Tutorial
on

Evolutionary Multiobjective Combinatorial Optimization (EMCO)

Rajeev Kumar
Department of Computer Science and Engineering
Indian Institute of Technology (IIT) Kharagpur

rkumar @ cse.iitkgp.ernet.in
rajeevkumar.cse @ gmail.com
www.facweb.iitkgp.ernet.in/~**rkumar**

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-131-6/08/07.

GECCO 2008
July 13, 2008 @ AM 2

About Me

Professor, Computer Sc
IIT Kharagpur

Teaching:

- Compiler Construction
- OO Lang. Implementation
- Prog. Lang. & Methodology
- Object-Oriented Computing
- Trustworthy Computing
- Software Engineering
- Data Struct. & Algorithm
- Multimedia & Embedded Sys.
- Compt. Intelligence . . .

Research Interests:

- Evo. Algo & Combinatorial Optimization
- Prog. Lang. & Software Engineering
- Embedded System Soft Tools
- Multimedia System & QoS
- Machine Intelligence

Education:

- Sheffield, UK
- Roorkee
- Allahabad

Worked for:

- IIT KGP / KAN
- National Germany
- BITS Pilani
- DST and DRDO

Recent Projects:

- MHRD, India
- Microsoft, USA
- National, Germany

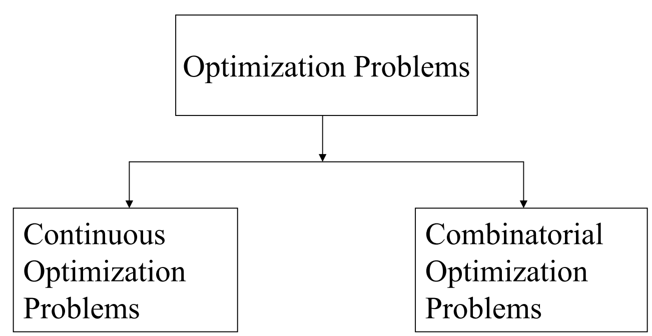
Rajeev Kumar
rkumar AT cse.iitkgp.ernet.in
www.facweb.iitkgp.ernet.in/~**rkumar**

Tutorial Overview

- Optimization
- Combinatorial Optimization
- Single Objective Combinatorial Optimization
- Multiobjective Combinatorial Optimization
- Issues and Challenges
- Hybridization of MOEAs
- Case Studies

Optimization

refers to the design and operation of a system or process to make it as good as possible in some defined sense.



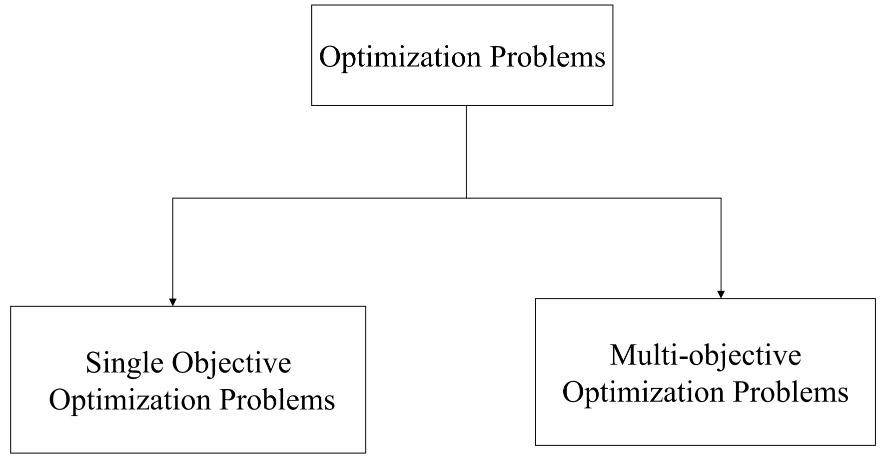
Combinatorial Optimization

refers to the optimization problem where solution vector is discrete in finite set of feasible solutions.

Continuous Optimization

As opposed to discrete optimization, the variables used in the objective function can assume real values, e.g., values from intervals of the real line.

Combinatorial Optimization Problems



Single Objective Optimization (Problem Definition)

Maximize / Minimize

$$f(x)$$

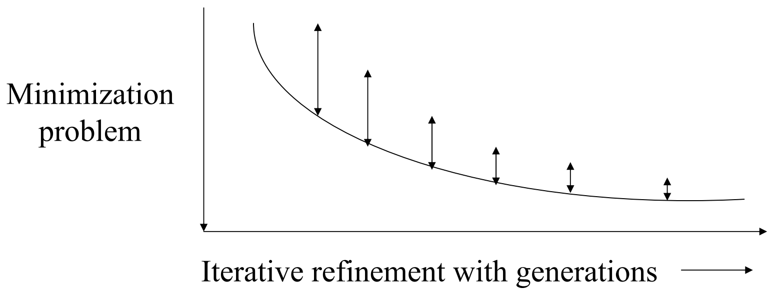
Subject to

$$
\begin{aligned}
& g_j(x) \geq 0, && j = 1, 2, \ldots, j \\
& h_k(x) = 0, && k = 1, 2, \ldots, k \\
& x_i^{(L)} \leq x_i \leq x_i^{(U)} && i = 1, 2, \ldots, n
\end{aligned}
$$

Single Objective Optimization (What to do?)

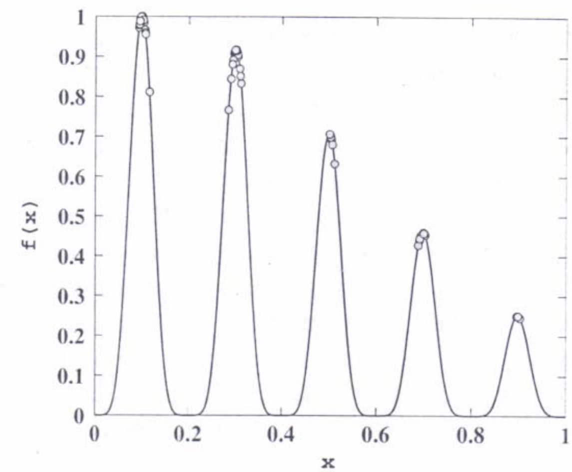
- Solution is clearly defined as the search space is often totally ordered.
- We simply seek one best solution that optimizes the sole objective function (except multimodal optimization problems).

Single Objective Space



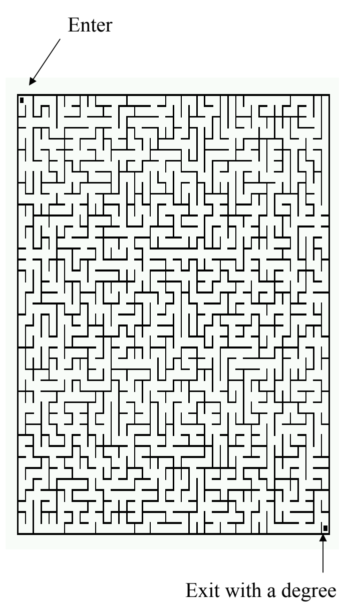
Performance monitoring and termination criteria both are **trivial**.

Multimodal Function



A Sample Maze . . .

- What is the goal ?
  - Exit with a degree ? Y/N
    - Have a ***Decent*** degree ?
    - Degree with ***minimal*** Cost ?
      - Attending to teaching etc.
      - Self efforts (study/practices)
      - Collaborations,
      - Expenditures.
- Multiple objectives

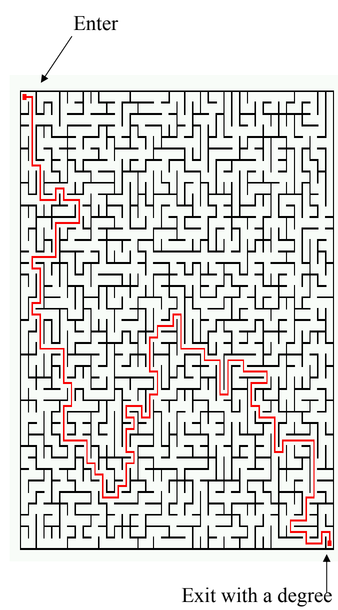


Optimized Maze

- No solution
- Single solution
- Multiple solutions
  - DM picks one.

Combinatorial (discrete) Optimization/decision Problem

- variables are ***discrete***.

