Asia Pacific Journal of Multidisciplinary Research
P-ISSN 2350-7756 | E-ISSN 2350-8442 | Volume 2, No. 6 | December 2014

Finding Solutions to Different Problems Simultaneously in a Multi-molecule Simulated Reactor

Jaderick P. Pabico[1], Ma. Christine A. Gendrano[2] and Jose Rene L. Micor[3]

[1]Institute of Computer Science, University of the Philippines Los Baños
[2]College of Computer Studies, De La Salle University – Science and Technology Complex
[3]Institute of Chemistry, University of the Philippines Los Baños
[1]jppabico@uplb.edu.ph, [2]ma.christine.gendrano@dlsu.edu.ph, [3]jrlmicor@uplb.edu.ph

Date Received: November 3, 2014; November 30, 2014

Abstract – *In recent years, the chemical metaphor has emerged as a computational paradigm based on the observation of different researchers that the chemical systems of living organisms possess inherent computational properties. In this metaphor, artificial molecules are considered as data or solutions, while the interactions among molecules are defined by an algorithm. In recent studies, the chemical metaphor was used as a distributed stochastic algorithm that simulates an abstract reactor to solve the traveling salesperson problem (TSP). Here, the artificial molecules represent Hamiltonian cycles, while the reactor is governed by reactions that can re-order Hamiltonian cycles.*

*In this paper, a multi-molecule reactor (*MMR-*n) that simulates chemical catalysis is introduced. The* MMR-*n solves in parallel three NP-hard computational problems namely, the optimization of the genetic parameters of a plant growth simulation model, the solution to large instances of symmetric and asymmetric TSP, and the static aircraft landing scheduling problems (ALSP). The* MMR-*n was shown as a computational metaphor capable of optimizing the cultivar coefficients of CERES-Rice model, and at the same time, able to find solutions to TSP and ALSP. The* MMR-*n as a computational paradigm has a better computational wall clock time compared to when these three problems are solved individually by a single-molecule reactor (*MMR-1*).*

Keywords – Artificial chemistry, combinatorial optimization, traveling salesperson problem, TSP

I. INTRODUCTION

Problems such as the traveling salesman problem (TSP), job-shop scheduling, vehicle routing, scheduling of aircraft landing, gene sequencing, and many others are problems whose solutions are of real-world importance. Because of the combinatorial nature of the problem, exact solutions have been proposed but these solutions were proven inefficient for large problem instances (i.e., they are NP-hard) [1]. Several heuristics have been developed to find time-restrained optimal and near optimal solutions for these problems. These heuristics can be classified into two: graph-based and distributed multi-agent- based. An example of graph-based heuristic is branch and bound [2], while examples of distributed multi-agent-based algorithms are genetic algorithms (GA) [3], memetic algorithms [4–6], tabu search [7], simulated annealing (SA) [8], simulated jumping [9], neural networks [10], and swarm intelligence [11–13].

In recent years, different researchers have shown that the chemical systems of living organisms possess inherent computational properties [14–16]. Because of these, the chemical metaphor has emerged as a computational paradigm [17–27]. Under this computational framework, abstract objects such as atoms or molecules are considered as data or solutions, while interactions (i.e., molecular collisions or reactions) among objects are defined by an algorithm.

Using the chemical metaphor, a distributed stochastic algorithm was designed to simulate a reactor where the molecules are being represented either by a Hamiltonian cycle, a vector of coefficients for a plant growth simulation model, or a schedule for landing aircrafts. The chemical universe in the reactor is governed by reaction rules that can re-order Hamiltonian cycles, alter model coefficients, or program a landing schedule for aircrafts. This computational paradigm can be used to solve, in parallel, very hard real-world problems.

In this effort, a multi-molecule reactor (MMR-*n*) was designed and implemented [25, 26, 28–30] to simulate chemical catalysis for the purpose of concurrently solving three NP-hard computational problems. These problems are:

1. Finding for the solutions to large instances of symmetric and asymmetric TSP;

2. Optimization of cultivar coefficients in CERES-Rice; and

3. Static aircraft landing scheduling problem (ALSP).

The MMR-n was used as a computational metaphor for finding the optimal combination of cultivar coefficients such that the difference between the CERES-Rice predicted growth factors and the observed growth factors are minimized. The MMR-n simulation of catalytic reactions combined better cultivar coefficients than those computed by GENCALC [31]. In addition, MMR-n was also used to find, in parallel, solutions to the TSP and the ALSP.

II. DEVELOPMENT OF MMR-*N*

This section briefly defines the three problems used in this study: the cultivar coefficient optimization, the TSP, and the ALSP. The development of the MMR-n is then discussed, while its underlying reaction algorithms defined, with a focus on solving the three problems in parallel.

A. Traveling Salesperson Problem

The TSP is formally defined as the problem of finding the shortest Hamiltonian cycle of a graph $G(V, E)$ composed of a set of cities $V = \{v_1, v_2, ..., v_n\}$, and a path set $E = \{(v_i, v_j): v_i, v_j \in V\}$. Associated with G is a cost matrix C where each element $c_{i,j} \in \mathbb{R}$ is the cost measure associated with path $(v_i, v_j) \in E$. A Hamiltonian cycle is a closed tour that visits each city $v_i \in V$ once. The Hamiltonian cycle $H = \{h_i \mid i = 1, 2, ..., n, 1\}$ is a vector of indexes h_i, where each h_i encodes a number within the permutation between $1...n$.

