A Learning Automata Based Algorithm For Solving Capacitated Vehicle Routing Problem

Mir Mohammad Alipour[1]

[1] **Department of Computer Engineering, University of Bonab, Bonab 5551761167, Iran**

Abstract

This paper presents an approximate algorithm based on distributed learning automata for solving capacitated vehicle routing problem. The vehicle routing problem (VRP) is an NP-hard problem and capacitated vehicle routing problem variant (CVRP) is considered here. This problem is one of the NP-hard problems and for this reason many approximate algorithms have been designed for solving it. Distributed learning automata that is a general searching tool and is a solving tool for variety of NP-complete problems, is used to solve this problem and tested on fourteen benchmark problems. Our results were compared to the best known results. The results of comparison have shown the efficiency of the proposed algorithm.

Keywords: *Vehicle routing problem, Capacitated vehicle routing problem, Distributed learning automata, 2-opt local search heuristic, Candidate list, Mutation operation.*

1. Introduction

Finding efficient vehicle routes is an important logistics problem which has been studied for several decades. When a firm is able to reduce the length of its delivery routes or is able to decrease its number of vehicles, it is able to provide better service to its customers, operate in a more efficient manner and possibly increase its market share. A typical vehicle routing problem includes simultaneously determining the routes for several vehicles from a central supply depot to a number of customers and returning to the depot without exceeding the capacity constraints of each vehicle.

The process of selecting vehicle routes allows the selection of any combination of customers in determining the delivery route for each vehicle. Therefore, the vehicle routing problem is a combinatorial optimization problem where the number of feasible solutions for the problem increases exponentially with the number of customers to be serviced. In addition, the vehicle routing problem is closely related to the traveling salesman problem where an out and back tour from a central location is determined for each vehicle. Since there is no known polynomial algorithm that will find the optimal solution in every instance, the vehicle routing problem is considered NP-hard [1].

Because of this nature of the problem, it is not realistic to use exact methods to solve large instances of the problem. Most approaches are based on heuristics. Several, mostly hybrid, heuristics have been applied for solving the VRP problem. Some of them are: a hybrid search method which associates non-monotonic simulated annealing to hill-climbing and random restart [2], hybrid discrete particle swarm optimization algorithm (DPSO) [3], an improved ant colony optimization (IACO) [4], honey bees mating optimization algorithm [5], optimized crossover genetic algorithm [6].

This paper uses an efficient Distributed Learning Automata (DLA) algorithm to find solutions to the capacitated vehicle routing problem. In order to test the proposed algorithm we used fourteen benchmark problems. This paper is organized as follows. Section 2 presents the capacitated vehicle routing problem. Section 3 describes the Learning Automata and Distributed Learning Automata. Section 4 describes three improvement strategies. In Section 5 the implementation details of the DLA algorithm for the CVRP problem are described. Section 6 presents the experimentations and results. Conclusion and some possible plans for future work are in Section 7.

2. Capacitated Vehicle Routing Problem

Many versions of the vehicle routing problem have been described. The capacitated vehicle routing problem (CVRP) is discussed here and it can be described as follows:
Goods are to be delivered to a set of customers by a fleet of vehicles from a central depot. Each vehicle has limited capacity and each customer has a certain demand. The locations of the depot and the customers, the capacity of each route and the demand of each customer are given. The objective is to determine a viable route schedule which minimizes the distance or the total cost with the following constraints:
1. Each customer is served exactly once by exactly one vehicle.
2. Each vehicle starts and ends its route at the depot.

3. The total length of each route must not exceed the constraint.
4. The total demand of any route must not exceed the capacity of the vehicle.

