COMP815 Nature Inspired Computing

GA for Other Types of Optimization Problems

Example of Practical Optimization Problem

A butter production company wants to optimize the use of the machineries in its daily production of butter. Two types of butter are made – sweet and raw. One kilogram of sweet butter gives the company a profit of $10 and one of raw a profit of $15. Two machines are used in the production: a pasteurization machine and a whipping machine. The daily use time of the pasteurization machine is 3.5 hours and 6 hours for the whipping machine. The processing times (in minutes) for 1kg of butter are given below:

Machine	Sweet butter	Raw butter
Pasteurization	3	3
Whipping	3	6

Let's first refer to the General Problem Format!

Mathematical Description

Objective function

Minimize $f(\boldsymbol{x})$

Parameters

$$\boldsymbol{x} = [x_1, x_2, \dots, x_N]$$

Subject to $\boldsymbol{x} \in M$

Feasible Set

Example of Practical Optimization Problem

One kilogram of sweet butter gives the company a profit of $10 and one of raw a profit of $15.

The daily use time of the pasteurization machine is 3.5 hours (210 min) and 6 hours (480 min) for the whipping machine.

The processing times (in minutes) for 1kg of butter are given below:

Machine	Sweet butter	Raw butter
Pasteurization	3	3
Whipping	3	6

Objective function	*maximum the profit from two butters*	$10∗sweet(kgs) + $15∗raw(kgs)
Parameters		
Feasible set		

Example of Practical Optimization Problem

One kilogram of sweet butter gives the company a profit of \$10 and one of raw a profit of \$15.

The daily use time of the pasteurization machine is 3.5 hours (210 min) and 6 hours (480 min) for the whipping machine.

The processing times (in minutes) for 1kg of butter are given below:

Machine	Sweet butter	Raw butter
Pasteurization	3	3
Whipping	3	6

Objective function	*maximum the profit from two butters*	$f(x) = 10 * x_1 + 15 * x_2$
Parameters	Sweet(kgs), raw(kgs)	x_1, x_2
Feasible set	Obviously	$x_1 \geq 0, x_2 \geq 0$
	daily use time of two machines: pasteurization and whipping	P: $3 * x_1 + 3 * x_2 <= 210$ W: $3 * x_1 + 6 * x_2 \leq 480$

Example of Practical Optimization Problem

One kilogram of sweet butter gives the company a profit of $10 and one of raw a profit of $15.

The daily use time of the pasteurization machine is 3.5 hours (210 min) and 6 hours (480 min) for the whipping machine.

The processing times (in minutes) for 1kg of butter are given below:

Machine	Sweet butter	Raw butter
Pasteurization	3	3
Whipping	3	6

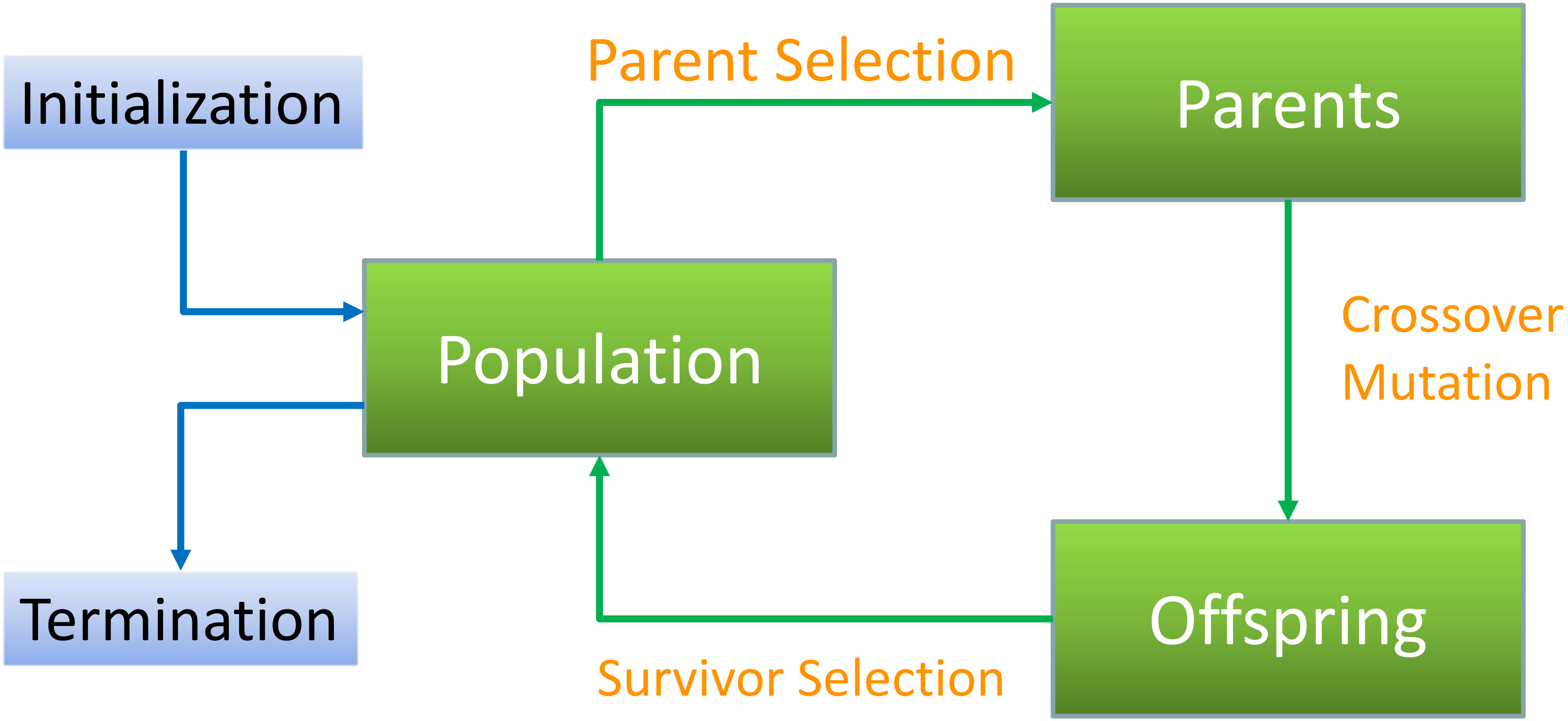
Objective function	*maximum the profit from two butters*	$f(x) = 10 * x_1 + 15 * x_2$
Parameters	Sweet(kgs), raw(kgs)	x_1, x_2
Feasible set	Obviously	$x_1 \geq 0, x_2 \geq 0$
	daily use time of two machines: pasteurization and whipping	P: $3 * x_1 + 3 * x_2 <= 210$ W: $3 * x_1 + 6 * x_2 \leq 480$

