Available online at www.sciencedirect.com
ScienceDirect
Operations Research Letters 35 (2007) 660–668
Operations Research Letters
www.elsevier.com/locate/orl

Interior point stabilization for column generation

Louis-Martin Rousseau[a], Michel Gendreau[b,*], Dominique Feillet[c]

[a]*Cirrelt, École Polytechnique de Montréal, Canada*
[b]*Cirrelt, Université de Montréal, C.P. 6128, Succursale Centre-ville, Montréal, Canada H3C 3J7*
[c]*Laboratoire d'Informatique d'Avignon, Canada*

Received 3 July 2003; accepted 14 November 2006
Available online 20 December 2006

Abstract

Interior point stabilization is an acceleration method for column generation algorithms. It addresses degeneracy and convergence difficulties by selecting a dual solution inside the optimal space rather than retrieving an extreme point. The method is applied to the case of the vehicle routing problem with time windows.

Keywords: Column generation; Stabilization; Vehicle routing

0. Introduction

Column generation was introduced by Dantzig and Wolfe [3] to solve linear programs with decomposable structures. It has been applied to many problems with success and has become a leading optimization technique to solve routing and scheduling problems [5,1]. However, some column generation methods often show very slow convergence partly due to heavy degeneracy problems. One of these problems arises when multiple dual solutions are associated with each primal solution. Choosing the dual solution then becomes a crucial part of the column generation algorithm as the subproblem solution often heavily depends on the dual values. This observation led to the introduction of several stabilization methods, which attempted to accelerate convergence by implementing diverse mechanisms to control the selection of the dual solution. The method proposed here tries to achieve the same goal by using a different approach.

The next section, which gives a brief overview of the column generation framework, is followed by a short description of some existing stabilization methods that we will use for comparison purposes. Section 3 introduces interior point stabilization and results are reported in Section 4.

Future work on this technique involves its application to different problems in order to clearly assess in which contexts it can be useful.

* Corresponding author. Tel.: +1 514 3437435; fax: +1 514 3437121.
E-mail address: michelg@crt.umontreal.ca (M. Gendreau).

0167-6377/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.orl.2006.11.004

1. Column generation

Column generation is a general framework that can be applied to numerous problems. However, since an example is often useful to give a clear explanation, all results in this paper will be presented with respect to the vehicle routing problem with time windows (VRPTW). The description of IPS is done without loss of generality nor restrictions to deal with this particular problem. Although it cannot be said that the VRPTW is representative of all domain applications solved by column generation, it is sufficiently degenerate and unstable to make it a good test case for the proposed method. The VRP can be described as follows: given a set of customers, a set of vehicles, and a depot, find a set of routes of minimal length, starting and ending at the depot, such that each customer is visited by exactly one vehicle. Each customer having a specific demand, there are usually capacity constraints on the load that can be carried by a vehicle. In addition, there is a maximum amount of time that can be spent on the road. The time window variant of the problem (VRPTW) imposes the additional constraint that each customer must be visited during a specified time interval. The vehicle can wait in case of early arrival, but late arrival is not permitted.

In the first application of column generation to the field of vehicle routing problems with time windows, presented by Desrochers et al. [4], the basic idea was to decompose the problem into sets of customers visited by the same vehicle (routes) and to select the optimal set of routes between all possible ones. Letting r be a feasible route in the original graph (which contains N customers), R be the set of all possible routes r, c_r be the cost of visiting all the customers in r, $A = (a_{ir})$ be a Boolean matrix with coefficient $a_{ir} = 1$ if a particular customer (denoted by index $i \in \{1 \dots N\}$) is visited in route r, and x_r be a Boolean variable specifying whether the route r is chosen ($x_r = 1$) or not ($x_r = 0$), the set partitioning formulation is defined as (S)

$$\min \quad \sum_{r \in R} c_r x_r \tag{1}$$

$$\text{s.t.} \quad \sum_{r \in R} a_{ir} x_r = 1 \quad \forall i \in \{1 \dots N\}, \tag{2}$$

