Available online at www.sciencedirect.com

SCIENCE DIRECT®

European Journal of Operational Research 162 (2005) 122–125

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/dsw

Dual variable based fathoming in dynamic programs for column generation

Marco E. Lübbecke *

Technische Universität Berlin, Institut für Mathematik, Sekr. MA 6-1, Straße des 17. Juni 136, Berlin D-10623, Germany

Received 1 January 2002; accepted 1 May 2003
Available online 13 February 2004

Abstract

In this note, we aim at reducing the state space of dynamic programming algorithms used as column generators in solving the linear programming relaxation of set partitioning problems arising from practical applications. We propose a simple generic lower bounding criterion based on the respective dual optimal solution of the restricted master program.

Keywords: Dynamic programming; Column generation; State space reduction

1. Motivation

Column generation is a well-established method to solve large-scale (integer) linear programs, see e.g., [3,5,8]. In recent years we have seen the optimal solution of problems with millions of variables in various practical application areas. Often, a natural modeling approach is a formulation as a set partitioning style program. Classical examples are airline crew pairing [2] and vehicle routing [5]. Here, the pricing problem, i.e., the task to compute a favorable column to enter the basis or to prove that none such exists, commonly constitutes an $\mathcal{NP}$-complete combinatorial optimization problem; dynamic programming algorithms have been proposed for exact solutions, see e.g., [2,4–6,9]. This note provides a simple means to reduce the state space of these algorithms, rediscovering the *fathoming* technique in dynamic programming [1,10,12].

2. Dynamic programs as column generators

Stressing their practical relevance [3], we confine ourselves to linear programming (LP) relaxations of set partitioning problems. That is, we minimize $c^T x$ subject to $Ax = 1$, $x \geqslant 0$, where $A \in \{0,1\}^{m \times n}$, and n is typically *very* large. We assume $c \geqslant 0$. The columns of A encode a set S of *admissible subsets* of an m-set $\mathcal{S}$ via $a_{ij} = 1$ iff $i \in j \in S \subseteq 2^{\mathcal{S}}$, $i = 1, \ldots, m$. Usually $n = |S| \ll 2^m$, and set membership in S is defined by a set $\mathcal{C}$ of problem specific constraints. An example: a set $\mathcal{S}$ of m customers has to be visited, each exactly once,

* Tel.: +49-30-31425735; fax: +49-30-31425191.
E-mail address: m.luebbecke@math.tu-berlin.de (M.E. Lübbecke).

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2003.05.006

by a fleet of vehicles. Rules $\mathscr{C}$ ensure e.g., that time windows, vehicle capacities, and precedence relations among customers are respected. This results in the set S of (incidence vectors of) feasible vehicle routes, and associated costs c_j, $j \in S$.

The generic column generation scheme starts off with a small subset of S, the associated variables constitute the initial *restricted master program* (RMP). Adjoining negative reduced cost columns, and re-optimization of the RMP alternately proceed. Modern implementations usually use problem specific heuristics for solving the pricing problem, cf. [3,4], but this is not followed up here. Instead, we concentrate on an exact algorithm which has to be executed at *some* point in time, at least when one wishes to prove optimality of the RMP.

Let x and u denote the primal and dual optimal solutions, respectively, associated with the current RMP. Then, the pricing problem amounts to determining

$$z := \min\left\{c_j - \sum_{i \in j} u_i \mid j \in S\right\}. \tag{1}$$

If $z \geqslant 0$, *no* reduced cost coefficient is negative, and x, embedded in $\mathbb{R}^n$, optimally solves the original LP as well. Otherwise, when the minimum in (1) is attained for index $j' \in S$, column $a_{j'}$ is adjoined to the RMP. Implicit enumeration of S can be accomplished by a dynamic program, which starts from $R = \emptyset$, and successively appends elements from $\mathscr{S}$, the most recent of which is denoted by k. The current stage of the algorithm is represented by *state* (R,k) with $R \subseteq \mathscr{S}$, $k \in R$. A *transition* from (R,k) to (R',k') is feasible iff (R',k') is compliant to a set of constraints $\mathscr{C}$, which usually depend on (R,k). In the simplest case, each state is associated with its reduced cost $c(R,k)$. Note the assumption of additive costs along transitions. For every R generated during the process there exists a superset $\overline{R} \supseteq R$ with $\overline{R} \in S$. These so-called final states—which represent the feasible solutions for the pricing problem—are finally considered by the dynamic program, at least implicitly. The minimum reduced cost feasible solution among these determines z in (1).

