FUDMA Journal of Sciences (FJS)
ISSN online: 2616-1370
ISSN print: 2645 - 2944
Vol. 6 No. 3, June, 2022, pp 157 - 159
DOI: https://doi.org/10.33003/fjs-2022-0603-991

ON THE APPLICATION OF MULTISET TO DETERMINISTIC COOPERATIVE SPECIALIZATION P SYSTEM UNDER CATALYSIS

[*1]Peter, C. M., [2]Singh D., [3]Atawodi S. E.

[1]Department of Mathematical Sciences & Information Technology, Federal University, Dutsin-ma – Nigeria
[2]Department of Mathematics, Ahmadu Bello University, Zaria – Nigeria
[3]Biochemistry Department, Federal University, Lokoja, – Nigeria

*Corresponding authors' email: **macpee3@yahoo.com**

ABSTRACT
Quasi-general models of membrane computing have been designed by means of rules which represent the processes taking place in a biological cell without explicitly specifying the specialties of the rules, viz: their mixture, chemical or physical characteristics as is obtainable in a biological cell. In this paper, an attempt to model a variant of membrane computing called *specialization P system* is made. It is capable of simulating biological activities at cellular level, and as the name implies, in a specialized manner. It is shown that with only one membrane and three rules, a deterministic cooperative specialization P system under catalysis is able to characterize the family of recursively enumerable languages.

Keywords: Multiset, membrane computing, specialization rules, determinism, computational completeness

INTRODUCTION

Membrane computing is a distributed and parallel computing model introduced in Păun (2000) which aims to abstract computing ideas from the activities taking place in living cells and the way the cells interact with one another or with their environment in a hierarchy of *membranes* called *membrane structure*.*Multisets* of objects are placed in the *regions* of a typical membrane structure and are processed by *rules* which act upon them.

The ability of the system to *evolve* by the action of the rules on the objects give rise to a computing device called P system. P systems are powerful and efficient Păun (2010). Of the several variants that have emerged in the literature we present an improvement of the P system working in the accepting mode introduced in Freund and Păun (2003) where an input is provided in the form of a number and the system accepts this number or not, provided that a halting computation exists starting from the initial configuration. The input is introduced into the system in the form of the multiplicity of a specified object at the initial configuration in a pre-specified input membrane.

An important characteristic that gave rise to the variant of P system is the way in which the tasks of the rules arecarried out. Much as the biochemical activities within a living cell may be classified into their respective chemical, physical and mixture forms, the rules representing these biological activities are dedicated to separately carry out their functions in these forms. Thus, one can say that the P system introduced is of *specialized type*.

In the regions delimited by the membranes are placed rules representing chemical process, physical process and mixture. The objects, which are also placed in the regions represent chemical compounds. As is typical of most variants of P system, these objects are said to *evolve* – they can be transformed from one object to another and can combine to form another object. (See Păun (2000), Păun (2002) and Păun (2006) for details on P systems).

It is observed that with only three rules adeterministic specialization P systemhas computational completeness – itcan characterize the recursively enumerable sets of natural numbers. It is interesting to note that very simple membrane structures are enough –only onemembranesuffices. The method usedto determine computational completenessis similar to that of the accepting P system provided in Freund and Păun (2003), only that this time the rules are fewer in number.

BIOCHEMICAL INTERPRETATION OF THE RULES

We shall use the arrow symbol $\rightarrow$ for composition among chemical substancesin the sense that $a \rightarrow b$ means a substance b is composed of a substance a. Thus, if b chemically combines with c to produce a, we write $b + c \rightarrow a$, meaning that a is composed of the substances b and c.

