Bull. Nov. Comp. Center, Comp. Science, 2 (1993), 1–11

Correctness of mixed cellular computations*

S.M. Achasova

Operation of parallel substitutions over cellular arrays in mixed (synchronous-asynchronous) mode is studied. Correctness conditions for parallel substitution systems in this mode of execution are stated.

1. Introduction

This paper discusses the problem of correctness of cellular computations in the context of the Parallel Substitution Algorithm theory. The Parallel Substitution Algorithm (PSA) is an abstract model of distributed (cellular) computations [1]. A PSA is specified by a set of parallel substitutions which operate over a cellular array in parallel (everywhere and at the same time).

Two ideas are in the heart of the PSA: naming functions which allow to specify parallel interactions of any type and a context which serves to represent control of a computation process in time. They render this model particularly appropriate for organizing cellular computations.

Attractive for the practical applications is the class of systems of stationary parallel substitutions. Unlike general-type parallel substitutions, which may either decrease or increase the cardinality of a cellular array at hand, stationary substitutions result in no change of the cardinality of the array.

The paper only concernes with systems of stationary parallel substitutions. In [2], [3] the termination, determinacy and correctness conditions of stationary parallel substitution systems (in [1–4] the term "parallel microprograms" is used instead of "systems of parallel stationary substitutions") in the synchronous and asynchronous modes of execution are studied. Considering these modes as the pure modes of execution of parallel substitutions, this paper focuses on a mixed mode which combines the features of

*This work was supported by Russian Fund of Fundamental Research

the synchronous and asynchronous modes. In this mode, at each computation step one from the applicable substitutions or any group of them can be executed. It is just how applicable substitutions can be executed in a cellular array (in a net of automata which interprets systems of parallel substitutions in one or other mode) without clock pulses. In other words, in transient regime can occur such a situation when some automata of a net change their states for a fixed time domain and thus, effect of synchronous execution of some substitutions appears.

2. Basic concepts

A cellular array under processing is represented by a set of pairs (a_i, m_i) where a_i is a data item from an alphabet A and m_i is a place of the data item, $\{m_i, i = 1, 2, \ldots\} = M$. The sets A and M are finite. A pair (a_i, m_i) is termed a cell, a_i is a state of the cell, m_i is a name of the cell. The name of a cell is specified by integer coordinates $m_i = m_i^1, \ldots, m_i^n$ and a cellular array is viewed as an integer grid of any dimensionality in theory, and of 1D, 2D, 3D in practice. *A cellular array* is a finite set of cells with no pair of cells having one and the same name.

An expression

$$S_1 * S_2 \rightarrow S_3$$

is an elementary substitution, where S_1, S_2, S_3 are cellular arrays, S_1 is *the base*, S_2 is *the context* of a substitution. The base and the right-hand side S_3 of a substitution are of the same cardinality and contain cells with the same set of names. This is the condition of stationary substitutions.

A substitution is applicable to a cellular array W if the left-hand side $S_1 * S_2$ is contained in W, i.e., occurring the base and context cells in W is the applicability condition of a substitution in W. The substitution is executed by substituting S_3 for S_1, i.e., the base cells change their states and thus the result is $W' = W \setminus S_1 \cup S_3$.

To represent substitutions of the same type by one and the same expression, a substitution is generalized to *a parallel substitution*

$$\Theta(m) : S_1(m) * S_2(m) \rightarrow S_3(m),$$

here m is a variable name from M, $S_1(m)$, $S_2(m)$, $S_3(m)$ are *configurations* which are the following expressions:

$$\begin{aligned} S_1(m) &= \{(a_1, \varphi_1(m)), (a_2, \varphi_2(m)), \ldots, (a_p, \varphi_p(m))\}, \\ S_2(m) &= \{(b_1, \psi_1(m)), (b_2, \psi_2(m)), \ldots, (b_q, \psi_q(m))\}, \\ S_3(m) &= \{(c_1, \varphi_1(m)), (c_2, \varphi_2(m)), \ldots, (c_p, \varphi_p(m))\}, \end{aligned}$$

