Applications

J.M. Valério de Carvalho
vc@dps.uminho.pt

Departamento de Produção e Sistemas
Escola de Engenharia, Universidade do Minho
Portugal

Ciclo di Seminari 'Column Generation'
Metodi e Modelli per l'Ottimizzazione Combinatoria
Corso di Laurea Magistrale in Informatica
Dipartimento di Matematica Pura e Applicata
Università degli Studi di Padova
19^{th} - 28^{th} October 2011

Global outline

- Part I - Decomposition methods
- Part II - Applications
- Part III - Branch-and-price algorithms
- Part IV - Branch-and-price algorithms (cont.)
- Part V - Practical issues, stabilization, accelerating strategies and heuristics
- Bibliography

Part II

Applications

Outline

- Reasons for using decomposition
- Block angular structure: examples
- Solving LP relaxations with column generation
- Application: Cutting Stock (CSP) and Bin Packing (BPP) Problems
- Application: Vehicle routing

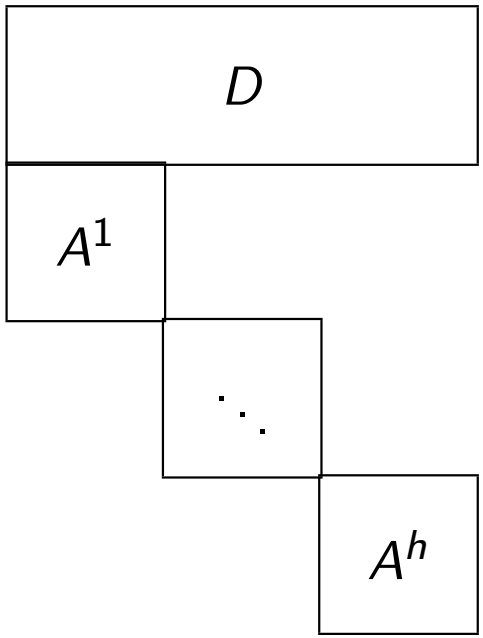
Reasons for using decomposition

Models from DW decomposition:

- become manageable in size: number of constraints is reduced and column generation is used.
- are suitable to deal with non-linear constraints: they are tackled in a dynamic programming subproblem.
- may be stronger: subproblems do not have the integrality property.
- may be the only models at hand, because compact models may not be known.

General structure: block angular with linking constraints

- DW decomposition partitions model into levels: Main problem and subproblem(s) (or Master and slave(s)).
- Subproblem(s) has(ve) nice structure that can be exploited (*e.g.*, network).



- Block D - Linking constraints
- Each of the blocks $A^1,\ldots,A^h$ defines a different subproblem

Examples of models resulting from structured problems

Problem	D block	A blocks
Production planning	Availability of common resources required for production (*e.g.*, machine capacities).	One block for each product. Production requirements of each product (for example, forced by existing demand).
Vehicle routing	Constraints imposed on the fleet of vehicles (*e.g.*, it must visit all the clients).	One for each vehicle. Route and vehicle constraints (*e.g.*, a route must end at a depot and vehicle capacity cannot be exceeded).
Generalised assignment	Constraints imposed on the group of agents (all the tasks must be performed).	One for each agent, related with its capacity.
Machine scheduling	Job constraints (*e.g.*, all jobs must be done).	One for each machine. Machine constraints (*e.g.*, two tasks cannot be made at the same time).

Decomposition framework

Master Problem (MP)

- "combines" independent solutions of SPs
- constraints in MP tell how resources are used by subproblem solutions

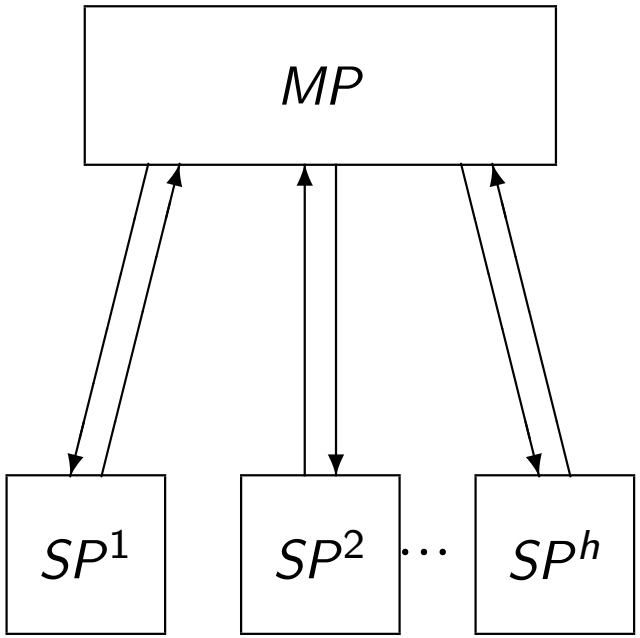
Subproblem(s) (SP)

- usually subproblem solutions are paths.
- difficult constraints (non-linearities) are tackled in the subproblem (solved with dynamic programming)
- SP use resources when economically efficient

