Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

THORSTEN KOCH, TOBIAS ACHTERBERG[1],
ERLING ANDERSEN[2], OLIVER BASTERT[3],
TIMO BERTHOLD[⋆], ROBERT E. BIXBY[4],
EMILIE DANNA[5], GERALD GAMRATH,
AMBROS M. GLEIXNER, STEFAN HEINZ[⋆],
ANDREA LODI[6], HANS MITTELMANN[7],
TED RALPHS[8], DOMENICO SALVAGNIN[9],
DANIEL E. STEFFY, KATI WOLTER[⋆⋆]

MIPLIB 2010

Mixed Integer Programming Library version 5

http://miplib.zib.de/

[1] IBM Deutschland, Böblingen, Germany
[2] MOSEK, Copenhagen, Denmark
[3] FICO, Munich, Germany
[4] Gurobi, Houston, USA
[5] Google, Mountain View, USA
[6] Università di Bologna, Bologna, Italy
[7] Arizona State University, Tempe, USA
[8] Lehigh University, Bethlehem, USA
[9] Università degli Studi Padova, Padua, Italy

[⋆] Supported by the DFG Research Center MATHEON *Mathematics for key technologies* in Berlin.
[⋆⋆] Funded by DFG Priority Program 1307 “Algorithm Engineering”.

ZIB-Report 10-31

MIPLIB 2010

Mixed Integer Programming Library version 5

Thorsten Koch · Tobias Achterberg · Erling Andersen · Oliver Bastert · Timo Berthold · Robert E. Bixby · Emilie Danna · Gerald Gamrath · Ambros M. Gleixner · Stefan Heinz · Andrea Lodi · Hans Mittelmann · Ted Ralphs · Domenico Salvagnin · Daniel E. Steffy · Kati Wolter

Abstract This paper reports on the fifth version of the Mixed Integer Programming Library. The MIPLIB 2010 is the first MIPLIB release that has been assembled by a large group from academia and from industry, all of whom work in integer programming. There was mutual consent that the concept of the library had to be expanded in order to fulfill the needs of the community. The new version comprises 361 instances sorted into several groups. This includes the main *benchmark* test set of 87 instances, which are all solvable by today's codes, and also the *challenge* test set with 164 instances, many of which are currently unsolved. For the first time, we include scripts to run automated tests in a predefined way. Further, there is a solution checker to test the accuracy of provided solutions using exact arithmetic.

Keywords Mixed Integer Programming · Problem Instances · IP · MIP · MIPLIB

Mathematics Subject Classification (2000) 90C11 · 90C10 · 90C90

Thorsten Koch, Timo Berthold, Gerald Gamrath, Ambros M. Gleixner, Stefan Heinz, Daniel E. Steffy, Kati Wolter
Zuse Institute Berlin, Takustr. 7, 14195 Berlin, Germany
E-mail: {koch,berthold,gamrath,gleixner,heinz,steffy,wolter}@zib.de

Tobias Achterberg
IBM Deutschland, E-mail: achterberg@de.ibm.com

Erling Andersen
MOSEK, E-mail: e.d.andersen@mosek.com

Oliver Bastert
FICO, E-mail: oliverbastert@fico.com

Robert E. Bixby
Gurobi, E-mail: bixby@gurobi.com

Emilie Danna
Google, E-mail: edanna@google.com

Andrea Lodi
University of Bologna, E-mail: andrea.lodi@unibo.it

Hans Mittelmann
Arizona State University, E-mail: mittelmann@asu.edu

Ted Ralphs
Lehigh University, E-mail: ted@lehigh.edu

Domenico Salvagnin
Università degli Studi di Padova, E-mail: salvagni@dei.unipd.it

1 Introduction

The MIPLIB is now going into its fifth incarnation. Starting in 1992 with the first two versions by Bixby, Boyd, and Indovina [23], the update to MIPLIB 3 by Bixby, McZeal, Ceria, and Savelsbergh [24] in 1996, and the compilation of MIPLIB 2003 by Achterberg, Koch, and Martin [3], we have finally arrived at MIPLIB 2010. The motivation for this update is the same as in the previous versions: the continuous progress in the field of mixed integer programming.

A *mixed integer (linear) program* (MIP) is an optimization problem in which a linear objective function is minimized subject to linear constraints over real- and integer-valued variables. For details on mixed integer programming, see, e. g., [69, 106]. The MIPLIB is a diverse collection of challenging real-world MIP instances from various academic and industrial applications suited for benchmarking and testing of MIP solution algorithms.

In this paper, we provide a detailed description of the instances in MIPLIB 2010, including their origin, information on the coefficient matrices, and the types of constraints and variables they contain. The complete library is available online at `http://miplib.zib.de`, where we have collected further information, such as references to papers that have used the MIPLIB as a test set for their algorithms. If available, we also provide a problem description in an algebraic modeling language, such as AMPL [57] or ZIMPL [70]. In addition to the role of MIPLIB as a test suite for integer programming algorithms, we strongly encourage investigation of alternative models for the given problems.

