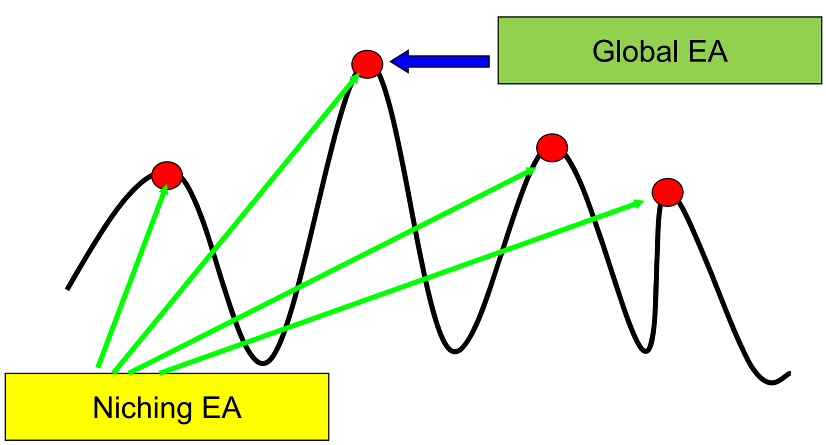
Multi-modal Optimization

- Aim: To find multiple global and local optima of a single objective problem.
- Evolutionary Algorithms vs. Classical Optimization Methods: Evolutionary methods with populations are more suited compared to single solution based methods.
- Standard GAs/EAs converge to the global or a sub-optimal point
- Prevent convergence to a single solution and maintain multiple solutions – Niching (each desired solution is called a niche)

Niching and Multimodal Optimization

- Traditional evolutionary algorithms with elitist selection are suitable to locate a single optimum of **functions**.
- Real problem may require the identification of optima along with several optima.
- For this purpose, **niching methods** extend the simple evolutionary algorithms by *promoting the formation of subpopulations in the neighborhood of the local optimal solutions.*
- ***Multiple solutions are shown below for a maximization problem.***



Multi-modal Optimization Methods

- Some existing Niching Techniques
  - Sharing
  - Clearing
  - Crowding
  - Restricted Tournament Selection
  - Clustering
  - Species Based
  - Neighborhood based DE

Multi-modal Optimization - Sharing

- Sharing
  - First among Niching Techniques
  - Proposed by Holland – Improved by Goldberg & Richardson
  - Population divided into subgroups based on similarity of individuals

    (σ - threshold of dissimilarity or niche radius)
  - Information sharing with other individuals in the same niche

    (Fitness sharing)

$$f_i' = \frac{f_i}{m}$$

    m is niche count
  - Complexity – $O(NP)$

    NP – population size

Multi-modal Optimization - Clearing

- Clearing
  - Retain the best members while eliminating the worst individuals of each niche
  - Complexity – $O(cNP)$

    NP – population size, c – number of subpopulations

  - Advantages
    - Lower Complexity
    - Significant reduction in genetic drift due to selection noise
    - Population can be much smaller

Multi-modal Optimization - Crowding

- Crowding
  - Proposed by De Jong
  - Newly generated individuals can replace similar individuals in the population.
  - Similarity determined by a distance metric
  - 2 parents randomly selected and produce 2 offspring by Mutation and crossover
  - Offspring replace nearest or similar parent if offspring are of greater fitness
  - Complexity $O(NP)$

Multi-modal Optimization - RTS

- Restricted Tournament Selection (RTS)
  - Proposed by Harick
  - Similar to Crowding (except with a limited-windowed population)
  - Corresponding to each offspring randomly choose *w* individuals from the population

  w – window size
  - From *w*, pick the nearest or similar individual to the offspring
  - Restricts competition with some of the similar individuals
  - Complexity – $O(NP * w)$

Neighborhood Mutation Based DE

Steps of Generating Offspring Using Neighborhood Mutation

Input	A population of solutions of current generation (current parents)
Step 1	For $i = 1$ to NP (population size)
	1.1 Calculate the Euclidean distances between individual i and other members in the population.
	1.2 Select m smallest Euclidean distance members to individual i and form a subpopulation (*subpop*) using these m members.
	1.3 Produce an offspring u_i using DE equations within $subpop_i$, i.e., pick r_1, r_2, r_3 from the subpopulation.
	2.3 Reset offspring u_i within the bounds if any of the dimensions exceed the bounds.
	2.4 Evaluate offspring u_i using the fitness function.
	Endfor
Step 2	Selection NP fitter solutions for next generation according to the strategies of different niching algorithm.
Output	A population of solutions for next generation

Compared with about 15 other algorithms on about 27 benchmark problems including recent IEEE TEC articles.

B-Y Qu, P N Suganthan, J J Liang, "Differential Evolution with Neighborhood Mutation for Multimodal Optimization," *IEEE Trans on Evolutionary Computation*, Doi: 10.1109/TEVC.2011.2161873, 2012.

Multi-modal Optimization

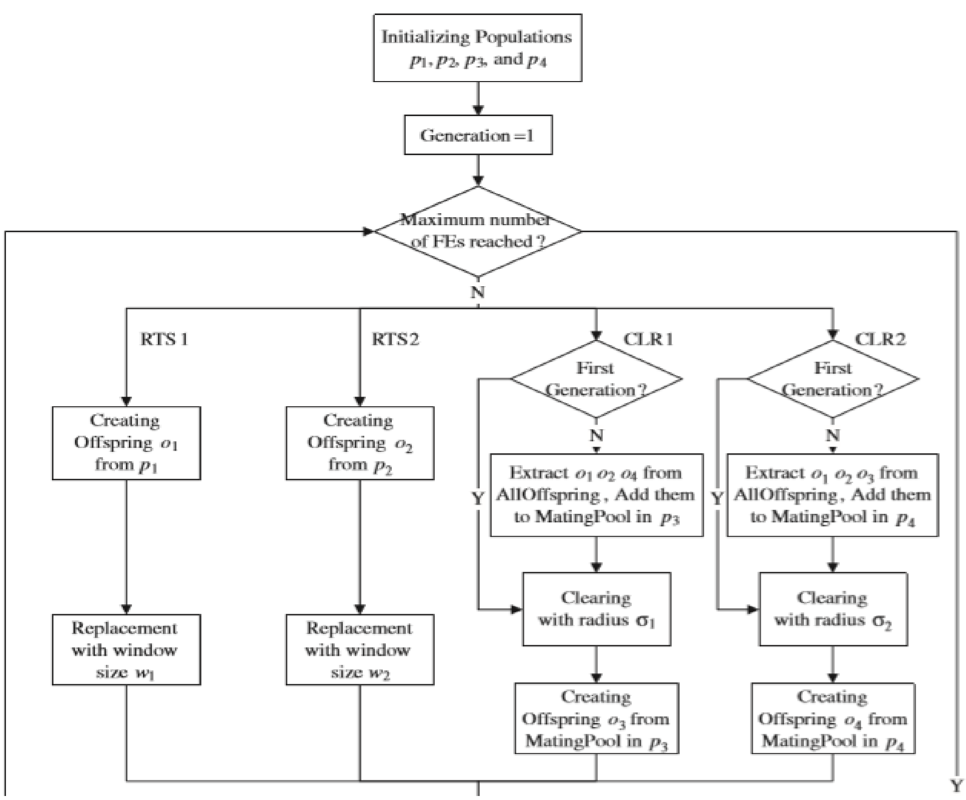
- Species based
  - Separating population into several species based on similarity
  - Similar to sharing – except no change in fitness

    (σ - species distance)

