Local Search

PREPARED BY

DR. ALIYA ALERYANI

Outlines

- Local Search and Optimization Problems
- Terminologies
- Hill Climbing Algorithm
- Examples
- Hill Climbing Pros/Cons
- Hill Climbing Types
- Simulated Annealing

Local Search and Optimization Problems

- **Local Search** algorithms operate by searching from a start state to neighbouring states, without keeping track of the paths, nor the set of states that have been reached.
- might never explore a portion of the search space where a solution actually resides.
- solve **optimization problems**, in which the aim is to **find the best state** according to an objective function.

Key advantages:

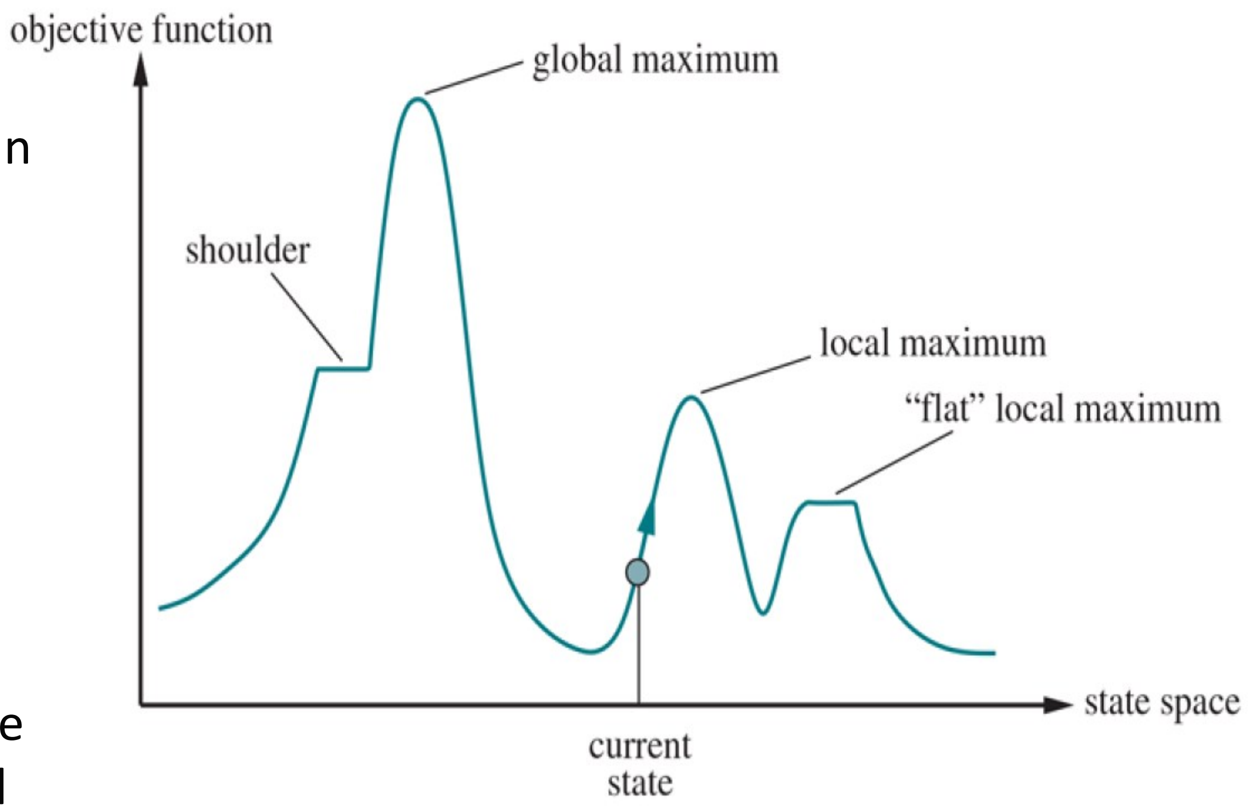
1. use very little memory
2. often find reasonable solutions in large or infinite state spaces for where systematic algorithms are unsuitable.

Local Search

- Each point (state) in the landscape has an "elevation," defined by the value of the objective function.

- If elevation corresponds to an objective function, then the aim is to find the highest peak—a global maximum—and we call the process **hill climbing**.

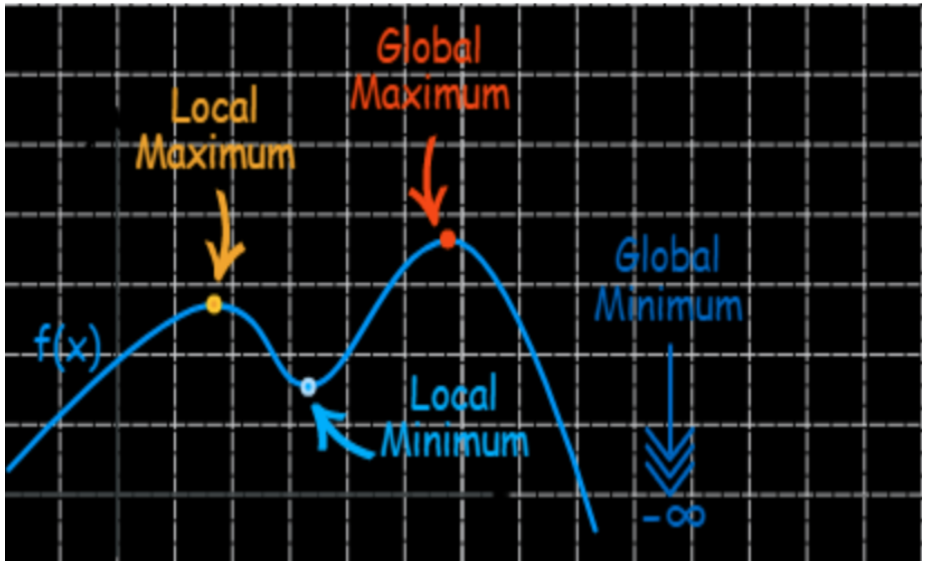
- If elevation corresponds to cost, then the aim is to find the lowest valley—a global minimum—and we call it **gradient descent**.



A one-dimensional state-space landscape in which elevation corresponds to the objective function. The aim is to find the global maximum.

Terminologies

- A high point is called a **maximum (plural maxima).**
- A low point is called a **minimum (plural minima).**
- We say **local maximum (or minimum)** when there may be higher (or lower) points elsewhere but not nearby.
- The **maximum or minimum** over the entire function is called an "**Absolute" or "Global" maximum or minimum.**



Example

Find the maxima and minima for a function:

$$y = 5x^3 + 2x^2 - 3x \qquad (1)$$

<u>**Solution:**</u>

1) Apply first derivative to equation (1)

   $y' = 15x^2 + 4x - 3 \qquad (2)$

2) Find the value of x

   $15x^2+9x-5x-3=0$

   $3x(5x+3)-1(5x+3)=0$ using factorization

   $(3x-1)(5x+3)=0$

   $x=1/3, \quad x=-3/5$

   <u>At $x=1/3$:</u>

   $y'=15(1/3)^2 +4(1/3) - 3 = 0$

   <u>At $x=-3/5$:</u>

   $y'= 15(-3/5)^2 +4(-3/5) - 3 = 0$

Example

3) Apply derivative once more to equation (2), the second derivative

$y'' = 30x + 4$ (3)

<u>At $x = -3/5$:</u>

$y'' = 30(-3/5) + 4 = -14$

it is less than 0, so **−3/5 is a local maximum**

<u>At $x = +1/3$:</u>

$y'' = 30(+1/3) + 4 = +14$

it is greater than 0, so **+1/3 is a local minimum**

Hill-Climbing Search

- The hill-climbing search algorithm keeps track of one current state and on each iteration moves to the neighbouring state with highest value.
- heads in the direction that provides the **steepest ascent**.
- terminates when it reaches a “peak” where no neighbour has a higher value.
- does not look ahead beyond the immediate neighbours of the current state.
- one way to use hill-climbing search is to use the negative of a heuristic cost function as the objective function; that will climb locally to the state with smallest heuristic distance to the goal.
- sometimes called greedy local search because it grabs a good neighbour state without thinking ahead about where to go next.

Hill-Climbing Search

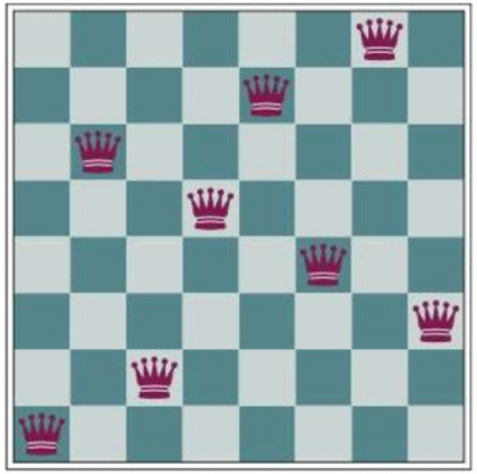
The hill-climbing search algorithm, which is the most basic local search technique. At each step the current node is replaced by the best neighbour.

function HILL-CLIMBING(*problem*) **returns** a state that is a local maximum
    current ← *problem*.INITIAL
    while *true* **do**
        neighbor ← a highest-valued successor state of *current*
        if VALUE(*neighbor*) ≤ VALUE(*current*) **then return** *current*
        current ← *neighbor*

Example

8 Queens

- The goal is to minimize the number of queens attacking each other.
- Heuristic cost estimate, h=number of queens (Q) attacking each others directly or indirectly.
- Successors are all possible states
  moving one Q in the same column.
- 8 * 7= 56 possible successors.
- (a) h=1
- (b) h= 17



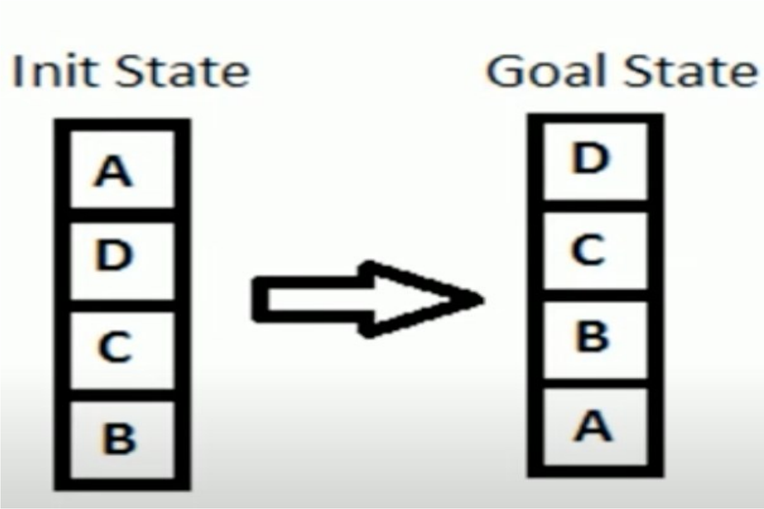
(a)

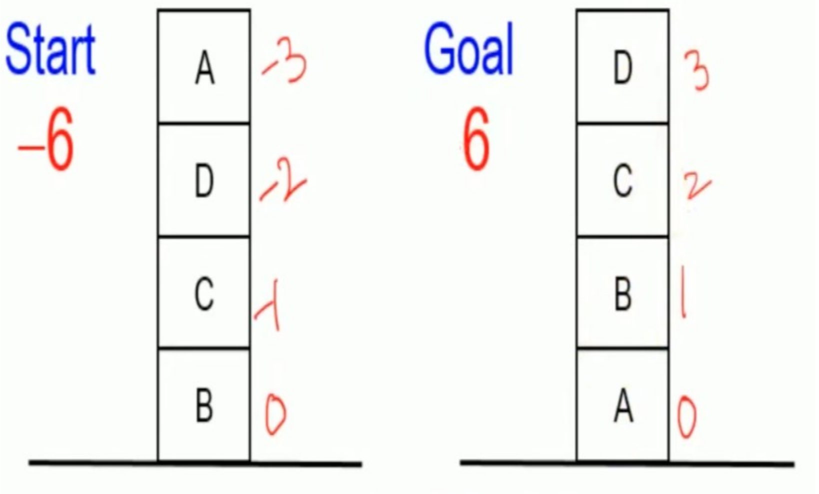
(b)