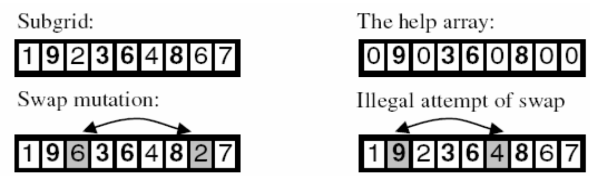


Sudoku Puzzle

- How to solve ?
- How to ***generate*** Sudoku with ***Different*** complexity levels.
- Constraint Satisfaction Problem
  - Each row, col. and 3x3 grid has each digit from 1 to 9
  - Given digits must remain in positions

Sudoku Puzzle :: Solving with EA

Mutation



Multiobjective Combinatorial Optimization (MOCO) problems

Definition

minimize/maximize	$f_m(x)$	$m = 1, 2, \ldots, M$
	$g_k(x) \le c_k$	$k = 1, 2, \ldots, K$
	$x_i^{(L)} \le x_i \le x_i^{(U)}$	$i = 1, 2, \ldots, n$

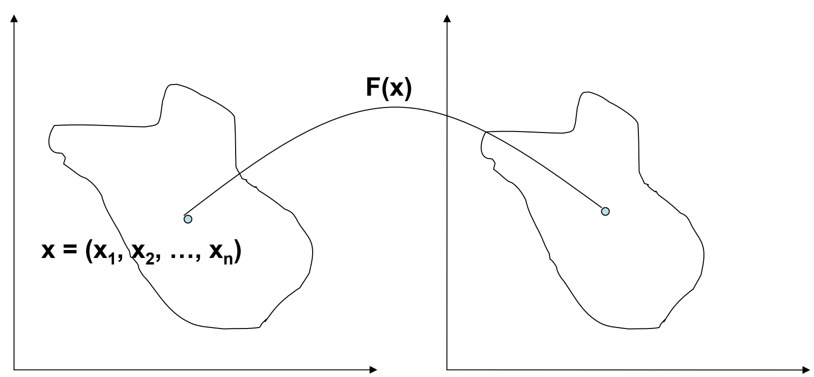
where $x = (x_1, x_2, \ldots, x_n)$ is discrete solution vector in X, which is a finite set of feasible solutions.

Objective vector $F(x) = (f_1(x), f_2(x), \ldots, f_m(x))$ maps solution vector (x) in decision space to objective space for $m \ge 2$.

There is *no single* solution to the problem instead, we get a set of solutions known as Pareto-optimal set.

MOCO problems . . .

Decision/Search space **Objective space**



MOCO problems . . .

Characteristics

- We desire to get a set of solutions known as Pareto-optimal set.
- A aggregation of objectives through weighted sum finds only the supported optimum solutions and not all the solutions as MOCO deals with *discrete, non-continuous* problems.
- Any efficient method to find all the Pareto-optimal solutions may not be possible as the size of the Pareto-optimal set usually *grow exponentially* with the problem size.
- Search space further adds to the complexity as it is only partial ordered.
- Most MOCO problems are NP-hard problems.

MOCO problems . . .

Solution methodologies

- Exact methods
  - May solve only small problems
  - Not expendable
- Heuristics
  - Usually problem specific
  - Finds local optimal set instead of global
- Metaheuristics
  - General problem solver
  - Explore and exploit the search space in a better way

MOCO problems . . .

Solution methodologies (Metaheuristics)

Non evolutionary

Evolutionary

Non population based

Population based

Population based methods look for global convergence as

- Whole population contributes in the evolutionary process.
- Population and genetic operators combine principles of cooperation and self adaptation.
- Generation mechanism is parallel along the frontier.

Multiobjective Evolutionary Algorithms

General purpose search and optimization tool that mimics natural evolution process and aims to search whole solution space and provide a set of feasible results corresponding to extreme values of objectives.

Working of MOEA at abstract level

generate a set of feasible solutions (initial population)
while stopping criteria is not satisfied do
    select
    crossover
    mutate
output a set of optimal results

Additional Issues in Multiobjective Optimization

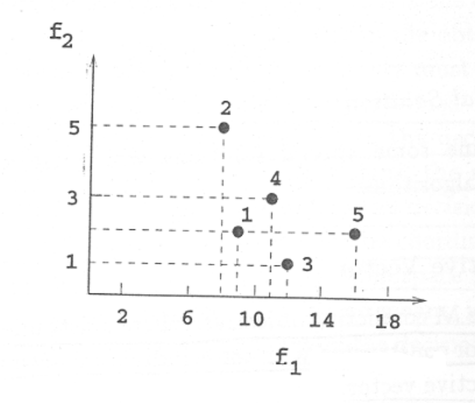
- A set of optimal solutions, known as Pareto-optimal set/ Pareto-front, instead of a single solution,
- Search space is not often totally ordered but only partially ordered.
- Achieving and monitoring convergence towards true Pareto-front,
- Achieving Diversity along Pareto-front, and
- Avoiding local convergence.

Pareto-dominance (Definition)

f_i dominates f_j if and only if

$f_{mi} \leq f_{mj}$ for all m and

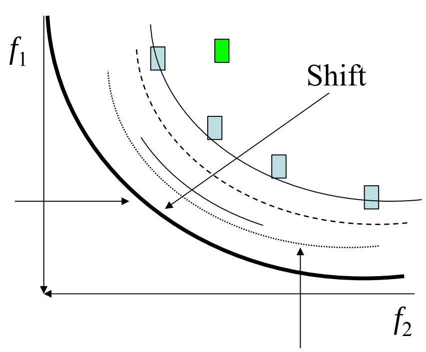
$f_{mi} < f_{mj}$ for some (at least one) m



Multi - Objective Space . . .

Minimization problem
f_1 and f_2

Actual
Pareto-front



Obtained
Pareto-front at '*t*'

Challenge I : Extent

Challenge II : Diversity

Challenge III : Convergence

Drawbacks of Classical Methods

- Some techniques are sensitive to the shape of pareto-optimal front.
- Problem specific knowledge may be required which may not be available.
- Convergence to an optimal solution depends upon chosen initial solution.
- An algorithm efficient in solving one problem may not be efficient in solving other problem.
- These are not efficient for problems having discrete search space
- Most algorithms tend to get stuck at suboptimal solution.
- Cannot be used efficiently on parallel machines.

Evolutionary Algorithms

- Suitable for ***Search***, ***Optimization, and MI***
- ***Inspired*** from ***Biological*** phenomenon
  - Set of Population (rather a single point search),
  - Population evolves through (***superior***) generations,
    - Productive Operators for children
      - Crossover (inherit from parents)
      - Mutation (Own properties)
    - Survival of the fittest
  - A multipoint search leads to (near-) optimal sol.
- Randomized, Stochastic, Meta-heuristics. . .
- Do not need much problem specific knowledge. . .

They are not Bio-Informatics or Bio-computers.

Primary Reasons for their Success

- Broad Applicability
  - works with the coding of the decision variables, instead of variables themselves.
  - uses only objective function values, not derivatives or other auxiliary knowledge.
- Global Prospective
  - work on a set of populations and uses synergy between the solutions.
  - uses probabilistic transition rules, not the deterministic rules, to guide the search.
- It can be conveniently used on parallel systems.

EA : A Brief Detour

- Randomized Search Algorithm mimicking evolutionary process
- Works on Iterative Refinement scheme like many other techniques, e.g., Hill - climbing etc.

```
Initialize(Population)
While ( ! Termination) {
    Produce (New Individuals) // EvoOpr
    Insert (Into Population)
}
```


EA :: Can do?

- Generic problem solving strategy,
- Most problems can be attempted through EAs
- Excellent at getting some solution w/o much problem specific knowledge,
- Expect to get ***near-optimal*** solution without any approximation bounds,
- Expect to get ***superior solution*** than any other known techniques, and
- Improve iteratively the solution quality

EA :: Can Not or Difficult to do?

- Do not aim for optimal solutions through EAs,
- Very difficult to find time-bounds and approximate solution quality bounds,
- At times, difficult to ***recast*** the problem into genetic/evolutionary domain,
- At times, difficult to design productive operators
- More efforts to translate quick/early gains into better solutions.

Learning from Experiences (1995s)

- While working on a ***partitioning*** problem taken from a RWA
  - I thought of entering into the world of fantasy, because
- Try Evolutionary Algorithms (EA) when nothing else works,
- With a little problem-specific knowledge, one gets good performance

Stage I : Recast the problem into genetic domain.

Stage II : Selection & Tuning of a couple of genetic operators.
// A bit of clever work

Within a few days of work, I was thrilled to realize that it does work.

Black Box Optimization

The very next day – it was a **catastrophe** . . .

