European Journal of Operational Research 000 (2017) 1–12

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Innovative Applications of O.R.

Dynamic penalization of fractional directions in the integral simplex using decomposition: Application to aircrew scheduling

Samuel Rosat[a,b,*], Frédéric Quesnel[a,b], Issmail Elhallaoui[a,b], François Soumis[a,b]

[a] Polytechnique Montréal, C.P. 6079, Succ. Centre-ville, Montreal, QC H3C 3A7, Canada
[b] GERAD, 3000, ch. de la Côte-Sainte-Catherine, Montreal, QC H3T 2A7, Canada

ARTICLE INFO

Article history:
Received 11 January 2016
Accepted 24 May 2017
Available online xxx

Keywords:
Integer programming
Primal algorithms
Integral simplex
Normalization
Aircrew scheduling

ABSTRACT

To solve integer linear programs, primal algorithms follow an augmenting sequence of integer solutions leading to an optimal solution. In this work, we focus on a particular primal algorithm, the integral simplex using decomposition (ISUD). To find the next point, one solves a linear program to select an augmenting direction for the current point from a cone of feasible directions. To ensure that this linear program is bounded, a normalization constraint is added and the optimization is performed on a section of the cone. The solution of the linear program, i.e., the direction proposed by the algorithm, strongly depends on the chosen normalization weights, and so does the likelihood that the next solution is integer. We modify ISUD so that the normalization is dynamically updated whenever the direction leads to a fractional solution, to penalize that direction. We propose update strategies, based on new theoretical and experimental results. To prove the efficiency of our strategies, we show that our version of the algorithm yields better results than the former version and than classical branch-and-bound techniques on a benchmark of industrial aircrew scheduling instances. The benchmark that we propose here is, to the best of our knowledge, comparable to no other from the literature. It provides large-scale instances with up to 1700 flights and 115,000 pairings, hence as many constraints and variables, and the instances are given in a set-partitioning form together with initial solutions that accurately mimic those of industrial applications. Our work shows the strong potential of primal algorithms for the crew scheduling problem, which is a key challenge for large airlines.

© 2017 Elsevier B.V. All rights reserved.

1. Introduction

Crew pairing is a key challenge for large airlines: it is both financially significant and notably hard to solve. The fundamental issue is the size of the instances: companies have grown continually since the very beginning of passenger air traffic, as a result of increases in passenger volumes and occasional mergers (Continental Airlines and United Airlines, Air France and KLM, etc.). The problem involves determining a set of pairings (where a pairing is a sequence of flights and layovers that starts and ends at the same base) that covers all scheduled flights at minimal cost over the planning horizon. Standard solution techniques are based on the branch-and-bound algorithm. First, the integrality constraints are relaxed and an optimal fractional solution is found. Second, a branching tree is explored until integrality is restored and the integrality gap reduced to a given threshold. The exploration of the tree involves solving numerous linear relaxations of slightly modified versions of the original problem. The main drawbacks of this strategy are as follows. The size of the tree, and therefore the solution time, grows exponentially with the size of the data. Furthermore, the method used to solve the linear relaxations, the simplex algorithm, performs poorly on problems that have many degenerate solutions. A (basic) feasible solution is *degenerate* when some variables in the corresponding simplex basis have the value zero or, equivalently, when too many of the constraints are simultaneously saturated. Moreover, such methods may take a long time to find a first (and possibly good) feasible integer solution. Another approach for these problems (and for integer linear programming in general) uses a *primal*, or *augmentation*, algorithm: it starts from a known feasible solution and iteratively improves it until optimality is reached. At each step it solves an augmentation subproblem that either furnishes an improving direction or asserts that the current solution is optimal. In the former case, this direction is followed from the current solution to a strictly better one. The main

* Corresponding author at: Polytechnique Montréal, C.P. 6079, Succ. Centre-ville, Montreal, QC H3C 3A7, Canada.
E-mail addresses: samuel.rosat@polymtl.ca (S. Rosat), frederic.quesnel@polymtl.ca (F. Quesnel), issmail.elhallaoui@gerad.ca (I. Elhallaoui), francois.soumis@gerad.ca (F. Soumis).

http://dx.doi.org/10.1016/j.ejor.2017.05.047
0377-2217/© 2017 Elsevier B.V. All rights reserved.

Please cite this article as: S. Rosat et al., Dynamic penalization of fractional directions in the integral simplex using decomposition: Application to aircrew scheduling, European Journal of Operational Research (2017), http://dx.doi.org/10.1016/j.ejor.2017.05.047

drawbacks of this method are the need for an initial feasible solution and the need to ensure that the improved solution is still integer.

