Universidade do Minho
Escola de Engenharia
Departamento de Informática

Local Search and Optimization

Single Solution Algorithms

LICENCIATURA EM ENGENHARIA INFORMÁTICA
MESTRADO integrado EM ENGENHARIA INFORMÁTICA
Inteligência Artificial
2025/26

Agenda

- Search vs Optimization
- Exploration vs Exploitation
- Meta-heuristics
- Local vs global search
- Single solution vs Population-Based
- Single solution
  - Hill-Climbing Search
  - Simulated Annealing
  - Tabu Search
- Search with non-deterministic actions

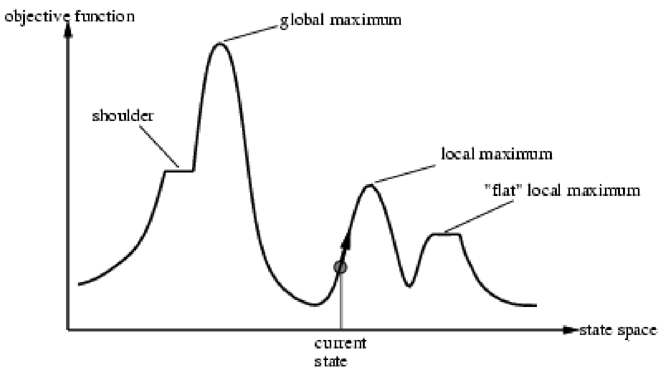
- Population-Based

Optimization Problems

- How can a cutlery manufacturer get the maximum number of pieces out of a piece of sheet metal?
- How can a carrier organize itself to distribute parcels in a given container or transport?
- How can a telephone operator route calls to get the best use out of its lines?
- How can a university draw up class schedules to make the best use of classrooms without conflicts?

Search and Optimization

- So far we have essentially dealt with a single category of problems: observable, deterministic and known environments, where the solution is a sequence of actions.
    - **Not all environments are like this!**
- Algorithms that perform purely local search in the state space, evaluating and modifying one or more current states, rather than systematically exploring paths from an initial state.
- These algorithms are suitable for problems in which all that matters is the state of the solution, not the cost of the path to reach it.



Source: Russell and Norvig, (2009) Artificial Intelligence - A Modern Approach.

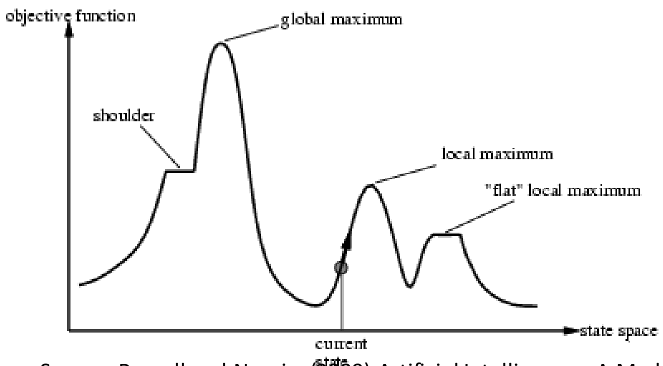
Searching for solutions

- It doesn't matter how far you get

    ex: queens problem, schedules, optimization - of networks, shop floors, etc.

- In an optimization problem - you may not know if you've already reached the optimum value

    if we don't know the optimum of the function we're optimizing...



Source: Russell and Norvig, (2009) Artificial Intelligence - A Modern Approach.

Exploration vs Exploitation

- **Solving complex problems requires obtaining the best possible solutions in good time;**
- **To do this, instead of trying out all possible solutions (guaranteeing the optimum solution), it is necessary to identify solutions that are as close as possible to the optimum solution, in a limited number of attempts;**
- **An appropriate balance between :**
  - ***Exploration***: general exploration of the search space;
  - ***Exploitation***: search focused on the most promising areas.

