Practical issues, stabilization, accelerating strategies and heuristics

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

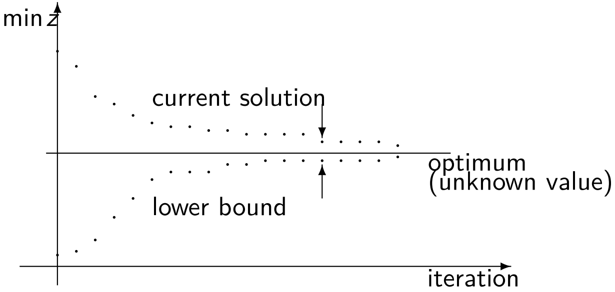
- Part I - Decomposition methods
- Part II - Applications
- Part III - Branch-and-price algorithms
- Part IV - Branch-and-price algorithms (cont.)
- Part V - Practical issues, stabilization, accelerating strategies and heuristics
- Bibliography

Part VI

Practical issues, stabilization, accelerating strategies and heuristics

Outline

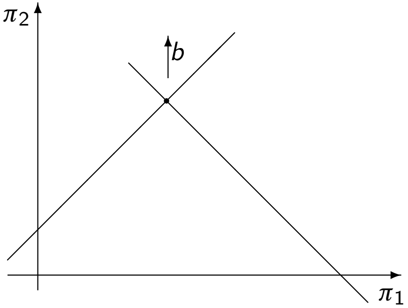
- Dual-cutting and Stabilization
  - Primal and dual perspectives
  - Stabilizing terms: examples
  - Degeneracy and perturbation
  - Perfect Dual Information
  - (Weak and deep) dual-optimal inequalities
  - Application: cutting stock problem
- Pre-processing
- Master problem
- Subproblem
- Branch-and-bound



Slow convergence: large changes in the values of the dual variables, which oscillate from one iteration to the next.
Degeneracy: in many iterations, adding new columns to restricted master problem does not improve objective value.

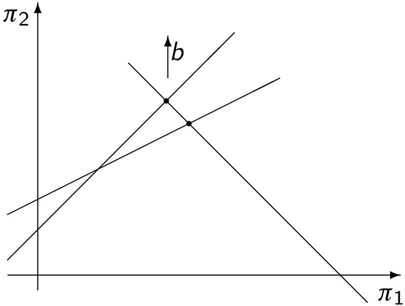
Instability of the values of the dual variables

- Dual objective: maximize dual function πb, with gradient b.
- Domain is successively restricted by adding dual constraints.
- π_2 gets smaller at each iteration (πb also does).
- π_1 oscillates until optimum dual solution is reached.



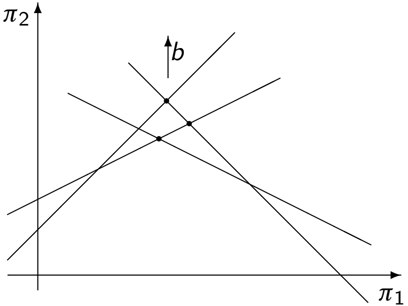
Instability of the values of the dual variables

- Dual objective: maximize dual function πb, with gradient b.
- Domain is successively restricted by adding dual constraints.
- π_2 gets smaller at each iteration (πb also does).
- π_1 oscillates until optimum dual solution is reached.



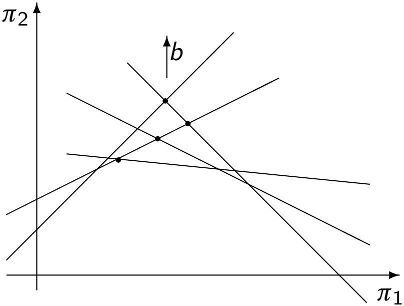
Instability of the values of the dual variables

- Dual objective: maximize dual function πb, with gradient b.
- Domain is successively restricted by adding dual constraints.
- π_2 gets smaller at each iteration (πb also does).
- π_1 oscillates until optimum dual solution is reached.



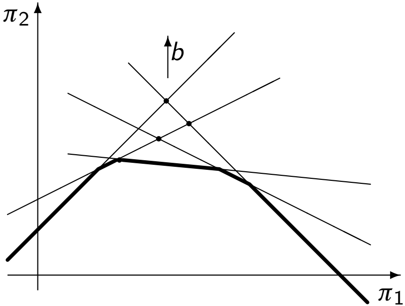
Instability of the values of the dual variables

- Dual objective: maximize dual function πb, with gradient b.
- Domain is successively restricted by adding dual constraints.
- π_2 gets smaller at each iteration (πb also does).
- π_1 oscillates until optimum dual solution is reached.



Instability of the values of the dual variables

- Dual objective: maximize dual function πb, with gradient b.
- Domain is successively restricted by adding dual constraints.
- π_2 gets smaller at each iteration (πb also does).
- π_1 oscillates until optimum dual solution is reached.



Acceleration of column generation: motivation

Restricting the dual space may accelerate column generation.
Better convergence: smaller number of attractive columns in subproblem.
Less degeneracy: alternative dual solutions ≡ degenerate primal solutions.

How to do it [VC, 2005]:
Add valid dual cuts to the model before starting column generation.

Restricting the dual space by setting lower bounds on dual variables:
GilmoreGomory'61: for any optimal primal solution with slack for the CSP, there is an alternative optimal primal solution without slack.
What to do do: allow solutions with slack, substituting primal $=$ for $\geq$ constraints.
Faster convergence: at a given restricted master problem, there may be a solution with slack better than all solutions without slack.
Dual perspective: dual variables are restricted to be ≥ 0, instead of unrestricted.

Same happens in many practical applications, when it is valid to work with set covering instead of set partitioning formulations.

Column generation: dual perspective

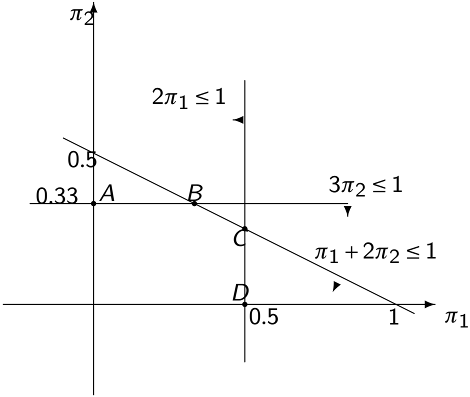
Cutting plane algorithm: adding a column in the primal is equivalent to adding a cut in the dual.

$$(Primal)\quad \begin{array}{ll} \min & cx \\ s.t. & Ax \geq b \\ & x \geq 0 \end{array} \qquad\qquad (Dual)\quad \begin{array}{ll} \max & \pi b \\ s.t. & \pi A \leq c \end{array}$$

