CSE3013 Artificial Intelligence

Module-3: Local Search

Dr Sunil Kumar P V

SCOPE

Overview

Local Search

- Hill-Climbing Search
- Simulated annealing

Local Search

Local search- overview

- Up to now, single category of problems:
  - observable, deterministic, known environments
  - where the solution is a sequence of actions
- Now these assumptions are relaxed
- Local search- evaluate and modify one or more current states rather than systematically exploring paths from an initial state
- Useful when only the solution state is relevant, not the path cost to reach it

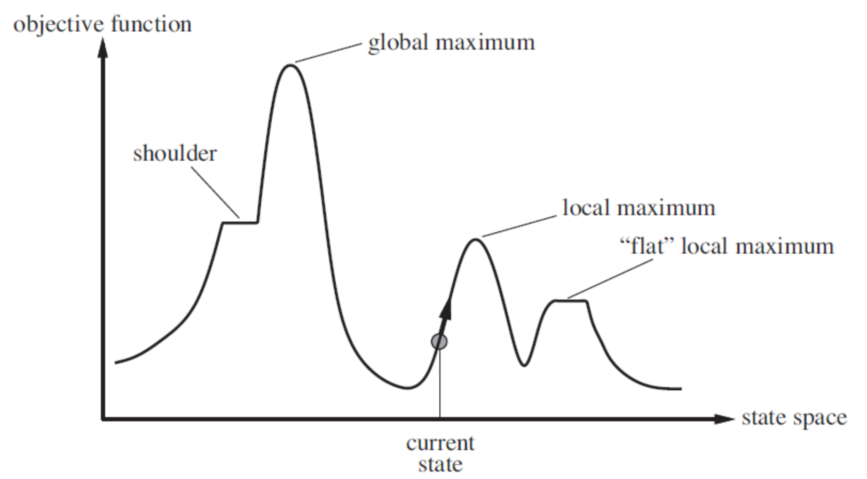
When local search?

- Up to now, paths to goal states and various alternatives are memorized
- Systematic exploration of the search space
- Path to the goal is also part of the solution
- For many problems, path to the goal is irrelevant
    - N-queens problem
    - integrated-circuit design
    - factory-floor layout
    - job-shop scheduling
    - automatic programming
    - telecommunications network optimization
    - vehicle routing
    - portfolio management.
- If the path to the goal does not matter, we might consider a different class of algorithms, which do not worry about paths at all

Local search- features

- Local search algorithms operate using a single current node (rather than multiple paths) and generally move only to neighbors of that node
- Typically, the paths followed by the search are not retained
- Advantages: -
  1. They use very little memory—usually a constant amount
  2. They yield solutions in large or infinite state spaces for which systematic algorithms are unsuitable
- Especially useful for optimization problems, to find the best state for a given objective function

State-space landscape



Terminologies

- **State-space landscape:** State-space v/s objective function/heuristic cost function plot
  - Has **location** (defined by the state) and **elevation** (defined by the value of the heuristic cost function or objective function)
- If elevation corresponds to cost, then the aim is to find the lowest valley—a **global minimum**
- If elevation corresponds to an objective function, then the aim is to find the highest peak—a **global maximum**
- Local search algorithms explore this landscape
- Local search varieties:
  - **Complete local search** algorithm always finds a goal if one exists
  - **Optimal** algorithm always finds a global minimum/maximum

Hill-Climbing Search

- Steepest ascent version
- Is a loop that continually moves in the direction of increasing value—that is, uphill
- Terminates when it reaches a "peak" where no neighbor has a higher value
- Does not look ahead beyond the immediate neighbors of the current state

Hill-climbing search algorithm

function HILL-CLIMBING(*problem*) **returns** a state that is a local maximum

current $\leftarrow$ MAKE-NODE(*problem*.INITIAL-STATE)
loop do
    neighbor $\leftarrow$ a highest-valued successor of *current*
    if neighbor.VALUE $\leq$ current.VALUE **then return** *current*.STATE
    current $\leftarrow$ *neighbor*

- Greedy local search $\longrightarrow$ it grabs a good neighbor state without thinking ahead about where to go next
- Local maxima $\longrightarrow$ a local maximum is a peak that is higher than each of its neighboring states but lower than the global maximum
- Ridges $\longrightarrow$ result in a sequence of local maxima that is very difficult for greedy algorithms to navigate
- Plateaux $\longrightarrow$ a plateau is a flat area of the state-space landscape

Difficulties in hill-climbing

- Local maxima
- Ridges
- The algorith halts If it reaches a plateau where the best successor has the same value as the current state
- Allow sideways move- but can result in infinite loop, if not a shoulder
- Solution $\longrightarrow$ Limit the number of sideways moves

Hill-climbing variations

- We discussed steepest ascent now
- Stochastic hill-climbing $\longrightarrow$ chooses at random from among the uphill moves; the probability of selection can vary with the steepness of the uphill move
- First-choice hill climbing $\longrightarrow$ implements stochastic hill climbing by generating successors randomly until one is generated that is better than the current state
  - good strategy when a state has many (e.g., thousands) of successors
- Random-restart hill climbing $\longrightarrow$ conducts a series of hill-climbing searches from randomly generated initial states, until a goal is found
  - Useful as all other hill-climbing versions are **incomplete**- often fail to find a goal when one exists because they can get stuck on local maxima

Simulated annealing

- Hill climbing with no downward movement to lower value or higher cost is **incomplete,** due to local maximum
- Purely random walk—that is, moving to a successor chosen uniformly at random from the set of successors—is complete but extremely inefficient
- Hybrid of hill climbing and random walk $\longrightarrow$ efficiency and completeness

function SIMULATED-ANNEALING(*problem*, *schedule*) **returns** a solution state
  inputs: *problem*, a problem
        schedule, a mapping from time to “temperature”

  current $\leftarrow$ MAKE-NODE(*problem*.INITIAL-STATE)
  for t = 1 **to** ∞ **do**
    $T \leftarrow schedule(t)$
    if $T = 0$ **then return** *current*
    next $\leftarrow$ a randomly selected successor of *current*
    $\Delta E \leftarrow$ *next*.VALUE – *current*.VALUE
    if $\Delta E > 0$ **then** *current* $\leftarrow$ *next*
    else *current* $\leftarrow$ *next* only with probability $e^{\Delta E/T}$

Algorithm

- Temperature in annealing is mapped to time here
- Instead of picking the best move the algorithm picks a random move. If the move improves the situation, it is always accepted
- Otherwise, the algorithm accepts the move with some probability less than 1
- The probability decreases exponentially with the "badness" of the move—the amount ΔE by which the evaluation is worsened
- The probability also decreases as the "temperature" T goes down
- "bad" moves are more likely to be allowed at the start when T is high, and they become more unlikely as T decreases

More local search algorithms

- Local beam search
- Genetic algorithms

Thank you...