DM63
HEURISTICS FOR
COMBINATORIAL OPTIMIZATION

Lecture 5

# Search Landscape Analysis

Marco Chiarandini

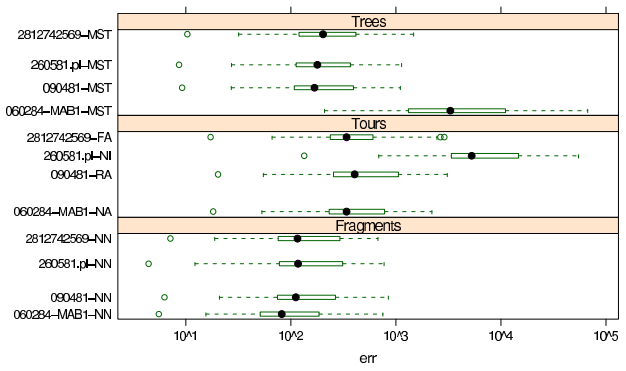
## Outline

1. Competition

2. Fundamental Search Space Properties
   - Fitness-Distance Correlation
   - Ruggedness
   - Plateaux
   - Barriers and Basins

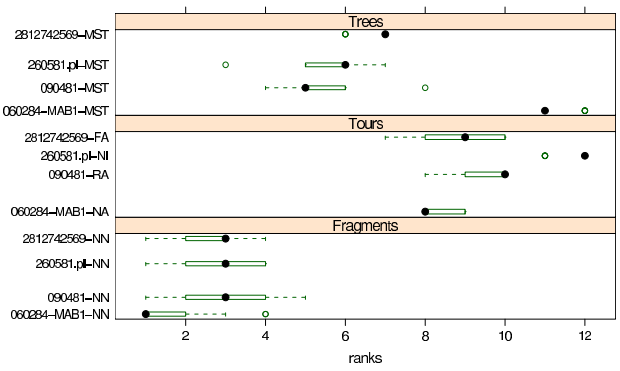


## Results - Boxplots of Errors






## Results - Boxplots of Ranks





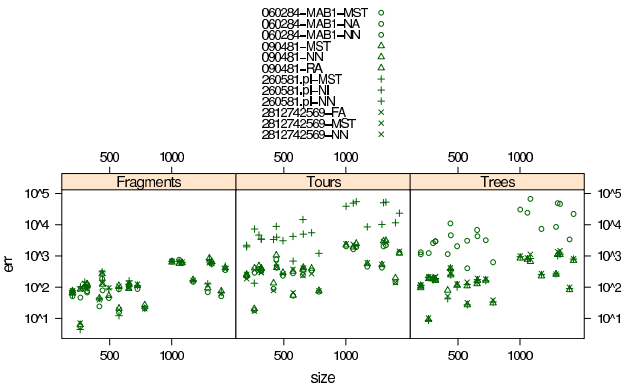

## Numerical Results

| Heuristic | best | median error |
|---|---|---|
| 060284-MAB1-MST | 209.24 | 3298.475 |
| 060284-MAB1-NA | 18.21 | 338.945 |
| 060284-MAB1-NN | 5.53 | 81.915 |
| 090481-MST | 9.22 | 167.860 |
| 090481-NN | 6.26 | 111.385 |
| 090481-RA | 20.26 | 405.155 |
| 260581.pl-MST | 8.62 | 178.680 |
| 260581.pl-NI | 134.02 | 5270.635 |
| 260581.pl-NN | 4.42 | 117.210 |
| 2812742569-FA | 17.21 | 339.465 |
| 2812742569-MST | 10.34 | 202.600 |
| 2812742569-NN | 7.13 | 115.595 |

In the literature: the NN has an error of 26%

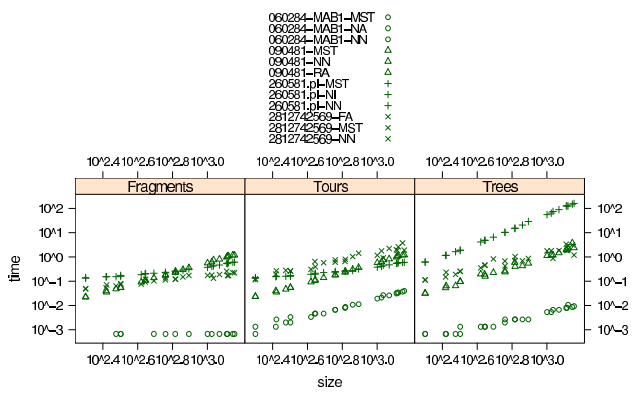


## Results - Scatter Plots: size vs error






## Results - Scatter Plots: size vs time






## Learning Goals

- Understand purpose and goals of search space analysis.
- Get an overview of basic concepts, approaches, and techniques
- Understand relationships between search space features and LS performance.