Example

4-Blocks: global heuristic function

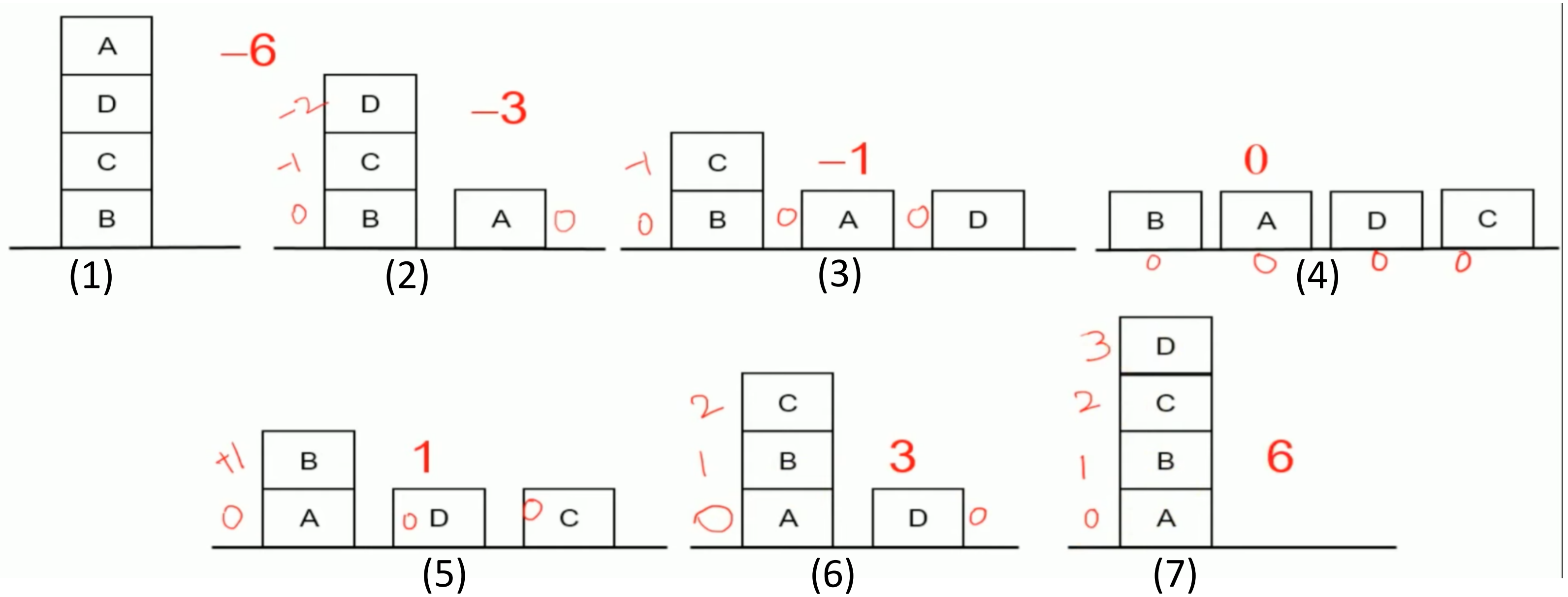
- h= +1 for all the blocks in the support structure if the block is correctly positioned
- otherwise h= -1





Example

Start with h= -6

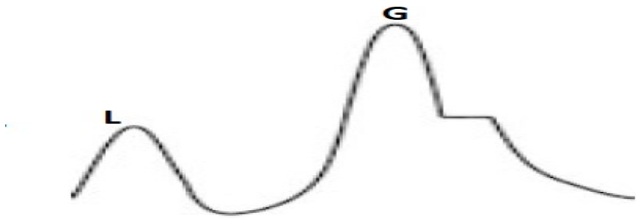


End with h=-6

Hill-Climbing Search Pros/Cons

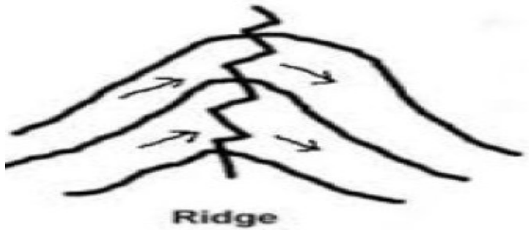
Pros:

- It can make rapid progress toward a solution because it is usually quite easy to improve a bad state.
- No backtracking



Cons:

- **LOCAL MAXIMA**: a peak that is higher than each of its neighbouring states but lower than the global maximum.
- **PLATEAUS**: a flat area of the state-space landscape. It can be a flat local maximum, from which no uphill exit exists, or a shoulder, from which progress is possible.
- **RIDGES**: result in a sequence of local maxima that is very difficult for greedy algorithms to navigate.



Hill Climbing Types

1. Stochastic hill climbing
   - chooses at random from among the uphill moves
   - the probability of selection can vary with the steepness of the uphill move
   - converges more slowly than steepest ascent
   - finds better solutions in some state
2. First-choice hill climbing
   - implements stochastic hill that generates successors randomly until one is generated that is better than the current state
   - a good strategy when a state has many (e.g., thousands) of successors.
3. Random-restart hill climbing
   - conducts a series of hill-climbing searches from randomly generated initial states, until a goal is found
   - complete with probability 1, because it will eventually generate a goal state as the initial state

Simulated Annealing

- Key Idea: escape local maxima by allowing some "bad" moves but gradually decrease their frequency
- Take some uphill steps to escape the local minimum
- Instead of picking the best move, it picks a random move
- If the move improves the situation, it is executed. Otherwise, move with some probability less than 1.
- Physical analogy with the annealing process:
  - Allowing liquid to gradually cool until it freezes
- The heuristic value is the energy, E
- Temperature parameter, T, controls speed of convergence.

Simulated Annealing

- **Basic inspiration**: What is annealing?
  - In mettallurgy, annealing is the physical process used to temper or harden metals or glass by heating them to a high temperature and then gradually cooling them, thus allowing the material to coalesce into a low energy cristalline state.
    - **Heating then slowly cooling a substance to obtain a strong cristalline structure.**
- **Key idea:** Simulated Annealing combines Hill Climbing with a random walk in some way that yields both efficiency and completeness.
- Used to solve VLSI layout problems in the early 1980

Simulated Annealing

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 $t \leftarrow$ 1 **to** ∞ **do**
    $T \leftarrow$ *schedule*[t]
    if $T = 0$ **then return** *current*
    next ← a randomly selected successor of *current*
    $\Delta E \leftarrow$ VALUE[*next*] – VALUE[*current*]
    if $\Delta E > 0$ **then** *current* ← *next*
    else *current* ← *next* only with probability $e^{\Delta E/T}$

Simulated Annealing

Temperature T

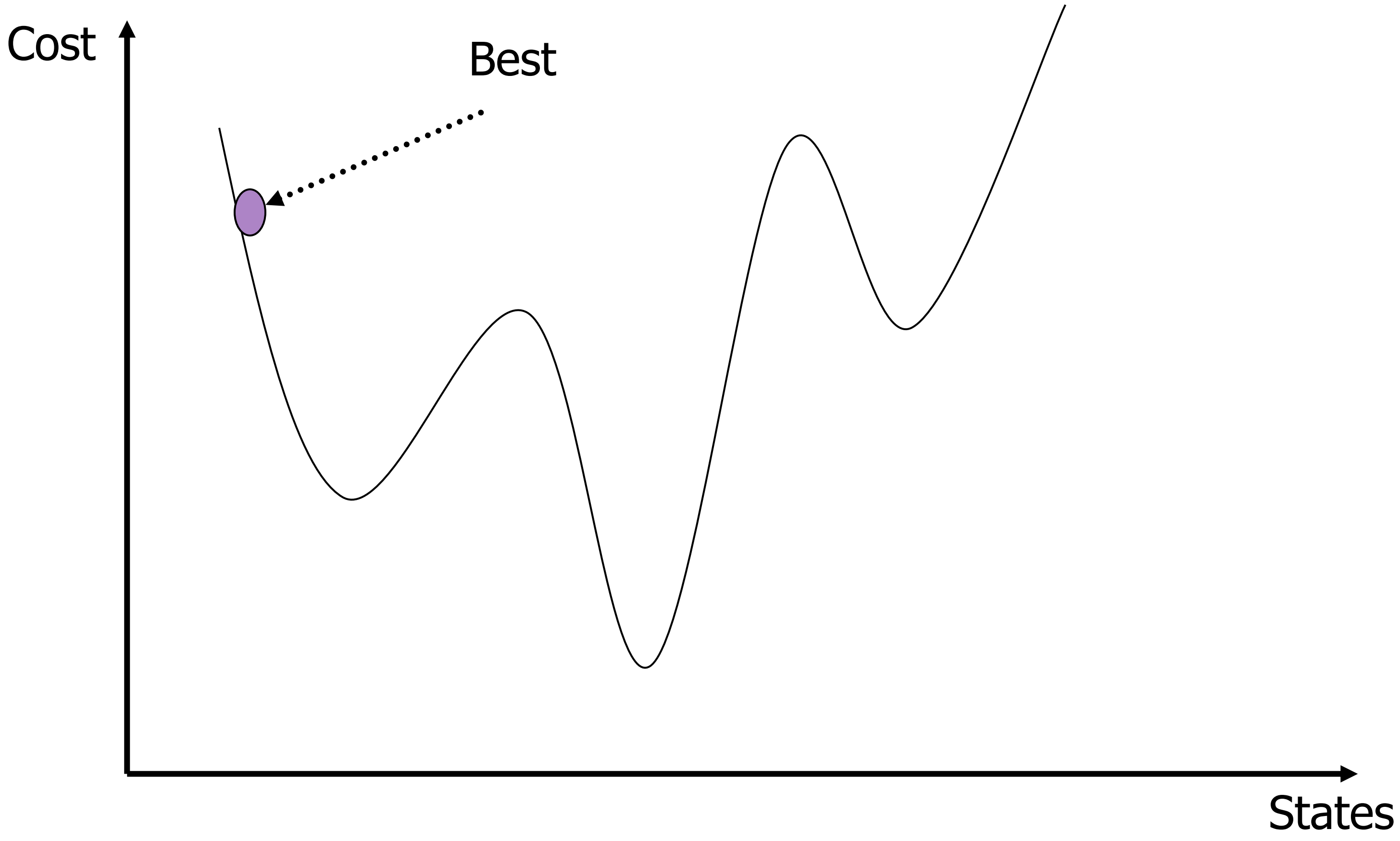
- Used to determine the probability
- High T : large changes
- Low T : small changes

Cooling Schedule

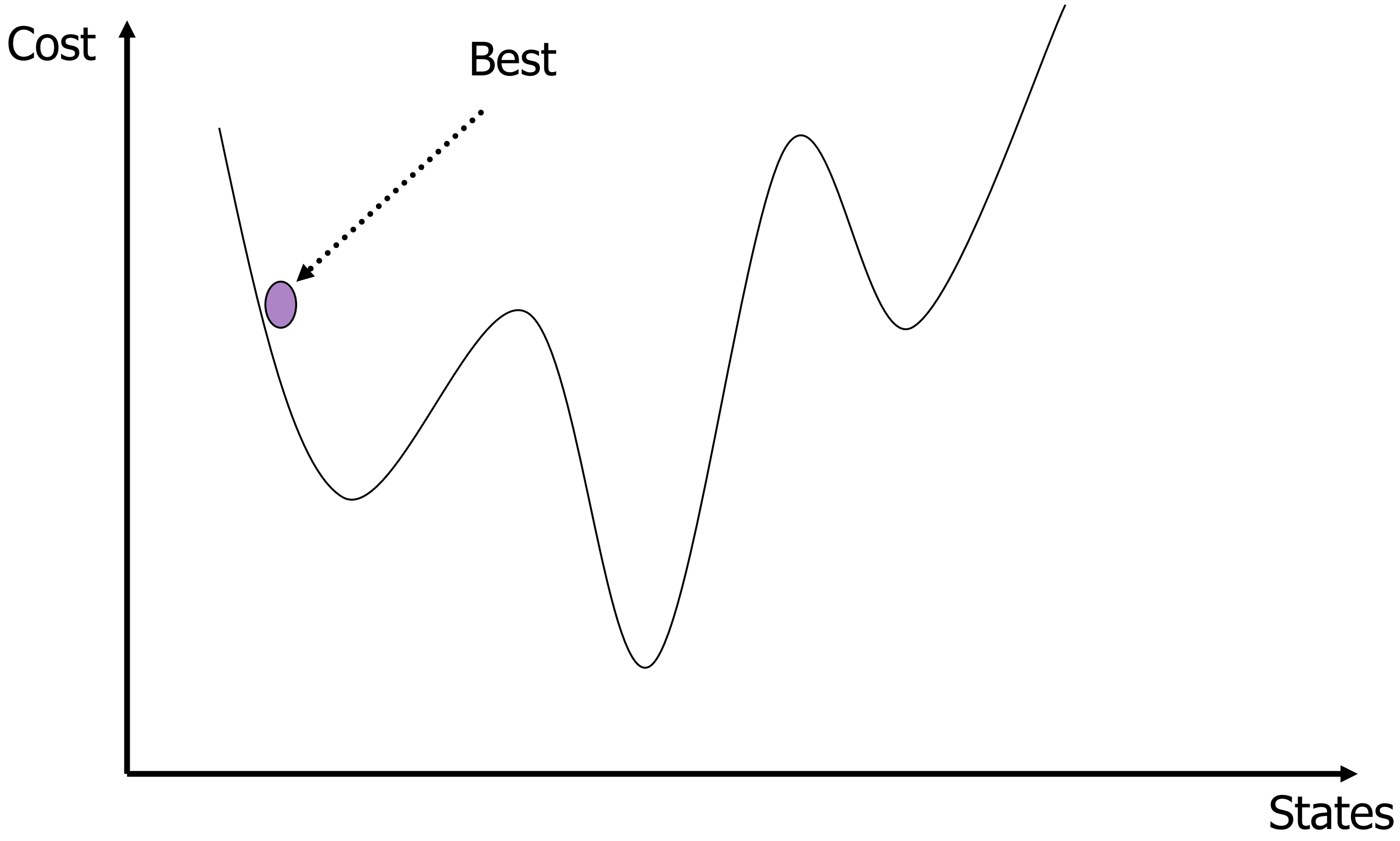
- Determines rate at which the temperature T is lowered
- Lowers T slowly enough, the algorithm will find a global optimum

