

Computational Intelligence

Unit # 2

Acknowledgement

- The slides of this lecture have been taken from the lecture slides of “CSE659 – Computational Intelligence” by Dr. Sajjad Haider.

Evolutionary Algorithm

(Source: Wikipedia)

Evolution (also known as biological or organic evolution) is the change over time in one or more inherited traits found in populations of organisms.

In artificial intelligence, an evolutionary algorithm (EA) is a subset of evolutionary computation, a generic population-based meta-heuristic optimization algorithm.

- An EA uses some mechanisms inspired by biological evolution: reproduction, mutation, recombination, and selection.
- Candidate solutions to the optimization problem play the role of individuals in a population, and the fitness function determines the environment within which the solutions "live".
- Evolution of the population then takes place after the repeated application of the above operators.



History

- Several efforts were started in parallel during 1960s
 - Evolutionary Strategies (Berlin Technical University)
 - Genetic Algorithms (University of Michigan)
 - Evolutionary Programming (UCLA)
- During 1990s the above communities agreed to the term “**Evolutionary Algorithms**”.

Biology

- A gene is a sequence of DNA bases that code for a trait, e.g., eye color, hair color etc.
- An allele is a value of a trait. The eye color gene could have a blue allele or a hazel allele in different people.
- **Definition:** Evolution is the variation of allele frequencies in populations over time.

Theory of Evolution

- You do not need to accept the theory of evolution in biology to do evolutionary computation.
- Evolutionary computation uses the ideas in the theory of evolution, asserting nothing about their validity in biology.

Variation and Selection in Biology

- There are two opposing forces that drive evolution: variation and selection.
- Variation is the process that produces new alleles.
- Selection is the process whereby some alleles survive and others do not.
- Variation builds up genetic diversity; selection reduces it.

Survival of the Fittest

- “evolution is the result of survival of the fittest” is a pretty good description of many evolutionary computation systems.
- When we use evolutionary computation to solve a problem, we operate on a collection (population) of data structures (creatures).
- These creatures will have explicitly computed fitnesses used to decide which creatures will be partially or completely copied by the computer (have offspring)

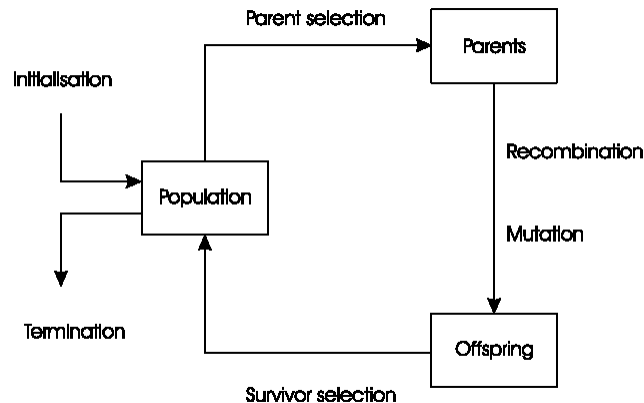
Exploitation vs. Exploration

- Any efficient optimization algorithm must use two techniques to find a global maximum:
 - Exploration: to investigate new and unknown areas in the search space
 - Exploitation: to make use of knowledge found at points previously visited to help find better points.
 - These two requirements are contradictory, and a good search algorithm must find a tradeoff between the two.

Basic Idea of EAs

- Generate a population of structures
- Repeat
 - Test the structures for quality
 - Select structures to reproduce
 - Produce new variations of selected structures
 - Replace old structures with new ones
- Until Satisfied

General Scheme of EAs



Artificial Intelligence Lab - IBA, Karachi

Fall 2014

11

A Typical Evolutionary Algorithm Cycle

- **Step 1:** Initialize the population randomly or with potentially good **solutions**.
- **Step 2:** Compute the **fitness** of each individual in the population.
- **Step 3:** Select parents using a **selection procedure**.
- **Step 4:** Create offspring by **crossover** and **mutation** operators.
- **Step 5:** Compute the **fitness** of the new offspring.
- **Step 6:** Select members of population to die using a **selection procedure**.
- **Step 7:** Go to Step 2 until termination criteria are met.