3. Dual variable based fathoming

Besides classical *dominance* relations among states, *lower bounds* can be exploited to reduce the state space of dynamic programs [1,10,12]. Consider a particular state (R,k) during the assumed dynamic programming pricing algorithm. Denote by $\bar{z}$ the cost of a currently cheapest final state—referred to as the *incumbent*, initialized with a known upper bound, possibly infinity. Candidates for extension of R come from (but because of $\mathscr{C}$ need not be identical to) $\mathscr{S} \setminus R$. Now only final states having negative cost are interesting. Therefore, if $\mathrm{LB}(R,k) \geqslant \min\{0,\bar{z}\}$ holds for a lower bound $\mathrm{LB}(R,k)$ on the best possible reduced cost coefficient obtainable by (subsequent) transitions of state (R,k), we prune the search, which we call *fathoming* of the current state. In the following we present such a simple bound.

The idea is to relax constraints in $\mathscr{C}$ imposed on the extension of R. An immediate choice would be a *total relaxation*, i.e., disregarding $\mathscr{C}$ altogether. Denote by $\mathscr{S}_+ = \{i \in \mathscr{S} \mid u_i > 0\}$. No state derived from (R,k) can have cost smaller than

$$\mathrm{LB}(R,k) = c(R,k) - \sum_{i \in \mathscr{S}_+ \setminus R} u_i, \tag{2}$$

which—although straightforward—has not been generally stated, to the best of our knowledge. In general, this lower bound is possibly weak, but can be refined as follows. Not relaxing all $\mathscr{C}$, and utilizing *some* structural information on elements in S one might be able to restrict $\mathscr{S}_+$ to a more *meaningful* set to be substituted in (2). For instance, upper bounds on $\max_{j\in S}|j|$ can be applied to strengthen $\mathrm{LB}(R,k)$. In practical applications, such information is often available, e.g., the maximal number of customers on a vehicle tour. Another likely situation is to have $i_1, i_2 \in \mathscr{S}_+$ such that there is no $j' \in S$ with $R \cup \{i_1, i_2\} \subseteq j'$, that is, two *incompatible* elements, implied by $\mathscr{C}$, e.g., due to conflicting time windows. We would eliminate from $\mathscr{S}_+$ one which attains $\min\{u_{i_1}, u_{i_2}\}$. This can be generalized to more than two elements [7], still, the tradeoff between such efforts and computational gains should be kept in mind. If known, the minimal cost incurred by transitions from the current to a final state can be added to $\mathrm{LB}(R,k)$, e.g., the cost for a vehicle to return to a depot.

Suppose we fathom a state (R,k) when $\gamma \cdot \mathrm{LB}(R,k) \geqslant \min\{0,\bar{z}\}$ with $0 \leqslant \gamma \leqslant 1$. We already discussed the case $\gamma = 1$. For $\gamma = 0$, the inequality becomes redundant; all states would be eliminated. Assume that $\mathrm{LB}(R,k) \leqslant 0$, for otherwise the multiplication by γ would have no effect on the criterion. Then, for $0 < \gamma < 1$, we have $\gamma \cdot \mathrm{LB}(R,k) > \mathrm{LB}(R,k)$; *more* states than before are fathomed. In other words, an incumbent possibly with $\bar{z} > z$ will be considered optimal. Note, that $(z - \bar{z})/z < 1$ always holds for any incumbent with $0 > \bar{z} \geqslant z$. That is, as long as some negative reduced cost column is computed, we have a trivial upper bound on the relative error incurred for *any* pricing heuristic. Most notably, in this case the heuristic just described allows for a better approximation guarantee.

Lemma 1. *Let* $z \leqslant \bar{z} < 0$. *When* $\mathrm{LB}(R,k)$ *is replaced by* $\gamma \cdot \mathrm{LB}(R,k)$ *with* $0 < \gamma \leqslant 1$, *then* $(z - \bar{z})/z \leqslant 1 - \gamma$.