Usually, the CVRP is represented by a complete weighted graph G = (V , E), with n + 1 nodes, where $V = \{0,1,\dots,N\}$ is a set of vertices corresponding to the customers (or delivery points) i ($i = 1,\dots,n$) are to be served by K vehicles and the depot 0 and $E = \{(i,j) : i \neq j\}$ is a set of edges. Each edge (i,j) is associated with a non-negative d_{ij} which represents the distance (or travels time) between i and j. The demand of customer i is q_i, the capacity of vehicle k is Q_k and the maximum allowed travel distance by vehicle k is D_k. Then the mathematical model [7] of the CVRP is described as follows:

$$\text{Minimize} \sum_{k=1}^{K}\sum_{i=0}^{N}\sum_{j=0}^{N} C_{ij}^{k}X_{ij}^{k} \quad (1)$$

subject to:

$$X_{ij}^{k} = \begin{cases} 1, & \text{if vehicle k travel from customer i to j} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

$$\sum_{k=1}^{K}\sum_{i=0}^{N} X_{ij}^{k} = 1\,, j = 1,2,\dots,N \quad (3)$$

$$\sum_{k=1}^{K}\sum_{j=0}^{N} X_{ij}^{k} = 1\,, i = 1,2,\dots,N \quad (4)$$

$$\sum_{i=0}^{N} X_{it}^{k} - \sum_{j=0}^{N} X_{tj}^{k} = 0, k = 1,2,\dots,K; t = 1,2,\dots,N \quad (5)$$

$$\sum_{i=0}^{N}\sum_{j=0}^{N} d_{ij}^{k}X_{ij}^{k} \leq D_k\,, k = 1,2,\dots,K \quad (6)$$

$$\sum_{j=0}^{N} q_j\left(\sum_{i=0}^{N} X_{ij}^{k}\right) \leq Q_k, k = 1,2,\dots,K \quad (7)$$

$$\sum_{j=1}^{N} X_{0j}^{k} \leq 1, k = 1,2,\dots,K \quad (8)$$

$$\sum_{i=1}^{N} X_{i0}^{k} \leq 1, k = 1,2,\dots,K \quad (9)$$

$$X_{ij}^{k} \in \{0,1\}, i,j = 0,1,\dots,N\,, k = 1,2,\dots,K \quad (10)$$

Where N represents the number of customers, K is the number of vehicles, C_{ij}^{k} is the cost of travelling from customer i to customer j by vehicle k and d_{ij}^{k} is the travel distance from customer i to customer j by vehicle k. The objective function Eq. (1) is to minimize the total cost by all vehicles. Constraint Eq. (3) and Eq. (4) ensure that each customer is served exactly once. Constraint Eq. (5) ensures the route continuity. Constraint Eq. (6) shows that the total length of each route has a limit. Constraint Eq. (7) shows that the total demand of any route must not exceed the capacity of the vehicle. Constraints Eq. (8) and Eq. (9) ensure that each vehicle is used no more than once. Constraint Eq. (10) ensures that the variable only takes the integer 0 or 1.

The vehicle routing problem is of great practical significance in real life. It appears in a large number of practical situations, such as transportation of people and products or delivery service. An example of a single solution consisting of a set of routes constructed for a CVRP is presented in Fig. 1, where k=4 and n=18. So, the solution of this example is that: 0-15-14-1-16-0; 0-3-2-4-10-5-0; 0-6-17-18-8-0; 0-7-9-11-12-13-0.

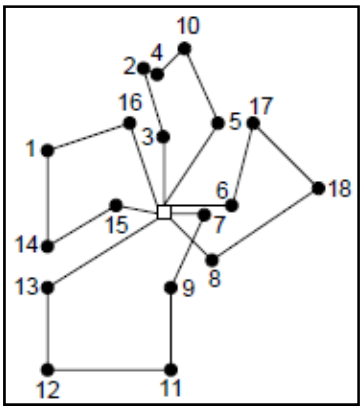

Fig. 1 Example of CVRP solution.

3. Learning automata

A learning automaton [8-17] is an adaptive decision-making unit that improves its performance by learning how to choose the optimal action from a finite set of allowed actions through repeated interactions with a random environment. The action is chosen at random based on a probability distribution kept over the action set and at each instant the given action is served as the input to the random environment.

The environment responds the taken action in turn with a reinforcement signal. The action probability vector is updated based on the reinforcement feedback from the environment. The objective of a learning automaton is to find the optimal action from the action set so that the

average penalty received from the environment is minimized [8].

