PA184 - Heuristic Search Methods

Lecture 7 – Evaluating Heuristic Performance

- Fitness Landscapes
- Empirical Analysis of Fitness Landscapes
- Parameter Tuning and Performance Analysis
- Experiments with Heuristic Search

Learning outcomes:

- Understand the concept of fitness landscape.
- Understand the relationship between features of fitness landscapes and heuristic search.
- Appreciate the purpose and good principles of experimental design for evaluating the performance of heuristic methods.

Fitness Landscapes

The idea of fitness landscape originated in theoretical biology and within the context of optimization, fitness landscapes have been studied mostly for evolutionary algorithms although there are also studies for other meta-heuristics.

Stadler (2002) states about fitness landscapes:

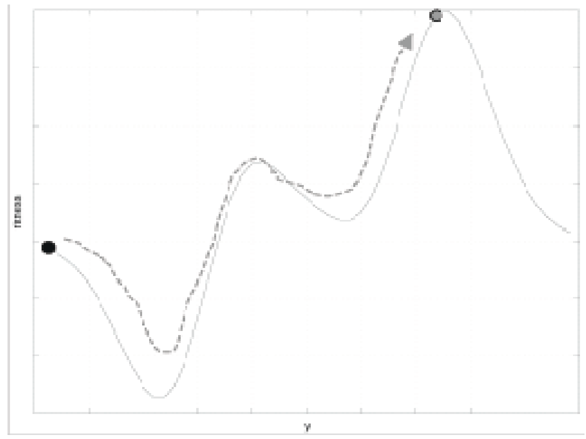
Implicit in this idea is both a fitness function f that assigns a fitness value to every possible genotype (or organism), and the arrangement of the set of genotypes in some kind of abstract space that describes how easily or frequently one genotype is reached from another one.

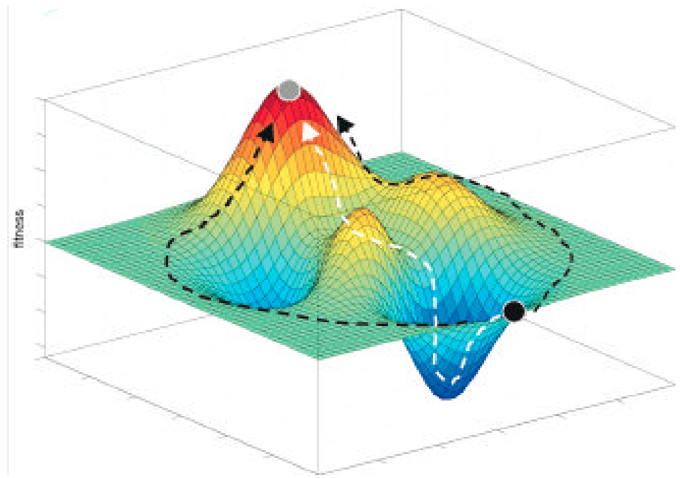
Peter F. Stadler. Fitness Landscapes. In: M. Lässig and A. Valleriani (eds.): iological Evolution and Statistical Physics. Springer-Verlag, Berlin, 2002, pp. 187-207.

It is often easier to 'visualise' the fitness landscape for continuous optimization problems than for combinatorial optimization problems.

Use of Fitness Landscapes

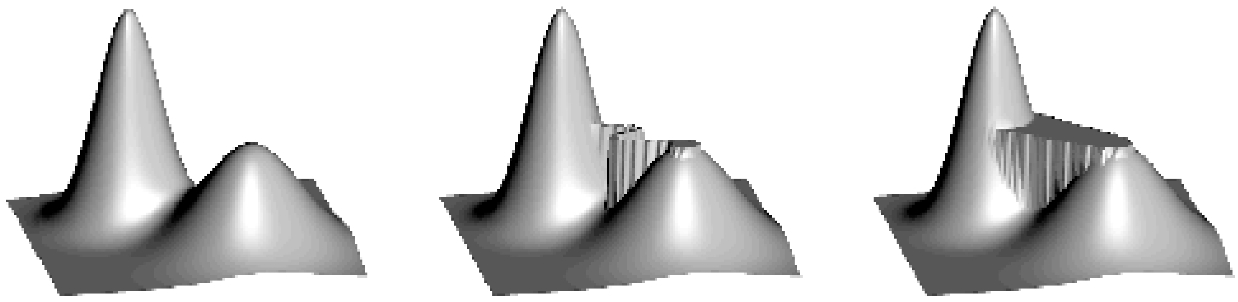
- Describe dynamics of adaptation in Nature (Wright, 1932). Later, describe dynamics of meta-heuristics
- Search: adaptive-walk over a Landscape
- Three components $L = (S, N, f)$
  - Search Space
  - Neighborhood relation or distance metric (operator dependant!)
  - Fitness function





