

Applying the Intelligence of Ant and Tabu Search to Solve The 8-puzzle Problem

Ruqaya Z. Sha'ban

nnabeel2013@gmail.com

College of Medicine

University of Mosul, Mosul, Iraq

Received on: 25/01/2012

Accepted on: 19/04/2012

ABSTRACT

The research tackled artificial intelligent methods to solve one of the optimization problems by using artificial ant by applying ant colony optimization algorithm and also tabu search algorithm to find the solution of sliding tile 8-puzzel problem. In ant colony algorithm generated many possible solutions depending on finding the difference tiles in initial state from the goal and moving accordingly in the current state of the problem. In Tabu search, many possible solutions have been generated according to the replacement relation between different tiles in initial state to find the optimal solution from many solutions. In this research, the experimental show is very speed to obtain the goal. The source code is written in MATLAB language to simulate these two algorithms.

Keywords: Ant colony optimization algorithm, Tabu search, 8-puzzle problem.

تطبيق ذكاء النمل والبحث الممنوع في حل مسألة اللغز المحير -8

رقية زيدان شعبان

كلية الطب، جامعة الموصل

تاريخ قبول البحث: 2012/04/19

تاريخ قبول البحث: 2012/01/25

المخلص

تطرق البحث إلى استخدام طرق الذكاء الاصطناعي لحل مسألة من مسائل الأمثلية باستخدام ذكاء النمل من خلال تطبيق خوارزمية مستعمرة النمل المثلى وتطبيق خوارزمية البحث الممنوع لإيجاد حل مسألة اللغز المحير -8، تم توليد الحلول الممكنة في خوارزمية النمل من خلال الاعتماد على إيجاد المواقع المختلفة في الحل الابتدائي عن الحل الهدف وتتحرك حسب موقعها في الحالة الحالية لرقعة اللعبة، إما في خوارزمية البحث الممنوع تم اعتماد توليد الحلول الممكنة من خلال علاقة تبادل المواقع المختلفة في الحل الابتدائي للوصول إلى الحل الأمثل من بين العديد من الحلول. أثبتت الطريقتان سرعة الوصول إلى حل الهدف. تم إعداد برنامج حاسوبي بلغة ماتلاب ليحاكي كلتا الخوارزميتين. الكلمات المفتاحية: خوارزمية مستعمرة النمل، البحث الممنوع، مسألة اللغز المحير 8.

1. Introduction

The 8-puzzle is a square tray in which are placed 8 square tiles. The remaining ninth square is uncovered. Each tile has a number on it. A tile that is adjacent to be the blank space can be slid into that space. A game consists of a starting position and a specified goal position [12]. A player can slide an adjacent tile into a position occupied by the blank tile. The goal of this game is to move the tiles so as to reach the goal state where all numbers are placed in an increasing order from left to right and from top to bottom [10]. The objective of the 8-puzzle is to rearrange a given initial configuration of eight squared tiles on a 3×3 board into a specified goal configuration by successively sliding tiles into the orthogonally adjacent empty square(the blank square) [11 and 13]. The research aims to solve the 8-puzzle problem by ant colony optimization algorithm and tabu search algorithm through proposed algorithm . This algorithm solves the problem by depending on the different tails.

In this paper, besides this introductory section, ant colony optimization and tabu search are described in sections 2 and section 3. Section 4 presents the Pseudo -code of

algorithms. Practically, Representation is explained in section 5. Finally, section 6 concludes this paper.

2. Ant Colony Optimization

Ant algorithms were first proposed by Dorigo and co-authors [6] as some system of cooperating agents (a multi-agent system) for various optimization problems. Ant algorithms were inspired by the observation of real ant colonies. While walking from the nest to food sources and vice versa, ants deposit on the ground a substance called pheromone, forming in this way a pheromone trail. The role of this trail is to guide ants toward the source of food (or to the nest). It has been shown that the quantity of pheromone left by an ant depends on the amount of food found, it is also obvious that the more the ants which reach the source of food, the stronger the pheromone left [3,4,5,6 and 9].

