International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:6, No:8, 2012

Combined Simulated Annealing and Genetic Algorithm to Solve Optimization Problems

Younis R. Elhaddad

***Abstract*—**Combinatorial optimization problems arise in many scientific and practical applications. Therefore many researchers try to find or improve different methods to solve these problems with high quality results and in less time. Genetic Algorithm (GA) and Simulated Annealing (SA) have been used to solve optimization problems. Both GA and SA search a solution space throughout a sequence of iterative states. However, there are also significant differences between them. The GA mechanism is parallel on a set of solutions and exchanges information using the crossover operation. SA works on a single solution at a time. In this work SA and GA are combined using new technique in order to overcome the disadvantages' of both algorithms.

***Keywords*—**Genetic Algorithm, Optimization problems, Simulated Annealing, Traveling Salesman Problem

I. Introduction

GENETIC ALGORITHM (GA) and Simulated Annealing (SA) have been used to solve optimization problems. Both GA and SA search a solution space throughout a sequence of iterative states. However, there are also significant differences between them. The GA mechanism is parallel on a set of solutions and exchanges information using the crossover operation. SA works on a single solution at a time. GA uses the same selection strategy during the run of the algorithm, while SA regulates the temperature parameter it uses to evaluate the solution. These differences lead to different search criteria. Both GA and SA have advantages and disadvantages. A disadvantage of GA is that it might trap at local minima, or it is time-consuming to find an optimal solution [1]. In case of SA only one candidate solution is used, thus it does not build up an overall view of the search space. Also, SA is slow because of its sequential nature. In other words, SA can find good quality solutions in a neighborhood, but most likely it will get trapped in local minima and takes longer to escape, while GA rapidly discovers the search space, but has difficulty finding the exact minima [2] [3]. The hybridization of SA and GA tries to combine the advantages of GA. In this work GA and SA are combined in order to improve the quality of solutions and reduce execution time.

A. Simulated Annealing

The purpose of physical annealing is to accomplish a low energy state of a solid. This is achieved by melting the solid in a heat bath and gradually lowering the temperature in order to allow the particles of the solid to rearrange themselves in a crystalline lattice structure. This structure corresponds to a minimum energy state for the solid. The initial temperature of the annealing process is the point at which all particles of the solid are randomly arranged within the heat bath. At each temperature, the solid must reach what is known as thermal equilibrium before the cooling can continue [4].

Younis Elhaddad is with University of Benghazi, Libya e-mail:yrh2010 @yahoo.com

If the temperature is reduced before thermal equilibrium is achieved, a defect will be frozen into the lattice structure and the resulting crystal will not correspond to a minimum energy state.

The Metropolis Monte Carlo simulation [4] can be used to simulate the annealing method at a fixed temperature T. The Metropolis method randomly generates a sequence of states for the solid at the given temperature. A solid's state is characterized by the positions of its particles. A new state is generated by small movements of randomly chosen particles. The change in energy ΔE caused by the move is calculated and acceptance or rejection of the new state as the next state in the sequence is determined according to Metropolis acceptance condition [4]. If $\Delta E < 0$ the move is acceptable and if $\Delta E > 0$ the move is acceptable with probability, if $e^{-\frac{-\Delta E}{t}} > \Omega$.

The move is acceptable otherwise rejected, where Ω is random number and $0 < \Omega < 1$. Simulated annealing was first introduced by Metropolis et al [5], but it was Kirkpatrick et al [6], in 1983 who proposed SA as the basis of an optimization technique for combinatorial optimization problems. Simulated annealing is one of the most popular and general adaptive heuristic algorithms [7]. Simulated annealing algorithms have been applied to solve numerous combinatorial optimization problems. The name and idea of SA comes from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. The heat frees the atoms to move from their initial positions (initial energy). By slowly cooling the atoms the material continuously rearranges, moving toward a lower energy level. They gradually lose mobility due to the cooling, and as the temperature is reduced the atoms tend to crystallize into a solid. In the simulated annealing method, each solution s in the search space is equivalent to a state of a physical system and the function $f(s)$ to be minimized is equivalent to the internal energy of that state. The objective is to minimize the internal energy as much as possible. For successful annealing it is important to use a good annealing schedule, where the temperature reducing gradually.

