Multiobjective Optimization

Winter 2024

Outline

- What is multi-objective optimization?
- Dominance and Pareto Front
- Classical Approaches
- Multi-objective GA
  - NSGA II (Elitism Non-dominated sorting GA)
  - SPEA II (Strength Pareto Evolutionary Algorithm)
- Multi-objective PSO

Multi-objective optimization (MO)

- So far, our optimization target is expressed as a single objective function
- Practical problems, however, often have multiple (conflicting) objectives
- How to simultaneously optimize multiple objectives is practically relevant
- Website of references

http://delta.cs.cinvestav.mx/~ccoello/EMOO/

Examples of multi-objective problems

- In truss construction, a good design is characterized by low total mass and high reliability
- Time/cost/quality/safety/sustainability tradeoff in project management
- Design of product leads to different cost and performance level
- In design of infrastructure facility, objectives include life-cycle maintenance cost and degradation status
- Portfolio optimization is to minimize the risk and maximize the return

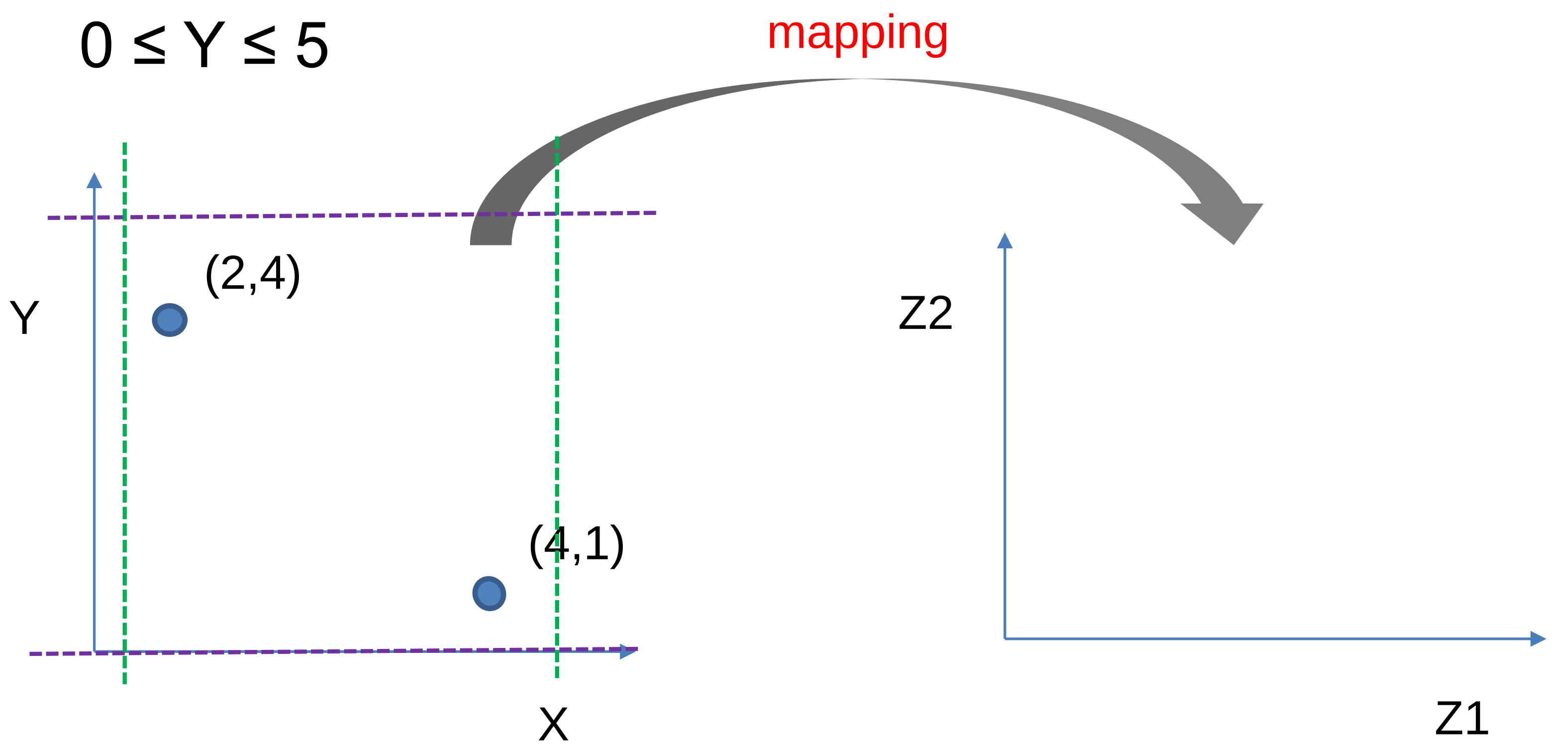
Search Space and Objective Space

Min $Z1 = Y/X$

Min $Z2 = 2X - Y$

$1 \le X \le 6$

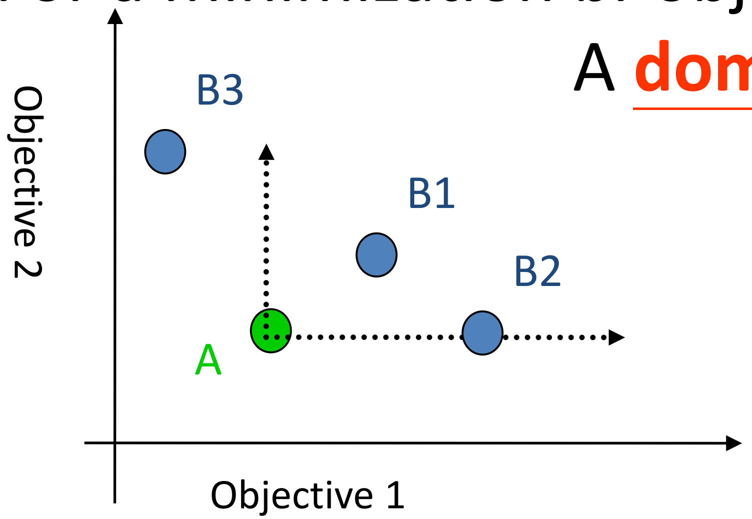
$0 \le Y \le 5$



Dominance

- Solution A **dominates** B if A has a better objective value for at least one of the objective functions and is not worse with respect to the remaining objective functions
- For a minimization bi-objective problem,

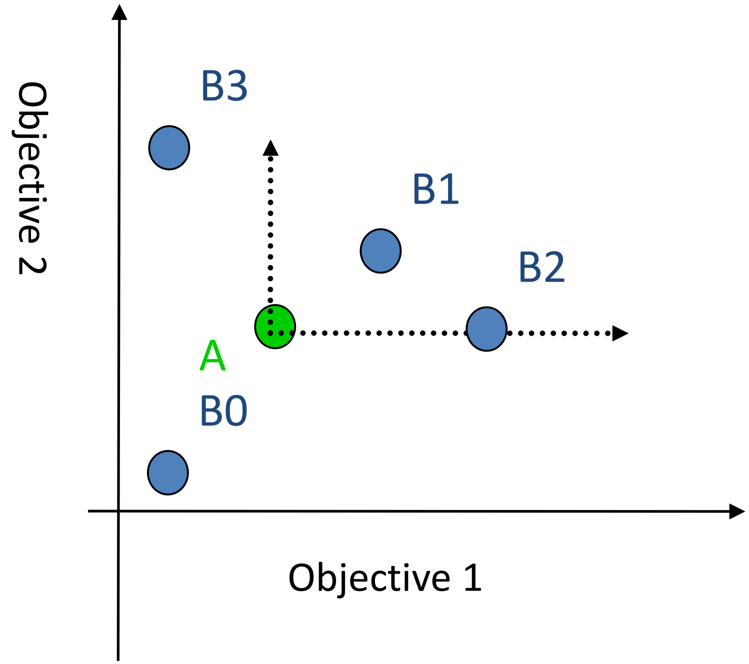
A **dominates** B1, B2, but not B3



Which space this is?

Dominance relationships

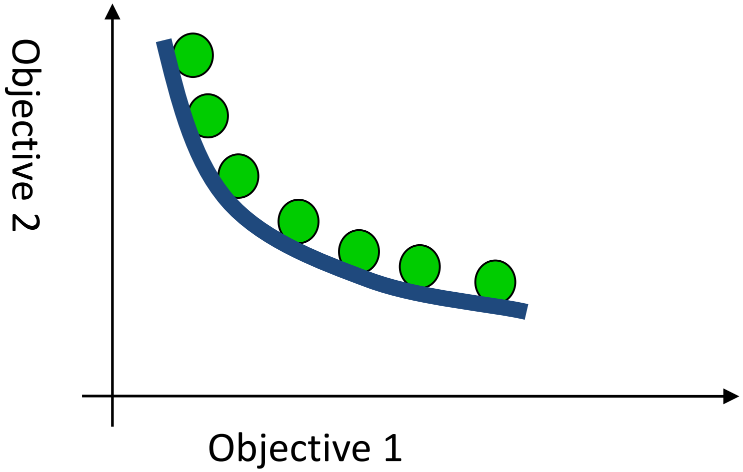
- Three possibilities (minimization problem)
  1. A dominates B1 and B2
  2. A is dominated by B0
  3. A and B3 are non-dominated to each other



