Computers & Operations Research 40 (2013) 2145–2160

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

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

Improved bounds for large scale capacitated arc routing problem

Rafael Martinelli [a,*], Marcus Poggi [a], Anand Subramanian [b]

[a] Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio) – Departamento de Informática – Rua Marquês de São Vicente, 225 – RDC, 4º andar – Gávea – Rio de Janeiro, RJ 22451-900, Brazil

[b] Universidade Federal da Paraíba (UFPB) – Departamento de Engenharia de Produção – Centro de Tecnologia, Campus I – Bloco G, Cidade Universitária – João Pessoa, PB 58051-970, Brazil

ARTICLE INFO

Available online 28 February 2013

Keywords:
Arc routing
Integer programming
Dual ascent
Capacity cuts
Iterated local search

ABSTRACT

The Capacitated Arc Routing Problem (CARP) stands among the hardest combinatorial problems to solve or to find high quality solutions. This becomes even more true when dealing with large instances. This paper investigates methods to improve on lower and upper bounds of instances on graphs with over 200 vertices and 300 edges, dimensions that, today, can be considered of large scale. On the lower bound side, we propose to explore the speed of a dual ascent heuristic to generate capacity cuts. These cuts are next improved with a new exact separation enchained to the linear program resolution that follows the dual heuristic. On the upper bound, we implement a modified Iterated Local Search procedure to Capacitated Vehicle Routing Problem (CVRP) instances obtained by applying a transformation from the CARP original instances. Computational experiments were carried out on the set of large instances generated by Brandão and Eglese and also on the regular size sets. The experiments on the latter allow for evaluating the quality of the proposed solution approaches, while those on the former present improved lower and upper bounds for all instances of the corresponding set.

1. Introduction

The Capacitated Arc Routing Problem (CARP) can be defined as follows. Let $G=(V,E)$ be an undirected graph, where V and E are the vertex and edge set respectively. There is a special vertex called *depot* (usually vertex 0) where a set I of identical vehicles with capacity Q is located. Each edge in E has a cost $c: E \rightarrow \mathbb{Z}^+$ and a demand $d: E \rightarrow \mathbb{Z}_0^+$. Let $E_R = \{e \in E : d_e > 0\}$ be the set of required edges. The objective is to find a set of routes, one for each available vehicle, which minimizes the total traversal cost satisfying the following constraints: (i) every route starts and ends at the depot; (ii) each required edge must be visited exactly once; (iii) the total load of each vehicle must not exceed Q.

This problem can arise in many real life situations. According to Wølhk [1], some of the applications studied in the literature are garbage collection, street sweeping, winter gritting, electric meter reading and airline scheduling.

The CARP is $\mathcal{NP}$-hard and it was first proposed by Golden and Wong in 1981 [2]. Since then, several solution approaches were proposed in the literature involving algorithms based on heuristics, metaheuristics, cutting plane, column generation, branch-and-bound, among others.

In 2003, Belenguer and Benavent [3] proposed a mathematical formulation for the CARP which makes use of two families of cuts as constraints, the *odd-edge cutset cuts* and the *capacity cuts*. With this formulation and other families of cuts, they devised a cutting plane algorithm in order to obtain good lower bounds for well-known CARP instance datasets. Before this work, the best known CARP lower bounds were found mainly by heuristic algorithms.

Since the work of Belenguer and Benavent, the best known lower bounds were found using exact algorithms. In 2004, Ahr [4] devised a mixed-integer formulation using an exact separation of capacity cuts. However, due to memory limitations, the author did not manage to apply his algorithm in all known instances, which illustrates the difficulty in separating such cuts.

The main drawback of the exact approaches is the fact of being prohibitive on larger instances. Up to this date, the larger instance solved to optimality is the *egl-s3-c* from the *eglese* instance dataset, proposed almost 20 years ago by Li [5] and Li and Eglese [6]. This instance has 140 vertices and 190 edges, 159 of these required ones, and it was solved for the first time by Bartolini et al. in 2011 [7] using a cut-and-column based technique combined with a set partitioning approach. Other recent works using exact approaches which solved to optimality instances from *eglese* instance dataset are those of Bode and Irnich [8], which used a cut-first branch-and-price-second exploiting the sparsity of the instances, and Martinelli et al. [9], which used a branch-cut-and-price with non-elementary routes.

