Computers & Operations Research 40 (2013) 1938–1946

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Hybrid column generation for large-size Covering Integer Programs: Application to transportation planning

L. Alfandari [a,*], J. Sadki [b], A. Plateau [c], A. Nagih [d]

[a] ESSEC Business School, 95021 Cergy-Pontoise, France
[b] Université Paris 13, Sorbonne Paris Cité LIPN, CNRS(UMR 7030), 93430 Villetaneuse, France
[c] CEDRIC, CNAM, 75141 Paris, France
[d] LCOMS, Université de Lorraine, Ile du Saulcy, 57045 Metz Cedex 1, France

ARTICLE INFO

Available online 27 February 2013

Keywords:
Column generation
Covering integer programming
Approximation
Greedy heuristic
Fractional problems
Transportation planning

ABSTRACT

The well-known column generation scheme is often an efficient approach for solving the linear relaxation of large-size Covering Integer Programs (*CIP*). In this paper, this technique is hybridized with an extension of the best-known *CIP* approximation heuristic, taking advantage of distinct criteria of columns selection. This extension uses fractional optimization for solving pricing subproblems. Numerical results on a real-case transportation planning problem show that the hybrid scheme accelerates the convergence of column generation both in terms of number of iterations and computational time. The integer solutions generated at the end of the process can also be improved for a significant proportion of instances, highlighting the potential of diversification of the approximation heuristic.

1. Introduction

Covering Integer Programming (*CIP*) is an NP-hard minimization problem that models real-case applications like location problems [8]. It can also appear as the master problem of a Dantzig–Wolfe decomposition in other applications, e.g. transportation problems [1,29] and cutting stock problems [12]. This paper is devoted to the second category of large-size covering integer programs. For solving the linear relaxation of these problems, the column generation method is an efficient approach when the pricing subproblem can be solved in reasonable time. However, as it only provides a lower bound of the optimal solution, it is often combined with other solving approaches to obtain integer solutions: exact methods (e.g. branch-and-bound and cuts) or approximation methods (e.g. heuristics, metaheuristics and Lagrangian methods) [22,17,30,19,3]. We will denote by *CG+MIP* the two-stage process that consists in, firstly, solving the linear relaxation of the master problem by Column Generation (CG), and secondly, running a Mixed-Integer Programming (MIP) solver on the last restricted master problem in order to get integer solutions. Various improvements of column generation have been proposed both for the general process (e.g. initialization, resolution of the master problem and subproblems, choice of inserted columns) as for the integer resolution scheme (e.g. branch-and-price, column generation combined to heuristics or metaheuristics) [2,26,27,16,5]. Since the goal of the paper is not to design the best-possible solving method for a specific problem but to show the added value of hybridization in column generation *before* branching, we keep a basic MIP branching scheme without exploring branch-and-price techniques.

The main contribution of this paper is an original and efficient hybridization of *CG* with (i) the greedy heuristic of Dobson [10], denoted by *Gr*, which achieves a logarithmic approximation ratio for CIP, and (ii) an extension of *Gr* to large-size *CIP* problems, denoted by Gr^+ Although the principle of such an extension was already described in [6], no paper has ever analyzed the fractional subproblems associated with Gr^+ as it is done in this paper, nor studied how to implement it.

Numerical experiments are conducted on real-case instances of a *locomotive assignment problem*. Both *CG+MIP* and Gr^+ are first evaluated separately on the problem. Although the greedy heuristic does not generally have outstanding performance in terms of solution quality, it shows however an interesting potential for generating diversified columns with controlled quality and running time. This is a strong motivation to design strategies for an efficient hybridization of the two resolution approaches. This hybrid scheme has two objectives:

1. Accelerate the column generation process that solves the linear relaxation of the CIP master problem,
2. output an integer solution that is strictly better than both the *CG+MIP* solution and the Gr^+ solution.

* Corresponding author. Tel.: +33 134433284.
E-mail addresses: alfandari@essec.edu (L. Alfandari), jalila.sadki@lipn.univ-paris13.fr (J. Sadki), agnes.plateau_alfandari@cnam.fr (A. Plateau), anass.nagih@univ-lorraine.fr (A. Nagih).

0305-0548/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2013.02.020

The paper is organized as follows. In Section 2, we briefly present the *CIP* formulation and describe the standard Column Generation and *Gr* principles. Section 3 presents the Gr^+ extension of *Gr* to large-size *CIP*, and makes the link between the heuristic subproblem and fractional optimization. The generic hybridization scheme is described in Section 4. Section 5 describes the transportation planning application and analyses the greedy subproblem tractability. Section 6 provides and analyses computational results on the real-case problem for *CG+MIP* and Gr^+ tested separately, then for hybridization. Different hybridization strategies are proposed and evaluated. Final conclusions are given in Section 7.

2. Existing solving approaches for CIP

We first introduce in this section the formulation of CIP programs. We recall then the column generation principles on *CIP*. We finish by a brief description of *Gr*.