Ant colony optimization algorithm makes use of simple agents called ants which iteratively construct candidate solutions to a combinatorial optimization problem [4]. The ant's solution construction is guided by (artificial) pheromone trail and problem dependent heuristic information. Ant colony optimization algorithm can be applied to any combinatorial optimization problem by defining solution components which the ants use to iteratively construct candidate solutions and on which they may deposit a pheromone. Partial problem solutions are seen as states:

Each ant moves from a state i to another state j corresponding to a more complete partial solution.

At each step, each ant k computes a set of feasible expansion to its current state and moves to one of these state according to a probability distribution [3and 7].

2-1 Pheromone Update

Real ants deposit a substance called pheromone while moving from one point to another point. Artificial ants perform this action by adding a value called trace on the trail levels of moves chosen by them [2,3 and 14].

In this paper, each movement (direction) in the sliding tile 8-puzzelis is given an initial pheromone value denoted by fer equal to 1. The size of the colony of ant is the number of differences between current solution and goal denoted by n . The pheromone update denoted by Δfer , then the update rule is as follows:

$$fer_{new} = fer_{old} + \Delta fer \quad \dots(1)$$

$$\Delta fer = n/9, \text{ nine is the number of squares with blank in problem.} \quad \dots(2)$$

The heuristic information indicates the desirability of assigning the i to the location j . There are several methods to estimate the desirability depending on the problem. In 2011, Alkallak and Sha'ban [2] do not include the heuristic information because the ants movement in some deterministic way and they visit all edges in the search space depending on the constraints of the problem. Also, in this paper, the proposed algorithm is used without heuristic information.

Ant colony optimization algorithm is similar to other metaheuristic. It can be terminated in several manners, repeating the algorithm for a maximized number of iterations or running for a stipulated time [3 and 14]. In this paper, termination manner belongs to the solution of the problem obtained

2-2 Pheromone Evaporate

Pheromones evaporate over time, preventing levels becoming unbounded, and allowing the ant colony to "forget" old information. We implemented this by reducing the amount of pheromone on each edge once per cycle, The pheromone evaporation is denoted by p . where $0 < p < 1$ [8].

In this research, the evaporation rate is 0.2, pheromone information is updated (Pheromone Evaporate).

The update rule is as follows:

$$fer_{new} = (1-p) * fer_{old} \quad \dots(3)$$

3. Tabu Search

Tabu search is based on the neighborhood search with local-optima avoidance, but in a rather deterministic way. The key idea of tabu search is to allow climbing movements when no improving movement (neighboring solution) exists. However, some movements have to be forbidden in order to avoid cycling. So, the tabu search starts from an initial solution s , may be, randomly generated in S and movements repeatedly from a solution to a neighboring one. At each step of the procedure, a set (subset) of the neighboring solutions of the current solution s is considered and the movement that improves most the objective function value f is chosen. If there are no improving movements, tabu search chooses one that at least degrades the objective function, a movement is performed to the best neighbor s' .

In order to avoid returning to the local optimal solution just visited, the reverse movement must be forbidden (prohibited). This is done by storing this movement (or an attribute of the move) into a memory (or more precisely short-term-memory) managed like a circular list T and called a tabu list. The tabu list keeps information on the last movements which have been done during the search process (the parameter is called a tabu list size). Thus, a movement from s to s' is considered as tabu if it (or its attribute) is contained in the list T . This way of proceeding hinders the algorithm from returning to a solution reached in the last steps. However, it might be worth returning after a while to a solution visited previously to search in another direction. Consequently, an aspiration criterion is introduced to permit the tabu status to be dropped under certain favorable circumstances. Typically, a tabu movement from s to s' is permitted where the best solution is found so far [1 and 9].

4. Pseudo-code of Algorithms

In this research, we solved 8-puzzle problem by two algorithms as ant colony and tabu search. In 2010, Sha'ban et al proposed a table which contains the number of tile's move for index of tile and direction of tile's move. Also, in this research, we used this table to generate the number of tile's movement for the index of tile and direction of tile's movement. Below this table (1) [13].

