Optimization

Human-centered CAD Lab.

Mathematical Formulation

- Search for $\mathbf{X}^* \in R^n$ such that $F(\mathbf{X}^*) = \min F(\mathbf{X})$ subject to

$$\mathbf{X}_l \leq \mathbf{X}^* \leq \mathbf{X}_u \qquad \leftarrow \text{Regional constraint}$$

$$G_i(\mathbf{X}^*) \geq 0 \qquad i = 1, 2, \ldots\ldots., m$$

$$H_j(\mathbf{X}^*) = 0 \qquad j = 1, 2, \ldots\ldots., q$$

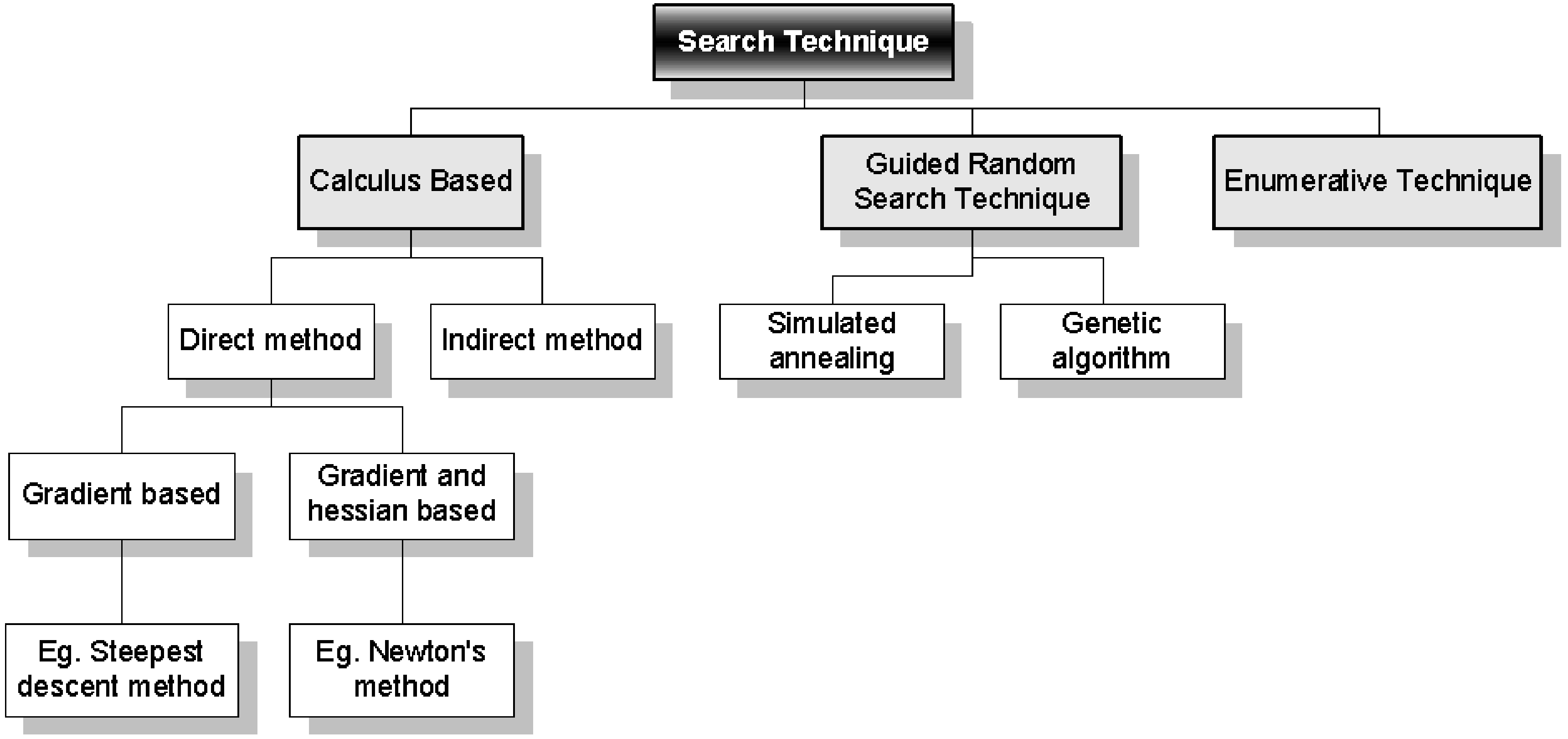
$\leftarrow$ Functional constraint

Mathematical formulation – cont’

