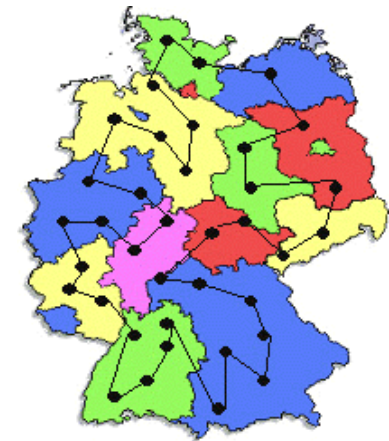


Where would we use Optimization?

- Engineering
- Architecture
- Nutrition
- Electrical circuits
- Economics
- Transportation
- Etc.

But also ...

- Optimization is also applied in:
 - Protein folding
 - System identification
 - Financial market forecasting (options pricing)
 - Logistics (traveling salesman problem), route planning, operations research
 - Controller design
 - Spacecraft trajectory planning



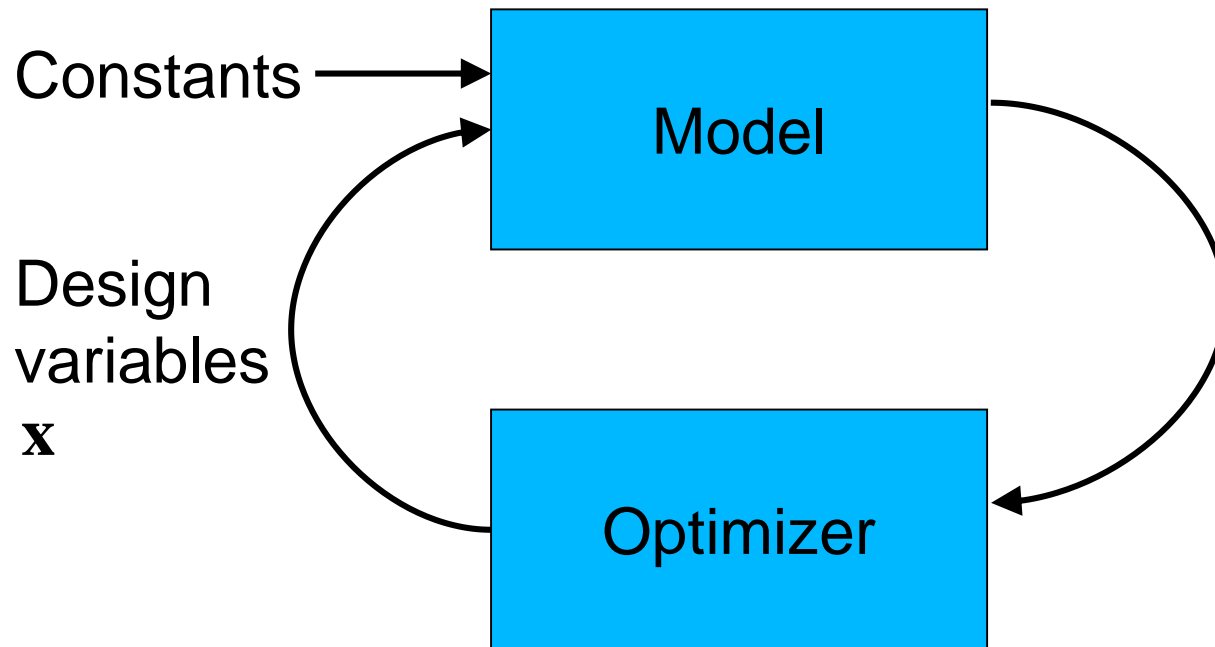
Optimization popularity

Increasingly popular:

- Increasing availability of numerical modeling techniques
- Increasing availability of cheap computer power
- Increased competition, global markets
- Better and more powerful optimization techniques
- Increasingly expensive production processes
(trial-and-error approach too expensive)
- More engineers having optimization knowledge

Solving optimization problems

- Optimization problems are typically solved using an iterative algorithm:



Curse of dimensionality

Looks complicated ... why not just sample the design space, and take the best one?

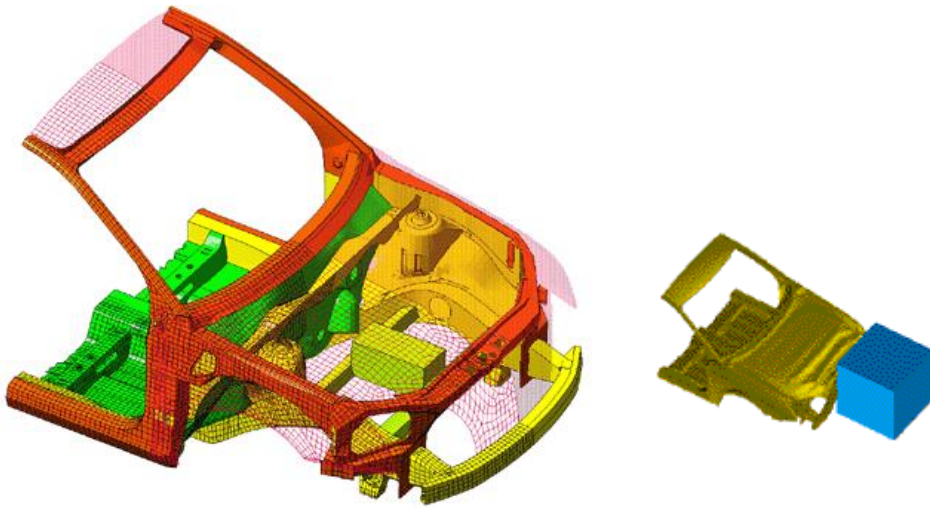
- Consider problem with n design variables
- Sample each variable with m samples
- Number of computations required: m^n

Take 1 second per computation,
10 variables, 10 samples:
total time *317 years!*



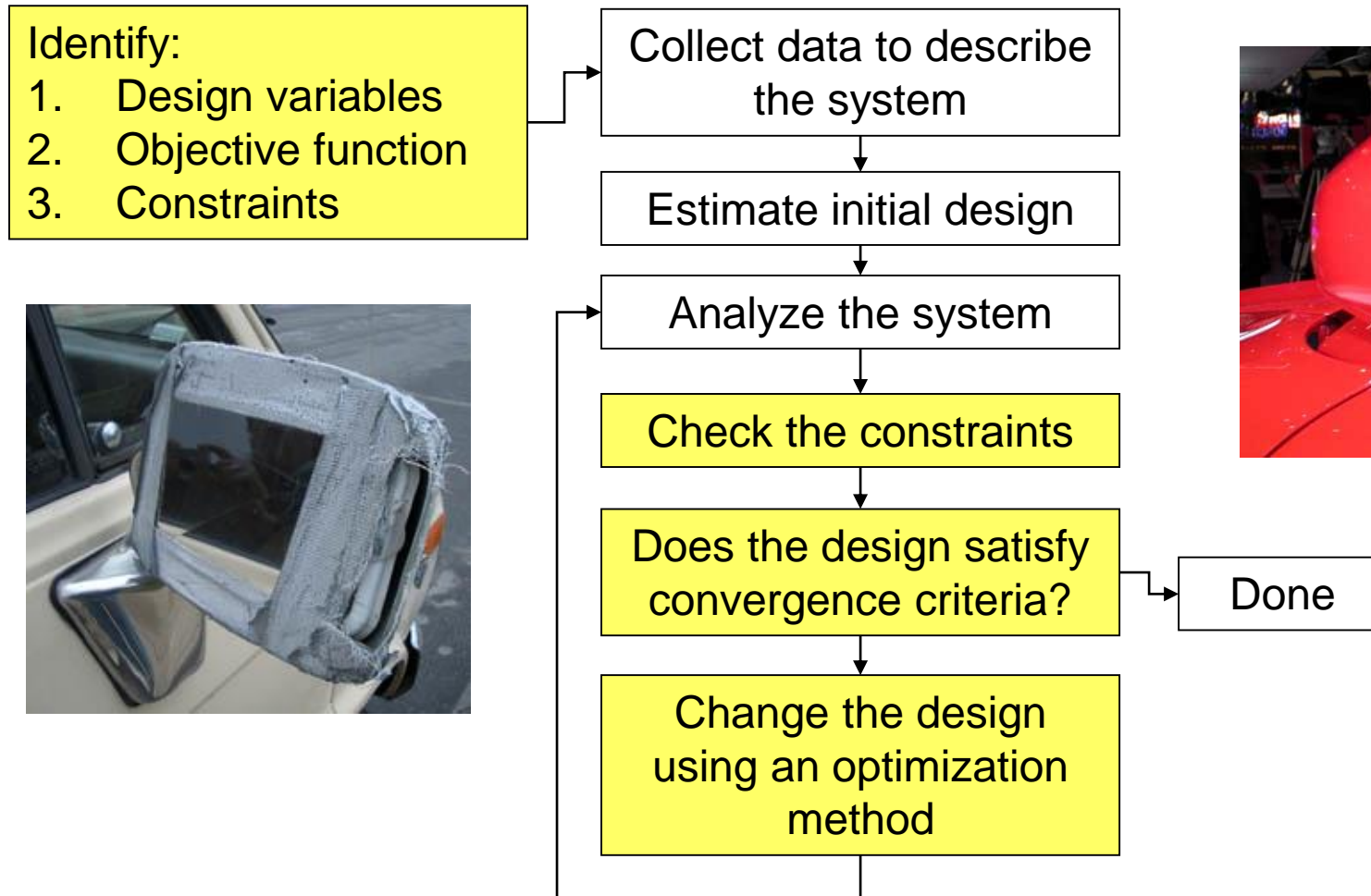
Parallel computing

- Still, for large problems, optimization requires lots of computing power
- Parallel computing



Optimization in the design process

Optimization-based design process:



What makes a design optimization problem interesting?

- Good design optimization problems often show a *conflict of interest / contradicting requirements*:
 - Aircraft wing & F1 car: stiffness vs. weight
 - Oil bottle: stiffness / buckling load vs. material usage
- Otherwise the problem could be trivial!

A Standard Optimization Model

Optimization concerns the minimization or maximization of functions:

▪ Standard Optimization Problem:

Objective function: $\min \text{ or } \max f(x)$

Subject to:

$$h_i(x) = 0 \quad i = 1, \dots, l \quad \text{Equality Constraints}$$

$$g_j(x) \leq 0 \quad j = 1, \dots, m \quad \text{Inequality Constraints}$$

$$x_k^L \leq x_k \leq x_k^U \quad k = 1, \dots, N \quad \text{Side Constraints}$$

where:

$f(x)$ is the *objective function*, which measure and evaluate the performance of a system. In a standard problem, we are *minimizing* the function. For maximization problems, it is equivalent to minimization of $-f(x)$.

x is a column vector of *design variables*, which can affect the performance of the system.

$$\max f(x) = \min [-f(x)]$$

Example of an optimization problem

Objective Function:

$$\min f(x) = -x_1 x_2 x_3$$

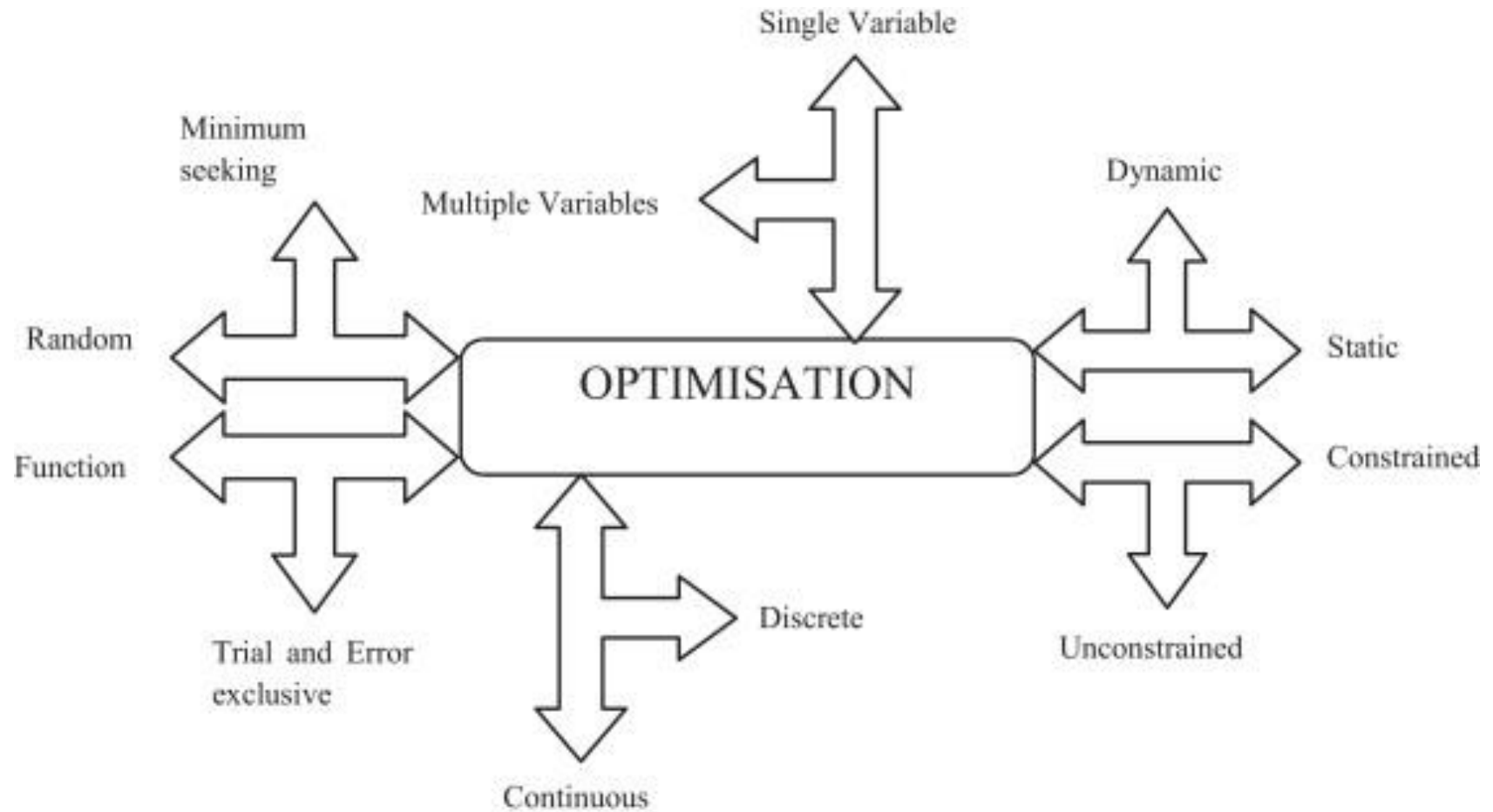
Subject to:

The constraints are grouped into categories as follows:

- $2x_1^2 + x_2 \leq 0$ points to **Nonlinear inequalities** (red box).
- $5x_1^2 + 3x_2^3 = 0$ points to **Nonlinear equalities** (purple box).
- $-x_1 - 2x_2 - 2x_3 \leq 0$ and $x_1 + 2x_2 + 2x_3 \leq 72$ are grouped by a bracket pointing to **Linear inequalities** (black box).
- $-5x_1 - 2x_2 = 0$ and $8x_2 + 4x_3 = 5$ are grouped by a bracket pointing to **Linear equalities** (blue box).
- $0 \leq x_1, x_2, x_3 \leq 30$ points to **Side Constraints** (green box).

$$\begin{aligned} 2x_1^2 + x_2 &\leq 0 \\ 5x_1^2 + 3x_2^3 &= 0 \\ -x_1 - 2x_2 - 2x_3 &\leq 0 \\ x_1 + 2x_2 + 2x_3 &\leq 72 \\ -5x_1 - 2x_2 &= 0 \\ 8x_2 + 4x_3 &= 5 \\ 0 \leq x_1, x_2, x_3 &\leq 30 \end{aligned}$$

Optimization Category



Optimization classification

Single Variable Vs. Multiple Variables

- If there is only one variable, the optimization is one-dimensional.

Dynamic Vs. Static

- Dynamic optimization means that the output is a function of time, while static means that the output is independent of time.

Discrete Vs. Continuous

- Discrete variables have only a finite number of possible values, whereas continuous variables have an infinite number of possible values.

Try-and-error Vs. Function

- Trial-and-error optimization refers to the process of adjusting variables that affect the output without knowing much about the process that produces the output.
- In contrast, a mathematical formula describes the objective function in function optimization.
- Experimentalists prefer the try-and-error, while theoreticians love the theoretical and mathematical approach.

Random Search Vs. Minimum seekers

- Some algorithms try to minimize the cost by starting from an initial set of variable values. These minimum seekers easily get stuck in local minima but tend to be fast. They are the traditional optimization algorithms and are generally based on calculus methods. Moving from one variable set to another is based on some determinant sequence of steps.
- On the other hand, random methods use some probabilistic calculations to find variable sets. They tend to be slower but have greater success at finding the global minimum.

Minimum Seeking Algorithms: Classic way

Random Search Algorithms: New way

Constraint Vs. Unconstraint

- Variables often have limits or constraints. Constrained optimization incorporates variable equalities and inequalities into the cost function.
- Constraints can be *hard* (must be satisfied) or *soft* (is desirable to satisfy).

Example: In your course schedule a hard constraint is that no classes overlap. A soft constraint is that no class be before 10 AM.

- Constraints can be *explicit* (stated in the problem) or *implicit* (obvious to the problem).