Economic interpretation of DW decomposition

Master Problem (MP)

controls usage of resources: tells SP the price of the usage of resources



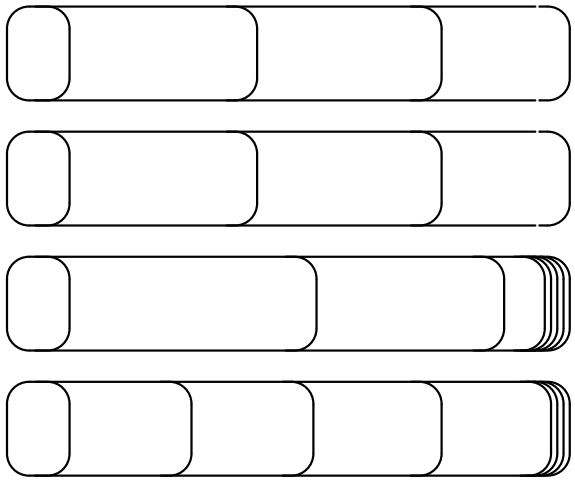
Subproblem(s) (SPs)

compete for resources: each SP makes its best bid to MP

Application I

- Cutting Stock Problem (CSP) and Bin Packing Problem (BPP)
- Kantorovich model
- Gilmore-Gomory model
- Solution of Gilmore-Gomory model by column generation
- Example (solution of LP relaxation)

Cutting Stock Problem



W: width of large rolls
w_i : width of rolls for client i, $i = \ldots, m$
b_i : demand of rolls of width w_i (many items of each size)
Objective: cut the minimum number of rolls to satisfy demand.

Cutting and packing problems

Bin packing problem:

- given an unbounded number of bins of capacity W and a list of n items of size w_i, $0 < w_i \leq W$, $i = 1, \ldots, n$,
- pack all the items in the smallest number of bins without exceeding their capacity.
- few items of each size.

Cutting stock problem:

- given an unbounded number of rolls of size W, and given m clients with demands of b_i rolls of size w_i, $0 < w_i \leq W$, $i = \ldots, m$,
- cut the minimum number of rolls to satisfy demand.
- many items of each size.

Cutting Stock Problem: a weak model

$$\text{Decision variables } x_{ij} = \begin{cases} 1 & \text{, if item } j \text{ is placed in roll } i \\ 0 & \text{, otherwise} \end{cases}$$

$$\text{Decision variables } y_i = \begin{cases} 1 & \text{, if roll } i \text{ is used} \\ 0 & \text{, otherwise} \end{cases}$$

$$\min z_{IP} = \sum_{i=1}^{n} y_i$$

$$\text{subj. to} \quad \sum_{j=1}^{n} w_j x_{ij} \leq W y_i, \ \forall i \in I$$

$$\sum_{i=1}^{n} x_{ij} = 1, \ \forall j \in J$$

$$y_i = 0 \text{ or } 1, \ \forall i$$

$$x_{ij} = 0 \text{ or } 1, \ \forall i,j$$

L. Kantorovich, "Mathematical methods of organising and planning production" (translated from a paper in Russian, dated 1939), Management Science 6, 366–422, 1960.

Quality of the relaxation: value of the lower bound

LP relaxation: replace the last two constraints by $0 \leq y_i \leq 1, \forall i$, and $0 \leq x_{ij} \leq 1, \forall i, j$, respectively.

LP relaxation optimum, z^*_{LP}, is a lower bound for the IP optimum.

Proposition (Martello and Toth, 90)

Lower bound $LB_1 = \lceil \sum_{i=1}^{n} w_i / W \rceil$.

Proof: No solution can have an objective value lower than $\sum_{i=1}^{n} w_i / W$.

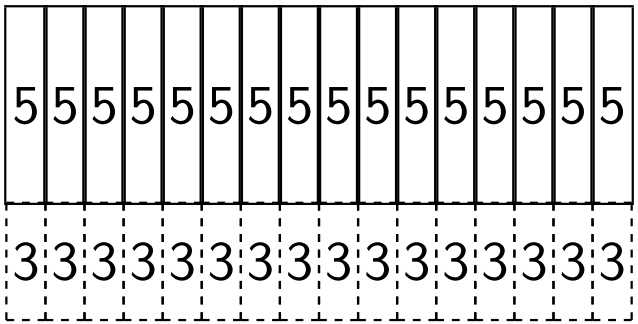
Solution $x_{ii} = 1, x_{ij} = 0, \forall j \neq i$, and $y_i = w_i / W, \forall i$, has an objective value, $z^*_{LP} = \sum_{i=1}^{n} w_i / W$, equal to that value. So, it is an optimal LP solution.

Round up, because the number of bins must be integer. □

Bound can be very poor for instances with large loss: there may be cases in which $z^*_{LP} \to 1/2\ z_{IP}$.

Example: Bins of capacity 8 and 16 items of size 5

Integer optimum is 16:



Linear relaxation optimum is 10:

$$\begin{array}{rcl} x_{11} = 1 & , & y_1 = 5/8 \\ x_{22} = 1 & , & y_2 = 5/8 \\ \ldots & & \ldots \\ x_{16,16} = 1 & , & y_{16} = 5/8 \end{array}$$

$$\sum_i y_i = 10$$

Gap between Integer and Linear Relaxation optima, $z_{IP} - z_{LP} = 6$.

