Bounded-Diameter MST Instances with Hybridization of Multi-Objective EA

Soma Saha
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Kharagpur West Bengal India 721302

Rajeev Kumar
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Kharagpur West Bengal India 721302

ABSTRACT
The Bounded Diameter (a.k.a Diameter Constraint) Minimum Spanning Tree (BDMST/DCMST) is a well-known combinatorial optimization problem. In this paper, we recast a few well-known heuristics, which are evolved for BDMST problem to a Bi-Objective Minimum Spanning Tree (BOMST) problem and then obtain Pareto fronts. After examining Pareto fronts, it is concluded that none of the heuristics provides the superior solution across the complete range of the diameter. We have used a Multi-Objective Evolutionary Algorithm (MOEA) approach, Pareto Converging Genetic Algorithm (PCGA), to improve the Pareto front for BOMST, which in turn provides better solution for BDMST instances. We have considered edge-set encoding to represent MST and then applied recombination operators having strong heritability and mutation operators having negligible complexity to improve the solutions. Analysis of MOEA solutions confirms the improvement of Pareto front solutions across the complete range of the diameter over Pareto front solutions generated from individual heuristics. We have considered multi-island scheme using Inter-Island rank histogram and performed multiple run of the algorithm to avoid from trapping into local-optimal solution-set.

General Terms
Algorithm, Design, Experimentation.

Keywords
Combinatorial optimization, multi-objective optimization, heuristics, MST, BDMST problem, evolutionary algorithm, Pareto front, edge list encoding.

1. INTRODUCTION
BDMST has many applications in real-world [2, 4, 22]; it is an NP-hard problem within diameter bound (D) ranges $4 \leq D < |V| - 1$ [9], where diameter bound (D) is a constraint, the maximum feasible longest path between two vertices of a connected, undirected, weighted graph G to generate feasible MSTs and V is the set of vertices of G. Well-known heuristics which are evolved to provide solutions to BDMST problem, are: e.g., one time tree construction (OTTC) [8], iterative refinement (IR) [8], randomized greedy heuristics (RGH) [21], random tree construction (RTC) [10], center based tree construction (CBTC) [10] and center-based recursive clustering (CBRC) [20]. Initially, algorithmic complexity directed the development of various heuristics. The complexity of OTTC, IR, CBTC and CBRC is $O(n^3)$ [8, 10, 20]. Depending on initial vertex, generated spanning tree (ST) differs for OTTC, IR and CBTC heuristics; therefore, one needs to execute the heuristics with each of the vertices as initial vertex with the expectation to get better low-weight spanning tree and then select the best generated tree which makes the complexity $O(n^4)$. RGH has complexity $O(n^2)$ [21]. RGH generates ST randomly; thus, by increasing number of execution of the algorithm (say, no. of execution = n), the possibility to produce better solution is increased and the complexity of RGH raises to $O(n^3)$.

In this work, we have adapted well-known heuristics to formulate bi-objective MST problem and then obtained Pareto fronts [23, 24]. BDMST is a specialize instance of BOMST. Considering Pareto front solutions for each heuristic as initial population, we provide an MOEA approach to improve Pareto front which in turn improves the BDMST solution within a particular diameter constraint for that heuristic. We have considered various performance metrices to analyze the MOEA solution set which supports the improvement of quality of solutions.

The paper is organized as follows, in Section 2, we include the basic definitions and outline problem formulation. We describe the overview of those heuristics which we deal in this work in Section 3. Next, Section 4 contains the evolutionary algorithm and the description of recombination and mutation operators. The results are contained in Section 5. We conclude with short discussion and future work in Section 6.

2. PRELIMINARY
2.1 Basic Definitions
Definition 1 Multiobjective Optimization Problem (MOP).
In an MOP, a number of objectives have to be minimized/ maximized along with constraints (optional) to achieve goal vectors which can be written as:
Maximize/Minimize : $F(\mathbf{X}) = \{f_1(\mathbf{X}), f_2(\mathbf{X}),\ldots, f_m(\mathbf{X})\}$
subject to satisfaction of the constraints:
$C(\mathbf{X}) = \{c_1(\mathbf{X}), c_2(\mathbf{X}),\ldots, c_k(\mathbf{X})\} \leq (0,\ldots,0)$

A set of objective values constitutes an *objective* space and the collection of decision variables forms a *decision* space.

Definition 2 Pareto-optimal set.
Without loss of generality, we assume Multi-Objective Spanning Tree (MOST) (includes BOMST) and BDMST are m-objective minimization problem. In an m-objective minimization problem, a vector of decision variables $x \in X'$ includes in Pareto-optimal (P) set as a Pareto-optimal point if another x^* does not exist such that $f_i(x^*) \leq f_j(x)$ for all $i = 1, 2, 3,\ldots,m$ and $f_i(x^*) < f_j(x)$ for atleast one j. Here, X' denotes the feasible region of the problem (i.e. where the constraints are satisfied).

