the line with n cells in the minimal time, $2n-2$ steps, and requiring several thousands of states. A bit later, Minsky found his famous solution which works in $3n$, see [5], with a much smaller number of states, 13 states. Then, a race to find a cellular automaton with the smallest number of states which synchronizes in $3n$ started. See the above papers for references and for the best results; for generalizations to the planar case, see [9] for results and references.

The synchronization problem appears in many different contexts, in particular in biology. As P systems model the work of a living cell constituted of many

D. Corne et al. (Eds.): WMC9 2008, LNCS 5391, pp. 118–128, 2009.

micro-organisms, represented by its membranes, it is a natural question to raise the same issue in this context. Take as an example the meiosis phenomenon, which probably starts with a synchronizing process which initiates the division process. Many studies have been dedicated to general synchronization principles occurring during the cell cycle; although some results are still controversial, it is widely recognized that these aspects might lead to an understanding of general biological principles used to study the normal cell cycle, see [8].

We may translate FSSP in P systems terms as follows. Starting from the initial configuration where all membranes, except the root, contain same objects, the system must reach a configuration where all membranes contain a distinguished symbol, F. Moreover, this symbol must appear in all membranes only during at the synchronization time.

The synchronization problem as defined above was studied in [1] for two classes of P systems: transitional P systems and P systems with priorities and polarizations. In the first case, a non-deterministic solution to FSSP was presented and for the second case a deterministic solution was found. These solutions need a time $3h$ and $4n + 2h$ respectively, where n is the number of membranes of a P system and h is the depth of the membrane tree.

In this article we significantly improve the previous results in the non-deterministic case. In the deterministic case, another type of P system was considered and this permitted to improve the parameters. The new algorithms synchronize the corresponding P systems in $2h + 3$ and $3h + 3$ steps respectively.

2 Definitions

In the following we briefly recall the basic notions concerning P systems. For more details we refer the reader to [6] and [12].

A transitional P system of degree n is a construct

$$\Pi = (O, \mu, w_1, \ldots, w_n, R_1, \ldots, R_n), \text{ where:}$$

1. O is a finite alphabet of symbols called objects,
2. μ is a membrane structure consisting of n membranes labeled in a one-to-one manner by $1, 2, \ldots, n$ (the outermost membrane is called the *skin* membrane),
3. $w_i \in O^*$, for each $1 \leq i \leq n$ is a multiset of objects associated with the region i (delimited by membrane i),
4. for each $1 \leq i \leq n$, R_i is a finite set of rules associated with the region i which have the following form $u \to v_1, tar_1; v_2, tar_2; \ldots; v_m, tar_m$, where $u \in O^+$, $v_i \in O$, and $tar_i \in \{in, out, here, in!\}$.

A transitional P system is defined as a computational device consisting of a set of n hierarchically nested membranes that identify n distinct regions (the membrane structure μ) where, to each region i, a multiset of objects w_i and a finite set of evolution rules R_i, $1 \leq i \leq n$, are assigned.

An evolution rule $u \to v_1, tar_1; v_2, tar_2; \ldots; v_m, tar_m$ rewrites u by $v_1, \ldots, v_m$ and moves each v_j accordingly to the target tar_j. If the tar_j target is *here*, then

v_j remains in membrane i. Target *here* can be omitted in the notation. If the target tar_j is *out*, then v_j is sent to the parent membrane of i. If the target tar_j is *in*, then v_j is sent to any inner membrane of i chosen non-deterministically. If the target tar_j is equal to $in!$, then v_j is sent to all inner membranes of i (a necessary number of copies is made).

A computation of the system is obtained by applying the rules in a non-deterministic maximally parallel manner. Initially, each region i contains the corresponding finite multiset w_i. A computation is successful if starting from the initial configuration it reaches a configuration where no rule can be applied. With a successful computation a result can be associated, but in what follows we are interested in the computation itself, not in any result of it.

A *transitional P system with promoters and inhibitors* is a system as defined as in the previous definition, where the set of rules may contain rules of the form

$$u \to v_1, tar_1; v_2, tar_2; \ldots; v_m, tar_m \mid_{P,\neg Q},$$

where $P \in O$ is the *promoter*, $Q \in O$ is the *inhibitor*, $tar_i \in \{in, out, here, in!\}$, $u \in O^+$ and $v_i \in O$. If P and/or Q are absent, we shall omit them. The meaning of promoter and inhibitor (if present in a rule) is that the rule is not applicable unless the promoter object exists in the current membrane, while the rule is applicable unless the inhibitor object is present in the current membrane.