where φ_i, ψ_j, $(i = 1, \ldots, p, j = 1, \ldots, q)$ are *naming functions* such that for any m $\varphi_i(m) \neq \varphi_j(m)$, $\psi_i(m) \neq \psi_j(m)$, $i \neq j$, in addition, $S_1(m) * S_2(m)$ is also a configuration, i.e., $\varphi_i(m) \neq \psi_j(m)$ for all i, j. $S_1(m)$ is termed the base and $S_2(m)$ is termed the context of a substitution.

The pair (*a cell state, a naming function*) in a configuration is a component of the configuration. A cellular array resulting from substituting a certain $m_t \in M$ in a configuration is referred to as a configuration element and denoted $S(m_t)$. In this paper only coordinate shift functions are taken as the naming functions. One of the naming functions is convenient to be taken equal to the trivial function $f(m) = m$, it is agreed that this is a function of the base, the corresponding component of the base configuration is termed the central one.

The expression

$$\vartheta(m_t) : S_1(m_t) * S_2(m_t) \rightarrow S_3(m_t)$$

is a substitution $\theta(m)$ for a specific name $m = m_t$ (i.e., the elementary substitution), which is further called a microoperation. The configuration elements $S_1(m_t)$ and $S_2(m_t)$ are termed the base and the context of the microoperation $\vartheta(m_t)$, respectively. The cell of the base $S_1(m_t)$, which corresponds to the central component of the configuration $S_1(m)$, is called central, as well. This cell has the name m_t.

A parallel substitution $\theta_i(m) : S_{i1}(m) * S_{i2}(m) \rightarrow S_{i3}(m)$ (in what follows it is merely a substitution) is applicable to a cellular array W if among names of the cells belonging to W, there is a name m_t such that $S_{i1}(m_t) \cup S_{i2}(m_t) \subseteq W$ (i.e., the microoperation $\vartheta_i(m_t)$ is applicable to W). If the number of these names in W is more than one (let there be a set $\{m_1, \ldots, m_t\}$), then we consider the applicable substitution to associate with a set of microoperations $\{\vartheta(m_1), \ldots, \vartheta(m_t)\}$.

A finite set of substitutions $\Phi = \{\theta_1, \ldots, \theta_v\}$ is called *a Parallel Substitution System.*

Example 1. A parallel substitution system Φ_1 realizing a parallel binary adder for an arbitrary number of items comprises two substitutions [1], [4].

$$\theta_1 : \{(1, \langle i, j\rangle)(1, \langle i+1, j\rangle)(0, \langle i, j-1\rangle)\} * \{(0, \langle i-1, j-1\rangle)(0, \langle i-1, j\rangle)\} \rightarrow \{(0, \langle i, j\rangle)(0, \langle i+1, j\rangle)(1, \langle i, j-1\rangle)\},$$

$$\theta_2 : \{(1, \langle i, j\rangle)(0, \langle i+1, j\rangle)\} * (0, \langle i-1, j\rangle) \rightarrow \{(0, \langle i, j\rangle)(1, \langle i+1, j\rangle)\}.$$

In Figure 1 there is a visual representation of the substitutions. In Figure 2 a two-dimensional cellular array (for $A = \{0, 1\}$ and $M = \{\{1, 2, 3, 4\} \times$

$\{1,2,3\}\}$) is shown, to which the substitution $\theta_1 \in \Phi_1$ is applicable. Two microoperations $\vartheta_1(3,2)$, $\vartheta_1(2,3)$ therewith associate with the applicable substitution.

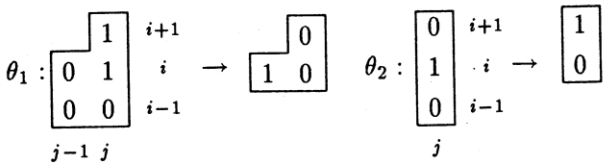

Figure 1

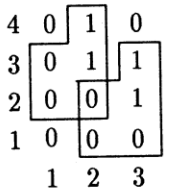

Figure 2

A parallel substitution system may be executed in the synchronous and asynchronous modes.