A symmetric TSP is when $c_{i,j} = c_{j,i}$, while an asymmetric TSP is when $c_{i,j} \neq c_{j,i}$. The solution to a n-city TSP is a Hamiltonian tour with the minimum cost f_v (Equation 1).

$$f_v = c_{n,1} + \textstyle\sum_{i=1..n-1} c_{i,i+1} \qquad (1)$$

B. CERES-Rice Cultivar Coefficients

Simulating the plant growth in the CERES-Rice model f can be generally viewed as an evaluation of f over the time growth t:

$$\int_{t=\text{planting..harvesting}} f(\{\Psi_1, \Psi_2, ..., \Psi_n\})dt = \{O\}.$$

In this evaluation, $\{\Psi_1, \Psi_2, ..., \Psi_n\}$ is a set of n real-valued vectors that are inputs to f, while $O = \{o_1, o_2, ..., o_m\}$ is a set of m real-valued vectors that describe the time-dependent model outputs [31]. The cultivar coefficients are contained in the input vector $\Psi_1 = (\psi_1, \psi_2, ..., \psi_k)$, where each real-valued ψ represents a quantitative value that is physiologically feasible for a given rice cultivar. All possible combinations of ψ define points in the (k+1)-dimensional space, where each coordinate axis corresponds to one of the ψ's and the (k+1)th axis corresponds to the difference between the predicted (O) and the actual (Ω) growth and development characteristics. The (k+1)th dimension define the terrain of the search space where the lowest points dictates the most desirable combinations of ψ's. It is not known whether Ψ_1 is unique for each rice cultivar (i.e., Does there exists another Ψ^* such that $|O-\Omega|=|O^*-\Omega|$?). Thus, the modality (the number of global optima) of the search space for cultivar determination is unknown. The Ceres-Rice cultivar coefficients [32] used in this paper are listed in Table 1.

Table 1. List of cultivar coefficients used in CERES-Rice.

Variable	Description
P1	Time period (expressed as growing degree days or GDD in °C above a base temperature of 9 °C) from seedling emergence during which the rice plant is not responsive to changes in photoperiod.
P2R	Extent to which phasic development leading to panicle initiation is delayed (expressed as GDD in °C) for each hour increase in photoperiod above P20.
P5	Time period in GDD (°C) from beginning of grain filling (3 to 4 days after flowering) to physiological maturity with a base temperature of 9 °C.
P20	Critical photoperiod or the longest day length (in hours) at which the development occurs at maximum rate.
G1	Potential spikelet number coefficient as estimated from the number of spikelets per gram of main culm dry weight (less lead blades and sheath plus spikes at anthesis).
G2	Single gain weight (g) under ideal growing conditions.
G3	Tillering coefficients relative to IR64 cultivar under ideal conditions.
G4	Temperature tolerance coefficient.

C. Aircraft Landing Scheduling Problem

The ALSP is the problem of deciding a landing time for a set of aircrafts P such that the total penalty cost f_p for landing earlier or later than a target time is minimized. Given, for each aircraft $p_i \in P$ are:

1. the earliest landing time e_i;
2. the latest landing time l_i;
3. the target landing time t_i;
4. the penalty cost per unit time, g_i, for landing before t_i;
5. the penalty cost per unit time, h_i, for landing after t_i;
6. the required separation time between p_i landing and p_j landing s_{ij} (where p_i lands before p_j and $p_j \in P$);
7. the unknown landing time x_i; and
8. a dependency variable δ_{ij}, where $\delta_{ij} = 1$ if p_i lands before p_j and 0 otherwise.

The ALSP is formally defined in the mathematical programming formulation [33,34] shown in Equation 2.

$$f_p = \sum_{i=1..|P|} \{ g_i \max(0, t_i - x_i) + h_i \max(0, x_i - t_i) \} \qquad (2)$$

$$\text{s.t. } \delta_{ij} + \delta_{ji} = 1, j > i; i,j = 1, ..., |P|$$
$$x_j \geq x_i \mid s_{ij} \delta_{ij} - (l_i - e_i) \delta_{ij}, i=j; i,j = 1, ..., |P|$$
$$e_i \leq x_i \leq l_i, i = 1, ..., |P|.$$

D. Multi-molecule Reactor

The MMR-*n* is defined by a triple (M, R, A), where M is a set of artificial molecules, R is a set of reaction rules describing the interaction among molecules, and A is an algorithm driving the reactor. In this paper, the molecules in M are Hamiltonian tours, vector of coefficients, or aircraft landing schedules. The rules in R are reordering algorithms that create new molecules when two molecules collide. The algorithm A describes how the rules are applied to a vessel of artificial molecules simulating a well-stirred, topology-less reactor. A partitions the reactor into different levels of reaction activities. The level of reaction activity is a function of molecular mass.

