Facility location problems

Discrete models and local search methods

Yuri Kochetov

Sobolev Institute of Mathematics
Novosibirsk Russia
e-mail: jkochet@math.nsc.ru

Lecture 2. Local Search Methods

Content

1. Standard local descent
2. Neighborhoods and properties of local optima
3. Advanced local search strategies
   - 3.1. Simulated Annealing
   - 3.2. Great Deluge
   - 3.3. Tabu Search
   - 3.4. Variable Neighborhood Search
   - 3.5. Genetic algorithms

Local Search Problems

- Input:
  - Optimization problem $\min \{f(s), s \in Sol\}$
  - Neighborhood function $N: Sol \rightarrow 2^{Sol}$
- Output:
  - feasible solution $s \in Sol$;
- Goal:
  - find a local optimum $f(s) \leq f(s')$, $s' \in N(s)$.

.

Standard local descent algorithm

1. Select a starting solution $s \in Sol$
2. Try to find better neighboring solution s':

$$f(s') < f(s), \quad s' \in N(s)$$

3. If it exists then move:

$$s := s' \text{ and go to 2} \quad \text{else STOP}$$

4. Return local minimum s.

Examples: – Siplex method for linear programming problem;

– Ford–Falkerson method for maximum flow problem;

– Bubble sort algorithm for the sorting problem.

The *p*-median problem with Swap neighborhood

- Input:
  - a set I of facilities;
  - a set J of users;
  - a number p of opened facilities;
  - a production–transportation cost c_{ij} to service user j from facility i;
- Output:

  a set $S \subset I$, $|S| = p$, of opened facilities;
- Swap neighborhood:

$$N(S) = \{S' \subset I \,\big|\, |S'| = p, |S \setminus S'| = |S' \setminus S| = 1\}$$
- Goal: fine a local optimal solution

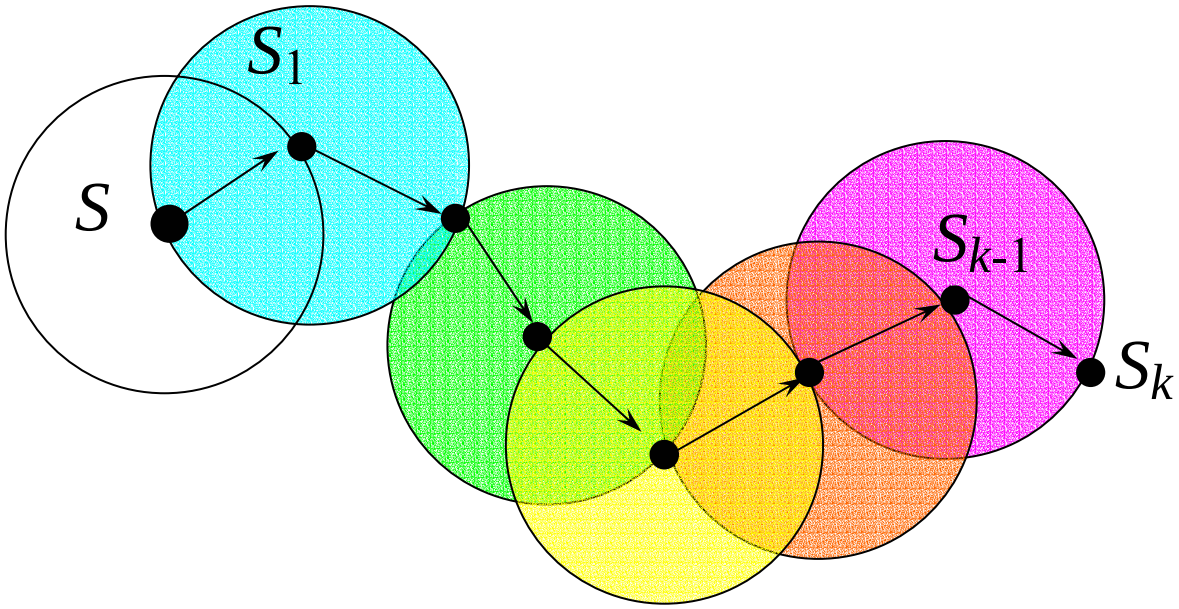
$$F(S) = \sum_{j \in J} \min_{i \in S} c_{ij} \le \sum_{j \in J} \min_{i \in S'} c_{ij}, \forall S' \in N(S).$$

Lin-Kernighan Neighborhoods

Basic steps:

1. Choose two facilities $i_{ins} \in I \setminus S$ and $i_{rem} \in S$ such that $F(S \cup \{i_{ins}\} \setminus \{i_{rem}\})$ is minimal even if it greater than $F(S)$
2. Perform excange of i_{ins} and i_{rem}.
3. Repeat steps 1, 2 k times so that a facility can not be chosen to be inserted in S if it has been removed from S in one of the previous iterations of step 1 and step 2.

