Metaheuristic Optimization Methods: Simulated Annealing

Simulated Annealing

- A metaheuristic inspired by statistical thermodynamics
- Used in optimization for 20+ years
- Very simple to implement
- A lot of literature
- Converges to the global optimum under weak assumptions (- usually slowly)

Annealing

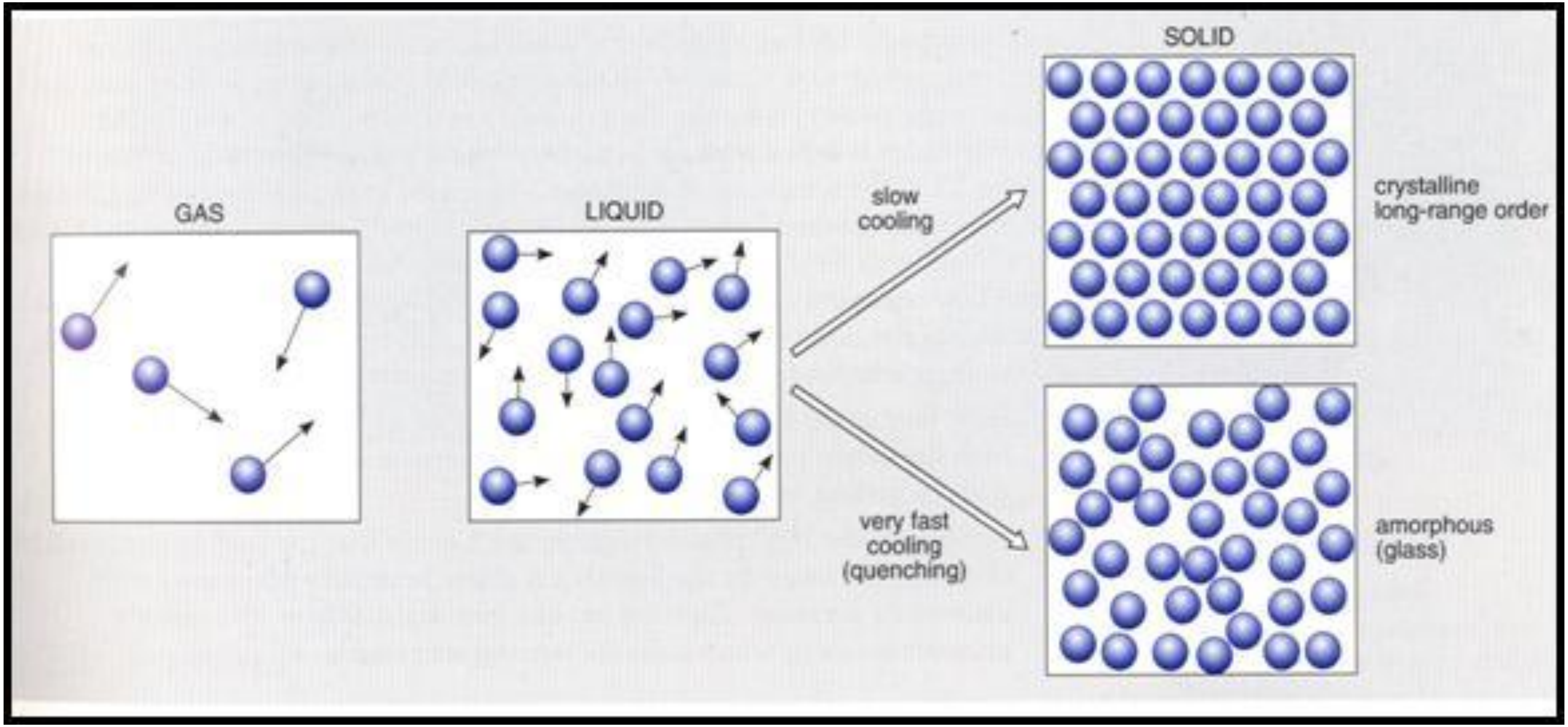
Annealing is the process of heating a solid until thermal stresses are released. Then, in cooling it very slowly to the ambient temperature until perfect crystals emerge.