Features of Fitness Landscapes Relevant to Heuristic Search

- Number, fitness, and distribution of local optima or peaks.
- Fitness differences between neighboring points (ruggedness).
- Topology of the basins of attraction of the local optima (lengths of adaptive walks to optima)
- Presence and structure of *neutral networks o plateaus* (terrains with equal fitness)



(a) Unconnected peaks (b) Single neutral pathway (c) Broad neutral plateau

Ruggedness of Fitness Landscape

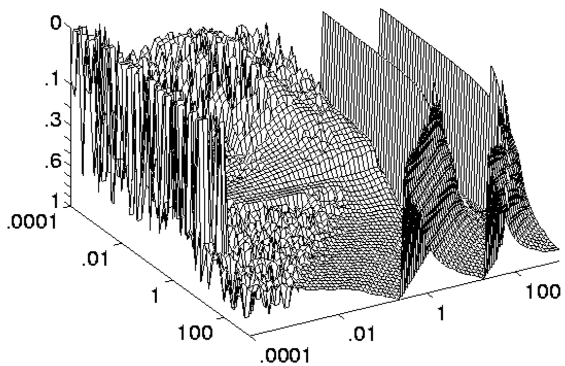
This concept of ruggedness gives a notion of the difficulty for finding the optima in a given landscape.

The ruggedness (shape) of the fitness landscape is induced by:

- Representation
- Fitness function
- Search process (operators, strategy, etc.)

The following features of fitness landscape are of interest:

- Peaks and Valleys
- Basins of attraction
- Peaks concentration
- Local and global optima
- Dynamic behaviour, etc.



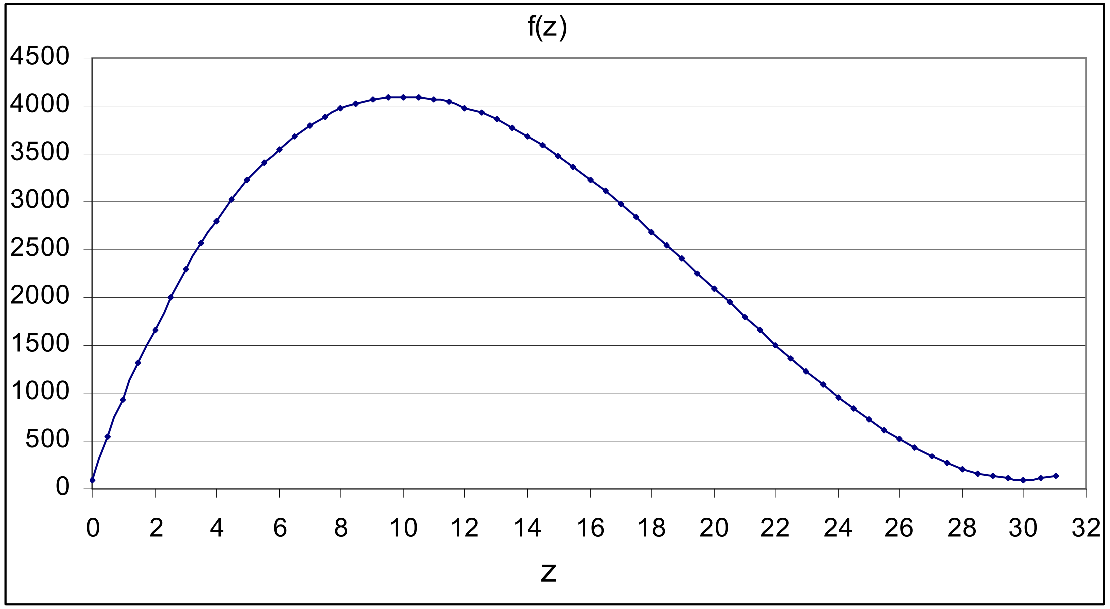
There are some 'static' features and some 'dynamic' behaviour induced by the fitness function and the search process.

