Bio-inspired Computation and Optimization

Project 1: Knapsack Problem – Part I
Discussion of Task 2

a) Neighbourhood Function

- Define the neighbourhood function to return the set of solutions withing a Hamming distance *d* from the solution *x*.

If we denote the Hamming distance between two binary vectors as $h(x, y)$, then we can define the neighborhood function $N_d(x)$ as:

$$N_d(x) = \{y \in X | h(x, y) \leq d\}$$

a) Proof

We have to show the following property: for any $x \in X, d \in \mathbb{N}, d < n$: $N_d(x) \subset N_{d'}(x) \iff d < d'$. We prove both directions of the equivalence separately.

Proof for $N_d(x) \subset N_{d'}(x) \Leftarrow d < d'$ For $d < d'$ and $\forall x \in X$:

$$\begin{array}{rcll} \forall y \in N_d(x) & : & h(x,y) \leq d & \text{(definition of } N_d(x)\text{)} \\ & \Rightarrow & h(x,y) \leq d' & (d < d') \\ & \Rightarrow & y \in N_{d'}(x) & \text{(definition of } N_d(x)\text{)} \end{array}$$

This shows that $N_d(x) \subseteq N_{d'}(x)$. Now, we have to show that the subset is a strict subset. Because $d < n$, we have $d + 1 \leq n$. This means that there exists a bit vector z, which is created from the bit vector x by flipping $d+1$ bits. Thus, $h(x,z) = d + 1 > d \Rightarrow z \notin N_d(x)$. It follows that:

$$\begin{array}{rc} & d < d' \\ \Rightarrow & d + 1 \leq d' \\ \Rightarrow & h(x,z) \leq d' \\ \Rightarrow & z \in N_{d'}(x) \\ \Rightarrow & N_d(x) \subset N_{d'}(x) \end{array}$$

solution of Michael Haspra, Stefan Mueller, and Timo Wuersch

a) Proof continuation

Proof for $N_d(x) \subset N_{d'}(x) \Rightarrow d < d'$ If we assume $N_d(x) \subset N_{d'}(x)$, there exists z for which holds that $z \in N_{d'}(x)$ but $z \notin N_d(x)$. Thus, $h(x,z) \leq d'$ and $h(x,z) > d$ and so $d < h(x,z) \leq d' \Rightarrow d < d' \Rightarrow N_d(x) \subset N_{d'}(x) \iff d < d'$.

solution of Michael Haspra, Stefan Mueller, and Timo Wuersch

a) Pitfalls

- Exercise care in writing

$$\leq \text{ vs } <$$

$$\subseteq \text{ vs } \subset$$

$$\in \text{ vs } \subset$$

- Do not write something like „it is obvious, that the property holds“, try prooving the property.
- Proofing the property for a specific example is not sufficient.
- Show both directions! <= as well as =>

