*Journal of Universal Computer Science, vol. 18, no. 13 (2012), 1760-1781*
*submitted: 15/11/11, accepted: 30/3/12, appeared: 1/7/12* © *J.UCS*

# A Variant of Distributed P Systems for Real Time Cross Layer Optimization

**Susan Elias and Vanaja Gokul**
(Sri Venkateswara College of Engineering, Chennai, India
susan@svce.ac.in,vanaja@svce.ac.in)

**Kamala Krithivasan**
(Indian Institute of Technology Madras, Chennai, India
kamala@iitm.ac.in)

**Marian Gheorghe**
(University of Sheffield, Sheffield, UK
m.gheorghe@dcs.shef.ac.uk)

**Gexiang Zhang**
(Southwest Jiaotong University, Sichuan, China
zhgxdylan@126.com)

**Abstract:** Membrane computing models (also known as *P Systems*) that solve optimisation problems using genetic algorithms, ant colony optimisation, quantum-inspired evolutionary algorithm and particle swarm optimisation have been defined and are efficiently used in several applications. This paper describes the design of a variant of the existing *Distributed P system (dP system)* that is augmented with new features that enable centralised monitoring and communication with all the other components of the distributed system. This proposed model is titled *Monitored Distributed P System (MDP System)* and its innovative application in performing Cross Layer Optimisation in wireless adhoc networks is also presented. In the proposed *MDP System* each node in the network is represented by a *P system* that can independently perform Cross Layer Optimisation using particle swarm optimisation. Discussions on the communication complexity of the proposed model and the experimental results presented are also suggestive of the fact that the proposed *Monitored Distributed P System* is suitable for real time optimisation in a dynamic and distributed environment.

**Key Words:** Membrane Computing, Distributed P Systems, Particle Swarm Optimisation, Cross Layer Optimisation
**Category:** H.1

## 1 Introduction

The aim of this research work is to design a membrane computing model that would represent distributed systems effectively for real time computations. In the existing *distributed P (dP)* system [Paun and Perez-Jimenez, 2010] a centralised control or monitoring process has not been explicitly defined or integrated. In real time systems that are dynamic in nature a centralised monitoring system to co-ordinate the activities is required for the effective utilisation of resources. Responding efficiently to the need for a change within the system is also an important functional requirement in dynamic environments. We propose a variant of the existing *dP system* titled *Monitored Distributed P System (MDP System)*. The proposed *MDP System* has a centralised *P system* for monitoring and communicating with the other *P systems* whenever required. Each *P system* in the proposed distributed model has been designed to adopt the divide and conquer paradigm for computing. This results in a very quick response by each *P system* thereby collectively leading to a distributed system with very efficient response time to requests. In addition, local environments are also taken into consideration in our proposed model to enable multi-casting and independent control of each component of the distributed system.

Utilisation of *Cross Layer Optimization (CLO)* in networked communication systems is a challenging and emerging research area. Enabling the efficient utilization of resources is the primary goal of *CLO* and it is effectively being used currently in wired communication systems. Wireless communication systems that are dynamic in nature require optimizations to be carried out in real time [Song et al., 2008; Van der Schaar and Sai Shankhar, 2005] . Hence there is a need for improved *CLO* techniques having better response time. Wireless multimedia applications are becoming increasingly popular for business, education and entertainment. These delay sensitive, bandwidth intensive and loss-tolerant multimedia applications have an ever demanding need for better Quality of Service (QoS). The existing Cross Layer Designs for multimedia transport having significant improvements with respect to various *QoS* parameters have been presented in [Hager et al., 2008; Syed and Hayder, 2009; Toni, 2009]. In the layered network operating systems traditionally only adjacent systems have been communicating and fine tuning themselves to work in an optimized way. But the Cross Layer Designs explore the possibilities of communicating across all the layers in order to work in an optimized manner. *CLO* can be integrated into the existing wired and wireless systems without drastically changing their original design. Centralized and decentralized schemes [Nicholas and Van der Schaar, 2010] have also been proposed to cater to network layers from the same and different manufacturers respectively.

*Particle Swarm Optimization (PSO)* [Riccardo et al., 2007] is an evolutionary optimization technique having relatively few parameters to deal with and is hence easy to incorporate and implement in various applications. Several variants of the basic *PSO* have evolved over the years [Sedighizadeh and Masehian, 2009] and have proved that *PSO* is a very promising optimization technique. In this paper we use *PSO* for *CLO* and present a generalised framework for using them in an integrated manner. *PSO* has been used effectively in real-time optimization [Liu et al., 2011; Meigin et al., 2010; Zhang and Mahfouf, 2006] and also in Cross Layer Design schemes in very narrow domain specific applications [Tang et al., 2009]. The variant of *PSO* that integrates the concept of digital pheromones [Kalivarapu et al., 2009] has been utilised in our paper to overcome limitations of the basic *PSO* and to obtain faster convergence. Thus an improved *PSO with Digital Pheromone* has been used to compute solutions for Cross Layer Optimization [Elias et al., 2011]. The *P variant* for the improved *PSO* can be implemented as a parallel algorithm as its membrane structure and evolution rules have been designed for distributed computing to obtain a speed up in computation. Thus in this paper we present,

1. The design of the proposed *Monitored Distributed P System* which is a variant of the *Distributed P system* integrated with features to monitor and communicate within a distributed system.

2. *Real time Cross Layer Optimisation* using *PSO with digital pheromones* presented as an application of the proposed *Monitored Distributed P System.*

## 2 Terminology

### 2.1 *P* System

Membrane Computing [Paun, 2000] is a research area that aims to abstract computing ideas and models from the structure and functioning of living cells, as well as from the way the cells are organized in tissues or higher order structures. Membrane systems also known as *P systems*, deal with distributed and parallel computing models, processing multi-sets of symbol objects in a localized manner. Evolution rules and evolving objects are encapsulated into compartments delimited by membranes with an essential role played by the communication between compartments and with the environment.

