Heuristic Algorithms

Master's Degree in Computer Science/Mathematics

Roberto Cordone

DI - Università degli Studi di Milano

Schedule:	Thursday 14.30 - 16.30 in classroom 503
	Friday 14.30 - 16.30 in classroom 503
Office hours:	on appointment
E-mail:	roberto.cordone@unimi.it
Web page:	https://homes.di.unimi.it/cordone/courses/2024-ae/2024-ae.html
Ariel site:	https://myariel.unimi.it/course/view.php?id=4466

Exchange algorithms

In Combinatorial Optimization every solution x is a subset of B

An exchange heuristic updates a current subset $x^{(t)}$ step by step

1. start from a feasible solution $x^{(0)} \in X$ found somehow
   (*often by a constructive heuristic*)
2. generate a family of feasible solutions by exchanging elements, i.e. add subsets A external to $x^{(t)}$ and delete subsets D internal to $x^{(t)}$

$$x'_{A,D} = x \cup A \setminus D \text{ with } A \subseteq B \setminus x \text{ and } D \subseteq x$$

3. use a selection criterium $\varphi(x, A, D)$ to choose the subsets to exchange

$$(A^*, D^*) = \arg\min_{(A,D)} \varphi(x, A, D)$$

4. perform the chosen exchange to generate the new current solution

$$x^{(t+1)} := x^{(t)} \cup A^* \setminus D^*$$

5. if a termination condition holds, terminate;
   otherwise, go back to point 2

Neighbourhood

An exchange heuristic is defined by:

1. the pairs of exchangeable subsets (A, D) in every solution x, i.e. the solutions generated by a single exchange starting from x
2. the selection criterium $\varphi(x, A, D)$

Neighbourhood $N : X \to 2^X$ is a function which associates to each feasible solution $x \in X$ a subset of feasible solutions $N(x) \subseteq X$

The situation can be formally described with a search graph in which

- the nodes represent the feasible solutions $x \in X$
- the arcs connect each solution x to those of its neighbourhood $N(x)$, moving elements into and out of x (they are often denoted as moves)

Every run of the algorithm corresponds to a path in its search graph

How does one define a neighbourhood and select a move?

Neighbourhoods defined on the basis of distance

Every solution $x \in X$ can be represented by its incidence vector

$$\xi_i(x) = \begin{cases} 1 & \text{if } i \in x \\ 0 & \text{if } i \in B \setminus x \end{cases}$$

Hamming distance between two solutions x and x'
is the number of elements in which their incidence vectors differ

$$d_H(x, x') = \sum_{i \in B} |\xi_i(x) - \xi_i(x')|$$

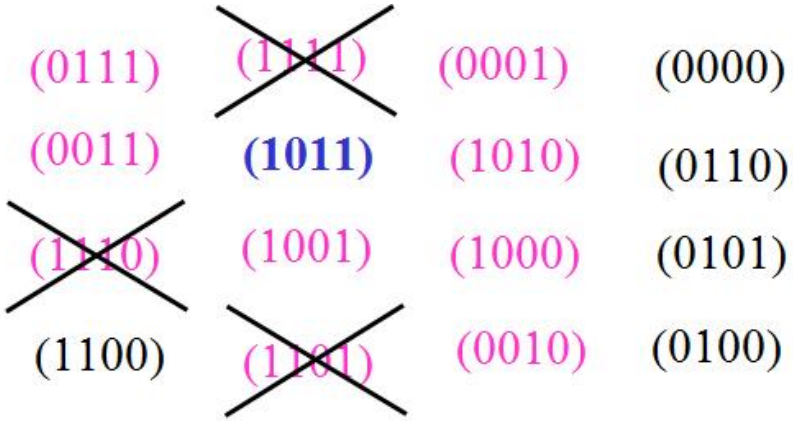
Referring to the subsets, $d_H(x, x') = |x \setminus x'| + |x' \setminus x|$

A typical definition of neighbourhood, with an integer parameter k, is the set of all solutions with a Hamming distance from x not larger than k

$$N_{H_k}(x) = \{x' \in X : d_H(x, x') \leq k\}$$

Example: the *KP*

The *KP* instance with $B = \{1, 2, 3, 4\}$, $v = [\ 5\ 4\ 3\ 2\ ]$ and $V = 10$, has 13 feasible solutions out of 16 subsets



since subsets $\{1, 2, 3, 4\}$, $\{1, 2, 3\}$ and $\{1, 2, 4\}$ are unfeasible

10 subsets (pink) have Hamming distance ≤ 2 from $x = \{1, 3, 4\}$ (blue)