The quality of the results strongly depends on the cooling process. The final state can be interpreted as an energy state (crystaline potential energy) which is lowest if a perfect crystal emerges.

Top 3 Crystals for Energy Healing,
Stress Management

Simulated Annealing - SA

- Metropolis' algorithm (1953)
  - A solid material is heated past its melting point and then cooled back into a solid state (annealing).
  - The final structure depends on how the cooling is performed
    - slow cooling → large crystal (low energy)
    - fast cooling → imperfections (high energy)


GAS
LIQUID
slow cooling
very fast cooling (quenching)
SOLID
crystalline long-range order
amorphous (glass)

Simulated Annealing - SA

- Metropolis’ algorithm simulates the change in energy of the system when subjected to the cooling process; the system converges to a final “frozen” state of a certain energy.
- Kirkpatrick, Gelatt and Vecchi (1983)
  - The Metropolis simulation can be used to explore the feasible solutions of a problem with the objective of converging to an optimal solution.

SA - Analogy

Thermodynamics	Discrete optimization
1. Configuration of particles	1. Solution
2. System state	2. Feasible solution
3. Energy	3. Objective Function
4. State change	4. Move to neighbor solution
5. Temperature	5. Control Parameter
6. Final state	6. Final Solution

Thermodynamics

- In thermodynamics, the probability *p* to go from a state with energy E_1 to a state of energy E_2 is given by

$$p = e^{\frac{-(E_1 - E_2)}{kT}}$$

where k is a constant and T is the temperature.

- The idea is that probability decreases exponentially with E2−E1 increasing; the probability also decreases as temperature decreases.

Simulated Annealing (using minimization)

- Idea: **escape local minima by allowing some bad moves but gradually decrease their size and frequency**
- Instead of picking the best move from the neighborhood of current solution, pick a random move from the neighborhood.
- If the move improves objective, it is accepted. Otherwise, accept the move with probability:

$$p = e^{\frac{-(f(s_1) - f(s_2))}{T}}$$

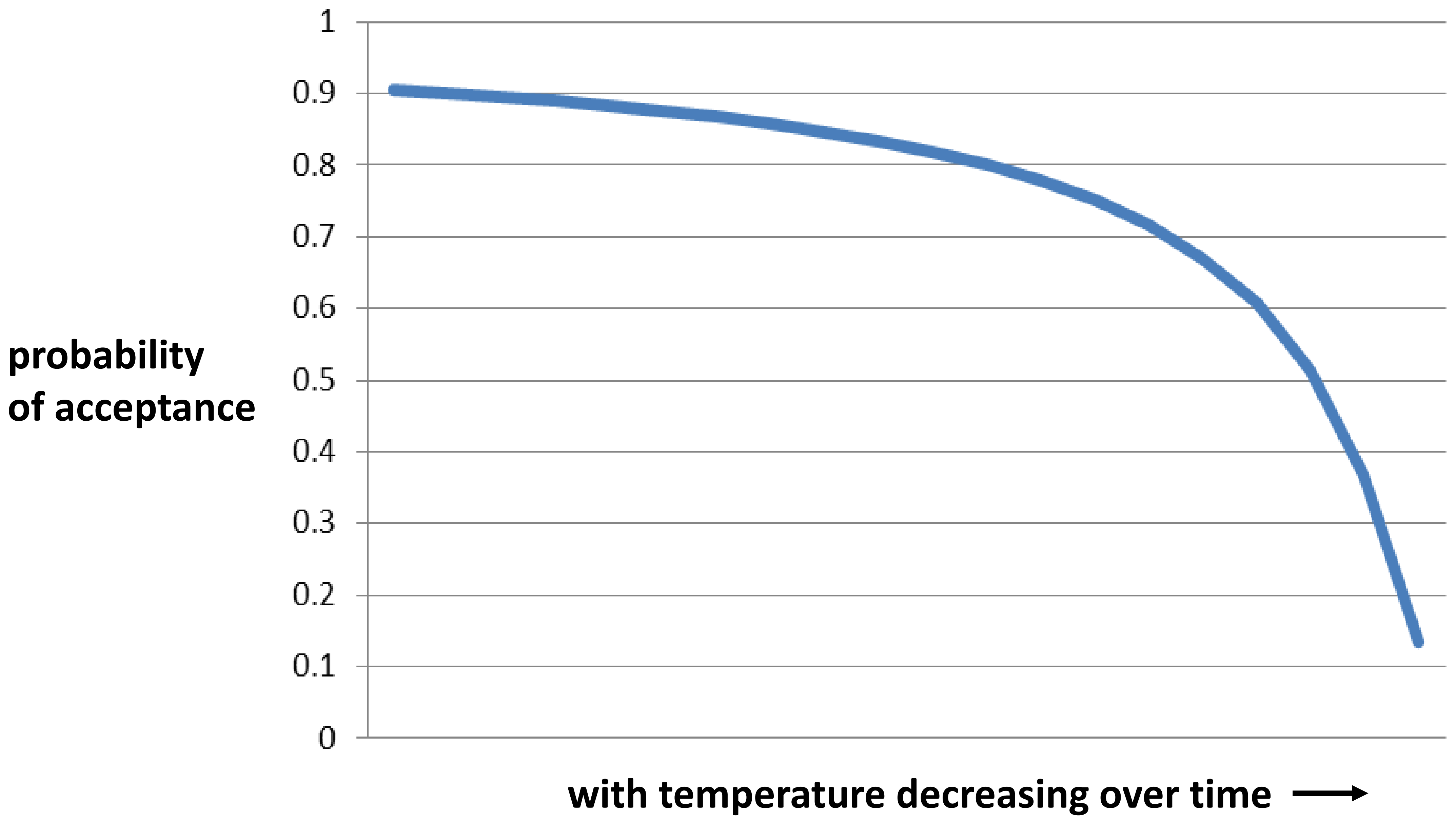
where T decreases according a pre-defined schedule as iterations increase