The environment can be described by a triple $E \equiv \{\alpha, \beta, c\}$, where $\alpha \equiv \{\alpha_1, \alpha_2, ..., \alpha_r\}$ represents the finite set of the inputs, $\beta \equiv \{\beta_1, \beta_2, ..., \beta_m\}$ denotes the set of the values can be taken by the reinforcement signal, and $c \equiv \{c_1, c_2, ..., c_r\}$ denotes the set of the penalty probabilities, where the element c_i is associated with the given action α_i. If the penalty probabilities are constant, the random environment is said to be a stationary random environment, and if they vary with time, the environment is called a non stationary environment. The environments depending on the nature of the reinforcement signal $\underline{\beta}$ can be classified into P-model, Q-model and S-model. The environments in which the reinforcement signal can only take two binary values 0 and 1 are referred to as P-model environments. Another class of the environment allows a finite number of the values in the interval [0, 1] can be taken by the reinforcement signal.

Such an environment is referred to as Q-model environment. In S-model environments, the reinforcement signal lies in the interval [a, b]. The relationship between the learning automaton and its random environment has been shown in Fig. 2.

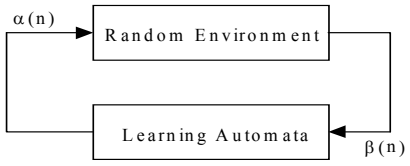

Fig. 2 The relationship between the learning automaton and its random environment.

Learning automata can be classified into two main families [8-13] fixed structure learning automata and variable structure learning automata. Variable structure learning automata are represented by a triple $\{\underline{\beta}, \underline{\alpha}, T\}$, where $\underline{\beta}$ is the set of inputs, $\underline{\alpha}$ is the set of actions, and T is learning algorithm. The learning algorithm is a recurrence relation which is used to modify the action probability vector. Let α(k) and p(k) denote the action chosen at instant k and the action probability vector on which the chosen action is based, respectively. The recurrence equation shown by Eq. (11) and Eq. (12) is a linear learning algorithm by which the action probability vector p is updated. Let $\propto_i$ (k) be the action chosen by the automaton at instant k.

$$p_j(n+1) = \begin{cases} p_j(n) + a[1 - p_j(n)] & j = i \\ (1-a)p_j(n) & \forall j \ j \neq i \end{cases} \quad (11)$$

When the taken action is rewarded by the environment (i.e., $\beta(n) = 0$) and

$$p_j(n+1) = \begin{cases} (1-b)p_j(n) & j = i \\ \frac{b}{r-1} + (1-b)p_j(n) & \forall j \ j \neq i \end{cases} \quad (12)$$

When the taken action is penalized by the environment (i.e., $\beta(n) = 1$). r is the number of actions can be chosen by the automaton, a(k) and b(k) denote the reward and penalty parameters and determine the amount of increases and decreases of the action probabilities, respectively. If a(k) = b(k), the recurrence equations (11) and (12) are called linear reward-penalty (L_{R-P}) algorithm, if a(k) >> b(k) the given equations are called linear reward-ε penalty ($L_{R-\varepsilon P}$), and finally if b(k) = 0 they are called linear reward-Inaction (L_{R-I}). In the latter case, the action probability vectors remain unchanged when the taken action is penalized by the environment.

Learning automata have been found to be useful in systems where incomplete information about the environment, wherein the system operates, exists. Learning automata are also proved to perform well in dynamic environments. It has been shown in Refs. [15-16] that the learning automata are capable of solving the NP-hard problems.

3.1 Distributed learning automata

A distributed learning automata (DLA) is a network of the learning automata which collectively cooperate to solve a particular problem [14]. Formally, a DLA can be defined by a quadruple $< A, E, T, A_0 >$ where $A = \{A_1, \ldots, A_n\}$ is the set of learning automata, $E \subset A \times A$ is the set of the vertices in which vertex $e_{(i,j)}$ corresponds to the action $\propto_j$ of the automaton A_i, T is the set of learning schemes with which the learning automata update their action probability vectors and A_0 is the root automaton of DLA from which the automaton activation is started. An example of a DLA has been shown in Fig. 3.

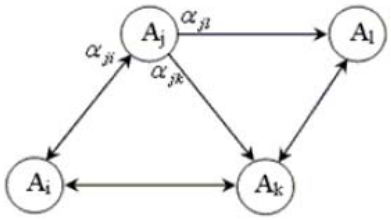

Fig. 3 Distributed learning automata.