The neighbourhood $N_{H_2}(x)$ consists of the 7 feasible subsets in pink

$N_{H_2}(x)$ excludes

- the 3 crossed subsets in pink because they are unfeasible
- the 5 subsets in black because their Hamming distance from x is > 2

x is part of the neighbourhood, but useless

Neighbourhoods defined on the basis of operations

A more general definition of neighbourhood considers

- a family $\mathcal{O}$ of operations on the solutions of the problem
- the set of all solutions generated applying to x the operations of $\mathcal{O}$

$$N_{\mathcal{O}}(x) = \{x' \in X : \exists o \in \mathcal{O} : o(x) = x'\}$$

Considering again the *KP*, $\mathcal{O}$ can be defined as the union of

- $\mathcal{A}_1$: adding to x one element of $B \setminus x$
- $\mathcal{D}_1$: deleting from x at most one element (*to impose* $x \in N(x)$)
- $\mathcal{S}_1$: swapping one element of x with one of $B \setminus x$

The resulting neighbourhood $N_{\mathcal{O}}$ is related to those defined by the Hamming distance, but does not coincide with any of them

$$N_{H_1} \subset N_{\mathcal{O}} \subset N_{H_2}$$

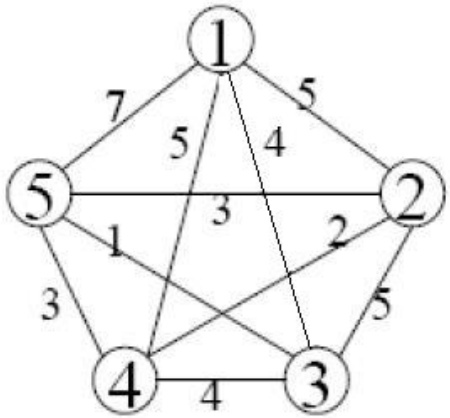
As the distance-based ones, these neighbourhoods can be parameterised considering sequences of k operations of $\mathcal{O}$ instead of a single one

$$N_{\mathcal{O}_k}(x) = \{x' \in X : \exists o_1, \ldots, o_k \in \mathcal{O} : o_k(o_{k-1}(\ldots o_1(x))) = x'\}$$

Distance and operation-based neighbourhoods

In general, an operation-based neighbourhood includes solutions with different Hamming distances from x

For the *TSP* one can define a neighbourhood $N_{\mathcal{S}_1}$ including the solutions obtained swapping two nodes in their visit order



The neighbourhood of solution $x = (3, 1, 4, 5, 2)$ is:

$$N_{\mathcal{S}_1}(x) = \{(1,3,4,5,2), (4,1,3,5,2), (5,1,4,3,2), (2,1,4,5,3), (3,4,1,5,2), (3,5,4,1,2), (3,2,4,5,1), (3,1,5,4,2), (3,1,2,5,4), (3,1,4,2,5)\}$$

If the two nodes are adjacent, the modified arcs are $3+3$; otherwise, they are $4+4$

Distance and operation-based neighbourhoods: relations

Some neighbourhoods can be naturally defined in both ways

- for the *MDP*
  - neighbourhood N_{H_2} (solutions at Hamming distance equal to 2)
  - neighbourhood $N_{\mathcal{S}_1}$ (swap one element of x with one of $B \setminus x$)
- for the *BPP*
  - neighbourhood N_{H_2} (solutions at Hamming distance equal to 2)
  - neighbourhood $N_{\mathcal{T}_1}$ (transfer an object into a different container)
- for *Max-SAT*
  - neighbourhood N_{H_2} (solutions at Hamming distance equal to 2)
  - neighbourhood $N_{\mathcal{F}_1}$ ("flip" a variable: invert its truth assignment)

This is typical of problems with solutions of fixed cardinality:

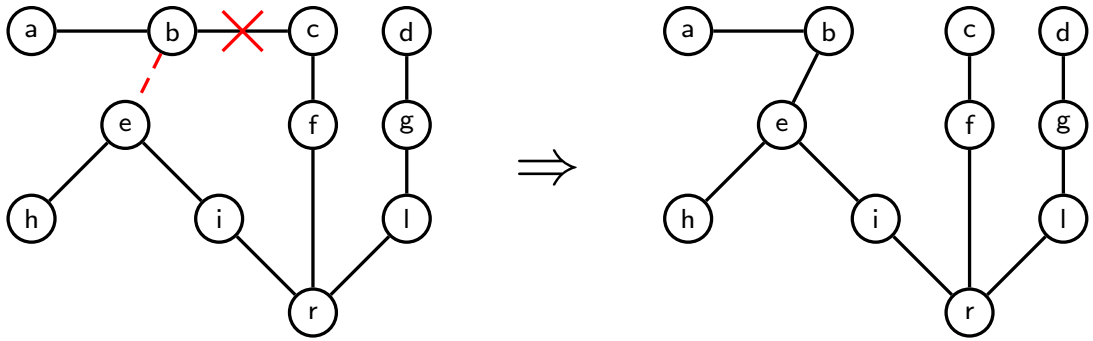
- perform a sequence of k swaps between single elements ($|A| = |D| = 1$): k elements go into x and k elements out of it
- the Hamming distance between the two extreme solutions is $\leq 2k$ (*if all exchanged elements are different, it is exactly* $2k$)

