International Journal of Engineering, Mathematical and Physical Sciences
World Academy of Science, Engineering and Technology
ISSN: 2517-9934
Vol:4 2010-07-24
Vol:4, No:7, 2010

A Hybrid Approach Using Particle Swarm Optimization and Simulated Annealing for N-queen Problem

Vahid Mohammadi Saffarzadeh, Pourya Jafarzadeh, Masoud Mazloom

***Abstract*—**This paper presents a hybrid approach for solving n-queen problem by combination of PSO and SA. PSO is a population based heuristic method that sometimes traps in local maximum. To solve this problem we can use SA. Although SA suffer from many iterations and long time convergence for solving some problems, By good adjusting initial parameters such as temperature and the length of temperature stages SA guarantees convergence. In this article we use discrete PSO (due to nature of n-queen problem) to achieve a good local maximum. Then we use SA to escape from local maximum. The experimental results show that our hybrid method in comparison of SA method converges to result faster, especially for high dimensions n-queen problems.

***Keywords*—**PSO, SA, N-queen, CSP

I. Introduction

A CSP is defined by a set of variables and a set of constraints. Variable has a nonempty domain of possible values. Each constraint involves some subset of the ariables and specifies the allowable combination of values for ıe subset. A state of the problem is defined by an assignment f values to some or all variables. A complete assignment is ne in which every variable is mentioned, and solution to CSP s a complete assignment that satisfies all the constraints. Some SPs also require a solution that maximize an objective unction [1].

The n-queens problem is a CSP that consists of placing n ueens on an N by N chess board, so that they do not attack ach other, i.e. on every row, column or diagonal, there is only ne queen, Its complexity is O(n!) [2][3]. There are many euristics for solving n-queen problem, some of these euristics cooperate better with some search methods than the thers. The complexity of heuristic used in this paper is O(n). here are several search strategies for n-queen problem such as epth First Search, Beam Search, Branch and Bound, local earch methods and Evolutionary algorithms (EA).

The sections of the papers are as follows: Section II reviews ıe basic and discrete forms of PSO. Section III reviews the SA lgorithms. In section IV we described our hybrid algorithm, nd section V summarizes experimental results. Finally, conclusions and future works are presented in VI.

II. Particle Swarm Optimiztion

A. Basic PSO

The Particle Swarm Optimization (PSO) [Kennedy and Eberhart, 1995, Eberhart et al.,2001] evolved from an analogy drawn with the collective behavior of the animal displacements [4].

The system is initialized with a population of random solutions and searches for optima by updating potential solution over generation. In PSO, the potential solutions, called particles, "fly" through the problem space by following the current better performing particle [2]. The solution updating can be represented by the concept of velocity [5]. By definition, a velocity is a vector or, more precisely, an operator, which, applied to a position (solution), will give another position (solution). It is in fact a displacement, called velocity because the time increment of the iteration is always implicitly regarded as equal to 1 [6]. Velocity of each particle can be modified by the following equation:

$$v_i^{k+1} = wv_i^k + c_1 rand_1 \times \left(pbest_i - s_i^k\right) + c_2 rand_2 \times \left(nbest_i - s_i^k\right) \quad (1)$$

Where v_k^i is velocity of particle i at iteration k, w is weighting function, c_j is weighting coefficients, rand is random number between 0 and 1, s_i^k is the current position of particle i at iteration k, $pbest_i$ is the best state of particle i and $nbest_i$ is the best state among the neighbors of particle i (until iteration k) [5]. A general flowchart of basic PSO is shown in figure 1.

B. Disceret PSO

The basic PSO treats nonlinear optimization problem with continuous variables. However, practical engineering problems are often formulated as combinatorial optimization problems. Kennedy and Eberhart developed a discrete binary version of PSO for these problems. They proposed a model wherein the probability of an agent's (particle) deciding yes or no, true or false and 0 or 1 by the following factors:

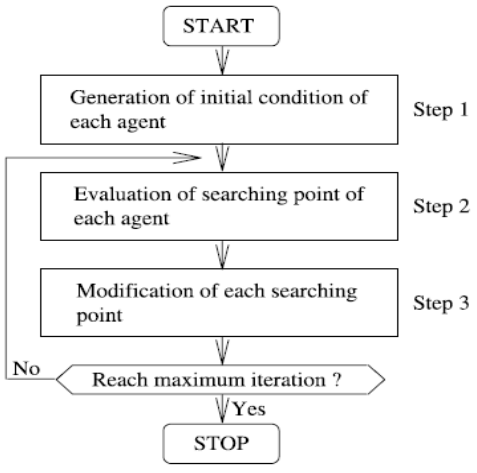

Fig. 1. A general flowchart of PSO [5]. Agent is the same particle.

$$P(s_i^{k+1} = 1) = f(s_i^k, v_i^k, pbest_i, nbest_i) \quad (2)$$

The parameter v, a particle's tendency to make one or the other choice, will determine a probability threshold. If v is higher, the particle is more likely to choose 1, and lower values favor 0 choice. Such a threshold requires staying in the range [0, 1]. The proper function for this feature is the sigmoid function:

$$sig(v_i^k) = \frac{1}{1+\exp(-v_i^k)} \quad (3)$$

Like the basic continues version, the formula for the binary (discrete) version of PSO can be describe as follows [5]:

$$v_i^{k+1} = v_i^k + rand_1 \times (pbest_i - s_i^k) + rand_2 \times (nbest_i - s_i^k) \quad (4)$$

$$\rho < sig(v_i^{k+1}) \quad then \quad s_i^{k+1} = 1;$$

$$else \ s_i^{k+1} = 0; \quad (5)$$

Where rand and ρ are random numbers in $[0,1]$. The entire algorithm of the binary version of PSO is almost the same as that of the basic continous version (figure 1) except for the above equations[5].

III. SIMULATED ANNEALING (SA)

In statistical mechanics, a physical process called annealing s often perform in order to relax the system to a state with inimum free energy [8]. The idea to use annealing technique n order to deal with optimization problems gave rise to the imulated annealing technique. It consists in introducing a ontrol parameter in optimization, which plays the rule of emperature. The "temperature" of the system to be optimized ost have the same effect as the temperature of the physical ystem: it must condition the number of accessible states and ead to the optimal state, if the temperature is lowered radually in a slow and well controlled manner (as in the nnealing technique) and towards a local minimum if the emperature is lowered abruptly. The flowchart of the lgorithm is shown in figure 2 [9].

If $\Delta E \leq 0$ (i.e. new state is better than the current state) the odification will be accepted, when $\Delta E > 0$ if the probability $\exp(-\Delta E/T)$ is greater than a random number drawn niformly between 0 and 1, the modification, making the state orse, also will be accepted (Metropolis rule [9]).

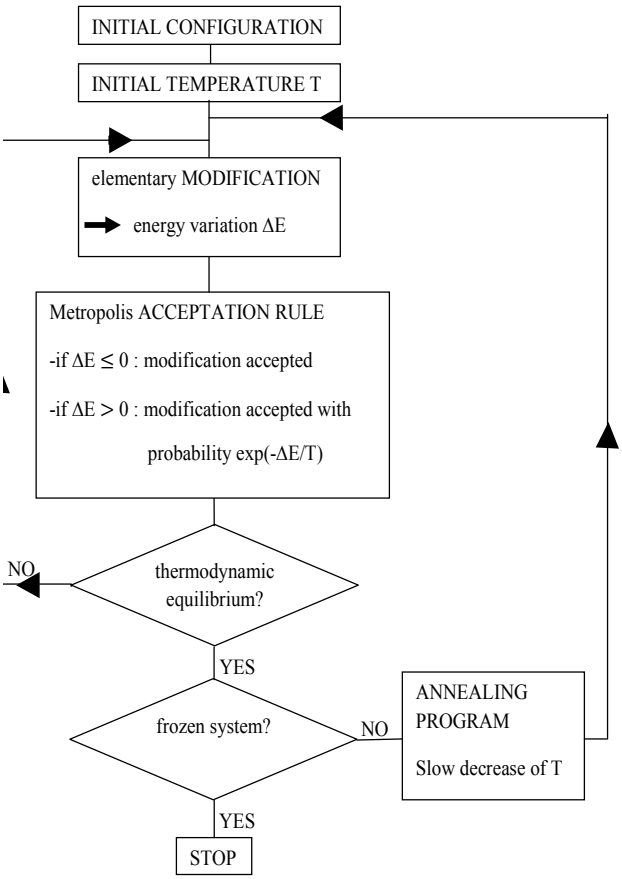

Fig. 2. Flowchart of SA [10].