Cutting Stock Problem: Gilmore-Gomory model

Cutting Pattern: possible arrangement of items in the roll:

$$\sum_{i=1}^{m} a_{ij} w_i \leq W$$
$$a_{ij} \geq 0 \text{ and integer, } \forall j \in J.$$

a_{ij} : number of items of width w_i obtained in the cutting pattern j
J : set of valid cutting patterns.
x_j : number of rolls cut according cutting pattern j.

$$\begin{aligned} min\ z_{IP} \quad &= \quad \sum_{j \in J} x_j \\ subj.\ to \quad & \quad \sum_{j \in J} a_{ij} x_j \geq b_i,\ i = 1, 2, \ldots, m \\ & \quad x_j \geq 0 \text{ and integer, } \forall j \in J \end{aligned}$$

Example (cont.): Bins of capacity 8 and 16 items of size 5

The only valid cutting pattern is:

Mathematical formulation:

$$\begin{array}{lcl} \min\ z_{LP} & = & x_1 \\ \text{subj. to} & & 1x_1 \geq 16 \\ & & x_1 \geq 0 \end{array}$$

Optimal value of linear relaxation $z_{LP} = 16$, when $x_1 = 16$.

Gap between Integer and Linear Relaxation optima, $z_{IP} - z_{LP} = 0$.

Gilmore-Gomory's bound in practice

- Bound given by the LP relaxation of Gilmore-Gomory's model is very tight.
- Most of the one-dimensional cutting stock instances have gaps smaller than one.
- There are instances with gaps equal to 1 (O.Marcotte'1985,86).
- Largest gap known is $\frac{7}{6}$ (Rietz,Scheithauer'2002).
- Conjecture: all instances have gaps smaller than 2 (modified integer round-up property) (Scheithauer,Terno'1995).

Column generation for CSP [Gilmore, Gomory, 1961]

Generally, it is unpractical to enumerate all valid cutting patterns.

Solve linear programming relaxation of CSP using column generation:

Choose an initial restricted set of cutting patterns
While (there is an a attractive cutting pattern) do
    add attractive cutting pattern to restricted problem
    reoptimize
End While

To get an integer solution, round up fractional values of cutting patterns.
Solutions are of good quality, if the quantities demanded are high.

Cutting Stock Problem: Restricted Problem

$$
\begin{aligned}
min \quad & z_{LP} = \sum_{j \in \bar{J}} x_j \\
subj.to \quad & \sum_{j \in \bar{J}} a_{ij} x_j \geq b_i, \ i = 1, 2, \ldots, m \\
& x_j \geq 0, \ \forall j \in \bar{J},
\end{aligned}
$$

$\bar{J}$: subset of cutting patterns in restricted problem
$\pi = \pi(\bar{J}) = (\pi_1, \pi_2, \ldots, \pi_m)$: optimal dual solution with subset $\bar{J}$

Pricing cutting patterns out of the restricted problem:

$$
\begin{aligned}
\text{Reduced cost of cutting pattern } j \quad &= \quad c_j - c_B B^{-1} A_j = \\
&= \quad 1 - \sum_{i=1}^{m} a_{ij} \pi_i
\end{aligned}
$$

Column is attractive if its reduced cost < 0

Cutting Stock Problem: objective function of subproblem

Find most attractive cutting pattern $\in J \setminus \overline{J}$:

$$\min_{j \in J \setminus \overline{J}} \; 1 - \sum_{i=1}^{m} a_{ij} \pi_i$$

Columns in $\overline{J}$ have reduced costs ≥ 0; so, search over J:

$$\min_{j \in J} \; 1 - \sum_{i=1}^{m} a_{ij} \pi_i$$

Maximize symmetric function:

$$\min_{j \in J} \; 1 - \sum_{i=1}^{m} a_{ij} \pi_i \;\equiv\; \max_{j \in J} \; \sum_{i=1}^{m} a_{ij} \pi_i - 1$$

Cutting Stock Problem: knapsack subproblem

Knapsack subproblem:

$$\max \; z_s = \sum_{i=1}^{m} \pi_i y_i$$

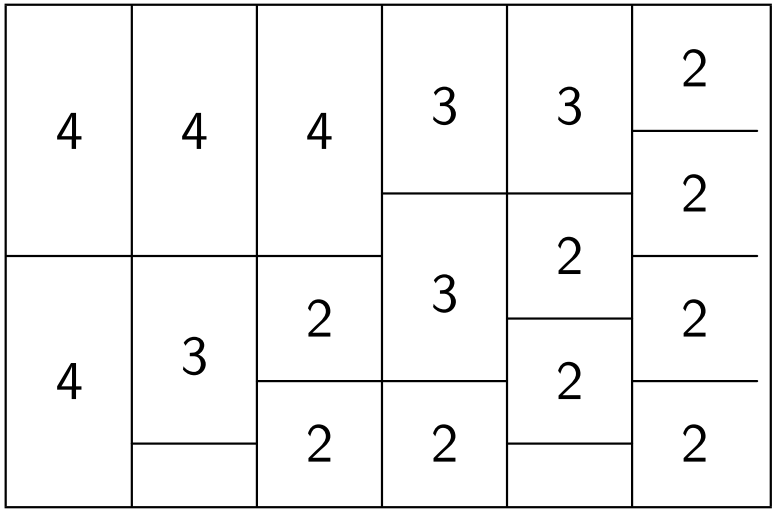
$$\textit{subj. to} \quad \sum_{i=1}^{m} w_i y_i \leq W$$

$$y_i \geq 0 \text{ and integer}, \; i = 1, 2, \ldots, m,$$

y_i : number of items of size w_i in the new cutting pattern

If optimum $z_s^* > 1$, cutting pattern is attractive.
If no attractive columns, solution is optimal.

