INT J COMPUT COMMUN, ISSN 1841-9836
7(5):916-923, December, 2012.

Evolutionary Algorithm based on the Automata Theory for the Multi-objective Optimization of Combinatorial Problems

E. Niño-Ruiz

Elias D. Niño-Ruiz
Universidad del Norte
KM5 Via Puerto Colombia
Barranquilla, Colombia
Web: `http://combinatorialoptimization.blogspot.com/`
E-mail: enino@uninorte.edu.co

Abstract:
This paper states a novel, Evolutionary Metaheuristic Based on the Automata Theory (EMODS) for the multiobjective optimization of combinatorial problems. The proposed algorithm uses the natural selection theory in order to explore the feasible solutions space of a combinatorial problem. Due to this, local optimums are often avoided. Also, EMODS exploits the optimization process from the Metaheuristic of Deterministic Swapping to avoid finding unfeasible solutions. The proposed algorithm was tested using well known multi-objective TSP instances from the TSPLIB. Its results were compared against others Automata Theory inspired Algorithms using metrics from the specialized literature. In every case, the EMODS results on the metrics were always better and in some of those cases, the distance from the true solutions was 0.89%.
Keywords: Combinatorial Optimization, Multi-objective Optimization, Automata Theory, Metaheuristic of Swapping.

1 Introduction

As well known, Combinatorial Optimization is a branch of the Optimization. Its domain is optimization problems where the set of feasible solutions is discrete or can be reduced to a discrete one, and the goal is to find the best possible solution [8]. In this field it is possible to find a large number of problems denominated NP-Hard, that is mean that the problem does not have a solution in polynomial time. One of the most classical problems in the combinatorial optimization field is the Traveling Salesman Problem (TSP), it has been analyzed for years [6] either in a mono or multi-objective manner. Formally, TSP is defined as follows:

$$min \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} \cdot X_{ij}, \tag{1}$$

subject to:

$$\sum_{j=1}^{n} X_{ij} = 1, \forall i = 1, \ldots, n, \tag{2a}$$

$$\sum_{j=1}^{n} X_{ij} = 1, \forall j = 1, \ldots, n, \tag{2b}$$

$$\sum_{i \in \kappa} \sum_{j \in \kappa} X_{ij} \leq |\kappa| - 1, \forall \kappa \subset \{1, \ldots, n\}, \tag{2c}$$

$$X_{ij} = 0, 1 \forall i, j, \tag{2d}$$

where C_{ij} is the cost of the path X_{ij} and κ is any nonempty proper subset of the cities $1, \ldots, m$. (1) is the objective function. The goal is the optimization of the overall cost of the tour. (2a), (2b) and (2d) fulfill the constrain of visiting each city only once. Lastly, Equation (2c) set the subsets of solutions, avoiding cycles in the tour.

TSP has an important impact on different sciences and fields, for instance in Operations Research and Theoretical Computer Science. Most problems related to those fields, are based in the TSP definition. For instance, The Hard Scheduling Optimization [5] had been derived from TSP. Although several algorithms have been proposed for the solution of TSP, there is not one that optimal solves it. For this reason, this paper discuss novel metaheuristics based on the Automata Theory in order to approach the solution of the Multi-objective Traveling Salesman Problem. This paper is structured as follows: in section 2 important definitions about the multi-objective combinatorial optimization and the metaheuristics based on the automata theory are given, section 3 discusses an evolutionary metaheuritic based on the automata theory for the multi-objective optimization of combinatorial problems, lastly, in section 4 and 5 experimental results are given for each algorithm in order to estimate their performance using multi-objective metrics from the specialized literature.

2 Preliminaries

2.1 Multi-objective Optimization

The multi-objective optimization consists in two or more objectives functions to optimize and a set of constraints. Mathematically, the multi-objective optimization model is defined as follows:

$$\textit{optimize} \quad F(X) = \{f_1(X), f_2(X), \ldots, f_n(X)\}, \tag{3}$$

subject to:

$$H(X) = 0, \tag{4a}$$

$$G(X) \leq 0, \tag{4b}$$

$$X_l \leq X \leq X_u, \tag{4c}$$

where $F(X)$ is the set of objective functions, $H(X)$ and $G(X)$ are the constraints of the problem. Lastly, X_l and X_u are the bounds for the set of variables X.