Definition 3 Pareto dominance.
A vector $\vec{u} = (u_1,\ldots,u_k)$ dominate $\vec{v} = (v_1,\ldots,v_k)$ denoted by ($\vec{u} \ll \vec{v}$) , iff *u* is partially less than *v*; it can be represented as follows,

$i \in 1,2,\ldots,k, u_i \leq v_i \wedge \exists\, i \in 1\,2,\ldots, k : u_i < v_i$.

Definition 4 Pareto front.
Pareto front (*PF**), for a given MOP $\vec{f}(x)$ and Pareto-optimal set *P*, can be stated as, $PF^* = u = \vec{f} = (f_1(x),\ldots, f_k(x)), x \in P$.

Definition 5 Bi-Objective MST Problem (BOMST).
A connected, undirected, weighted graph $G = (V, E)$ is given; a minimum spanning tree (MST) or minimum-weight spanning tree is a spanning tree *T* with weight less than or equal to the weight of every other spanning tree. The single-objective MST problem can be formally stated as:

Minimizes, $Wt(T) = \sum_{e \in T} wt(e)$

The diameter of a tree is the maximum eccentricity of its vertices, the maximum longest path exists between two vertices. The BDMST problem is formulated as:

Minimizes, $Wt(T) = \sum_{e \in T} wt(e)$
with constraint, diameter of $T \leq D$, where *D*, the diameter bound, is given.

It was not known which of the algorithms yield the best solution, there are claims of getting a better tree for a specific diameter value. None of the heuristic shows the best result for complete range of the diameter. To illustrate this point, we have plotted results for obtaining DCMST for all values of the diameter with different heuristics.

We choose only one random plot which is shown in Figure 1 for 100 node non-Euclidean instance. OTTC and CBTC, both generate superior solutions leaving only negligible drift across the complete diameter bound for non-Euclidean instances. RGH generates better solutions within very small diameter bound; whereas for Euclidean instances it generates superior solutions across a much larger diameter bound range. For Euclidean instances, CBTC gives superior results for larger diameter range and IR provides nearly similar solution to CBTC for larger diameter range. We have seen similar pattern of solutions across the Pareto front for different instances. Therefore, on visualizing one plot either for Euclidean or non-Euclidean instances, we can conclude which heuristic generates superior solution on a certain range across the Pareto front.

Lately, the research efforts were directed on improving the solution quality. It was not attempted which heuristic to use for a given application needing the entire range of diameters. Some work has been done on improving the quality of solutions by EA [18].

In an attempt to yield, always the best solutions possible across the entire ranges of the diameter, the work in this paper is focused on obtaining a Pareto front which gives the best possible solution across the whole Pareto front. With that aim, we recast BDMST problem and then we provide the MOEA approach (PCGA [15] and modified PCGA) to the solutions to yield an improved Pareto front across the full range of diameter. Through an improved Pareto front, one can get the best possible instance of an MST satisfying the specified diameter constraint.

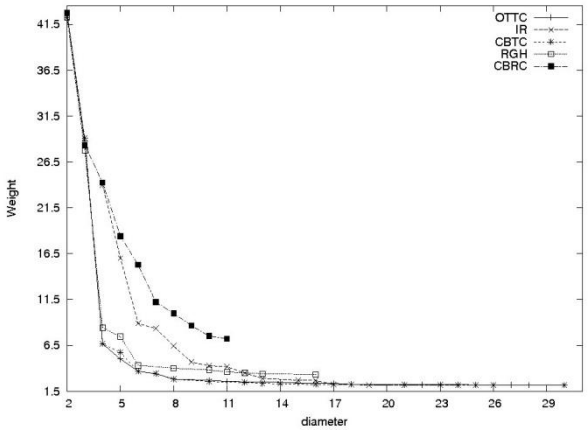


Figure 1: Pareto front obtained from OTTC, IR, CBTC, RGH and CBRC heuristics for 100 node non-Euclidean instance. OTTC and CBTC, both generates superior solutions leaving only negligible drift across the complete diameter bound for non-Euclidean instances. RGH generates better solutions within very small diameter bound.

3. OVERVIEW OF HEURISTICS

We have considered the existing well-known heuristics for BDMST problem.

3.1 One Time Tree Construction

Deo and Abdalla [8] proposed *One Time Tree Construction* (*OTTC*), based on *Prim*'s algorithm. It builds a spanning tree, starting from each node and connecting the nearest neighbor that does not violate the diameter constraint. The complexity of this algorithm is $O(n^3)$, where *n* is the number of vertices in the graph. There are number of research papers for BDMST problem using OTTC for both Euclidean and non-Euclidean instances [3, 8, 10, 20, 21, 25]; all those works generalize MST problem with particular diameter constraint. But, for bi-objective MST problem using OTTC heuristics very little work has been done [14].

3.2 Iterative Refinement

Iterative Refinement (IR) heuristic is proposed again by Deo and Abdalla [8]; initially, an unconstrained minimum spanning tree (MST) for the input graph is computed using the *Prim*'s algorithm, then in each iteration one edge is removed that breaks the longest path in the spanning tree and replaced it by a non-tree edge without increasing the diameter. This process continues until diameter constraint is satisfied or the algorithm fails. IR is computationally expensive [21] due to large number of iteration of edges and not guaranteed a feasible solution. The complexity of this algorithm is $O(n^3)$.