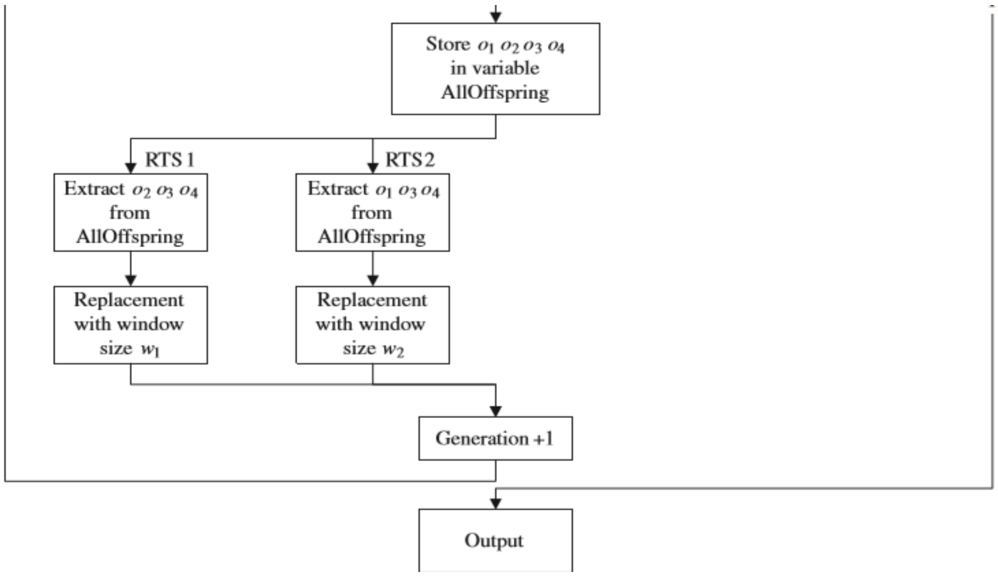
- Ensemble of Niching Algorithms (ENA)
  - Population divided into niches using various niching methods
  - Same selection and survival criteria used

E. L. Yu, P. N. Suganthan, "Ensemble of niching algorithms", *Information Sciences*, Vol. 180, No. 15, pp. 2815-2833, Aug. 2010.

Ensemble of niching algorithms – Page 1



Ensemble of niching algorithms – Page 2



Multi-Objective GAs

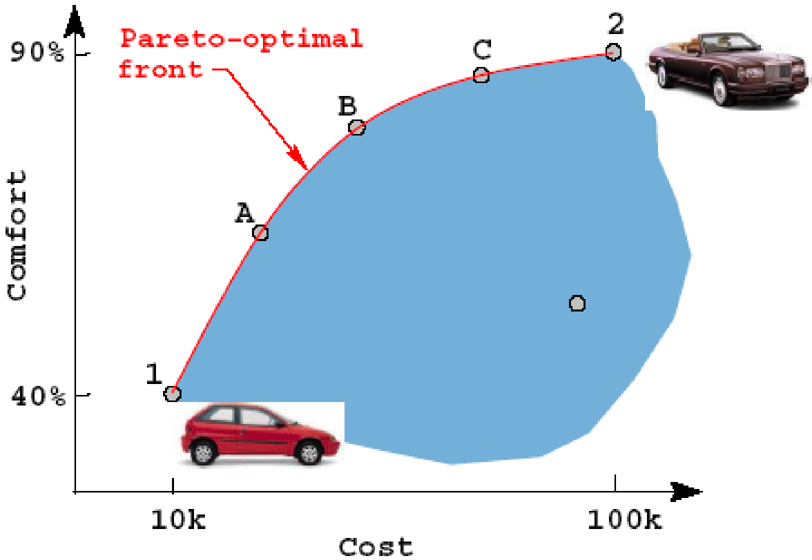
- Introduction
- Pareto Optimality
- Dominance
- Non-dominated Sorting GA II (NSGA-II)
- Maintaining Diversity

Reference:

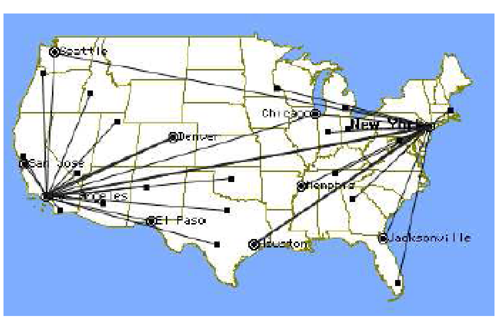
K. Deb. *Multi-objective optimization using evolutionary algorithms.* Chichester, UK: Wiley, 2001. (Second edition, with exercise problems)

Multi-Objective Optimization

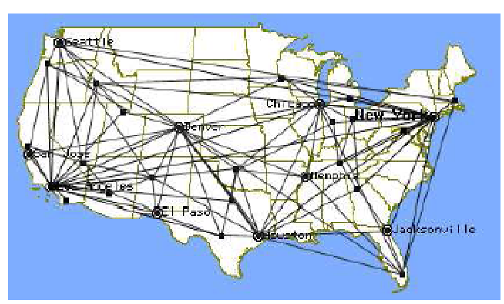
We often face them



More Examples



A cheaper but inconvenient flight



A convenient but expensive flight

Mathematical Representation

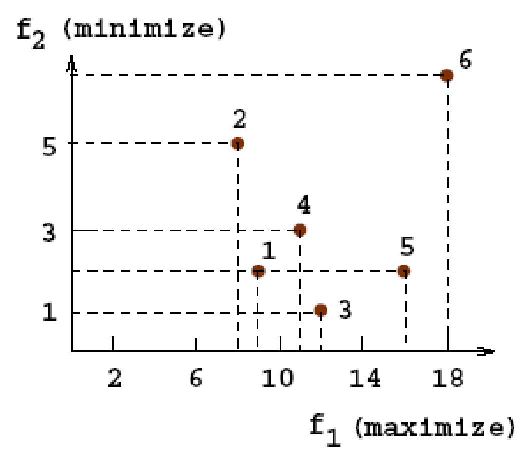
$$
\begin{aligned}
\text{Min/Max} \quad & (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_M(\mathbf{x})) \\
\text{Subject to} \quad & g_j(\mathbf{x}) \geq 0 \\
& h_k(\mathbf{x}) = 0 \\
& \mathbf{x}^{(L)} \leq \mathbf{x} \leq \mathbf{x}^{(U)}
\end{aligned}
$$

Which Solutions are Optimal?

Relates to the concept of domination

$\mathbf{x}^{(1)}$ dominates $\mathbf{x}^{(2)}$ if

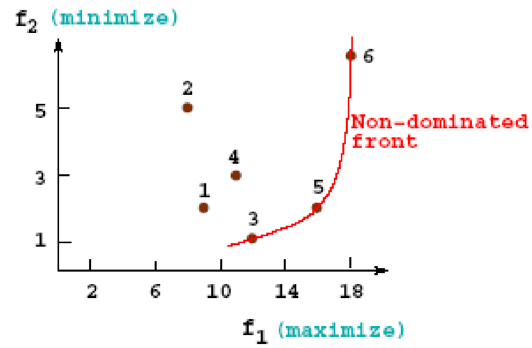
1. $\mathbf{x}^{(1)}$ is no worse than $\mathbf{x}^{(2)}$ in all objectives
2. $\mathbf{x}^{(1)}$ is strictly better than $\mathbf{x}^{(2)}$ in at least one objective



Pareto-Optimal Solutions

Non-dominated solutions: Among a set of solutions P, the non-dominated set of solutions P' are those that are not dominated by any member of the set P. $O(N \log N)$ algorithms exist.