We formulate the FSSP to P systems as follows:

Problem 1. *For a class of P systems $\mathcal{C}$ find two multisets $\mathcal{W}, \mathcal{W}' \in O^*$, and two sets of rules $\mathcal{R}$, $\mathcal{R}'$ such that for any P system $\Pi \in \mathcal{C}$ of degree $n \geq 2$ having*

$w_1 = \mathcal{W}'$, $R_1 = \mathcal{R}'$, $w_i = \mathcal{W}$ and $R_i = \mathcal{R}$ for all $i \in \{2, \ldots, n\}$, assuming that the skin membrane has the number 1

it holds

- *If the skin membrane is not taken into account, then the initial configuration of the system is stable (cannot evolve by itself).*
- *If the system halts, then all membranes contain the designated symbol F which appears only at the last step of the computation.*

3 The Non-deterministic Case

In this section we discuss a non-deterministic solution to the FSSP using transitional P systems. The main idea of such a synchronization is based on the fact that if a signal is sent from the root to a leaf, then it will take at most $2h$ steps to reach a leaf and return back to the root. In the meanwhile, the root may guess the value of h and propagate it step by step down the tree. This takes also $2h$ steps: h to guess the root, and h to end the propagation and synchronize. Hence, if the signal sent to the leaf, having depth $d \leq h$, returns at the same moment that the root ended the propagation, then the root guessed the value d. Now, in order to finish the construction it is sufficient to cut off cases when $d < h$.

In order to implement the above algorithm in transitional P systems we use the following steps.

- Mark leaves and nodes (nodes by $\bar{S}$ and leaves by S).
- From the root, send a copy of symbol a down. Any inner node must take one a in order to pass to state S'. If some node is not passed to state S' then when the signal c will come inside, it will be transformed to #.
- Then end of the guess is marked by signal c. Symbols S in leaves are transformed to S''' and those in inner nodes to S''.
- In the meanwhile the height is computed with the help of a symbol C_3. If a smaller height $d \leq h$ is obtained at the root node, then either the symbol C_3 will arrive to the root node and it will contain some symbols b – then the symbol # will be introduced at the root node, or the guessed value will be d and then there will be an inner node with $\bar{S}$ or a leaf with S (because we have at most d letters a) which leads to the introduction of # in the corresponding node.

Now let us present the system in details.

Let $\Pi = (O, \mu, w_1, \ldots, w_n, R_1, \ldots, R_n)$ be the P system to be synchronized. To solve the synchronization problem, we make the following assumptions on the objects, the membranes, and the rules. We consider that μ is an arbitrary membrane structure and

$$O = \{S, \bar{S}, S_1, S_2, S_3, C_1, C_2, C_3, S', S'', S''', a, b, c, F, \#\}.$$

We also assume that $w_1 = \{S_1\}$ and that all other membranes but the skin one are empty. The sets of rules, $R_1, \ldots, R_n$ are all equal and they are described below.

Start:

$$S_1 \to S_2; C_2; S, in!; C_1, in \tag{1}$$

Propagation of S:

$$S \to \bar{S}; S, in! \tag{2}$$

Root counter (guess):

$$S_2 \to S_2; b; a, in! \qquad S_2 \to S_3; c, in! \tag{3}$$

Propagate a:

$$\bar{S}a \to S' \qquad a \to b; a, in! \tag{4}$$

Propagate c:

$$cS' \to S''; c, in! \qquad cSa \to S''' \tag{5}$$

Decrement:

$$S''b \to S'' \qquad S'''a \to S''' \tag{6}$$
$$S'' \to F \qquad S''' \to F \tag{7}$$
$$S_3 b \to S_3 \tag{8}$$

Height computing:

$$C_1 \to C_1, in \qquad C_2 \to C_2, in \tag{9}$$
$$C_1 C_2 \to C_3 \qquad C_2 \to \# \tag{10}$$
$$C_3 \to C_3, out \tag{11}$$

Root firing:

$$C_3 S_3 \to F \tag{12}$$

Traps:

$$c\bar{S} \to \# \qquad cS \to \# \qquad C_3 \to \# \tag{13}$$
$$aF \to \# \qquad bF \to \# \qquad \# \to \# \tag{14}$$

The system Π has the desired behavior. Indeed, let us consider the functioning of this system.