Maximize $f(x) = 10 * x_1 + 15 * x_2$

Subject to $x_1 \geq 0, \quad x_2 \geq 0,$

$3 * x_1 + 3 * x_2 \leq 210, \qquad 3 * x_1 + 6 * x_2 \leq 480$

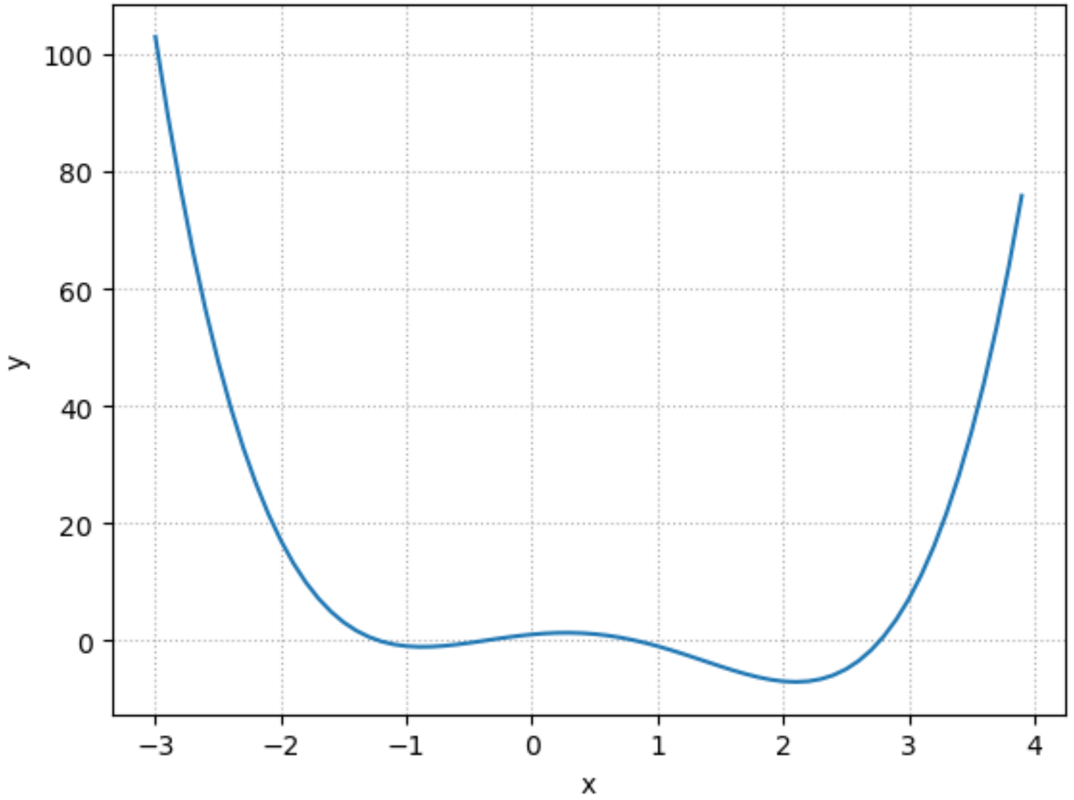
Last Lecturer Review



Last Lecturer Review

min $f(x) = x^4 - 2x^3 - 3x^2 + 2x + 1$

for $-3 \leq x \leq 4$



Chromosome Encoding

Each candidate solution encodes a value of $-3 \leq x \leq 4$

Each candidate solution is represented by a *chromosome*

n binary digits: $\{0,1\}^n$

Let *n* = 10

x	chromosome
-3	00 0000 0000
0	01 1011 0110
4	11 1111 1111

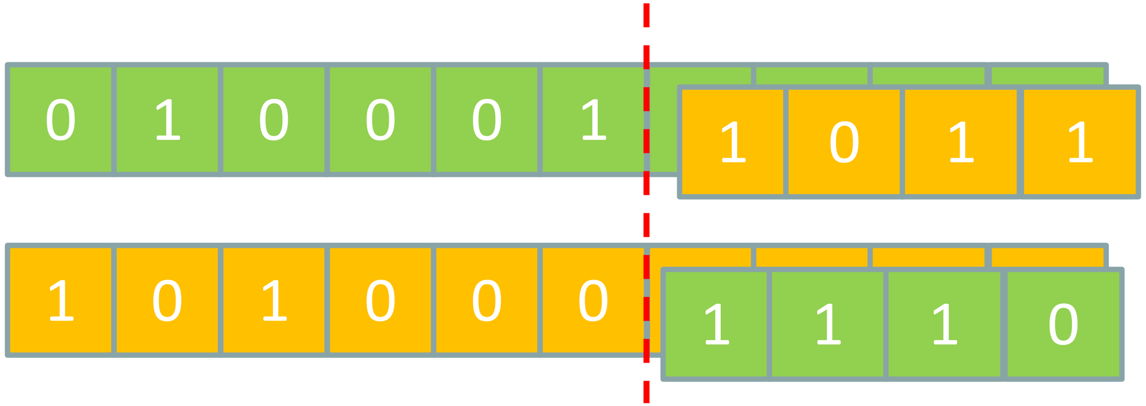
$\Longleftrightarrow$

n=10 *a* = -3 *b* = 4

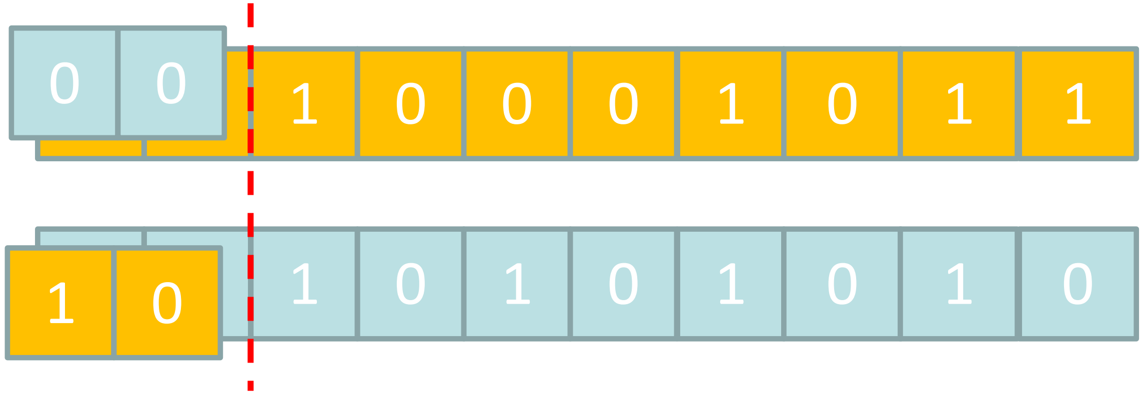
chromosome	x
00 0000 0000	-3
01 1011 0110	-0.003
11 1111 1111	4

