UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial,
Aeroespacial i Audiovisual de Terrassa

Experimental evaluation of heuristics

Jose M Sallan

Introduction to metaheuristics

Outline

Evaluation of heuristics

Two issues evaluating heuristics:

- How fast can solutions be obtained?
- How close do they come to be optimal?

Experimental evaluation: apply procedures to a collection of instances and compare the observed solution quality and computational time.

Evaluation contexts

The way we perform an experimental evaluation may depend on context:

- Research vs development.
- Design, planning or control applications.
- Life cycle of the problem studied.

Research vs development

Experimental evaluation may depend on the goals of the researcher:

- **Research:** discovering new technologies for existing problems or apply existing technology in creative ways to new problems.
- **Development:** evolve the most efficient solution procedure for a specific environment.

Research focuses on heuristics refinement, development on software implementation for an specific context.

Design, planning or control applications

1. **Design:** problems solved infrequently that seek answers that cover a extended period of time (e.g. capacity location).
2. **Control:** problems to be solved frequently that involve decisions in a short horizon (e.g. data transmision in network, vehicle assignment and routing).
3. **Planning:** intermediate problems (e.g. shift scheduling).

Design problems call for application of exact, time-consuming techniques (e.g. branch and bound), control problems for fast techniques (heuristics).

Problem life cycle

Every problem has its own life cycle in the scientific literature:

- **Early stages:** heuristics to find feasible solutions, workable algorithms running on small instances.
- **Late stages:** insights on issues not considered on previous approaches, algorithms that outperform existing methods.

Design of computational experiments

Distinction between generic problem instance as a particular numerical case.

- **Problem:** Travelling salesman problem (TSP): hamiltonian cycle of minimum value.
- **Instance:** solving TSP on a specific distance matrix of size *n*.

A computational experiment should deal with:

- **Problem characteristics:** e.g. size of matrix distance, type of distance considered, spatial arrangement of nodes.
- **Algorithm parameters:** e.g., stopping rules, search neighbourhoods, move selection.

instances vs procedures design

	Procedure 1	Procedure 2	...	Procedure n
Instance 1				
Instance 1				
...				
Instance n				

Procedure definition

Each **procedure** can be a particular implementation of an algorithm with specific values of components or **parameters**.

- Local search algorithms.
  - Neighbourhood and move definitions.
  - Specific parameters: μ of SA, tabu list size of tabu search...
- Genetic algorithms.
  - Population size.
  - Crossover or mutation operators.

If possible, it is convenient to define two levels por each component or parameter

Definition of the set of instances

Instances should cover structural problem **characteristics** of the different particular cases (e.g., problem size, constraints, distance definition).

- **Multiple replicates:** several instances generated using the same characteristics.
- **Multiple runs:** for randomized procedures, several runs must be made with different random number seeds.

Additional considerations

- **Blocking on instances:** differences among procedures are more likely to be detected if the same instances are solved by all.
- All algorithms must be allowed to consume the **same amount of time** (proxy: number of evaluations of objective function).
  - Example: tabu search vs SA.

Additional considerations

A computational experiment can be resource-consuming:

- Three heuristic **parameters** with two levels each lead to $2^3 = 8$ procedures.
- Four **problem characteristics** with two levels each lead to $2^4 = 16$ variations.
- If we define three **replicates** for each variation we have a set of 48 instances.

Running five times each procedure-instance combination (full factorial design) leads to 1920 runs.

- Fractional factorial designs.
- Discarding non-relevant parameters in pilot studies.

Test instances

The experimental evaluation of an heuristic must be done on a **set of instances** that have the size and variety to span all problem characteristics of interest.

Sources of test instances:

- Real world data sets.
- Published and online instance libraries.
- Randomly generated instances.

Real world data sets

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

Real world data sets are not adequate for testing new algorithms:

- Usually these datasets are proprietary and it is problematic to use for research purposes (reproducibility).
- Real world instances may not cover problem characteristics of future implementations.

These data sets are more adequate for development than for research.

Instance libraries

Many problems have sets of instances published on the internet:

- TSPLIB: `http://bit.ly/2rqu6lV`.
- QAPLIB: `http://bit.ly/2qHur2Z`.
- Taillard scheduling instances: `http://bit.ly/2q8I1sh`.

These libraries usually have benchmarks to assess new developments (even optimal solutions).

Instance libraries

Drawbacks of instance libraries:

- Some benchmarks can come from earlier stages of problem life cycle (too small, not representative).
- Researchers may post instances on which their procedures perform particularly well.
- Too much effort may be devoted to develop heuristics that peform well on a given set of instances, instead of heuristics that perform well on any instance.

Randomly generated data sets

Random generation of instances has several relevant advantages:

- Problem characteristics are under control, so a set of instances representative of **problem space** can be generated.
- Instance generation is **reproducible**, so future researchers can know about differences and similarities between instances.
- Randomly-generated instances are **portable**: we only need the code use to generate them.

Randomly generated data sets

When defining a set of instances, we must pay special attention about how are related problem parameters:

- **TSP:** random matrices, vs distance matrices of nodes located at random.
- **KP:** easy vs difficult, interesting instances to solve.

Example:
Pisinger, D. (2005). Where are the hard knapsack problems? *Computers & Operations Research*, 32(9), 2271-2284.

Performance measurement

Usually heuristic performance is assessed with the **heuristic to best ratio** (for minimization problems):

$$\frac{z - z_{best}}{z_{best}}$$

where z is the value of the objective function obtained with a run of an algorithm and z_{best} the benchmark solution.

Sources of benchmarks:

- Best known solution of all runs.
- Benchmark from previous research (online libraries).
- Optimal solution (obtained with exact algorithms).

Performance measurement

Pitfalls:

- Ratio can be adjusted manipulating instances (e.g., adding a constant to all values of a matrix distance).
- The ratio can be problematic for objective functions taking positive and negative values (e.g., tardiness).

Analysis

The value of the objective function can be different for each run due to:

- **Sampling error:** coming by random generation of instance replicates and random decision made by procedures.
- **Parameter values:** some parameters of the heuristic can influence the value of the solution.

The effect of heuristic parameters on the obtained value of the objective function can be assessed using **statistical analysis**.

Analysis

Several statistical techniques can be used to assess parameter influence:

- Analysis of variance (ANOVA).
- Multivariate regression with dummy variables representing the levels of each parameter.

Sometimes can be signficant the **interaction** of parameters

To be considered, results must be:

- **Significant:** the probability that the results comes from random sampling (p-value) is low.
- **Relevant:** the influence of the parameter on the objective value should be large enough to be of practical relevance.

Reporting results

Results must be reported for each combination of procedure and instance. If several runs have been made, variability of objective function must be reported.

Some alternatives:

- Maximum, minimum and average values.
- Mean and standard deviation.
- Confidence intervals (assuming some probability distribution).
- Graphical representations (histogram, boxplot).

Further reading

Rardin, R. L., & Uzsoy, R. (2001). Experimental evaluation of heuristic optimization algorithms: A tutorial. *Journal of Heuristics*, 7(3), 261-304.