B. Annealing schedule

The annealing schedule is a major step of SA because it controls the uphill movement of the algorithm. In order to use simulated annealing, the annealing schedule must be at a proper setting. The initial temperature is set at a high value and then it is decided how to decrease the temperature gradually as a function of time. If $T(0)$ is the initial temperature, a simple schedule is $T(k+1) = \beta T(k)$, for some fixed β, such that $0 < \beta < 1$. A cooling schedule depends on the problem to be solved.

Choosing an annealing schedule for a given problem is still a problem for most researchers. Therefore the annealing schedule can be determined by experiments or using heuristics methods to find the optimal annealing parameters.

C. Description of SA

As shown in figure (1) SA starts from a random solution x_p , selects a neighboring solution x_n and computes the difference in the objective function values, $\Delta f = f(x_n) - f(x_p)$. If the objective function is better ($\Delta f < 0$), then the present solution x_p is replaced by the new one x_n .

Otherwise the solution that decreases the value of the objective function with a probability $pr = 1/(1 + e^{-\frac{\Delta E}{t}})$ is accepted, where pr is decreased as the algorithm progresses, and where (t) is the temperature or control parameter. This acceptance is achieved by generating a random number (rn) where $(0 \leq rn \leq 1)$ and comparing it against the threshold. If $pr > rn$ then the current solution is replaced by the new one. The procedure is repeated until a termination condition is satisfied.

Create initial solution x_p
Calculate $f(x_p)$
Set t
Set $0 < \beta < 1$.
Loop
  Select a neighboring solution x_n
  Calculate $f(x_n)$
  Compute $\Delta f = f(x_n) - f(x_p)$.
  If ($\Delta f < 0$)
    Then $x_p = x_n$
    Else
      If $\frac{1}{1+e^{-\frac{-\Delta f}{t}}} > random\ (0,1)$
        Then $x_p = x_n$
        Else
          $t(k+1) = \beta t(k)$,
Until termination condition is satisfied.

Fig. 1 Simulated Annealing

II. Genetic Algorithm

Genetic algorithms (GAs) are a specific type of Evolutionary Algorithm (EA). GAs will be the center of attention appearing to be the best suited evolutionary algorithms for combinatorial optimization problems. The power of GAs comes from their reliable, robust optimization method and applicability to a variety of complex problems [8].

A. The basic principles of GA

The basic principles of Genetic Algorithms (GAs) were introduced by Holland (1975). [9]. GAs are an optimization and search technique which emerged from the study of biological evolution [8].

In general GAs can be described as follows:

Genetic algorithms start with generating random populations of possible solutions. Each individual of the population is represented (coded) by a DNA string, called a chromosome, and the chromosome contains a string of problem parameters. Individuals from the population are selected based on their fitness values. The selected parents are recombined to form a new generation. This process is repeated until some termination condition is met. Figure (2) shows the Abstract Genetic Algorithm (AGA).

BEGIN GA
  Create initial population randomly
    While NOT stop DO
      BEGIN
        Select parents from the population.
        Produce children from the selected parents.
        Mutate the individuals.
        New population = children
      END
  Output the best individual found.
END GA

Fig. 2 Abstract Genetic

III. Principle of SAGA

In this work two SA algorithms are used to generate new two individuals which are transferred to a genetic algorithm, using multi crossover technique and swapped inverted crossover [10] to produce offspring consist of 94 children, from these offspring the best two individuals are selected according to their fitness values, once again these two individuals are used as inputs to both of the SA algorithms in order to improve these solutions, If solutions are no longer improved within consecutive iterations, then the best memorized solutions from the SA algorithms will be moved to the GA to repeat the above process.