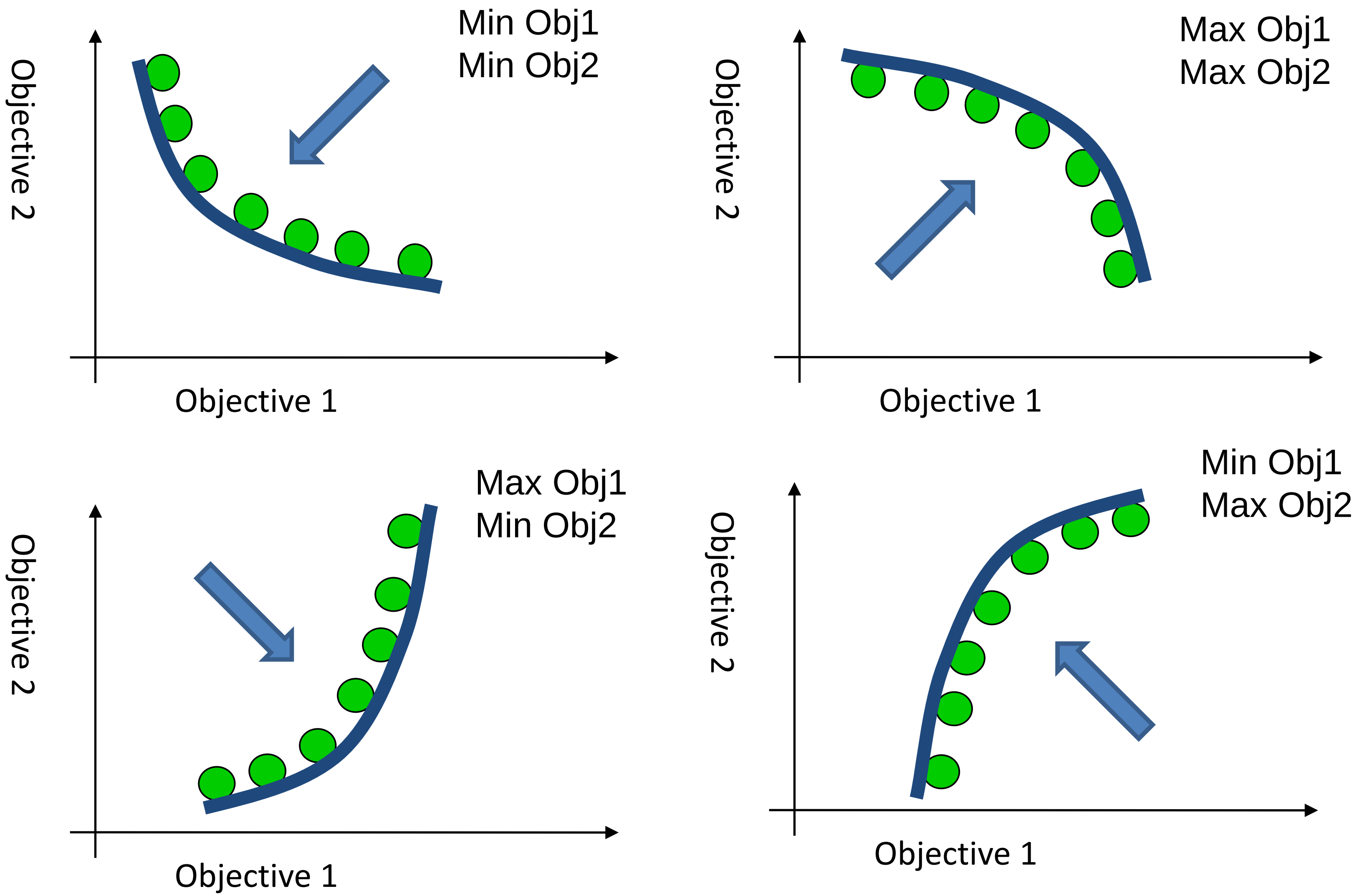
Pareto front

- A non-dominated solution **has no solution can be found that dominates it**
- The set of non-dominated solutions is called the Pareto front

Vilfredo Pareto
(1848–1923)



Front directions



Classical approach 1: weighted sum

- Weighted sum of objective functions

  Obj* = α (Obj_1) + β (Obj_2)

  Then, solve it as a single objective problem
- Drawback: adding different units is unnatural, e.g., how to add public safety to life-cycle cost? Also, weights may be hard to assess
- Improvement: adaptive weights; changing the weight during the optimization process

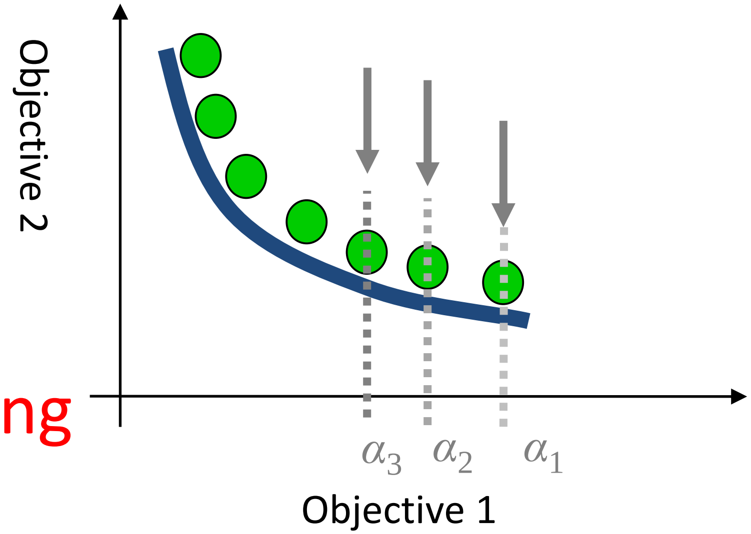
Classical approach 2: ε- constraint

- Optimize one objective function while treating others as constraints
- Gradually change the constraints to constitute the entire Pareto front

Minimize Obj_2

s.t.

Obj_1 $\leqq \alpha$

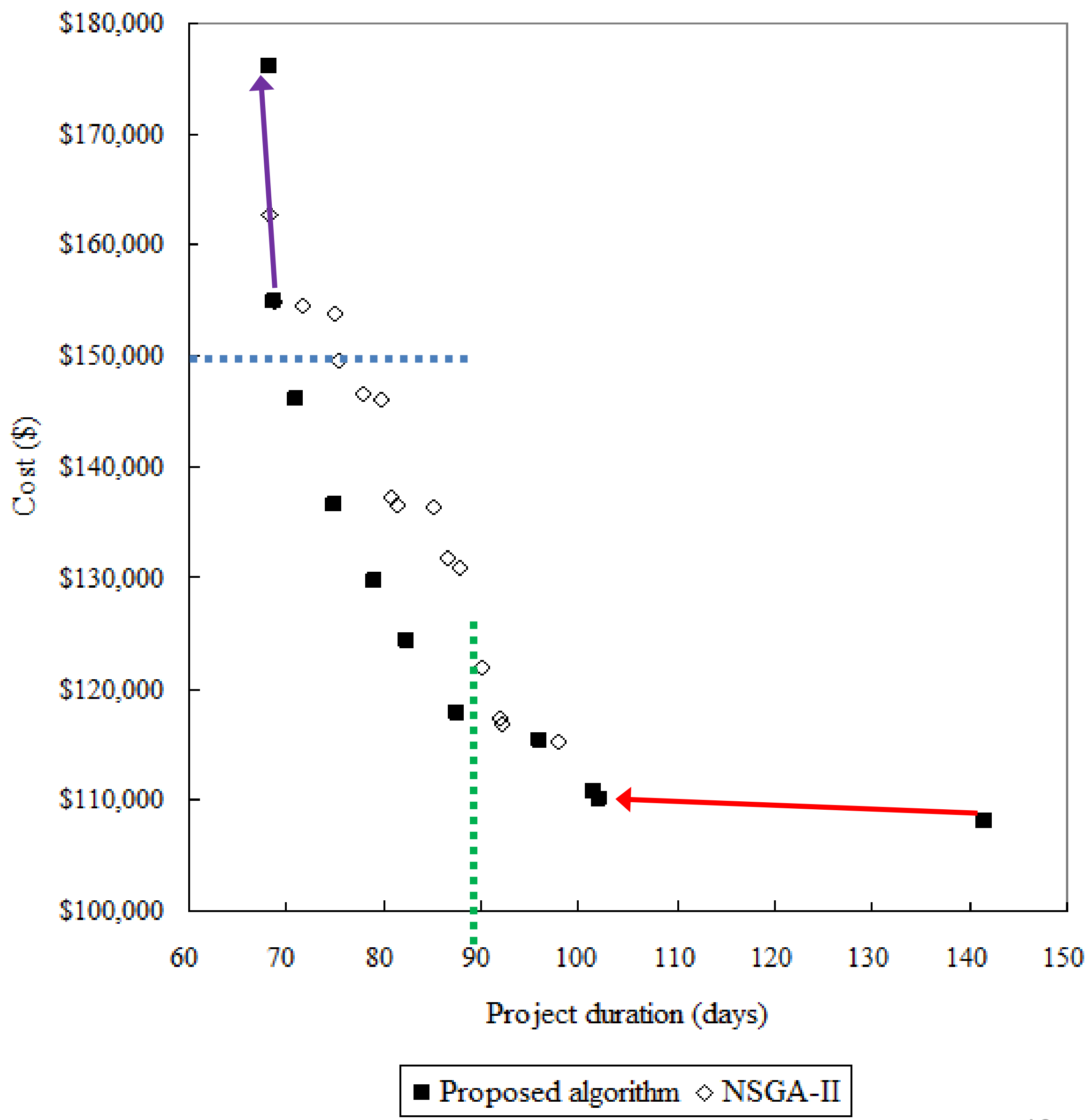


- Disadvantage: time-consuming for a large or infinite set

Goal of MO

- Find the entire Pareto front
- Let decision makers choose the optimal compromise; it is much easier once the alternatives are present
- Some decision support systems allow decision makers to manually guide the search during process

