



An Optimum Solution to 8- Puzzle Problem using Particle Swarm Optimization

¹Parul Saxena, ² Ashish Mehta

¹Assistant Professor and Campus Head, Department of Computer Science,
Kumaun University, S.S.J. Campus, Almora, 263601, Uttarakhand, INDIA.

²Assistant Professor and Head, Department of Computer Science,
Kumaun University, D.S.B. Campus, Nainital, Uttarakhand. INDIA.

Abstract: The 8-puzzle problem is one of the most classical problems in artificial intelligence. In this problem starting from current state we reach to the goal state through state space search procedure which depends on the number and position of misplaced tiles in the current state as compared with the goal state. Particle Swarm Optimization (PSO) technique has been developed under the inspiration of behavior laws of bird flocks, fish schools and human communities. Compared with the genetic algorithm, PSO algorithm has high convergence speed. In this paper, we have applied the Particle swarm optimization technique to find the solution to 8 puzzle problem.

Index Terms: 8-Puzzle Problem, Particle Swarm Optimization, State Space Search

I. INTRODUCTION

The 8 puzzle is a game invented by Sam Loyd. It is a square tray in which there 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. [1]. One cell of the frame is always empty thus making it possible to move an adjacent numbered tile into the empty cell. Such a puzzle is illustrated in figure 1.

A move transforms one problem state into another state. The 8-puzzle is conveniently interpreted by following the moves. Move empty space (blank) to the left, to upward, to downward and to the right. These moves are modeled by production rules that operate on the state descriptions in the appropriate manner.

The program is to change the initial configuration into the goal configuration. A solution to the problem is an appropriate sequence of moves, such as “move tile 5 to the right”, “move tile 7 to the left” and “move tile 6 to the down”, etc”.

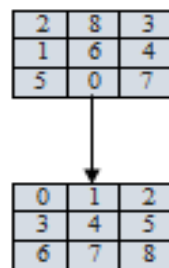


Figure 1: Initial State to Goal State

To solve a problem using a production system, we must specify the global database the rules, and the control strategy. These elements are the problem states, moves and goal. In this problem each tile configuration is a state. Once the problem states have been conceptually identified, we must construct a computer representation, or description of them. this description is then used as the database of a production system. For the 8-puzzle, a straight forward description is a 3X3 array of matrix of numbers. The initial global database is this description of the initial problem state. Virtually any kind of data structure can be used to describe states.

The problem goal condition forms the basis for the termination condition of the production system. The control strategy repeatedly applies rules to state descriptions until a description of a goal state is produced. It also keeps track of rules that have been applied so that it can compose them into sequence representing the problem solution.

II. PARTICLE SWARM OPTIMIZATION

PSO is a robust stochastic optimization technique based on the movement and intelligence of swarms. PSO applies the concept of social interaction to problem solving. Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995[2], inspired by

social behavior of bird flocking or fish schooling. PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA).

The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. Compared to GA, the advantages of PSO are that PSO is easy to implement and there are few parameters to adjust. PSO has been successfully applied in many areas: function optimization, artificial neural network training, fuzzy system control, and other areas where GA can be applied. [3]

In PSO, each single solution is a "bird" in the search space. We call it "particle". All of particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. The particles fly through the problem space by following the current optimum particles. PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called *pbest*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called *gbest*. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called *lbest*.

III. PSO ALGORITHM

In a PSO system, a swarm of individuals (called particles) fly through the search space. Each particle represents a candidate solution to the optimization problem. The position of a particle is influenced by the best position visited by itself (i.e. its own experience) and the position of the best particle in its neighborhood (i.e. the experience of neighboring particles). When the neighborhood of a particle is the entire swarm, the best position in the neighborhood is referred to as the global best particle, and the resulting algorithm is referred to as a *gbest* PSO. When smaller neighborhoods are used, the algorithm is generally referred to as a *lbest* PSO [4][5]. The performance of each particle (i.e. how close the particle is from the global optimum) is measured using a fitness function that varies depending on the optimization problem. The Pseudo code of basic PSO algorithm is as follows [6]:

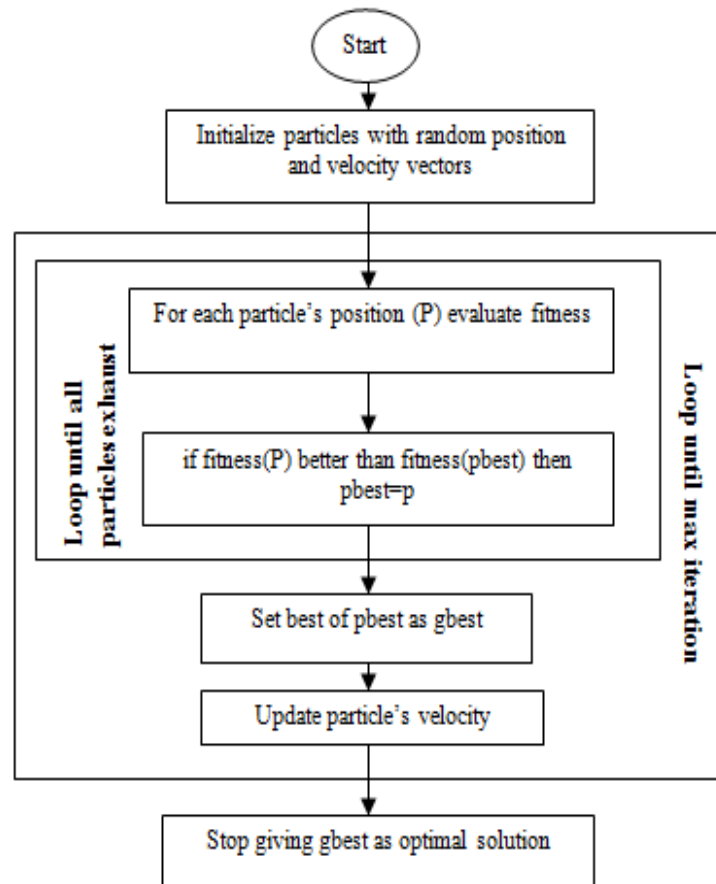


Figure 2: PSO Algorithm

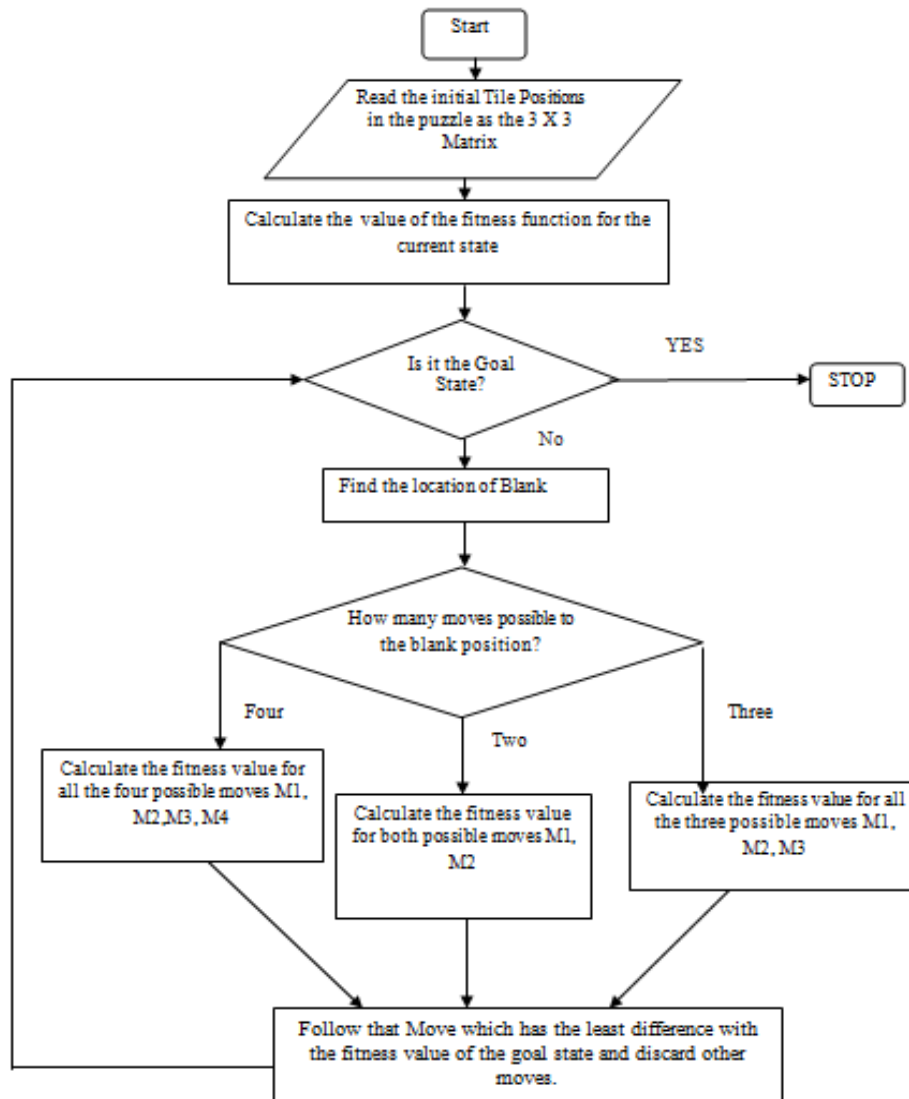


Figure 3: Flow Chart for the solution of 8 Puzzle Problem through PSO

IV. APPROACH TO IMPLEMENT PSO FOR 8 PUZZLE PROBLEM

The different tile positions can be treated as the particles in the swarm after following the appropriate move, which is found by calculating the fitness value of all possible moves in every state. Now we are entering in the next and local best state which is having the closest optimum value to the fitness value of the Goal state, as soon as the goal state is achieved or the maximum number of iterations is complete, we stop the process.