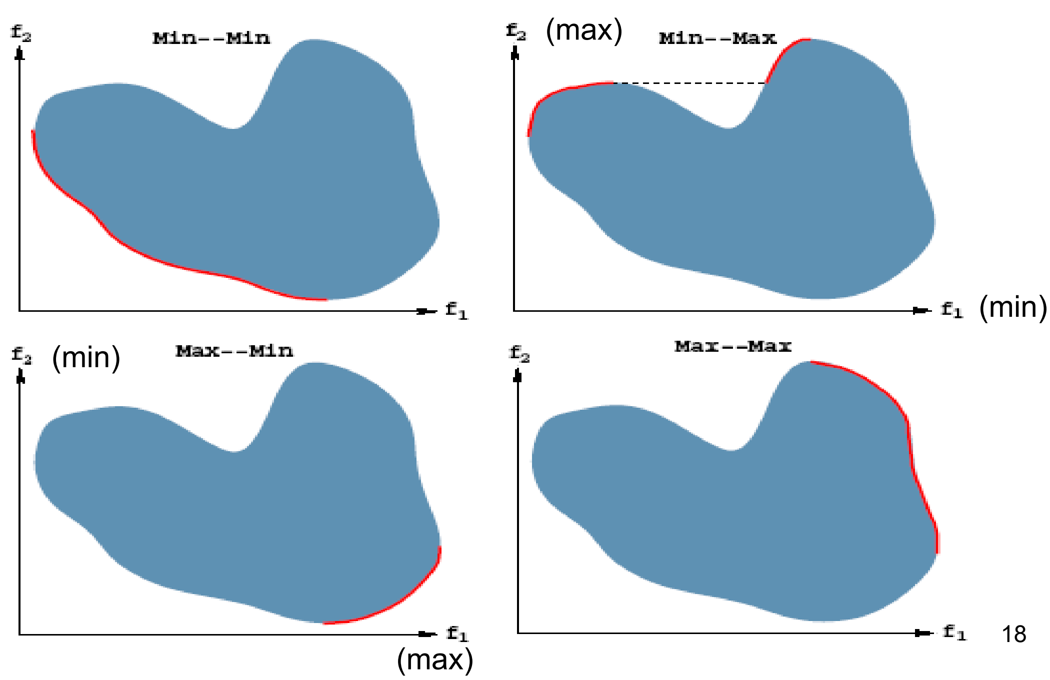
Pareto-Optimal solutions: When $P = \mathcal{S}$, the resulting P' is Pareto-optimal set



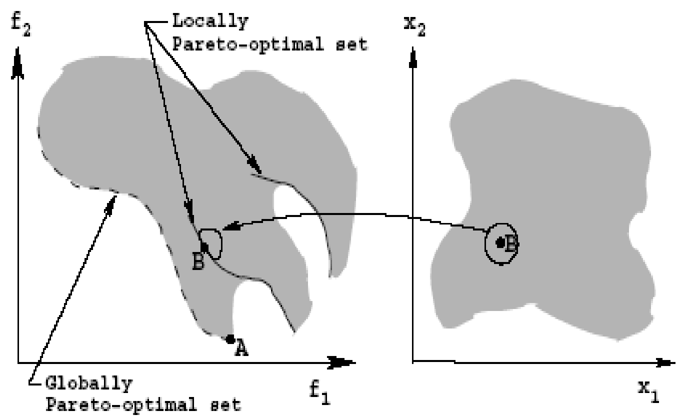
A number of solutions are optimal

Where *S* is the set of all feasible solutions

Pareto-Optimal Fronts

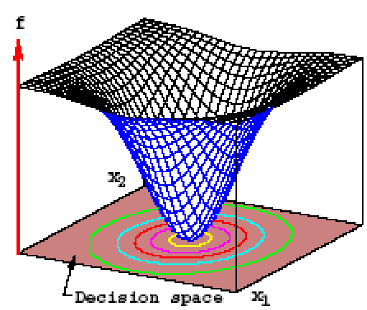


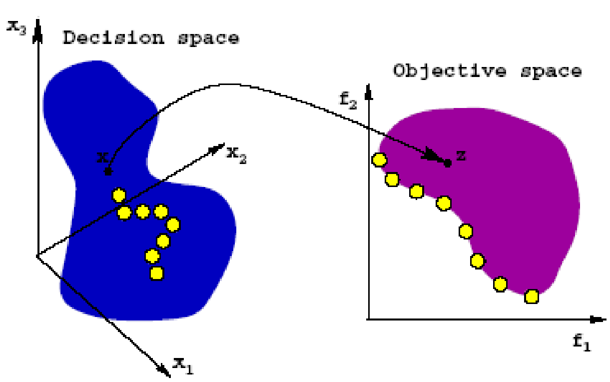
Like single-objective optimization, local and global P-O fronts exist:



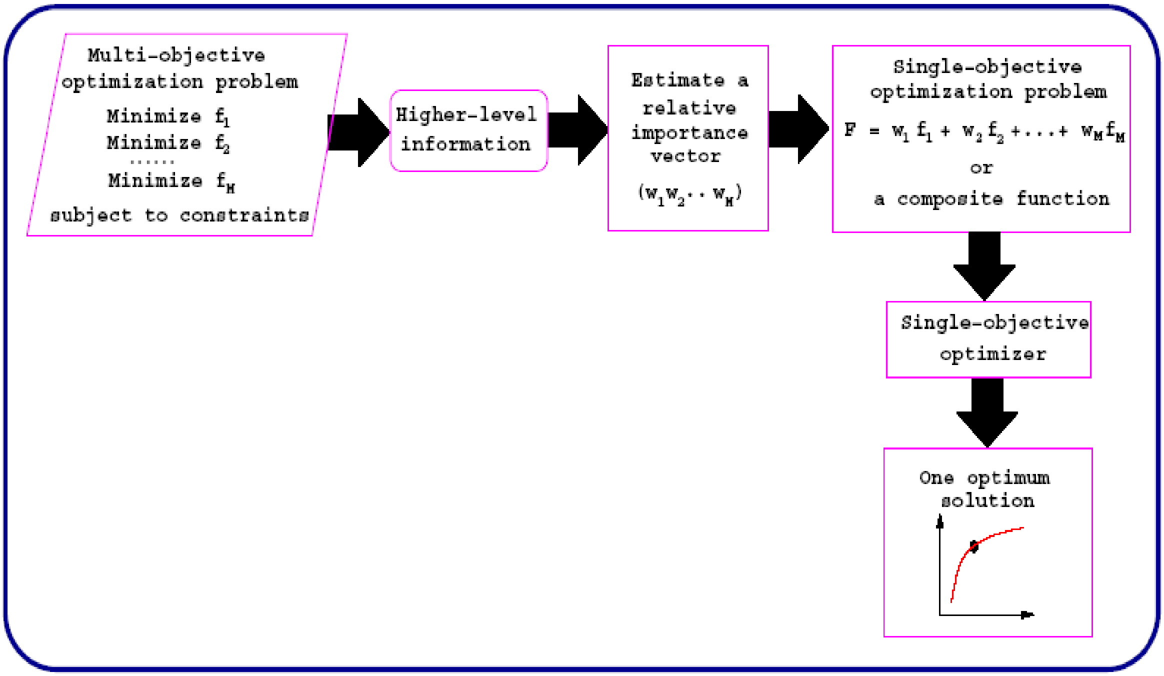
Differences with Single-Objective Optimization

- One optimum versus multiple optima
- Requires search and decision-making
- Two spaces of interest, instead of one