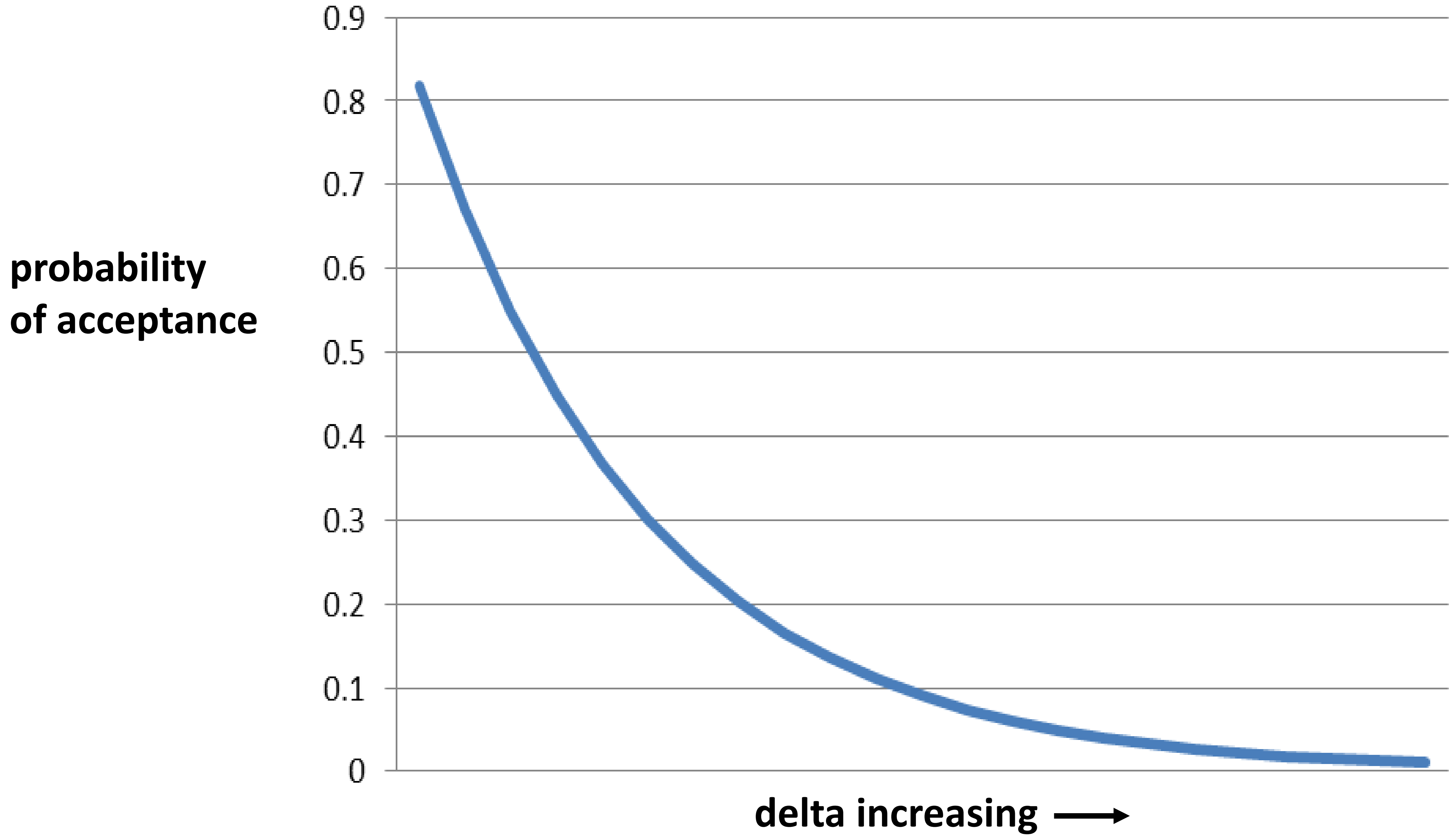
Simulated Annealing

Can be interpreted as a modified random descent in the space of solutions

1. Choose a *random* neighbor
2. Improving moves are always accepted
3. Deteriorating moves are accepted with a probability that depends on the amount of the deterioration and on the *temperature* (a parameter that decreases with time)

Can escape local optima


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
probability of acceptance
with temperature decreasing over time


probability of acceptance
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
delta increasing

Basic Simulated Annealing Structure

- Initial temperature t_0 high

  Note: if $\infty \rightarrow$ random walk
- Reduce t regularly: need a *cooling schedule*
  - if too fast $\rightarrow$ stop in some local optimum too early
  - if too slow $\rightarrow$ too slow convergence
- Do a certain number of iterations with the same temperature
- Stopping criterion – e.g., minimum temperature or max number of iterations
  - might consider restarts

Simulated Annealing

```
1:  input: starting solution, $s_0$
2:  input: neighborhood operator, $N$
3:  input: evaluation function, $f$
4:  input: the cooling schedule, $t_k$
5:  input: the number of iterations for each temperature, $M_k$
6:  $current \Leftarrow s_0$
7:  $k \Leftarrow 0$
8:  while stopping criterion not met do
9:      $m \Leftarrow 0$
10:     while $m < M_k$ do
11:         $s \Leftarrow$ randomly selected solution from $N(current)$
12:         if $f(s) \leq f(current)$ then
13:             $current \Leftarrow s$
14:         else
15:             $\Delta \Leftarrow f(s) - f(current)$
16:             $\xi \Leftarrow$ a random number, uniformly drawn from $[0, 1]$
17:             if $\xi \leq e^{-\Delta/t_k}$ then
18:                 $current \Leftarrow s$
19:             end if
20:         end if
21:         $m \Leftarrow m + 1$
22:     end while
23:     $k \Leftarrow k + 1$
24: end while
```