Single objective vs multi-objective

- Minimize $f(\mathbf{x})$ \longleftarrow Vector!

s.t.: $g(\mathbf{x}) \leq 0$, $h(\mathbf{x}) = 0$

- Input from designer required! Popular approach:
replace by weighted sum:

$$F(\mathbf{x}) = \sum_i w_i f_i(\mathbf{x})$$

- Optimum, clearly, depends on choice of weights
- *Pareto optimal point*: “no other feasible point exists that has a smaller f_i without having a larger f_j ”

Multi-objective problems (cont.)

- Examples of multi-objective problems:
 - Design of a structure for
 - Minimal weight *and*
 - Minimal stresses
 - Design of reduction gear unit for
 - Minimal volume
 - Maximal fatigue life
 - Design of a truck for
 - Minimal fuel consumption @ 80 km/h
 - Minimal acceleration time for 0 – 40 km/h
 - Minimal acceleration time for 40 – 90 km/h



Types of Solutions

- A solution to an optimization problem specifies the values of the decision variables, and therefore also the value of the objective function.
- A feasible solution satisfies all constraints.
- An optimal solution is feasible and provides the best objective function value.
- A near-optimal solution is feasible and provides a superior objective function value, but not necessarily the best.
- Near-optimal solution ~ Local optimal solution

Simple Vs. Hard problem

Hard

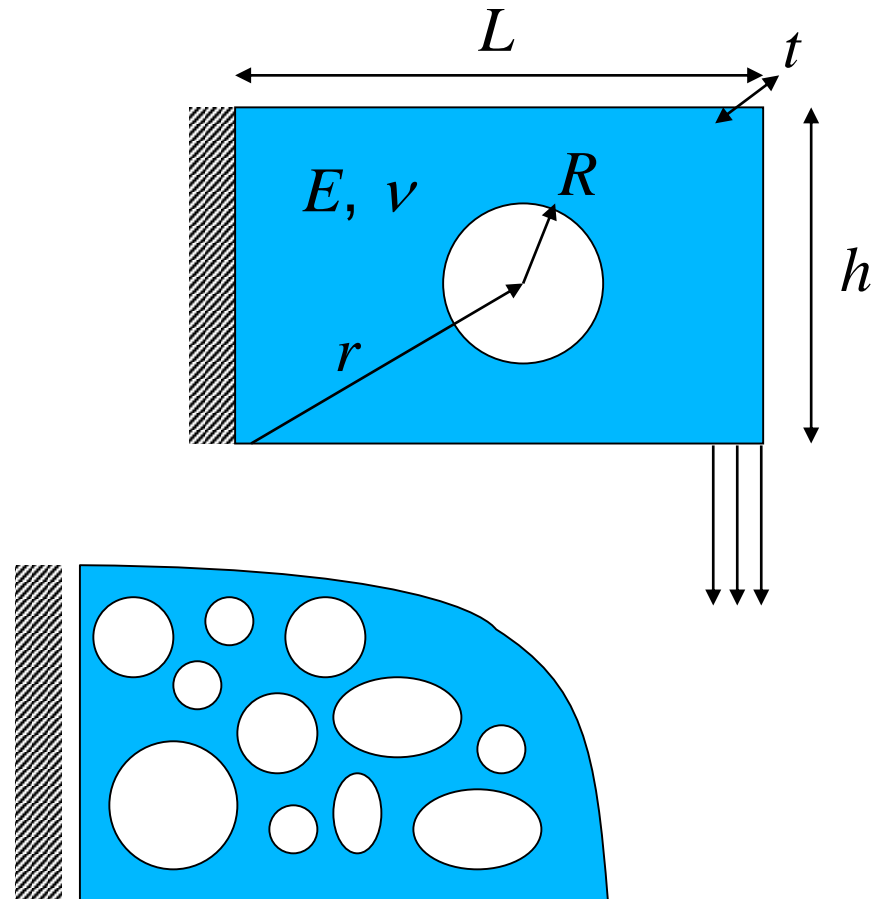
- Many decision variables
- Discontinuous, combinatorial
- Multi modal
- Objective difficult to calculate
- Severely constraints
- Feasibility difficult to determine
- Multiple objectives
- Stochastic

Simple

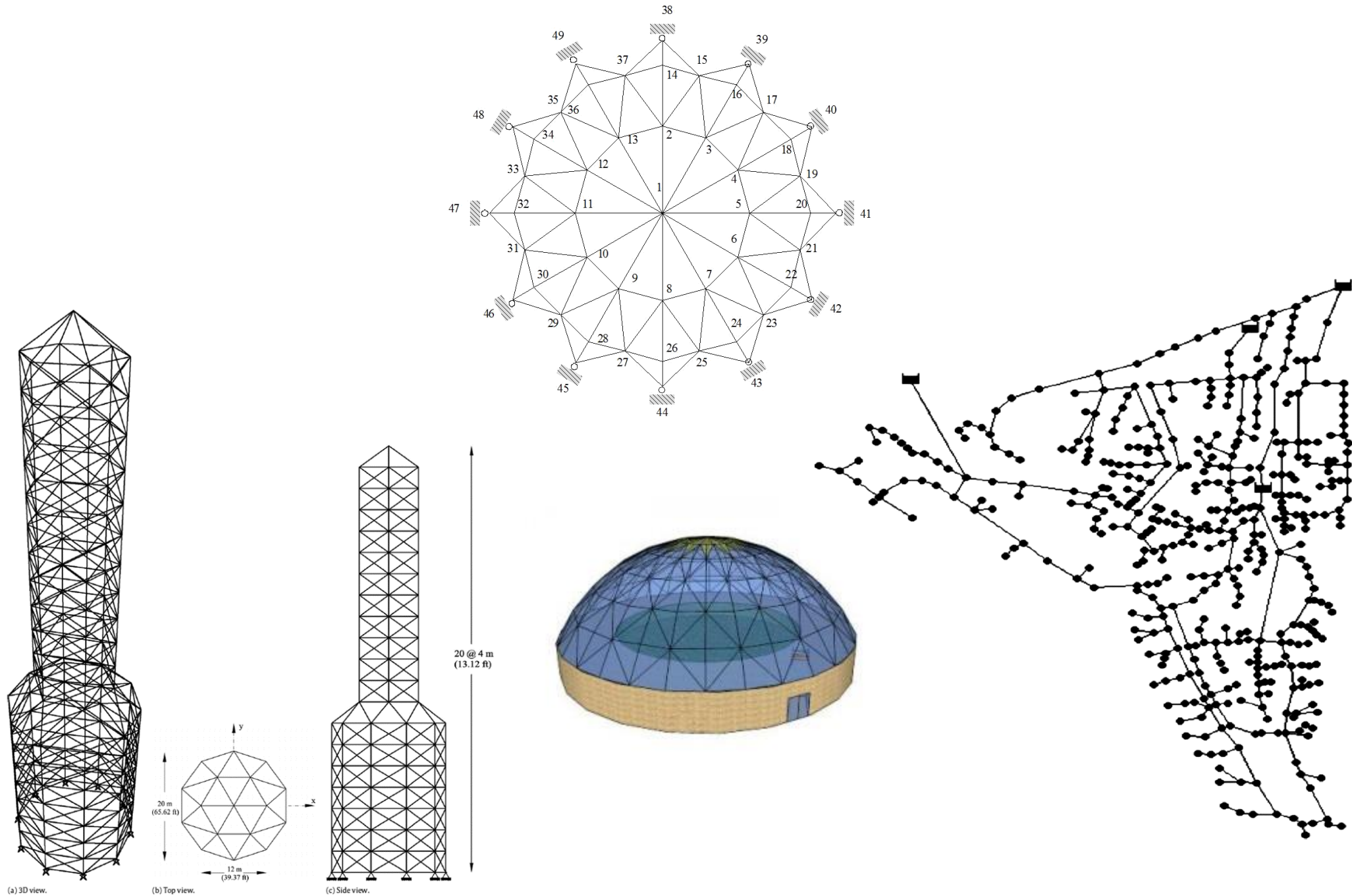
- Few decision variables
- Differentiable
- Uni-modal
- Objective easy to calculate
- No or light constraints
- Feasibility easy to determine
- Single objective
- deterministic

Structural optimization

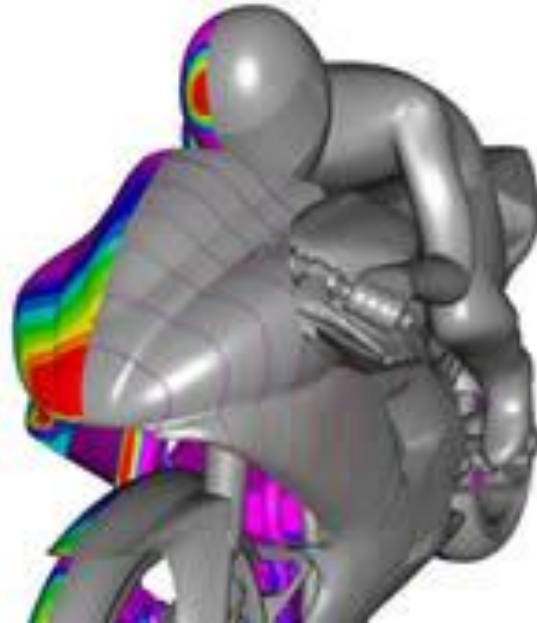
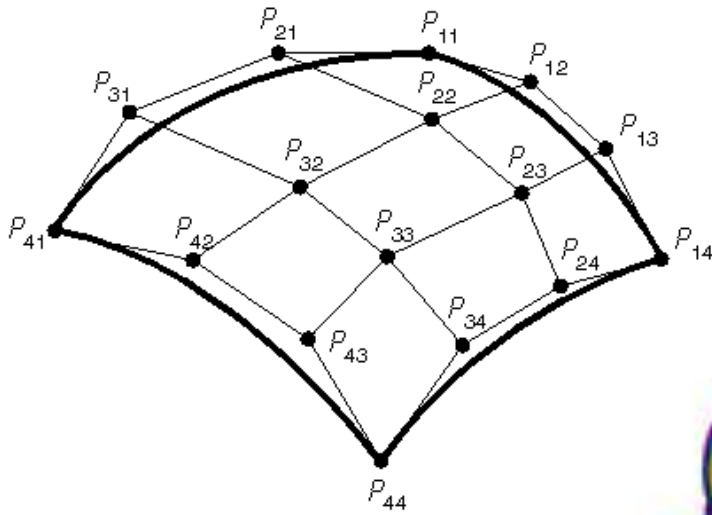
- Structural optimization = optimization techniques applied to structures
- Different categories:
 - Sizing optimization
 - Shape optimization
 - Topology optimization



Sizing Optimization

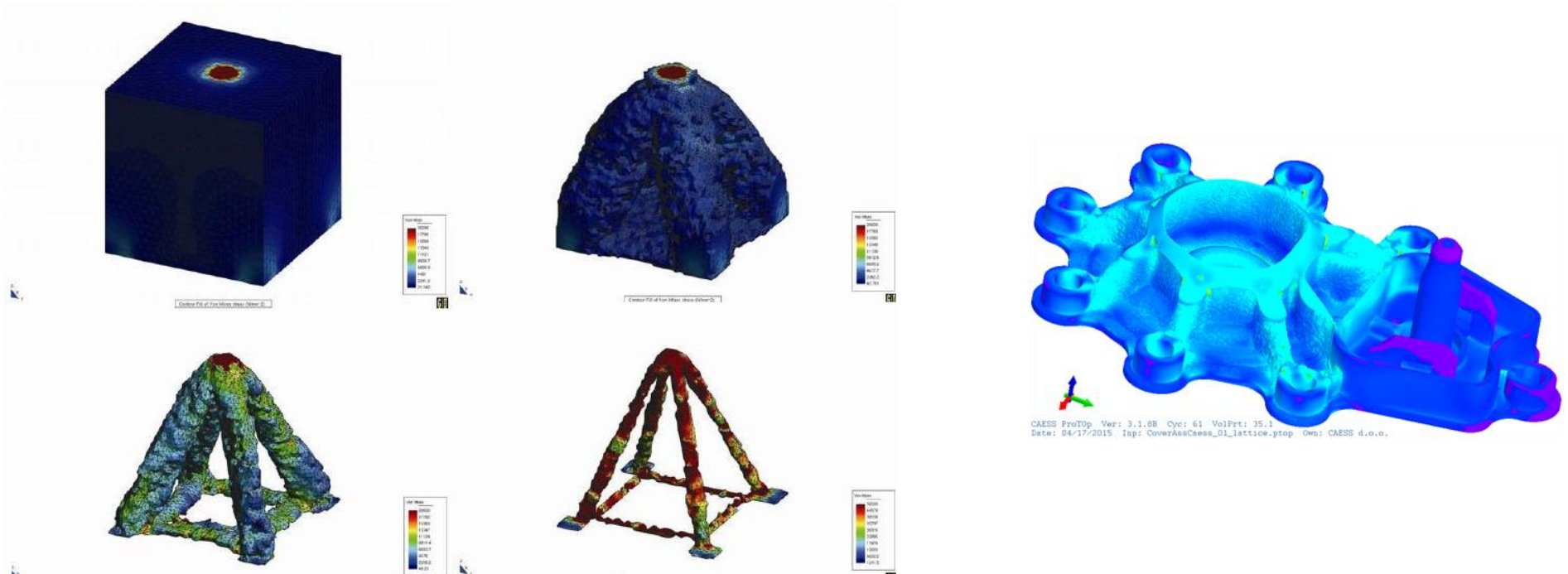


Shape optimization

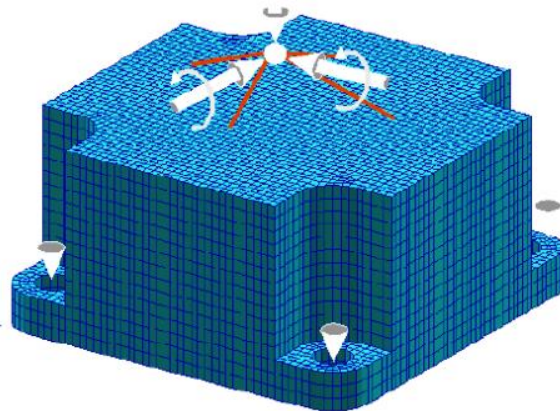


Yamaha R1

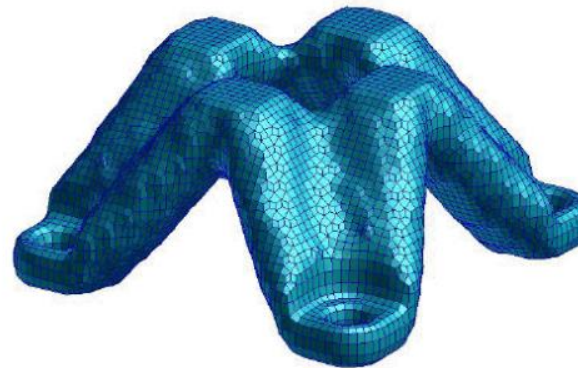
Topology optimization



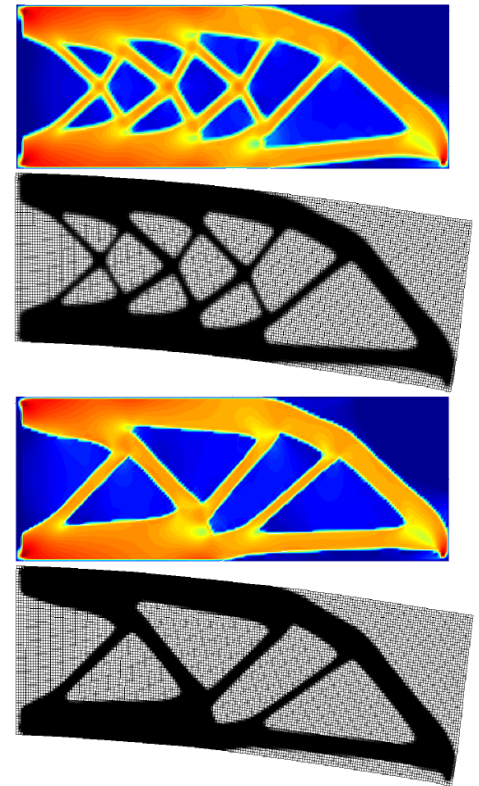
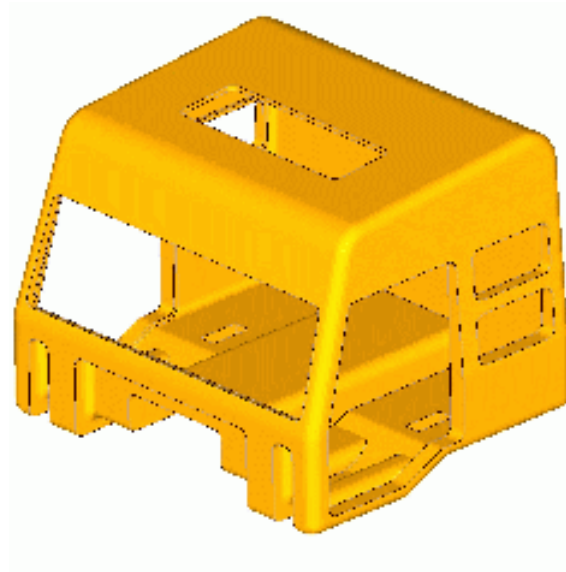
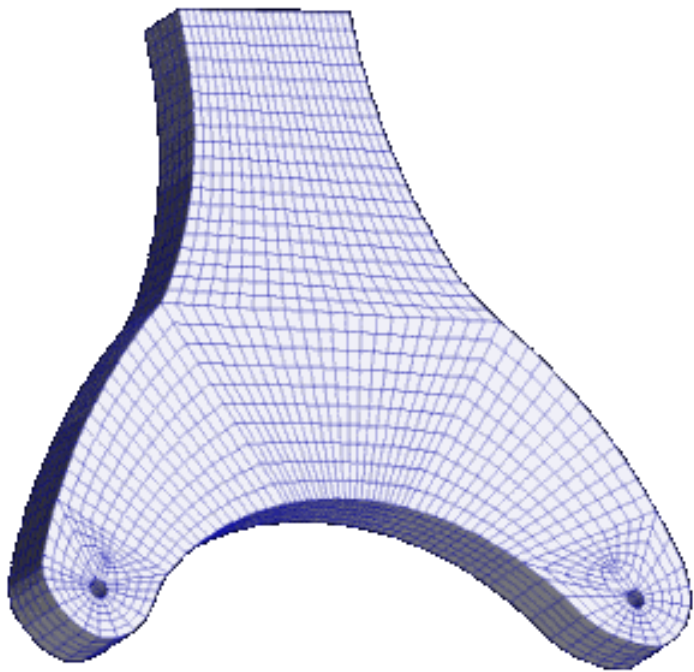
Before...