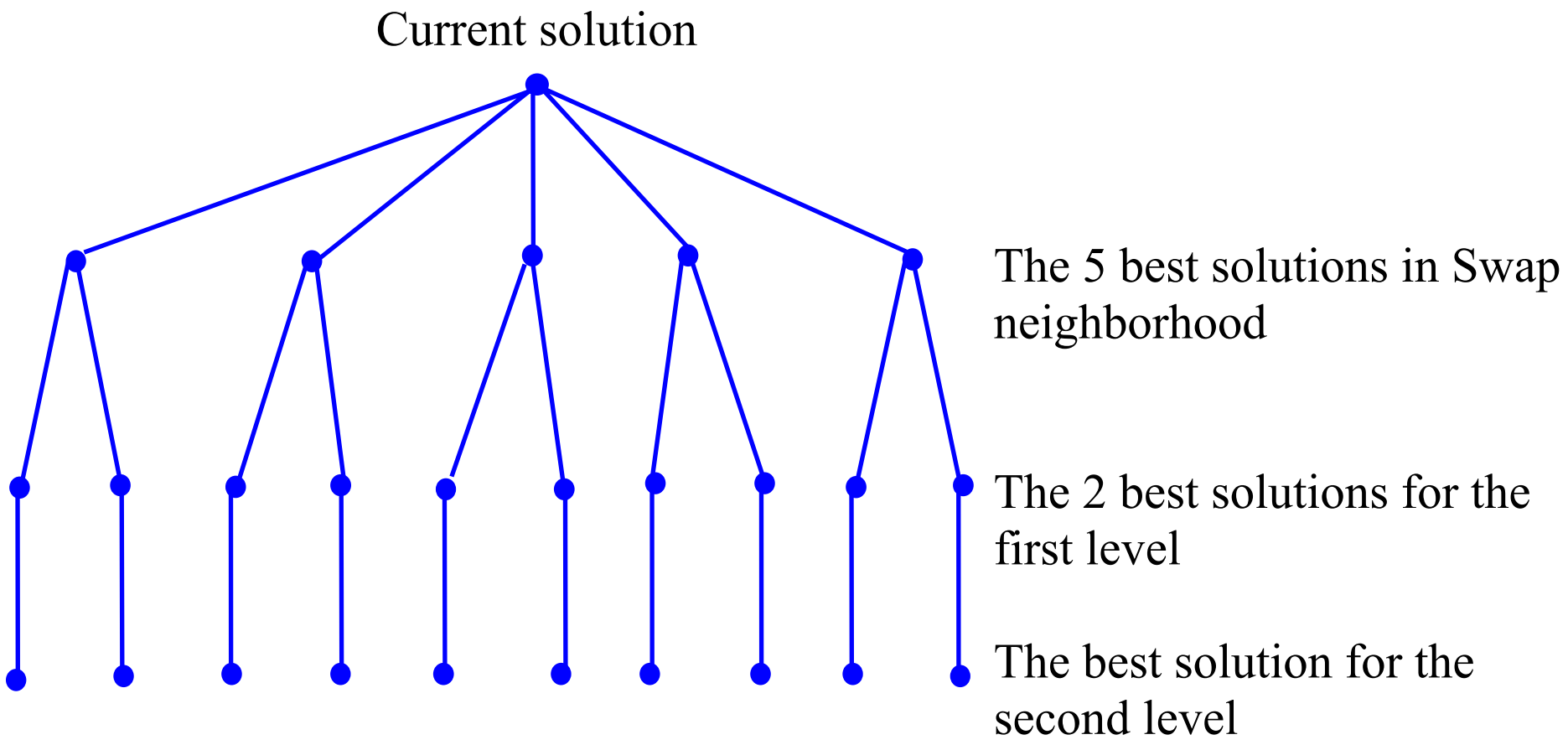
Lin-Kernighan Neighborhoods



The sequence $\{(i^{\tau}_{ins}, i^{\tau}_{rem})\}_{\tau \le k}$ defines k neighbors S_{τ} for the solution S.

The best element in the LK–neighborhood can be found in $O(kIJ)$ time.

Balas – Vazacopoulos Neighborhood



Theorem 2.1. Assuming P≠NP, no polynomial searchable neighborhood N can guarantee that each local optimum S of the p-median problem is ρ-approximate solution for any fixed constant $\rho > 1$, that is $F(S)/Opt \le \rho$ for any instances.

Proof. Let us consider the vertex cover problem (VC): given graph $G = (V,E)$ and integer positive number k. Is there a subset $V' \subset V$, $|V'| \le k$ such that each edge is incident to at least one vertex of V'? It is NP–complete problem.

Consider the family of the p-median problems with $I = V$, $J = E$, $p = k$ and

$$c_{ij} = \begin{cases} 1 & \text{if edge } e_j \text{ is incident to vertex } i \\ (|E|+1)\rho, & \text{otherwise.} \end{cases}$$

Let us select an arbitrary subset $S \subset I$, $|S| = \rho$ and apply Standard Local Search Descent algorithm with neighborhood N. For this family, it is polynomial time procedure. But each local optimum must have the same value if it is ρ-approximate solution! Otherwise $F(S) \ge |E| - 1 + (|E| + 1)\rho > |E|\rho$. ■

Remark. The statement holds for $\rho = 2^{q(|I|,|J|)}$ where q is any fixed polynomial.

Corollary. Assuming $P \neq NP$, there is no exact polynomial searchable neighborhood for the p-median problem.

Theorem 2.2. [4] For the metric p-median problem, standard local descent algorithm with *Swap*–neighborhood produces 5-approximate solution.

Theorem 2.3. [Arua et al. 2004] For the metric p-median problem, standard local descent algorithm with k–*Swap*–neighborhood produces $(3+\frac{2}{k})$-approximate solution.

