International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

Comparison of IBEA and NSGA-II on MOTSP

Gerbie Atsede Mitiku

Tianjin University of Technology And Education, Department of Applied Computer Science, Dagu South Road,Hexi District,Tianjin,Prc-300222,China

Abstract: ***Multi-objective traveling salesman problem (MOTSP) is a well-known NP hard problem. In this paper, IBEA and NSGA-II algorithm is proposed to solve the MOTSP and compare the result of each algorithm and find a better algorithm based on the execution time it takes to find the Pareto-optimal solutions. The result shows that even though the two algorithm finds the true pare to front NSGA_II is better algorithm for MOTSP problem compare to IBEA***

Keywords: Multi-objective traveling salesman problem, Indicator-Based Evolutionary Algorithm (IBEA), Non-dominated Sorting based Genetic Algorithm –II (NSGA-II)

1. Introduction

Multi-traveling salesman problem (MOTSP) is an extension of traveling salesman problem, which is a famous NP hard problem, and can be used to solve many real world problems, such as railway transportation, routing and pipeline laying. The literature related to this MOTSP and its practical applications has been reviewed (Sofge, D., Schultz, A., & De Jong, K., 2002)In this work, different formulations are highlighted, and exact and different heuristics methods .For solving the problem are described Many Population-based metaheuristic algorithms for the MTSP were proposed by (Sofge, D., Schultz, A., & De Jong, K., 2002), (Zhao, 2008), (Király, 2011)and (Yu, 2012). (Chang, 2012) considered the routing problem of the courier service for a city. The problem is formulated as a MOmTSP by considering hard time windows (MOmTSPTW). A multi-objective scatter search is proposed to minimize the operating costs and to improve the level of service. This paper gone find optimal solution by considering two objectives(i) The length of the total tour between the cities and (ii)the total cost between the cities

This paper use NSGA-II and IBEA to find the optimal solutions and compare the result based on the execution time it takes to find the optimal solution

2. Non-dominated Sorting based Genetic Algorithm-II

The NSGA-II is a multi-objective evolutionary optimization algorithm based on non-dominated sorting of the population. In multi-objective problem, there is a set of solutions better than all the other ones in the search space which is named the Pareto optimum or set of non-dominated solutions. The Pareto optimum is the set of non-dominated solutions considering the entire search space. Multi-objective evolutionary methods try to find this solution set by using each objective separately, without aggregating them as a unique objective

Defination1. Dominance: A solution $x^{(1)}$ is said to dominate the other solution $x^{(2)}$ if both the following conditions are true:

i) The solution $x^{(1)}$ is no worse than $x^{(2)}$ in all objectives, or

$$\forall j = 1,2..m\ fj(x^{(1)}) \leq fj(x^{(2)})$$

Thus, the solutions are compared based on their objective function values.

ii)The solution $x^{(1)}$ is strictly better than $x^{(2)}$ in at least one objective, or

$$\exists j = 1,2..m\ fj(x^{(1)}) < fj(x^{(2)})$$

If any of the above condition is violated, the solution $x^{(1)}$ does not dominated the solution $x^{(2)}$.

Defination2. Non-dominated sorting approach: First, for each solution calculate two entities:

i) domination count ni the number of solutions which dominate the solution i , and

ii) Si , a set of solutions which the solution i dominates. At the end of this procedure, all solutions in the first non-dominated front will have their domination count as zero. Now, for each these solutions (each solution i with $ni = 0$), we visit each member (j) of its set Si and reduce its domination count by one. In doing so, if for any member j the domination count becomes zero, we put it in a separate listP'. After such modifications on Si are performed for each i with $ni = 0$, all solutions of P' would belong to the second nondominated front. The above procedure can be continued with each member of P' and the third non-dominated front can be identified. This process continues until all solutions are classified.

Step1 for each $i \in p, ni = 0$ initialize $si = \emptyset$For all j $\neq$ i and j $\in$ p , perform *Step 2* and then proceed to *Step 3*.

Step2 If i°j (i dominate j) , update S p = Sp ∪ {j} . Otherwise, j °i , set $ni = ni + 1$

Step3 If $ni = 0$, keep i in the first non-dominated front $p1$ (we called this set P' in the above paragraph). Set a front counter = 1 .

