Conformons-P Systems*

Pierluigi Frisco\(^1\) and Sungchul Ji\(^2\)

\(^1\) L.I.A.C.S., Leiden University
Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
pier@liacs.nl

\(^2\) Dep. of Pharm. and Toxic., Rutgers University
Piscataway, N.J. 08855, U.S.A.
sji@echsi.rutgers.edu

Abstract. The combination of a theoretical model of the living cell and membrane computing suggests a new variant of a computational model based on membrane-enclosed compartments defined and presented in this paper for the first time. This variant is based on simple and basic concepts: conformons, a combination of information and energy; locality of the interactions of conformons, permitted by the presence of membrane-enclosed compartments; communication via conformons between membrane-enclosed compartments.

The computational power of this new system is sketched. Possible other variants of this model and links with Petri nets are outlined.

1 Introduction

One of the aspects of natural computing is to interpret all processes present in a cell as computational processes. The extrapolation of some basic principles of the functioning of a cell and their definition from a mathematical point of view have led to the creation of theoretical computational models. This contribution expounds the theoretical facet of biomolecular computing considering also the investigation of the generative capability, complexity, universality, etc, of such models.

In our research we have combined some basic principle of biocybernetics [9], a general molecular theory of living processes, with membrane computing, a novel distributed parallel way of computing.

In [9] biocybernetics is formulated on the basis of principles, concepts and analogies imported from physics, chemistry and cybernetics. The most novel physical concept to emerge in that theory is that of gnergy, a hybrid physical entity composed of free energy and information that is postulated to be ultimately responsible for driving all molecular machines. Discrete physical entities carrying gnergy are called gnergons and there are two examples of gnergons identified in biology: conformons, sequence-specific conformational strains of biopolymers.

* Work partially supported by contribution of EU commission under The Fifth Framework Program, project "MolCoNet" IST-2001-32008

© Springer-Verlag Berlin Heidelberg 2003
and IDS (intracellular dissipative structures) intracellular chemical and mechanical stress gradients and waves. Conformons and IDSs are utilized to formulate what appears to be the first coherent theoretical model of the living cell known as the Bhopalator [8, 12].

Conformons (as better explained in Section 2) are visualized as a collection of a small number of catalytic residues of enzymes or segments of nucleic acids that are arranged in space and time with appropriate force vector so as to cause chemical transformations or physical changes on a substrate or a bound ligand.

Membrane computing is based on membrane systems (also called P systems) a new class of distributed and parallel computing devices introduced in [18]. In that paper the author considers systems based on a hierarchically arranged, finite cell-structure consisting of several cell-membranes embedded in a main membrane called the skin. The membranes delimit regions where objects, elements of a finite set, and evolution rules can be placed.

The objects evolve according to given evolution rules associated to a region, and they may also move between regions. A computation starts from an initial configuration of the system, defined by a cell-structure with objects and evolution rules in each cell, and terminates (halts) when no further rule can be applied.

It is possible to assign a result to a computation in two ways: (1) a multiset, considering the multiplicity of objects present in a specific (output) membrane in a halting configuration, or (2) a set of strings, composed of the strings over a specific alphabet sent out of the system. Combining the outputs of each possible computation the behaviour of the system is obtained, a multiset-language (a set of vectors) or a string-language (a set of strings).

In [18] the author examines three ways to view P systems: transition, rewriting and splicing P systems. Starting from these, several variants were considered (see for instance [2, 14, 15, 19, 21]). Each of these variants has been shown to generate recursively enumerable sets or vectors of natural numbers. The latest information about P systems can be found at the url http://psystems.disco.uninibs.it/.

Conformons-P systems, the variant of P systems presented in Section 3, consider as objects conformons - an ordered pair of name and value. This way to consider objects is substantially different from the others described in the literature. Until now an object has been considered either as simple entity without internal structure or a string, that is an entity with a well defined structure. Objects considered in our research may be placed in between these two categories as the only structure related to the name is the value related to it.

The main result of this paper, presented in Section 4, states that conformons-P systems with priorities generate precisely the family of recursively enumerable sets of natural numbers.

In Section 5 we outline possible variants of conformons-P systems with priorities. Moreover in that section we delineate how this model may be used to describe and study the massive parallelism so fundamental in biomolecular computing, and to which fields the computation based on conformons might be extended.
Due to lack of space we omit here the proofs of our results. The interested reader may refer to [4].

