DM204, 2010
SCHEDULING, TIMETABLING AND ROUTING

Lecture 30
Vehicle Routing with Time Windows

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Course Overview

- ✔ Problem Introduction
  - ✔ Scheduling classification
  - ✔ Scheduling complexity
  - ✔ RCPSP
- ✔ General Methods
  - ✔ Integer Programming
  - ✔ Constraint Programming
  - ✔ Heuristics
  - ✔ Dynamic Programming
  - ✔ Branch and Bound
- ✔ Scheduling Models
  - ✔ Single Machine
  - ✔ Parallel Machine and Flow Shop
  - ✔ Job Shop
  - ✔ Resource-Constrained Project Scheduling
- Timetabling
  - ✔ Reservations and Education
  - ✔ Course Timetabling
  - ✔ Workforce Timetabling
  - ✔ Crew Scheduling
- Vehicle Routing
  - ✔ Capacitated Models
  - Time Windows models
  - Rich Models

Outline

Outline

VRPTW

$$
\begin{aligned}
\min \quad & \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} x_{ijk} && (1)\\
\text{s.t.} \quad & \sum_{k \in K} \sum_{(i,j) \in \delta^+(i)} x_{ijk} = 1 & \forall i \in V \quad & (2)\\
& \sum_{(i,j) \in \delta^+(0)} x_{ijk} = \sum_{(i,j) \in \delta^-(0)} x_{ijk} = 1 & \forall k \in K \quad & (3)\\
& \sum_{(i,j) \in \delta^-(i)} x_{jik} - \sum_{(i,j) \in \delta^+(i)} x_{ijk} = 0 & i \in V, k \in K \quad & (4)\\
& \sum_{(i,j) \in A} d_i x_{ijk} \leq C & \forall k \in K \quad & (5)\\
& x_{ijk}(w_{ik} + t_{ij}) \leq w_{jk} & \forall k \in K, (i,j) \in A \quad & (6)\\
& a_i \leq w_{ik} \leq b_i & \forall k \in K, i \in V \quad & (7)\\
& x_{ijk} \in \{0,1\} && (8)
\end{aligned}
$$

Pre-processing

- Arc elimination
  - $a_i + t_{ij} > b_j$ ➜ arc (i,j) cannot exist
  - $d_i + d_j > C$ ➜ arcs (i,j) and (j,i) cannot exist
- Time windows reduction
  - $[a_i, b_i] \leftarrow [\max\{a_0 + t_{0i}, a_i\}, \min\{b_{n+1} - t_{i,n+1}, b_i\}]$

- Time windows reduction:
  - Iterate over the following rules until no one applies anymore:

1) Minimal arrival time from predecessors:

$$a_l = \max\left\{a_l, \min\left\{b_l, \min_{(i,l)}\{a_i + t_{il}\}\right\}\right\}.$$

2) Minimal arrival time to successors:

$$a_l = \max\left\{a_l, \min\left\{b_l, \min_{(l,j)}\{a_j - t_{lj}\}\right\}\right\}.$$

3) Maximal departure time from predecessors:

$$b_l = \min\left\{b_l, \max\left\{a_l, \max_{(i,l)}\{b_i + t_{il}\}\right\}\right\}.$$

4) Maximal departure time to successors:

$$b_l = \min\left\{b_l, \max\left\{a_l, \max_{(l,j)}\{b_j - t_{lj}\}\right\}\right\}.$$

Lower Bounds

- Combinatorial relaxation
  relax constraints (5) and (6)
  reduce to network flow problem
- Linear relaxation
  fractional near-optimal solution has capacity and time windows constraints inactive