(Very Small) Example



$W = 8$	cutting patterns						Demand b_i	
	x_1	x_2	x_3	x_4	x_5	x_6		
$w_i = 4$	2	1	1				$\geq$	5
3		1		2	1		$\geq$	4
2			2	1	2	4	$\geq$	8
min	1	1	1	1	1	1		

(Very Small) Example (cont.)

$W = 8$	cutting patterns						Demand b_i	
	x_1	x_2	x_3	x_4	x_5	x_6		
$w_i = 4$	2	1	1				$\geq$	5
3		1		2	1		$\geq$	4
2			2	1	2	4	$\geq$	8
min	1	1	1	1	1	1		

Optimal fractional solution

2.5	2.0	1.5

6 rolls

Fractional solution rounded up

3.0	2.0	2.0

7 rolls

Excess production: 1 item of width 4 and 2 items of width 2

Restricted problem: first iteration

Initial solution: 3 columns, each with items of the same size (as suggested by Gilmore and Gomory'61).

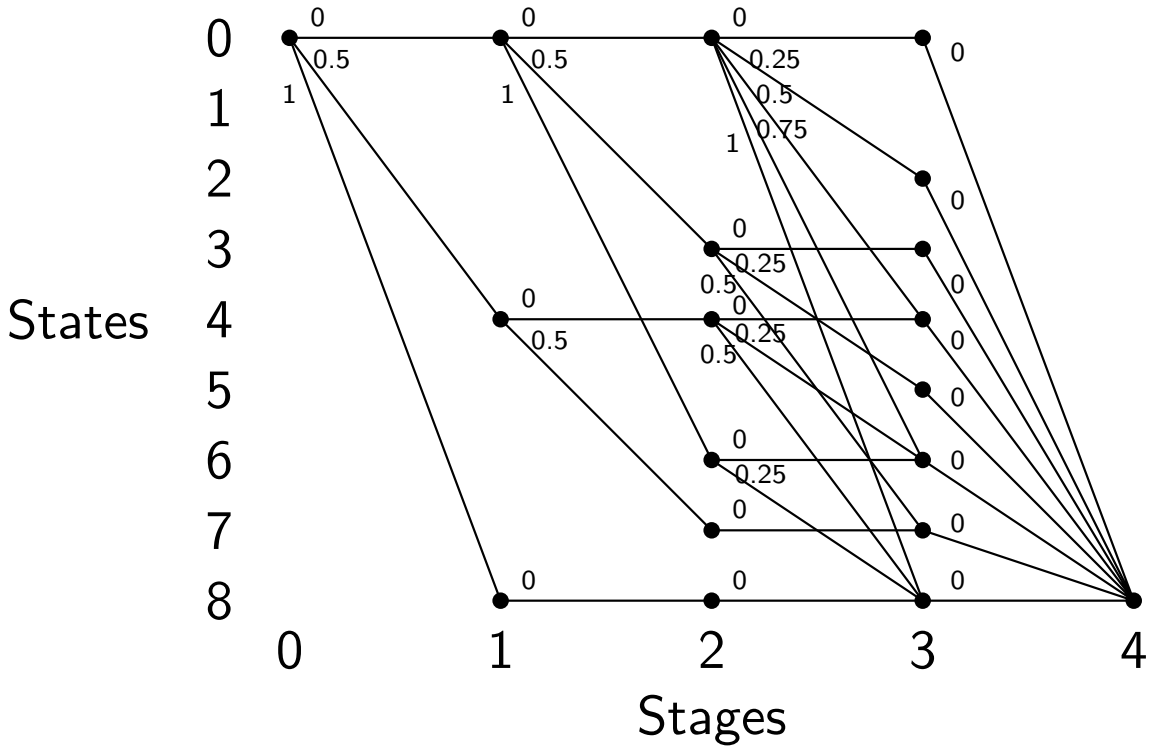
Using an LP solver, we obtain the following optimal solution (primal and dual):

	x_1	x_2	x_3			dual
$w_d =$ 4	2			$\geq$	5	0.5
3		2		$\geq$	4	0.5
2			4	$\geq$	8	0.25
min	1	1	1			

primal	2.5	2.0	2.0	$z^0 =$	6.5

Subproblem: first iteration

$$
\begin{array}{lrrrl}
\max\ z_s = & 0.5y_1 + & 0.5y_2 + & 0.25y_3 & \\
\textit{subj. to} & 4y_1 + & 3y_2 + & 2y_3 & \leq 8 \\
 & y_j \geq 0 \text{ and integer}, \forall j & & &
\end{array}
$$



Optimal solution: $(y_1, y_2, y_3) = (0, 2, 1), z_s^* = 1.25 \rightarrow$ Attractive

Subproblem: knapsack problem

- Dynamic programming
- $F_i(x)$: maximum value from placing items with index less than or equal to i using x units of space.
- Recursive equations of Knapsack Problem with upper bounds on variables:

$$\begin{aligned} F_0(0) &= 0 \\ F_i(x) &= \max_{x - l w_i \geq 0} \{F_{i-1}(x - l w_i) + l\overline{\pi}_i : 0 \leq l \leq l_i^{max}\}, \end{aligned}$$

$$x = 0, 1, \ldots, W; \quad i = 1, 2, ..., m.$$

- Largest number of items of a given size in a cutting pattern (element a_{ij} in column j) must also be less than or equal to the demand of that size:

$$l_i^{max} = a_{ij}^{max} = \min\left\{b_i, \left\lfloor \frac{W}{w_i} \right\rfloor\right\}$$

- Computational complexity is $O(mW^2)$
- weakly NP-hard (pseudo-polynomial)

Restricted problem: second iteration

Attractive cutting pattern: 2 items of size 3 and 1 item of size 2.