In real-life, variety of problem domain exists which leads the consideration of IR in our work. We have seen in our obtained Pareto front that for a particular range of diameter, IR works better than other few heuristics for Euclidean instances and it supports that IR works better for large diameter.

3.3 Randomized Greedy Heuristics or Randomized Tree Construction

Possessing the complexity of a reduced factor of *n* in comparison to other existing heuristics [21], *Randomized Greedy Heuristics* (RGH) provides randomness by choosing the start vertex and all subsequent vertices at random from the set of vertices which are not yet in the spanning tree. But the connection of new vertex to the tree is yet greedy; it always connects lowest-weight edge between new vertex and tree-eligible vertex whose depth is less than *D*/2. This heuristic is named as *Randomized Tree Construction* (RTC) in [10] and as *Randomized Greedy Heuristics* in [21]. RTC has the complexity of $O(n^3)$. RGH is a well-known heuristics for generating better solutions (i.e. MSTs) for low-range diameter on Euclidean instances for BDMST problem. We consider RGH for both Euclidean and non-Euclidean instances in our bi-objective MST problem to easily visualize the solution characteristics over different diameter range through obtained Pareto front.

3.4 Center Based Tree Construction

Center Based Tree Construction (CBTC) [10] is a variation of RGH/RTC and proposed by Julstrom [10]. Here, spanning tree is built by centering a chosen starting vertex and then adding subsequent vertices with the satisfaction of given depth bound.

3.5 Center Based Recursive Clustering

Nghia and Binh [20] proposed *Center Based Recursive Clustering* (CBRC) [20] heuristic which is based on RGH/RTC, only the concept of center changes in each level with the growing spanning tree. This algorithm recursively cluster the vertices of the graph, in that every in-node of the spanning tree is the center of the sub-graph. These sub-graphs are composed of nodes in the sub-tree rooted at that in-node. The inspiration of introducing CBRC came from the observation [1, 21] that good solutions for BDMST problem have "*star-like structures*".

In most of the previous work, Euclidean data sets have been considered and little efforts have been done with non-Euclidean instances. However, it is shown that the conclusions drawn on Euclidean data set are not necessarily similar for non-Euclidean instances.

4. EVOLUTIONARY APPROACH

Evolutionary algorithms show better solutions for both BDMST [3, 11, 21] and MOST problem [17, 19]. Researchers have shown that better solutions are obtained with the consideration of either random spanning trees (STs) or STs generated from a particular heuristic as initial population. Solution quality also depends on the encoding considered to represent STs. In an attempt, considering well-known edge-set encoding to represent STs, we obtained improved solutions for each heuristic by MOEA and genetic operators. Well-known MOEAs that do not require problem specific knowledge to the extent are Non-dominated Sorting Genetic algorithm II (NSGA-II) [7], Pareto Converging Genetic Algorithm (PCGA) [16], Strength Pareto Evolutionary Algorithm 2 (SPEA-2) [26], PAES [12] etc. Researchers revealed that in this particular case, the variations of solution quality for different MOEAs are negligible [18]. We have considered a steady-state MOEA, PCGA with monitoring of convergence by rank histogram. Regarding MOEA, it is a common concern that the solution should not stuck into local optima and it directs the consideration of multi-island scheme using Inter-Island rank histogram.

4.1 Evolutionary Operators

We have used one recombination operator and two simple mutation operator namely greedy edge replacement mutation and edge-delete mutation. We have modified the edge-delete mutation operator and recombination operator as proposed in [3, 17, 20, 21].

4.1.1 Recombination Operator

We have adapted well-known recombination operator proposed by Raidl and Julstrom [21]; it supports strong heritability by providing favor to select common parental edges into its offspring. In this work, the modified recombination operator generates offspring by considering a common parental edge as an initial edge; it provides a high probability of a common edge of parent to be in the offspring. Originally, this recombination operator is proposed to generate offspring-MSTs for a particular diameter constraint. Here, in BOMST problem, we have worked on wide range of diameter; therefore, discarded the strict diameter-constraint checking of original proposed operator. We have modified the diameter-constraint checking in original proposed operator to generate partially directed MSTs within diameter constraint.

We use the notations that a set F1 contains edges appearing in both parents, F2 contains set of edges appear in only in one parent; U represents set of non-tree vertices, C represents set of vertices included in tree i.e. tree-vertices. Two set A1 and A2 maintain set of edges from F1 and F2, respectively during the growing phase of spanning tree and are modified if the newly added vertex *v* from U satisfies diameter constraint. Until all non-tree vertices are considered as a tree-edge, set A1 is checked; if A1 set is non-empty, a random edge from A1 is chosen for growing spanning tree; otherwise, set A2 is checked. If both set A1 and A2 are empty, then only a random edge connecting vertices from set C and set U are considered.