$$x \in \{0, 1\}^{|R|}. \tag{3}$$

This formulation, however, raises some issues. Firstly, since it is impractical to construct and to store the set R because of its very large size, it is usual to work with a partial set R' that is enriched iteratively by solving a subproblem. Secondly, the linear relaxation of the set partitioning formulation (S_{LP}) allows negative dual values which can be problematic for the subproblem (a negative dual means there is a negative marginal cost to visit a node…). A first stabilization approach is thus to use the following relaxed set covering formulation as a master problem (M):

$$\min \quad \sum_{r \in R'} c_r x_r \tag{4}$$

$$\text{s.t.} \quad \sum_{r \in R'} a_{ir} x_r \geqslant 1 \quad \forall i \in \{1 \dots N\}, \tag{5}$$

$$x \geqslant 0. \tag{6}$$

To enrich R', it is necessary to find new routes that offer a better way to visit the customers, that is, routes with a negative reduced cost. The reduced cost of a route is computed by replacing the cost of an arc (the distance between two customers) d_{ij} by the reduced cost of that arc $c_{ij} = d_{ij} - \lambda_i$, where λ_i is the dual value associated with the covering constraint (5) of customer i. The dual value associated with a customer can be interpreted as the marginal cost of visiting that customer in the current optimal solution (given R'). The objective of the subproblem is then the identification of a negative reduced cost path, i.e., a path for which the sum of the traveled distances is smaller than the sum of the marginal costs (dual values). Such a path represents a new and better way to visit the customers it serves.

The optimal solution of the unrestricted master ((M) defined over R) has been identified when there exists no more negative reduced cost path. This solution can however be fractional. If this is the case, it is necessary to start a branching scheme in order to identify an integer solution.

2. Stability problems

Column generation frameworks depend heavily on marginal costs to guide the search at the subproblem level. In some cases it is possible that, during the first iterations, the marginal costs associated with each customer are not appropriately estimated by the dual

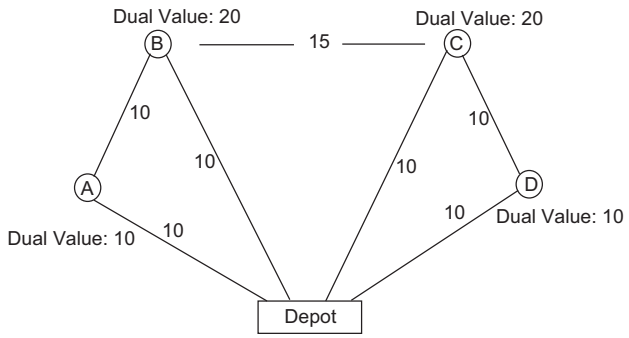

Fig. 1. Path depot–B–C–depot has a negative reduced cost of −5.

values. For instance, it is possible that in some routes some customers pick up most of the total dual values. If this is the case (as illustrated in Fig. 1), then in the subproblem a path that visits each of those overweighted customers (B, C) will be considered a good route (with reduced cost of −5), even though it is not the case (it is unlikely to be selected in an optimum IP solution). With a more realistic distribution of dual values, all nodes would have been given a value of 15 and no more reduced cost path would have been found, thus saving the need for a last iteration.

In this example, the unwanted behavior occurs because S_{LP} is degenerate and thus its dual has an infinite number of optimal solutions. The bases of these primal solutions to S_{LP} all exhibit a common set of strictly positive variables, but different sets of null variables. To each of these equivalent primal optimal solutions is associated a different dual solution. The standard function that returns dual values in the LP codes returns an extreme point of the dual polyhedron. Extreme solutions are characterized by very large values for some marginal costs while others are at zero. Much better approximations of the optimal marginal cost for the unrestricted master would be obtained if the dual variables would take values in the center (or at least the interior) of the optimal dual polyhedron.