Among particular features of crew scheduling problems are the following. First, they have a high proportion of so-called set-partitioning constraints, and these tend to increase drastically the proportion of degenerate solutions. Then, the determination of good initial solutions proves to be relatively easy (see Section 4). Hence they are well-fitted for the primal approach. The advantages of this approach are that it takes advantage of existing solutions in a way which is not possible with branch and bound, and it can quickly improve the given solution. The former is particularly important since it allows the approach to be used both for planning and for reoptimization after unforeseen events, when the existing solution can be used as a starting point.

This paper is organized as follows. In Section 2, we give an overview of the existing literature on augmentation methods, particularly in the context of the set partitioning problem (SPP) where these algorithms can be based on simplex pivots and are hence called *integral simplex* methods. We concentrate on a specific augmentation algorithm, the integral simplex using decomposition (ISUD, Rosat, Elhallaoui, Soumis, & Chakour, 2017; Zaghrouti, Soumis, & Elhallaoui, 2014). We detail the main contributions of this paper in Section 2.4. In Section 3 we improve the ISUD by dynamically modifying one of the constraints in the augmentation subproblem, called the normalization constraint. We provide new theoretical results on the geometry of the set of possible normalization constraints, and we describe our update method, giving its theoretical basis. In Section 4, we propose a new benchmark of SPPs generated from aircrew scheduling applications; these problems have specific features that distinguish them from other benchmarks. In Section 5, we report numerical results that demonstrate the improvements obtained by our dynamic update of the constraints. Section 6 provides concluding remarks.

2. Integral simplex algorithms for the set partitioning problem

This section gives an overview of the existing literature on augmentation methods. In Section 2.1, we give a description of primal algorithms and integral simplex methods. In Section 2.2, we review the literature on integral simplex methods. A detailed description of the ISUD algorithm is given in Section 2.3. Finally, the contributions of this paper are summarized in Section 2.4.

2.1. Problem formulation and integral simplex definition

We assume that the reader is familiar with integer linear programming (see for example Schrijver, 1998). A *primal* (or *augmentation*) algorithm is a method that, given an initial solution of the integer linear program, outputs a sequence of feasible (integer) solutions of strictly augmenting cost such that the last one is optimal. By *augmenting*, we mean increasing for a maximization problem and decreasing for a minimization problem. In this paper, we consider minimization but the ideas also apply to maximization. Every primal method is based on the iterative solution of the *integral augmentation problem*, which can be stated as follows:

i AUG Find an augmenting vector $\boldsymbol{z} \in \mathbb{Z}^n$ such that $(\boldsymbol{x}+\boldsymbol{z})$ is feasible and $\boldsymbol{z}$ is of negative cost, or assert that $\boldsymbol{x}$ is optimal.

Here $\mathbb{Z}^n$ is the set of integers, and $\boldsymbol{x}$ is a (known) solution of the integer linear program. If an augmenting vector $\boldsymbol{z}$ is found, $\boldsymbol{x}' = (\boldsymbol{x}+\boldsymbol{z})$ becomes the next solution in the sequence, and **iAUG** is then solved again with $\boldsymbol{x}'$ as the initial solution. The process stops when $\boldsymbol{x}$ is determined to be optimal. Unfortunately, in general, the theoretical complexity of a single augmentation step is the same as that of solving the integer linear program (Schulz, Weismantel, & Ziegler, 1995), and integer linear programming is $\mathcal{NP}$-hard. However, in practice, **iAUG** or a good relaxation of **iAUG** is relatively easy to solve and often produces integral augmentations on specific problems, including the one we consider here. For a detailed review of primal algorithms, see Spille and Weismantel (2005).

In this paper, we concentrate on a specific integer linear program, the SPP, because it is the core of scheduling models. Its formulation is

$$z^\star_{\text{SPP}} = \min_{\boldsymbol{x}\in\mathbb{R}^n}\left\{\boldsymbol{c}^T\boldsymbol{x} \,|\, \boldsymbol{A}\boldsymbol{x}=\boldsymbol{e},\ \boldsymbol{0}\le\boldsymbol{x}\le\boldsymbol{e},\ \boldsymbol{x} \text{ is integer}\right\}, \tag{SPP}$$