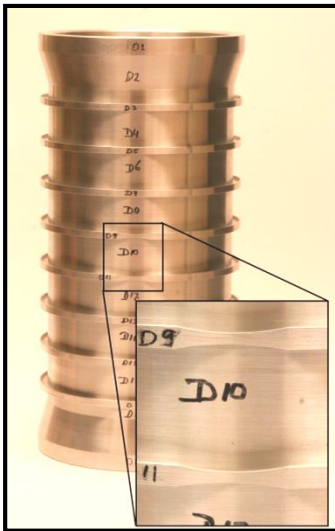
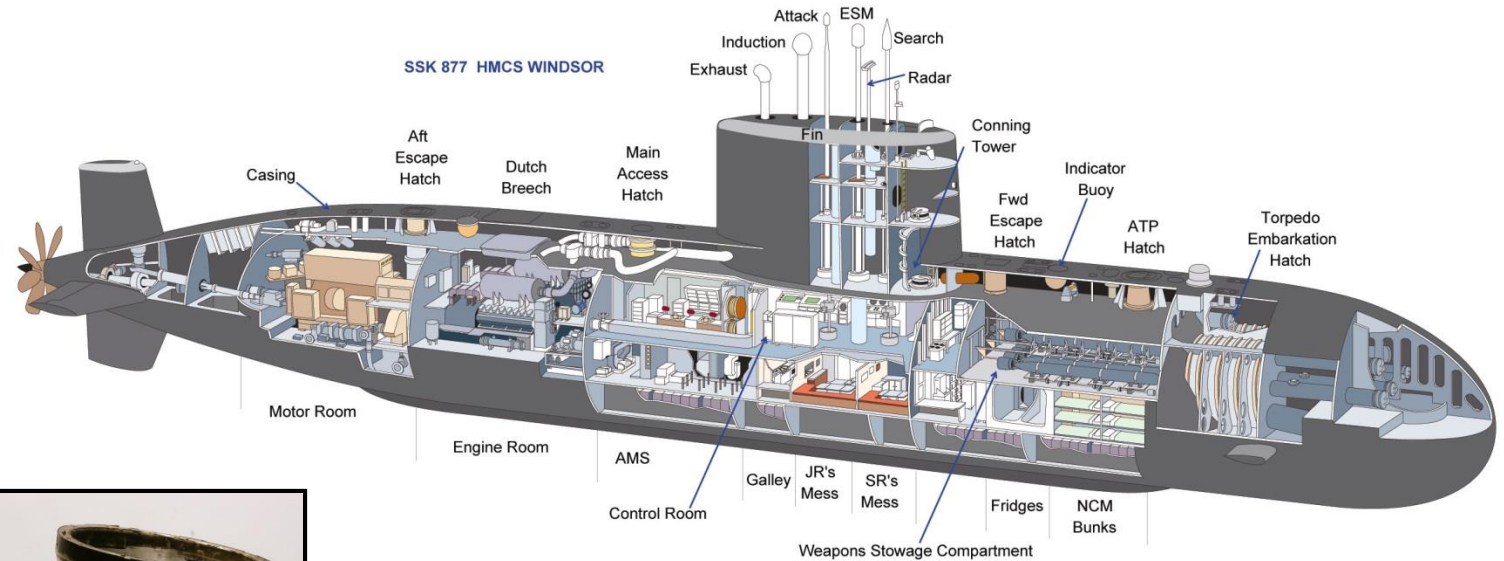
After!



Topology optimization



Practical Examples: Submarines



Practical Example: Airbus A380

- Wing stiffening ribs of Airbus A380:

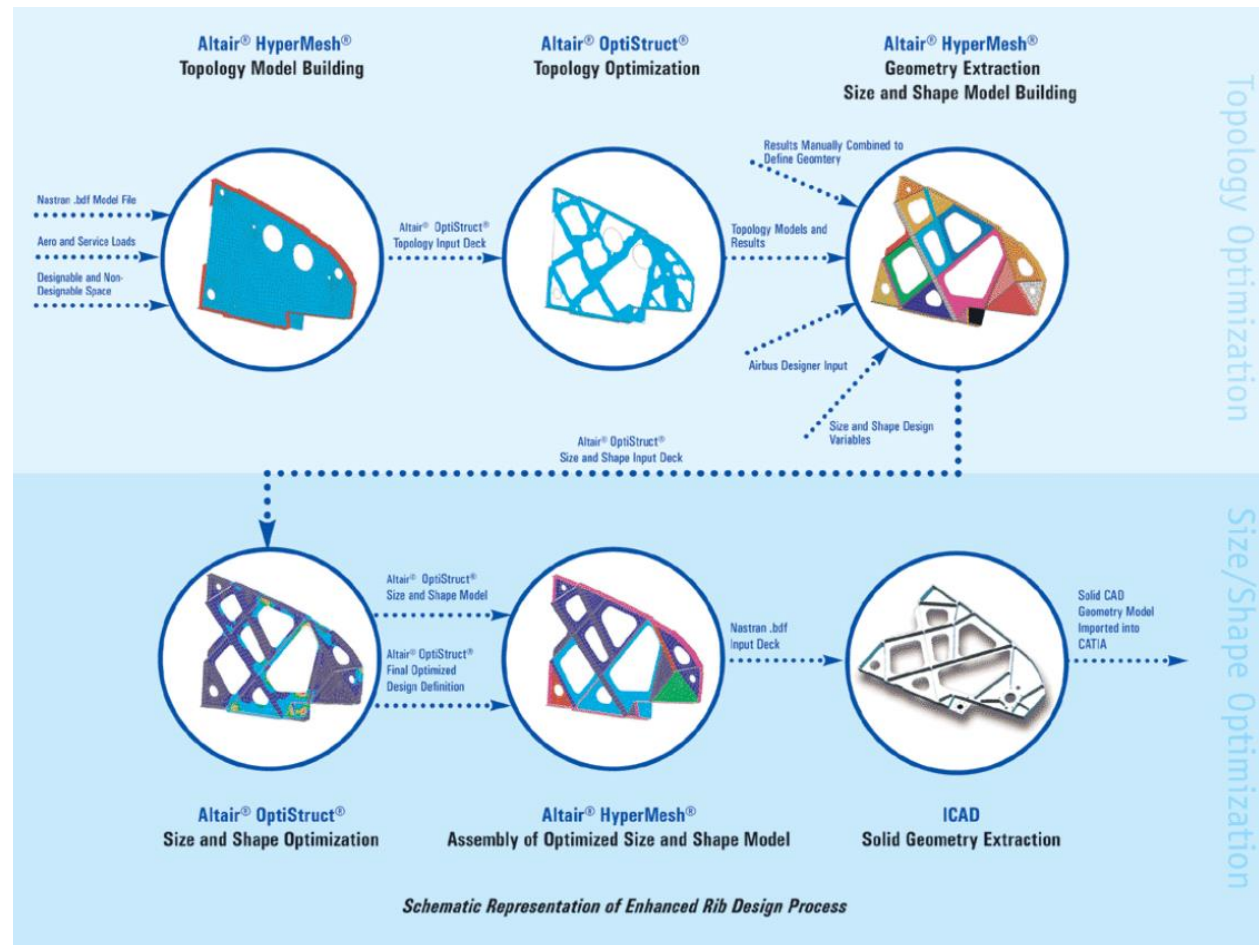


- Objective: reduce weight
- Constraints: stress, buckling

Leading
edge ribs

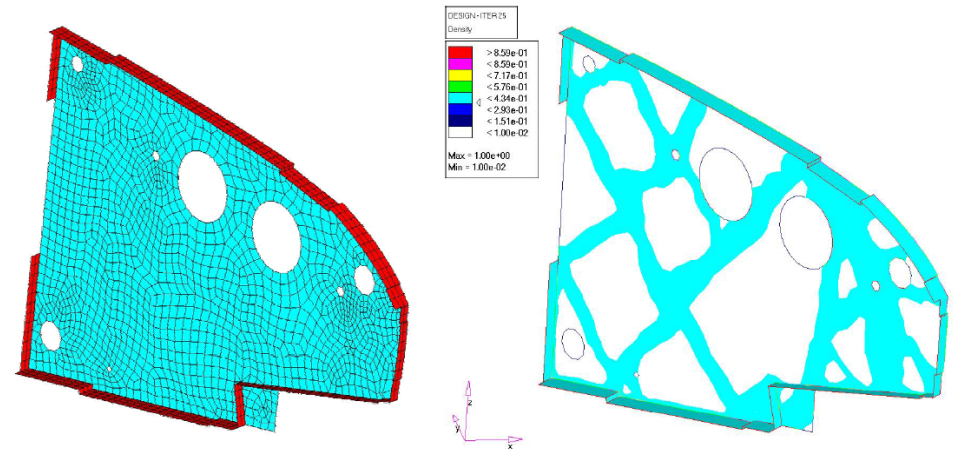
Airbus A380 example (cont.)

- Topology and shape optimization

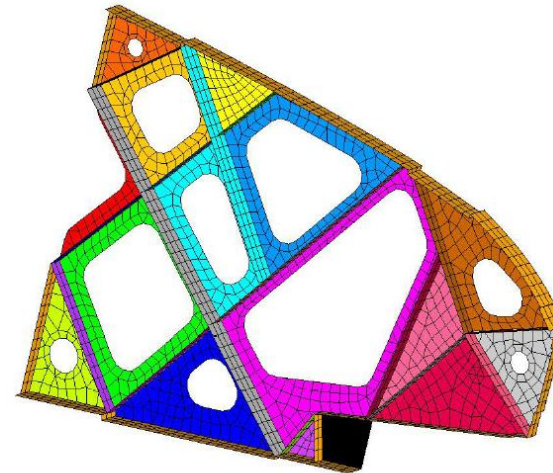


Airbus A380 example (cont.)

- Topology optimization:

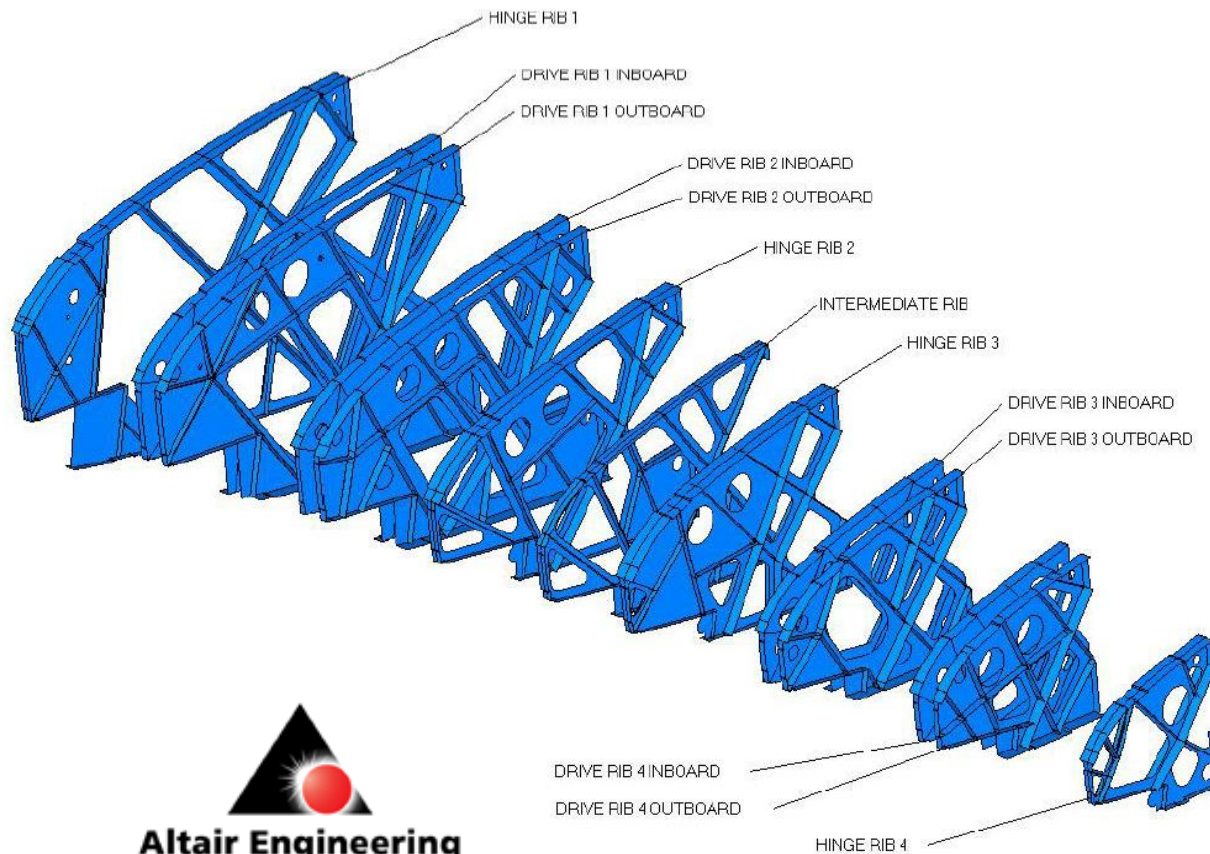


- Sizing / shape optimization:



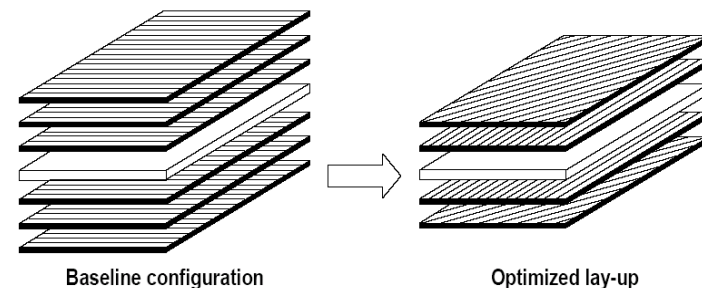
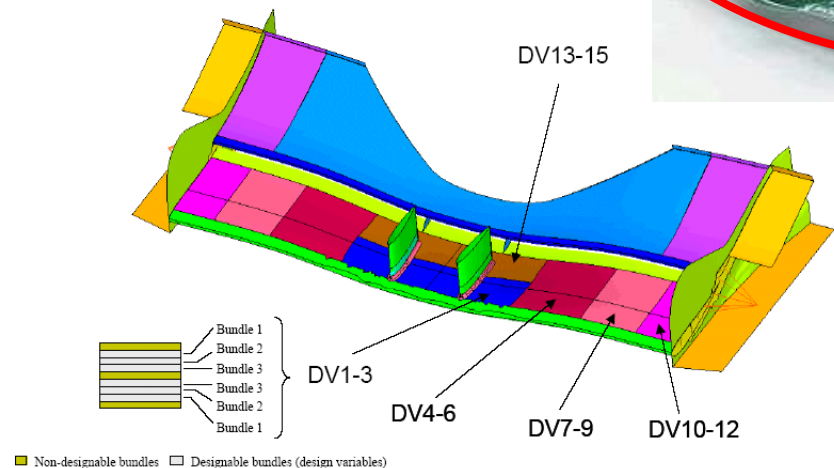
Airbus A380 example (cont.)

- Result: 500 kg weight savings!



Other examples

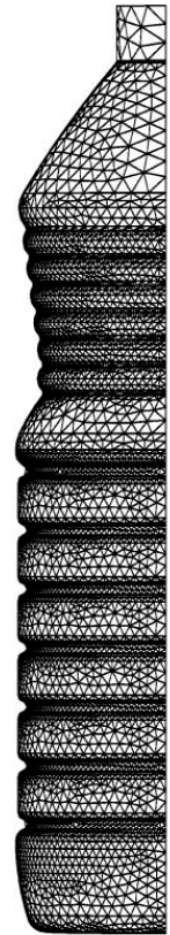
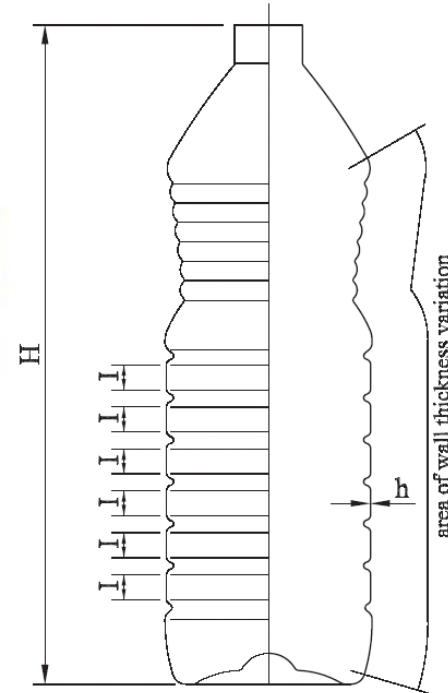
- Jaguar F1 FRC front wing:
reduce weight
constraints on
max. displacements



5% weight saved

Other examples (cont.)