CSP Example: rolls of width 10, items of size 4 and 3

$$(Primal)\quad \begin{array}{llllll} \min & 1x_1 & +1x_2 & +1x_3 & & \\ s.t. & 2x_1 & +1x_2 & & \geq & b_1 \\ & & +2x_2 & +3x_3 & \geq & b_2 \\ & & x_1, x_2, x_3 \geq 0 & & & \end{array} \qquad (Dual)\quad \begin{array}{lllll} \max & b_1\pi_1 & +b_2\pi_2 & & \\ s.t. & 2\pi_1 & & \leq & 1 \\ & 1\pi_1 & +2\pi_2 & \leq & 1 \\ & & 3\pi_2 & \leq & 1 \\ & \pi_1, \pi_2 \geq 0 & & & \end{array}$$

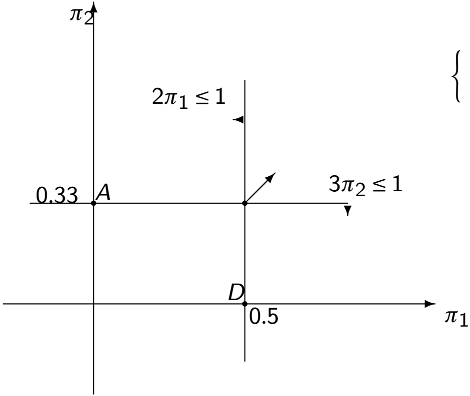
Dual space of CSP: rolls of size 10, items of size 4 and 3



knapsack:

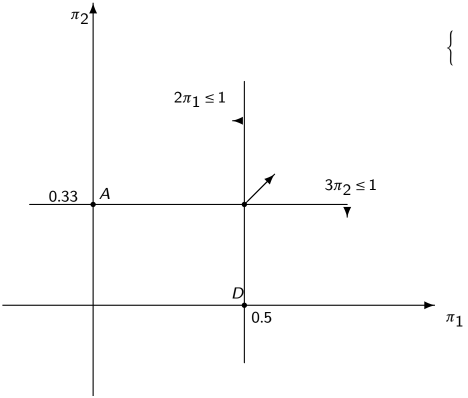
$K = \{(y_1, y_2) : 4y_1 + 3y_2 \leq 10, y_1, y_2 \geq 0 \text{ and integer}\} = \{(2,0),(1,2),(0,3)\}$

Cutting plane algorithm for dual of CSP: starting solution



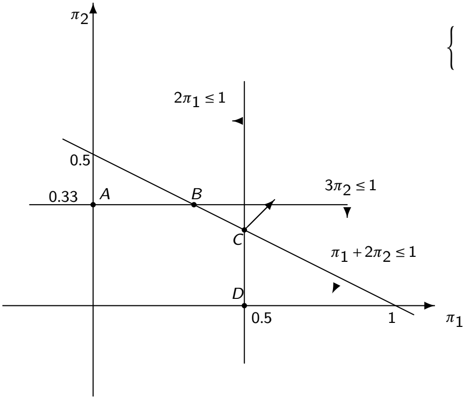
$$\begin{cases} 2\pi_1 & & \leq 1 \\ & 3\pi_2 & \leq 1 \end{cases}$$

Dual space of CSP: first iteration



$$\begin{cases} 2\pi_1 & & \leq 1 \\ & 3\pi_2 & \leq 1 \end{cases}$$

Dual space of CSP: second iteration



$$\begin{cases} 2\pi_1 & & \leq 1 \\ & 3\pi_2 & \leq 1 \\ 1\pi_1 & +2\pi_2 & \leq 1 \end{cases}$$

Dual cuts

$Ax = b$ is column generation model.

$$(P) \quad \begin{array}{ll} \min & cx \\ s.t. & Ax = b \\ & x \geq 0 \end{array} \qquad\qquad (D) \quad \begin{array}{ll} \max & \pi b \\ s.t. & \pi A \leq c \end{array}$$

Adding a set of inequalities to the dual problem, $\pi D \leq d$, we get the extended primal-dual pair:

$$(P^e) \quad \begin{array}{ll} \min & cx + dy \\ s.t. & Ax + Dy = b \\ & x, y \geq 0 \end{array} \qquad\qquad (D^e) \quad \begin{array}{ll} \max & \pi b \\ s.t. & \pi A \leq c \\ & \pi D \leq d \end{array}$$

Usually, restricting the dual $\equiv$ relaxing the primal.
In this case, that does not happen.

Assumption

Assume that we can map any solution $(\overline{x}, \overline{y})$ *of the extended model to a solution* $\overline{\overline{x}}$ *that is valid in the original space, i.e.,* $\overline{\overline{x}} \in \mathscr{X} = \{x : Ax = b, x \geq 0\}$*, and has the same objective value, i.e.,* $c\overline{\overline{x}} = c\overline{x} + d\overline{y}$*.*

Solve the extended model, and eventually recover an optimal solution to the original problem.

Proposition

Under Assumption 1, mapping the optimal solution of the extended model $(\overline{x}^*, \overline{y}^*)$ *gives a solution* $\overline{\overline{x}}^*$ *that is optimal to the original problem.*

Proof: Let z_P^* and $z_{P^e}^*$ be the optimal values of problems P and P^e, respectively. Clearly, $z_{P^e}^* \leq z_P^*$. Let $(\overline{x}^*, \overline{y}^*)$ be the optimal solution of problem P^e. Then, $\overline{\overline{x}}^* \in \mathscr{X}$ and $c\overline{\overline{x}}^* = c\overline{x}^* + d\overline{y}^* = z_{P^e}^* \leq z_P^*$, which means that $\overline{\overline{x}}^*$ is optimal to the original problem. It follows that $z_{P^e}^* = z_P^*$. $\square$

Insight

Dual inequalities may effectively cut portions of the space of the dual problem D ($\pi A \leq c$), but

Corollary

Under Assumption 1, the dual inequalities do not cut all optimal dual solutions of the original problem.

Proof: Let z_D^* and $z_{D^e}^*$ be the optimal values of problems D and D^e, respectively. Suppose that all optimal dual solutions were cut. Then, $z_{D^e}^* < z_D^*$, and, by the strong duality theorem, $z_{P^e}^* < z_P^*$, contradicting the previous Proposition. $\square$

That also happens, if, at the optimum of the extended model, the dual inequality is obeyed with slack, that is, $\pi D < d$.

A family of valid dual cuts

Proposition

For any width w_i, and a set S of item widths, indexed by s, such that $\sum_{s \in S} w_s \leq w_i$, the dual cuts

$$-\pi_i + \sum_{s \in S} \pi_s \quad \leq \quad 0, \quad \forall i, S,$$

are valid inequalities to the space of optimal solutions of the dual of the cutting stock problem.