The operation of a DLA can be described as follows: At first, the root automaton randomly chooses one of its outgoing vertices (actions) according to its action probabilities and activates the learning automaton at the other end of the selected vertex. The activated automaton also randomly selects an action which results in activation of another automaton. The process of choosing the actions and activating the automata is continued until a leaf automaton (an automaton which interacts to the environment) is reached. The chosen actions, along the path induced by the activated automata between the root and leaf, are applied to the random environment. The environment evaluates the applied actions and emits a reinforcement signal to the DLA. The activated learning automata along the chosen path update their action probability vectors on the basis of the reinforcement signal by using the learning schemes. The paths from the unique root automaton to one of the leaf automata are selected until the probability with which one of the paths is chosen is close enough to unity. Each DLA has exactly one root automaton which is always activated, and at least one leaf automaton which is activated probabilistically.

4. Three improvement strategies

Research shows that the attainment of improved solutions to the VRP is dependent on route improvement strategies in the algorithm [17]. The first of these strategies involves the inclusion of a local exchange procedure to act as an improvement heuristic within the routes found by individual vehicles. The technique used for this purpose is the common 2-opt local search heuristic [18] where all possible pair wise exchanges of customer locations visited by individual vehicles are tested to see if an overall improvement in the objective function can be attained.

The principle of the 2-opt is to delete 2 arcs of the same route and to replace them by 2 other arcs in order to improve the cost of this route and to delete the road junctions. In the Fig. 4, the two scattered arcs are deleted in the left graph. They are replaced by the two arcs labeled (A and B) in the right graph in order to find a new route. The direction of some arcs in directed graphs, that are not concerned by a 2-opt operation, can be modified during the construction of the new route.

The second improvement strategy is the use of a candidate list for determining the next location selected in a vehicle route. Each individual location is assigned a candidate list based on the distance to all other locations in the location set. Only the closest locations are included in the candidate list for the current location and are made available for selection as the next location to be visited in the route.

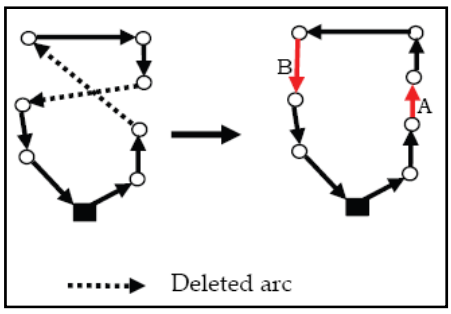

Fig. 4 The heuristic principle of local search 2-OPT.

The size of the candidate list has been determined in the past by restricting its size to a fraction of the total number of customers in the problem. For example, it is possible to set the candidate list size equal to one forth of the total number of customers regardless of the number of customers. For problems with fifty customers the candidate list is restricted to the rounded integer value of twelve (n/4). Fig. 5 depicts the twelve closest candidate locations for the current location during the construction of a vehicle route.

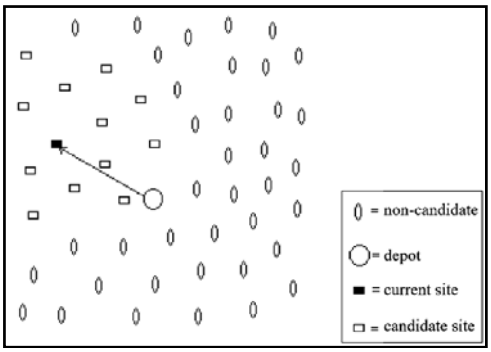

Fig. 5 Route selection with candidate list.

It is believed that this restriction prevents the algorithm from wasting effort considering moves to locations that are a great distance from the current location and have very little chance of creating an improved solution to the problem.

The third improvement strategy (SwapMutation) is the use of the mutation operation to exchange the customers of two routes in a random fashion. The steps for this mutation operator are as follows:

1. Randomly select two routes from the last solution and randomly select two customers from each selected route.
2. Exchange the customers in the different routes and generate the new solution.

When the pair of customers whose exchange produces the shortest distance is found, the route is rearranged. If we don't find such pair of customers, the route stays unchanged.

