Iranian Journal of Operations Research
Vol. 2, No. 2, 2011, pp. 63-71

A hybrid metaheuristic using fuzzy greedy search operator for combinatorial optimization with specific reference to the travelling salesman problem

K. Sheibani

We describe a hybrid meta-heuristic algorithm for combinatorial optimization problems with a specific reference to the travelling salesman problem (TSP). The method is a combination of a genetic algorithm (GA) and greedy randomized adaptive search procedure (GRASP). A new adaptive fuzzy a greedy search operator is developed for this hybrid method. Computational experiments using a wide range of standard benchmark problems indicate that the proposed hybrid meta-heuristic approach is very efficient.

Keywords: *Combinatorial optimization, Genetic algorithms, GRASPS, Meta-heuristics.*

Manuscript received on 20/02/2010 and Accepted for publication on 26/12/2010.

1. Introduction

Combinatorial optimization problems are normally easy to describe but difficult to solve. To illustrate this point, consider the travelling salesman problem (TSP). There are many variations of the TSP; see Gutin and Punnen [9]. In this paper, the Euclidean TSP is considered. This problem can be represented by n cities numbered 1, 2,…,n and a Euclidean distance ($c_{i,j}$) between any two cities i and j. The goal in the TSP is to find a tour, which visits each city exactly once and is of minimum length. As the starting point is arbitrary and the distance between every pair of cities is the same, there are clearly $(n - 1)!/2$ feasible solutions. The TSP is a classical NP-hard optimization problem. Hence, approximation methods are generally considered to be the only practical way to solve most real-life problems. In recent years, there has been a growth of interest in the development of systematic search methods for solving such problems in operationsl research. A much newer area of research is the hybridization of metaheuristics; see Sheibani [18]. The use of search techniques on a solution space is central to the design of a solution method. Indeed, adopting a robust search technique significantly improves the overall performance.

The systematic study of the TSP as a combinatorial optimization problem began with the work of Dantzig et al. [4]. Various approaches have been proposed for the problem as demonstrated in Merz and Freisleben [13], Schmitt and Amini [16], Laporte and Palekar [11] and Laporte [12]. We will now discuss some methods that are most relevant to our discussion. The early efforts to find approximate solutions of the TSP by using genetic algorithms (GA) were those made by Goldberg and Lingle [7], using partially mapped crossover (PMX), Grefenstette et al. [8], using greedy crossover, Davis [5], using order crossover (OX), and Oliver et al. [14], using cycle crossover (CX). Other related works include Cheng and Gen [3], using greedy selection crossover (GSX), Chatterjee et al. [2], using a non-random initial population obtained by the nearest neighbour heuristic, and an asexual scheme in the new population generations, Qu and Sun [15], using a modification of the GSX, and a measure to prevent premature convergence.

Iran Telecom Research Centre (ITRC), North Kargar Street, P.O. Box 14155-3961, Tehran 14399-55471 Iran.
Email: ksheibani@itrc.ac.ir

Here, we introduce a new adaptive fuzzy greedy search operator for a hybrid meta-heuristic, which is a combination of GA and greedy randomized adaptive search procedures (GRASP) (Feo and Resende [6]) to find near-optimum solutions for the TSP. The concluding remarks contain some suggestions for further research.

2. Methodology

2.1. A Hybrid Metaheuristic

The aim of the design of a hybrid method is to combine the strengths of some different techniques in order to improve the efficiency of a single approach; see Sheibani [17]. Here, we propose a hybrid metaheuristic for the TSP. The proposed method is based on the use of GA and some of the ideas of the GRASP. The methodology uses the solutions from the construction phase of GRASP as the initial population of GA. A new adaptive fuzzy greedy search operator is developed for this hybrid method. Figure 1 illustrates the proposed hybrid method in a pseudo-code (the variable $P(t)$ represents a set of population members at generation t).

```
HYBRID METHOD PROCEDURE
BEGIN
  t ← 0;
  P(t) ← Ø;
  Solution ← Ø;
  WHILE (P(t) is not complete) DO
    P(t) ← GRASP_construction(seed);
  evaluate(P(t));
  WHILE (not termination condition) DO
    BEGIN
      t ← t + 1;
      P(t) ← select(P(t - 1));
      recombine(P(t));
      evaluate(P(t));
    END
 END
```