#### 2.1.1 The Basic Model

The fundamental component of a *P system* is the membrane structure. The membrane structure is a hierarchical arrangement of membranes, as in a cell or a network of membranes, within an external membrane referred to as the *skin* membrane. A membrane without any other membrane inside it is called an *elementary* membrane. The membranes are identified by *labels* as shown in Figure 1 presented in [Paun et al., 2010; Paun, 2000].

Objects evolve by means of *evolution rules* which are localized and associated with the regions of the membrane structure. There are three main types of evolution rules (i) Multiset rewriting rule (ii) Communication rules (iii) Rules for handling membranes. These evolution rules are applied non-deterministically

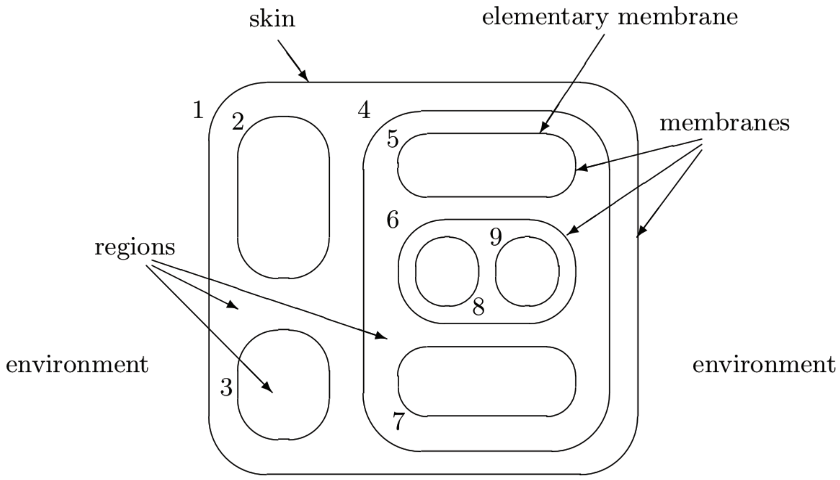


**Figure 1:** Basic P System

in a maximally parallel manner. Some of the existing variants that were of interest to this research work are the *P Automata* [Csuhag-Varju 2010], *dP Automata* [Paun and Perez-Jimenez, 2010], *Membrane Algorithm* [Nishida, 2006], *P Colonies* [Cienciala et al., 2010], *Population P system* [Bernardini and Gheorghe, 2004], *Using Genetic Algorithm to evolve P systems* [Escuela et al., 2010], *Membrane Algorithm for optimisation* [Zhang and Huang, 2009], *P variant for ant colony* [Zhang et al., 2010], *P variant for Quantum inspired evolutionary algorithm* [Zhang et al., 2008] and the *P variant for PSO* [Zhou et al., 2010]. The variant referred to as *distributed P systems* was found to be most suitable for real-time applications and it contributed to the basic design of the membrane computing model proposed in this paper.

#### 2.1.2 Distributed P System

A Distributed P system (dP system) [Paun and Perez-Jimenez, 2010] of degree $n \geq 1$ is a construct

$$\Delta = (O, \pi_1, \pi_2, ..., \pi_n, R) \tag{1}$$

where:

1. $O$ is an alphabet of objects;
2. $\pi_1, \pi_2, ..., \pi_n$ are cell-like P systems with O as the alphabet of objects and the skin membranes labeled as $s_1, ..., s_n$, respectively;

3. $R$ is a finite set of rules of the form $(s_i, u/v, s_j)$, where $1 \leq i,\ j \leq n,\ i \neq j$ and $u, v \in O^*$, with $uv \neq \lambda$; $|uv|$ is called the weight of the rule $(s_i, u/v, s_j)$

$\pi_1, \pi_2, ..., \pi_n$ are called the components of the system and the rules in $R$ are called *inter-component communication rules*. The application of the rule $(s_i, u/v, s_j)$ means that the systems $\pi_i$ and $\pi_j$ communicate and the set of objects in $u$ are sent from $\pi_i$ to $\pi_j$ while the set of objects in $v$ are sent from $\pi_j$ to $\pi_i$. Each component can take an input, work on it, communicate with other components, and provide the answer to the problem after the halting computation. In [Paun and Perez-Jimenez, 2010] it has been suggested that *local environments* can be considered for each component in future variants and its significance could be analysed and explored. The variant proposed in this paper has incorporated this concept of *local environments* in the distributed model and also demonstrates a very innovative application of this new feature. Details of the proposed model will be presented in section 3.

## 2.2 Particle Swarm Optimization: *PSO*

### 2.2.1 The Basic *PSO*

*Particle Swarm optimization* is an emerging technique that is simple and easy to implement and helps to achieve relatively faster convergence. This technique is briefly explained in this section, followed by the variant with digital pheromones that has been adopted as the optimization technique in our proposed *MDP-CLO* model. In the basic *PSO*, $P$ number of particles are randomly distributed in a problem solution space $S$ with $N$ number of dimensions represented as $S^N$. Each particle will compute the solutions and determine their suitability by using the fitness function $f(s^1, s^2, ..., s^n)$, where $0 < n \leq N$ and $s^n \in S^N$. The objective of the optimization is to find a set of $\hat{S} \subset S$ to maximize/minimise the fitness function $\hat{S} = argmax\ f(s^1, s^2, ..., s^n)$.

The *PSO* technique initially generates random positions and velocities for a population of particles. Each particle represents an alternative solution in the multidimensional search space. Each particle computes its way through the search space with the velocity constantly updated according to its own search experience and its neighbours's best experience. The position vector and velocity vector of the $i^{th}$ particle in the D-dimensional search space can be represented as $X_i = (x_{i_1}, x_{i_2}, ..., x_{i_D})$ and $V_i = (v_{i_1}, v_{i_2}, ..., v_{i_D})$ respectively. According to a predefined fitness function mentioned as before, the best previous position of the $i^{th}$ particle among all the particles found so far is $Pg = (P_{g_1}, P_{g_2}, ..., P_{g_D})$. The velocity and position of the particles are updated according to the following equations,

$$V_{id}(t+1) = w*v_{id}(t)+c_1*r_1(t+1)*[P_{id}(t)-x_{id}(t)]+c_2*r_2(t+1)[P_{gd}(t)-x_{id}(t)]$$

$$x_{id}(t+1) = x_i dt + V_{id}(t+1) \tag{2}$$

where $t$ is the index of the iterations, $w$ is the inertia weight, $C_1$ and $C_2$ are positive constants known as acceleration coefficients, $r_1(t)$ and $r_2(t)$ are two uniformly distributed random variables in the range (0,1). The second part of velocity update equation is known as *cognitive* component. It represents the personal thinking of each particle. This component encourages the particles to fly towards their own best positions found so far. The third part of velocity update shown in the above equation is the *social* component, which represents the co-operative effect of the particles in optimization searching. This component always leads the particles towards the global best particle found so far. Generally a maximum velocity vector $V_{max}$ is defined and acts as an upper limit for the achievable velocity of the particles. It is used to control the ability of the particles to search and is often confined within the search space.