## Concepts that Define the Search Landscape

- Search space $S$
- Neighborhood structure $N \subseteq S \times S$
  Often defined through an operator collection $\Delta_N$
- Evaluation Function $g(\pi) : S \mapsto \mathbb{R}$

Definition:
The **Search Landscape** $L(\pi)$ of $\pi$ is the triple

$$\mathcal{L}(\pi) := \langle S(\pi), N(\pi), g(\pi) \rangle$$



## Fundamental Search Space Properties

The behavior and performance of an SLS algorithm on a given problem instance crucially depends on properties of the respective search space.

Simple properties of search space $S$:

- search space size $|S|$
- search space diameter $diam(G_N)$
  (= maximal distance between any two candidate solutions)
  **Note:** it depends on the neighborhood size $|N|$
- number of (optimal) solutions $|S'|$, *solution density* $|S'|/|S|$
- distribution of solutions within the neighborhood graph

Solution densities and distributions can generally be determined by:

- exhaustive enumeration;
- sampling methods;
- counting algorithms (often variants of complete algorithms).

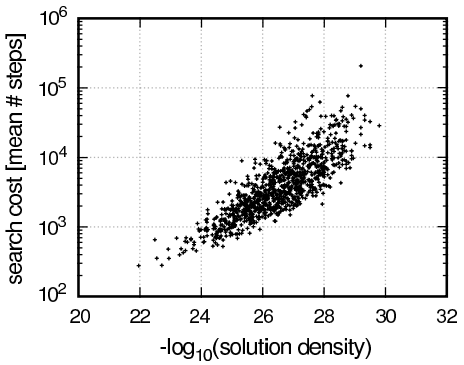


## **Example:** Search space size and diameter for the TSP

- **Given:** Symmetric TSP instance with $n$ vertices
- Candidate solutions = permutations of vertices
- Search space size $= (n-1)!/2$
- Size of 2-exchange neighborhood
  $= \binom{n}{2} = n \cdot (n-1)/2$
- Size of 3-exchange neighborhood
  $= \binom{n}{3} = n \cdot (n-1) \cdot (n-2)/6$
- Diameter of neighborhood graphs: Exact values unknown.
  - Bounds for 2-exchange neighborhood $= [n/2, n-1]$
  - Bounds for 3-exchange neighborhood $= [n/3, n-1]$



## **Example:** Correlation between solution density and search cost for GWSAT over set of hard Random-3-SAT instances:






**A landscape** $L := (S, N, g)$ **is ...**

- *invertible* (or *non-degenerate*), iff
  $\forall s, s' \in S : [g(s) = g(s') \Longrightarrow s = s']$;
- *locally invertible*, iff
  $\forall r \in S : \forall s, s' \in N(r) \cup \{r\} : [g(s) = g(s') \Longrightarrow s = s']$;
  *Note:* Every invertible landscape is also locally invertible
  (but not necessarily vice versa).
- *non-neutral*, iff
  $\forall s \in S : \forall s' \in N(s) : [g(s) = g(s') \Longrightarrow s = s']$.
  *Note:* Every locally invertible landscape is also non-neutral
  (but not necessarily vice versa).



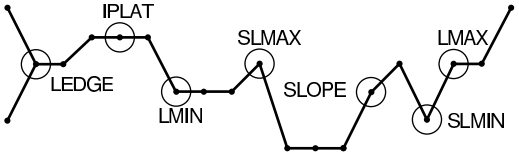
## Classification of search positions

| *position type* | $>$ | $=$ | $<$ |
|---|---|---|---|
| SLMIN (strict local min) | + | – | – |
| LMIN (local min) | + | + | – |
| IPLAT (interior plateau) | – | + | – |
| SLOPE | + | – | + |
| LEDGE | + | + | + |
| LMAX (local max) | – | + | + |
| SLMAX (strict local max) | – | – | + |

"+" = present, "–" absent; table entries refer to neighbors with larger (">") , equal ("="), and smaller ("<") evaluation function values