To prevent dual variables from taking extreme values, the general strategy proposed in the literature is to try to limit the distance traveled by the dual variables in the dual space from one iteration to another. A few techniques are available. A first technique consists in defining a box around the previous value of each dual variable and modifying the master problem so that the feasible dual space is limited to the area defined by these boxes. A second technique is to adapt the master problem so that the distance separating a dual solution from the previous optimal dual solution is linearly penalized. These two techniques were, respectively, proposed by Marsten et al. [13] and by Kim et al. [11] for improving the convergence of Kelley's well-known cutting plane algorithm [10].

du Merle et al. [7] have proposed to stabilize column generation with a method that combines these two concepts. They impose soft limits on the dual values λ associated with the constraints (5) of (M). These limits, defined by (d^-, d^+), are first given as parameters to the model and then automatically updated during the resolution. The λ variables can take values outside the given limits (when $\omega^- > 0$ or $\omega^+ > 0$), but the objective is then penalized by $(\varepsilon^-, \varepsilon^+)$. The dual form of the master then becomes:

$$\max \quad \sum_{i\in\{1\ldots N\}} \lambda_i - \varepsilon_i^- \omega_i^- - \varepsilon_i^+ \omega_i^+ \tag{7}$$

$$\text{s.t.} \quad \sum_{i\in\{1\ldots N\}} \lambda_i a_{ir} \leqslant c_r \quad \forall r \in R', \tag{8}$$

$$\lambda_i + \omega_i^- \geqslant d_i^- \quad \forall i \in \{1\ldots N\}, \tag{9}$$

$$\lambda_i - \omega_i^+ \leqslant d_i^+ \quad \forall i \in \{1\ldots N\}, \tag{10}$$

$$\lambda, \omega^-, \omega^+ \geqslant 0, \tag{11}$$

while the primal form of the stabilized master is

$$\min \quad \sum_{r\in R'} c_r x_r + \sum_{i\in\{1\ldots N\}} -d_i^- y_i^- + d_i^+ y_i^+$$

$$\text{s.t.} \quad \sum_{r\in R'} a_{ir} x_r - y_i^- + y_i^+ \geqslant 1 \quad \forall i \in \{1\ldots N\},$$

$$y_i^- \leqslant \varepsilon^- \quad \forall i \in \{1\ldots N\},$$

$$y_i^+ \leqslant \varepsilon^+ \quad \forall i \in \{1\ldots N\},$$

$$x, y^-, y^+ \geqslant 0.$$

This method is subsequently referred as BoxPen stabilization since the bounds (d^-, d^+) on the λ dual variables can be represented by a set of *boxes* defined by centers (B^{c}) and sizes (B^{s}). The mapping between the box representation and the LP formulation (7)–(11) is the following: $d_i^- = B_i^{\mathrm{c}} - B_i^{\mathrm{s}}$ and $d_i^+ = B_i^{\mathrm{c}} + B_i^{\mathrm{s}}$, $\forall i \in \{1\ldots N\}$. The values of ε^- and ε^+ express the penalty imposed to the objective when the values of λ are chosen outside the boxes.

In order to use this method efficiently, one must define an initializing procedure for B^c and a size control procedure for B^s. If we can easily see that initial values of B^c could be provided by any heuristic estimating reasonable marginal costs, there is little information in the literature about the control of the box size and penalty factor (ε) during resolution. The approach described in [6] offers two possible approaches to update the parameters: one is to enlarge the size (B_i^s) of a box when the dual value λ_i (obtained from the LP) lies on or outside its boundary and to reduce it otherwise. The other is to keep increasing the values of ε^- and ε^+ as long as the subproblem is able to identify new columns and to reduce them otherwise. In both cases, when no new column can be found, all boxes are re-centered on the last dual values obtained ($B^c = \lambda$). The convergence criterion of the method is that the dual variables λ take values which are strictly within $[d^-, d^+]$ or that $\varepsilon^- = \varepsilon^+ = 0$.

