Metaheuristics for combinatorial optimization

Yury Kochetov

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

Contents

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

Combinatorial Optimization Problem

- Input:

$$\min \{f(s), s \in Sol\}$$

where *Sol* is a feasible domain.

- Output:
  - feasible solution $s \in Sol$;
- Goal:
  - find global minimum $f(s) \le f(s'),\ \ s' \in Sol.$

.

Standard local descent algorithm

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

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

3. If it exists then move:

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

4. Return local minimum s.

Examples: – Simplex method for linear programming;

– Ford–Fulkerson method for maximum flow problem;

– Bubble sort algorithm for sorting problem.

The p-median problem

- 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: find a local optimal solution

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

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

Threshold algorithms

1. Construct an initial solution $s \in Sol$, $s^* := s$, $k := 0$.
2. Repeat until a stop criterion is satisfied

   Generate $s' \in N(s)$ and put $k := k+1$;

   If $F(s') - F(s) < t_k$ then $s := s'$;

   If $F(s^*) > F(s)$ then $s^* := s$.

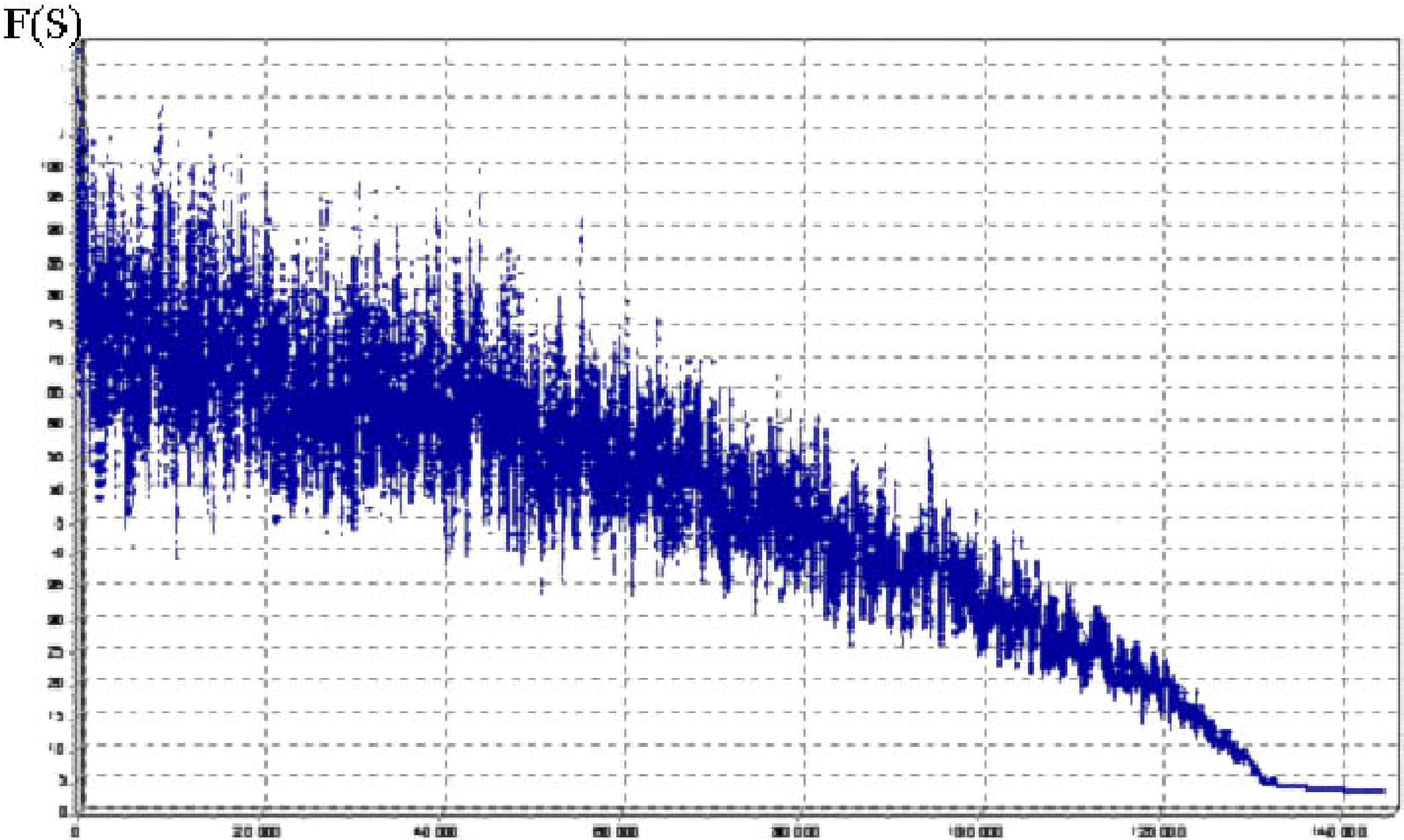
3. Return s^*.