Synthetic Intelligence Lab

Exploration vs Exploitation

- *Exploration* without *Exploitation* gives an overview of the search space, but without getting too close to the optimum value;
- Exploiting an area at an early stage in the search process can lead to the search getting stuck in a local optimum;
- This balancing is usually successfully managed using metaheuristics

Synthetic Intelligence Lab

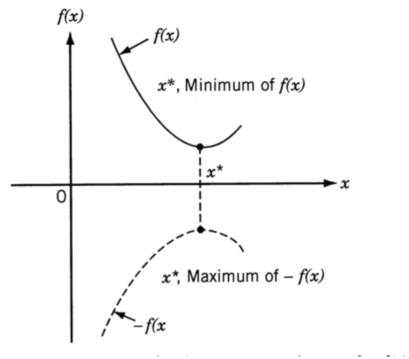
- A metaheuristic is a heuristic method for solving optimization problems in a generic way;
- Meta-heuristics are generally applied to problems for which no efficient algorithms are known;
- They use a combination of random choices and historical knowledge of previous results acquired by the method to guide themselves and carry out their searches in neighborhoods within the search space, which can avoid local optima.

Meta-heurístics

- **Usually inspired by natural phenomena;**
- **Due to their random component, they are non-deterministic;**
- **They do not guarantee the identification of the optimum solution:**
  - a close solution;
  - In the shortest execution time;
  - Using fewer computing resources than traditional techniques.

- **Mathematical Optimization Problem (minimization):**

$$\begin{aligned}&\text{minimize} \quad f_0(x)\\&\text{subject to} \quad g_i(x) \le b_i, \quad i = 1,\dots,m\end{aligned}$$



- ***$f_0 : R^n \rightarrow R$:***
  - Objective function (calculates a value)
- ***$x = (x_1,\dots\dots,x_n)$:***
  - Controllable variables (linearly independent)
- ***$g_i : R^n \rightarrow R$: ($I = 1,\dots,m$):***
  - Restrictions (can be of other types)

Beyond Classical Search
Iterative Improvement Algorithms

- In many optimization problems, the path to the goal is irrelevant.
- State space = set of complete configurations.
- Iterative algorithms maintain a single (current) state and try to improve it.
- Iterative Improvement Algorithms:
  - Hill-Climbing Search
  - Simulated Annealing
  - Tabu Search
  - Genetic Algorithms
  - Ant Colony Optimization
  - Particle Swarm Optimization
- **Strategy:** Start with an initial solution to the problem and make changes to improve its quality.

- **Local Demand vs. Global Demand**
  - Some metaheuristics apply local search methods, where the new solutions explored are "neighbors" of previous solutions (e.g. Simulated Annealing, Tabu Search);
  - Other metaheuristics distribute the search process over the entire search space (usually through population-based approaches).

- **Single solution vs Population-based**
  - Single solution approaches are iterative and guide the search process by improving the previous solution;
  - Population-based approaches use a parallel search by several members of the population, and there may or may not be an exchange of information between individuals (e.g. Particle Swarm optimization, Genetic Algorithms, Ant Colony optimization).

ISLab
Synthetic Intelligence Lab

Iterative Improvement Algorithms
Single Solution

"Individual Based" (single solution)!

- Hill-Climbing Search:
  - Choose a state at random from the state space
  - Consider all the neighbors of that state
  - Choose the best neighbor
  - Repeat the process until there are no better neighbors
  - The current state is the solution

- Simulated Annealing:
  - Similar to Hill-Climbing Search but allows the exploration of worse neighbors
  - Temperature that is successively reduced defines the probability of accepting worse solutions
- Tabu Search:
  - Similar to Hill-Climbing Search, explores neighboring states but eliminates the worst ones (taboo neighbors)
  - Deterministic algorithm

Hill-Climbing Search

- **O Hill Climbing** is a classic algorithm and is very efficient at finding local maxima or minima by exploring.

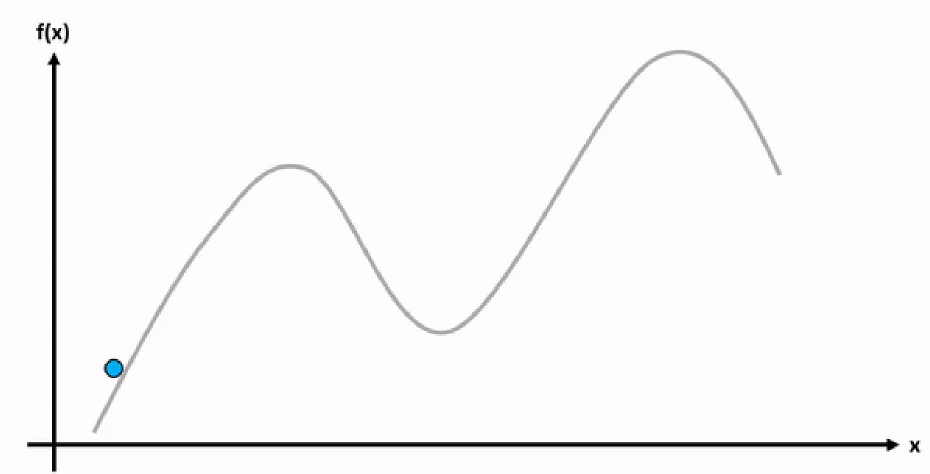
- **Startegy:**
  - It starts at a random point X and evaluates that point;
  - Move from the original point X to a new point Y near point X;
  - if this new point Y is a better solution than the original point X, fix point Y and start the process again, but
  - o visit another neighbor.

```
function HILL-CLIMBING(problem) returns a state that is a local maximum
   inputs: problem, a problem
   local variables: current, a node
                    neighbor, a node
   current ← MAKE-NODE(INITIAL-STATE[problem])
   loop do
      neighbor ← a highest-valued successor of current
      if VALUE[neighbor] ≤ VALUE[current] then return STATE[current]
      current ← neighbor
```