Crossover

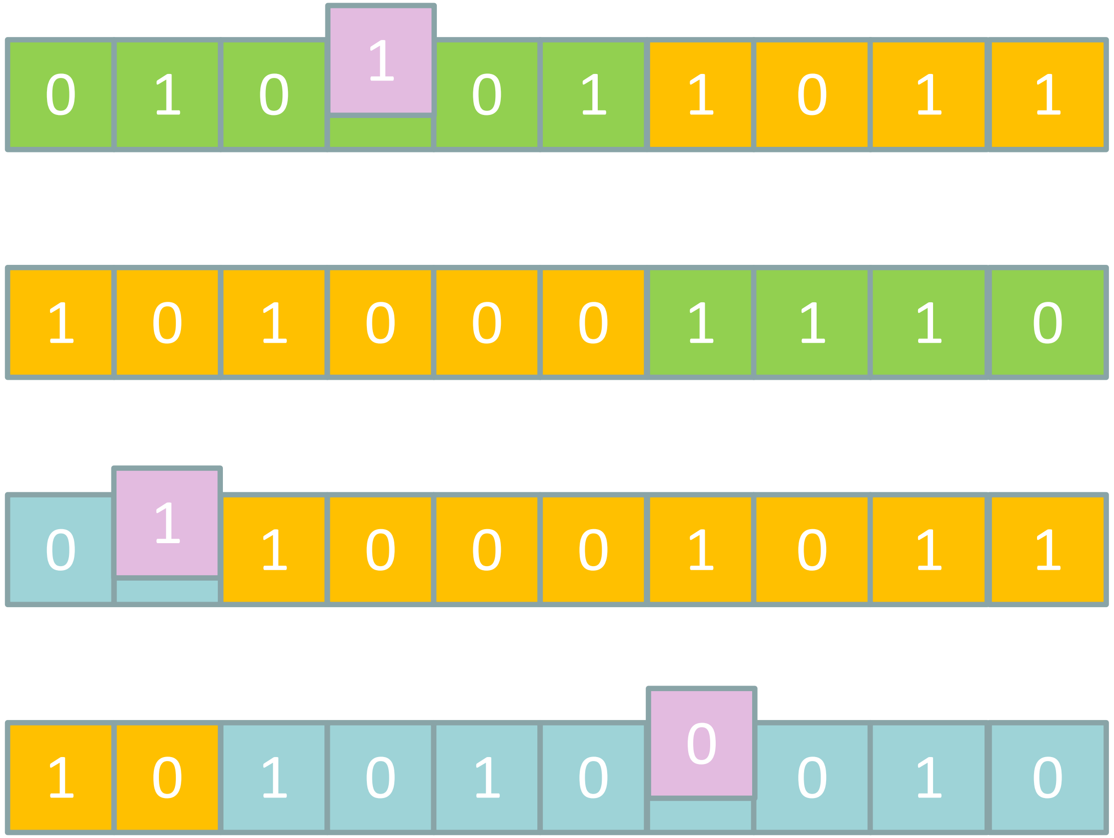
Parents chosen



Single-point crossover
Crossover point randomly selected

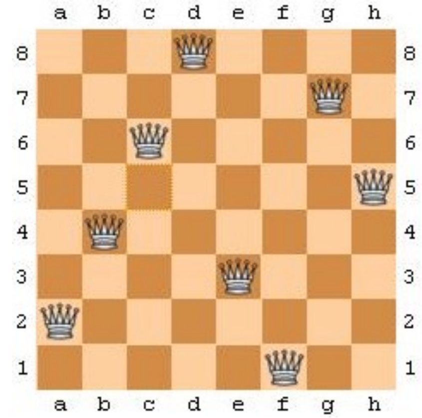


Single-bit Mutation

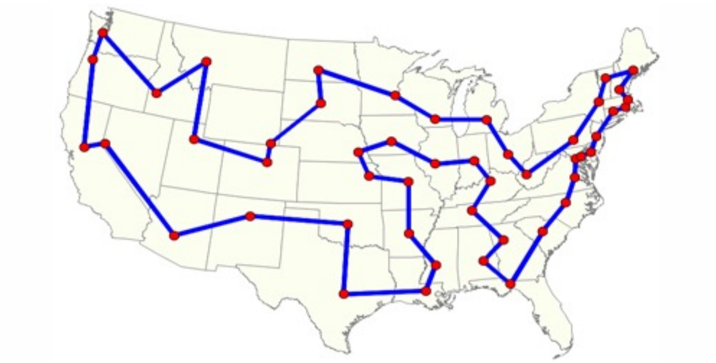


More Problems

- N-Queen, 8-Queen



- Travelling Salesman Problem

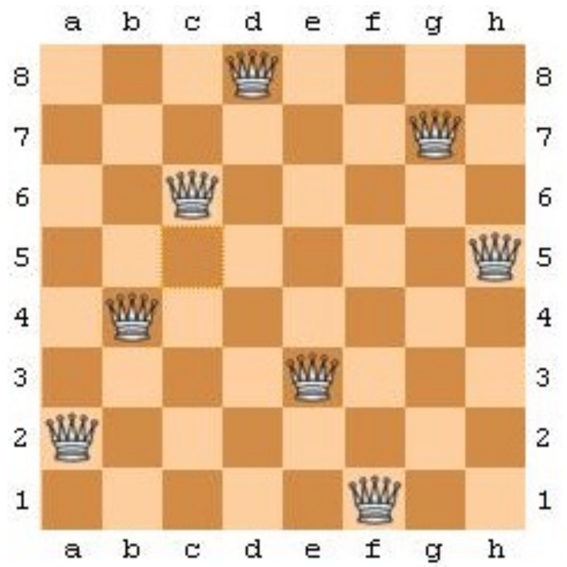


- New Strategies for
  - Chromosome
  - Crossover
  - Mutation

N-queens Problem

- A constraint satisfaction problem