2 Conformons in Molecular Biology

The developments, during the second half of the last century, of the automata theory, neural nets, genetic programming, evolutionary computation, and most recently DNA-based molecular computing all attest to the fecundity of the interaction between computer science and biology. The computer science-biology interactions are a two-way street: not only can biology serve as a rich source of ideas and inspirations for computer scientists aspiring to discover and design novel computational frameworks but may also absolutely require computer science to formalize and test its theories in order to solve the mysteries of life on the molecular and cellular levels. The present contribution is primarily concerned with the former aspect of the computer science-biology interactions, and the latter aspect has been dealt with elsewhere by one of us [8, 9, 10, 12].

One of the basic concepts to develop in molecular biology during the past three decades is the notion of conformons, defined as sequence-specific mechanical strains embedded in biopolymers, such as DNA supercoils and protein conformational deformations, that provide both the free energy and information needed for biopolymers to drive molecular processes essential for life [7, 11]. The free energy content of conformons has been estimated to be in the range of 5 ~10 Kcal/mole in proteins and 500 ~ 2,000 Kcal/mole in DNA, while the information content per conformon has been calculated to be in the range of 20 ~ 40 bits (note that 20 ~ 200 bits as reported in [10] is an error) in proteins and 200 ~ 600 bits in DNA [9, 11].

Conformons and conformon-like entities invoked in the biological literature during the past three decades have been classified into 10 families based on their biological functions, including the origination of life, thermal fluctuations, substrate and product bindings, formation of the transition-state complex, free energy transfer, DNA replication, timing in proteins, and timing in DNA [11]. Given such a multiplicity of conformon families, each with a large number (from $10^5$ to $10^6$?) of members, it is possible, at least in principle, to account for all living processes in the cell in molecular terms. This has led to the postulates (1) that the number of conformons active in and utilized by living cells are finite in number and (2) that conformons are quanta of biological actions, akin to quanta of action in quantum mechanics [11].

Another fundamental feature of the living cell, postulated to be the smallest molecular computing system in nature [10], is the biological membranes consisting of a phospholipid bilayer of about 50 angstroms (i.e., $50 \times 10^{-8}$ cm) in thickness with many different kinds of proteins, either attached to its surface or deeply embedded in it. The basic function of biomembranes is to divide the Euclidean space into multiple compartments, to be referred to as membrane-enclosed compartments or simply as membranes when there is no danger of ambiguity. The principle of biological membranes began to be capitalized in developing new
computing paradigms during the past several years, giving rise to the P system [18, 19, 20, 21].

3 Basic Definitions

As sketched in the Introduction, the basic ideas underlying the concept of conformation and the interaction between two conformations have been of inspiration for us to define a new computability model. What in bioncybernetics is a pair of information and free energy in this section is defined from a mathematical point of view as an ordered pair name-value, the interaction between two conformations is modeled as passage of all or a part of the value from one pair to another.

Let V be a finite alphabet and N the set of natural numbers. A conformation is an element of the relation name-value: V × N (where N = N ∪ {0}) denoted by [X, x]. We will refer to x as the value of X and to X as the name of the conformation [X, x]. The symbol X will also indicate the conformation itself; the context will help the reader to understand when we refer only to the name aspect of the conformation or to the whole conformation. Moreover let \( r = (A, e, B) \), \( A, B \in V \), \( e \in N \), be a rule (also indicated as \( A \xrightarrow{e} B \)) defining the passage of (part of) the value from one conformation to another so that:

\[
\begin{align*}
[A, a] & \quad [A, a - e] \\
\Rightarrow_r & \\
[B, b] & \quad [B, b + e]
\end{align*}
\]

with \( a, b \in N \), \( a \geq e \) indicating that \([A, a]\) and \([B, b]\) interact according to \( r \). Informally this means that \( e \) is subtracted from the value of the conformation (with name) \( A \) and \( e \) is added to the value of the conformation (with name) \( B \) only if the value of \( A \) is at least \( e \).

A multiset (over \( V \)) is a function \( M : V \rightarrow N \cup \{+\infty\} \); for \( a \in V \), \( M(a) \) defines the multiplicity of \( a \) in the multiset \( M \). We will indicate this also with \( (a, M(a)) \). In case the multiplicity of an element of a multiset is \( 1 \) we will indicate just the element. The support of a multiset \( M \) is the set \( \text{supp}(M) = \{a \in V \mid M(a) > 0\} \). Informally we will say that a symbol belongs to a multiset if it belongs to the support of \( M \).