Step4 While$pk \neq \emptyset$, perform the following steps.

Step5 Initialize $Q = \emptyset$ for storing next non-dominated solutions. For each $i \in pk$ and for each $\in si$,

Step5a Update $nj = nj - 1$.

Step5b If $nj = 0$ keep j in , or perform Q = Q ∪ {j} .

Step6 Set $k = k + 1$ and $pk = Q$.Go to *Step4*

In NSGA-II procedure, at any generation t, the offspring population Qt is first created by using the parent population Pt and the usual genetic operators. Thereafter, the two populations are combined together to form a new population Rt of size $2N$. Then, the population Rt classified into different non-domination classes. Thereafter, the new population is filled by points of different non-domination

fronts, one at a time. The filling starts with the first non-domination front(of class one) and continues with points of the second non-domination front, and so on. Since the overall population size of is $2N$, not all fronts can be accommodated in N slots available for the new population. All fronts which could not be accommodated are deleted. When the last allowed front is being considered, there may exist more points in the front than the remaining slots in the new population. Instead of arbitrarily discarding some members from the last front, the points which will make the diversity of the selected points the highest are chosen. The crowded-sorting of the points of the last front which could not be accommodated fully is achieved in the descending order of their crowding distance values and points from the top of the ordered list are chosen. The crowding distance i d of point is a measure of the objective space around i which is not occupied by any other solution in the population (Elham Zhoulaian, 2014)

3. Indicator-Based Evolutionary Algorithm (IBEA)

Indicator-Based Evolutionary Algorithm (IBEA) is developed by (Eckart Zitzler and Simon Künzli) (Deb, 2001).it is s one of the very first multiobjective optimizers to integrate user preferences in a clear and mathematically sound way. The main contribution of IBEA was to open up a new research area on the design of multiobjective optimization algorithms which employ a so-called quality indicator in their (environmental) selection procedure (Brockhoff, 9 Jun 2015).

Before we describe the original IBEA algorithm in more detail, let us mention that we consider, w.l.o.g., minimization problems here where the Pareto dominance relation $\prec$ is defined between solutions x^1 and x^2 as $x^1 \prec x^2$ if and only if $fi(x^1) \leq fi(x^2)$ for all objective functions $fi: X \to Z\ (1 \leq i \leq k)$ and $fi(x^1) < fi(x^2)$ for at least one objective function. In this case, we also say x^1 dominates x^2 An $m-$ary quality indicator is furthermore a function $I: \Omega m \to R$ that maps m solution sets $x_1 \ldots, x_m$ from the set of all possible solutions $(x_1 \ldots, x_m \in \Omega = 2^x)$ to a real number. Nowadays, mostly unary quality indicators such as the standard hypervolume indicator are used in both performance assessment and the definition of solution quality within the environmental selection. Instead, IBEA itself is based on binary quality indicators that map two solution sets to a real number. To be more precise, the fitness of a solution x1 in IBEA's population P is assigned by

$$\sum_{x^2 \in P/\{x^1\}} -e - I(\{x^2\},\{x^1\})/k$$

where $\kappa > 0$ is a parameter of the algorithm and $c = \max x^1, x^2 \in P\ |I(x^1, x^2)|$ is the maximum indicator value between any two population members. This fitness assignment scheme of IBEA has the theoretical property that if a solution x^1 dominates solution x^2 then also $F(x^1) > F(x^2)$ as long as the chosen binary indicator I itself is "dominance preserving"1 (Brockhoff, 9 Jun 2015). Note that in the following, we abuse the mathematical notation and write I(x,y) instead of I({x},{y}) if x and y are single solutions. Examples of dominance preserving binary indicators are the binary hypervolume and the binary ε-indicator which, for that reason, have been proposed to be used in the original IBEA publication. The binary (additive) ε-indicator assigns to two solution sets A and B the minimal objective value ε by which all solutions in A have to be improved (along each objective) in order to (weakly) dominate all solutions in B (Brockhoff, 9 Jun 2015)