Challenge I :

How to know that I was advancing ?

Performance monitoring

Challenge II :

How to know that I had achieved ?

Convergence

- Did not aim to have EA as a ***Testing*** tool.
- Selected EA as the ***Solution*** tool ?

What difference does this make ?

EA :: A Reality Check . . .

- Difficult to assess quality of solutions,
- Adopt Hybridization with others, e.g., local search
- Incorporate as much problem specific knowledge as you can into representation and operators,
- Use hybridization to learn and improve each other, and
- . . .

3 Classes of problems . . .

- One, mostly Analytical functions : ***known***
  - Simple, Multi-modal . . .
- Second, hard-class of ***known*** problems
  - Solutions are ***verifiable***
  - E.g., MST, Knapsack . . .
- Third, hard-class of ***unknown*** problems
  - solutions are NOT verifiable, directly.
  - E.g., TSP, Network, Partitioning & many other problems . . .

Hard Problems

- Computational problems fall into two categories:
  - **Decision problem**
    - Output: Yes/No
  - **Optimization problem**
    - Output: Solution with max./min.
- *Polynomial-time* algorithms do not exist:
  - If the problem is ***not hard***, someone can find it.
  - If the problem is ***really hard***, other smart people cannot find it either.
- It is hard to find a needle in a haystack,
- It is harder to say that there is ***no* needle** in a haystack.

Biobjective 0-1 Knapsack Problem

Problem Definition

We use a biobjective 0-1 Knapsack problem consisting of a single knapsack.

For a knapsack of n items with positive
weights $w_1, w_2, \ldots, w_n$,
profits of $p_1, p_2, \ldots, p_n$ and
decision variables $x_1, x_2, \ldots, x_n$
where for each $1 \le i \le n$, x_i is either 0 or 1

We aim to maximize $P = \Sigma^n_{j=1} p_j x_j$ and minimize $W = \Sigma^n_{j=1} w_j x_j$ and find full solution front.

It has been shown NP-hard problem for arbitrary value of p_j and x_j as Pareto-optimal set grows exponential to n.

Biobjective 0-1 Knapsack Problem . . .

Motivation

A good heuristic is available that arranges the items in descending order of their profit to weight ratio and generate a subset of n solutions.

Another algorithm of dynamic programming paradigm is available that generate good solutions in whole range of solutions.

We aim to solve the problem using MOEA to judge the efficacy and quality of solutions.

Biobjective 0-1 Knapsack Problem . . .

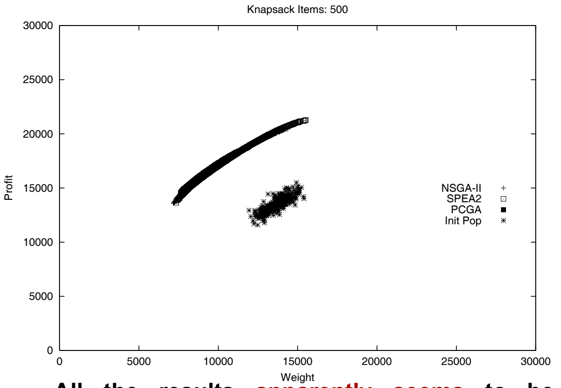
MOEA Solution

- **Pareto-ranking based MOEA**
- **Complete Elitism**
- **Parameter less diversity preservation**
- ***Encoding* of chromosome : bit encoding**
- ***Crossover operator* : 2-point crossover**
- **Mutation operator : Bit mutation**

1 0 0 1 0 1 1 0 1 1 0 1
chromosome

Biobjective 0-1 Knapsack Problem . . .

MOEA Results



All the results apparently seems to be very promising. Initial population is also shown here.

Biobjective 0-1 Knapsack Problem . . .

Improving MOEA Results

We observed that solution in the Pareto-front are heavily skewed towards 0s in left hand side and 1s towards right hand side.

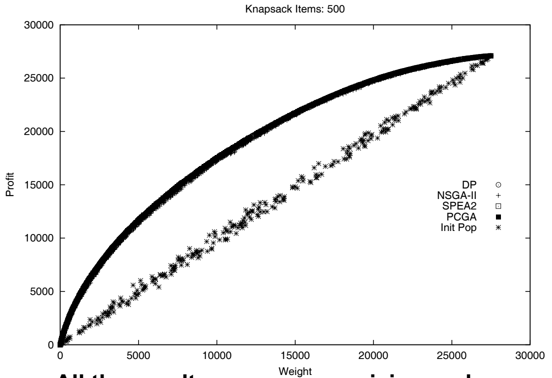
Further, we observed that MOEA did not generate these skewed solutions. It was due to the fact the 0s and 1s have been generated randomly in the chromosome.

The solutions are concentrated in the middle portion only and not spread in the whole range of solutions.

We inject two special chromosomes one with all 0s and other with all 1s and other chromosomes have randomly generated fix number of 1s and 0s.

Biobjective 0-1 Knapsack Problem . . .

Improving MOEA Results



All the results are very promising and comparable to results of heuristics. Initial population is also shown here.

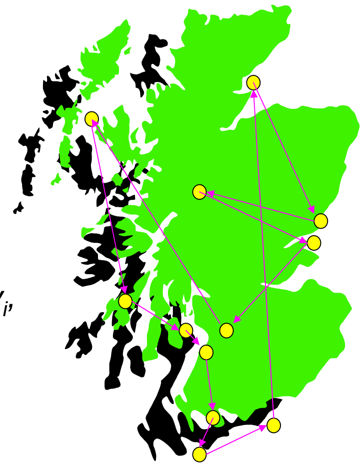
Biobjective 0-1 Knapsack Problem . . .

Important findings

- **Had it not been known to us about the solution front by other algorithms we would have taken MOEA results as very promising.**
- **With the knowledge of solution front we incorporated the problem-specific knowledge in the evolution process of MOEA and got comparable results.**
- **It is a paradox that we must know the solution set in advance to effectively solve the problem.**

TSP

Hamilton circuit : a circle uses every vertex of the graph exactly once except for the last vertex, which duplicates the first vertex. (NP-complete)



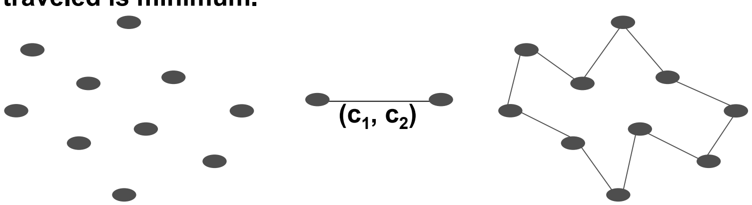
Traveling Salesman problem (TSP):
Input: $V=\{v_1, v_2, ..., v_n\}$ be a set of nodes (cities) in a graph and $d(v_i, v_j)$ the distance between v_i and v_j, find a shortest circuit that visits each city exactly once. (NP-complete)

- (Weighted Hamilton circuit)

Traveling Salesman Problem

Problem Definition

Make a tour starting from a random city, visit every city exactly once and return back to starting city such that the distance traveled is minimum.



It is a NP-hard problem even for single objective optimization.

We intend to find a tour that minimize two costs defined between each pair of cities.

Traveling Salesman Problem . . .

Previous work in single objective TSP

Heuristics

- **Tour construction heuristics: Builds a tour afresh from scratch and terminates when a feasible tour is constructed, e.g., nearest neighbor, greedy.**
- **Tour improvement heuristics: Improve upon a feasible tour, e.g., 2-opt, 3-opt, lk.**

Few polynomial time approximation algorithms (PTAS) are also available

Evolutionary methods

Various solutions by genetic algorithm, ant colony optimization, particle swarm optimization, simulated annealing, tabu search have been proposed.

Since the problem is hard, most researchers have hybridized the evolutionary methods with local search heuristics to obtain good results.

Traveling Salesman Problem . . .

Previous work in biobjective TSP

- *Jaszkiewicz* has presented a hybrid genetic algorithm known as MOGLS.
- *Paquete* and others have presented a two phase (non evolutionary) method hybridized with local search.
- *Zhenyu* and others have presented a genetic algorithm without any local search and emphasize o effective genetic operators.
- *Li* have presented a non evolutionary solution attractor method without any local search.
- Some other studies using branch-and-bound, ε-constrained method, aggregation of two objectives are also available in literature.

Traveling Salesman Problem . . .