Source: Russell and Norvig, (2009) Artificial Intelligence - A Modern Approach.

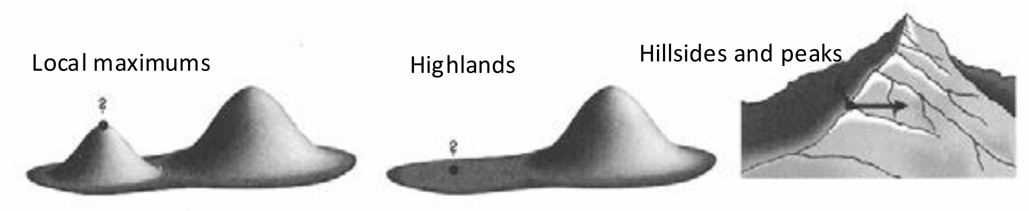
Hill-Climbing Search

- Hill Climbing is great for finding good solutions (local minimum/maximum) but is unlikely to find the best solution (unless you get "lucky" in initializing the starting point).



Source: https://www.globalsoftwaresupport.com/wp-content/uploads/2018/04/ezgif.com-video-to-gif-47.gif

Hill-Climbing Search



- In the cases presented, the algorithm reaches a point where it can no longer make progress.
- ***Solution****: random restart*
    - The algorithm performs a series of searches from randomly generated initial states.
- Each search is executed:
    - Until a stipulated maximum number of iterations is reached, or
    - Until the results found do not show a significant improvement.
- The algorithm chooses the best result obtained from the different searches.

- Success depends very much on the morphology (shape) of the state space surface:
    - If there are few local maxima, random restart finds a good solution quickly
    - Otherwise, the time cost is exponential.

ISLab
Synthetic Intelligence Lab

Simulated Annealing

- **Simulated Annealing** is an algorithm inspired by nature, as are Artificial Neural Networks and Genetic Algorithms, among others.
- Assumption: Escape the local minimum by allowing some "bad" movements but gradually reducing their size and frequency.!

- **Strategy:**
  - Similar to Hill Climbing, it starts at a random point X and evaluates it;
  - the algorithm moves to one of its neighbors Y and evaluates this new point;
    - If the results have improved at this new point Y then move to Y and redo the previous process, however if point Y is lower, move to this point Y if the **probability** of going to a negative point is greater than a random number.

```
function SIMULATED-ANNEALING(problem, schedule) returns a solution state
   inputs: problem, a problem
           schedule, a mapping from time to "temperature"
   local variables: current, a node
                    next, a node
                    T, a "temperature" controlling prob. of downward steps

   current ← MAKE-NODE(INITIAL-STATE[problem])
   for t ← 1 to ∞ do
      T ← schedule[t]
      if T = 0 then return current
      next ← a randomly selected successor of current
      ΔE ← VALUE[next] – VALUE[current]
      if ΔE > 0 then current ← next
      else current ← next only with probability e^(Δ E/T)
```

probability(p) = Exp(Y- X / T)

This exponential function calculates the difference of point Y subtracted by point X (previous position), divided by the temperature (variable T).
As a result, in the first few iterations, when the temperature T is higher, there is a greater probability of accepting negative values and with subsequent iterations, the probability decreases.

Over time (decreasing temperature), the algorithm starts to behaves like Hill-Climbing

Source: Russell and Norvig, (2009) Artificial Intelligence - A Modern Approach.