- F(X) surface of dimension n embedded in a space of dimension n+1

- n= 2 surface in a three dimensional space
  - Similar to mountain climbing
  - Move in uphill or downhill direction while measuring local altitude
  - Falling into a trap is equivalent to violating constraints

Classification of Search Method



Search Method

- Quasi-Newton method:
  - Approximate Hessian from gradient without calculating second order derivatives
- Calculus based techniques:
  - Applied to “well behaved” problems
  - Search local minima
- Enumerative technique:
  - Evaluate the objective function everywhere in design space $\rightarrow$ search global minimum, heavy computation

Search Method – cont’

- Guided random search:
  - Search in the design space more efficiently than enumerative technique, better probability to find global minimum than calculus-base techniques
    - Simulated Annealing
    - Genetic algorithm
  - Suitable for combinatorial optimization problems
  - Select the best combination of design variable values or configuration when design variables have discrete values

Simulated Annealing

- Kirhpatrick, Cerny
- Simulate the process of reaching an equlibrium state in annealing

Physical System	Optimization Problem
State	Configuration
Energy	Cost function
Ground state	Optimal solution
Annealing	Simulated annealing

Generic Simulated Annealing Algorithm

```
begin
  S : =   Initial solution   $S_O$ ;
  T : =   Initial temperature   $T_O$ ;
  While (stopping criterion is not satisfied) do
    begin
      while (not yet in equilibrium) do
        begin
          S′ : = Some random neighboring solution of   S ;
          $\Lambda$:=C (S′) - C (S) ;
          Prob : = min (1, $e^{-\Lambda/T}$ ) ;
          if   random (0,1) $\leq$   Prob   then   S:=S′ ;
        end ;
      update T;
    end;
  Output best solution;
```

Generic Simulated Annealing Algorithm – cont'

- To use Simulated Annealing
  1. Formulate the problem with a concise description of the configurations
  2. Generate systematically the neighboring solutions of each solution
  3. Choose a suitable cost function
  4. Define an annealing schedule(specify initial temperature, rule for Changing the temperature, the duration of search at each temperature, termination condition of the algorithm)

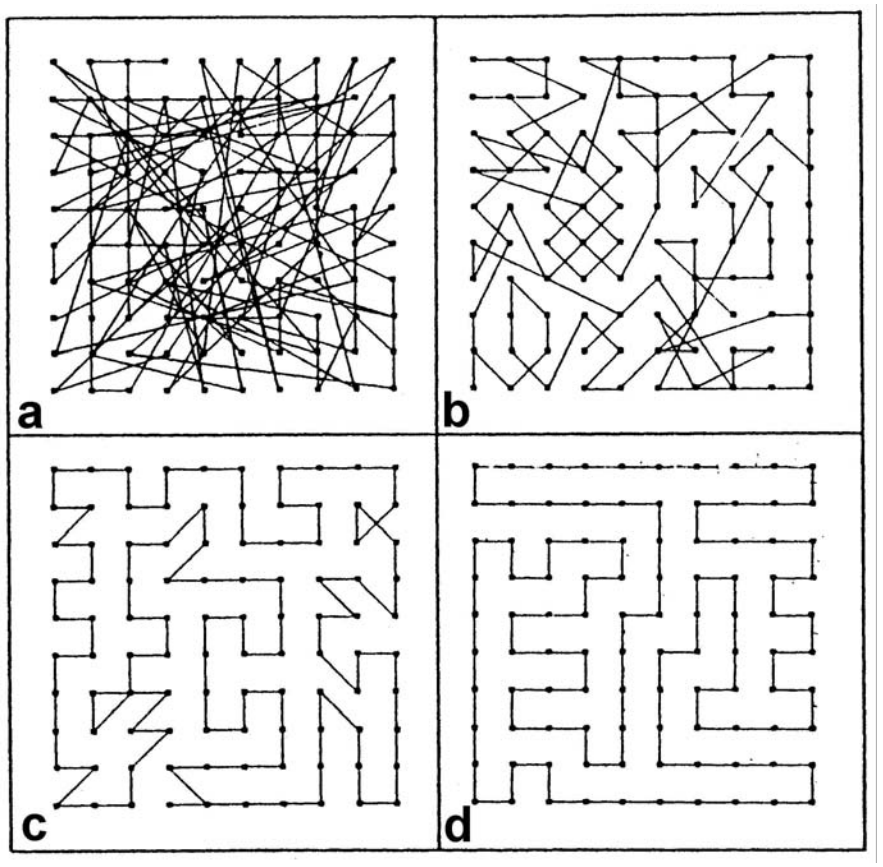
Examples –
Traveling Salesman Problem (TSP)

