Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

Tobias Achterberg[1] Stefan Heinz Thorsten Koch

Counting solutions of integer programs using unrestricted subtree detection

[1] ILOG Deutschland, Ober-Eschbacher Str. 109, 61352 Bad Homburg, Germany

Supported by the DFG Research Center Matheon *Mathematics for key technologies* in Berlin.

ZIB-Report 08-09 (February 2008)

Counting solutions of integer programs using unrestricted subtree detection

Tobias Achterberg[1], Stefan Heinz[2*], and Thorsten Koch[2]

[1] ILOG Deutschland, Ober-Eschbacher Str. 109, 61352 Bad Homburg, Germany
tachterberg@ilog.de
[2] Zuse Institute Berlin, Takustr. 7, 14195 Berlin, Germany
{heinz,koch}@zib.de

Abstract. In the recent years there has been tremendous progress in the development of algorithms to find optimal solutions for integer programs. In many applications it is, however, desirable (or even necessary) to generate *all* feasible solutions. Examples arise in the areas of hardware and software verification and discrete geometry.
In this paper, we investigate how to extend branch-and-cut integer programming frameworks to support the generation of all solutions. We propose a method to detect so-called *unrestricted subtrees*, which allows us to prune the integer program search tree and to collect several solutions simultaneously. We present computational results of this *branch-and-count* paradigm which show the potential of the unrestricted subtree detection.

1 Introduction

In the last decades much progress has been made in finding optimal solutions to integer linear programs (IP) [6]. Recently, more attention has been given to the task of finding all feasible solutions to a given IP, since it arises in applications, for instance, in the context of hardware and software verification and the analysis of polyhedra (see De Loera et al. [9] and references therein). Furthermore, for IP problems that evolve from industry applications, it is desirable to find multiple or even all optimal solutions as discussed in [8].

A common way to solve IP counting or enumeration problems is to transform them into an equivalent binary representation and use specialized solvers. For Boolean satisfiability instances an algorithm for counting solutions is introduced in [13]. A method based on binary decision diagrams is stated in [4]. This algorithm is capable of counting or enumerating all feasible solutions of binary linear programs (BP), which are IPs containing only binary variables. Alternative methods for these type of problems are given in [7] and [10]. Both approaches make use of a search tree. The first one additionally uses linear programming (LP) relaxations to detect infeasible subproblems.

* Supported by the DFG Research Center Matheon *Mathematics for key technologies* in Berlin.