Application 2: Genetic Algorithm

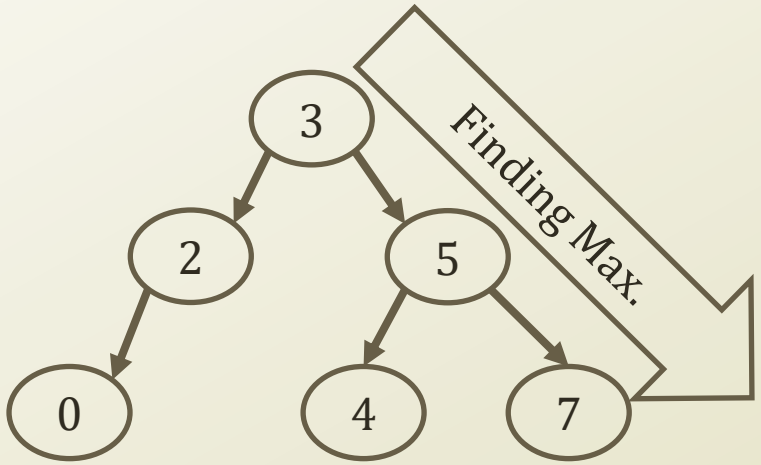
Il-Chul Moon
Dept. of Industrial and Systems Engineering
KAIST
icmoon@kaist.ac.kr

Weekly Objectives

- This week, we study the genetic algorithm and the traveling salesman problem.
- Objectives are
  - Understanding when and why the genetic algorithm is used
  - Memorizing the structure of the genetic algorithm
  - Understanding the encoding strategies of the genetic algorithms
  - Understanding the roles and the rational of the algorithm steps
    - Selection step
    - Crossover step
    - Mutation step
    - Substitution step

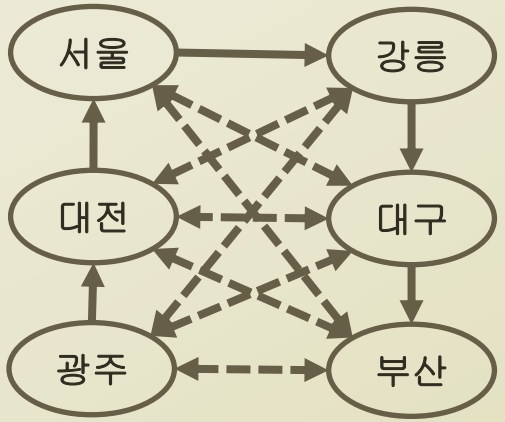
A difficult problem…

Easy Problem



Finding a maximum value of a structure
O(logN) or O(N)

Difficult Problem

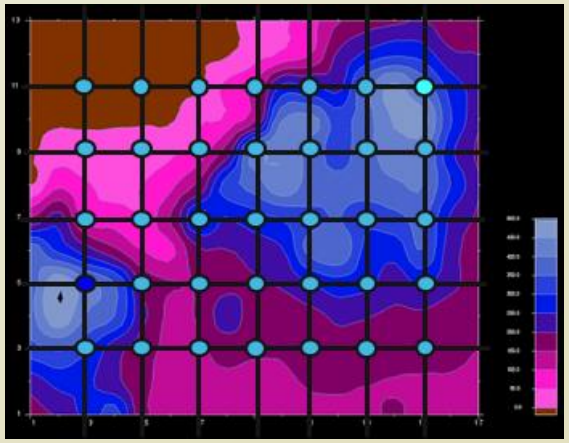


Finding a travel plan minimizing the trip time
O(N!)

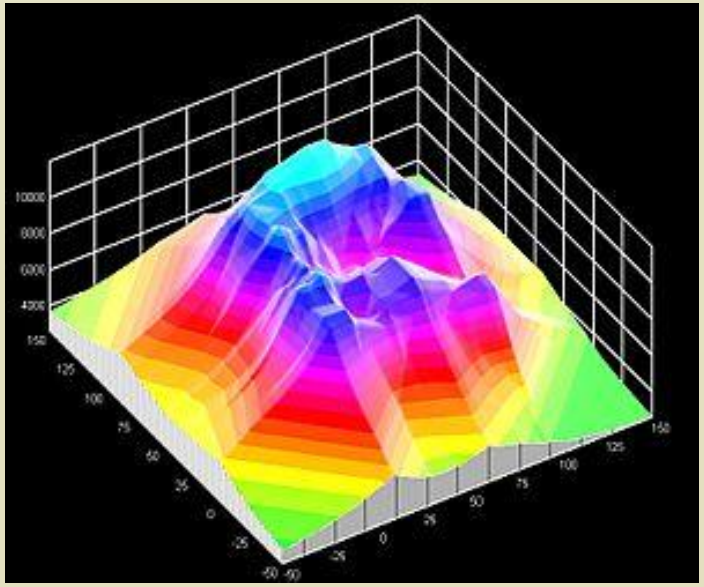
- What makes a problem difficult?
  - Difficulty in implementing an algorithm?
  - Difficulty in understanding an algorithm?
  - Difficulty in calculating a solution in a given time, or your lifetime…
- Imagine your algorithm solves a problem size X in a month…
  - with O(N!) complexity
  - If you increase a size to (X+1), then now it takes (X+1) month to solve it

