

## **INTRODUCTION**

### **1. Definition of Optimization**

Optimization is an important tool for engineers who strive to improve the quality of their designs and products. Since, the notion of *optimum* was adopted by Francis Y. Edgeworth in 1896, optimization remained largely, a task almost solely addressed by mathematicians and not widely spread or applied.

An optimum process usually begins with problem classification and formulation. In the formulation of the optimization problem, the objective should be clearly defined. This can be accomplished when the problem is sufficiently understood. The dependent and independent parameters and input and output relation must be clearly defined. Optimization problems occur in engineering, physical and social sciences. Frequent occurrence and the advent of fast large-scale computers has produced, beginning in 1948, a mathematical optimization discipline which has been concerned with the theory and computational algorithms associated with these problems.

A common characteristic of all design problems is the existence of many feasible solutions. The selection of the best possible solution depends on a clear definition of the interaction of all the pertinent variables affecting the problem, an explicit statement of the design objective, and an effective search procedure for locating the optimum solution in accordance with the stated objective

### **2. Search Methods: A Conceptual Overview**

#### **Classification of Optimization Problems**

1. Unconstrained Problems with Single Variables
  - a- Differential Functions,
  - b- Computable Functions (Exhaustive, Dichotomous, Golden Section, Fibonacci, etc..)
2. Unconstrained Problems with Several Variables
  - a- Continuous and Differential Functions,
  - b- Computable and Unimodal Functions (Exhaustive and sequential Exhaustive search, Univariate Search, Gradient Search, etc..)
3. Constrained Problems with Several Variables
  - a- Linear Problems – Linear Programming (Simplex Algorithm).
  - b- Non-Linear Problems – Non-Linear Programming Methods (Lagrange Multiplier, Geometric Programming, etc..)
4. Problems Structured for Multistage Decision. (Dynamic Programming)

### **3. Geometry Optimization Classes**

There are three distinct classes of geometry optimization problems. In order of computational complexity, these are: size, shape, and topological optimization.

1. Size optimization (also called cross-sectional optimization) refers to the determination of specific geometric dimensions for a pre-selected design class, such as the thickness of a shell, the size of a truss member or the radius of a

circular stress element. This class of problems has been under modern investigation for decades.

2. Shape optimization introduces additional design variables, which allow for boundary movement. Due to its increased difficulty relative to size optimization, the geometrical changes have historically been limited; however, it has gained importance in the aircraft and automotive industries, as well as others, providing improvements to turbines, airfoil shapes and connecting arms. Size optimization is a subset of shape optimization.

3. Topological optimization involves topological as well as shape and size modifications. Topological modifications can also deal with assemblies of components. The components in the assembly may be modified and components may be added, deleted or moved in the assembly in the attempt to generate an improved design.

Fig. 1 shows examples of each for the design of a beam cross-section. In sizing optimization, as shown in Fig. 1(a), a parameterized shape are held constant, while an optimal set of parameters is found. These are commonly rectilinear dimensions. Shape optimization maintains a constant topology but changes the shape, as shown in Fig. 1(b). Fig. 1(c) indicates that topology optimization is the next step; new boundaries may be created.

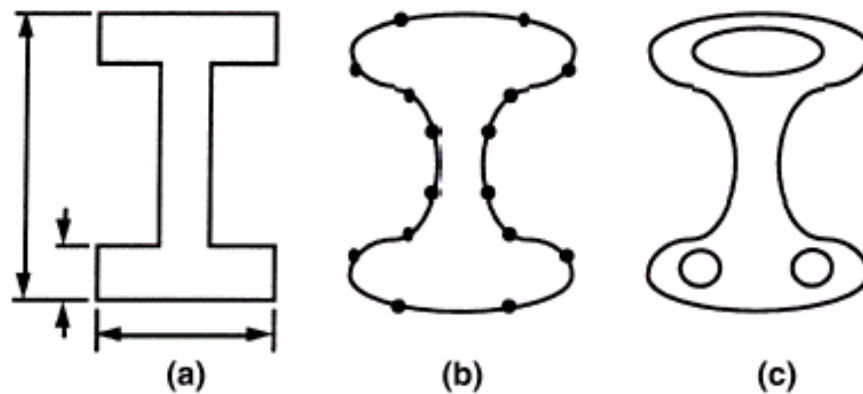


Fig. 1, Geometry Optimization Classes  
(Size, Shape, and Topology)