F1 ← edges appearing in both parents ;
F2 ← edges appearing in only single parent ;
depth[v] ← *-1 ; v = 1,2,...*total_nodes *;*
T ← ∅ *;*
// Randomly chosen edge from F1
T ← *{(v0,v1)} ;*
C ← *{v0,v1} ;*
U ← *V – {v0,v1} ;*
depth[v0] ← *0 ;*
depth[v1] ← *0 ;*
A1 ← edges from F1 incident on *{v0,v1} ;*
A2 ← edges F2 incident on *{v0,v1} ;*
// Next part is done iteratively for rest of the nodes to include in T
while U ≠ ∅ **do**
  if A1 ≠ ∅ **then**
    pick an edge *{(u,v)}* ∈ A1 at random ;
    T ← T ∪ *{(u,v)}* ;
    A1 ← A1 – *{(u,v)}* ;
  else
    if A2 ≠ ∅ **then**
      pick an edge *{(u,v)}* ∈ A2 at random ;
      T ← T∪ *{(u,v)}* ;
      A2 ← A2 – *{(u,v)}* ;
    else
      pick *u* ∈ C at random ;
      pick *v* ∈ U at random ;

```
        T ← T ∪ {(u,v)} ;
      end if
    end if
    if v ∈ U then
      U ← U – {v} ;
      depth[v] ← depth[u] + 1 ;
      if depth[v] < ⌊ D/2 ⌋ then
        A1 ← A1 ∪ all edges from F1 incident on v ;
        C ← C ∪ v ;
      end if
    else
      if u ∈ U then
        U ← U – {u} ;
        A1 ← A1 ∪ all edges from F1 incident on u ;
        A2 ← A2 ∪ all edges from F2 incident on u ;
        C ← C ∪ u ;
      else
        T ← T – {(u,v)} ;
      end if
    end if
end while
```

Algorithm 1**: Recombination operator.**

4.1.2 Mutation Operators

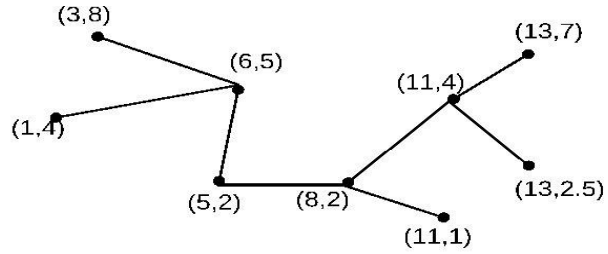
The EA applies two mutation operators. Greedy edge replacement mutation generates valid tree. But, the edge-delete mutation operator only randomly deletes an edge from an individual to reduce the probability of occurrence of that edge into the offspring; thus, this mutation operator never produces a valid tree. Therefore, mutated individual using edge-delete mutation operator always takes part in recombination with other valid tree individual. We delete an edge from a valid tree to incorporate a poor solution with the expectation of generating better solution in next generations; thus, the selection of the probability of occurrence for this mutation operator should be much less than the greedy edge replacement operator.

4.1.2.1 Greedy Edge Replacement Mutation Operator

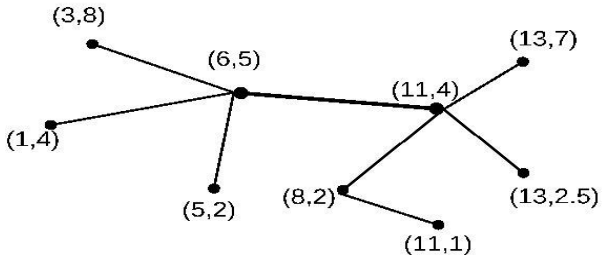
In greedy edge replacement mutation operation, an edge from a valid spanning tree is deleted randomly and replaced with a lowest cost edge that forms a bridge between two separated sub-tree [21]. This mutation operation is simple and has a complexity of $O(n)$.

4.1.2.1. Modified Edge Delete Mutation Operator

In modified edge delete mutation operator, we randomly delete an edge from a spanning tree to reduce the probability of that edge during the generation of offspring. The probability of occurrence of this operator is considered very small with the hope that a new unexpected edge will be considered in offspring to give better solutions in successive generations. The complexity of this operation is $O(1)$. This mutation operator is easy and simple to implement; only one random delete operation is needed and then that invalid ST is chosen as one of the parent to perform the recombination operation.



a) A random individual



b) Mutated individual

Figure 2: Greedy edge replacement operation applied on a randomly selected individual: a randomly selected edge is deleted and replaced with a minimum cost edge connecting two disconnected sub-tree within diameter bound = 5.

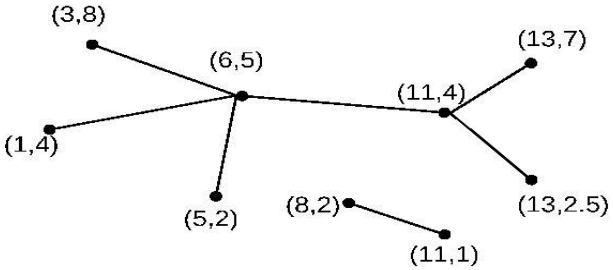


Figure 3: Modified Edge Delete Mutation operation applied on a randomly selected individual: a randomly selected edge is deleted to reduce the probability to consider that edge for newly generated offspring.

