Applied Mathematical Modelling 34 (2010) 1042–1050

Contents lists available at ScienceDirect

Applied Mathematical Modelling

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

Solving a cut problem in bipartite graphs by linear programming: Application to a forest management problem

Alain Billionnet

ENSIIE, 1 square de la résistance, 91025 Evry cedex, France

ARTICLE INFO

Article history:
Received 10 October 2008
Received in revised form 26 June 2009
Accepted 24 July 2009
Available online 3 August 2009

Keywords:
Bipartite graph
Cut
Linear programming
Land allocation modelling
Wildlife habitat
Experiments

ABSTRACT

Given a bipartite graph $G = (V, E)$, a weight for each node, and a weight for each edge, we consider an extension of the MAX-CUT problem that consists in searching for a partition of V into two subsets V_1 and V_2 such that the sum of the weights of the edges from E that have one endpoint in each set plus the sum of the weights of the nodes from V that are in V_1, is maximal. We prove that this problem can be modeled as a linear program (with real variables) and therefore efficiently solved by standard algorithms. Then, we show how this result can be applied to model a land allocation problem by a 0–1 linear program. This problem consists in determining the cells of a land area, divided into a matrix of identical square cells, which must be harvested and the cells which must be left in old growth so that the weighted sum of the expected populations of some species is maximized. Some computational results are presented to illustrate the efficiency of the method.

1. Introduction

Integer linear programming is a classical tool in operations research that can be applied to a lot of optimization problems. Many studies have already shown that (integer) linear programming was a useful tool to optimize a management objective in an ecosystem (see, for example [1–4]). The technique is known and well-tried but must be carefully implemented since some formulations may require a prohibitive computation time (see, for example [5–7]). Indeed, several models are often possible for the same problem, and the choice of a good one is important. If a particular formulation of a problem fails, another one may appear more efficient in practice, as we will see when considering a cut problem in a bipartite graph. We prove that a classical 0–1 linear model of this cut problem, of quadratic nature, can be solved by linear programming (in real variables). So, it is possible to solve large sized instances of this problem only using a linear programming tool, and finally a standard, commercially available, software. Then we use this formulation to efficiently model a land allocation problem by 0–1 linear programming. This problem, presented in [1] arises in land allocation modelling and consists in determining the cells of a land area, divided into a matrix of identical square cells, which must be harvested and the cells which must be left in old growth so that the weighted sum of the expected populations of three species is maximized. The species S1 prefers recently harvested areas contrary to the species S2, and the species S3 population is determined as a linear function of the amount of edge between harvested cells and non-harvested cells. Edges concern the interface between forest and non-forest elements and have environmental characteristics that are different from either of the adjacent elements. We suppose that the species S3 uses edges as its habitat. In Section 2, we recall some properties and mathematical programming formulations of the well-known MAX-CUT problem. In Section 3, we present the considered cut problem, a variant of MAX-CUT, and we prove that, in bipartite graphs, it can be solved (in polynomial time) by linear programming. In Section 4, we recall the forest management problem presented in [1], and a numerical

E-mail address: Alain.Billionnet@ensiie.fr

0307-904X/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.apm.2009.07.014

example in Section 5. In Section 6, we give the linear 0–1 programming model proposed in [1]. In Section 7, we propose another linear 0–1 programming formulation based on the results of Section 3. In Section 8, we report comparisons of the two formulations, from a computational time viewpoint, which show the efficiency of the second formulation to compute the species S3 population. Section 9 is a conclusion.