In the beginning, aggressive for searching alternatives, become conservative when time goes by

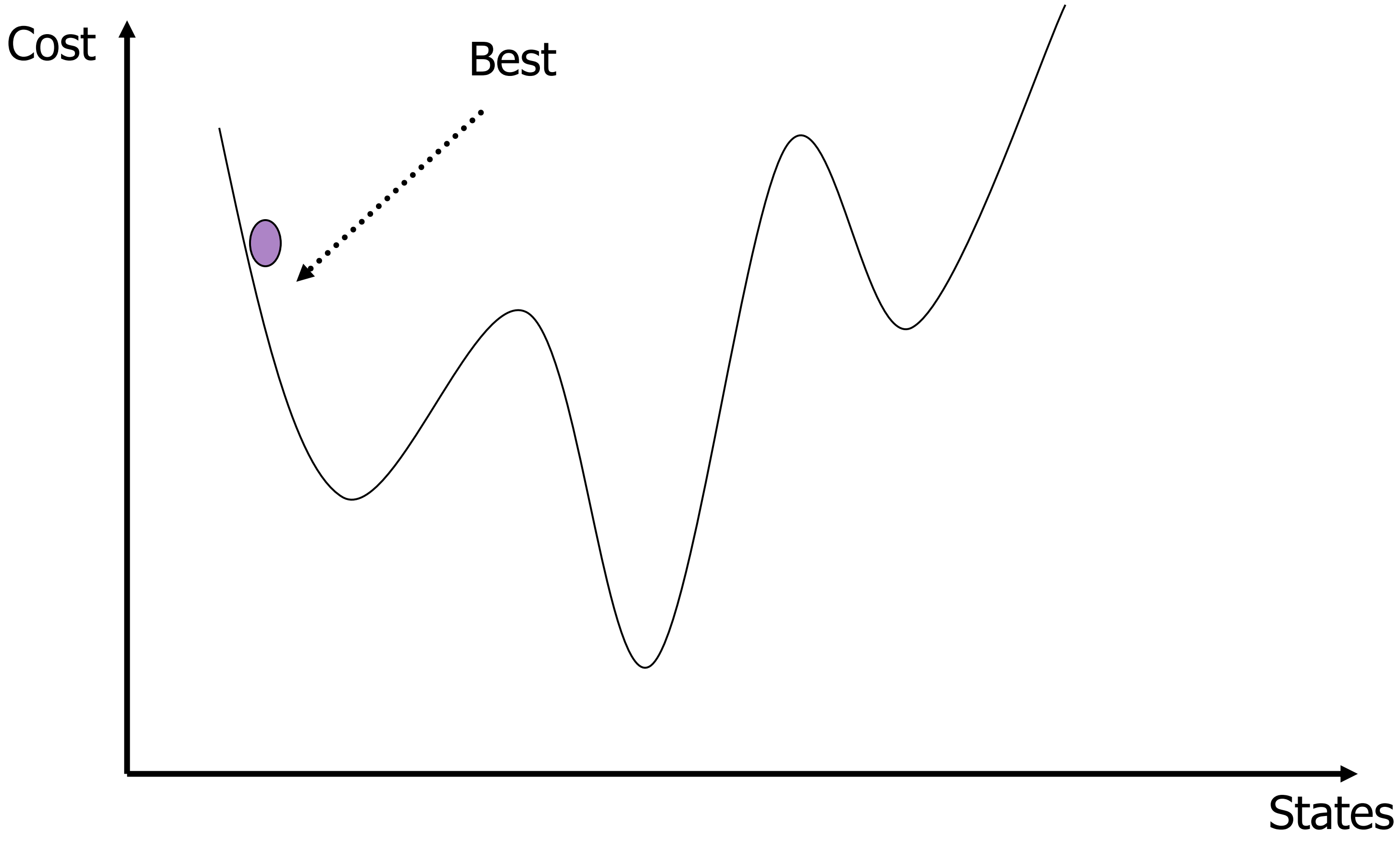
Simulated Annealing



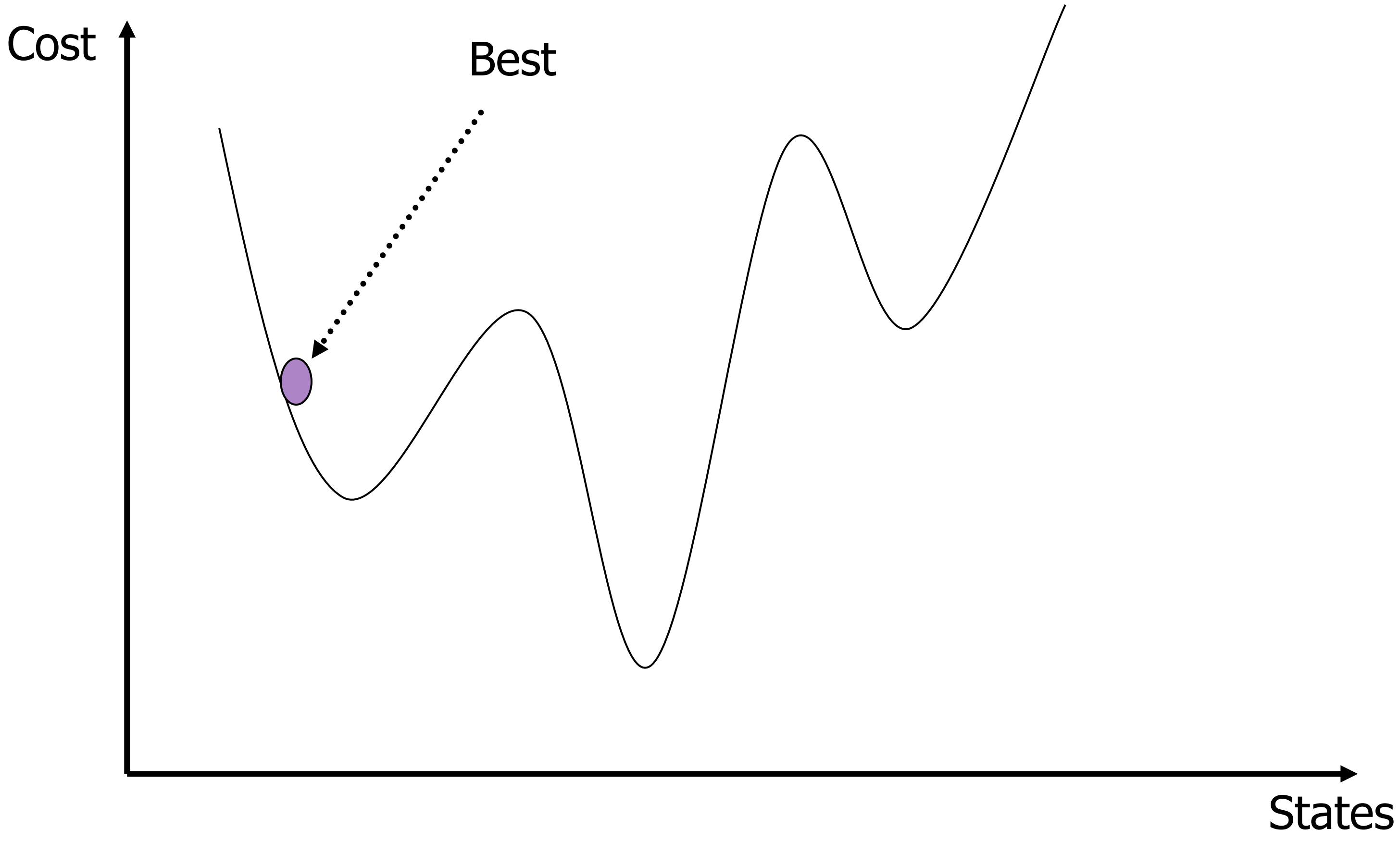
Simulated Annealing