Different neighbourhoods for the same problem: the *CMST*

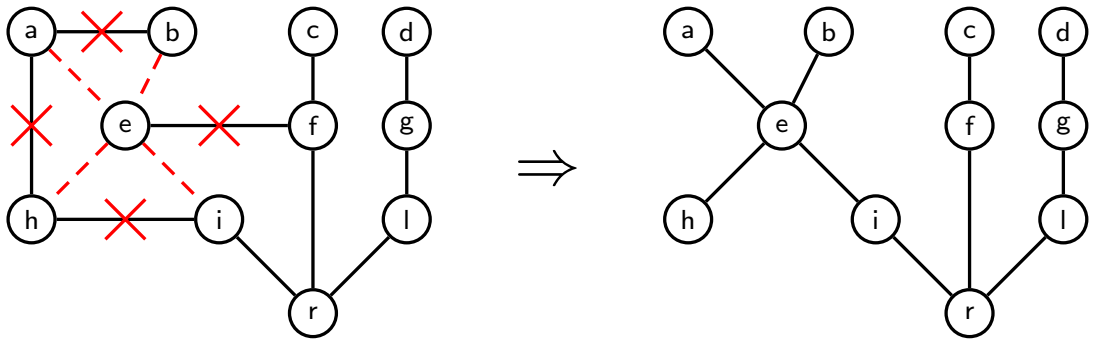
Different ground sets yield different neighbourhoods

In the *CMST* it is possible to set $B = E$ or $B = V \times T$ and

- exchange edges: delete (b, c) and add (b, e), making a swap



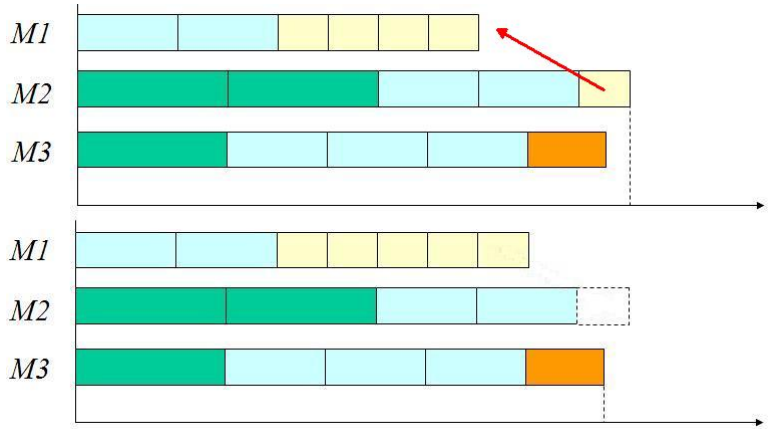
- exchange vertices: transfer e from subtree 2 to subtree 1, and recompute the two minimum spanning subtrees



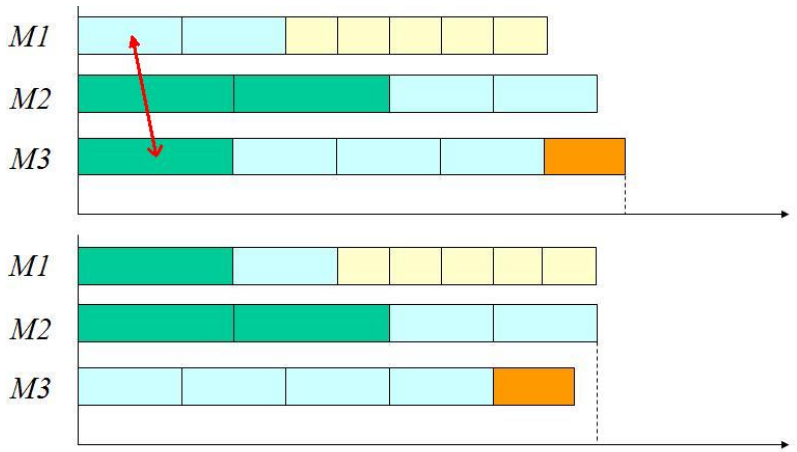
Different neighbourhoods for the same problem: the *PMSP*