Search space and global optimum

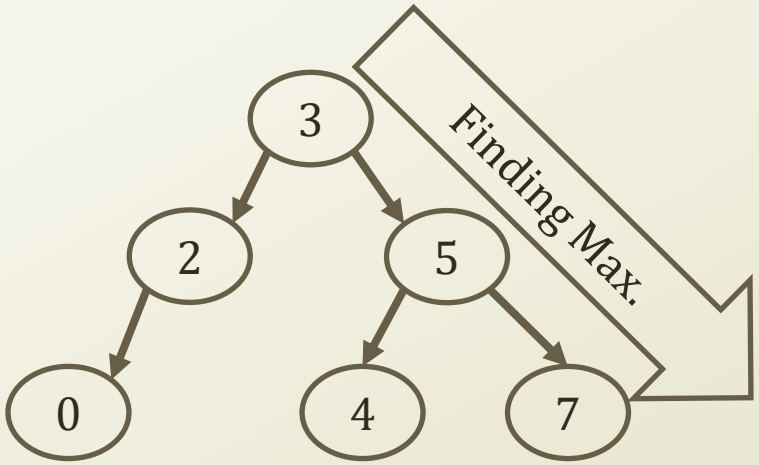
- Optimization is tightly linked to space search
  - Finding a maximum (a type of optimum) in a binary search tree
    - Search through a BST following a pre-defined paths
    - Data structure is a safe path with a consistency keeping the assumptions
- What-if there is no paths, just a big list sizing N!
  - Searching a record by a record must not be feasible
  - The list is not structured with a pattern
    - **No deterministic ways to search the space!**