Proof.

(contradiction): there would be an attractive cutting pattern. □

Primal point of view: an item of size w_i can be cut, and used to fulfill the demand of smaller orders, provided the sum of their widths is $\leq w_i$.

Example

Combining a cutting pattern and a valid dual cut gives a new cutting pattern.

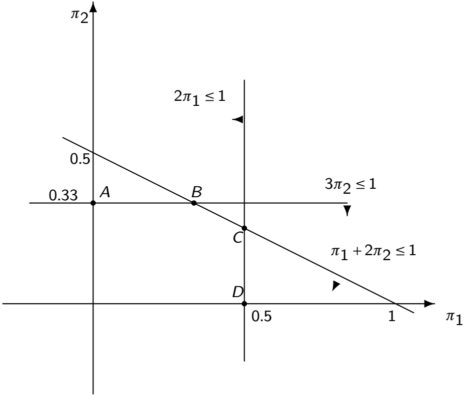
$$
\begin{array}{ccccccccc}
W = 100 & & A_1 & & D_1 & & A_1 & & A_1^{new} \\
\begin{array}{c} 25 \\ 10 \\ 6 \\ 3 \\ 2 \end{array} & & \boxed{\begin{array}{c} 2 \\ 4 \\ 1 \\ 0 \\ 2 \end{array}} & + & \boxed{\begin{array}{c} 0 \\ -1 \\ 1 \\ 1 \\ 0 \end{array}} & --> & \boxed{\begin{array}{c} 2 \\ 4 \\ 1 \\ 0 \\ 2 \end{array}} & + & \boxed{\begin{array}{c} 2 \\ 0 \\ 5 \\ 4 \\ 2 \end{array}} \\
x_j & & 0.3 & & 0.8 & & 0.1 & & 0.2
\end{array}
$$

Implementation issues

- Exponential number of cuts of this family.
- Use only cuts from sets S of small cardinality.
- Sets of size 1 and 2 provide a polynomial number $O(m^2)$ of cuts.

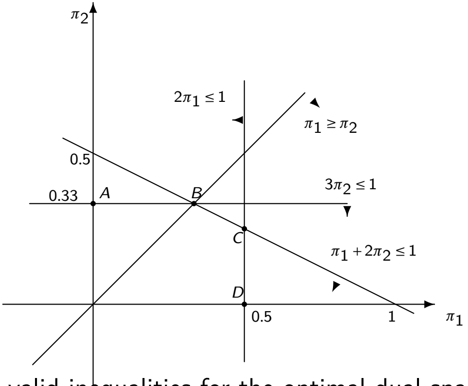
Cuts selected:

- **Cuts of Type 1:** $-\pi_i + \pi_{i+1} \leq 0, \quad i = 1, 2, \ldots, m-1$
- **Cuts of Type 2:** $-\pi_i + \pi_j + \pi_k \leq 0, \quad \forall i, j, k : w_i \geq w_j + w_k$



knapsack:
$K = \{(y_1, y_2) : 4y_1 + 3y_2 \leq 10, y_1, y_2 \geq 0$ and integer$\} = \{(2,0), (1,2), (0,3)\}$

Dual space of CSP with cut $\pi_1 \geq \pi_2$



Dual cuts are valid inequalities for the optimal dual space: $\pi_1 \geq \pi_2$ cuts the dual space but obeys all the dual optimal solutions.

Computational implementation of column generation:

- Add dual cuts to model before starting column generation.
- Add starting solution: as suggested by GG, or any other.
- Proceed as usual.

	dual cuts							GG initial solution					
100	d_1	d_2	d_3	d_4	d_5	d_6	d_7	x_1	x_2	x_3	x_4	x_5	
25	-1				-1			4					$\geq d_{25}$
10	1	-1			1	-1			10				$\geq d_{10}$
6		1	-1		1	1	-1			16			$\geq d_6$
3			1	-1		1	1				33		$\geq d_3$
2				1			1					50	$\geq d_2$
min	0	0	0	0	0	0	0	1	1	1	1	1	

Note: every column is a dual constraint.

Computational experiments: instances

Binpacking instances (OR-Library, Beasley'90)

t class: instances with an integer optimum solution in which all bins have three items, which fulfill exactly the capacity of the bin (triplet instances).
Bin capacity is $W = 100$, and item sizes vary between 25.0 and 49.9.
No dual cuts of Type II, because no item can be divided into two smaller items.
Larger instances were tested: the t501–instances, with 501 items.

Cutting stock instances (as in Vance'93)

Rolls with widths of 100, 120 or 150, a number of items equal to 200 or 500, with randomly generated real values drawn from a uniform distribution u(1,100).
Existence of small items leads to an explosion in the number of feasible columns.
The more difficult instances are those with larger roll widths and larger number of items, because they have more feasible columns.

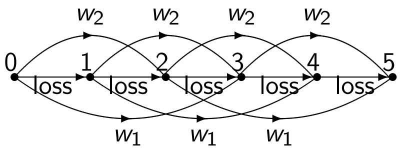
Computational results: summary

Binpacking instances (OR-Library, Beasley'90)
Reduction in number of columns: 43.0 % (from 263.3 to 150.0).
Reduction in computational time: 20.1 % faster.
Reduction in degenerate pivots: percentage falls from 9.3% to 5.4%.

Cutting stock instances (as in Vance'93)
Reduction in number of columns: 75.9 % (from 5309.1 to 1281.6).
Reduction in computational time: 78.2 % (4.5 times faster).
Reduction in degenerate pivots: percentage falls from 39.8% to 8.5%.

Dual cuts in the arc-flow model

Dual cuts are cycles in the space of the original variables.



Exactly one arc (the largest) is traversed in the direction opposite to its orientation.
Combining a cycle and a path produces a new path.
For each arc with negative flow (direction opposite to its orientation), there is always one (or plus) arc(s) with positive flow(s) with larger value: the net sum of flows in arcs that correspond to a given width is positive (equal to the demand).

Dual-optimal inequalities and deep dual-optimal inequalities

Instead of just referring to "dual cuts", at some points, we will make a distinction between the two different classes:

Dual-optimal inequalities:
All dual optimal solutions are preserved (as in the dual cuts for Gilmore-Gomory model for the CSP).

Deep dual-optimal inequalities:
One may even effectively cut a subset of dual optimal solutions, if, at least, one dual optimal solution is preserved.
One optimal dual solution is sufficient to drive the process to find the optimum.