MO-assisted Decision-making Example

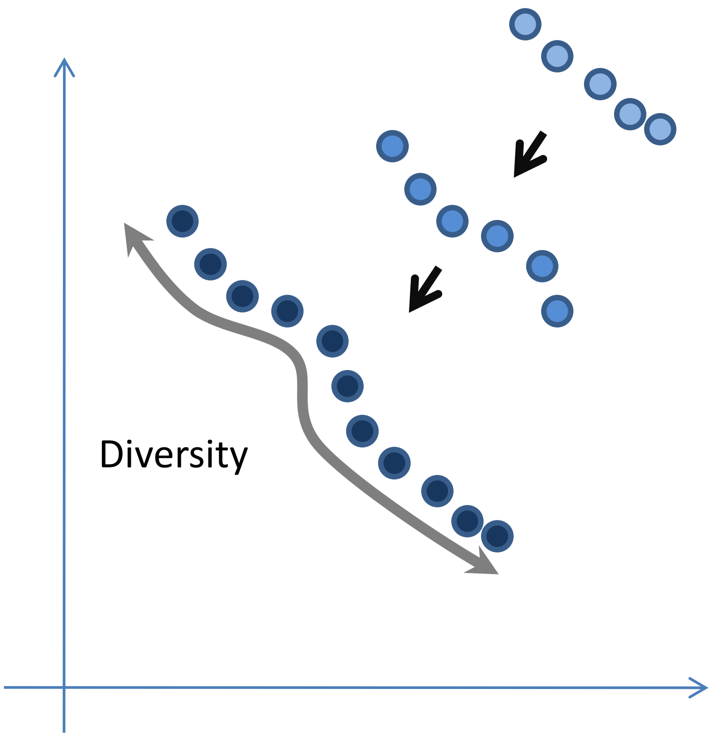


Keys

- Fitness assignment (approach the entire Pareto front)
- Preserve diversity (keep proper distances among solutions)
- Elitism (external collection of good solutions)

Issues

- Density estimation:
  - How to maintain a diverse non-dominated set
- Archiving (Elitism):
  - How to prevent non-dominated solution being lost
- Fitness assignment:
  - How to guide the population towards the front



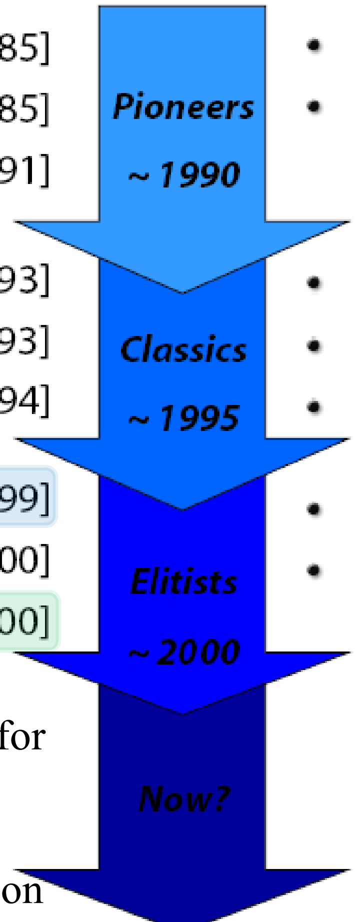
What is this space?

History of Multiobjective GA

VEGA [Schaffer 85]
[Fourman 85]
ESVO [Kursawe 91]

MOGA [Fonseca, Fleming 93]
NPGA [Horn, Nafpliotis 93]
NSGA [Srinivas, Deb 94]

SPEA [Zitzler, Thiele 99]
PAES, PESA [Knowles, Corne 99/00]
NSGA-II [Deb et al. 00]



- Objective-wise selection
- Proof-of principle

- Pareto-based selection
- Niching
- Visual comparisons

- Archiving + elitism
- Quantitative performance metrics

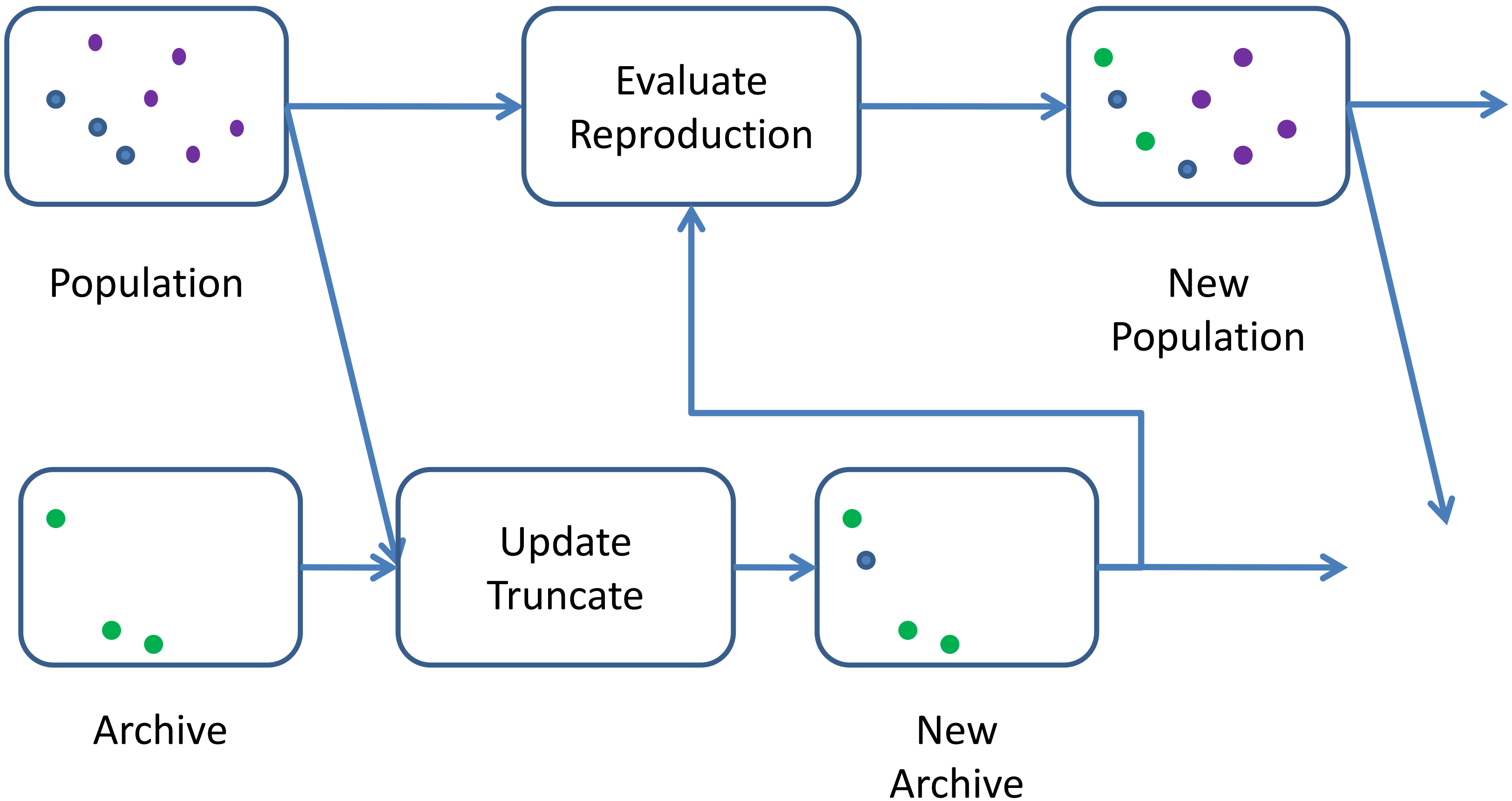
Zitzler, E.

Deb, K.

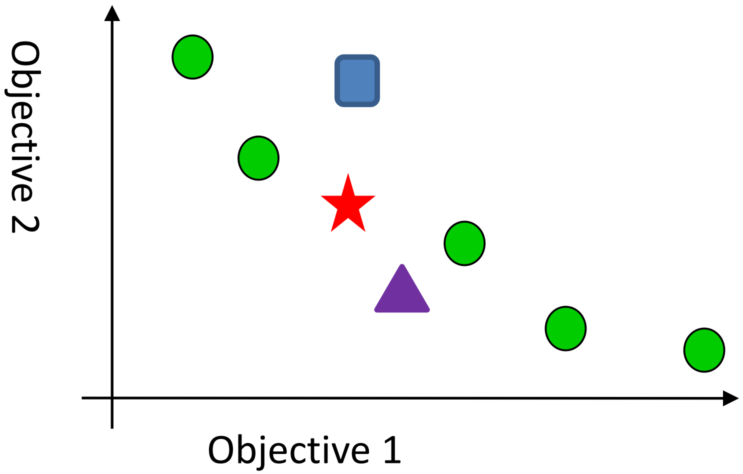
Zitzler (2001) "Evolutionary Algorithms for Multiobjective Optimization"

Singh (2016) "A Comprehensive Review on Multi-Objective Optimization Using Genetic Algorithms"