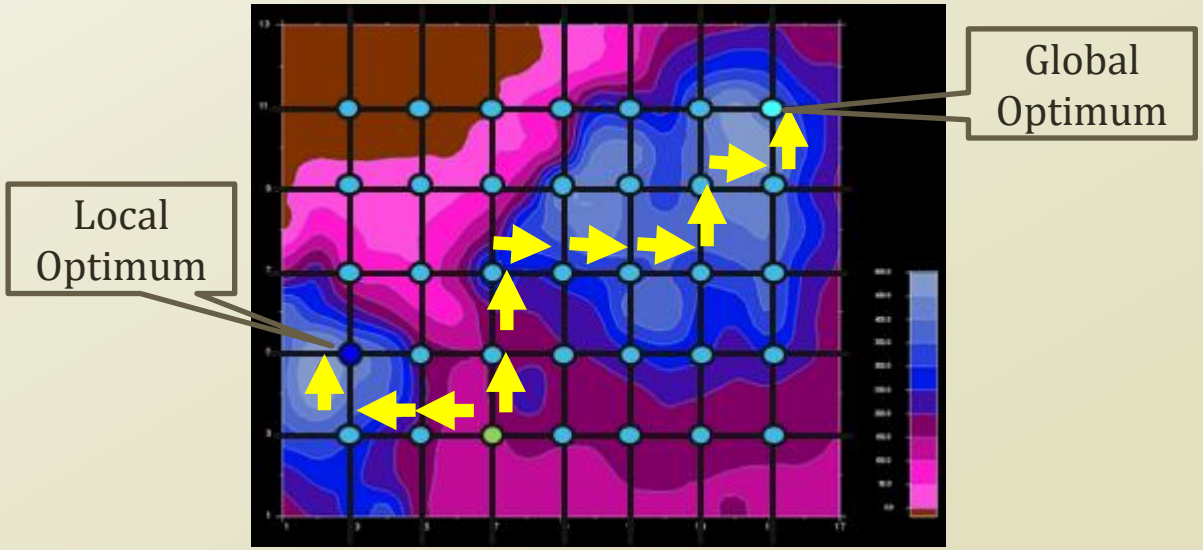


Finding the highest point in the list



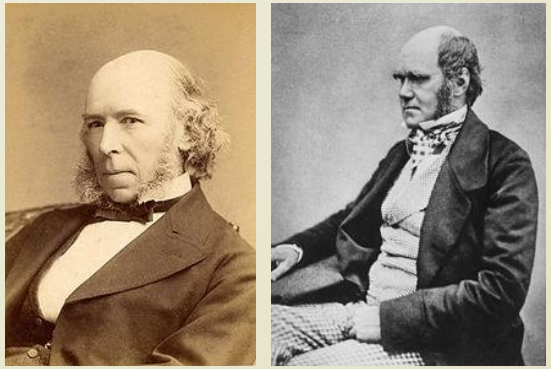
Heuristic based Optimization Techniques





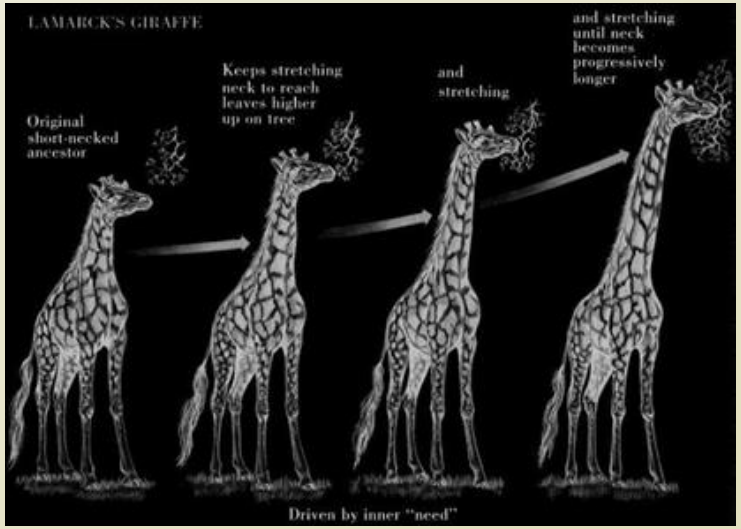
- One way to do the search….
  - Follow the higher neighbor until the search hit no way to go
  - Called *Hill-climbing heuristics* or *gradient-ascent heuristics*
  - Problem???
    - Okay, that is the highest place around the search
    - But, not the entire map
    - Falling into a local optimum
  - Why does hill-climbing have a trouble of falling into a local optimum?
- Then, is there any other heuristics?

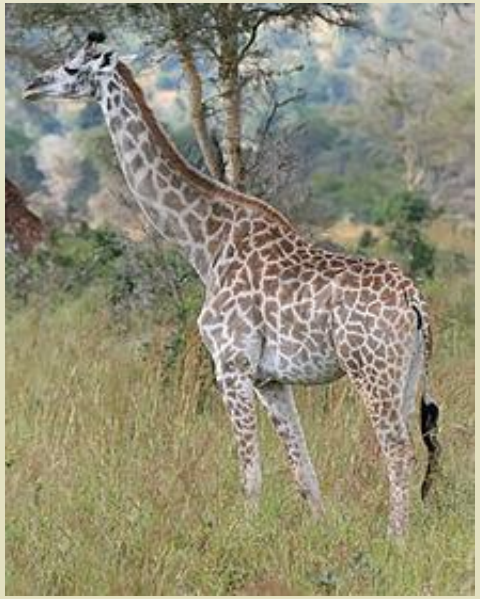
Learning from the nature



© Wikipedia

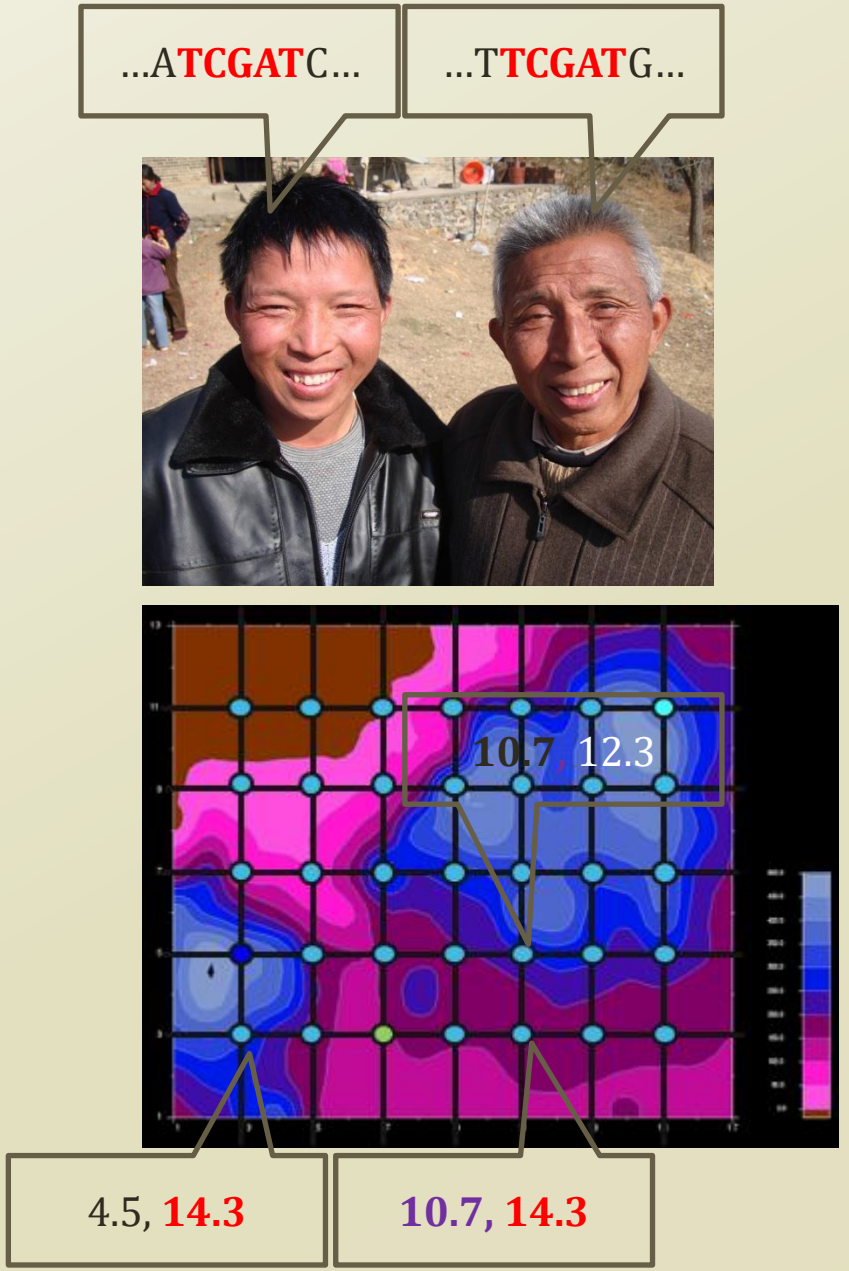
- Survival of the fittest by Herbert Spencer
- Natural selection by Charles Darwin
- Who gets to be chosen by the nature?
  - Someone who is the optimal in surviving the environment
  - Giraffe in the forest
  - By the way…
    - What is selected?
    - The giraffe or the gene of the giraffe