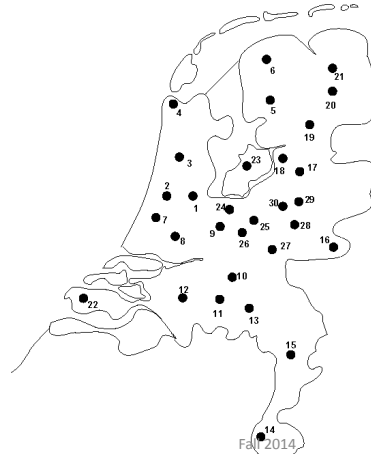
Artificial Intelligence Lab - IBA,
Karachi

Fall 2014

12

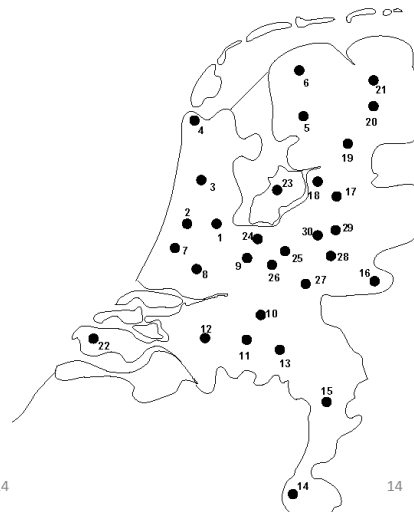
Traveling Sales Person Problem

- Given a number of cities and the costs of traveling from one city to any other city, what is the cheapest round-trip route that visits each city exactly once and then returns to the starting city?



Permutation Representation: TSP

- Problem:
 - Given n cities
 - Find a complete tour with minimal length
- Encoding:
 - Label the cities $1, 2, \dots, n$
 - One complete tour is one permutation (e.g. for $n=4$ $[1,2,3,4]$, $[3,4,2,1]$ are OK)
- Search space is BIG:
 - for 30 cities there are $30! \approx 10^{32}$ possible tours



TSP: Nearest Neighbor

	A	B	C	D	E	F	G	H
A	0	8	3	1	4	9	3	6
B	8	0	5	10	11	4	3	6
C	3	5	0	8	7	1	5	12
D	1	10	8	0	9	11	6	4
E	4	11	7	9	0	5	17	3
F	9	4	1	11	5	0	4	1
G	3	3	5	6	17	4	0	7
H	6	6	12	4	3	1	7	0

- Start with A: A – D – H – F – C – B – G – E Cost?
- Start with E: E – H – F – C – A – D – B – G Cost?
- Start with G: G – B – F – H – E – A – D – C Cost?

Fall 2014

Initialize The Population

- Suppose the population size is 6

Candidate Solutions
A C B F H D E G
H B G E A C D F
A H G C B D F E
E G B C D H F A
F H A D C B E G
C D B A H E G F

Compute Fitness

Candidate Solutions	Fitness
A C B F H D E G	43
H B G E A C D F	52
A H G C B D F E	49
E G B C D H F A	47
F H A D C B E G	49
C D B A H E G F	56

	A	B	C	D	E	F	G	H
A	0	8	3	1	4	9	3	6
B	8	0	5	10	11	4	3	6
C	3	5	0	8	7	1	5	12
D	1	10	8	0	9	11	6	4
E	4	11	7	9	0	5	17	3
F	9	4	1	11	5	0	4	1
G	3	3	5	6	17	4	0	7
H	6	6	12	4	3	1	7	0

Components of Evolutionary Algorithms

- Representation (definition of individuals)
- Evaluation function (or fitness function)
- Population
- Parent selection mechanism
- Variation operators, recombination and mutation
- Survivor selection mechanism (replacement)

Representation

- The first step in defining an EA is to link the “real world” to the “EA world”.
- Objects forming possible solution within the original problem context are referred to as **phenotypes**, while their encoding in the EA are called **genotypes**.