Figure 1. The proposed hybrid metaheuristic in a pseudo-code

2.2. The Fuzzy Greedy Evaluation

In order to apply our hybrid method to the TSP, a solution is represented by a string of integers, each being a sequence of n arranged cities, numbered 1 to n, which represents the order of the cities on a circle. The proposed method uses a set of candidate cities, each of which can be incorporated into the partial solution under construction without causing infeasibility. The priority of the cities in the list is determined according to an evaluation function through equation (1), below this is a modification of the general formulas of the families of fuzzy membership functions Klir; see Yuan [10]:

$$\mu(x) = \frac{1}{1 + \lambda^2 \left(\left(\frac{1-\lambda}{\lambda} \right) x - \theta \right)^2}. \quad (1)$$

In (1), x is the distance between any two cities ($c_{i,j}$), the parameter θ is a basic measure for evaluating the priority to be assigned to x, which is the average distance between successive cities in the current best solution obtained by the algorithm. This parameter will be seen to play an adaptive role, so that good choices made at previous stages (giving rise to the best solution so far) will also influence future choices. The parameter λ is a tuning parameter that is chosen by experimentation, such that $0 \leq \lambda < 1$, to adjust θ.

The proposed function has the following properties: $\mu(\lambda\theta / (1 - \lambda)) = 1$ and $0 < \mu(x) < 1$ for all $x \neq \lambda\theta / (1 - \lambda)$. This implies that the choice of the next city to travel will have a distance closest numerically to $\lambda\theta / (1 - \lambda)$. Further details on the methodology can be found in Sheibani [19].

2.3. Initial Population Generation

The initial population was generated using the construction phase of GRASP. This process will be referred to as the GRASP-link. In this phase, a candidate list is formed from the possible cities (i.e., those not yet visited). Then, we select two of the candidate cities from the list, which have maximum μ values in order to build the restricted candidate list (RCL). One of these elements in the RCL is selected at random to incorporate into the partial solution under construction. This process is repeated until a full TSP tour is formed.

2.4. Selection Scheme

A mixed strategy based on the roulette wheel selection and the elitist replacement is adopted. The roulette wheel method uses a probability distribution for selection of a chromosome, which is proportional to its fitness ($1/c$), where c is the tour length. Elitist replacement puts the fittest chromosome in the current population directly into the next generation.

2.5. An Adaptive Fuzzy Greedy Search Operator

A new search operator (called FGSX) was proposed by modifying the method of Qu and Sun [15]. The operating principle of FGSX is shown schematically in Figure 2. Let P_1 and P_2 be two randomly selected chromosomes from the previous generation. Each is a sequence of 9 arranged cities, numbered 1 to 9, which represents the order of the cities on a circle. First, we arbitrarily select a city, say 4, as the starting point in the offspring O_1. Then, we duplicate all cities in the selected parent chromosomes, which have not been incorporated in the offspring O_1 (between two cut points marked by '|'), as shown under the heading "Duplication" in the figure. This guarantees that the next two possible candidate cities have not already been incorporated in the offspring under construction. The next city in the offspring is determined as shown under the heading "Selection". Assume that $\mu(c_{4,3})$ is greater than $\mu(c_{4,5})$, indicating that the choice of travel from city 4 to 3 is more suitable than 5. So, we should select city 3 as the second city in the offspring O_1. The process is continued until a completely new offspring is formed. It is important to note that the FGSX can be adaptive in the sense that it attempts to learn from the best solution obtained in the previous generation by updating the parameter θ at each generation.

Parents

P_1: (1 2 3 4 5 6 7 8 9)
P_2: (4 3 2 1 5 6 7 8 9)

Duplication

1 2 3 **4** | **5** 6 7 8 9 1 2 3 | 4 5 6 7 8 9
4 | **3** 2 1 5 6 7 8 9 | 4 3 2 1 5 6 7 8 9

Selection

Assuming that $\mu(c_{4,5}) < \mu(c_{4,3})$,
city 3 is selected for offspring O_1

Offspring

O_1: (4 3 - - - - - - -)

Figure 2. An illustration of the proposed FGSX

2.6. Mutation Operator

We use a simple reordering operator as a mutation by selecting two points along the length of the single chromosome at random and then reverse the order of the sub-sequence between these two points. We should also mention that this reordering operator is commonly known as a 2-opt move in the TSP literature, which involves the reversal of a tour segment. The operating principle is shown schematically in Figure 3.