The set of all permutations $\Pi(V)$ of the $|V|$ cities in V) of the TSP, the set of all binary strings w of finite length L that encode all combinations of cultivar coefficients, and the set of all permutations $\Pi(P)$ of the $|P|$ aircrafts in P in the ALSP were considered as molecules. A permutation π_V encodes a Hamiltonian cycle that solves the TSP, a string w_i encodes a vector of coefficients, and a permutation π_P encodes a landing schedule that solves the ALSP. The cost f_v (Equation 1) of traversing a specific π_V, the difference $|O - \Omega|$ brought about by a w_i, and the cost f_p (Equation 2) of scheduling a specific π_P were considered as molecular mass.

If two molecules m_1 and m_2 collide and they encode solutions to the same problem, they react following a zero-order catalytic reactions of the form

$$m_1 + m_2 + C \rightarrow m_3 + m_4,$$

where m_i are molecules ($\forall i = 1, ..., 4$) and C is a catalyst. The reaction can be mathematically thought of as a function

$$R_1 : M \times M \times M \rightarrow M \times M,$$

where $m_i \in M$. R_1 performs reordering of solutions as described by the following algorithm (let n be the length of the number of cities in the TSP, or the vector of coefficients, or the number of aircrafts in the ALSP – atom is taken as either a city, a vector, or an aircraft):

1. Let an integer $l \in [1, n]$ be the index of the lth atom in any molecule m. Let $i = 1$.
2. Take a random integer between 1 and n and assign it to l. Let $l^0 = \text{l}$.
3. Taking the reactant m_i, locate the lth atom and move it as the lth atom for m_{i+2}.
4. Take note of the lth atom in m_{i+2} and locate it in m_i. Replace l with the value of the index of the atom in m_i.
5. Repeat steps 3 to 4 until the lth atom in m_{i+2} is the same as the l^0th atom in m_i.
6. For all indexes l with no atoms yet in m_{i+2}, move the lth atom from reactant m_i as the lth atom in product m_{i+2}.
7. Repeat steps 2 to 6 for $i = 2$.

If two molecules m_5 and m_6 collide and they encode solutions to different problems, they react following a zero-order catalytic reaction of the form

$$m_5 + m_6 \rightarrow m_7 + m_8.$$

The reaction follows a mathematical function

$$R_2 : M \times M \rightarrow M \times M$$

and is described by the following algorithm:

1. Let $i = 5$.
2. Take molecule m_i and mark the point of collision as l.

3. Take the lth atom in m_i and swap it with the (l + 1)th atom in m_i. If $l = n$, swap the lth atom with the first atom, instead.
4. The resulting molecule is the product m_{i+2}
5. Repeat steps 2 to 4 for i = 6.

If a molecule m_9 hits the bottom or walls of the reactor, a zero-order catalytic reaction of the form

$$m_9 + C \rightarrow m_{10}$$

happens. The reaction is a mathematical function

$$R_3 : M \times M \rightarrow M$$

described by the following algorithm:

1. Mark the point of collision in m_9 as l.
2. Take the lth atom in m_9 and swap it with the (l + k)th atom. With a probability > 0.5, assign $k = 1$, else $k = -1$.
3. If $l = n$, swap the lth atom with the first atom instead (for $k = 1$).
4. If $l = 1$, swap the lth atom with the nth atom instead (for $k = -1$).

The reactor algorithm A operates on a universe of molecules $S = \{m_1 , ..., m_{|S|}\}$, $|S| \ll |M|$. The development of S is realized by applying the following algorithm:

1. Initialize S with $|S|$ molecules selected randomly from M.
2. Using stochastic sampling with replacement, select two molecules m_1 and m_2 from S without removing them.
3. Apply the reaction rule R_1 if m_1 and m_2 encode solutions to the same problem. Otherwise, apply R_2 instead to get the products m_3 and m_4.
4. Apply the reaction rule R_3 for heavy molecules that collide with the reactor walls and bottom.
5. Decay the heavier molecules by removing them out of S and replacing them with randomly selected molecules from M.
6. Repeat steps 2 to 5 until S is saturated with lighter molecules.

One iteration of A constitutes one epoch in the artificial reactor. The sampling procedure gives molecules with low molecular mass a higher probability to react or collide with other molecules. This mimics the level of excitation energy the molecule needs to overcome for it to react with another molecule. This means that the lighter the molecule, the higher the chance that it will collide with other molecules. Step 6 of algorithm A requires a metric for measuring saturation of molecules. In this study, when the number of molecules in that level of excitation has reached 90% of the total molecules encoding the same problem, the MMR-n will stop applying the reaction rules for the same problem and will consider it solved while continuing the simulation for the remaining problems.

III. RESULTS AND DISCUSSION

Using the same datasets and parameters from results on published results on TSP [12], CERES-Rice cultivar coefficient optimization [35] and ALSP [34], the MMR-1 [27] and the MMR-n were run to solve the three problems independently and in parallel, respectively. A single-processor x86 machine with 1.2GHz processing speed running on a multiprogramming operating system was used to run the MMR-n simulations. The MMR-n simulation was repeated 10 times while each of the problem's metrics (i.e., the best minimum for each run) were recorded. The values recorded were averaged and the standard deviation computed. The results of the runs were compared to those of the recent literature as well as to runs with a single-molecule reactor (MMR-1).

