PA184 - Heuristic Search Methods

Lecture 3 – Local Search

- Neighbourhood Search
- Iterative Improvement
- Escaping Local Optima

Learning outcomes:

- Understand the principles of neighbourhood search and the key issues to consider when implementing it.
- Analyse and design neighbourhood moves for some COPs
- Understand the main iterative improvement search strategies
- Describe and discuss the main strategies to escape local optima in heuristic search
- Design competent local search strategy for

Neighbourhood Search

In COPs, the neighbourhood of a solution x in the search space S can be defined as:

$$N(x) = \{y \in S \mid y = \Phi(x)\}$$

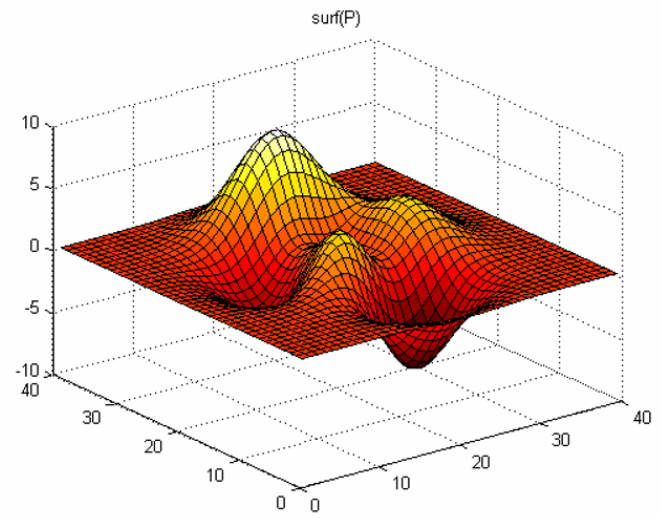
where Φ is a move or sequence of moves

A feasible solution x is a local optimum with respect to N(x) if:

$$f(x) \leq f(y) \quad \forall y \in N(x) \quad \text{(assuming minimisation)}$$

if the inequality above is strict then x is a strict local optimum.

The graph illustrates a fitness landscape with local optima, global optima, hills, valleys and plateaus.



Neighbourhood search refers to exploring solutions within the neighbourhood of the current solution with the goal to find the best solution in the vicinity, i.e. a local optimum.

Trade-off in neighbourhood search: $|N(x)|$ vs. Search Time

Note: defining appropriate neighbourhoods according to the problem domain and the search strategy is crucial in heuristic local search.

Neighbourhood Size

The k-exchange neighbourhood is widely used in many COPs.

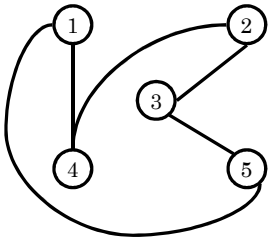
Typically, $|\text{k-exchange neighbourhood}| = O(n^k)$

Neighbourhood pruning: consider only a subset of neighbours which are selected based on the state of the current solution.

Example 3.1 Considering the 2-edge-exchange neighbourhood structure, illustrate neighbour solutions for a symmetric TSP with 5 cities. Start with the following tour: 14235

edges
14
42
23
35
51

tour: 14235

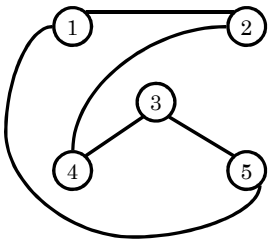


possible exchanges
14,23 to 12,43
14,35 to 13,45
42,35 to 43,25
42,51 to 45,21
23,51 to 25,31