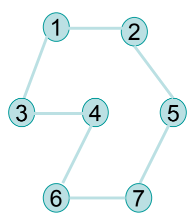
Motivation

- Single objective TSPs with moderate number of cities have been solved to optimality, so, the results can be verified but it is no validated results are available for biobjective TSP.
- Jaszkiewicz argued that Pareto-ranking based MOEAs are neither well suited for MOCO problems nor suited to local search.
- In the literature, we did not come across any solution of biobjective TSP using Pareto-ranking based Multi-Objective Evolutionary Algorithm (MOEA) hybridized with local search.

Traveling Salesman Problem . . .

MOEA Solution

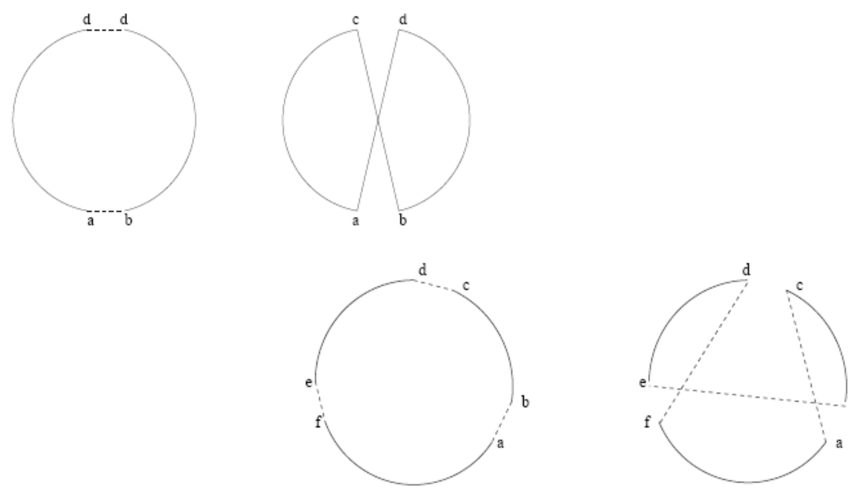
- Pareto-ranking based MOEA
- Complete Elitism
- Parameter less diversity preservation
- *Encoding* of chromosome: path representation
- *Crossover operator*: distance preserving crossover (DPX)
- Mutation operator: double-bridge



Chromosome:
{1, 3, 4, 6, 7, 5, 2}

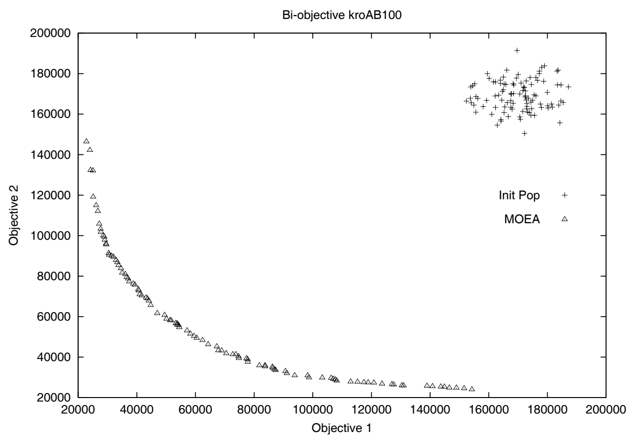
Path representation

Exchange Operators



Traveling Salesman Problem . . .

Results



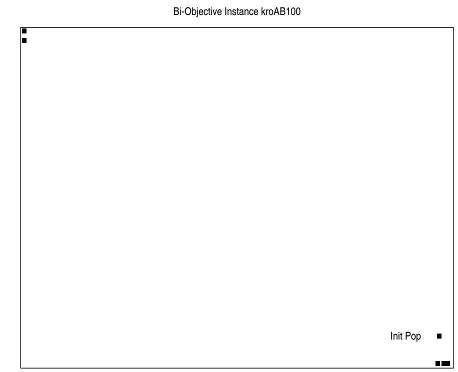
Pure **MOEA result for 100 cities biobjective TSP. Initial population is also shown in figure.**

Traveling Salesman Problem . . .

Hybridization of Pareto-ranked based MOEA

We did 3-opt steepest local search with single objective while generating initial population. It gave us very good solutions distributed at both ends.

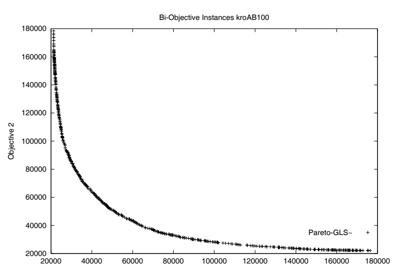
The local search applied after recombination was different in a way that it considered both the objectives simultaneously using Pareto-ranking.

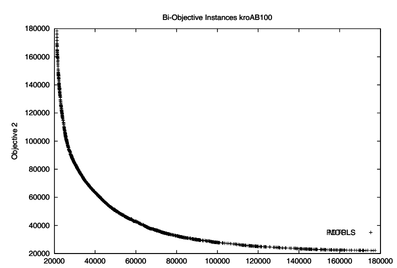


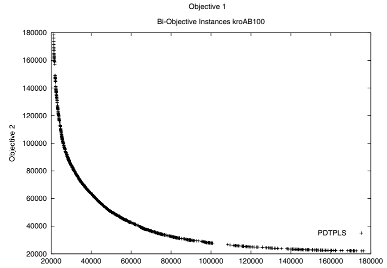
Initial population it has clustered to extremes after local search.

Traveling Salesman Problem . . .

Hybridization of Pareto-ranked based MOEA







All the results are comparable after application of local search (hybridization) in MOEA.

Traveling Salesman Problem . . .

	KroAB100	KroAC100	KroBD100	KroBE100
R Measure				
Pareto-GLS Avg.	0.9350	0.9323	0.9345	0.9334
Std.	0.0000	0.0000	0.0001	0.0001
MOGLS	0.9344	0.9314	0.9338	0.9327
PDTPLS	0.9344	0.9316	0.9340	0.9329
C Measure				
MOGLS covers	36%	25%	32%	32%
covered by	41%	55%	37%	34%
PDTPLS covers	40%	38%	45%	48%
covered by	35%	40%	30%	24%
Spread				
Pareto-GLS	0.6030	0.5229	0.5374	0.5122
MOGLS	0.7587	0.7125	0.7080	0.7124
PDTPLS	0.7750	0.7731	0.6918	0.7224
Convergence				
Pareto-GLS	0.0004	0.0004	0.0007	0.0006
MOGLS	0.0005	0.0008	0.0007	0.0006
PDTPLS	0.0003	0.0003	0.0003	0.0003

Traveling Salesman Problem . . .

Important findings

- **We effectively hybridized Pareto-ranking based MOEA with local search and solved a MOCO problem.**
- **Our results are comparable to the best results available in literature (to the best of our knowledge).**

Network Design

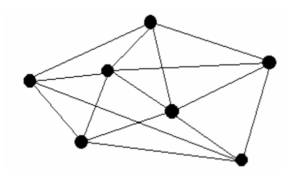
- Minimize {Cost, Diameter, Degree, Intersection Points}
  - Yields a Spanning/Steiner Tree
- Minimize **multiple costs** with different cost measures
  - Example: Multicast Routing – 2 Cost functions
  - Tree construction cost : Channel bw, buffer space and others
  - Delay cost : txn. and queue delays

Subject to a set of constraints

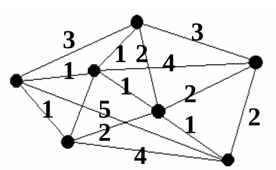
And many other applications :: In almost every sphere of life

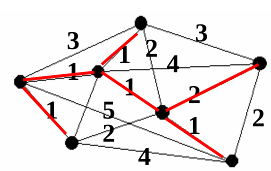
Spanning Tree

A **spanning tree** of a graph G is a subgraph of G that is a tree containing all the vertices of G.



In a weighted graph, a **minimum spanning tree** is a spanning tree whose sum of edge weights is as small as possible. It is the most economical tree of a graph with weighted edges.



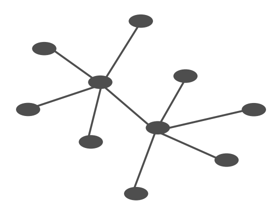


Biobjective MST Problems

Diameter-Cost Minimum Spanning Tree Problem

Problem Definition

Construct a minimum spanning tree (MST) for a given complete graph minimizing simultaneously edge cost and diameter of the tree.



Cost : C'
Diameter : 3

It is a NP-hard problem for 4 ≤ D ≤ (n-1) where D is diameter of the tree and n is the number of nodes.