5. DLA Algorithm for CVRP

At first k networks of learning automata which are isomorphic to the graph of VRP instance is created (k DLAs). In these networks each node is a learning automaton and each outgoing edge of this node (connecting this city to other city according to its candidate list) is one of the actions of this learning automaton.

Each DLA simulates a vehicle, and its route is constructed by incrementally selecting customers until all customers have been visited or the capacity constraint of the vehicle is met. Initially, First DLA starts at the depot (automaton at the depot is activated) and the set of customers included in its tour is empty.

Then at the each stage, active automaton of first DLA chooses one of its actions (from candidate list of actions) based on its action probability vector and heuristic information (it prefer to choose short edges) according to the probability distribution given in Eq. (13) and the storage capacity of the vehicle is updated before another automaton (customer) is selected.

$$\acute{P}^{j} = \{\acute{P}_i^{j} \left| \acute{P}_i^{j} = \frac{[p_i^{j} \times W^{-1}(j,i)]^{\beta}}{\sum_{i=1}^{r}[p_i^{j} \times W^{-1}(j,i)]^{\beta}} \right. : \; i = 1,2,\dots,r\} \quad (13)$$

Where P_i^j is the is the probability with which automaton j chooses to move from city j to city i, $W^{-1}(j,i)$ is the inverse of the distance between city j to city i, i is the one of cities that can be visited by automaton on city j (to make the solution feasible), and $\beta \geq 1$ is a parameter which determines the relative importance of P_i^j versus distance. In Eq. (13) we multiply the P_i^j by the corresponding heuristic value $W^{-1}(j,i)$. In this way we favor the choice of edges which are shorter and which have a greater P_i^j.

This action activates automaton on the other end of edge. The process of choosing an action and activating an automaton is repeated until the capacity constraint of the vehicle is met or when all customers are visited.

In order to exclude loops from the traversed paths, the algorithm meets every node along a path being traversed once (except depot). To implement this, if an automaton chooses action $\propto_k$ from the list of its actions, then all inactivated automatons (unvisited nodes) of all DLAs will disable action $\propto_j$ in their list of actions. However, at the next iteration of the algorithm, all the disabled actions will be enabled.

The action of a DLA is a sequence of actions that represents a particular route in the graph. After this, the solution is improved by 2-opt heuristic. The DLA algorithm constructs a complete tour for the first DLA prior to the second DLA starting its tour. This continues until a predetermined number of DLA k each construct a feasible route.

After k DLAs construct a feasible route, the third improvement strategy (SwapMutation) is used and randomly two routes from the last solution selected and randomly two customers of these route exchanged. When the pair of customers whose exchange produces the shortest distance is found, the routes are rearranged. If we don't find such pair of customers, the route stays unchanged.

The total distance L is computed as the objective function value for the complete route of the each DLA. The environment uses the length of these routes to produce its response. This response causes the actions along the best route be rewarded globally in all DLAs according to linear Reward-Inaction learning algorithm (L_{R-I}).

This updating encourages the use of shorter routes and increases the probability that future routes will use the arcs contained in the best solutions. This process is repeated for a predetermined number of iterations and the best solution from all of the iterations is presented as an output of the model and should represent a good approximation of the optimal solution for the problem. The outline of our proposed algorithm is shown in Fig. 6.

Construct K DLAs isomorphic to the graph of VRP instance
and initialize those probability vectors
Repeat
For m = 1 to = k
Produce a route by DLA_m
Perform 2-opt on the route of DLA_m
Next
Perform third improvement strategy (SwapMutation)
Compute the route length of each DLA
Globally reward the selected actions of all LAs along the best route according to the L_{R-I}
All DLAs enable disabled actions of each LAs
Until (termination condition reached)

Fig. 6 DLA for CVRP

6. Experimentations and Results

In this section, fourteen VRP benchmark problems described in Christofides et al. [19] are tested by our proposed algorithm. These problems contain between 50 and 199 customers as well as the depot. The customers in

problems 1-10 are randomly distributed in the plane, while they are clustered in problems 11-14. Problems 1-5 and 6-10 are identical, except that the total length of each vehicle route is limited for the latter problems. Problems 13-14 are the counterparts of problems 11-12 with additional route length constraint. In addition, the best known solutions of C1 and C12 had been proved to be optima [20, 21]. Information on these instances is summarized in Table 1. Columns 2-7 show the problem size n, the vehicle capacity Q, the maximum route length T, the best known solutions BKS [22,23], the best solution of DLA algorithm, the worst solution, the average solution and average run time (second).

