# Assignment 04: Optimizing TSP with GA

# Write Short Notes on Different Search Methods

MCE 079 05536 Shyed Shahriar Housaini

## Short Notes on Different Search Methods-

***Local search (optimization):***

In computer science, local search is a heuristic method for solving computationally hard optimization problems. Local search can be used on problems that can be formulated as finding a solution maximizing a criterion among a number of candidate solutions. Local search algorithms move from solution to solution in the space of candidate solutions (the search space) by applying local changes, until a solution deemed optimal is found or a time bound is elapsed.

Local search algorithms are widely applied to numerous hard computational problems, including problems from computer science (particularly artificial intelligence), mathematics, operations research, engineering, and bioinformatics.

***Global search (optimization):***

Global optimization is a branch of applied mathematics and numerical analysis that attempts to find the global minima or maxima of a function or a set of functions on a given set. It is usually described as a minimization problem because the maximization of the real-valued function

$$f(x) := (-1) \cdot g(x)$$

Global optimization is distinguished from local optimization by its focus on finding the minimum or maximum over the given set, as opposed to finding *local* minima or maxima. Finding an arbitrary local minimum is relatively straightforward by using classical *local optimization* methods. Finding the global minimum of a function is far more difficult: analytical methods are frequently not applicable, and the use of numerical solution strategies often leads to very hard challenges.

### Types of search algorithms:

Based on the search problems we can classify the search algorithms into uninformed (Blind search) search and informed search (Heuristic search) algorithms.

### Uninformed/Blind Search:

The uninformed search does not contain any domain knowledge such as closeness, the location of the goal. It operates in a brute-force way as it only includes information about how to traverse the tree and how to identify leaf and goal nodes. Uninformed search applies a way in which

search tree is searched without any information about the search space like initial state operators and test for the goal, so it is also called blind search.It examines each node of the tree until it achieves the goal node.

It can be divided into five main types:

***Breadth First Search:***

Breadth-first search (BFS) is an algorithm for traversing or searching tree or graph data structures. It starts at the tree root (or some arbitrary node of a graph, sometimes referred to as a 'search key'), and explores all of the neighbor nodes at the present depth prior to moving on to the nodes at the next depth level. It is implemented using a queue.

***Uniform Cost Search:***

UCS is different from BFS and DFS because here the costs come into play. In other words, traversing via different edges might not have the same cost. The goal is to find a path where the cumulative sum of costs is the least. Sorting is done in increasing cost of the path to a node. It always expands the least cost node. It is identical to Breadth First search if each transition has the same cost. It explores paths in the increasing order of cost.

Disadvantage − There can be multiple long paths with the cost ≤ C*. Uniform Cost search must explore them all.

***Depth First Search:***

Depth-first search (DFS) is an algorithm for traversing or searching tree or graph data structures. The algorithm starts at the root node (selecting some arbitrary node as the root node in the case of a graph) and explores as far as possible along each branch before backtracking. It uses last in-first-out strategy and hence it is implemented using a stack.

***Iterative Deepening Depth-First Search***

It performs depth-first search to level 1, starts over, executes a complete depth-first search to level 2, and continues in such way till the solution is found. It never creates a node until all lower nodes are generated. It only saves a stack of nodes. The algorithm ends when it finds a solution at depth d. The number of nodes created at depth d is bd and at depth d-1 is bd-1.

***Bidirectional Search:***

It searches forward from initial state and backward from goal state till both meet to identify a common state. The path from initial state is concatenated with the inverse path from the goal state. Each search is done only up to half of the total path.

Informed Search:

Informed search algorithms use domain knowledge. In an informed search, problem information is available which can guide the search. Informed search strategies can find a solution more efficiently than an uninformed search strategy. Informed search is also called a Heuristic search.

A heuristic is a way which might not always be guaranteed for best solutions but guaranteed to find a good solution in reasonable time. Informed search can solve much complex problem which could not be solved in another way. An example of informed search algorithms is a traveling salesman problem.

***Greedy Search:***

In greedy search, we expand the node closest to the goal node. The "closeness" is estimated by a heuristic h(x). Heuristic: A heuristic h is defined as-

h(x) = Estimate of distance of node x from the goal node.

Lower the value of h(x), closer is the node from the goal.

Strategy: Expand the node closest to the goal state, i.e. expand the node with a lower h value.