In their work of 2008, Brandão and Eglese [10] proposed a new set of CARP instances, called *egl-large*, containing 255 vertices,

* Corresponding author.
E-mail addresses: rmartinelli@inf.puc-rio.br (R. Martinelli), poggi@inf.puc-rio.br (M. Poggi), anand@ct.ufpb.br (A. Subramanian).

0305-0548/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2013.02.013

375 edges and 347 or 375 required edges. They ran the path-scanning heuristic from Golden [11] and compared the results with their deterministic tabu search, giving the first upper bounds for this instance dataset. In 2009, Mei et al. [12] improved these upper bounds using a repair-based tabu search algorithm. To the best of our knowledge, there are no lower bounds reported in the literature for this instance dataset.

The contributions of this paper are twofold: (i) provide a methodology capable of obtaining good lower bounds and (ii) improve the existing upper bounds by means of a heuristic algorithm; both approaches with emphasis on large scale instances. In order to find the first lower bounds for the *egl-large* instance dataset, we devise a *dual ascent heuristic* to speed up a cutting plane algorithm which uses a new exact separation of the capacity cuts and a known exact separation of the odd edge cutset cuts. The upper bounds are found using a known transformation to the Capacitated Vehicle Routing Problem (CVRP) and then applying an *Iterated Local Search* (ILS) based heuristic. We report new improved upper bounds for all 10 instances of the *egl-large* set.

The remainder of the paper is organized as follows. Section 2 presents the mathematical formulation needed for the dual ascent heuristic and the known exact separation algorithms. Section 3 introduces a new exact separation for the capacity cuts. Section 4 describes our dual ascent heuristic and how it generates cuts to hot-start the cutting plane algorithm. Section 5 explains the known transformation to the CVRP and the ILS heuristic. Section 6 presents extensive computational experiments. Finally, conclusions are given in Section 7.

2. Mathematical formulation

2.1. The one-index formulation

In their work, Belenguer and Benavent [3] developed a CARP formulation, usually referred as the *One-Index Formulation* [13]. In contrast to other approaches, this formulation only makes use of variables representing the deadheading of an edge. An edge is deadheaded when a vehicle traverses this edge without servicing it. In addition, all vehicles are aggregated. Due to these simplifications, this formulation is not complete, i.e., it may result in an infeasible solution for the problem. Moreover, even when a given solution is feasible, it is a very hard task to find a complete solution. Nevertheless, these issues do not prevent such formulation of giving very good lower bounds in practice.

For each deadheaded edge e, there is an integer variable z_e representing the number of times the edge e was deadheaded by *any* vehicle. Let $S \subseteq V\backslash\{0\}$ be a subset of vertices not including the depot. We can define $\delta(S)=\{(i,j)\in E : i\in S \wedge j\notin S\}$ as being the set of edges which have one endpoint inside S and the other outside S. Similarly, $\delta_R(S)=\{(i,j)\in E_R : i\in S \wedge j\notin S\}$ is the set of *required* edges which have one endpoint inside S and the other outside S. Analogously, $E(S)=\{(i,j)\in E : i\in S \wedge j\in S\}$ and $E_R(S)=\{(i,j)\in E_R : i\in S \wedge j\in S\}$ are the sets of edges with both endpoints inside S.