A. Symmetric and Asymmetric TSPs

Table 2 compares the average tour lengths found by MMR-n, MMR-1 [27], SA [8], and self-organizing maps (SOM) on five sets of random instances of symmetric 50–city TSPs. The table shows the average value of 10 runs for MMR-n, MMR-1, SA, and SOM, and their respective standard deviations. For each problem, the values were statistically pairwise-compared using the Duncan Multiple Range Test (DMR) at 5% significance level (i.e., statistics α=0.05). For all problems, the respective average tour lengths found by MMR-n are statistically better than thos found by the other three methods. In solving problem 1, MMR-n is best while MMR-1 and SA performed equally well. In solving problem 2, both MMR-n and SA had the best performance. In solving problems 3, 4, and 5, both MMR-based methods had the best performance. Among the four methods, SOM had the worst mean performance in solving the five sets of symmetric 50-city TSPs.

Table 2. Comparison of average tour length found by MMR-*n*, MMR-1 [27], SA [8], and SOM on five sets of random instances of symmetric 50-city TSPs.

Problem	**MMR-***n*	**MMR-**1	**SA**	**SOM**
1	**5.81** *c* (0.0636)	5.87 *b* (0.0564)	5.88 *b* (0.0568)	6.06 *a* (0.0604)
2	**5.99** *c* (0.0455)	6.15 *b* (0.0682)	6.01 *c* (0.0550)	6.25 *a* (0.0873)
3	**5.55** *c* (0.0534)	5.59 *bc* (0.0615)	5.65 *b* (0.0859)	5.83 *a* (0.0849)
4	**5.64** *c* (0.0553)	5.67 *c* (0.0502)	5.81 *b* (0.0803)	5.87 *a* (0.0677)
5	**6.10** *c* (0.0562)	6.15 *c* (0.0535)	6.33 *b* (0.0705)	6.70 *a* (0.0882)

Note: Values are averaged over 10 runs for each problem and search technique combination. The average values in boldfaces are the lowest in each row, while means with the same letter are not statistically different from each other using Duncan's Multiple Range Test at α = 0.05. Figures in parenthesis are standard deviations.

Table 3 compares the integer tour length found by MMR-*n*, MMR-1, GA and ant colony optimization (ACO) on four examples of asymmetric instances of TSP. Aside from the problems being asymmetric, the problem size is increasing: 30-, 50-, 75-, and 100-city TSPs. These problems are based on real-world road networks where the presence of one-way traffic provides the asymmetric constraints. The same DMRT at α=0.05 were conducted on pairwise comparison of mean integer tour length found by the four methods, Analysis shows that at low problem size (30 cities), all methods performed equally well. When the problem size was increased to 50 cities, MMR-*n*, MMR-1 and ACO statistically performed better than GA. When the problem size was increased 25 cities more (75-city TSP), MMR-*n* and GA performed best, while MMR-1 had the second average integer tour length. When the problem size became 100 cities, only MMR-*n* had the best average integer tour length, followed by MMR-1, GA, and ACO, in this order.

Table 3. Comparison of the best integer tour length found by MMR-*n*, MMR-1 [27], GA, and ACO on four examples of asymmetric instances of TSP.

Problem Size	**MMR-***n*	**MMR-**1	**GA**	**ACO**
30	**421** *a*	**421** *a*	**421** *a*	**421** *a*
50	**424** *b*	**424** *b*	428 *a*	**424** *b*
75	**545** *c*	550 *b*	**545** *c*	555 *a*
100	**20,995** *d*	21,280 *c*	21,761 *b*	22,363 *a*

Note: Values are averaged over 10 runs for each problem and search technique combination. The average values in boldfaces are the lowest in each row, while means with the same letter are not statistically different from each other using Duncan's Multiple Range Test at α = 0.05.

B. CERES-Rice Cultivar Coefficients

Using the cultivar coefficients generated from both the MMR-1 and the MMR-*n* runs, the main growth and development variables are shown in Table 4. Results of various model runs using the cultivar coefficients found by GENCALC, by MMR-1, and by MMR-*n* were compared through a metric called *mean normalized absolute error* ε (Equation 3). In this equation, $|X|$ is the cardinality of the set X of growth variables $\{x_1, x_2, ..., x_{|X|}\}$, O_i is the ith growth variable as observed in the actual rice plant, and P_i is the ith growth variable as predicted by CERES-Rice. This metric provides a single value of comparison from all the different simulated and actual growth characteristics. This metric simply shows how far away was the simulated growth variables from the respective observed ones taking in consideration the different units of measure and scales of magnitude of the variables. Thus ε, is unit-less. Note here that statistical comparisons between methods were not performed because GENCALC is a deterministic algorithm which always outputs the same sets of cultivar coefficients from different runs, and therefore gathering a population of data for the purpose of representing the variabilities in the real-world is not possible (i.e., the variance is zero across all runs). In this analysis, the metric provided by Equation 3, which is the simplest among the many that exist, already suffices for comparative performance analysis purposes.

