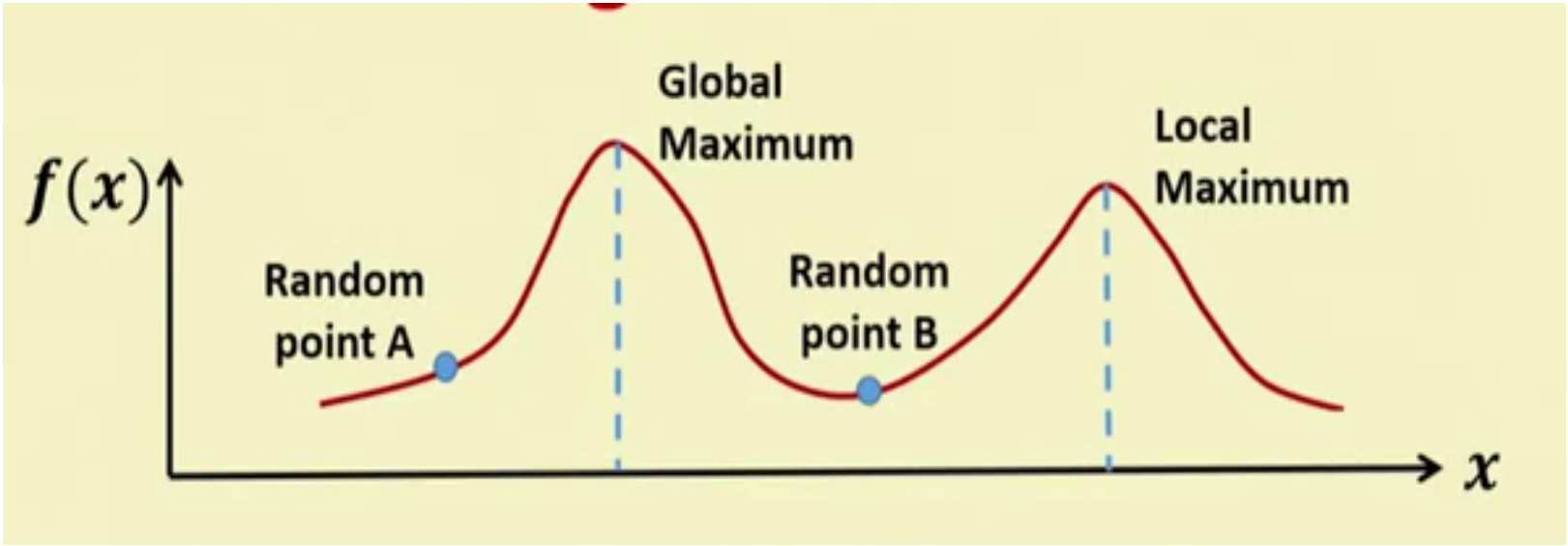
Simulated Annealing

Simulated Annealing Algorithm

- **Annealing:**
- Annealing is a thermal process for obtaining low energy states of a solid in a heat bath.
- The process contains two steps:
  - Increase the temperature of the heat bath to a maximum value at which the solid melts.
  - Decrease carefully the temperature of the heat bath until the particles arrange themselves in the ground state of the solid. Ground state is a minimum energy state of the solid.
- The ground state of the solid is obtained only if the maximum temperature is high enough and the cooling is done slowly

Simulated Annealing

- Simulated Annealing is a variant of the hill climbing method.
- However it allows downhill moves as well.
- Additionally it explores the search space in such a manner that the starting point does not influence much the final solution.



- SA is a heuristic solution generation process that uses logic and rules to iteratively change a suboptimal solution to a problem.
- It's based on the idea of slowly cooling a material to find the lowest energy state, or the most optimal solution.
- It is a technique used in AI to solve optimization problems. It's a probabilistic technique that approximates the global optimum of a function.
- We can apply this algorithm to generate a solution to combinatorial optimization problems assuming an analogy between them and physical many-particle systems with the following equivalences:
  - Solutions in the problem are equivalent to states in a physical system.
  - The cost of a solution is equivalent to the “energy” of a state.