MIPLIB classifies instances into three categories: *easy* for those that can be solved within an hour on a contemporary PC with a state-of-the-art solver, *hard* for those that are solvable but take a longer time or require specialized algorithms, and finally *open* instances for which the optimal solution is not known. This classification is kept up-to-date on the website, and we ask for notification whenever an optimal solution is determined for an *open* instance. The website also contains details if specific settings have been used to solve an instance.

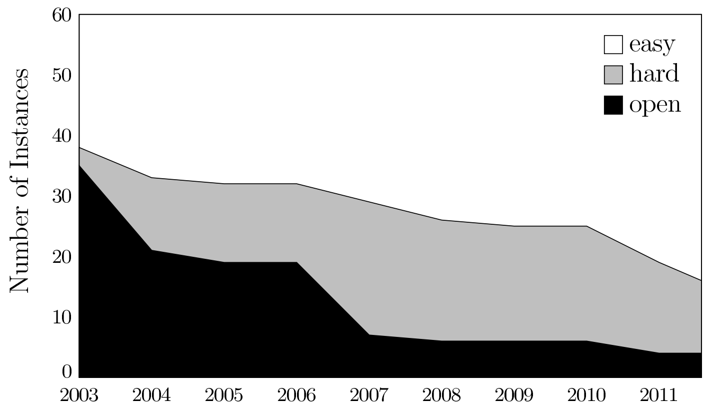
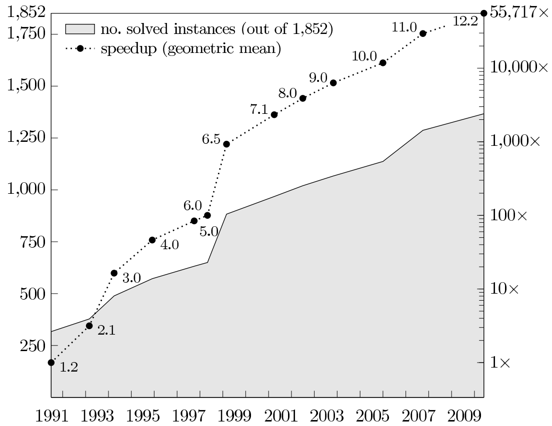


Fig. 1: Number of MIPLIB 2003 instances classified as *easy*, *hard*, and *open* over time

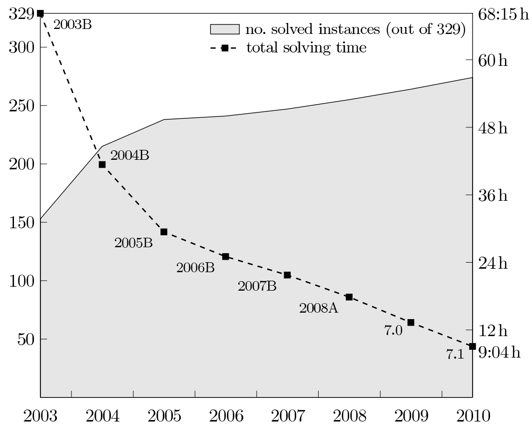
The progress in solving real-world MIP instances has been exceptional over the last decades. It is recorded in various articles [22, 72, 77] and numerous talks. One example is the solvability of the MIPLIB 2003 instances. As shown in Figure 1, at the start of MIPLIB 2003 there were 22 easy, three hard, and 35 open instances. By the end of 2010, 41 were classified as easy, 15 as hard, and only four open instances remained.

Another showcase is the speedup of commercial MIP solvers. Figure 2 depicts the progress made by two of the commercial solver vendors with long traditions, CPLEX and XPRESS. These figures are based on their internal test sets and record purely the speedup due to algorithmic improvements. For CPLEX the geometric mean of the speedup is drawn and for XPRESS the reduction in total solution time. Though these measures cannot be compared directly, they both show the impressive performance improvements gained during the last years.

Combining a pure algorithmic speedup of 55,000 with the speedup in computing machinery, we see that solving MIPs has become something like 100 million times faster in the last 20 years. This easily translates into the difference between considering an instance to be trivial versus unsolvable.



(a) Geometric mean speedup and number of instances solved by CPLEX versions 1.2 to 12.2 for an internal test set of 1,852 instances.



(b) Total solution time and number of instances solved by XPRESS versions 2003B to 7.1 for an internal test set of 329 instances.

Fig. 2: Performance improvements of CPLEX and XPRESS

This might give the misleading impression that all mixed integer programs are easy to solve nowadays. However, keep in mind that practitioners often experiment with tractable instances. In this sense, MIP codes tend to be tuned to more efficiently solve those models that are already known to be solvable. In order to compensate for this, we added a large set of instances to MIPLIB 2010 that are out-of-scope for today's solvers. As of this writing, the 361 instances of MIPLIB 2010 are classified as follows: 185 *easy*, 42 *hard*, and 134 *open*.

2 The test sets

During the initial discussions among the authors, it became evident that a single test set that also includes many very hard or even unsolved instances was not going to be sufficient. Researchers have often focused their attention on subsets of the MIPLIB 2003 instances that were suited to their particular topic of study. This often resulted in inadequate test sets because the full library only contained 60 instances, and added restrictions further reduced this size.

Therefore, we identified several areas for which dedicated test sets should be made available. Please note that a particular instance can be part of more than one test set. Table 1 gives an overview of all test sets.