The synchronous mode of execution of a parallel substitution system Φ in a cellular array W is the following iterative procedure. Let a cellular array W_i be the result of i-th step of the procedure, then:

1) if no substitution of Φ is applicable to W_i, then W_i is the result,
2) if there exists a subset of substitutions $\{\theta_{k_1}, \ldots, \theta_{k_g}\} \subset \Phi$ applicable to W_i for subsets of the names $M_{k_1}, \ldots, M_{k_g}$, respectively, then W_i is substituted for W_{i+1}, which is obtained from W_i by executing the microoperations $\vartheta_{k_1}(m)$ for all $m \in M_{k_1}$, $\vartheta_{k_2}(m)$ for all $m \in M_{k_2}$, $\ldots$, $\vartheta_{k_g}(m)$ for all $m \in M_{k_g}$, i.e., by removing the bases $S_{k_i 1}(m)$, $m \in M_{k_i}$, $i = 1, \ldots, g$, of all applicable microoperations from W_i and next by adding the right-hand sides $S_{k_i 3}(m)$, $m \in M_{k_i}$, $i = 1, \ldots, g$, of the microoperations,
3) if W_{i+1} is not a cellular array (i.e., there are cells with the same name in it), then the procedure is stopped without a result.

A parallel substitution system Φ is called *contradictory* if the synchronous execution of Φ mets the condition of Item 3 of the above procedure at least for one cellular array W. In other words, this is the case when in Φ there is at least two substitutions which being simultaneously executed change the state of one and the same cell to different ones.

The asynchronous mode of execution of a parallel substitution system Φ in a cellular array W is the following iterative procedure. Let a cellular array W_i be a result of i-th step of the procedure, then:

1) if no substitution of Φ is applicable to W_i, then W_i is a result,

2) if there are several substitutions of Φ applicable to W_i, then W_i is transformed to a cellular array W_{i+1} by executing a single microoperation, anyone from the subset of microoperations applicable to W_i.

A set of cellular arrays, which can be produced from an initial cellular array W by executing a parallel substitution system in the asynchronous (synchronous, in this case a parallel substitution system is assumed to be non-contradictory) mode, together with the succession relation of cellular arrays is called *an asynchronous (synchronous) computation* and is denoted by $\tilde{\Phi}(W)$ ($\bar{\Phi}(W)$). The succession relation graph is referred to as *an asynchronous (synchronous) computation graph*. A sequence of cellular arrays, which forms a path from an initial vertex of a graph to a terminal one is termed *a computation realization*.

A computation (asynchronous or synchronous) is *terminating* if its graph contains no loop. *A parallel substitution system is terminating in the asynchronous (synchronous) mode* if a asynchronous (synchronous) computation by the substitution system for any initial cellular array is terminating.

A terminating computation (asynchronous or synchronous) whose graph has a single terminal vertex (it is a vertex corresponding to a cellular array to which no substitution is applicable) is called *determinate*.

Example 2. Figure 3, in which graphs of synchronous and asynchronous computations by the parallel substitution system Φ_1 are shown, illustrates the above concepts. Both computations are terminating. The first computation graph has only one realization, the second has three realizations which converge to the only terminal vertex and hence, the computations are determinated.

A parallel substitution system is determinate in the synchronous (asynchronous) mode of execution if a synchronous (asynchronous) computation by the substitution system is determinate for any initial cellular array.

It is worth to notice that the notion of determinacy of a parallel substitution system in the synchronous mode is exhausted by the notion of non-contradictoriness.