Although the approach of du Merle et al. [7] is the most widely used for stabilizing column generation, several other stabilization approaches have been proposed in the literature. Neame [14] proposes the following scheme. The dual prices used to generate new columns are obtained by taking a linear combination of the dual prices of the previous iteration and an extreme dual solution of the current restricted master problem. The subproblem might then provide columns with a negative pricing but a positive reduced cost (according to the current extreme dual solution). These columns are rejected. If no negative reduced cost columns are obtained, the subproblem is solved again with a convex combination stepping closer to the current extreme dual solution. The genuine current extreme dual solution is finally chosen for pricing when its distance with the convex combination falls under a given parameter ε. For complete reviews and discussions on this topic, see Neame [14] and Lübbecke and Desrosiers [12].

3. Interior point stabilization

The idea behind interior point stabilization is to generate a dual solution that is in the interior of the convex hull of optimal dual solutions to the master instead of using one of its extreme points. Bixby et al. [2] have proposed to achieve this by using an interior point method to solve (M). The authors note that, although the interior point method requires less iterations than a simplex based method, each iteration takes more time. They argue that each call to the subproblem generates a relatively small amount of columns compared to the size of R', thus enabling a simplex based method to reoptimize incrementally (M) at each iteration. Since interior point methods cannot be warm started, they do not take advantage of the successive resolution of column generation approach.

The proposed way to achieve the centralization of dual values is to generate several extreme points of the optimal dual polyhedron and to compute an interior point corresponding to a convex combination of all these extreme points. This section provides details on how this is achieved.

3.1. Theoretical issues

As mentioned above, the original relaxed set covering problem (M) is given by (4)–(6) and its dual is denoted by (D) and can be written as

$$\begin{aligned} \max \quad & \sum_{i\in\{1\ldots N\}} \lambda_i \\ \text{s.t.} \quad & \sum_{i\in\{1\ldots N\}} \lambda_i a_{ir} \leqslant c_r \quad \forall r \in R, \\ & \lambda \geqslant 0. \end{aligned}$$

Once the unrestricted master is solved to optimality, let R^* be defined as the set of routes r for which $x_r > 0$ (i.e., some of the basic columns), and let C be the set of nodes for which the covering constraint (5) is not tight. Using complementary slackness conditions, the optimal dual polyhedron $\mathscr{D}$ containing all optimal values of λ is defined by

$$\sum_{i\in\{1\ldots N\}} \lambda_i a_{ir} \leqslant c_r \quad \forall r \in R\backslash R^*, \tag{12}$$

$$\sum_{i\in\{1\ldots N\}} \lambda_i a_{ir} = c_r \quad \forall r \in R^*, \tag{13}$$

$$\lambda_i = 0 \quad \forall i \in C, \tag{14}$$

$$\lambda_i \geqslant 0 \quad \forall \quad i \in \{1\ldots N\}\backslash C. \tag{15}$$

To obtain a point in this polyhedron, one can simply define a random objective function $u\lambda$ with $u \sim U(0,1)$ (i.e., each element of u is uniformly distributed between 0 and 1) and solve the following LP, denoted $(\mathscr{D}^u)$:

$$\max \quad \sum_{i\in\{1\ldots N\}} u_i\lambda_i, \tag{16}$$

$$\sum_{i\in\{1\ldots N\}} \lambda_i a_{ir} \leqslant c_r \quad \forall r \in R\backslash R^*, \tag{17}$$

$$\sum_{i\in\{1\ldots N\}} \lambda_i a_{ir} = c_r \quad \forall r \in R^*, \tag{18}$$

$$\lambda_i = 0 \quad \forall i \in C, \tag{19}$$

$$\lambda_i \geqslant 0 \quad \forall i \in \{1\ldots N\}\backslash C. \tag{20}$$

If this problem is solved with a simplex based method, then the optimal solution obtained will always be an extreme point of $\mathscr{D}$. Different instances of $(\mathscr{D}^u)$ can be generated by defining multiple objective functions (u vectors) and thus several extreme points of $\mathscr{D}$ can be obtained. For each u vector generated, both problems $(\mathscr{D}^u)$ and $(\mathscr{D}^{-u})$ are successively solved in order to favor the identification of distant extreme points. Since $\mathscr{D}$ is a convex set, any convex combination of its extreme points will lie within it. Furthermore, except in pathological cases (i.e., when the solutions obtained for different u correspond to the same extreme dual solution of (12)–(15), one would expect convex combinations of several extreme points with strictly positive coefficients to yield an interior point of $\mathscr{D}$. In particular, if we take the average of all obtained extreme points, we should obtain an interior point of $\mathscr{D}$ that gives much more centered dual values.