Elitism

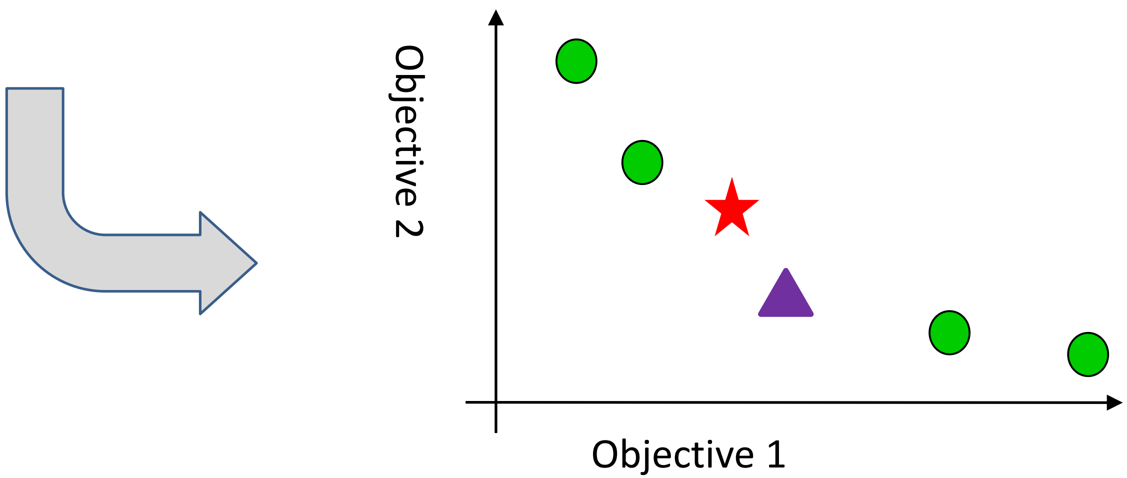

Evaluate
Reproduction
Population
New
Population
Update
Truncate
Archive
New
Archive

Archive Update

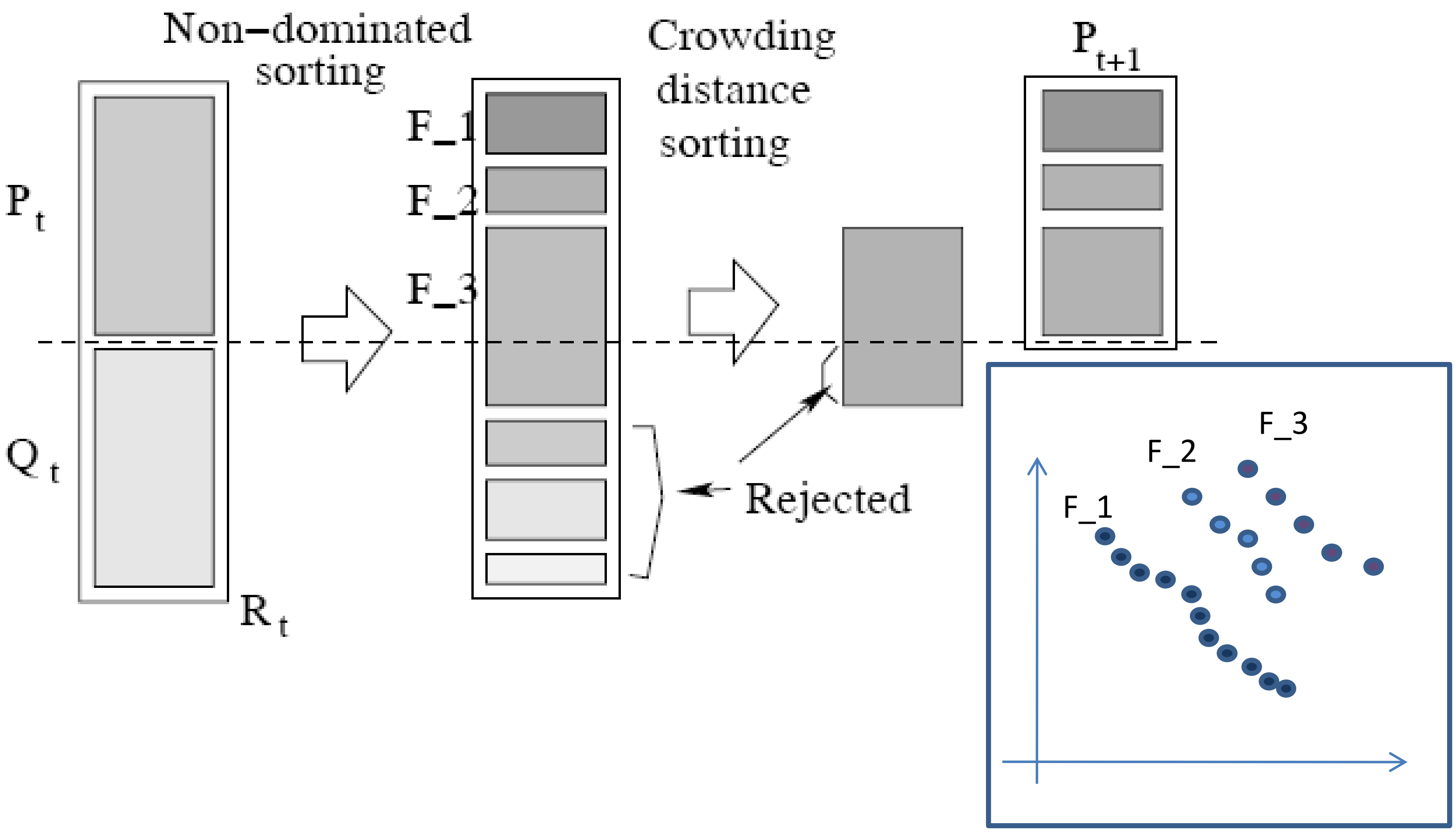


Three possibilities

- New solution is dominated by some of the front solutions
- New solution dominates some of the front solutions
- New solution is non-dominated to any of the front solutions



NSGA II: Sketch



NSGA II: Main loop

$R_t = P_t \cup Q_t$	combine parent and children population
$\mathcal{F} =$ `fast-nondominated-sort(`R_t`)`	$\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2, \ldots)$, all non-dominated fronts of R_t
$P_{t+1} = \emptyset$ and $i = 1$	
until $\lvert P_{t+1}\rvert + \lvert\mathcal{F}_i\rvert \leq N$	till the parent population is filled
`crowding-distance-assignment(`$\mathcal{F}_i$`)`	calculate crowding distance in $\mathcal{F}_i$
$P_{t+1} = P_{t+1} \cup \mathcal{F}_i$	include i-th non-dominated front in the parent pop
$i = i + 1$	check the next front for inclusion
Sort$(\mathcal{F}_i, \prec_n)$	sort in descending order using $\prec_n$
$P_{t+1} = P_{t+1} \cup \mathcal{F}_i[1 : (N - \lvert P_{t+1}\rvert)]$	choose the first $(N - \lvert P_{t+1}\rvert)$ elements of F_i
$Q_{t+1} =$ `make-new-pop(`P_{t+1}`)`	use selection, crossover and mutation to create a new population Q_{t+1}
$t = t + 1$	increment the generation counter

NSGA II: fast-nondominated-sorting

At the end, solutions are categorized into multiple fronts

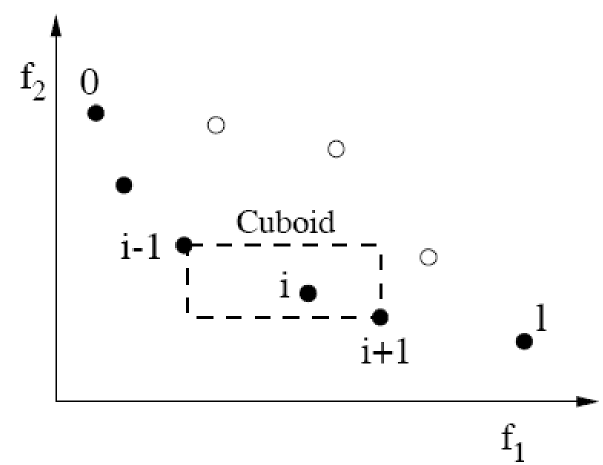
$\mathcal{F} = \texttt{fast-non-dominated-sort}(P)$	$\mathcal{F}$ is a set of non-dominated fronts
$i = 1$	i is the front counter and is initialized to one
until $P \neq \emptyset$	
$\quad \mathcal{F}_i = \texttt{find-nondominated-front}(P)$	find the non-dominated front
$\quad P = P \backslash \mathcal{F}_i$	remove non-dominated solutions from P
$\quad i = i + 1$	increment the front counter

$P' = \texttt{find-nondominated-front}(P)$	
$P' = \{1\}$	include first member in P'
for each $p \in P \wedge p \notin P'$	take one solution at a time
$\quad P' = P' \cup \{p\}$	include p in P' temporarily
$\quad$ for each $q \in P' \wedge q \neq p$	compare p with other members of P'
$\quad\quad$ if $p \succ q$, then $P' = P' \backslash \{q\}$	if p dominates a member of P', delete it
$\quad\quad$ else if $q \succ p$, then $P' = P' \backslash \{p\}$	if p is dominated by other members of P', do not include p in P'

NSGA II: crowding-distance-assignment

`crowding-distance-assignment`$(\mathcal{I})$	
$l = \lvert\mathcal{I}\rvert$	number of solutions in $\mathcal{I}$
for each i, set $\mathcal{I}[i]_{distance} = 0$	initialize distance
for each objective m	
$\quad \mathcal{I} = \mathrm{sort}(\mathcal{I}, m)$	sort using each objective value
$\quad \mathcal{I}[1]_{distance} = \mathcal{I}[l]_{distance} = \infty$	so that boundary points are always selected
$\quad$ for $i = 2$ to $(l-1)$	for all other points
$\qquad \mathcal{I}[i]_{distance} = \mathcal{I}[i]_{distance} + \underline{(\mathcal{I}[i+1].m - \mathcal{I}[i-1].m)}$	

m-th objective function value
for the *i*-th individual



NSGA II: partial order

$$i \prec_n j \quad \text{if } (i_{rank} < j_{rank}) \text{ or } ((i_{rank} = j_{rank}) \text{ and } (i_{distance} > j_{distance})\ )$$

Put *i* in front of *j* if *i* is in a sparse area; this is to encourage diversity

- *i* and *j* are individuals
- *rank* (non-domination rank):
  The current non-dominated subset of the population is assigned rank 0 and is then temporarily removed from consideration.
  The remaining population is then evaluated to determine another non-dominated subset, which is then given rank 1 and removed from consideration…
- *distance*: crowding distance