### 2.2.2 Variant of PSO with Digital Pheromones

A variant of *PSO* [Kalivarapu et al., 2009] uses digital pheromones for aiding communication within the swarm to improve the search efficiency and reliability. This variant has been used effectively in other applications [Chandar et al., 2010]. An additional pheromone component potentially causes a swarm member to move in a direction different from the combined influence of the particle's best and global best positions, thereby increasing the probability of finding the global optimum. The velocity update is done using the formulation given below,

$$V_{id}(t+1) = w*v_{id}(t)+c_1*r_1(t+1)*[P_{id}(t)-x_{id}(t)]$$
$$+c_2*r_2(t+1)*[P_{gd}(t)-x_{id}(t)]$$
$$+c_3*r_3(t+1)*[Targetpheromone_{id}(t)-x_{id}(t)]$$

The parameter $c_3$ is a user defined *confidence parameter* for the pheromone component of the velocity vector. The *confidence parameter* $c_3$ combines the knowledge from the *cognitive* and *social* components of the velocity of a particle and complements their deficiencies. The *confidence parameter* $c_3$ determines the extent of influence a target pheromone can have on the swarm when the information from particle's best and global best alone are not sufficient to determine a particle's next move. The particle is attracted to a target pheromone that has the highest $P'_h$ value based on its proximity to other pheromones and their pheromone level $P_h$. $P'_h$ is given by $P'_h = (1-d)P_h$ where $d$ is the distance between the particle and the target pheromone.

### 2.3 Cross Layer Optimization: CLO

Cross Layer optimization solutions are traditionally classified [Van der Schaar and Tekalp, 2005] into several categories and briefly presented below,

1 *Top Down Approach:* The higher layer protocols optimise their parameters and strategies at the next lower layer.

2 *Bottom Up Approach:* The lower layers try to insulate the higher layers from losses and bandwidth variations.

3 *Application Centric Approach:* The $APP$ layer optimises the lower layer parameters one at a time in a *Bottom Up* or *Top Down* manner based on its requirements.

4 *MAC Centric Approach:* The $APP$ layer passes its traffic information and requirements to the $MAC$ which decides which $APP$ layer packets should be transmitted and at what $QoS$ level.

5 *Integrated Approach:* The integrated approach exhaustively computes the fitness values of all possible joint strategies and determines the best strategy.

The Cross Layer Design problem is an optimization problem with the objective to select a joint strategy across multiple $OSI$ layers. Here we limit our discussion to the Physical ($PHY$), Medium Access Control ($MAC$) and Application ($APP$) layers respectively. Let $N_P$, $N_M$, $N_A$ denote the number of adaptation and protection strategies available at $PHY$, $MAC$ and $APP$ layers respectively. For instance, the strategies $PHY_i, i \in \{1, 2, ..., N_P\}$, may represent the various modulation and channel coding schemes. The strategies $MAC_i$, $i \in \{1, 2, ..., N_M\}$, corresponds to different packetization, Automatic Repeat Request ($ARQ$), scheduling, admission control and Forward Error Correction ($FEC$) mechanisms. The strategies $APP_i, i \in \{1, 2, ..., N_A\}$ may include adaptation of video compression parameters, packetization, traffic shaping, traffic prioritisation, scheduling, $ARQ$ and $FEC$ mechanisms. We define the joint cross layer strategy $S$ as $S = \{PHY_1, ..., PHY_{N_P}, MAC_1, ..., MAC_{N_M}, APP_1, ..., APP_{N_A}\}$. Accordingly, there are $N$ possible joint design strategies where $N = N_P * N_M * N_A$. The CLO problem seeks to find the optimal composite strategy $S^{opt}(x) = arg_S \ max \ Q(S(x))$. This strategy results in the best network utilisation subject to many environmental constraints like bandwidth utilization ($BU$), packet loss ratio ($PLR$), etc.

## 3 Design of the Proposed Monitored Distributed P System

A *Monitored Distributed P System* is a construct

$$\Delta^{'} = (O, \pi_c, \pi_1, \pi_2, ..., \pi_n, S) \tag{3}$$

where:

1. $O$ is an alphabet of objects;
2. $\pi_c$ is the centralized $P$ system with the following construct,

$$\pi_c = (V_c, \mu_c, w_c, E_c, R_c) \tag{4}$$

   where:

   (a) $V_c$ is an alphabet of objects, $V_c \subseteq O$;

   (b) $\mu_c$ is a membrane structure consisting of skin membrane $[_0]_0$. The skin membrane (i.e., the $0^{th}$ membrane) in the rule is referred as $s_c$;

   (c) $w_c$ are strings representing the multisets over $V$ associated with skin region;

   (d) $E_c \subseteq V_c$ represents the objects available in arbitrarily many copies in the environment;

   (e) $R_c$ represents evolution and symport communication rules [Cavaliere and Genova, 2004] of the form:

      i. $a \rightarrow b$
      Here $a$ evolves as $b$ in the skin membrane of $\pi_c$ (Note that $\pi_c$ has only skin membrane);

      ii. $a \rightarrow (b, entry)$
      The symbol $a$ enters the local environment of $\pi_c$ as $b$;

      iii. $a \rightarrow (b, exit)$
      The symbol $a$ in $LE_c$ changes to $b$ and exits where $LE_c$ is the local environment of $\pi_c$

      iv. $a \rightarrow (b, in)$
      The symbol $a$ in $LE_c$ changes to $b$ and enters the skin membrane of $\pi_c$;

      v. $a \rightarrow (b, out)$
      The symbol $a$ in the skin membrane changes to $b$ and goes from the skin membrane of $\pi_c$ to the local environment $LE_c$;

      vi. $(s_c, u, LE_j)$
      The symbol $u$ is sent from the skin membrane of $\pi_c$ to the local environment of $\pi_j$.