Three Types of Threshold Algorithms

- Iterative improvement: $t_k = 0$, $k \geq 0$, it is standard LD algorithm with random pivoting rule.
- Threshold accepting: $t_k \geq 0$, $t_k \geq t_{k+1}$, $k \geq 0$, $\lim\limits_{k\to\infty} t_k = 0$.
- 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

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

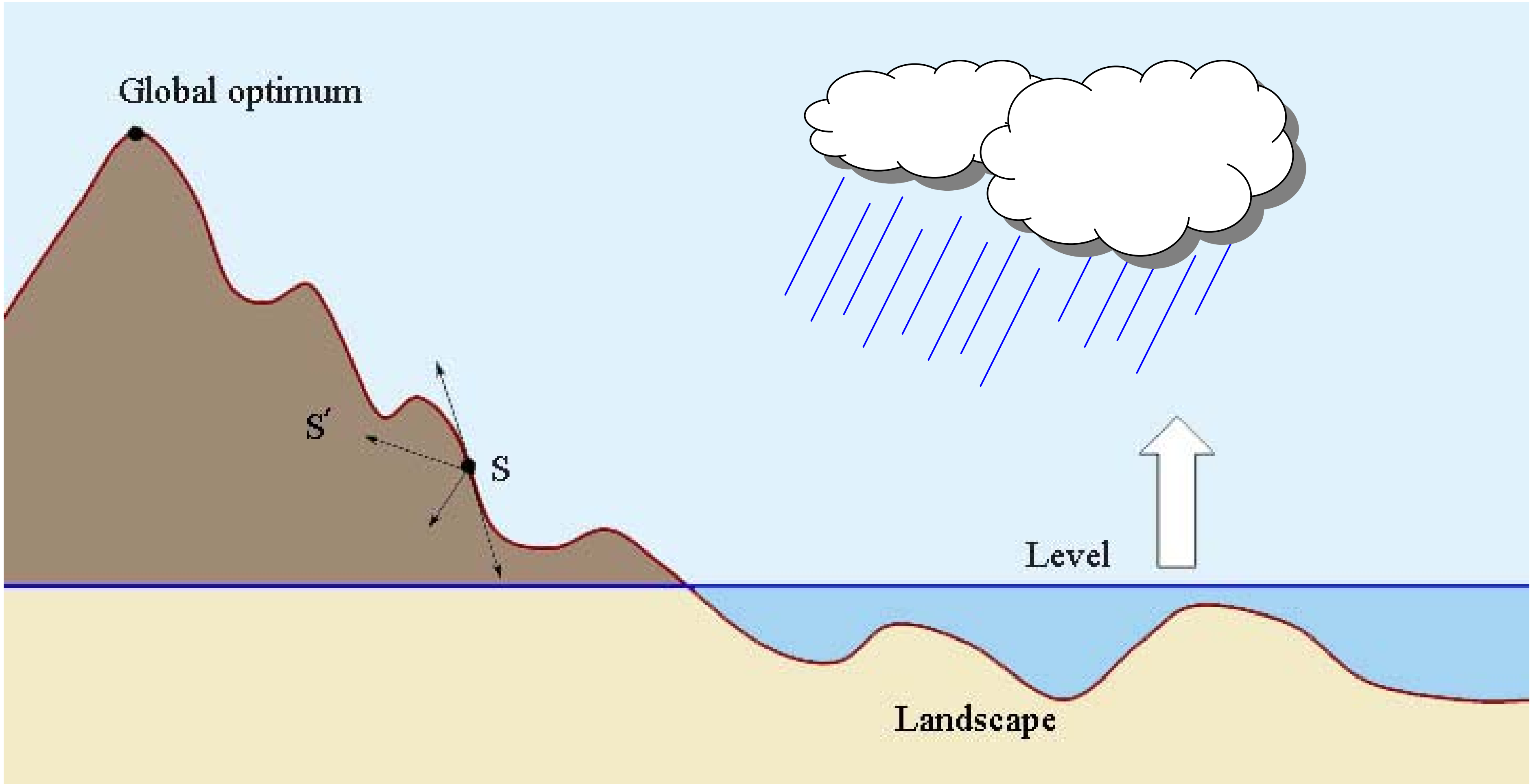
Typical Behavior of SA



Simulated Annealing Algorithm

1. Construct an initial solution s, put $t = t_{\max}$, $s^* := s$;
2. Repeat until a stop criterion is satisfied.
3. For k=1 to K do
    - 3.1 Randomly generate $s' \in N(s)$;
    - 3.2 $\Delta := F(s') - F(s)$;
    - 3.3 If $\Delta \le 0$ then $s := s'$ else $s := s'$ with probability $\exp(-\Delta/t)$;
    - 3.4 If $F(s^*) > F(s)$ then $s^* := s$;
4. Decrease the temperature $t := \gamma t$
5. Return s^*.

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 stopping criterion is satisfied

   Randomly generate $s' \in N(s)$;