Advanced Local Search Strategies

Optimal problem $\min \{F(S) \mid s \in Sol\}$

Neighborhood $N: Sol \rightarrow 2^{Sol}$

Threshold algorithms

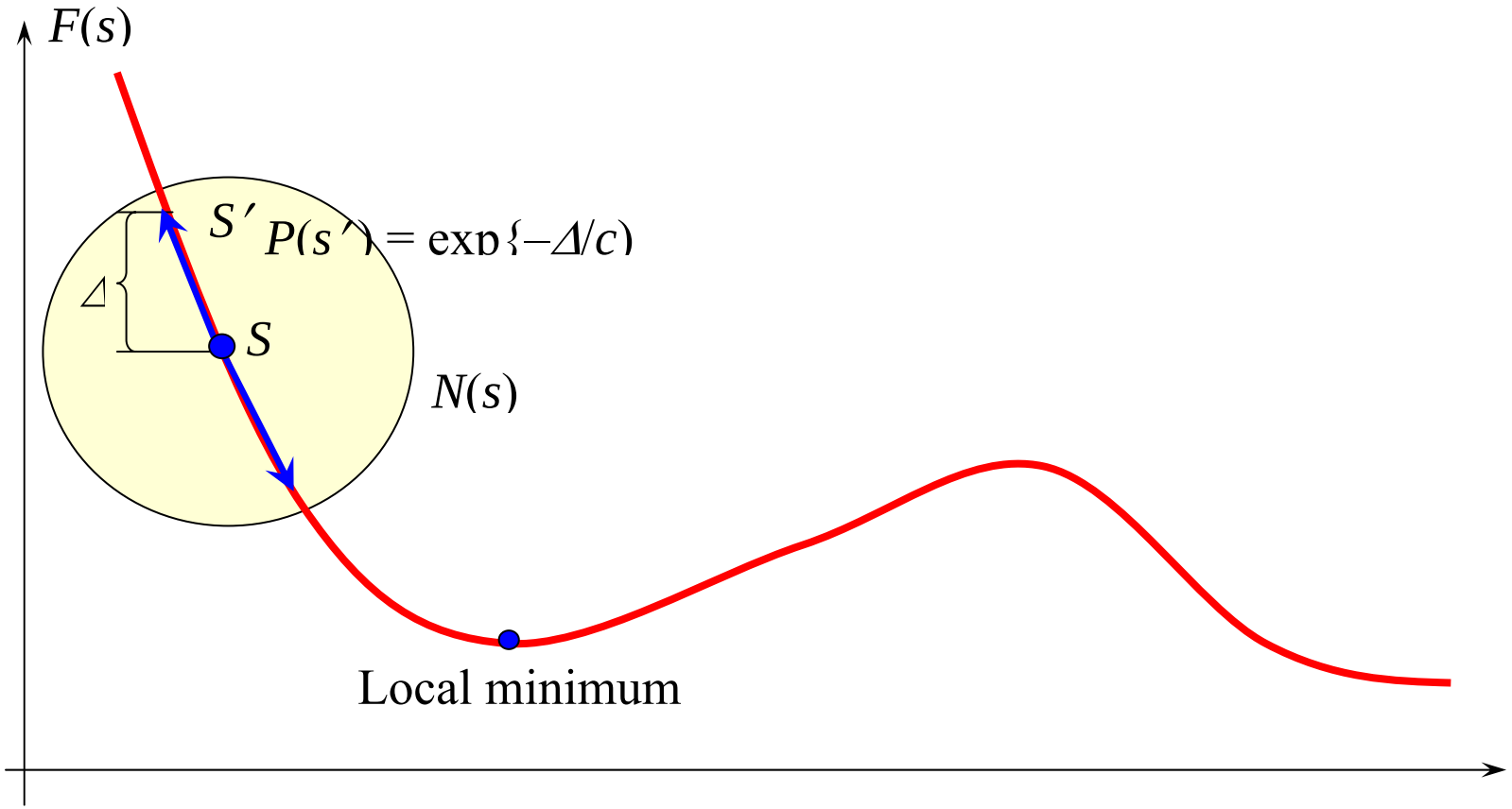
1. Construct an initial solution $s \in Sol$, $s^* := s$, $k := 0$.
2. Repeat until a stop criterion is satisfied
   - 2.1. Generate $s' \in N(S)$ and put $k := k+1$;
   - 2.2. If $F(S) - F(S') < t_k$ then $s := s'$;
   - 2.3. If $F(S^*) > F(S)$ then $s^* := s$.
3. Return s^*.

Three Types of Threshold Algorithms

- Iterative improvement: $t_k = 0$, $k \geq 0$, it is standard LD algorithm with random pivoting rule.
- Threshold accepting: $t_k \geq 0$, $t_k \geq t_{k+1}$, $k \geq 0$, $\lim_{k \to \infty} t_k = 0$.
- Simulated annealing: t_k is a random variable with expected value $E(t_k) = c_k$, $k \geq 0$; more exactly, the probability of accepting $s' \in N(S)$ at the k^{th} iteration is given by

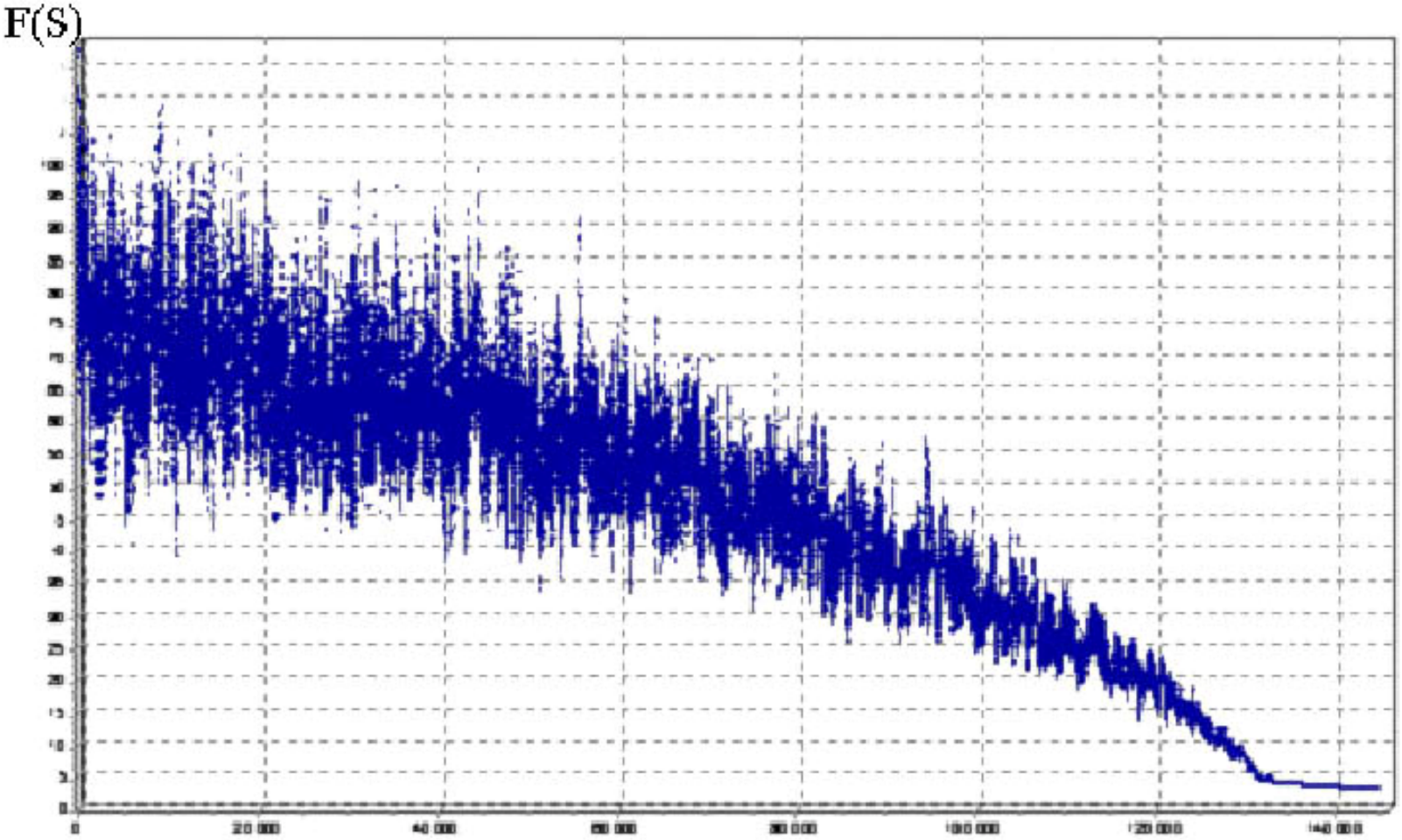
$$P_{c_k}\{\text{accept } s'\} = \begin{cases} 1 & \text{if } F(S') \leq F(S) \\ \exp\left(\dfrac{F(S) - F(S')}{c_k}\right) & \text{if } F(S') > F(S) \end{cases}$$

