Discrete Optimization 20 (2016) 40–61

Contents lists available at ScienceDirect

Discrete Optimization

www.elsevier.com/locate/disopt

Covering problems in edge- and node-weighted graphs

Takuro Fukunaga*

National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan
JST, ERATO, Kawarabayashi Large Graph Project, Japan

ARTICLE INFO

Article history:
Received 30 March 2015
Received in revised form 27 November 2015
Accepted 11 March 2016

MSC:
05C85
90C27

Keywords:
Edge dominating set
Multicut
Primal–dual algorithm

ABSTRACT

This paper discusses the graph covering problem in which a set of edges in an edge- and node-weighted graph is chosen to satisfy some covering constraints while minimizing the sum of the weights. In this problem, because of the large integrality gap of a naive linear programming (LP) relaxation, LP rounding algorithms based on the relaxation yield poor performance. Here we propose a stronger LP relaxation for the graph covering problem. The proposed relaxation is applied to designing primal–dual algorithms for two fundamental graph covering problems: the prize-collecting edge dominating set problem and the multicut problem in trees. Our algorithms are an exact polynomial-time algorithm for the former problem, and a 2-approximation algorithm for the latter problem. These results match the currently known best results for purely edge-weighted graphs.

1. Introduction

1.1. Motivation

Choosing a set of edges in a graph that optimizes some objective function under constraints on the chosen edges constitutes a typical combinatorial optimization problem and has been investigated in many varieties. For example, the spanning tree problem seeks an acyclic edge set that spans all nodes in a graph, the edge cover problem finds an edge set such that each node is incident to at least one edge in the set, and the shortest path problem selects an edge set that connects two specified nodes. In this paper, we call the generic problem including those problems by the *graph covering problem*, which we will formally define later.

Most previous investigations of the graph covering problem have focused on edge-weighted graphs, over which the objective of the problem is to minimize the sum of the weights assigned to the edges in the solution. To our knowledge, when node weights have been considered in the graph covering problem, it has been restricted to the Steiner tree problem or its generalizations, possibly because the inclusion of

* Correspondence to: National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan.
E-mail address: takuro@nii.ac.jp.

http://dx.doi.org/10.1016/j.disopt.2016.03.001

node weights greatly complicates the problem. For example, the Steiner tree problem in edge-weighted graphs can be approximated within a constant factor (the best currently known approximation factor is 1.39 [1,2]). On the other hand, the Steiner tree problem in node-weighted graphs is known to extend the set cover problem (see [3]), indicating that achieving an approximation factor of $o(\log |V|)$ is NP-hard. The literature is reviewed in Section 2. As revealed later, the inclusion of node weights generalizes the set cover problem in numerous fundamental problems.

However, from another perspective, node weights can introduce a rich structure into the above problems, and provide useful optimization problems. In fact, the node-weighted network design problem (including the node-weighted Steiner tree problem) is considered in the field of networking motivated by an application for wireless networks [4,5]. The objective function counts the weight of a node only once, even if the node is shared by multiple edges. Hence, the objective function defined by node weights includes a certain subadditivity, which cannot be captured by edge weights.

The aim of the present paper is to present an algorithmic technique for dealing with the graph covering problem in node-weighted graphs. We focus on algorithms based on linear programming (LP) relaxations. Many algorithms for combinatorial optimization problems are typically designed using LP relaxations. However, in the graph covering problem with node-weighted graphs, the integrality gap of naive relaxations may be excessively large. Therefore, we propose tighter LP relaxations that preclude unnecessary integrality gaps. Unfortunately, there is no generic algorithm that rounds an optimal solution for this relaxation. To demonstrate the usefulness of the relaxation, we discuss its integrality gap in two fundamental graph covering problems: the edge dominating set problem and multicut problem in trees. We prove upper bounds on the integrality gap by designing primal–dual algorithms for both problems. The approximation factors of our proposed algorithms match the current best approximations in purely edge-weighted graphs.

1.2. Problem definitions

The *edge dominating set (EDS) problem* covers edges by choosing adjacent edges in undirected graphs. For any edge e, let $\delta(e)$ denote the set of edges that share end nodes with e, including e itself. We say that an edge e *dominates* another edge f if $f \in \delta(e)$, and a set F of edges dominates an edge f if F contains an edge that dominates f. Given an undirected graph $G = (V, E)$, a set of edges is called an EDS if it dominates each edge in E. The EDS problem seeks to minimize the weight of the EDS. In other words, the EDS problem is the problem of finding a minimum weight edge set F such that $F \cap \delta(e) \neq \emptyset$ for each $e \in E$.

In the *multicut problem*, an instance specifies an undirected graph $G = (V, E)$ and demand pairs $(s_1, t_1), \ldots, (s_k, t_k) \in V \times V$. A *multicut* is an edge set C whose removal from G disconnects the nodes in each demand pair. This problem seeks a multicut of minimum weight. Let $\mathcal{P}_i$ denote the set of paths connecting s_i and t_i. The multicut problem is equivalent to finding a minimum weight edge set C such that $C \cap P \neq \emptyset$ for each $P \in \bigcup_{i=1}^{k} \mathcal{P}_i$.

Our proposed algorithms for solving these problems assume that the given graph G is a tree. In fact, our algorithms are applicable to the prize-collecting versions of these problems, which additionally specifies a penalty function $\pi \colon \mathcal{E} \to \mathbb{R}_+$, where $\mathcal{E} = \{\delta(e) \colon e \in E\}$ in the EDS problem and $\mathcal{E} = \bigcup_{i=1}^{k} \mathcal{P}_i$ in the multicut problem. In this scenario, an edge set F is a feasible solution even if $F \cap C = \emptyset$ for some $C \in \mathcal{E}$, but imposes a penalty $\pi(C)$. The objective is to minimize the sum of $w(F)$, $w(V(F))$, and the penalty $\sum_{C \in \mathcal{E} : F \cap C = \emptyset} \pi(C)$. The prize-collecting versions of the EDS and multicut problems are referred to as the *prize-collecting EDS problem* and the *prize-collecting multicut problem*, respectively. The prize-collecting multicut problem can be easily reduced to the multicut problem if the graph is a tree, which we will explain in Section 5.