In both cases the bounds are weak

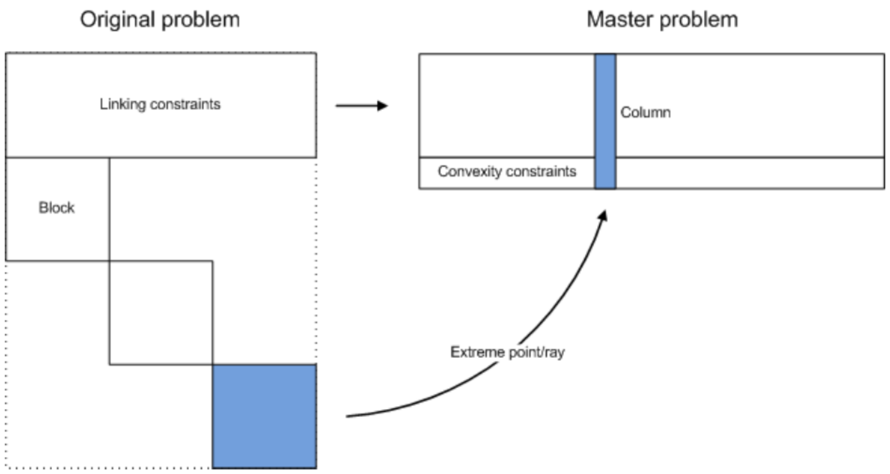
Dantzig Wolfe Decomposition

The VRPTW has the structure:

$$
\begin{aligned}
\min \quad & c^k x^k \\
& \sum_{k \in K} A^k x^k \leq b \\
& D^k x^k \leq d^k && \forall k \in K \\
& x^k \in \mathbb{Z} && \forall k \in K
\end{aligned}
$$

Dantzig Wolfe Decomposition

Illustrated with matrix blocks



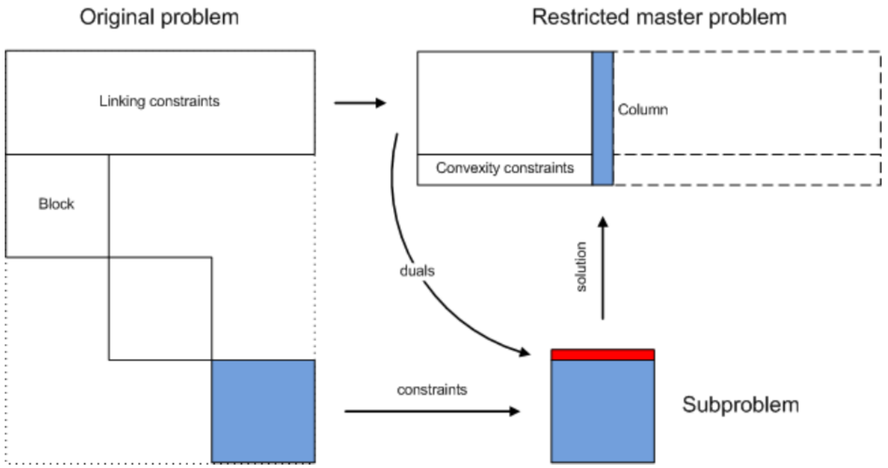
[illustration by Simon Spoorendonk, DIKU]

Linking constraint in VRPTW is $\sum_{k \in K} \sum_{(i,j) \in \delta^+(i)} x_{ijk} = 1, \quad \forall i$. The description of the block $D^k x^k \leq d^k$ is all the rest:

$$
\begin{aligned}
& \sum_{(i,j) \in A} d_i x_{ij} \leq C && (9) \\
& \sum_{j \in V} x_{0j} = \sum_{i \in V} x_{i,n+1} = 1 && (10) \\
& \sum_{i \in V} x_{ih} - \sum_{j \in V} x_{hj} = 0 & \forall h \in V \quad & (11) \\
& w_i + t_{ij} - M_{ij}(1 - x_{ij}) \leq w_j & \forall (i,j) \in A \quad & (12) \\
& a_i \leq w_i \leq b_i & \forall i \in V \quad & (13) \\
& x_{ij} \in \{0, 1\} && (14)
\end{aligned}
$$

where we omitted the index k because, by the assumption of homogeneous fleet, all blocks are equal.