Perturbing dual-optimal inequalities

Consider now the set of dual-optimal inequalities perturbed by an ε : $\pi D \leq d + \varepsilon$:

$$(P^{e'}) \quad \begin{array}{ll} \min & cx + (d+\varepsilon)y \\ s.t. & Ax + Dy = b \\ & x, y \geq 0 \end{array} \qquad\qquad (D^{e'}) \quad \begin{array}{ll} \max & \pi b \\ s.t. & \pi A \leq c \\ & \pi D \leq d + \varepsilon \end{array}$$

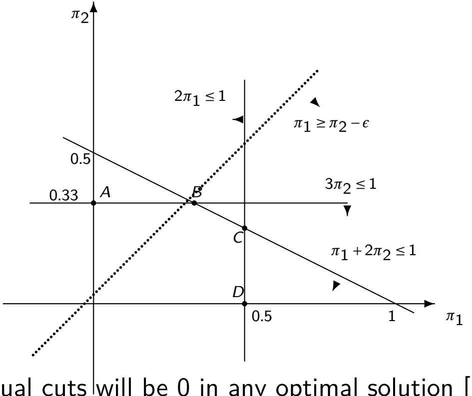
Proposition

Let $\pi D \leq d$ be a set of dual-optimal inequalities, and (x^*, y^*) an optimal solution for $P^{e'}$. Then, $y^* = 0$ and x^* is an optimal solution for P.

Proof: All dual-optimal solutions obey $\pi D \leq d$ and have slack in $\pi D \leq d + \varepsilon$. By complementary slackness, the corresponding primal variables $y^* = 0$. $\square$

(Ben Amor,Desrosiers,VC'2006)

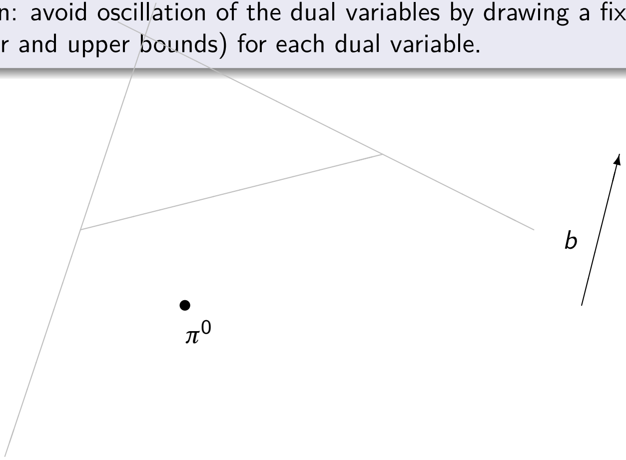
Dual space of CSP with cut $\pi_1 \geq \pi_2$ perturbed by ϵ



Columns of dual cuts will be 0 in any optimal solution [Ben Amor, Desrosiers, VC, 2006].

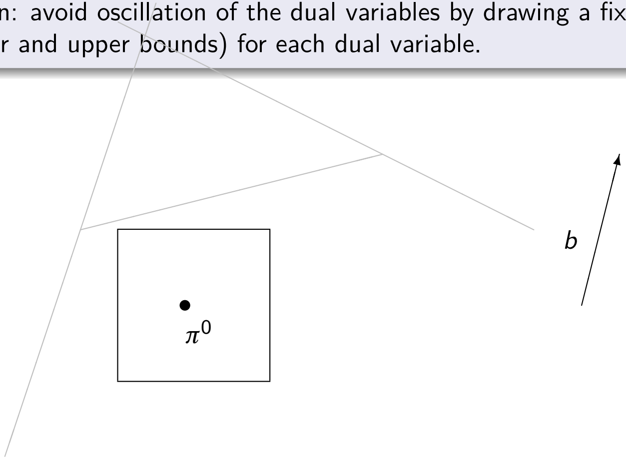
Boxstep method (Marsten et al. 1975)

Motivation: avoid oscillation of the dual variables by drawing a fixed-size Box (lower and upper bounds) for each dual variable.



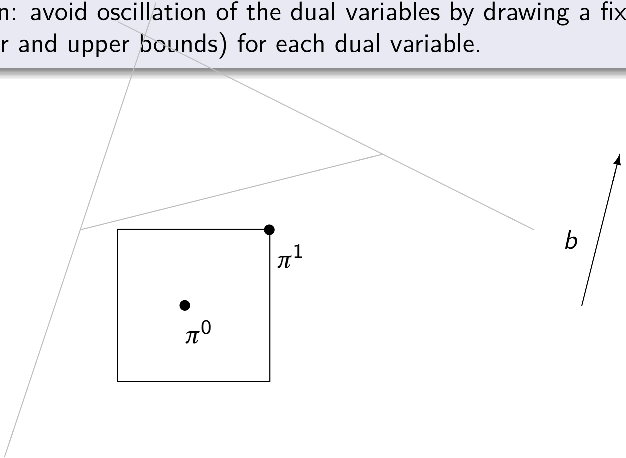
Boxstep method (Marsten et al. 1975)

Motivation: avoid oscillation of the dual variables by drawing a fixed-size Box (lower and upper bounds) for each dual variable.



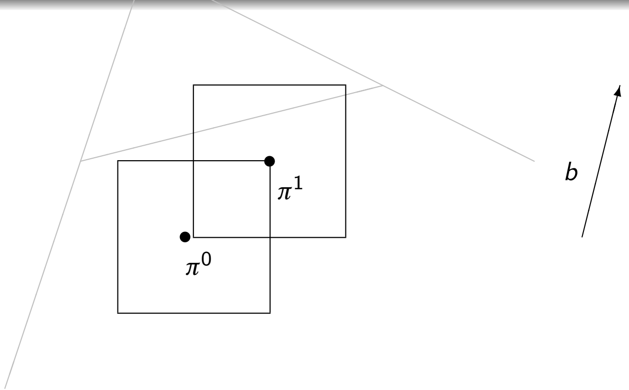
Boxstep method (Marsten et al. 1975)

Motivation: avoid oscillation of the dual variables by drawing a fixed-size Box (lower and upper bounds) for each dual variable.



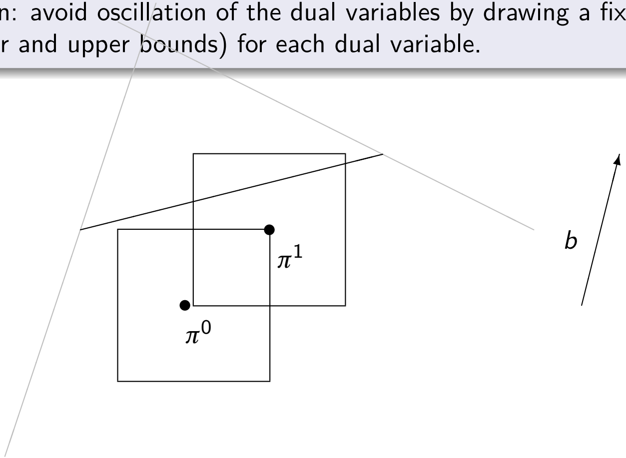
Boxstep method (Marsten et al. 1975)

Motivation: avoid oscillation of the dual variables by drawing a fixed-size Box (lower and upper bounds) for each dual variable.