The good qualities of these operator, in terms of heritability is demonstrated by the empirical results presented later in this paper.

5. RESULTS

5.1 Problem Instances

Without loss of generality, we have considered complete graphs for both Euclidean and non-Euclidean problem instances; the standard benchmark BDMST problem instances (as used in [10, 21, 25]) are used for Euclidean problem set, downloadable from http://tomandtun.googlepages.com/phd. For 50 node non-Euclidean problem instances, weights are randomly generated within range [0.01, 0.99] and for 100 node, we have considered the instances used by Julstrom [10]. Due to dissimilar nature of input problem instances, nature of solutions varies over wide range of diameter for Euclidean and non-Euclidean problem instances.

5.2 Pareto front solutions from Heuristics

We have performed operation on first five instances of the input for both Euclidean and non-Euclidean complete graphs; then, obtained Pareto fronts for each instance considering OTTC, IR, CBTC, RTC, CBRC heuristics in one plot. Due to space limitation we choose only one random plot shown in Figure 1 for 100 node non-Euclidean instance.

5.3 Results from EA

We have applied the solutions across the Pareto front for each heuristic individually to the EA as initial population and obtained an improved Pareto front for each heuristic. Improved Pareto front plot for OTTC is shown in Figure 4, for 50-node 1*st* Euclidean problem instance. Figure 5 depicts the CBTC, RGH and improved RGH Pareto front obtained for a non-Euclidean instance. We have included few sampler plots in this paper due to space concern; performance measurement of heuristics and EA is easily visualized from any sampler instance plot. Solutions generated from RGH and CBTC heuristic, together provides the superior solutions for Euclidean instance; thus Pareto fronts generated for RGH and CBTC, are considered in each plot. On visualizing the plots we conclude that EA on OTTC solutions improve the solution set across complete range of diameter for Euclidean instances and generated improved Pareto front is closer to RGH Pareto front; whereas RGH provides best solutions for Euclidean instances within small range of diameter.

OTTC Pareto front is closer to CBTC Pareto front in higher diameter bound range. Similar nature of improved Pareto front is observed for IR and CBRC for Euclidean instances. Improved RGH Pareto front does not show much improvement on smaller diameter range; but, it provides better solutions for larger diameter range. Whereas improved CBTC Pareto front provides a smaller improvement over CBTC Pareto front. RGH generates poor solution set than OTTC and CBTC for non-Euclidean instances; CBTC and OTTC provide similar solutions with negligible drift and EA over OTTC solutions provide new solutions along with old solutions across the Pareto front. On applying EA over RGH solutions generates better solution set as shown in Figure 5.

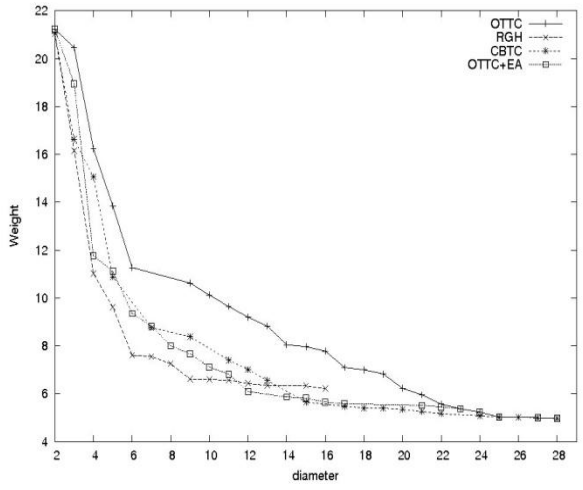


Figure 4: Pareto front obtained from OTTC, RGH, CBTC and OTTC+EA heuristics for 50 node Euclidean 1st instance. MOEA on OTTC Pareto front solutions improves the Pareto front over entire range of diameter.

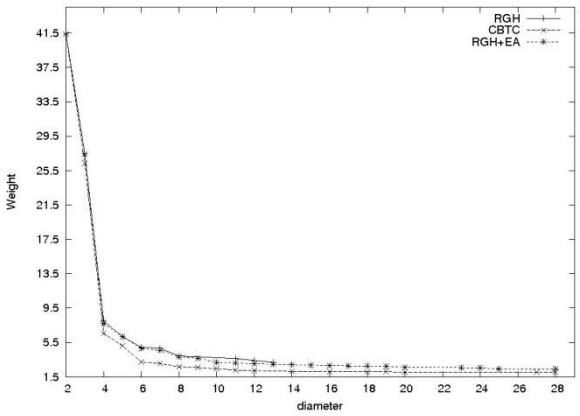


Figure 5: Pareto front obtained from RGH, CBTC and RGH+EA heuristics for 100 node non-Euclidean instance. EA on RGH Pareto front solutions extends the Pareto front over large diameter range.

Table 1 depicts the improved best solution for BDMST problem from various well-known heuristics using EA for Euclidean instances. For node size n = 50, diameter bound D is set to 5 and for n = 100, D = 10; the benchmark diameter bound is set for particular node size as used in [3, 10, 20, 21, 25]. The average weight of solutions within particular diameter bound is also improved by EA for all Euclidean and non-Euclidean instances. Considering a directed initial population (as generated from each heuristics) EA operators try to generate new improved solutions in successive generations across the complete range of diameter bound.