## Example for various types of search positions:






## **Example:** Complete distribution of position types for hard Random-3-SAT instances

| instance | *avg sc* | SLMIN | LMIN | IPLAT |
|---|---|---|---|---|
| uf20-91/easy | 13.05 | 0% | 0.11% | 0% |
| uf20-91/medium | 83.25 | $< 0.01\%$ | 0.13% | 0% |
| uf20-91/hard | 563.94 | $< 0.01\%$ | 0.16% | 0% |

| instance | SLOPE | LEDGE | LMAX | SLMAX |
|---|---|---|---|---|
| uf20-91/easy | 0.59% | 99.27% | 0.04% | $< 0.01\%$ |
| uf20-91/medium | 0.31% | 99.40% | 0.06% | $< 0.01\%$ |
| uf20-91/hard | 0.56% | 99.23% | 0.05% | $< 0.01\%$ |

(based on exhaustive enumeration of search space;
*sc* refers to search cost for GWSAT)

**Example:** Sampled distribution of position types for hard Random-3-SAT instances

| instance | *avg sc* | SLMIN | LMIN | IPLAT |
|---|---|---|---|---|
| uf50-218/medium | 615.25 | 0% | 47.29% | 0% |
| uf100-430/medium | 3 410.45 | 0% | 43.89% | 0% |
| uf150-645/medium | 10 231.89 | 0% | 41.95% | 0% |

| instance | SLOPE | LEDGE | LMAX | SLMAX |
|---|---|---|---|---|
| uf50-218/medium | $< 0.01\%$ | 52.71% | 0% | 0% |
| uf100-430/medium | 0% | 56.11% | 0% | 0% |
| uf150-645/medium | 0% | 58.05% | 0% | 0% |

(based on sampling along GWSAT trajectories; *sc* refers to search cost for GWSAT)



## Local Minima

**Note:** Local minima impede local search progress.

**Simple properties of local minima:**

- *number of local minima:* $|lmin|$, *local minima density* $|lmin|/|S|$
- *localization of local minima:* distribution of local minima within the neighbourhood graph

**Problem:** Determining these measures typically requires exhaustive enumeration of search space.

⇒ Approximation based on sampling or estimation from other measures (such as autocorrelation measures, see below).



**Example:** Distribution of local minima for the TSP

**Goal:** Empirical analysis of distribution of local minima for Euclidean TSP instances.

**Experimental approach:**

- Sample sets of local optima of three TSPLIB instances using multiple independent runs of two TSP algorithms (3-opt, ILS).
- Measure pairwise distances between local minima (using *bond distance* = number of edges in which two given tours differ).
- Sample set of purportedly globally optimal tours using multiple independent runs of high-performance TSP algorithm.
- Measure minimal pairwise distances between local minima and respective closest optimal tour (using bond distance).



Empirical results:

| Instance | avg $sq$ [%] | avg $d_{lmin}$ | avg $d_{opt}$ |
|---|---|---|---|
| *Results for 3-opt* | | | |
| rat783 | 3.45 | 197.8 | 185.9 |
| pr1002 | 3.58 | 242.0 | 208.6 |
| pcb1173 | 4.81 | 274.6 | 246.0 |
| *Results for ILS algorithm* | | | |
| rat783 | 0.92 | 142.2 | 123.1 |
| pr1002 | 0.85 | 177.2 | 143.2 |
| pcb1173 | 1.05 | 177.4 | 151.8 |

(based on local minima collected from 1 000/200 runs of 3-opt/ILS)



Interpretation:

- Average distance between local minima is small compared to maximal possible bond distance, $n$.
  ⇒ *Local minima are concentrated in a relatively small region of the search space.*
- Average distance between local minima is slightly larger than distance to closest global optimum.
  ⇒ *Optimal solutions are located centrally in region of high local minima density.*
- Higher-quality local minima found by ILS tend to be closer to each other and the closest global optima compared to those determined by 3-opt.
  ⇒ *Higher-quality local minima tend to be concentrated in smaller regions of the search space.*

Note: These results are fairly typical for many types of TSP instances and instances of other combinatorial problems.
In many cases, local optima tend to be clustered; this is reflected in multi-modal distributions of pairwise distances between local minima.



## Fitness-Distance Correlation (FDC)

**Idea:** Analyse correlation between solution quality (fitness) $g$ of candidate solutions and distance $d$ to (closest) optimal solution.

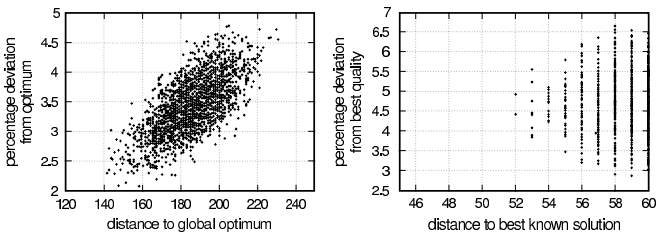
**Measure for FDC**: *empirical correlation coefficient* $r_{fdc}$.