3. $\pi_i$ represent the P system for each node.

$$\pi_i = (V_i, \mu_i, w_{i,1}, ..., w_{i,m_i}, E_i, R_{i,1}, ..., R_{i,m_i}) \tag{5}$$

where:

(a) $V_i$ is an alphabet of objects, $V_i \subseteq O$;

(b) $\mu_i$ is a membrane structure of the $i^{th}$ $P$ system containing $m_i$ membranes, $[_0[_1]_1[_2]_2, ..., [_{m_i}]_{m_i}]_0$ apart from the skin membrane with label 0. The skin membrane (i.e., the $0^{th}$ membrane) in the rule is referred as $s_i$;

(c) $w_{i,1}, ..., w_{i,m_i}$ represents the multisets of objects available in each membrane;

(d) $E_i \subseteq V_i$ represents the objects available in arbitrarily many copies in the environment;

(e) $R_{i,1}, ..., R_{i,m_i}$ represent the evolution and symport communication rules [Cavaliere and Genova, 2004] used in $P$ system $\pi_i$. The rules are of the form:

i. $a \rightarrow (b, here)$
Here the symbol $a$ changes to $b$ and remains in the same membrane;

ii. $a \rightarrow (b, out)$
The symbol $a$ changes to $b$ and goes out either to the outer membrane or to $LE_i$;

iii. $a \rightarrow (b, in)$
The symbol $a$ changes to $b$ and goes from $LE_i$ to the skin membrane of $\pi_i$;

iv. $a \rightarrow (b, in_j)$
The symbol $a$ changes to $b$ and goes from the skin membrane of $\pi_i$ to a membrane j contained in it;

v. $(s_i, u/v, s_c)$
The rule represents the skin to skin rule between $\pi_i$ and $\pi_c$ with $uv \neq \lambda$; $|uv|$ is called the weight of the rule $(s_i, u/v, s_c)$.

4. $S$ is a finite set of inter-component communication rules of the form $(s_i, u/v, s_j)$, where $1 \leq i,\ j \leq n,\ i \neq j$ and $u, v \in O^*$, with $uv \neq \lambda$; $|uv|$ is called the weight of the rule $(s_i, u/v, s_j)$.

If $\alpha_i$ denotes the configuration of the MDP system at an instant $i$ and the configuration changes to $\alpha_{i+1}$ at instant *i+1* by the application of evolution and/or communication rules, we write $\alpha_i \Rightarrow \alpha_{i+1}$.

### 3.1 Application of MDP System in CLO

Let us refer to this model as the *MDP-CLO* model that can be used to study the behaviour of an adhoc wireless network where *PSO* and *MDP* are used to select the best strategy in *CLO*. In this application we consider an adhoc wireless network and assume one of the nodes to be stationed within the network to perform network monitoring. This node is represented by $\pi_c$ in the proposed *MDP System*. All other distributed nodes are allotted a unique identifier and represented as $\pi_i$ in general. The evolution rules that indicate the methodology adopted in the system to perform the desired function, has also been formally represented in this paper. More precisely, a component $P$ system in the Monitored Distributed P System framework will consist of:

i. A membrane structure $\mu = [_0[_1]_1, [_2]_2, ..., [_m]_m]_0$ with *m+1* regions delimited by $m$ elementary membranes and the skin membrane denoted by 0;

ii. A vocabulary that consists of all the particles (i.e., the solutions);

iii. A set of terminal symbols, $T$;

iv. Initial multisets $w_0 = \lambda$,
$w_1 = P_1P_2, ..., P_{M_1}$
$w_2 = P_{M_1+1}P_{M_1+2}, ..., P_{M_2}$
.
.
.
$w_m = P_{M_{(m-1)}+1}P_{M_{(m-1)}+2}, ..., P_{M_m}$
where $P_i, 1 \leq i \leq M$, is a solution(particle); $M_j, 1 \leq j \leq m$, is the number of solutions in the $w_j$; $\sum_{j=1}^{m} M_j = M$, where $M$ is the total number of particles (solutions) in this computation.

v. Rules which are categorized as, developmental/transformation rules perform the communication between the particles in the elementary membranes and those in the skin membrane.

In the *MDP-CLO* the initial swarm of particles is scattered across the membrane structure. The initial swarm will consist of the multisets $w_1, ..., w_m$. The particle swarm optimization using digital pheromones worked out in each elementary membrane determines the best cross layer strategy for that corresponding elementary membrane. Thus $m$ best particles (solutions) from $m$ elementary membranes form the initial multisets in the skin membrane. Now by performing a linear search among the particles, the best particle (best solution) is chosen based on the fitness value. The process will stop according to a pre-set termination condition, such as a certain number of iterations. The algorithms presented here are, 1 *MDP-CLO()*: Algorithm that utilises *MDP* model in performing real time *CLO*

2 *C-PSO()*: Algorithm to perform *CLO* using a variant of *PSO*

3 *PSO-H()*: Algorithm that uses the variant of *PSO* with digital pheromones

---

**Algorithm 1**: *MDP-CLO()*

---

Create a Centralised P system $\pi_c$ with its local environment $LE_c$ which obtains the network profile.
**for** *each node* $Nod_i$ *entering into the system* **do**
    Create an individual P system $\pi_i$ for the corresponding node, with its local environment $LE_i$
    Broadcast its arrival to all other nodes already present through the centralised P system $\pi_c$
**end**
**for** *each node* $Nod_i$ *in the system* **do**
    let $\pi_c$ perform the communication to all nodes informing about the environmental inputs and changes in the environment
**end**
**for** *each node* $Nod_i$ **do**
    call *C-PSO()*
**end**
From each node $Nod_i$ the centralised P system obtains the computed best joint strategy and taking the current network profile into consideration, computes the need for modification of strategies. This is carried out through suitable algorithm or methods.