Simulated Annealing Algorithm

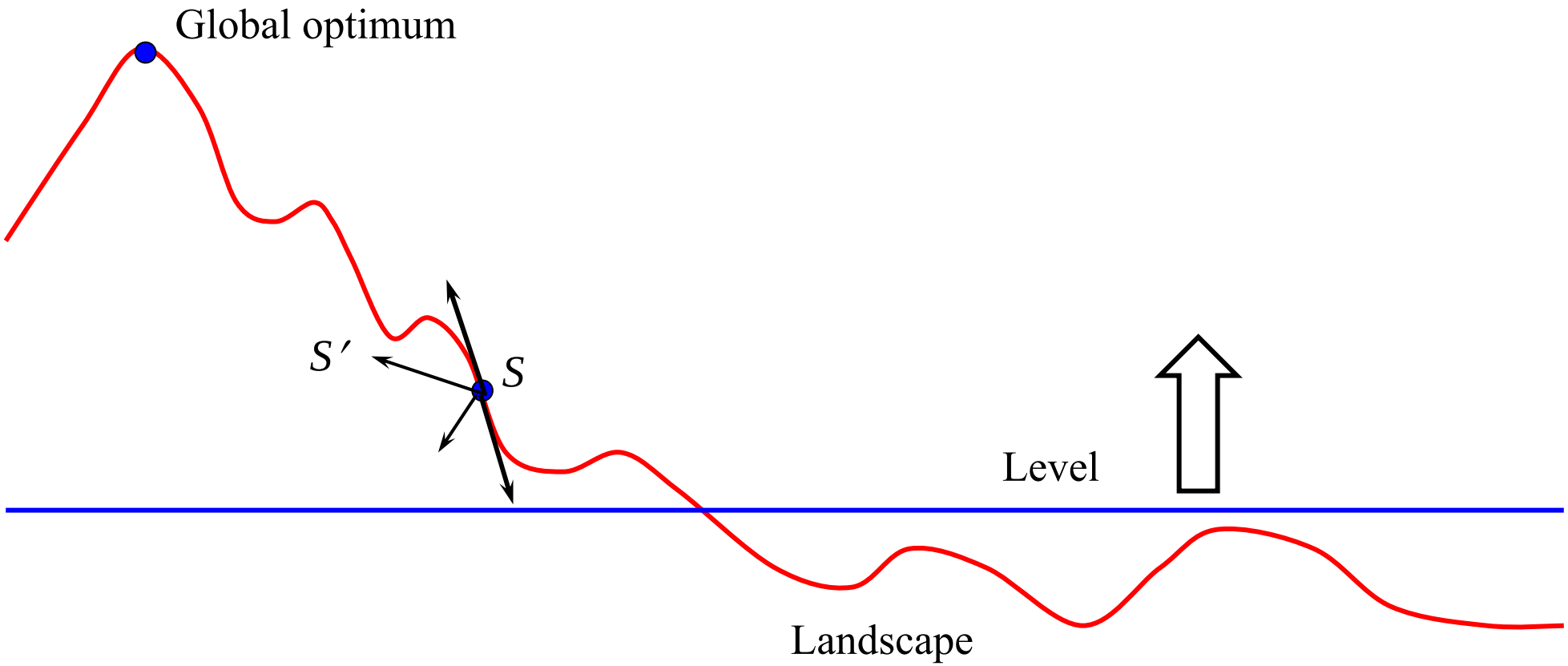


SA was introduced by Kirkpatrick, Gellat, and Vecchi, 1983; Černỳ, 1985.