Simulated Annealing

- Inspired by the Annealing process in which materials are raised to high energy levels for melting and are then cooled to solid state.
- The probability of moving to higher energy state, instead of lower is

$$p = e^{\frac{-\Delta E}{kT}}$$

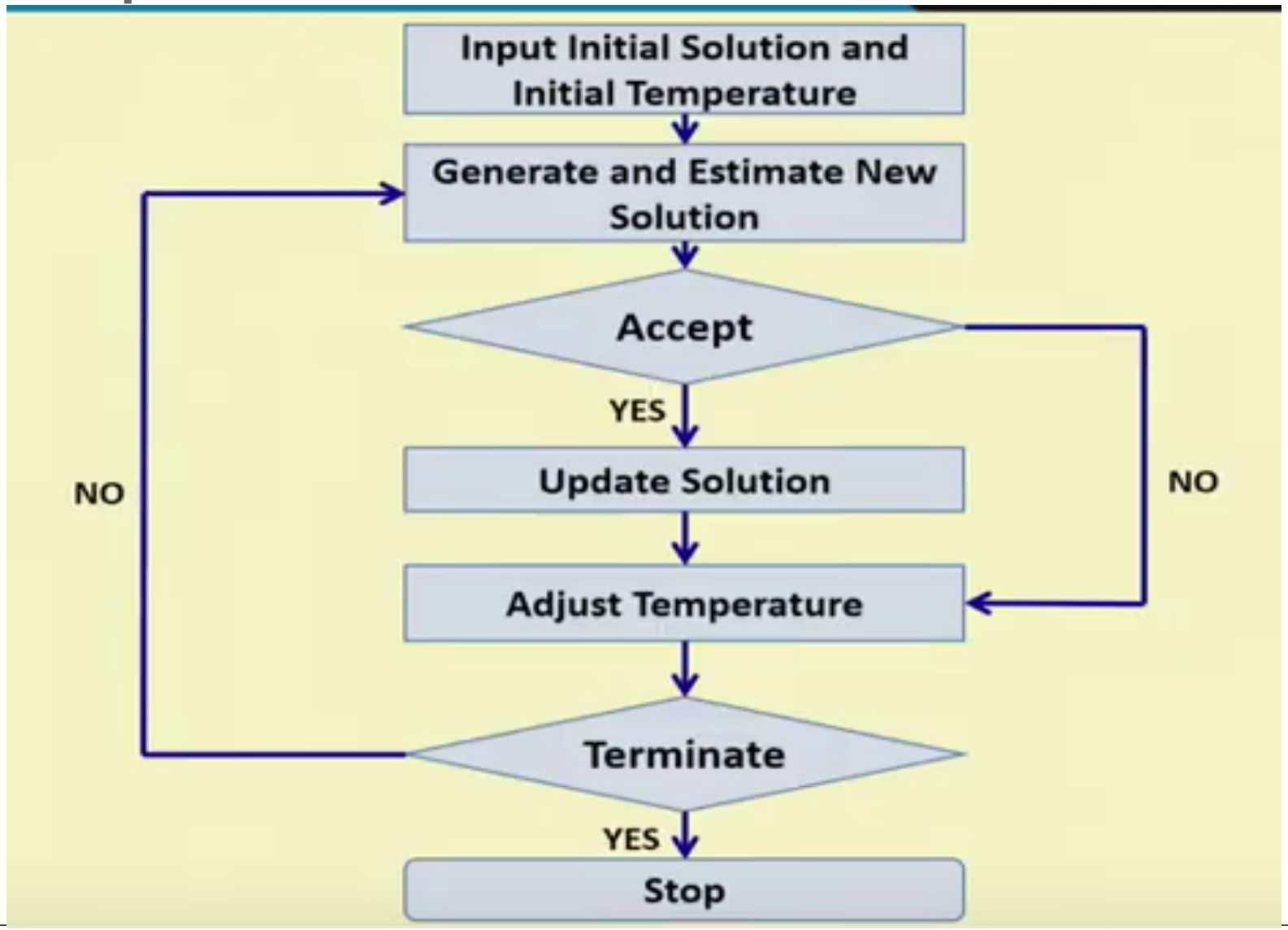
- Where ΔE is the positive change in energy level, T is the temperature and k is the Boltzmann's constant.
- Temperature is high at the beginning
- A temperature is lowered, probability of a downhill move gest smaller and smaller.
- If temperature is lowered very slowly, the best energy state is resulted.