---

The step wise description of the *C-PSO()* algorithm is presented here for clarity.

- **Step 1.** In this step, the *OLMS* (One Level Membrane Structure as considered in [Zhang et al., 2008] ), is constructed and the parameter $m$(number of elementary membranes) is determined as follows,

  $m = (N_P * N_M * N_A)$/Number of particles=$N/M$ where M represents number of particles inside an elementary membrane (chosen randomly)

- **Step 2.** The loop is repeated until the termination condition

- **Step 3.** The $M$ particles forming a swarm are scattered across the $m$ elementary membranes and contains at least two particles. Thus, the

**Algorithm 2**: *C-PSO()*

Initialize the membrane structure as OLMS<br>
**while** *not termination condition* **do**<br>
    Scatter particles into the elementary membranes<br>
    Determine iterations for each of the elementary membranes<br>
    **for** *i=1 to m* **do**<br>
        Perform *PSO-H()* inside the $i^{th}$ elementary membrane<br>
    **end**<br>
    Form a swarm of particles in the skin membrane<br>
    Perform linear search in the skin membrane to determine the best joint strategy<br>
**end**

**Algorithm 3**: *PSO-H()*

Populate swarm with random initial values<br>
**while** *!converged* **do**<br>
    Evaluate the fitness value of each swarm member<br>
    Store pbest and gbest<br>
    Decay digital pheromones in the solution space (if any)<br>
    **if** *Iteration==1* **then**<br>
        Randomly choose 50 percentage of swarm to release pheromones<br>
    **end**<br>
    **else**<br>
        Improved particles releases pheromones<br>
    **end**<br>
    Find target pheromone towards which each particle moves<br>
    Update velocity vector and position of each particle<br>
**end**<br>
Stop the algorithm

number of particles in an elementary membrane varies from *2* to *M-2m+2*

- **Step 4.** This step determines the number of iterations for each elementary membrane required to independently perform the *PSO* with digital pheromones. The number $A_i(i = 1, 2, ..., m)$ of iterations for the $i^{th}$ elementary membrane is generated randomly between $g_{min}$ and $g_{max}$, i.e., $A_i = A_{min} + \lfloor rand(0, 1).(A_{max} - A_{min})\rfloor$ where $A_{min}$ and $A_{max}$ are lower and upper limits of iterations for elementary membranes; $\lfloor . \rfloor$ is a function rounding an element to the nearest smaller integer.

- **Step 5,6,7.** In each of the $m$ elementary membranes, the *PSO-H()* algorithm shown in Algorithm 3 is performed independently, (i.e., for i=1,2, ..., k) iterations.

- **Step 8.** The swarm of particles in the skin membrane is formed by using the best particles of elementary membranes. Each compartment sends the best particle out into the skin membrane and therefore there are $m$ particles in total.

- **Step 9.** A linear search is performed for all the best particles in skin membrane, and based on the best fitness value the best particle is chosen as the optimal solution for the joint cross-layer strategy.

### 3.2 Evolution rules for Communication in the proposed MDP System

The evolution rules presented in this section model the communication between the components of the distributed system and with the environment and is illustrated in Figure 2. The parameter *Nod* used in the evolution rules represents the total number of components in the system (except $\pi_c$)

1. Communication rules associated with $\pi_c$ during entry and exit of nodes in the distributed system:
   a. To register the Nodes in $\pi_c$'s local environment $LE_c$
   $Node_i \rightarrow (Nodereg_{i_1}, entry)$ where $1 \leq i \leq Nod$;
   b. To communicate the registration information from $LE_c$ to the skin membrane,
   $Nodereg_{i_1} \rightarrow (Nodereg_{i_1}, in)$ where $1 \leq i \leq Nod$;
2. To record the exit of a node i.e cancel the registration (Note: $Nodereg_{i_1}$ is the symbol that represents the presence of a node, $Nodereg_{i_0}$ is the symbol that represents the absence of a node i.e., the node that wishes to exit the system)
   a. $Nodereg_{i_1} \rightarrow (Nodereg_{i_0}, here)$ where $1 \leq i \leq Nod$ (according to rule 3.(e).i in section 3)
   b. $(s_i, Nodereg_{i_0}/\lambda, s_c)$ where $1 \leq i \leq Nod$ (according to rule 3.(e).v in section 3)
   c. $Nodereg_{i_0} \rightarrow (Nodereg_{i_0}, out)$ where $1 \leq i \leq Nod$ (according to rule 2.(e).v in section 3)

   d. $Nodereg_{i_0} \rightarrow (Node_i, exit)$ where $1 \leq i \leq Nod$ (according to rule 2.(e).iii in section 3)

3. To broadcast from $\pi_c$, the information regarding the arrival and departure of every node to every other node
   a. $(s_c, Nodereg_{i_1}/\lambda, s_j)$ where $1 \leq i, j \leq Nod$;
   b. $(s_c, Nodereg_{i_0}/\lambda, s_j)$ where $1 \leq i, j \leq Nod$;
4. To perform *skin-to-skin* communication, when any node needs to communicate with any other node other than $\pi_c$
   $(s_i, data1/data2, s_j)$ where $1 \leq i, j \leq Nod, i \neq j$;
5. To communicate the computation results of the $j^{th}$ membrane of the $i^{th}$ node to its $0^{th}$ membrane.
   $Ans_{ij} \rightarrow (Ans_{ij}, out)$ where $1 \leq i \leq Nod$;
   $Ans_{ij}$ denotes the best joint strategy in membrane $j$ of $\pi_i$;
6. To communicate the best joint strategy to the skin membrane of $\pi_c$
   $(s_i, Ans_i/\lambda, s_c)$ where $1 \leq i \leq Nod$;
7. To communicate the network profile to $\pi_c$'s local environment $LE_c$ as well as to its skin membrane
   a. $NP \rightarrow (NP, entry)$;
   b. $NP \rightarrow (NP, in)$;
8. To communicate the need to modify the current strategy
   $(s_c, changestrategy_i, LE_i)$ where$1 \leq i \leq Nod$;
   The change in the joint strategy used by the node $i$ is communicated from the skin membrane of the centralised node to the local environment of $i$. For example a code such as $ABCDE$ could be formulated such that each symbol can be used like a flag indicating and suggesting a change in a particular aspect. Node $i$ may have limitations and may not be able to carry out all suggested changes. The local environment sends only those changes which are possible, to the skin membrane of $i$. This could possibly trigger the re-computation of the optimised values using the $PSO$ to obtain a better strategy taking into consideration the current scenario.