Insert attractive column in the restricted problem, and reoptimize.

Optimal solution:

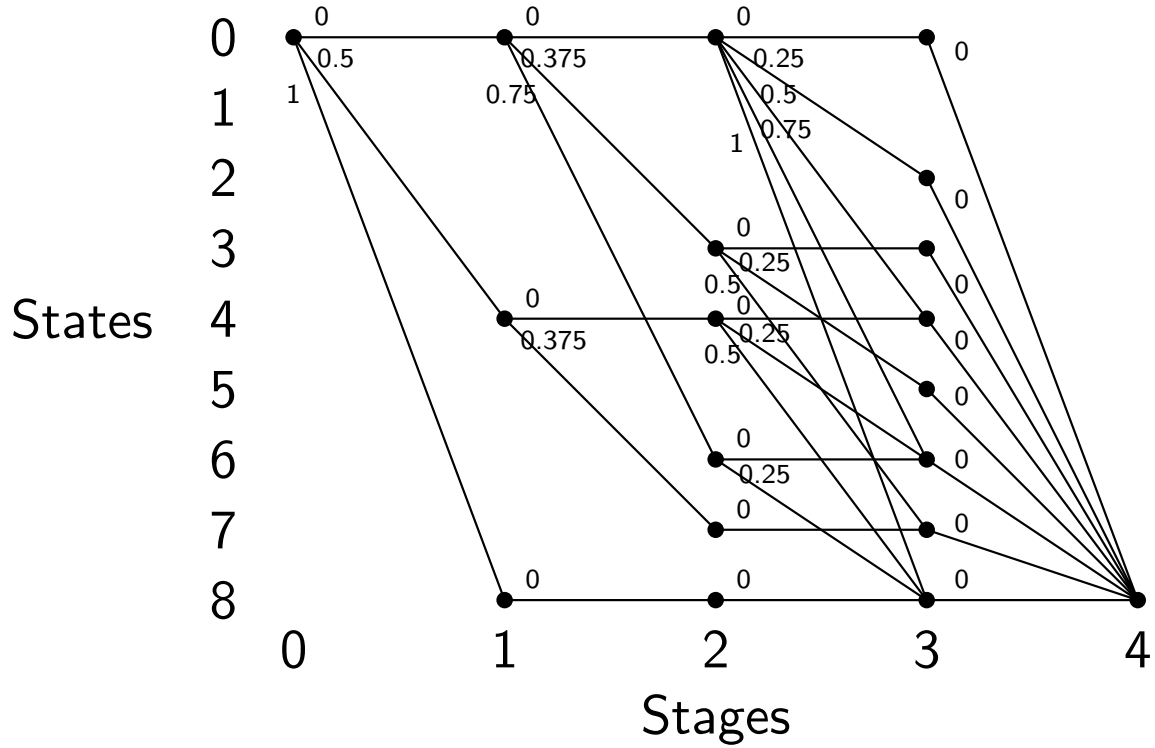
	x_1	x_2	x_3	x_4			dual
$w_d =$ 4	2				$\geq$	5	0.5
3		2		2	$\geq$	4	0.375
2			4	1	$\geq$	8	0.25
min	1	1	1	1			

primal	2.5	0.0	1.5	2.0

$z_{LP} =$ 6.0

Subproblem: second iteration

$$\begin{aligned} \max\ z_s = \quad & 0.5y_1 + \quad 0.375y_2 + \quad 0.25y_3 \\ subj.\ to \quad & 4y_1 + \quad 3y_2 + \quad 2y_3 \quad \leq 8 \\ & y_j \geq 0 \text{ and integer}, \forall j \end{aligned}$$



Alternative optima (Value $z_s^* = 1.0$) → No attractive columns. So...

Optimal solution of the linear relaxation

	x_1	x_2	x_3	x_4			dual
$w_d =$ 4	2				$\geq$	5	0.5
3		2		2	$\geq$	4	0.375
2			4	1	$\geq$	8	0.25
min	1	1	1	1			

primal | 2.5 | 0.0 | 1.5 | 2.0 |

$z_{LP} =$ 6.0

	A_1	A_3	A_4
4	2	3	
	2		
4	2	3	
	2		2
$x_j =$	2.5	1.5	2.0

Strengthening the formulation

LP relaxation has an optimal value z_{LP}. Optimal solution has an integer value.
Round-up: use a number of rolls $\geq$ LP optimum rounded up:

$$\sum_{j \in \bar{J}} x_j \quad \geq \quad \lceil z_{LP} \rceil$$

In this case, z_{LP} is integer: new constraint does not change the optimal solution.

	x_1	x_2	x_3	x_4			dual
$w_d =$ 4	2				$\geq$	5	0.5
3		2		2	$\geq$	4	0.375
2			4	1	$\geq$	8	0.25
round-up	1	1	1	1	$\geq$	6	0.0
min	1	1	1	1			

primal	2.5	0.0	1.5	2.0

$z_{LP} =$ 6.0

Easy to transfer dual information to subproblem.