***A* Tree Search:***

A* Tree Search, or simply known as A* Search, combines the strengths of uniform-cost search and greedy search. In this search, the heuristic is the summation of the cost in UCS, denoted by g(x), and the cost in the greedy search, denoted by h(x). The summed cost is denoted by f(x).

Heuristic: The following points should be noted wrt heuristics in A* search. f(x) = g(x) + h(x)

*[Search Heuristics: In an informed search, a heuristic is a function that estimates how close a state is to the goal state. For example – Manhattan distance, Euclidean distance, etc. (Lesser the distance, closer the goal.)]*

***A* Graph Search:***

A* tree search works well, except that it takes time re-exploring the branches it has already explored. In other words, if the same node has expanded twice in different branches of the search tree, A* search might explore both of those branches, thus wasting time

A* Graph Search, or simply Graph Search, removes this limitation by adding this rule: do not expand the same node more than once. Graph search is optimal only when the forward cost between two successive nodes A and B, given by h(A) – h (B), is less than or equal to the backward cost between those two nodes g(A -> B). This property of the graph search heuristic is called consistency.

***Hill-Climbing Search:***

It is an iterative algorithm that starts with an arbitrary solution to a problem and attempts to find a better solution by changing a single element of the solution incrementally. If the change produces a better solution, an incremental change is taken as a new solution. This process is repeated until there are no further improvements. Hill-Climbing (problem), returns a state that is a local maximum.

***Best First Search:***

It expands the node that is estimated to be closest to goal. It expands nodes based on f(n) = h(n). It is implemented using priority queue. Disadvantage − It can get stuck in loops. It is not optimal.

***Travelling salesman problem:***

The travelling salesman problem (also called the travelling salesperson problem or TSP) asks the following question: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?" It is an NP-hard problem in combinatorial optimization, important in theoretical computer science and operations research. The travelling purchaser problem and the vehicle routing problem are both generalizations of TSP. In the theory of computational complexity, the decision version of the TSP (where given a length L, the task is to decide whether the graph has a tour of at most L) belongs to the class of NP-complete problems. Thus, it is possible that the worst-case running time for any algorithm for the TSP increases superpolynomially (but no more than exponentially) with the number of cities.

The problem was first formulated in 1930 and is one of the most intensively studied problems in optimization. It is used as a benchmark for many optimization methods. Even though the problem is computationally difficult, many heuristics and exact algorithms are known, so that some instances with tens of thousands of cities can be solved completely and even problems with millions of cities can be approximated within a small fraction of 1%.

Other search methods are:-

***Tabu search:***

Tabu search is a metaheuristic search method employing local search methods used for mathematical optimization. Local (neighborhood) searches take a potential solution to a problem

and check its immediate neighbors (that is, solutions that are similar except for very few minor details) in the hope of finding an improved solution. Local search methods have a tendency to become stuck in suboptimal regions or on plateaus where many solutions are equally fit. Tabu search enhances the performance of local search by relaxing its basic rule. First, at each step worsening moves can be accepted if no improving move is available (like when the search is stuck at a strict local minimum). In addition, prohibitions (henceforth the term tabu) are introduced to discourage the search from coming back to previously-visited solutions.

***Pattern search:***

Pattern search (also known as direct search, derivative-free search, or black-box search) is a family of numerical optimization methods that does not require a gradient. As a result, it can be used on functions that are not continuous or differentiable. One such pattern search method is "convergence" (see below), which is based on the theory of positive bases. Optimization attempts to find the best match (the solution that has the lowest error value) in a multidimensional analysis space of possibilities.

***Random Search:***

Random Search (RS) is a family of numerical optimization methods that do not require the gradient of the problem to be optimized, and RS can hence be used on functions that are not continuous or differentiable. Such optimization methods are also known as direct-search, derivative-free, or black-box methods.

***Iterated Local Search:***

Iterated Local Search (ILS) is a term in applied mathematics and computer science defining a modification of local search or hill climbing methods for solving discrete optimization problems.

Local search methods can get stuck in a local minimum, where no improving neighbors are available. A simple modification consists of iterating calls to the local search routine, each time starting from a different initial configuration. This is called repeated local search, and implies that the knowledge obtained during the previous local search phases is not used. Learning implies that the previous history, for example the memory about the previously found local minima, is mined to produce better and better starting points for local search.