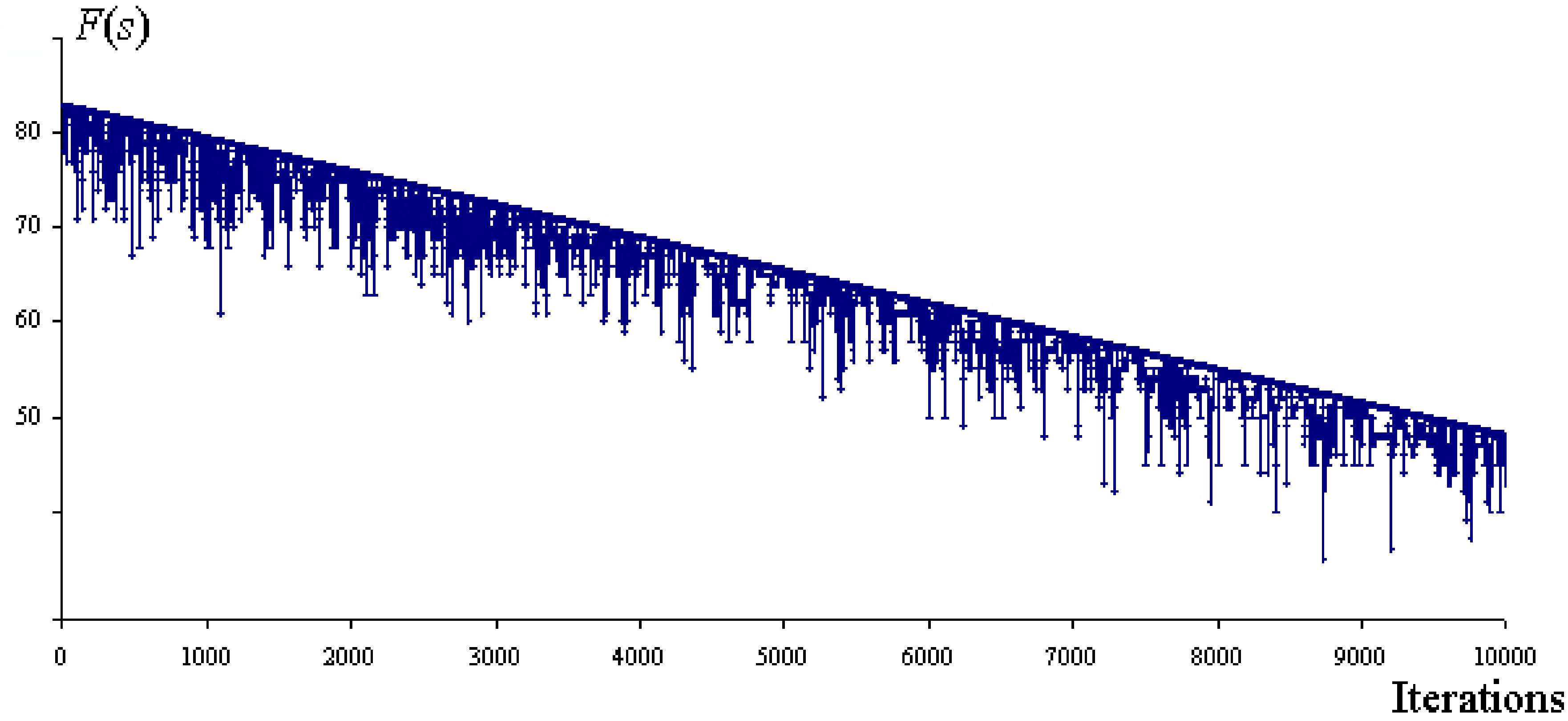
   If $F(s') \le F(s)$ then $s := s'$ else if $F(s') \le L$ then $s := s'$;

   $L := L - \Delta L$

   If $F(s^*) > F(s)$ then $s^* := s$;

