Optimization and Landscape Search

Jonathan Scott[1] and Narges Norouzi[2]

[1] *University of California, Santa Cruz*
[2] *University of California Santa Cruz & University of California, Berkeley*

1 Optimization Problems

An optimization problem involves finding the "best" solution from all possible solutions. We can classify an optimization problem as discrete or continuous. An example of a discrete optimization includes graph search. An example of continuous optimization includes finding the maximizing or minimizing value of a continuous function. In this assignment, we explore local search (i.e. job scheduling, traveling salesman), which implies we are searching for something. In this search, the path is not as critical as the goal or solution. If the path is also important, we refer to this as a systematic search (i.e. route finding, chess).

2 Local Search Algorithms

In many optimization problems, the path to the goal is irrelevant; the goal state itself is the solution. Local search is widely used for problems with very large search spaces, the solutions returned are good but usually not optimal or the best. Local search algorithms keep a single "current" state or a small set of states and iteratively try to improve on them. These usually make this class of algorithms very memory efficient.

3 Algorithms

In this assignment, you have the option of exploring the Ackley, Rastrigin, or Rosenbrock functions. They are all non-convex functions with a global minimum used to test optimization algorithms. An example is given in the attached Python notebook of how to plot the Ackley function, this can be adapted to the other optimization problems.

Ackley function[2]:

$$f(x,y) := -20exp[-0.2\sqrt{0.5(x^2+y^2)}] - exp[0.5(cos(2\pi x)+cos(2\pi y))] + e + 20$$

It has a global optima of f(0,0) = 0

Rastrigin function[3]:

$$f(\mathbf{x}) = An + \sum_{i=1}^{n} \left[x_i^2 - A\cos(2\pi x_i)\right]$$

where $A = 10$ and $x_i \in [-5.12, 5.12]$.

It has a global minimum at $\mathbf{x} = \mathbf{0}$ where $f(\mathbf{x}) = 0$.

Rosenbrock function[4]:

$$f(x, y) = (a - x)^2 + b(y - x^2)^2$$

It has a global minimum at $(x, y) = (a, a^2)$, where $f(x, y) = 0$ and $a = 1, b = 100$.

Select a landscape and find solutions to these problems by implementing three optimization algorithms: Stochastic Hill Climbing with Restarts (SHCR), Simulated Annealing (SA), and Local Beam Search (LBS).

4 Stochastic Hill Climbing with Restarts

Hill climbing treats our optimization surface as a landscape with peaks and valleys. Hill climbing algorithms choose among available uphill moves according to the steepness of these moves. Therefore one needs to define a function that captures steepness, but since there are valleys and plateaus, our algorithm can get stuck. Hence, we incorporate randomness and restarts to search different paths in parallel. This encapsulates the idea of "if at first you don't succeed, try again.". This is done until a goal is reached or the algorithm exhausted all allowed number of restart attempts. This algorithm is asymptotically complete with probability approaching 1, because it will eventually generate a goal state as the initial state (with infinite number of restarts). Below is the pseudo-code for the hill-climbing algorithm which can be restarted and run multiple times until a goal is reached.

```
function Hill-Climbing (problem) returns a solution

inputs: a problem
static: Current, a node
        Next, a node

Current ← Make_Node(Initial State[problem])
for k → 0 to $k_{max}$ do
   Next ← highest-valued successor of the Current
   if Value[Next] < Value[Current] then return Current
   Current ← Next
end
```

5 Simulated Annealing

Simulated annealing is a probabilistic technique for approximating the global optimum of a function. Simulated annealing escapes local minima by allowing some "bad" moves and gradually decreasing their frequency. A typical annealing scheduling is exponential, the temperature schedule or decay schedule should be tuned to the problem. Below is the pseudo-code for the simulated annealing algorithm[1, 5].

```
function Simulated_Annealing (problem) returns a solution

inputs: A problem
        A schedule: a mapping from time to temperature

static: Current, a node
        Next, a node
        T: a temperature controlling the probability
        of downward steps
        decayRate: value in [0,1) used to decay the temperature

Current ← Make_Node(Initial State[problem])
for k → 0 to k$_{max}$ do
   Next ← randomly selected successor of Current
   ΔE ← Value[Next] - Value[Current]
   if ΔE> 0 then Current ← Next
   else Current ← Next with probability $e^{\Delta E/T}$
   T  ← Update_Temperature(decayRate, T, ΔE)
end
```

6 Local Beam Search

Local beam search is a greedy technique that keeps track of k states rather than just one. At each iteration successors of the k states are generated and the best k nodes are selected until a goal is found.

References

[1] Todd W Neller. Teaching stochastic local search. In *FLAIRS Conference*, pages 8–14, 2005.

[2] Wikipedia. Ackley function — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Ackley\%20function&oldid=1069641594, 2022. [Online; accessed 23-November-2022].

[3] Wikipedia. Rastrigin function — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Rastrigin\

%20function&oldid=1069631188, 2022. [Online; accessed 23-November-2022].

[4] Wikipedia. Rosenbrock function — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Rosenbrock\%20function&oldid=1094200520, 2022. [Online; accessed 23-November-2022].

[5] Wikipedia. Simulated annealing — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Simulated\%20annealing&oldid=1115450447, 2022. [Online; accessed 29-November-2022].