A key notion for finding the determinacy conditions of a parallel substitution system is a concept of intersection of substitutions. Two substitutions θ_i and θ_j intersect if there exists at least one pair of microoperations $\vartheta_i(m')$ and $\vartheta_j(m'')$ such that their left-hand sides $S_{i1}(m') * S_{i2}(m')$ and $S_{j1}(m'') * S_{j2}(m'')$ have common cells and $S_{i1}(m') \cup S_{i2}(m') \cup S_{j1}(m'') \cup S_{j2}(m'')$ is a cellular array (i.e., there are no cells with the same names in

it). Considering that the naming functions are shifts, all pairs of microoperations $\vartheta_i(m_t)$ and $\vartheta_j(m_g)$ such that the difference of the names of the central cells $m_g - m_t = k$, where $k = m'' - m'$, intersect.

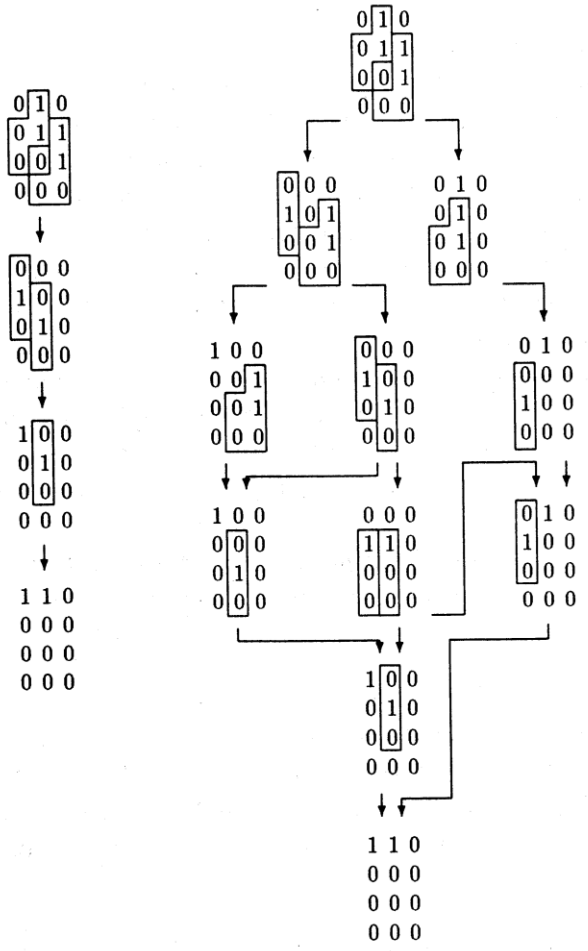

Figure 3

Example 3. Figure 4 shows the intersections (and self-intersections) of the binary adder substitutions with the associated values of the vector k.

Certain types of the intersections may be a reason for arising nondeterminate synchronous and asynchronous computations. They are termed critical intersections. For synchronous computations, a base-base intersection of microoperations $\vartheta_i(m')$ and $\vartheta_j(m'')$ (the common cells of the intersected microoperations belong to their bases) is critical if the expression

$$S_{i3}(m') \cup S_{i2}(m') \cup S_{j3}(m'') \cup S_{j2}(m'')$$

is not a cellular array, i.e., in it there are at least two cells with the same names and different states. An intersection of this type is called contradictory.

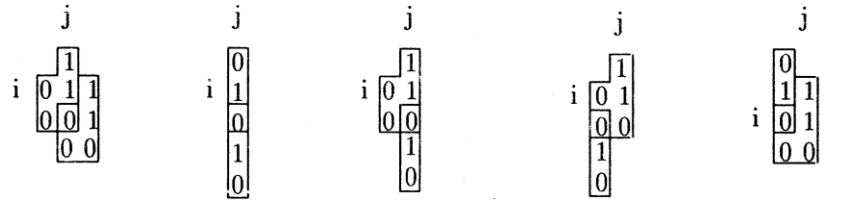

Figure 4

For asynchronous computations, among critical intersections are base-base ones and base-context intersections (common cells belong to the base of one microoperation and to the context of the other). When two microoperations intersect critically, in the asynchronous computation the applicability condition of one of the microoperations is broken after the other has been executed and, as a result, being branched, the computation does not have a common end for all its realizations. Context-context intersections are called safe. In the synchronous mode, context-context intersecting microoperations are executed simultaneously not hindering each other. In an asynchronous mode, executing one of context-context intersecting microoperations does not break the applicability condition for the other.