Mutation

S: (1 2 | 3 4 5 6 7 | 8 9) $\rightarrow$ **S′: (1 2 | 7 6 5 4 3 | 8 9)**

Figure 3. A simple reordering operator

3. Experimental Results

3.1. Parameter Setting

The proposed method was implemented in C++ code. The experimental results were obtained for 10 replications with different seeds common in all problem instances for the random number generation. The random number generator used the standard C++ library function. The test problems that were chosen were part of the extensive set of standard benchmark problems available in TSPLIB. The performance was measured for both the solution quality and algorithm computational time (CPU time) on a 2.80 GHz processor. The solution quality was determined with the percentage deviation of the obtained solution (tour length) from the best-known solution through equation (2):

$$Error\% = \frac{c - c'}{c'} \times 100, \tag{2}$$

where c is the obtained tour length and c' is the best-known solution taken from the TSPLIB library (http://comopt .ifi.uni-heidelberg.de/software/TSPLIB95).

The performance of a GA greatly depends on the structure of the problem considered, architecture of the algorithm and the settings for the algorithm parameters such as population size, crossover and mutation rates. Finding universal parameter values to prevent premature convergence still remains an unsolved problem (Back [1] and Yun and Gen [20]). Our experimentation showed that good performance of the proposed hybrid method was obtained when setting the genetic

parameters as follows: population size *popsize* = 45, crossover probability P_c = 0.45, mutation probability P_m = 0.55 and the number of iterations equal to50,000 for the termination condition of the algorithm. At the end of this section, we briefly discuss the dependency between the values of genetic parameters and the rate of convergence to good quality local optima.

We introduced the tuning parameter λ in addition to the above GA parameters. The performance of the proposed method is sensitive to the chosen value of λ. Extensive experimentation showed that good performance was obtained when setting λ to 0.1, 0.2, or 0.3. We found that usually this range of λ values gave the best solutions. Figure 4 illustrates the effect of different values of λ on the computational performance of the proposed method.

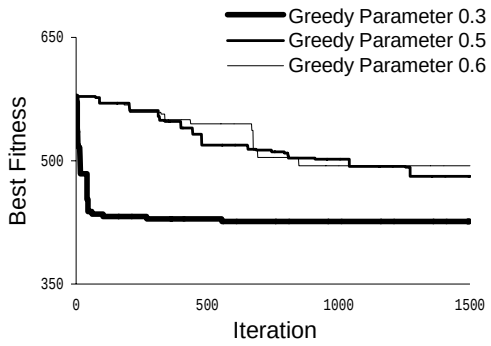

Figure 4. A comparison of the proposed method on the eil51 problem with different values of λ

3.2. Effectiveness of the Initial Population

We evaluated the performance of the proposed method using random and the GRASP-link initializations; all other parameter settings were the same. A comparison of the results is tabulated in Table 1.

Table1. Comparison of two different initialization methods

Problem	*GRASP-Link*		*Rand-Initial*	
	Mean Error%	*Variance*	*Mean Error%*	*Variance*
eil51	0.11	0.01	0.51	0.35
eil76	0.89	0.69	0.53	0.20
kroA100	0.22	0.05	0.31	0.08
lin105	0.08	0.02	0.74	1.15
bier127	0.83	0.14	1.36	0.31
ch130	1.47	0.41	2.36	0.24
kroA150	1.37	0.26	1.88	1.09
tsp225	1.86	1.29	2.89	2.20
Average	**0.86**	**0.36**	**1.33**	**0.70**

Given that the initialization process is computationally inexpensive, compared to the rest of the GA, the results show the value of using an effective initialization process. The evolution of solutions using these two methods is illustrated in Figure 5. The results indicate that for a small number of generations, the GRASP-Link initialization leads to better performance than a random initialization.