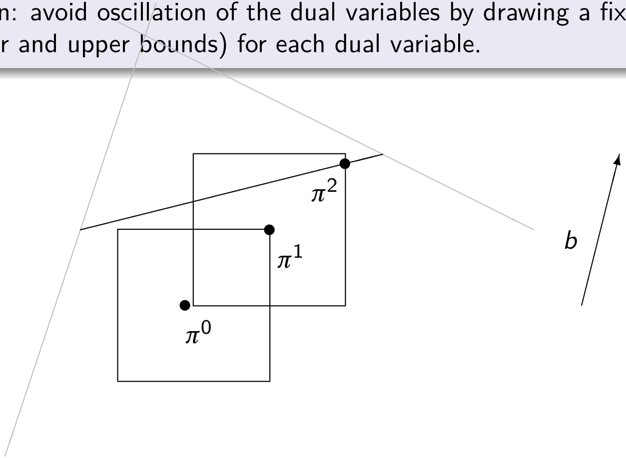
Boxstep method (Marsten et al. 1975)

Motivation: avoid oscillation of the dual variables by drawing a fixed-size Box (lower and upper bounds) for each dual variable.



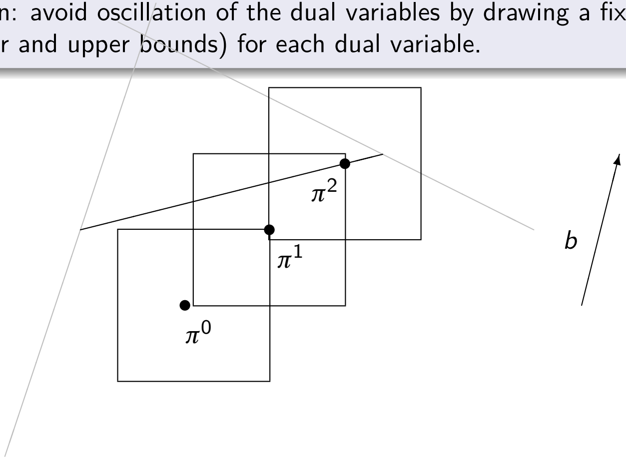
Boxstep method (Marsten et al. 1975)

Motivation: avoid oscillation of the dual variables by drawing a fixed-size Box (lower and upper bounds) for each dual variable.



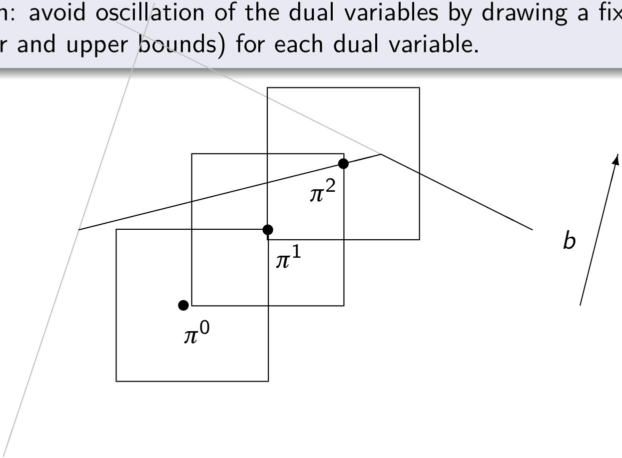
Boxstep method (Marsten et al. 1975)

Motivation: avoid oscillation of the dual variables by drawing a fixed-size Box (lower and upper bounds) for each dual variable.



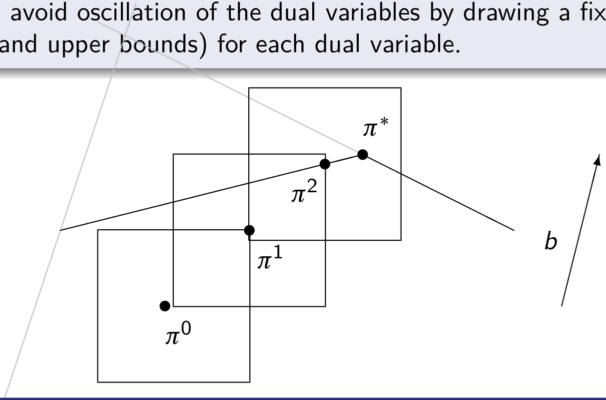
Boxstep method (Marsten et al. 1975)

Motivation: avoid oscillation of the dual variables by drawing a fixed-size Box (lower and upper bounds) for each dual variable.



Boxstep method (Marsten et al. 1975)

Motivation: avoid oscillation of the dual variables by drawing a fixed-size Box (lower and upper bounds) for each dual variable.



Solution process:

- If the optimal dual solution is strictly inside the box, then it is an optimal solution to the original problem.
- If any dual variable lies in the boundary of the box (its value equals the lower or the upper bound), the box is re-centered for the next iteration.

$Ax = b$ is original column generation model:

$$(P)\quad \begin{array}{ll} \min & cx \\ s.t. & Ax = b \\ & x \geq 0 \end{array} \qquad\qquad (D)\quad \begin{array}{ll} \max & \pi b \\ s.t. & \pi A \leq c \end{array}$$

Modified column generation model:

$$(P^b)\quad \begin{array}{ll} \min & cx - \delta^- u^- + \delta^+ u^+ \\ s.t. & Ax - u^- + u^+ = b \\ & x, u^-, u^+ \geq 0 \end{array} \qquad\qquad (D^b)\quad \begin{array}{ll} \max & \pi b \\ s.t. & \pi A \leq c \\ & -\pi \leq -\delta^- \\ & \pi \leq \delta^+ \end{array}$$

Bounds in dual variables define a Box: $\delta^- \leq \pi \leq \delta^+$.

Analysis of Boxstep method (Marsten et al. 1975)

Computational implementation of column generation:

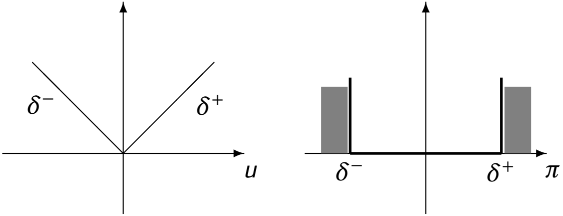
similar to standard, but the restricted master problems has slack and surplus variables with a cost (penalty).

$$(P^b) \quad \begin{array}{ll} \min & cx - \delta^- u^- + \delta^+ u^+ \\ s.t. & Ax - u^- + u^+ = b \\ & x, u^-, u^+ \geq 0 \end{array} \qquad (D^b) \quad \begin{array}{ll} \max & \pi b \\ s.t. & \pi A \leq c \\ & -\pi \leq -\delta^- \\ & \pi \leq \delta^+ \end{array}$$

- Primal view: penalize deviation from valid solution.
- Dual view: Set Box (Trust region) for dual variables.