Weighted Sum Approach



Conventional non-evolutionary methods usually follow this approach

Weighted Sum Method

- Construct a weighted sum of objectives and optimize

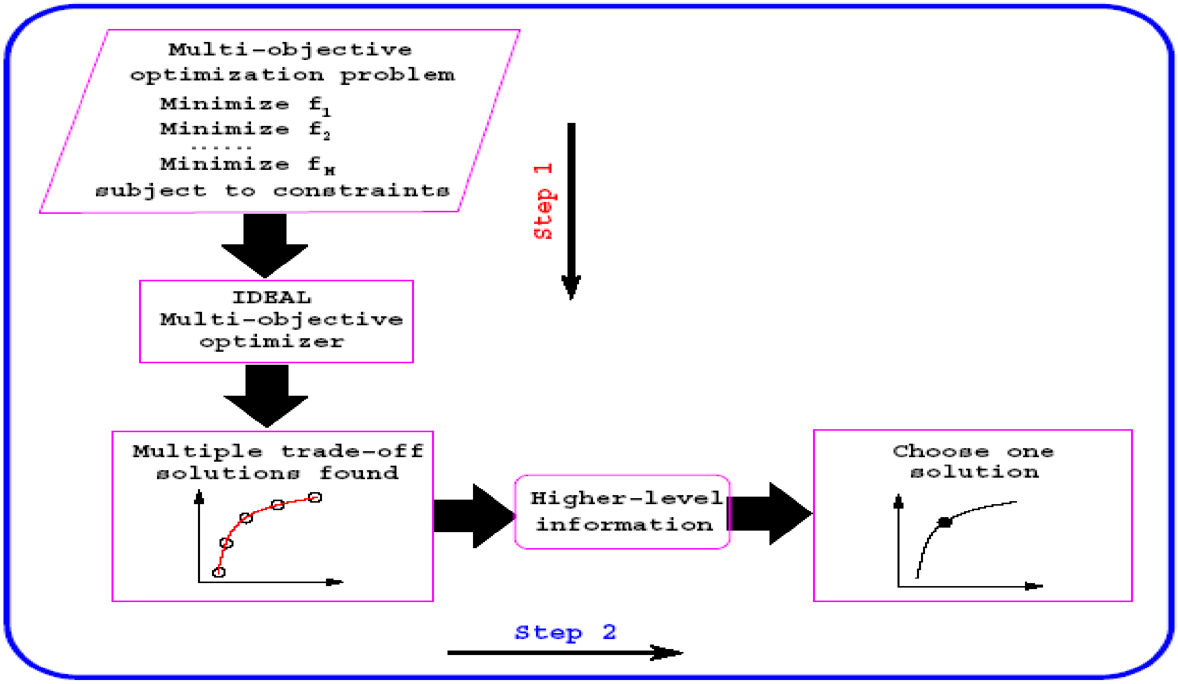
$$F(\mathbf{x}) = \sum_{m=1}^{M} w_m f_m(\mathbf{x}).$$

- User supplies weight vector $\mathbf{w}$

Difficulties with Weighted Sum Method

- Need to know $\mathbf{w}$
- Non-uniformity in Pareto-optimal solutions
- Inability to find some Pareto-optimal solutions

Ideal Multi-Objective Optimization

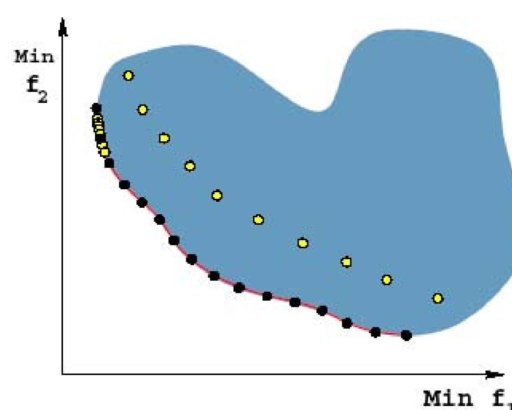


Step 1 Find a set of Pareto-optimal solutions

Step 2 Choose one from the set

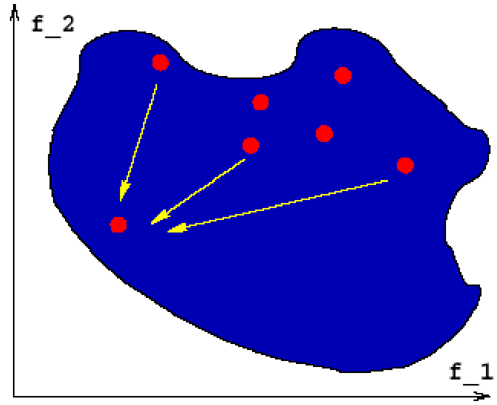
Two Goals in Ideal Multi-Objective Optimization

1. Converge on the Pareto-optimal front
2. Maintain as diverse a distribution as possible



Why Use Evolutionary Algorithms?

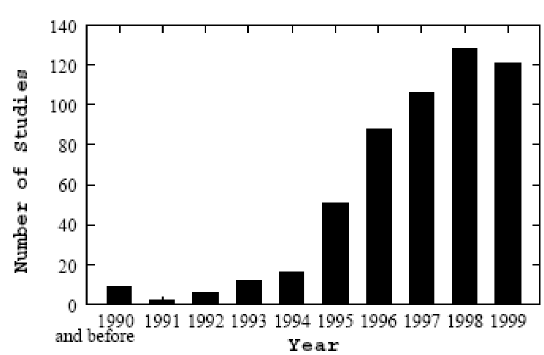
- Population approach suits well to find multiple solutions
- Niche-preservation methods can be exploited to find diverse solutions
- Implicit parallelism helps provide a parallel search



- Multiple applications of classical methods do not constitute a parallel search

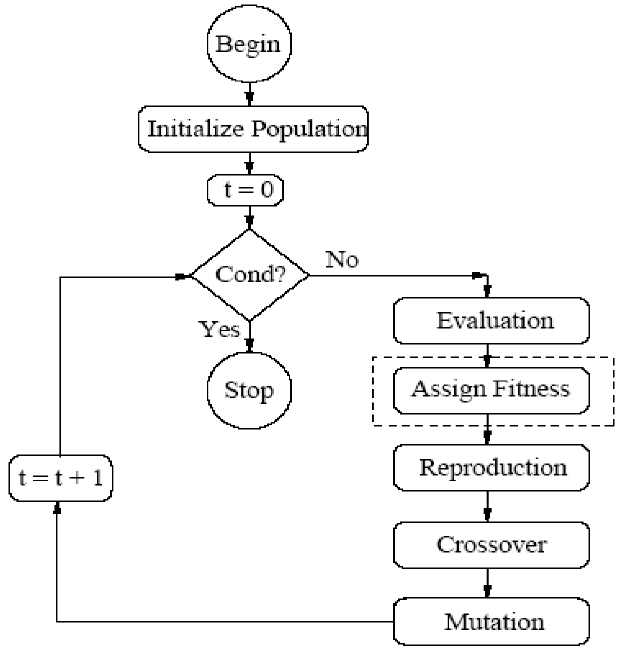
History of Evolutionary Multi-Objective Optimization (EMO)

- Early penalty-based approaches
- VEGA (1984)
- Goldberg's (1989) suggestion
- MOGA, NSGA, NPGA (1993-95) used Goldberg's suggestion
- Elitist EMO (SPEA, NSGA-II, PAES, MOMGA etc.) (1998 – Present)



What to Change in a Simple GA?

- Modify the fitness computation
- Emphasize non-dominated solutions for convergence
- Emphasize less-crowded solutions for diversity



Identifying the Non-dominated Set

Step 1 Set $i = 1$ and create an empty set P'.

Step 2 For a solution $j \in P$ (but $j \neq i$), check if solution j dominates solution i. If yes, go to Step 4.

Step 3 If more solutions are left in P, increment j by one and go to Step 2; otherwise, set $P' = P' \cup \{i\}$.

Step 4 Increment i by one. If $i \leq N$, go to Step 2; otherwise stop and declare P' as the non-dominated set.