For the *PMSP* it is possible to define

- the transfer neighbourhood $N_{\mathcal{T}_1}$, based on the set $\mathcal{T}_1$ of all transfers of a task on another machine



- the swap neighbourhood $N_{\mathcal{S}_1}$, based on the set $\mathcal{S}_1$ of the swaps of two tasks between two machines (one task for each machine)



Connectivity of the search graph

An exchange heuristic can return the optimum only if
every feasible solution can reach at least one optimal solution, that is
there is a path from x to X^* for every $x \in X$

Such a search graph is denoted as weakly connected to the optimum

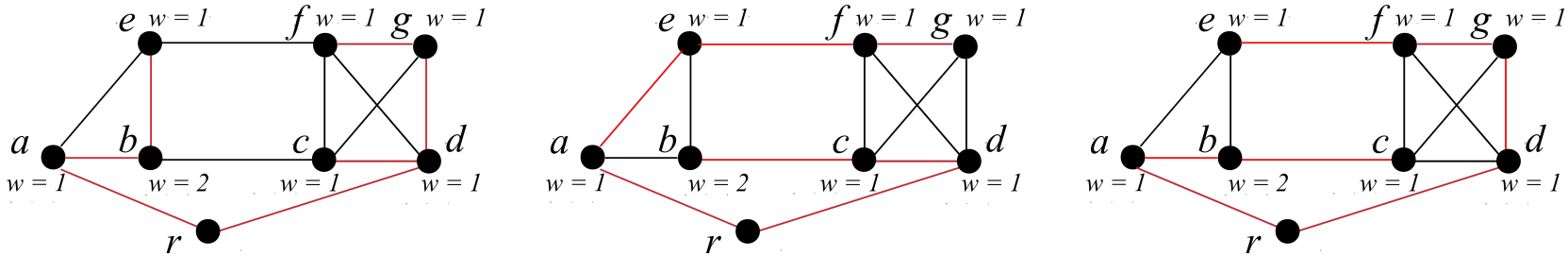
Since X^* is unknown, a stronger condition is often used: a search graph is strongly connected when it admits a path from x to y for every $x, y \in X$

A good neighbourhood should guarantee some connectivity conditions

- in the *MDP*, neighbourhood $N_{\mathcal{S}_1}$ connects any pair of solutions with at most k swaps
- in the *KP* and the *SCP*, no neighbourhood $N_{\mathcal{S}_k}$ gives that guarantee
  (*feasible solutions can have any cardinality*)
- the search graph becomes connected also in the *KP* and the *SCP* if swaps are combined with both additions and deletions

Connectivity of the search graph

If feasibility is defined in a sophisticated way, exchanging, adding and deleting **single** elements can be insufficient to reach all solutions:
the unfeasible subsets can break all paths between some feasible solutions



If $V = 4$, only three solutions are feasible, all with two subtrees:

- $x = \{(r, a), (a, b), (b, e), (r, d), (c, d), (d, g), (f, g)\}$
- $x' = \{(r, a), (a, e), (e, f), (r, d), (c, d), (b, c), (f, g)\}$
- $x'' = \{(r, a), (a, b), (e, f), (r, d), (b, c), (d, g), (f, g)\}$

The three solutions are mutually reachable only exchanging at least two edges at at time; exchanging only one yields unfeasible subsets

Steepest descent (*hill-climbing*) heuristics

The simplest selection criterium $\varphi(x, A, D)$ is the objective function

It is used in nearly all exchange heuristics

When $\varphi(x, A, D) = f(x \cup A \setminus D)$, the heuristic moves from $x^{(t)}$ to the best solution in $N\left(x^{(t)}\right)$

To avoid cyclic behaviour, only strictly improving solutions are accepted
Consequently, the best known solution is the last visited one

Algorithm SteepestDescent$\left(I, x^{(0)}\right)$

$x := x^{(0)}$;

Stop := *false*;

While Stop = *false do*

$\tilde{x} := \arg \min_{x' \in N(x)} f(x')$;

If $f(\tilde{x}) \geq f(x)$ *then* Stop := *true*; *else* $x := \tilde{x}$;