3.2. Practical issues

This procedure for obtaining an interior point is very simple, but it requires to either write the dual of the master problem or to automatically generate it. This step, which is not very difficult, can be a bit troublesome when additional constraints are present in (M). To avoid it, let us consider the dual of $(\mathscr{D}^u)$ denoted by $(\mathscr{P}^u)$ and defined by

$$\min \quad \sum_{r\in R} c_r x_r \tag{21}$$

$$\text{s.t.} \quad \sum_{r\in R} a_{ir} x_r \geqslant u_i \quad \forall i \in \{1\ldots N\}\backslash C, \tag{22}$$

$$\sum_{r\in R} a_{ir} x_r \geqslant -\infty \quad \forall i \in C, \tag{23}$$

$$x_r \geqslant 0 \quad \forall r \in R\backslash R^*, \tag{24}$$

$$x_r \text{ free} \quad \forall r \in R^*. \tag{25}$$

Interior point stabilization can thus be achieved by making simple modifications to the original primal problem. Extreme points of $\mathscr{D}$ are obtained if, after modifying the right-hand side, $(\mathscr{P}^u)$ is solved and its dual values are collected. The interior point is again generated by collecting a number of extreme points (here multipliers of $(\mathscr{P}^u)$) and averaging them.

Furthermore, it is not really necessary to modify all right-hand side terms (those set to random values). In order to improve efficiency, only a small subset of these terms can be changed so that the LP solver is able to resolve the new problem efficiently. One can also decide to solve both problems $(\mathscr{P}^u)$ and $(\mathscr{P}^{-u})$ for each u vector generated, in order to favor the identification of distant extreme points. Note that solving $(\mathscr{P}^{-u})$ is equivalent to minimizing, instead of maximizing $(\mathscr{P}^u)$.

This procedure is very general and could turn out to be very useful in many column generation applications. The extra time needed to compute the solutions of $(\mathscr{P}^u)$ is sometimes negligible compared to the time necessary to solve one subproblem; and as we will see in the experimental results, this procedure can significantly cut down the number of iterations needed to reach optimality.

3.3. Implications for the subproblem

In the context of unstabilized column generation, the first iterations are usually characterized by extreme dual values (either 0 or large numbers). At the subproblem level, this tends to make the search for negative reduced cost paths much easier, since all nodes which were given a 0 marginal cost can be discarded. Of course, the paths generated are of poor quality and these first iterations are only useful in stabilizing the whole process.

Unfortunately, when one introduces a stabilization technique, one also makes the first few subproblems harder. In some cases this can be problematic, namely when the subproblem solution algorithm tries to identify all non-dominated paths of negative reduced cost. This is due to the improved estimation of the marginal costs that yields more average values (fewer zeroes) and thus many more negative reduced cost paths. The performance decrease does not lie in the identification of these paths, but rather in the fact that many more paths can be identified.

Since there is no real need to generate all negative reduced cost paths, it is possible, and sometimes essential, to limit the total number of columns generated. This can be done easily (and is probably already implemented) in almost all subproblem algorithms whether they are based on dynamic programming [4,5,1] or on constraint programming [9,15]. A simple implementation is to stop the subproblem when a sufficient number of paths have been identified, but more sophisticated strategies could be devised. This limit insures that the computational time of the subproblem is unaffected by our stabilization technique.

4. Experimental results

In this section, we attempt to demonstrate the effectiveness of interior point stabilization with regard to two criteria. We first evaluate IPS as a simple (almost parameter free) stabilization method for column generation and then compare it to the stabilization techniques described in Section 2. The problem chosen to perform experiments is the vehicle routing problem with time windows (VRPTW).