In [2], [3] the determinacy conditions are stated. *A parallel substitution system in the synchronous (asynchronous) mode of execution is determinate iff the synchronous (asynchronous) computations by the substitution system are determinate for all critical words* (these are cellular arrays which are made up from the left-hand sides of critically intersecting microoperations, for example, the critical words for the binary adder substitution system are shown in Figure 4). For instance, the binary adder substitutions have no base-base intersections at all, that is why the binary adder substitution system is determinate in the synchronous mode. In addition, the asynchronous computations for all critical words for this substitution system are determinate, from this it follows that the substitution system is determinate in the asynchronous mode, too.

A determinate asynchronous computation by a parallel substitution system for a cellular array W is correct if its result is equal to that of the synchronous computation by this substitution system for the same cellular array. A parallel substitution system is correct in the asynchronous mode if

an asynchronous computation by the substitution system is correct for any initial cellular array. For example, the parallel binary adder substitution system is correct in the asynchronous mode.

3. Mixed mode

We define *the mixed mode of execution of a non-contradictory parallel substitution system* as an iterative procedure, whose i-th step is the transformation of a cellular array W^{i-1} to W^i by simultaneous executing some applicable microoperations, but not necessarily all applicable ones at once. It can be any group of applicable microoperations, including a group consisting of one microoperation or of all applicable microoperations.

Like the asynchronous computation by a parallel substitution system, *a mixed computation* is defined as a set of cellular arrays, which are produced from an initial cellular array W by executing a parallel substitution system Φ in the mixed mode, together with the succession relation of the cellular arrays. The mixed computation is denoted by $\hat{\Phi}(W)$. In Figure 5 the graph of a mixed computation by the binary adder substitution system Φ_1 is shown.

In a mixed computation graph all realizations of the asynchronous and synchronous computations for the one and same initial cellular array are contained as the realizations of the mixed computation for the same cellular array.

By analogy with the asynchronous computation, we define a terminating, determinate and correct mixed computation.

A mixed computation $\hat{\Phi}(W)$ is *terminating* if the graph of the computation contains no loop. *A parallel substitution system* Φ is *non-terminating in the mixed mode* if there exists at least one cellular array W such that the mixed computation $\hat{\Phi}(W)$ is non-terminating.

A terminating mixed computation $\hat{\Phi}(W)$ is *determinate* if the graph of it's mixed computation has the only one terminal vertex, and *the determinate mixed computation* is *correct* if the cellular array of the terminal vertex is equal to the result of the synchronous computation $\bar{\Phi}(W)$. For instance, the mixed computation of Figure 5 is terminating, determinate and correct.

Note that a determinate mixed computation is always correct, since among the realizations of a mixed computation $\hat{\Phi}(W)$ there is always the realization of the synchronous computation $\bar{\Phi}(W)$. In the context of the fact we define only the notion of correctness of a parallel substitution system in the mixed mode. *A parallel substitution system* is *correct in the mixed*

mode if a mixed computation by the substitution system is correct for any initial cellular array.