- Place *N* queens on an *N*×*N* chessboard so that no two of them can check each other

Design a GA for the 8-queens problem



A solution of the 8-queens problem

Chromosome Representation

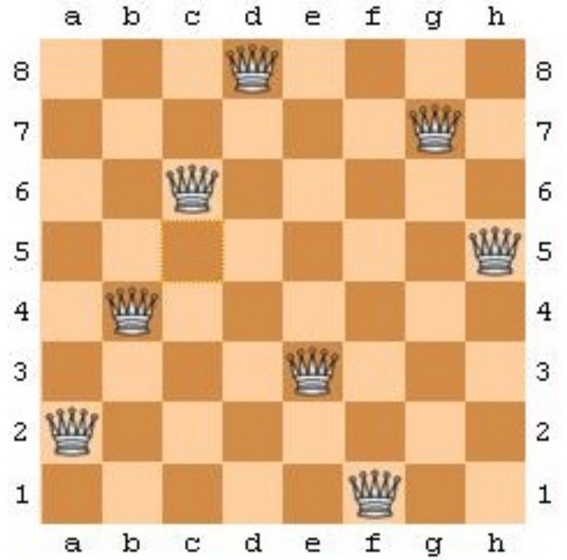
Could use a matrix (2D) of binary genes

Since we know that no two queens could be in the same row and the same column, we can use a permutation representation

Example:

2	4	6	8	3	1	7	5

A permutation of integers 1 to 8



Mutation Operator

Need to remain a permutation after mutation

Swap mutation:

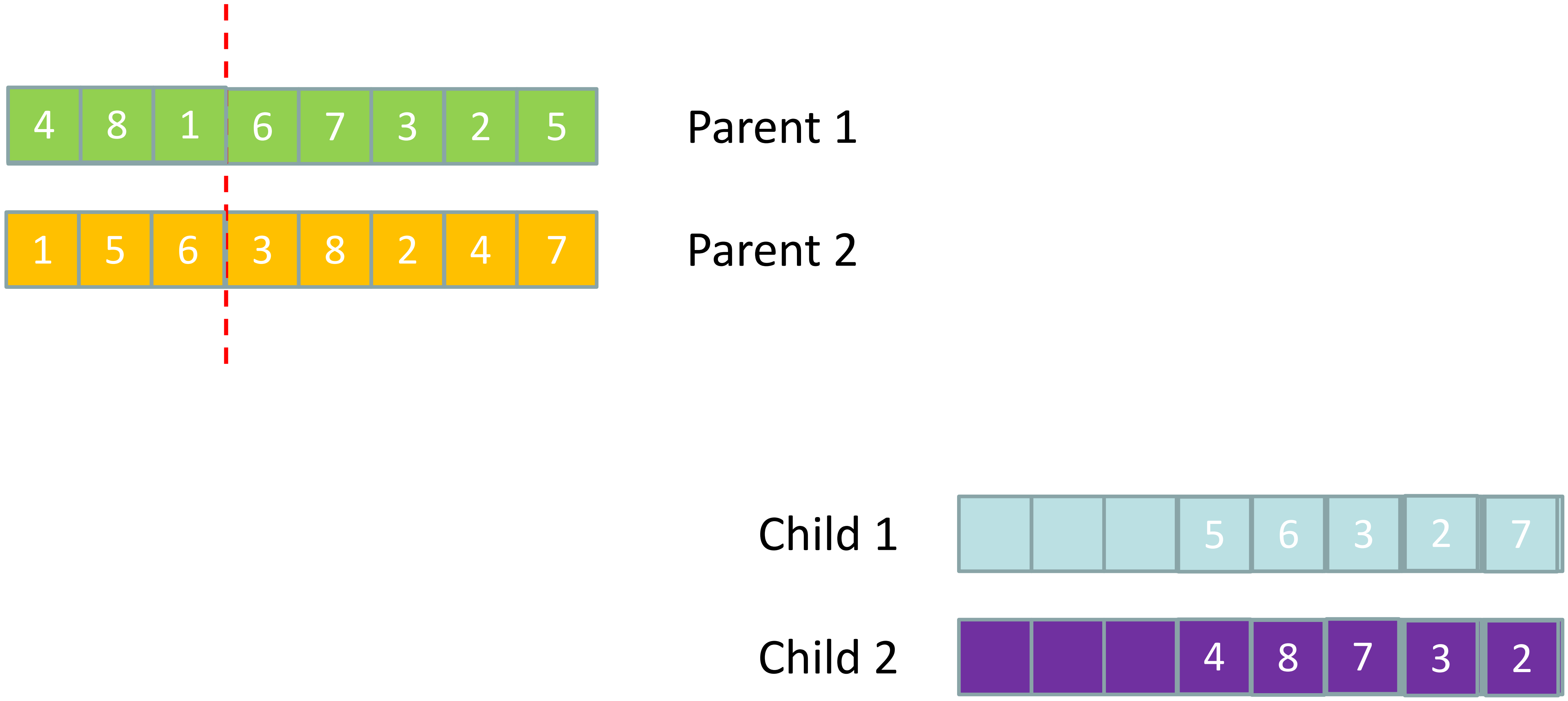
1. Randomly select two genes
2. Swap their positions

1	2	3	4	5	6	7	8

Crossover Operator

Cut-and-Crossfill

- Select a random crossover point
- Cut the parents into two segments at this position
- Copy the first segment of parent 1 into child 1 and the first segment of parent 2 into child 2
- Scan parent 2 from left to right and fill the second segment of child 1 with values from parent 2, skipping those that the first segment already contains
- Do the same for parent 1 and child 2


4 8 1 6 7 3 2 5
Parent 1
1 5 6 3 8 2 4 7
Parent 2
Child 1
5 6 3 2 7
Child 2
4 8 7 3 2

Parent Selection

- Let there be *n* individuals in the first generation
- In each cycle, select 2 parents to produce 2 children
  - Randomly select 5 individuals
  - Choose the best 2 as parents

Replacement Strategy

- Combine the population and the two offsprings
  - Total $n + 2$ individuals
- Rank their fitness
- Remove the worst two

Fitness

$q(x)$ = the number of queen pair violations by x

$$fitness = \frac{1}{q(x)}$$

Can be zero

$$fitness = \frac{1}{q(x) + \epsilon}$$

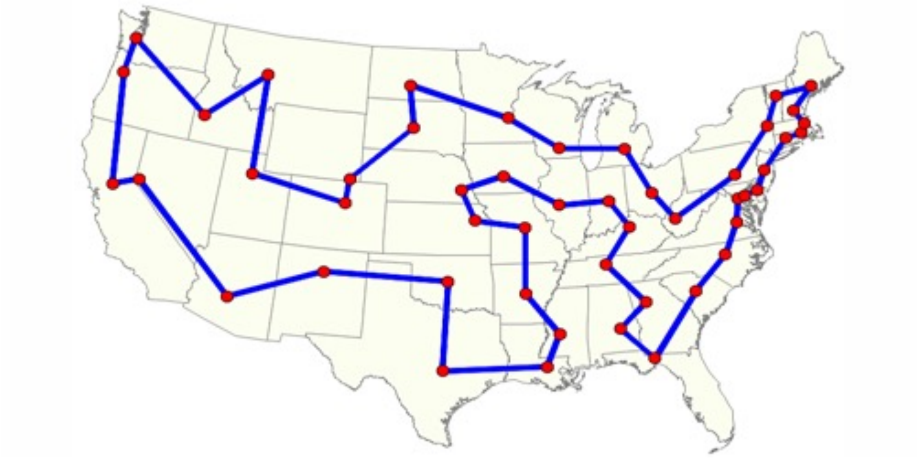
A small value

Travelling Salesman Problem

- Given N cities and the distances between each pair of them, find the shortest path that goes through each city once and returns to the starting city
- The number of possible paths that goes through all the cities is

$$\frac{(N-1)!}{2}$$

Why?