We intend to find the solutions in full front ranging from 2 to (n-1).

Biobjective MST Problems . . .
Diameter-Cost Minimum Spanning Tree Problem

Motivation

- It is essentially a multiobjective problem as it is better to provide all the solutions to the decision maker (DM) to enable him to opt for best alternate solution.
- No such study is available in the literature. Earlier studies treated diameter as a constraint and solved MST to provide single solution for a particular value of diameter.
- Researchers could not assess the performance of their algorithms over the entire range of solutions. Their claims were localized and cannot be generalized for complete solution front.
- They could not assess the quality of solutions in absence of any reference.

Biobjective MST Problems . . .
Diameter-Cost Minimum Spanning Tree Problem

Motivation

The problem has following characteristics:

- No a priori knowledge of the solution space is available.
- There does not exist any information regarding a reference set.
- No experimental results for polynomial time good approximation algorithm is available.

Biobjective MST Problems . . .
Diameter-Cost Minimum Spanning Tree Problem

Previous work

- **Exact methods**
  - Achuthan & others have presented an exact solution for the diameter constrained MST (DCMST) problem.
  - Kortsarz & others have presented an algorithm for DCMST that combines greedy heuristic and exhaustive search.

  They are restricted to small problems only because of complexity of the problem.
- **Heuristics**
  - Deo & others, Ravi & others, and Raidl & others have presented several approximation algorithms for diameter constraint MST problem.

  Example: OTTC, RGH, and RGH
- **Metaheuristics**
  - Solutions with Genetic algorithms, variable neighborhood search, ant colony optimization are available in literature for DCMST.

Biobjective MST Problems . . .
Diameter-Cost Minimum Spanning Tree Problem

Analysis of search space

Let the cost of unconstrained MST is \`C' and diameter is \`D'.
So, the solution tuple is (C,D)

Now, let us consider a spanning tree with diameter \`D+1'.
Its cost will be either (i) C - ε or (ii) C + ε

Case (i):
- It is not possible. Otherwise MST algorithms are wrong.

Case (ii):
- It is a possibility.
- For trees having diameter \`D+x', we will get cost C + ε where 1 < x < (n-1)-D. Hence, the solution tuple is (C + ε, D+x).
- All such solutions are dominated by MST.

Unconstrained MST is a one extreme solution to the problem.
Best tree with diameter 2 is another extreme solution.

Biobjective MST Problems . . .
Diameter-Cost Minimum Spanning Tree Problem

One Time Tree Construction (OTTC)

- **It is a modification of prim's algorithms. It builds a tree as prim keeping in view that any time diameter constraint is not violated.**

Iterative Refinement (IR)

- **Initially, it generates a MST and then reduce the diameter iteratively to achieve the target diameter or it fails to produce result.**

Random Greedy Heuristic (RGH)

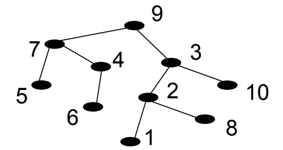
- **It is a center based algorithm. Initially it fix a center and then iteratively and randomly adds edges to complete the tree.**

Pareto versions of the algorithms

- **We run these algorithm for each diameter and initial node to generate a solution front. Since, RGH is a stochastic algorithm we run it multiple time to get best results.**

Biobjective MST Problems . . .
Diameter-Cost Minimum Spanning Tree Problem

MOEA Solution

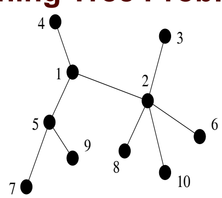


Chromosome:{(5,7),(7,4),(7, 9),(4,6),(9,3),(3,2),(3,10),(2,1) ,(2,8)}

- **Pareto-ranking based MOEA**
- **Complete Elitism**
- **Parameter less diversity preservation**
- ***Encoding* of chromosome: edge-set**
- ***Crossover operator*: selects common parental edges before selecting any non-common edge to make an offspring to preserve locality and heritability from parents**
- **Mutation operator:**
  - ***Edge delete mutation*: deletes an edge randomly and join the two subtrees with another random edge**
  - ***Greedy edge replace mutation*: deletes a random edge and then join the two subtrees with lowest cost edge.**

Biobjective MST Problems . . .
Diameter-Cost Minimum Spanning Tree Problem

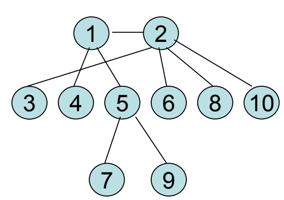
MOEA Solution



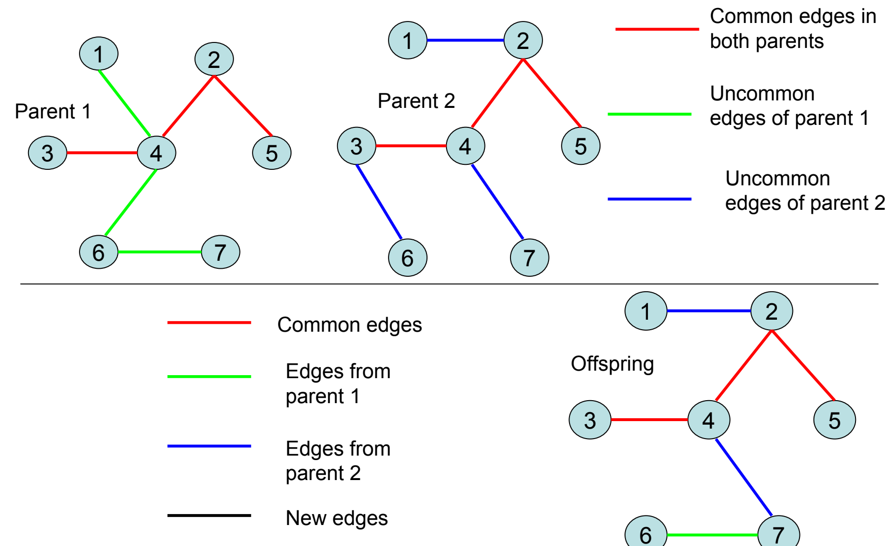
Nodes	Level
1 2	0
3 4 5 6 8 10	1
7 9	2

Chromosome: 0 0 1 1 1 1 2 1 2 1 – – 2 1 1 2 5 2 5 2
Levels Predecessors

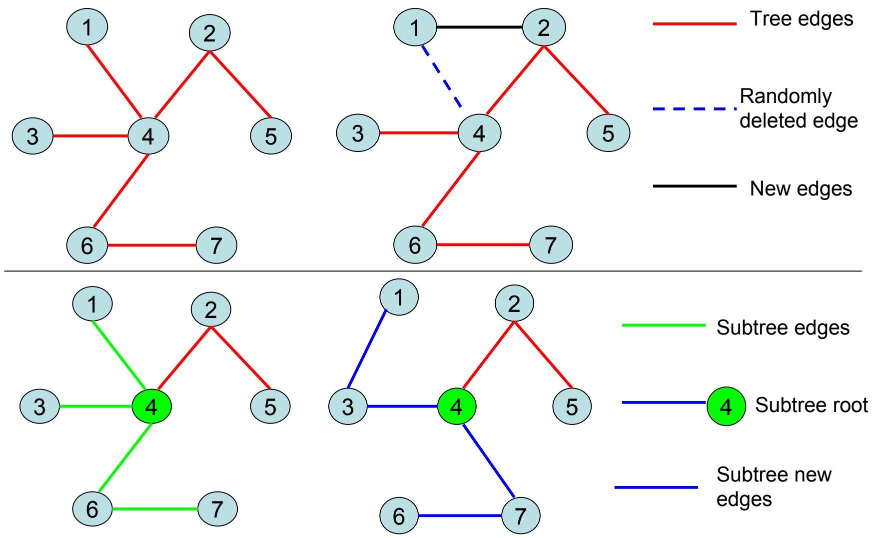
- **Pareto-ranking based MOEA**
- **Complete Elitism**
- **Parameter less diversity preservation**
- **Encoding of chromosome: level encoding**
- ***Crossover operator*: uniform**
- **Mutation operator: Bit mutation**



EA :: Crossover Illustrated



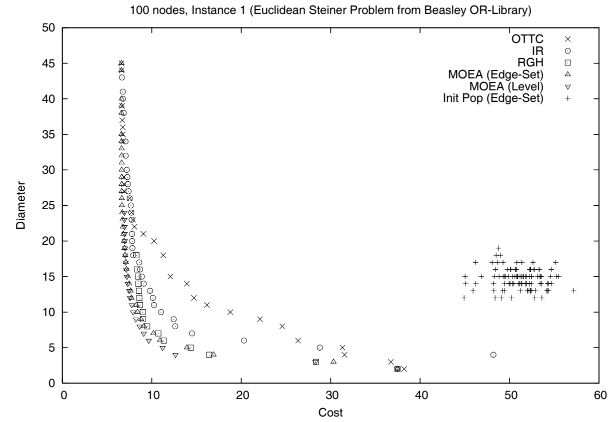
EA :: Mutation Illustrated



Biobjective MST Problems . . .

