2.2 S-metaheuristics
Simulated Annealing (SA)

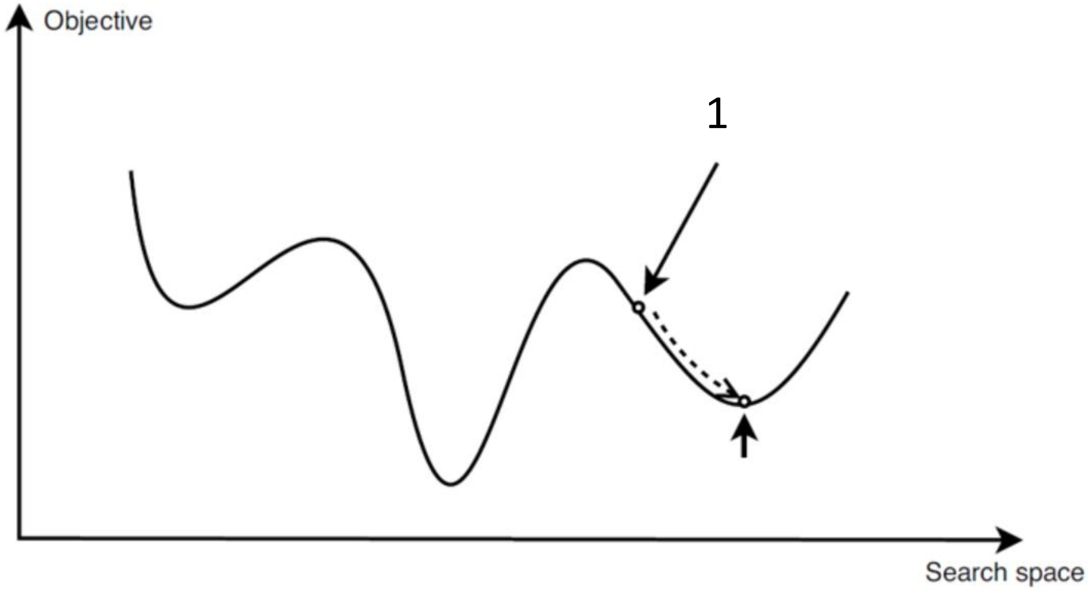
S-metaheuristics – Simulated Annealing Algorithm (SA)


Casting: pouring molten gold into an ingot

- Annealing process in metallurgy
  - Heating and then slowly cooling a substance to obtain a strong crystalline structure
  - Structure strength depends on metal cooling rate
  - Imperfections (**metastable states**) are obtained
    - Initial temperature is not sufficiently high
    - Fast cooling
  - Strong crystals (**equilibrium state**) are obtained
    - Careful and slow cooling

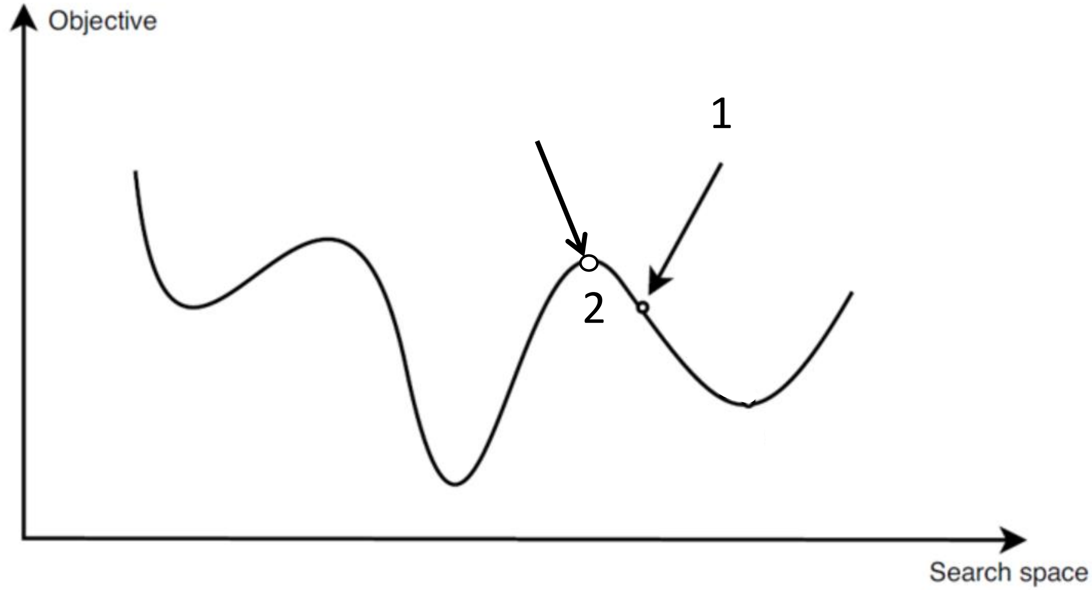
S-metaheuristics – Simulated Annealing Algorithm (SA)

- Simulated Annealing algorithm
  - Escape from local optima (how?)