Tabu Search

- The basic idea of Tabu search is to penalize moves that take the solution to previously visited search spaces (also known as tabu).
- Tabu search, however, deterministically accepts solutions that don't improve to avoid getting stuck in local minima.
- **Strategy:**
  - Key idea: maintaining the sequence of nodes already visited (Taboo list);
  - Starting from an initial solution, the search moves, at each iteration, to the best solution in the neighborhood, not accepting movements that lead to solutions that have already been visited; these known movements are stored in a taboo list;
  - The list remains in memory, storing the solutions already visited (tabu) for a certain amount of time or a certain number of iterations (tabu deadline). The end result is expected to be a global optimum or a value close to the global optimum.

Tabu Search

Algorithm 1 Tabu search algorithm

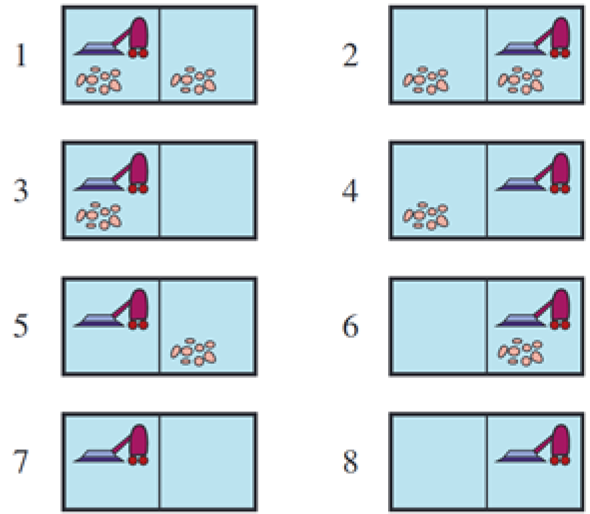
```
Set x = x_0;                                ▷ Initial candidate solution
Set length(L) = z;                          ▷ Maximum tabu list length
Set L = {};                                 ▷ Initialize the tabu list
repeat
    Generate a random neighbor x';
    if x' ∉ L then
        if length(L) > z then
            Remove oldest solution from L;  ▷ First in first out queue
            Set x' ∈ L;
        end if
    end if
    if x' < x then
        x = x';
    end if
until (Stopping criteria satisfied)         ▷ e.g.Number of iterations
return x;                                   ▷ Best found solution
```

Source: F. Glover and M. Laguna (1997) Tabu Search, Kluwer Academic Publishers, Springer

Search with Non-Deterministic Actions
Faults in the suction mechanism

Assuming there are vacuuming faults :