Proof. The modified lower bound is more effective only if $\mathrm{LB}(R,k) < \bar{z} \leqslant \gamma \cdot \mathrm{LB}(R,k)$ for some state (R,k). When such an additional elimination takes place, we obtain for the incumbent

$$\bar{z} - z \leqslant \gamma \cdot \mathrm{LB}(R,k) - z$$
$$\iff \frac{\bar{z} - z}{z} \geqslant \gamma \cdot \frac{\mathrm{LB}(R,k)}{z} - 1 \geqslant \gamma \cdot 1 - 1.$$

The last inequality follows from $\mathrm{LB}(R,k) \leqslant z < 0$, which holds by definition of the lower bound. Constraining $\bar{z}$ to be non-positive immediately yields the claim. $\square$

The case $z \geqslant 0$ is uninteresting; the assumed dynamic program truly returns that no negative reduced cost columns exist.

Let us finally remark that a variant of (2) is suited for *preprocessing* the data prior to solving a pricing problem. Again, we assume non-negative cost coefficients. If $u_p < 0$ and $\sum_{i \in \mathscr{S}_+} u_i + u_p \leqslant 0$, then the element in $\mathscr{S}$ corresponding to p cannot be promisingly incorporated in *any* feasible solution to the pricing problem, and therefore can be discarded throughout the calculation.

4. Concluding remarks

Our fathoming criterion also applies in the common presence of convexity constraints in the LP. When the cost structure is such that $j_1 \subseteq j_2 \subseteq \mathscr{S} \Rightarrow c_{j_1} \leqslant c_{j_2}$, considering the set covering relaxation $Ax \geqslant 1$ is no loss of optimality. The assumption $A \in \{0,1\}^{m \times n}$ may be temporarily relaxed for computational ease. The non-binary columns encode the repeated appearance of elements in a set S. In this case, (2) is no longer a lower bound. Independently of our work a similar bound has been proposed [11], however, in a pricing algorithm of *branch-and-bound* style, where the use of bounds is essential to the method.

Traditionally used dominance rules in dynamic programming require an efficiently manageable *global* overview of states, and only allow for discarding states which are *already generated*. Opposed to that, the benefit and advantage of our criterion is its ability to be checked *locally* in the sense that only knowledge about the current state is necessary. Future state transitions may be avoided in the first place. Computational experiments with (2), conducted in the context of a locomotive scheduling problem at in-plant railroads [9], indicate considerable state space reductions of more than 90%.

Acknowledgements

This research was funded by the German Federal Ministry of Education and Research (BMBF) under grants no. 03-ZI7BR2-1 and 03-ZIM2BS. It was done while the author was with the Department of Mathematical Optimization, Braunschweig University of Technology. I acknowledge the comments of two anonymous referees which lead to a more readable note.

References

[1] O.G. Alekseev, I.F. Volodos, Combined use of dynamic programming and branch-and-bound methods in discrete-programming problems, Automat. Remote Control 37 (1976) 557–565.

[2] R. Anbil, J.J. Forrest, W.R. Pulleyblank, Column generation and the airline crew pairing problem, in: Proceedings of the International Congress of Mathematicians Berlin, Extra Volume ICM 1998 of Doc. Math. J. DMV, pages III 677–686, August 1998.

[3] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, P.H. Vance, Branch-and-price: Column generation for solving huge integer programs, Operations Research 46 (3) (1998) 316–329.

[4] G. Desaulniers, J. Desrosiers, M.M. Solomon, Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems, in: C.C. Ribeiro, P. Hansen (Eds.), Essays and Surveys in Metaheuristics, Kluwer, Boston, 2001, pp. 309–324.

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

[6] J. Desrosiers, P. Pelletier, F. Soumis, Plus court chemin avec contraintes d'horaires. *RAIRO Recherche Opérationnelle*, 17 (4) (1983) 357–377, in French.

[7] M.E. Lübbecke, Engine Scheduling by Column Generation. Ph D thesis, Braunschweig University of Technology, Cuvillier Verlag, Göttingen, 2001.

[8] M.E. Lübbecke, J. Desrosiers, Selected topics in column generation. Les Cahiers du GERAD G-2002-64, HEC Montréal, Canada, under revision for Operations Research.

[9] M.E. Lübbecke, U.T. Zimmermann, Engine routing and scheduling at industrial in-plant railroads, Transportation Science 37 (2) (2003) 183–197.

[10] R.E. Marsten, T.L. Morin, A hybrid approach to discrete mathematical programming, Mathematical Programming 14 (1978) 21–40.

[11] A. Mehrotra, M.A. Trick, Cliques and clustering: A combinatorial approach, Operations Research Letters 22 (1) (1998) 1–12.

[12] T.L. Morin, R.E. Marsten, Branch-and-bound strategies for dynamic programming, Operations Research 24 (4) (1976) 611–627.