How To Divide (and Conquer): Using General Disjunctions For Solving Mixed Integer Programs

Ashutosh Mahajan and Ted Ralphs

Industrial and Systems Engineering, Lehigh University

Abstract

Many real life decision problems are modelled and solved as Mixed Integer Programs (MIPs). In this work, we develop a novel technique that substantially improves the performance of branch-and-bound which is the most commonly used algorithm to solve MIPs. Computational results on benchmark instances are provided.

Introduction

MIPs are used to express mathematically many real life problems:

- Objective function (Maximize returns, efficiency etc., or Minimize delays, costs, congestion)
- Constraints (Physical constraints, capacity, etc.)
- Discrete or Indivisible Choices (yes/no decisions, Number of bridges, ships etc.)

Selected Applications:

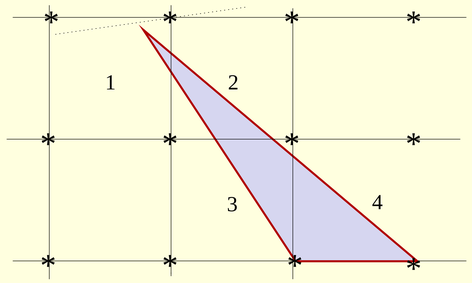
Instance	Description
10teams	Scheduling games in the English Football League.
bell3a	Capacity expansion of local access (computer) networks with minimum cost.
flugpl	Modeling operations of an airline company.
gt2	Routing trucks to meet customer demands.
mod008	Optimizing work loads on machines.

Problem

- The Branch-and-Bound algorithm is the most successful algorithm for solving MIPs. It is based on the divide-and-conquer paradigm.
- The time taken to solve can grow exponentially with the size of the problem. Even small instances can not be solved in reasonable time.
- MIPs are computationally difficult. Belong to the class of $\mathcal{NP}-$hard problems.

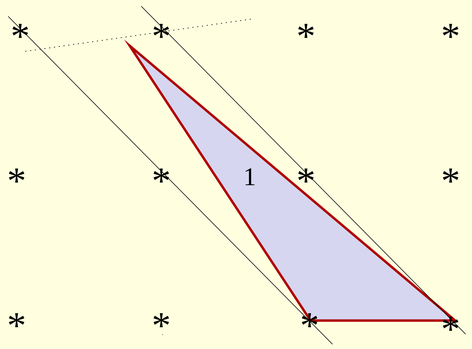
Improving Branch-and-Bound

We develop a novel strategy for dividing the search space. Existing methods of dividing using a single variable . . .



(Number of iterations: 4)

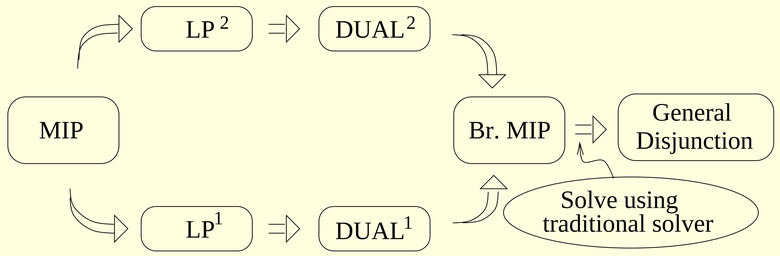
. . . we use several variables together in a so called general disjunction.



(Number of iterations: 1, Improvement by a factor of 4!)

Selecting General Disjunctions

There are an infinite number of general disjunctions. We select the one that improves the objective function the most. This problem can be modelled as a MIP and solved using a traditional approach.

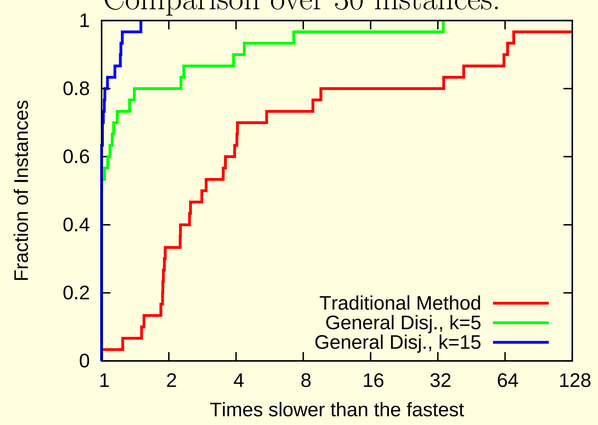


Computational Results

Number of iterations for selected instances:

Instance	Traditional	Gen. Disj.	Improvement
10teams	115	12	9.58
bell3a	16387	259	63.27
flugpl	394	6	65.67
gt2	340	10	34
mod008	2840	68	41.76
vpm1	263111	20	13155.55

Comparison over 30 instances:



Conclusions and Future Work

General disjunctions can reduce significantly the number of iterations in branch-and-bound. However, fast heuristics to discover such disjunctions need to be developed.