Genetic Algorithm

- Gene
  - Is an element consisting of genotype
- Genotype is the identify of an entity
  - Genotype of an individual: …A**TCGAT**C…
  - Genotype of a location: (**10.7,** 12.3)
- Phenotype is the observation on an entity
  - Phenotype of an individual: the look of his face
  - Phenotype of an location: the altitude of the location
- Fittest phenotype → Optimal value
- Driving genotype → Optimal solution

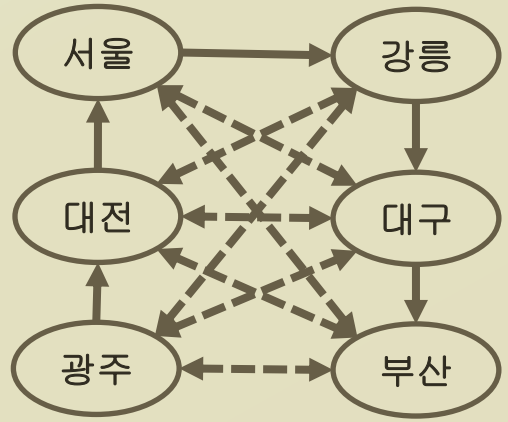


Traveling Salesman Problem

© A Walk in the Clouds

Example of difficult problems: Traveling salesman problem (TSP)

- There is a traveling salesman in a county
  - He has a big trunk where he stores his goods
    - He sells watches, jewelries, chocolates…
    - A job throughout a great depression…
  - He needs to visit as many as city possible
  - He needs to conserve his travel money
- His travel choice ranges from
  - (Seoul, Kangleung, Daejeon, Daegu, Gwangju, Pusan)…
    → Genotype
  - How many possible choices?
    - O(N!)
- Traveling salesman problem (TSP)
  - A common problem in SCM and logistics
  - A search among N! possible cases
  - Finding an cheapest route passing through all of the cities



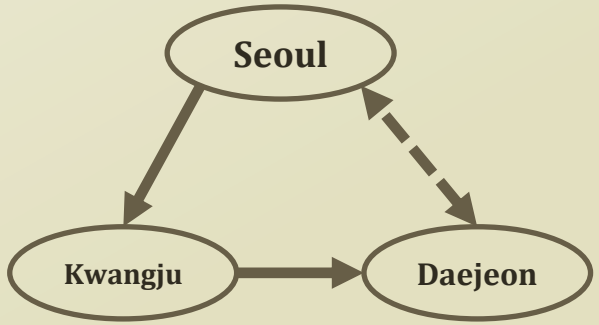
Terminology of genetic algorithm

Terminology	Implication	Application in TSP
A problem to solve	A problem that is converted and solved by GA	Traveling Salesman Problem
Gene	A factor of a solution	Seoul or Kwangju or Daejeon…
Genotype	A solution	(Seoul, Kwangju, Daejeon…)
Encoding	Representation method of a solution	(1, 2, 3, …) when assuming Seoul is 1…
Phenotype	Quality of a solution	Trip cost of (Seoul, Kwangju, Daejeon…)
Population	A set of solutions	(Seoul, Kwangju, Daejeon…) (Kwangju, Seoul, Daejeon…) (Daejeon, Kwangju, Seoul…)…
Initialization	Producing the initial set of solutions	Randomly generating the population of (Daejeon, Kwangju, Seoul…)…
Selection	Selecting a prominent sub-set of solutions	Solutions producing top K minimum trip costs
Offspring production	Producing a derived solutions from the selected sub-set	Swapping the cities in the tuple or the vector
Environment	The measurement for the evaluation of a solution	Trip cost calculator