SPEA II

- Initialization:
  - generate an initial population P(0) and empty archive A(0)
- Fitness assignment
  - Assign fitness based on the strengths of the dominators
  - Minimize the fitness value
  - Also consider density (the smaller the better)
- Environmental selection
  - Copy nondominated solutions in A(t) and P(t) to A(t+1)
- Termination check
- Mating selection
  - Tournament selection on A(t+1) to fill the mating pool
- Variation
  - Generate P(t+1)

SPEA II: strengths of the dominators

$$S(i) = |\{ j \mid j \in P(t) + A(t) \wedge i \succ j \}|$$

S(i): number of solutions individual i dominates

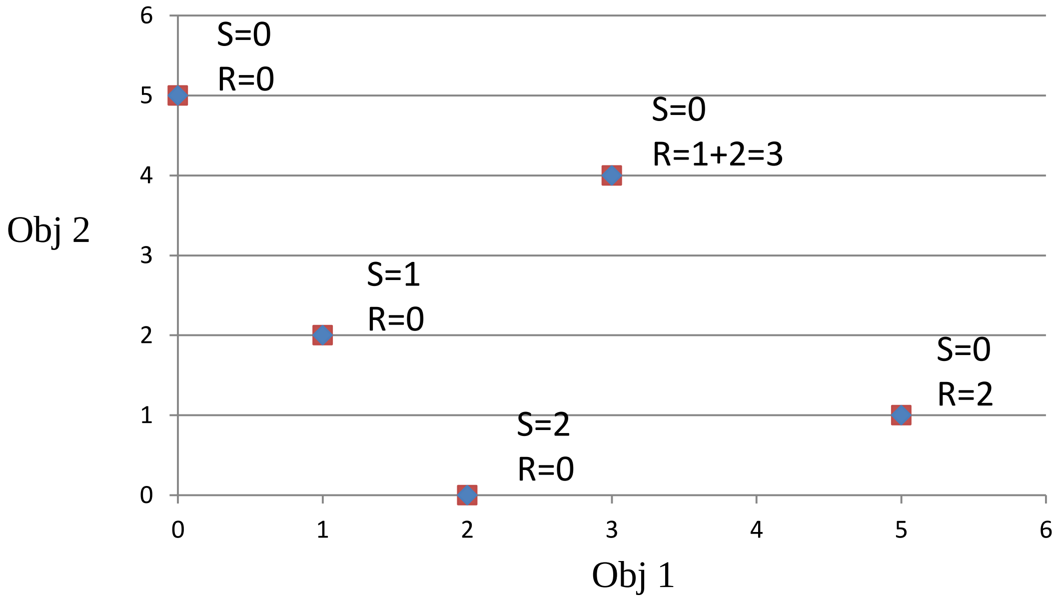
$$R(i) = \sum_{j \in P(t) + A(t) \wedge j \succ i} S(j)$$

R(i): strengths of the dominators for individual i

Give preference to stronger individuals that are dominated by none or few individuals

Example: strengths of the dominators

- Consider a bi-objective minimization problem
- Five solution: (1,2) (0,5) (3,2) (5,1) (2,7)



$$S(i) = |\{ j \mid j \in P(t) + A(t) \wedge i \succ j \}|$$

$$R(i) = \sum_{j \in P(t) + A(t) \wedge j \succ i} S(j)$$

SPEA II: density

- To discriminate individuals having identical strengths of dominators
- The *k*-th nearest neighbor method
  - The **inverse** of the distance to the *k*-th nearest neighbor; usually *k*=1
- Give preference to **more sparse areas** (less represented areas, where individual has a larger distance to the neighbors)

Multi-objective (Elitist) PSO (1)

- Proposed by Coello Coello (2002)
  "MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization"

Coello Coello

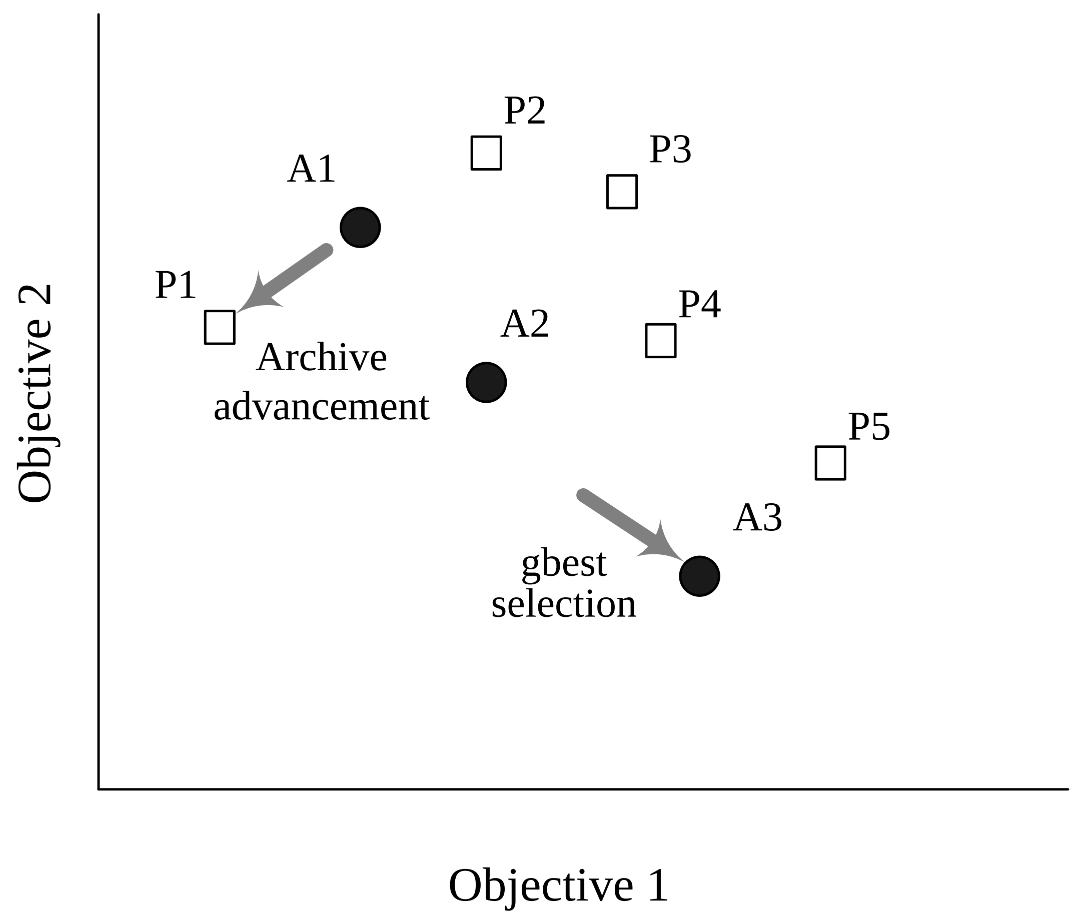
- Maintaining an **elite archive** as a repository for non-dominated solutions and **choosing members of the archive to direct further search**
- The elite archive accepts a new position of a particle if it is non-dominated by all solutions already stored. All dominated members of the archive are deleted.

Multi-objective (Elitist) PSO (2)

- The selection of pBest is simply replacing the previous best experience by the current position **if the former does not dominate the latter**.
- The selection of gBest is to promote exploration at the edges and sparse areas. This is implemented **by using the archive members that dominates fewest particles** in the current iteration as the gBest.

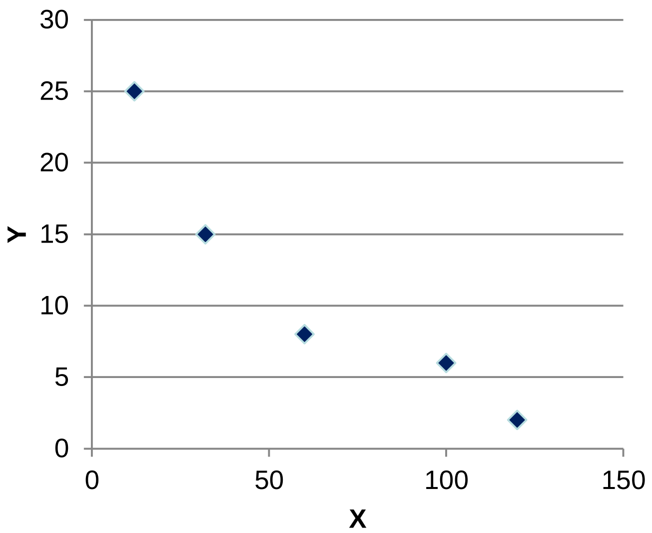
Elitist PSO: Sketch

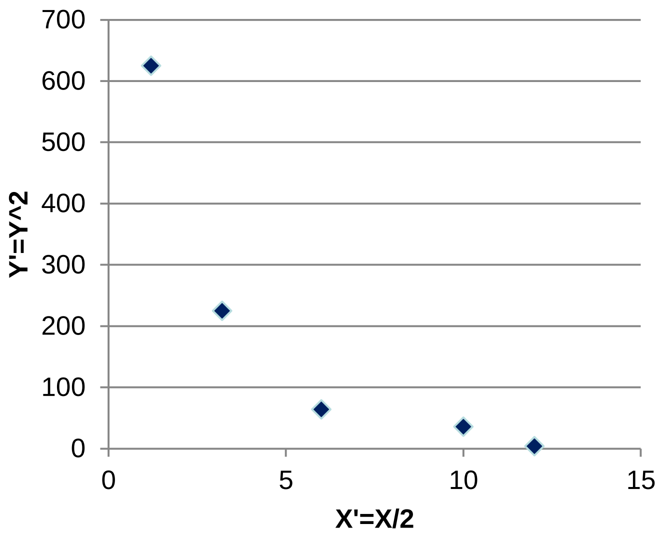
Key:
No consideration of "**distance**" between solutions



Advantage of Elitist PSO

- Using "distance" to force diversity may involve "**scaling problem**", i.e., different scales of objectives (i.e., measurement unit) would influence the optimization performance.