Choose solution

Allow to choose bad moves

Simulated Annealing

```
input: starting solution, $\boxed{s_0}$
input: neighborhood operator, $\boxed{N}$
input: evaluation function, $\boxed{f}$
input: the cooling schedule, $\boxed{t_k}$
input: the number of iterations for each temperature, $\boxed{M_k}$
$current \Leftarrow s_0$
$k \Leftarrow 0$
while $\boxed{\text{stopping criterion not met}}$ do
    $m \Leftarrow 0$
    while $m < M_k$ do
        $s \Leftarrow$ randomly selected solution from $N(current)$
        if $f(s) \leq f(current)$ then
            $current \Leftarrow s$
        else
            $\Delta \Leftarrow f(s) - f(current)$
            $\xi \Leftarrow$ a random number, uniformly drawn from $[0, 1]$
            if $\xi \leq e^{-\Delta/t_k}$ then
                $current \Leftarrow s$
```

```
 8: while stopping criterion not met do
 9:   $m \Leftarrow 0$
10:   while $m < M_k$ do
11:     $s \Leftarrow$ randomly selected solution from $N(current)$
12:     if $f(s) \leq f(current)$ then
13:       $current \Leftarrow s$
14:     else
15:       $\Delta \Leftarrow f(s) - f(current)$
16:       $\xi \Leftarrow$ a random number, uniformly drawn from $[0, 1]$
17:       if $\xi \leq e^{-\Delta/t_k}$ then
18:         $current \Leftarrow s$
19:       end if
20:     end if
21:     $m \Leftarrow m + 1$
22:   end while
23:   $k \Leftarrow k + 1$
24: end while
```

SA in Practice

- Behaviour strongly dependent on the cooling schedule
  - Much research on this
  - Static schedules: specified in advance
  - Adaptive schedules: react to information from the search

many iterations at each temperature, few temperatures?

or

few iterations at each temperature, many temperatures?

Cooling schedule

Initial temperature must be high

How to do this... (a couple of ways to consider)

1. If maximal difference in cost between neighboring solutions is known, initial temperature can be calculated so that increases of that magnitude are initially accepted with sufficiently large probability

Why might this be important?

First iterations are free and the final is independant of the starting

$$p = e^{-\Delta_{max}/T}$$

Cooling schedule

Initial temperature must be high

How to do this…

2. Before starting the effective algorithm a *heating procedure* is run:

   the temperature is increased until the proportion of accepted moves to total number of moves reaches a desired value.

But, in any case, experimental tuning is needed!

Cooling…

The rate at which temperature is reduced is governed by:

- Temperature length (M_k): number of iterations at a given temperature
- Cooling ratio: rate at which temperature is reduced

Cooling…

In practice, very often:

$$t_{k+1} = \alpha t_k$$

$$\text{where } 0.8 \leq \alpha \leq 0.99$$

(and usually closer to 0.99; cooling is slow)

Cooling…

There are others:

$$t_k = \frac{T_0}{1+k}$$

Cauchy schedule

$$t_k = \frac{T_0}{\log(1+k)}$$

Boltzmann schedule

$$t_k = T_0 e^{-ck^Q}$$

“Very Fast” schedule
(c and Q are constants)

How long to stay at a temperature?

The number of iterations at each temperature depends on:

- size of the neighborhood
- size of the solution space

The number of iterations may vary from temperature to temperature. It is important to spend sufficiently long time at lower temperatures.

→ e.g., increase M_k as temperature decreases

How long to stay at a temperature?

M_k can be also determined using feedback from the SA process:

- e.g., accept a certain number of moves before decreasing temperature.

⬇

- small number of iterations at high temperature
- large number of iterations at small temperatures

When to stop?