Rule (1) produces objects S, C_1, C_2 and S_2. Object S will propagate down the tree structure by rule (2), leaving $\bar{S}$ in all intermediate nodes and S in the leaves. Objects C_1 and C_2 will be used to count the time corresponding to twice the depth d of some elementary membrane by rules (9)-(11) (trying to guess the maximal depth). Finally, object S_2 will produce objects b in some multiplicity by rules (3).

Together with objects b, objects a are produced by the first rule from (3), and they propagate down the tree structure by (4), one copy being subtracted at each level.

After the root finishes guessing the depth (second rule of (3)), object c propagates down the tree structure by (5), producing objects S'' at intermediate nodes and objects S''' at leaves; recall that the root has object S_3. These three objects perform the countdown (and then the corresponding nodes fire) by rules (6). As for the root, at firing by (12) it also checks that the timing matches twice the depth of the node visited by C_1 and C_2. The rules (13)-(14) handle possible cases of behavior of the system, not leading to the synchronization.

Now we present a more formal proof of the assertion above. We have the following claims.

- *The symbol C_3 will appear at the root node at the time $2d+2$, $d \leq h$, where h is the height of the membrane structure and d is the depth of the leaf visited by C_1.* Indeed, by rules (9) symbols C_1 and C_2, initially created by rule (1), go down until they reach a leaf. If they do not reach the same leaf, then the symbol # is introduced by C_2. The symbol C_2 reaches the leaf (of depth d) after $d+1$ steps. After that C_1 and C_2 are transformed to the symbol C_3 (1 step) which starts traveling up until it reaches the root node (d steps).

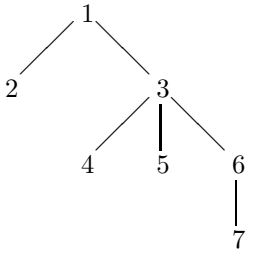
- *All nodes inner nodes will be marked by $\bar{S}$ and the leaves will be marked by S.* Indeed, rule (2) permits to implement this behavior.
- Let $d+2$, $0 \leq d \leq h$ be the moment when the root stops the guess of the tree height (the second rule from (3) has been applied). At this moment the contents of w_1 is $S_3 b^d$ and c starts to be propagated. Now consider any node x except the root. Then:

  x is of depth i then symbol c will reach x at time $d+i+1$ and the number of letters a (respectively letters b) present at x if it is a leaf (respectively inner node) is $a^{d \ominus i}$ (respectively $b^{d \ominus (i+1)}$), where $\ominus$ denotes the positive subtraction.

  The proof of this assertion may be done by induction. Initially, at step $d+2$, symbol c is present in all nodes of depth 1. Let x be such a node. If x is a leaf, then it received d copies of a. Otherwise, if x is an inner node, it must contain $d \ominus 1$ letters b (d letters a reached this node and all of them except one were transformed to b). The induction step is trivial since the letter c propagates each step down the tree and because the number of letters a reaching a depth i is smaller by one than the number of a reaching the depth $i-1$.
- From the above assertion it is clear that all nodes at time $2d+2$ will reach the configuration where there are no more letters b and a. Hence, all nodes, including the root node, up to depth d will synchronize at time $2d+3$.

Now, in order to finish the proof it is sufficient to observe that if $d \neq h$, then either there will be a symbol $\bar{S}$ in an inner node or the deepest leaf (having the depth h) will not contain object a (because only d letters a will be propagated down). Hence, when c will arrive at this node, it will be transformed to #.

Example 1. Consider a system Π having 7 membranes with the following membrane structure:



Now consider the evolution of the system Π constructed as above. We represent it in a table format where each cell indicates the contents of the corresponding membrane at the given time moment. Since the evolution is nondeterministic, we consider firstly the correct evolution and after that we shall discuss unsuccessful cases.