Application II

- Vehicle Routing Problem with Time Windows
- Flow model
- Reformulated model
- Subproblem
- Dealing with subproblem
- Resource constraints: a general framework

Vehicle Routing Problem with Time Windows

Statement

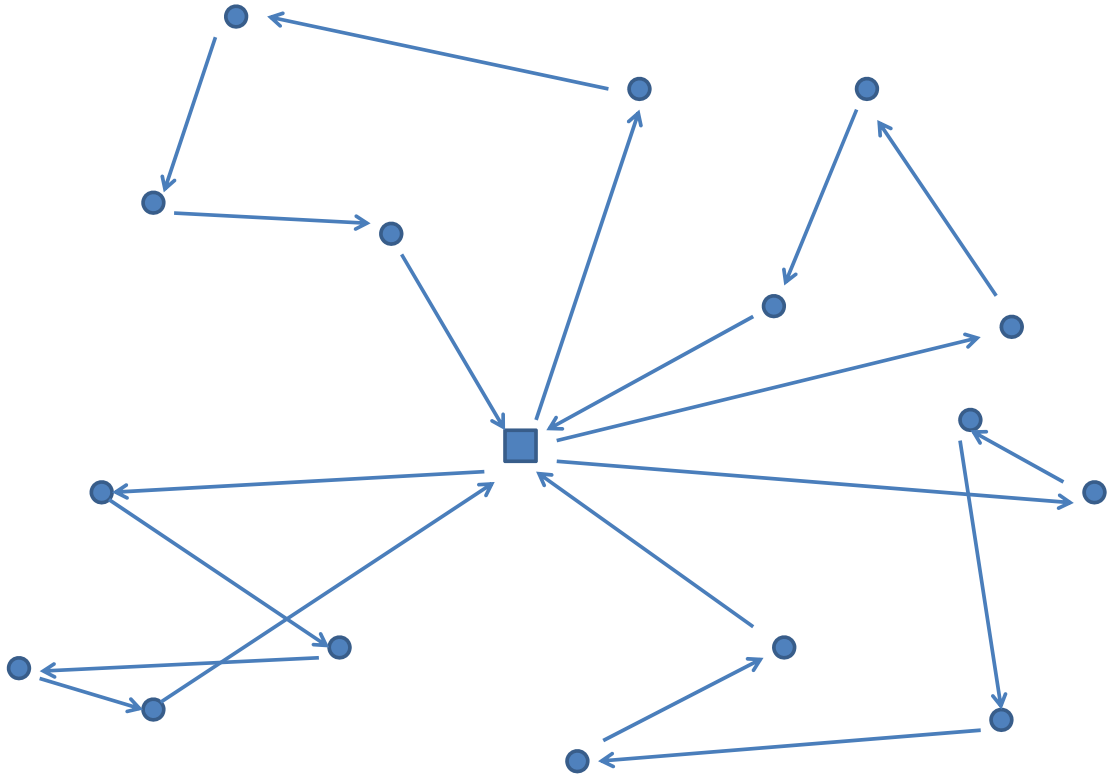
Given

- a set of vehicles with given capacities,
- a set clients with given demands and time windows,

find

- a set of routes, all starting and ending at the depot,
- such that each client is visited by one vehicle in a way that minimizes costs.

A set of routes



Mathematical model

Set of clients $N = \{1, 2, \ldots, n\}$

- demands d_i, $i \in N$
- time windows $[a_i, b_i]$, $i \in N$.

Set of homogeneous vehicles $\{1, 2, \ldots, K\}$

- K is known
- capacity Q

Mathematical model

Single depot, which is the origin and the destination of all vehicle routes:

split into 2 nodes:

- origin node $o \equiv$ vertex 0
- destination node $d \equiv$ vertex $n+1$
- no demand: $d_0 = d_{n+1} = 0$
- time windows $[a_0, b_0] = [a_{n+1}, b_{n+1}]$

Mathematical model

Graph $G = (V, A)$

- $V = N \cup \{o, d\}$ represents the set of nodes
- A the set of oriented arcs.

arc $(i, j) \in A \subset V \times V$:

- c_{ij} : cost incurred in travelling through the arc
- t_{ij} : travel time (includes service time of node i)
- for an arc to be feasible,

$$a_i + t_{ij} \leq b_j$$

The optimization objective of the plan is to minimize the total cost of the vehicles routes.