Table (1): Represents the number and direction of tile's move

Index of tile	The number of tile's move	The direction of tile's move Where: R: Right L: Left U: Up D: Down
1	2	D,R
2	3	R,L,D
3	2	D,L
4	3	U,D,R
5	4	U,D,R,L
6	3	U,D,L
7	2	U,R

8	3	U,R,L
9	2	U,L

4-1 Pseudo -code of Ant Colony Algorithm

Below are steps to solve the problem by ant colony algorithm:

- 1- Input the initial state of problem 8-puzzel
- 2- Input the goal of the problem
- 3- Find the numbers of tail difference between the goal and initial state =d and denoted to the number of ants.
- 4- Initialize the matrix of pheromone trails in Fig.(1)by the creation of the matrix called *fer* which consists of 9 row and 4 column for each row describes in general how the different tail from the goal must be moved to which direct (R,L,U,D) by:
 - a- Put 1 according to which direction the next movement of the tail
 - b- Put 0 if the tail does not need to moving
 therefore, for general problem of the matrix should be as follows:

fer =

Tail sequence	L	R	U	D
1	0	1	0	1
2	1	1	0	1
3	1	0	0	1
4	0	1	1	1
5	1	1	1	1
6	1	0	1	1
7	0	1	1	0
8	1	1	1	0
9	1	0	1	0

Fig.(1) Matrix of Initializing the Pheromone Trails

- 5- Generate all possible solutions by taking for each different tail of all the possible movements by using Table(1).
- 6-Consider each solution as a result for the previous step which is a path built by Ant
- 7- Apply the steps of ant algorithm
 - a- Find the Δfer of pheromone for each path built by ant formulated as in (equa.-2) :

$$\Delta fer = 9/n$$
 , where 9 is the number of all tail in problem 8-puzzel
 ,where n is the number of different tails in each solution from the goal
 - b- Update the pheromone by using (equa.-1)

$$fer_{new} = fer_{old} + \Delta fer$$
 , now the new pheromone is created.
 - c- Calculate the evaporation according to (equa.-3)

$$fer_{new} = (1-p) * fer_{old}$$
 , in this paper ,it is assumed that $p=0.2$
 Where $0 < p < 1$
 - d- The path of ant which has higher pheromone is the best solution
 - e- Compare the new solution with the goal
 - f- Compare the numbers of tail difference between the goal and initial state
 if the difference=0
 Stop
 else

go to step 8

8- Repeat the steps from step 5

4-2 Pseudo –Code of Tabu Search Algorithm

Below the steps to solve the problem by tabu search algorithm:

Begin

Notation

S , current solution

S^* , the best -known solution

f^* , the objective function of the problem in this paper represent the number .of difference between current state and the goal state

$N(S)$, the neighborhood of S

It , stopping after number of iterations

t , tabu list length {is the number of iterations for move will be kept tabu

Step1: Initialization:

- Input the initial state of problem 8-puzzel and assume it the current solution.
- Input the Goal of the problem
- Input the stop criterion it
- Set tabu list length $t=3$
- Initialize tabu list is empty:= \emptyset

Step2: Find the numbers of tail difference between the goal and initial state $n:=n$

Step3:Set $S:=S_0$, $S^* = S_0$, $f^*=f(S_0)$

Step4: begin the search i.e Iteration:

While termination criterion not satisfied Do

Begin

Identify Neighborhood set $N(S)$ by the creation of many solutions with used Move operation. The move operation in this paper about 8_puzzel is making swap between each different tail with others of the current

solution S ,so we obtain R solutions as follows : $R = \sum_{i=1}^{n-1} i$

Where R is the number of solutions we will obtain

Where n = number of difference between current solution and the Goal

Step 4.1: Select the best admissible move that transforms s into s' with objective function value $f(s')$ and add its attributes to the tabu list.

If solution s' is not Tabu go to step 4.2

Else go to step 4.3

Step4.2: Perform exchanges:

$s = s'$, $f(s) = f(s')$.

If $f(s) < s^*$ then $f^* = f(s)$, $s^* = s$.

end if

go to step 5

Step 4.3: Check s' is Aspiration

IF Yes go to step 4.2

Else

Check neighbor set.

Step 5 : Record tabu for the current move in tabu list, i.e. update the tabu list by changing the contents of tabu list as follows:

- If solution is not tabu, add solution to tabu list by changing the position of tail to 3 in the next iteration the tabu tenure decrees by one.