Structure of Genetic Algorithm

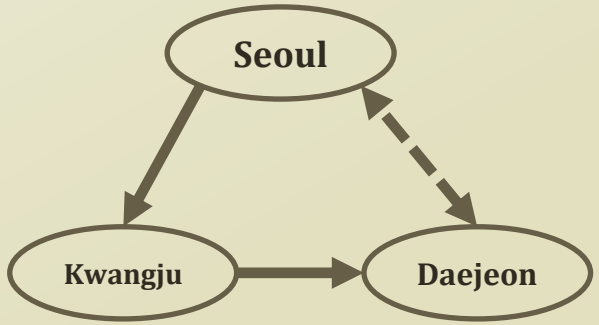
Encoding

- Encoding step
  - How to represent a solution as a vector of genes
    - Traveling 3 cities: Total 3! cases
    - Ex. (Seoul, Kwangju, Daejeon)
  - Three possible encoding strategies
    - Convenience of the representation
      - K-ary coding
      - (0, 1, 2) when 3-ary coding → 012
    - Convenience of the future operations
      - binary coding
      - (00, 01, 10) when binary coding → 000110
    - Adaptation to the problem at hand
      - Position based coding (0 = Seoul, 1 = Kwangju, 2 = Daejeon)
      - (1, 2, 0) when position based coding → 120



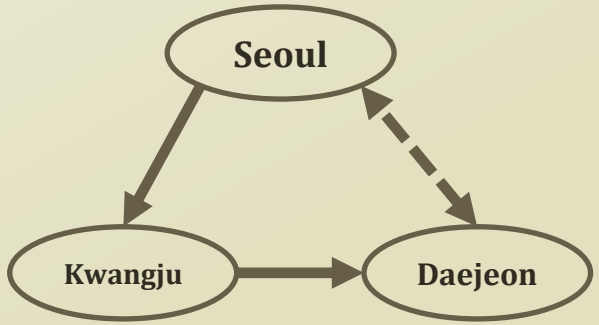
Encoding : *k*-ary coding

- Convenience of the representation
  - K-ary coding
    - (0, 1, 2) when 3-ary coding → 012
    - (00, 01, 10) when binary coding → 000110
- *K*-ary encoding
  - K = # of different genes
  - Easiest encoding…
- Binary encoding
  - Turn the K-ary numbers to the binary numbers
  - Pros: Why we turn the K-ary to the binary?
    - Easier for the cross-over and the mutation step: Flipping 0 to 1, or vice versa
  - Cons: When we perform the cross-over at the arbitrarily chosen position
    - We are losing the digit information



Encoding : position based coding

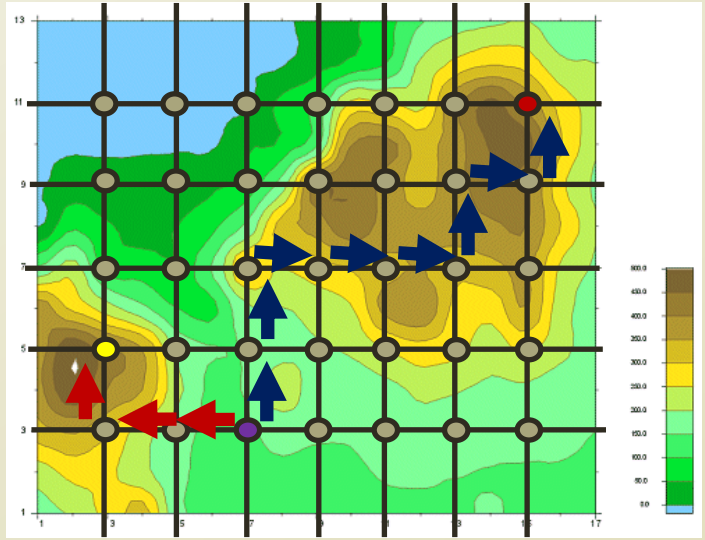
- Adaptation to the problem at hand
  - Position based coding (0 = Seoul, 1 = Kwangju, 2 = Daejeon)
    - (1, 2, 0) when position based coding → 120
  - How to?
    - The value on position *i*
      - The city to visit next to visiting the city of *i*
      - 1,2,0
        - 1 → Kwangju is the city to visit next to visiting Seoul
        - 2 → Daejeon is the city to visit next to visiting Kwangju
        - 0 → Seoul is the city to visit next to visiting Daejeon
  - Is this any different to K-ary encoding?
    - In K-ary encoding: 012 == 120 == 201
      - Three representations are the same
    - In position based encoding: 012 != 120 != 201
      - No more wasted representations
      - Now, some invalid solutions exist!
        - Is 012 valid?