- To apply simulated annealing with optimization purposes we require the following:
  - A successor function that returns a “close” neighboring solution given the actual one. This will work as the “disturbance” for the particles of the system.
  - A target function to optimize that depends on the current state of the system. This function will work as the energy of the system.
- The search is started with a randomized state. In a polling loop we will move to neighboring states always accepting the moves that decrease the energy while only accepting bad moves accordingly to a probability distribution dependent on the “temperature” of the system.

Analogy

Physical Annealing	Simulated Annealing
Metal	Optimization Problem
Energy State	Cost Function
Temperature	Control Parameter
Crystalline Structure	The optimal Solution
Global optima solution can be achieved as long as the cooling process is slow enough	

Steps in SA


Input Initial Solution and Initial Temperature
Generate and Estimate New Solution
Accept
YES
NO
NO
Update Solution
Adjust Temperature
Terminate
YES
Stop

How it works:

1. Initialize a random solution and evaluate its quality (energy or cost).
2. Set the initial temperature and perturbation size.
3. Generate a new solution by perturbing the current one.
4. Evaluate the new solution's quality.
5. Calculate the difference in quality (ΔE) between the new and current solutions.
6. If ΔE is positive (new solution is better), accept it.
7. If ΔE is negative (new solution is worse), accept it with a probability based on the temperature and ΔE (e.g., $\exp(\Delta E/T)$)
8. Decrease the temperature and perturbation size.
9. Repeat steps 3-8 until the stopping criterion is met.

Detailed steps of algorithm

- Initialization:
  - Start with an initial solution (S) and an initial temperature (T).
  - Define the cooling schedule, which determines how the temperature decreases over time.
- Perturbation:
  - Generate a new solution (S') by making a small random change to the current solution (S).
- Evaluation:
  - Calculate the energy (or cost) of the new solution (E(S')) and compare it to the energy of the current solution (E(S)).
- Acceptance Probability:
  - If (E(S') < E(S)), accept the new solution (S') (i.e., move to a lower energy state).
  - If (E(S') >= E(S)), accept the new solution with a probability (P) given by the Boltzmann distribution:

$$P = \exp\left(\frac{E(S) - E(S')}{T}\right)$$

- This probability allows the algorithm to accept worse solutions early on, helping it escape local minima.

- Cooling:
  - Reduce the temperature (T) according to the cooling schedule. Common schedules include:
  - Linear Cooling: [T(t) = T0 - (T0 - Tf) * (t / t_max)]
  - Exponential Cooling [T(t) = T0 * (Tf / T0)^(t / t_max)]
  - Logarithmic Cooling [T(t) = T0 / (1 + (T0 - Tf) * log(1 + t) / log(1 + t_max))
- Repeat the perturbation, evaluation, and acceptance steps until the system “freezes” (temperature approaches zero).
  - Where:
  - T(t) is the temperature at iteration t
  - T0 is the initial temperature
  - Tf is the final temperature (freezing temperature)-
  - t is the current iteration
  - t_max is the maximum number of iterations

Key Concepts

- **Energy Function**: Represents the objective function to be minimized. In optimization problems, this could be the cost, distance, error, etc.
- **Temperature**: Controls the probability of accepting worse solutions. Higher temperatures allow more exploration, while lower temperatures focus on exploitation.
- **Cooling Schedule**: Determines how quickly the temperature decreases. A slower cooling schedule allows more thorough exploration but increases computation time.

Applications

- **Traveling Salesman Problem (TSP)**: Finding the shortest route that visits a set of cities and returns to the origin.
- **Job-Shop Scheduling**: Optimizing the sequence of jobs in a manufacturing process to minimize completion time.
- **VLSI Design**: Optimizing the layout of circuits on a chip.
- **Protein Folding**: Predicting the three-dimensional structure of a protein from its amino acid sequence.
- **Graph Partitioning**: Dividing a graph into smaller components while minimizing the number of edges between them.

Advantages and Disadvantages

- **Advantages:**
  - **Escapes Local Minima**: By accepting worse solutions with a certain probability, SA can escape local minima and potentially find a global optimum.
  - **Flexibility**: Can be applied to a wide range of optimization problems without requiring specific knowledge of the search space.
  - **Simplicity**: Easy to implement and understand.
- **Disadvantages:**
  - **Computation Time**: Can be slow, especially with a slow cooling schedule.
  - **Parameter Sensitivity**: Performance depends on the choice of initial temperature, cooling schedule, and perturbation mechanism.

