STOCHASTIC LOCAL SEARCH
FOUNDATIONS AND APPLICATIONS

Search Space Structure and SLS Performance

Holger H. Hoos & Thomas Stützle

Outline

1. Fundamental Search Space Properties
2. Search Landscapes and Local Minima
3. Fitness-Distance Correlation
4. ~~Ruggedness~~
5. Plateaux
6. Barriers and Basins

Fundamental Search Space Properties

The behaviour 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: The diameter of a given search space depends on the *neighbourhood size*.

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 neighbourhood
  $= \binom{n}{2} = n \cdot (n-1)/2$
- Size of 3-exchange neighbourhood
  $= \binom{n}{3} = n \cdot (n-1) \cdot (n-2)/6$
- Diameter of neighbourhood graphs: Exact values unknown.
  - Bounds for 2-exchange neighourhood $= [n/2, n-1]$
  - Bounds for 3-exchange neighourhood $= [n/3, n-1]$

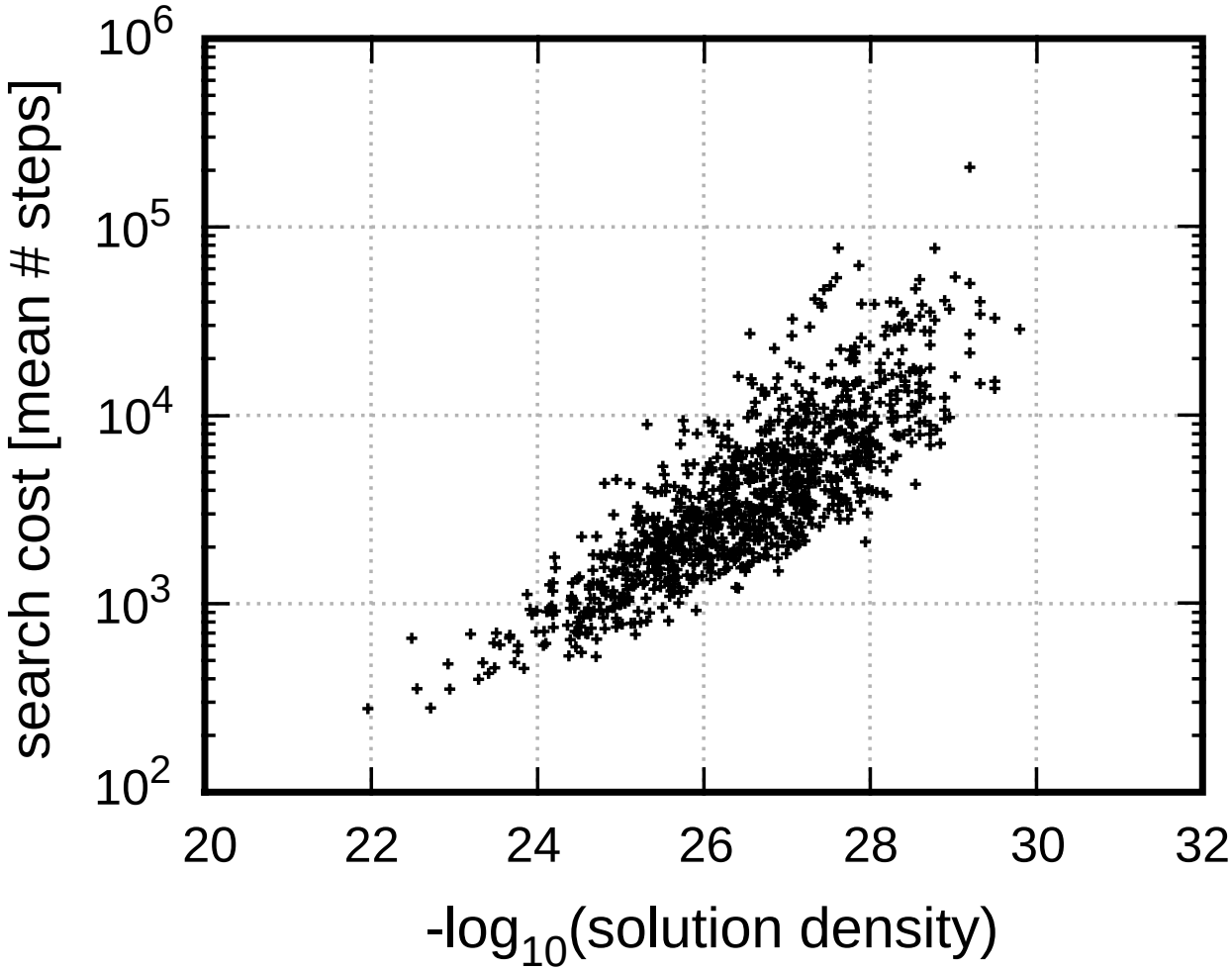
Simple properties of search space S (continued):