$$\varepsilon = \sum_{i=1..|X|} | O_i - P_i | / O_i \qquad (3)$$

Based on ε, one can clearly see from Table 4 that, in particular, the MMR-*n* performance in predicting dates (flowering date and physiological maturity) having only a maximum error of one day (i.e., ε_i=0.01) is better compared to GENCALC's maximum error of three days (i.e., ε_i=0.05) and to MMR-1's maximum error of two days (i.e., ε_i=0.04). Similarly, MMR-*n* performed better than both GENCALC and MMR-1 in terms of most yield prediction variables (weight per grain, biomass and stalk at harvest maturity, and biomass, stalk and seed N). However, MMR-*n*'s performance in predicting grain yield (i.e., ε_i=0.07) is not better than that of GENCALC's (i.e., ε_i=0.03) nor of MMR-1's (i.e., ε_i=0.01). In fact MMR-1 has the best performance in predicting grain yield with an absolute predictive error of only 49 Kg/ha compared to GENCALC and MMR-*n* with respective absolute predictive errors of 166 Kg/ha and 345 Kg/ha. With only ε_i=0.03 in predicting grain N, GENCALC on the other hand beats both MMR-1 and MMR-*n* with ε_i=0.08 and ε_i=0.09, respectively. In general, however, MMR-1 performed better than GENCALC having 0.07 less ε than GENCALC, but MMR-*n* performed best among the three cultivar-estimation methods having 0.10 and 0.03 less ε than GENCALC and MMR-1, respectively.

Table 4. Comparison between the observed experimental values and the predicted values simulated using GENCALC-generated, MMR-1-generated, and MMR-*n*-generated cultivar coefficients.

Growth Variable	**Actual Observed Data**	**GENCALC**		**MMR-1**		**MMR-*n***	
		Predicted	ε_i	**Predicted**	ε_i	**Predicted**	ε_i
Flowering Date (Day)	57	60	0.05	55	0.04	57	0.00
Physiological maturity (Day)	87	90	0.03	88	0.01	86	0.01
Grain yield (Kg/ha)	5,073	4,907	0.03	5,122	0.01	5,418	0.07
Weight per grain (g)	0.020	0.025	0.25	0.019	0.05	0.021	0.05
Biomass at harvest maturity (Kg/ha)	7,420	8,564	0.15	8,198	0.10	7,725	0.04
Stalk at harvest maturity (Kg/ha)	3,058	4,343	0.42	3,022	0.01	3,066	0.00
Grain N (Kg N/ha)	54.3	56.0	0.03	58.7	0.08	59.0	0.09
Biomass N (Kg N/ha)	72.3	85.0	0.18	84.9	0.17	84.0	0.16
Stalk N (Kg N/ha)	18.0	29.0	0.61	28.3	0.57	26.0	0.44
Seed N (%)	1.17	1.32	0.13	1.30	0.11	1.26	0.08
Mean Normalized Absolute Error, ε			0.19		0.12		0.09

C. Scheduling Aircraft Landings

Table 5 shows the performance of MMR-1 and MMR-*n* as compared to the optimal solutions of eight instances of ALSP at increasing number of aircrafts to be scheduled, four of which are the simplier one-runway problems and four are the harder two-runway problems. From the data presented, it can be seen that both MMR-1 and MMR-*n* were able to find solutions that are near to the optimal ones for all eight problems, although the mean costs found by MMR-1 are numerically greater than the mean costs found by MMR-*n*. Statistically, the mean costs found respectively by MMR-1 and by MMR-*n* are not significantly different from each other for all one-runway problems at increasing number of aircrafts. However, for all the complex two-runway problems and at increasing number of aircrafts, the mean costs found by MMR-1 is statistically significantly different from the mean costs found by MMR-*n*. In particular, MMR-*n*'s performance in solving problems 4 and 8 is significantly better than that of the MMR-1's for the two-runway, 20- and 50-aircraft problem instances. Additionally, the performance of MMR-*n* in solving problems 2 and 6 is highly significantly better than the performance of MMR-1 for the two-runway, 10- and 30-aircraft problem instances. It is, thus, very easy to see that regardless of the number of aircrafts to schedule, MMR-1 and MMR-*n* have statistically equal performance in solving the simpler one-runway problems, and that MMR-*n* is statistically a better solver than MMR-1 for the harder two-runway problems.

Table 5. Comparison between the optimal cost of scheduling aircraft landings and the near-optimal costs /found by MMR-1 and MMR-*n* on eight instances of ALSP. The t-Test is a statistical comparison between the mean cost found by MMR-1 and the mean cost found by MMR-*n*.

Problem Number	**Number of Aircrafts**	**Number of Runways**	**Optimal Solution**	**MMR-1**	**MMR-*n***	**t-Test** $\alpha > \lvert t \rvert$
1	10	1	700	721.4	720.5	0.3306^{ns}
2	10	2	90	94.4	91.9	0.0007**
3	20	1	1,480	1,483.8	1,483.4	0.6051^{ns}
4	20	2	210	219.5	215.9	0.0130*
5	30	1	24,442	24,536.4	24,535.8	0.9763^{ns}
6	30	2	554	561.1	556.6	0.0005**
7	50	1	1,950	2,000.8	2,000.1	0.5202^{ns}
8	50	2	135	139.9	138.1	0.0346*

Note: t-Test is the Pooled 2-tailed Student t-Test at α=0.05 with the assumption that MMR-1 and MMR-*n* are two samples with homoscedastic (i.e., equal) variances. The difference between the mean MMR-1 and mean MMR-*n* costs are (a) ** = highly significantly different from zero (i.e., $\alpha \leq 0.01$), (b) * = significantly different from zero (i.e., $0.01 < \alpha \leq 0.05$), and (c) ns = not significantly different from zero (i.e., $\alpha > 0.05$).