3. Return s^*.

Typically behavior of Great Deluge



Probabilistic Tabu Search

1. Construct an initial solution $s \in Sol$; $s^* := s$;
2. Repeat until a stopping criterion is satisfied

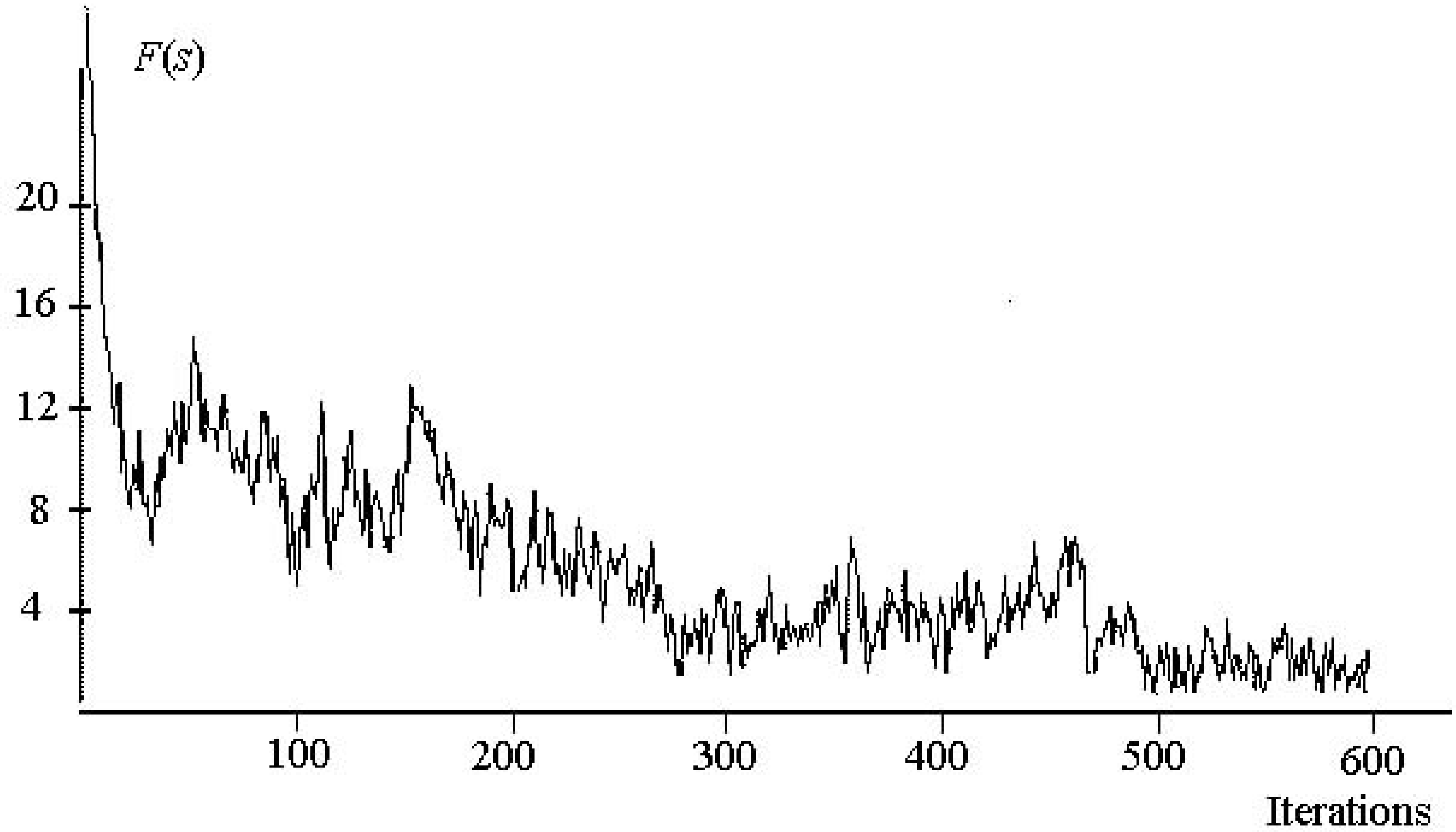
   2.1 Generate a random subneighborhood $N'(s) \subseteq N(s)$

   2.2 Select the best legal neighboring solution 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

1. Construct an initial solution $s \in Sol$, select the set of neighborhoods N_k, $k = 1,\ldots, k_{max}$

2. Repeat until a stopping criterion is satisfied

Set $k := 1$;

Repeat until $k = k_{max}$;

(a) Generate $s' \in N_k(s)$ at random;