Let \( M_1, M_2 : V \rightarrow N \) be two multisets. The union of \( M_1 \) and \( M_2 \) is the multiset \( M_1 \cup M_2 : V \rightarrow N \cup \{+\infty\} \) defined by \( (M_1 \cup M_2)(a) = M_1(a) + M_2(a) \), for all \( a \in V \). The difference \( M_1 \setminus M_2 \) is here defined only when \( M_2 \) is included in \( M_1 \) (which means that \( M_1(a) \geq M_2(a) \) for all \( a \in V \)) and it is the multiset \( M_1 \setminus M_2 : V \rightarrow N \cup \{+\infty\} \) given by \( (M_1 \setminus M_2)(a) = M_1(a) - M_2(a) \) for all \( a \in V \).

A conformons-P system with priorities of degree \( m, m \geq 1 \), is a construct \( \Pi = (V, \mu, l, a, l_1, \ldots, l_m, R_1, \ldots, R_m) \), where \( V \) is an alphabet; \( \mu = (N, E) \) is a directed labeled graph underlying \( \Pi \). The set \( N \subseteq N \) contains vertices, for simplicity we define \( N = \{1, \ldots, m\} \). Each vertex in \( N \) defines a membrane of the system \( \Pi \). The set \( E \subseteq N \times N \times \text{pred}(N) \) defines directed labeled edges between vertices, indicated by \( (i, j, \text{pred}(n)) \) where for each \( n \in N \) we consider
pred(n) = \{ \geq n, \leq n, = n \} set of predicates. For x ∈ \mathbb{N}_0, p \in pred(n), p(x) may be (\geq n)(x) or (\leq n)(x) or (= n)(x) (only one of them), indicating x ≥ n, x ≤ n and x = n respectively. The symbol l ∈ N defines the final membrane while a ∈ N the acknowledgment membrane that is initially empty.

The multisets I_i over V × \mathbb{N}_0 \cup \{+∞\}, i ∈ N, contain conformons; R_i, i ∈ N, are finite sets of rules.

Two conformons present in a membrane i may interact according to a rule r present in the same membrane such that the multiset of conformons M_i changes into M'_i. So, for i ∈ N, [A, a, |B, b|] ∈ M_i and r = (A, e, B) ∈ R_i, A, B ∈ V, a, b, e ∈ \mathbb{N}_0 we have what indicated in (1) so that

M'_i = (M_i \{[A, a, |B, b|]\} \cup \{[A, a - e, |B, b + e|]\}.

A conformon [X, x] present in a membrane i may pass to a membrane j if it cannot interact with any other conformon present in the same membrane, if (i, j, p) ∈ E and p(x) holds, changing the multisets of conformons M_i and M_j to M'_i and M'_j respectively. In this case M'_i = M_i \{[X, x]\} and M'_j = M_j \{[X, x]\}.

The fact that the passage of an object to a membrane is regulated by some features present in the membranes is already discussed by others in literature when membrane with electrical charge and variable thickness have been used [20] or only communication was used to compute [22].

The application of a rule and the passage of a conformon from one membrane to another are the only operations that may be performed by a conformons-P system with priorities. A conformon present in a membrane may be involved in one of these two operations or none of them.

It is important to note that the interaction between conformons has priority on the passage of a conformon to another membrane. This anyhow does not mean that if a conformon may interact with another one or pass to another membrane it has to. So the feature “all the objects which can evolve should evolve”, present in most of the other variants of P system introduced until now, is not applied here. The presence of such a universal clock, common in digital computers but not in biological processes, is very powerful from a computational point of view as it forces the system to a maximal parallelism. Not considering it does not limit the parallelism of conformons-P systems with priorities as it is possible that an operation is performed when it can be performed.

The possibility to perform one of the two allowed operations in a same membrane or none of them let conformons-P systems with priorities to be non-deterministic. Non-determinism may also arise from the configurations of a conformons-P system with priorities if in a membrane a conformon may interact with more than one conformon.

A configuration of II is an m-tuple (M_1, ..., M_m) of multisets over V × \mathbb{N}_0 \cup \{+∞\}. The m-tuple (L_1, ..., L_m) is denoted as initial configuration. For two configurations (M_1, ..., M_m), (M'_1, ..., M'_m) of II we write (M_1, ..., M_m) ⇒ (M'_1, ..., M'_m) indicating a transition from (M_1, ..., M_m) to (M'_1, ..., M'_m) that is the parallel application of operations or of no operation in each membrane of μ. If no operation is applied to a multiset M_i then M_i = M'_i. The reflexive and transitive closure of ⇒ is indicated by ⇒*.
A computation is a finite sequence of transitions between configurations of a system \( H \) starting from \( (L_1, \ldots, L_m) \). Initially \( L_a = \emptyset \). The result of a computation is given by the multisets of conformons present in membrane \( l \) and having elements of \( T \) as name when a (generic) conformon is present in membrane \( a \). When this happens the computation halts, that is no other operation is performed. This feature is new in the area of membrane computing: it provides an alternative to the way of defining successful computations as halting computations. When this happens the multisets of all such conformons present in membrane \( l \) define the language generated by \( H \), indicated by \( L(H) \).