Step	w_1	w_2	w_3	w_4	w_5	w_6	w_7
0	S_1						
1	S_2C_2	S	SC_1				
2	S_2b	Sa	$\bar{S}aC_2$	S	S	SC_1	
3	S_2bb	Saa	$S'a$	S	S	$\bar{S}C_2$	SC_1
4	S_2bbb	$Saaa$	$S'ba$	Sa	Sa	$\bar{S}a$	SC_1C_2
5	S_3bbb	$Saaac$	$S'bbc$	Saa	Saa	$S'a$	SC_3
6	S_3bb	$S'''aa$	$S''bb$	$Saac$	$Saac$	$S'bcC_3$	Sa
7	S_3b	$S'''a$	$S''bC_3$	$S'''a$	$S'''a$	$S''b$	Sac
8	S_3C_3	S'''	S''	S'''	S'''	S''	S'''
9	F	F	F	F	F	F	F

The system will fail in the following cases:

1. Signals C_1 and C_2 go to different membranes.
2. Some symbol $\bar{S}$ is not transformed to S' (or the deepest leaf does not contain a letter a).
3. S_3 appears in the root membrane after C_3 appears in a leaf.
4. The branch chosen by C_3 is not the longest (it has the depth d, $d < h$).

A possible evolution for the first unsuccessful case is represented in the table below:

Step	w_1	w_2	w_3	w_4	w_5	w_6	w_7
0	S_1						
1	S_2C_2	S	SC_1				
2	S_2b	SaC_2	$\bar{S}a$	S	S	SC_1	
3	S_2bb	$Saa\#$	$S'a$	S	S	$\bar{S}$	SC_1

A possible evolution for the second unsuccessful case is represented in the table below:

Step	w_1	w_2	w_3	w_4	w_5	w_6	w_7
0	S_1						
1	S_2C_2	S	SC_1				
2	S_2b	Sa	$\bar{S}aC_2$	S	S	SC_1	
3	S_2bb	Saa	$\bar{S}ba$	Sa	Sa	$\bar{S}aC_2$	SC_1
4	S_2bbb	$Saaa$	$\bar{S}bba$	Saa	Saa	$S'a$	SC_1C_2
5	S_3bbb	$Saaac$	$\bar{S}bbbc$	$Saaa$	$Saaa$	$S'ab$	aSC_3
6	S_3bb	$S'''aa$	$\#bbb$	$Saaac$	$Saaac$	$S'bbcC_3$	Saa

A possible evolution for the third unsuccessful case is represented in the table below:

Step	w_1	w_2	w_3	w_4	w_5	w_6	w_7
0	S_1						
1	S_2C_2	S	SC_1				
2	S_2b	Sa	$\bar{S}aC_2$	S	S	SC_1	
3	S_2bb	Saa	$S'a$	S	S	$\bar{S}C_2$	SC_1
4	S_2bbb	$Saaa$	$S'ba$	Sa	Sa	$\bar{S}a$	SC_1C_2
5	S_2bbbb	$Saaaa$	$S'bba$	Saa	Saa	$S'a$	aSC_3
6	S_3bbbb	$Saaaac$	$S'bbbc$	$Saaa$	$Saaa$	$S'baC_3$	Sa
7	S_3bbb	$S'''aaa$	$S''bbbC_3$	$Saaac$	$Saaac$	$S'bbc$	Saa
8	S_3bbC_3	$S'''aa$	$S''bb$	$S'''aa$	$S'''aa$	$S''bb$	$Saac$
9	$S_3b\#$	$S'''a$	$S''b$	$S''a$	$S'''a$	$S''b$	$S'''a$

A possible evolution for the fourth unsuccessful case is represented in the table below:

Step	w_1	w_2	w_3	w_4	w_5	w_6	w_7
0	S_1						
1	S_2C_2	S	SC_1				
2	S_2b	Sa	$\bar{S}aC_2$	S	SC_1	S	
3	S_2bb	Saa	$S'a$	S	SC_1C_2	$\bar{S}$	S
4	S_2bbb	$Saaa$	$S'ba$	Sa	SaC_3	$\bar{S}a$	S
5	S_3bbb	$Saaac$	$S'bbcC_3$	Saa	Saa	$S'a$	S
6	S_3bbC_3	$S'''aa$	$S''bb$	$Saac$	$Saac$	$S'bc$	Sa
7	$S_3b\#$	$S'''a$	$S''bC_3$	$S'''a$	$S'''a$	$S''b$	Sac

4 The Deterministic Case

Consider now the deterministic case. We take the class of P systems with promoters and inhibitors and solve Problem 1 for this class.