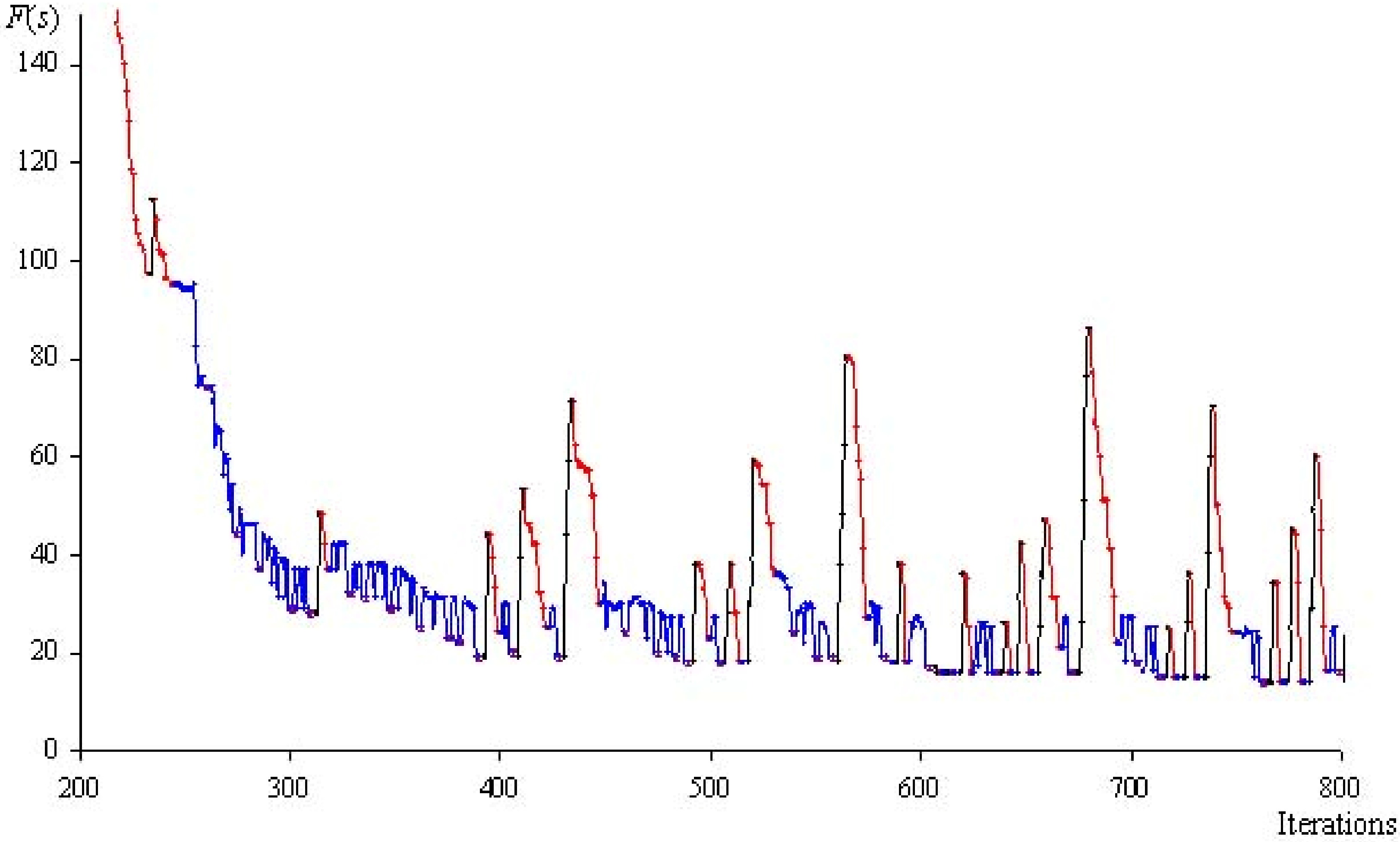
(b) Apply some local search method with s' as initial solution; denote with s'' the so obtained local minimum;