$O(MN^2)$ computational complexity

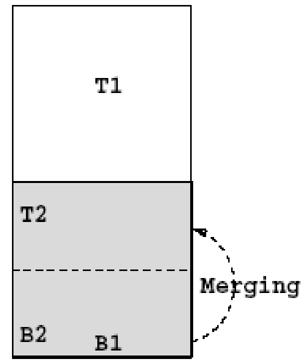
M – total number of objectives, *N* – population size

Finding the Non-dominated Set: An Efficient Approach

Kung Algorithm

Step 1 Sort the population in descending order of importance of f_1

Step 2, Front(P) If $|P| = 1$, return P as the output of **Front**(P). Otherwise, T = **Front**$(P^{(1)} - - P^{(|P|/2)})$ and B = **Front**$(P^{(|P|/2+1)} - - P^{(|P|)})$. If the i-th solution of B is not dominated by any solution of T, create a merged set $M = T \cup \{i\}$. Return M as the output of **Front**(P).



$O\left(N(\log N)^{M-2}\right)$ for $M \geq 4$ and $O(N \log N)$ for $M = 2$ and 3

Example on Kung Algorithm from Slides 5-6

Step 1: Sort the popln according 1st objective: P={6,5,3,4,1,2}

Step 2: P's size is not 1. We'll have to divide recursively until size is 1.

{6,5,3,4,1,2}

[front T] {6,5,3} (label A) [front B] {4,1,2} (label B)

[front T]{6} [front B] {5,3} (label C) [front T] {4} {1,2} (label D)

{5} {3} {1} {2}

5 & 3 do not dominate each other →{5,3} at C.
6, 5 & 3 do not dominate each other → {6,5,3} at A.

1 dominates 2 → discard 2 to have {1} at D.
4 & 1 do not dominate each other → {4,1} at B.

Example *cont.*

Step 2 (cont.): when all the sizes are 1, we work upward.

Finally we have 2 non-dominated sets {6,5,3} derived from front T and {4,1} derived from front B.

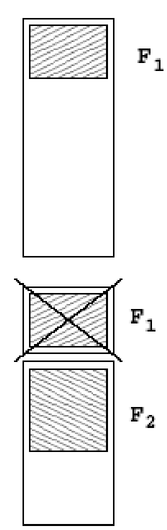
Now check each solution in {4,1} if they are dominated by any solution in {6,5,3}.

In this example, 4 & 1 are dominated by 3 & 5.

Hence, the final non-dominated set is {6,5,3}.

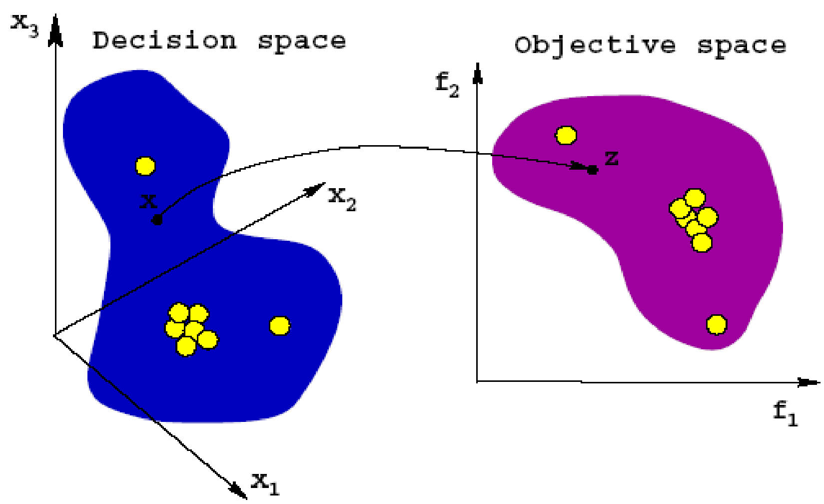
A Simple Non-Dominated Sorting Procedure

- Identify the best non-dominated set
- Discard them from population
- Identify the next-best non-dominated set
- Continue till all solutions are classified
- We discuss a $O(MN^2)$ algorithm later



Which are Less-Crowded Solutions?

- Crowding can be in decision variable space or in objective space



Non-Elitist EMOs

- Vector evaluated GA (VEGA) (Schaffer, 1984)
- Vector optimized EA (VOES) (Kursawe, 1990)
- Weight based GA (WBGA) (Hajela and Lin, 1993)
- Multiple objective GA (MOGA) (Fonseca and Fleming, 1993)
- Non-dominated sorting GA (NSGA) (Srinivas and Deb, 1994)
- Niched Pareto GA (NPGA) (Horn et al., 1994)
- Predator-prey ES (Laumanns et al., 1998)

Vector-Evaluated GA (VEGA)

- Divide population into M equal blocks
- Each block is reproduced with one objective function
- Complete population participates in crossover and mutation
- Bias towards to individual best objective solutions
- A non-dominated selection: Non-dominated solutions are assigned more copies
- Mate selection: Two distant (in parameter space) solutions are mated

Shortcomings of Non-Elitist EMOs

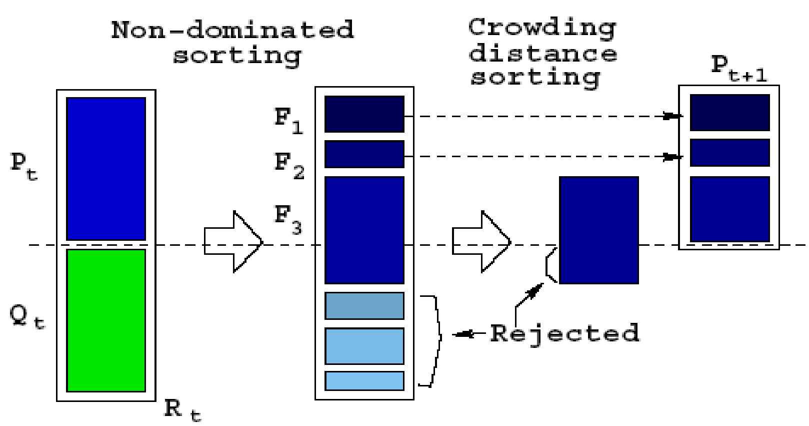
- Elite-preservation is missing
- Elite-preservation is important for proper convergence in SOEAs
- Same is true in EMOs
- Three tasks
  - Elite preservation
  - Progress towards the Pareto-optimal front
  - Maintain diversity among solutions

Elitist EMOs

- Distance-based Pareto GA (DPGA) (Osyczka and Kundu, 1995)
- Thermodynamical GA (TDGA) (Kita et al., 1996)
- Strength Pareto EA (SPEA) (Zitzler and Thiele, 1998)
- Non-dominated sorting GA-II (NSGA-II) (Deb et al., 1999)
- Pareto-archived ES (PAES) (Knowles and Corne, 1999)
- Multi-objective Messy GA (MOMGA) (Veldhuizen and Lamont, 1999)
- Other methods: Pareto-converging GA, multi-objective micro-GA, elitist MOGA with coevolutionary sharing

Elitist Non-dominated Sorting Genetic Algorithm (NSGA-II)

