DM811

Heuristics for Combinatorial Optimization

**Lecture 2**

# Introductory Topics

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

# Outline

# Outline

# Construction Heuristics

**Construction heuristics**

(aka, single pass heuristics or dispatching rules in scheduling)
They are closely related to tree search techniques but correspond to a single path from root to leaf

- search space = partial candidate solutions
- search step = extension with one or more solution components

Construction Heuristic (CH):
$s := \emptyset$
**while** $s$ is not a complete candidate solution **do**
  choose a solution component $(X_i = v_j)$
  add the solution component to $s$

# Designing Constr. Heuristics

Which variable should we assign next,
and in what order should its values be tried?

- Select-Unassigned-Variable
  - *Static*: Degree heuristic (reduces the branching factor) also used as tie breaker
  - *Dynamic*: Most constrained variable = Fail-first heuristic = Minimum remaining values heuristic
- Order-Domain-Values
  eg, least-constraining-value heuristic (leaves maximum flexibility for subsequent variable assignments)

# Designing Constr. Heuristics

- Ideas for variable selection
  - with smallest min value
  - with largest min value
  - with smallest max value
  - with largest max value
  - with smallest domain size
  - with largest domain size

  The degree of a variable is defined as the number of constraints it is involved in.

  - with smallest degree. In case of ties, variable with smallest domain.
  - with largest degree. In case of ties, variable with smallest domain.
  - with smallest domain size divided by degree
  - with largest domain size divided by degree

  The min-regret of a variable is the difference between the smallest and second-smallest value still in the domain.

  - with smallest min-regret: $i = \operatorname{argmin} \Delta f_i^{(2)} - \Delta f_i^{(1)}$
  - with largest min-regret: $i = \operatorname{argmax} \Delta f_i^{(2)} - \Delta f_i^{(1)}$
  - with smallest max-regret: $i = \operatorname{argmin} \Delta f_i^{(n)} - \Delta f_i^{(1)}$
  - with largest max-regret: $i = \operatorname{argmax} \Delta f_i^{(n)} - \Delta f_i^{(1)}$

# Designing Constr. Heuristics

- Ideas for value selection
  - Select smallest value
  - Select median value
  - Select maximal value

  Look-ahead:
  - Select value that leaves the largest number of feasible values to the other variables
  - Select value that leaves the smallest number of feasible values to the other variables (fail early)

# Greedy best-first search

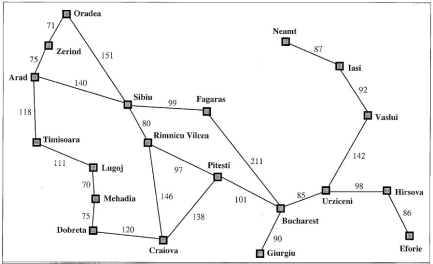


**Figure 3.2** A simplified road map of part of Romania.

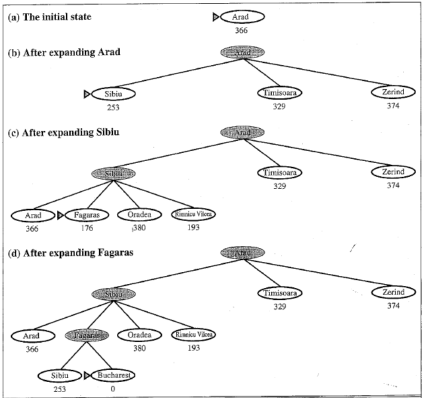


**Figure 4.2** Stages in a greedy best-first search for Bucharest using the straight-line distance heuristic $h_{SLD}$. Nodes are labeled with their $h$-values.

- Sometimes greedy heuristics can be proved to be optimal
    - minimum spanning tree,
    - single source shortest path,
    - total weighted sum completion time in single machine scheduling,
    - single machine maximum lateness scheduling

- Other times an approximation ratio can be proved

# Local Search Paradigm

- search space = complete candidate solutions
- search step = modification of one or more solution components
- neighborhood candidate solutions in the search space reachable in a step
- iteratively generate and evaluate candidate solutions
  - decision problems: evaluation = test if solution
  - optimization problems: evaluation = check objective function value

Iterative Improvement (II):

determine initial candidate solution $s$
**while** $s$ has better neighbors **do**
    choose a neighbor $s'$ of $s$ such that $f(s') < f(s)$
    $s := s'$

# Local Search Algorithm

Basic Components:

- solution representation $\rightsquigarrow$ search space
- initial solution
- neighborhood relation (determines the move operator)
- evaluation function