4.1. Benchmarks

We have tested the proposed method on the well-known Solomon instances [16]. The geographical data are randomly generated in problem set R1, clustered in problem set C1, and a mix of random and clustered structures in problem set RC1. The customer coordinates are identical for all instances within one type (i.e., R, C and RC). The instances differ with respect to the width of the time windows. Some have very tight time windows, while others have time windows which are hardly constraining. Each instance contains 100 customers.

In the data sets, the distance matrix is not explicitly stated, but customer locations are given. Euclidean distances between these customers are computed with one decimal place to allow comparisons with other published methods.

4.2. Implementations

The column generation framework used to evaluate the stabilization method is quite straightforward since the restricted master problem is exactly the one described by Eqs. (4)–(6). Columns are generated using a resource constrained elementary shortest path algorithm based on dynamic programming [8]. At each iteration of the subproblem, the number of non-dominated columns generated at the destination node is limited to 500. The maximum allowed time to find a solution was set to 3600 s.

As the aim of this project was to assess the impact of stabilization, no particular effort was made to optimize the implementation of the framework. Furthermore, since a reduction of both the number of iterations and the CPU time can be observed at the root node of the search tree, we do not include in the following experiments the branch and price search for integer solutions. Typically, the branching procedure starts when the root node has been solved and thus a sufficiently large number of columns (dual cuts) have been generated. It can be expected that, beyond the root node, centering the dual value is not as essential since the intervals for dual values have already been significantly narrowed.

Computational experiments were carried out on a 1.8 GHz Pentium 4 PC with 256 Megabytes of RAM. The master problem was modeled and solved with Cplex 7.5 leaving all parameters to their default values.

4.2.1. Neame's method

We implemented a version of Neame's algorithm [14] (see Section 2), which exhibits lots of similarities with IPS. Indeed, if extreme solutions of the dual optimal face are produced in successive solutions of the restricted master problem, the dual prices provided by Neames's method will also be a convex combination of optimal dual solutions. The method needs two

parameters: the convex combination coefficients, which we set to $(\frac{1}{2}, \frac{1}{2})$, and ε, which we set to 0.1.

4.2.2. BoxPen stabilization

We also re-implemented the BoxPen stabilization method described in Section 2 using two different sets of initial values. In the first implementation, potentially good initial values are defined using some knowledge we have about the problem at hand. In the present case, we estimated the marginal costs of each node as the cost of its cheapest outgoing arc ($\lambda_i = \min_{j\in\{1\dots N\}} c_{ij}$) and initialized each box center with these values ($B_i^c = \lambda_i, \forall i \in \{1\dots N\}$).

The second strategy is to use as initial centers for the stabilizing boxes the optimal dual value associated with each node. Each problem is solved a first time and the optimal dual values (λ_i^*) are stored and used as initial box centers ($B_i^c = \lambda_i^*, \forall i \in \{1\dots N\}$) for the stabilized resolution. This process is clearly not applicable in practice, since each problem needs to be solved twice. However, since it is not clear a priori how BoxPen stabilization is sensitive to the quality of the initial boxes, this last evaluation permits to determine the potential best gain achievable with this technique.

For both strategies, ε^- and ε^+ are initially set to 0.001 and the parameter control process we used is the following: all boxes are initialized to a size of 0.2 ($B_i^s = 0.1, \forall i \in \{1\dots N\}$). If the subproblem is able to find negative reduced cost paths, then the values of ε^- and ε^+ are increased by 10%. However, when there is no more such column, the values of ε^- and ε^+ are considerably decreased (divided by 1000) and every box is re-centered on the last dual value ($B_i^c = \lambda_i, \forall i \in \{1\dots N\}$). This is performed in order to achieve convergence through the second criterion ($\varepsilon^- = \varepsilon^+ = 0$). We have experimented with different updating strategies and the one described above provided the best results.