Stopping criterion

Can be subjective

Possible Criteria for stopping:

- A given minimum value of the temperature has been reached.
- A certain number of iterations (or temperatures) has passed without acceptance of a new solution.
- The proportion of accepted moves relative to attempted moves drops below a given limit.
- A specified number of total iterations has been executed

Simulated Annealing and TSP

n cities: $v_i, i = 1 \ldots n$

Tour: permutation of numbers 1 to n

$d(v_i, v_j) = d(v_j, v_i)$ is distance between v_i and v_j

Given a permutation π of the n cities,
v_i and $v_i + 1$ are adjacent cities

The permutation π has to be found that minimizes:

$$\sum_{i=1}^{n-1} d(v_i, v_{i+1}) + d(v_n, v_1)$$

2-opt neighborhood

- 2-opt neighborhood is defined by tours obtained by removing 2 edges and replacing them by a different set of links in a way that maintains feasibility.
- Size of neighborhood:

$$\frac{n(n-3)}{2}$$

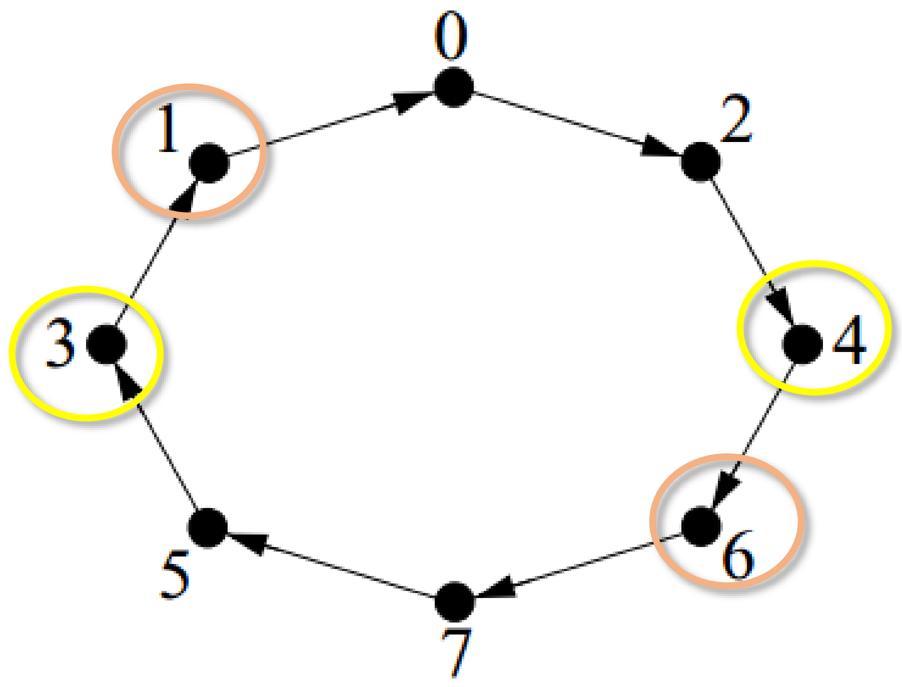
All 2-opt neighbors of a certain solution are defined by the pair i, j so that $i < j$.

A neighboring solution is generated by randomly generating i and j.

The change of the cost function can be computed incrementally:

$$d(v_i, v_j) + d(v_{i+1}, v_{j+1}) - d(v_i, v_{i+1}) - d(v_j, v_{j+1})$$

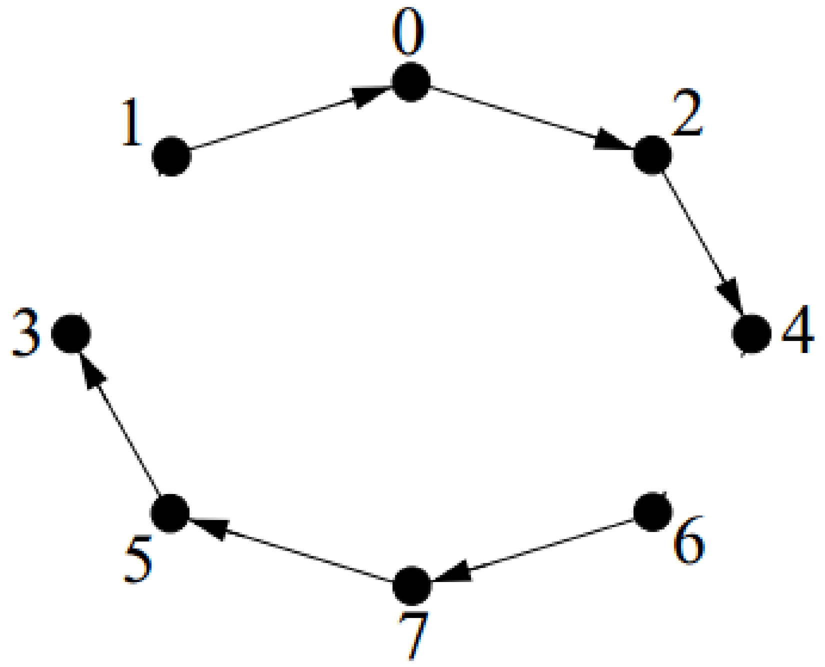
Permutation:
[0 2 4 6 7 5 3 1]