- small Box may lead to many iterations.
- standard column generation is Boxstep method with infinite dimension box.

Stabilizing terms: penalty / trust region



- larger penalty in primal, wider box in dual
- smaller penalty in primal, thinner box in dual

Stabilization (du Merle *et al.* 99)

$Ax = b$ is original column generation model:

$$(P) \quad \begin{array}{ll} \min & cx \\ s.t. & Ax = b \\ & x \geq 0 \end{array} \qquad\qquad (D) \quad \begin{array}{ll} \max & \pi b \\ s.t. & \pi A \leq c \end{array}$$

Stabilized column generation model:

$$(P^s) \quad \begin{array}{ll} \min & cx - \delta^- u^- + \delta^+ u^+ \\ s.t. & Ax - u^- + u^+ = b \\ & u^- \leq \epsilon^- \\ & u^+ \leq \epsilon^+ \\ & x, u^-, u^+ \geq 0 \end{array} \qquad (D^s) \quad \begin{array}{ll} \max & \pi b - \epsilon^- w^- - \epsilon^+ w^+ \\ s.t. & \pi A \leq c \\ & -\pi - w^- \leq -\delta^- \\ & \pi - w^+ \leq \delta^+ \\ & w^-, w^+ \geq 0 \end{array}$$

Dual variables may be outside the Box: $\delta^- \leq \pi \leq \delta^+$, but there is a penalty.

Analysis of Stabilization method (du Merle *et al.* 99)

Computational implementation of column generation:

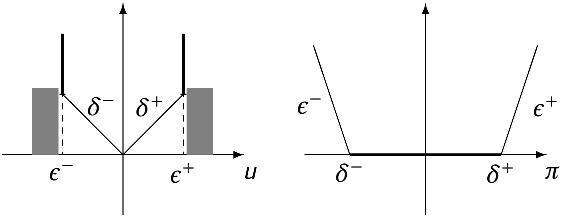
now, the slack and surplus variables with a cost also have bounds.

$$(P^s) \quad \begin{array}{ll} \min & cx - \delta^- u^- + \delta^+ u^+ \\ s.t. & Ax - u^- + u^+ = b \\ & u^- \le \epsilon^- \\ & u^+ \le \epsilon^+ \\ & x, u^-, u^+ \ge 0 \end{array} \qquad (D^s) \quad \begin{array}{ll} \max & \pi b - \epsilon^- w^- - \epsilon^+ w^+ \\ s.t. & \pi A \le c \\ & -\pi - w^- \le -\delta^- \\ & \pi - w^+ \le \delta^+ \\ & w^-, w^+ \ge 0 \end{array}$$

- Primal view: penalize deviation from valid solution (now deviation is limited).
- Last two groups of dual constraints: $\delta^- - w^- \le \quad \pi \quad \le \delta^+ + w^+$.
- Dual view: dual variables outside a pre-defined box are penalized (if π is outside the interval $[\delta^-, \delta^+]$, the variables w^- or w^+ take a positive value, penalizing the objective function).

- size of Box is not so critical, because solutions outside Box are allowed.

Stabilizing terms: penalty / trust region



Getting to the optimal solution

Adjustment of penalties at a given restricted master problem:

- If any π is on the border of the interval, the penalty is not sufficiently large, and the optimal solution of the stabilized problem may not be valid for the original problem.
- Adjust penalties !
- The algorithm needs appropriate strategies for the adjustment of the penalties so that the optimal solution is found rapidly.

At the optimal solution:

- Complementary Slackness Theorem: if the optimal value of π is strictly inside the interval $[\delta^-, \delta^+]$, the constraints have slack and the corresponding dual variables are null, that is, $u^- = u^+ = 0$, which implies that $Ax = b$ (valid for original model).
- The same happens with $\epsilon^- = \epsilon^+ = 0$.

When an optimal dual solution is known

Given an optimal dual solution $\overline{\pi}^*$ for $\overline{D}$ and a vector of scalars $\Delta > \mathbf{0} \in \mathbb{R}^m$, use the stabilized pair of primal and dual problems:

$$\begin{array}{c|c} \begin{array}{c} v(\overline{P}(\overline{\pi}^*)) := \min \mathbf{c}^T\mathbf{x} - (\overline{\pi}^* - \Delta)^T\mathbf{y}_1 + (\overline{\pi}^* + \Delta)^T\mathbf{y}_2 \\ A\mathbf{x} - \mathbf{y}_1 + \mathbf{y}_2 = \mathbf{b} \\ \mathbf{x} \geq \mathbf{0},\ \mathbf{y}_1 \geq \mathbf{0},\ \mathbf{y}_2 \geq \mathbf{0} \end{array} & \begin{array}{c} v(\overline{D}(\overline{\pi}^*)) := \max \mathbf{b}^T\pi \\ A^T\pi \leq \mathbf{c} \\ \overline{\pi}^* - \Delta \leq \pi \leq \overline{\pi}^* + \Delta. \end{array} \end{array}$$

Proposition

Let $E^T\pi \leq \mathbf{e}$ be a set of deep dual-optimal inequalities and $(\overline{\mathbf{x}}^*, \mathbf{y}_1^*, \mathbf{y}_2^*)$ be an optimal solution for $\overline{P}(\overline{\pi}^*)$. Then, $\mathbf{y}_1^* = \mathbf{y}_2^* = \mathbf{0}$ and $\overline{\mathbf{x}}^*$ is an optimal solution for P.

Proof: This result was proved in a previous session. $\square$

When an optimal dual solution is known: example

Proposition

Consider a CSP instance with no loss at optimality. Then, $\pi_i^* = \frac{w_i}{W}, i \in I$ is an optimal dual solution.

Proof: All dual constraints are obeyed. The dual objective function reaches the optimal value $\sum_{i=1}^{m} b_i w_i / W$. Therefore, this dual solution is optimal. $\square$

Computational results for binpacking triplet instances (OR-Library, Beasley'90)

- Reduction in number of columns: 90.2 % (from 124.2 to 12.2).
- Size of box $\Delta = 10^{-2}$

(Ben Amor,Desrosiers,VC'2006)

A word about a nice result

Different primal models with equally constrained dual spaces take the same number of iterations.