Selection step

- Selection is the most delicate part of the GA
  - Natural selection
  - Selection is based upon the fitness
  - However, just adapting to the environment that we have is enough?
    - Often falls into a local optimum
  - To obtain a global optimum
    - You have to swallow some bitter tasting pills at the beginning.
    - Accept the inferior population
    - See what they eventually contribute to the solution populations
- Hence, the selection is not just dropping the inferior solutions
  - Delicately balance
    - the bigger portion of good solutions
    - the smaller portion of bad solutions

Global Optimum vs. Local Optimum

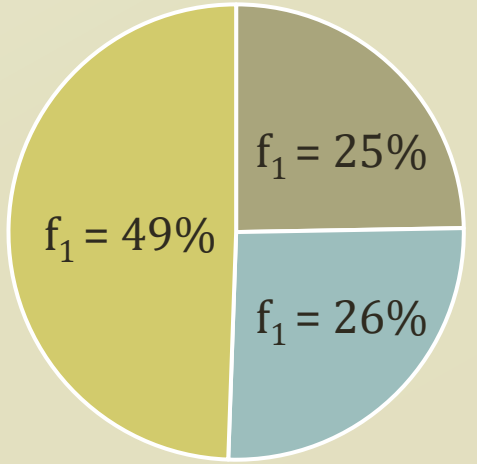


Dinosaur in the ice age
© Ice Age

Selection step: roulette wheel based

- Selection by proportion
  - $f_i = (c_w - c_i) + \frac{(c_w - c_b)}{(k-1)}, k > 1$
    - C_w = travel cost of the worst solution in the population
    - C_i = travel cost of the examined solution in the population
    - C_b = travel cost of the best solution in the population
- For example,
  - 3! possible cases, 3 cases in the population
    - (Seoul, Kwangju, Daejeon) = 463 km
    - (Kwangju, Seoul, Daejeon) = 457 km
    - (Seoul, Daejeon, Kwangju) = 329 km
  - When k = 5,
    - $f_1 = 463 - 463 + \frac{463-329}{5-1} = 33.5$
    - $f_2 = 463 - 457 + \frac{463-329}{5-1} = 39.5$
    - $f_3 = 463 - 329 + \frac{463-329}{5-1} = 167.5$
- Fitness-based proportional survival of population
  - What-if the fitness has a highly skewed distribution?

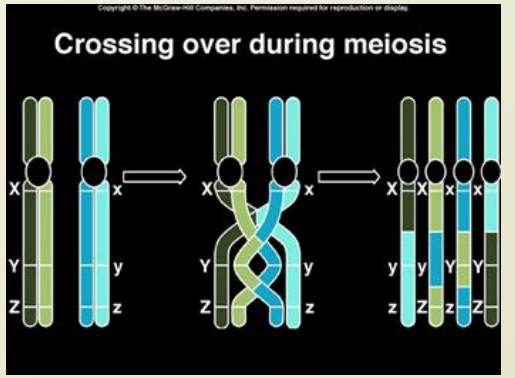
	k=2		k=5		k=11	
f1	134	25%	33.5	14%	13.4	7%
f2	140	26%	39.5	16%	19.4	11%
f3	268	49%	167.5	70%	147.4	82%
Sum	542	100%	240.5	100%	180.2	100%



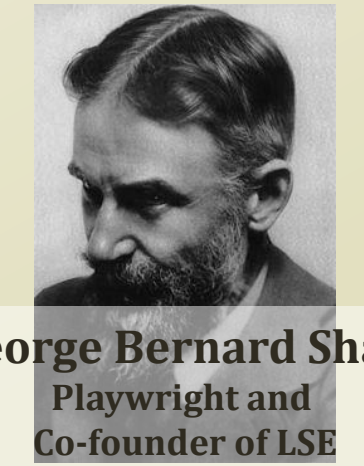
k=2

Crossover step

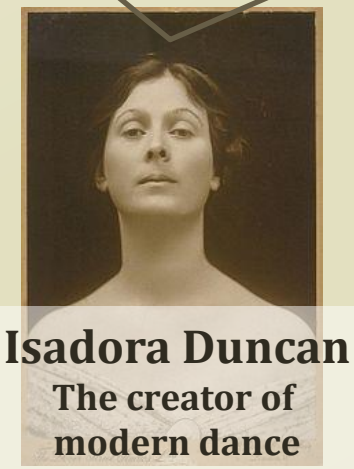
- Through the selection step, we find two solutions to be the parent of a new solution
  - The new solution is generated by mixing up the genes of the two solutions
  - Mixing up == crossover
  - The best case scenario:
    - The new solution takes the best parts from the two solutions
  - The worst case scenario:
    - The new solution takes the worst parts from the two solutions
  - Hence, we need a good crossover