D. Wall Clock Computation Time

The plots in Figure 1 present the time-dependent saturation of the number of minimum-cost molecules in the reactor over all the total number of molecules that encode the cost. For comparison purposes, this number is represented as a percentage. For MMR-1, the percentage is for independently solving the TSP, CERES-Rice, and ALSP using separate MMR-1 runs. For MMR-*n*, however, the percentage is for solving the three problems concurrently using only one MMR-*n* run. The respective plots for both MMR-1 runs and MMR-*n* run show the improvement of the percentage over time. All plots exhibit a jagged sigmoid-like pattern that contains three parts: (1) the *base* where the percentage ranges up to 25%; (2) The *inflection* where the percentage shoots up to more than 50%; and (3) The *head* where the percentage does not go down 75%. Figure 1a shows these parts on a smooth sigmoid plot for visualization purposes. Figure 1b shows the jagged

MMR-1 and the MRR-*n* plots for a representative TSP run. Figure 1c contains the respective representative jagged plots for the CERES-Rice runs, and Figure 1d are jagged plots that represent the runs for solving the ALSP. The random dynamic interactions between molecules in the MMRs brought the jaggedness to the resulting plots. For all plots, it is in the head part where the near-optimal solutions were found by both the MMR-1 and the MMR-*n*. The near-optimal solution was deemed found when the percentage did not exhibit significant improvement after an arbitrary time has elapsed. It can be seen from Figures 1b through 1d that the inflection occur earlier in all MMR-*n* runs than in MMR-1. This means that MMR-*n* had earlier percent saturation of molecules that encode the minimum costs than MMR-1. The near-optimal solutions were found in MMR-*n* earlier than in MMR-1, which intuitively show that MMR-*n* solves the problems faster together than MMR-1 which solves the problems separately.

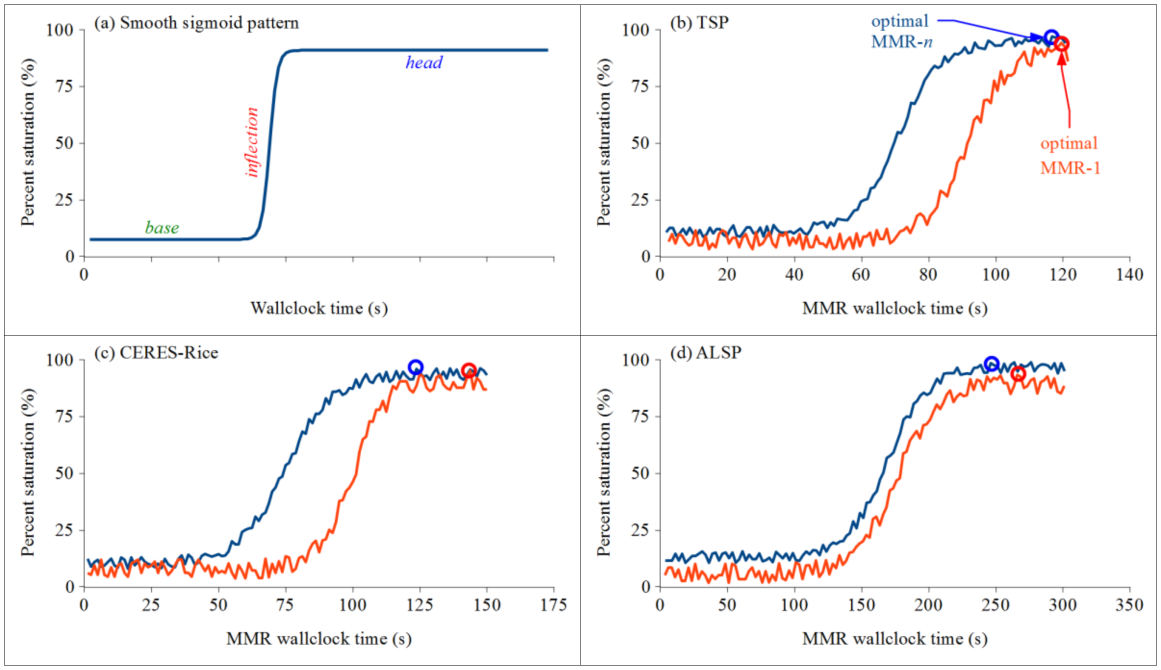

Figure 1. Percent (%) saturation of molecules encoding the minimum cost solutions over wallclock time (s) for the (a) smooth sigmoid-like pattern showing the parts, and the jagged sigmoid-like patterns obtained by MMR-1 and by MMR-*n* solving (b) the TSP, (c) the CERES-Rice cultivar coefficients, and (d) the ALSP.

IV. CONCLUSION

A MMR-*n* that mimicked catalytic reactions was designed to simultaneously solve three unrelated problems. MMR-*n* was found to be better applicable than either MMR-1, SA, SOM, GA, and ACO in finding near-optimal solutions to TSPs, particularly at solving problem instances involving larger number of cities. The catalytic reactions of the MMR-*n* were able to construct a vector of cultivar coefficients with better qualities than those found by either the GENCALC deterministic algorithm or the MMR-1. In addition, MMR-*n* is also better than MMR-1 in solving the ALSPs. As shown in this paper, the MMR-*n* does not only provide nearer optimal solutions but also solves the three unrelated problems faster together than MMR-1 solves them separately.