Diameter-Cost Minimum Spanning Tree Problem

Results



RGH & MOEA (level) generated solutions only in lower diameter range only whereas OTTC, IR & MOEA (edge-set) generated solutions in whole range. Comparatively MOEA (edge-set) is better in whole range.

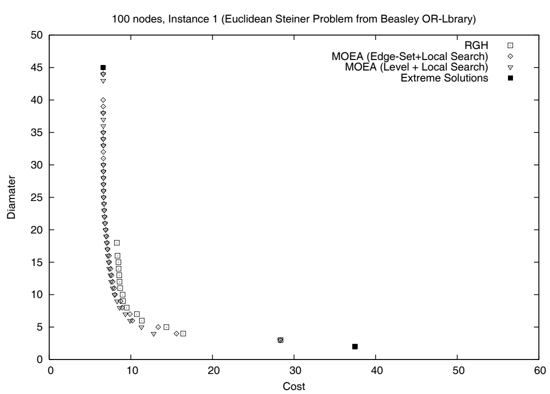
Biobjective MST Problems . . .

Diameter-Cost Minimum Spanning Tree Problem

Improvements in MOEA results

Local search

Injection of extreme solutions in initial population



Though MOEA (edge-set) has generated better than heuristics but MOEA (level) generated the best results after incorporation of problem specific knowledge in the evolution process of MOEA.

Biobjective MST Problems . . .

Diameter-Cost Minimum Spanning Tree Problem

Important findings

- **We analyzed the search space and were able to access the solution front.**
- **We got problem specific knowledge in terms of extreme solutions of the solution front.**
- **We found that heuristics were not able to generate good results over the entire range of solution front.**
- **We got comparatively good solutions in whole range of solution front using MOEA.**
- **We further improved the MOEA results with problem specific knowledge.**

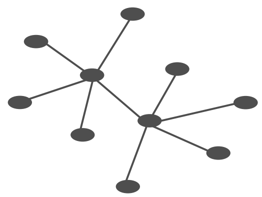
We generated, validated and further improved the results in whole range using MOEA and problem-specific knowledge.

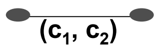
Biobjective MST Problems . . .

Multiple Edge Cost Minimum Spanning Tree Problem

Problem Definition

Construct a minimum spanning tree (MST) for a given complete graph when a vector of costs is associated with each edge.





It is a NP-hard problem.

We intend to find a set of solutions in full front.

Biobjective MST Problems . . .

Multiple Edge Cost Minimum Spanning Tree Problem

Previous work

Exact and approximation algorithms

- Zhou & others have presented an enumeration algorithm.
- Ramos and Steiner & others have presented two-phase exact algorithm.
- Erghott & others and Hamacher & others have presented approximation algorithms.

Evolutionary Algorithms

- Zhou & others and Knowles & others have solved the problem using MOEA.
- Rocha & others have solved the problem using MOEA hybridized with tabu search.
- Lin & others presented solutions in order to solve communication network problems.

Biobjective MST Problems . . .

Multiple Edge Cost Minimum Spanning Tree Problem

Motivation

- Most of the researchers have done their experiments on small problems.
- Researchers have compared their results with some earlier published results to show efficacy of their algorithms and superiority of their results.
- Though Rocha and others have considered large problem but they present their findings in such a way that it fails to assess the quality of obtained results.
- It is simple to get a reference set for this problem using aggregated sum method. It is preferred to compare the solutions using a true reference set and judge the quality of solutions.
- Moreover, the claims regarding superiority must be made only after experiments with varying complexity and fairly large problems.

Biobjective MST Problems . . .

Multiple Edge Cost Minimum Spanning Tree Problem

Heuristic to generate supported as well as unsupported solutions

Input : G = Graph 1 and # iterations

Output : PF = A set of MSTs over G

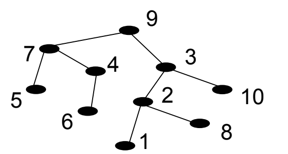
Algorithm :

```
PF ⟵ Ø
For #iterations do
    Generate scalarizing vector λ
    /** Generate supported Pareto-optimal solutions **/
    Use λ on edge costs to aggregate and generate tree using
      standard Prim algorithm
    Update PF
    /** Generate unsupported Pareto-optimal solutions **/
    Use λ on edge costs to aggregate and generate tree using
      standard Kruskal algorithm
    Update PF
Output PF
```

Biobjective MST Problems . . .

Multiple Edge Cost Minimum Spanning Tree Problem

MOEA Solution



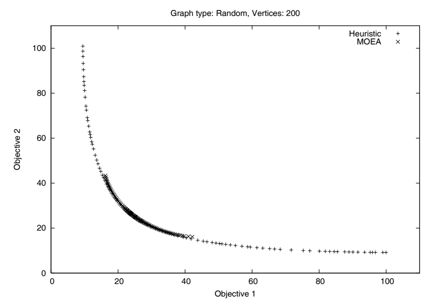
Chromosome:{(5,7),(7,4),(7, 9),(4,6),(9,3),(3,2),(3,10),(2,1) ,(2,8)}

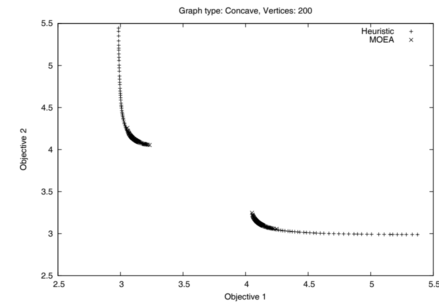
- Pareto-ranking based MOEA
- Complete Elitism
- Parameter less diversity preservation
- *Encoding* of chromosome: edge-set
- *Crossover operator*: selects common parental edges before selecting any non-common edge to make an offspring to preserve locality and heritability from parents
- Mutation operator:
  - *Edge delete mutation*: deletes an edge randomly and join the two subtrees with another random edge
  - *Greedy edge replace mutation*: deletes a random edge and then join the two subtrees with lowest cost edge.

Biobjective MST Problems . . .

Multiple Edge Cost Minimum Spanning Tree Problem

Results





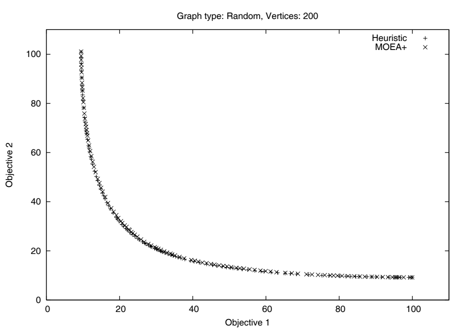
Heuristics has generated solutions in whole range whereas MOEA solutions are concentrated to a part region only (they are visually comparable) for random graph.

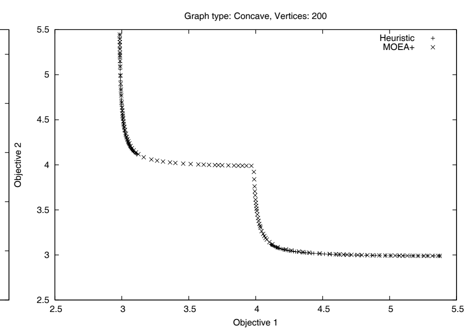
Neither heuristic nor MOEA generated solutions in concave region. Again, MOEA solutions are concentrated to a part region only (they are visually comparable) for concave graph.

Biobjective MST Problems . . .

Multiple Edge Cost Minimum Spanning Tree Problem

Improving the MOEA results





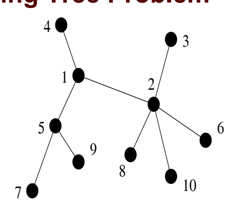
Heuristics and MOEA both results are comparable.

MOEA generated comparable solutions in whole range whereas heuristic is limited to concave region only.

Biobjective MST Problems . . .