Along with easily visualizable Pareto front plots, we have considered convergence metric [6], spread [5], *C*-measure [27] and hypervolume/*S*-metric [13] to assess the performance of MOEA. The reference set is considered as the Pareto front obtained from all EA solutions.

5.3.1 Avoiding Local Optimal

We have used Kumar and Rockett [15] proposed rank-histogram scheme to assess the movement of solution-front generated by heuristics towards convergence. It is a common concern that whether the obtained solution-set is near to the optimum Pareto front or not. In several problems continuing the EA search where search gets trapped in local optima is a waste of computational resources and time [16]. We have considered multi-island scheme using Inter-Island rank histogram [16]. Therefore, we perform multiple runs of the algorithm and combine the genetic information from different runs to extract the Pareto front. Table 2 shows different metric values (Convergence, Spread and Hypervolume) for different tribes and their combination for RGH on 50-node Euclidean 1*st* instance. The hypervolume and spread metrices show improved performance by combining tribes. We have considered Inter-Island rank histogram scheme to calculate the empirical results and all metric values for improved Pareto fronts on each heuristics.

5.3.2 Convergence

Table 3 contains average convergence value obtained from 1st five standard instances of 50 node Euclidean and 100 node non-Euclidean problem instances. EA gives better convergence over heuristics for both Euclidean and non-Euclidean problem instance. Convergence of EA on Euclidean CBTC Pareto front solutions is increased due to its superior solutions across larger diameter range includes into reference

Pareto front and negligible improvement is achieved. Convergence of EA on non-Euclidean RGH Pareto front solutions is increased due to increasing number of solutions on improved Pareto front in comparison to Pareto front obtained from RGH heuristic and for higher range EA achieved near-equal solutions to reference Pareto front.

5.3.3 Spread

Spread measurement implies the diversity with respect to a reference front. Table 4 shows the average spread measurement for OTTC, IR, RGH, CBRC and CBTC Pareto fronts and improved Pareto fronts. We have considered reference front as the Pareto front which is generated from large EA solution set consisting all improved heuristics solution. Diversity is improved for Pareto fronts generated by MOEA; degraded negligible diversity value in few cases depicts the reduced number of solution set in improved Pareto front.

5.3.4 C-Measure

C-measure calculates the percentage of a Pareto front solutions which are dominated by a reference front. C-measure is not symmetric. If *A* is the comparator Pareto front and *B* is Pareto front generated from MOEA, $C(A,B) = 0$ implies that no points in *A* dominates any point in *B*; $C(A,B) = 1$ implies that all points in *B* are dominated by, or equivalent to point lies in front *A*. Pareto fronts generated by the conventional heuristics *A* are completely dominated by, or equivalent to Pareto fronts generated by the MOEA *B*; thus, $C(A,B)$ value is 0. Whereas $C(B,A)$ value range from 0.55 to 1.00 for all such all Euclidean and non-Euclidean instances.

5.3.5 Hypervolume

Table 5 contains average S-measure/ hypervolume [13] of the multi-dimensional region which is enclosed by obtained Pareto fronts. Hypervolume measurement depends on selection of a particular reference point. In this work, we have chosen a particular reference point as a maximum obtained solution for each objective for a particular instance by accumulating obtained Pareto fronts from heuristics and improved Pareto fronts using EA for that instance. The hypervolume depicted in table 5 is calculated by averaging the hypervolume obtained from 1*st* five instances for 50 and 100 node Euclidean and non-Euclidean problem data set. RGH shows superior hypervolume for Euclidean instances; whereas, CBTC and OTTC show close superior hypervolume for non-Euclidean instances. This metric shows the superiority of obtained MOEA Pareto fronts over Pareto fronts obtained from different heuristic for both Euclidean and non-Euclidean instances. The improvement of Pareto fronts obtained for OTTC and CBTC for non-Euclidean instances using MOEA is smaller for non-Euclidean instances in comparison to other improved Pareto fronts obtained for both Euclidean and non-Euclidean instances.

6. CONCLUSIONS AND FUTUREWORK

In this paper, We have recasted a few known BDMST heuristics for BOMST problem and concluded with Pareto front plots that none of the heuristics shows superior solution across the complete range of diameter bound for both Euclidean and non-Euclidean instances. Again, nature of superiority of heuristics over different region varies on type of instances (i.e. Euclidean or non-Euclidean). On applying MOEA approach on each heuristics Pareto front solutions, the solution-set for each heuristics is improved; thus, improved Pareto front is generated. We have considered multi-island rank histogram scheme to avoid from trapping into local-optimal solution-set. MOEA improves the solutions generated from each heuristics across the complete range of diameter which in turn improves the BDMST solutions for a particular diameter constraint.

6. REFERENCES

[1] Abdalla, A. 2001. Computing a Diameter-Constrained Minimum Spanning Tree. Doctoral thesis, The School of Electrical Engineering and Computer Science, University of Central Florida, Florida.