The idea is to present a model of the biochemical processes taking place in the cell by means of multisets of objects. The changes in the substances involve chemical properties for chemical change as well as for physical change. The processes to consider include mixture, reversible chemical reaction, physical change, catalytic reaction and*derived reaction*. In modelling the rules, the alphabets (Greek or English) represent the chemical substances, the binary operation '$+$' represents the process of "combining" in a chemical reaction or a physical change as the case may be. The arrow '$\rightarrow$' points to the result of the reaction or change process, while the arrow '$\Longrightarrow$' is used where such changes do not take place. Moreover, the use of the symbol '$\emptyset$' is essential and is interpreted to mean that there exists a particular substance which has not reacted with another substance after they have been combined – a case of mixture, hence its use in conjunction with '$\Longrightarrow$'.

Mixture

One would think of representing mixture by $\alpha + \beta \Longrightarrow \beta + \alpha$. The thought of such a representation is probably with the intention that mixing two substances would render thesubstances as they were (without any form of reaction taking place). This however, is crude and inappropriate to our model. The idea is to model the process in reaction terms which should reflect in the notation as in $\alpha + \emptyset \Longrightarrow \beta$.Itrepresent a more logical concept of the representation of mixture. It means 'nothing' has reacted with α and we can still obtain β. Moreover, α cannot change to β unless a second element (or energy) is involved. Hence, the symbol $\emptyset$ denotes the absence or lack of such substance. Furthermore, $\emptyset$ cannot be omitted to avoid the impression that α underwent

a unilateral change or reaction to the substance β which is not a property of a biochemical process.

Catalytic reaction
We consider a catalytic case $\alpha + \xi \rightarrow \beta + \xi$of the chemical reaction, where ξ is a catalyst. We do not lay emphasis on the other element which reacts with α for which ξ is catalyst to the reaction of such element and α. The model is so since by the end of the reaction at least a new substance is produced – not minding the number of substances used. In other words, the state of the catalyst is the focus for the catalytic case – it is not consumed in the process.

Physical change
Physical change is modelled by $\alpha + \beta \rightarrow \alpha$. This captures the idea that α and β combine via the change process '+' which leaves the chemical properties of α unchanged, much as water transforms into water vapor by heating. Thus, again the notation is in terms of the chemical properties of the substance – leaving the chemical properties unchanged. This process is reversible and the reverse is $\alpha \rightarrow \alpha + \beta$, similar to separating heat from water vapor to produce heat and water.

Reversible chemical reaction
Chemical reaction is modelled by $\alpha + \beta \underset{rev}{\longrightarrow} \gamma$, meaning that two substances α and β react to produce the resultant substance γ. The process can be reversed under certain conditions as indicated by the rev below the arrow. Unlike in the physical change, not all chemical change is reversible, hence the need to include the notation rev.

Incomparability in reactions
It is possible that two substances cannot undergo any of the processes described above. In such a scenario, we insert # between the objects representing the substances as in $\alpha \# \beta$ for the substances represented by the objects α and β. We say the α and β are incomparable..

Derived reactions
It is possible to derive other reaction types in terms of a combination of some of the reactions already mentioned. A typical example of a *derivedreaction* is the resultant reaction that converts the initial substance to the final substance in a series of reactions. That is, if for instance, a substance a changes to b by a reaction $a \rightarrow b$ and b changes to c by $b \rightarrow c$ then the derived reaction that changes a to c is $a \rightarrow c$. In particular, by the use of reversible chemical reactions, catalytic reactions and physical change we derive a typical reaction which is of interest to the model of P system under investigation in the following proposition.

Proposition 1 *Given a reversible chemical reaction* $f_1 + a \underset{rev}{\longrightarrow} f$*, its catalytic form* $f_1 + a \rightarrow f + a$ *and its physical form* $f_1 + a \rightarrow f_1$*, there exists a derived reaction of the form* $f_1 \rightarrow f + a$ *under catalysis.*