Mathematical model

Feasible route: path $p = (v_0, v_1, \ldots, v_H)$

- starts at origin node $(v_0 = o)$
- ends at destination node $(v_H = d)$
- visits customers $v_i \in N,\ i = 1, \ldots, H-1$
- obeys capacity constraints $\sum_{i=1}^{H-1} d_i \leq Q$
- obeys time windows:

$$\begin{aligned} T_0 &= a_{v_0} \\ T_{i+1} &= \max\{a_{v_i+1}, T_i + t_{v_i,v_{i+1}}\} \leq b_{v_{i+1}},\ \forall i = 0, \ldots, H-1 \end{aligned}$$

Decision variables

Flow variables:

- $x_{ij}^k = \begin{cases} 1 & \text{, if vehicle } k \text{ travels from client } i \text{ to client } j \\ 0 & \text{, otherwise} \end{cases}$

  $\forall k = 1, \ldots, K, (i,j) \in A$

Time variables:

- T_i^k : start of service of vehicle k at node i

  $\forall k = 1, \ldots, K, i \in V$

Model with arc variables

$$\min \quad \sum_{k=1}^{K} \sum_{(i,j)\in A} c_{ij} x_{ij}^k \tag{1}$$

$$s.to \quad \sum_{k=1}^{K} \sum_{(i,j)\in \delta^+(i)} x_{ij}^k = 1, \ \forall i \in N \tag{2}$$

$$\sum_{(0,j)\in \delta^+(0)} x_{0j}^k = 1, \ \forall k = 1,\ldots,K \tag{3}$$

$$\sum_{(i,j)\in A} d_i x_{ij}^k \leq Q, \ \forall k = 1,\ldots,K \tag{4}$$

$$\sum_{(i,j)\in \delta^-(j)} x_{ij}^k = \sum_{(j,i)\in \delta^+(j)} x_{ji}^k, \ \forall j \in N, k = 1,\ldots,K \tag{5}$$

$$\sum_{(i,d)\in \delta^-(d)} x_{id}^k = 1, \ \forall k = 1,\ldots,K \tag{6}$$

$$T_i^k - T_j^k + M x_{ij}^k \leq M - t_{ij}, \ \forall k = 1,\ldots,K, (i,j) \in A \tag{7}$$

$$a_i \leq T_i^k \leq b_i, \ \forall k = 1,\ldots,K, i \in V \tag{8}$$

$$x_{ij}^k \in \{0,1\}, \ \forall k = 1,\ldots,K, (i,j) \in A \tag{9}$$

Time constraints (7)

$$T_i^k - T_j^k + Mx_{ij}^k \leq M - t_{ij},\ \forall k = 1,\ldots,K,(i,j) \in A$$

- $M = b_i - a_j + t_{ij}$ provides a tighter constraint
- an alternative way of expressing constraint is the *non-linear* constraint:

$$(T_i^k - T_j^k + t_{ij})x_{ij}^k \leq 0,\ \forall k = 1,\ldots,K,(i,j) \in A$$

Dantzig-Wolfe decomposition

- Keep in the master problem the partitioning constraints
- Remaining constraints in subproblem k
- Subproblem k finds solutions which are *elementary shortest paths* with capacity constraints and time windows
- extreme points are feasible route $\equiv$ paths
- each decision variable corresponds to a path for vehicle k
- if the fleet is homogeneous, all blocks are identical

Reformulated model

- P^k : set of feasible routes for vehicle k, each obeying the constraints,
- $y_p^k \in \{0,1\}$: vehicle k does route $p \in P^k$
- $c_p^k = \sum_{i=0}^{h} c_{v_i,v_{i+1}}^k$: cost of vehicle k in path $p \in P^k$

$$\begin{aligned}
\min \quad & \sum_{k=1}^{K} \sum_{p \in P^k} c_p^k y_p^k \\
\text{s. to} \quad & \sum_{k=1}^{K} \sum_{p \in P^k} \delta_{ip}^k y_p^k = 1 \ , \ \forall i \in V \\
& \sum_{p \in P^k} y_p^k = 1, \ k = 1, \ldots, K \\
& y_p^k \in \{0,1\}, \ \forall p \in P^k, k = 1, \ldots, K
\end{aligned}$$

- where

$$\delta_{ip}^k = \begin{cases} 1 & \text{, if route } p \text{ of vehicle } k \text{ visits client } i \\ 0 & \text{, otherwise} \end{cases}$$

Reformulated model with all vehicle identical

- $P = P^k,\ k = 1,\ldots,K$: all vehicles are identical
- convexity constraints $\sum_{p\in P^k} y_p^k = 1,\ k = 1,\ldots,K$ can be aggregated into a single constraint $\sum_{k=1}^{K}\sum_{p\in P^k} y_p^k = K$.
- Vehicles with path (o,d) do not leave the depot;
- they act as slack variables: above constraint is a $\leq$ constraint.
- Using $y_p = \sum_{k=1}^{K} y_p^k$, then

$$
\begin{aligned}
\min \quad & \sum_{p\in P} c_p y_p \\
\text{s. to} \quad & \sum_{p\in P} \delta_{ip} y_p = 1\ ,\ \forall i \in V \\
& \sum_{p\in P} y_p \leq K \\
& y_p \in \{0,1\},\ \forall p \in P
\end{aligned}
$$

where

$$
\delta_{ip} = \begin{cases} 1 & \text{, if route } p \text{ visits client } i \\ 0 & \text{, otherwise} \end{cases}
$$

Example with 8 clients

	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}	y_{12}	y_{13}	y_{14}	y_{15}	
node 1	1	1	1	1	1	1										$= 1$
2	1	1					1	1	1	1	1					$= 1$
3			1	1			1					1	1	1		$= 1$
4	1							1	1			1				$= 1$
5					1					1					1	$= 1$
6		1	1					1					1			$= 1$
7				1			1				1	1		1		$= 1$
8						1					1		1	1	1	$= 1$
vehicles	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	≤ 4
cost	8	7	10	9	8	7	10	11	7	6	10	9	10	12	7	