"Think of it!" she said, *"With your brains and my body, what a wonder it would be."*



George Bernard Shaw
Playwright and Co-founder of LSE



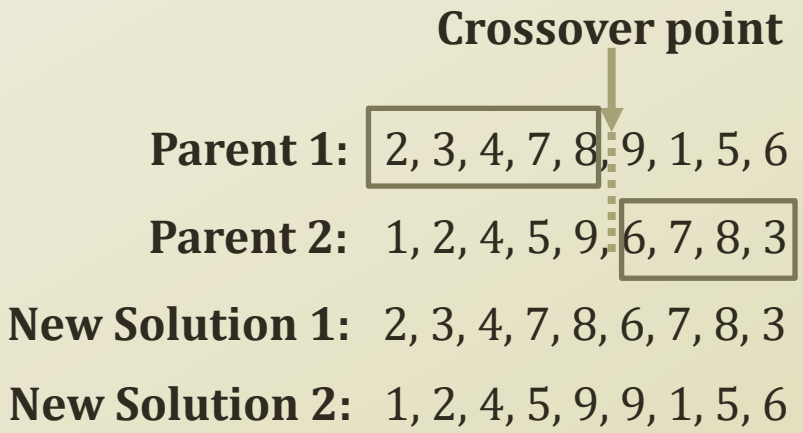
Isadora Duncan
The creator of modern dance

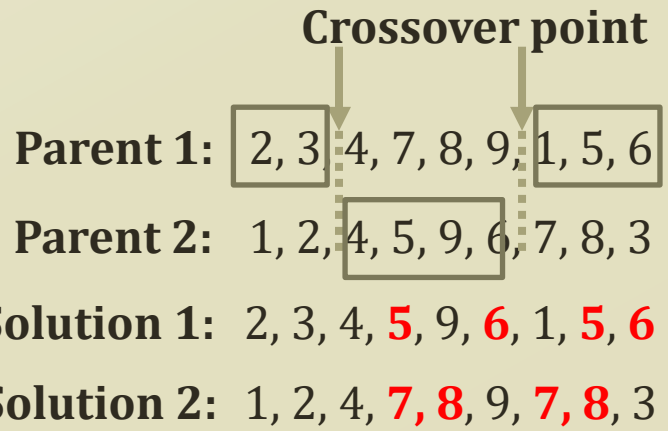
Shaw thought for a moment and replied, *"Yes, but what if it had my body and your brains?"*

© Wikipedia

Crossover step: multi-point crossover

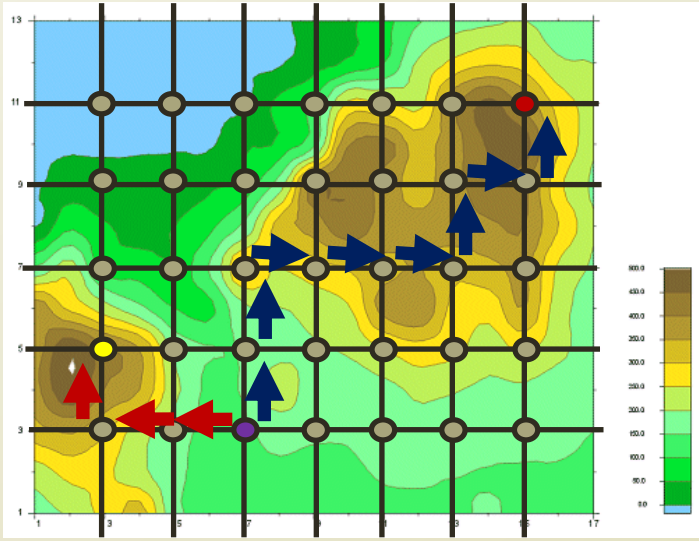
- One-point crossover
  - Select a position over a genotype
  - Choose the parts alternatively
- Multi-point crossover
  - Select multiple positions to crossover
  - Choose the parts alternatively
  - How many possible solutions?
    - Still, 2
    - Either starting the parent 1 or the parent 2
- Any problem?
  - The inconsistency problem….
    - Might not produce a valid solution
    - Two styles: produce and repair; or careful crossover





Mutation step

- Problem of hill-climbing
  - Equal to problem of Nazi
  - Strong driving force in pursuing a goal
  - However, does not see a long-term evolution
    - Hence, they eliminate the diversity of population
    - No starting points for the new ways to improve
- In GA, What-if we eliminate the diversity of the population through the selection and the crossover step?
  - At the end of the day, what we are going to play with is **selected** parents
  - Sometimes, you need to think outside the box
- Therefore, you need a **mutation**
  - In biology, mutation is one cause of producing much stronger viruses