- number of (optimal) solutions $\#S'$, *solution density* $\#S'/\#S$
- distribution of solutions within the neighbourhood graph

Note:

- Solution densities and distributions can generally be determined by:
  - exhaustive enumeration;
  - sampling methods;
  - counting algorithms (often variants of complete algorithms).
- In many cases, (optimal) solutions tend to be clustered; this is reflected in uneven distributions of pairwise distances between solutions.

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



Search Landscapes

The behaviour of all but the simplest SLS algorithms depends on an *evaluation function* that guides the search process.

Definition:

Given an SLS algorithm A and a problem instance π with associated

- search space $S(\pi)$,
- neighbourhood relation $N(\pi)$,
- evaluation function $g(\pi) : S \mapsto \mathbb{R}$

the *search landscape of* π, $L(\pi)$, is defined as $L(\pi) := (S(\pi), N(\pi), g(\pi))$.

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).

Local Minima

Note: Local minima impede local search progress.

Simple measures related to local minima:

- number of local minima $\#lmin$, *local minima density* $\#lmin/\#S$
- distribution of local minima within the neighbourhood graph

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

Solution: 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.

  $\Rightarrow$ *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.

  $\Rightarrow$ *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.

  $\Rightarrow$ *Higher-quality local minima tend to be concentrated in smaller regions of the search space.*

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*

$$r_{fdc} := \frac{\widehat{Cov}(g,d)}{\widehat{\sigma}(g) \cdot \widehat{\sigma}(d)},$$

where

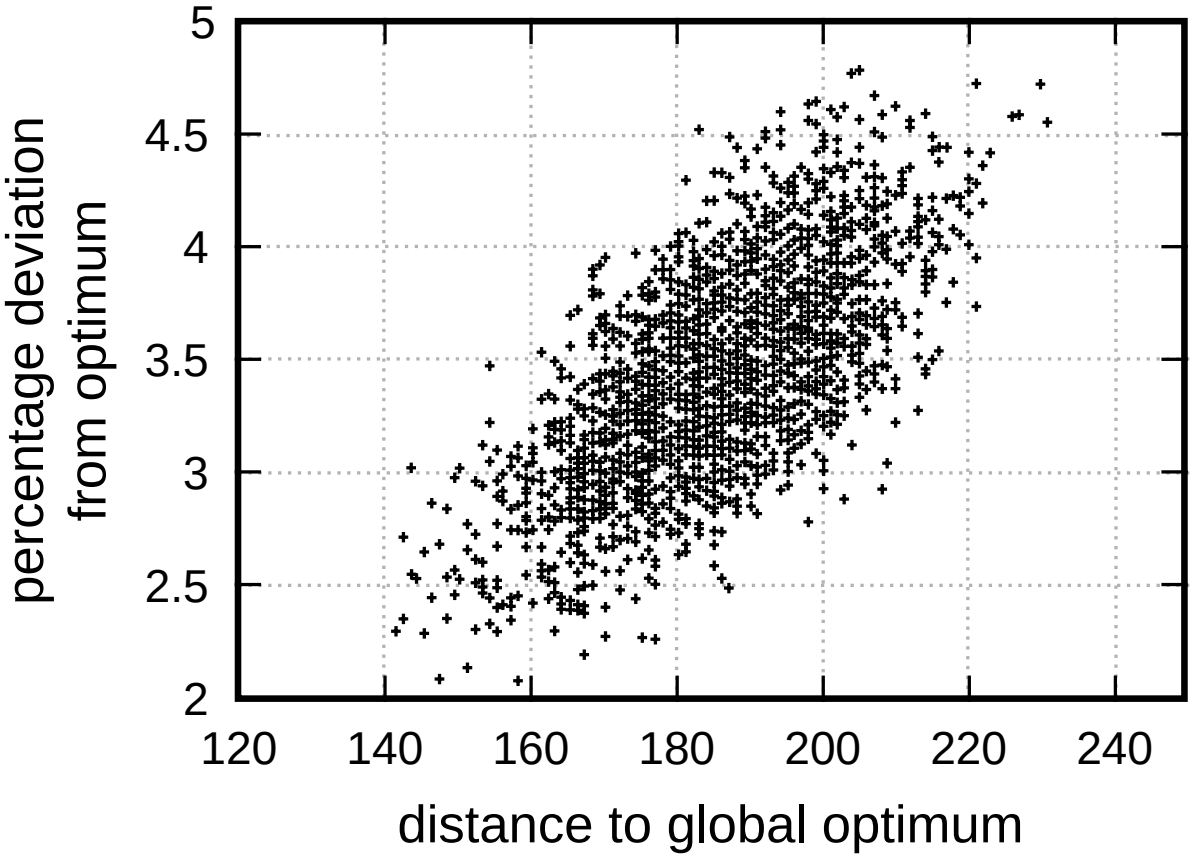
$$\widehat{Cov}(g,d) := \frac{1}{m-1} \sum_{i=1}^{m} (g_i - \bar{g})(d_i - \bar{d}),$$