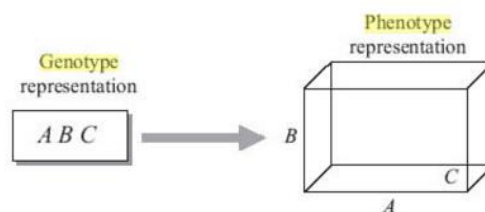
Representation

Within the EC literature many synonyms can be found:

Candidate solution and **individuals** denote points in the **phenotype space**.

Chromosome and **individuals** denote points in the **genotype space**.

A placeholder is commonly called a variable, a **locus**, or – in a biology-oriented terminology – a **gene**. An object in such a place can be called a value or an **allele**.



Representation Example

- An individual is typically described as a fixed length vector of L features which are chosen presumably because of their (potential) relevance to estimating an individual's fitness. For example,
 - <hair color, eye color, skin color, height, weight>

Evaluation Function

- The role of the evaluation function is to represent the requirements the population should adopt to.
- Technically, it is a function or procedure that assigns a quality measure to genotypes.
- The evaluation function is commonly called the **fitness function** in EC. This might cause a counterintuitive terminology if the original problem requires minimization, because the term fitness is usually associated with maximization.

Parent Selection Mechanism

- The role of parent selection is to distinguish among individuals based on their quality, and, in particular, to allow the better individuals to become parents of the next generation.
- An individual is a **parent** if it has been selected to undergo variation in order to create offspring.
- In EC, parent selection is typically probabilistic.

Variation Operators

- The role of **variation operators** is to create new individuals from old ones.
- In the corresponding phenotype space this amounts to generating new candidate solutions.
- Variation operators in EC are divided into two types:
 - Mutation
 - Crossover

Crossover

- A binary variation operators is called **recombination** or **crossover**.
- This operator merges information from two parent genotypes into one or two offspring genotypes.
- Like mutation, recombination is a stochastic operator: the choices of what parts of each parent are combined and how this is done, depend on random drawings.
- The principle behind recombination is simple – *by mating two individuals with different but desirable features, we can produce an offspring that combines both of those features.*

Mutation

- Mutation is a unary variation operator.
- It is applied to one genotype and delivers a (slightly) modified mutant, the **child** or **offspring**.
- A mutation operator is always stochastic: its output – the child – depends on the outcomes of a series of random choices.

Mutation

- Mutations of data structures can be “good” or “bad.”
- A good mutation is one that increases the fitness of a data structure.
- A bad mutation is one that reduces the fitness of a data structure.

Crossover OR Mutation?

- **Exploration**: Discovering promising areas in the search space, i.e. gaining information on the problem
- **Exploitation**: Optimizing within a promising area, i.e. using information
- There is co-operation AND competition between them
- Crossover is explorative, it makes a big jump to an area somewhere “in between” two (parent) areas
- Mutation is exploitative, it creates random small diversions, thereby staying near (in the area of) the parent
- Only crossover can combine information from two parents
- Only mutation can introduce new information (alleles)

Survival Selection Mechanism

- The role of survival selection (**replacement**) is to distinguish among individuals based on their quality.
- It is similar to parent selection, but it is used in a different stage of the evolutionary cycle.
- In contrast to parent selection, which is typically stochastic, survivor selection is often deterministic.

Initialization and Termination

- Initialization is kept simple in most EA applications, the first population is seeded by randomly generated individuals.
- Problem specific heuristics can also be used in this step to create an initial population with higher fitness.
- The following options are used for **termination**:
 - The maximally allowed CPU time elapses.
 - The fitness improvement remains under a threshold value for a given period of time.
 - The population diversity drops under a given threshold.