Elitist PSO in Pseudocode (1)

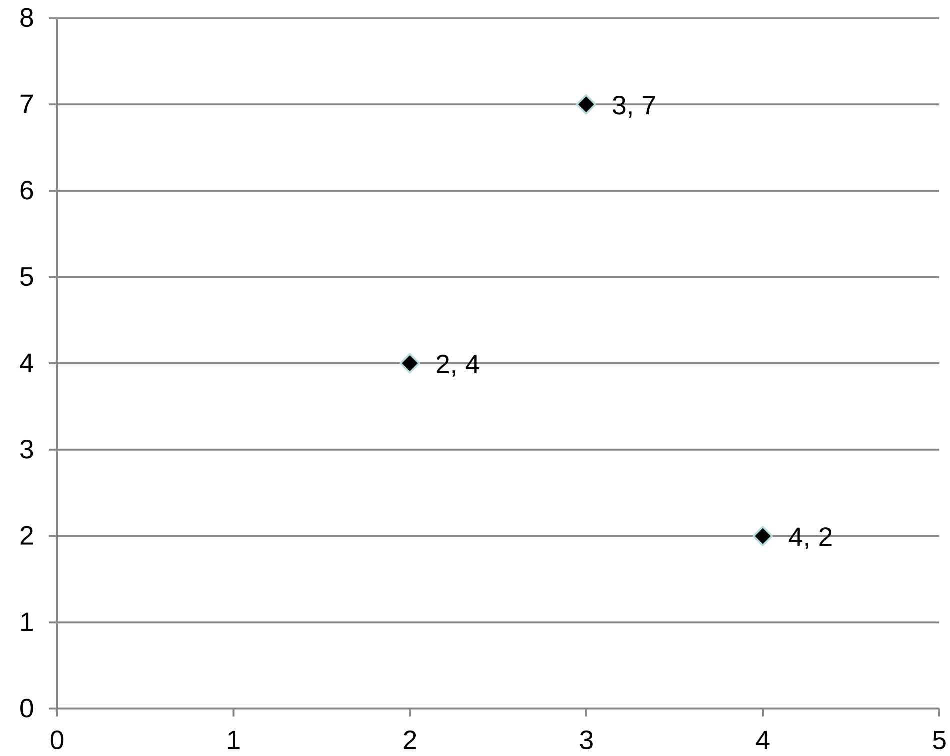
```
FOR each particle
    Initialize particle randomly
    Use the initial locations as the archive
WHILE maximum iteration or convergence criteria is not met
    FOR each particle
        Calculate fitness value F(i,t) corresponded to location X(i,t)
        IF F(i,t) is not dominated by pBest
          Replace pBest and pBestLocation
        END
        Update the archive (truncate the archive if necessary)
    END
```

Elitist PSO in Pseudocode (2)

```
Choose an archive element (which dominates the fewest
    particles in the current iteration) as gBest; if there is a
    tie, assign gBest randomly among the candidates
  FOR each particle
     Calculate particle velocity
     Update particle location
  END
END WHILE
Return Archive
```

Numerical Example (1)

- Minimization problem
- Initial swarm

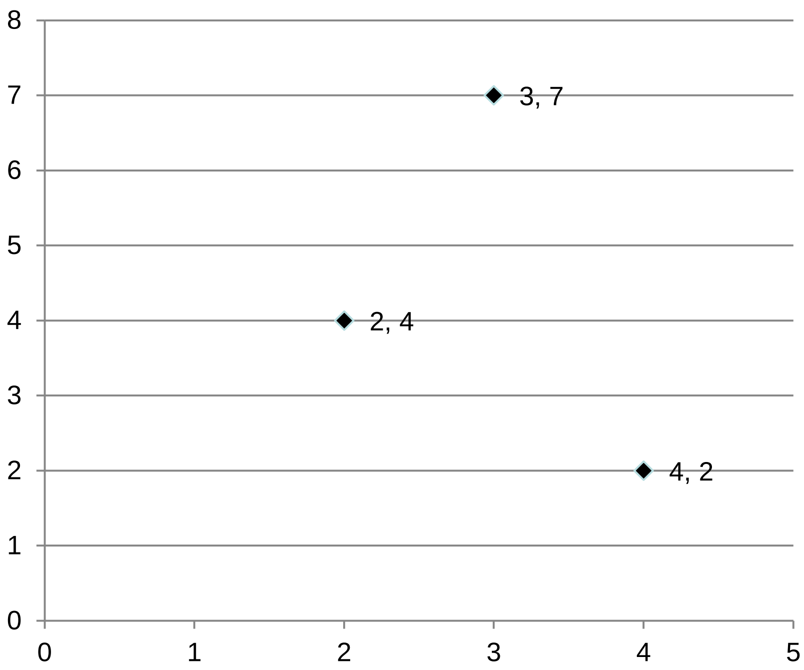


Which space this is?

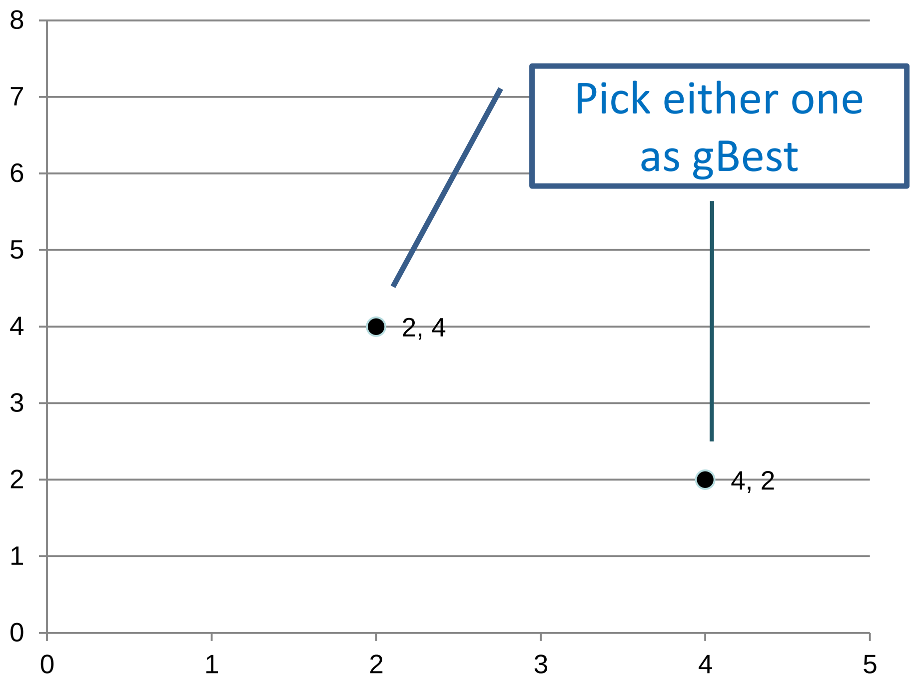
objective space

Numerical Example (2)

- pBest

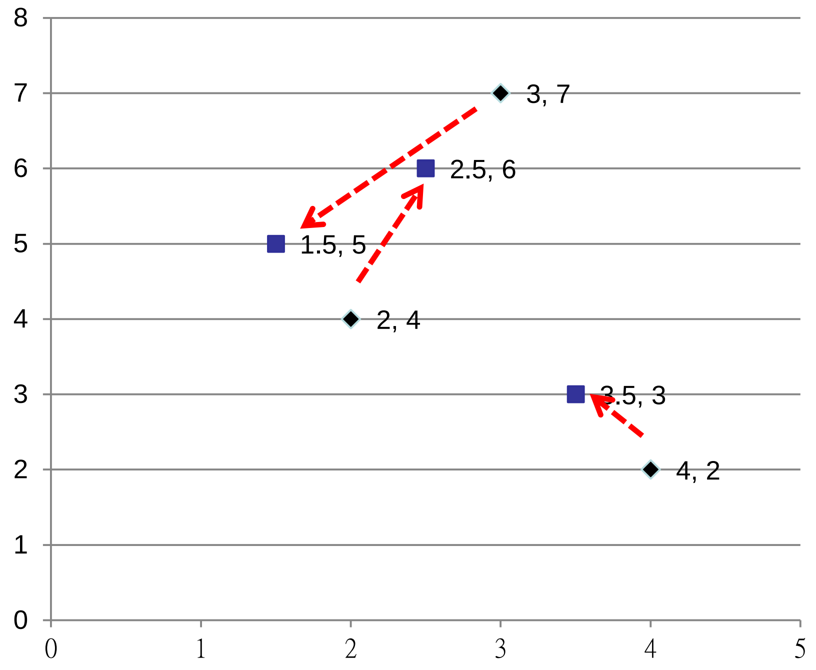


- Initial Archive

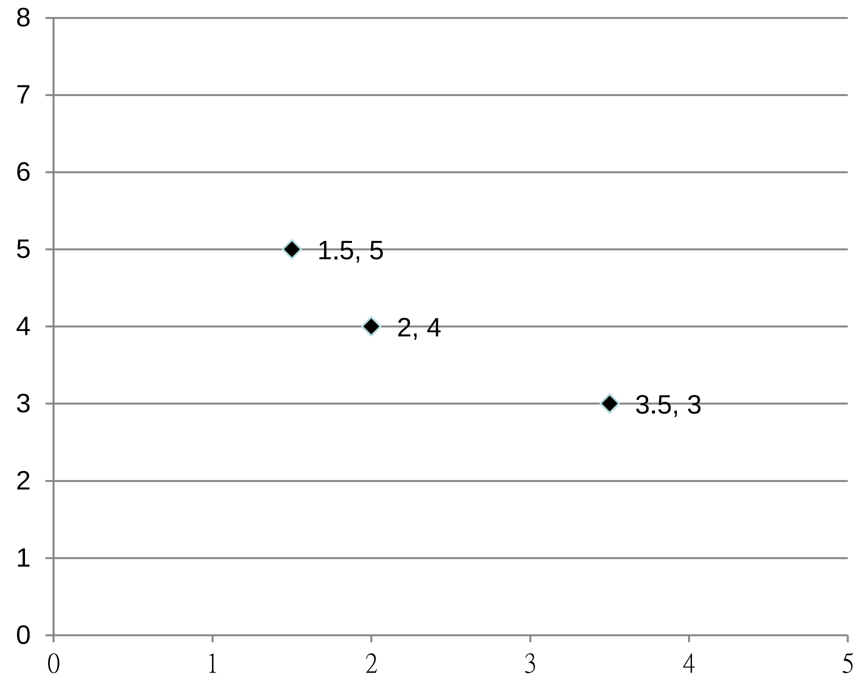


Numerical Example (3)