By repeatedly observing the rule of acceptance, described above, a sequence of configurations (states) is generated, which constitutes a Markov chain (in a sense that each configuration depends on only that one which immediately precedes it) [9].

IV. THE HYBRID ALGORITHM

According to the diagram of our method (figure 3), PSO and SA algorithms were implemented consecutively. The evaluation functions of two algorithms are the same. Diagram shows that PSO works with four particles, then SA takes the state of best particle from PSO and after generating the initial temperature, performs sufficient iterations (Markov chains) toward a global maximum. In the following paragraphs the important parameters and steps of algorithm are represented.

A. Representaiton of problem states

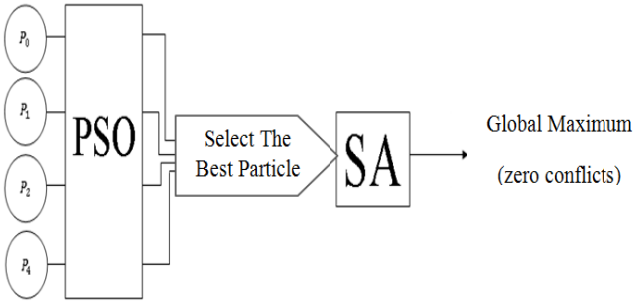

Fig. 3. The operation schema of the algorithm.

For adaptation between the problem and discrete PSO, states were represented by binary matrices. Notice that this representation is used only when we want to update the state according to formulas (4) and (5), whereas for other parts of algorithm these states representation will be transformed to a one dimensional, due to a function that has a complexity of . In matrix (that is a matrix for chess board) each column shows the binary representation of row number that is the position of queen on that column in chess board. For example the 4 4 chess board in figure (4.a) will be represented like 2 4 matrix shown in figure (4.b), used in updating time, and the state shown in figure (4.c), the transformed schema of figure (4.a), used in evaluation rocedure.

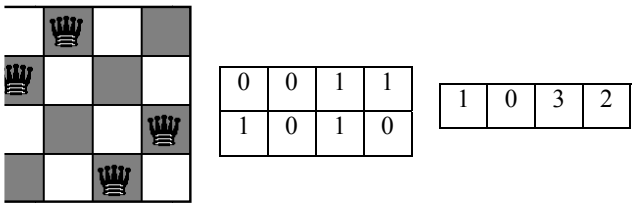

Fig. 4 Queens pos ition representation.

Number of particles

The number of particles in PSO is low because of:

1. We don't want to solve the problem by PSO, we just want to achieve a good local maximum (low number of conflicts) for the next step of algorithm (SA).
2. If the number of particles is high, more iterations will be required to achieve the local maximum.

Markov chains

According to figure 2 the modifications of the states in each temperature stage should be sufficient to receive the thermal equilibrium (i.e. no move can decrease the evaluation of the state) in that temperature. After the thermal equilibrium the temperature decreases by a coefficient (e.g. 0.99). The number of temperature stages (Markov chains) is a function of dimension of the problem, and the number of moves in each stage (Markov length) is multiple of problem's dimension and the stage index (figure 4).

```
N = dimension ( number of Markov chain);
for n=0…N×N/2{
 for m=0…(n+1)×N {//Markov chain length
    Modify state to neighbor configuration;
    …..
 }
 T = 0.99 × T;
                    }
```

Fig. 5. Pseudo code of using Markov chain in SA step

The initial temperature, after the PSO step and in the beginning of SA step, is generated by using the evaluation of the best state, the formula is as follow [11]:

$$T=(0.001)\times(\text{bestEvaluation})/(-\log(0.15)) \quad (6)$$

The coefficients are appropriate to avoid from the initial divergence of SA, i.e. the state won't go to very bad situations.

D. Determining the local maximum of PSO

The algorithm traps in local maximum in PSO step when the evaluation of , and state are equal for each particle.