Typical Behavior of SA



Great Deluge Algorithm

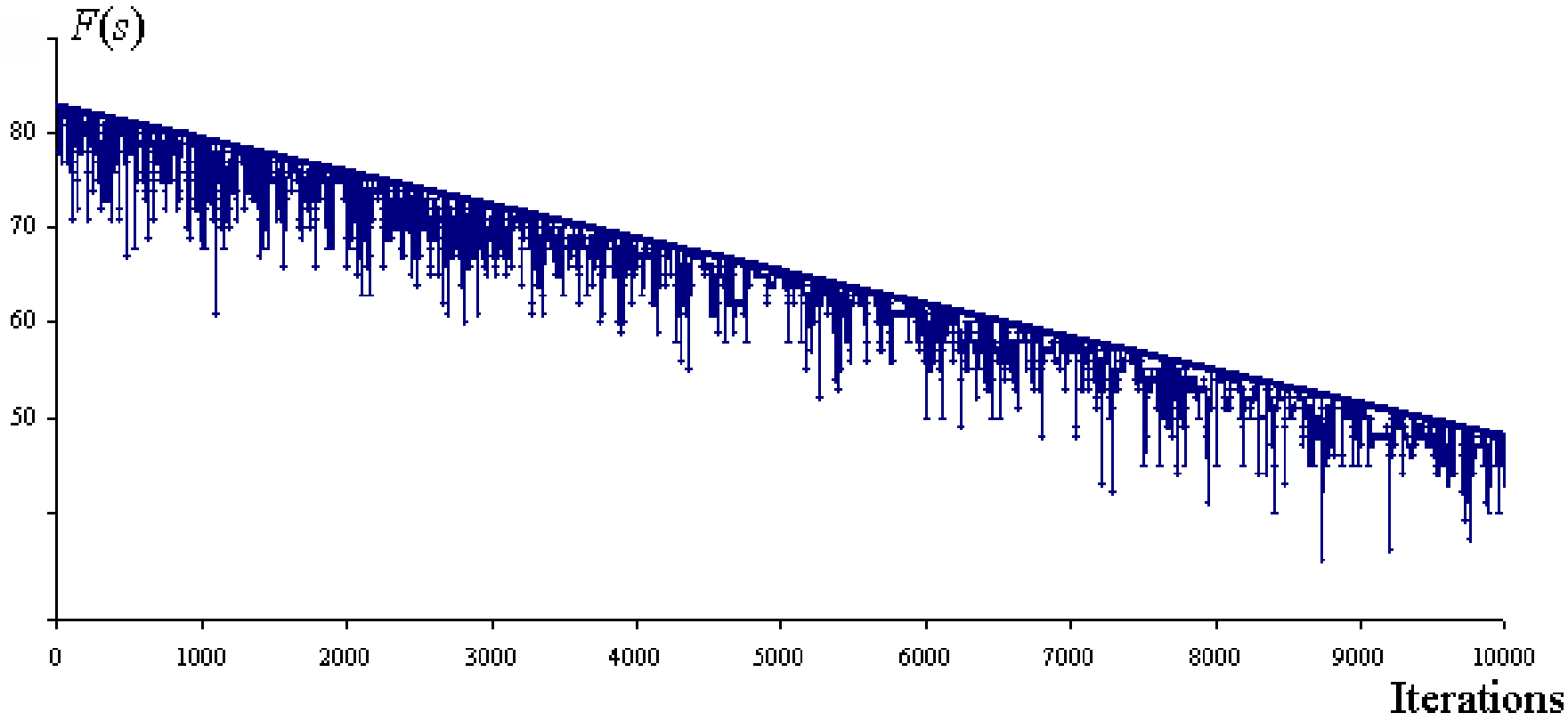


GD was introduced by Dueck, 1993

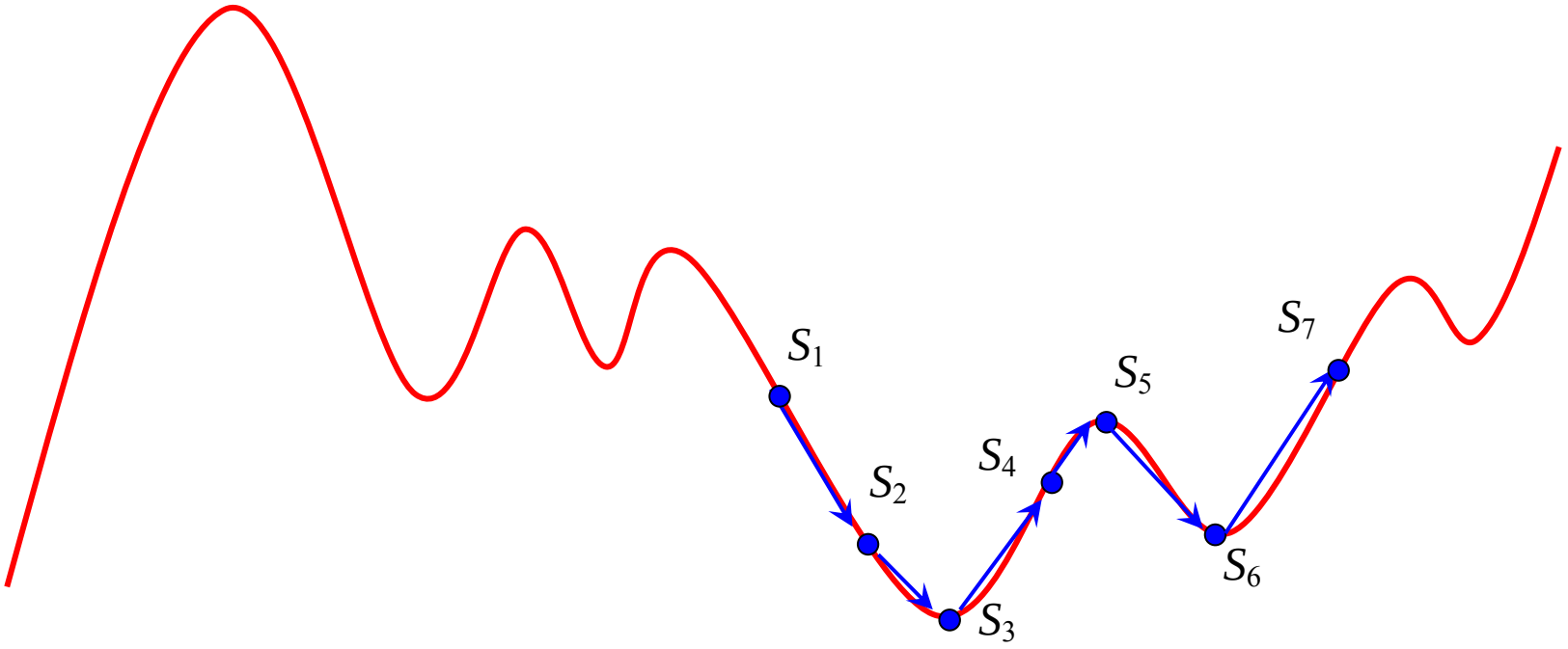
Great Deluge Algorithm

1. Construct an initial solution s, put $L = F(s)$, $s^* := s$;
2. Repeat until a stop criterion is satisfied;
   2.1. Randomly generate $s' \in N(s)$;
   2.2. If $F(s') \leq F(s)$ then $s := s'$ else if $F(s') \leq L$ then $s := s'$;
   2.3. $L := L - \Delta L$
   2.4. If $F(s^*) > F(s)$ then $s^* := s$;
3. Return s^*.