End while

Step 6: the solution s^* as the same of the goal state

End

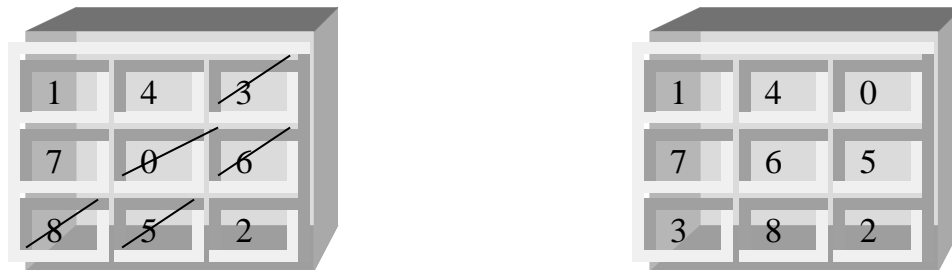
5. Practical Representation

5-1 Practical Representation of Tabu search Method

Below results illustrate that the approach can be applied for solving 8_puzzle problem by tabu search algorithm . In our experiment, input the used initial state and the goal state as array which consist of 1 row and 9 column :

Apply the steps of Tabu search algorithm:

1- Initial state : 1 4 3 7 0 6 8 5 2 → Goal state: 1 4 0 7 6 5 3 8 2



Iteration (1):

2- To generate the neighbor solution by the creation many solutions by using Move operation. The move operation in the 8_puzzel is making swap between any different tail with others of the current solutions ,so we obtain R solutions as in table(2)

In the above example, the number of different tails $n=5$

The tails (3, 5,6,7 and 8)

The number of neighbor solution =10 as in Fig.(2):

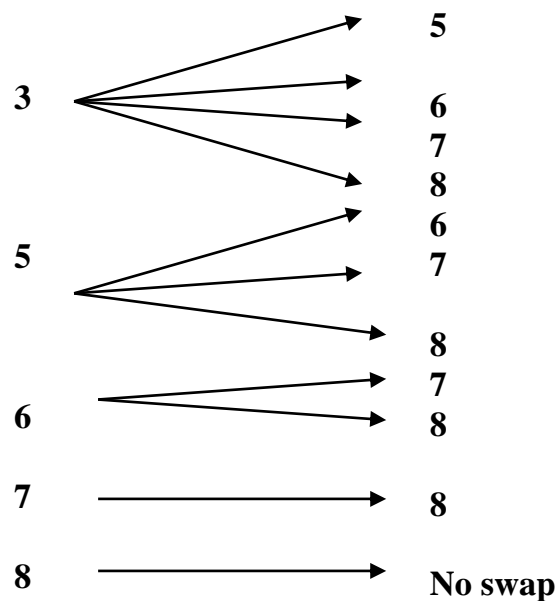


Fig.(2) number of neighbor solution of iteration (1)

Table(2):Generate neighbor solution to create many solutions by using Move operation by iteration (1)

Seq.of neighbor solution	tails	Move with tail	neighbor solution	No.of different tail with the Goal
1	3	5	1 4 0 7 3 6 8 5 2	4
2		6	1 4 6 7 0 3 8 5 2	5
3		7	1 4 8 7 0 6 3 5 2	4
4		8	1 4 5 7 0 6 8 3 2	5
5	5	6	1 4 3 7 6 0 8 5 2	4
6		7	1 4 3 7 8 6 0 5 2	5
7		8	1 4 3 7 5 6 8 0 2	5
8	6	7	1 4 3 7 0 8 6 5 2	5
9		8	1 4 3 7 0 5 8 6 2	4
10	7	8	1 4 3 7 0 6 5 8 2	4

After sorting the solution according to the different tail, there are many solutions which have it has minimum difference, therefore select the one of them, select the solution s which moves from tail 3 to tail 5 with $f(s)=4$ tail difference.

$$3 \rightarrow 5 \quad s = [1 \ 4 \ 0 \ 7 \ 3 \ 6 \ 8 \ 5 \ 2]$$

The (short term memory) or tabu list must be as follows:

Iteration (2):

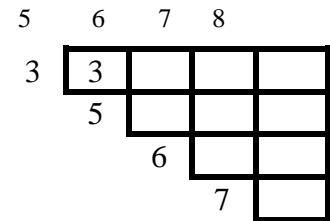
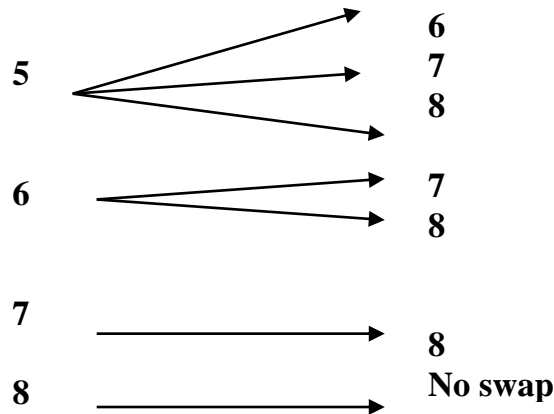


Fig.(3) number of neighbor solution of iteration(2)

Table(3):Generate neighbor solution to create many solutions by using Move operation by iteration(2)

Seq.of neighbor solution	tails	Move with tail	neighbor solution	No.of different tail with the Goal
1	5	6	1 4 0 7 6 3 8 5 2	3
2		7	1 4 0 7 8 6 3 5 2	3

3		8	1 4 0 7 5 6 8 3 2	4
4	6	7	1 4 0 7 3 8 6 5 2	4
5		8	1 4 0 7 3 5 8 6 2	3
6	7	8	1 4 0 7 3 6 5 8 2	3

After sorting the solution according to difference of tail, there are many solutions with minimum difference, therefore select one of them as example, select the solution s which moves from tail 5 to tail 6 with $f(s)=3$ tail difference.

$$5 \rightarrow 6 \quad s = [1 \quad 4 \quad 0 \quad 7 \quad 6 \quad 3 \quad 8 \quad 5 \quad 2 \quad]$$

Now the tabu list must be as follows:

	5	6	7	8
3	2			
	5	3		
		6		
			7	

After iteration(3)

Select the solution s which moves from tail 6 to tail 7 with $f(s)=2$ tail difference.

$$6 \rightarrow 7 \quad s = [1 \quad 4 \quad 0 \quad 7 \quad 6 \quad 8 \quad 3 \quad 5 \quad 2 \quad]$$

The tabu list must be as follows:

	5	6	7	8
3	1			
	5	2		
		6	3	
			7	

At iteration (4), we obtain the best solution S^* which has zero different tail one neighbor solution table (4) :

Table(4):Generate one neighbor solution in iteration(4)

Seq.of neighbor solutio	tails	Move with tail	neighbor solution	No.of different tail with the Goal
1	6	8	1 4 0 7 6 5 3 8 2	0

The tabu list must be as follows:

	5	6	7	8
3	0			
	5	1		
		6	2	3
			7	

5-2 Practical Representation of Ant Colony Method

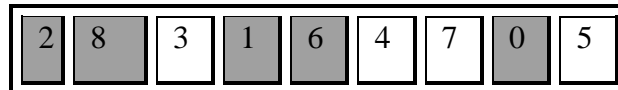
The results below illustrate that the approach can be applied for solving 8_puzzle problem by Ant colony method. It is implemented to simulate the ants behavior .In our experiment ,the inputs used are initial state and the goal state as an array consists of 1 row and 9 column :

e.g:

- 1- Initial state : 2 8 3 1 6 4 7 0 5
- Goal state: 1 2 3 8 0 4 7 6 5

Initialize the matrix of pheromone trails by using Fig.(1)

- 2- Find the differences of tails between initial state and the goal state for example n=5 , where n is the number of difference tails in columns (1,2,4,5and 8) in array.



-input the matrix of initial state

2 8 3 1 6 4 7 0 5

-input the matrix of the goal

1 2 3 8 0 4 7 6 5


- 3- The number of possible solutions depends on tail moves for all misplaced tiles in initial state . See table (5), here we have 15 possible solutions, we have 15 Ant

4-Table (5) describes the solutions(paths) for each ant, the direction, and number of different tail between the path and the goal state ,as well as the pheromone update Δfer last column.