Experiment 1
BinaryCSP model (disaggregated demand): there is a constraint for each item of the same size (demand is equal to 1).
Solution of BinaryCSP takes more iterations than CSP.
Experiment 2
Add dual constrains saying that dual variables of items of the same size should be equal.
Solution of BinaryCSP takes approximately the same number of iterations as CSP.

(Ben Amor,Desrosiers,VC'2006)

Stabilization
General framework, which is not problem dependent.
Adjustments of stability center may be needed.

Dual cuts
Derivation relies on characterization of the space of dual optimal solutions.
Problem dependent, not easy to derive.
Valid through entire column generation process.

Combination
Using dual cuts amounts to solving an alternative primal model (equally strong) with a more restricted dual space.
Stabilization and dual cuts can be combined.

Concluding remarks

- Strength of models is of crucial importance.
- Dual cuts make column generation faster keeping models strong.
- Restriction of dual space may be an important factor for faster convergence.

Pre-processing

- Arc elimination
- Initial Primal Solutions

Arc elimination

Using a feasible primal integer solution and a feasible dual solution to the relaxation of the problem to fix a *path variable* to 0:

Lemma

- *Given* (IP): $\min\{cx : Ax = b, x = (x_i)_{i \in \{1,\ldots,n\}} \in \{0,1\}^n\}$.
- π : *feasible dual solution of the LP-relaxation of IP, with value* πb.
- U : *upper bound for IP, given by a feasible primal integer solution.*
- *If, for some* $p \in \{1,\ldots,n\}$, $\pi b + (c_p - \pi A_p) > U$, *then* $x_p = 0$ *in all optimal solutions of IP.*

- In the integer problem (IP), x_p is a binary variable.
- Idea of proof: the lagrangean lower bound of the problem with $x_p = 1$ is above the upper bound.

Arc elimination (cont.)

Proof:

- Consider modified problem where x_p is fixed to 1 and $x_i \geq 0, \forall i \in \{1,\dots,n\} \setminus \{p\}$:

$$\min\{cx + c_p : Ax = b - A_p, x_i \geq 0, \forall i \in \{1,\dots,n\} \setminus \{p\}, x_p = 0\}.$$

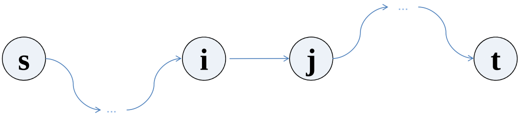
- Dual of modified problem: $\max\{(b - A_p)\pi : \pi A \leq c\}$
- lagrangean of modified problem:

$$\begin{aligned} z_{LR}(\pi) &= c_p + \min_{x \setminus x_p}\{cx + \pi((b - A_p) - Ax)\} = \\ &= (c_p - \pi A_p) + \pi b + \min_{x \setminus x_p}\{(c - \pi A)x)\} \end{aligned}$$

- in dual feasible solution, $\pi A \leq c$.
- given a feasible dual solution π, the lagrangean lower bound of modified problem $> U$.

Arc elimination (cont.)

Using a feasible primal integer solution and a feasible dual solution to the relaxation of the problem to fix an *arc variable* to 0:



Variable fixing

- Consider the paths $p = (s, \ldots, i, j, \ldots, t)$ that contain arc (i,j).
- If $\pi b + \min_{p:(i,j) \in p}(c_p - \pi A_p) > U$, then arc (i,j) can be eliminated.

Initial Primal solutions

Starting solutions for column generation:

- Polynomial heuristics: First Fit Decreasing and Best Fit Decreasing
- Pseudo-polynomial heuristics

First Fit Decreasing (FFD) and Best Fit Decreasing (BFD) heuristics

FFD : largest unplaced item is assigned to the bin with smallest index already used with sufficient remaining capacity; if there is none, a new bin is started.

BFD : largest unplaced item is assigned to the bin with smallest remaining capacity, but still sufficient to accommodate the item; if there is none, a new bin is started.

FFD and BFD have absolute performance ratios of 3/2, *i.e.*, $z_H \leq 3/2\ z^*$, where z^* is the value of the optimum (Simchi-Levi'94).

Absolute performance ratios:

First-Fit Decreasing – $\frac{3}{2}$.

Best-Fit Decreasing – $\frac{3}{2}$.

Asymptotic performance ratios:

First-Fit Decreasing – $\frac{11}{9}$.

Best-Fit Decreasing – $\frac{11}{9}$.

FFD: example of asymptotic performance ratio of 11/9

Asymptotic performance ratio: happens even in large instances.
Example: optimal solution uses $9N$ bins, heuristic solution uses $11N$ bins.

51	26	23	
51	26	23	
51	26	23	
51	26	23	
51	26	23	
51	26	23	
27	27	23	23
27	27	23	23
27	27	23	23

$$z^* = 9N$$

51	27			
51	27			
51	27			
51	27			
51	27			
51	27			
26	26	26		
26	26	26		
23	23	23	23	
23	23	23	23	
23	23	23	23	

$$z_{FFD} = 11N$$

Absolute performance ratio: only in instances with a small number of bins.

Pseudo-polynomial heuristics

Greedy (myopic) heuristic, based on iterative solution of knapsack problems:

Build list with all items
While (there are items in the list) do
    solve knapsack problem
    remove items in the solution from the list
    (repeat removal, if there are multiple copies of all items)
End While

- Computation time is not significant in the column generation framework.
- Usually provides good starting solutions, with, at least, some very good cutting patterns.
- Last patterns may be very poor.

Pseudo-polynomial heuristics (two implementations)

Vanderbeck'99:

- among the solutions with maximum capacity usage, $\sum_i w_i y_i$, choose the one that is lexicographically smaller when considering a solution vector where the items are ordered by non-increasing sizes $(w_1 \geq w_2 \geq \ldots \geq w_m)$.

VC'05:

- use weights $w_i =$ item sizes and profits $p_i = (1 - (j-1)/n)\ w_i, \forall i$, to favor choice of solutions with larger items, leaving the smaller items, which should be easier to combine, to subsequent iterations
- preferable to solving a subset-sum problem, $\max\{\sum_i w_i y_i : \sum_i w_i y_i \leq W, y_i \geq 0 \text{ and integer}, \forall i\}$, which is, in practice, difficult to solve [Martello, Toth'90].

They provide much better starting solutions than FFD or BFD.

Master Problem

- Column elimination
- Constraint aggregation
- Multiple columns at each iteration
- Stabilization [Part V]
- Dual cuts (including covering vs. partitioning constraints) [Part V]

Column elimination

Most probably many columns of the RMP will not be used in the optimal solution. Periodically,

- purge columns with reduced cost above a pre-defined threshold, or
- purge columns with zero value for a predefined number of iterations.