4.2.3. Interior point stabilization

IPS was implemented using the model (*P*) ((21)–(25)) described in Section 3.2. The only parameter to set is the number of sample points to identify in order to calculate an interior point of the dual optimal space. The trade-off is that a larger number of points will produce a more centered point, but also requires more computing time (since a LP has to be solved for each point). Our experiments show that 20 points seems enough to ensure a stabilization effect, while computing more points does not really improve the global performance of the column generation framework. The values of u in (16) are simply set with the use of the standard random number generator of C++. As we mentioned earlier, for each u vector we solve both the minimization ($\mathscr{D}^{-u}$) and the maximization ($\mathscr{D}^{u}$) problems to favor diverse extreme points.

In the present case, since solving (*P*) takes less than 0.01 s, we were able to modify the right-hand side of all constraints (22) without significantly slowing down the global resolution process.

4.3. Comparisons and discussion

Table 1 compares the different stabilization approaches. *BPHeur* and *BPOpt* stand for the BoxPen stabilization method, respectively, with a heuristic or an optimal choice for the box centers. *IPS* is for interior point stabilization. IPS performs better than the BPHeur algorithm and exhibits results similar to BPOpt. We did not observe any stabilization effect of Neame's method on our benchmark instances. In fact, the number of iterations sometimes increased significantly, a result opposite to the one sought for.

Table 1 also shows that stabilization appears particularly effective for C instances, where customers are clustered. This can be easily interpreted: in these instances, clusters are commonly serviced with a single vehicle, which should discard every column not visiting all nodes in a cluster. The visit of every customer in a cluster is enhanced by the IPS, since low values for dual variables (and notably null dual variable) are avoided. Improvements are, however, smaller for the R and RC instances.

Table 2 further compares the effectiveness of the different stabilization approaches (except Neame's, which performs poorly) . For each algorithm, it indicates the computing time (*time*) and the number of iterations (*iter*). The optimal solution value is identical for all approaches and is mentioned in column *value*. *Unstabilized* corresponds to the standard column generation scheme for the solution of the master problem, ((4)–(6)), where we rely solely on the LP code to select the dual values.

Results reported in Table 2 are the time and number of iterations needed to identify the optimal

Table 1
Average impact on time and iterations of stabilization methods over the unstabilized algorithm

Class	Neame		BPHeur		BPOpt		IPS	
	Time (%)	Iter (%)	Time (%)	Iter (%)	Time (%)	Iter (%)	Time (%)	Iter (%)
R	107	108	103	87	86	71	79	64
C	161	200	88	75	49	26	29	30
RC	121	117	106	97	88	75	89	71

Table 2
Comparison of IPS with BoxPen-stabilized and unstabilized column generation algorithms

Instances	Value	Unstabilized		BoxHeur		BoxOpt		IPS	
		Time	Iter	Time	Iter	Time	Iter	Time	Iter
R101	1631.15	4.35	33	5.19	38	4.28	29	5.33	23
R102	1466.60	38.77	43	45.83	48	27.77	32	26.86	25
R103	1206.78	307.89	68	318.49	59	235.91	44	226.06	42
R105	1346.14	17.04	46	17.88	46	16.00	40	17.17	34
R106	1226.91	181.85	69	192.03	56	172.48	49	147.73	44
R107	1053.60	3451.77	88	3569.92	74	3018.99	66	2656.92	58
R109	1134.28	81.93	62	71.65	50	69.68	45	65.16	40
R110	1055.57	510.29	77	506.29	58	447.92	45	413.44	45
R111	1034.73	817.61	88	864.52	69	656.37	55	690.78	56
C101	827.30	19.25	87	12.17	45	4.18	16	7.56	23
C102	827.30	844.45	110	747.62	88	554.87	38	230.69	38
C105	827.30	30.01	81	28.27	70	8.47	21	13.05	27
C106	827.30	46.35	84	37.50	62	14.92	25	18.50	24
C107	827.30	32.27	74	38.31	65	17.49	30	17.85	30
C108	827.30	123.91	118	84.29	78	26.22	25	43.52	37
C109	827.30	317.53	124	296.89	101	67.45	22	80.97	27
RC101	1584.09	11.19	39	11.69	39	10.85	38	13.40	33
RC102	1406.26	81.63	56	85.63	54	69.91	43	68.48	38
RC103	1225.65	916.65	76	978.19	74	768.04	54	802.75	52
RC105	1471.93	41.84	51	48.00	50	38.10	38	36.75	33
RC106	1318.80	66.03	56	76.44	57	54.55	38	62.89	41
RC107	1183.37	514.70	68	524.21	61	498.82	48	469.34	47

solution to the linear relaxation of ((1)–(3)). Results show a clear improvement when implementing the interior point stabilization within the standard column generation scheme, both in time and in the number of iterations.