$$\widehat{\sigma}(g) := \sqrt{\frac{1}{m-1} \sum_{i=1}^{m} (g_i - \bar{g})^2}, \quad \widehat{\sigma}(d) := \sqrt{\frac{1}{m-1} \sum_{i=1}^{m} (d_i - \bar{d})^2}$$

Note:

- 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).
- *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.

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.

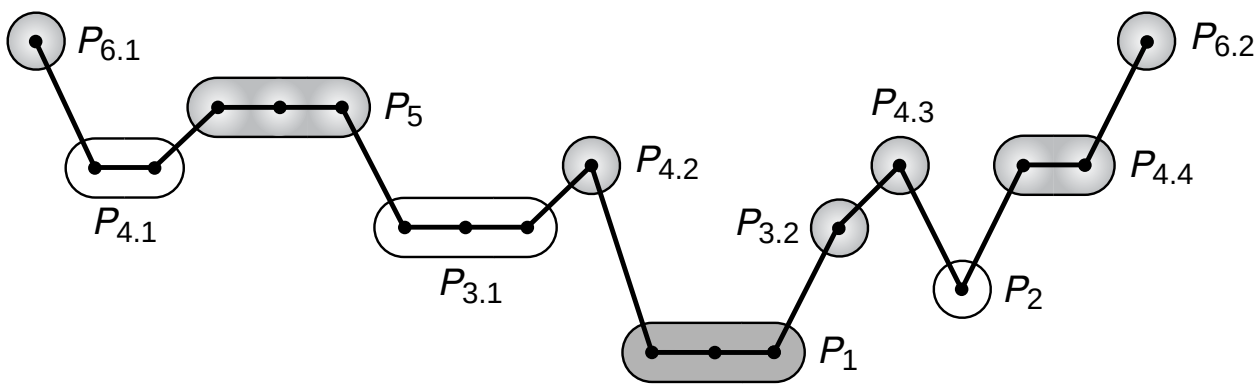
Limitations and short-comings:

- *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.

Plateaux

Plateaux, *i.e.*, 'flat' regions in the search landscape, are characteristic for the neutral landscapes obtained for combinatorial problems such as SAT.

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



Definition

- *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.

Definition

- *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
- *plateau branching factor* = fraction of neighbours of a plateau position that are also on the plateau.
- *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:

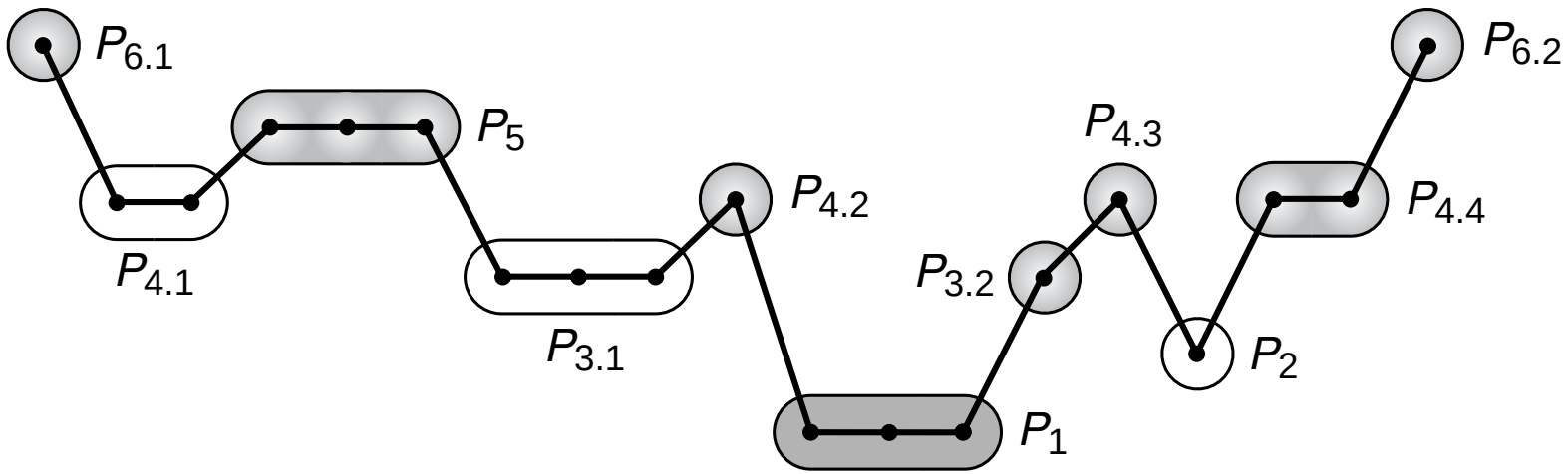
- Plateaux typically don't have an interior, *i.e.*, almost every position is on the border.
- The diameter of plateaux, particularly at higher levels, is comparable to the diameter of search space. (In particular: plateaux tend to span large parts of the search space, but are quite well connected internally.)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

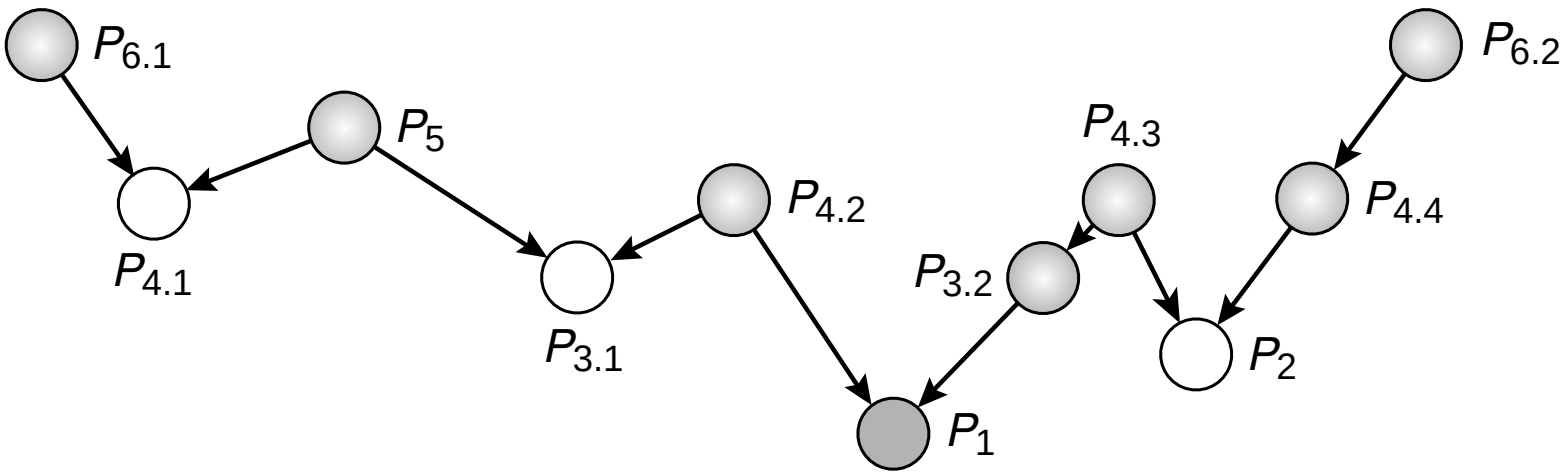
- *Vertices:* plateaux of given landscape
- *Edges (directed):* connect plateaux that are directly connected by one or more exit.
- Additionally, *edge weights* can be used to indicate the relative numbers of exits from one plateau to its PCG neighbours.