$I\varepsilon+(A,B) = \min \varepsilon \in \forall x^2 \in B \exists x^1 \in A: fi(x^1) - \varepsilon \leq fi(x^2)$ for $i \in \{1, \ldots, k\}$. The binary ε-indicator is negative if all solutions in B are dominated by at least one solution in A. The binary hypervolume indicator used in (Eckart Zitzler and Simon K)assigns to two solution sets A and B the "volume of the space that is dominated by B but not by A with respect to a predefined reference point" in objective space (Eckart Zitzler and Simon K):

$$I_{HD}(A,B) = \begin{cases} I_H(B) - I_H(A) \text{ if } \forall x^2 \in B \exists x^1 \in A: x^1 < x^2 \\ I_H(A+B) - I_H(A) \text{else} \end{cases}$$

where I_H (.) denotes the standard (unary) hypervolume proposed in (Brockhoff, 9 Jun 2015)and the index "HD" stands for "hypervolume difference". Also I_{HD} (A,B) is negative if all solutions in B are dominated by at least one solution in A. Note also that neither of the two binary indicators is symmetric, i.e., typically $I(A,B) \neq I(B,A)$ holds. The corresponding IBEA variants using the above defined hypervolume and ε-indicator are denoted $IBEA_{HD}$ and $IBEA_{\varepsilon+}$ respectively here. A binary quality indicator I is called dominance preserving if for all solutions $x^1, x^2, x^3 \in X$ both $x^1 < x^2 \Longrightarrow I(\{x^1\},\{x^2\}) < I(\{x^2\},\{x^2\})$ and $x^1 < x^2 \Longrightarrow I(\{x^3\},\{x^1\}) \geq I(\{x^3\},\{x^2\})$ hold. (Brockhoff, 9 Jun 2015)

4. Problem Description

The Multi-objective traveling salesman problem (MOTSP) could be defined by a complete (directed or undirected) graph G =(N,E,c) with being the set of N nodes, E being the set of edges fully connecting the nodes, and c being a function that assigns to each $edge\ (i,j) \in E$ a vector $cij^1 \ldots . cij^k$, where each element $c\ ij^k$ corresponds to a certain measure like distance, cost, etc. between nodes i and j. For the following we assume that $cij^k \ldots . cji^k$, for all pairs of nodes i, j and objectives k, that is, we consider only symmetric problems. The MOTSP is also the problem of finding "minimal" Hamiltonian circuits of the graph, that is, a set of closed tours visiting each of the $n = |N|$ nodes of G exactly once; here "minimal" refers to the notion of Pareto optimality.

5. Implementation

Initial population

Every gene of the chromosome represent a city and the sequence between the genes reflects the trip, for example the code is 0 1 2 3 4 5 0, which express that the trip go through the number 0 city to number 1 then 2, ... at last return the original city. After initialize population, every chromosome represents a random arranged of city's natural number. Suppose a coding of chromosome is a0a1...an-1a0 for the bi-objective TSP, the two evaluative fitness functions are defined as follows: The length of the total tour: $Length = dist(a0, a1) + dist(a1, a2) + \cdots + dist(an -$

$1, a0)$. $Here, dist(ai, aj)$ means the distance between city ai, and city aj. The total cost: $Costs = cost(a0, a1) + cost(a1, a2) + \cdots + cost(an - 1, a0)$. Here, $cost(ai, aj)$ means the cost between city ai, and city aj.

IBEA Algorithm (algorthim-1)

Input: *α (population size)*
N (maximum number of generations)
κ (fitness scaling factor)

Output: A (Pareto set approximation)
Step 1: Initialization: Generate an initial population P of size α; set the generation counter m to 0.
Step 2: Fitness assignment: Calculate fitness values of individuals in P, i.e., for all $x1 \in P$ *set* $F(x1) = x2 \in P\backslash\{x1\} - e - I(\{x2\}, \{x1\})/\kappa$.
Step 3: Environmental selection: Iterate the following three steps until the size of population P does not exceed α:
1. Choose an individual $x* \in P$ with the smallest fitness value, i.e., $F(x*) \leq F(x)$ for all $x \in P$.
2. Remove $x*$ from the population.
3. Update the fitness values of the remaining individuals, i.e., $F(x) = F(x) + e - I(\{x*\}, \{x\})/\kappa$ for all $x \in P$.
Step 4: Termination: If $m \geq N$ or another stopping criterion is satisfied then set A to the set of decision vectors represented by the no dominated individuals in P. Stop.
Step 5: Mating selection: Perform binary tournament selection with replacement on P in order to fill the temporary mating pool P'.
Step 6: Variation: Apply recombination and mutation operators to the mating pool P' and add the resulting offspring to P. Increment the generation counter (m = m +1) and go to Step 2. (Eckart Zitzler and Simon K)
On this paper we use κ (fitness scaling factor) =0.05

NSGA-II algorithm fitness assignment
First objective function for each path is calculated then paths are ranking based on dominance concept of Defination1 and non-dominated sorting of Defination2.