Dantzig Wolfe Decomposition



[illustration by Simon Spoorendonk, DIKU]

Master problem

A Set Partitioning Problem

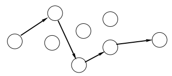
$$
\begin{aligned}
\min \quad & \sum_{p\in\mathcal{P}} c_{ij}\alpha_{ijp}\lambda_p & & (15)\\
& \sum_{p\in\mathcal{P}} \sum_{(i,j)\in\delta^+(i)} \alpha_{ijp}\lambda_p = 1 & \forall i \in V \quad & (16)\\
& \lambda_p = \{0,1\} & \forall p \in \mathcal{P} \quad & (17)
\end{aligned}
$$

where $\mathcal{P}$ is the set of valid paths and $\alpha_{ijp} = \begin{cases} 0 & \text{if } (i,j) \notin p \\ 1 & \text{otherwise} \end{cases}$

Subproblem

Elementary Shortest Path Problem with Resource Constraints (ESPPRC)

- arcs modified with duals (possible negative costs), NP-hard
- find shortest path without violating resource limits



Subproblem

$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in A} \hat{c}_{ij} x_{ij} & & (18)\\
\text{s.t.} \quad & \sum_{(i,j)\in A} d_i x_{ij} \leq C & & (19)\\
& \sum_{j\in V} x_{0j} = \sum_{i\in V} x_{i,n+1} = 1 & & (20)\\
& \sum_{i\in V} x_{ih} - \sum_{j\in V} x_{hj} = 0 & \forall h \in V \quad & (21)\\
& w_i + t_{ij} - M_{ij}(1 - x_{ij}) \leq w_j & \forall (i,j) \in A \quad & (22)\\
& a_i \leq w_i \leq b_i & \forall i \in V \quad & (23)\\
& x_{ij} \in \{0,1\} & & (24)
\end{aligned}
$$

Subproblem

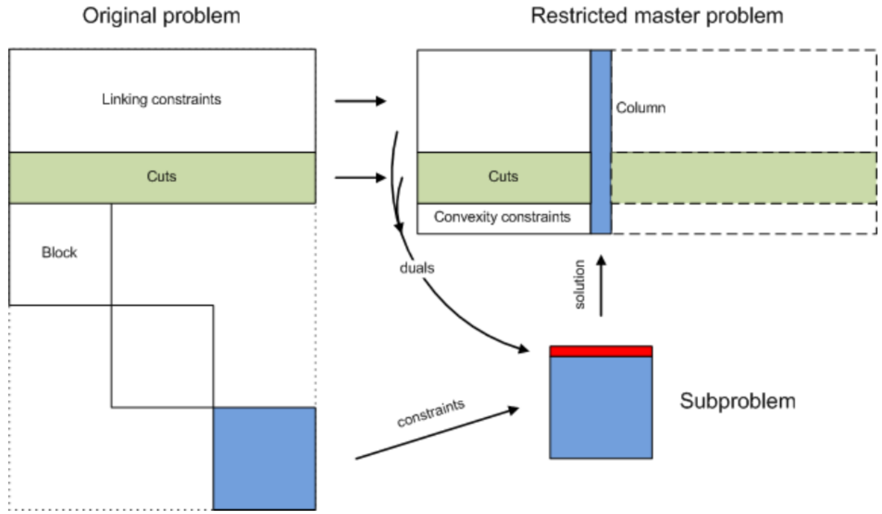
Solution approach:

- ESPPRC Solved by dynamic programming. Algorithms maintain labels at vertices and remove dominated labels. Domination rules are crucial.
- Relaxing and allowing cycles the SPPRC can be solved in pseudo-polynomial time.
  Negative cycles are however limited by the resource constraints.
  Cycle elimination procedures by post-processing
- Further extensions (arising from brnaching rules on the master):
  SPPRC with forbidden paths
  SPPRC with (i,j)-antipairing constraints
  SPPRC with (i,j)-follower constraint

For details see chp. 2 of [B11]

Branch and Bound

Cuts in the original three index problem formulation (before DWD)



[illustration by Simon Spoorendonk, DIKU]

Branching

- branch on original variables
  - $\sum_k x_{ijk} = 0/1$ imposes follower constraints on visits of i and j
  - choose a variable with fractional not close to 0 or 1, ie, $\max c_{ij}(\min\{x_{ijk}, 1 - x_{ijk}\})$

- branch on time windows
  split time windows s.t. at least one route becomes infeasible
  compute $[l_i^r, u_i^r]$ (earliest latest) for the current fractional flow
  $L_i = \max_{\text{fract. routes } r} \{l_i^r\} \qquad \forall i \in V$
  $U_i = \max_{\text{fract. routes } r} \{u_i^r\} \qquad \forall i \in V$
  if $L_i > U_i$ ➨ at least two routes have disjoint feasibility intervals