The idea of the algorithm is very simple. A symbol C_2 is propagated down to the leaves and at each step, being at a inner node, it sends back a signal C. At the root a counter starts to compute the height of the tree and it stop if and only if there are no more signals C. It is easy to compute that the last signal C will arrive at time $2h-1$ (there are h inner nodes, and the last signal will continue for $h-1$ steps). At the same time the height is propagated down the tree as in the non-deterministic case.

The P system $\Pi = (O, \mu, w_1, \ldots, w_n, R_1, \ldots, R_n)$ for deterministic synchronization is present below. We consider that μ is an arbitrary membrane structure. The set of objects is $O = \{S_1, S_2, S_3, S_4, S, \bar{S}, S', S'', S''', C_1, C_2, C, a, a', b, F\}$,

the initial contents of the skin is $w_1 = \{S_1\}$, the other membranes are empty. The set of rules $R_1, \ldots, R_n$ are identical, they are presented below.

Start:

$$S_1 \to S_2; C_2'; S, in!; C_1, in! \tag{15}$$

Propagation of S:

$$S \to \bar{S}; S, in! \tag{16}$$

Propagation of C (height computing signal):

$$C_1 \to C_1, in! \qquad C_2 \to C; C_2, in!; C, out \tag{17}$$
$$C_1 C_2 \to \varepsilon \qquad C_2' \to C; C_2, in! \tag{18}$$
$$C \to C, out \tag{19}$$

Root counter:

$$S_2 \to S_3 \qquad S_3 \to S_3'; b; a, in! \mid_C \tag{20}$$
$$C \to \varepsilon \mid_{S_3} \qquad S_3' \to S_3 \mid_C \tag{21}$$
$$C \to \varepsilon \mid_{S_3'} \qquad S_3' \to S_4; a', in! \mid_{\neg C} \tag{22}$$

Propagation of a:

$$\bar{S}a \to S' \qquad a \to b; a, in! \mid_{S'} \tag{23}$$

End propagate of a:

$$a' S' \to S''; a', in! \qquad a' S a \to S''' \tag{24}$$

Decrement:

$$S'' b \to S'' \qquad S''' a \to S''' \tag{25}$$
$$S'' \to F \mid_{\neg b} \qquad S''' \to F \mid_{\neg a} \tag{26}$$

Root decrement:

$$S_4 b \to S_4 \qquad S_4 \to F \mid_{\neg b} \tag{27}$$

We now give a structural explanation of the system. Rule (15) produces four objects. Similar to the system from the previous section, the propagation of object S by (16) leads to marking the intermediate nodes by $\bar{S}$ and the leaves by S. While objects C_1, C_2 propagate down the tree structure and send a continuous stream of objects C up to the root by (17)-(19), object S_2 counts, producing by rules (20)-(22) an object b every other step.

When the counting stops, there will be exactly h copies of object b in the root. Similar to the construction from the previous section, objects a are produced together with objects b by the second rule from (20). Objects a are propagated down the structure and decremented by one at every level by (23).

After the counting stops in the root (the last rule from (22)), object a' is produced. It propagates down the tree structure by (24), leading to the appearance of objects S'' in the intermediate nodes and S''' in the leaves. These two

objects perform the countdown and the corresponding nodes fire by (25). The root behaves in a similar way by (27).

The correctness of the construction can be argued as follows. It takes $h+1$ steps for a symbol C_2 to reach all leaves. All this time, symbols C are sent up the tree. It takes further $h-1$ steps for all symbols C to reach the root node, and one more step until symbols C disappear. Therefore, symbols b appear in the root node every odd step from step 3 until step $2h+1$, so h copies will be made. Together with the production of b^h in the root node, this number propagates down the tree, being decremented by one at each level. For the depth i, the number $h-i$ is represented, during propagation, by the multiplicity of symbols a (one additional copy of a is made) in the leaves and by the multiplicity of symbols b in non-leaf nodes. After $2h+2$ steps, the root node starts the propagation of the countdown (*i.e.*, decrement of symbols a or b). For a node of depth i, it takes i steps for the countdown signal (a') to reach it, another $h-i$ steps to eliminate symbols a or b, so every node fires after $2h+2+i+(h-i)+1 = 3h+3$ steps after the synchronization has started.