Each of the possible 2-edge-exchanges generates a neighbour solution

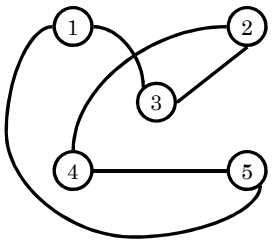
14,23 to 12,43

tour: 12435



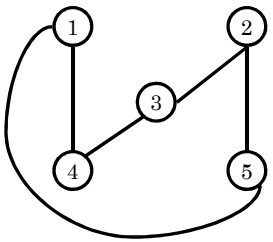
14,35 to 13,45

tour: 13245



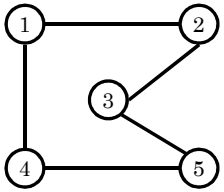
42,35 to 43,25

tour: 14325



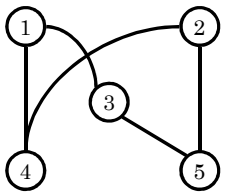
42,51 to 45,21

tour: 12354



23,51 to 25,31

tour: 13524



Based on the 2-edge-exchange move, each tour has 5 neighbour solutions, i.e. $|N(x)| = 5$

Delta Fitness Evaluation

Refers to making only incremental updates to the fitness function by calculating only the effect of the differences between the candidate solution x' and the current solution x.

The goal of delta evaluation is to make local search more efficient particularly in problems with computationally expensive evaluation functions.

Example 3.2 Given solution s for a symmetric TSP with 5 cities and its corresponding fitness value:

$$s = c_1c_3c_4c_2c_5 \quad \text{then} \quad f(s) = d(c_1,c_3)+d(c_3,c_4)+d(c_4,c_2)+d(c_2,c_5)+d(c_5,c_1)$$

after a 2-edge-exchange move which gives solution s':

$$s = c_1c_3c_4c_2c_5 \text{ and } s' = c_1c_4c_2c_5c_3 \text{ then delta fitness evaluation is :}$$

$$f(s') = d(c_1,c_4)+d(c_4,c_2)+d(c_2,c_5)+d(c_5,c_3)+d(c_3,c_1)$$

$$f(s') = f(s)-d(c_3,c_4)-d(c_5,c_1)+d(c_1,c_4)+d(c_5,c_3)$$

Iterative Improvement

General Iterative Improvement

1. Generate initial solution x
2. Select x' ∈ N(x)
3. If f(x') is better than f(x) then x = x'
4. If stop condition true then finish, otherwise go to 2

Factors that influence the overall search strategy:

- Quality of initial solution
  - Random, Greedy, Peckish
- Quality of the selected neighbour solution
  - Random, First improving, Best improving, Random improving, Least improving, Probabilistic improving
- The neighbourhood explored
  - Size, Quality, Single/Multiple
- Stopping condition
  - Time Limit, Idle Iterations, Max Iterations, etc.

Steepest (Greedy) Iterative Improvement

1. Initialise best solution x_{best}
2. Generate initial solution x
3. local_opt = *false*
4. Select the best x' $\in$ N(x)
   a) If f(x') is better than f(x) then x = x'
   b) Else local_opt = *true*
5. If local_opt = false then go to 4
6. If f(x) is better than f(x_{best}) then x_{best} = x
7. If stop condition true then finish, otherwise go to 2

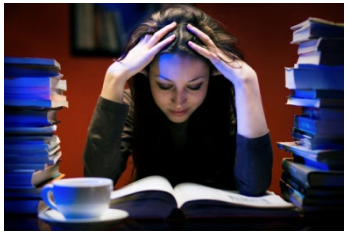
In steepest iterative improvement (also called greedy iterative improvement), the best neighbour solution is selected every time.

Although simple to implement, steepest iterative improvement has several limitations.

Escaping Local Optima

Some Strategies…

- Re-initialise the search from a different starting solution.
- Accept non-improving candidate solutions to replace the current solution in deterministic or probabilistic manner.
- Explore only a fraction of the neighbourhood (e.g. pruning).
- Use more than one neighbourhood to generate candidate solutions.
- Design composite (independent or not) neighbourhoods.
- Maintain a memory of already visited solutions or regions.
- Modify the fitness function to change the search landscape.
- Perturb local optima in deterministic or probabilistic manner.
- Improve more than one solutions simultaneously.