Multiple Edge Cost Minimum Spanning Tree Problem

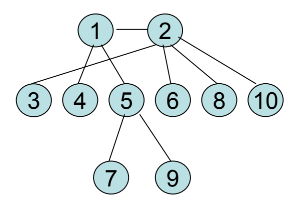
MOEA Solution



Nodes	Level
1 2	0
3 4 5 6 8 10	1
7 9	2

Chromosome: 0 0 1 1 1 1 2 1 2 1 – – 2 1 1 2 5 2 5 2
Levels Predecessors

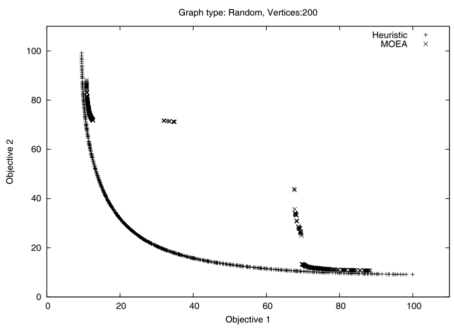
- Pareto-ranking based distributed MOEA where one population optimize one objective and other population optimize other objective. They exchange few good chromosomes after every iteration.
- Complete Elitism
- Parameter less diversity preservation
- Encoding of chromosome: level encoding
- *Crossover operator*: uniform
- Mutation operator: Bit mutation



Biobjective MST Problems . . .

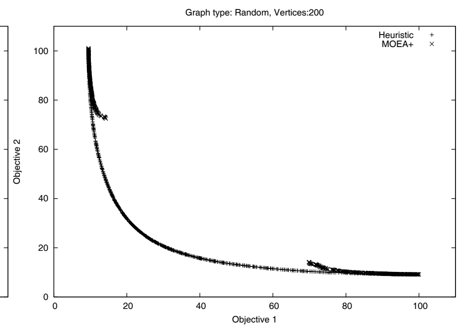
Multiple Edge Cost Minimum Spanning Tree Problem

Results



MOEA generated results only towards both ends without extremes. Few very poor results are scattered in other part region.

Improving the MOEA results



MOEA still generated results only towards both ends including extremes. There are no solutions in other part region.

Biobjective MST Problems . . .

Multiple Edge Cost Minimum Spanning Tree Problem

Improvement in MOEA (edge-set) results

	Random graph	Concave graph
C Measure MOEA covers H-MOEA MOEA covered by H-MOEA	14.33% 75.87%	02.45% 94.64%
Spread MOEA H-MOEA	0.60 0.54	0.59 0.52
Convergence MOEA H-MOEA	0.006 0.004	0.002 0.001

Biobjective MST Problems . . .

Multiple Edge Cost Minimum Spanning Tree Problem

Important findings

- We generated very good results using little problem-specific knowledge, for varying complexities of the problem, in whole range whereas heuristics could not generate solutions in whole range for all the problems.
- Though hybridization of MOEA with a local search heuristic has been proved very effective to generate good solutions for hard problems but in few cases it is possible to generate good solutions with little problem-specific knowledge only.
- It is preferable to devise good representation (encoding of chromosome) and genetic operator to solve the problem effectively.

Intersecting Spanning Trees from Multiple Geometric Graphs

Problem Definition

Given two geometric graphs (corresponds to two net lists), find Minimum Spanning Tree (MST) with two objectives

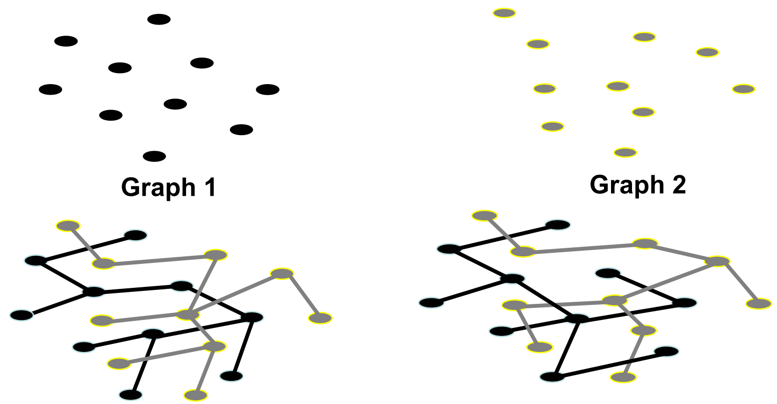
- ❑ Minimize total edge cost
- ❑ Minimize number of intersections among the tree edges

Characteristic of the problem

- ❖ Multiobjective combinatorial optimization
- ❖ NP-hard

Intersecting Spanning Trees from Multiple Geometric Graphs

Contd … Problem Definition



Cost = C’ Intersections = 5

Cost = C” Intersections = 6

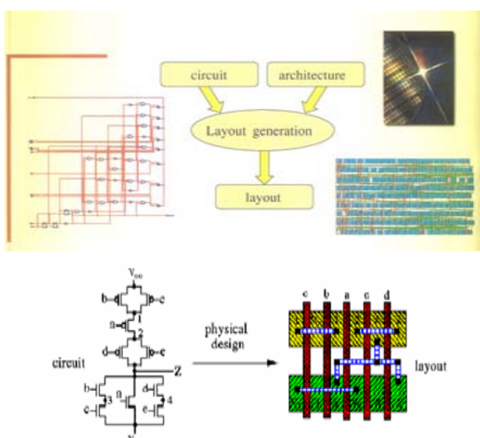
Intersecting Spanning Trees from Multiple Geometric Graphs

Motivation: CAD for VLSI

- Specification
- Architecture Design
- Logic Design
- Circuit Design
- Physical Design
  - Partitioning
  - Placement
  - Routing
    - Global Routing
    - Detailed Routing
  - Compaction
  - Extraction & Verification
- Fabrication
- Testing / Debugging

Physical Design Flow

Circuit Partitioning
⇩
Floor-planning & Placement
⇩
Routing
⇩
Layout Compaction
⇩
Extraction and Verification



- Physical design converts a circuit description into a geometric description.
- The description is used to manufacture a chip.

Steiner Tree

Let G be shown in Figure a. R={a,b,c}. The Steiner minimum tree T={(a,d),(b,d),(c,d)} which is shown in Figure b.

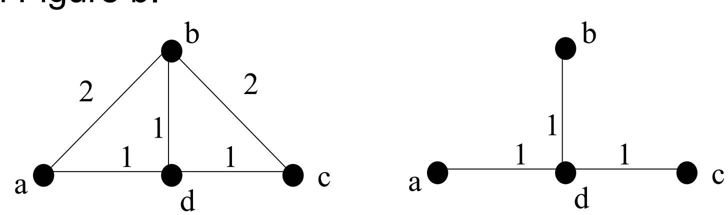
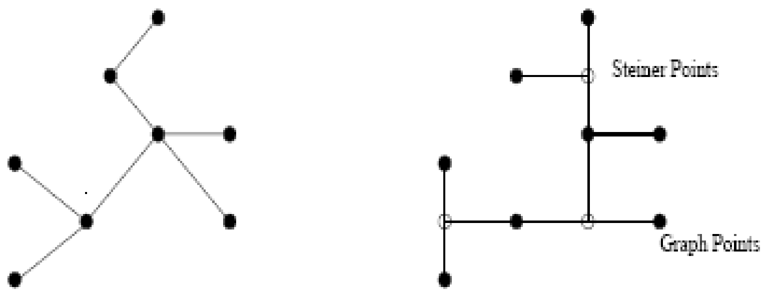


Figure a Figure b

Minimum Steiner tree problem is NP-complete.

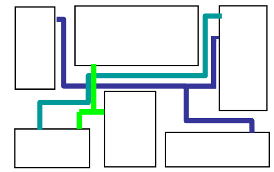
Rectilinear Steiner Tree

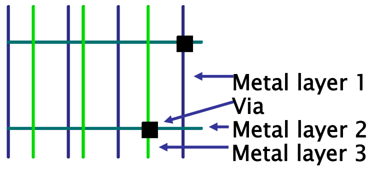


(a) Minimum Spanning Tree (b) Rectilinear Steiner Tree

Intersecting Spanning Trees from Multiple Geometric Graphs

Contd …Motivation: CAD for VLSI





Two geometrically crossing edges belonging to two distinct nets can not be routed on a single metal layer preserving their embeddings. Hence, we require a multilayer design. To make use of another routing layer, each crossing among the tree edges requires *vias* so that the wires can change layers.