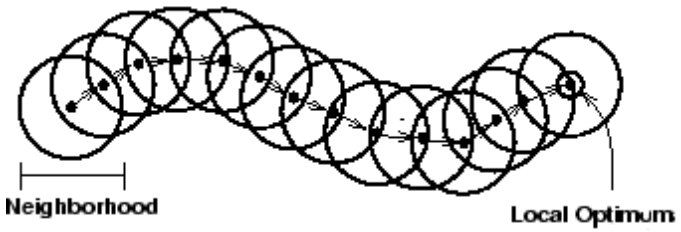
EndWhile;

Return $(x, f(x))$;

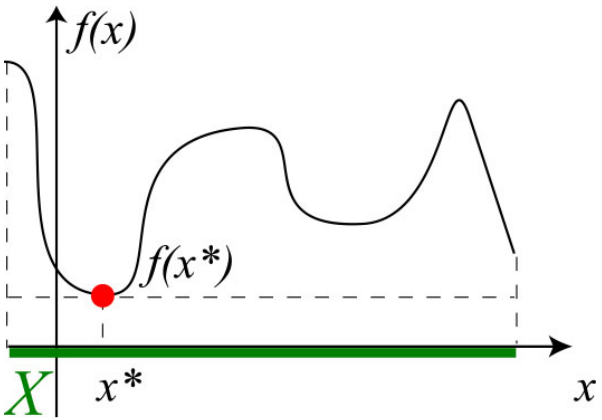
Local and global optimality

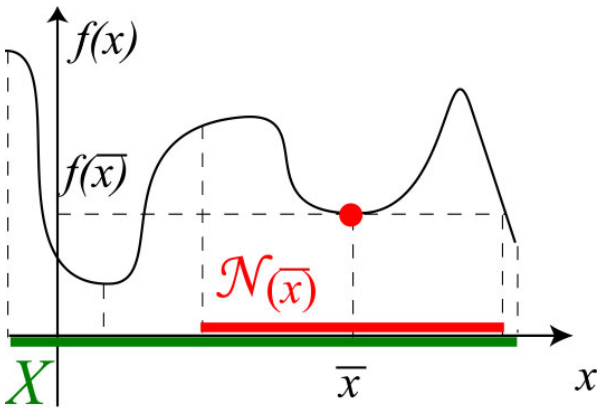
A *steepest descent* heuristic terminates when it finds a locally optimal solution, that is a solution $\bar{x} \in X$ such that

$$f(\bar{x}) \leq f(x) \text{ for each } x \in N(x)$$



A globally optimal solution is always also locally optimal, but the opposite is not true in general: $X^* \subseteq \bar{X}_N \subseteq X$





Exact neighbourhood

Exact neighbourhood is a neighbourhood function $N : X \to 2^X$ such that each local optimum is also a global optimum

$$\bar{X}_N = X^*$$

Trivial case: the neighbourhood of each solution coincides with the whole feasible region ($N(x) = X$ for each $x \in X$)

It is a useless neighbourhood: too wide to explore

The exact neighbourhoods are extremely rare

- exchange between edges for the Minimum Spanning Tree problem
- exchange between basic and nonbasic variables used by the simplex algorithm for Linear Programming

In general, the *steepest descent* heuristic does not find a global optimum

Its effectiveness depends on the properties of search graph and objective

Properties of the search graph (1)

Some relevant properties for the effectiveness of an algorithm are

- the size of the search space $|X|$
- the connectivity of the search graph (as discussed above)
- the diameter of the search graph, that is the number of arcs of the minimum path between the two farthest solutions: larger neighbourhoods produce graphs of smaller diameter
  (*but other factors exist: see the "smallworld" effect*)

Consider neighbourhood N_{S_1} for the symmetric *TSP* on complete graphs

- the search space includes $|X| = (n-1)!$ solutions
- N_{S_1} (swap of two nodes) includes $\binom{n}{2} = \frac{n(n-1)}{2}$ solutions
- the search graph is strongly connected and has diameter $\leq n-2$: every solution turns into another after at most $n-2$ swaps

  For example, $x = (1, 5, 4, 2, 3)$ becomes $x' = (1, 2, 3, 4, 5)$ in 3 steps

$$x = (1, 5, 4, 2, 3) \rightarrow (1, 2, 4, 5, 3) \rightarrow (1, 2, 3, 5, 4) \rightarrow (1, 2, 3, 4, 5) = x'$$

(*the first node is always* 1, *the last one is automatically in place*)

Properties of the search graph (2)

Other relevant properties

- the density of global optima ($\frac{|X^*|}{|X|}$) and local optima ($\frac{|\bar{X}_N|}{|X|}$):
  if the local optima are numerous, it is hard to find the global ones
- the distribution of the quality $\delta(\bar{x})$ of local optima (SQD diagram):
  if local optima are good, it is less important to find a global one
- the distribution of the locally optimal solutions in the search space:
  if local optima are close to each other, it is not necessary to explore the whole space