Typically behavior of Great Deluge



Tabu Search Algorithm

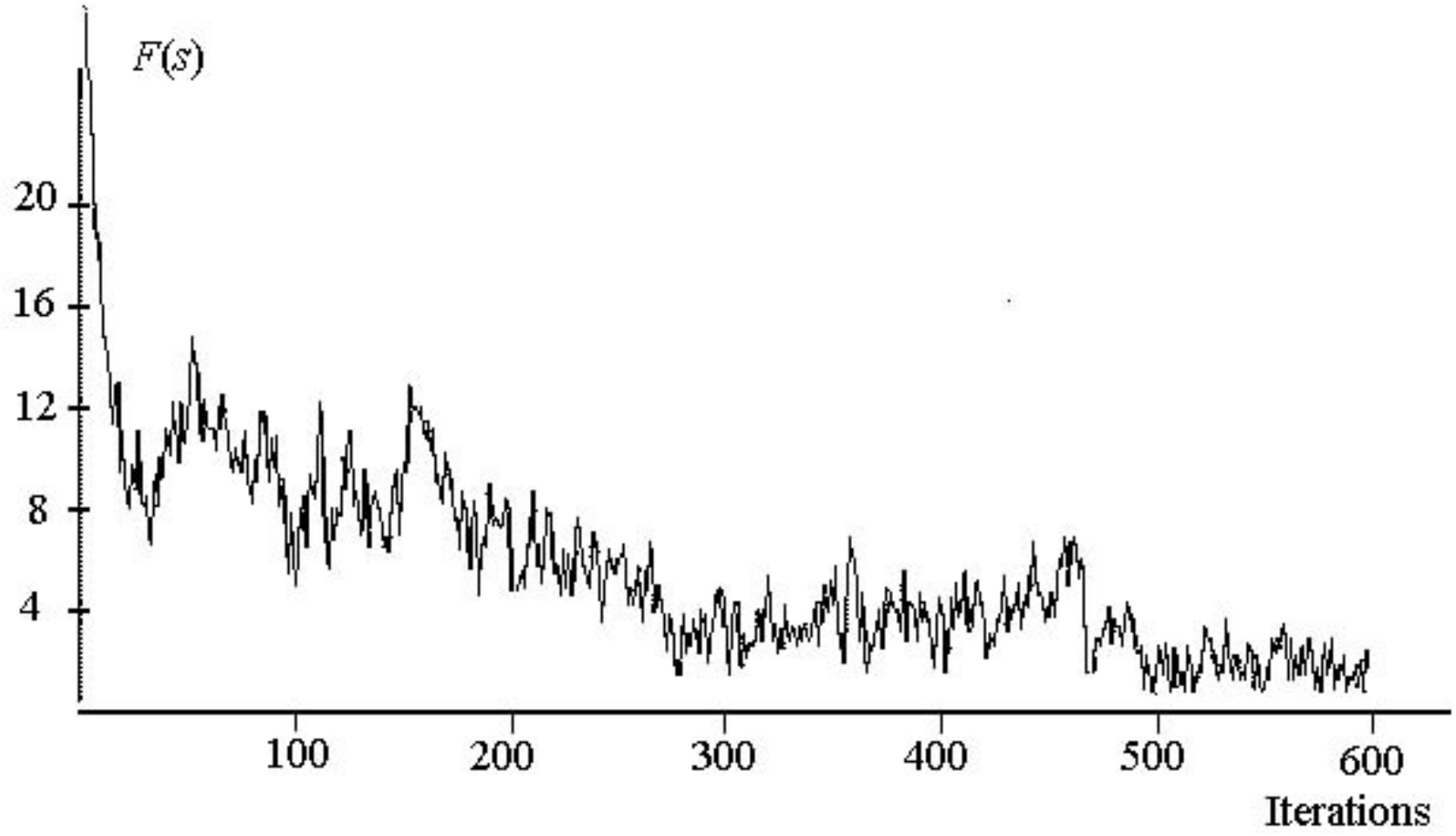


Tabu Search was introduced by F. Glover 1989

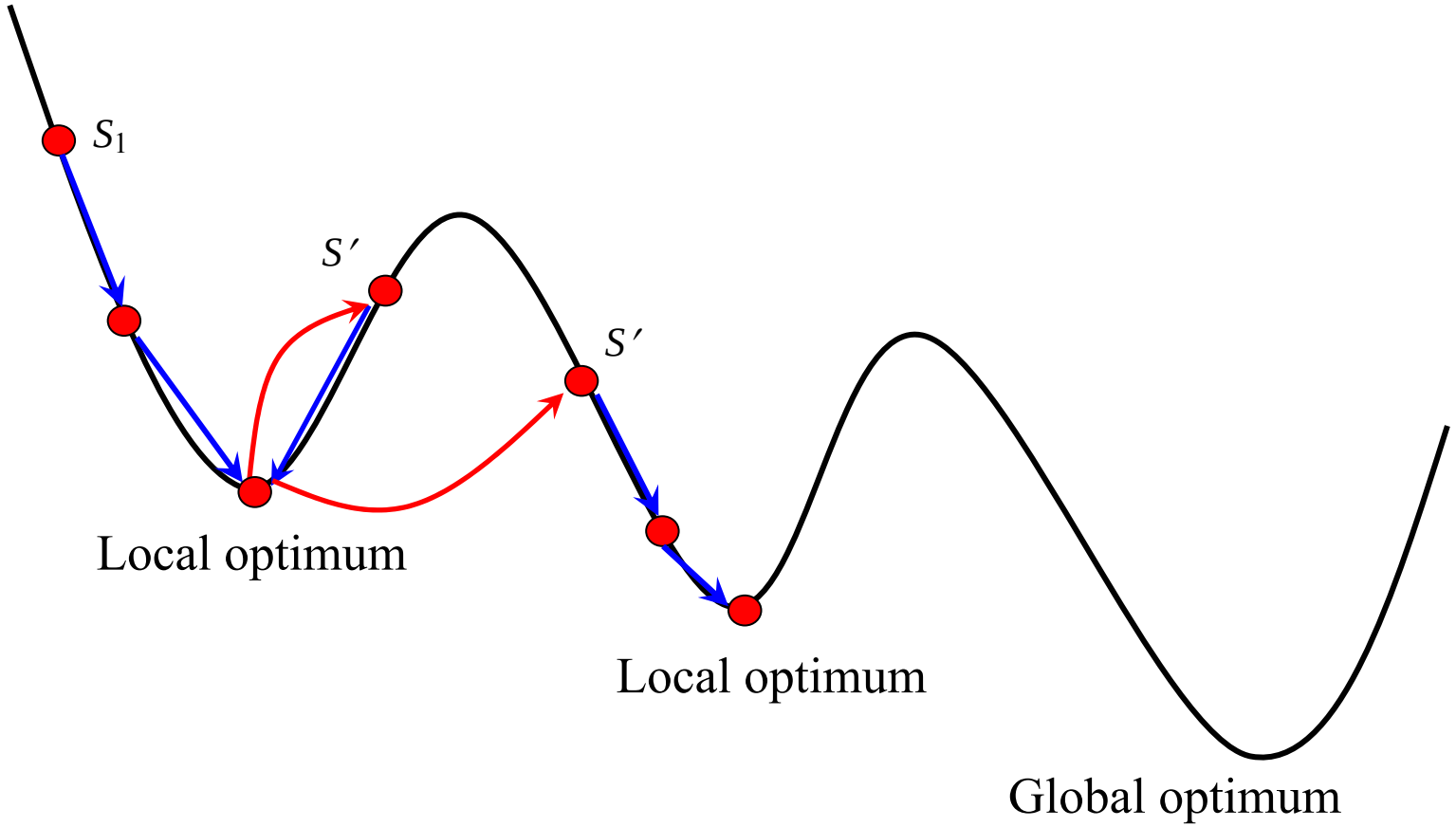
Probabilistic Tabu Search

1. Construct an initial solution $s \in Sol$; $s^* := s$;
2. Repeat until a stop criterion is satisfied
   2.1. Generate a random subneighborhood $N'(s) \subseteq N(s)$