- Design optimization of packaging products
- Objective: minimize material used
- Constraints: stress, buckling
- Result: 20% saved



Structural optimization examples

- Typical objective function: weight

$$f = \frac{W(\mathbf{x})}{W(\mathbf{x}_0)} \quad \text{Note the scaling!}$$

- Typical constraint: maximum stress, maximum displacement

$$g = \frac{\sigma_{\max}(\mathbf{x})}{\sigma_{\text{allowed}}} - 1 \leq 0$$

Scaled

vs.

$$g = \sigma_{\max}(\mathbf{x}) - \sigma_{\text{allowed}} \leq 0$$

Unscaled

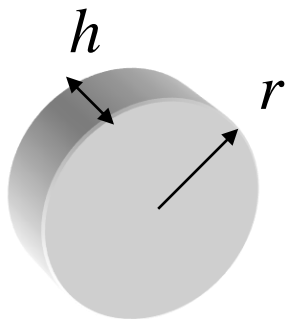


Aspirin pill design

- Proper bounds are necessary to avoid unrealistic solutions:

- Example: aspirin pill design

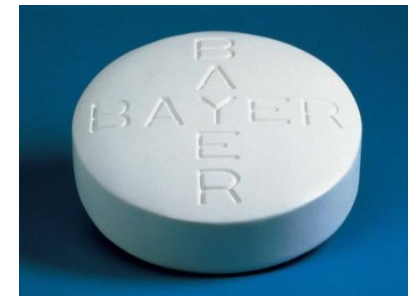
Objective: minimize dissolving time
= maximize surface area
(fixed volume)



\Rightarrow

$$\max_{r,h} 2\pi r^2 + 2\pi rh$$

$$\text{s.t. } \pi r^2 h = 1$$

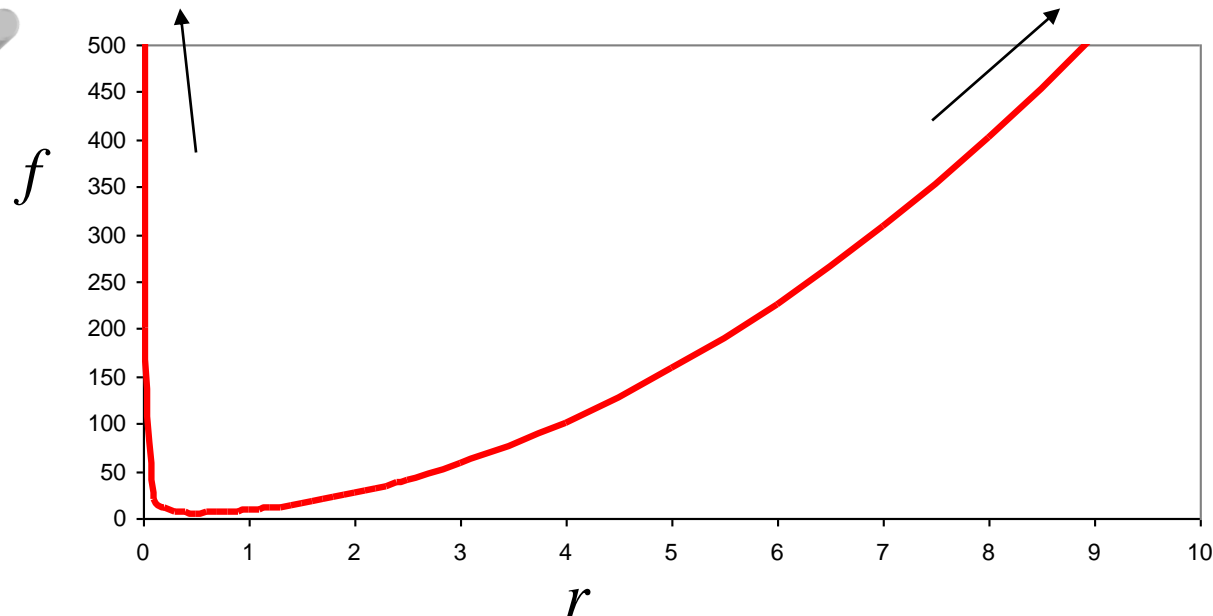
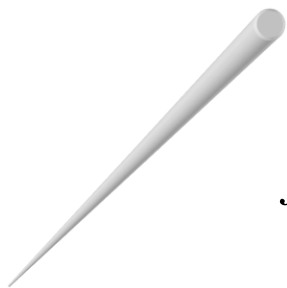


Aspirin pill design (cont.)



- Volume equality constraint can be substituted, yielding:

$$h = \frac{1}{\pi r^2} \quad \Rightarrow \quad \max_r \quad 2\pi r^2 + \frac{2}{r}$$



Using Optimization in Mathematics

$$y' + 2y + 5 \int_0^x y(t)dt = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}, y(0) = 0 \quad (36)$$

where the solution interval varies from 0 to π . Using the mathematical approach, the exact solution of the considered IVP is obtained as follows:

$$y(x) = \frac{1}{2}e^{-x} \sin(2x).$$

This ODE problem is solved using different approximate solution is represented in terms of Fourier series (NT) is chosen variables). The best approximate solution square weight (LSW) function using the follows:

$$\begin{aligned} Y_{\text{appx}}(x) = & (1.9996E-02) + (1.1002E-02) \\ & + (3.8363E-02)\sin(x) - (7.99 \\ & + (1.1888E-01)\sin(2x) - (3.6 \\ & + (4.7980E-02)\sin(3x) - (4.2 \\ & + (6.5364E-02)\sin(4x) + (2.7 \\ & + (4.9891E-02)\sin(5x) + (2.0154E-02)\cos(6x) \\ & - (4.1469E-03)\sin(6x) \end{aligned} \quad (38)$$

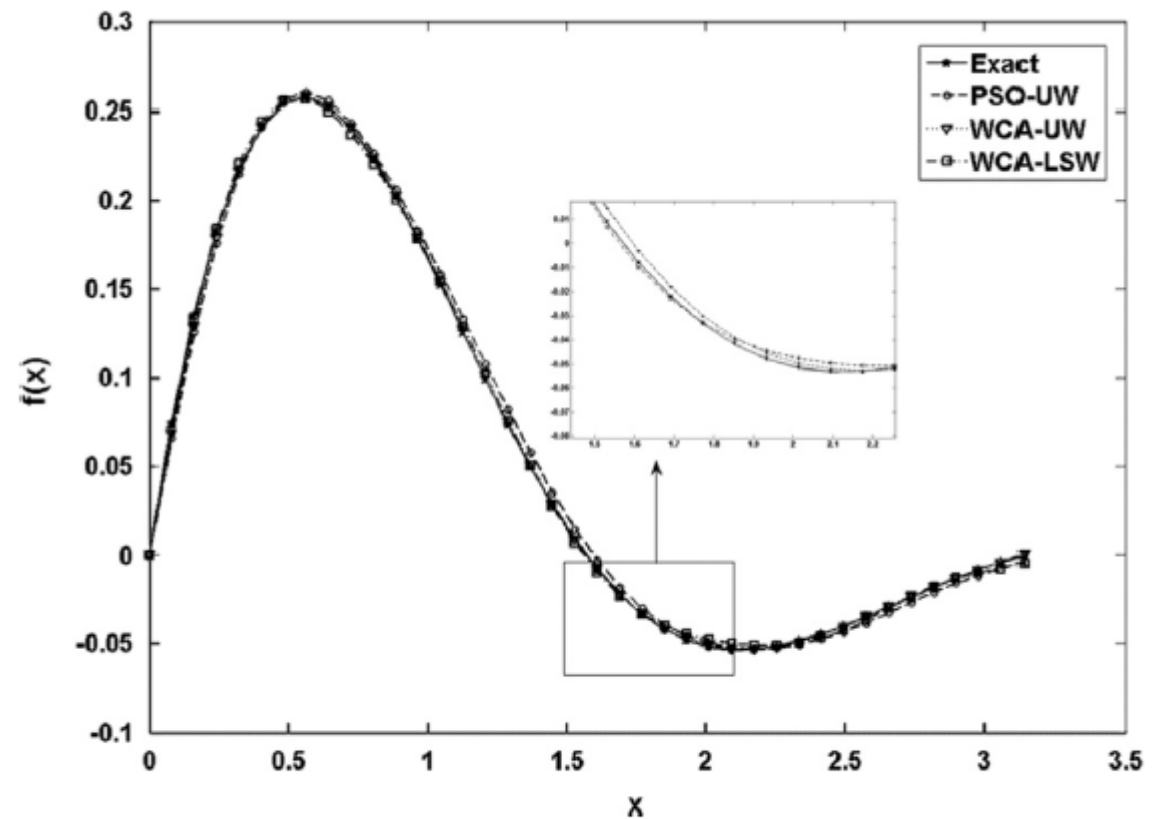
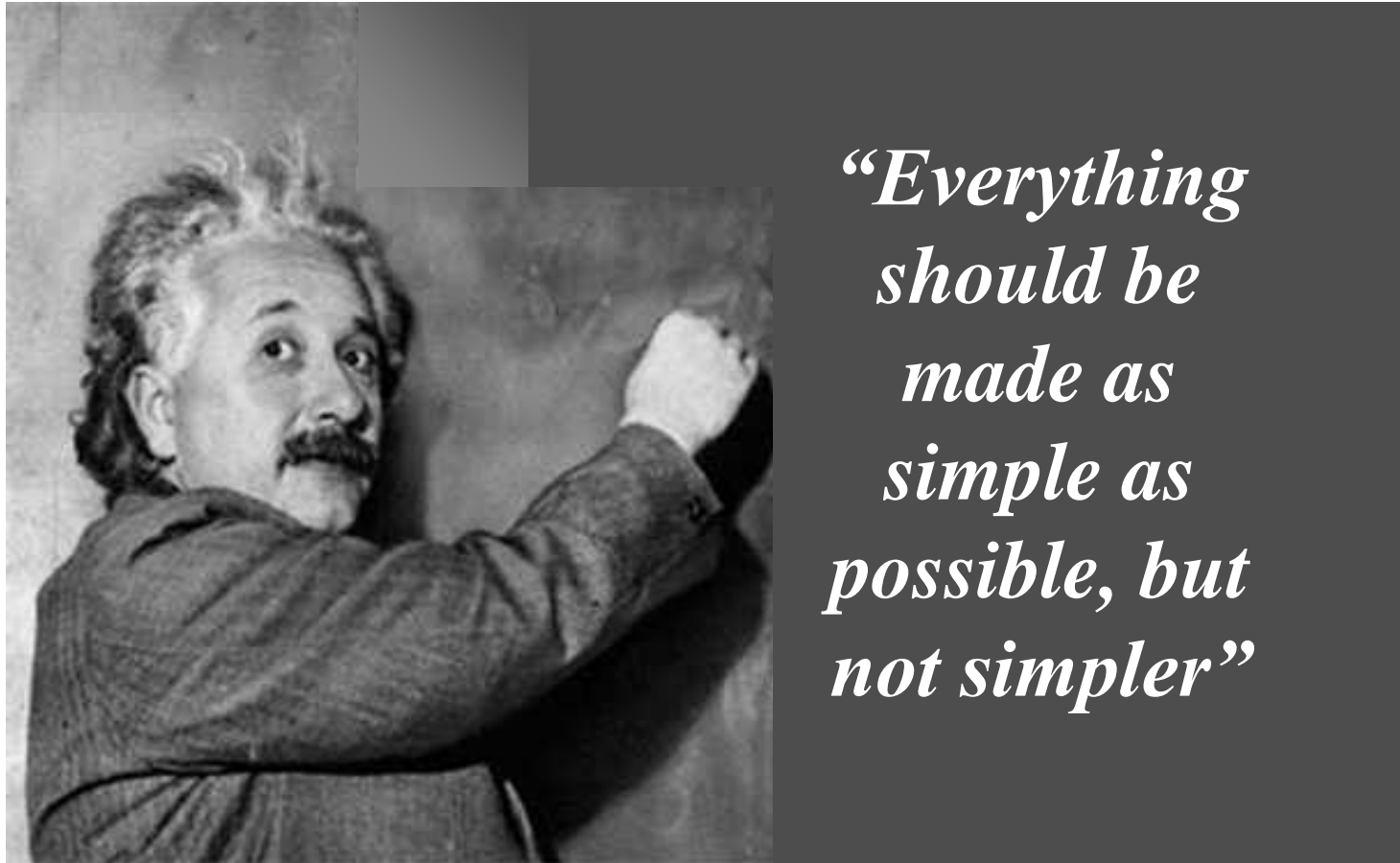


Fig. 3. Comparison of the best solutions between PSO and WCA optimizers having two different weight functions for the test problem 1.

Einstein's advice

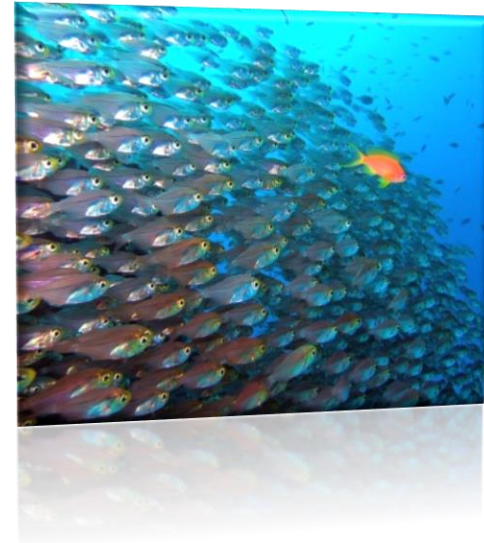
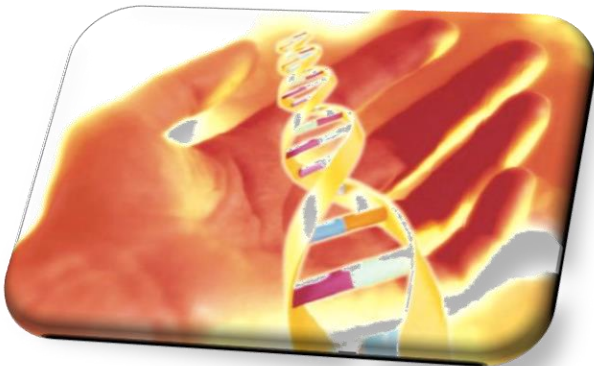


- Model simplification is important for optimization!

What are the metaheuristics?

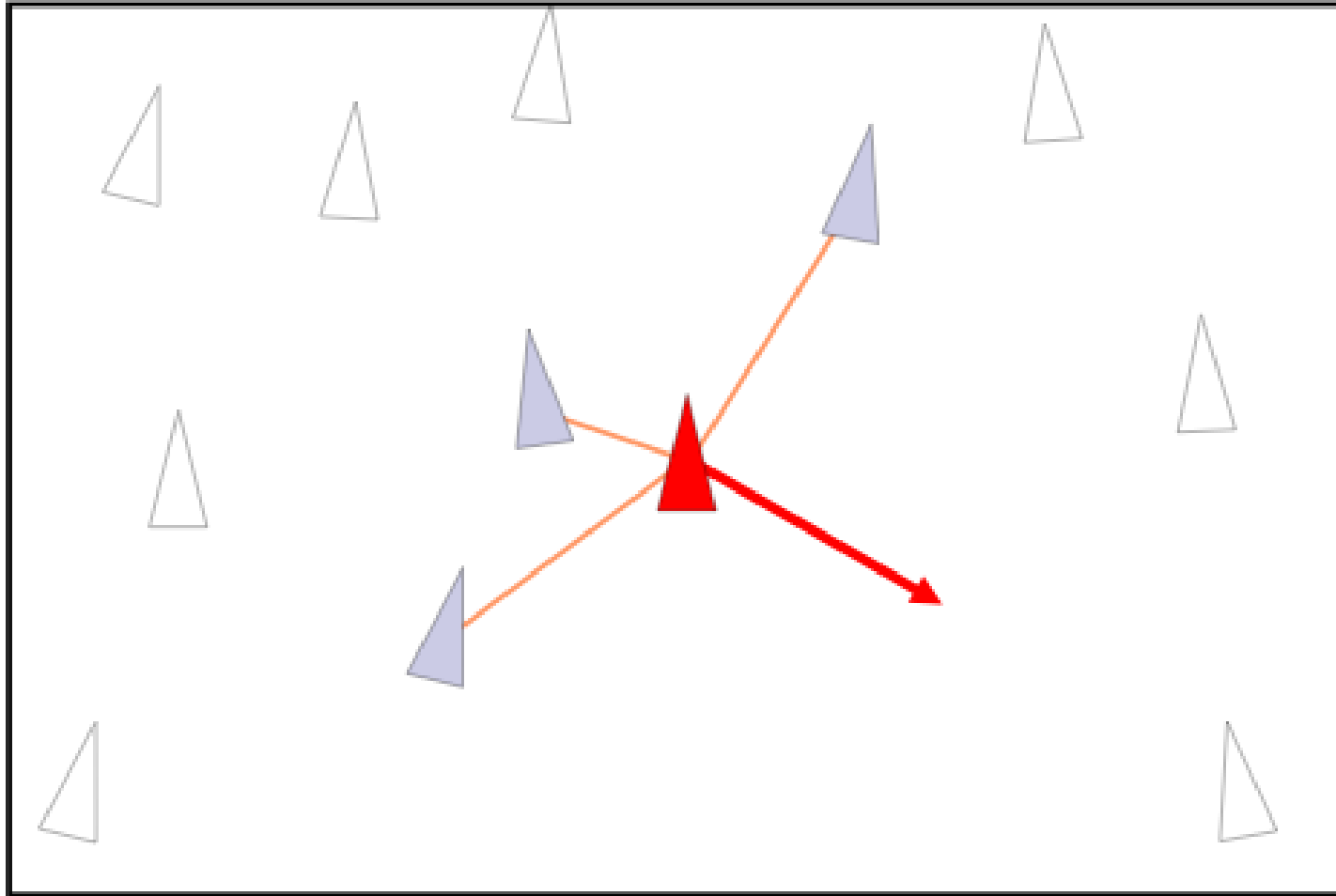
Metaheuristics

- Computational methods
- Iteratively improve
- Inspired by nature, real life events, etc
- No assumption of problem being solved
- Derivative free method



