**Definitions**

**Heuristic:**

Technique in problem solving which sacrifices accuracy/optimality for speed. Finds a satisfactory solution where an optimal solution is impractical/impossible.

**Metaheuristic:**

A heuristic designed to find, generate, or select a heuristic (partial search algorithm).

These are properties that characterize most metaheuristics:[[2]](https://en.wikipedia.org/wiki/Metaheuristic" \l "cite_note-blum03metaheuristics-2)

* Metaheuristics are strategies that guide the search process.
* The goal is to efficiently explore the search space in order to find near–optimal solutions.
* Techniques which constitute metaheuristic algorithms range from simple local search procedures to complex learning processes.
* Metaheuristic algorithms are approximate and usually non-deterministic.
* Metaheuristics are not problem-specific.

**Variable neighbourhood search:**

Explored distant neighbours of the current solution and moves to a new one if an improvement is found. The purpose is to get from neighbourhood solutions to local optima using repeated local searches.

**Local Search:**

Heuristic for solving computationally hard optimization problems. Local searches move among solutions in the search space by applying local changes until an optimal solution is found or a time is elapsed.

**Random Search:**

Family of optimization methods that do not require the gradient of a model to be optimized, and therefore can be used on models that are not continuous (and therefore not differentiable).

Basically the algorithm chooses a random position in the search space, searches a local area (hypersphere) until solution is found or moves on to the next area (starting at one of the bounds of the local area).

**Computational Complexity Theory:**

Classifies computational problems (collection of processes producing multiple solutions) according to difficulty.

**Complexity Class:**

A set of problems of related resource-based complexity. Complexity classes are concerned with the rate of growth of the requirement in the resources as the input increases.

Resources can be time or storage space.

<https://en.wikipedia.org/wiki/Complexity_class> - for how to categorize

**NP Complexity:**

Nondeterministic polynomial time. A nondeterministic algorithm may exhibit different results/paths to the result on different runs. An algorithm is of polynomial time if its running time is upper bounded by a polynomial expression in the size of the input for the algorithm.

**P Complexity:**

Deterministic polynomial time.

**Reduction:**

Problem A is reducible to problem B if an algorithm for solving problem B efficiently (if it existed) could also be used as a subroutine to solve problem A efficiently.

**NP-Complete**

A complexity class which represents the set of all problems X in NP for which it is possible to reduce any other NP problem Y to X in polynomial time.

**NP-Hard**

A problem X is hard for a complexity class C if every problem in C can be reduced to X; i.e. no problem in C is harder than X.

So, and NP-hard problem is a problem of at LEAST NP complexity.

**P versus NP Problem**

If a problem can be verified quickly, can it also be solved quickly?

<https://en.wikipedia.org/wiki/P_versus_NP_problem>

**Combinatorial Optimization (as opposed to continuous optimization):**

Optimizing discrete number of solutions.

**Continuous Optimization:**

Solutions chosen from a continuous set of real numbers.

**Objective Function:**

Loss/cost functions map events/values onto a real number representing some cost. The objective function seeks to maximize a loss function.

**Swarm Intelligence:**

Collective behaviour of decentralized (resource distributed), self-organized (local interaction of individual constituents) systems. Typically SI consists of a population of simple agents interacting locally with the environment and each other.

**Particle Swarm Optimization**

Global optimization (as opposed to local optimization) dealing with problems with solutions represented as a point or surface in n-dimensional space.

Iteratively tries to improve a candidate solution (member of solution set) with regard to a given measure of quality.

Basic procedure involves having a number of candidate solutions (these are the particles) traverse the solution space. The movement is influenced by local optimum solutions and also global optimums, with the goal being to move the swarm to an optimum solution.

PSO is a metaheuristic; it does not concern optimal solutions, and there is no guarantee any solutions are found.

PSO is a combinatorial optimization problem.

**Multi-swarm Optimization**

Variant of PSO using multiple sub-swarms rather than one swarm. The purpose is to find multiple local optima.

**Ant Colony Optimization**

Class of optimization modelled around ant colonies. Probabilistic technique that is useful for finding paths through graphs. Simulation agents traverse the solution space and record their position and quality of their solutions, so that more ants can find better solutions.

**Artificial Bee Colony (ABC) Algorithm**

Meta-heuristic algorithm simulating foraging of bees. Three phases: employed, onlooker, and scout bees.

Employed bees search food around food sources in their memory. Onlooker bees select good food sources based on information provided by employed bees via a fitness test.

In the scout bee phase, non-beneficial solutions are abandoned by employed bees as they search for new sources.

**Bees Algorithm**

*Exploratory Search:* Information exploration which represents activities carried out by searchers who are uninformed agents.

*Exploitative Search:* Local search which hopes to improve a current best solution

Combines global explorative search with local exploitative search. A small number of scout bees explores the solution space randomly for solutions of high fitness. The rest of the bees search the neighbourhood of the fittest solutions looking for the fitness optimum.

The scouts report back to the forager who are sent to perform local searches in areas of favourable fitness.

In summary, the Bees Algorithm searches concurrently the most promising regions of the solution space, whilst continuously sampling it in search of new favourable regions.

**Evolutionary Algorithm**

Trial and error global optimization methods distinguished by the use of a population of candidate solutions.

Evolutionary algorithms use iterative processes to select a final population by guided random searches usually using parallel processing.

Evolutionary computing techniques usually involve metaheuristic optimization algorithms.

**Genetic Algorithm**

Search heuristic emulating natural selection.

**Direct/Pattern Search Algorithms**

Do not require the gradient to be optimized and is therefore useful for functions that are not differentiable.

E.g. Hill climbing, random walk

**Intelligent Swarm-based Optimization**

*SOA vs Direct Search:* SOAs use a population of solutions for each iteration rather than a single solution, and the outcome is also a population of solutions. If the solution space has a global optimum, SOAs will tend towards it. If there are multiple optimal solutions, SOA will capture all of them in the final result population.

**Bees Algorithm (From Paper)**

**Condition:** There must be some measure of topological distance between the solutions in the solution space.

**Goal:** Locate the global optimum of the search space

**Structure:** Combines neighbourhood search and random search.