## 4. Random Search Optimization Methods

### 4.1 Genetic Algorithm and Evolutionary Strategies

Genetic algorithms are drawn from analogy with what occurs in nature. They are conceptually different from classical optimization techniques. Where classical techniques generally explore new candidate designs one by one, genetic algorithms operate from a *population* of designs. The initial population is typically generated randomly. A given population is used to generate new *members* (designs). Generation of the new members is performed through *reproduction* operations. The reproduction is attained by randomly selecting candidate designs from the current population and

recombining to generate new ones. The essential driving mechanism of the optimization is the selection process, which is based on *survival of the fittest*, that is, the better a candidate design is, the higher its probability of being selected. After enough new population members are produced, old members die and the new ones become the current population. Several iterations are performed while maintaining a copy of the *current best design*, which guarantees that a new generation will contain better designs or at least no worse than the current best design. Given enough generations, the current best design converges onto the global optimum.

### Advantages

- Has the global optimization capability
- Produces many design alternatives rather than just one in the case of classical methods

### Disadvantages

- May sometimes not reach the exact value of the optimum, but only close to it
- Requires much more computational operations than most classical methods

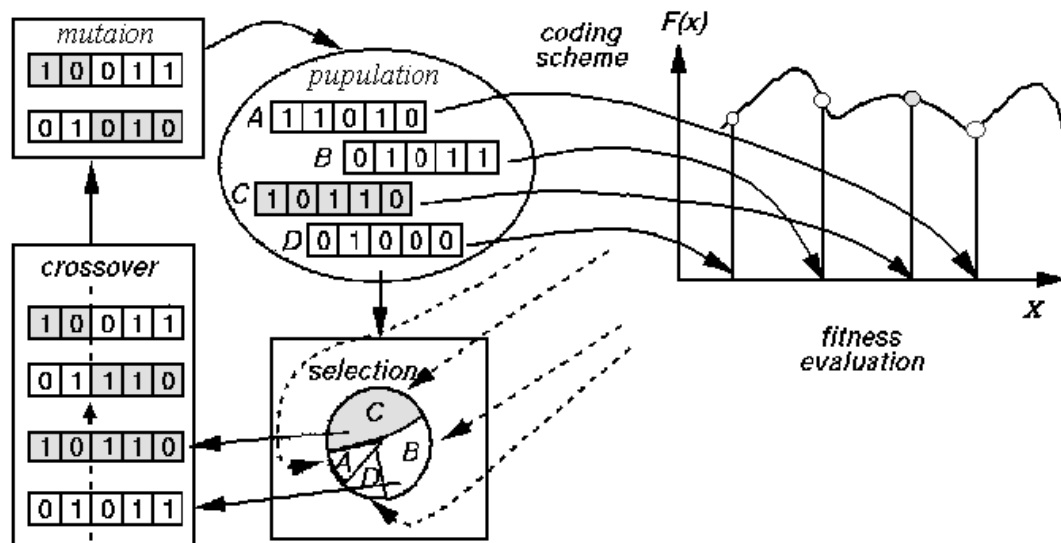


Fig. 2 GA Operators

## 4.2 Simulated Annealing

Simulated annealing is drawn from the analogy to the crystals of hot metal being annealed. When the temperature of the metal is high there is a chance that the molecules arrangements within the crystals move from a low energy arrangement to a higher energy arrangement, which is difficult to occur when the metal temperature is low. Similarly, simulated annealing starts at a single candidate design and an imaginary temperature value. The repetitive step of simulated annealing is fairly simple. A point in the neighborhood of the current point is examined. If the examined point is better than the current point, it is immediately accepted, but if not, it may be

accepted or rejected according to a certain probability that is function of the imaginary temperature value. When the imaginary temperature value is still high, accepting worse designs than the current one may allow the optimizer to escape local optima. As the iterations are repeated, the imaginary temperature value is slowly decreased until the probability of selecting a worse point becomes close to zero, which then renders simulated annealing equivalent to a local optimizer. The iterations are stopped shortly after.