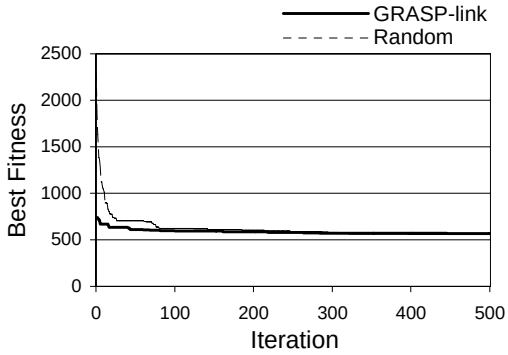

Figure 5. Two different initializations for the eil76 problem

3.3. Effectiveness of the FGSX Operator

A comparison of the proposed operator FGSX with the simplified GSX is illustrated in Table 2. It should be noted that GSX has been the most effective operator in comparison with PMX, OX, and CX for the TSP problems; see Qu and Sun [15]. On average, the proposed FGSX has an error of 0.14% less than that of GSX.

Table 2. Comparison of the FGSX with the simplified GSX

Problem	*FGSX*		*GSX*	
	Mean Error%	*Variance*	*Mean Error%*	*Variance*
eil51	0.11	0.01	0.44	0.33
eil76	0.89	0.69	0.89	0.69
kroA100	0.22	0.05	0.42	0.69
kroB100	0.91	0.07	0.91	0.07
kroC100	0.59	0.22	0.66	0.56
kroD100	1.29	0.63	1.18	0.35
kroE100	0.42	0.05	0.49	0.09
rd100	0.70	0.68	1.00	0.98
eil101	1.46	0.58	1.46	0.58
lin105	0.08	0.02	0.27	0.17
bier127	0.83	0.14	1.19	0.67
ch130	1.47	0.41	1.70	0.63
ch150	0.58	0.09	0.87	0.40
kroA150	1.37	0.26	1.63	0.41
kroB150	1.62	1.05	1.62	1.05
tsp225	1.86	1.29	1.86	1.29
Average	**0.90**	**0.39**	**1.04**	**0.56**

3.4. Overall Results

The performance of the proposed method on a range of problems of varying sizes is reported in Table 3. We evaluated our method on some bigger standard problems to study its behavior in the next section.

Table 3. The performance of the proposed method

Problem	*Mean Error%*	*Variance*	*CPU sec*
eil51	0.11	0.01	97
eil76	0.89	0.69	197
kroA100	0.22	0.05	326
kroB100	0.91	0.07	326
kroC100	0.59	0.22	326
kroD100	1.29	0.63	326
kroE100	0.42	0.05	328
rd100	0.70	0.68	326
eil101	1.46	0.58	329
lin105	0.08	0.02	355
bier127	0.83	0.14	375
ch130	1.47	0.41	528
ch150	0.58	0.09	685
kroA150	1.37	0.26	683
kroB150	1.62	1.05	684
tsp225	1.86	1.29	1494
Average	**0.90**	**0.39**	**462**

3.5. Sensitivity Analysis

Given the results of Table 3, it might be reasonable to plan experimentation in such a way that larger problems are given a larger number of iterations. In order to keep the computational effort approximately unchanged, we keep the maximum number of tour evaluations constant; this would mean simultaneously reducing the size of the population and rising the number of iterations. Some bigger problems with different parameter values but with the identical tour requirements were considered. The results are tabulated in Table 4.

Table 4. Comparison with different values of the parameters

Problem	*Popsize*	*Iteration*	*Mean Error%*	*Variance*	*CPU sec*
lin318	25	90,000	2.37	0.72	2858
	45	50,000	2.33	0.31	2919
pcb442	25	90,000	2.26	0.20	5476
	45	50,000	3.55	1.14	5580
vm1084	15	150,000	3.75	0.31	32270
	45	50,000	6.99	1.79	33580

The results reported in Table 4 show that the quality of solutions for the two larger problems significantly improved by increasing the number of iterations and reducing the population size. This experimentation thus led us to set the parameters to the values that depended on the size of the problem.

4. Concluding Remarks

We introduced a new idea to integrate approaches for solving hard combinatorial optimization problems. The proposed methodology evaluates objects in a way that combines fuzzy reasoning with a greedy mechanism. In other words, we exploit a fuzzy solution space (fuzzy set) using greedy methods. Our methodology also attempted to adapt its knowledge from previous experiments, thereby improving the exploration of the promising areas of the search space. In this context, a hybrid meta-heuristic based on a combination of genetic algorithms (GA) and greedy randomized adaptive search procedures (GRASP) was developed for the travelling salesman problem (TSP). We examined the effectiveness and the efficiency of the proposed hybrid method for the Euclidean TSP on a wide range of standard benchmark problems. The proposed method has the potential for application to other combinatorial optimization problems, if a suitable evaluation measure can be properly defined based on global information for a given problem solution. This would correspond to the role of the tour length in the TSP. For future research, we believe that the following topics are potentially useful: (1) extending our method to other objectives, (2) developing efficient methods using the fuzzy greedy evaluation concept in other areas of combinatorial optimization.