Proof
Consider the catalytic form$f_1 + a \rightarrow f + a$and the physical change $f_1 + a \rightarrow f_1$. One of four cases must hold. That is, either$f$ with a is incomparable with f_1(i. e., $f_1 \# f + a$) or f reacts with a to produce f_1(i. e., $f + a \rightarrow f_1$) or f_1 produces f with a(i. e., $f_1 \rightarrow f + a$) or f_1 is not different from f with a(i. e., $f_1 = f + a$) . Suppose $f_1 = f + a$ thenthe two are identical and are both a product of the catalytic chemical form and the physical form. This is a contradiction of real life situation. Since a cannot react with f to produce f_1 which it reacts with to produce f by the hypothesis, then $f + a \underset{rev}{\longrightarrow} f_1$ also contradicts real life situation.Since every physical change is reversible, consider extending the reverse $f_1 \rightarrow f_1 + a$ of $f_1 + a \rightarrow f_1$ with the catalytic form $f_1 + a \rightarrow f + a$ (or applying catalyst to the reverse of the physical change). Thus, we see that $f_1 \rightarrow f + a$ is a derived reaction. Hence, f_1 and $f + a$ are not incomparable under catalysis. □

TURING COMPUTABILITY
Classical computability can be presented in various equivalent mathematical formalisms such as *Turing machines*, *register machines*, *Chomsky type-zero grammars*, *partial recursive functions* and so on (Soare, 2016). In this paper, we exploit the mechanism of a register machine. It is a device that consists of a given number of *registers* each of which can hold an arbitrarily large non-negative integers and a sequence of labeled instructions called *program*. The instructions determine how the integers in the registers can change and which instructions should be executed next after another. Formally, a typical*register machine* is a device $M = (m, B, l_0, l_h, R)$, where $m \geq 1$ is the number of registers, B is the set of instruction labels, l_0 is the intial label, l_h is the halting label and R is the set of instructions labeled by elements from B (R is also called the *program* of the machine). The labeled instructions are of the following forms:
$l_1 : (\mathrm{ADD}(r), l_2)$ (add 1 to register r and go to the instruction with label l_2),
$l_1 : (\mathrm{SUB}(r), l_2, l_3)$ (if register r is not empty, then subtract 1 from it and go to the instruction with label l_2, otherwise go to the instruction with label l_3),
l_h : HALT (the halt instruction which can only have the label l_h).
A register machine is used to recognize a number in the following manner: starting with all registers being empty, a number is introduced in a distinguished register (say, the first register), computation is made starting with the instruction labeled l_0; a number is inputted, say n in this register while all other registers hold the value 0. If the computation reaches the instruction labeled by l_h: HALT (that is, it halts), then the number is recognized, otherwise it is not recognized. The set of all numbers recognized by M is denoted by $N(M)$. It is known (see Freund and Oswald (2002) and Minsky (1967)) that register machines (with three registers only, recognize exactly the family *NRE*, of Turing computable numbers.

THE DETERMINISTIC COOPERATIVE SPECIALIZATION P SYSTEM AND ITS COMPUTATIONAL POWER
We introduce a class of membrane system considered in this paper. We show universality for the (cell-like) P system, in the accepting mode using symbol objects, processed by some specialized rules. For now, we consider a system with only one membrane and obtain universality by means of a register machine. In general, aspecialization P system (of degree $m \geq 1$)is a construct of the form

$$\Pi = (O \cup H, \mu, w_1, \ldots, w_m, R_1, \ldots, R_n)$$

where O is a finite alphabet, called objects. $H = \{h_1, \ldots, h_n\}$ is a set of labels of the membranes in Π, μ is a membrane structure of degree m, w_i is a string over O representing the multiset of objects present in region i of the membrane system, where $1 \leq i \leq m$ and R_j is the set of rules which act in the region of membrane j such that $1 \leq j \leq n$. For this work in particular, the form of the rules are as follows:

i. Rulesimulating reversible chemical reaction are of the form $u + v \underset{rev}{\longrightarrow} w$.

ii. Rule simulating derived reaction are of the form $u \rightarrow v + w$.
iii. Rule simulating mixture are of the form $u + \emptyset \Longrightarrow w$.