*Fitness-distance plots*, *i.e.*, scatter plots of the $(g_i, d_i)$ pairs underlying an estimate of $r_{fdc}$, are often useful to graphically illustrate fitness distance correlations.

- The FDC coefficient, $r_{fdc}$ depends on the given neighbourhood relation.
- $r_{fdc}$ is calculated based on a sample of $m$ candidate solutions (typically: set of local optima found over multiple runs of an iterative improvement algorithm).



**Example:** FDC plot for TSPLIB instance rat783, based on 2500 local optima obtained from a 3-opt algorithm






High FDC ($r_{fdc}$ close to one):

- 'Big valley' structure of landscape provides guidance for local search;
- search initialisation: high-quality candidate solutions provide good starting points;
- search diversification: (weak) perturbation is better than restart;
- typical, *e.g.*, for TSP.

Low FDC ($r_{fdc}$ close to zero):

- global structure of landscape does not provide guidance for local search;
- typical for very hard combinatorial problems, such as certain types of QAP (Quadratic Assignment Problem) instances.

Applications of fitness-distance analysis:

- algorithm design: use of strong intensification (including initialisation) and relatively weak diversification mechanisms;
- comparison of effectiveness of neighbourhood relations;
- analysis of problem and problem instance difficulty.

Limitations and short-comings:

- *a posteriori* method, requires set of (optimal) solutions, **but:** results often generalise to larger instance classes;
- optimal solutions are often not known, using best known solutions can lead to erroneous results;
- can give misleading results when used as the sole basis for assessing problem or instance difficulty.



## Ruggedness

**Idea:** Rugged search landscapes, *i.e.*, landscapes with high variability in evaluation function value between neighbouring search positions, are hard to search.

Example: Smooth *vs* rugged search landscape

**Note:** Landscape ruggedness is closely related to local minima density: rugged landscapes tend to have many local minima.



The ruggedness of a landscape $L$ can be measured by means of the *empirical autocorrelation function* $r(i)$:

$$r(i) := \frac{1/(m-i) \cdot \sum_{k=1}^{m-i}(g_k - \bar{g}) \cdot (g_{k+i} - \bar{g})}{1/m \cdot \sum_{k=1}^{m}(g_k - \bar{g})^2}$$

where $g_1, \ldots g_m$ are evaluation function values sampled along an uninformed random walk in $L$.

Note: $r(i)$ depends on the given neighbourhood relation.

- Empirical autocorrelation analysis is computationally cheap compared to, *e.g.*, fitness-distance analysis.
- (Bounds on) AC can be theoretically derived in many cases, *e.g.*, the TSP with the 2-exchange neighbourhood.
- There are other measures of ruggedness, such as *empirical autocorrelation coefficient* and *(empirical) correlation length*.



High AC (close to one):

- "smooth" landscape;
- evaluation function values for neighbouring candidate solutions are close on average;
- low local minima density;
- problem typically relatively easy for local search.

Low AC (close to zero):

- very rugged landscape;
- evaluation function values for neighbouring candidate solutions are almost uncorrelated;
- high local minima density;
- problem typically relatively hard for local search.



Note:

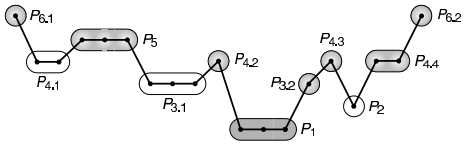
- Measures of ruggedness, such as AC, are often insufficient for distinguishing between the hardness of individual problem instances;
- but they can be useful for
  - analysing differences between neighbourhood relations for a given problem,
  - studying the impact of parameter settings of a given SLS algorithm on its behaviour,
  - classifying the diffculty of combinatorial problems.



## Plateaux

*Plateaux*, *i.e.*, 'flat' regions in the search landscape

**Intuition:** Plateaux can impede search progress due to lack of guidance by the evaluation function.






Definitions

- **Region:** connected set of search positions.
- **Border of region** $R$**:** set of search positions with at least one direct neighbour outside of $R$ (**border positions**).
- **Plateau region:** region in which **all** positions have the same level, *i.e.*, evaluation function value, $l$.
- **Plateau:** maximally extended plateau region, *i.e.*, plateau region in which no border position has any direct neighbours at the plateau level $l$.
- **Solution plateau:** Plateau that consists entirely of solutions of the given problem instance.
- **Exit of plateau region** $R$**:** direct neighbour $s$ of a border position of $R$ with lower level than plateau level $l$.
- **Open / closed plateau:** plateau with / without exits.