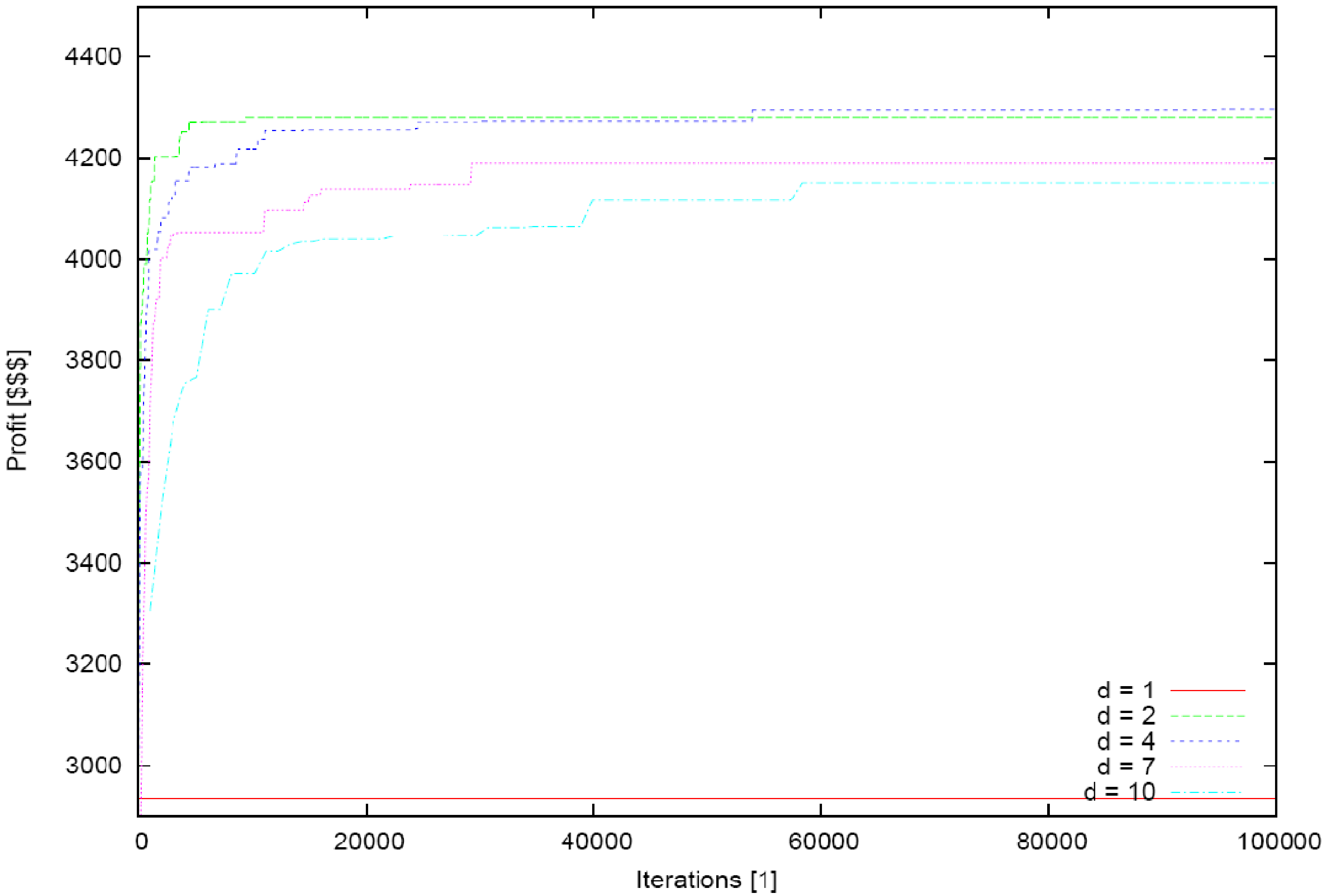
b) Randomized Local Search

Randomly choose an initial solution x_1 from X
Calculate $f(x_1)$
Initialize memory, i.e., $M = \{(x_1, f(x_1))\}$
Set iteration counter $t = 0$
loop
  $t = t + 1$
  Choose $x_t \in N(x) \subseteq X$ where $M = \{(x, f(x))\}$
  Calculate $f(x_t)$
  if $f(x_t) \geq f(x)$ **then**
    $M = \{(x_t, f(x_t))\}$
  end if
  if $t \geq t_{MAX}$ **then**
    Output best solution x^* stored in M
    Stop
  end if
end loop