Example 7.1 Consider the following simple maximisation problem:

$$f(z) = z^3 - 60z^2 + 900z + 100$$

where z is an integer within $[0,31]$

0	100
1	941
2	1668
3	2287
4	2804
5	3225
6	3556
7	3803
8	3972
9	4069
10	4100
11	4071
12	3988
13	3857
14	3684
15	3475
16	3236
17	2973
18	2692
19	2399
20	2100
21	1801
22	1508
23	1227
24	964
25	725
26	516
27	343
28	212
29	129
30	100
31	131



Clearly, the is a single global optimum at z = 10.

Directly encoding z as an integer and using the add/subtract 1 operator with iterative improvement, induces one optimum.

Example 7.1 (cont.)

However, encoding z as a binary string of length 5, using the 1 bit flip operator with steepest iterative improvement, induces four optima, three of them local optima.

This can be shown by 'exploring' the neighbourhoods for the following sample of solutions (configurations):

00000 00100 10111 10000

Example 7.1 (cont.)

Using first iterative improvement instead of steepest iterative improvement and considering a forward search direction (exploring bit flips left to right), induces the same optima but different 'basins of attraction'.

This can be shown by 'exploring' the neighbourhoods for the following sample of solutions (configurations):

00000 00100 10111 10000

Example 7.1 (cont.)

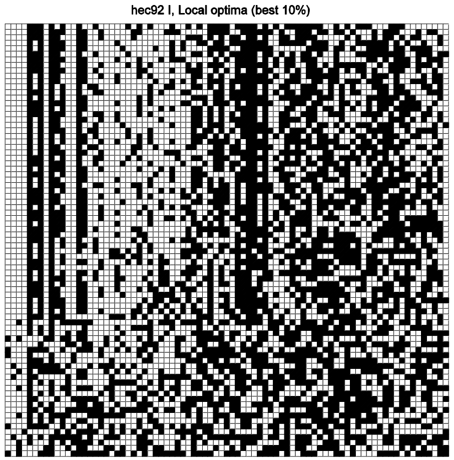
Following the previous case but considering a backward search direction (exploring bit flips right to left), induces again the same optima but different 'basins of attraction'.

This can be shown by 'exploring' the neighbourhoods for the following sample of solutions (configurations):