2. The maximum cut problem in graphs: MAX-CUT

Given a graph $G = (V, E)$, and a weight $w(e)$ for each edge $e \in E$, the well-known MAX-CUT problem consists in determining a partition (V_1, V_2) of V such that the sum of the weights of the edges from E that have one endpoint in each set is maximal. MAX-CUT is very easy to state but hard to solve. This problem is among the first problems whose NP-hardness was established in the famous paper by Karp [8]. Note that the problem remains NP-hard when the weights of all edges are one [9,10], and if, in addition, no vertex has degree exceeding 3 [11]. Using a semidefinite relaxation, Goemans and Williamson [12] achieve an approximation ratio of 0.87856 for this difficult combinatorial optimization problem. MAX-CUT has applications in many fields (see, for example [13]) and we show in this paper an application of a variant of MAX-CUT to spatial optimization in ecosystems. There are several classes of graphs for which MAX-CUT is solvable in polynomial time. These include planar graphs [14,15], weakly bipartite graphs with non-negative edge weights [16], graphs without K_5 minors [17]. Note that all planar graphs, and more generally all graphs not contractible to K_5 are weakly bipartite [18]. The MAX-CUT problem is also solvable in polynomial time in cographs when the weights of all edges are one [18]. A cograph is a graph which does not contain a path on four vertices (and hence of length 3) as an induced graph. In a bipartite graph the MAX-CUT problem is trivial. On the contrary, the problem is NP-hard on complements of bipartite graphs [19] cited by Poljak and Tuza [18]. The solution of MAX-CUT can be approached by mathematical programming. Using the variables $x_i \in \{-1, 1\}$ for every vertex in the graph where $x_i = 1$ if $v_i \in V_1$ and $x_i = -1$ if $v_i \in V_2$, an integer quadratic programming formulation of MAX-CUT is given by (P1).

$$(\text{P1}) \quad \begin{cases} \max \frac{1}{2} \sum_{(i,j)\in S} w_{ij}(1 - x_i x_j), \\ \text{s.t.} \mid x_i \in \{-1, 1\}, \quad i = 1, \ldots, n, \quad (1.1) \end{cases}$$

where n is the number of nodes in G and $S = \{(i,j) \in \{1, \ldots, n\}^2 : [v_i, v_j] \in E, i < j\}$. We can also associate the Boolean variables x_i with every vertex of G. In this case $x_i = 1$ if $v_i \in V_1$ and $x_i = 0$ if $v_i \in V_2$, and a 0–1 quadratic programming formulation of MAX-CUT is given by (P2).

$$(\text{P2}) \quad \begin{cases} \max \sum_{(i,j)\in S} w_{ij}(x_i + x_j - 2x_i x_j), \\ \text{s.t.} \mid x_i \in \{0, 1\}, \quad i = 1, \ldots, n. \quad (2.1) \end{cases}$$

It is also easy to formulate MAX-CUT as the 0–1 linear program (P3).

$$(\text{P3}) \quad \begin{cases} \max \sum_{(i,j)\in S} w_{ij}(x_i + x_j - 2y_{ij}), \\ \text{s.t.} \begin{cases} y_{ij} \geqslant x_i + x_j - 1, & (i,j) \in S, \quad (3.1) \\ y_{ij} \geqslant 0, & (i,j) \in S, \quad (3.2) \\ x_i \in \{0, 1\}, & i = 1, \ldots, n. \quad (3.3) \end{cases} \end{cases}$$

(P3) can be viewed as a classical linearization of the 0–1 quadratic program (P2): the product $x_i x_j$ is replaced by the variable y_{ij} and the constraints $y_{ij} \geqslant x_i + x_j - 1$ and $y_{ij} \geqslant 0$ are added to force the equality $y_{ij} = x_i x_j$ in an optimal solution (see, for example [20]). Kahruman et al. [21] propose another 0–1 linear programming formulation of MAX-CUT:

$$(\text{P4}) \quad \begin{cases} \max \sum_{(i,j)\in S} w_{ij} y_{ij}, \\ \text{s.t.} \begin{cases} y_{ij} - x_i - x_j \leqslant 0, & (i,j) \in S, \quad (4.1) \\ y_{ij} + x_i + x_j \leqslant 2, & (i,j) \in S, \quad (4.2) \\ x_i \in \{0, 1\}, & i = 1, \ldots, n. \quad (4.3) \end{cases} \end{cases}$$

From a practical viewpoint, an efficient method is presented in [22] for finding exact solutions of the MAX-CUT problem. The authors use a semidefinite relaxation combined with triangle inequalities. Their experiments show that exact solutions are obtained in a reasonable time for any instance of size up to $n = 100$, independent of the graph density.

3. A variant of MAX-CUT: N-MAX-CUT

In the classical MAX-CUT problem, the objective function is based on the weights of the edges. In the problem that we consider, the objective function not only incorporates the weights of the edges but also the weights of the nodes. Given a graph $G = (V, E)$, a weight for each node, and a weight for each edge, we consider an extension of the MAX-CUT problem,