Maximization problem:
$(X, \mathfrak{R}, f, \geq)$

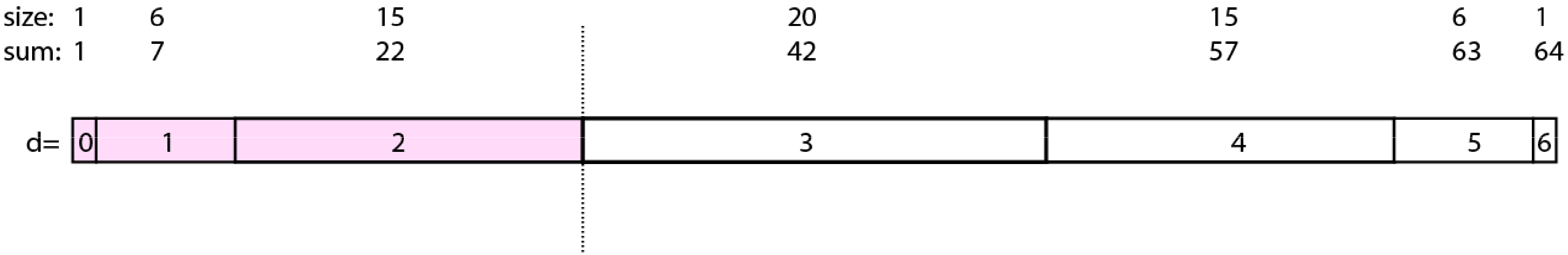
- There is always just one solution in the memory.
- A new solution is generated by locally perturbing the current one.

b) Result


4400
4200
4000
3800
3600
3400
3200
3000
Profit [$$$]
0
20000
40000
60000
80000
100000
Iterations [1]
d = 1
d = 2
d = 4
d = 7
d = 10

b) Select uniformly in $N_d(x)$

- To get a solution $x_t \in N_d(x)$ get first the distance d' you want, then flip d' bits randomly.
- Pay attention to pick d' according to the size of the neighbourhoods!
- Example: n = 6:



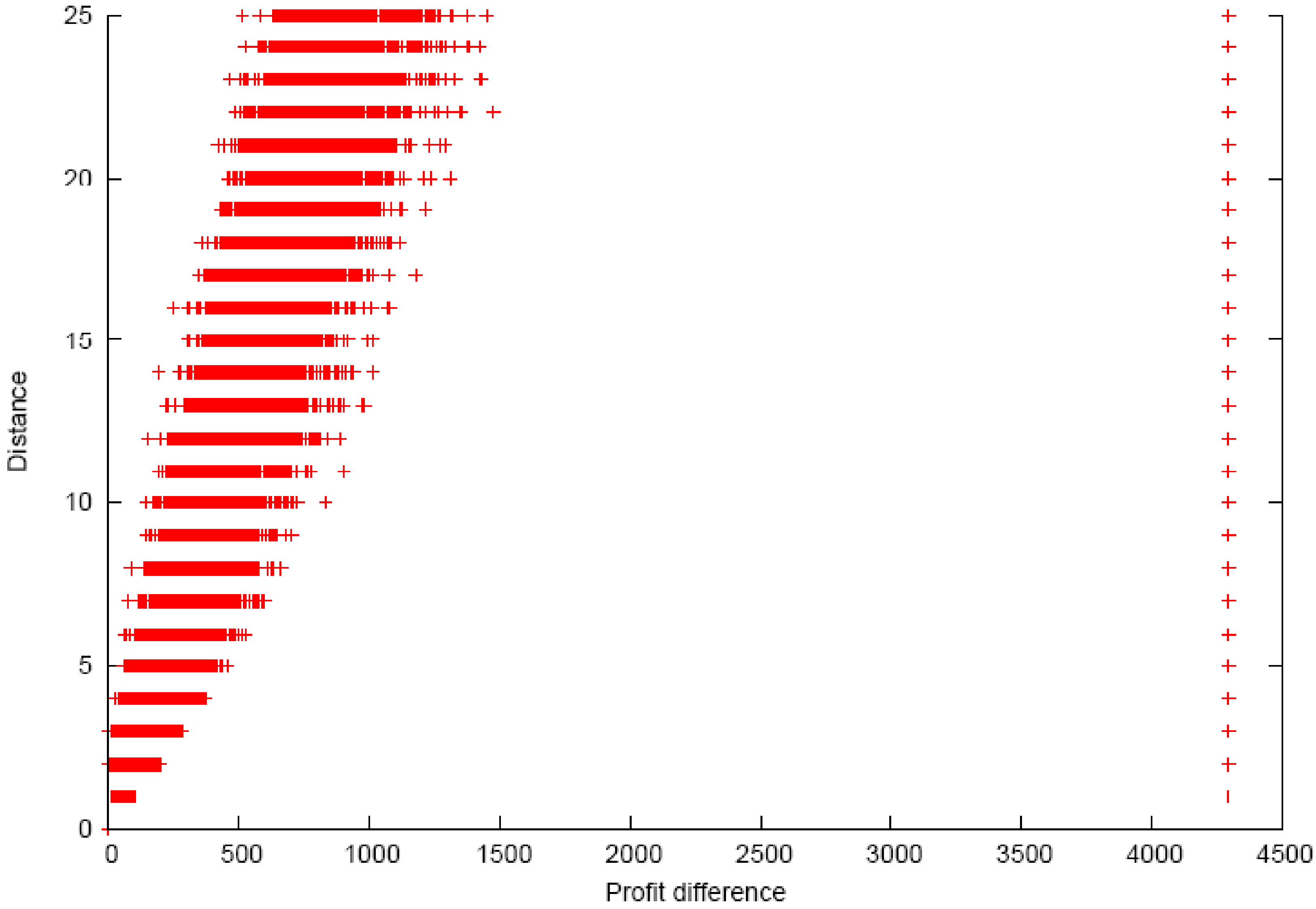
- Randomly pick a number between 0 and 63. Select the highest interval, such that the according sum is greater than the number you picked.
- E.g., 15: 15 is smaller than 22 and greater equal than 7 => Flip 2 bits.
- E.g., 42: 42 is smaller than 57 and greater equal than 42 => Flip 4 bits.