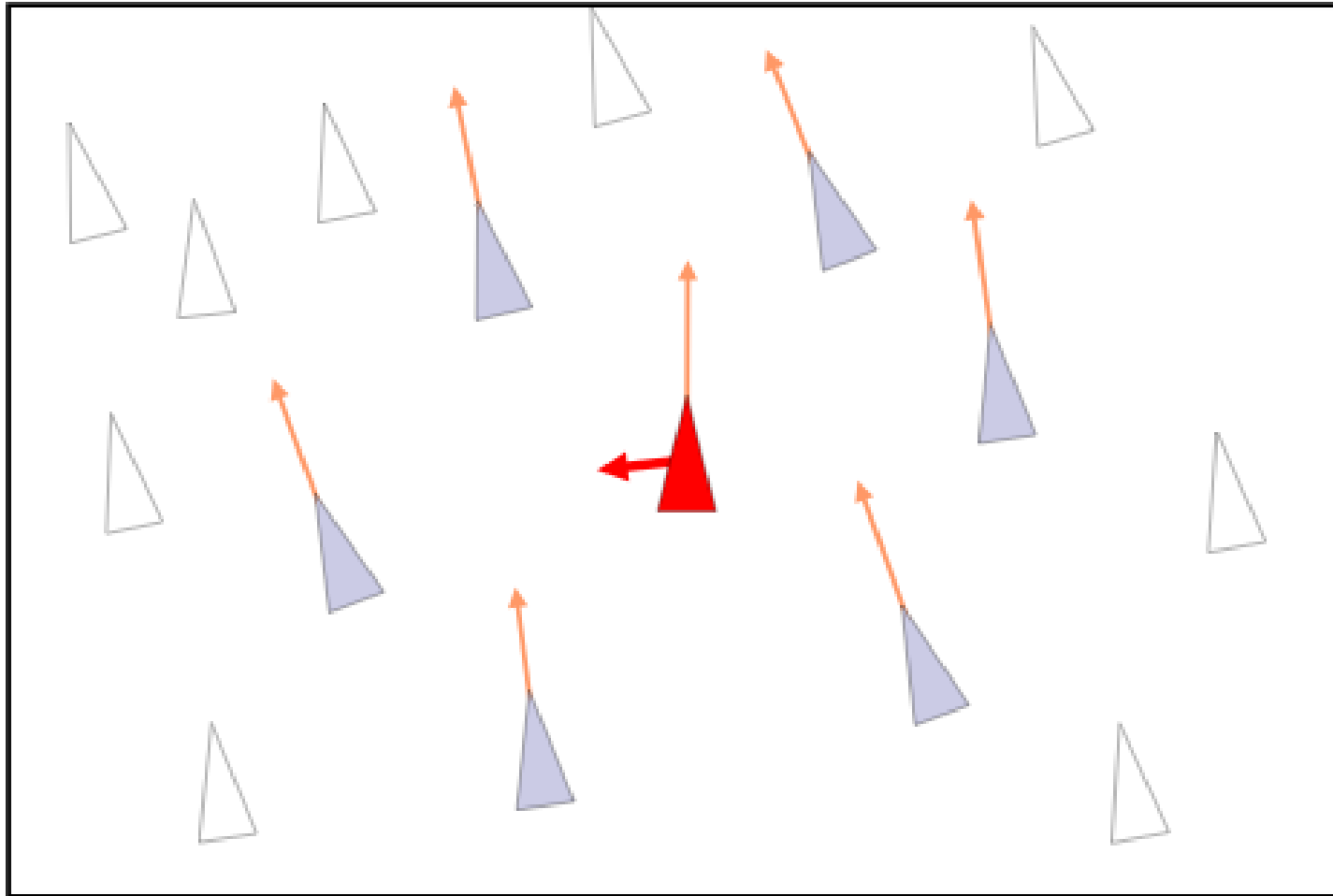
Rule 1: Collision Avoidance

Avoid Collision with neighboring birds



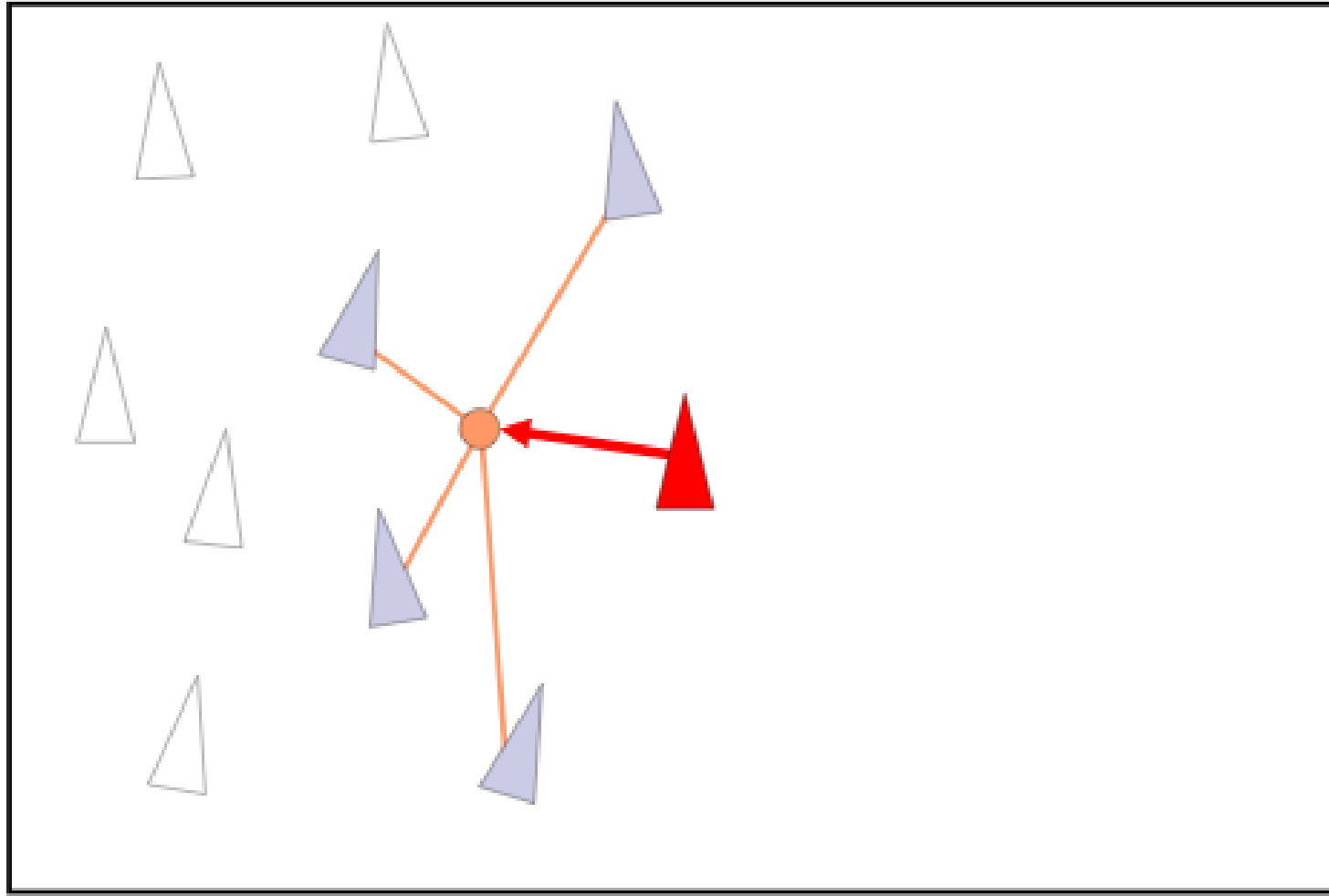
Rule 2: Velocity Matching

Match the velocity of neighboring birds



Rule 3: Flock (Swarm) Centering

Stay near neighboring birds



Ideal Optimizer

An Ideal Optimization Method

1. Guarantee finding global optimum point
2. No need any initial or user parameters
3. Fast convergence
4. Simple concept (simple programming)
5. High solution stability
6. Great solution quality
7. Independent to the nature of a given problem
8. Independent to the number of D.Vs

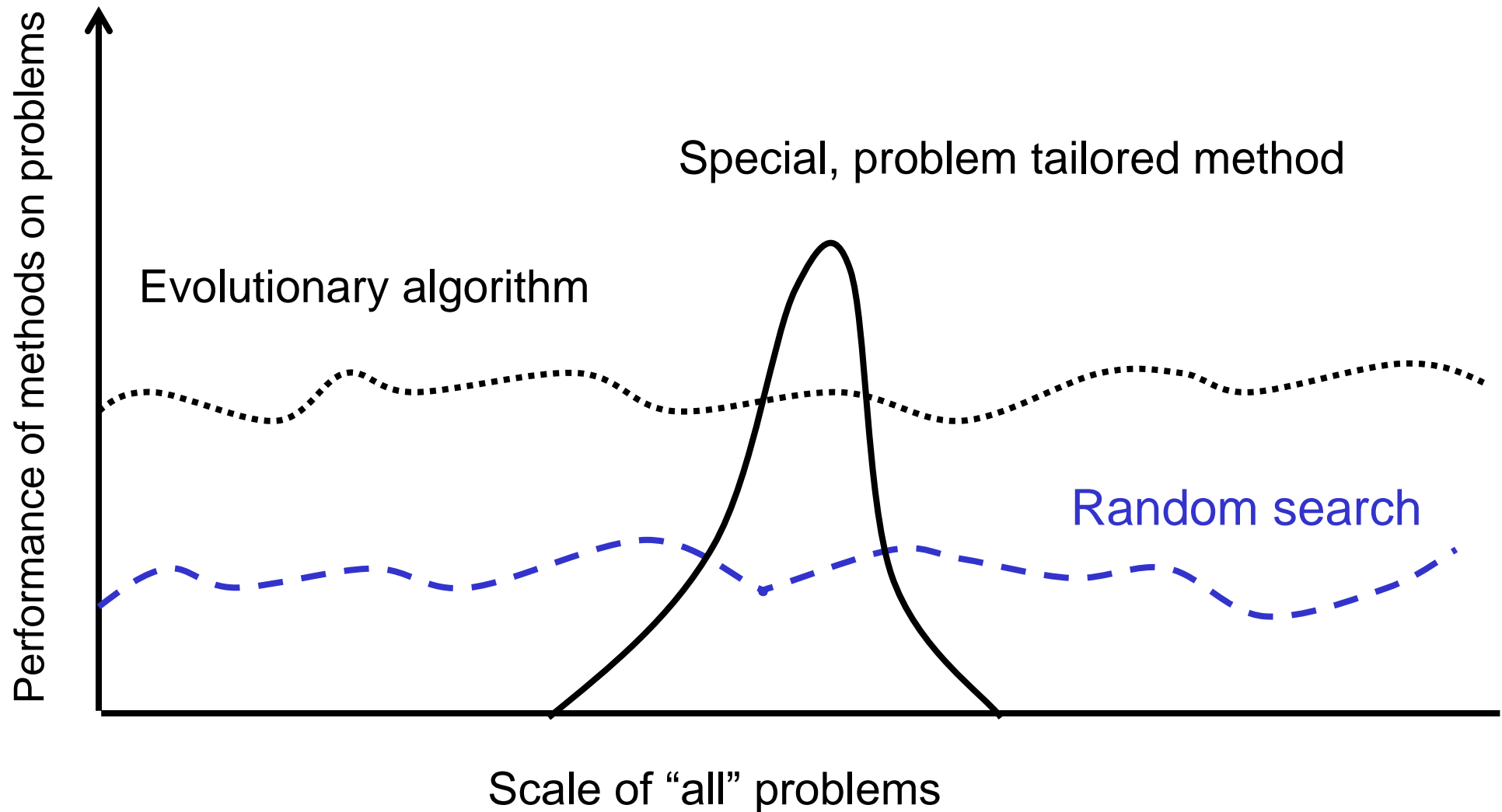


Simultaneously

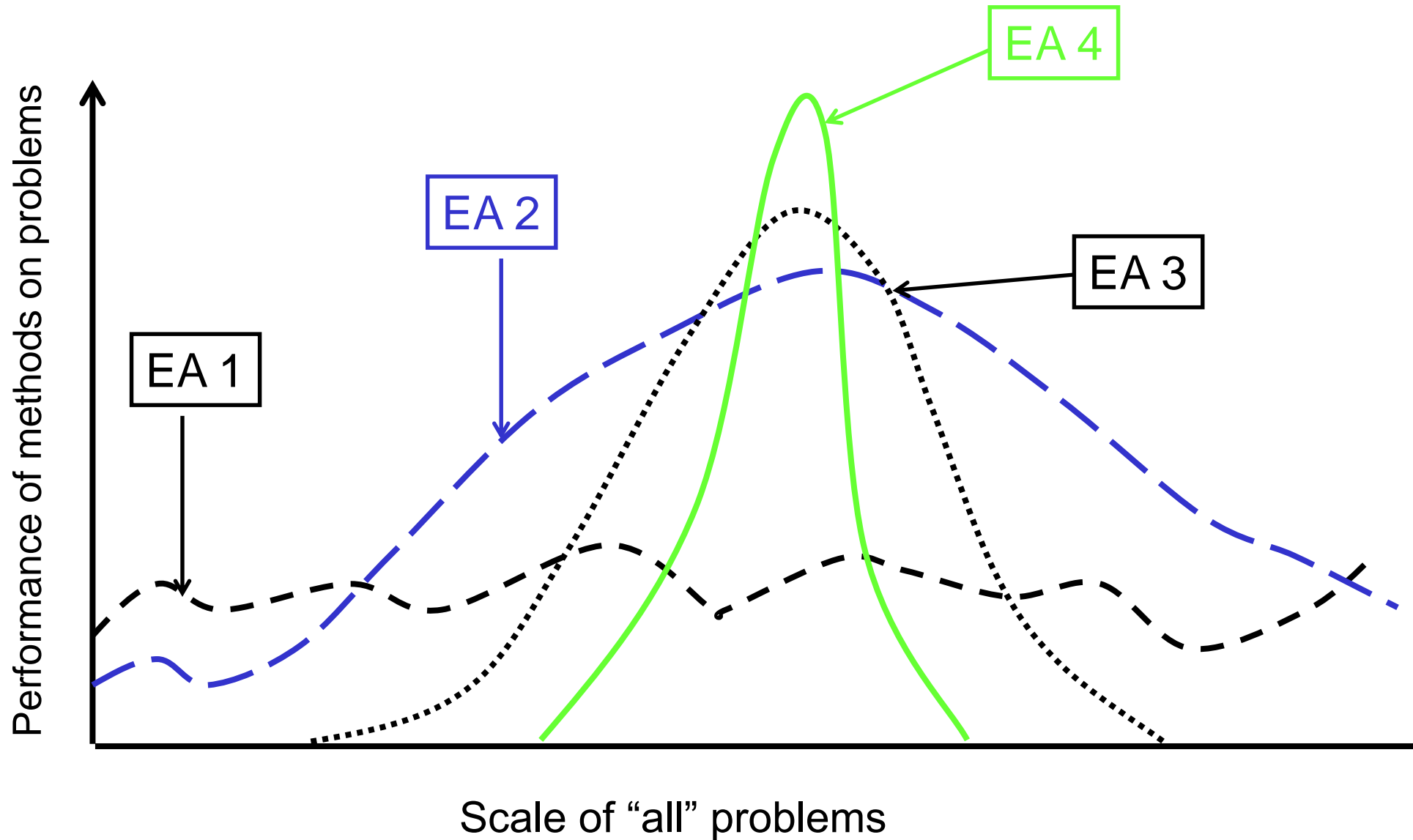
When to Use Metaheuristic algorithms?!

- When space to be searched is large.
- When the “best” solution is not necessarily required.
- Approach to solving a problem not well-understood.
- Problems with many parameters that need to be simultaneously optimized.
- Problems that are difficult to describe mathematically.

