Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 177 (2007) 102–115
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Discrete Optimization

Revisiting dynamic programming for finding optimal subtrees in trees ☆

Christian Blum

ALBCOM, LSI, Universitat Politècnica de Catalunya, Jordi Girona 1-3, Campus Nord, Omega 112, 08034 Barcelona, Spain

Received 21 January 2005; accepted 9 November 2005
Available online 19 January 2006

Abstract

In this paper we revisit an existing dynamic programming algorithm for finding optimal subtrees in edge weighted trees. This algorithm was sketched by Maffioli in a technical report in 1991. First, we adapt this algorithm for the application to trees that can have both node and edge weights. Second, we extend the algorithm such that it does not only deliver the values of optimal trees, but also the trees themselves. Finally, we use our extended algorithm for developing heuristics for the k-cardinality tree problem in undirected graphs G with node and edge weights. This NP-hard problem consists of finding in the given graph a tree with exactly k edges such that the sum of the node and the edge weights is minimal. In order to show the usefulness of our heuristics we conduct an extensive computational analysis that concerns most of the existing problem instances. Our results show that with growing problem size the proposed heuristics reach the performance of state-of-the-art metaheuristics. Therefore, this study can be seen as a cautious note on the scaling of metaheuristics.

Keywords: Heuristics; Dynamic programming; Combinatorial optimization

☆ This work was supported by "SegraVis" (grant HPRN-CT-2002-00275), a Research Training Network funded by the Improving Human Potential program of the CEC. The information provided is the sole responsibility of the authors and does not reflect the Community's opinion. The Community is not responsible for any use that might be made of data appearing in this publication. This work was also supported by the Spanish CICYT project TRACER (grant TIC-2002-04498-C05-03), and by the "Juan de la Cierva" program of the Spanish Ministry of Science and Technology of which Christian Blum is a post-doctoral research fellow.

E-mail address: cblum@lsi.upc.edu

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2005.11.005

1. Introduction

The k-cardinality tree (KCT) problem—also referred to as the *k-minimum spanning tree* (k-MST) problem, or just the *k-tree* problem—is an NP-hard [15,25] combinatorial optimization problem which generalizes the well-known minimum weight spanning tree problem. Given is an undirected graph $G = (V, E)$ and a cardinality $k < |V|$. Two versions of the problem are studied in the literature. In the first version of the problem the given graph G has positive weights on the edges, whereas in the second version G has positive weights on the nodes. In both versions, the problem consists of finding a subtree in G with exactly k edges, such that the sum of the weights is minimal. The problem was first described in [21] and it has gained considerable interest in recent years due to various applications, e.g. in oil-field leasing [20], facility layout [16,17], open pit mining [26], matrix decomposition [7,8], quorum-cast routing [11] and telecommunications [19].

In this paper we deal with a generalized problem version in which the given graph G can have both node and edge weights. More formally, let $G = (V, E)$ be a graph with a weight function $w_E : E \to \mathbb{N}$ on the edges and a weight function $w_V : V \to \mathbb{N}$ on the nodes. We denote the weight of a node v by $w_V(v)$ (or just w_v), and the weight of and edge e by $w_E(e)$ (or just w_e). Furthermore, we denote by $\mathcal{T}_k$ the set of all k-cardinality trees in G. Then, the problem consists of finding a k-cardinality tree $T_k \in \mathcal{T}_k$ that minimizes

$$f(T_k) = \left(\sum_{e \in E(T_k)} w_e \right) + \left(\sum_{v \in V(T_k)} w_v \right). \tag{1}$$

In this equation, as well as in the rest of the paper, when given a tree T, $E(T)$ denotes the set of edges of T, and $V(T)$ the set of nodes of T.

The edge weighted version of the KCT problem was first tackled by exact approaches [18,11,28] and heuristics [14,13,11]. The best ones of these heuristics are based on a polynomial time dynamic programming algorithm [24] that finds the best k-cardinality tree in a graph that is itself a tree. However, the interest in heuristics was quickly lost and research focused on the development of more appealing metaheuristics [6]. Among these approaches are two evolutionary computation (EC) approaches [2,4], three tabu search (TS) methods [3,22,4], different variations of variable neighborhood search (VNS) [29], and two ant colony optimization (ACO) approaches [10,4]. Two sets of benchmark instances exist. One was introduced for the empirical evaluation of the VNS-based approaches in [29], and the other one for the metaheuristics proposed in [4]. Generally it is assumed that the variable neighborhood decomposition search (VNDS) algorithm proposed in [29] is the state-of-the-art method for the first set, and the ACO algorithm proposed in [10] for the second set.

Much less research efforts were directed at the node weighted KCT problem. Simple greedy as well as dual greedy based heuristics were proposed in [14], and the first metaheuristic approaches (TS and EC) were presented in [5]. The currently best metaheuristic is a recent VNDS [9]. In the same paper the only existing benchmark set for the node weighted KCT problem is introduced.

Our contribution: In this paper we revisit the dynamic programming (DP) algorithm of Maffioli for finding the best k-cardinality tree in an edge weighted graph that is itself a tree. The idea for this algorithm, which has polynomial running time, was sketched by Maffioli in 1991 in a technical report [24]. Several heuristics for the edge weighted KCT problem are based on this algorithm [14]. These heuristics first construct—in different ways—a spanning tree of the given graph. Then dynamic programming is applied to obtain the best k-cardinality tree in the previously constructed spanning tree. In [14]—because of the low computational power of the time—these heuristics were only applied to graphs of up until 30 nodes, which is very small compared to the biggest graphs from the nowadays available benchmark sets (i.e., 5000 nodes and 50 000 edges). Furthermore, the DP algorithm as described in [24] does not solve the problem, because it only outputs the objective function value of an optimal k-cardinality tree and not the tree itself. Possibly due to these reasons, the heuristics proposed in [14] never attracted much attention, until Urošević et al. in