b) Pitfalls

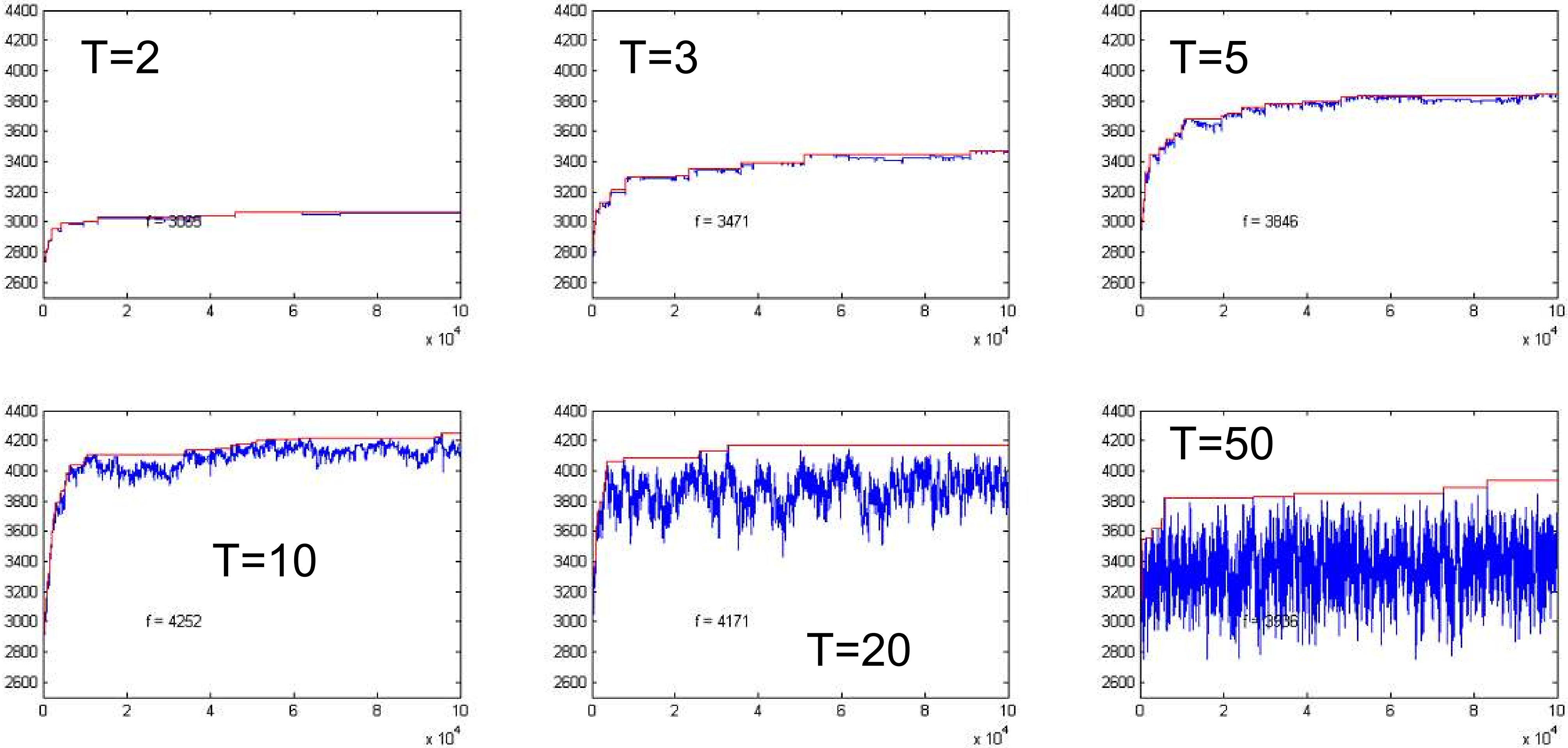
- Take care of sampling uniformly from the neighbourhood!
- Don‘t forget to add a legend!
- Use a logarithmic or linear scale
- Plot 100‘000 iterations!