We denote by $NSPP_1(coo)$ the family of sets $N(\Pi)$ of numbers computed by a deterministic cooperative specializationP system with single membrane.

Theorem 1 $NSPP_1(coo) = NRE$
Proof
Consider a register machine $M = (m, B, l_o, l_h, P)$ and the set of alphabets $U = \{a_1, a_2, \ldots, a_m\}$ where a symbol a_i is associated with register i. The content of the register is represented by the number of occurrence of a_i in the P system being constructed. An accepting P system of degree 1 under catalysis is a construct of the form

$$\Pi = (O, [_1]_1, l_o, E, R_1)$$

where
$O = U \cup \{l_1, l_2, l_3\}$
$R_1 = \{(l_1 \rightarrow a_r + l_2) | \text{ for} l_1 : (\text{ADD}(r), l_2) \in R\}$
$\cup \left\{\left(l_1 + a_r \underset{rev}{\longrightarrow} l_2\right), (l_1 + \emptyset \Longrightarrow l_3) \mid \text{for} l_1 : (\text{SUB}(r), l_2, l_3) \in R\right\}$

Assume that we introduce n copies of the object a in the system of Π just like starting the work of the register machine M where n is in the first register. We show that Π simulates the work of the register machine when analyzing the input n. Each of the ADD instructions of M is simulated by the rule $l_1 \rightarrow a_r + l_2$. It introduces an object a_r together with the instruction label l_2 erasing the instruction label l_1. The SUB instruction is simulated by the two rules $l_1 + a_r \underset{rev}{\longrightarrow} l_2$ and $l_1 + \emptyset \Longrightarrow l_3$. In the first case if there exists an object a_r in the system then the reaction erases both the object a_r and the label instruction l_1 while introducing the label instruction l_2. If, on the other hand, no element a_r exists in the system, then the rule $l_1 + \emptyset \Longrightarrow l_3$ is executed. It transforms the label instruction l_1 to the label instruction l_3. We continue to simulate the instructions of M in this way. In the event where the register machine halts, that is, it reaches the label l_h then the computation for Π halts as well, and conversely. Therefore, $N(\Pi) = N(M)$. □

CONCLUSION

The model of P system presented with only three rules has been shown to generate all recursively enumerable sets of natural numbers by simulating a register machine. Moreover, such rules simulate chemical, biological, physical and other cellular activities in a specialized manner, hence, better simulating the activities of the cell. It is also necessary to state that the model of P system is an improvement to its symport and antiport counterpart presented in Freund and Păun (2003) having minimal rules and no more than one membrane.

ACKNOWLEDGEMENT

We wish to appreciate Mr. Alisi Ikechukwu Ogadinma and Mrs. Bilkisu O. Abdulrahman for their swift response fields of Applied Chemistry and Biochemistry, respectively. We also wish to thank the anonymous reviewers for their helpful comments and suggestions.

REFERENCES

Freund, R. and Oswald, M. (2002), A short note on analysing P systems with antiport rules. *Bulletin of the EATCS*, Vol. 78: pp. 231-236.

Freund, R. and Păun, Gh. (2003), On deterministic P systems. *See P Systems Web Page at http://psystems. disco. unimib. it.*

Minsky, M. L. (1967),*Computation: finite and infnite machines*. Prentice-Hall, Inc.

Păun, Gh., (2000), Computing with membranes. *Journal of Computer and System Sciences*, Vol. 61(1): 108-143.

Păun, Gh., (2002), Membrane Computing. *An Introduction*, Springer-Verlag, Berlin.

Păun, Gh., (2006). Introduction to membrane computing. In *Applications of Membrane Computing*. pp. 1-42 Springer Berlin Heidelberg.

Păun, G. (2010). A quick introduction to membrane computing. *The Journal of Logic and Algebraic Programming*, *79*(6), 291-294.

Soare, R. I. (2016). *Turing computability: Theory and applications*. Springer Berlin Heidelberg.