Discrete Optimization 16 (2015) 76–89

Contents lists available at ScienceDirect

Discrete Optimization

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

Local search inequalities

Giuseppe Lancia *, Franca Rinaldi, Paolo Serafini

Dipartimento di Matematica e Informatica - University of Udine, via delle Scienze 206, Udine, Italy

ARTICLE INFO

Article history:
Received 7 October 2014
Received in revised form 14 February 2015
Accepted 19 February 2015
Available online 13 March 2015

Keywords:
Integer linear programming
Local search
Branch and cut

ABSTRACT

We describe a general method for deriving new inequalities for integer programming formulations of combinatorial optimization problems. The inequalities, motivated by local search algorithms, are valid for all optimal solutions but not necessarily for all feasible solutions. These local search inequalities can help in either pruning the search tree at some nodes or in improving the bound of the LP relaxations.

1. Optimization and local search

One of the most effective ways to solve an NP-hard combinatorial optimization problem is to formulate it as an integer program, which is in turn solved by branch and bound [1]. Let the formulation be

$$z_{IP} := \min\{c\,x : x \in S\} \tag{1}$$

where $S = \{x : A\,x \geq b, x \geq 0, x \in \mathbb{Z}^n\}$. Moreover, let $P = \{x : A\,x \geq b, x \geq 0\}$ and $P^I = \text{conv}(S)$.

Valid inequalities (also called cuts) can be added to (1) to strengthen the quality of the LP bound to be used in the branch-and-bound. The process of solving a combinatorial optimization problem with the addition of valid cuts is called *branch-and-cut* [2].

Local search is a general framework for finding good (not necessarily optimal) solutions of an optimization problem [3,4]. In local search, a neighborhood function $\mathcal{N}(s)$ is specified, which, for a feasible solution s, defines a set of feasible solutions "close" to s. A *local optimum* is a solution s^* such that $c\,s^* \leq c\,s$ for all $s \in \mathcal{N}(s^*)$. Clearly, an optimal solution of the problem is also a local optimum for each possible neighborhood but not vice-versa. Local search heuristics usually work by quickly finding as many local optima as possible, and then returning the best one.

The results presented in this paper are based on the following observation:

> *Given a local search neighborhood $\mathcal{N}$, a global optimum of the problem must also be a local optimum for $\mathcal{N}$. This is therefore an additional constraint on the global optimum.*

If it is possible to express the above constraint via linear inequalities, we call each such linear constraint a *local search inequality* (LSI). Basically, local search inequalities are constraints saying that, for each move which changes a feasible solution x into a feasible solution $x' \in \mathcal{N}(x)$, it must be $c\,x' \geq c\,x$. These constraints are valid for local optima (and hence for global optima), but may be violated by some other feasible solutions in P^I. Therefore, they cut *through* the set P^I and are not valid inequalities in the usual sense.

We mention that the idea of using inequalities that are not valid for all feasible solutions but only for some subsets of solutions can be also found in other settings. As examples of such inequalities, we cite the *logic cuts* which can be found in [5,6] and the *conditional inequalities* appearing in [7].

* Corresponding author.
E-mail addresses: giuseppe.lancia@uniud.it (G. Lancia), franca.rinaldi@uniud.it (F. Rinaldi), paolo.serafini@uniud.it (P. Serafini).

http://dx.doi.org/10.1016/j.disopt.2015.02.003
1572-5286/

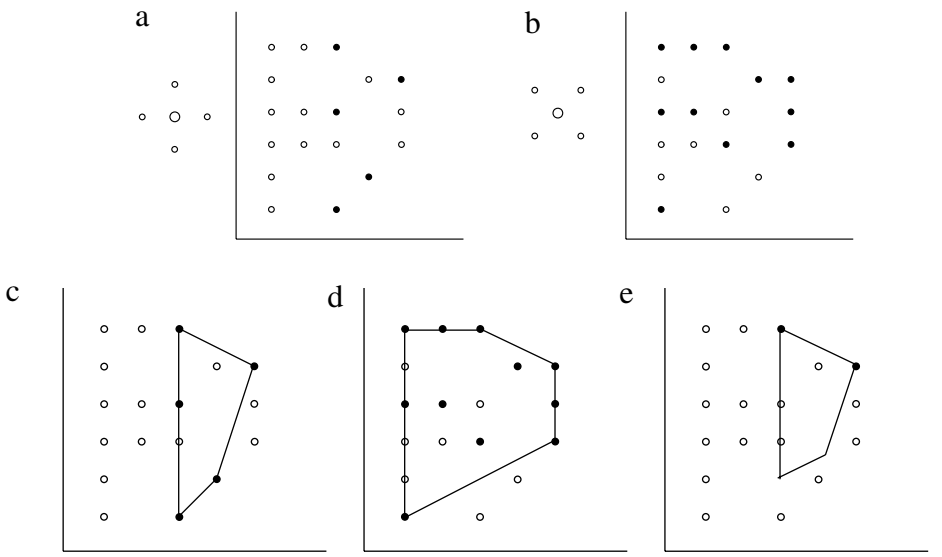

Fig. 1. A feasible set with two neighborhood functions $\mathcal{N}^1$ (a) and $\mathcal{N}^2$ (b). In (a) neighbor points are obtained by moving vertically or horizontally; in (b) by moving diagonally. The objective function is $\max x_1 + x_2$ and the local optima are the black dots. $P^{\mathcal{N}_1}$ is shown in (c) and $P^{\mathcal{N}_2}$ is shown in (d). $P^{\mathcal{N}_1} \cap P^{\mathcal{N}_2}$ is shown in (e) which has only two local optima points, one of which is the global optimum.

