Evolutionary Computation

Evolutionary Computing

Evolutionary Computing is a research area within **computer science**. As the name suggests, it is inspired by *Darwin's theory of natural evolution*.
In an environment with limited resources, the **fittest individuals survive**. Also, they have more chances to have **offspring**.

Evolutive Algorithms Terminology

Individual	Represents a solution
Phenotype	The representation of an individual
Genotype	The representation used for solving the problem
Gene	A simple value from the genotype
Fitness	A numeric value that represents the quality of the solution
Population	It is a group of individuals that recombine and mutate their properties. The initial population is randomly created
Selection of parents:	The parents must be selected based on their fitness
Crossover (Reproduction)	The parents inherit their characteristics to their offspring.
Mutation	Individuals modify their characteristics or behavior to improve themselves
Survival	The fittness. individuals survive and can live more time

Evolutive Algorithms Terminology (cont)

Termination Condition	If you know the value of good fitness, the algorithm can stop when you find an individual with good fitness

Pseudocode

```
BEGIN
          INI TIALISE population
with random candidate solutions;
          EVA LUATE each
candidate;
          REPEAT UNTIL (
TERMIN ATION CONDITION is
satisfied ) DO
                   1 SELECT
parents;
                   2 RECOMBINE
pairs of parents;
                   3 MUTATE the
resulting offspring;
                   4 EVALUATE
new candid ates;
                   5 SELECT
indivi duals for the next
genera tion;
         OD
END
```

Evolutionary Programming (EP)

Evolutionary Programming (EP)

In the classical example of EP, predictors were evolved in the form of finite state machines.
A *finite state machine* (FSM) is a transducer that can be stimulated by a finite alphabet of input symbols and can respond in a finite alphabet of output symbols.
It consists of a number of states S and a number of state transitions.
The state transitions define the working of the FSM: depending on the current state and the current input symbol, they define an output symbol and the next state to go to.

Mutation operators to generate new FSMs

The idea was evolving finite state machines (FSMs). There are five generally usable mutation operators to generate new FSMs:

- Changing an output symbol
- Changing a state transition
- Adding a state
- Deleting a state
- Changing the initial state

Evolutionary Programming Terminology

Representation	Real-valued vectors
Parent selection	Deterministic (each parent creates one offspring via mutation)
Recombination	None
Mutation	Gaussian perturbation
Survivor selection	$(\mu + \mu)$
Specialty	Self-adaptation of mutation step sizes

Particle Swarm Optimization

Particle Swarm Optimization

Inspired by the movement of a flock of birds when searching for food.

Particle Representation

Each particle i represents a solution for the problem. In the time t, it has a position $xi(t) \in \mathbb{R}d$ and a velocity $vi \in \mathbb{R}d$.

Position and Velocity Update

The positions and velocities are updated following the next equations, where *Pbest i* is the best position where the particle *i* has been, *Gbest* is the best location founded until the moment, *r*1 *and* □□2 are random numbers between 0 and 1, and *w*, *c*1, *and c*2 are hyper parameters. Those last values can be initialized at 0.9 and gradually reducing it until 0.1.

Genetic Algorithms (GA)

Genetic Algorithms (GA)

John Holland proposed genetic Algorithms in the 1970s. Initially, they were called *"Reproductive Plans."* These algorithms are maybe the most famous of the evolutive algorithms family.
The inspiration comes from the **DNA structure**, which is people's genetic code. All the information is stored in chromosomes that have a lot of genes. Holland's proposal consists of representing the solutions by binary arrays.

Selection of Parents

Roulette selection	You can imagine a roulette where each section is assigned to an individual. If we have 10 individuals, the roulette is divided into 10 sections. The section size is proportioned to the individual's fitness.
Tournament selection	It consists of randomly choosing k individuals and selecting the fittest one. k represents the tournament size.

Reproduction (crossover or recombination)

1 point crossover	This technique divides the parents into two sections randomly choosing a crossover point.
N point crossover	The parents are divided into several sections.
Uniform crossover	For each gene, it copies the gene of the first or the second parent randomly.

Mutation

Bitwise mutation	Consists of randomly selecting one or several genes and changing their values.
Random resetting	Consists of randomly selecting one or several genes and resets its values.
Uniform mutation	It randomly selects one or several genes and chooses a random value between the minimum and maximum values.
Swap mutation	Consists of randomly selecting two elements and swapping their values.