- 5- Update the pheromone value by equ-1,(see Fig.-4).

- 6- Update the matrix of pheromone evaporation by equ-3 where $p=0.2$,(see Fig.-5)

Table (5): Describe the path for each ant, the direction, number of different tail between the path and the goal state, the pheromone update .

 Ants	The direction to Move	The solution after the ant move	The numbers of different	$\Delta fer = 9/n$
Ant1	R	8 2 3 1 6 4 7 0 5	4	2.25
Ant2	D	1 8 3 2 6 4 7 0 5	4	2.25
Ant3	D	2 6 3 1 8 4 7 0 5	5	1.8
Ant4	R	2 3 8 1 6 4 7 0 5	6	1.5
Ant5	L	8 2 3 1 6 4 7 0 5	4	2.25
Ant6	D	2 8 3 7 6 4 1 0 5	6	1.5
Ant7	U	1 8 3 2 6 4 7 0 5	4	2.25
Ant8	R	2 8 3 6 1 4 7 0 5	5	1.8
Ant9	D	2 8 3 1 0 4 7 6 5	3	3
Ant10	U	2 6 3 1 8 4 7 0 5	5	1.8
Ant11	R	2 8 3 1 4 6 7 0 5	6	1.5
Ant12	L	2 8 3 6 1 4 7 0 5	5	1.8
Ant13	U	2 8 3 1 0 4 7 6 5	3	3
Ant14	R	2 8 3 1 6 4 7 5 0	6	1.5
Ant15	L	2 8 3 1 6 4 0 7 5	6	1.5

fer =

Tails sequence	L	R	U	D
1	0	3.25	0	3.25
2	3.25	2.5	0	2.8
3	1	0	0	1
4	0	2.8	3.25	2.5
5	2.8	2.5	2.8	4
6	1	0	1	1
7	0	1	1	0
8	2.5	2.5	4	0
9	1	0	1	0

Fig.(4): Update the pheromone

fer =

Tails sequence	L	R	U	D
1	0	2.6	0	2.6
2	2.6	2	0	2.24
3	1	0	0	1
4	0	2.2	2.6	2
5	2.2	2	2.2	3.2
6	1	0	1	1
7	0	1	1	0
8	2	2	3.2	0
9	1	0	1	0

Fig.(5): The matrix of pheromone after evaporation

7- The Ant puts the maximum pheromone as current solution

In this example, the ant number 9 has maximum pheromone=3.2

Therefore, the current solution=

The path of ant9 the pheromone the Move direction

2 8 3 1 0 4 7 6 5 3.2 5 →D

8-Repeat from step 2 until n=0

The optimal solution was obtained when tail 1 moves to Down:

1 2 3 8 0 4 7 6 5 1→D

6. Conclusions

The 8-puzzle problem can be solved by many general heuristics algorithms in artificial intelligence ,so in this paper, it is solved by two heuristics algorithms for combinatorial optimization problems , tabu search and simulating the behavior of real ants through mathematical formulations by using ant colony algorithm.

To solve the 8-puzzle problem in the beginning, we must find the number of different tails between the initial state and the goal state, this is considered similarly in the two algorithms of this paper.

The difference between the used tabu and ant colony algorithms is the way the trial solutions are generated. In ant colony algorithm trial solutions are constructed incrementally, based on created solutions from all possible moves for the different tail to the suitable direction i.e. to the (Right, Left, Up, Down), and consider each result of solution is a path of ant and then apply the steps of ant colony algorithm.

On the other hand, in tabu search the trial solutions are use move (swap) operation, with relationships between the different tails that were obtained from the beginning of solution. The relationships are used to construct the neighborhoods and then choose the solution that has minimum number of different tails, then continue with the steps of tabu algorithm, this is considered efficient and creative good solution.

Tabu search and ant colony optimization perform better than any local optimization method and yield a solution close to global optimum. Despite the classical method (manual method) can give the exact solution, it takes a large space and need more computations. The two algorithms perform significantly better than the traditional search methods. This is dependent on the number of different tail as shown in table (6).