Example: Simple neutral search landscape L ...

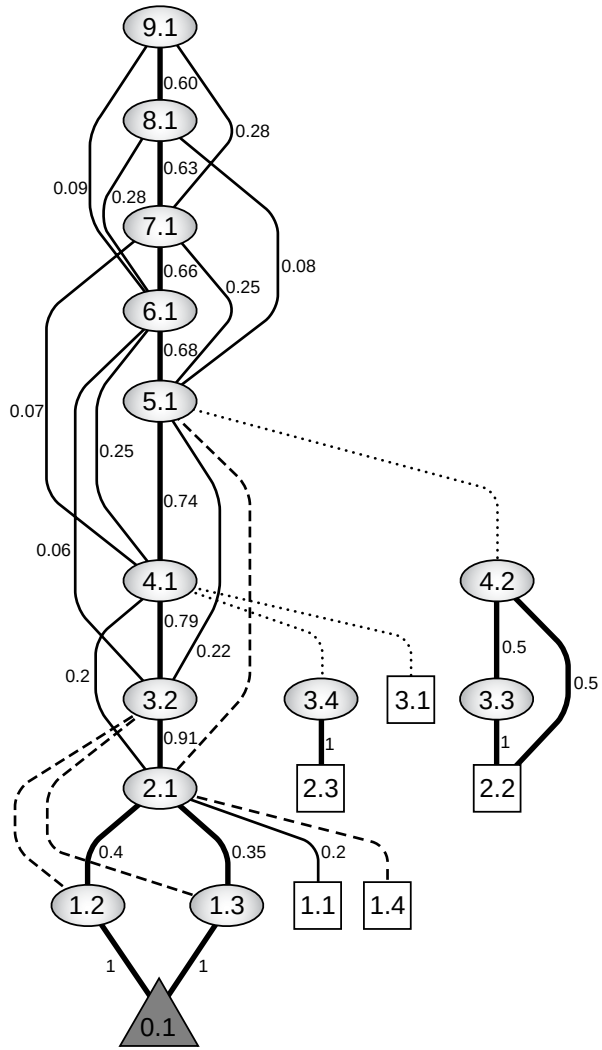


Note: The plateaux form a partition of L, *i.e.* every position in L is part of exactly one (possibly degenerate) plateau.

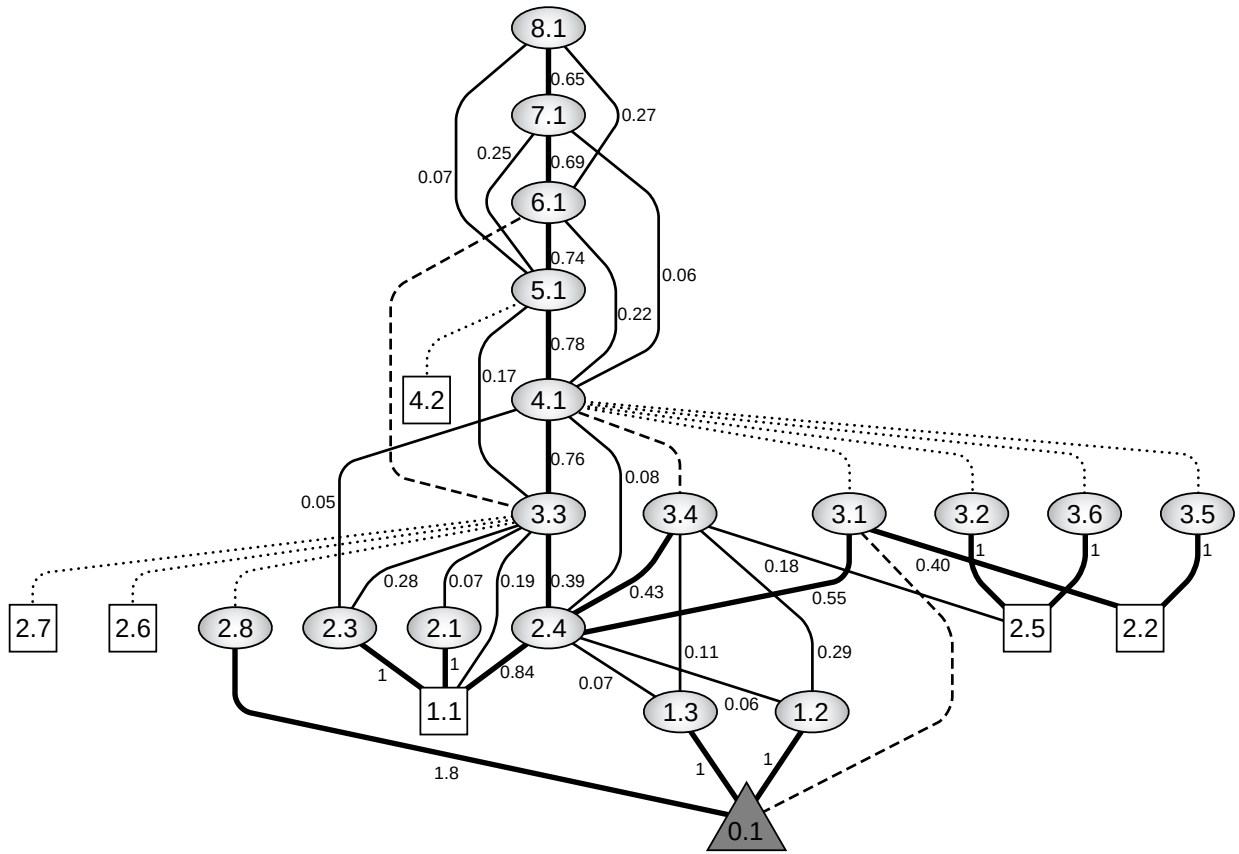
Example: . . . and the respective plateau connection graph



