DM811

Heuristics for Combinatorial Optimization

Lecture 9
Examples

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Examples

Iterative Improvement for TSP

TSP-2opt-first(s)
input: *an initial candidate tour* $s \in S(\in)$
output: *a local optimum* $s \in S_\pi$

for $i = 1$ to $n - 1$ **do**
  for $j = i + 1$ to n **do**
    if $P[i] + 1 = P[j]$ or $P[j] + 1 = P[i]$ **then** *continue*
    if $P[i] + 1 \geq n$ or $P[j] + 1 \geq n$ **then** *continue*

$$\begin{aligned} \Delta_{ij} \quad = \quad & d(\pi_{P[i]}, \pi_{P[j]}) + d(\pi_{P[i]+1}, \pi_{P[j]+1}) + \\ & -d(\pi_{P[i]}, \pi_{P[i]+1}) - d(\pi_{P[j]}, \pi_{P[j]+1}) \end{aligned}$$

    if $\Delta_{ij} < 0$ **then**
      $\texttt{UpdateTour}(\texttt{s}, \texttt{i}, \texttt{j})$

is it really?

Examples

Iterative Improvement for TSP

TSP-2opt-first(s)
input: *an initial candidate tour* $s \in S(\in)$
output: *a local optimum* $s \in S_\pi$

Improvement=*TRUE*;
while Improvement is TRUE **do**
  Improvement=*FALSE*;
  for $i = 1$ to $n - 1$ **do**
    for $j = i + 1$ to n **do**
      if $P[i] + 1 = P[j]$ or $P[j] + 1 = P[i]$ **then** *continue*
      if $P[i] + 1 \geq n$ or $P[j] + 1 \geq n$ **then** *continue*

$$\begin{aligned}\Delta_{ij} \quad = \quad & d(\pi_{P[i]}, \pi_{P[j]}) + d(\pi_{P[i]+1}, \pi_{P[j]+1}) + \\ & -d(\pi_{P[i]}, \pi_{P[i]+1}) - d(\pi_{P[j]}, \pi_{P[j]+1})\end{aligned}$$

      if $\Delta_{ij} < 0$ **then**
        $\mathtt{UpdateTour}(\mathtt{s}, \mathtt{i}, \mathtt{j})$
        Improvement=*TRUE*

Summary: Local Search Algorithms
(as in [Hoos, Stützle, 2005])

For given problem instance π:

1. search space S_π
2. neighborhood relation $\mathcal{N}_\pi \subseteq S_\pi \times S_\pi$
3. evaluation function $f_\pi : S \to \mathbf{R}$
4. set of memory states M_π
5. initialization function $\texttt{init} : \emptyset \to S_\pi \times M_\pi)$
6. step function $\texttt{step} : S_\pi \times M_\pi \to S_\pi \times M_\pi$
7. termination predicate $\texttt{terminate} : S_\pi \times M_\pi \to \{\top, \bot\}$

Outline

Examples

- Permutations
  - TSP
  - SMWTP
- Assignments
  - SAT
  - Coloring
  - Parallel machines
- Sets
  - Max Weighted Independent Set
  - Steiner tree

Single Machine Total Weighted Tardiness

Given: a set of n jobs $\{J_1, \dots, J_n\}$ to be processed on a single machine and for each job J_i a processing time p_i, a weight w_i and a due date d_i.

Task: Find a schedule that minimizes
the total weighted tardiness $\sum_{i=1}^{n} w_i \cdot T_i$
where $T_i = \max\{C_i - d_i, 0\}$ (C_i completion time of job J_i)

Example:

Job	J_1	J_2	J_3	J_4	J_5	J_6
Processing Time	3	2	2	3	4	3
Due date	6	13	4	9	7	17
Weight	2	3	1	5	1	2

Sequence $\phi = J_3, J_1, J_5, J_4, J_1, J_6$

Job	J_3	J_1	J_5	J_4	J_2	J_6
C_i	2	5	9	12	14	17
T_i	0	0	2	3	1	0
$w_i \cdot T_i$	0	0	2	15	3	0

The Max Independent Set Problem

Also called “stable set problem” or “vertex packing problem”.
Given: an undirected graph $G(V, E)$ and a non-negative weight function ω on V ($\omega : V \to \mathbf{R}$)

Task: A largest weight independent set of vertices, i.e., a subset $V' \subseteq V$ such that no two vertices in V' are joined by an edge in E.

Related Problems:

Vertex Cover

Given: an undirected graph $G(V, E)$ and a non-negative weight function ω on V ($\omega : V \to \mathbf{R}$)
Task: A smallest weight vertex cover, i.e., a subset $V' \subseteq V$ such that each edge of G has at least one endpoint in V'.

Maximum Clique

Given: an undirected graph $G(V, E)$
Task: A maximum cardinality clique, i.e., a subset $V' \subseteq V$ such that every two vertices in V' are joined by an edge in E

Graph Partitioning

Input: A graph $G = (V, E)$, weights $w(v) \in Z^+$ for each $v \in V$ and $l(e) \in Z^+$ for each $e \in E$.
Task: Find a partition of V into disjoint sets $V_1, V_2, \ldots, V_m$ such that $\sum_{v \in V_i} w(v) \leq K$ for $1 \leq i \leq m$ and such that if $E' \subseteq E$ is the set of edges that have their two endpoints in two different sets V_i, then $\sum_{e \in E'} l(e)$ is minimal.

Consider the specific case of graph bipartitioning, that is, the case $|V| = 2n$ and $K = n$ and $w(v) = 1, \forall v \in V$.

Example: Scheduling in Parallel Machines

Total Weighted Completion Time on Unrelated Parallel Machines Problem

Input: A set of jobs J to be processed on a set of parallel machines M. Each job $j \in J$ has a weight w_j and processing time p_{ij} that depends on the machine $i \in M$ on which it is processed.

Task: Find a schedule of the jobs on the machines such that the sum of weighted completion time of the jobs is minimal.

Example: Steiner Tree

Steiner Tree Problem

Input: A graph $G = (V, E)$, a weight function $\omega : E \mapsto \mathbf{N}$, and a subset $U \subseteq V$.

Task: Find a Steiner tree, that is, a subtree $T = (V_T, E_T)$ of G that includes all the vertices of U and such that the sum of the weights of the edges in the subtree is minimal.

Vertices in U are the special vertices and vertices in $S = V \setminus U$ are Steiner vertices.