The work described in this paper can be extended as follows. In order to assess its efficiency, additional experiments are needed with the MMR-*n* simulating more than three problems simultaneously. This experiment will answer the question: “Up to how many problems will the MMR-*n* be still efficient?” Using a single-processor machine as the number of problems being solved simultaneously is increased, it is expected that the MMR-*n* will suffer considerably in terms of efficiency. Thus, further investigations on the use of multi-processor machines are needed to achieve better MMR-*n* performance.

ACKNOWLEDGEMENTS

This research effort is funded partly by and was conducted at the Research Collaboratory for Advanced Intelligent Systems, Institute of Computer Science, University of the Philippines Los Baños, College, Laguna. It is also funded partly by the 2008 Metro Manila Professorial Chair Lecture of the first author. The authors thank their collaborator, Prof. Elmer-Rico E. Mojica, Ph.D., Assistant Professor of Analytical Chemistry, Department of Chemistry and Physical Sciences, Pace University, New York, USA for the idea on multi-molecular interactions.

REFERENCES

[1] M.R. Garey and D.S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco, CA, 1979 (ISBN 0716710455).

[2] S. Tschoke, R. Luling, and B. Monien. *Solving the traveling salesman problem with a distributed branch-and-bound algorithm on a 1024 processor network*. In Proceedings of the 9th International Parallel Processing Symposium (IPPS95), pp 182–189, 1995 (DOI: 10.1109/IPPS.1995.395930).

[3] J.P. Pabico and E.A. Albacea. *The Interactive Effects of Operators and Parameters to GA Performance Under Different Problem Sizes*. Philippine Computing Journal, 3(2):26–37 2008 (ISSN 1908-1995).

[4] P. Moscato and M.G. Norman. *A memetic approach for the traveling salesman problem: Implementation of a computational ecology for combinatorial optimization on message-passing systems*. In M. Valero, E. Onate, M. Jane, J.L. Larriba, and B. Suarez (eds) Parallel Computing and Transputer Applications, pp 187–194. IOS Press, Amsterdam, 1992 (DOI: 10.1.1.50.1940).

[5] B. Freisleben and P. Merz. *A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems*. In Proceedings of the 1996 IEEE International Conference on Evolutionary Computation, pp 616–621, 1996 (DOI: 10.1109/ICEC.1996.542671).

[6] B. Freisleben and P. Merz. *New genetic local search operators for the traveling salesman problems*. In H.M. Voigt, W. Ebeling, I. Rechenberg, and H.P. Schwefel (eds) Proceedings of the 4th Conference on Parallel Problem Solving from Nature (PPSN IV), Volume 1141 of Lecture Notes in Computer Science, pp 890–899. Springer, 1996 (DOI: 10.1007/3-540-61723-X_1052).

[7] M. Zachariasen and M. Darn. *Tabu search on the geometric traveling salesman problem*. In Metaheuristics International Conference 95, Breckenridge, CO, 1995 (DOI: 10.1.1.50.5792).

[8] O.C. Martin and S.W. Otto. *Combining simulated annealing with local search heuristics*. Annals of Operations Research, 63(1):57–75, 1996 (DOI: 10.1007/BF02601639).

[9] S. Amin. *Simulated jumping*. Annals of Operations Research, 86:23–38, 1999 (DOI: 10.1023/A:1018954718550).

[10] O. Miglino, D. Menezer, and P. Bovet. *A neuro-ethological approach for the TSP: Changing metaphors in connectionist models*. Journal of Biological Systems, 2(3): 357–366, 1994 (DOI: 10.1142/S0218339094000210).

[11] L.M. Gambardella and M. Dorigo. *Ant-Q: A reinforcement learning approach to the traveling salesman problem*. In Proceedings of the Twelfth International Conference on Machine Learning, pp 252–260, Tahoe City, CA, 1995.

[12] L.M. Gambardella and M. Dorigo. *Solving symmetric and asymmetric TSPs by ant colonies*. In Proceedings of the IEEE International Conference on Evolutionary Computation, May 20–22 1996, pp 622–627, Nagoya, Japan, 1996 (DOI: 10.1109/ICEC.1996.542672).

[13] M. Dorigo and L.M. Gambardella. *Ant colonies for the traveling salesman problem*. BioSystems, 43(2):73–81, 1997 (DOI: 10.1016/S0303-2647(97)01708-5) .

[14] A. Hjemfelt, E.D. Weinberger, and J. Ross. *Chemical implementation of neural networks and Turing Machines*. Proceedings of National Academy of Sciences of the United States, 88(24):10983–10987, 1991 (PMID: 1763012).

[15] L.M. Adleman. *Molecular computation of solutions to combinatorial problems*. Science, 266(5187):1021–1024, 1994 (DOI: 10.1126/science.7973651).

[16] A. Arkin and J. Ross. *Computational functions in biochemical reaction networks*. Journal of Biophysics, 67(2):560–578, 1994 (PMID: 7948674).

[17] G. Berry and G. Boudol. *The chemical abstract machine*. Journal of Theoretical Computer Science, 96:217–248, 1992 (DOI: 10.1145/96709.96717).