Example: PCG of *easy* Random 3-SAT instance



Example: PCG of *hard* Random 3-SAT instance



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).

Definition:

- 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'*.

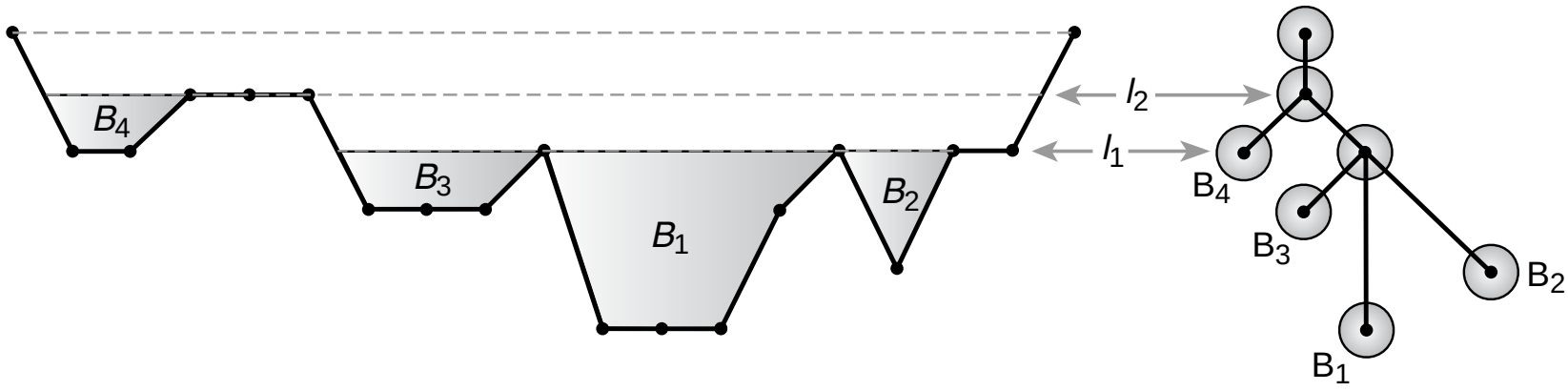
- The *depth of a position s* is the minimal barrier height between s and any position s' at a level lower than s, *i.e.*, for which $g(s') < g(s)$.

Basins, *i.e.*, maximal (connected) regions of search positions below a given level, form an important basis for characterising search space structure.

Note:

- Basins of a given landscape form a *hierarchy*, *i.e.*, two basins are either disjoint, or one is contained in the other.
- Basin hierarchies can be formally represented as *basin trees*.

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.