Example 2. Consider a P system having the same membrane structure as the system from Example 1. The evolution of the system is as follows:

Step	w_1	w_2	w_3	w_4	w_5	w_6	w_7
0	S_1						
1	S_2C_2'	SC_1	SC_1				
2	S_3C	SC_1C_2	$\bar{S}C_2$	SC_1	SC_1	SC_1	
3	$S_3'bC$	Sa	$\bar{S}aC$	SC_1C_2	SC_1C_2	$\bar{S}C_2$	SC_1
4	S_3bC	Sa	$S'C$	S	S	$\bar{S}C$	SC_1C_2
5	$S_3'bbC$	Saa	$S'aC$	S	S	$\bar{S}$	S
6	S_3bbC	Saa	$S'b$	Sa	Sa	$\bar{S}a$	S
7	$S_3'bbb$	$Saaa$	$S'ba$	Sa	Sa	S'	S
8	S_4bbb	$a'Saaa$	$a'S'bb$	Saa	Saa	$S'a$	S
9	S_4bb	$S'''aa$	$S''bb$	$a'Saa$	$a'Saa$	$a'S'b$	Sa
10	S_4b	$S'''a$	$S''b$	$S'''a$	$S'''a$	$S'b$	$a'Sa$
11	S_4	S'''	S''	S'''	S'''	S'	S'''
12	F	F	F	F	F	F	F

5 Conclusions

In this article we presented two algorithms that synchronize two given classes of P systems. The first one is non-deterministic and it synchronizes the class of transitional P systems (with cooperative rules) in time $2h+3$, where h is the depth of the membrane tree. The second algorithm is deterministic and it synchronizes the class of P systems with promoters and inhibitors in time $3h+3$.

It is worth to note that the first algorithm has the interesting property that after $2h$ steps either the system synchronizes and the object F is introduced, or an object # will be present in some membrane.

The results obtained in this article rely on a rather strong target indication, *in*!, which sends an object to all inner membranes. Such a synchronization was already considered in neural-like P systems where it corresponds to the target *go*. It would be interesting to investigate what happens if this target is not used. We conjecture that a synchronization would be impossible in this case.

The study of the synchronization algorithms for different classes of P systems is important as it permits to implement different synchronization strategies which are important for such a parallel device as P systems. In particular, with this approach it is possible to simulate P systems with multiple global clocks by P systems with one global clock. It is particulary interesting to investigate the synchronization problem for P systems which cannot create new objects, for example for P systems with symport/antiport.

Acknowledgments. The first and the third authors acknowledge the support of the Science and Technology Center in Ukraine, project 4032.

References

1. Bernardini, F., Gheorghe, M., Margenstern, M., Verlan, S.: How to synchronize the activity of all components of a P system? In: Vaszil, G. (ed.) Proc. Intern. Workshop Automata for Cellular and Molecular Computing, MTA SZTAKI, Budapest, Hungary, pp. 11–22 (2007)
2. Goto, E.: A minimum time solution of the firing squad problem. Harward Univ. Course Notes for Applied Mathematics, 298 (1962)
3. Kruskal, J.B.: On the shortest spanning subtree of a graph and the traveling salesman problem. Proc. American Mathematical Society 7, 48–50 (1956)
4. Mazoyer, J.: A six-state minimal time solution to the firing squad synchronization problem. Theoretical Science 50, 183–238 (1987)
5. Minsky, M.: Computation: Finite and Infinite Machines. Prentice-Hall, Englewood Cliffs (1967)
6. Păun, G.: Membrane Computing. An Introduction. Springer, Heidelberg (2002)
7. Prim, R.C.: Shortest connection networks and some generalizations. Bell System Technical Journal 36, 1389–1401 (1957)
8. Spellman, P.T., Sherlock, G.: Reply whole-cell synchronization - effective tools for cell cycle studies. Trends in Biotechnology 22, 270–273 (2004)
9. Umeo, H., Maeda, M., Fujiwara, N.: An efficient mapping scheme for embedding any one-dimensional firing squad synchronization algorithm onto two-dimensional arrays. In: Bandini, S., Chopard, B., Tomassini, M. (eds.) ACRI 2002. LNCS, vol. 2493, pp. 69–81. Springer, Heidelberg (2002)
10. Schmid, H., Worsch, T.: The firing squad synchronization problem with many generals for one-dimensional CA. In: Proc. IFIP TCS 2004, pp. 111–124 (2004)
11. Yunès, J.-B.: Seven-state solution to the firing squad synchronization problem. Theoretical Computer Sci. 127, 313–332 (1994)
12. The P systems web page, `http://ppage.psystems.eu`