where $\boldsymbol{A} \in \{0,1\}^{m\times n}$ is an $m \times n$ binary matrix, and $\boldsymbol{c} \in \mathbb{N}^n$ is the cost vector. Here $\boldsymbol{0}$ and $\boldsymbol{e}$ are vectors of zeroes and ones with size dictated by the context. Without loss of generality, we assume that $\boldsymbol{A}$ is full rank, contains no zero rows or columns, and has no identical rows or columns. The equalities $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{e}$ are called the linear constraints and $\boldsymbol{0}\le\boldsymbol{x}\le\boldsymbol{e}$ are the bound constraints. The set of all feasible solutions of SPP is denoted $\mathcal{F}_{\text{SPP}}$; the optimal value (or optimum) is $z^\star_{\text{SPP}}$; and any feasible solution $\boldsymbol{x}^\star \in \mathcal{F}_{\text{SPP}}$ such that $\boldsymbol{c}^T\boldsymbol{x}^\star = z^\star_{\text{SPP}}$ is an optimal solution of SPP. Because of the bounds ($\boldsymbol{0}$ and $\boldsymbol{e}$) and the integrality constraints, $\mathcal{F}_{\text{SPP}}$ contains only $\{0,1\}$-vectors, i.e., $\mathcal{F}_{\text{SPP}} \subseteq \{0,1\}^n$. Finally, the linear relaxation of SPP, denoted SPP^{LR}, is the linear program obtained by removing the integrality constraints. Its feasible domain is $\mathcal{F}_{\text{SPP}^{\text{LR}}}$ and its optimal value is $z^\star_{\text{SPP}^{\text{LR}}}$. If k augmentations have been performed, $\boldsymbol{x}^k$ designates a known integer solution that we seek to improve, and $\boldsymbol{x}^{k+1}$ is the next solution that we want to determine.

The SPP has specific features that allow the design of integral simplex methods. First, SPP is a $\{0,1\}$-program, so all the integer points of the domain of $\mathcal{F}_{\text{SPP}^{\text{LR}}}$ are extreme points of that polytope. Second, as shown by Trubin (1969), it possesses the *quasi-integral* property, also called the *Trubin property*: every edge of the convex hull of the integer domain $Conv(\mathcal{F}_{\text{SPP}})$ is also an edge of $\mathcal{F}_{\text{SPP}^{\text{LR}}}$. These two observations show that from any initial solution there exists a finite sequence of strictly augmenting integer solutions $(\boldsymbol{x}^k)_k$ that reaches an optimal solution and has the property that any two consecutive points are adjacent in $\mathcal{F}_{\text{SPP}^{\text{LR}}}$. Hence, it is possible to go from one to the other by performing a sequence of pivots. Moreover, it is always possible to find a sequence of pivots between two adjacent vertices such that only one of them is nondegenerate; otherwise they would not be adjacent. An integral simplex method is thus a primal algorithm in which an augmentation is performed through one or several simplex pivots, only one of which is nondegenerate.

2.2. Literature review

Set partitioning problems have intrinsic degeneracy, and this is a challenge for primal algorithms, especially algorithms based on simplex pivots. Therefore, integral simplex implementations need techniques that are specifically designed to cope with degeneracy. One of the first algorithms to take degeneracy into account was that of Balas and Padberg (1975). Given an initial solution $\boldsymbol{x}^k$ of SPP, they generate a large number of augmenting directions and consider only those that lead to other integral solutions. For an integer solution $\boldsymbol{x}^{k+1} \neq \boldsymbol{x}^k$, let $\boldsymbol{d} = \boldsymbol{x}^{k+1} - \boldsymbol{x}^k$ be the corresponding direction. The positive support of $\boldsymbol{d}$, $\mathcal{Q}^+ = \{j | d_j > 0\}$, defines the set of indices of columns that should be pivoted into the basis to go from $\boldsymbol{x}^k$ to $\boldsymbol{x}^{k+1}$.

In the terminology of Balas and Padberg (1975), $\mathcal{Q}^+$ is *nondecomposable* if, for any strict subset $\tilde{\mathcal{Q}}^+ \subsetneq \mathcal{Q}^+$, performing pivots on the variables indexed by $\tilde{\mathcal{Q}}^+$ results in no change in the solution (degenerate pivots only). They prove that $\mathcal{Q}^+$ is nondecomposable if and only if $\boldsymbol{x}^{k+1}$ is a neighbor of $\boldsymbol{x}^k$. After restricting the set of directions to those that are integral, they choose the steepest and either perform the corresponding pivots if it is augmenting, or prove

Please cite this article as: S. Rosat et al., Dynamic penalization of fractional directions in the integral simplex using decomposition: Application to aircrew scheduling, European Journal of Operational Research (2017), http://dx.doi.org/10.1016/j.ejor.2017.05.047

Download English Version:

https://daneshyari.com/en/article/4959356

Download Persian Version:

https://daneshyari.com/article/4959356

Daneshyari.com