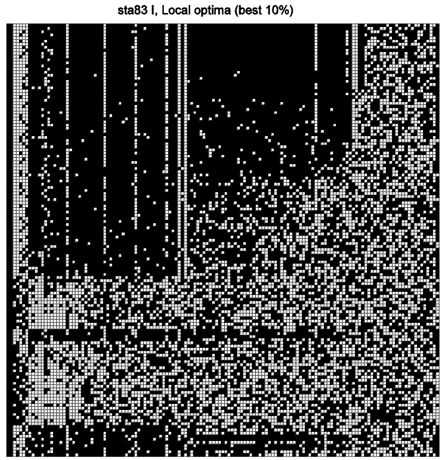
00000 00100 10111 10000

Global Features: Visualising local optima

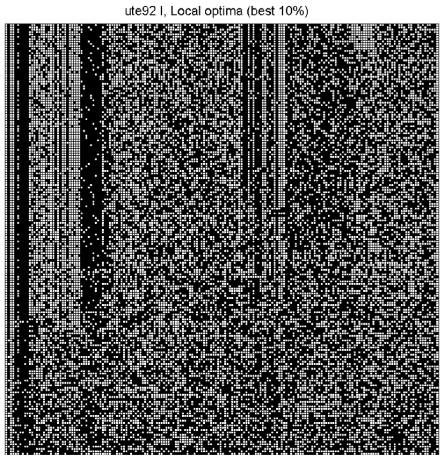
hec92
n= 81



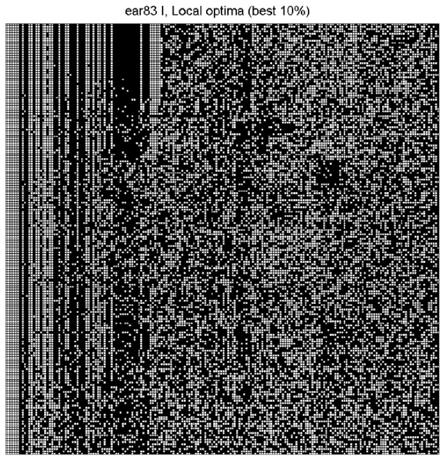
sta83
n = 139



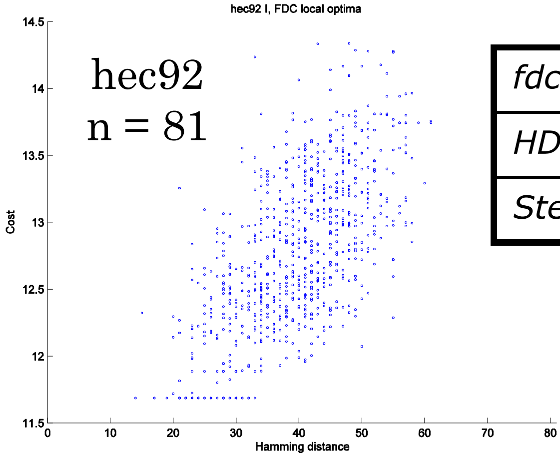
ute92
n = 184



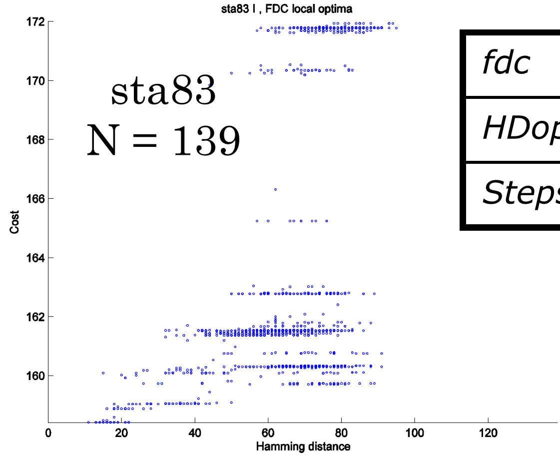
ear83
n = 190



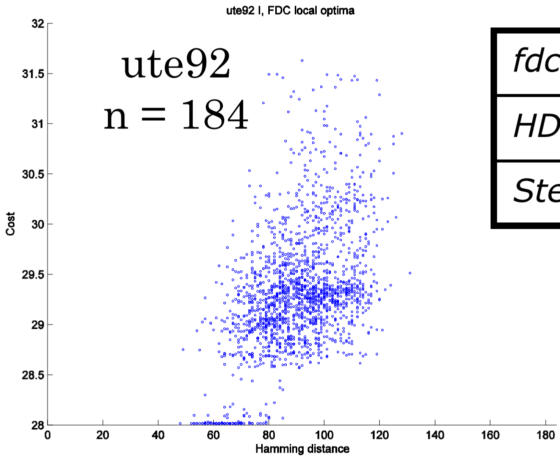
Global Features: Cost-distance scatter plots (local optima)