Differential Evolution

Diferencial Evolution (DE)

It is a robust algorithm for solving *continuous multidimensional optimization problems*. In this algorithm, individuals as seen as *vectors*.
The novelty is the mutation operator, that uses three individuals for mutate another one, and the mutation depends on the distance

Differential Evolution Example

Video: http://youtu.be/BsfJDg0a0Z4

Differential Evolution Terminology

Representation	The individuals are represented as vectors whose entries are the variables values.
Mutation	Mutation is the main operation in Differential Evolution. The new individual *v i* is calculated as follows: *vi* = *xr*1 + *F*(*xr*2 − □□*r*3)
Crossover	For each variable k of *u* □□ , the value is selected randomly between *vi* or □□*i*
Selection	The selection is performed by tournament.

Constraint Handling

Disadvantages of Constrains

In general, the presence of constraints will divide the space of potential solutions into two or more disjoint regions, *the feasible region*, containing those candidate solutions that satisfy the given feasibility condition, and the *infeasible region* containing those that do not.

Penalty Functions

Static	Relies on the ability to specify a distance metric that accurately reflects the difficulty of repairing the solution, which is obviously problem-dependent, and may also vary from constraint to constraint

Penalty Functions (cont)

Dynamic	The fitness function used to evaluate the population is a combination of the distance function for constraint i with a death penalty for all solutions violating constraints

Is a distance metric of the infeasible point to the feasible region

Constrains in EA

The presence of constraints implies that not all possible combinations of variable values represent valid solutions to the problem at hand.

Unfortunately, constraint handling is not straightforward in an EA, because the variation operators are typically “blind" to constraints.

There is no guarantee that even if the parents satisfy some constraints, the offspring will satisfy them as well.

Repair Functions

Takes an infeasible point and generates a feasible solution based on it. In some problems, this technique is relatively simple.

In general, defining a repair function may be as complex as solving the problem itself.

Evolution Strategies (ES)

Evolution Strategies (ES)

The goal is to solve *continuous multidimensional optimization problems*.
The main characteristic is the **self-adaptation of parameters**. It means that some evolutive algorithm parameters change during the execution.
Those parameters are included in the individual representation and **evolve** at the same time that the solution.

Evolution Strategies Terminology

Individual's Representation	The individuals’ solutions are represented as vectors whose inputs are the values of the variables
Mutation	The individual’s position is modified by adding a random number, noise, to each entry
Recombination	In ES there are two recombination variants
Intermediate recombination	The values of the parents are averaged.
Discrete recombination	One of the parent’s values is randomly chosen with equal chance for either parents
Parent selection	Parent selection in ES is completely random, because here the whole population is seen as parent
Survivor Selection	(μ, λ) selection, where only the best μ offspring are selected. (μ + λ) selection, where the best μ individuals (from the union of parents and offspring) are selected
Specialty	Self-adaptation of mutation step sizes

Genetic Programming

Genetic Programming (GP)

Automatically solves problems without requiring the user to know or specify the structure of the solution in advance.
The main idea of GP is to evolve a population of computer programs, where individuals are commonly represented as syntax trees.

Elements of a GP individual

Terminals	Leave nodes in a syntax tree. Variables that can be predefined or randomly generated.
Functions	Internal nodes in a syntax tree. Operations

Genetic Programing Terminology

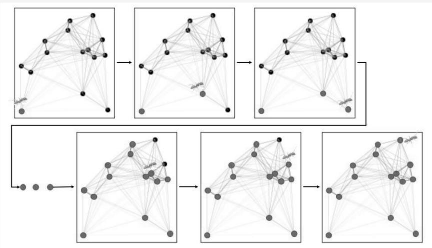
Initial population	1. **Full**, where all the trees are randomly created, and all the leaves have the same depth 2. **Grow**, each node selects an element randomly from either the function set or the terminal set 3. **Ramped half-and-half** where half of the population is created with the full technique and the other half with grow
Selection	Tournament Selection
Crossover	Classic Crossover
Mutation	**Subtree mutation**, randomly selects a mutation point in a tree and substitutes the subtree rooted there with a randomly generated subtree

Ant Colony Optimization (ACO)

Ant Colony Optimization (ACO)

Solves problems of finding paths in graphs. It is inspired by the ants' behavior when searching for food. The ants leave pheromones that allow other ants to follow the path to food. The more ants go for a specific path, the more pheromones.

Example



In this algorithm, an artificial ant must simulate a path starting at a specific point. It moves node by node, choosing based on the pheromones of each path.

Ant Colony Terminology

Cij	Path from the node i to the node j
Tij	Pheromones in the path from the node i to the node j
Nij	Heuristic in the path from the node i to the node j
p	Evaporation rate, between 0 and 1

Steps

First	All the pheromones can be initialized with a small value.
Second	Ants start to move around the graph node by node using the previous equation.
Last	The pheromones must be updated. Ants deposit pheromones to their path proportional to its distance. The pheromones evaporate.