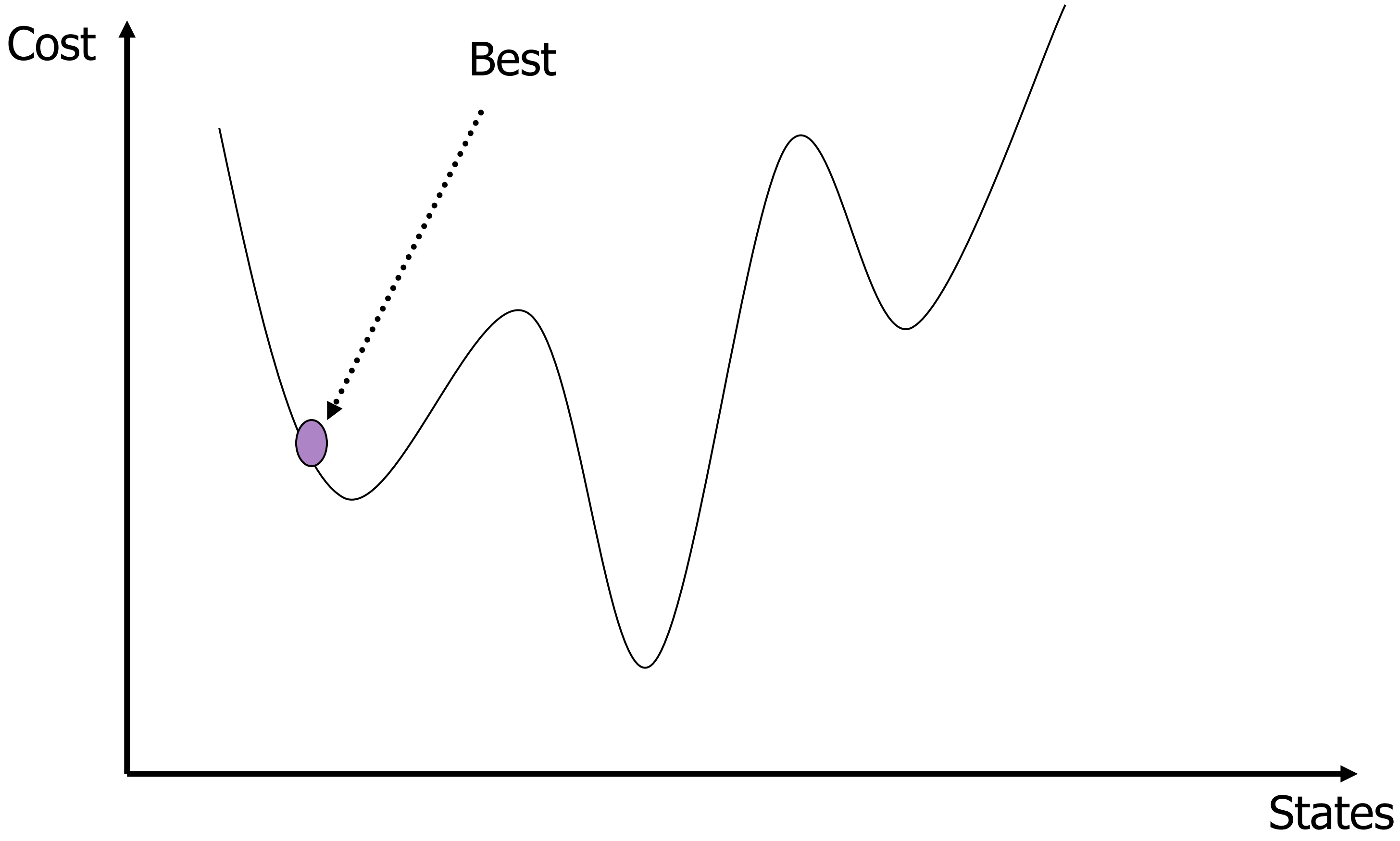
Simulated Annealing



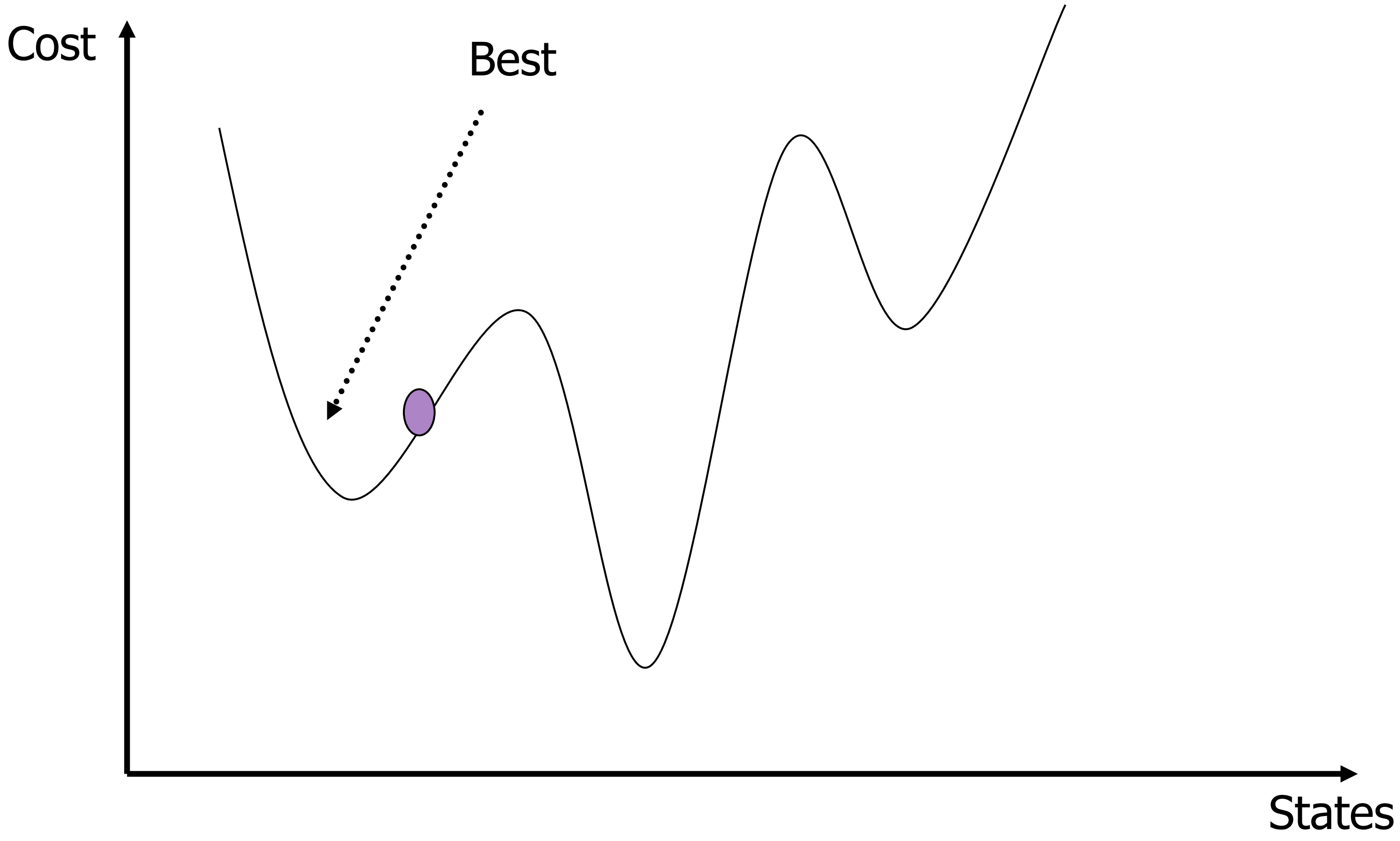
Simulated Annealing



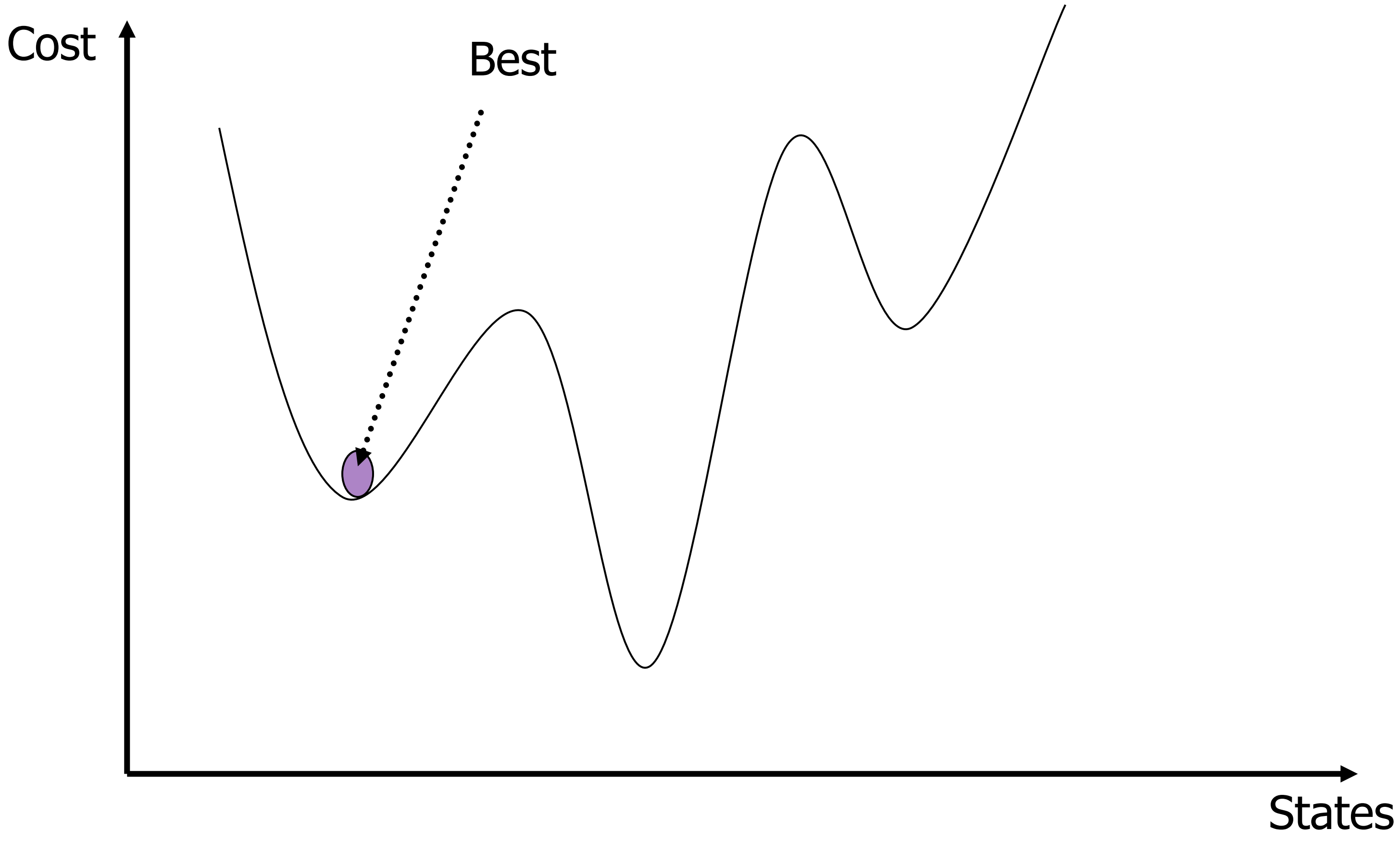
Simulated Annealing



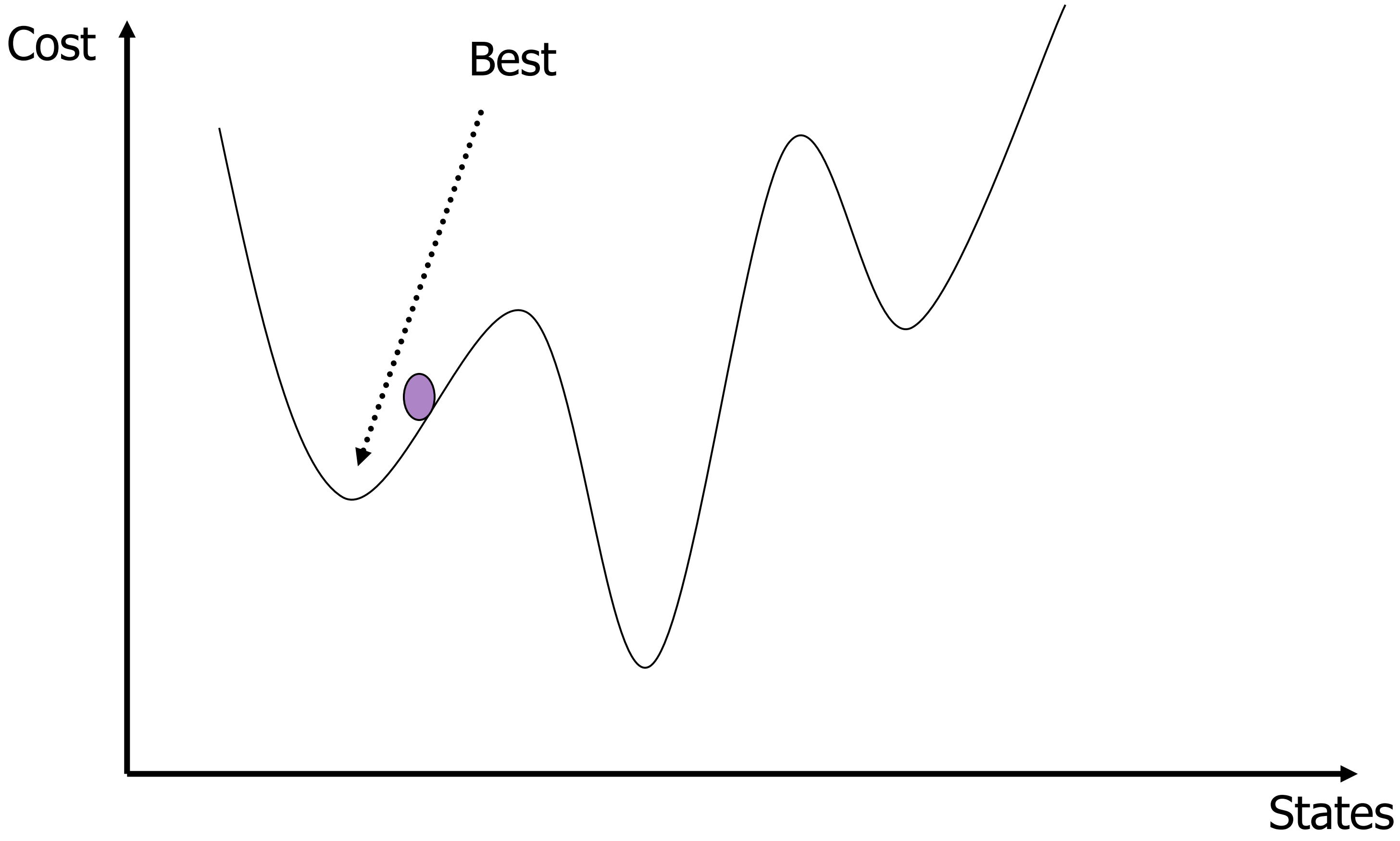
Simulated Annealing