Constraint aggregation

CSP [Alves, VC'07]

- Basic algorithm:
  - pick two similar item sizes and aggregate demands (*e.g.*, use larger item size).
  - less constraints and smaller subproblem
  - solution of aggregated model may be "sub-optimal"
  - at the end, disaggregate to check if re-optimization is necessary.
- Also more effective *n*-phase algorithm.

VR & CS [Elhallaoui, Villeneuve, Soumis, Desaulniers'05]

- Vehicle routing and crew scheduling:
- Aggregation according to an equivalence relation that changes dynamically over time.
- Shortest path problem used to recover the non-aggregated dual information
- Master problem time reduced by a factor of 8.

Multiple columns at each iteration

- when problem has several subproblems, use the dual information of the RMP to generate columns from all the subproblems, and insert them all in the RMP, before re-optimizing.
- if possible, pick not only the optimal solution of subproblem, but also $2^{nd}, \ldots, k^{th}$ best solutions (there are cases with 3 to 10 columns).
- furthermore, use heuristics to find columns that are orthogonal together with the most attractive (*i.e.*, that may combine better to form a solution).

Subproblem

- Heuristic pricing
- State space reduction

Heuristic pricing

- Subproblem may be a problem difficult to solve practically.
- Try to get close to the optimal solution using heuristics.
- Only resort to solving subproblem optimally when no more attractive solutions are found.

Example

- use different heuristic algorithms along the process.

Space state reduction

- Temporarily reduce the burden of the dynamic programming subproblem,
- Then resort to solving subproblem optimally.

Example

- in problems with time constraints, start with a less precise definition of time,
- start with a subset of clients or items.

Branch-and-bound

- Early branching
- Upper bounds

Early branching

- For problems with integer cost coefficients, c_j, $\forall j$, given
- $\bar{z}$: value of the current solution of column generation process, and
- LB : a lower bound,
- if $\lceil \bar{z} \rceil = \lceil LB \rceil$,
- column-generation process can be cut off to reduce the tail.

- One can also terminate heuristically earlier.

Upper bounds

- use depth first search (possibly making a single dive) to try to get a good incumbent solution, which may help fathoming nodes later during the full exploitation of the tree.
- use heuristics (more or less elaborate) at each node of the tree.

Other issues

- G. Desaulniers, J. Desrosiers, M. Salomon, Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems, Cahiers GERAD G-99-36, and in Essays and Surveys in Metaheuristics, C. Ribeiro and P. Hansen (eds.), Kluwer Academic Publishers, 309-324, 2002.

Concluding remarks

- Practical issues may reduce computational time significantly.
- They are problem dependent, and have to be tailored.

Bibliography - I

- R.K. Ahuja, T. Magnanti and J.B. Orlin, "Network flows: Theory, Algorithms and Applications". Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1993.
- S. Martello and P. Toth "Knapsack Problems: Algorithms and Computer Implementations", Wiley, New York, 1990.
- G.L. Nemhauser and L.A. Wolsey, "Integer and Combinatorial Optimization", Wiley Interscience, 1988.
- Column Generation, G. Desaulniers, J. Desrosiers and M.M. Solomon (eds.), Springer, 2005.

Bibliography - II

- C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh and P.H. Vance, "Branch-and-Price: Column Generation for Solving Huge Integer Programs", Operations Research 46(3), 316-329, 1998.
- H. Ben Amor, J. Desrosiers and A. Frangioni "On the Choice of Explicit Stabilizing Terms in Column Generation", Discrete Applied Mathematics, to appear, 2009.
- H. Ben Amor and J.M. Valério de Carvalho, "Cutting Stock Problems", in "Column Generation", G. Desaulniers, J. Desrosiers and M.M. Solomon (eds.), 131-162, Springer, 2005.
- O. Briant, C. Lemaréchal, Ph. Meurdesoif, S. Michel, N. Perrot, and F. Vanderbeck, "Comparison of bundle and classical column generation", Mathematical Programming 113(2), 299-344, 2008.
- S. Fekete and J. Schepers, "New classes of fast lower bounds for bin packing problems". Mathematical Programming 91, 11-31, 2001.
- A. Frangioni "About Lagrangian Methods in Integer Optimization" Annals of Operations Research 139, 163-193, 2005.
- A. Frangioni "Generalized Bundle Methods" SIAM Journal on Optimization 13(1), 117-156, 2002.

- O. du Merle, D. Villeneuve, J. Desrosiers and P. Hansen, "Stabilized column generation", Discrete Mathematics 194, 229-297, 1999.
- F. Vanderbeck, "Exact algorithm for minimising the number of setups in the one-dimensional cutting stock problem". Operations Research 48(6), 915-926, 2000.
- F. Clautiaux, C. Alves and J.M. Valério de Carvalho, "A survey of dual-feasible and superadditive functions", to appear in Annals of Operations Research, 2009.
- C. Alves and J.M. Valério de Carvalho, "A branch-and-price-and-cut algorithm for the pattern minimization problem", RAIRO Operations Research, 42, 435-453, 2008.
- C. Alves and J.M. Valério de Carvalho, "Stabilized Branch-and-Price -and-Cut Algorithm for the Multiple Length Cutting Stock Problem", Computers and Operations Research 35(4), 1315-1328, 2008.
- F. Alvelos and J.M. Valério de Carvalho, "An extended model and a column generation algorithm for the planar multicommodity flow problem", Networks 50(1), 3-16, 2007.
- M. Lopes and J.M. Valério de Carvalho, "A branch-and-price algorithm for scheduling parallel machines with sequence dependent setup times", European Journal of Operational Research 176(3), 1508-1527, 2007.

Bibliography - IV

- C. Alves and J.M. Valério de Carvalho, "Accelerating Column Generation for Variable Sized Bin-Packing Problems", European Journal of Operational Research 183(3), 1333-1352, 2007.
- H. Ben Amor, J. Desrosiers and J.M. Valério de Carvalho, "Dual-optimal Inequalities for Stabilized Column Generation", Operations Research 54(3), 454–463, 2006.
- H. Ben Amor and J.M. Valério de Carvalho, "Cutting Stock Problems", in Column Generation, Guy Desaulniers, Jacques Desrosiers, and Marius M. Solomon (eds.), Springer, 2005.
- J.M. Valério de Carvalho, "Using extra dual cuts to accelerate convergence in column generation", INFORMS Journal on Computing 17(2), 175-182, 2005.
- J.M. Valério de Carvalho, "LP Models for Bin-Packing and Cutting Stock Problems", European Journal of Operational Research 141(2), 253-273, 2002.
- J.M. Valério de Carvalho, "Exact solution of bin-packing problems using column generation and branch-and-bound", Annals of Operations Research 86, 629-659, 1999.