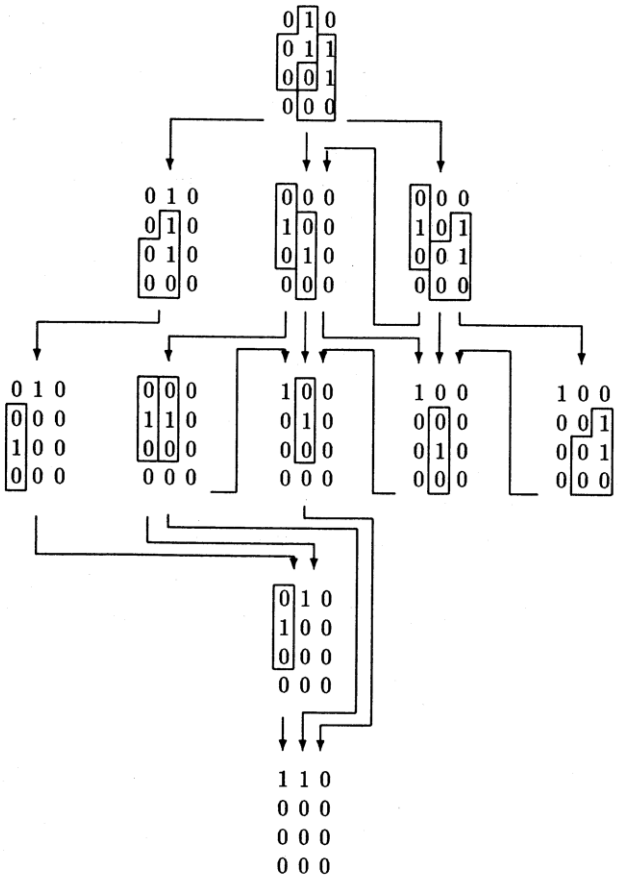

Figure 5

4. Correctness of parallel microprograms in the mixed mode

We formulate two theorems which establish the conditions of correctness and termination of a parallel substitution system in the mixed mode through the same conditions in the synchronous and asynchronous modes.

Theorem 1. *A parallel substitution of system* Φ *is correct in the mixed mode iff it is correct in the asynchronous mode.*

Proof. Necessity is obvious. Sufficiency is proved by contradiction. We suppose that the condition of the theorem is fulfilled, i.e., the parallel substitution of system Φ is correct in the asynchronous mode. Then let us take the case that there was found a cellular array W for which the mixed computation $\hat{\Phi}(W)$ turned out to be incorrect, i.e., a realization of $\hat{\Phi}(W)$ ends with a cellular array not equal to the result of the synchronous computation $\bar{\Phi}(W)$. It means that in the asynchronous computation $\tilde{\Phi}(W)$ there is the realization which ends with the same (i.e., incorrect) result. Indeed, since any realization of a mixed computation has its respective realization (and maybe not the only one) then in the asynchronous computation there is a realization also ending with the incorrect result. But this contradicts the condition of the theorem. From this it follows that the assumption of existence of a cellular array W for which the mixed computation is incorrect was in error. □

In [2], [3] the conditions of restricted termination of a parallel substitution system in the synchronous and asynchronous modes are established, which ensure termination of a parallel substitution system in these modes for a cellular array, whose cardinality is not greater than a fixed one. The termination conditions are related to constructing a so called provocation graph for each substitution of the substitution system in question. The looplessness of every provocation graph is the termination condition of the substitution system.

Theorem 2. *A parallel substitution system Φ is terminating in the mixed mode for any initial cellular array, whose cardinality is not greater than a fixed one, iff it is terminating in both the synchronous mode and the asynchronous one for any cellular array whose cardinality is not greater than the same fixed value.*

Proof. Necessity is obvious. Like in Theorem 1, sufficiency follows from the fact that any realization of the mixed computation has its respective one in the asynchronous computation for the same initial cellular array. □

References

[1] O.L. Bandman, S.V. Piskunov, Parallel substitution algorithms as a model for distributed computations, J. New Gener. Comput. Syst., 4, 1991, 1,3–18.

[2] S.M. Achasova, Correctness of interpretations of parallel substitution systems, J. New Gener. Comput. Syst., 4, 1991, 1,19–27.

[3] S.M. Achasova, O.L. Bandman, Correctness of Parallel Computation Processes, Nauka, Novosibirsk, 1990 (in Russian).

[4] O.L. Bandman, S.V. Piskunov, Parallel microprogramming as a tool for multimicroprocessor system design, Lect. Notes in Comp. Sci., 342, 1989, 57–68.