Formally:

\[
L(H) = \{ \text{supp}(M_l) \mid (L_1, \ldots, L_m) \Rightarrow (M'_1, \ldots, M'_n) \Rightarrow (M_1, \ldots, M_m), \text{supp}(M'_a) = \emptyset, \text{supp}(M_a) \neq \emptyset \}.
\]

In Section 4 we sketch the proof that conformons-P systems with priorities are computationally complete. In the proof of this result we need the notion of program machines.

Non-rewriting Turing machines were introduced by M. L. Minsky in [16] and then reconsidered in [17] under the name of program machines. After their introduction such machines and some variants of them have been studied under different names: in [5] they were called (multi)counter machines, in [1] multi-pushdown machines, in [13] register machines and in [6] counter automata. Such devices have counters (also called registers) each of infinite capacity recording a natural number or zero. Simple operations can be performed on the counters: addition of one unit and conditional subtraction of one unit. After each of these operations the machine may change state. The main difference between the original models and some of the subsequent variants indicated above is that the latter may have a read only tape where the input is recorded. In the model introduced by M. L. Minsky and considered by us such tape is not present and the input is recorded as a number in one of the counters of the machine.

Formally a program machine with \( n \) counters \( (n \in \mathbb{N}) \) is defined as \( M = (S, R, s_0, f) \), where \( S \) is a finite set of states, \( s_0, f \in S \) are respectively called the initial and final states, \( R \subseteq (s, op(i), a, b), s, a, b \in S, s \neq f, op(i) \in \{i_+, i_\ldots\}, 1 \leq i \leq n, \) is the set of instructions of the following form:

- \((s, i_-, a, b)\): in state \( s \) if the contents of counter \( i \) is greater than 0 then subtract 1 from that counter and change state into \( a \), otherwise change state into \( b \);
- \((s, i_+, a, a)\): in state \( s \) add 1 to counter \( i \) and change state into \( a \).

A configuration of a program machine \( M \) with \( n \) counters is given by the \( n+1 \)-tuple \((s, \mathbb{N}^n)\), \( s \in S \). Given two configurations \((s, x_1, \ldots, x_n), (t, y_1, \ldots, y_n)\) we define a computational step as \((s, x_1, \ldots, x_n) \vdash (t, y_1, \ldots, y_n)\) if \((s, op(i), a, b) \in R\) and:

- if \( op(i) = i_- \) and \( x_i \neq 0 \), then \( t = a, y_i = x_i - 1, y_j = x_j, j \neq i, 1 \leq j \leq n \);
- if \( op(i) = i_- \) and \( x_i = 0 \), then \( t = b, y_j = x_j, 1 \leq j \leq n \);
- if \( op(i) = i_+ \) then \( t = a, y_i = x_i + 1, y_j = x_j, j \neq i, 1 \leq j \leq n \).
A computation is a finite sequence of transitions between configurations of a program machine $M$ starting from the initial configuration $(s_0, x_1, \ldots, x_n)$ with $x_1 \neq 0$, $x_j = 0$, $2 \leq j \leq n$. If the last of such configurations has $f$ as state then we say that $M$ accepted the number $x_1$. The language accepted by $M$ is defined as $L(M) = \{ x_1 \in \mathbb{N} \mid M$ accepts $x_1 \}$. For every program machine it is possible to create another one accepting the same language and having all counters empty in the final state.

4 The System

In this section we discuss how a conformmons-P system with priority may generate all recursive enumerable (RE) sets of natural numbers, but first we introduce the notion of module.

A module is a group of membranes in a conformmons-P system with priority able to perform a specific task. In the figure representing conformmons-P system with priority in this paper, modules are depicted as unique vertices with a thicker line. Such modules will be elements of the set $W$ in the graph underlying a conformmons-P system with priority. Each element of $W$ will have a label indicating the kind of module. A subscript is add to differentiate labels referring to the same kind of module.