2.1. CIP formulation

Given a matrix A of integer non-negative coefficients $a_{ij} \in \mathbb{N}_0, i \in I = \{1,\dots,n\}, j \in J = \{1,\dots,m\}$, positive vectors $c \in \mathbb{N}_0^m$, and $b \in \mathbb{N}_0^n$, a covering integer program *CIP* consists in finding a vector $y \in \mathbb{N}_0^m$ minimizing the cost function cy while satisfying covering constraints $Ay \geq b$. The mathematical formulation of *CIP* is given as follows:

$$(CIP)\begin{cases} \min & z = \sum_{j \in J} c_j y_j \\ \text{s.t} & \sum_{j \in J} a_{ij} y_j \geq b_i \quad \forall i \in I \\ & y_j \in \mathbb{N}_0 \quad \forall j \in J \end{cases} \tag{1}$$

The *CIP* defined above is sometimes called *multiset multicover problem* [28] or *integer covering problem* [11,6]. The well-known set covering problem corresponds to the case when $a_{ij} \in \{0,1\}$ for $i \in I, j \in J$, $b_i = 1$ for all $i \in I$, and $y_j \in \{0,1\}$ for $j \in J$.

2.2. Column generation for large-size CIP

In many applications formulated as a *CIP*, $|J|$ is a very large number which can be an exponential function of $n = |I|$. For example, J can represent the set of all paths satisfying given constraints in a graph, as for the transportation planning application tested in this paper. Complete enumeration of J is then impossible for solving these problems.

Column Generation (CG) is a resolution scheme particularly suited to solve the linear relaxation of these large-size problems. It is an extension of the Simplex algorithm to the case when J is very large which implies a non-trivial computation of reduced costs. CG is typically applied when the subproblem has a particular combinatorial structure that makes it easy to solve despite the exponential size of J.

The linear relaxation of the (CIP)-formulation is called the Master Problem (MP). As J is very large, the CG scheme only generates a subset of columns of J. It iteratively solves the MP restricted to a subset of columns, called Restricted Master Problem (*RMP*). At each iteration, the *RMP* is enlarged by adding new columns with negative reduced costs output by a pricing subproblem. This subproblem generates the column with minimum reduced cost that will enter the basis, i.e.,

$$\min_{j \in J} \left\{ c_j - \sum_{i \in I} u_i a_{ij} \right\}$$

where $u = (u_i)_{i \in I}$ denotes the vector of dual variables associated with the covering constraints (1). As mentioned earlier, this subproblem should be tractable in reasonable time despite the size of J.

If at some iteration, the pricing subproblem outputs no column with negative reduced cost, then the current solution is optimal for the MP (this only holds if the subproblem is solved to optimality).

The approach *CG+MIP* developed in this paper is a two-stage process. The first stage uses the above CG process to solve the linear relaxation of the CIP master problem. At the second stage, a MIP solver is applied to the last RMP for getting integer solutions.

We describe the *Gr* heuristic that provides the best-possible approximation ratio, and its extension to large-size *CIP* in the following section.

2.3. Greedy approximation heuristic

When J is given explicitly, the greedy heuristic *Gr* designed by Dobson [10] is the best approximation heuristic known for *CIP*. *Gr* is a generalization of the greedy heuristic for the set covering problem analyzed by Chvatàl [4]. It achieves a logarithmic approximation ratio equal to

$$H\left(\max_{j \in J}\left(\sum_{i \in I} a_{ij}\right)\right)$$

where $H(k) = \sum_{i=1}^{k} 1/i$ is the harmonic series.

At each iteration, algorithm *Gr* selects a column $j \in J$ with minimum ratio $c_j / \sum_{i \in I} a_{ij}$. Then the number of units to cover b_i is updated by subtracting the number of units covered at the current iteration, and all a_{ij} that become higher than the updated value of b_i are set to this value. This process is reiterated until all demand units are covered. It is formally described as follows:

Algorithm 1. Greedy algorithm *Gr* for *CIP* [10].

- **Input** :
  $J_{Gr} \leftarrow \emptyset,\ y \leftarrow 0$
- **while** $b \neq 0$ **do**
  1. $j^* \leftarrow arg \min_{j \in J} \frac{c_j}{\sum_{i \in I} a_{ij}}$
  2. $y_{j^*} \leftarrow y_{j^*} + \Delta_{j^*}$ with $\Delta_{j^*} = \min_{i \in I} \left\lfloor \frac{b_i}{a_{ij^*}} \right\rfloor$
  3. $J_{Gr} \leftarrow J_{Gr} \cup \{j^*\}$
  4. $b_i \leftarrow b_i - \Delta_{j^*} a_{ij^*}\ \forall i \in I$
  5. $a_{ij} \leftarrow \min(a_{ij}, b_i)\ \forall i \in I, j \in J$
- **output** J_{Gr}

In the next section, Gr^+, the extension of *Gr* to the case when $|J|$ is an exponential function of $|I|$ is analyzed. For such large-size *CIP*, as for the standard CG process and under some conditions, *Gr* can be used efficiently despite the size of J.

3. Extension of *Gr* to large size *CIP* using fractional optimization

To get the column of minimum ratio (step 1. of algorithm *Gr*), one has to evaluate $c_j / \sum_{i \in I} a_{ij}$ for every $j \in J$ and then find $\min_{j \in J} c_j / \sum_{i \in I} a_{ij}$. Gr^+ is the extension of *Gr* to a large-size context where explicit enumeration of J is impossible. Crama and Van de Klundert [6] called it *greedy column generation*. However, we prefer