### Advantages

- Has the global optimization capability
- Works well when the objective is quasi-monotonic as in many structural optimization problems

### Disadvantages

- May sometimes not reach the exact value of the optimum, but only close to it
- May perform bad if the imaginary temperature regulation was in-appropriately performed

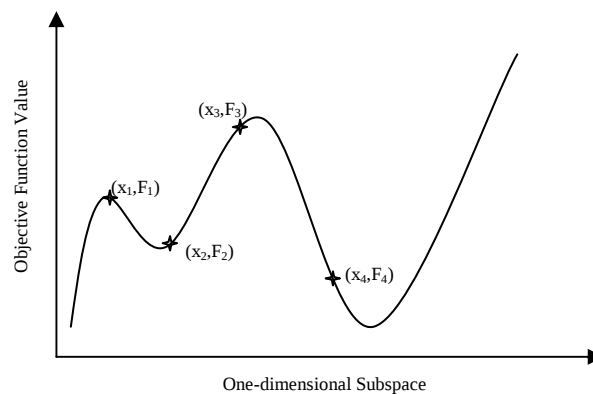


Fig. 3 Accepting higher energy level for the sake of a lower energy level

## 4.3 Tabu Search

Tabu search is heuristic optimization technique that has less stochastic content than genetic algorithms and simulated annealing. Tabu search starts at a single design point. The main optimization mechanism then evaluates candidate designs in the neighborhood of the current design and moves into the best among them. The neighborhood exploration acts like a local optimizer until a local optimum is found. Tabu search prevents entrapment at a local optimum by *memorizing* its previous moves and placing tabu conditions that discourage returning to an already explored design. Tabu search also occasionally performs random restarts.

### Advantages

- Has the global optimization capability
- Attains an optimum accurately just like classical methods

- Is more efficient in terms of the required computations than most other global optimizers

### Disadvantages

- Requires more computer memory than most other global optimizers
- Explored region of search space is smaller and more dense than genetic algorithm, which tends to be more widely spread

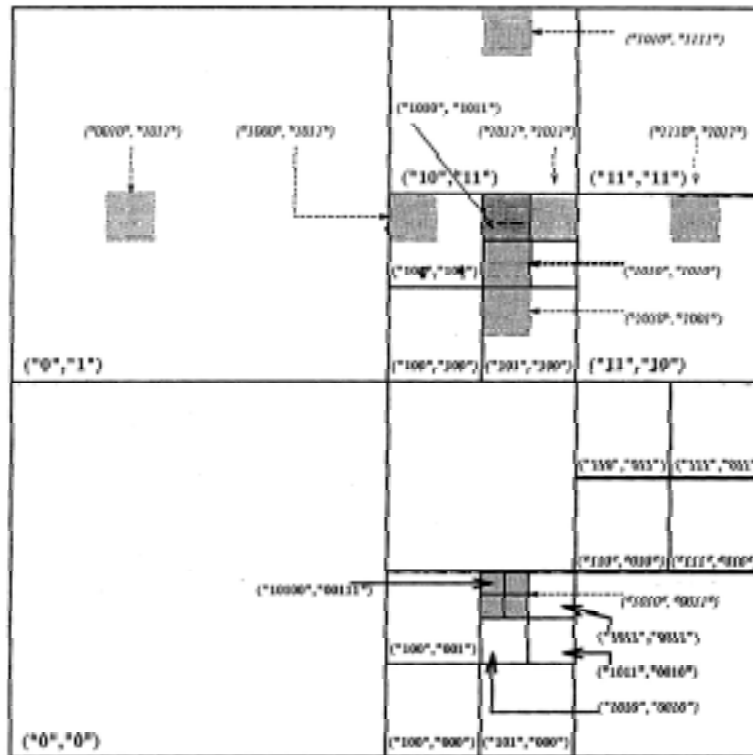


Fig. 4 Reactive Tabu Search (R-TS): Tree of search boxes