2.2 Metaheuristic of Deterministic Swapping (MODS)

Metaheuristic Of Deterministic Swapping (MODS) [4] is a local search strategy that explores the feasible solution space of combinatorial problems based on a data structure named Multi Objective Deterministic Finite Automata (MDFA) [3]. A MDFA is a Deterministic Finite Automata that allows the representation of the feasible solution space of combinatorial problems. Formally, a MDFA is defined as follows:

$$M = (Q, \Sigma, \delta, Q_0, F(X)), \tag{5}$$

where Q represents all the set of states of the automata (feasible solution space), Σ is the input alphabet that is used for δ (transition function) to explore the feasible solution space of a combinatorial problem, Q_0 contains the initial set of states (initial solutions) and $F(X)$ are the objectives to optimize. MODS explores the feasible solution space represented through a MDFA using a search direction given by an elitist set of solutions (Q_*). The elitist solution are states that, when were visited, their solution dominated at least one solution in Q_ϕ. Q_ϕ contains all the states with non-dominated solutions.

Lastly, the template algorithm of MODS is defined as follows:

1. Create the initial set of solutions Q_0 using a heuristic relative to the problem to solve.

2. Set Q_ϕ as Q_0 and Q_* as ϕ.

3. Select a random state $q \in Q_\phi$ or $q \in Q_*$

4. Explore the neighborhood of q using δ and Σ. Add to Q_ϕ the solutions found that are not dominated by elements of Q_f. In addition, add to Q_* those solutions found that dominated at least one element from Q_ϕ.

5. Check stop condition, go to 3.

2.3 Simulated Annealing Metaheuristic of Deterministic Swapping (SAMODS)

Simulated Annealing & Metaheuristic Of Deterministic Swapping [2] (SAMODS) is a hybrid local search strategy based on the MODS theory and Simulated Annealing algorithm for the multi-objective optimization of combinatorial problems. Its main propose consists in optimizing a combinatorial problem using a Search Direction and an Angle Improvement. SAMODS is based in the next Automata:

$$M = (Q, Q_0, P(q), F(X), A(n)), \tag{6}$$

Alike MODS, Q_0 is the set of initial solutions, Q is the feasible solution space, $F(X)$ are the functions of the combinatorial problem, $P(q)$ is the permutation function ($P(q) : Q \to Q$) and $A(n)$ is the weighted function ($A(n) : \mathbb{N} \to \Re^n$). n represents the number of objective for the combinatorial problem.

SAMODS exploits the search directions given by MODS and it proposed an angle direction given by the function $A(n)$. Due to this, SAMODS template is defined as follows:

1. Setting sets. Set Q_0 as the set of Initial Solutions. Set Q_ϕ and Q_* as Q_0.

2. Settings parameters. Set T as the initial temperature, n as the number of objectives of the problem and ρ as the cooler factor.

3. Setting Angle. If T is equal to 0 then got to 8, else set $T_{i+1} = \rho \times T_i$, randomly select $s \in Q_\phi$, set $W = A(n) = \{w_1, w_2, \cdots, w_n\}$ and go to 4.

4. Perturbing Solutions. Set $s' = P(s)$, add to Q_ϕ and Q_* according to the next rules:

$$Q_\phi = Q_\phi \cup \{s'\} \Leftrightarrow (\nexists r \in Q_\phi)(r \text{ is better than } s'), \tag{7a}$$

$$Q_* = Q_* \cup \{s'\} \Leftrightarrow (\exists r \in Q_*)(s' \text{ is better than } r), \tag{7b}$$

   then, if Q_ϕ has at least one element that dominated to s' go to step 5, otherwise go to 7.

5. Guess with dominated solutions. Randomly generated a number $n \in [0, 1]$. Set z as follows:
$$z = e^{(-(\gamma/T_i))}, \tag{8}$$
where T_i is the temperature value in moment i and γ is defined as follows:
$$\gamma = \sum_{i=1}^{n} w_i \cdot f_i(s_X) - \sum_{i=1}^{n} w_i \cdot f_i(s'_X), \tag{9}$$
where s_X is the vector X of solution s, s'_X is the vector X of solution s', w_i is the weight assigned to the function i and n is the number of objectives of the problem. If $n < z$ then set s as s' and go to 4 else go to 6.
6. Change the search direction. Randomly select a solution $s \in Q_*$ and go to 4.
7. Removing dominated solutions. Remove the dominated solutions for each set (Q_* and Q_ϕ). Go to 3.
8. Finishing. Q_ϕ has the non-dominated solutions.