The role of the SA is to improve or change the population which caused the GA to become trapped at local minima. The algorithm presented in this work can be effective to solve TSP with different sizes and tend to produce better quality results in lesser time.

```
Initiate two random individuals (a1,a2)
j = 0
Start SAGA
While termination condition not met
 j = 0
 While j ≤ 10
        a3 ← SA1(a1)
        a4 ← SA2(a2)
        Memorize best result  a3,a4
 End (while j)
 C ← 0
  P1 ← (a3, a4)
 While C ≤ 10
    C ← C + 1
    F1 ← f(p1)
   P2 ← GA (p1)
   F2 ← f(p2)
        If F(2) < F(1)
              C ← 0
              P1 ← p2
              P3 ← p2
             end
                Else
                   C ← C + 1
                   P3 ← p1
                  End
 End (while C)
   (a1, a2) ← P1
End (HGSAA)
```

Fig. 3 SAGA algorithm

IV. Experimental Results of SAGA

The proposed algorithm was tested using symmetric Traveling Salesman Problem (TSP) instances from known TSPLIB [11] and the results show that the algorithm is able to find an optimal solution or near optimal solution for varying sizes of these instances. Table I shows some results produced by proposed algorithm

TABLE I
RESULTS OF SAGA

problem	Optimal	Best result
eil101	629	629
kroA200	29368	29368
Pcb1173	56892	57142

V. Conclusion

This work proposed a new approach to combined simulated annealing and a genetic algorithm (SAGA) in order to reap the benefits of SA and reduce the time that GA spends stuck at local minima. The proposed algorithms tend to produce better quality results in smallest amount of time.

References

[1] An Introduction to Genetic Algorithms for Electromagnetics. Haupt, R.L. 2, s.l. : IEEE, April 1995, Vol. 37, pp. 7-15.

[2] Parallel simulated annealing and genetic algorithms: A space of hybrid methods. H Chen, N S Flann. s.l: International Conference on Evolutionary Computation the Third Conference Parallel Problem Solving, 1994. 866.

[3] Genetic algorithms and very fast simulated reannealing: A comparison. Rosen, L. Ingber and B. 11, s.l. : Mathematical Computer Modeling, 1992, Vol. 16. 87-100.

[4] Metropolis, N, et al. Equation of State Calculations by Fast Computing Machines. Florida State University. [Online] 1953. [Cited: 2 17, 2008.] www.csit.fsu.edu/~beerli/mcmc/metropolis-et-al-1953.pdf.

[5] W. Thomas. Global Optimization Algorithms Theory and Application. Thomas Weise. [Online] 2008. [Cited: 11 7, 2008.] http://www.it-weise.de/projects/book.pdf.

[6] S. Kirkpatrick, C. D. Gelatt, Jr., M. P. Vecchi. Optimization by Simulated Annealing. Computer Engineering Research Group. [Online] May 13, 1983. [Cited: 8 13, 2007.] http://www.eecg.utoronto.ca/~janders/ece1387/readings/sim_anneal.pdf.

[7] Bradley, J and Lambert, C. Simulated Annealing Applications. University of Victoria . [Online] November 18, 1999. [Cited: 4 12, 2008.] http://www.me.uvic.ca/~zdong/courses/mech620/SA_App.PDF.

[8] Beasley, D, Bull, D R and Martin, R. An Overview of Genetic Algorithms :. Part 1, Fundamentals. Norwegian University of Science and Technology. [Online] 93. [Cited: 3 24, 2008.] http://www.idi.ntnu.no/emner/it3704/lectures/papers/Beasley93GA.pdf.

[9] Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. Holland, J. s.l. : The University of Michigan Press, 1975.

[10] An Improved Genetic Algorithm to Solve the Traveling Salesman Problem. Sallabi, Omar M and Elhaddad, Younis R. Rome : World Academy of Science, Engineering and Technology, 2009. pp. 471-474. Issue 52.

[11] http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/. Heidelberg University. [Online] [Cited: 1 22, 2007.]