[18] W. Fontana. *Algorithmic chemistry*. In C.G. Langton, C. Taylor, J.D. Farmer, and S. Rasmussen (eds) Proceedings of the Workshop on Artificial Life (ALIFE90), volume 88, pp 159–209, Redwood City, CA, 1992. Addison-Wesley.

[19] W. Banzhaf. *Self-organizing algorithms derived from RNA interactions*. In W. Banzhaf and F.H. Eeckman (eds) Evolution and Biocomputing, volume 899, pp 69–103. Springer, Berlin, 1995 (DOI: 10.1007/3-540-59046-3_6).

[20] T. Ikegami and T. Hashimoto. *Coevolution of machines and tapes*. In F. Moran, A. Moreno, J.J. Merelo, and P. Chacon (eds) Advances in Artificial Life: Proceedings of Third European Conference on Artificial Life, volume 929, pp 234– 245, Berlin, 1995. Springer-Verlag (DOI: 10.1007/3-540-59496-5_302).

[21] W. Banzhaf, P. Dittrich, and H. Rauhe. *Emergent computation by catalytic reactions*. Nanotechnology, 7(1):307–314, 1996 (DOI: 10.1.1.32.8076).

[22] P. Dittrich, W. Banzhaf, H. Rauhe, and J. Ziegler. *Macroscopic and microscopic computation in an artificial chemistry*. In P. Dittrich, H. Rauhe, and W. Banzhaf (eds) Proceedings of the Second German Workshops on Artificial Life (GWAL97), pp 19–22, University of Durtmond, 1998 (DOI: 10.1.1.51.1060).

[23] J.P. Pabico, E.R.E. Mojica, and J.R.L. Micor. *Artificial chemistry: Basic concepts and applications to combinatorial problems*. Kimika: The Journal of Chemical Society of the Philippines, 19(2):77–82, 1999 (ISSN: 0115-2130).

[24] J.P. Pabico, M.J.M. Mendoza, and M.C.A. Gendrano. *Solving symmetric and asymmetric TSPs by artificial chemistry*. In Proceedings of the 4th Philippine Computing Science Congress (PCSC 2004), University of the Philippines Los Baños, College, Laguna, 2004 (ISSN: 1908-1146).

[25] J.P. Pabico. *Simultaneously solving computational problems using an artificial chemical reactor*. In R.P. Saldaña (ed) Proceedings of the 6th Philippine Computing Science Congress (PCSC 2006), Ateneo De Manila University, Loyola Heights, Quezon City, 2006 (ISSN: 1908-1146).

[26] J.P. Pabico. *Artificial catalytic reactions in 2D for combinatorial optimization*. In H.N. Adorna, editor, Proceedings of the 3rd Symposium on Mathematical Aspects of Computer Science (SMACS 2006), Adventist University of the Philippines, Silang, Cavite, 2006.

[27] J.P. Pabico, J.R.L. Micor, and M.C.A. Gendrano. *A molecular dynamics heuristic for solving the traveling salesperson problem*. Asia Pacific Journal of Education, Arts and Sciences, 1(1):38–46, 2014 (ISSN : 2362-8022).

[28] J.P. Pabico and E.R.E. Mojica. *Coevolution in artificial catalytic reactions*. In 28th Annual Philippine-American Academy of Science and Engineering (PAASE) Meeting and Symposium (28th APAMS), Georgetown University, Washington DC, USA, 2008.

[29] J.P. Pabico. *Molecular dynamics as a computational metaphor for search, optimization and machine learning*. In First DOST-PCASTRD National Symposium on Science and Technology (NSST 2008), SEARCA, University of the Philippines Los Baños, College, Laguna, 2008.

[30] J.P. Pabico. *Solving different problems simultaneously with artificial chemistry*. In Proceedings of the 9th Philippine Computing Science Congress (PCSC 2009), Silliman University, Dumaguete City, 2008 (ISSN: 1908-1146).

[31] L. A. Hunt, S. Pararajasingham, J. W. Jones, G. Hoogenboom, D. T. Imamura, and R. M. Ogoshi. *GENCALC: Software to facilitate the use of crop models for analyzing field experiments*. Agronomy Journal, 85(5):1090–1094, 1993 (DOI:10.2134/ agronj1993.00021962008500050025x).

[32] G.Y. Tsuji, G. Uehara, and S. Balas (eds). DSSAT v3, volume 1–3. University of Hawaii, Honolulu, HI, 1994.

[33] J.E. Beasley, M. Krishnamoorthy, Y.M. Sharaiha, and D. Abramson. *Scheduling aircraft landings—The static case*. Transportation Science, 34:180–197, 2000 (DOI: 10.1287/ trsc.34.2.180.12302).

[34] J.E. Beasley, M. Krishnamoorthy, Y.M. Sharaiha, and D. Abramson. *The displacement problem and dynamically scheduling aircraft landings*. Journal

of the Operational Research Society, 55:54–64, 2004 (DOI: 10.1.1.163.9605).

[35] J.P. Pabico, G. Hoogenboom, and R.W. McClendon. *Determination of cultivar coefficients of crop models using a genetic algorithm: A conceptual framework*. Transactions of the ASAE, 42(1):223–232, 1999 (DOI: 10.13031/2013.13199).