Computers and Operations Research 81 (2017) 247–256

Contents lists available at ScienceDirect

Computers and Operations Research

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

An exact algorithm for a Vehicle-and-Driver Scheduling Problem

Bencomo Domínguez-Martín, Inmaculada Rodríguez-Martín*, Juan-José Salazar-González

DMEIO, Facultad de Matemáticas, Universidad de La Laguna, Tenerife, Spain

ARTICLE INFO

Article history:
Received 11 July 2016
Revised 15 November 2016
Accepted 27 December 2016
Available online 28 December 2016

Keywords:
Task scheduling
Branch-and-cut
Multi-depot
Vehicle routing

ABSTRACT

This article introduces a combinatorial optimization problem that consists of assigning tasks to machines and operators, and sequencing the tasks assigned to each one. Two configurations exist. Machines alternate configurations, while the operators must start and finish the process in the same configuration. Moreover, machines and operator have limited capacities. The sequencing of the tasks must guarantee that each one is performed by a machine and an operator at the same time, and it is determined in order to minimize an overall cost function. Two critical aspects of the problem are the need of synchronizing the machine and the operator performing each task, and the need of minimizing the changeovers, which are pairs of tasks done consecutively by the same machine but by different operators. The problem is modeled as a vehicle routing problem with two types of vehicles and with two depots. We propose a mixed integer programming formulation, and introduce valid inequalities to strengthen its linear programming relaxation. We describe separation routines for these inequalities and design a branch-and-cut algorithm for the problem. The algorithm is tested on a set of benchmark instances showing that it is able to solve to optimality instances with up to 50 customers.

1. Introduction

The problem addressed in this paper can be stated in a general way as a task scheduling problem where each task must be assigned to one machine and one operator. Tasks assigned to the same machine or operator cannot be performed in parallel, and the execution of a task cannot be interrupted. Thus, the tasks assigned to a machine or operator must be ordered. There are two configurations on which machines and operators must be before and after processing their tasks. A machine changes from an initial to a final configuration during the processing of its tasks, while an operator must start and finish in the same configuration. Moreover, machines and operator can only process a limited number of tasks. There is a cost for changing from a task to the next one in a schedule, a cost for each operator used, and a cost associated to each pair of consecutive tasks performed by the same machine but by different operators (changeovers). The objective of the problem is to assign the tasks to machines and operators, and to sequence the tasks assigned to each one, in order to minimize the total solution cost.

The problem was inspired by some particular planning characteristics of an air transportation company operating in the Canary Islands. A number of commercial flights (tasks) must be operated every day, using several aircraft (machines) and crews (operators). All the flights must be operated between 07:00 and 23:00, which allows to decompose the problem in a daily base. There are two hub airports, TFN and LPA, where the crews' homes are (the two configurations). To avoid overnight costs to the company, the flights assigned to a crew must allow that crew to start and end the day at the airport where its home is. The aircraft need to be checked every two days, and this operation is called *short-time maintenance*. The short-time maintenances take place during the night (between 23:00 and 07:00) and can only be performed at LPA airport, where the required equipments are located. The short-time maintenances have a major planning impact as they force that, in a particular night, half of the aircraft fleet stay at LPA while the other half stay at TFN. Thus, the flights assigned to each aircraft must be such that it starts the day in one hub airport and ends in the other one. This movement guarantees the short-time maintenance on each aircraft. In addition there are other constraints like the maximum number of flights that can be assigned to a crew.

Salazar-González [15] studied the complex real-world problem posed by the air carrier in the same regional context. That problem also involves, among other aspects, the so-called *aircraft routing* and *crew routing* problems, where a *route* is a set of flights that can be sequentially operated. However, in the real-world problem the departure and arrival times for each flight are given in advance. This fact makes the graph of connecting flights acyclic, besides being sparse, and thus dynamic programming provides efficient approaches to find min-cost routes (see for example Freling

* Corresponding author.
E-mail addresses: bdomingu@ull.es (B. Domínguez-Martín), irguez@ull.es (I. Rodríguez-Martín), jjsalaza@ull.es (J.-J. Salazar-González).

http://dx.doi.org/10.1016/j.cor.2016.12.022

et al. [8]). Under this condition, Salazar-González [15] described a heuristic approach and Cacchiani and Salazar-González [4] gave an exact method to solve the airline problem. In this paper we study a more general problem, without assuming precedences among the tasks, or, using the air transportation example, with unfixed departure times for the flights.

The problem without given precedence constraints on the tasks can be modeled as a vehicle routing problem. A route represents a sequence of tasks. The tasks can be considered as customers that need to be served by two types of vehicles. One type of vehicles (type 1) represents the operators. The other type of vehicles (type 2) represents the machines. The two machine configurations are called depots. There are costs associated to consecutive customers in a route. Such cost could represent, in the air transportation example, the waiting time in an airport for the crews, and the cleaning operations for aircraft between two consecutive flights. There are limits on the maximum number of tasks in a route (e.g. a crew cannot operate more than 6 flights each day). There is a cost associated to each operator (crew) used, and to each changeover (two consecutive flights done by the same plane but by different crews). The problem consists of designing min-cost routes so that each customer is served by one vehicle of each type. Vehicles of type 1 must start and end their routes at the same depot, while vehicles of type 2 must leave from one depot and arrive to the other.