## 4 Results and Analysis

In this section we discuss the computational efficiency of the proposed variant the *Monitored Distributed P System* (*MDP* System). We analyse the improvement

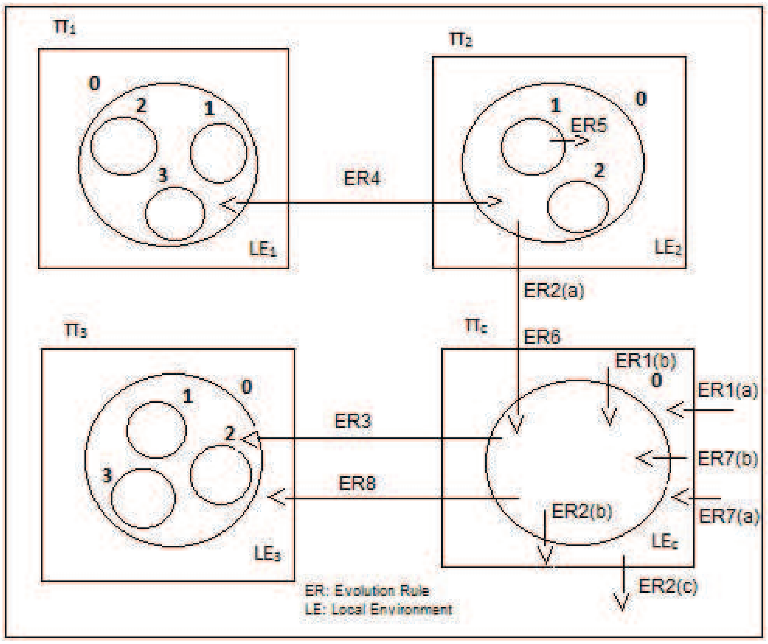


**Figure 2:** The proposed Monitored Distributed P System

in communication complexity and also the time space trade-off of the proposed model in comparison with the existing distributed P system. The experimental results presented in this section indicates the significant improvement in response time and efficient use of space by the proposed computing model.

### 4.1 Complexity Analysis of the proposed work

The computing efficiency is determined by evaluating the utilisation of resources especially in time and space. P systems that can generate an exponential workspace in linear time are considered to be efficient problem solvers. The efficiency in speed-up is obtained by trading space for time. There are three basic ways to create an exponential workspace: Membrane Division, Membrane Creation and String Replication

Our proposed model generates workspace by membrane creation and efficiently performs time-space trade off dynamically. This is achieved by incorporating the Expected Response Time ($ERT$) parameter while computing the number of elementary membranes required to be created to solve the problem. The number of joint strategies $N$, particles $M$ and membranes $m$ are computed as follows:

$$N = N_P * N_M * N_A \tag{6}$$

The terms $N_P$, $N_M$, $N_A$ are referred in Section 2.3. In Equation (7) the parameter $M_{init}$ represents the initial number of particles. The value for the number of particles used for obtaining experimental results normally ranges between 1 and 100 depending on the problem size. $M_{init}$ is assigned a value within this range initially. The number of particles are however determined by Equation (7) where the expected response time ($ERT$) to determine the new joint strategy plays a role in varying the required number of particles by either increasing or decreasing them. The number of particles ($M$) in-turn varies the number of membranes ($m$) thereby trading time and space effectively.

$$M = M_{init} * ERT \tag{7}$$

$$m = N/M \tag{8}$$

Communication complexity of the proposed *Monitored Distributed P System* (MDP System) can be theoretically analysed. In [Adorna et al., 2010] the parameters used to compute and evaluate the communicating complexity in *P systems* is presented. The communication complexity of the existing dP system presented in [Paun and Perez-Jimenez, 2010] helps to compare and evaluate the performance of our proposed *MDP System*. There are several possibilities for estimating *the cost of communication*. Let us consider an *MDP System* $\Delta^{'}$, and let $\delta : \alpha_0 \Rightarrow \alpha_1 \Rightarrow, ..., \Rightarrow \alpha_h$ be a computation in $\Delta^{'}$, with $\alpha_0$ being the initial configuration. In the existing *dP system* a halting computation is defined as the one where the computation after starting at the initial configuration stops after reaching a stable final configuration. In a wireless, adhoc network there is nothing such as halting configuration. Nodes enter and exit dynamically. Hence for calculating the complexity measures, a sequence $\delta : \alpha_0 \Rightarrow \alpha_1 \Rightarrow, ..., \Rightarrow \alpha_h$ is taken where $\alpha_h$ is the configuration at time *t*.

Then for each $i = 0, 1, ..., h - 1$ we can write,

1 ComN($\alpha_i \Rightarrow \alpha_{i+1}$) is equal to one, if a communication rule is used in this transition, and 0 otherwise

2 ComR($\alpha_i \Rightarrow \alpha_{i+1}$) is equal to the number of communication rules used in this transition

3 ComW($\alpha_i \Rightarrow \alpha_{i+1}$) is equal to the total weight of the communication rules used in this transition

The number of communication rules *ComR* will indicate the *cost of communication* of our proposed model. In the *MDP System* the centralised node directs all messages relating to control and monitoring of the system directly to each node and each node responds to the central node in turn. Suppose we represent the same using dP system without a centralized system, then these communication complexity will be comparable and almost the same. Here, the evolution rules that define this communication are associated with the designated central node only. Let us assume that there are $m$ such communication rules required in a *MDP System*. The equivalent *dP System* will therefore require *O(nm)* number of communication rules where $n$ is the number of nodes or distributed components. This is because in the *dP System* the communication rules associated with the proposed *MDP System* has to be replicated in all the $n$ nodes of the *dP System* in-order to make it computationally equivalent. This indicates that the number of communication rules in the proposed *MDP System* is less when compared to the existing *dP System*. The main purpose of having a centralized system is to decide on the change to be executed, in the network profile to the respective P system $\pi_i$.