- Cities are numbered and each configuration is a list of these numbers.

1. Choose an arbitrary portion of the list and reverse the order

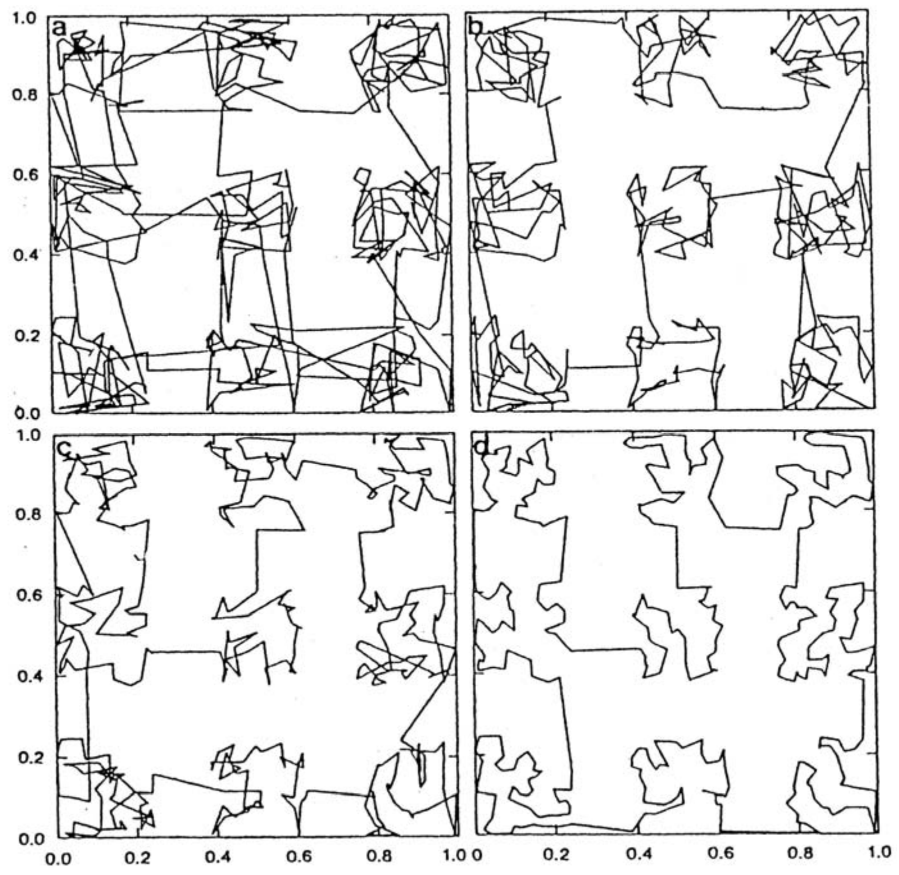
2. Move the portion chosen to another arbitrary place

Examples – Traveling Salesman Problem (TSP)