Example: Traveling Salesman Problem (TSP)

- Imagine a salesman needs to visit several cities and return to the starting point, minimizing the total travel distance. Here's how simulated annealing can be applied:
- **Initialization**: Start with a random route and a high temperature.
- **Perturbation**: Swap the order of two cities in the route to create a new route.
- **Evaluation**: Calculate the total distance of the new route.
- **Acceptance**: Accept the new route if it's shorter. If it's longer, accept it with a probability based on the temperature.
- **Cooling**: Gradually reduce the temperature and repeat the process until the temperature is very low.

- Problem Setup:
- We have five cities with the following distances between them:

City	A	B	C	D	E
A	0	2	9	10	7
B	2	0	6	4	3
C	9	6	0	8	5
D	10	4	8	0	6
E	7	3	5	6	0

- Steps of Simulated Annealing
- Initialization:
  - Start with an initial route: A → B → C → D → E → A.
  - Initial temperature (T = 100).
  - Cooling rate (alpha = 0.95).
- Perturbation:
  - Swap two cities in the route to create a new route. For example, swap B and D: A → D → C → B → E → A.

- Evaluation:
  - Calculate the total distance of the new route.
  - Original route distance: (2 + 6 + 8 + 6 + 7 = 29).
  - New route distance: (10 + 8 + 6 + 3 + 7 = 34).
- **Acceptance**:
- Since the new route is longer, calculate the acceptance probability:

$$P = \exp\left(\frac{29 - 34}{100}\right) = \exp(-0.05) \approx 0.9512$$

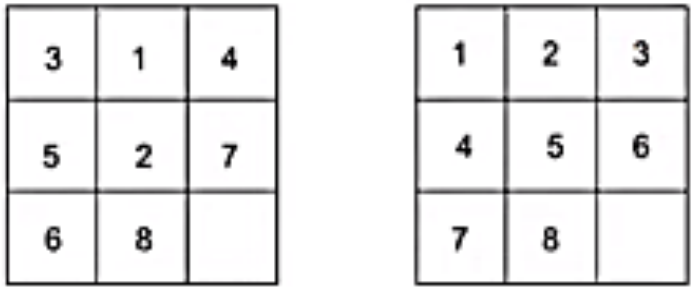
- Generate a random number between 0 and 1. If the number is less than 0.9512, accept the new route.
- Cooling:
  - Reduce the temperature: (T = T \times \alpha = 100 \times 0.95 = 95).
  - Repeat the perturbation, evaluation, and acceptance steps until the temperature is very low.

- Iteration 1
  - **Current Route**: A → B → C → D → E → A
  - **New Route**: A → D → C → B → E → A
  - **Distance:** 34
  - **Acceptance Probability**: 0.9512
  - **Random Number**: 0.8 (accept the new route)
  - **New Temperature**: 95
- Iteration 2
  - **Current Route**: A → D → C → B → E → A
  - **New Route**: A → D → E → B → C → A
  - **Distance**: 10 + 6 + 3 + 6 + 9 = 34
  - **Acceptance Probability**: 1 (since the distance is the same)
  - **Random Number**: 0.5 (accept the new route)
  - **New Temperature**: 95 × 0.95 = 90.25

- Iteration 3
  - **Current Route**: A → D → E → B → C → A
  - **New Route**: A → E → D → B → C → A
  - **Distance**: 7 + 6 + 4 + 6 + 9 = 32
  - **Acceptance Probability**: 1 (since the new distance is shorter)
  - **New Temperature**: 90.25 × 0.95 = 85.7375

- Final Solution
- After many iterations, the algorithm will converge to a near-optimal solution.
- For example, it might find the route A → B → E → D → C → A with a total distance of 24.