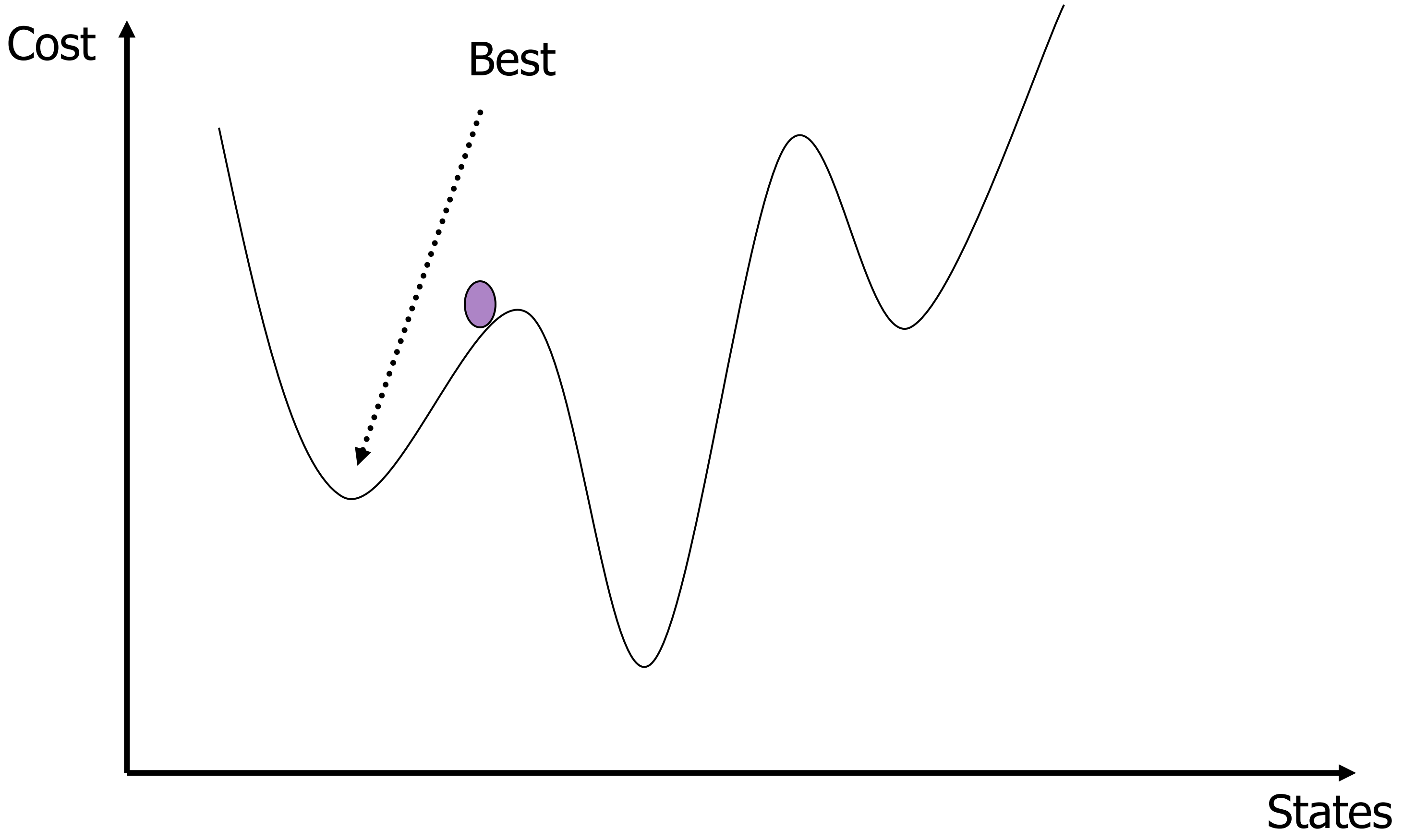
Simulated Annealing



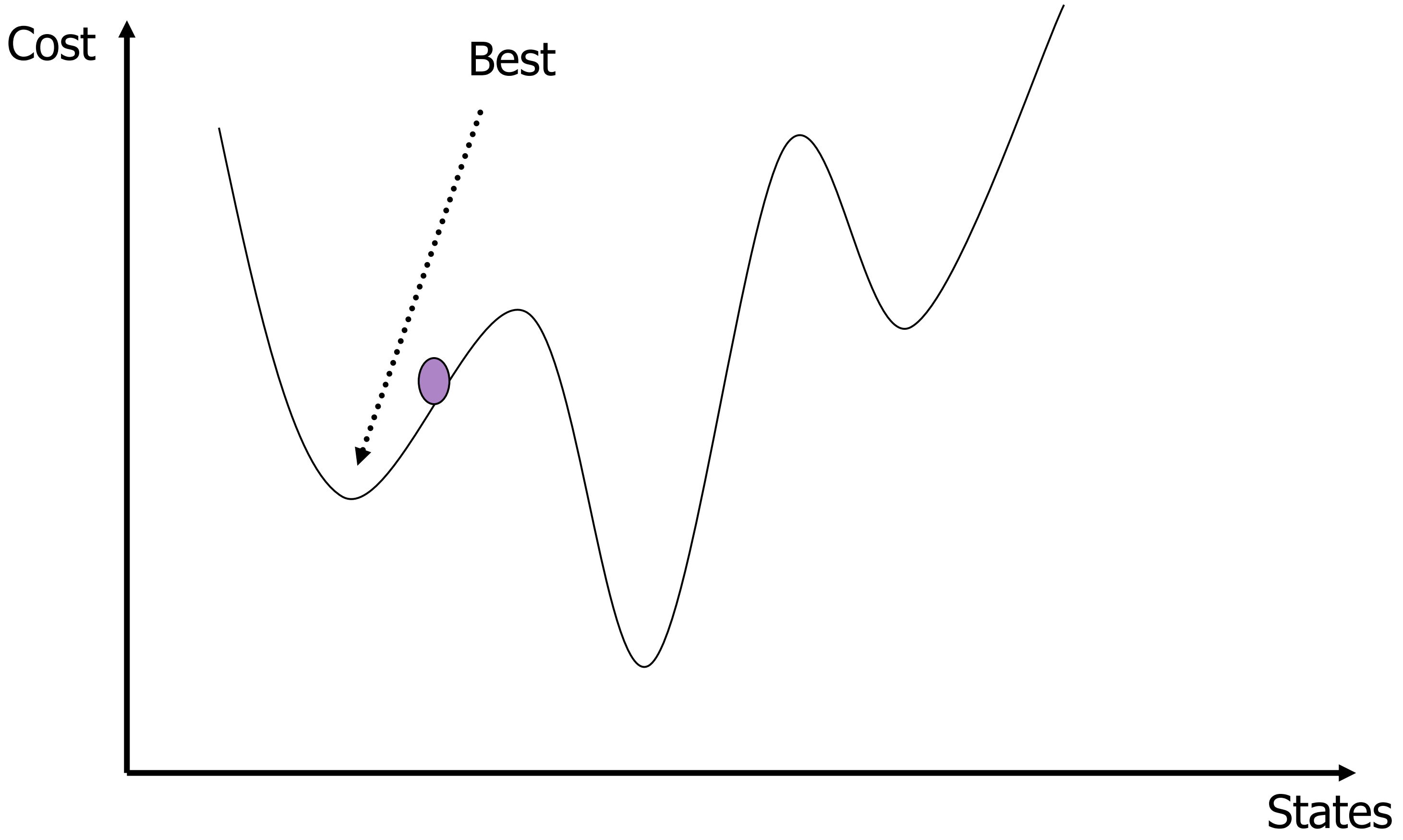
Simulated Annealing



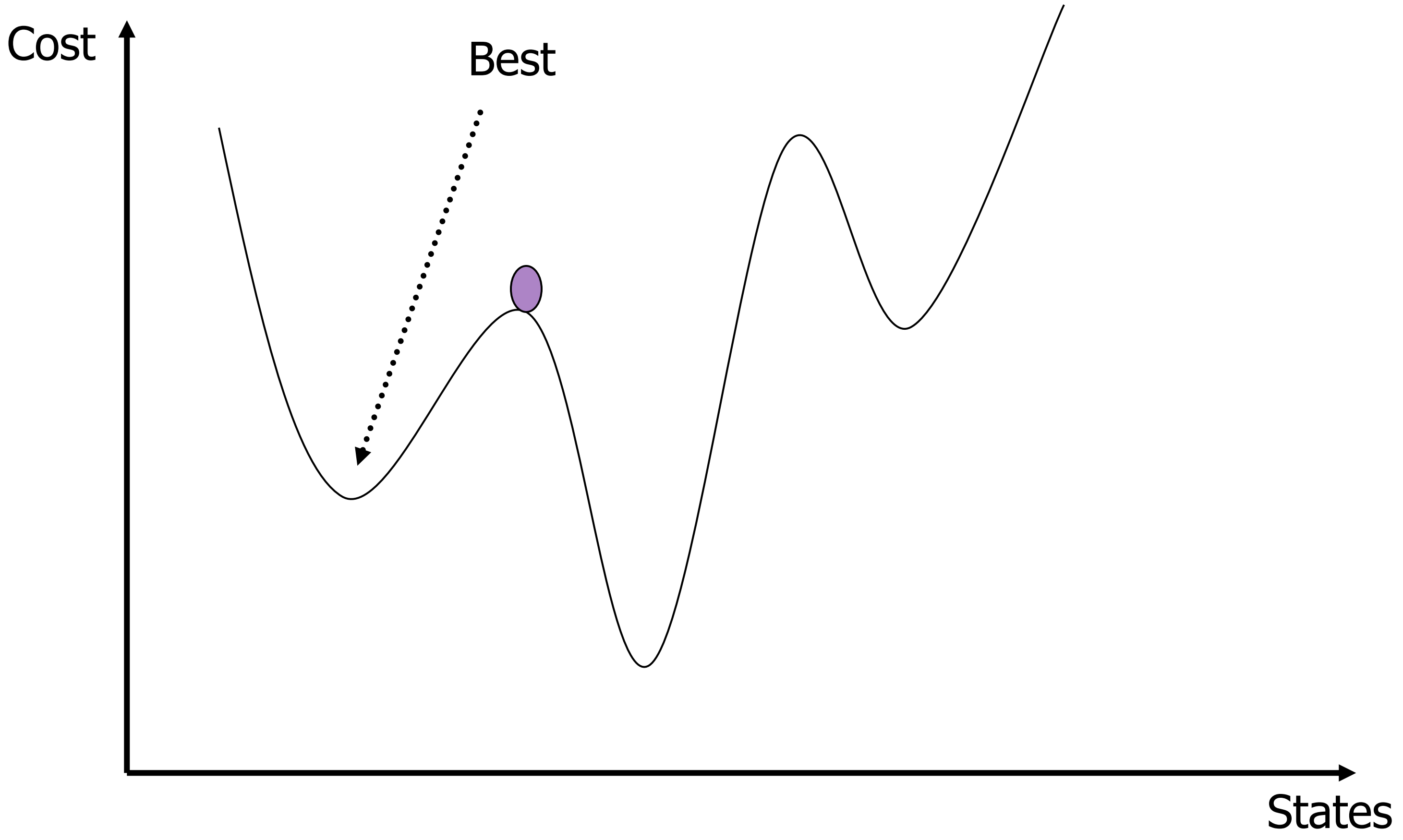
Simulated Annealing



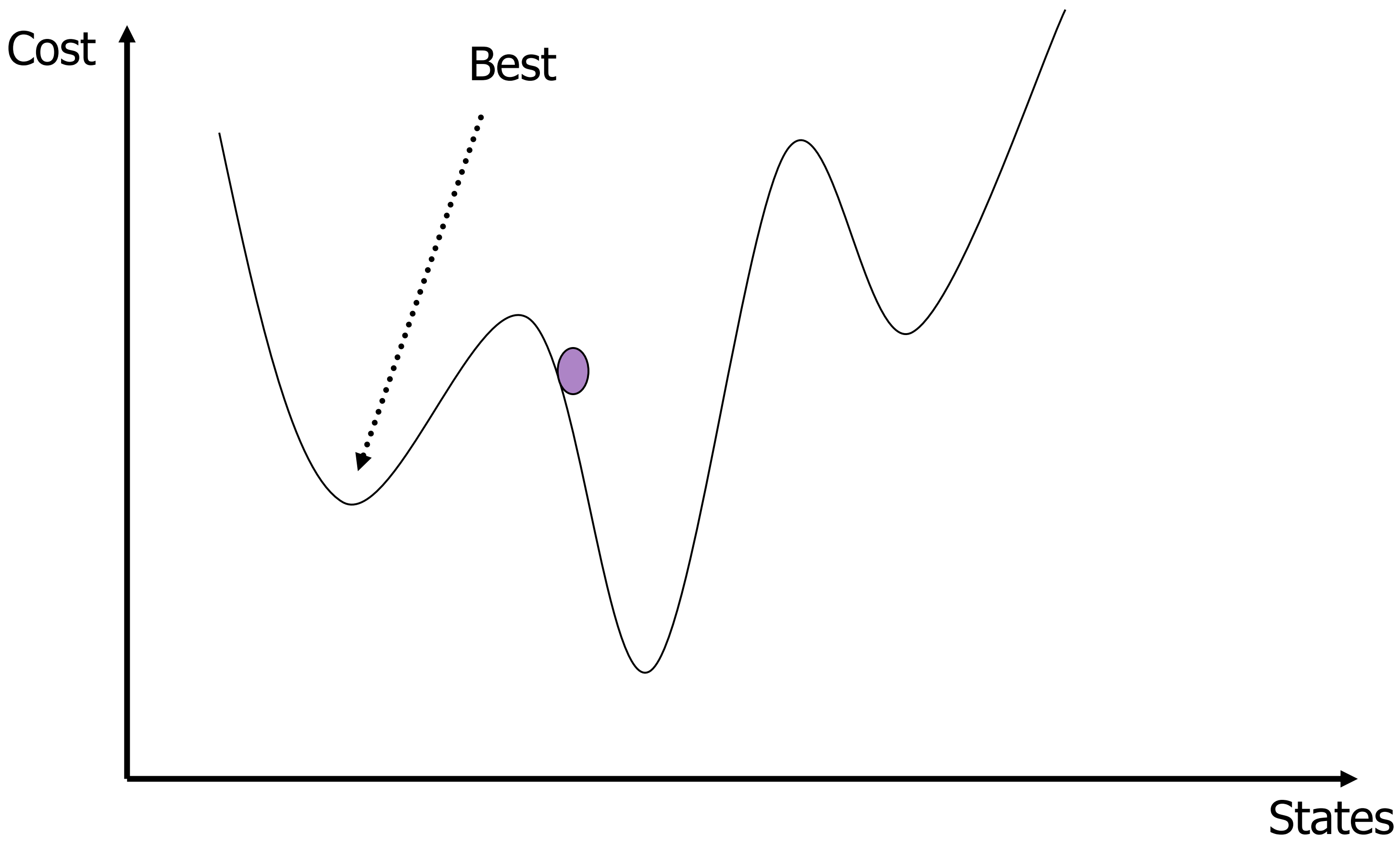