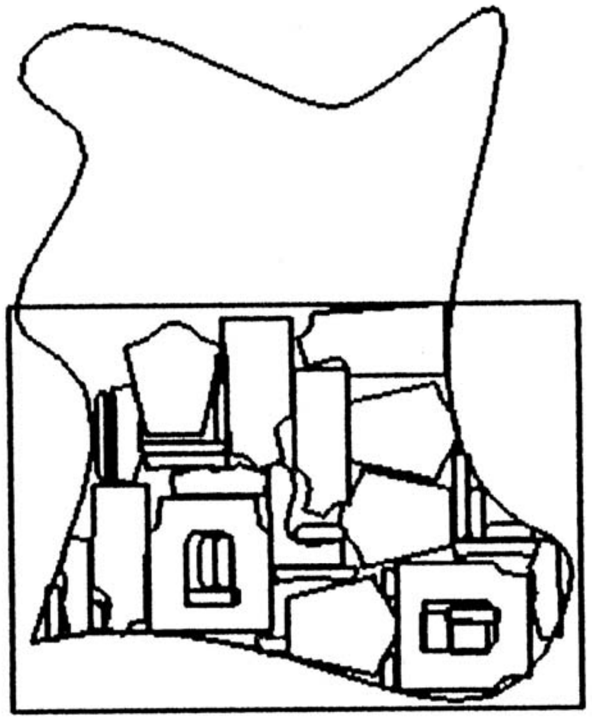
100-city traveling salesman problem solved by simulated annealing

Examples – Traveling Salesman Problem (TSP)



400-city traveling salesman problem solved by simulated annealing

Examples – Nesting Problem



Nesting of 36 patterns on an irregular raw sheet with an interior defect

Genetic Algorithm

- Simulate evolution of lives
- Encode each possible state of design variables into a string composed of 0 and 1
- Let the strings evolve into the most fitable chromosome

Genetic Algorithm – cont’

- Example : design variable x, y

$220 \leq x \leq 260$ => 32 step => 5 digits

$50 \leq y \leq 70$ => 16 sep => 4digits

$\underbrace{1\ 1\ 0\ 0\ 1}_{x}\ \underbrace{0\ 1\ 0\ 1}_{y}$: 유전자

Genetic Algorithm – cont’

```
BEGIN /* Genetic algorithm */
    Generate initial population ;
    Compute fitness of each individual ;

    WHILE NOT finished DO
    BEGIN /* Produce new generation */

    FOR population_size/2 DO
    BEGIN /* Reproduction cycle */
    Select two individuals from old generation for mating ;
    /* Biased in favor of the fitter ones */
    Recombine the two individuals to give offspring ;
    Compute fitness of the two offspring ;
    Insert offspring in new generation ;
    END

    IF population has converged THEN
    finished := TRUE ;

    END
END
```

Genetic Algorithm – cont’