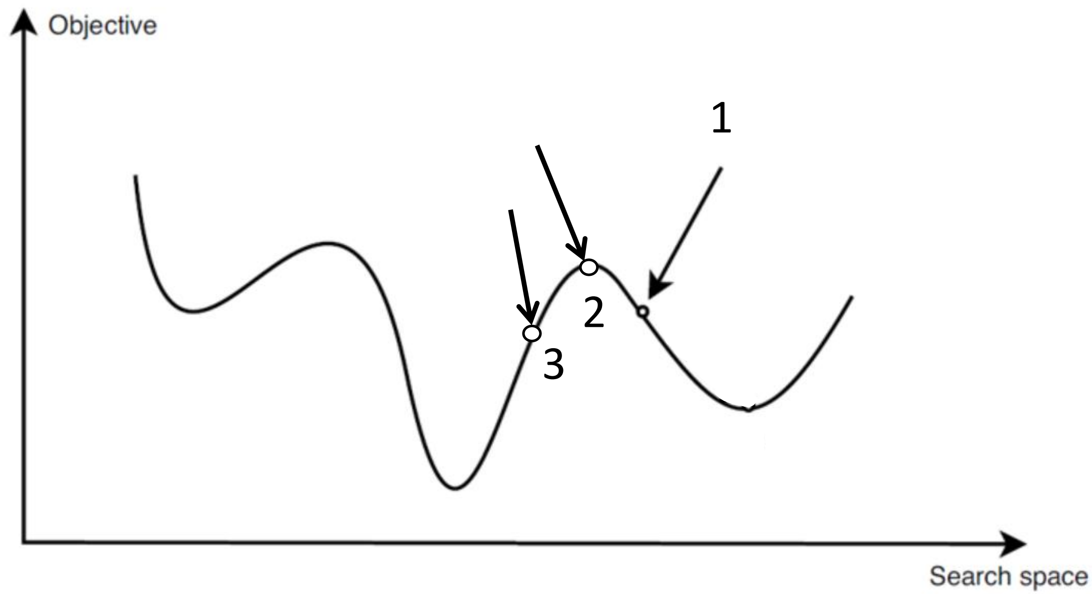
S-metaheuristics – Simulated Annealing Algorithm (SA)

- Simulated Annealing algorithm
  - Escape from local optima →Enable under some conditions the degradation of a solution



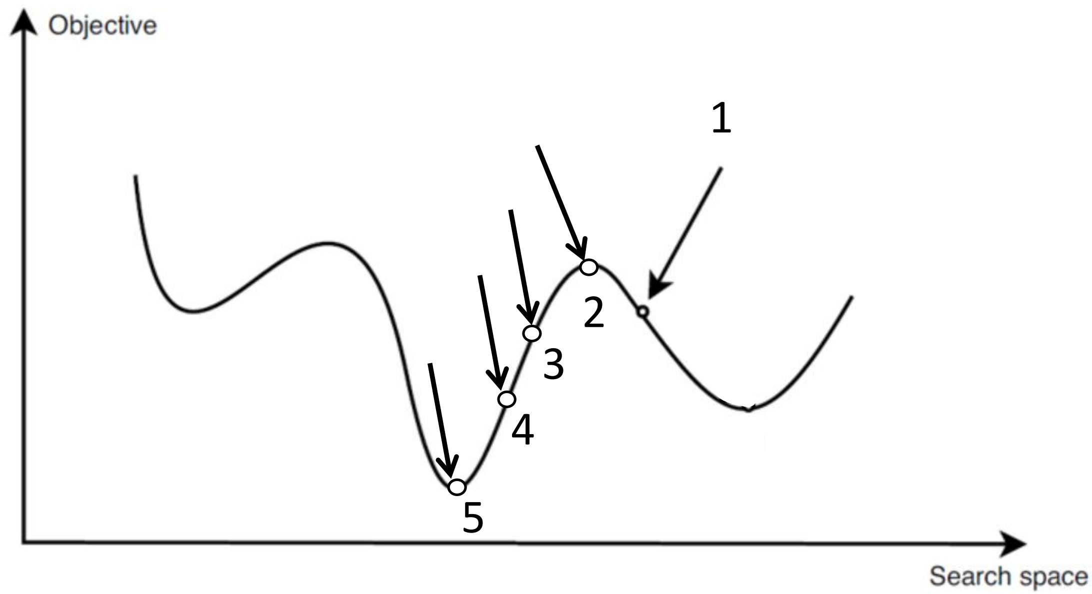
S-metaheuristics – Simulated Annealing Algorithm (SA)