References

[1] Back, T. (1992), The interaction of mutation rate, selection, and self-adaptation within a genetic algorithm. *Proceedings of the 2nd International Conference on Parallel Problem Solving From Nature*, Elsevier, Amsterdam, pp. 85-94.

[2] Chatterjee, S., Carrera, C. and Lynch, L.A. (1996), Genetic algorithms and traveling salesman problems, *European Journal of Operational Research*, 93, 490-510.

[3] Cheng, R. and Gen, M. (1994), Crossover on intensive search and traveling salesman problem, *Computer and Industrial Engineering*, 27, 485-488.

[4] Dantzig, G.B., Fulkerson, D.R. and Johnson, S.M. (1954), Solution of a large scale traveling salesman problem, *Operations Research*, 2, 393-410.

[5] Davis, L. (1985), Applying adaptive algorithms to epistatic domains, *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, Los Angeles, pp. 162-164.

[6] Feo, T.A. and Resende, M.G.C. (1995), Greedy randomized adaptive search procedures, *Journal of Global Optimization*, 6, 109-133.

[7] Goldberg, D.E. and Lingle, R. (1985), Alleles, loci and the travelling salesman problem, *Proceedings of the 2nd International Conference on Genetic Algorithms*, Lawrence Erlbaum, London, pp. 154-159.

[8] Grefenstette, J.J., Gopal, R., Rosmaita, B. and Van Gucht, D. (1985), Genetic algorithms for the traveling salesman problem, *Proceedings the 2nd International Conference on Genetic Algorithms*, Lawrence Erlbaum, London, pp. 160-168.

[9] Gutin, G. and Punnen, A.P. (2002), *The Traveling Salesman Problem and Its Variations*, Kluwer Academic Publishers, Dordrecht.

[10] Klir, G.J. and Yuan, B. (1995), *Fuzzy Sets and Fuzzy Logic: Theory and Applications*, Prentice Hall, Englewood Cliffs, NJ.

[11] Laporte, G. and Palekar, U. (2002), Some applications of the clustered travelling salesman problem, *Journal of the Operational Research Society*, 53, 972-976.

[12] Laporte, G. (2010), A concise guide to the traveling salesman problem, *Journal of the Operational Research Society*, 61, 35-40.

[13] Merz, P. and Freisleben, B. (2001), Memetic algorithms for the traveling salesman problem, *Complex Systems*, 13, 297-345.

[14] Oliver, I.M., Smith, D.J. and Holland, J.R.C. (1987), A study of permutation crossover operators on the traveling salesman problem, *Proceedings the 2nd International Conference on Genetic Algorithms*, Lawrence Erlbaum, London, pp. 224-230.
[15] Qu, L. and Sun, R. (1999), A synergetic approach to genetic algorithms for solving travelling salesman problem, *Information Sciences*, 117, 267-283.
[16] Schmitt, L.J. and Amini, M.M. (1998), Performance characteristics of alternative genetic algorithmic approaches to the traveling salesman problem using path representation: an empirical study, *European Journal of Operational Research*, 108, 551-570.
[17] Sheibani, K. (2005), *Fuzzy Greedy Evaluation in Search, Optimisation, and Learning*, PhD. dissertation, London Metropolitan University, London, UK.
[18] Sheibani, K. (2008), *Fuzzy Greedy Search in Combinatorial Optimisation*, Tadbir Institute for Operational Research, Systems Design and Financial Services, Tehran.
[19] Sheibani, K. (2010), A fuzzy greedy heuristic for permutation flow-shop scheduling, *Journal of the Operational Research Society*, 61, 813-818.
[20] Yun, Y. and Gen, M. (2003), Performance analysis of adaptive genetic algorithms with fuzzy logic and heuristics, *Fuzzy Optimization & Decision Making*, 2, 161-175.