Intensification  vs.  Diversification

ILS – Iterated Local Search

1. Initialise best solution x_{best}
2. Generate initial solution x
3. local_opt = *false*
4. Perform perturbation of x
5. Select the best x' ∈ N(x)
   a) If f(x') is better than f(x) then x = x'
   b) Else local_opt = *true*
6. If local_opt = false then go to 5
7. If f(x) is better than f(x_{best}) then x_{best} = x
8. If stop condition true then finish, otherwise go to 3

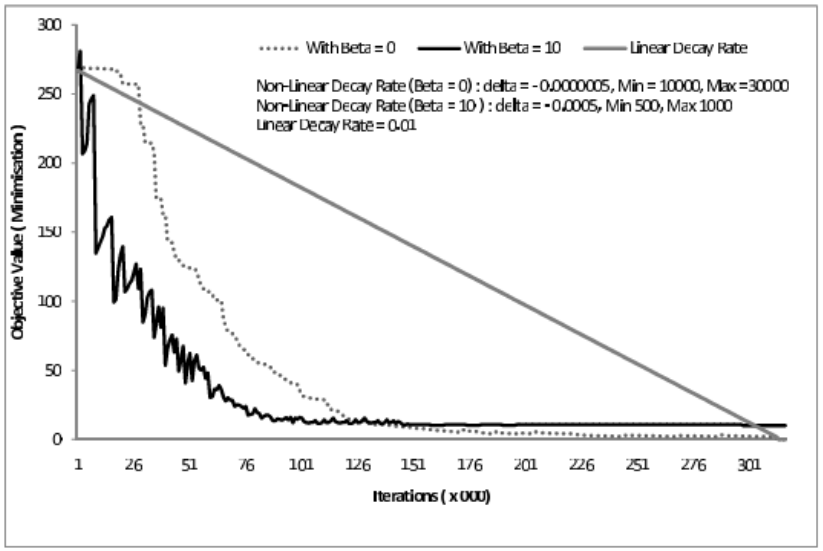
The iterated local search approach is an intuitive method to combine intensification and diversification.

The local search and perturbation operators should be carefully designed so that they work well in combination.

Neighbourhood Search With Learning

A Non-linear Great Deluge (NLGD) algorithm that employs more than one neighbourhood, uses a simple 'learning mechanism' to select the move to apply at each step during the search.

$$B = B \times \left(\exp^{-\delta(rnd[\min,\max])} \right) + \beta$$



Non-linear Great Deluge Search

Static Memory

$$p_i = \frac{w_i}{\sum_{i \in H} w_i}$$

Dynamic Memory

$$p_i = \frac{w_i + w_{\min}}{\sum_{i \in H} w_i + w_{\min}}$$

$$w_{\min} = \min\{0, w_i\} \qquad w_{ih} = \sum_{j=k}^{h} \sigma^i \Re_{ij}$$

$$\Re_{ij} \begin{cases} r & \text{if } \Delta < 0 \\ -r & \text{if } \Delta > 0 \\ \Im & \text{if } \Delta = 0 \text{ and new solution} \\ \Im & \text{if } \Delta = 0 \text{ and no new solution} \\ 0 & \text{if not selected} \end{cases}$$

Additional Reading

Chapters 2 and 5 of (Michalewicz,2004)

Chapter 2 of (Holger and Stuetzle, 2005)

Joe H. Obit, Dario Landa-Silva, Djamilah Ouelhadj, Marc Sevaux. *Non-Linear Great Deluge with Learning Mechanism for Solving the Course Timetabling Problem.* Proceedings of the 8th Metaheuristics International Conference (MIC 2009), Hamburgh Germany, 2009.

E. Aarts, J.K. Lenstra, J. (eds). *Local search in combinatorial optimisation*. W*iley,* 1997.

Seminar Activity 3

The purpose of this seminar activity is to achieve an understanding of basic local search with focus on neighbourhood search.

For the GAP described in Lecture 1 and the neighbourhood moves described in your Seminar Activity 2, do the following:

1. Estimate the size of the neighbourhood, in terms of the problem size, defined by each move.

2. Explain if delta fitness evaluation can be used to evaluate candidate solutions produced with the moves defined and if that is the case, outline a delta fitness evaluation function.

3. Describe some ways in which solutions could be 'perturbed' in order to apply an Iterated Local Search strategy to the problem.