



Heuristics

Jesper Larsen

jesla@man.dtu.dk

Department of Management Engineering
Technical University of Denmark



Why do we need heuristics?

There are several reasons for using a heuristic instead of an optimal method:

1. Most of the practical problems that we wish to solve are $\mathcal{NP}$ -hard. Finding an optimal solution will typically be almost impossible.
2. It is complicated or impossible to formulate the problem as a MIP/IP of reasonable size.
3. It is difficult or impossible for the B&B to come up with (good) feasible solutions in reasonable time.



Type of heuristics

Heuristics are divided into categories:

1. Approximation heuristics
2. Metaheuristics
3. MIP-based heuristics

Heuristics can also be classified into:

1. Construction heuristics
2. Improvement heuristics



Approximation heuristics

Advantages:

- ▶ A priori bound for worst-case behavior.
- ▶ Polynomial running time.

Drawbacks:

- ▶ A priori bound is typically very weak.
- ▶ Approximation algorithms are purpose-build for a given problem.



Metaheuristics

- **Advantages:**
 - ▶ Metaheuristics is a framework that fits many different problems.
 - ▶ In practice it delivers good bounds on most problems. Easy to change.
- **Drawbacks:**
 - ▶ Hardly any theory regarding quality of solution, running time etc.
 - ▶ No a priori bound for worst-case behaviour.
 - ▶ Experiments are needed.

Jesper Larsen

5



Examples from Wolsey

Wolsey mentions four examples of MIP-based heuristics:

- Dive-and-Fix
- Relax-and-Fix
- Cut-and-Fix
- Beam Search

Jesper Larsen

7



MIP-based heuristics

- **Advantages:**
 - ▶ It is a framework that fits many different problems.
 - ▶ A bound is available during computations.
- **Drawbacks:**
 - ▶ No feasible solution may be found.
 - ▶ Running time unknown.
 - ▶ No a priori bound for worst-case behaviour.
 - ▶ Experiments are needed to set up the algorithms.

Jesper Larsen

6



Construction or Improvement

- A construction heuristic starts from scratch and builds a solution step by step.
- Approximation heuristics are often construction heuristics.
- An improvement heuristic tries to improve on an already established solution.
- Metaheuristics are a popular example of improvement heuristics.

Jesper Larsen

8





Example: STSP

For the STSP we have:

- **Nearest neighbour:** Start at some city and then visit the nearest city. Continue to visit the nearest city that has not yet been visited until a tour is complete. (greedy)
- **Cheapest insertion heuristic:** Start with a single node. One by one add nodes whose insertion makes the smallest increase to the length of the tour. (also greedy)

Jesper Larsen

9



The Tree heuristic

1. Determine the MST for the graph (opt: z_T)
2. Double the number the edges in the graph
3. Perform the pre-order walk, but jump directly to the next unvisited node in the pre-order walk

Jesper Larsen

11



Example: STSP II

- **Furthest insertion heuristic:** Start with a single node. Insert the node with the largest minimal distance to the existing tour. Continue until all nodes are inserted. (also greedy)
- **Savings heuristic:** Rank the arcs in ascending order of cost. Add the arcs in this order, so long as they do not violate any constraints, and until all cities have been visited. (again greedy)

Jesper Larsen

10



A graph $G = (V, E)$ is Eulerian if the degree of all nodes are even. In a connected Eulerian graph it is possible to construct a **walk** starting and ending at a node v in which each edge is traversed exactly once.

Proposition 12.3: Now given a complete graph H on V with edge length satisfying the triangle inequality, and let $G = (V, E)$ be a connected Eulerian subgraph of H . Then the original graph contains a STSP tour of length at most $\sum_{e \in E} c_e$.

Jesper Larsen

12





Proof of the bound

We know:

- A tour is a tree and an edge — $z_T \leq z$
- By construction the length of the Eulerian subgraph is $2z_T$
- By construction of the tour: $z_H \leq 2z_T$

So $\frac{z_H}{2} \leq z_T \leq z \leq z_H$ or $\frac{z_H}{z} \leq 2$.



Proof of the bound (Christofides)

- As before we have $z_T \leq z$.
- Now assume the optimal tour of length z is $1, 2, \dots, n$. Let $j_1, j_2, j_3, \dots, j_{2k}$ be the nodes in increasing order in V' .
- Consider $M_1 = \{(j_1, j_2), (j_3, j_4), \dots, (j_{2k-1}, j_{2k})\}$ and $M_2 = \{(j_2, j_3), (j_4, j_5), \dots, (j_{2k}, j_1)\}$ of length z_{M_1} and z_{M_2} .
- The triangle inequality gives:

$$c_{j_1 j_2} + c_{j_2 j_3} + \dots + c_{j_{2k} j_1} \leq \sum_{i=1}^{n-1} c_{i, i+1} + c_{n, 1} = z.$$



Christofides' Heuristic

1. Determine the MST for the graph (opt: z_T)
2. Let V' be the set of nodes of odd cardinality in (V, E_T)
3. Find a perfect matching M in the graph $G' = (V', E')$. (opt: z_M)
4. Perform pre-order walk, but jump directly to the next unvisited node in the pre-order walk



Proof II

- So $z_M \leq z_{M_i}$ for $i = 1, 2$, and therefore $2z_M \leq z_{M_1} + z_{M_2} \leq z$.
- Then by proposition 12.3 we have $z_C \leq z + z/2 = (3/2)z$.