   2.2. Select the best legal neighbor s':
   $$s' : F(s') \min\{F(s''), s'' \in N'(s) \setminus \text{Tabu}(s)\}$$
   2.3. Put $s := s'$, update TabuList
   2.4. If $F(s^*) > F(s)$ then $s^* := s$.
3. Return s^*

Typical Behavior of Tabu Search



Variable Neighborhood Search Algorithm

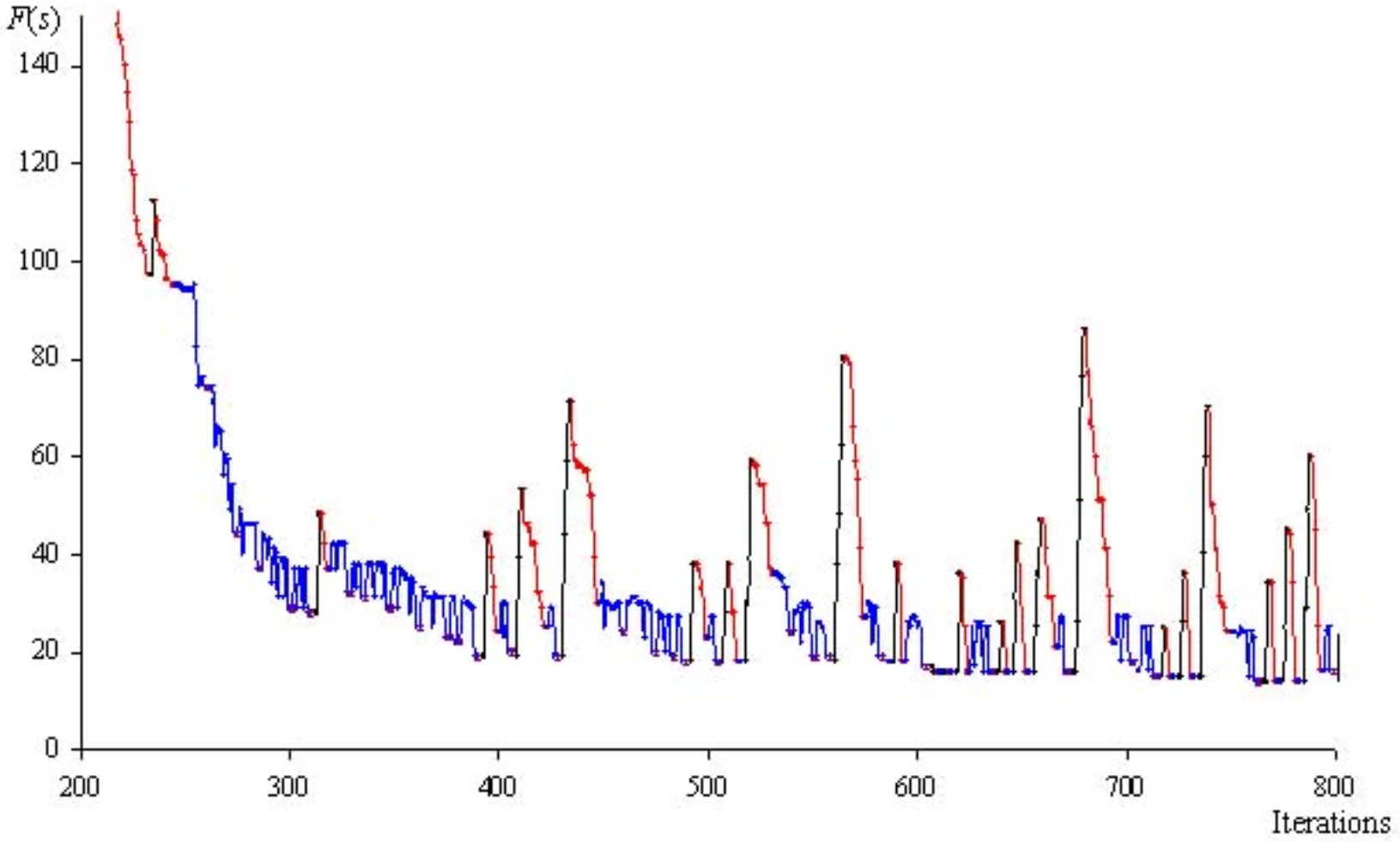


VNS was introduced by P. Hansen and N. Mladenović 1997.