Given a vertex set S, with $|\delta_R(S)|$ odd, it is easy to conclude that at least one edge in $\delta(S)$ must be deadheaded because each vehicle entering the set S must leave and return to the depot. This is the principle of the *odd-edge cutset cuts*:

$$\sum_{e\in\delta(S)} z_e \ge 1 \quad \forall S\subseteq V\backslash\{0\},\ |\delta_R(S)|\,\text{odd} \tag{1}$$

Furthermore, we can define a lower bound on the number of vehicles needed to meet the demands in $\delta_R(S)\cup E_R(S)$ as $k(S)=\lceil\sum_{e\in\delta_R(S)\cup E_R(S)} d_e/Q\rceil$. These $k(S)$ vehicles must enter and leave the set S, in such a way that at least $2k(S)-|\delta_R(S)|$ times an edge in $\delta(S)$ will be deadheaded. If this value is positive, we can define the following *capacity cut*:

$$\sum_{e\in\delta(S)} z_e \ge 2k(S)-|\delta_R(S)| \quad \forall S\subseteq V\backslash\{0\} \tag{2}$$

Since the left-hand side of both (1) and (2) are the same, they can be represented in the formulation by only using a single constraint. This can be done by introducing $\alpha(S)$, which is defined as follows:

$$\alpha(S)=\begin{cases}\max\{2k(S)-|\delta_R(S)|,1\} & \text{if } |\delta_R(S)|\,\text{is odd,}\\ \max\{2k(S)-|\delta_R(S)|,0\} & \text{if } |\delta_R(S)|\,\text{is even}\end{cases} \tag{3}$$

These two families of cuts define the one-index formulation:

$$\text{Min} \quad \sum_{e\in E} c_e z_e \tag{4}$$

$$\text{s.t.} \quad \sum_{e\in\delta(S)} z_e \ge \alpha(S) \quad \forall S\subseteq V\backslash\{0\} \tag{5}$$

$$z_e\in\mathbb{Z}_0^+ \quad \forall e\in E \tag{6}$$

The objective function (4) minimizes the cost of the deadheaded edges. Constraints (5) combine cuts (1) and (2). In order to obtain the total cost for the problem, one needs to add the costs of the required edges ($\sum_{e\in E_R} c_e$) to the solution cost.

2.2. Exact odd-degree cutset cuts separation

The exact separation of the odd-degree cutset cuts (1) can be done in polynomial time using the *Odd Minimum Cutset Algorithm* of Padberg and Rao [14]. We believe that the application of the algorithm is not immediate and therefore we decided to provide a brief description of the separation routine, which is as follows.

The odd minimum cutset algorithm creates a *Gomory-Hu Tree* [15] using just the vertices with odd $|\delta_R(\{v\})|$, called *terminals*. This tree represents a *maximum flow tree*, i.e., the maximum flow of any pair of vertices is represented on this tree. In order to obtain the maximum flow between a pair of vertices, one only needs to find the least cost edge on the unique path between these two vertices. This edge also represents the minimum cut between them. Hence, to determine a violated odd-degree cutset cut, one needs to find any edge with a value less than one. This can be done during the execution of the algorithm, but we prefer to run it until the end to find as many violated cuts as possible.

This whole operation can be done running at most $|V|-1$ times any maximum flow algorithm. In this work we use the *Edmonds–Karp Algorithm* [16], which takes $\mathcal{O}(|V|\cdot|E|^2)$, resulting in a total complexity of $\mathcal{O}(|V|^2\cdot|E|^2)$.

2.3. Ahr's exact capacity cut separation

The only exact separation routine for the capacity cuts available in the CARP literature was proposed by Ahr [4] in 2004. This algorithm runs a mixed-integer formulation several times, one for each possible number of vehicles. This approach was inspired on the exact separation of the capacity cuts for the CVRP proposed by Fukasawa et al. [17]. In Ahr's work, this separation was used to identify violated cuts on a complete formulation for the CARP. As we only wish to separate the cuts, we changed the objective function of the mixed-integer formulation to use it with the one-index formulation.

The formulation is composed by three types of variables. The first one is the binary variable h_e, $\forall e\in E$, which is 1 when exactly one endpoint of e is inside the cut (what we call *cut edge*) and 0 otherwise. The second variable is the binary variable f_e, $\forall e\in E$, which is 1 when both endpoints of e are inside the cut (called *inner edge*) and 0 otherwise. The last variable is the binary variable s_i, $\forall i\in V$, which is 1 if vertex i is inside the cut and