**Lemma 1.** (Splitter) There exists a module that, when a conformon $[Z, z]$ with $z \in \{ z_1, \ldots, z_s \}$, $z_i < z_{i+1}$, $1 \leq i \leq s-1$ is present in a specific membrane of it, may pass such a conformon to other specific membranes according to its value $z$.

*Sketch of the proof.* No conformon and no rule is present in the initial configuration of this module. If a conformon passes to a specific membrane $d$ a subsequent filtering on decreasing levels of values is performed via several membranes. If in a membrane the value of a conformon is smaller then a certain quantity then the conformon may pass to the next membrane, otherwise it leaves the module. \( \square \)

The number of membranes present in a splitter is equal to the number of edges outgoing this module. The label for a splitter is `spl`.

Considering that in [16] it is proved that a program machine with 2 counters may generate all RE sets of natural numbers we can reach our aim simulating such machine.

**Theorem 1.** The class of numbers generated by e-P systems with priorities coincides with the one generated by program machines.

*Sketch of the proof.* For each counter of the simulated program machine there are infinite occurrences of a specific conformon in a membrane $q$ of the conformmons-P system with priority. Every time that a unit is added to a counter an occurrence of the related conformon passes from membrane $q$ to the final one; conversely
for the inverse operation. Initially the final membrane contains the occurrences of the conformons related to the initial configuration of the simulated program machine.

Considering that the passage of conformons between membranes is determined only by the value present in a conformon, the value of specific conformons may be increased so that the conformons may move to different membranes through splitters in order to perform different tasks. The addition or the conditional subtraction of one unit to a counter are simulated via other conformons related to the states of the simulated program machine. Only one of such conformons per time may move through the system so to perform the operation associated to it.

When the passage of the program machine to a final state is simulated a conformon passes to the acknowledge membrane terminating the simulation. □

Figure 1 represents a conformons-P system with priorities with 21 membranes simulating a program machine with one counter \( c \). The simulation of program machines with more counters would not change the underlying graph of the conformons-P system with priority but its sets \( L_4 \) and \( R_4 \). In this figure conformons present in a membrane in the initial configuration are written in bold, while the others indicate in which membrane a conformon can be during the computation. What in Theorem 1 is indicated as membrane \( q \) in the picture is membrane 3; membrane 2 represent the acknowledge membrane while membrane 4 the final one. In this system one interaction per time may occur in a membrane and the nondeterminism of the system is limited only to membrane 1. It is interesting to note that the just described conformons-P system with priorities increases or decreases the value of conformons but the total amount of all values of the conformons present in it is always constant. We consider an infinite supply of conformons with value 0 in membrane 3 and the result of the computation is given by the conformons with value 0 present in membrane 4 (see details in [4] and Figure 1).

5 Final Remarks

It is possible to modify the definition of conformons-P with priorities given in Section 3 removing or adding features in order to get variants.

The simplest variant we were able to imagine is the one in which there are no priorities between the interaction and the passage of conformons to other membranes. We can also imagine that both the value and the number of conformons present in a system are finite or that a conformon cannot accumulate more that a fixed value defined by a membrane or by the total system. Moreover an operation, that is the interaction between two conformons or the passage of a conformon from one membrane to another, may subtract a finite amount from the value of the conformon involved in the operation. By the same token, we can also imagine that the value of conformons may increase upon entering membranes.
Fig. 1. The conformons-P system with priority related to Theorem 1

It is also possible to consider restrictions to the graph underlying a conformons-P system limiting it, for instance, to a tree, or to see how the power of a conformons-P system changes if maximal parallelism is present.

These variants are currently under investigation.

Also under investigation is the possibility to model the parallelism present in conformons-P systems with Petri nets. This kind of nets were introduced by C. A. Petri in his seminal PhD thesis in 1964 to model concurrent and distributed system. The parallelism, basic in the theoretical facet of biomolecular computing, has not yet been studied and formalized.
The interaction between two conformons defined in (1) in page 294, may be represented by the Petri net presented in Figure 2 where we used the notation used in the first chapter of [23].

The concept of the conformon and the filtering process performed by membranes may be interpreted in a different way than described in this paper. The value associated to a conformon may be seen also as the electrical charge, the mass, the size, the momentum, the spin, the speed or frequency of it. The interaction between conformons would allow the passage of one or more of such features from one conformon to another and membranes might allow the passage of conformons depending on one or more such parameters. Moreover a conformon may be seen not only as associated to a biopolymer but as a generic molecule or a particle akin to a photon or an electron.

References