2.4 Genetic Simulated Annealing Metaheuristic of Deterministic Swapping (SAGAMODS)

Simulated Annealing, Genetic Algorithm & Metaheuristic Of Deterministic Swapping [2] (SAGAMODS) is a hybrid search strategy based on the Automata Theory, Simulated Annealing and Genetics Algorithms. SAGAMODS is an extension of the SAMODS theory. It comes up as result of the next question: could SAMODS quickly avoid local optimums? Although, SAMODS avoids local optimums guessing, it can take a lot of time accepting dominated solutions for finding non-dominated. Thus, the answer to this question is based on the Evolutionary Theory. SAGAMODS proposes crossover step before SAMODS template is executed. Due to this, SAGAMODS supports to SAMODS for exploring distant regions of the solution space. Formally, SAGAMODS is based on the next automata:
$$M = (Q, Q_S, C(q, r, k), F(X)), \tag{10}$$
where Q is the feasible solutions space, Q_S is the initial solutions and $F(X)$ are the objectives of the problem. $C(q, r, k)$ is defined as follows:
$$C(q, r, k) : Q \to Q, \tag{11}$$
where $q, r \in Q$ and $k \in N$. q and r are named parents solutions and k is the cross point. Lastly, SAGAMODS template is defined as follows:

1. Setting parameters. Set Q_S as the solution set, x as the number of solutions to cross for each iteration.
2. Set Q_C (crossover set) as selection of x solutions in Q_S, Q_M (mutation set) as ϕ and k as a random value.
3. *Crossover.* For each $s_i, s_{i+1} \in Q_C/1 \le i < \|Q_C\| : \quad Q_M = Q_M \cup \{C(s_i, s_{i+1}, k)\}$
4. *Mutation.* Set Q_0 as Q_M. Execute SAMODS as a local search strategy.
5. Check stop conditions. Go to 2.

3 Evolutionary Metaheuristic of Deterministic Swapping (EMODS)

Evolutionary Metaheuristic of Deterministic Swapping (EMODS), is a novel framework that allows the Multiobjective Optimization of Combinatorial Problems. Its framework is based on MODS template therefore its steps are the same: create initial solutions, improve the solutions (optional) and execute the core algorithm. Unlike SAMODS and SAGAMODS, EMODS avoids the slowly convergence of Simulated Annealing's method. EMODS explores different regions from the feasible solution space and search for non-dominated solution using Tabu Search. The core algorithm is defined as follows:

1. Set θ as the maximum number of iterations, β as the maximum number of state selected in each iteration, ρ as the maximum number of perturbations by state and Q_ϕ as Q_0.
2. Randomly select a state $q \in Q_\phi$ or $q \in Q_*$.
3. *Mutation - Tabu Search* Set N as the new solutions found as result of perturbing q. Add to Q_ϕ and Q_* according to the next equations:

$$(Q_\phi = Q_\phi \cup \{q\}) \iff (\nexists r \in Q_\phi / r \text{ is better than } q) \tag{12a}$$

$$(Q_* = Q_* \cup \{q\}) \iff (\exists r \in Q_\phi / q \text{ is better than } r) \tag{12b}$$

   and then, the states with dominated solutions for each set are removed.

4. *Crossover.* Randomly, select states from Q_ϕ and Q_*. Generate a random point of cross.
5. Check stop condition, go to 3.

Step 2 and 3 support the algorithm in removing dominated solutions from the set of solutions Q_ϕ as can be seen in figure 3. However, one of the most important steps in the EMODS algorithm is 4 where new solutions are found after the crossover step.

4 Experimental Analysis

4.1 Experimental Settings

The algorithms were tested using well-known instances from the multi-objective TSP taken from TSPLIB [1]. The test of the algorithms was conducted using a dual core computer with 2 Gb RAM. The optimal solutions were constructed based on the best non-dominated solutions of all algorithms in comparison for each instance used. The instances were constructed using the combination of the mono-objective instances KROA100, KROB100, KROC100, KROD100 and KROE100. For instance, KROAB100 is a bi-objective instance whose matrices of distance are given by the instance KROA100 and KROB100. We full combine the instances (KROAB100, KROAC100, . . ., KROABCDE100) and then we run the experiments. The metrics used for the measurement of the different algorithms are described below, most of them use two Pareto fronts. The first one is PF_{true} and it refers to the real optimal solutions of a combinatorial problem. The second is PF_{know} and it represents the optimal solutions found by an algorithm. In all the cases $\|\cdot\|$ represents the number of elements.