Simulated Annealing



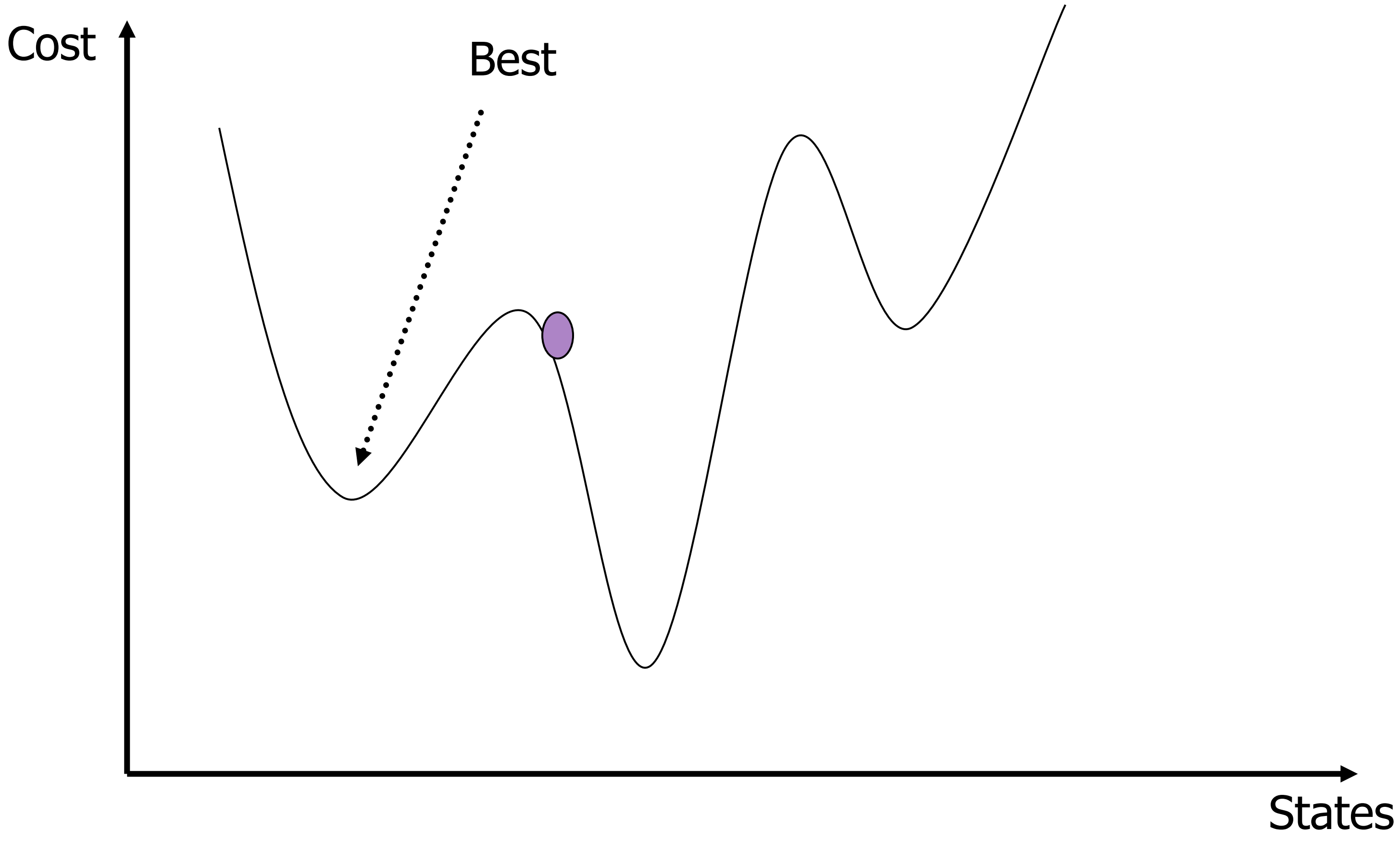
Simulated Annealing



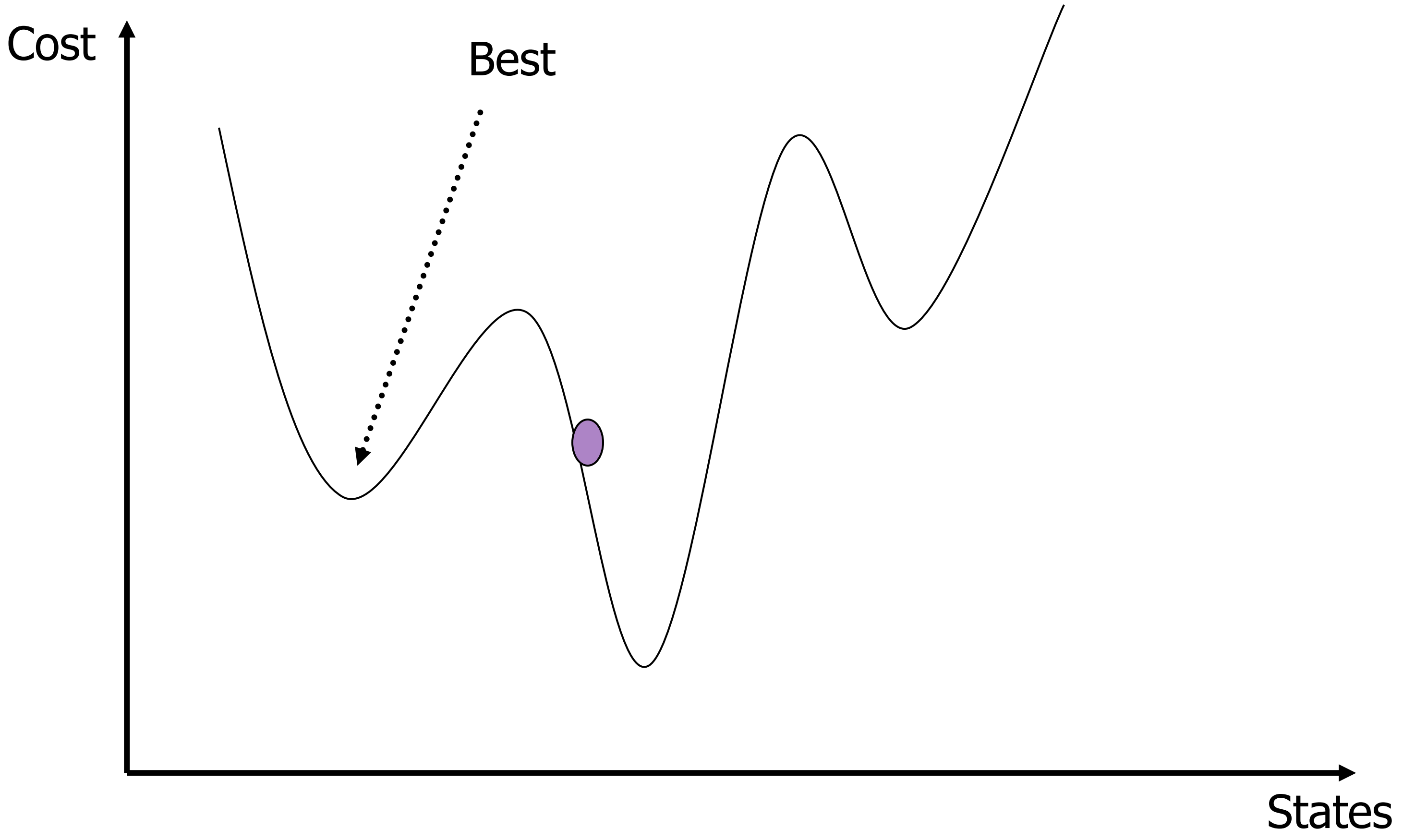
Simulated Annealing



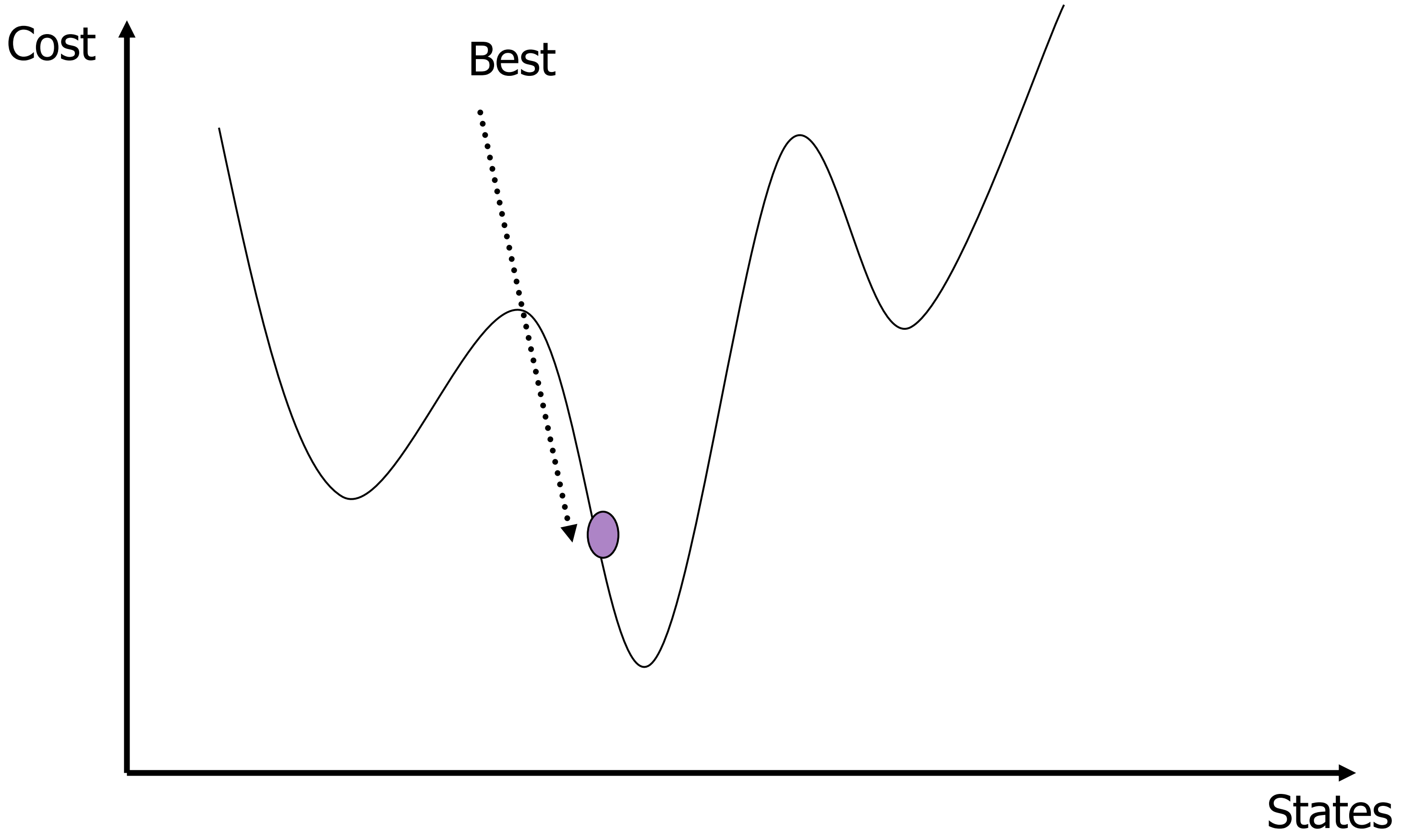
Simulated Annealing



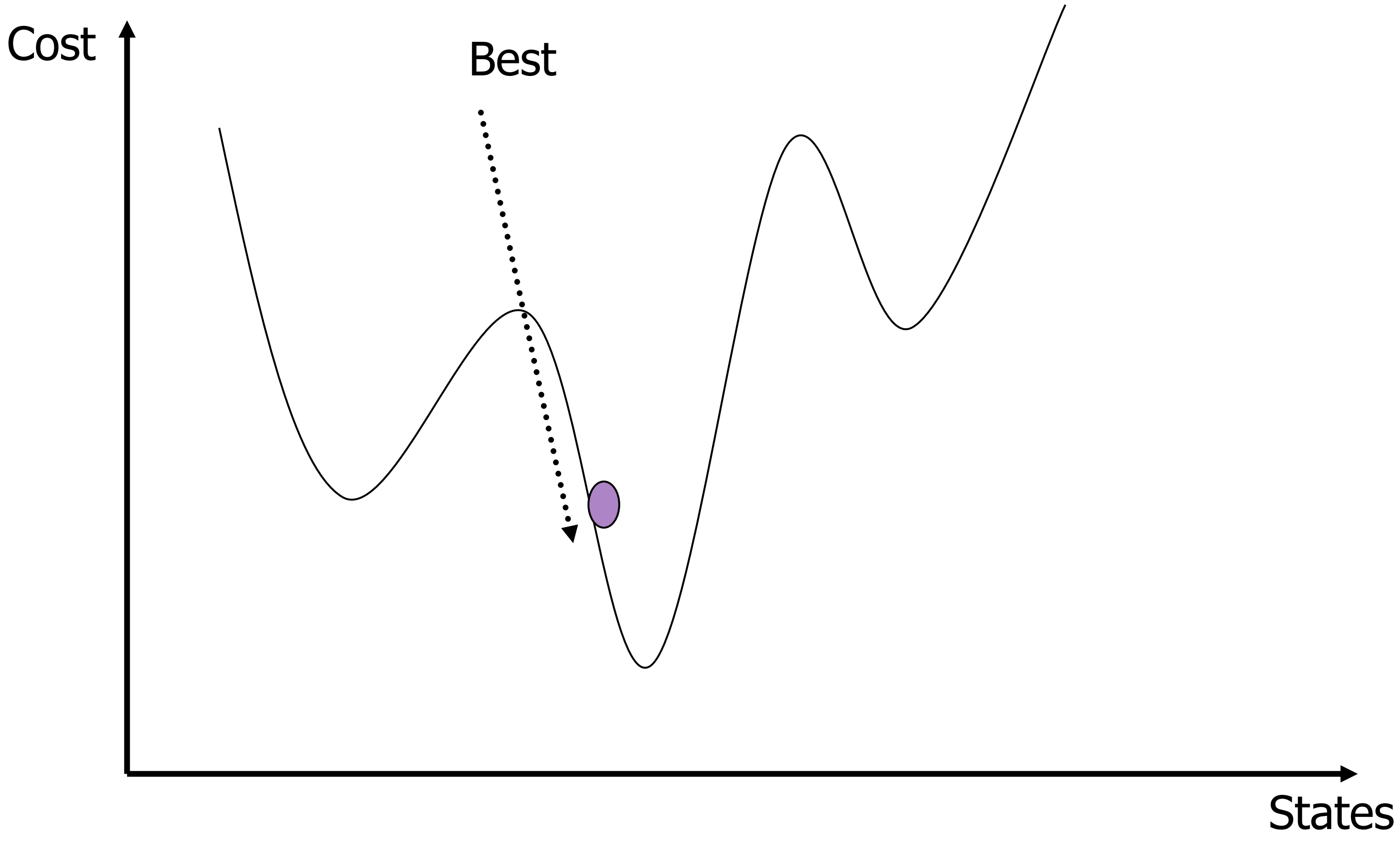