Evolutionary Algorithms (EAs) as problem solver: Goldberg's 1989 view



Michalewicz' 1996 view



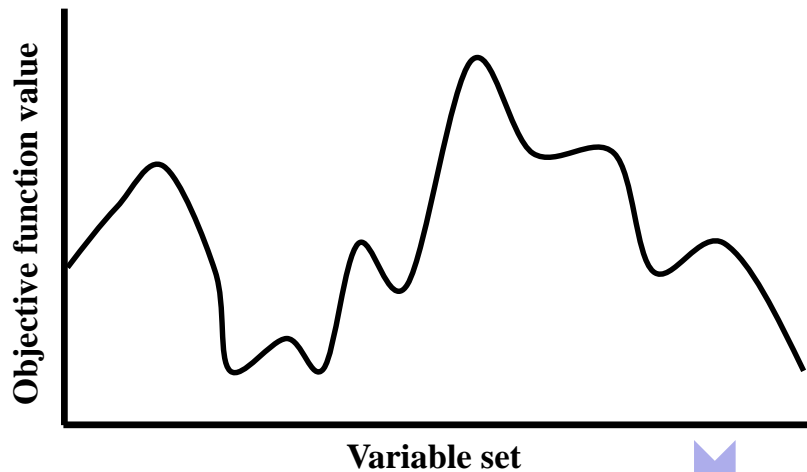
Two Important Characteristics of Metaheuristics

- **Diversification** – makes sure the algorithm explores the search space globally
- **Intensification** – intends to search locally and more intensively
- **A fine balance** between these two components is very important to the overall efficiency and performance of an algorithm
- Furthermore, needs *Survival of the Fittest*

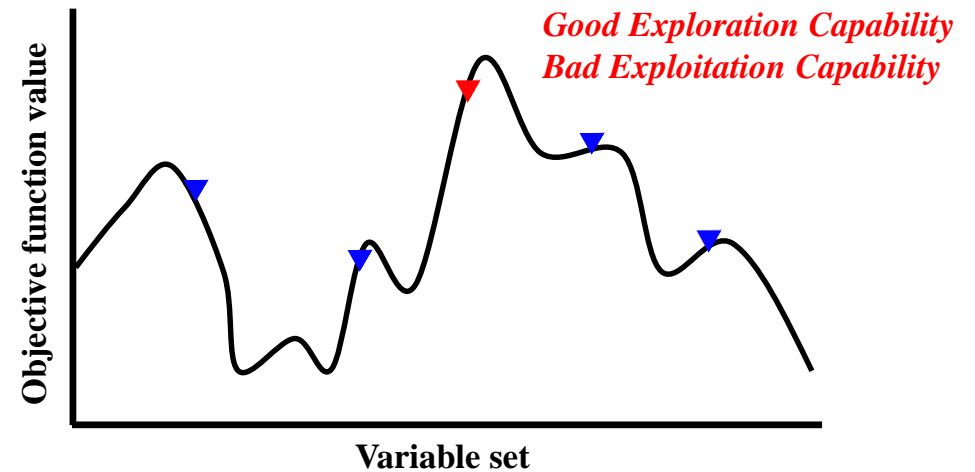
Exploration vs. Exploitation

Objective : Maximization of Function Value

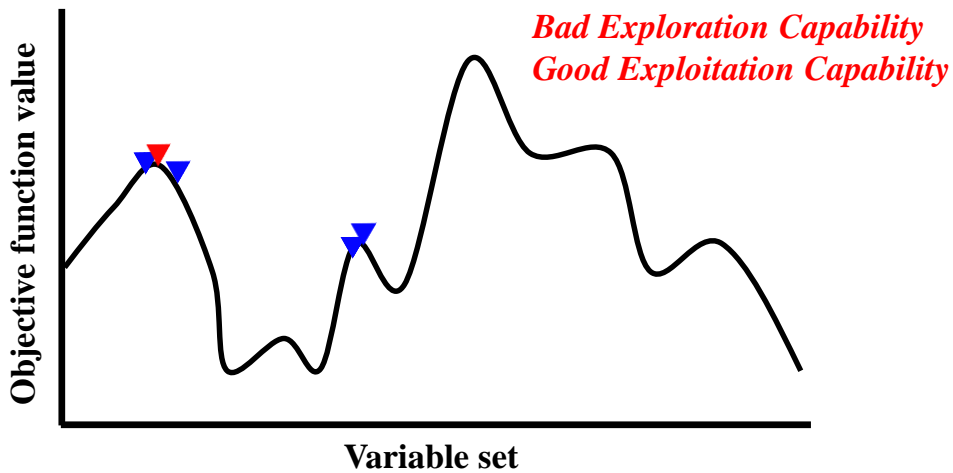
Searching Space



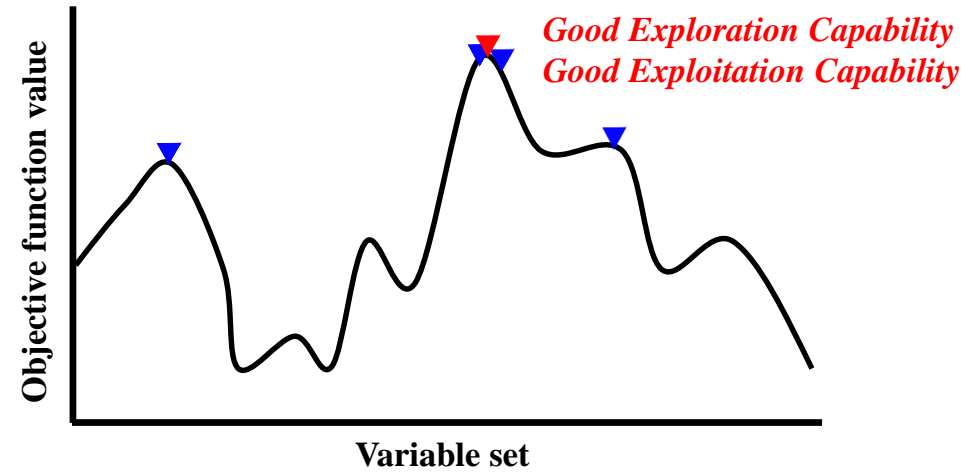
Final Solution Using Algorithm 1



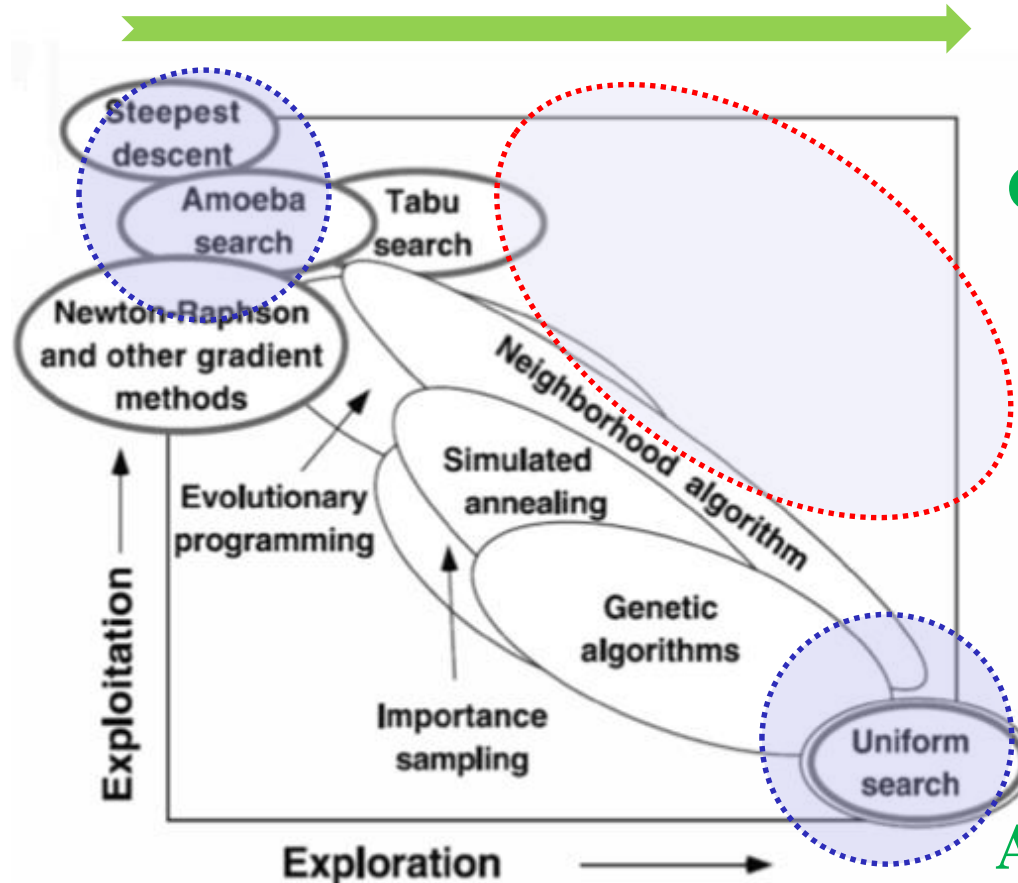
Final Solution Using Algorithm 2



Final Solution Using Algorithm 3



Exploration vs. Exploitation



Meta-heuristic Algorithms

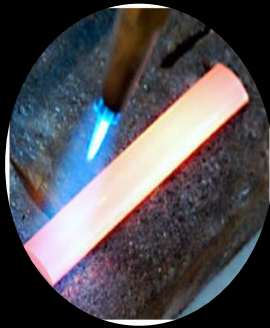
• Types of Algorithms - Initial Stage



Holland (1975)
Genetic Algorithm



Glover (1977)
Tabu Search



Kirkpatrick et al. (1983)
Simulated Annealing



Dorigo (1992)
Ant Colony Optimization



Kennedy and Eberhart (1995)
Particle Swarm Optimization



Storn and Price (1996)
Differential Evolution



Geem and Kim (2001)
Harmony Search

1970s

2001

Meta-heuristic Algorithms

• Types of Algorithms - Recent



**Geem and Kim
(2001)
Harmony
Search**



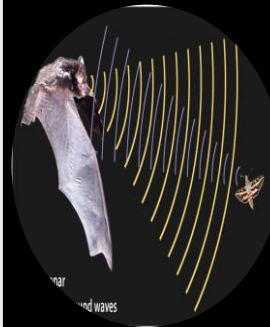
**Nakrani and
Tovey (2004)
Honeybee
Algorithm**



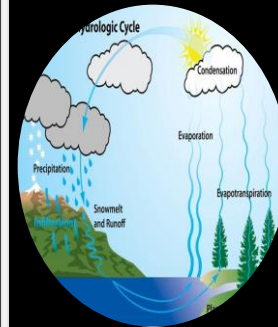
**Yang (2007)
Firefly
Algorithm**



**Yang and Deb
(2008) Cuckoo
Search**



**Yang (2010)
Bat Algorithm**



**Eskandar et al.
(2012)
Water Cycle
Algorithm**



**Sadollah et al.
(2012)
Mine Blast
Algorithm**

2000s

2013

An aerial photograph of a mountain valley. In the foreground, a winding river with a light blue-green hue flows through a lush green landscape. The river meanders through a valley floor covered in dense forest and patches of lighter green vegetation. In the background, steep, rugged mountains rise, their slopes covered in dense green forests. The mountain peaks are partially shrouded in white clouds, and the sky above is filled with large, billowing white clouds. The overall scene depicts a natural water cycle in a mountainous region.

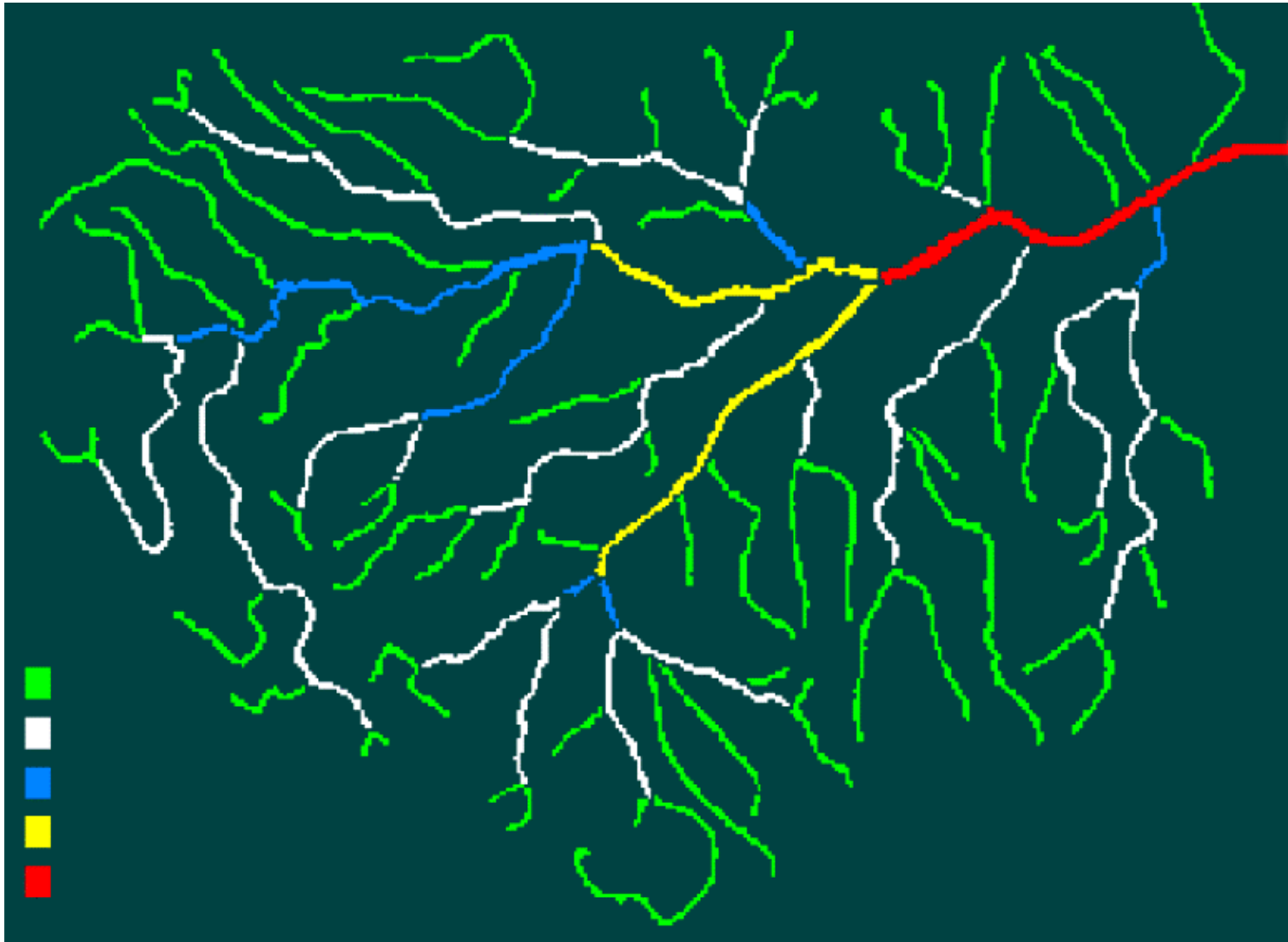
Water Cycle Algorithm

Water Cycle Algorithm: Basic Concept

Concepts: Water cycle process (Hydrologic cycle)