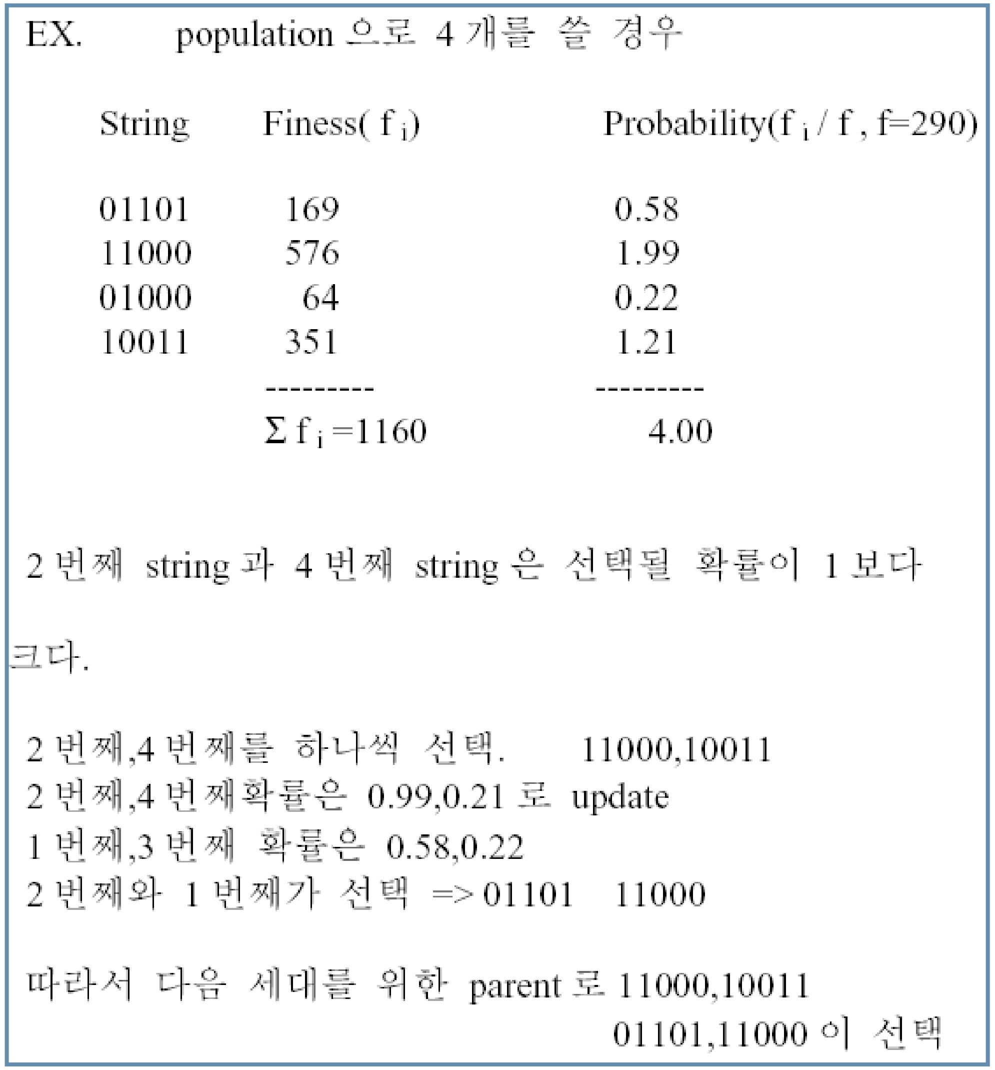
- Encoding the design variables is very simple in combinatorial optimization problems
- For real-valued continuous variables
  - Each variable is linearly mapped to an integer defined in a specified range ,then encode integer into binary bits
  - Ex: variables between -1.27 and 1.27 => -127,127

    7bit 1 bit for sign

Selection Mechanism

- Necessary to have good genes yielding good cost function participate in reproduction
- Solution with a bigger “fitness” is a good solution

Selection Mechanism – cont’



EX.　　population 으로 4 개를 쓸 경우

String	Finess(f_i)	Probability(f_i / $\bar{f}$, $\bar{f}$=290)
01101	169	0.58
11000	576	1.99
01000	64	0.22
10011	351	1.21
	---------	---------
	Σf_i =1160	4.00

2 번째 string 과 4 번째 string 은 선택될 확률이 1 보다

크다.

2 번째,4 번째를 하나씩 선택.　　11000,10011
2 번째,4 번째확률은 0.99,0.21 로 update
1 번째,3 번째 확률은 0.58,0.22
2 번째와 1 번째가 선택 => 01101　11000

따라서 다음 세대를 위한 parent 로 11000,10011
　　　　　　　　　　　　　　01101,11000 이 선택

Reproduction

- Select two parent arbitrarily
  - 11000 should have bigger probability to be selected

- Cross over
  - Cut the strings of the parents at an arbitrary location of the chromosome and exchange tail and head in a probability, generally between 0.6 and 1

$$\begin{array}{ccc} 11\ 000 & & 01\ 101 \\ & \downarrow & \\ 11101 & & 01000 \end{array}$$

Reproduction – cont’

- Apply mutation after cross over in a very low probability (e.g. 0.001)
  - Invert the gene while traversing all the genes

$$0 \rightarrow 1, \quad 1 \rightarrow 0$$

$$\begin{array}{c} 01000 \\ \;\downarrow\;\downarrow\; \\ 00010 \end{array}$$

  - Apply cross over and mutation as many as half of the population size
  - Mutation has an effect of recovering lost genetic material

Reproduction – cont’