Subproblem

Find path with minimum reduced cost:

$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in A} \overline{c}_{ij} x_{ij} \\
s.to \quad & \sum_{(0,j)\in \delta^+(0)} x_{0j} = 1 \\
& \sum_{(i,j)\in A} d_i x_{ij} \leq Q \\
& \sum_{(i,j)\in \delta^-(j)} x_{ij} = \sum_{(j,i)\in \delta^+(j)} x_{ji}, \ \forall j \in N \\
& (T_i - T_j + t_{ij}) x_{ij} \leq 0, \ \forall (i,j) \in A \\
& a_i \leq T_i \leq b_i, \ \forall i \in V \\
& x_{ij} \in \{0,1\}, \ \forall (i,j) \in A
\end{aligned}
$$

Subproblem

Elementary Shortest Path Problem with Time Windows and Capacity constraints (ESPPTWC)

- path p
- starts at origin node o, and ends at destination node $n+1$
- obeys capacity constraints
- obeys time windows, and
- with minimum reduced cost $\overline{c}_p = \sum \overline{c}_{ij} x_{ij}$

Reduced costs of arcs:

- $\overline{c}_{ij} = c_{ij} - \pi_i, \ \forall (i,j) \in A, i \neq o$
- $\overline{c}_{ij} = c_{ij} - \mu, \ \forall (o,j) \in A$, where
- π_i : dual variable of visit constraint to node i,
- μ : dual variable of number of vehicles constraint

arcs with negative reduced cost induce negative cost cycles in the network!

ESPPTWC

- Dynamic programming
- $F(S,i,t)$: minimum cost of path from o to $i, i \in N \cup \{d\}$, visiting all nodes in set $S \subseteq N \cup \{d\}$ *only once,* and servicing node i at time t or later.
- Recursive equations:

$$\begin{aligned} F(\emptyset, o, a_0) &= 0 \\ F(S,j,t) &= \min_{(i,j)\in A}\{F(S-\{j\}, i, t') + c_{ij} : \\ & \qquad i \in S-\{j\}, t' \leq t - t_{ij}, a_i \leq t' \leq b_i\}, \\ & \qquad \forall S \subseteq N \cup \{d\}, j \in S, a_j \leq t \leq b_j \end{aligned}$$

- Optimal solution:

$$\min_{S\subseteq N\cup\{d\}} \min_{a_d \leq t \leq b_d} F(S,d,t)$$

- Strongly NP-hard (and very hard to solve in practice)

Relax elementary path requirement to get an easier problem:

Shortest Path Problem with Time Windows and Capacity constraints

- $F(i,t)$: minimum cost of path from o to $i, i \in N \cup \{d\}$, and servicing node i at time t or later.
- Recursive equations:

$$\begin{aligned} F(o,a_0) &= 0 \\ F(j,t) &= \min_{(i,j)\in A}\{F(i,t') + c_{ij} : \\ & \quad i \in N \cup \{d\} - \{j\}, t' \leq t - t_{ij}, a_i \leq t' \leq b_i\}, \\ & \quad \forall j \in N \cup \{d\}, a_j \leq t \leq b_j \end{aligned}$$

- Optimal solution:

$$\min_{a_d \leq t \leq b_d} F(d,t)$$

- now, node i can be visited more than once in path p,
- in the RMP, we get coefficients δ_{ip} that may be larger than 1.
- Weakly NP-hard (pseudo-polynomial)

Different relaxations of ESPPTWC

It is possible to design a dynamic programming recursion that eliminates some of the cycles generated in the solution of the subproblem.

- SPPTWC: all cycles are allowed
- SPPTWC-2-cycles: 2-cycles are not allowed
- SPPTWC-k-cycles: cycles of length $\leq k$ are not allowed

Trade-off:

- larger values of k produce stronger LP-relaxation of master problem
- larger values of k induce much more complex recursion, more difficult to code

Shortest path problem with resource constraints (SPPRC)

Desrochers'86:

Generalization when there is a set of resources

Set of resources R

- travel time t_{ij} is replaced by the consumption of t_{ij}^r units of resource $r, \forall r \in R$
- time interval constraint $[a_i, b_i]$ of node i is replaced by $|R|$ constraints $[a_i^r, b_i^r]$, $\forall r \in R$
- T_i^r : amount of resource r used to reach node i, starting from o
- a path using less than a_i^r resources to reach node i wastes some resources and is feasible
- a path using more than b_i^r resources to reach node i is infeasible
- For an arc (i,j) to be feasible

$$a_i^r + t_{ij}^r \leq b_j^r, \ \forall r \in R$$

SPPRC

- extension of classical shortest path problem
- cost is replaced by multidimensional resource vector
- resources are accumulated / propagated along arcs
- resource values are constrained at the nodes
- Additional material

S. Irnich and D. Villeneuve, The Shortest-Path Problem with Resource Constraints and k-Cycle Elimination for $k \geq 3$, INFORMS Journal on Computing 18(3), pp. 391.406, 2006.

Concluding remarks

- column generation is an intuitive framework.
- very effective in many applications.
- field is growing.