This problem is called *Vehicle-and-Driver Scheduling Problem* (VDSP) from now on in the paper. It is closely related to the well-known Capacitated Vehicle Routing Problem (CVRP) where there is one depot and one type of vehicles. See e.g. Toth and Vigo [16] for a survey on this problem. There are also studies on CVRP variants that take into account several depots (see e.g. Laporte et al. [10,11]). However, we are not aware of any study on the VDSP. The major contribution of our work is to introduce and solve the VDSP. We do not intend to give an exact algorithm for the actual problem that was described in Salazar-González [15]. Instead, our article introduces and investigates a new problem in a wider context without assuming, for example, a sparse graph connecting the customers (as happens in Salazar-González [15] and Cacchiani and Salazar-González [4]). The new problem is presented as a routing problem with emphasis on the synchronization aspect between the two types of vehicles when serving a customer. The article gives a mathematical formulation for the problem and describes an exact approach to find optimal solutions.

The remainder of the paper is structured as follows. Section 2 formally describes the problem and details the mathematical formulation. It also presents several families of valid inequalities to strengthen the linear programming relaxation. Section 3 proposes a branch-and-cut algorithm to solve the problem and explains the separation procedures for each family of inequalities. Section 4 presents computational results obtained when we implemented the algorithm and used it to solve different instances. We compare the results using the valid inequalities at different stages of the algorithm. Finally, the paper ends with conclusions in Section 5.

2. Mathematical formulation and valid inequalities

In this section we formally describe the VDSP, give a mathematical model and present the valid inequalities used to strengthen its linear programming relaxation. We start by setting up the notation.

We are given n customers, two depots, and two types of vehicles. The set of customers is represented by $V_c = \{1, \dots, n\}$ and the set of depots by $V_d = \{0, n+1\}$. Let $G = (V, A)$ be a complete directed graph with vertex set $V = V_c \cup V_d$ and arc set $A = \{(i, j) : i, j \in V, i \neq j\}$. To refer to the set of arcs with tail in a set $S \subseteq V$ and head in $V \backslash S$, we use $\delta^+(S)$ instead of $\{(i, j) \in A : i \in S, j \notin S\}$, and we use $\delta^-(S)$ instead of $\delta^+(V \setminus S)$. The cost to pay when a vehicle of type k, for $k = 1$ and $k = 2$, traverses an arc (i, j) is denoted by c^k_{ij}, and it is assumed to be known. All vehicles of type k have a capacity equal to Q^k, which represents the maximum number of customers that can be served by the vehicle. We assume that the number of vehicles of type k available at depot d is K^k_d.

The aim of the problem is to design feasible routes in G in order to visit each customer with one vehicle of each type. A route for a vehicle of type 1 must end at the same depot where it starts (i.e., it must be a cycle). A route for a vehicle of type 2 must start and end at different depots (i.e., it must be an open path).

Ideally, if customers i and j are served by the same vehicle of type 1 then they should also be served by the same vehicle of type 2. In the air-transportation example, this would mean that each crew only flies one aircraft. However, this ideal situation is not always possible due to the capacity limits and the different types of routes, and in some cases a crew must change from one aircraft to another one. This case is called *changeover*, and is undesired by the transportation companies, not only because it forces extra work for the crew, but also because a delay in the first flight may affect other two flights. For that reason, a changeover is strongly penalized in the cost function with a big value M in our problem definition. There is also another cost N to be paid for using each vehicle of type 1 in a solution. This value N is usually lower than M.

To illustrate the problem, Fig. 1 shows the optimal solution of an instance with 2 depots (nodes 0 and 19) and 18 customers (nodes 1 to 18). There are two vehicles of type 2 that go from one depot to the other (solid lines). There are two vehicles of type 1 that make circular routes (dashed lines) from depot 0, and three vehicles of the same type that start and end their routes at depot 19. Vehicles of type 1 can serve at most 4 customers, and vehicles of type 2 can serve at most 9. Note that a line in this figure does not represent a space movement, like in the CVRP, but a precedence; for example, in the air-transportation context, a line from i to j means that flight j will be operated after flight i (being the arrival airport of i equal to the departure airport of j). Changeovers occur when, between two consecutive non-depot nodes in a route, there is a dashed line and not a solid line. For example, in Fig. 1, we can observe that there is a changeover between the nodes 8 and 17.

2.1. Mathematical model

We now model the VDSP, starting by defining the decision variables. Variable x^k_{ij} takes value 1 if a vehicle of type k traverses an arc $(i, j) \in A$, and value 0 otherwise. For brevity of notation, we will write $x^k(A')$ instead of $\sum_{(i,j)\in A'} x^k_{ij}$ for each vehicle of type k and each subset of arcs $A' \subseteq A$. Variable y_{ij} is used to indicate a changeover between nodes i and j, i.e., y_{ij} is equal to 1 when $x^1_{ij} = 1$ and $x^2_{ij} = 0$, and 0 otherwise. Variable $w_i \in \mathbb{R}$ represents the position in which customer i is served, and variable $z^k_i \in \mathbb{R}$ determines the number of customers that a vehicle of type k has served immediately after serving customer i.

Then a mathematical formulation for the VDSP is given by:

$$\min \sum_{(i,j)\in A} c^1_{ij} x^1_{ij} + \sum_{(i,j)\in A} c^2_{ij} x^2_{ij} + N \sum_{i\in V_d, j\in V_c} x^1_{ij} + M \sum_{(i,j)\in A} y_{ij} \tag{1}$$

subject to:

$$x^1(\delta^+(i)) = x^1(\delta^-(i)) = 1 \qquad \forall i \in V_c \tag{2}$$

$$x^1(\delta^+(j)) = x^1(\delta^-(j)) \le K^1_j \qquad \forall j \in V_d \tag{3}$$

$$x^1(\delta^+(S)) \ge \sum_{i\in S} (x^1_{0,i} + x^1_{i,n+1}) \qquad \forall S \subseteq V_c : S \neq \emptyset \tag{4}$$