### 4.2 Experimental Results

The following experimental results were obtained for a single node in the proposed *MDP model*. The simulations were performed on a cluster system using Message Passing Interface (*MPI*) programming and C language using a Pentium 4 multicore processor set up. The results depicted in the graphs are based on the bench mark fitness functions listed in the Table 1 where $D$ represents the number of dimensions, (any suitable fitness function can also be considered).

**Table 1:** Fitness Functions

| **Fitness** | **Function** | **Formula** | **Range** |
|---|---|---|---|
| Fit 1 | Rastrigin | f(x) $= \sum_{i=1}^{D}(x_i^2 - 10cos(2\pi x_i) + 10)$ | [-5.12, 5.12] |
| Fit 2 | Griewank | f(x) $= \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} cos(\frac{x_i}{\sqrt{i}}) + 1$ | [-600, 600] |
| Fit 3 | Rosenbrock | f(x) $= \sum_{i=1}^{D}(100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2)$ | [-5. 12, 5. 12] |
| Fit 4 | Quadric | f(x) $= \sum_{i=1}^{D} x_i^2$ | [-5.12, 5.12] |

The graph in Figure 3 shows the performance improvement. As number of elementary membranes increase the response time decreases.

The graph in Figure 4 indicates that as number of particles increase the response time also increases. This is because the number of particles $M$ and the number of membranes $m$ are inversely proportional in the equation $m$=$N/M$, .

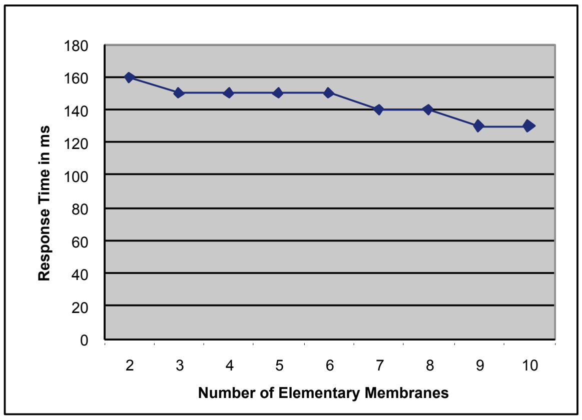


**Figure 3:** Number of Elementary membranes vs Response time

Here as the number of membranes $m$ decreases (since number of particles $M$ is increasing), the parallelism attained is less and hence the response time increases.

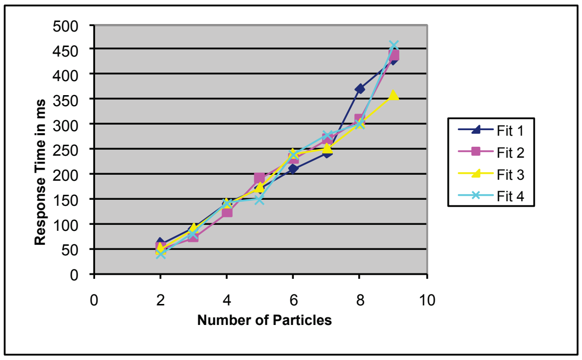


**Figure 4:** Number of particles vs Response time

The graph in Figure 5 helps to determine the optimal number of particles that could be associated within an elementary membrane to obtain an optimal solution (i.e., an optimal joint cross-layer design strategy) with minimal fitness.

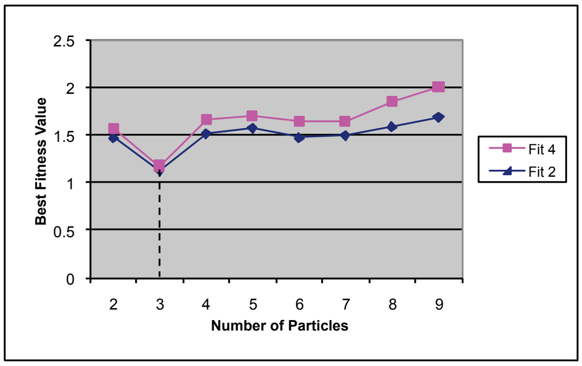


**Figure 5:** To determine the optimal number of particles

## 5 Conclusion and Future Work

This paper proposes a variant of the *distributed P system* that is suitable for real time applications. The proposed model has been incorporated with *local environments* and a centralised monitoring *P system* and hence referred to as *Monitored dP system (MDP system)*. The proposed *MDP model* was found to have efficient time, space and communication complexity. The Cross layer optimisation *CLO* technique that uses a variant of Particle Swarm Optimisation with digital pheromones for faster convergence has been used in the *MDP model* to illustrate its capabilities in handling real time optimisation in dynamic environments. Further enhancements of this research work could also include analysis of mathematical properties of the proposed MDP model.

### Acknowledgement

This research work was supported by the National Natural Science Foundation of China (61170016), the Program for New Century Excellent Talents in University (NCET-11-0715) and SWJTU supported project (SWJTU12CX008).

## References

[Adorna et al., 2010] Adorna, H., Paun, G., and Perez-Jimenez, M. J. : On Communication Complexity in Evolution-Communication P Systems. Romanian Journal of Information Science and Technology, 13: 113-130, Sept. 2010.

[Bernardini and Gheorghe, 2004] Bernardini, H., and Gheorghe, M.: Population P Systems. Journal of Universal Computer Science, 10(5): 509-539, 2004.

[Cavaliere and Genova, 2004] Cavaliere, M., and Genova, D.: P Systems with symport/antiport rules. Journal of Universal Computer Science, 10(5): 540-558, 2004.

[Cienciala et al., 2010] Cienciala, L., Ciencialova, L., Csuhag-Varju, E., and Vaszil, D. : PCol Automata: Recognising Strings with P Colonies. Proceedings of the 8th Brain Storming Week on Membrane Computing, Servilla, 2010.