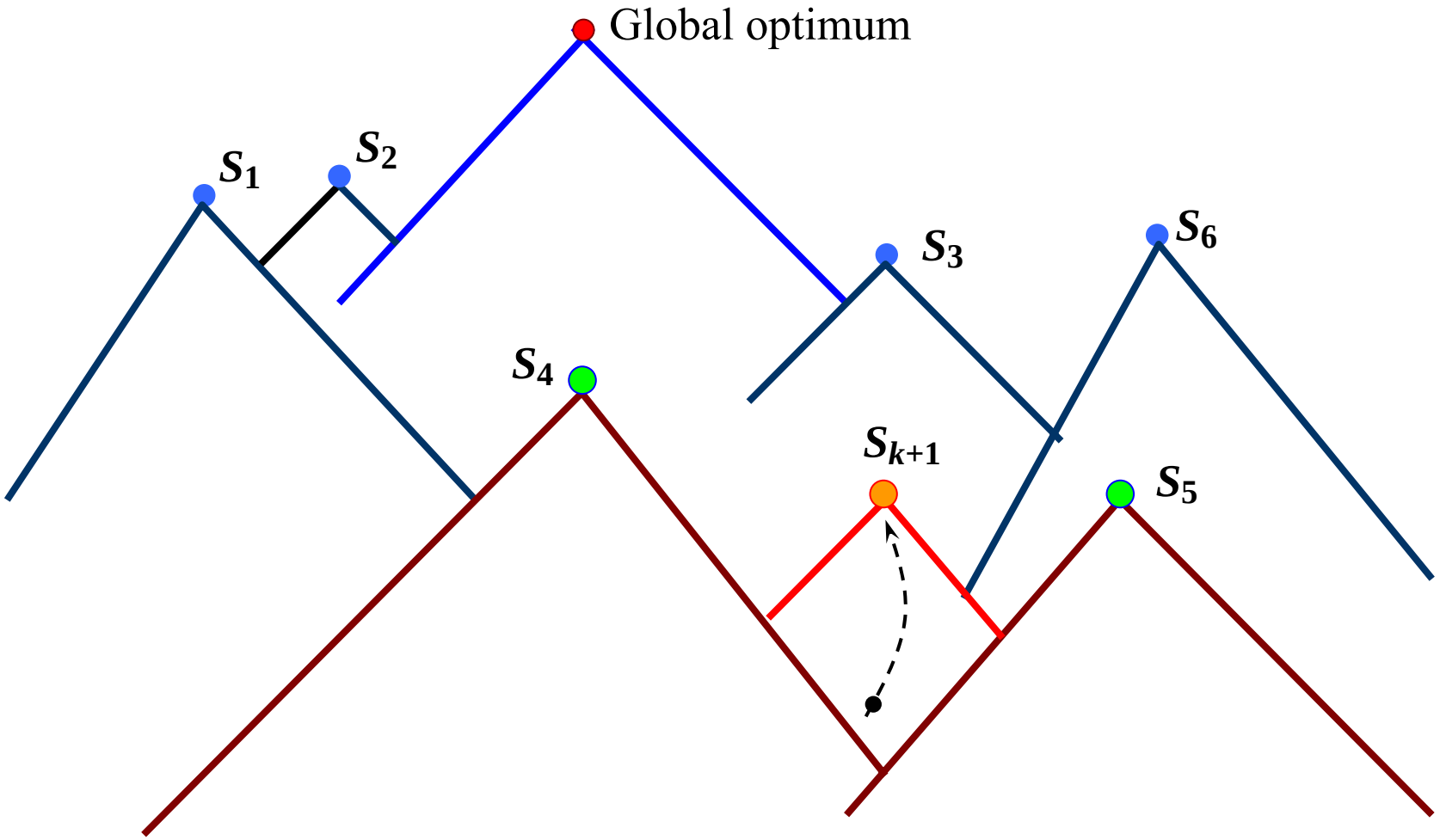
VNS Algorithm

1. Construct an initial solution s$\in$Sol, select the set of neighborhoods N_k, $k = 1,\ldots, k_{max}$
2. Repeat until a stop criterion is satisfied
    2.1. Set $k := 1$;
    2.2. Repeat until $k = k_{max}$;
        (a) Senerate $s' \in N_k(s)$ at random;
        (b) Apply some local search method with s' as initial solution; denote with s'' the so obtained local minimum;
        (c) If $F(s'') < F(s)$ then $(s := s'')$ & $(k := 1)$ else $k := k + 1$
3. Return s.

Typical Behavior of Variable Neighborhood Search



Genetic Local Search Algorithm



Genetic algorithms approach was introduced be J.H. Holland, 1975.

GLS Algorithm

1. Construct an initial population of k solutions.
2. Use local search to replace the k solutions in the population by l local optima.
3. Repeat until a stop criterion is satisfied
   3.1. Augment the population by adding m offspring solutions;
   3.2. Use local search to replace the m offspring solutions by m local optima;
   3.3. Reduce the population to its original size by selecting k solutions from the current population.
4. Return the best solution from the population.

Property of local optima

«On average, local optima are very much closer to the global optimum than are randomly chosen points and closer to each other than random point would be. The distribution of local optima is not isotropic, rather they are clustered in a big valley (or central massif for maximization problems).»

C. Reeves. 1999.