The DLA parameters used for CVRP instances are $a = \frac{2*(N-2)}{N*(N-1)}$, b = 0, k = N/10, $\beta = 5$ and the candidate list size is set at N/5 .The DLA algorithm were coded in Visual C++.Net 2008 and executed on a PC equipped with 512 MB of RAM and a Pentium processor running at 1.7 GHz.

Our proposed algorithm (DLA) are compared with five meta-heuristic approaches in the reference [24], which consisted of parallel tabu search algorithm (RR-PTS) by Rego and Roucairol, a tabu search algorithm (G-TS) by Gendreau et al., tabu search (OSM-TS), a simulated annealing algorithm (OSM-SA) by Osman and ant system algorithm (B-AS) by Bullnheimer et al. (1997). The comparison of the deviations from the best known solution is shown in Table 2. The performance of our proposed algorithm is best among all approaches, who produces in ten problems of fourteen test problems and yields the lowest average deviation. Also, compared with OSM-SA, the tabu search approaches are able to provide better solutions. Also, compared OSM-SA, the tabu search approaches can provide better solutions.

For a correct evaluation and comparison of the quality of six algorithms the computing times must be taken into account. However, a correct evaluation and comparison of the computing times is generally tough due to the enormous variety of computers available and used by different researchers. A very rough measure of computers' performance can be obtained using reference [25] tables where the number (in millions) of floating point operations per second (Mflop/seconds) executed by each computer was used, when solving standard linear equations, with LINPACK program. Regarding computational times, Rego and Roucairol used a sun sparc 4 (about 5.7 MFlop/s), Gendreau et al. used a 36 MHz Silicon Graphics (about 6.7 MFlop/s), Osman used a VAX 8600 (about 2.48 MFlop/s), Bullnheimer et al. used a Pentium 100 MHz (about 8 MFlop/s). In this research, the Pentium 1.7 GHz running DLA algorithm has an estimated power of 94 MFlop/s. Table 3 shows the origin computation times and the scaled computation times, which use Pentium 1.7 GHz as the baseline, of six approaches.

The performance of proposed algorithm is competitive when compared with other approaches, such as SA, and TS. Although the run times are not favor in DLA algorithm, our algorithm still seems to be superior in terms of solution quality with an average deviation of 0.13%. Considering the very rough measure, the scaled times are viewed as the assistant aspect of the performance. Regarding the computation efficiency, we find that the DLA algorithm can find very good solutions in an acceptable time.

7. Conclusion

In this paper, we presented the DLA algorithm for capacitated vehicle routing problem. The computational study of fourteen benchmark problems shows that the proposed algorithm is competitive when compared with other approximate approaches, such as SA, GA and TS. Though the run time is not favor in DLA, our DLA still produces the largest number of best know solutions among all approximate approaches. Regarding the computation efficiency, we find that the DLA can find very good solutions in a short time. This strength can be combined with other approximate approaches in the future work.

References

[1] M. Labbe, G. Laporte, and H. Mercure, Capacitated vehicle routing on trees, Operations Research, vol. 39, no. 4, pp. 61–622, 1991.

[2] Oliveira H. C. B., Vasconcelos G. C., A hybrid search method for the vehicle routing problem, Annals of Operations Research, Vol. 180, No. 1, 2010, pp. 125-144.

[3] Chen A., Yang G., Wu Z., Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem, Journal of Zheijang University SCIENCE A, Vol. 7, No. 4, 2006, pp. 607-614.

[4] Yu B., Yang Z., Yao B., An improved ant colony optimization for vehicle routing problem, European Journal of Operational Research, Vol. 196, Issue 1, 2009, pp. 171–176.

[5] Marinakis Y., Marinaki M., Dounias G., Honey Bees Mating Optimization algorithm for large scale vehicle routing problems, Natural Computing, Vol. 9, Issue 1, 2010, pp. 5-27.