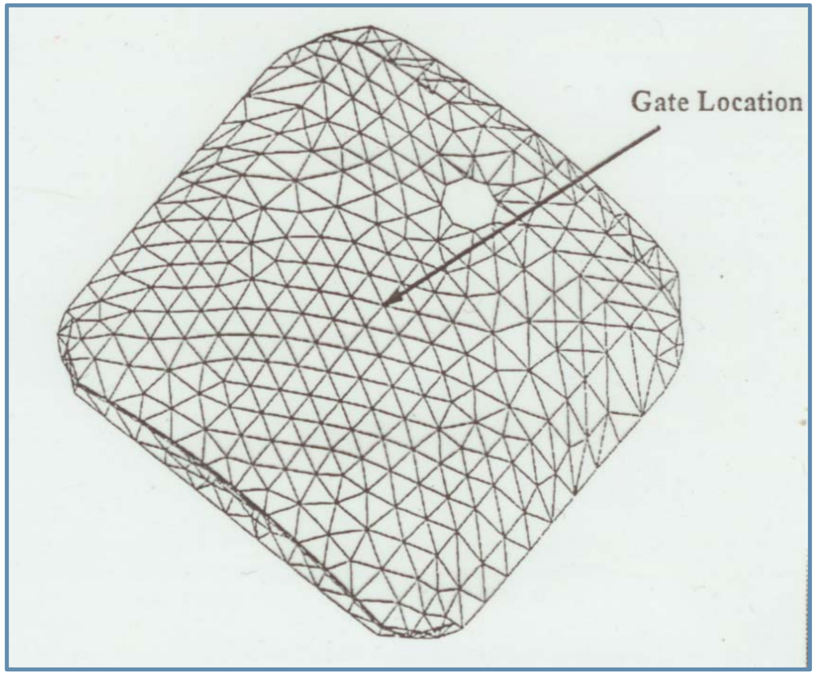
- If a specific gene is 0 for all the initial population, it cannot be changed to 1 by cross over

  → Optimal solution cannot be obtained if initial population is given badly

Convergence

- When about 95% of the population becomes the same string, the evolution is assumed to be converged

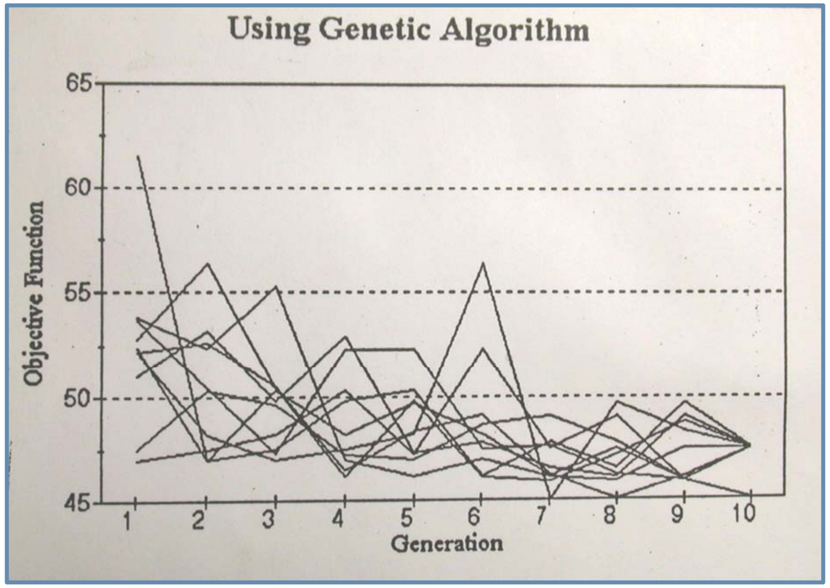
Case Study



	Minimum	Maximum	Values
Melt Temp.	220	260	32
Mold Temp.	50	70	32
Filling Time	1	4	16

Total Trials	100
Population Size	10
Crossover Rate	0.6
Mutation Rate	0.001

Case Study



	Melt Temp.	Mold Temp.	Filling Time	Object Function
Result	220.0	70.0	2.6	45.00