- Simulated Annealing algorithm
  - Escape from local optima →Enable under some conditions the degradation of a solution



S-metaheuristics – Simulated Annealing Algorithm (SA)

- Simulated Annealing algorithm
  - Escape from local optima →Enable under some conditions the degradation of a solution



S-metaheuristics – Simulated Annealing Algorithm (SA)

Minimization problem

Algorithm Template of simulated annealing algorithm.

Save ***S*** as ***best_solution***

Input: Cooling schedule.
$s = s_0$; /* Generation of the initial solution */
$T = T_{max}$; /* Starting temperature */
Repeat
  Repeat /* At a fixed temperature */
    Generate a random neighbor s' ;
    $\Delta E = f(s) - f(s')$;
    If $\Delta E \geq 0$ **Then** $s = s'$ /* Accept the neighbor solution */
    Else Accept s' with a probability $e^{\frac{-|\Delta E|}{T}}$;
  Until Equilibrium condition
  /* e.g. a given number of iterations executed at each temperature T */
  $T = g(T)$; /* Temperature update */
Until Stopping criteria satisfied /* e.g. $T < T_{min}$ */
Output: Best solution found.

Compare ***S*** with the ***best_solution*** and update if necessary

S-metaheuristics – Simulated Annealing Algorithm (SA)

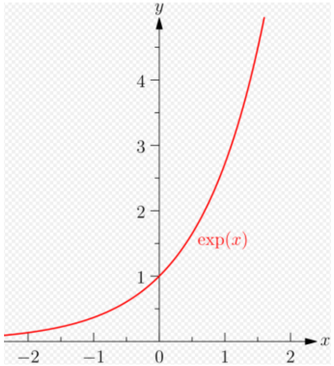
Minimization problem

Algorithm Template of simulated annealing algorithm.

Save ***S*** as ***best_solution***

Compare ***S*** with the ***best_solution*** and update if necessary

Input: Cooling schedule.
$s = s_0$; /* Generation of the initial solution */
$T = T_{max}$; /* Starting temperature */
Repeat
  Repeat /* At a fixed temperature */
    Generate a random neighbor s' ;
    $\Delta E = f(s) - f(s')$;
    If $\Delta E \geq 0$ **Then** $s = s'$ /* Accept the neighbor solution */
    Else Accept s' with a probability $e^{\frac{-|\Delta E|}{T}}$;
  Until Equilibrium condition
  /* e.g. a given number of iterations executed at each temperature T */
  $T = g(T)$; /* Temperature update */
Until Stopping criteria satisfied /* e.g. $T < T_{min}$ */
Output: Best solution found.



S-metaheuristics – Simulated Annealing Algorithm (SA)

- Simulated Annealing algorithm
  - Stochastic algorithm that enables under some conditions the degradation of a solution