[2] Bala, K., Petropoulos, K., and Stern, T. E. 1993. Multicasting in a linear lightwave network. In IEEE INFOCOM'93, 1350–1358.

[3] Binh, H. T. T., McKay, R. I., Hoai, N. X., and Nghia, N. D. 2009. New Heuristic and Hybrid Genetic Algorithm for Solving the Bounded Diameter Minimum Spanning Tree Problem. ACM.

[4] Bookstein, A. and Klein, S. T. 1996. Compression of correlated bit-vectors. Information Systems, 16:110–118.

[5] Deb, K. 2001. Multi-Objective Optimization using Evolutionary Algorithms. Jon Wiley Sons, Chichester.

[6] Deb, K. and Jain, S. 2002. Running performance metrices for evolutionary multiobjective optimization. SEAL'02, 13–20.

[7] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 6(2):182–197.

[8] Deo, N. and Abdalla, A. 2000. Computing a diameter-constrained minimum spanning tree in parallel. Lecture Notes in Computer Science, 1767:17–31.

[9] Garey, M. R., and Johnson, D. S. 1979. Computers and interactibility: A guide to the theory of NP-Completeness. W. H. Freeman, New York.

[10] Julstrom, B. A. 2009. Greedy heuristics for the bounded diameter minimum spanning tree problem. JOURNAL of Experimental Algorithmics (JEA), 14:1.1:1–1.1:14.

[11] Julstrom, B. A., and Raidl, G. R. 2003. A Permutation Coded Evolutionary Algorithm for the Bounded Diameter Minimum Spanning Tree Problem. In GECCO'03: Proceedings of the Genetic and Evolutionary Computation Conference, 2–7.

[12] Knowles, J. D. and Corne, D. W. 2000. Approximating the non-dominated front using the Pareto achieved evolution strategy. Evolutionary Computation, 8(2):149–172.

[13] Knowles, J. D. and Corne, D. W. 2002. On metrices for comparing nondominated sets. In Congress Evolutionary Computation (CEC'02), volume I, 711–716, Honolulu, Hawaii, IEEE.

[14] Kumar, R., Bal, B. K., and Rockett, P. I. 2009. Multiobjective Genetic Programming Approach to Evolving Heuristics for the Bounded Diameter Minimum Spanning Tree Problem. GECCO'09.

[15] Kumar, R. and Rockett, P. I. 1997. Assessing the convergence of rank-based multiobjective genetic algorithms. In IEE/ IEEE 2ndInt. Conf. Genetic

Algorithms in Engineering Systems: Innovations Applications (GALESIA-97), 19–23, Glasgow, UK. IEE Conference Publication No. 446.

[16] Kumar, R. and Rockett, P. I. 2002. Improved sampling of the Pareto-front in multiobjective genetic optimization by steady-state evolution: A Pareto converging genetic algorithm. MIT press, 10(3):283–314.

[17] Kumar, R. and Singh, P. K. 2007. On quality performance of heuristic and evolutionary algorithms for biobjective minimum spanning trees. In GECCO '07: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, 2259, New York,m USA. ACM Press.

[18] Kumar, R., Singh, P. K., and Chakrabarti, P. P. 2005. Multiobjective EA Approach for Improved Quality of Solutions for Spanning Tree Problem. In EMO-05: 3rd International Conference on Evolutionary Multi-Criterion Optimization, 811–825, Guanajuato, Mexico. Springer Berlin, Heidelberg.

[19] Neumann, F. and Wegener, I. 2006. Minimum spanning trees made easier via multi-objective optimization. Natural Computing, 5:305–319.

[20] Nghia, N. D. and Binh, H. T. T. 2008. Heuristic Algorithms for Solving Bounded Diameter Minimum Spanning Tree Problem and Its Application to Genetic Algorithm Development. Advances in Greedy Algorithms, 586.

[21] Raidl, G. R. and Julstrom, B. A. 2003. Greedy heuristics and an Evolutionary Algorithm for the Bounded-Diameter Minimum Spanning Tree Problem. SAC'03:18th ACM Symposium on Applied Computing, 747–752.

[22] Raymond, K. 1989. A tree-based algorithm for distributed mutual exclusion. ACM Transactions on Computer Systems, 7:61–77.

[23] Saha, S., Aslam, M., and R. Kumar. 2010. Assessing the Performance of Bi-Objective MST for Euclidean and Non-Euclidean Instances. In Proceedings of International Conference of Contemporary Computing (IC'03), volume I, 229–240, Noida, India. Springer.

[24] Saha, S. and Kumar, R. 2011. Improvement of Bounded-Diameter MST Instances with Hybridization of Multi-Objective EA. In Proceedings of International Conference on Communication, Computing & Security (ICCCS'11), Odissa, India. ACM.

[25] Singh, A. and Gupta, A. K. 2007. Impoved Heuristics for the Bounded-Diameter Minimum Spanning Tree Problem. JOURNAL Soft Computing, 11:911–921.