- Next iteration

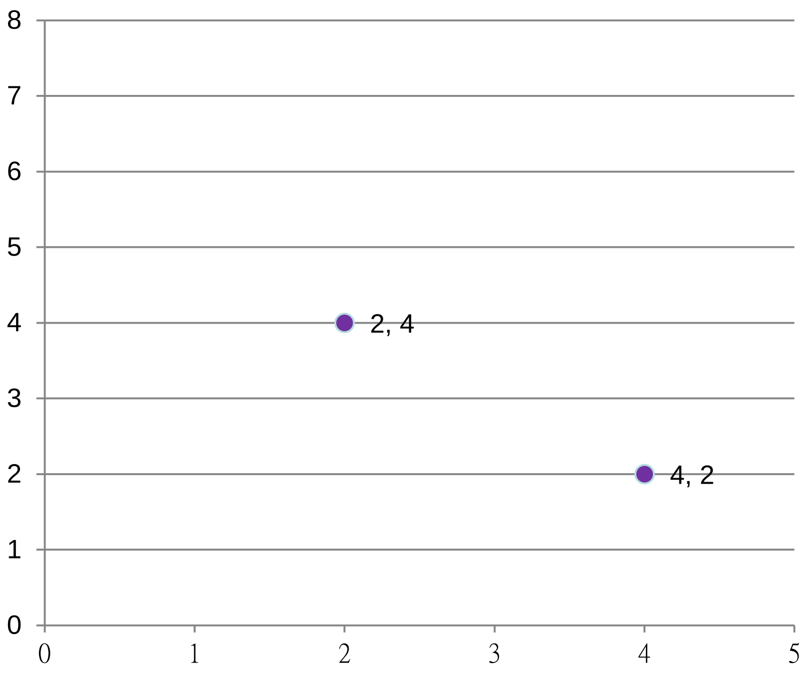


- Updated pBest

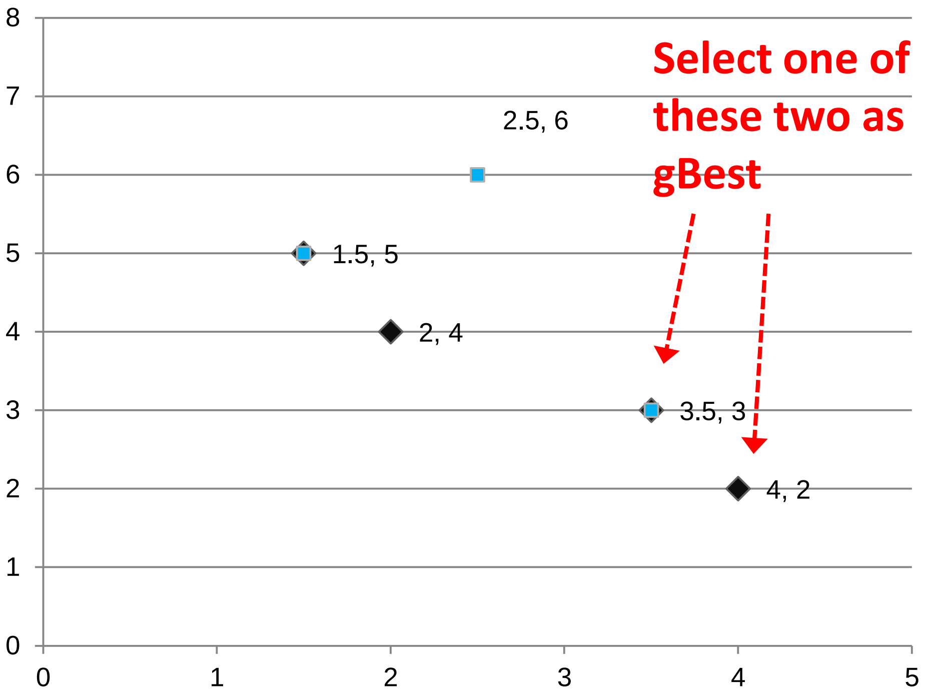


Numerical Example (4)

- Original archive



- Updated archive



Evaluation Metrics

- For single-objective optimization, it is straightforward to show the performance of a metaheuristic in terms of the objective value
- For MO-metaheuristic, we need to integrate different objective functions

we are concerned with

- ✓ Optimality of all the objectives
- ✓ Number of non-dominated solutions
- ✓ Solution diversity: good and even distribution

Multidimensional Objective Space

- The performance is measured in the N-dimensional objective space
- Since the ranges of objective function values may differ from one objective to another, one may normalize the original objective function values to values between 0 and 1
- We will discuss four kinds of metrics as follows

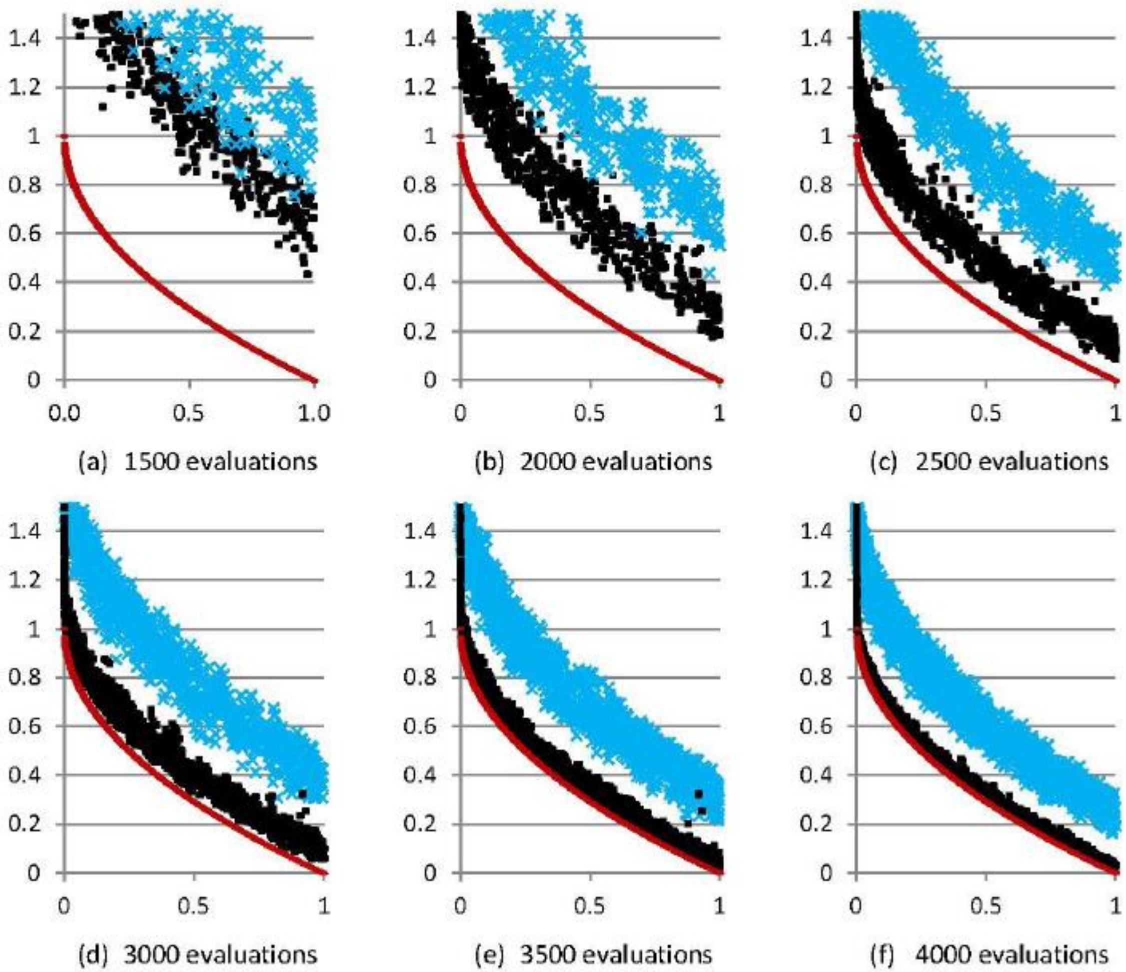
Possible Metrics (1)

- It may be necessary to consider **decision makers' personal preference**
- Simplest way is to transform the preference to individual weights of various objectives
- More advanced way is to obtain and apply the utility function of decision makers
- Utility function: the level of satisfaction a person derives from consuming a good or service

Possible Metrics (2)