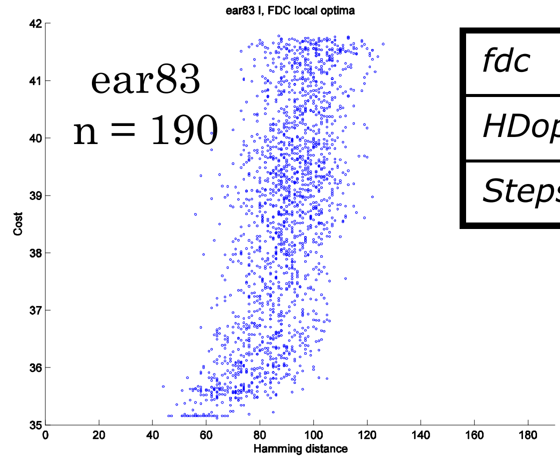
fdc	0.64
HDopt	39.66
Steps	2.45



fdc	0.51
HDopt	64.97
Steps	4.07

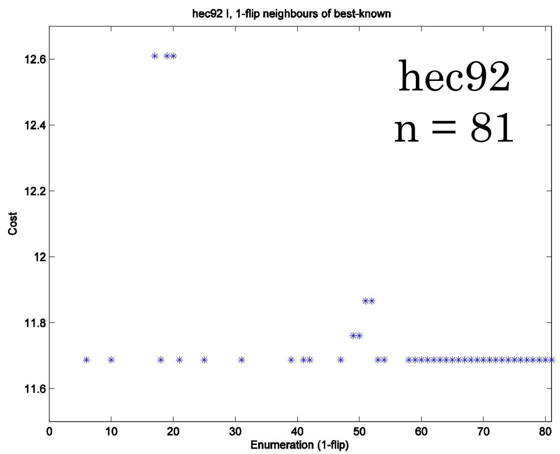


fdc	0.51
HDopt	90.37
Steps	11.81

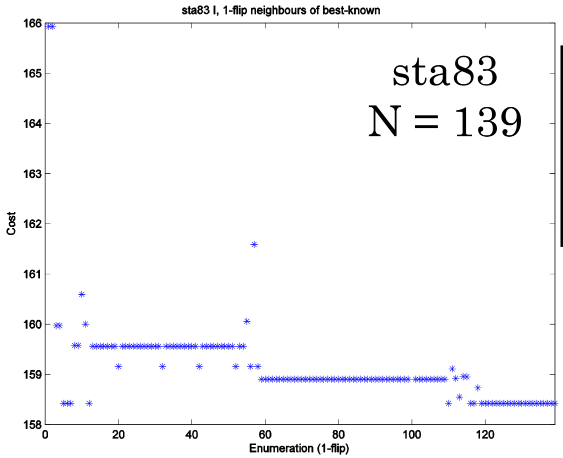


fdc	0.63
HDopt	88.61
Steps	8.29

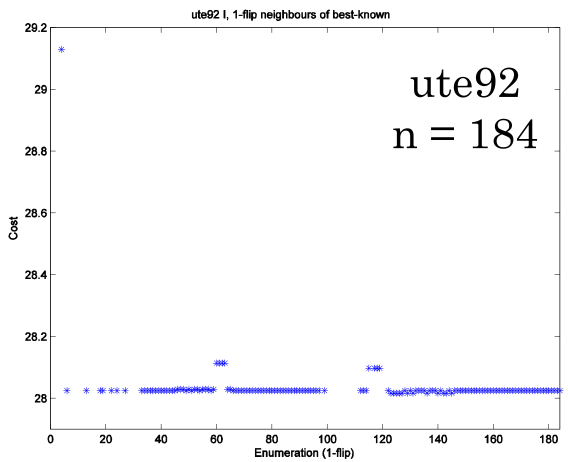
Local Features: 1-flip neighbourhood of best-known



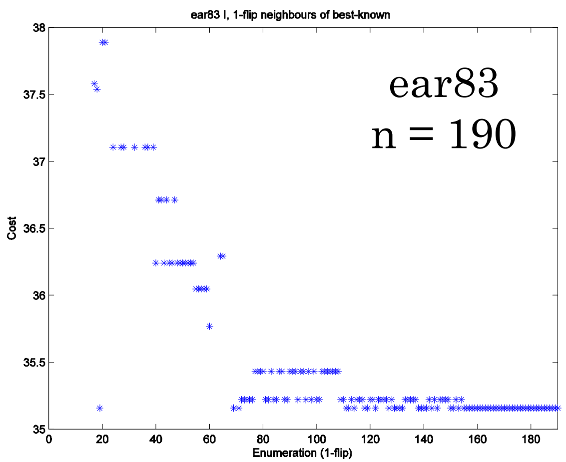
n/l	6.06
pbf	0.44
pbu	0.47



n/l	2.79
pbf	0.20
pbu	0.01



n/l	3.55
pbf	0.07
pbu	0.22



n/l	8.30
pbf	0.31
pbu	0.168

Empirical Analysis of Fitness Landscapes

Given a neighbourhood $N_w(X)$ that applies operator w to the set of solutions X and produces a set of solutions Y, then the canonical distance d_w can be defined as follows:

$$Y \in N_w(X) \Leftrightarrow dw(X,Y) = 1$$

Then, for a given fitness landscape, a solution x^o is local optima if:

$$f(x^o) > f(s) \quad \forall s \in N(x^o)$$

If X^o is the set of local optima and X^* is the set of global optima, a particular local optima $x^* \in X^o$ is a global optimum if:

$$f(x^*) \geq f(x^0) \quad \forall x^0 \in X^o$$

A basin of attraction can be defined as a function: $\mu : X \rightarrow X^o$

where if $\mu(x)$ is the optimum reached by given initial solution x, then the basin of attraction (also dependent of the search strategy) of x^o is:

$$B(x^o) = \{x : \mu(x) = x^o\}$$

Autocorrelation

Simple statistic measure that gives an idea of the type of fitness landscape by collecting data about fitness during a random walk of length T.

$$r_j = \frac{\sum_{t=1}^{T-j} \left(f_t - \overline{f}\right)\left(f_{t+j} - \overline{f}\right)}{\sum_{t=1}^{T} \left(f_t - \overline{f}\right)^2}$$

Typically:

$r_j \approx 1$ for 'smooth' landscapes
$r_j \approx 0$ for 'rugged' landscapes
$r_j < 0$ are possible but rare

Number of Optima

The number of local optima found can be used to estimate the difficulty of finding the global optima of a landscape.

Suppose that given a number r of initial solutions, k different final local optima solutions are found ($r \geq k$) then:

- If r exceeds k substantially, it can be estimated that all local optima have been found.
- If r is not much larger than k, this perhaps indicates that not many local optima have been seen. Still, this can be useful to compare different neighbourhoods.
- Other non-parametric estimates of the total number of optima v can be used. See: (Reeves and Eremeev, 2004).

In general, it is believed that:

- local optima are closer between them and to the global optima, than random sampled solutions would be.
- the better the local optimum, the larger its basin of attraction.

Parameter Tuning and Performance Analysis

Most meta-heuristics require parameter tuning which can be done either offline or online and for different problem instances.

A typical approach for parameter tuning is full factorial design where different values for each parameter are considered and then experiments are carried out with each combination of parameter values.

Offline parameter tuning is computationally expensive but online parameter tuning requires more design skills.

To evaluate the performance of meta-heuristics:

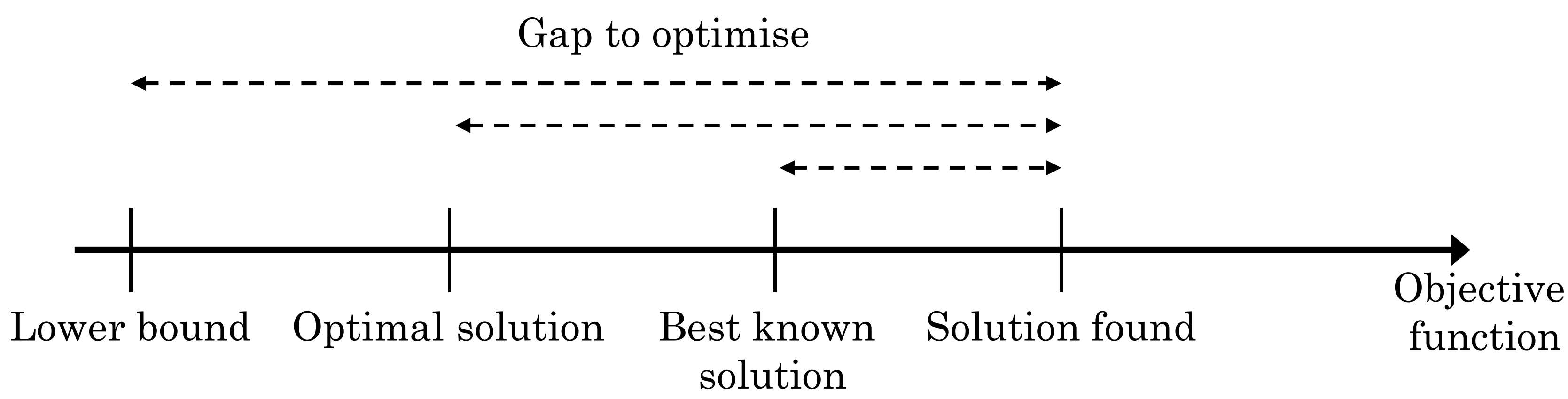
1. Experimental design
2. Measurement
3. Reporting

Measurement involves comparing the quality of the obtained solution against benchmark values. The following diagram taken from (Talbi,2009) illustrates this for a minimisation problem.