Order 1 Crossover

- Idea is to preserve relative order that elements occur
- Informal procedure:
 - Choose an arbitrary part from the first parent
 - Copy this part to the first child
 - Copy the numbers that are not in the first part, to the first child:
 - starting right from cut point of the copied part,
 - using the **order** of the second parent
 - and wrapping around at the end
 - Analogous for the second child, with parent roles reversed

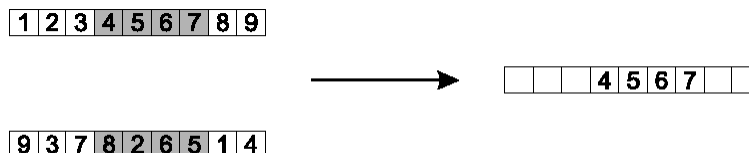
Artificial Intelligence Lab - IBA, Karachi

Fall 2014

31

Order 1 Crossover Example

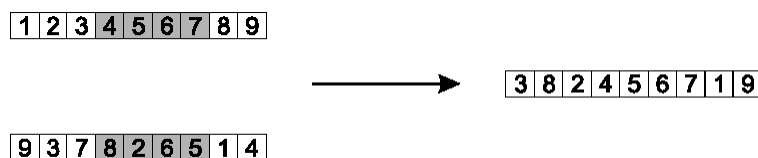
- Copy randomly selected set from first parent



Parent 2:

9	3	7	8	2	6	5	1	4
---	---	---	---	---	---	---	---	---

- Copy rest from second parent in order 1,9,3,8,2



Artificial Intelligence Lab - IBA, Karachi

Fall 2014

32

Crossover Example

- The following parents have been selected during parent selection:

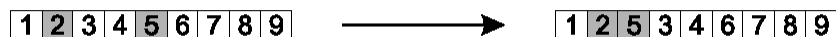
Parent 1	A	C	B	F	H	D	E	G	43
Parent 2	F	H	A	D	C	B	E	G	49

- Let's do crossover to produce two offspring. Suppose the crossover points are 3 and 5

Offspring 1	D	C	B	F	H	E	G	A
Offspring 2	F	H	A	D	C	E	G	B

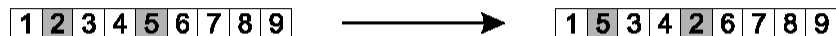
Insert Mutation for Permutations

- Pick two allele values at random
- Move the second to follow the first, shifting the rest along to accommodate
- Note that this preserves most of the order and the adjacency information



Swap Mutation for Permutations

- Pick two alleles at random and swap their positions
- Preserves most of adjacency information (4 links broken), disrupts order more



Mutation Example

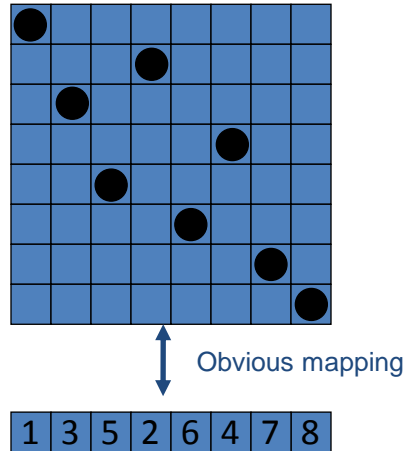
- Perform swap mutation on the two offspring produced in the previous slide.

Offspring 1	D C B F H E G A
Offspring 2	F H A D C E G B

- For Offspring 1, swap 2nd and 4th gene.
- For Offspring 2, swap 2nd and 6th gene.

Offspring 1	D F B C H E G A	55
Offspring 2	F E A D C H G B	40

The 8-Queen Problem: Representation



Artificial Intelligence Lab - IBA, Karachi

Fall 2014

37

The 8-Queen Problem: Fitness Evaluation

- Penalty of one queen:
the number of queens she can check.
- Penalty of a configuration:
the sum of the penalties of all queens.
- Note: penalty is to be minimized
- Fitness of a configuration:
inverse penalty to be maximized

Artificial Intelligence Lab - IBA, Karachi

Fall 2014

38

The 8-Queen Problem: Mutation

- Small variation in one permutation, e.g.:
 - swapping values of two randomly chosen positions



The 8-Queens Problem: Crossover

- Combining two permutations into two new permutations:
 - choose random crossover point
 - copy first parts into children
 - create second part by inserting values from other parent:
 - in the order they appear there
 - beginning after crossover point
 - skipping values already in child