Order of streams



Sea

Steps of Water Cycle Process

1. Precipitation 

2. Surface Runoff 

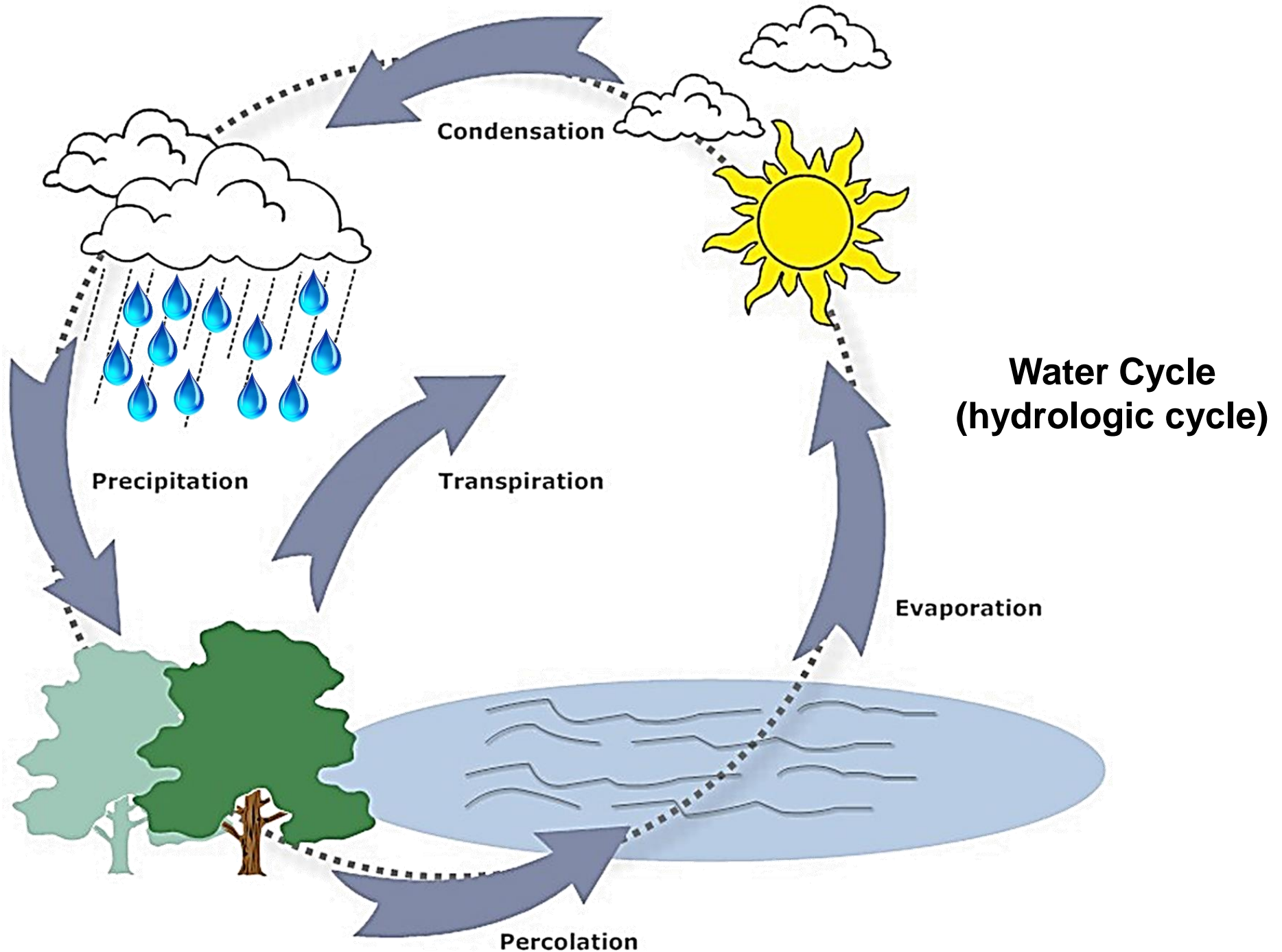
3. Infiltration 

4. Transpiration 

5. Evaporation and Condensation 



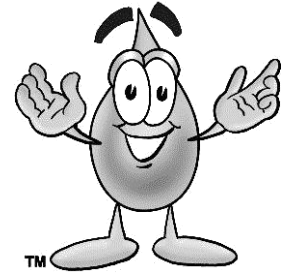
Schematic view of Water Cycle Process



Water Cycle Algorithm: Formulations

$$N_{SR} = \text{Number of Rivers} + \underbrace{1}_{\text{Sea}}$$

$$N_{Streams} = N_{Pop} - N_{SR}$$



$$\text{Population of Streams} = \begin{bmatrix} \text{Stream}_1 \\ \text{Stream}_2 \\ \text{Stream}_3 \\ \vdots \\ \text{Stream}_{N_{Streams}} \end{bmatrix} = \begin{bmatrix} x_1^1 & x_2^1 & x_3^1 & \dots & x_N^1 \\ x_1^2 & x_2^2 & x_3^2 & \dots & x_N^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{N_{Streams}} & x_2^{N_{Streams}} & x_3^{N_{Streams}} & \dots & x_N^{N_{Streams}} \end{bmatrix}$$

$$\text{Total Population} = \begin{bmatrix} \text{Sea} \\ \text{River}_1 \\ \text{River}_2 \\ \text{River}_3 \\ \text{Stream}_4 \\ \text{Stream}_5 \\ \text{Stream}_6 \\ \vdots \\ \text{Stream}_{N_{pop}} \end{bmatrix} = \begin{bmatrix} x_1^1 & x_2^1 & x_3^1 & \dots & x_N^1 \\ x_1^2 & x_2^2 & x_3^2 & \dots & x_N^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{N_{pop}} & x_2^{N_{pop}} & x_3^{N_{pop}} & \dots & x_N^{N_{pop}} \end{bmatrix}$$

$$\text{Cost}_i = f(\text{Stream}_i) = f(x_1, x_2, x_3, \dots, x_N) \quad i = 1, 2, 3, \dots, N$$

Water Cycle Algorithm: Formulations

In order to designate streams to rivers and sea which depends on the intensity of the flow:

$$NS_n = \text{round} \left\{ \left| \frac{Cost_n}{\sum_{i=1}^{N_{SR}} Cost_i} \right| \times N_{Streams} \right\}, \quad n = 1, 2, \dots, N_{SR}$$

New positions for streams and rivers may be given as: $1 \prec C \prec 2$

$$\vec{X}_{Stream}^{i+1} = \vec{X}_{Stream}^i + rand \times C \times (\vec{X}_{River}^i - \vec{X}_{Stream}^i)$$

$$\vec{X}_{Stream}^{i+1} = \vec{X}_{Stream}^i + rand \times C \times (\vec{X}_{Sea}^i - \vec{X}_{Stream}^i)$$

$$\vec{X}_{River}^{i+1} = \vec{X}_{River}^i + rand \times C \times (\vec{X}_{Sea}^i - \vec{X}_{River}^i)$$



WCA: Evaporation Condition



$$\text{If } \left\| \vec{X}_{Sea}^i - \vec{X}_{River}^i \right\| < d_{\max} \quad i = 1, 2, 3, \dots, N_{SR} - 1$$

Evaporation and Raining Process

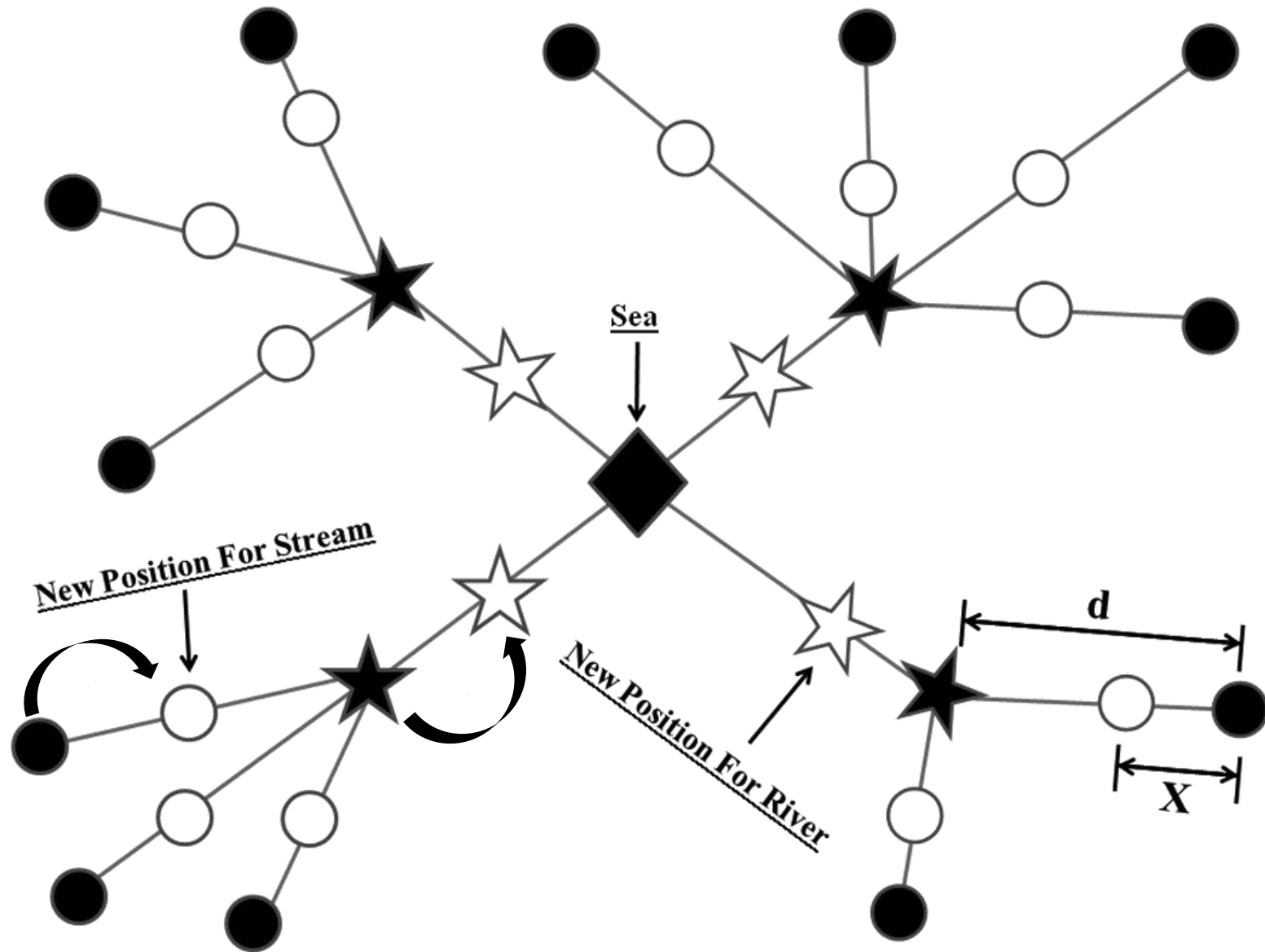
End

$$d_{\max}^{i+1} = d_{\max}^i - \frac{d_{\max}^i}{\text{Max Iteration}}$$

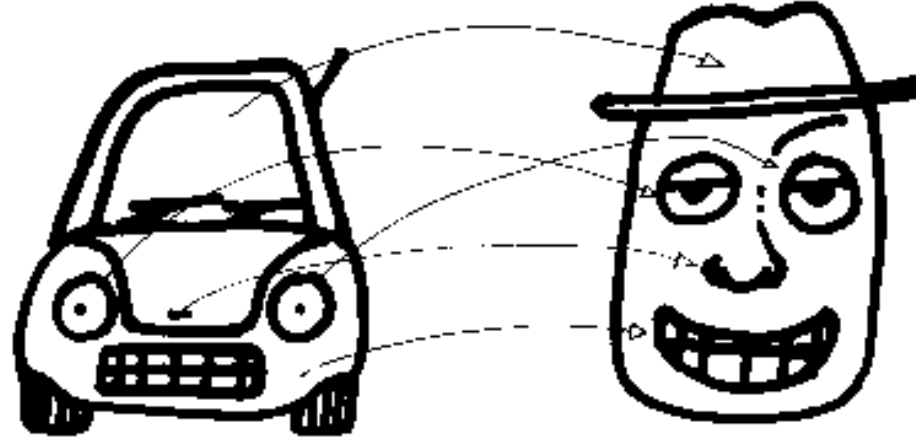
After satisfying evaporation condition, the raining process must be applied:

$$\vec{X}_{Stream}^{new} = LB + rand \times (UB - LB)$$

Processes of the WCA

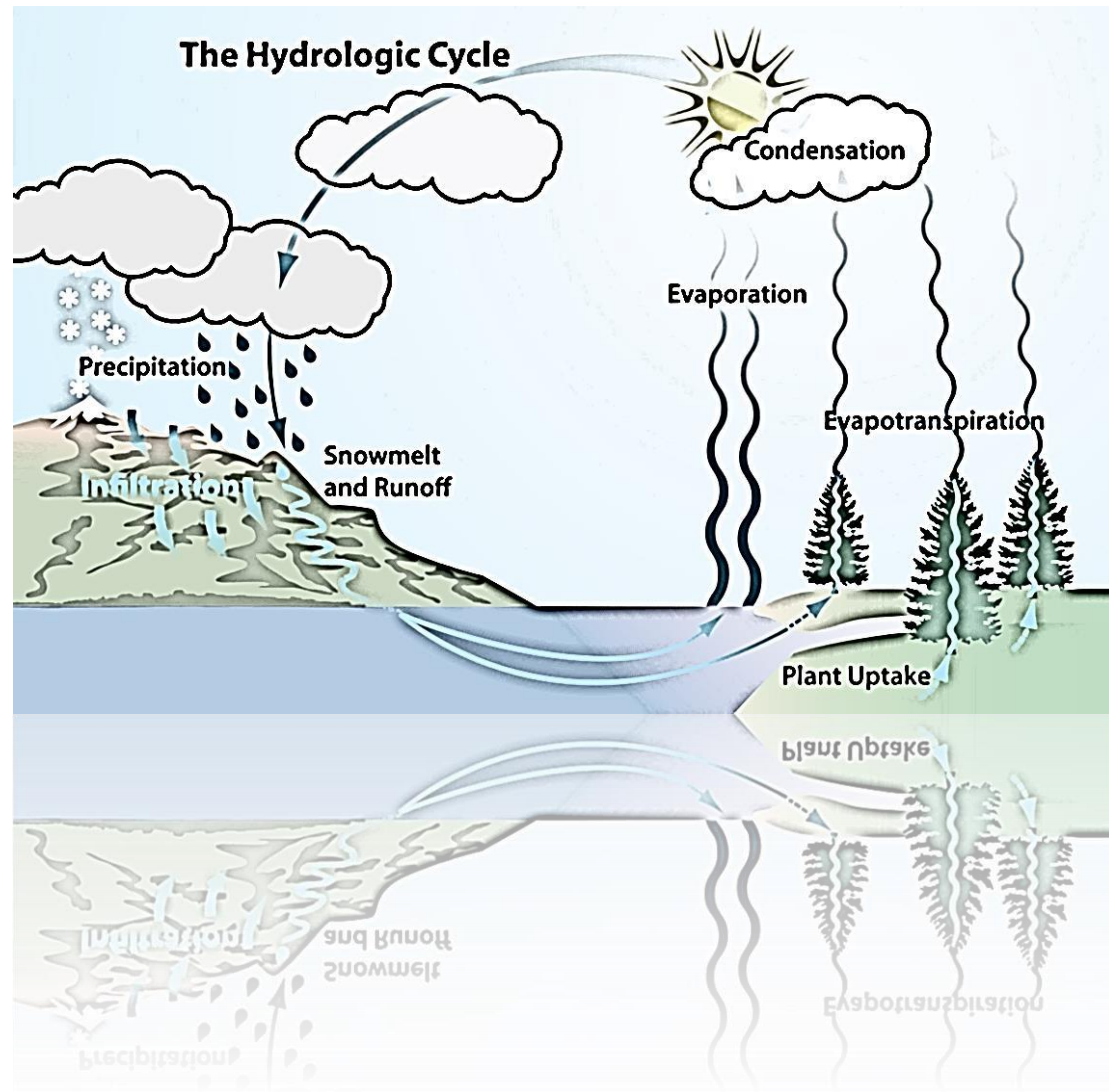
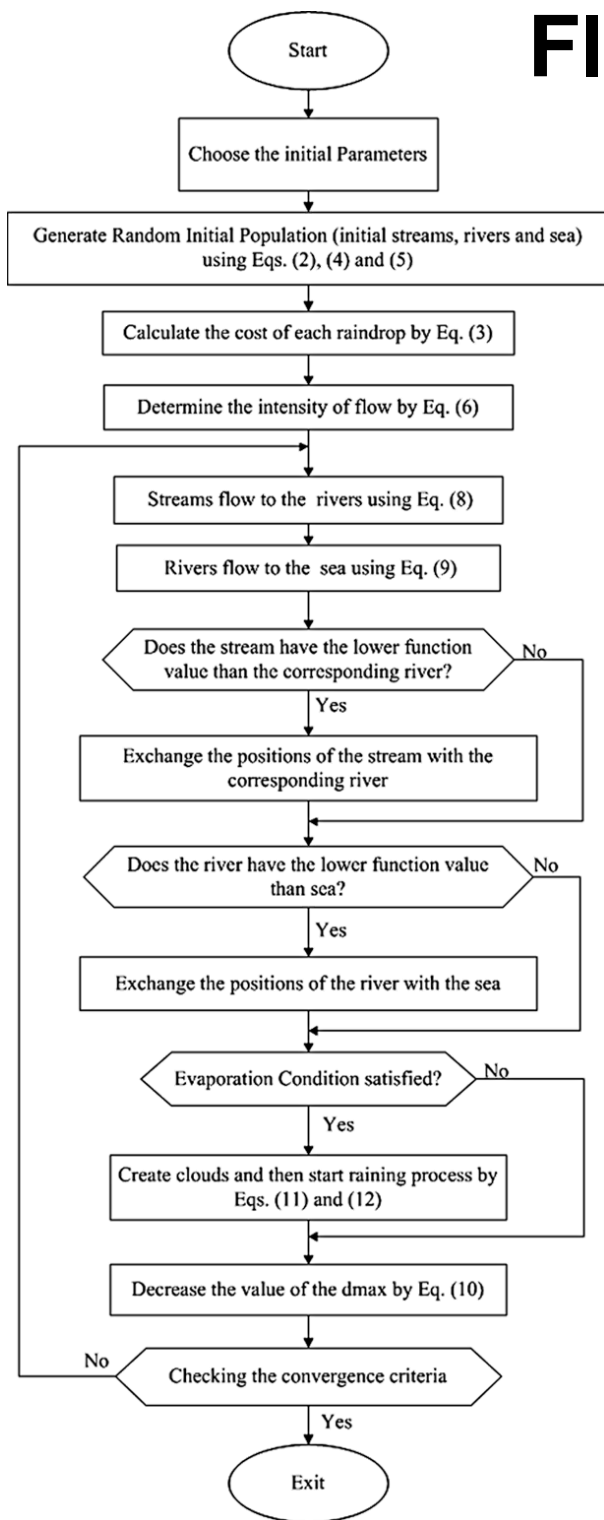


Analogy



Nature	Water Cycle Algorithm
Precipitation	Initial Population
Stream(s)	Individual(s) of population
River(s)	Second best solution (a number of best solution)
Sea	Best solution (optimum solution)
Surface Runoff	Moving streams to rivers, and rivers to sea
Evaporation	Evaporation condition
Water cycle process	Iteration