These indices would require an exhaustive exploration of the search graph

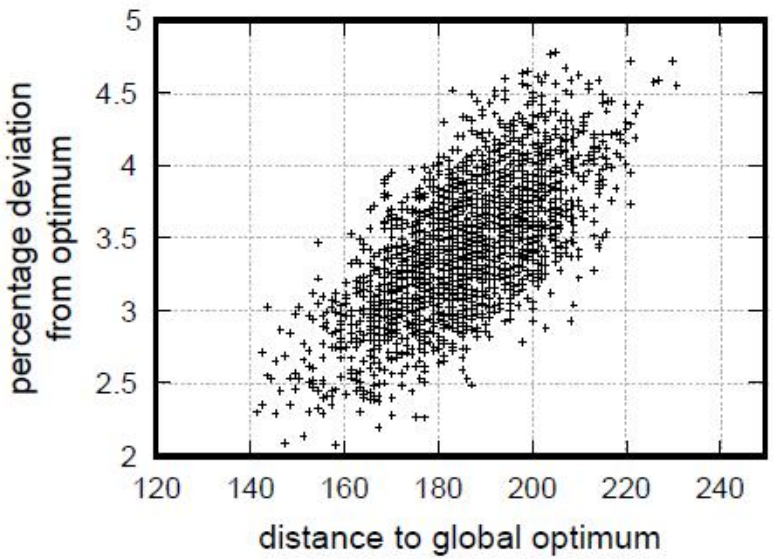
In pratice, one performs a sampling and these analyses

- require very long times
- can be misleading, especially if the global optima are unknown

Example: the *TSP*

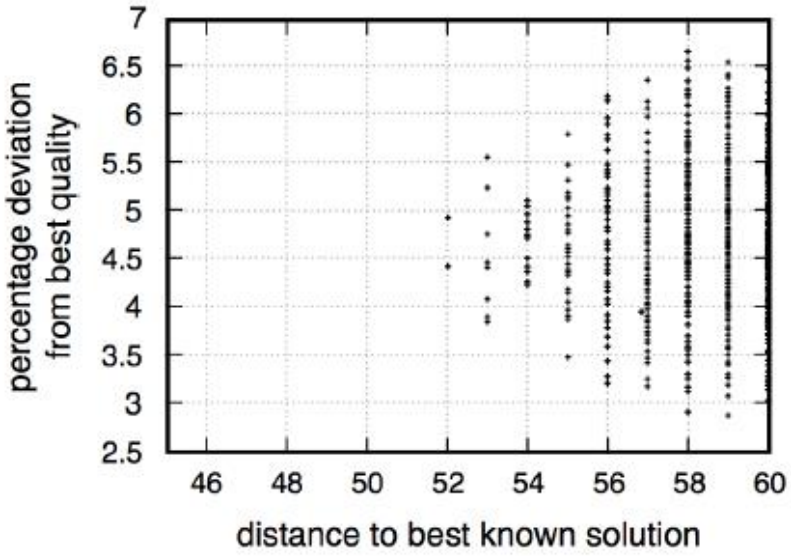
For the *TSP* on a complete symmetric graph with Euclidean costs

- the Hamming distance between two local optima is on average $\ll n$: the local optima concentrate in a small region of X
- the Hamming distance between local optima on average exceeds that between local and global optima:
  the global optima tend to concentrate in the middle of local optima
- the *FDC* diagram (*Fitness-Distance Correlation*) reports the quality $\delta(\bar{x})$ versus the distance from global optima $d_H(\bar{x}, X^*)$: if they are correlated, better local optima are closer to the global ones



Fitness-Distance Correlation

For the *Quadratic Assignment Problem* (*QAP*), the situation is different



If quality and closeness to the global optima are strongly correlated

- it is profitable to build good starting solutions,
  because they drive the search near a good local optimum
- it is better to intensify than to diversify

If the correlation is weak

- a good inizialization is less important
- it is better to diversify than to intensify

Landscape

The landscape is the triplet (X, N, f), where

- X is the search space, or the set of feasible solutions
- $N : X \to 2^X$ is the neighbourhood function
- $f : X \to \mathbb{N}$ is the objective function