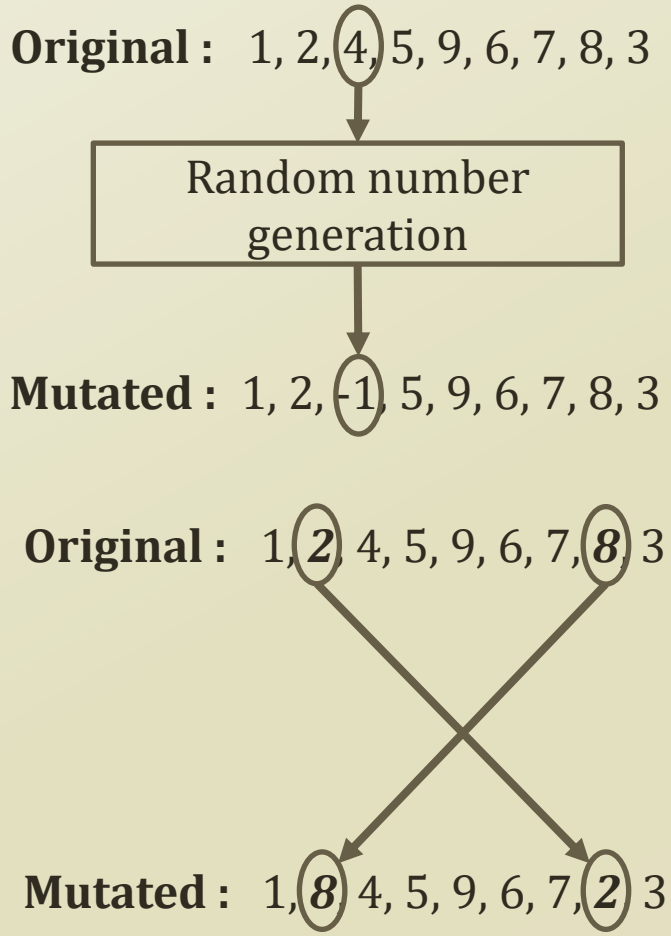
Many times, freak!

Few times, solution!

© Marvel

Mutation step: random and swap

- Simple mutations
  - Random mutations
    - If numeric variables
      - Assume a distribution
      - Draw a new number from the distribution
    - If categorical variables
      - Just the same: assume a distribution and draw a category
      - But, don't diverge from the set of the categorical values
  - Swap mutations
    - Select a number of variables to swap
    - Can be multiple or just two
    - More appropriate to the TSP
    - Why?
- Remember that this is the last step before the solutions are inserted into the population
- No invalid solutions!



Mutation step: repair

- Why we need a repair process
  - TSP
    - Every city should be visited
    - If a solutions is
      - Visiting 8 cities: 1, 2, ***5***, 4, 3, ***5***, 6, 7
      - 5 is visited twice
      - 8 is not visited
      - Typical infeasible solution
    - This can happen by
      - A random mutation
      - A crossover of various approaches
  - Infeasible solution
- How to repair
  - Random repair until the solution becomes a feasible solution
  - Greedy repair with heuristics
- Remember that the repair is a type of mutation

Substitution step

- N = the size of population, or the number of solutions in our hands
- K = the size of offspring, or the number of new solutions
- We need to create a feedback system to utilize the new solutions in the iteration process
  - How much feedback are we going to provide?
    - K/N = generation gap
    - K/N → 1
      - Replacing the most of the population
      - Called generational GA
      - Dynamic, but slow
    - K/N → 1/N
      - Replace just one solution of the population
      - Called steady-state GA
      - Steady, but fast
- Who to replace?
  - Inferior solutions? → Replacing the bottom K solutions with the new ones
    - Fast convergence
  - Parent? → Replacing the worse parent solution with the new one
    - Bigger diversity

Setting up and execute

Fixed number of iterations OR stop when no more diversities OR stop when reaching a plateau OR Time constraints

```
class GeneticAlgorithm:

    def performEvolution(self, numIterations, numOffstrings):

        population = self.createInitialPopulation()

        for itr in range(numIterations):

            offstring = {}

            for itr2 in range(numOffstrings):
                p1, p2 = self.selectParents()
                offstring[itr2] = self.crossoverParents(p1, p2)
                offstring[itr2] = self.mutation(offstring[itr2])

            self.substitutePopulation(population, offstring)

        mostFittest = self.findBestSolution(population)

        return mostFittest
```

Uniform random draw or shuffling: keep the diversity in the initial population

Large population: more diversity, slower convergence
Small population: opposite pros and cons
Usually 20~50

Further Reading

- 유전알고리즘, 문병로
  - pp. 1-68
- An introduction to genetic algorithm, Melanie Mitchell
  - pp. 1-34, 155-180