Experiments were also conducted on IPS using only one extreme point to artificially simulate an unstabilized behavior. The results obtained by this method were significantly poorer than those obtained by simply retrieving the marginal cost from the LP solver (here ILOG Cplex 7.5). We believe that this behavior is due to the fact that IPS with one sample point will generate dual value at random extreme points of the optimal polyhedron, a behavior not exhibited by the LP solver when performing incremental resolutions. This randomness probably worsens the instability when the number of sample points is insufficient.

All these results tend to demonstrate that even a very clever implementation of the BoxPen stabilization would not be far better than IPS. On the contrary, IPS outperforms a standard implementation of BoxPen, though being much simpler and intuitive.

References

[1] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, P.H. Vance, Branch-and-price: column generation for huge integer programs, Oper. Res. 46 (1998) 316–329.

[2] R.E. Bixby, J.W. Gregory, I.J. Lustig, R.E. Marsten, D.F. Shanno, Very large-scale linear programming: a case study in combining interior point and simplex methods, Oper. Res. 40 (1992) 885–897.

[3] G.B. Dantzig, P. Wolfe, Decomposition principles for linear programs, Oper. Res. 8 (1960) 101–111.

[4] M. Desrochers, J. Desrosiers, M.M. Solomon, A new optimisation algorithm for the vehicle routing problem with time windows, Oper. Res. 40 (1992) 342–354.

[5] J. Desrosiers, Y. Dumas, M.M. Solomon, F. Soumis, Time constrained routing and scheduling, in: M.O. Ball, T.L. Magnanti, C.L. Monma, G.L. Nemhauser (Eds.), Network Routing, Handbooks in Operations Research and Management Science, vol. 8, North-Holland, Amsterdam, 1995, pp. 35–139.

[6] O. du Merle, D. Villeneuve, J. Desrosiers, P. Hansen, Stabilisation dans le cadre de la génération de colonnes, Publication G-97-08, GERAD, 1997.

[7] O. du Merle, D. Villeneuve, J. Desrosiers, P. Hansen, Stabilized column generation, Discrete Math. 194 (1999) 229–237.

[8] D. Feillet, P. Dejax, M. Gendreau, C. Gueguen, An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems, Networks 44 (2004) 216–229.

[9] U. Junker, S.E. Karisch, N. Kohl, B. Vaaben, T. Fahle, M. Sellmann, A framework for constraint programming based column generation, in: Principles and Practice of Constraint Programming, Lecture Notes in Computer Science, 1999, pp. 261–274.

[10] J.E. Kelley, The cutting plane method for solving convex programs, J. SIAM 8 (1960) 703–712.

[11] S. Kim, K.-N. Chang, J.-Y. Lee, A descent method with linear programming subproblems for nondifferentiable convex optimization, Math. Programming 71 (1995) 17–28.

[12] M.E. Lübbecke, J. Desrosiers, Selected topics in column generation, Technical Report G-2002-64, GERAD, Montreal, December 2002.

[13] R.E. Marsten, W.W. Hogan, J.W. Blankenship, The BOXSTEP method for large-scale optimization, Oper. Res. 23 (1975) 389–405.

[14] P.J. Neame, Nonsmooth dual methods in integer programming, Ph.D. Thesis, University of Melbourne, 1999.

[15] L.-M. Rousseau, M. Gendreau, G. Pesant, Solving small VRPTWs with constraint programming based column generation, in: Proceedings of the Fourth Workshop on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 2002.

[16] M.M. Solomon, Algorithms for the vehicle routing and scheduling problem with time window constraints, Oper. Res. 35 (1987) 254–265.