Simulated Annealing



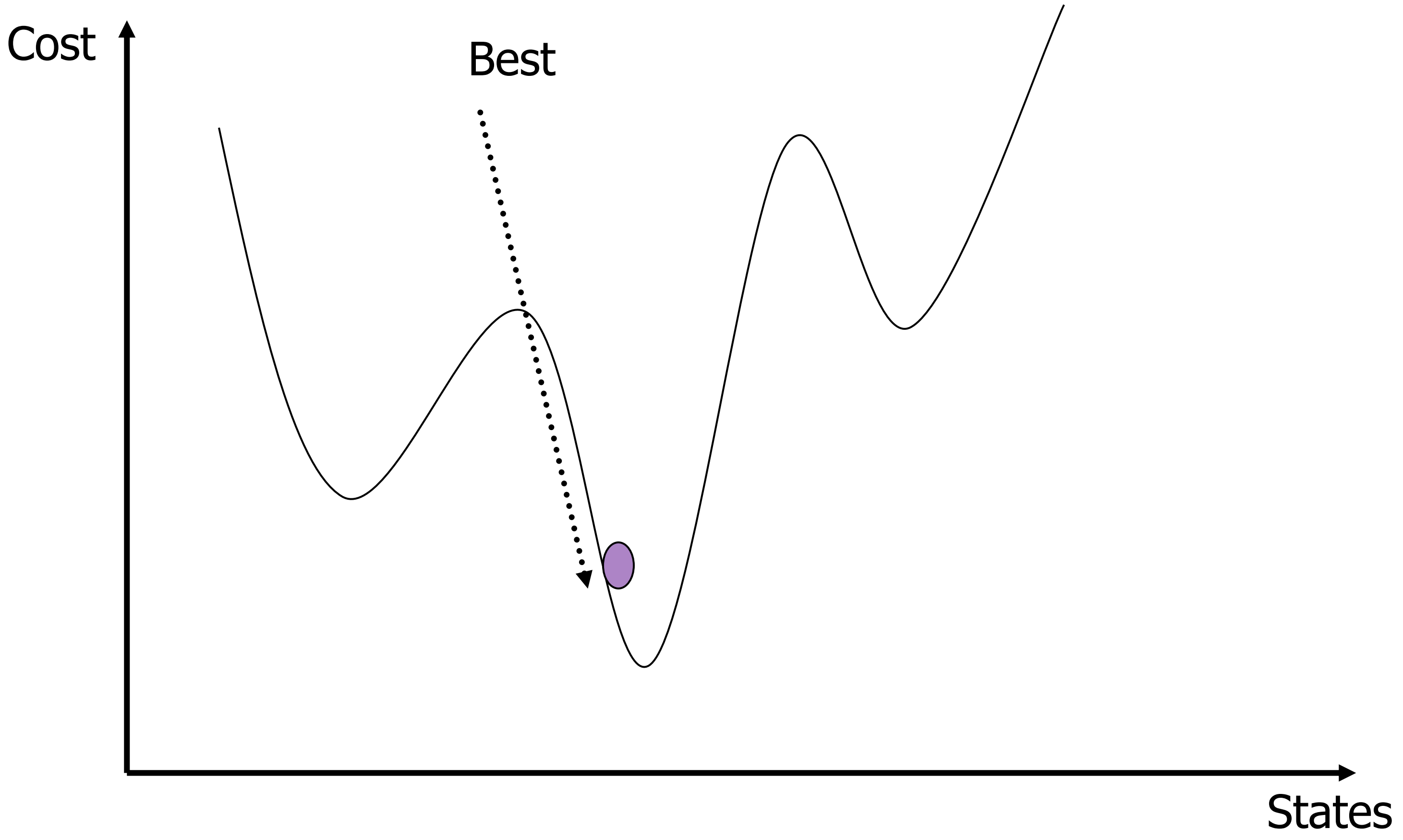
Simulated Annealing



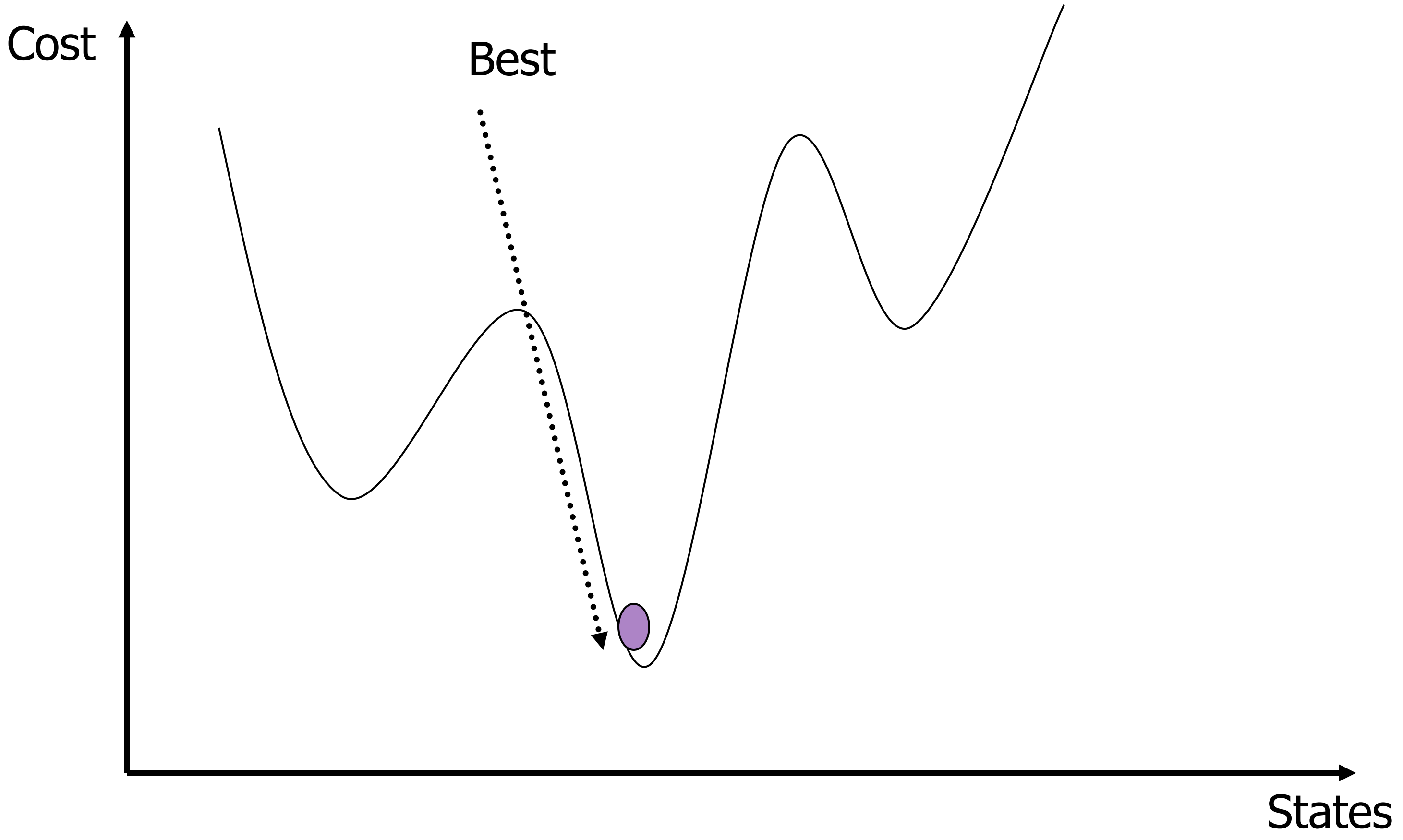
Simulated Annealing



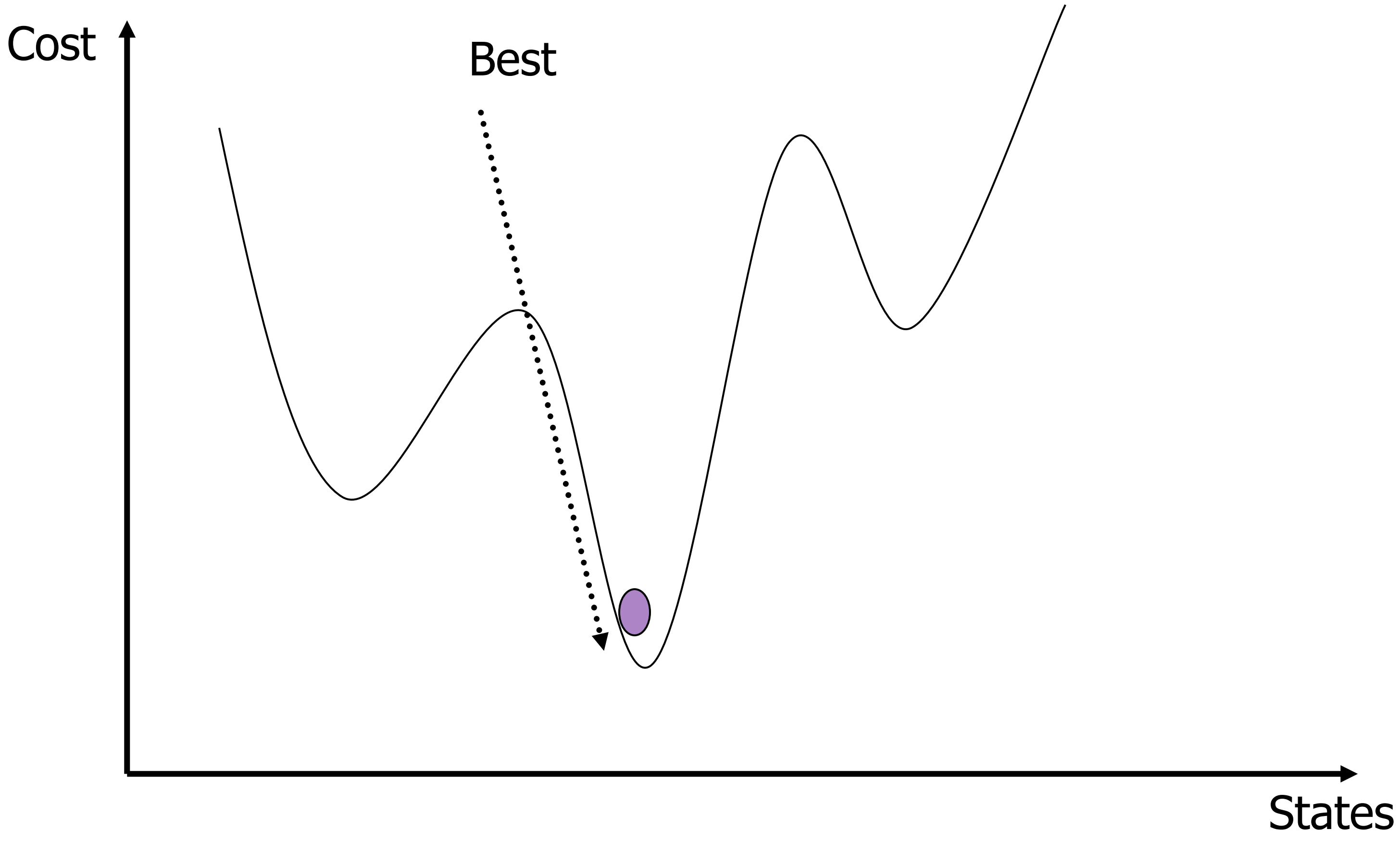
Simulated Annealing