- can clean two cells
- can deposit dirt



- starting in state 1

Contingency plan

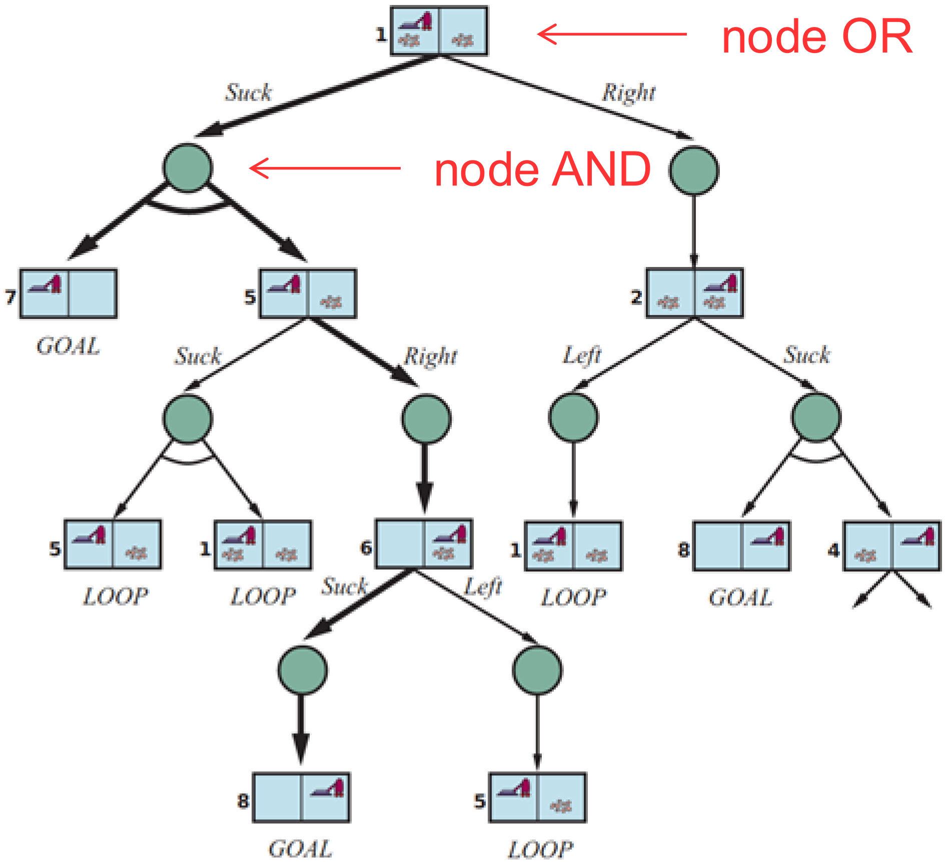
– vacuum

– **if** State = 5 **then**
Right, Vacuum

Source: Russell and Norvig, (2009) Artificial Intelligence - A Modern Approach.

ISLab
Synthetic Intelligence Lab

Trees E-OU (*AND-OR*)



- agent's actions - nodes OR
- Result. In world – nodes AND

- Generic Plan
  - –a goal on each leaf
  - –an action on each node OR
  - –all results in nodes AND

Source: Russell and Norvig, (2009) Artificial Intelligence – A Modern Approach.

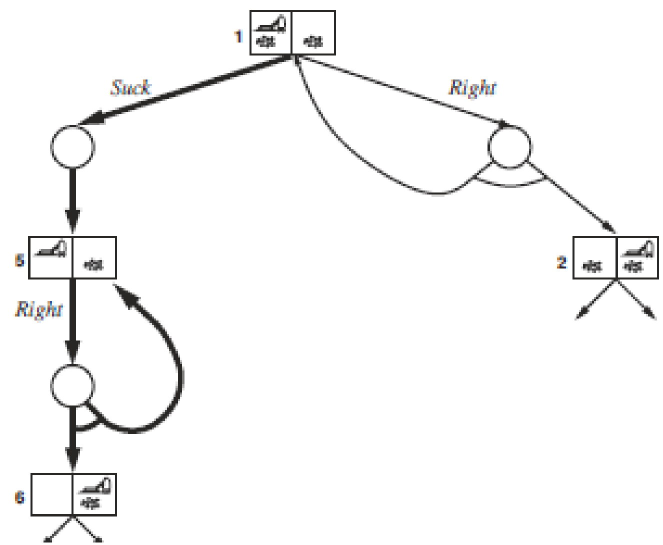
Synthetic Intelligence Lab

Search with non-deterministic actions
Faults in the movement mechanism

Assuming there is movement of the agent (vacuum cleaner):

- movement actions sometimes fail, leaving the agent in the same location.

- For example, moving to the right in state 1 leads to the set of states {1, 2}.
  - Cyclical solution that consists of continuing to try the Right until it works.

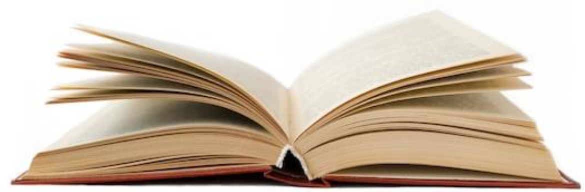


- Classical search methods essentially address a single category of problems: observable, deterministic and known environments, where the solution is a sequence of actions;
- In this context, we analyzed what happens when these assumptions are relaxed. Using algorithms that perform a purely local search in the state space, evaluating and modifying one or more states instead of systematically exploring paths from an initial state;
- These algorithms are suitable for problems where all that matters is the state of the solution, not the cost of the path to reach it;
- This family of local search algorithms includes methods inspired by statistical physics (Simulated annealing) and evolutionary biology (Genetic algorithms).

References

Recommended Bibliography

- Russell and Norvig, (2009) Artificial Intelligence - A Modern Approach, 3rd edition, ISBN-13: 9780136042594, chapter 4.



Universidade do Minho
Escola de Engenharia
Departamento de Informática

Local Search and Optimization

Single Solution Algorithms

LICENCIATURA EM ENGENHARIA INFORMÁTICA
MESTRADO integrado EM ENGENHARIA INFORMÁTICA
Inteligência Artificial
2025/26