If the goal is achieved we store the complete path through which we have reached to the goal state, and this path is the optimum solution for the 8 puzzle problem.

V. ALGORITHM FOR 8 PUZZLE PROBLEM USING PSO

1. Start.
2. iteration=0;
2. Read the initial Tile Positions in the puzzle as the 3 X 3 Matrix.
3. Calculate the value of the fitness function for the current state.
4. If it the Goal State OR (iterations>=100) STOP.
else iteration ++
5. Find the location of Blank.
6. Find the number of possible moves to the blank position.
7. If it is centre most position than the number of possible moves are four, Calculate the fitness value for all the four possible moves M1, M2,M3, M4.

else if it is corner most position than the number of mover are two, Calculate the fitness value for both possible moves M1, M2

else if it is boundary centre position than the number of moves are to three, Calculate the fitness value for all the three possible moves M1, M2, M3.

8. Follow the path of that Move which has the least difference with the fitness value of the goal state and discard other moves.

9. go to step 4.

V. FITNESS FUNCTION

If Puzzle [i][j] is a 3X3 Matrix then the fitness value of the current position is calculate by

$$\text{Fitness Value} = \sum_{i,j=0,1,2} (3 * i + j) * \text{Puzzle}[i][j]$$

For the goal state Fitnessvalue=

$$(3*0+0)*0+(3*0+1)*1+(3*0+2)*2+(3*1+0)*3+(3*1+1)*4+(3*1+2)*5+(3*2+0)*6+(3*2+1)*7+(3*2+2)*8 \\ = 0+1+4+9+16+25+36+49+64 = 204$$

which is Maximum hence we have to maximize the fitness function, for all other situations it is lower than 204.

VI. RESULTS AND DISCUSSION

The above algorithm has been developed in c programming language, The algorithm starts with the initial position of the tiles i.e. initial position of the particles. The tile value 0 represents the position of the blank space.

Fig. 4: Output of Algorithm for 8 Puzzle Problem having 3 iterations.

First the fitness value of the initial position is calculated if it is less than the fitness value of the goal state than the position is changed consequently i.e. the tiles are moved in such a way that they reaches closer to the goal state in every step. When the fitness value of the puzzle state is found equal to the fitness value of the goal state then we stop the process and hence we achieve the complete path through which we have reached to the solution.

VII. CONCLUSION

We can find out the optimum path of 8 Puzzle Problem through hybrid Particle Swarm Optimization by using proper fitness function. In future we want to find out the solution to those puzzle states which are infeasible to reach to the goal state as all initial board positions can not lead to the goal state. So we have to modify our program to detect and report such situations, also we shall implement it to N-Puzzle Problem in near future.

References

- [1]. Ruqaya Zedan Sha'ban Isra Natheer Alkallak J, "Genetic Algorithm to Solve Sliding Tile 8-Puzzle Problem", J. Edu.&Sci., Vol. (23), No. (3) 2010 K.
- [2]. J. Kennedy and R. Eberhart. Particle Swarm Optimization. In *Proceedings of IEEE International Conference on Neural Networks*, Perth, Australia, vol. 4, pp. 1942-1948, 1995.
- [3]. J. Kennedy and R. Eberhart. *Swarm Intelligence*. Morgan Kaufmann, 2001.
- [4]. Wen-Liang Zhong, Jun Zhang, Weineng Chen, "A Novel Discrete Particle Swarm Optimization to solve Travelling Salesman Problem", 2007 IEEE Congress on Evolutionary Computation.
- [5]. R. Eberhart and Y. Shi, "Comparison between Genetic Algorithms and Particle Swarm Optimization", *Proceedings of the Seventh Annual Conference on Evolutionary Programming*, pp. 611-619. Springer-Verlag, 1998.
- [6]. Mahamed G. H. Omran "Particle Swarm Optimization Methods for Pattern Recognition and Image Processing" thesis Submitted in partial fulfillment of the requirements for the degree Philosophiae
- [7]. Doctor in the Faculty of Engineering, University of Pretoria Pretoria November 2004.