In tabu search the number of neighborhood obtained by using the rule: $R = \sum_{i=1}^{n-1} i$

But in ACO algorithm the number of ant path obtained by depending on the different tail move for any suitable direction.

Table (7) explains some features used by tabu search and ant colony algorithms in this paper to solve 8-puzzle problem.

Table (6) : Explain the number of solutions that can be obtained by the algorithm used in this paper

Number of tails different	Number of Neighborhood In tabu search algorithm	Number of path In Ant colony algorithm
2	1	5 or 7
3	3	8 or 9
4	6	10 or 12
5	10	15 or 12
6	15	15 or 17
7	21	17 or 19 or 20
8	28	20 or 21 or 22

Table (7): Explain some features that used by tabu search and Ant colony algorithms

Tabu search algorithm	Ant colony algorithm
Number of Neighborhood	Number of ants
The process is used Move (swap)	The process is update the pheromone
Wider search space is guaranteed by Tabu list	Wider search space is guaranteed by pheromone evaporation

REFERENCES

- [1] Alkallak, I. N. and Sha'ban, R. Z., 2008, Tabu Search Method for Solving the Traveling Salesman Problem, *Al-Rafiden Journal of Computer Sciences and Mathematics*, Vol. 5, No. 2, pp.141-153.
- [2] Alkallak, I. N. and Sha'ban, R., 2011, A Survey of Two Optimization Methods to Solve A Modified Minimal Spanning Tree Problem in Undirected Tree Graph, *Al-Rafidain Journal of Computer Sciences and Mathematics*, Vol. 8, No. 1, pp.117-134.
- [3] Alkallak, I. N., 2009, A Hybrid Ant Colony Optimization Algorithm to Solve Assignment Problem by Hungarian Method, *Al-Rafiden Journal of Computer Sciences and Mathematics*, Vol. 6, No. 2, pp.159-175.
- [4] Dorigo, M., and Di Caro, G., 1999, *The Ant Colony Optimization Metaheuristic*, McGraw-Hill, pp. 11-32.
- [5] Dorigo, M., Gambardella, L., 1999, Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem, *IEEE Transactions on Evolutionary Computation*, 1, pp. 53-66.
- [6] Dorigo, M.; Maniezzo, V. and Colorni, A., 1991, Positive Feedback as a Search Strategy. Tech. Report 91-016, Dipartimento di Elettronica, Politecnico di Milano, Italy.
- [7] Fidanova, S., 2005, Heuristics for Multiple Knapsack Problem, *IADIS International Conference on Applied Computing*, pp.255-260.
- [8] Hingston,P.,and Kendall,G.,2004, Ant Colonies Discover Knight's Tours, Edith Cowan University, Australia, The University of Nottingham,UK,LNA13339, pp.1213-1218.
- [9] Misevicius, A.; Blazauskas, T.; Blonskis, J. and Smolinskas, J., 2004, An Overview of Some Heuristic Algorithms for Combinatorial Optimization Problems, *Informacines Technologijos Ir Valdymas*, Nr. 1(3), pp.21-31.
- [10] Qian, T., 1995, Using Genetic Algorithm to Solve Sliding Tile Puzzles, *Cognitive Science Problem*, Oswego,USA.
- [11] Reinefeld, A., 2006, Complete Solution of the Eight-Puzzle and the Benefit of Node Ordering in IDA*, Paderborn Center for Parallel Computing, Germany.
- [12] Rich, E., 1988, *Artificial Intelligence*, McGraw-Hill, Inc., Eight Edit, Singapore.
- [13] Sha'ban, R. Z, Alkallak, I. N., Sulaiman, M. M.,2010, Genetic Algorithm to Solve Sliding Tile 8-Puzzle Problem, *Journal of Education and Science*, Vol 23, No. 3, pp. 145-157.
- [14] Solimanpur, M., Vrat, P. and Shankar, R., 2004, Ant Colony Optimization Algorithm to the Inter-Cell Layout Problem in Cellular Manufacturing, *European Journal of Operational Research* 157, pp.592-606.

www.elsevier.com/locate/dsw