Measures of plateau structure:

- *plateau diameter* = diameter of corresponding subgraph of $G_N$
- *plateau width* = maximal distance of any plateau position to the respective closest border position
- *number of exits*, *exit density*
- *distribution of exits within a plateau*, *exit distance distribution* (in particular: avg./max. distance to closest exit)

Some plateau structure results for SAT:

- Plateaux typically don't have an interior, *i.e.*, almost every position is on the border.
- The diameter of plateaux, particularly at higher levels, is comparable to the diameter of search space. (In particular: plateaux tend to span large parts of the search space, but are quite well connected internally.)
- For open plateaux, exits tend to be clustered, but the average exit distance is typically relatively small.



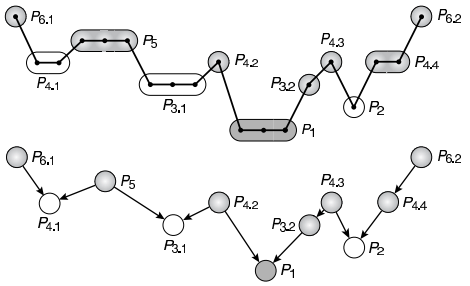
**Idea:** Obtain abstract view of neutral landscape by collapsing positions on the same plateau into 'macro positions'.

Plateau connection graphs (PCGs):

- *Vertices:* plateaux of given landscape
- *Edges (directed):* connect plateaux that are directly connected by one or more exit.
- Additionally, *edge weights* can be used to indicate the relative numbers of exits from one plateau to its PCG neighbours.



**Example:** Simple landscape $L$ and plateau connection graph $PCG$...




*Note:* The plateaux form a partition of $L$, *i.e.* every position in $L$ is part of exactly one (possibly degenerate) plateau.



## Barriers and Basins

Observation:

The *difficulty of escaping* from closed plateaux or strict local minima is related to the *height of the barrier*, *i.e.*, the difference in evaluation function, that needs to be overcome in order to reach better search positions:

Higher barriers are typically more difficult to overcome (this holds, *e.g.*, for Probabilistic Iterative Improvement or Simulated Annealing).

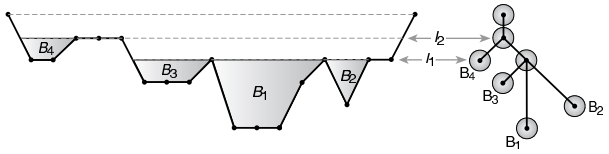


Definitions:

- Positions $s, s'$ are *mutually accessible at level* $l$ iff there is a path connecting $s'$ and $s$ in the neighbourhood graph that visits only positions $t$ with $g(t) \leq l$.
- The *barrier level between positions* $s, s'$, $bl(s, s')$ is the lowest level $l$ at which $s'$ and $s'$ are mutually accessible; the difference between the level of $s$ and $bl(s, s')$ is called the *barrier height between* $s$ *and* $s'$.
- **Basins**, *i.e.*, maximal (connected) regions of search positions below a given level, form an important basis for characterising search space structure.



**Example:** Basins in a simple search landscape and corresponding basin tree




*Note:* The basin tree only represents basins just below the critical levels at which neighbouring basins are joined (by a *saddle*).



Note:

- Like plateau connection graphs, basin trees can provide much deeper insights into SLS behaviour and problem hardness than global measures of search space structure, such as FDC or ACC.
- **But:** This type of analysis is computationally expensive, since it requires enumeration (or sampling) of large parts of the search space.



## Outline

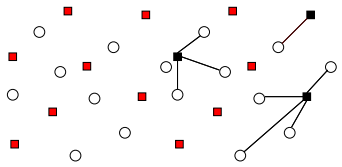
3. Example Problems
   The p-median Problem

## Example Problems: So far...

- Traveling Salesman Problem (TSP)
- Vertex Coloring Problem (GCP)
- Propositional Satisfiability (SAT and MAX-SAT)
- Constraint Satisfaction Problem (CSP and MAX-CSP)
- The Single Machine Total Weighted Tardiness Problem (SMTWTP)



## The p-median Problem

- *Given:*
  a set $U$ of locations for $n$ users
  a set $F$ of locations of $m$ facilities
  a distance matrix $\mathbf{D} = [d_{ij}] \in Re^{n \times m}$
- *Task:* Select $p$ locations of $F$ where to install facilities such that the sum of the distances of each user to its closest installed facility is minimized, *i.e.*,

$$\min_{J} \sum_{i \in U} \min_{j \in F} d_{ij} \qquad J \subseteq F \text{ and } |J| = p$$