E. Steps of algorithm

The algorithm consists of two parts (PSO & SA), and total steps are as follows:

1. Generate four particles with random initial states and velocities;
2. Perform the updating procedures (figure 6) of each particle;
3. If the local maximum is not happened, go to step 2; otherwise go to step 4;
4. Select the particle with best state;
5. Start SA with the selection result of the previous step;
6. Generate the initial temperature (with (6);
7. Make modification on the state, accept that according to the acceptation rule (Metropolis rule);
8. If the evaluation is zero it means that the algorithm has achieved to the final state, then go to 11, otherwise if the Markov length for the current temperature stage is not finished go to step 7 if not, go to step 9;
9. Decrease the temperature;
10. If Markov chains loop is finished, go to step 11, else go to 7;
11. End of procedure.

```
for i = 0 ... dimension {
  for j = 0 ... log (dimension){
      velocity[i][j] = velocity[i][j]
        + rand1 × (pbest[i][j] − state[i][j])
        + rand2 × (nbest[i][j] − state[i][j]);

      state[i][j] = sig(velocity[i][j])
                  > Math.random() ? 1 : 0;
    }
  }
if evaluation ≤ pbestevaluation {
    copy state array to pbest array;
    pbestevaluation = evaluation;
}

Update nbest array and evaluation
 of each particle;
```

Fig. 6. Pseudo code of the updating procedure.

V. Experimental results

The algorithm is implemented in Java language and tested by chess boards with different dimensions. The machine used was Intel Core 2 Duo CPU 2.40 GHz, 2GB of memory, running Windows 7 Ultimate and jdk1.7.0. The results are shown in tables 1 and 2 and figures 7 and 8.

Table 1 shows the number of iterations required to achieve a good local maximum in PSO step.

TABLE I THE NUMBER OF ITERATIONS THAT PSO PERFORMS TO TRAP IN LOCAL MAXIMUM (THE RESULTS ARE THE AVERAGE OF 10 RUNNING OF ALGORITHM)

Dimension	20	200	500	700	1000
PSO iteration	353	2489	5784	8078	11189

Following table is a comparison of the hybrid approach and SA method, for solving n-queen problem for 5 different dimensions (number of queens), according to their running times for finding solution. Afterward you can see diagrams of this comparison that are obtained from running algorithms with more different dimensions.

TABLE II COMPARISON OF SA AND HYBRID ALGORITHM FOR 5 DIFFERENT DIMENSION OF CHESS BOARDS, ACCORDING THE DURATION OF RUNTIMES IN MILISECOND

Dim / Algorithm	20	200	500	700	1000
Hybrid PSO&SA	57.6	5545.7	53601.6	131050.6	345855.3
SA	3.1	1911	63451.5	203503.9	776577.8

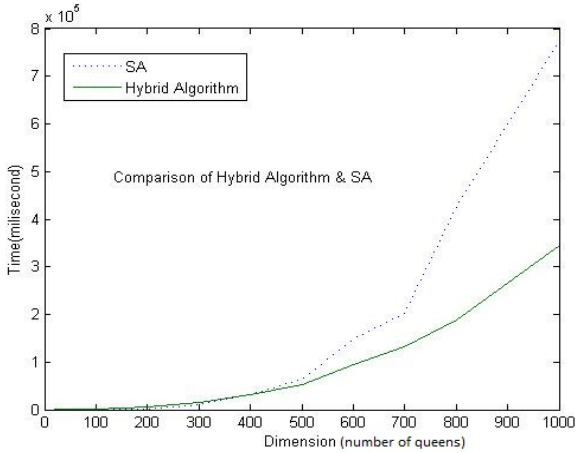

Fig. 7. Comparison of hybrid algorithm and SA run times for solving different dimension n-queen problems.

