**[Local Search Algorithms and Optimization Problem](https://www.tutorialandexample.com/local-search-algorithms-and-optimization-problem)**

The[informed](https://www.tutorialandexample.com/informed-search-heuristic-search/) and [uninformed search](https://www.tutorialandexample.com/uninformed-search-strategies/) expands the nodes systematically in two ways:

* keeping different paths in the memory and
* selecting the best suitable path,

Which leads to a solution state required to reach the goal node. But beyond these **“classical search algorithms**," we have some **“local search algorithms” where the path cost does not matters, and only focus on solution-state needed to reach the goal node.**

A local search algorithm completes its task by traversing on a single current node rather than multiple paths and following the neighbors of that node generally.

**Although local search algorithms are not systematic, still they have the following two advantages**:

* Local search algorithms use a very little or constant amount of memory as they operate only on a single path.
* Most often, they find a reasonable solution in large or infinite state spaces where the classical or systematic algorithms do not work.

**Does the local search algorithm work for a pure optimized problem?**

**Yes,** the local search algorithm works for pure optimized problems. A pure optimization problem is one where all the nodes can give a solution. But the target is to find the best state out of all according to the **objective function**. Unfortunately, the pure optimization problem fails to find high-quality solutions to reach the goal state from the current state.

**Note:** An objective function is a function whose value is either minimized or maximized in different contexts of the optimization problems. In the case of search algorithms, an objective function can be the path cost for reaching the goal node, etc.

**Working of a Local search algorithm**

Let's understand the working of a local search algorithm with the help of an example:

Consider the below state-space landscape having both:

* **Location:**It is defined by the state.
* **Elevation:** It is defined by the value of the objective function or heuristic cost function.



The local search algorithm explores the above landscape by finding the following two points:

* **Global Minimum:** If the elevation corresponds to the cost, then the task is to find the lowest valley, which is known as **Global Minimum.**
* **Global Maxima:** If the elevation corresponds to an objective function, then it finds the highest peak which is called as **Global Maxima**. It is the highest point in the valley.

We will understand the working of these points better in Hill-climbing search.

**Below are some different types of local searches:**

* Hill-climbing Search
* Simulated Annealing
* Local Beam Search

**We will discuss above searches in the next section.**

**Note:** Local search algorithms do not burden to remember all the nodes in the memory; it operates on complete state-formulation.

[← Prev](https://www.tutorialandexample.com/heuristic-functions/)[Next →](https://www.tutorialandexample.com/hill-climbing-algorithm/)