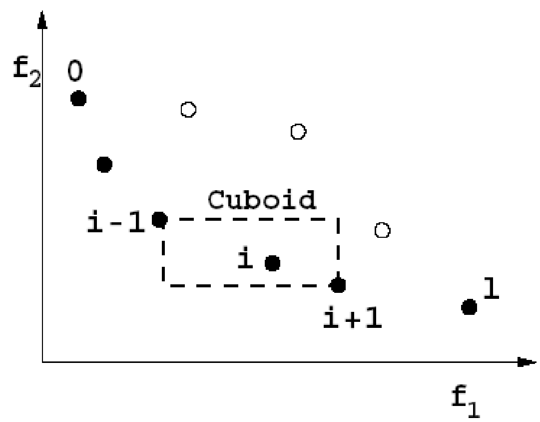
Elites are preserved



Maintaining Diversity in NSGA-II

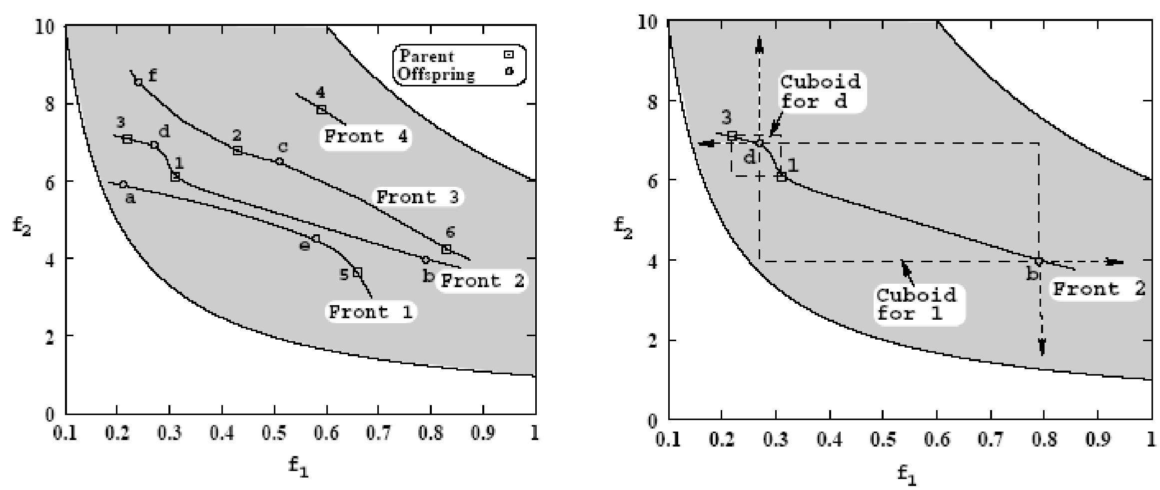
Fronts F_1 and F_2 are chosen by elitism.
Only a part of F_3 can be retained in the population of N.
Hence, choose diverse solutions in F_3 to have diversity.

Sort F_3 *population according one fitness at a time.*
Calculate the distances to the neighbors as shown below.
Solutions in F_3 *with the highest distances are chosen to fill in.*



An Illustration of NSGA-II

Six parents and six offspring

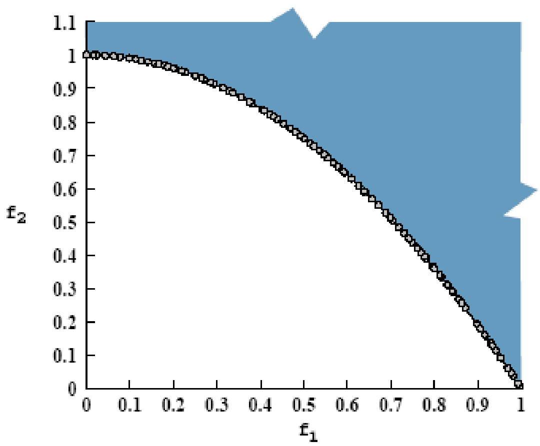


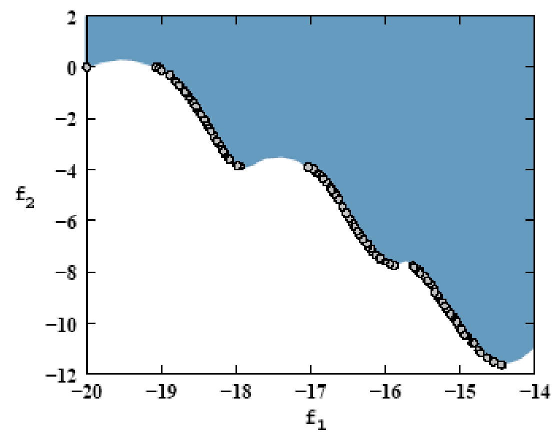
Parents after one iteration: (a,3,1,e,5,b)

NSGA-II on Test Problems

(Min) $f_1(\mathbf{x}) = x_1$
(Min) $f_2(\mathbf{x}) = g\left[1 - (f_1/g)^2\right]$
where $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$

(Min) $f_1(\mathbf{x}) = x_1$
(Min) $f_2(\mathbf{x}) = g\left[1 - \sqrt{\frac{f_1}{g}} - \frac{f_1}{g}\sin(10\pi f_1)\right]$
where $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$

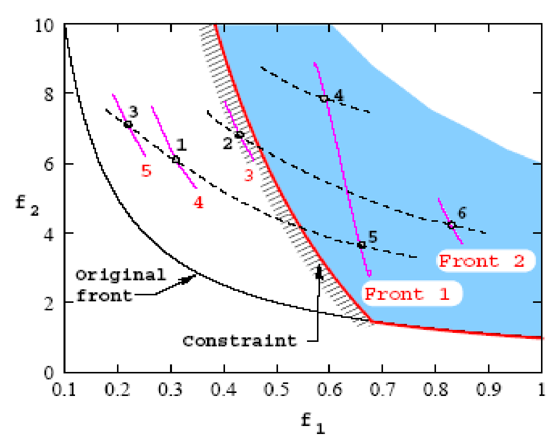




Constrain-Domination Principle

A solution i constrained-dominates a solution j, if any is true:

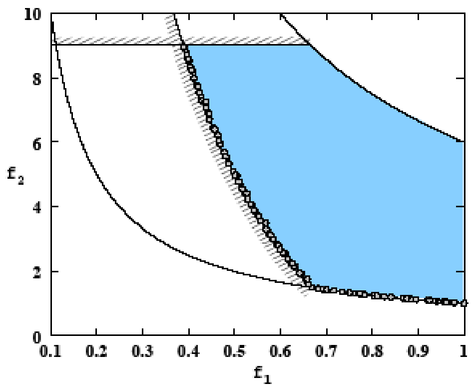
1. Solution i is feasible and solution j is not.
2. Solutions i and j are both infeasible, but solution i has a smaller overall constraint violation.
3. Solutions i and j are feasible and solution i dominates solution j.

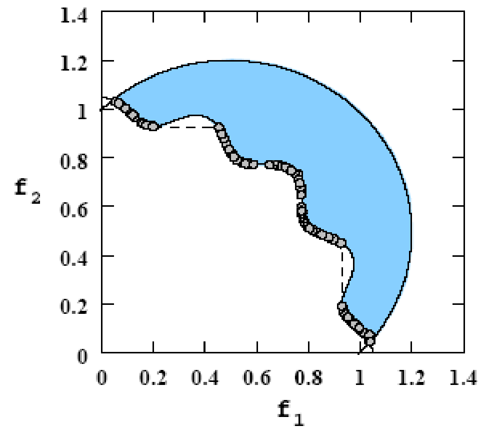


Constrained NSGA-II Simulation Results

$$
\begin{aligned}
&(\text{Min}) \quad f_1(\mathbf{x}) = x_1 \\
&(\text{Min}) \quad f_2(\mathbf{x}) = \frac{1+x_2}{x_1} \\
&x_2 + 9x_1 \geq 6 \\
&-x_2 + 9x_1 \geq 1
\end{aligned}
$$

$$
\begin{aligned}
&(\text{Min}) \quad f_1(\mathbf{x}) = x_1 \\
&(\text{Min}) \quad f_2(\mathbf{x}) = x_2 \\
&x_1^2 + x_2^2 - 1 - \tfrac{1}{10}\cos\left(16\tan^{-1}\tfrac{x_1}{x_2}\right) \geq 0 \\
&(x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5
\end{aligned}
$$

