DM811

Heuristics for Combinatorial Optimization

Lecture 5

Construction Heuristics, TSP and SAT

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

1. Exercises
   Heuristics for TSP
   Heuristics for GCP

Recap

- Combinatorial Optimization and Terminology
- Basic Concepts in Algorithmics
  Graphs • Notation and runtime • Machine model • Pseudo-code • Computational Complexity • Analysis of Algorithms
- Construction Heuristics + Local Search + Metaheuristics
- Software systems and Working Environment
  [Comet, EasyLocal++, unix]
- Assignment 1 + Analysis of Results in R
  [RStudio, Cheat Sheet, My Notes, script]
- Construction Heuristics (search tree, variable + value model)
  [from complete (DFS, Best first, A*) to incomplete search (greedy)]
- Metaheuristics on CH

Outline

1. Exercises
   Heuristics for TSP
   Heuristics for GCP

Construction Heuristics

Construction heuristics specific for TSP

- Heuristics that Grow Fragments
  - Nearest neighborhood heuristics
  - Double-Ended Nearest Neighbor heuristic
  - Multiple Fragment heuristic (aka, greedy heuristic)
- Heuristics that Grow Tours
  - Nearest Addition
  - Farthest Addition
  - Random Addition
  - Nearest Insertion
  - Farthest Insertion
  - Random Insertion
  - Clarke-Wright savings heuristic
- Heuristics based on Trees
  - Minimum spanning tree heuristic
  - Christofides' heuristics
  - Fast recursive partitioning heuristic

Construction Heuristics for TSP

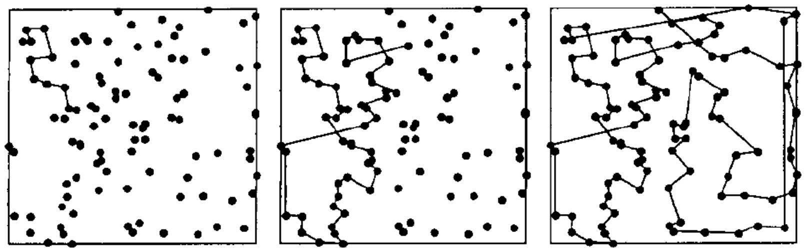


Figure 1. The Nearest Neighbor heuristic.

Construction Heuristics for TSP

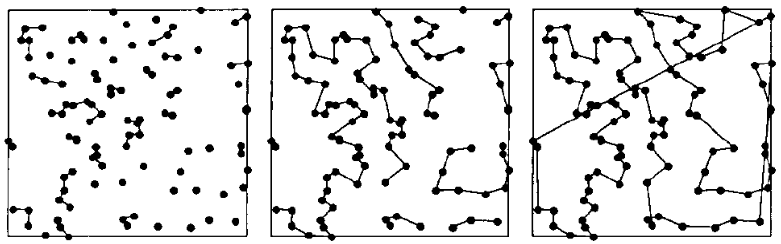


Figure 5. The Multiple Fragment heuristic.

Construction Heuristics for TSP

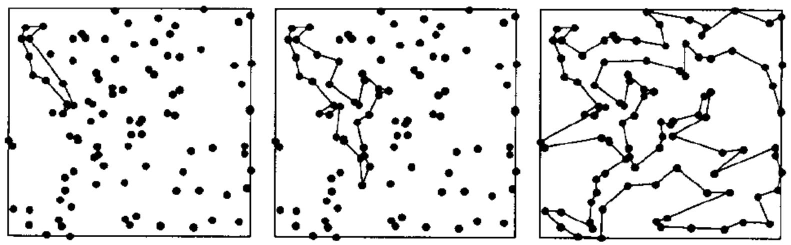


Figure 8. The Nearest Addition heuristic.

Construction Heuristics for TSP

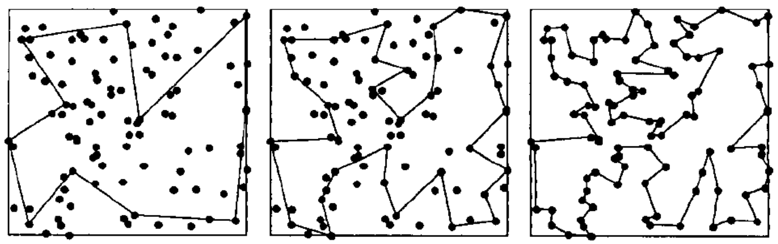


Figure 11. The Farthest Addition heuristic.

Construction Heuristics for TSP

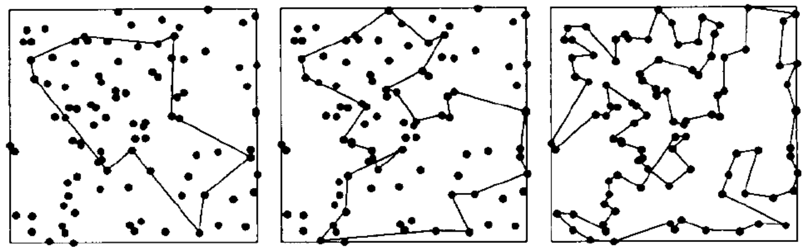


Figure 14. The Random Addition heuristic.