"A neighboring solution is generated by randomly generating i and j"
-- e.g., let v_i = node 4; let v_j = node 3

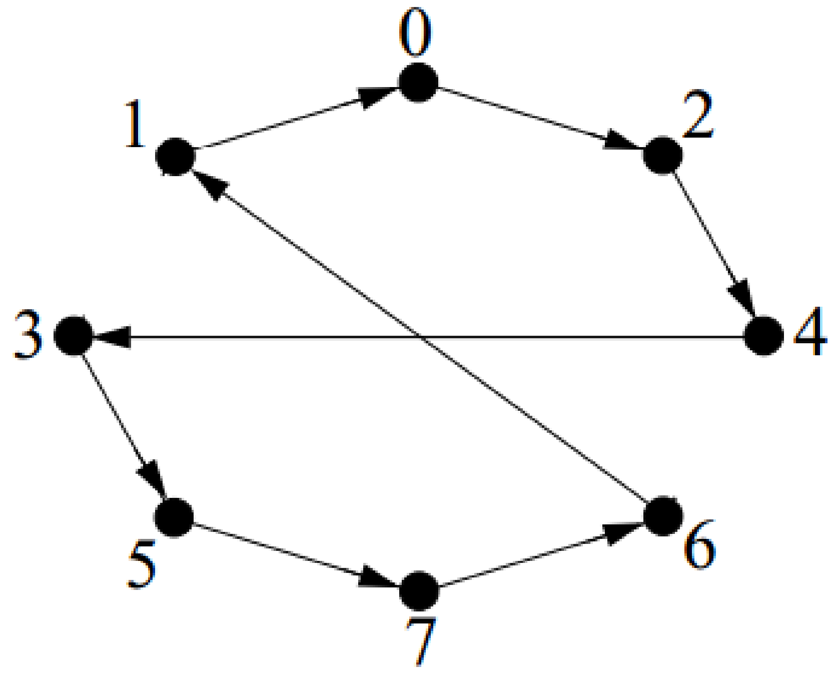
and thus, v_{i+1} = node 6; and v_{j+1} = node 1

edges (4,6) and (3,1)
are removed



edges (4,3) and (6,1)
are added

Permutation:
[0 2 4 3 5 7 6 1]



$$d(v_4, v_3) + d(v_6, v_1) - d(v_4, v_6) - d(v_3, v_1)$$

100 city TSP problem
optimal solution: cost = 21247

Best solution for T_0 = 1500, α=0.63: cost = 21331

- Time = 310 s
- Standard deviation over 10 trials: 30.3
- Average cost: 21372

Less than 0.4% from optimal!

Best solution for T_0 = 1500, α=0.90: C = 21255

- Time = 1340 s
- Standard deviation over 10 trials: 27.5
- Average cost: 21284

Less than 0.04% from optimal!

Simulated Annealing

- Based on hill climbing
- Very simple to implement
- Can escape local optimal using a temperature dependent probability
- Statistical guarantee that SA finds the global optimum
    - In practice this requires exponential (or ∞) running time
- The cooling schedule is vitally important
- Experimental tuning is very important!