When the temperature is High

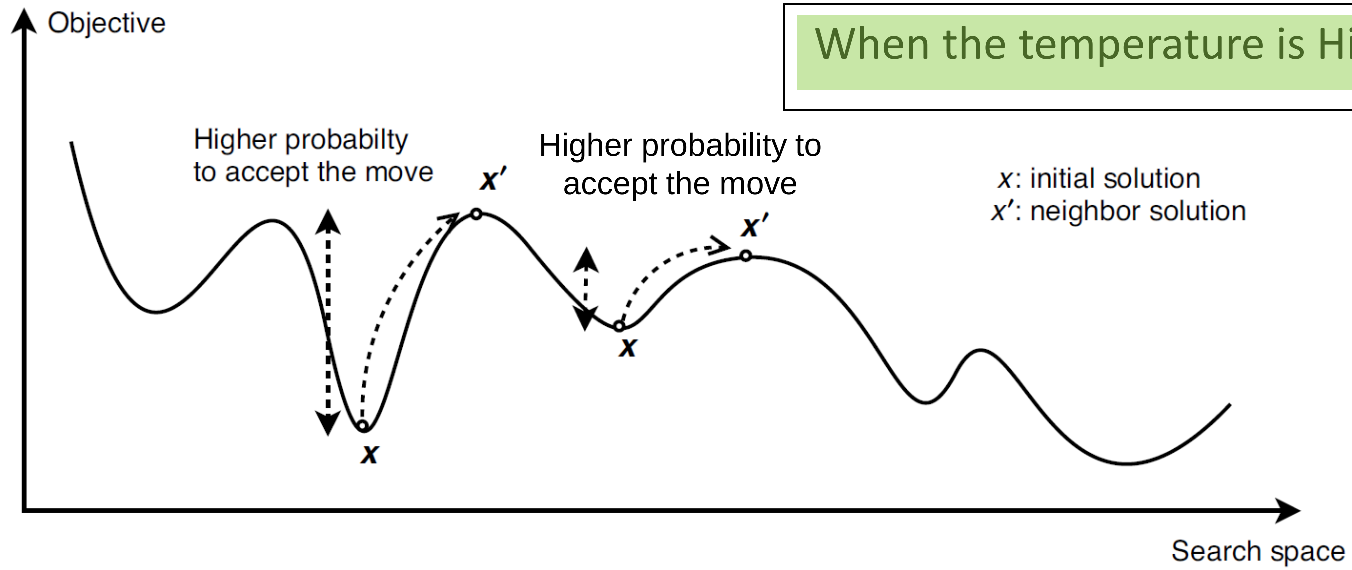


FIGURE Simulated annealing escaping from local optima. The higher the temperature, the more significant the probability of accepting a worst move. At a given temperature, the lower the increase of the objective function, the more significant the probability of accepting the move. A better move is always accepted.

S-metaheuristics – Simulated Annealing Algorithm (SA)

- Simulated Annealing algorithm
  - Stochastic algorithm that enables under some conditions the degradation of a solution

For static temperature or low

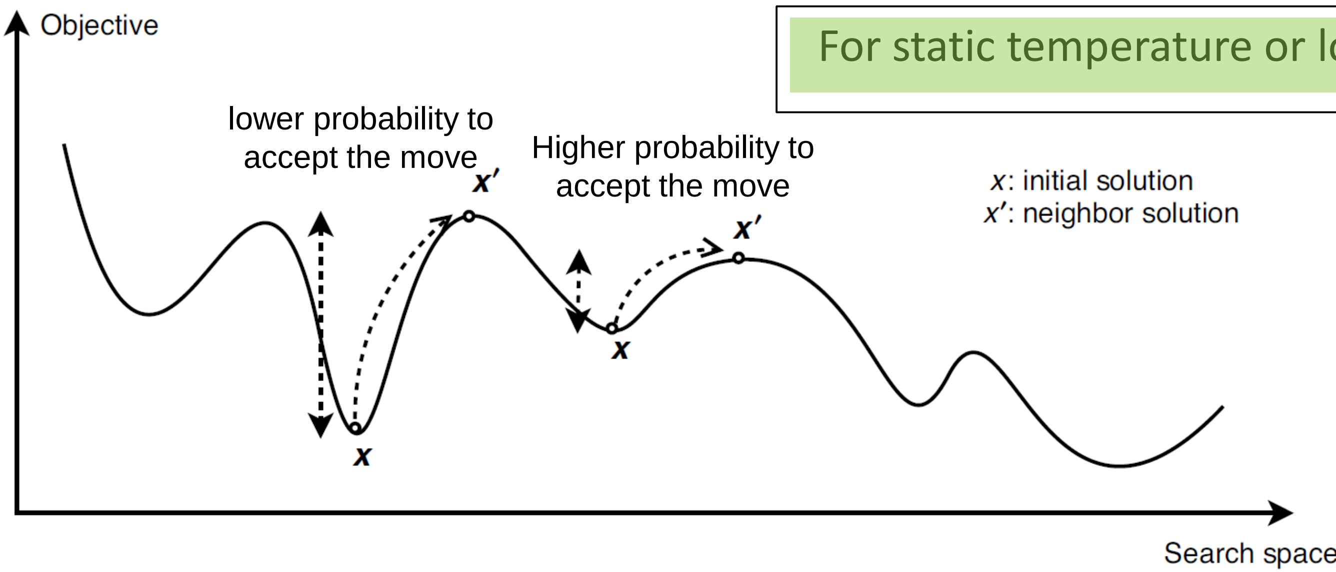


FIGURE Simulated annealing escaping from local optima. The higher the temperature, the more significant the probability of accepting a worst move. At a given temperature, the lower the increase of the objective function, the more significant the probability of accepting the move. A better move is always accepted.