[6] Nazif H., Lee L. S., Optimized Crossover Genetic Algorithm for Vehicle Routing Problem with Time Windows, American Journal of Applied Sciences, Vol. 7, Issue 1 2010, pp. 95-101.

[7] Bodin L, Golden B., Assad A., Ball M., Routing and scheduling of vehicles and crews: The state of the art, Computers and Operations Research, Vol. 10, No. 2, 1983, pp. 63-211.

[8] K. S. Narendra and K. S. Thathachar, Learning Automata: An Introduction, (Prentice-Hall, New York, 1989).

[9] M. A. L. Thathachar and P. S. Sastry, A hierarchical system of learning automata that can learn the globally optimal path, Information Science 42 (1997) 743–766.
[10] M. A. L. Thathachar and B. R. Harita, Learning automata with changing number of actions, IEEE Trans. Systems, Man, and Cybernetics SMG17 (1987) 1095–2400.
[11] M. A. L. Thathachar and V. V. Phansalkar, Convergence of teams and hierarchies of learning automata in connectionist systems, IEEE Trans. Systems, Man and Cyber-netics 24 (1995) 1459–1469.
[12] S. Lakshmivarahan andM. A. L. Thathachar, Bounds on the convergence probabilities of learning automata, IEEE Trans. Systems, Man, and Cybernetics, SMC-6 (1976) 756–763.
[13] K. S. Narendra and M. A. L. Thathachar, On the behavior of a learning automaton in a changing environment with application to telephone traffic routing, IEEE Trans. Systems, Man, and Cybernetics SMC-l0(5) (1980) 262–269.
[14] H. Beigy and M. R. Meybodi, Utilizing distributed learning automata to solve stochastic shortest path problems, Int. J. Uncertainty, Fuzziness and Knowledge-Based Systems 14 (2006) 591–615.
[15] J. Akbari Torkestani and M. R. Meybodi, A new vertex coloring algorithm based on variable action-set learning automata, J. Computing and Informatics 29(3) (2010) 447–466.
[16] J. Akbari Torkestani and M. R. Meybodi, Graph coloring problem based on learning automata, in Proc. Int. Conf. Information Management and Engineering (ICIME2009), Kuala Lumpur, Malaysia, April 3–5, 2009.
[17] Bullnheimer B, Hartl RF, Strauss C. Applying the ant system to the vehicle routing problem. In: Voss S, Martello S, Osman IH, Roucairol C, editors. Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization. Boston: Kluwer; 1998.
[18] Lin S, Kernighan BW. An effective heuristic algorithm for the TSP. Oper Res 1973;21:498–516.
[19] Christofides, N., A. Mingozzi and P. Toth, The Vehicle Routing Problem. In: Combinatorial Optimization, ed. N. Christofides, A. Mingozzi, P. Toth, & C. Sandi. Wiley, Chichester (1979).
[20] Golden, B. L., E. A. Wasil, J. P. Kelly and I-M. Chao, Metaheuristic in vehicle routing, In T. G. Crainic and G. Laporte, editors. Fleet Management and Logistics, Kluwer, Boston, 33-56 (1998).
[21] Hadjiconstantinou, E., N. Christofides and A. Mingozzi, A new exact algorithm for the vehicle routing problem based on q-paths and k-shortest paths relaxations, In M. Gendreau and G. Laporte, editors, Freight Transportation, Volume 61 of Annals of Operations Research, 21-43 (1995).
[22] Rochat, Y. and E. Taillard, Probabilistic and intensification in local search for vehicle routing, Journal of Heuristics, 1, 147-167 (1995).
[23] Taillard, E., Parallel iterative search methods for vehicle routing problems, Networks, 23, 661-673 (1993).
[24] Bullnheimer, B., Hartl, R.F., Strauss, C., 1997. Applying the ant system to the vehicle routing problem. In: Second Metaheuristics International Conference, MIC'97, Sophia-Antipolis, France.
[25] Dongarra, J., 2001. Performance of various computer using standard linear equations software. Report CS-89-85, University of Tennessee.

Mir Mohammad Alipour has a B.A. in Computer Engineering from Isfahan University, 2001 and a M.S. in Computer Engineering from Amirkabir University of Technology, 2004. Starting at University of Bonab, He has co-authored number of papers published in international and national conferences including CSI 2005, IKT 2005, ICEE 2006, CSICC 2006, CSCCIT 2011. His current interests are in optimization, learning automata and approximate algorithms.