(c) If $F(s'') < F(s)$ then $(s := s'')$ & $(k := 1)$ else $k := k + 1$

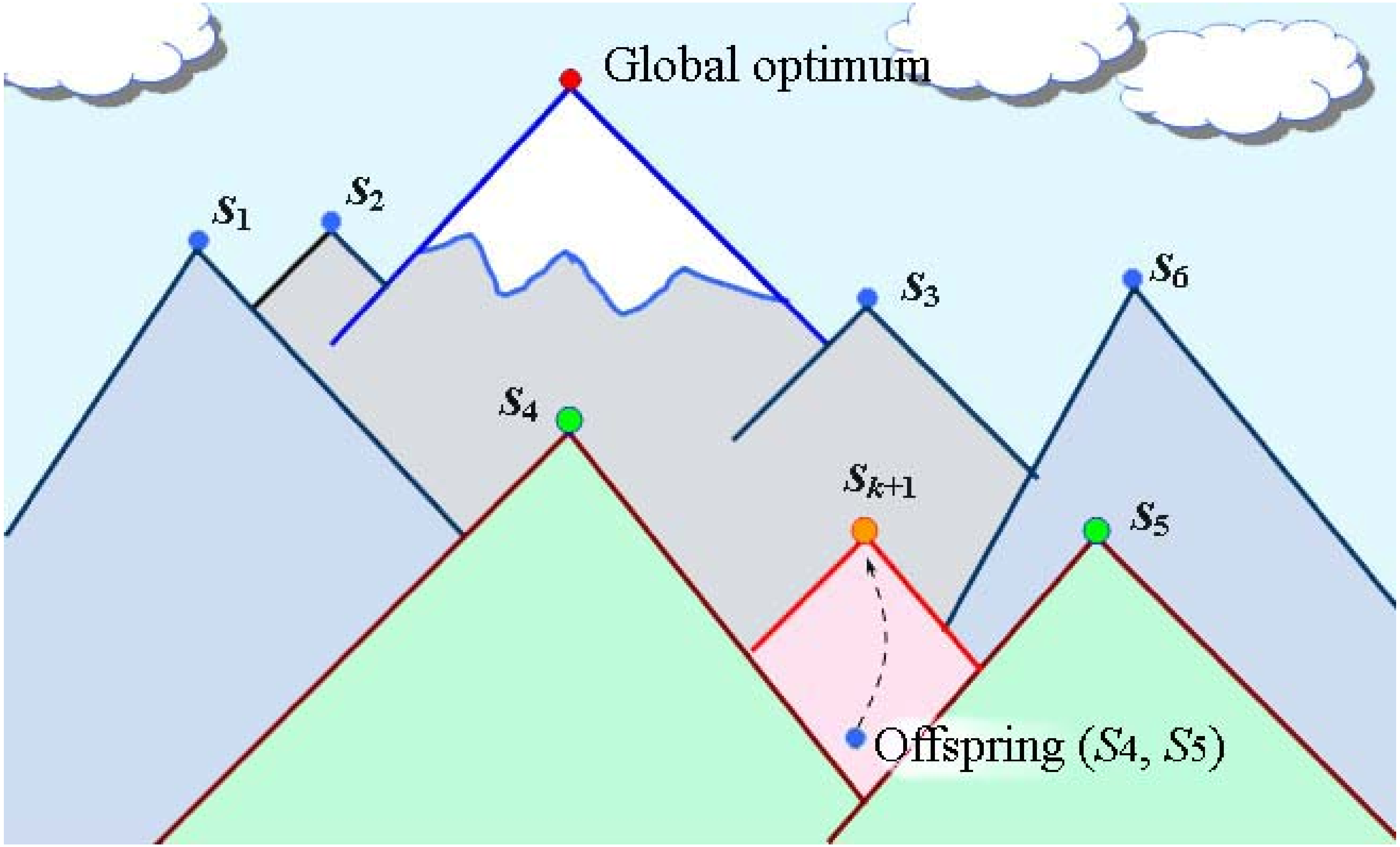
3. Return s.

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

Typical Behavior of Variable Neighborhood Search



Genetic Local Search Algorithm



Genetic approach was introduced by 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

   Augment the population by adding *m* offspring solutions;
   Use local search to replace the *m* offspring solutions by *m* local optima;
   Reduce the population to its original size by selecting *k* solutions from the current population.
4. Return the best solution from the population.


Easy case


Easy case


Hard case
Copyright : E. Beaudo


Extremely hard case

[General information](#) • [Authors](#) • [Other libraries](#) • [Russian page](#)

Discrete Location Problems

Benchmark library

- [Simple Plant Location Problem](#)
- [Capacitated Facility Location Problem](#)
- [Multi Stage Uncapacitated Facility Location Problem](#)
- [P-median Problem](#)
- Bilevel Location Problem

Benchmark library was created thanks to financial support
by Russian Foundation for Basic Research (grants NN 98-07-90259, 01-07-90212, 04-07-90096)

http://math.nsc.ru/AP/benchmarks/english.html