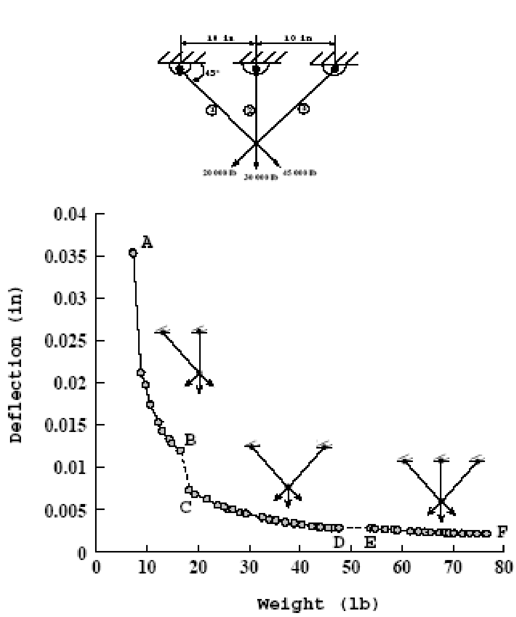


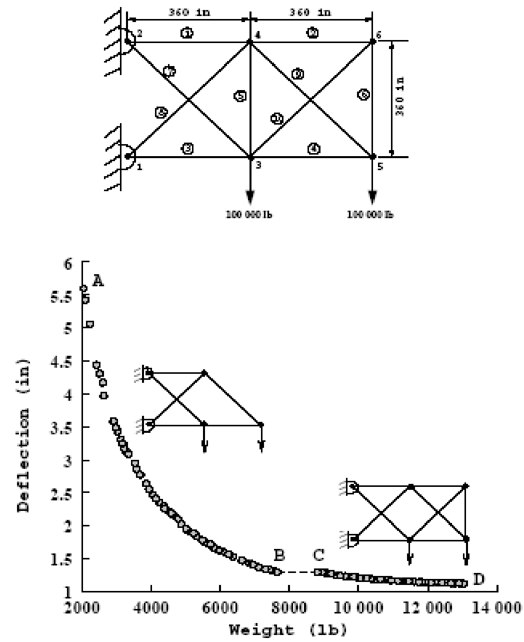


Applications of Multiobjective Optimization

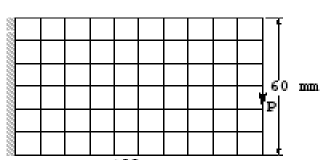
1. Identify different trade-off solutions for choosing one
2. Understanding insights about the problem
   - Reveal common properties among P-O solutions
   - Identify what causes trade-offs
   - Such information are valuable to users
   - May not exist other means of finding above
3. To aid in other optimization tasks

Revealing Salient Insights: Truss Structure Design

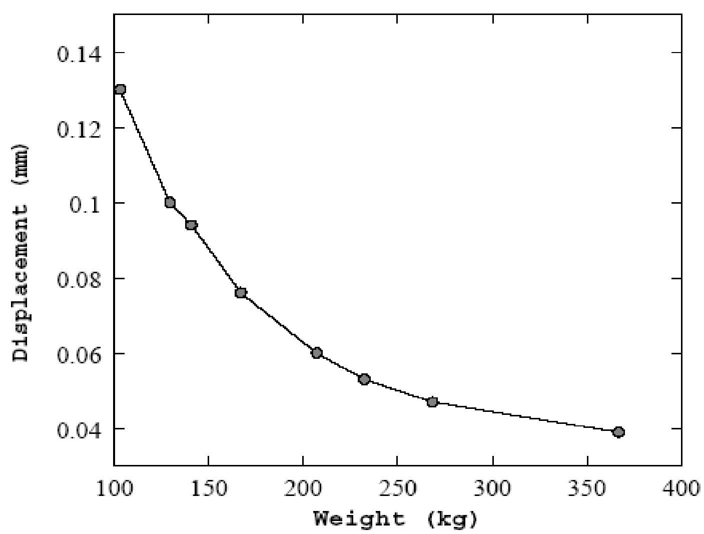




Revealing Salient Insights: A Cantilever Plate Design



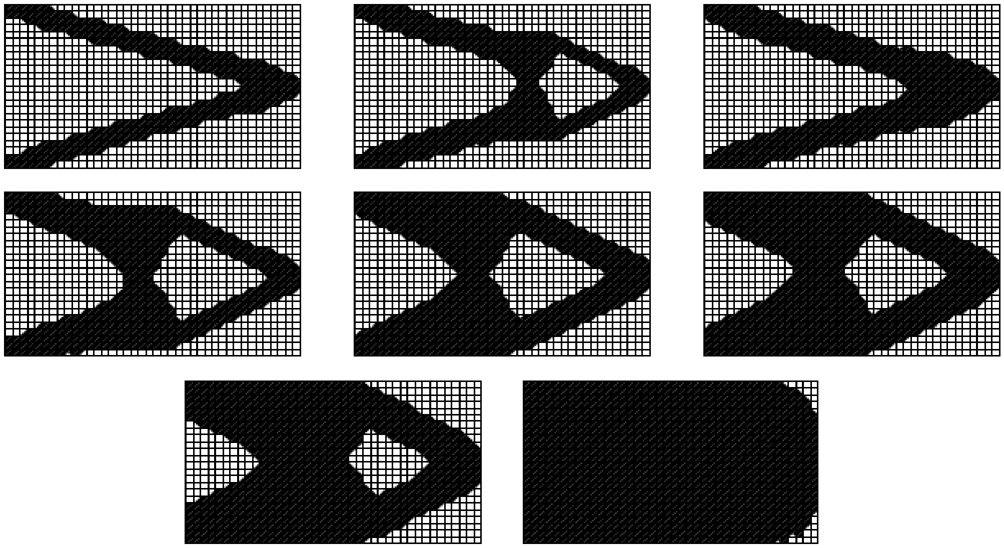
Base plate



Eight trade-off solutions are chosen

Trade-Off Solutions

- Symmetry in solutions about mid-plane, discovery of stiffener



Two-Objective Test Problems

- Pareto-optimal front is controllable and known
- ZDT problems:

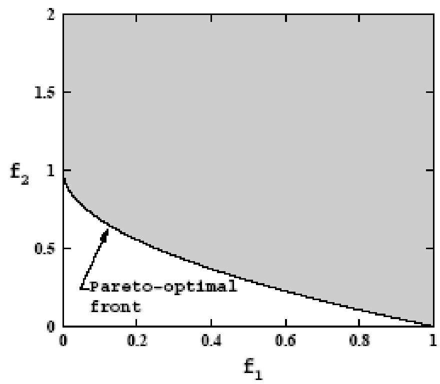
$$\begin{aligned} \text{Min.} \quad & f_1(\mathbf{x}) = f_1(\mathbf{x}_I), \\ \text{Min.} \quad & f_2(\mathbf{x}) = g(\mathbf{x}_{II})h(f_1, g). \end{aligned}$$

- Choose $f_1()$, $g()$ and $h()$ to introduce various difficulties

Zitzler–Deb–Thiele's Test Problems

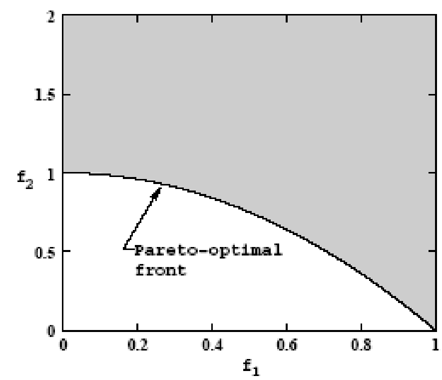
ZDT1

$$\begin{aligned} f_1(\mathbf{x}) &= x_1, \\ g(\mathbf{x}) &= 1 + \tfrac{9}{n-1} \textstyle\sum_{i=2}^{n} x_i, \\ h(f_1, g) &= 1 - \sqrt{f_1/g}. \end{aligned}$$