Table 1: CVRP experiment results of DLA

NO.	N	Q	T	BKS	Best	Worst	Average	Time(s)
C1	50	160	∞	524.61	524.61	526.22	525.04	2
C2	75	140	∞	835.26	835.26	843.14	840.23	10
C3	100	200	∞	826.14	829.62	864.2	848.75	28
C4	150	200	∞	1028.42	1028.42	1058.69	1040.9	253
C5	199	200	∞	1291.45	1306.1	1335.19	1317.44	528
C6	50	160	200	555.43	555.43	568.24	561.3	21
C7	75	140	160	909.68	909.68	940.5	922.78	20
C8	100	200	230	865.94	865.94	890.27	876.5	55
C9	150	200	200	1162.55	1162.55	1197.13	1190.32	298
C10	199	200	200	1395.85	1395.85	1440.38	1408.64	853
C11	120	200	∞	1042.11	1042.35	1050.4	1044.28	59
C12	100	200	∞	819.56	819.56	836.82	827.64	32
C13	120	200	720	1541.14	1543.71	1571.48	1557.18	121
C14	100	200	1040	866.37	866.37	868.15	867.8	45

Table 2: Deviations from the best known solution of several approaches

No.	RR-PTS	G-TS	OSM-TS	OSM-SA	B-AS	DLA
C1	0.00	0.00	0.00	0.65	0.00	0.00
C2	0.01	0.06	1.05	0.40	1.08	0.00
C3	0.17	0.40	1.44	0.37	0.75	0.42
C4	1.55	0.75	1.55	2.88	3.22	0.00
C5	3.34	2.42	3.31	6.55	4.03	1.13
C6	0.00	0.00	0.00	0.00	0.87	0.00
C7	0.00	0.39	0.15	0.00	0.72	0.00
C8	0.09	0.00	1.39	0.09	0.09	0.00
C9	0.14	1.31	1.85	0.14	2.88	0.00
C10	1.79	1.62	3.23	1.58	4.00	0.00
C11	0.00	3.01	0.09	12.85	2.22	0.02
C12	0.00	0.00	0.01	0.79	0.00	0.00
C13	0.59	2.12	0.31	0.31	1.22	0.17
C14	0.00	0.00	0.00	2.73	0.08	0.00
Average	0.55	0.86	1.03	2.10	1.51	0.13

Table 3: Computation times of several approaches

No.	RR-PTS		G-TS		OSM-TS		OSM-SA		B-AS		DLA
	Run times	Scaled times	Run times	Scaled times	Run times	Scaled times	Run times	Scaled times	Run times	Scaled times	Run times
C1	66	3.96	84	5.96	60	1.56	6	0.17	6	0.51	1
C2	2604	156.24	2352	166.99	48	1.25	3564	92.67	78	6.63	8
C3	1578	94.68	408	28.97	894	23.24	6174	160.52	228	19.38	32
C4	2910	174.6	3270	232.17	1764	45.86	4296	111.7	1104	93.84	194
C5	4626	277.56	5028	356.99	1704	44.30	1374	35.72	5256	446.76	537
C6	144	8.64	468	33.23	60	1.56	696	18.1	6	0.51	19
C7	1236	74.16	1908	135.47	744	19.34	312	8.11	102	8.67	20
C8	1134	68.04	354	25.13	1962	51.01	366	9.52	288	24.48	51
C9	1794	107.64	1278	90.74	2472	64.27	59,016	1534.42	1650	140.25	280
C10	2562	153.72	2646	187.87	4026	104.68	2418	62.87	4908	417.18	622
C11	672	40.32	714	50.7	780	20.28	264	6.86	552	46.92	54
C12	96	5.76	102	7.24	342	8.89	48	1.25	300	25.5	33
C13	120	7.2	2088	148.25	1578	41.03	4572	118.87	660	56.1	119
C14	1482	88.92	1782	126.52	582	15.13	300	7.8	348	29.58	41
Avg.	–	90.1	–	114.02	–	31.6	–	154.9	–	94.02	143.64