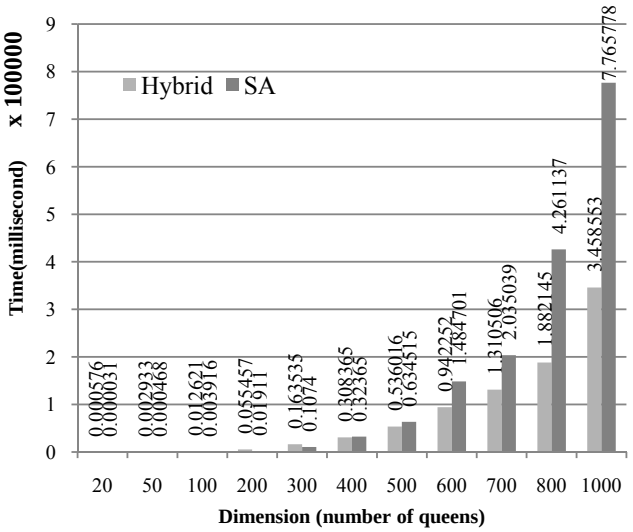

Fig. 8. Column chart of the diagram in figure 7.

Another result is that the majority of time consumes when the state receives to low conflicts (in range 10 to 1) situations (that is because of the nature of SA algorithm).

VI. Concolusion

N-queen is a good NP-complete problem to test the new heuristic algorithms and compare them with old ones. The results of this paper show that the combination of PSO and SA has a capacity to improve the performance of each one in solving this problem, separately. The results also show that n-queen problem can be solved in a reasonable time by this hybridization and this idea is better than SA (the same SA used in the hybrid approach) by an increasingly ratio for higher dimensions, that means when dimension becomes larger the hybrid algorithm receives to global maximum faster.

The hybrid algorithm is capable to be implemented in parallel approach, with parallel PSO it is possible to increase the number of particles and improve reliability of the algorithm. In another parallelism approach if we can give

more states of PSO (best states) to SA in parallel it is possible to achieve global maximum with a better speed. We think that this algorithm also can be useful in solving other CSPs (e.g. Graph Coloring, Time Scheduling, etc).

Described algorithm has many parameters (e.g. number of particles, number of PSO iterations, initial temperature, length of temperature stages, decreasing rate of temperature, etc) to be adjusted. Here, these parameters are determined by statistical tests, this work (adjusting parameters) can be done more precisely by neural networks.

REFERENCES

[1] Stuart Russell, Peter Norvig, "Artificial Inteligence: A Modern Approach," Constraint Satisfaction Problems, 2nd ed., Pearson Education, Inc, Upper Saddle River, New Jersey, 2003,1995, page: 137.

[2] Xiaohui Hu, Russell C. Eberhart, Yuhui Shi, "Swarm Inteligence for Permutation Optimization: A case Study of n-Queen Problem".

[3] Marko Božikovic, Marin Golub, Leo Budin, "Solving n-Queen problem using global parallel genetic algorithm".

[4] J. Dr´eo, A. P´etrowski, P.Siarry, E.Taillard, "Metaheuristics for Hard Optimization," Some Other Metaheuristics, Springer-Verlag Berlin Heidelberg 2006, pp. 162-166.

[5] Kwang Y.Lee, Mohamed Al-Sharkawi, "Modern Heuristic Optimization Techniques: Theory And Application To Power Systems," Fundamentals of Particle Swarm Optimization Techniques, Willey-Interscience, Hoboken, 2008, pp. 72-79.

[6] Maurice Clerc, "Particle Swarm Optimization," First Formulations, ISTE, United States, 2006, page: 39.

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[8] Kwang Y.Lee, Mohamed Al-Sharkawi, "Modern Heuristic Optimization Techniques: Theory And Application To Power Systems," Preface, Willey-Interscience, Hoboken, 2008, page: xxiv.

[9] J. Dr´eo, A. P´etrowski, P.Siarry, E.Taillard, "Metaheuristics for Hard Optimization," Simulated Annealing, Springer-Verlag Berlin Heidelberg 2006, pp. 25-31.

[10] J. Dr´eo, A. P´etrowski, P.Siarry, E.Taillard, "Metaheuristics for Hard Optimization," Introduction, Springer-Verlag Berlin Heidelberg 2006, page: 8.

[11] Kwang Y.Lee, Mohamed Al-Sharkawi, "Modern Heuristic Optimization Techniques: Theory And Application To Power Systems," Fundamentals of Simulated Annealing, Willey-Interscience, Hoboken, 2008, page(s): 128 and 129.