ZDT2

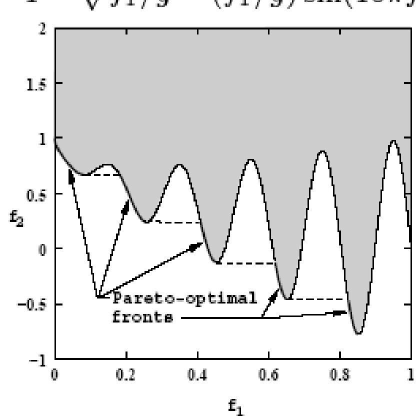
$$\begin{aligned} f_1(\mathbf{x}) &= x_1, \\ g(\mathbf{x}) &= 1 + \tfrac{9}{n-1} \textstyle\sum_{i=2}^{n} x_i, \\ h(f_1, g) &= 1 - (f_1/g)^2. \end{aligned}$$



Zitzler–Deb–Thiele's Test Problems

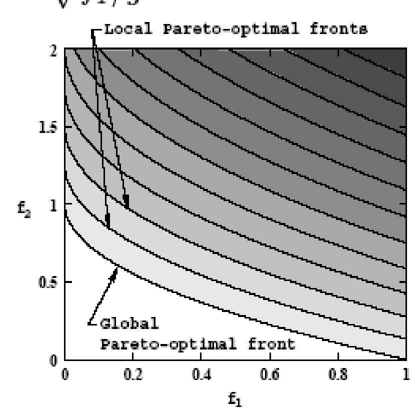
ZDT3

$$f_1 = x_1,$$
$$g = 1 + \tfrac{9}{n-1} \textstyle\sum_{i=2}^{n} x_i,$$
$$h = 1 - \sqrt{f_1/g} - (f_1/g)\sin(10\pi f_1).$$



ZDT4

$$f_1 = x_1,$$
$$g = 10n - 9 + \textstyle\sum_{i=2}^{n}(x_i^2 - 10\cos(4\pi x_i)),$$
$$h = 1 - \sqrt{f_1/g}.$$

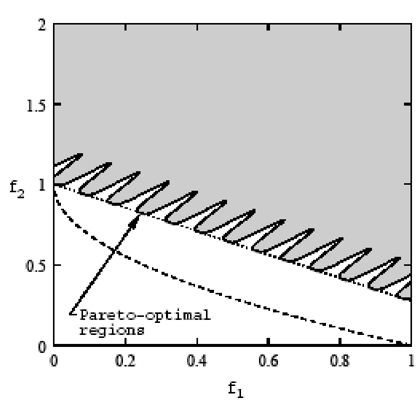


Constrained Test Problem Generator

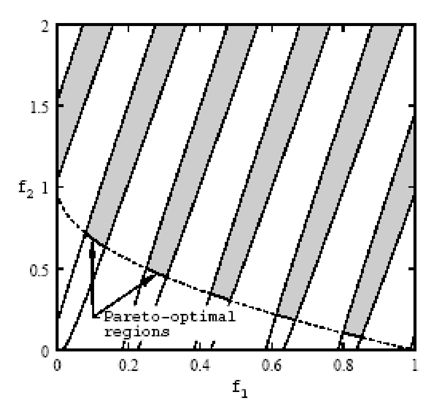
$$
\begin{aligned}
\text{Minimize} \quad & f_1(\mathbf{x}) = x_1 \\
\text{Minimize} \quad & f_2(\mathbf{x}) = g(\mathbf{x})\left(1 - \frac{f_1(\mathbf{x})}{g(\mathbf{x})}\right) \\
\text{Subject to} \quad & c(\mathbf{x}) \equiv \cos(\theta)(f_2(\mathbf{x}) - e) - \sin(\theta) f_1(\mathbf{x}) \geq \\
& \qquad a \left|\sin\left(b\pi \left(\sin(\theta)(f_2(\mathbf{x}) - e) + \cos(\theta) f_1(\mathbf{x})\right)^c\right)\right|^d
\end{aligned}
$$

Various Parameter Settings

$$\theta = -0.2\pi, \quad b = 10, \quad c = 1, \quad e = 1.$$



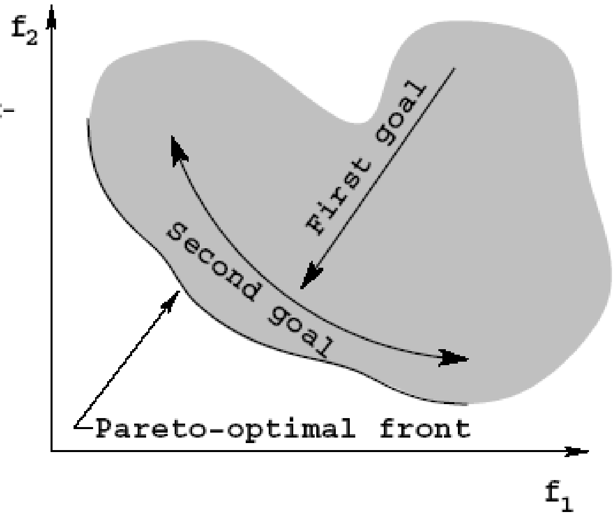
CTP2: $d = 6$ and $a = 0.2$



CTP 7: $\theta = -0.05\pi, a = 40, b = 5, c = 1, d = 6, e = 0$

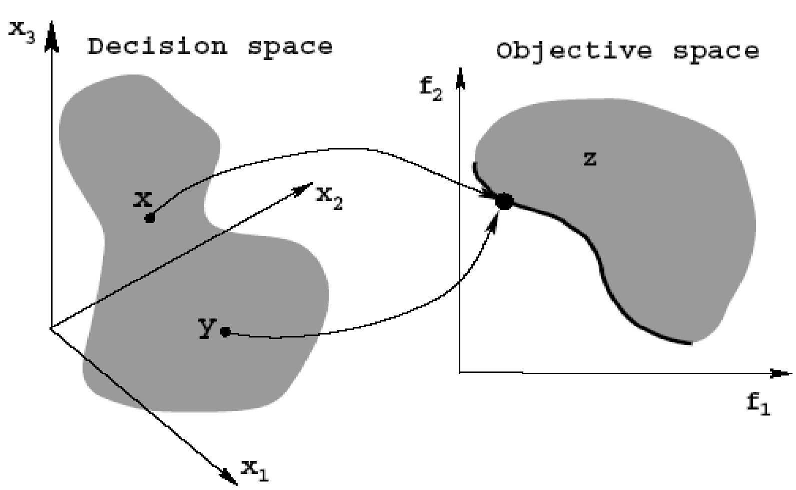
Performance Metrics

- Two essential metrics (functionally)
  - Convergence measure
  - Diversity measure



Multi-Modal EMOs

- Different solutions having identical objective values
- Multi-modal Pareto-optimal solutions: Design, Bioinformatics



Other Problem Scenarios

- In addition to the bound constrained single objective, general constraints, multi-objective, multimodal-niching, there are additional problem scenarios too.
- Dynamic Optimization: Objective function or constraints changing with time.
- Expensive Optimization: Evaluation of objective function or constraint equation suffering from excessive complexity. Surrogate methods are used.
- Large Scale Optimization: Huge number of decision variables or large number of objectives in a multiobjective optimization problem (MOP).
- Noisy Objective Function: Objective function is not accurate.
- Robust Optimization: Searches for robust optimal solution.
- Several combinations of the above scenarios possible.