Lower bounds can be obtained with relaxation techniques and it is good to have tight lower bounds.

Robustness is sometimes also measured when analysing the performance of meta-heuristics.

Visualisation tools and statistical analysis should be considered when reporting on the performance of meta-heuristics.



Experiments with Heuristic Search

The performance of a heuristic search method should be evaluated scientifically and in an objective manner.

Within this context, an experiment is a set of tests executed under controlled conditions to examine the performance of the heuristic.

According to Barr et al. (1995), experimentation involves the following steps:

1. Define goals of the experiment
2. Create measures of performance and factors to explore
3. Design and execute the experiment
4. Analyse the data and draw conclusions
5. Report and discuss the experiment results

A research experiment must have a clear purpose (question to answer).

Good Attributes of a Proposed Heuristic Method

- Fast – wrt other methods
- Accurate – finds higher-quality solutions
- Robust – less sensitive to tuning and problems
- Simple – wrt implementation
- High-impact – for ‘new’ problems or ‘better’ that other methods
- General – for any problems
- Innovative – original design

Computational experiments are undertaken to:

- Compare the performance of different algorithms on the same problems
- Characterize or describe the behaviour of a method in isolation

Target Questions When Evaluating Heuristics

Typically, heuristic performance is evaluated with respect to three aspects: solution quality, computational effort and robustness.

- Quality of the best solution found?
- How long takes to find the best solution?
- How long it takes to find 'good' solutions?
- How robust is the method?
- How far is the best solution from other 'good' solutions easily found?
- Trade-off between feasibility and solution quality?

Solution Quality. Compare to the optimal solution (if possible), to a tight (upper or lower) bound or to best known solutions.

Computational Effort. Several aspects can be measured, for example: time to find best solution, total execution time, time per stage (multi-stage method), convergence rate.

Robustness. Components and parameter values of the method should remain constant (or automatically set) for different problems.

When designing a computational experiment, it should be decided:

- Which factors to study?
- Which factors to fix?
- Which factors to ignore?

A good experimental design:

- Is unbiased
- Achieves the goals
- Clearly demonstrates performance of the method tested
- Has justifiable rationale
- Generates solid evidence to support conclusions
- Is reproducible

Real problems should be used when possible.

If artificially generated problems are used, these should be archived or its generation should be reproducible.

Once the experimental results are obtained, whenever possible statistical tools should be used to analyse the obtained data and then interpret results in order to collect evidence to reach conclusions.

Barr et al. (1995) propose the following guidelines:

- Ensure reproducibility
- Specify all experimental factors
- Report timing precisely
- Report parameter settings precisely
- Use statistical techniques
- Compare to other methods
- Reduce variability of results
- Produce comprehensive report of results

Of course, experimental evaluation is an alternative to theoretical analysis of an algorithm's performance and behaviour.

Additional Reading

R. Barr, B. Golden, J. Kelly, M. Resende, W. Stewart. *Design and reporting on computational experiments with heuristic methods*. Journal of heuristics, 1, 9-32, 1995.

J. Hooker. *Testing heuristics: we have it all wrong*. Journal of heuristics, 1, 33-42, 1995.

C. McGeoch. *Toward and experimental method for algorithm simulation*. INFORMS journal on computing, 8(1), 1-15 , 1996.

C.R. Reeves, A.V. Eremeev. *Statistical analysis of local search landscapes*. Journal of the Operational Research Society. 55, 687-693, 2004.

A. Tuson, P. Ross. *Adapting operator settings in genetic algorithms*. Evolutionary computation Journal, 6(2), 161-184, 1998.