Local search

- Local search are ways to improve a solution. In most cases (all?) no worst-case bound on the heuristic solution is known.
- For a local search algorithm we need to define a **solution**, a **neighbourhood**, and a **goal function**.
- A **neighbourhood** or **neighbourhood structure** is a function $\mathcal{N} : \mathcal{S} \rightarrow 2^{\mathcal{S}}$ that assigns to every $s \in \mathcal{S}$ a set of neighbours $\mathcal{N}(s) \subset \mathcal{S}$.



- Often the solution is a feasible solution to the original problem, the goal function is equal to the objective function and the neighbourhood then defines a “transition” of a feasible solution to another feasible solution. (Examples: TSP, VRP)
- In some situations it is beneficial (necessary) to allow infeasible solutions in your neighbourhood. The goal function will then incorporate a penalty term in order to punish infeasibility. (Examples: Set Partitioning)



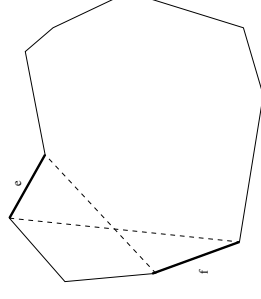
Local search algorithm

- (step 1): In local search one chooses an initial solution s . Then the neighbourhood of the solution is searched for a solution $s' \in \mathcal{N}(s)$ with $f(s') < f(s)$.
- (step 2): In none can be found stop and return s .
- (step 3): If one can be found repeat step 1 with $s = s'$.



Local search for STSP I

- We can construct a new (feasible) solution from the current one by removing 2 arcs and inserting 2 new ones.





Local search for STSP II

- This leads to the 2-exchange heuristic for STSP (also called 2-opt). Size of the neighbourhood is $O(n^2)$.
- The local search heuristic is applied with the following setting:
 - ▶ A set $s \subset E$ is the set of edges that form a tour
 - ▶ $\mathcal{N}(s) = \{s' : s' \neq s, |s \cap s'| = n - 2\}$
 - ▶ $f(S) = \sum_{e \in S} c_e$



Tabu Search

- Use the idea of moving to the best neighbour like in local search (even though it is worse).
- Problem can be cycling:
 $s_0 \rightarrow s_1 \rightarrow s_0 \rightarrow s_1 \rightarrow \dots$
- To avoid cycling certain solutions or moves are forbidden or tabu.
- A list of recent solutions or recent modifications are maintained. This list is denoted the **tabu list**.



Metaheuristics

- Metaheuristics are designed to overcome the problem of escaping from a local optimum.
- Today there exists many different variants of metaheuristics. The classical methods that still remains competitive:
 - ▶ Simulated annealing
 - ▶ Tabu Search
 - ▶ Genetic algorithms



The Tabu Search algorithm

$\mathcal{N}'(s)$ is $\mathcal{N}(s)$ with the tabu solutions removed.
 Choose initial solution s
 $s^* := s$
repeat
 Generate $\mathcal{N}'(s)$
 Choose the best s' in $\mathcal{N}'(s)$
 $s := s'$
if $f(s) < f(s^*)$ **then** $s^* := s$
until stopping criterion is met
return s^*



Elements of the TB

- Possible stopping criterion:
 - ▶ Stop after N iterations
 - ▶ Stop after N seconds
 - ▶ Stop after N iterations without improvement of the best solution.
- Length of tabulist: 7 (experiment!)
- Neighbourhood: Choose a “natural” one. Let it be polynomial in size.

Jesper Larsen

25



The Simulated Annealing algorithm

Choose initial solution s
 Get initial temperature T and reduction factor $0 < r < 1$
repeat
 loop L times
 Pick random solution $s' \in \mathcal{N}(s)$
 $\Delta = f(s') - f(s)$
 if $\Delta \leq 0$ **set** $s := s'$
 if $\Delta > 0$ **set** $s := s'$ with prob. $e^{-\frac{\Delta}{T}}$
 set $T := rT$
until stopping criterion **return** best solution found

Jesper Larsen

27



Simulated Annealing

- In principle it originates back to 1953 where the idea was used to simulate physical systems.
- *The laws of thermodynamics state that at temperature t , the probability of an increase in energy of magnitude δE is given by:*

$$p(\delta E) = \exp\left(-\frac{\delta E}{kt}\right).$$
- Was used for an optimization problem first time in 1983.

Jesper Larsen

26



SA-specific parameters

- Initial temperature T
- Cooling ratio r
- Loop length L
- Define stopping criterion (new one: threshold on T).
- We are not dependent on the size of the neighbourhood.

Jesper Larsen

28





More on metaheuristics

If you want to know more about metaheuristics
Thomas Stidsen is running the course “Optimization
using Metaheuristics” (42133).
The course is running in the Spring in module F2A
(Monday afternoons).