- You should get the best results using d between 2 and 5. The

c) Solution space shape

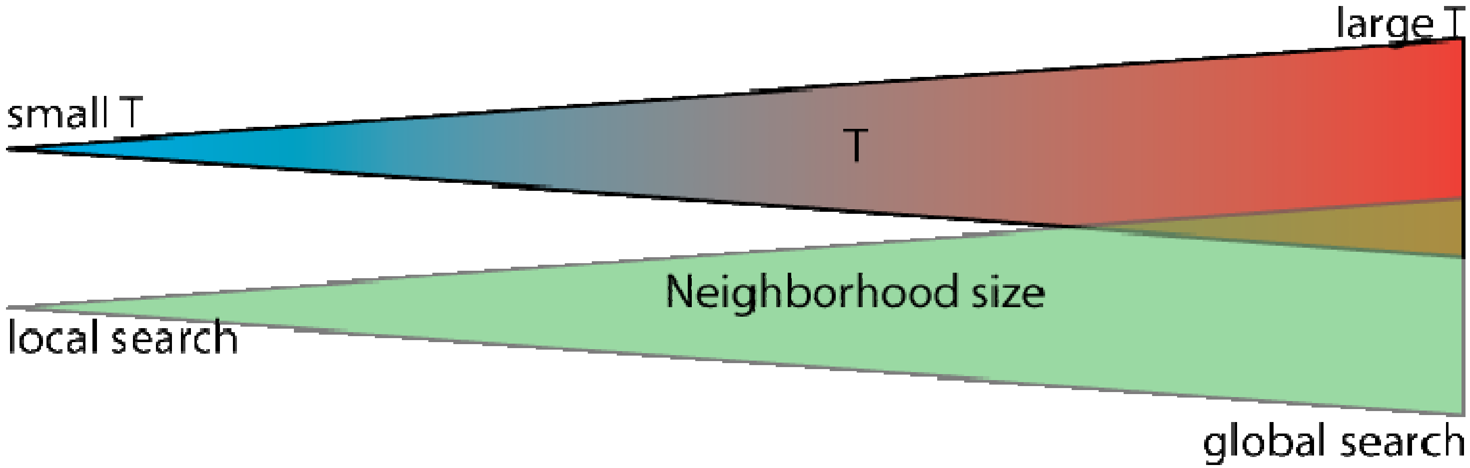


d) Metropolis Algorithm



solution of Markus Jehl and Stefan Dahinden

Influence of T, Influence of Neighborhoodsize



hill climber

- *accept only better solutions*
- *getting stuck in local optima*

randomized search

- *accept all solutions*
- *small chance to find (local) optima*