- **Distance to the true front**: If the true Pareto front is known (e.g., benchmark problems), we can calculate how far the obtained results are from the true Pareto front
- The Euclidean distance is defined between each of the obtained non-dominated solutions and the closest member of the true Pareto front
- We may report the average and maximum distances
- The shorter the distance, the better the solutions

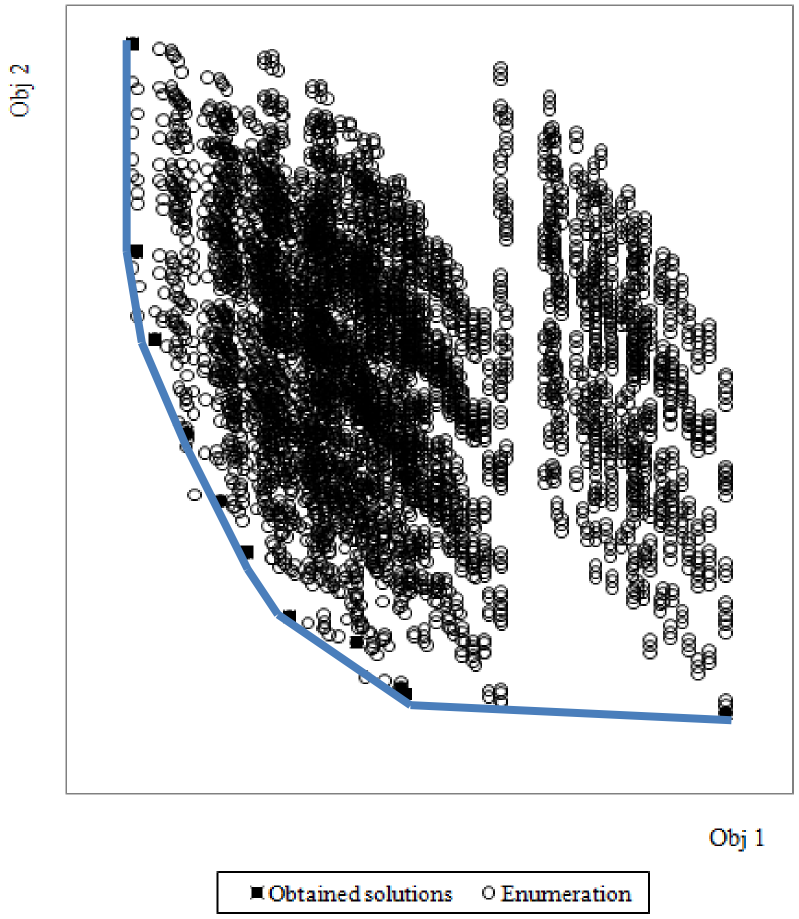
Converge to True Pareto Front



Seah et al. (2012). "Pareto Rank Learning in Multi-objective Evolutionary Algorithms." Proceedings of IEEE CEC

True Pareto front

- We may produce the true Pareto front by a complete enumeration and then compare the obtained results to it
- Enumeration, of course, may be impossible for large-scale problems



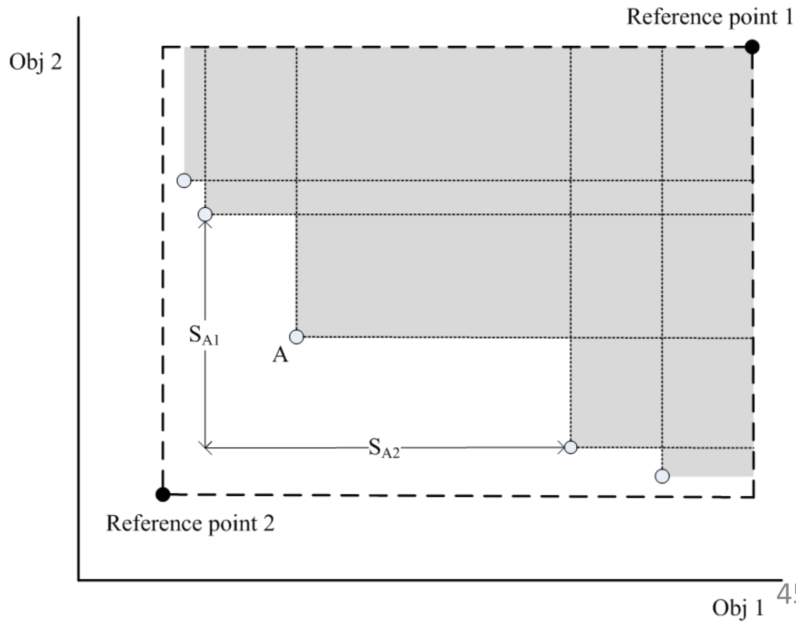
Possible Metrics (3)

- **Solution diversity (spread)**: we want the solutions to spread out on the front rather than gathering together in a small area
- To reflect the goal, one may measure

1. Number of non-dominated solutions, the greater the better
2. Distance between two extremes, the longer the better
3. Average distance between two adjacent non-dominated solutions , the smaller the better

Possible Metrics (4)

- **Hypervolume**: Area covered by the non-dominated solutions
- We may transform the hypervolume into a unit-less measurement, using reference points
- The greater the better



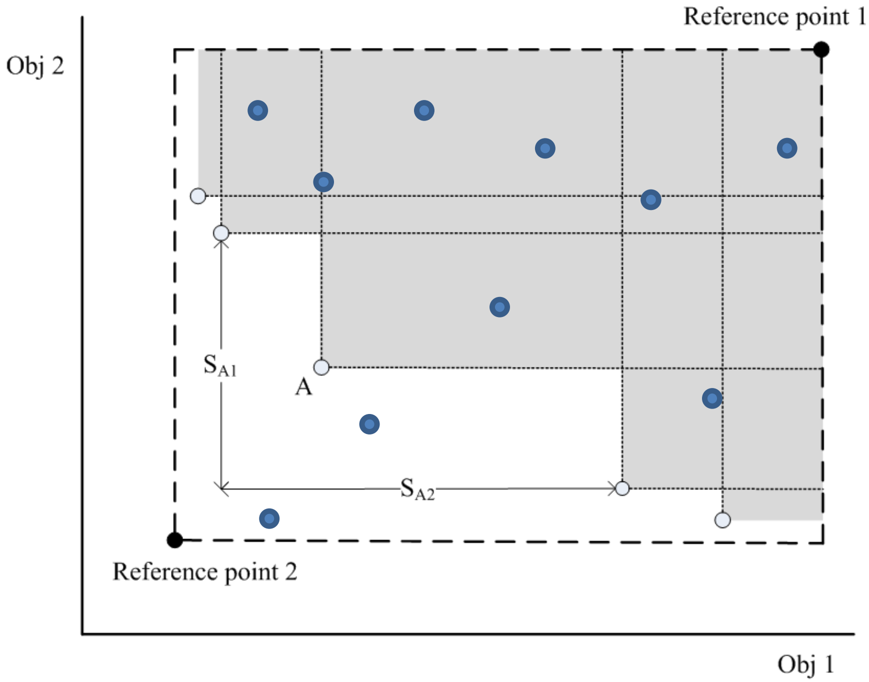
Hyper-volume

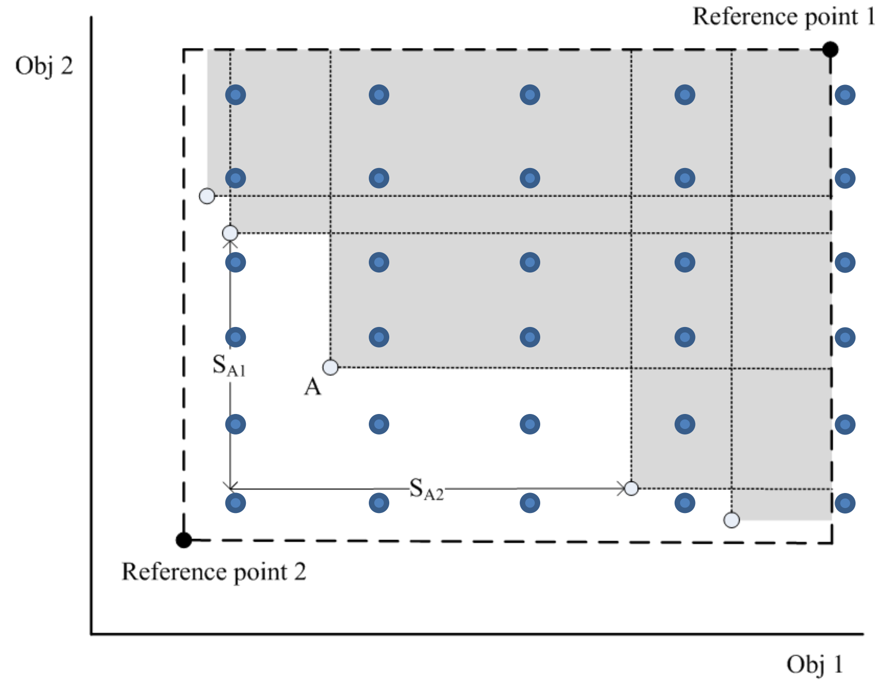
- **Hypervolume**: computational complexity increases exponentially with the number of objective functions
- Remedy: Using Monte Carlo simulation to approximate the true hypervolume
- Other metrics for multiobjective metaheuristics:

Okabe et al. (2003). “A Critical Survey of Performance Indices for Multi-Objective Optimization.” Proceedings of the 2003 Congress on Evolutionary Computation.

http://soft-computing.de/Final_CEC2003_Okabe_1.pdf

MCS to estimate Hypervolume





Final remarks

- A constraint may be transformed into an objective function
- Single objective optimization problem can therefore be upgraded to a multi-objective model
- A multiobjective optimization model is agile and flexible as non-dominated solutions provide decision makers with valuable tradeoff options to choose from
- It is more complicated to measure the performance of a multiobjective optimization algorithm