Travelling Salesman Problem

- Given *N* cities and the distances between each pair of them, find the shortest path that goes through each city once and returns to the starting city
- The number of possible paths that goes through all the cities is

Four cities: A B C D

1. All Permutations: N!
2. Circular

   A-B-C-D-**A** → **B**-C-D-A-**B** Fix any starting: (N-1)!
3. Reversal

   A-B-C-D-**A** → **A**-D-C-B-**A** (N-1)!/2

Chromosome Design

Cities can be numbered *1, 2, …, N*

Important Criteria for Design:

- A valid tour must consist of all cities
- Each city should only appear once, except the starting city

Permutation representation could be used

Mutation

Swap mutation could be used

Other possible mutation operations:

Insertion

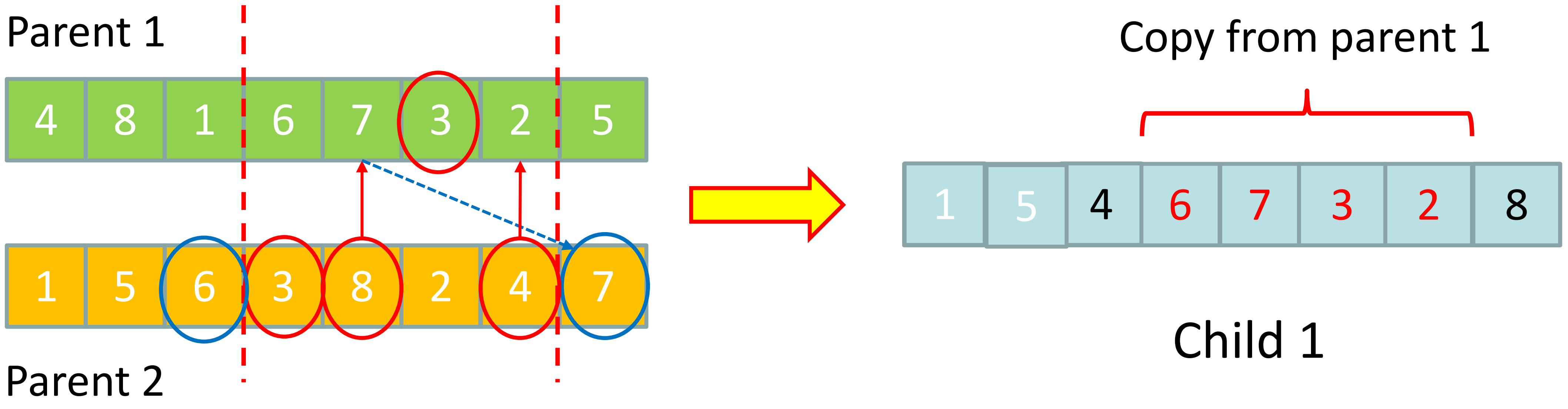
2	4	6	8	3	1	7	5	→	2	4	3	6	8	1	7	5

Inversion

2	4	6	8	3	1	7	5	→	2	3	8	6	4	1	7	5

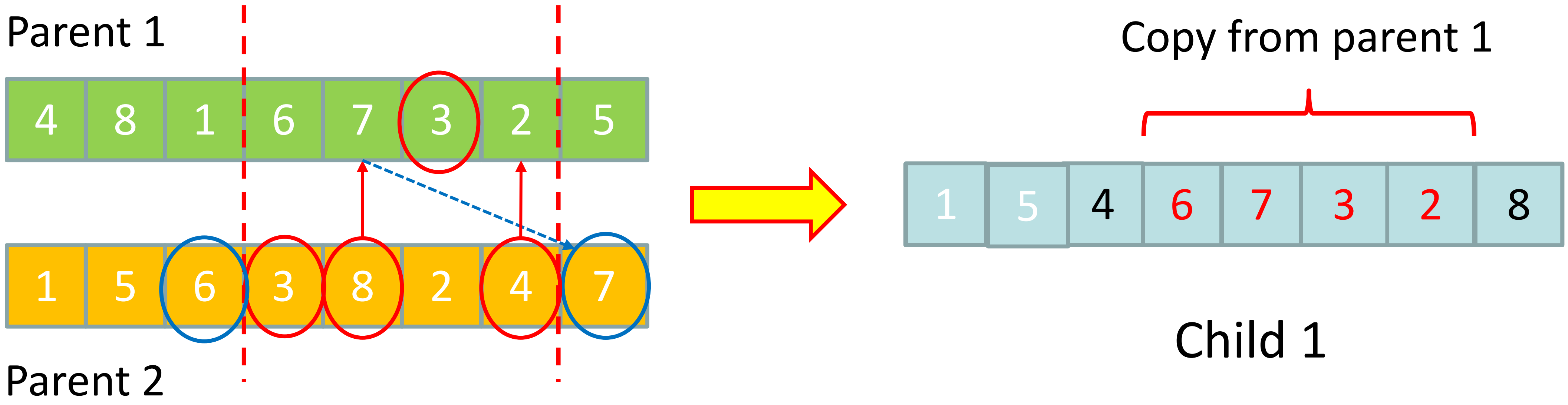
Crossover

Partially Mapped Crossover (PMX)



Crossover

Partially Mapped Crossover (PMX)

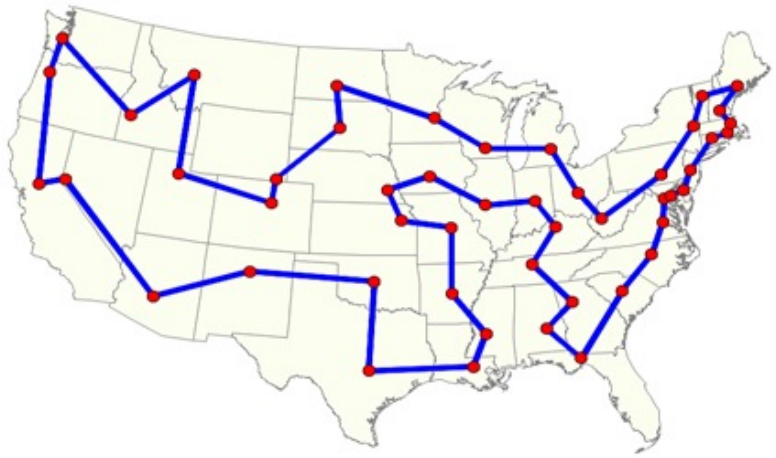


Fitness of TSP

Measure if a path is good or not

Total distance:

$$\sum d_{ij}$$



Parameter Tuning

- Difficulties:
  - Parameters interact:
    Population Size, #Generations, Crossover Rate, Mutation rate, etc
  - Trying different combinations is time consuming
- Good tuning algorithms proposed around 2005:
  - SPO
  - F-race
  - REVAC
  - Meta-GA

Still not widely adopted

Approach

- Treat the design of an EA as a search problem
- The tuning method is a search algorithm (the set of parameters is the vector of values to be searched)

- Tuning algorithms can provide information about an EA:
  - Robustness
  - Distribution of solution quality
  - Sensitivity

Algorithm Quality