Implications of Vias

- **Increase in number of vias decrease the yield as they involve processing of multiple layers.**
- **They introduce parasitic capacitance which in turn may affect the speed of chip.**

Desirable

- **Route not only with the minimum wire-length but also minimum intersections.**

Intersecting Spanning Trees from Multiple Geometric Graphs

Previous Work

- **Tokunaga & others derived theoretical results on the problem of finding geometric spanning trees such that they intersect in as few points as possible on two simple geometric graphs consisting of bi-colored point sets.**
- **Kano & others too theoretically attempted a problem similar to Tokunaga with multiple geometric graphs instead of only two and suggested an upper bound on the number of intersections of tree edges.**
- **Majumder & others studied similar problem and suggested a heuristic to construct a Rectilinear Steiner Tree (RST) of bi-colored point sets on two geometric graphs. The heuristic first generates a geometric MST and then convert it to rectilinear and provide a single solution.**

Intersecting Spanning Trees from Multiple Geometric Graphs

Heuristics for extreme solutions

Search over Minimum Spanning Trees

Input : **G_1 = Graph 1 and G_2 = Graph 2**

Output : **PF = A set of tuples (T_1, T_2) where T_1, T_2 are MSTs over G_1 and G_2 respectively**

Algorithm :

```
PF ⟵ Ø
For all nodes u1 of G1 do
    Make T1 considering u1 as start node of the tree
    For all nodes u2 of G2 do
        Make T2 considering u2 as start node of the tree
        Compute objective vector of tuple (T1, T2)
        Update PF
Output PF
```

Intersecting Spanning Trees from Multiple Geometric Graphs

Heuristics for extreme solutions

Heuristic for Fewer Intersection Points

Input : G_1 = Graph 1 and G_2 = Graph 2

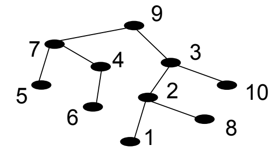
Output: PF = A set of tuples (T_1, T_2) where T_1, T_2 are STs over G_1 and G_2 respectively

Algorithm:

- *PF* ⟵ *Ø*
- u_1, u_2 ⟵ random initial node from Graphs G_1 and G_2 respectively to make T_1 and T_2
- T_1 and T_2 grows iteratively considering smallest cost edge that gives minimum number of intersections among the edges of trees
- Output PF

Intersecting Spanning Trees from Multiple Geometric Graphs

MOEA Solution



Chromosome:{(5,7),(7,4),(7,9),(4,6),(9,3),(3,2),(3,10),(2,1),(2,8)}

- Pareto-ranking based MOEA
- Complete Elitism
- Parameter less diversity preservation
- *Encoding* of chromosome: edge-set
- *Crossover operator*: selects common parental edges before selecting any non-common edge to make an offspring to preserve locality and heritability from parents
- Mutation operator:
  - *Edge delete mutation*: deletes an edge randomly and join the two subtrees with another random edge
  - *Greedy edge replace mutation*: deletes a random edge and then join the two subtrees with lowest cost edge.

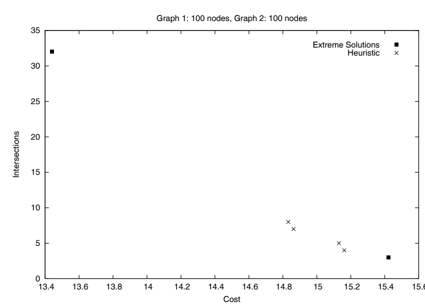
Intersecting Spanning Trees from Multiple Geometric Graphs

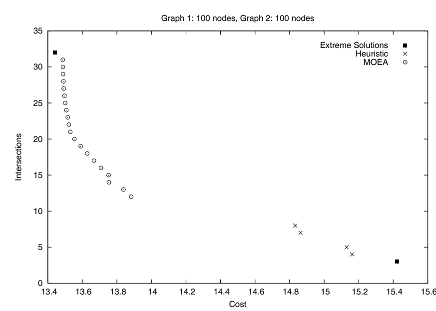
- For many combinatorial optimization problems good solutions usually lie in neighborhood.
- Neighborhood can be searched in finite steps.
- (T_1, T_2) ⟵ MSTs of G_1 and G_2 is one extreme optimal solution for this problem and hence a good start point.
- It usually produces good local optimal solutions.

ES ⟵ *Ø*
(T_1, T_2) ⟵ *MSTs of G_1 and G_2*
(T_1, T_2) ⟵ *unvisited*
ES ⟵ *(T_1, T_2)*
While there are unvisited solution S in ES do
  Sort intersecting edges in descending order of # intersections
  For each edge (u, v) do
    *S** ⟵ *neighborhood solutions \ (u, v)*
    Mark S as unvisited*
    *Update ES with S**
  Mark solution S visited
Output ES

Intersecting Spanning Trees from Multiple Geometric Graphs

Extreme and MOEA solutions



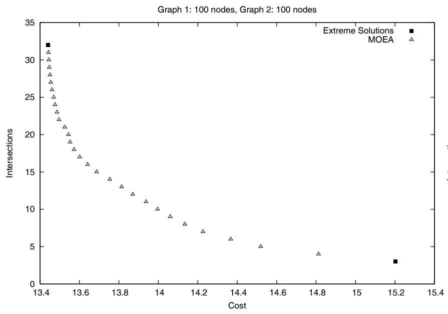


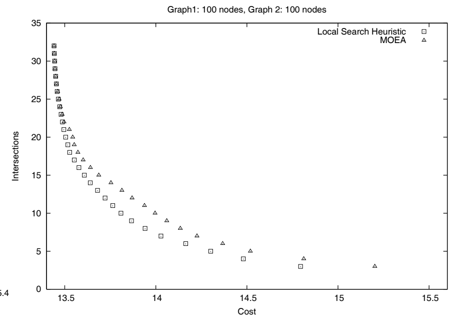
Solutions generated by heuristics designed to generate extremes.

Solutions generated by MOEA along with extreme solutions.

Intersecting Spanning Trees from Multiple Geometric Graphs

Informed MOEA and local search heuristic solutions



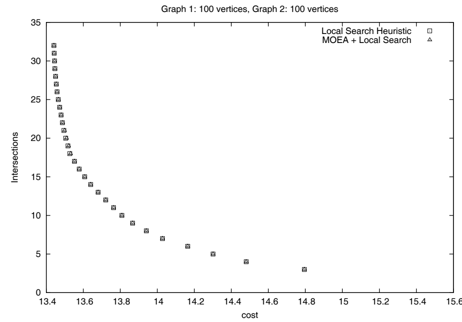


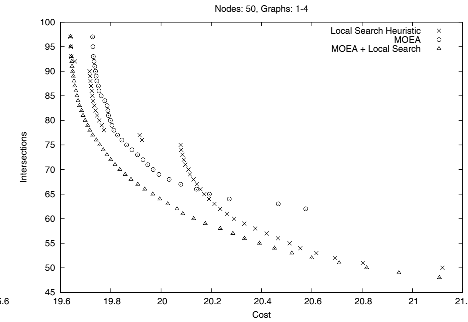
Extreme solutions generated by extreme heuristics were injected in initial population in MOEA. Now, MOEA finds full Pareto-front

Solutions generated by local search heuristics are better than even informed MOEA solutions.

Intersecting Spanning Trees from Multiple Geometric Graphs

Local search and MOEA+local search solutions





Extreme solutions generated by local search heuristics were injected in initial population in MOEA. Now, MOEA results almost matches local search heuristic results.

In case of multigraphs, solutions of MOEA injected with extreme solutions generated by local search heuristic are better than the solutions generated by local search heuristics itself.

Intersecting Spanning Trees from Multiple Geometric Graphs

Important Findings

- **The designed local search heuristic is**
  - **Simple neighborhood search**
  - **Scaleable to any number of nodes**
  - **Expendable to any number of graphs**
  - **Efficient compared to stochastic evolutionary algorithm.**
- **MOEA solution is effective and generates good solutions. The more problem-specific knowledge is introduced to evolution process, the better are the generated solutions.**
- **Solution space was effectively explored by incrementally designing and sandwiching strategies for evolutionary and heuristic search to serve each other, turn by turn, a reference set per se. In this scenario:**
  - **Can we effectively solve unknown problems using black-box optimization techniques?**
  - **How can one trust the solutions obtained for Real-World Applications by such black-box optimization specially on multiobjective optimization?**
  - **how can we effectively approximate the quality of solutions in real-world problems?**

Thanks

Questions !!!

The author gratefully acknowledges travel support from

Google, Inc., India

rkumar @ cse.iitkgp.ernet.in
www.facweb.iitkgp.ernet.in/~**rkumar**