S-metaheuristics – Simulated Annealing Algorithm (SA)

- SA parameters
  - Initial temperature
    - Can be set to high value but time consuming
    - Other strategies (page 130 from pdf doc)
  - Cooling schedule (page 132 from pdf doc) / temperature update, $T_i > 0, \forall i$ and $\lim_{i \longrightarrow \infty} T_i = 0$
    - Linear (with constant) $T_i = T_0 - i \times \beta$
    - Geometric (most popular) $T = \alpha T$ where $\alpha \in [0.5, 0.99]$
    - Logarithmic (convergence proof to Global Optimum) $T_i = \frac{T_0}{\log(i)}$
    - Very slow decrease (one iteration at each temperature) $T_{i+1} = \frac{T_i}{1 + \beta T_i}$ with $\beta = T_0 - T_F/(L-1)T_0 T_F$
  - Acceptance probability function (move's acceptance)
    - Neighbor solution always accepted/selected if it improves the current solution
    - Otherwis, it is selected with a **probability** ***P*** depending on the current temperature and amount of energy degradation.

Boltzmann distribution $P(\Delta E, T) = e^{-\frac{|f(s) - f(s')|}{T}}$

S-metaheuristics – Simulated Annealing Algorithm (SA) (page 128 from pdf doc)

Example Illustration of the SA algorithm. Let us maximize the continuous function $f(x) = x^3 - 60x^2 + 900x + 100$. A solution x is represented as a string of 5 bits. The neighborhood consists in flipping randomly a bit. The global maximum of this function is 01010 ($x = 10$, $f(x) = 4100$). The first scenario starts from the solution 10011 ($x = 19$, $f(x) = 2399$) with an initial temperature T_0 equal to 500 (Table 2.5). The second scenario starts from the same solution 10011 with an initial temperature T_0 equal to 100 (Table 2.6). The initial temperature is not high enough and the algorithm gets stuck by local optima.

TABLE 2.5 First Scenario $T = 500$ and Initial Solution (10011)

T	S'	Solution	f	Δf	Move?	New Neighbor Solution
500	3	00011	2287	112	Yes	00011
450	7	00111	3803	<0	Yes	00111
405	6	00110	3556	247	Yes	00110
364.5	14	01110	3684	<0	Yes	01110
328	12	01100	3998	<0	Yes	01100
295.2	8	01000	3972	16	Yes	01000
265.7	10	01010	**4100**	<0	Yes	01010
239.1	11	01011	4071	29	Yes	01011
215.2	27	11011	343	3728	No	01011

S-metaheuristics – Simulated Annealing Algorithm (SA) (page 128 from pdf doc)

Example Illustration of the SA algorithm. Let us maximize the continuous function $f(x) = x^3 - 60x^2 + 900x + 100$. A solution x is represented as a string of 5 bits. The neighborhood consists in flipping randomly a bit. The global maximum of this function is 01010 ($x = 10$, $f(x) = 4100$). The first scenario starts from the solution 10011 ($x = 19$, $f(x) = 2399$) with an initial temperature T_0 equal to 500 (Table 2.5). The second scenario starts from the same solution 10011 with an initial temperature T_0 equal to 100 (Table 2.6). The initial temperature is not high enough and the algorithm gets stuck by local optima.

TABLE 2.6 Second Scenario: T = 100 and Initial Solution (10011). When Temperature is not High Enough, Algorithm Gets Stuck

T	S'	Solution	f	Δf	Move?	New Neighbor Solution
100	3	00011	2287	112	No	10011
90	23	10111	1227	1172	No	10011
81	18	10010	2692	<0	Yes	10010
72.9	26	11010	516	2176	No	10010
65.6	16	10000	**3236**	<0	Yes	10000
59	20	10100	2100	1136	Yes	10000

S-metaheuristics – Simulated Annealing Algorithm (SA)

- Lab session 

Implement your second S-metaheuristic algorithm – The Simulated Annealing Algorithm (SA) with the parametrization strategies (cooling schedule) and apply it for

  - The TSP problem – Data available on Teams
  - The example of maximization function (slide 36, x in [0,31])
  - Show for each strategy the associated trajectory curve