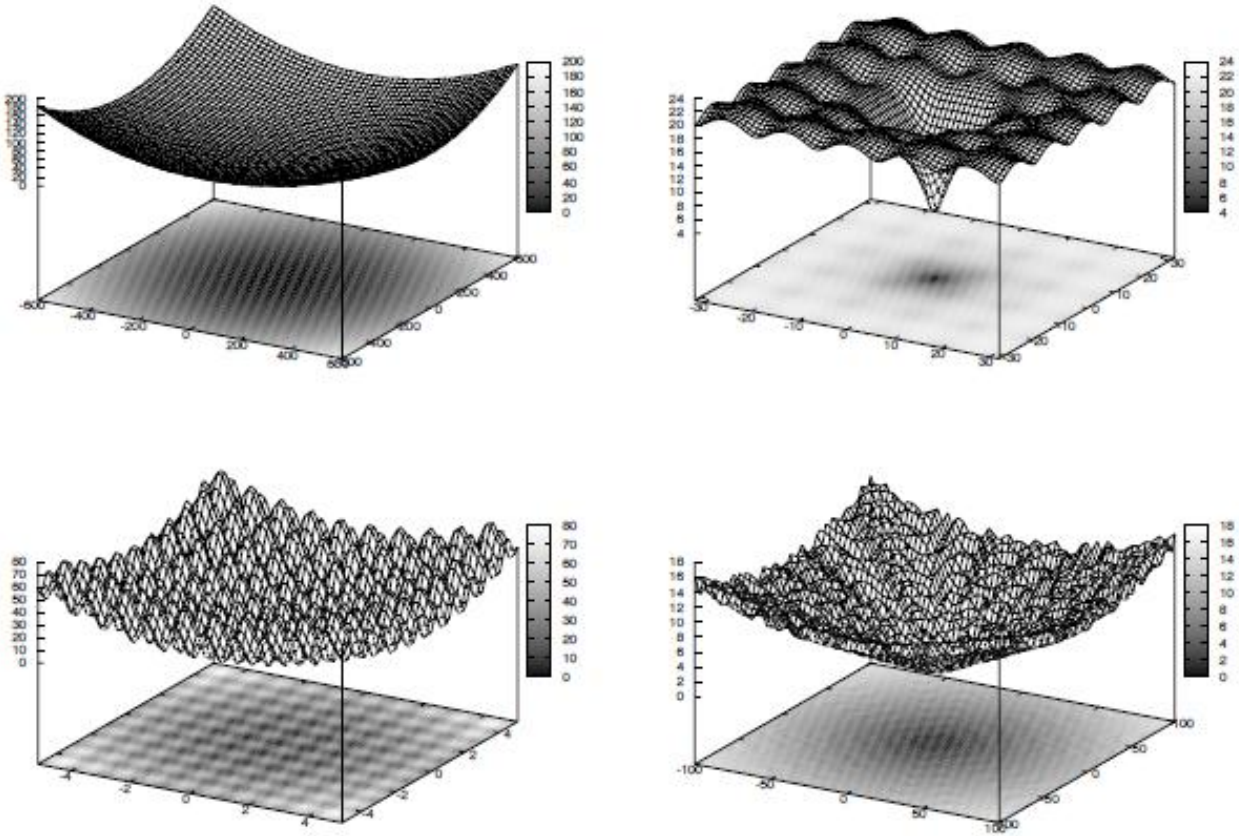
It is the search graph with node weights given by the objective

The effectiveness of steepest descent depends on the landscape

- smooth landscapes yield few local optima, possibly of good quality, hence to good results
- rugged landscapes yield several local optima of widespread quality, hence to bad results

Different kinds of landscape

There is a great variety of landscapes, very different from one another



Autocorrelation coefficient (1)

The complexity of a landscape can be empirically estimated

1. performing a *random walk* in the search graph
2. determining the sequence of values of the objective $f^{(1)}, \ldots, f^{(t_{\max})}$
3. computing the sample mean $\bar{f} = \dfrac{\sum_{t=1}^{t_{\max}} f^{(t)}}{t_{\max}}$
4. computing the empirical autocorrelation coefficient

$$r(i) = \frac{\dfrac{\sum_{t=1}^{t_{\max}-i} \left(f^{(t)} - \bar{f}\right)\left(f^{(t+i)} - \bar{f}\right)}{t_{\max} - i}}{\dfrac{\sum_{t=1}^{t_{\max}} \left(f^{(t)} - \bar{f}\right)^2}{t_{\max}}}$$

that relates the difference of the objective values in the solutions visited with the distance between these solutions along the walk

Autocorrelation coefficient (2)