Construction Heuristics for TSP

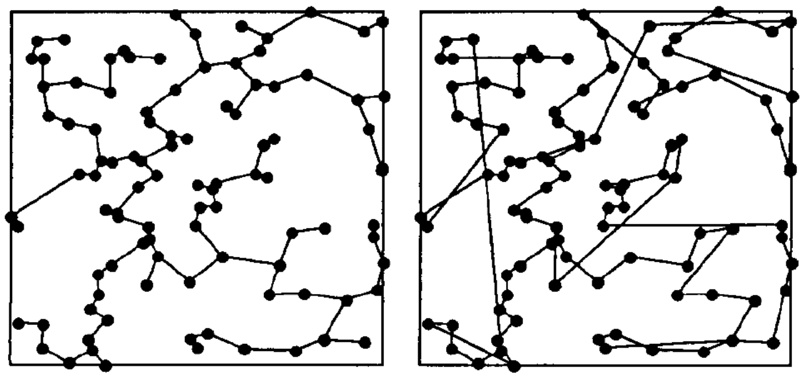


Figure 18. The Minimum Spanning Tree heuristic.

Construction Heuristics for TSP

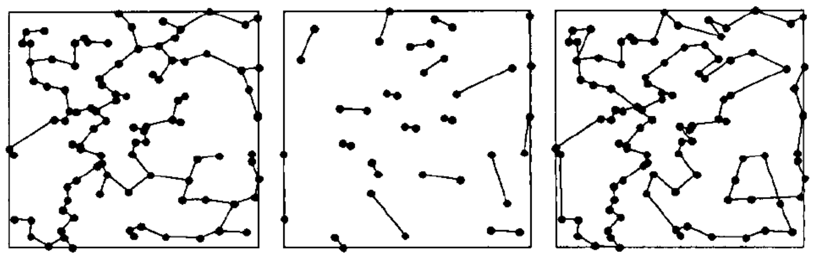


Figure 19. Christofides' heuristic.

Construction Heuristics

- sequential heuristics
  1. choose a variable (vertex)
     a) static order: random (ROS), largest degree first, smallest degree last
     b) dynamic order: saturation degree (DSATUR) [?]
  2. choose a value (color): greedy heuristic

Procedure ROS
RandomPermutation π(Vertices);
forall i in $1, \ldots, n$ **do**
  $v := \pi(i)$;
  select min{c : c not in saturated[v]};
  col[v] := c;
  add c in saturated[w] for all w adjacent v;

$$\mathcal{O}(nk + m) \leadsto \mathcal{O}(n^2)$$

Procedure DSATUR
select vertex v uncolored with max degree;
while uncolored vertices **do**
  select min{c : c not in saturated[v]};
  col[v] := c;
  add c in saturated[w] for all w adjacent v;
  select uncolored v with max size of saturated[v];

$$\mathcal{O}(n(n + k) + m) \leadsto \mathcal{O}(n^2)$$

- partitioning heuristics
  - recursive largest first (RLF) [?]
    iteratively extract stable sets

Alternative form of pseudo-code

Procedure ROS
RandomPermutation π(Vertices);
forall i in $1, \ldots, n$ **do**
  v := π(i);
  selectMin {c : c not in saturated[v]} **do**
    col[v] := c;
    forall w in Vertices: adj[v,w] **do**
      saturated[w].insert(c);

Procedure DSATUR
RandomPermutation π(Vertices);
forall i in $1, \ldots, n$ **do**
  v := π(i);
  selectMin {c : c not in saturated[v]} **do**
    col[v] := c;
    forall w in Vertices: adj[v,w] **do**
      saturated[w].insert(c);

RLF [?]

Procedure Recursive Largest First(G)
In $G = (V, E)$: input graph;
Out k : upper bound on $\chi(G)$;
Out c : a coloring $c : V \mapsto K$ of G;

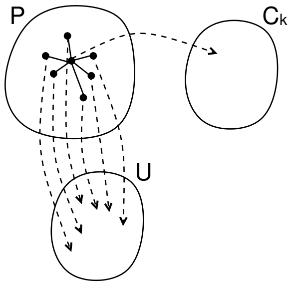
$k \leftarrow 0$ **while** $|V| > 0$ **do**
  $k \leftarrow k + 1$ /* Use an additional color */
  FindStableSet(V, E, k) /* $G = (V, E)$ is reduced */
return k

RLF

Key idea: extract stable sets trying to maximize edges removed.

Procedure FindStableSet(G, k)
In $G = (V, E)$: input graph
In k : color for current stable set
Var P : set of potential vertices for stable set
Var U : set of vertices that cannot go in current stable set

$P \leftarrow V$; $U \leftarrow \emptyset$;
forall $v \in P$ **do** $d_U(v) \leftarrow 0$; /* degree induced by U */
while P not empty **do**
  select v in P with max d_U;
  move v from P to C_k; $V \leftarrow V \setminus \{v\}$
  forall $w \in \delta_P(v)$ **do** /* neighbors of v in P */
    move w from P to U; $E \leftarrow E \setminus \{v, w\}$
    forall $u \in \delta_P(w)$ **do**
      $d_U(u) \leftarrow d_U(u) + 1$



$$\mathcal{O}(m + n\Delta^2) \leadsto \mathcal{O}(n^3)$$