$$GNDV = \|PF_{know}\|, \tag{13}$$

$$ReGNDV = \|\{y | y \in PF_{know} \wedge y \in PF_{true}\}\|, \tag{14}$$

where Generation of Non-dominated Vectors (GNDV) and Real Generation of Non-dominated Vectors (ReGNDV) measure the number of solutions and the number of true solutions found by an algorithm respectively. On the other measures the number of true solutions generated. On the other hand, Generational Distance (GD) and Inverse Generational Distance (IGD) measure the distance between FP_{know} and FP_{true}:

$$GD = \left(\frac{1}{\|PF_{know}\|}\right) \cdot \left(\sum_{i=1}^{\|PF_{know}\|} d_i\right)^{(1/p)}, IGD = \left(\frac{1}{\|PF_{true}\|}\right) \cdot \left(\sum_{i=1}^{\|PF_{know}\|} d_i\right), \tag{15}$$

where d_i is the smallest Euclidean distance between the solution i of FP_{know} and the solutions of FP_{true} and p is the dimension of the combinatorial problem. For the measurement of the range variance of neighboring solutions in PF_{know} the Spacing (S) is proposed:

$$S = \left(\frac{1}{\|PF_{know}\| - 1}\right)^2 \cdot \left(\sum_{i=1}^{\|PF_{know}\|} \left(\overline{d} - d_i\right)^2\right)^{(1/p)} \tag{16}$$

where d_i is the smallest Euclidean distance between the solution i and the rest of solutions in PF_{know}. $\overline{d} = \frac{1}{\|PF_{true}\|} \sum_{i=1}^{\|PF_{true}\|} d_i$. The Error Rate ($\varepsilon$) depicts the error rate respect to the precision of the solutions as follows:

$$\varepsilon = \left(\left|\frac{\|PF_{true}\| - \|ReGNDV\|}{\|PF_{true}\|}\right|\right) \cdot 100\% \tag{17}$$

4.2 Experimental Results

The average of the metrics applied to each algorithm are shown in table 1. Furthermore, a graphical comparison for tri-objectives instances is shown in figure 1.

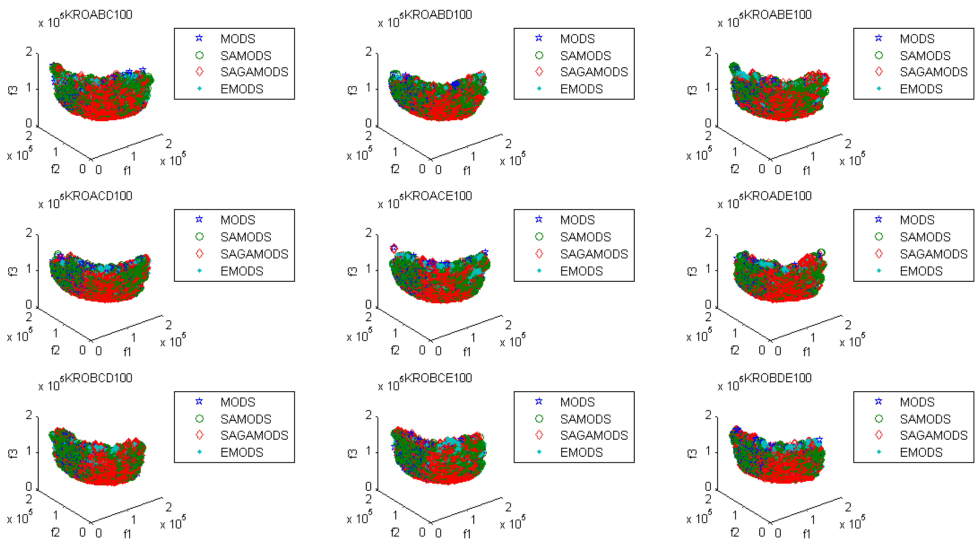

Figure 1: Graphical comparison between MODS, SAMODS, SAGAMODS and EMODS for tri-objective TSP instances.

Table 1: Average performance for the algorithms in comparison using multi-objective instances of TSP with multi-objective optimization metrics.

INSTANCE	*ALGORITHM*	*GNDV*	*ReGNDV*	$\left(\frac{ReGNDV}{GNDV}\right)\%$	*S*	*GD*	*IGD*	ε
Bi-objective TSP	MODS	262.7	0	0%	0.0286	21.6672	2329.4338	100%
	SAMODS	6487.2	1425.7	22.03%	0.0016	0.2936	265.8974	89.47%
	SAGAMODS	6554.5	1581.8	23.97%	0.0015	0.3062	286.547	88.3%
	EMODS	19758.6	10671.8	54.89%	0.0003	0.0492	75.2773	22.23%
Tri-objective TSP	MODS	1992.5	63.9	3.21%	0.1508	0.302	3206.7459	99.91%
	SAMODS	12444.2	269.3	2.16%	0.0727	0.0434	2321.5258	99.6%
	SAGAMODS	12332.5	271.1	2.2%	0.0743	0.0437	2312.3389	99.6%
	EMODS	68969.1	67097	97.3%	0.0468	0.0011	6.3914	0.89%
Quad-objective TSP	MODS	5364.8	3273.2	60.99%	0.3468	0.0252	5810.4824	94.31%
	SAMODS	27639.6	11594.2	41.94%	0.2325	0.0043	3397.7495	79.87%
	SAGAMODS	35649.6	14754.8	41.4%	0.2231	0.0032	3013.1894	74.39%
	EMODS	200420.6	27991.6	13.97%	0.176	0.0005	1891.9864	51.43%
Quint-objective TSP	MODS	7517	7517	100%	0.5728	0.0125	15705.6864	98.41%
	SAMODS	26140	26140	100%	0.4101	0.0033	10801.6382	94.46%
	SAGAMODS	26611	26611	100%	0.4097	0.0033	10544.8901	94.36%
	EMODS	411822	411822	100%	0.3136	0.0001	950.4252	12.77%