Given a neighborhood function $\mathcal{N}$, let $X(\mathcal{N})$ be the set of incidence vectors of all local optima for $\mathcal{N}$, and let $P^{\mathcal{N}} = \text{conv}(X(\mathcal{N}))$. Then, a LSI is nothing but a valid inequality for $P^{\mathcal{N}}$. The polytope $P^{\mathcal{N}}$ is contained in P^I, and if k distinct neighborhood functions are considered, the optimal solutions in P^I are contained in $P^{\mathcal{N}_1} \cap P^{\mathcal{N}_2} \cap \cdots \cap P^{\mathcal{N}_k}$. In Figs. 1(a) and (b) a feasible set is shown with two neighborhood functions $\mathcal{N}^1$ and $\mathcal{N}^2$. The neighborhood functions are schematically shown at the left of the figures (in the first case neighbor points are obtained moving vertically or horizontally; in the second case moving diagonally). If the objective function is $\max x_1 + x_2$ then the local optima are the black dots. In Figs. 1(c) and (d) we can see $P^{\mathcal{N}_1}$ and $P^{\mathcal{N}_2}$. In Fig. 1(e) we see $P^{\mathcal{N}_1} \cap P^{\mathcal{N}_2}$ which has only two local optima points, one of which is the global optimum.

From our computational experiments, we have noticed that $X(\mathcal{N}_1) \cap X(\mathcal{N}_2) \cap \cdots \cap X(\mathcal{N}_k)$ is typically much smaller than S, even for $k = 2, 3$. For example, consider the problem of finding the optimal TSP tour over the first 13 nodes of the TSPLIB instance `fri26`. We computed the local optima for three simple neighborhood functions defined by the moves that exchange two consecutive cities along the tour ($\mathcal{N}_1$), exchange any two cities along the tour ($\mathcal{N}_2$) and remove two edges from the tour and reconnect the two resulting paths ($\mathcal{N}_3$). Then, among the 239,500,800 tours, there are 432,507 local optima for $\mathcal{N}_1$, 7,293 local optima for $\mathcal{N}_2$ and 3 local optima for $\mathcal{N}_3$. Moreover, $X(\mathcal{N}_1) \cap X(\mathcal{N}_2) \cap X(\mathcal{N}_3)$ has just 2 elements, and they are both global optima.

Local search inequalities are, by their nature, very general, since they apply whenever a local optimality condition can be expressed by linear inequalities. In this paper, we will focus on the Traveling Salesman (TSP), on the Maximum Cut (Max-Cut) and on the Maximum Satisfiability problems (Max-SAT).

In a sense, LSIs are "formulation-independent", i.e., they can be added to any formulation whose variables include the variables appearing in the LSIs. For instance, there are several distinct formulations for the TSP which have, among their variables, binary variables x_{ij} associated to the edges of the graph (see [8]). Then, if we have a set of LSIs involving only the edge variables, they could be added to any of these formulations.

Another nice property of LSIs is that, as long as the neighborhood function is relatively simple (such as exchanging the order of two elements in a permutation), the number of corresponding inequalities is fairly small, and LSIs can be directly added to the formulation without the need of a separation algorithm. We will see examples of this type later on. On the other hand, for more complex neighborhood functions, it may be the case that there is an exponential number of LSIs, but still they can be dealt with in polynomial time via a separation algorithm. We will see examples of this type as well.

The remainder of the paper is organized as follows. In Section 2 we describe two neighborhoods for the Symmetric TSP and some corresponding LSIs. In Section 3 and Section 4 we describe LSIs for the Max-Cut and the Max-SAT problems, respectively. In Section 5 we report the results of some computational experiments. Some conclusions are drawn in Section 6.

Notation. We will adopt the following notation. For a graph $G = (V, E)$ and a set of nodes $S \subset V$, we denote by $\delta(S)$ the set of edges with one endpoint in S and the other in $V \setminus S$. With a slight abuse of notation, we write $\delta(v)$ instead of $\delta(\{v\})$ when $|S| = 1$. By $N(v)$ we denote the set of neighbors of $v \in V$, i.e. $N(v) := \{u : uv \in E\}$. If x are variables of a linear program with indices in a set I, and $J \subseteq I$, by $x(J)$ we denote the sum $\sum_{i \in J} x_i$.