- Standard performance metrics
  - Mean best fitness (MBF)
  - Average number of evaluations to a solution (AES)
  - Success rate (SR)

Generate-and-Test

- Robustness
  - Of a set of EA parameters to variations of problem instances
  - Of solution quality to variations of parameters for a particular problem instance

Dealing with Constrained Optimization

- Make use of the encoding method, e.g. *N*-queens problem

- Apply GA as normal and assign zero fitness to any individual that violates the constraints
  - Inefficient search for smaller problems
  - With highly constrained problems:
    - Can produce many illegal individuals, wasting effort
    - Cause over-rapid convergence – too few legal individuals generated

Dealing with Constrained Optimization

- Three approaches
  - Use of penalty functions
  - Use of repair operators
  - Creation of tailored diversity generation operations

Penalty Functions

Altered fitness function

Original fitness function

penalty function

$$\max \quad f^*(x) = f(x) - pen(x)$$

Weight of constraint *i*

$$pen(x) = \sum_{i=1}^{m} w_i b_i$$

$$b_i = \begin{cases} 1, & \text{if } x \text{ violates constraint } i \\ 0, & \text{otherwise} \end{cases}$$

Works for relatively few constraints

Repair Operators

Repair infeasible solutions
by moving them back into the feasible region

Tricky to design

Tailored Diversity-Generation Operators

Crossover and mutation operators are agnostic to the problem domain

Crossover of two legal parents may produce an illegal child

Mutation of a legal individual may result in an illegal one

Design problem-specific crossover and mutation operators

Summary

- N-Queen 8-Queen Problem
- Travelling Salesman Problem
- Parameter Tuning for Genetic Algorithm