Example

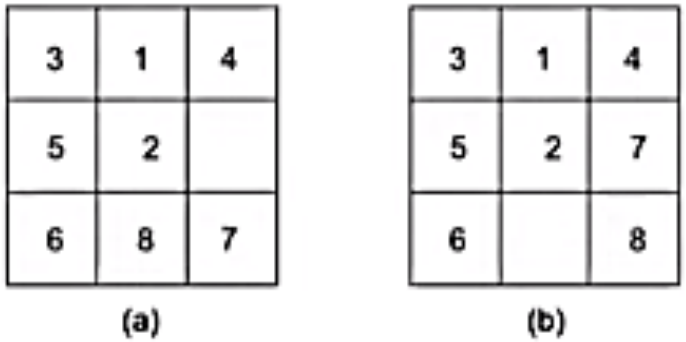


(a) Initial state, (b) Goal stat

Energy for the initial states is given as:

$(2 - 1) + (5 - 2) + (3 - 1) + (4 - 3) + (5 - 4) + (7 - 6) + (7 - 6) + (8 - 8) = 1 + 3 + 2 + 1 + 1 + 1 + 1 + 0 = 10$

We are aware that there are two different cases possible after this state. So, we have them, as shown in Figure 4.13 below.



(a) Energy 12, (b) Energy 11.

- Here energy function is calculated by variable position, where it is present and where should be. Like 1 is present at 2nd position but it should be in the first position. So, it is 2-1.

Example

- Let's consider another example of simulated annealing, the Knapsack Problem.
- Problem Setup
- In the Knapsack Problem, you have a set of items, each with a weight and a value, and a knapsack with a maximum weight capacity. The goal is to maximize the total value of the items in the knapsack without exceeding the weight capacity.

Item	Weight	Value
1	2	3
2	3	4
3	4	5
4	5	8
5	9	10

Knapsack Capacity
Maximum weight capacity: 15

- Steps of Simulated Annealing
- Initialization:
  - Start with an initial solution (a random selection of items). For example, select items 1, 2, and 3.
  - Initial temperature (T = 100).
  - Cooling rate (alpha = 0.95).
- Perturbation:
  - Generate a new solution by adding or removing an item. For example, remove item 3 and add item 4.
- Evaluation:
  - Calculate the total weight and value of the new solution.
  - Original solution: Items 1, 2, 3 (Weight: 2 + 3 + 4 = 9, Value: 3 + 4 + 5 = 12).
  - New solution: Items 1, 2, 4 (Weight: 2 + 3 + 5 = 10, Value: 3 + 4 + 8 = 15).

- **Acceptance**:
  - If the new solution is better (higher value without exceeding weight), accept it.
  - If the new solution is worse, accept it with a probability (P) given by:

$$P = \exp\left(\frac{\text{Value}_{\text{new}} - \text{Value}_{\text{current}}}{T}\right)$$

  - This allows the algorithm to explore different solutions and escape local optima.
- **Cooling**:
  - Reduce the temperature: (T = T \times \alpha).
  - Repeat the perturbation, evaluation, and acceptance steps until the temperature is very low.

- **Iteration 1**
  - **Current Solution**: Items 1, 2, 3 (Weight: 9, Value: 12)
  - **New Solution**: Items 1, 2, 4 (Weight: 10, Value: 15)
  - **Acceptance Probability**: 1 (since the new value is higher)
  - **New Temperature**: $100 \times 0.95 = 95$
- **Iteration 2**
  - **Current Solution**: Items 1, 2, 4 (Weight: 10, Value: 15)
  - **New Solution**: Items 1, 3, 4 (Weight: $2 + 4 + 5 = 11$, Value: $3 + 5 + 8 = 16$)
  - **Acceptance Probability**: 1 (since the new value is higher)
  - **New Temperature**: $95 \times 0.95 = 90.25$
- **Iteration 3**
  - **Current Solution**: Items 1, 3, 4 (Weight: 11, Value: 16)
  - **New Solution**: Items 2, 3, 4 (Weight: $3 + 4 + 5 = 12$, Value: $4 + 5 + 8 = 17$)
  - **Acceptance Probability**: 1 (since the new value is higher)
  - **New Temperature**: $90.25 \times 0.95 = 85.7375$

- Final Solution
- After many iterations, the algorithm will converge to a near-optimal solution.
- For example, it might find the solution with items 2, 3, 4, and 5 (Weight: 3 + 4 + 5 + 9 = 21, Value: 4 + 5 + 8 + 10 = 27), but since this exceeds the weight capacity, it will settle on the best feasible solution within the constraints.