[Csuhag-Varju, 2010] Csuhag-Varju, E. : P Automata: Concepts, Results, and New Aspects. Membrane Computing, Lecture Notes in Computer Science, Springer, Berlin/Heidelberg, 5957: 1-15, 2010.

[Elias et al., 2011] Elias, S., Gokul, V., Krithivasan, K., Gheorghe, M. and Zhang, G. : Real Time Cross Layer Design using Particle Swarm Optimization. Proceedings of the International Conference ACM Compute, Bangalore, 2011.

[Escuela et al., 2010] Escuela, G., Naranjo, G., and Miguel, A. : An Application of Genetic Algorithms to membrane Computing. Proceedings of the 8th Brain Storming Week on Membrane Computing, Servilla, 2010.

[Hager et al., 2008] Hager, C. T. R., Shyy, D. J., and Ma, J. : Cooperative Cross-Layer Design for Wireless Networks. Journal of Communications, 3(4): 49-58, 2008.

[Kalivarapu et al., 2009] Kalivarapu, V., Foo, J., and Winer, E. : Improving solution characteristics of particle swarm optimization using digital pheromones. Structural and Multidisciplinary Optimization, Springer Berlin / Heidelberg, 37(4): 415-427, 2009.

[Liu et al., 2011] Liu, W., Chung, I., Leng, S., Liu, L. and Cartes, D. A. : Real-time particle swarm optimization based current harmonic cancellation. Engineering Applications of Artificial Intelligence, Springer New York, 24(1): 132-141, 2011.

[Nicholas and Van der Schaar, 2010] Nicholas, M., and Van der Schaar, M. : Online layered learning for cross-layer optimization of dynamic multimedia systems. Proceedings of the first annual ACM SIGMM conference on Multimedia systems, Arizona, USA, 2010.

[Paun et al., 2010] Paun, G., Grzegorz, R., and Arto, S. : The Oxford Handbook of Membrane Computing. Oxford University Press, New York, USA, 2010.

[Paun, 2000] Paun, G. : Computing with membranes. J. Comput. Syst. Sci., 61(1): 108-143, 2000.

[Paun and Perez-Jimenez, 2010] Paun, G., and Perez-Jimenez, M. J. : Solving Problems in a distributed way in membrane computing : dP Systems. International Journal of Computers Communication and Control, 5(2): 238-250, 2010.

[Riccardo et al., 2007] Riccardo, P., James, K., and Tim, B. : Particle swarm optimization. Swarm Intelligence, 1(1): 33-57, 2007.

[Chandar et al., 2010] Sarath Chandar, A. P., Dheeban, S. G., Deepak, V., and Elias, S. : Personalised e-course composition approach using digital pheromones in improved particle swarm optimisation. Proceedings of the 6th International Conference on Natural Computing, ICNC, China, 2010.

[Sedighizadeh and Masehian, 2009] Sedighizadeh, D., and Masehian, E. : Particle Swarm Optimisation Methods Taxonomy and Applications. International Journal of Computer Theory and Engineering, 1(5): 1793-8201, 2009.

[Song et al., 2008] Song, C., Wang, H., Haohong, and Dalei, W. : A theoretical framework for quality-aware cross-layer optimized wireless multimedia communications. Adv. MultiMedia, 2(1): 2:1-2:10, 2008.

[Syed and Hayder, 2009] Syed, K., and Hayder, R. : Comparison of Conventional and Cross-Layer Multimedia Transport Schemes for Wireless Networks. Wireless Personal Communications, Springer, Netherlands, 51(3): 535-548, 2009.

[Meigin et al., 2010] Meigin, T., Chengnian, L., and Xinping, G. : Nonconvex maximization for communication systems based on particle swarm optimization. Comput. Commun., 33(7): 841-847, 2010.

[Tang et al., 2009] Tang, S., Qian, Y., and Chen, M. : Improved particle swarm optimization algorithm based cross-layer power allocation scheme in distributed antenna systems, Proceedings of the International Conference on Wireless Communications Signal Processing, 2009.

[Toni, 2009] Toni, L. : Rate-Distortion Curve Evaluation for Cross-Layer Optimization in Multimedia Transmission, Proceedings of the Fifth International Joint Conference on INC, IMS and IDC, Washington DC, USA, 2009.

[Nishida, 2006] Nishida, T. Y. : Membrane Algorithms. Lecture Notes in Computer Science, Springer, Berlin/Heidelberg, 3850(2): 55-66, 2006.

[Van der Schaar and Sai Shankhar, 2005] Van der Schaar, M., and Sai Shankar, N. : Cross-layer wireless multimedia transmission challenges principles and new paradigms. Wireless Communications, IEEE, 4(12): 50-58, 2005.

[Van der Schaar and Tekalp, 2005] Van der Schaar, M., and Tekalp, M. : Integrated multi-objective cross-layer optimization for wireless multimedia transmission. IEEE International Symposium on Circuits and Systems, 2005.

[Zhang et al., 2010] Zhang, G., Cheng, J., and Gheorghe, M. : An Approximate Algorithm Combining P Systems and Ant Colony Optimisation for Travelling Salesman Problems. Proceedings of the 8th Brain storming week on Membrane Computing, Servilla, Spain, 2010.

[Zhang et al., 2008] Zhang, G., Gheorge, M., and Wu, C. : A quantum-inspired evolutionary algorithm based on P systems for knapsack problem. Fundamenta Informaticae, 87(1): 93-116, 2008.

[Zhang and Mahfouf, 2006] Zhang, Q., and Mahfouf, M. : A new structure for particle swarm optimisation (nPSO) applicable to single objective and multi objective problems. Proceedings of the 3rd IEEE International Conference on Intelligent Systems, London, 2006.

[Zhang and Huang, 2009] Zhang, Y., and Huang, L. : A variant of P systems for optimisation. Neurocomputing, 72: 1355-1360, 2009.

[Zhou et al., 2010] Zhou, F., Zhang, G., Kong, H., Cheng, J., Gheorghe, M., Ipate, F., and Lefticaru, R. : A Particle Swarm optimisation based on P system. Proceedings of the 6th International Conference on Natural Computimg, ICNC, China, 2010.