[26] Zitzler, E., Laumanns, M., and Thiele, L. 2002. SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In Evolutionary Methods for Design, Optimization and Control, 95–100, CIMNE, Bercelona, Spain.

[27] Zitzle, E. and Thiele, L. 1999. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. IEEE Transactions on Evolutionary Computation, 3(4):257–271.

Table 1: Results of OTTC, IR, RGH, CBRC and CBTC on 5 Euclidean instances of BDMST (best solutions generated from heuristics and EA) problem of size 50 and 100. [The subscript over the size indicates a different instance.]

Instance	D	OTTC		IR		RGH		CBRC		CBTC	
$Size_{Instane}$		*Best*	$Best_{EA}$	*Best*	$Best_{EA}$	*Best*	$Best_{EA}$	*Best*	$Best_{EA}$	*Best*	$Best_{EA}$
50_1	5	13.84	11.12	11.54	11.21	9.62	9.59	12.61	11.80	10.88	10.80
50_2	5	14.19	12.42	12.89	12.81	8.80	8.80	13.09	11.00	9.47	9.47
50_3	5	12.53	12.53	11.97	11.14	10.17	8.91	11.00	10.46	11.03	11.03
50_4	5	11.04	11.04	11.56	10.72	8.35	8.23	10.85	9.83	10.10	10.10
50_5	5	13.04	12.30	11.34	11.00	9.13	9.00	13.98	10.98	11.13	11.13
100_1	10	18.79	11.88	11.04	10.82	9.28	8.84	17.26	14.45	14.22	12.01
100_2	10	17.69	14.78	10.45	10.10	9.51	9.32	17.43	11.24	13.61	11.70
100_3	10	19.90	15.75	10.77	9.92	9.49	9.01	22.64	17.46	13.66	11.19
100_4	10	17.64	15.33	11.16	10.69	9.65	9.47	18.42	14.23	14.05	12.13
100_5	10	16.63	16.56	11.55	11.09	9.81	9.44	17.37	13.88	14.50	12.85

Table 2: Different metric values (Convergence, Spread and Hypervolume) for different tribes and their combination for RGH on 50-node Euclidean 1st Instance. The hypervolume and spread metrices show improved performance by combining tribes.

Tribe	Convergence	Spread	Hypervolume
Tribe1	0.0119	0.6309	378.44
Tribe2	0.0119	0.6299	378.44
Tribe3	0.0110	0.6297	380.23
Combined	0.0116	0.6302	379.04

Table 3: Average convergence of OTTC, IR, RGH, CBRC and CBTC Pareto fronts on 5 Euclidean and non-Euclidean instances of BDMST (Pareto front solutions generated from heuristics and EA and the reference set is considered as the Merged Pareto front generated from all EA solutions) problem of size 50 and 100 respectively.

Instance $size_{type}$	Convergence on	OTTC	IR	RGH	CBRC	CBTC
50_E	Heuristic	0.0801	0.0425	0.0134	0.0458	0.0239
50_E	EA	0.0533	0.0259	0.0078	0.1187	0.0290
100_{nonE}	Heuristic	0.0126	0.0649	0.0303	0.1484	0.0074
100_{nonE}	EA	0.0107	0.0578	0.0308	0.0693	0.0038

Table 4: Average Spread of OTTC, IR, RGH, CBRC and CBTC Pareto fronts on 5 Euclidean and non-Euclidean instances of BDMST (Pareto front solutions generated from heuristics and EA and the reference set is considered as the Merged Pareto front generated from all EA solutions) problem of size 50 and 100 respectively.

Instance $size_{type}$	Spread on	OTTC	IR	RGH	CBRC	CBTC
50_E	Heuristic	0.289	0.456	0.711	0.655	0.423
50_E	EA	0.320	0.455	0.603	0.487	0.427
100_{nonE}	Heuristic	0.940	0.793	0.955	0.736	0.940
100_{nonE}	EA	0.938	0.763	0.937	0.760	0.950

Table 5: Average Hyper-Volume of OTTC, IR, RGH, CBRC and CBTC Pareto fronts on 5 **Euclidean and non-Euclidean instances of BDMST (Pareto front solutions are generated from heuristics and EA and the reference point for each individual instance is considered by choosing the maximum co-ordinates. This coordinates are the maximum diverse point which is generated by cumulating all heuristics and EA solutions for an instance.) problem of size 50 and 100 respectively.**

Instance size$_{type}$	**Hypervolume for**	**OTTC**	**IR**	**RGH**	**CBRC**	**CBTC**
50_{E}	Heuristic	256.14	272.09	280.42	222.58	278.45
50_{E}	EA	273.45	278.04	289.97	276.20	282.11
100_{E}	Heuristic	1109.20	1190.49	1211.90	948.95	1192.43
100_{E}	EA	1175.83	1209.19	1245.82	1186.93	1212.97
50_{NonE}	Heuristic	320.11	286.80	308.26	254.20	320.50
50_{NonE}	EA	320.82	295.51	312.77	300.04	320.86
100_{NonE}	Heuristic	1014.67	944.95	986.71	803.84	1016.42
100_{NonE}	EA	1015.48	958.21	999.55	952.69	1017.13