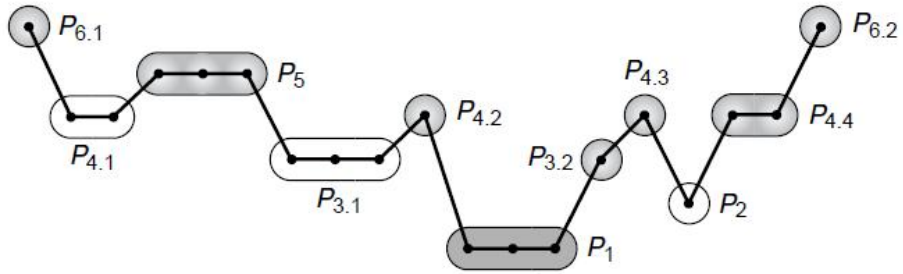
$$r(i) = \frac{\frac{\sum_{t=1}^{t_{max}-i} \left(f^{(t)} - \bar{f}\right)\left(f^{(t+i)} - \bar{f}\right)}{t_{max} - i}}{\frac{\sum_{t=1}^{t_{max}} \left(f^{(t)} - \bar{f}\right)^2}{t_{max}}}$$

- $r(0) = 1$ (*perfect correlation at* 0 *distance*)
- in general $r(i)$ decreases as the distance i increases
- if $r(i) \approx 1$ in a large range of distances, the landscape is smooth:
  - the neighbour solutions have values close to the current one
  - there are few local optima
  - the *steepest descent* heuristic is effective
- if $r(i)$ varies steeply, the landscape is rugged:
  - the neighbour solutions have values far from the current one
  - there are many local optima
  - the *steepest descent* heuristic is ineffective

Plateau

The search graph can be partitioned according to the objective value

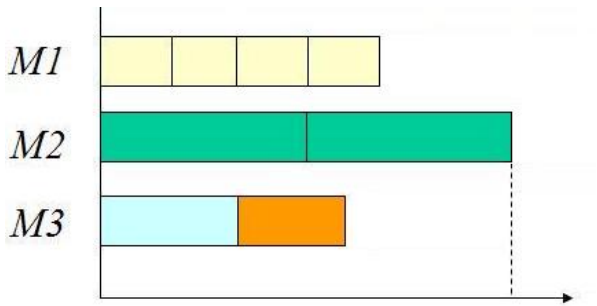
- plateau of value f is each subset of solutions of value f that are adjacent in the search graph



Large plateaus complicate the choice of the solution: most neighbours are equivalent, and the choice ends up depending on the visit order

An extremely uniform landscape is not an advantage!

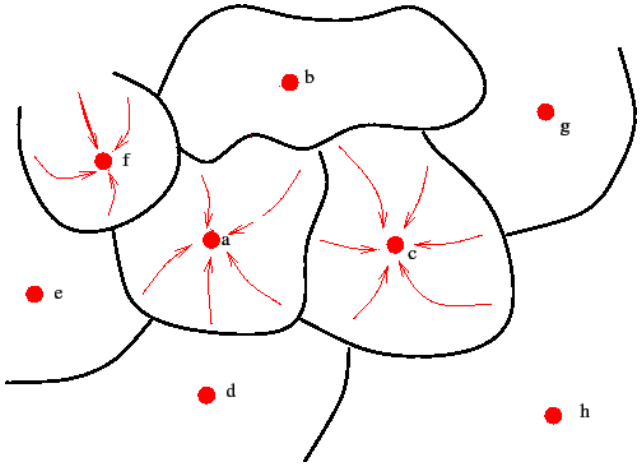
Example: all transfers and swaps between machines 1 and 3 leave the objective value unchanged (*most other moves worsen it*)

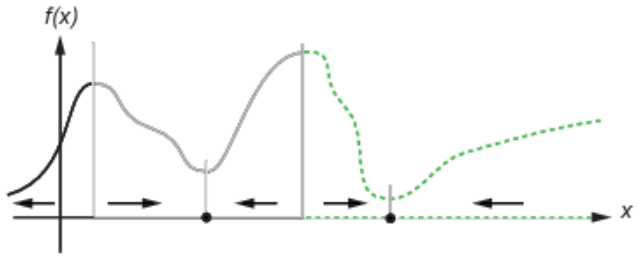


Attraction basins

Alternatively, the search graph can be partitioned into:

- attraction basins of the locally optimal solutions $\bar{x}$, that are the subsets of solutions $x^{(0)} \in X$ starting from which the *steepest descent* heuristic terminates in $\bar{x}$





The *steepest descent* heuristic is

- effective if the attraction basins are few and large (especially if the global optima have larger basins)
- ineffective if the attraction basins are many and small (especially if the global optima have smaller basins)