Flowchart of the WCA



- Set user parameter of the WCA: N_{pop} , N_{gr} , and $Max_Iteration$
- Determine the number of streams (individuals) which flow to the rivers and sea:

$$N_{gr} = \text{Number of Rivers} + \underbrace{1}_{\text{Sea}}$$

$$N_{Streams} = N_{pop} - N_{gr}$$

- Create randomly initial population.
- Define the intensity of flow (How many streams flow to their corresponding rivers and sea):

$$NS_n = \text{round} \left\{ \frac{Cost_n}{\sum_{i=1}^{N_{gr}} Cost_i} \times N_{Streams} \right\}, \quad n = 1, 2, \dots, N_{gr}$$

while ($t < \text{Maximum Iteration}$) or (Stopping Condition)

for $i = 1 : \text{Population size } (N_{pop})$

Stream flows to its corresponding rivers and sea:

$$\bar{X}_{Stream}^{i+1} = \bar{X}_{Stream}^i + rand \times C \times (\bar{X}_{River}^i - \bar{X}_{Stream}^i)$$

$$\bar{X}_{Stream}^{i+1} = \bar{X}_{Stream}^i + rand \times C \times (\bar{X}_{Sea}^i - \bar{X}_{Stream}^i)$$

Calculate the objective function of the generated stream

if $F_{New_Stream} < F_{river}$

River = New_Stream;

if $F_{New_Stream} < F_{Sea}$

Sea = New_Stream;

end if

end if

River flows to its corresponding sea:

$$\bar{X}_{River}^{i+1} = \bar{X}_{River}^i + rand \times C \times (\bar{X}_{Sea}^i - \bar{X}_{River}^i)$$

Calculate the objective function of the generated river

if $F_{New_River} < F_{Sea}$

Sea = New_River;

end if

end for

for $i = 1 : \text{number of rivers } (N_{gr})$

if (distance (Sea and River) $< D_{max}$) or ($rand < 0.1$)

New streams are created:

$$\bar{X}_{Stream}^{new} = L\bar{B} + rand \times (U\bar{B} - L\bar{B})$$

end if

end for

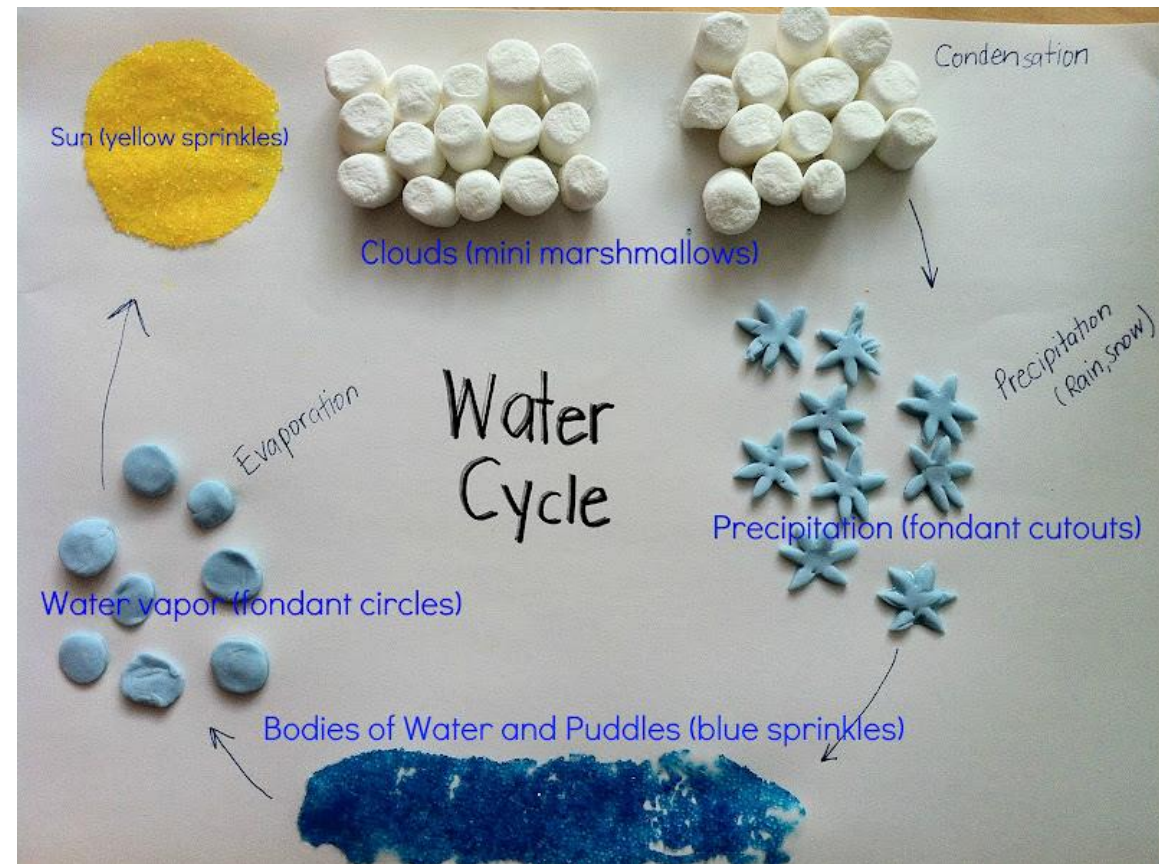
Reduce the D_{max} :

$$d_{max}^{i+1} = d_{max}^i - \frac{d_{max}^i}{\text{Max Iteration}}$$

end while

Postprocess results and visualization

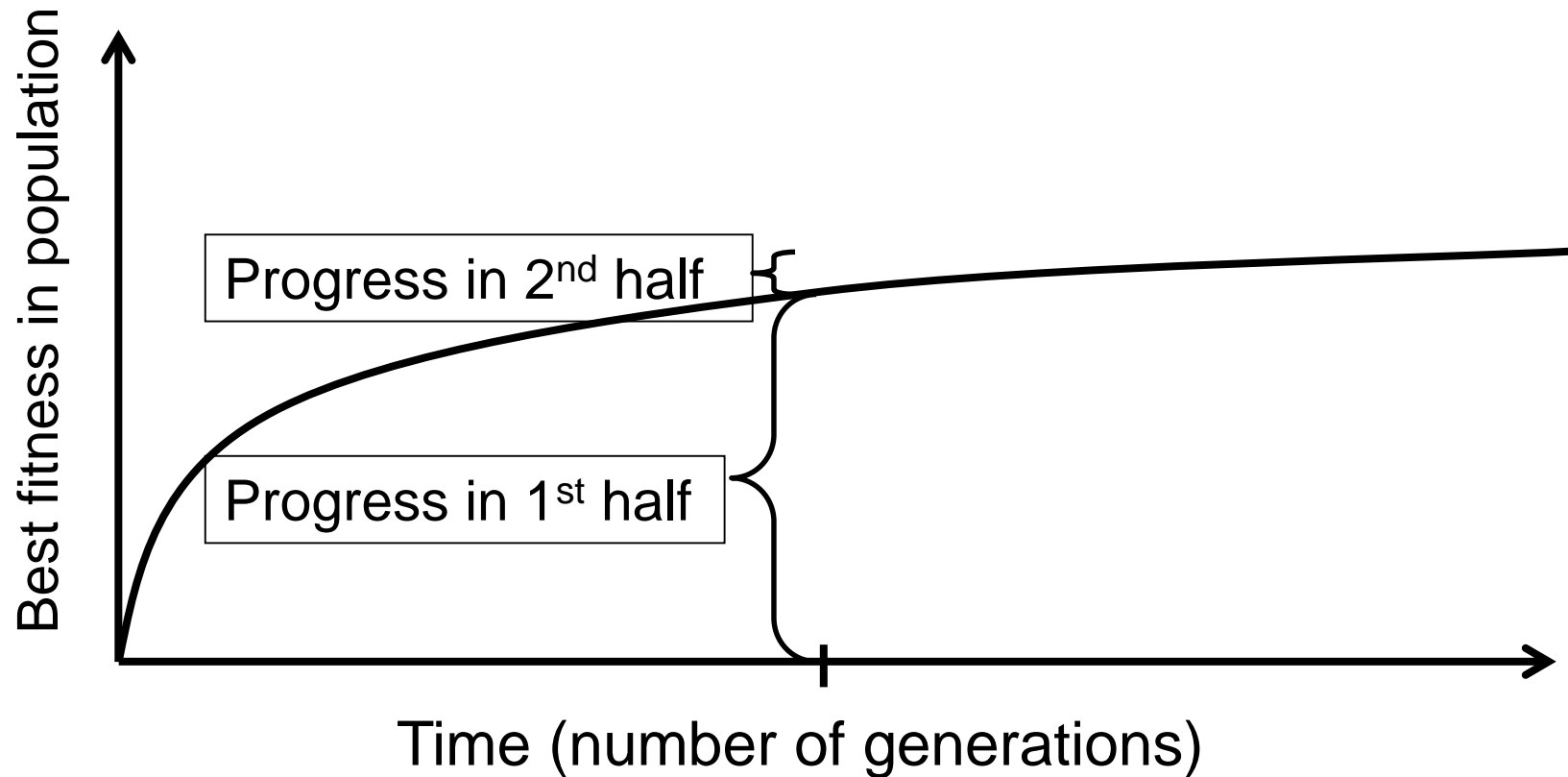
WCA: Pseudo Code



How do you know if they are converged?

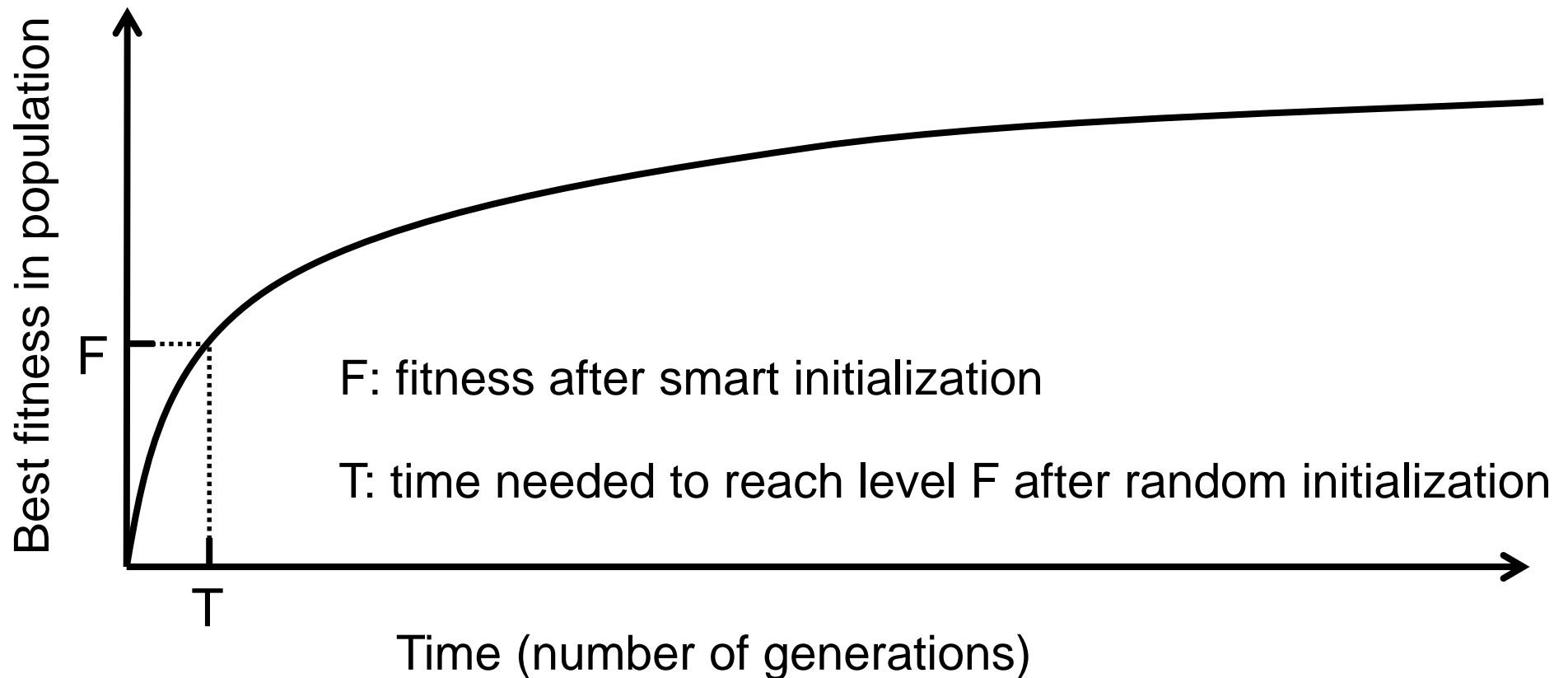
- **You don't**
- Metaheuristics are not a “black-box” optimizer for any function
- You can gain confidence by running several optimizations with different starting parameters, different algorithm options, and different parameter ranges.

Are long runs beneficial?!



- Answer:
 - it depends how much you want the last bit of progress.
 - it may be better to do more shorter runs.

Is it worth expending effort on smart initialization?



- Answer : it depends:
 - possibly, if good solutions/methods exist.
 - care is needed, see chapter on hybridization

Meta-heuristic Algorithms

● Classification of Metaheuristics

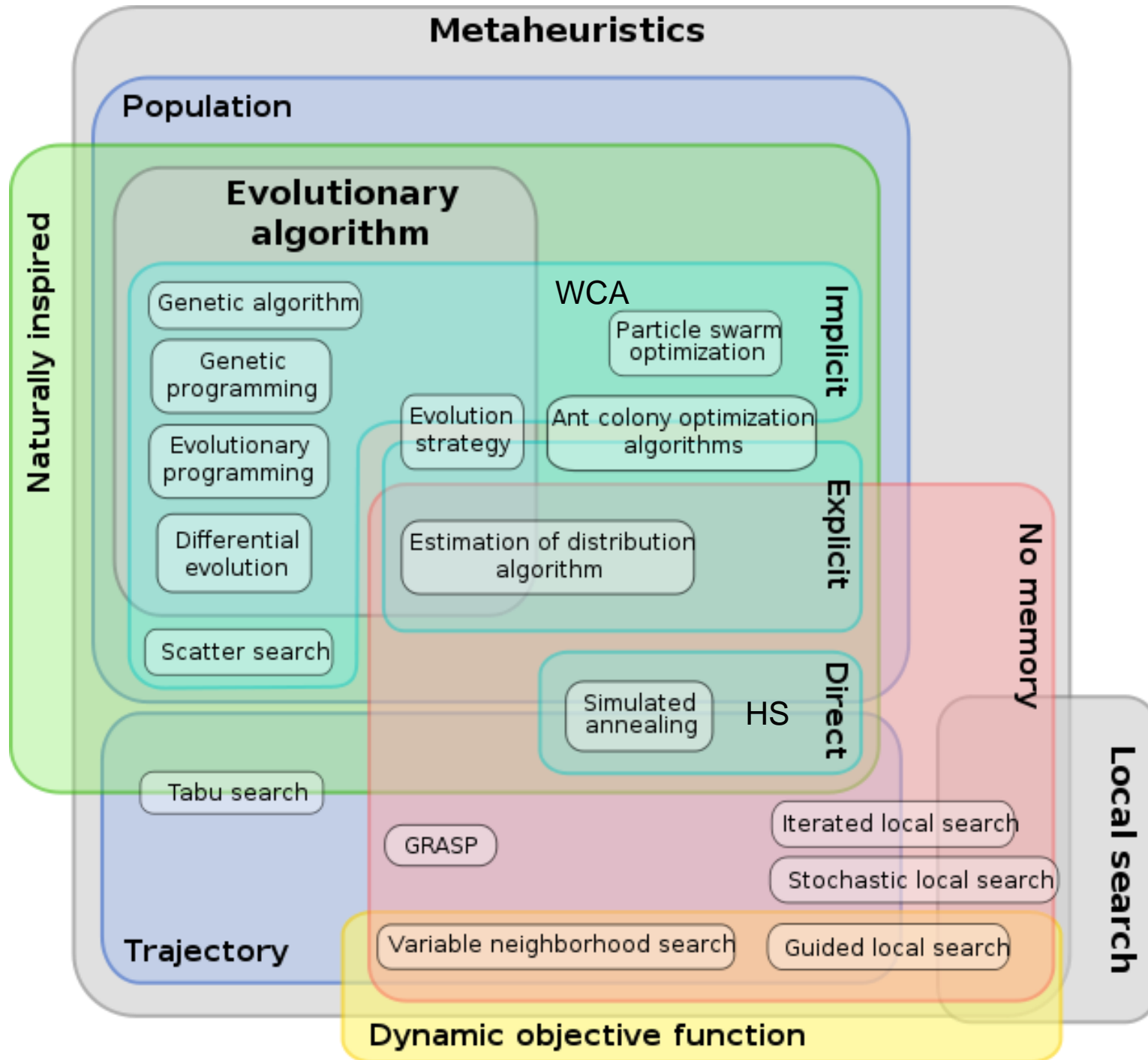
- ✓ Nature-inspired **vs.** non-nature inspired
- ✓ Population-based **vs.** single point search
- ✓ One **vs.** various neighborhood structures
- ✓ Memory usage **vs.** memory-less methods

Meta-heuristic Algorithms

• Comparison with other algorithms

Metaheuristics	Population-based vs. single point search	Using Memory	Generating Initial Solution	Number of Neighbor Solutions
GAs	Population-based	Memory less	Random	One neighbor
ACO	Population & Single based	Using memory to store amount of pheromones	Random / Local search	n neighbor solutions
SA	Single based	Memory less	Random	One neighbor
TS	Single based	Short term (tabu lists), mid term and long term memory	Local search	n neighbor solutions
HS	Population-based algorithm (Harmony Memory)	Using memory	Random	One neighbor

Metaheuristic Diagram



Thank you for your kind attentions