5 Conclusion

SAMODS, SAGAMODS and EMODS are algorithms based on the Automata Theory for the multi-objective optimization of combinatorial problems. All of them are derived from the MODS metaheuristic, which is inspired in the Theory of Deterministic Finite Swapping. SAMODS is a Simulated Annealing inspired Algorithm. It uses a search direction in order to optimize a set of solution (Pareto Front) through a linear combination of the objective functions. On the other hand, SAGAMODS, in addition to the advantages of SAMODS, is an Evolutionary inspired Algorithm. It implements a crossover step for exploring far regions of a solution space. Due to this, SAGAMODS tries to avoid local optimums owing to it takes a general look of the solution space. Lastly, in order to avoid slow convergence, EMODS is proposed. Unlike SAMODS and SAGAMODS, EMODS does not explore the neighborhood of a solution using Simulated Annealing, this step is done using Tabu Search. Thus, EMODS gets optimal solution faster than SAGAMODS and SAMODS. Lastly, the algorithms were tested using well known instances from TSPLIB and metrics from the specialized literature. The results shows that for instances of two, three and four objectives, the proposed algorithm has the best performance as the metrics values corroborate. For the last instance worked, quint-objective, the behavior of MODS, SAMODS and SAGAMODS tend to be the same, them have similar error rate but, EMODS has a the best performance. In all the cases, EMODS shows the best performance. However, for the last test, all the algorithms have different solutions sets of non-dominated solutions, and those form the optimal solution set.

Acknowledgment

First of all, I want to thank to God for being with me in my entire life, He made this possible. Secondly, I want to thank to my parents Elias Niño and Arely Ruiz and my sister Carmen Niño for their enormous love and support. Finally, and not less important, to thank to my beautiful wife Maria Padron and our baby Maria Gabriela for being my inspiration.

Bibliography

[1] University Of Heidelberg. Tsplib - office research group discrete optimization - university of heidelberg. `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/`.

[2] Elias D. Niño. Samods and sagamods: Novel algorithms based on the automata theory for the multi-objective optimization of combinatorial problems. *Int. J. of Artificial Intelligence - Special issue of IJAI on Metaheuristics in Artificial Intelligence*, accepted, 2012.

[3] Elias D. Niño, Carlos Ardila, Yezid Donoso, and Daladier Jabba. A novel algorithm based on deterministic finite automaton for solving the mono-objective symmetric traveling salesman problem. *Int. J. of Artificial Intelligence*, 5(A10):101-108, 2010.

[4] Elias D. Niño, Carlos Ardila, Yezid Donoso, Daladier Jabba, and Agustin Barrios. Mods: A novel metaheuristic of deterministic swapping for the multi objective optimization of combinatorials problems. *Computer Technology and Application*, 2(4):280-292, 2011.

[5] Elias D. Niño, Carlos Ardila, Adolfo Perez, and Yezid Donoso. A genetic algorithm for multiobjective hard scheduling optimization. *INT J COMPUT COMMUN*, 5(5):825-836, 2010.

[6] J.G. Sauer and L. Coelho. Discrete differential evolution with local search to solve the traveling salesman problem: Fundamentals and case studies. In *Cybernetic Intelligent Systems, 2008. CIS 2008. 7th IEEE International Conference on*, pages 1-6, 2008.

[7] Yang Xiawen and Shi Yu. A real-coded quantum clone multi-objective evolutionary algorithm. In *Consumer Electronics, Communications and Networks (CECNet), 2011 International Conference on*, 4683-4687, 2011.

[8] Qin Yong-Fa and Zhao Ming-Yang. Research on a new multiobjective combinatorial optimization algorithm. In *Robotics and Biomimetics, 2004. ROBIO 2004. IEEE International Conference on*, 187-191, 2004.