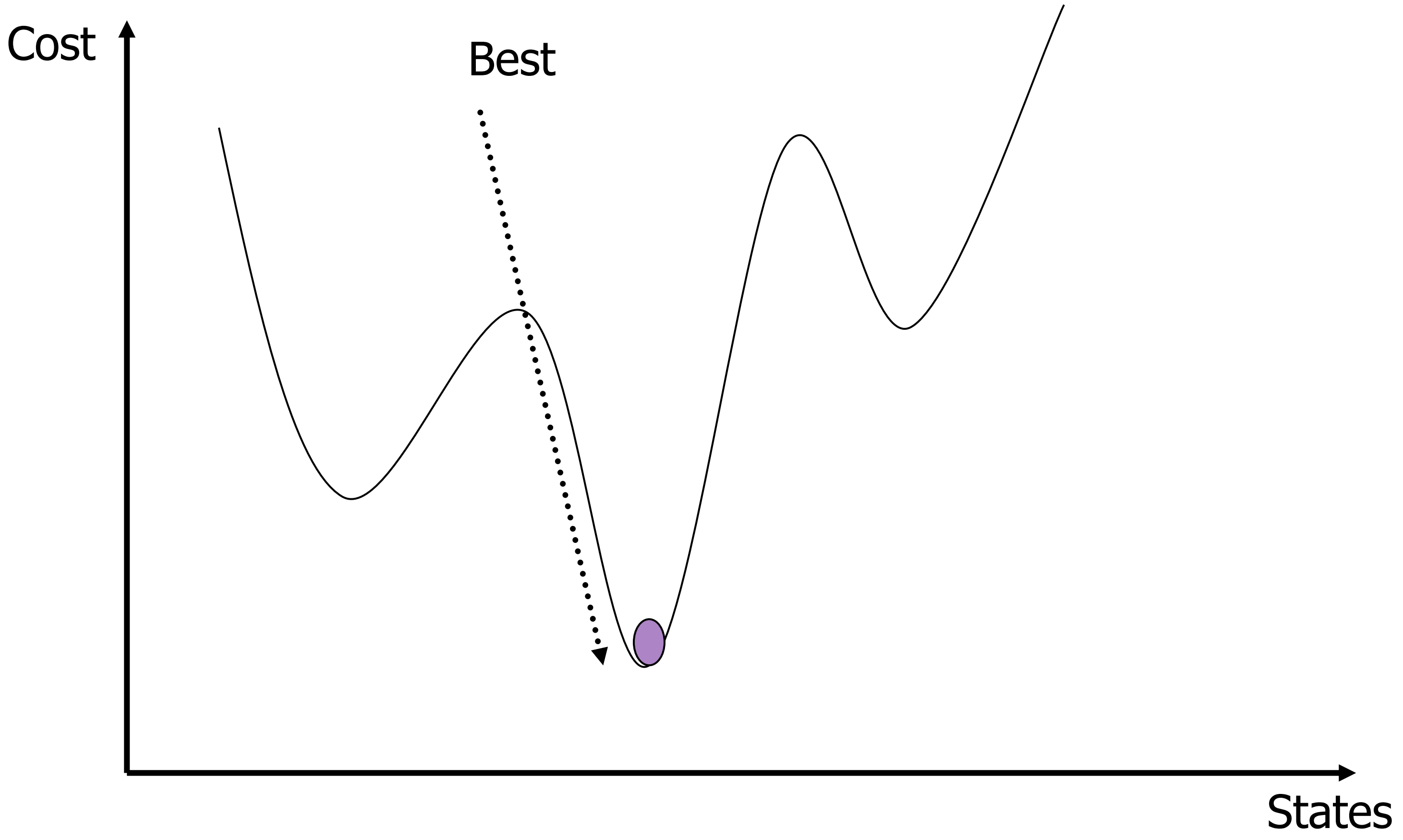
Simulated Annealing



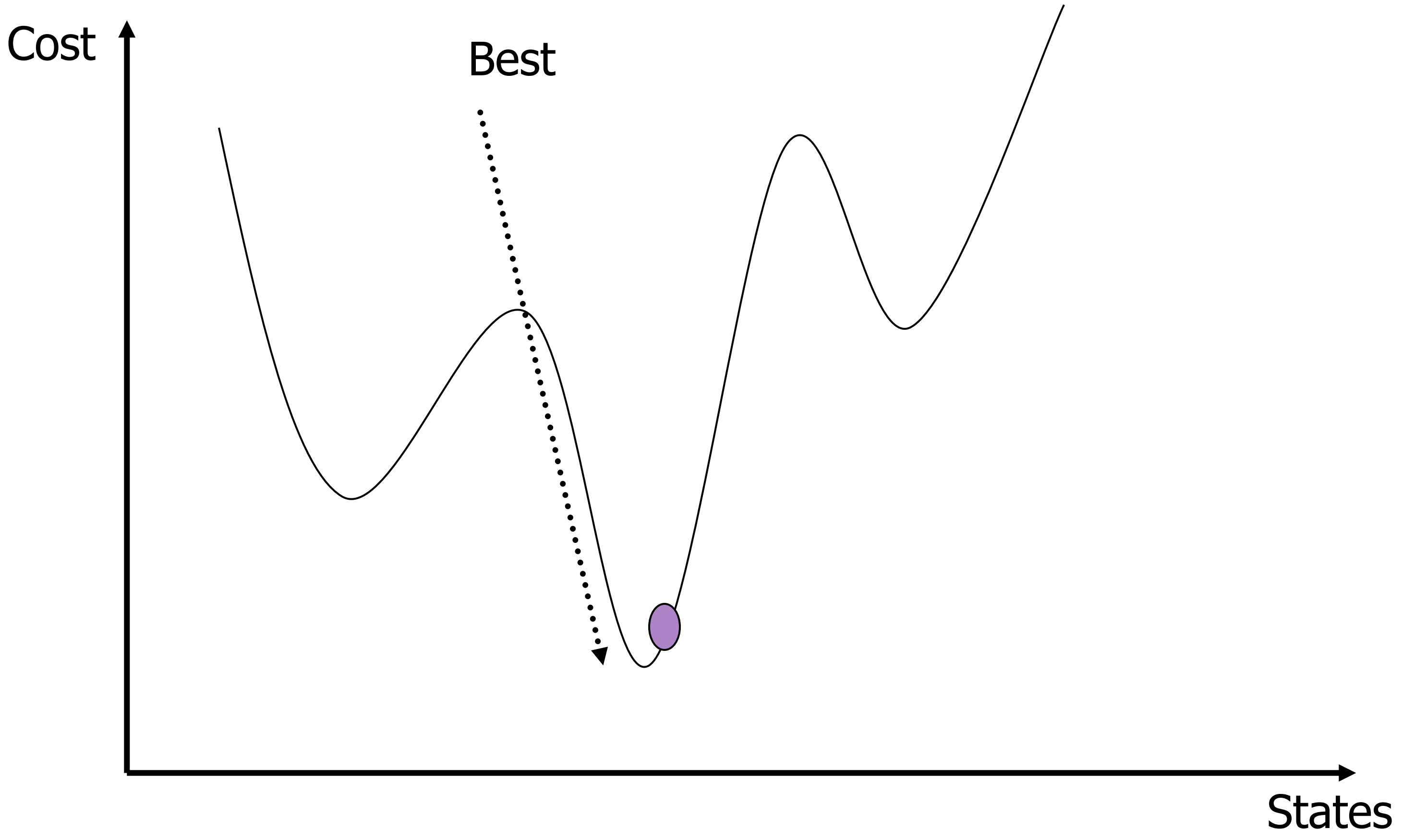
Simulated Annealing



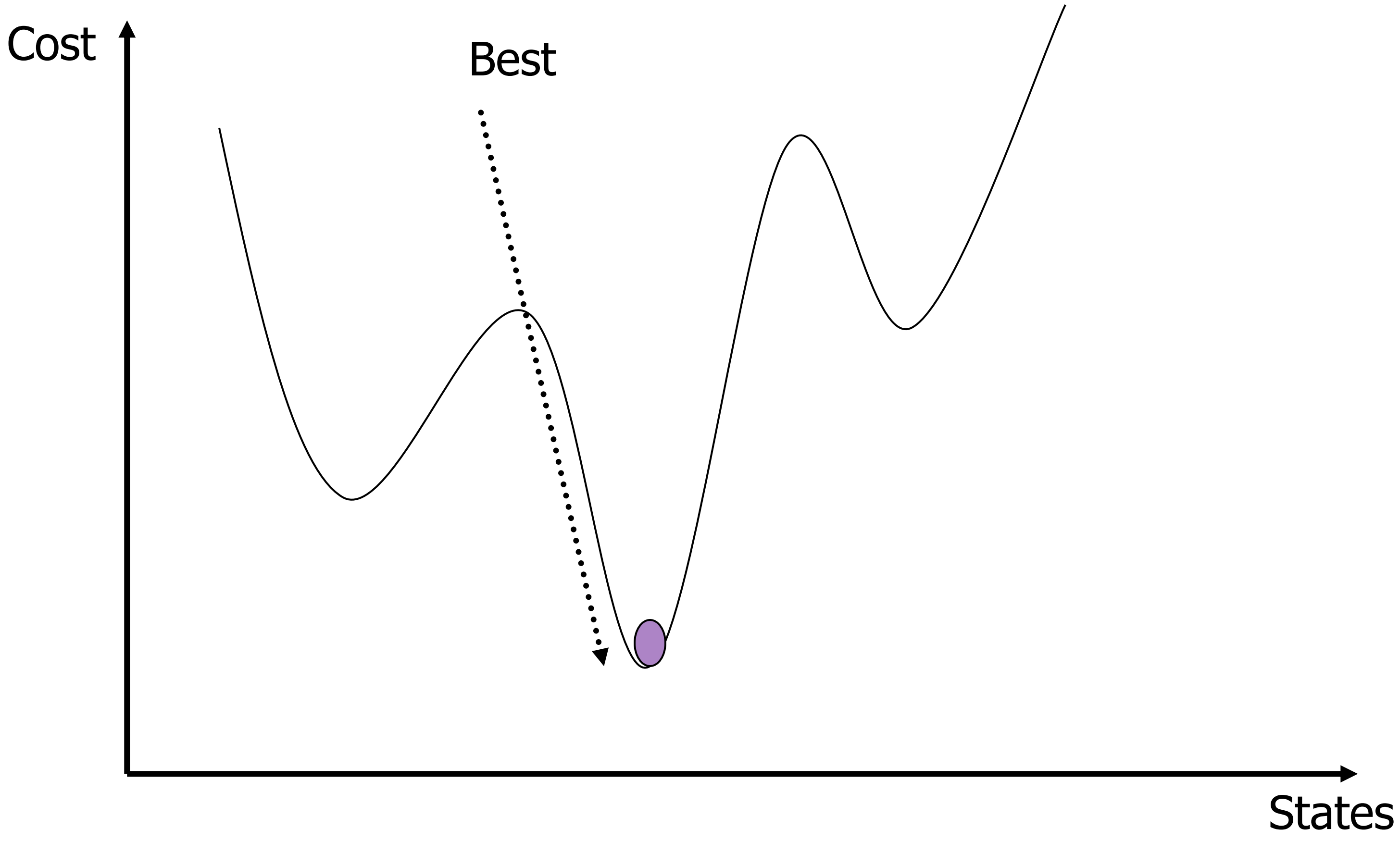
Simulated Annealing



Simulated Annealing



Simulated Annealing



Simulated Annealing