Objective function =[F1 F2] ……(1)
where :-

i) F1= dist(a0,a1)+dist(a1,a2)+…+dist(an-1,a0 ….(2)
(the total length it take for each chromosome)
ii) $F2 = cost(a0, a1) + cost(a1, a2) + \cdots + cost(an - 1, a0)$
……(3)
(The total cost it takes to travel for each chromosome)

To get an estimate of the density of solutions surrounding a particular solution in the population, the average distance of two solutions on either side of solution i along each of the objectives is calculated. This quantity serves as an estimate of the perimeter of the cuboid formed by using the nearest neighbors as the vertices. The crowding distance (i d) computation requires sorting the population according to each objective function value in ascending order of magnitude. Thereafter, for each objective function, the boundary populations are assigned an infinite or very large distance value. All other intermediate populations are assigned a distance equal to the absolute normalized difference in the function values of two adjacent populations. This calculation is continued with other objective functions. The overall crowding distance value is calculated as the sum of individual distance values corresponding to each objective. The procedure for calculating the crowding distance is given below: (Deb, 2001)

$$d_1^i = \frac{/f_1^{i+1} f_1^{i-1}/}{f_1^{max} f_1^{max}} \quad \text{……..(4)}$$

$$d_2^i = \frac{/f_2^{i+1} f_2^{i-1}/}{f_2^{max} f_2^{max}} \quad \text{……(5)}$$

Crowding distance

$$d_i = d_1^i + d_2^i \quad \text{……(6)}$$

In Equations (4) and (5) $f^{i-1}$and $f^{i+1}$are objective function values of two solutions on either side of solution i . f^{max} and f^{min} are maximum and minimum values of each objective function respectively
Tournament Selection is utilized as the selection operator and the order crossover (OX) used

6. Experiment Results and Analysis

The proposed algorithm has been implemented in matlab R2016a on a CPU Intel Corei7 with 2.5GHz and 8 GB of RAM. In the experiment process number of generation set to 10000 and test each algorithm with different number of population and by 5 times execution .The result shows in the form of Table 1

Table 2: Performance Comparison of NSGA-II and IBEA

Number of population	NSGA-II (running time)	IBEA (running time)
20	3.7514e-01	5.4825e_01
50	3.2714e-01	7.7695e-01

7. Conclusion

In this paper NSGA-II and IBEA has been implemented for solving MOTSP. The Experimental results obtained show that this two approach can find optimal path but based on Run time it takes to find the result NSGAII has better performance than IBEA for MOTSP problem

References

[1] Brockhoff, D. (9 Jun 2015). A Bug in the Multiobjective Optimizer IBEA: Salutary Lessons for Code Release and a Performance Re-Assessment.
[2] Chang, T. S. (2012). City-courier routing and scheduling problems.
[3] Deb, K. (2001). optimization Multi-objective using evolutionary algorithms. WileyInterscience series in systems and optimization.
[4] Eckart Zitzler and Simon K. (n.d.). Indicator-Based Selection in Multiobjective Search .
[5] Elham Zhoulaian, S. J. (2014). Multi-Objective Routing by using non -Dominated Sorting Genetic Algorithm in Computer Networks.
[6] Király, A. &. (2011). Optimization of multiple traveling salesmen problem by a novel representation based genetic algorithm.
[7] Sofge, D., Schultz, A., & De Jong, K. (2002). Evolutionary computational approaches to solving the

multiple traveling salesman problem using a neighborhood attractor schema. In Applications of Evolutionary Computing.
[8] Yu, Q. W. (2012). A novel two-level hybrid algorithm for multiple traveling salesman problems.
[9] Zhao, F. D. (2008). An improved genetic algorithm for the multiple traveling salesman problem.