

Revision 1

Xin Yao (x.yao@cs.bham.ac.uk)

CERCIA and Natural Computation Group
School of Computer Science

1. What have we covered in this module
2. Focus on **understanding** and
3. the **use** of knowledge (understanding)

Main Topics in This Module

1. Fundamentals

- Concepts and terminologies
- Representations and Operators

2. Basic Techniques

- Constraint handling and fitness landscapes
- Co-evolution, speciation, fitness sharing and niching
- Multi-objective evolutionary algorithms
- Genetic programming

3. Advanced Techniques

- Evolving learning machines
- Dynamic optimisation
- Evolutionary computation theory

Fundamentals: A Simple Evolutionary Algorithm

Evolutionary algorithms (EAs) are population-based stochastic search algorithms.

1. Generate the initial **population** $P(0)$ at random, and set $i \leftarrow 0$;
2. REPEAT
 - (a) Evaluate the fitness of each individual in $P(i)$;
 - (b) **Select** parents from $P(i)$ based on their fitness in $P(i)$;
 - (c) **Generate** offspring from the parents using *crossover* and *mutation* to form $P(i + 1)$;
 - (d) $i \leftarrow i + 1$;
3. UNTIL halting criteria are satisfied

Fundamentals: Concepts

1. The fundamental principles behind GAs, EP, ES, GP, etc. are **the same**. They can all be described as EAs (Evolutionary Algorithms). They differ in their histories, representations and search operators.
2. Key features of EAs: **population** and **stochasticity**.
3. The fundamental issues in evolutionary computation are **representation** and **search**, which are quite similar to the fundamental issues in classical artificial intelligence and computer science.
4. Evolutionary computation complements classical artificial intelligence and computer science.

Fundamentals: Representations

1. In terms of numerical problems,
 - real-valued representation
 - binary representation: different encoding schemes, Hamming cliff, etc.
2. In terms of discrete problems,
 - lists, sets, matrices, trees, graphs, ...
3. Mixed representation and adaptive representation are possible.

Representation reflects how one formulates and defines a problem. Domain knowledge is often used. Representation defines the search space operated by an EA.

Fundamentals: Search Operators

1. Every representation comes with its most appropriate search operators, including crossover and mutation operators.
2. Crossover (recombination) tries to combine useful information from two or more parents into offspring.
3. Mutation makes random changes (often small) to a single parent.
4. One can invent as many different forms of search operators, whether they are called crossover/mutation or not, as they like, as long as the probability of finding the global optimal solution from the offspring is higher than that from the parent. We can use the generic term of **search operators** or **random variations**.

Fundamentals: Selection

Selection schemes are not representation dependent. Although there are many different selection methods, there is only one single principle: fitter individuals are selected with higher probabilities.

Examples: Search Operators and Selection

1. Crossover for binary strings: k -point crossover, uniform crossover.
2. Crossover for real-valued vectors: global discrete recombination, intermediate recombination
3. Mutation for binary strings: bit-flipping
4. Mutation for real-valued vectors: Gaussian, Cauchy, self-adaptation.
5. Selection: fitness proportional (roulette wheel), ranking, tournament.

Techniques: Constraint Handling

The penalty function approach converts a constrained problem into an unconstrained one by introducing a penalty function into the objective function.

The repair approach maps (repairs) an infeasible solution into a feasible one.

The purist approach rejects all infeasible solutions in search.

The separatist approach considers the objective function and constraints separately.

The hybrid approach mixes two or more different constraint handling techniques.

Need to understand how constraint handling techniques change the fitness landscape of the problem

Techniques: Co-evolution, Speciation and Niching

1. Need to understand what co-evolution is and why it is different from ‘conventional’ evolution: the fitness evaluation is coupled!.
2. There are several concepts and techniques that are closely linked to co-evolution, but **not** equivalent to co-evolution: speciation, niching, fitness sharing. Need to understand how these techniques work.
3. Need to understand how fitness sharing can be used to locate multiple maxima (assuming maximisation) in a single evolutionary run.
4. What is the single most important point in all these (speciation, niching, fitness sharing, etc.)?

Techniques: Multi-objective Evolutionary Algorithms

1. Need to understand the **dominance** concept and definition clearly: *no worse in any objectives and strictly better in at least one.*
2. Need to understand the **Pareto optimality**, **Pareto front**, **non-dominated set**, etc.
3. For multi-objective optimisation, the solution is a **set**. Both **convergence** and **even distribution** are important performance metrics.
4. Need to understand how MOEAs work, at least for the most popular one, so that you can use it to solve a problem.

Techniques: Genetic Programming (GP)

1. Refers to EAs with structured representations, such as trees.
2. Representations and operators.
3. Other considerations, including initialisation, terminal set, non-terminal (functional) set, problem formulation, ...
4. Need to understand **when** to use it and **how** to use it to solve practical problems.
5. GP is often associated with (machine) learning and (data-driven) modelling.

Advanced Techniques: Evolving Learning Machines

1. Evolution is an extremely **general and powerful** tool for designing learning machines.
2. Evolutionary computation has been used in evolving/designing rule-based systems, artificial neural networks, fuzzy logic systems, hidden markov models, gene regulatory networks, finite state machines, and many other learning machines
3. The key issues here include **representation** and **fitness evaluation**.
4. Need to consider this topic in the general framework of machine learning, especially in terms of **generalisation**.

Advanced Techniques: Dynamic Optimisation

Common approaches:

1. Diversity: Hyper-mutations, Random immigrants,
2. Memory: Diploidy, external memory, memory-based diversity
3. Multi-populations: Closely related to speciation

Advanced Techniques: Evolutionary Computation Theory

1. It explains **how**, **when** and **why** EAs work.
2. No Free Lunch (NFL) theorem.
3. Convergence, convergence rate, computational time complexity.
 - Definition of computation time: expected first hitting time
 - Basic steps/approaches to analyse EAs
4. Key differences between EA-hardness and conventional NP-hardness.

Summary

1. Understanding is essential.
2. Needs to see **through** individual techniques.
3. Think carefully how various techniques are related to problem solving (what problems?)
4. Ensure a good understanding of questions before trying to answer them.