(DRAFT) Implicit enumeration with dual bounds

Nelson Frew

Introduction

The field of discrete optimisation (CITE) studies problems that model discrete decision making. Mixed-Integer Programming refers to a set of modelling and solving techniques for such discrete problems (CITE) by formulating them as Mixed-Integer Programs (MIPs). For many problems, this approach is the most efficient known method to find a solution. The standard MIP formulation, for maximisation, is as follows:

$$\begin{array}{ll} \text{maximise} & cx + hy \\ \text{subject to} & Ax + Gy \leq b \\ & x \geq 0 \text{ integral} \\ & y \geq 0 \end{array}$$

Generally, objective functions are constrained by linear inequalities, and variables can be specified to be integer or continuous.

Optimisation problems with linear constraints and continuous variables are known as Linear Programs (LPs), which are both solvable in polynomial time and very efficiently in practice (CITE). In this report, we consider maximisation problems whose constraints are all linear. MIPs, on the other hand, are characterised by having one more more variables constrained to be integer, generally resulting in an NP-Hard problem (CITE). By removing the integer constraints on variables in a MIP, we obtain its associated LP relaxation.

The state-of-the-art method for solving MIPs leverages this relationship to systematically search for what are known as primal solutions that are feasible, using bounds to implicitly enumerate through these.

Because the feasible region of continuous solutions naturally subsumes the set of feasible solutions for integers, we know, for optimal value for the LP relaxation z_{LP} and true optimal value z, that $z \leq z_{LP}$.

$$z \leq z_{LP} \tag{1}$$

Since z_{LP} is at least as large as our optimal value, we are provided with an upper bound in our case, which we refer to as the *dual bound.* Given our solved relaxation, we form a subproblem by adding constraints to our problem which constrain certain variables to be integer, and solve this. By solving LP relaxation again for this constrained subproblem, we can see if an optimal integer solution can exist in with these constraints: if our dual bound is lower than our best known value so far, we can prune this subproblem and switch to another constrained subproblem. Otherwise, if this subproblem has a dual bound above our best known value, we continue constraining and solving the relaxation until a new feasible solution is found, or a further constrained subproblem of this is pruned.

Should I note that x and y are vectors of assigned values within the objective?

This constitutes the core of the Branch and Bound approach, which we formally describe below: let (x^i, y^i) be the optimal solution to linear program LP_i, (x^*, y^*) denote an optimal solution to the MIP; $\underaccent{\bar}{z}$ denote the lower bound on the optimal value and z^* the optimal solution of the MIP, and let $\mathcal{L}$ denote the list of nodes of the Branch and Bound tree yet to be solved (i.e. not pruned nor branched on).

Algorithm 1: Branch and Bound algorithm for Mixed-Integer Programming

```
Data: A Mixed-Integer Program.
Result: The optimal solution value, and the associated solution
1  Set $\mathcal{L} := N_0$, set $\underline{z} := -\infty$, set $(x^*, y^*) := \emptyset$;
2  while $\mathcal{L}$ is not empty do
3      Select a node $N_i$ from $\mathcal{L}$, by some node selection scheme, deleting it from $\mathcal{L}$;
4      Solve $LP_i$ to get solution value $z_i$ and $(x^i, y^i)$ if it exists;
5      if $LP_i$ is infeasible, i.e. $z_i = -\infty$ then
6          Go to step 2;
7      end
8      else
9          Let $(x^i, y^i)$ be an optimal solution of $LP_i$ and $z_i$ be its objective value
10     end
11     if $z_i \leq \underline{z}$ then
12         Go to step 2;
13     else
14         if $(x^i, y^i)$ is feasible to the MIP then
15             Set $\underline{z} := z_i$;
16             Set $(x^*, y^*) := (x^i, y^i)$;
17             Go to step 2;
18         end
19     end
20     else
21         From $LP_i$ construct 2 linear programs $LP_{i1}, LP_{ik}$ with smaller feasible regions whose union does not
             contain $(x^i, y^i)$, but contains all the solutions of $LP_i$ with $x \in \mathbb{Z}$;
22         Add the corresponding new nodes $N_{i1}, N_{ik}$ to $\mathcal{L}$ and go to step 2.
23     end
24     Return $\underline{z}$;
25 end
```

While finding optimal solutions with Branch and Bound is in worst case exponential-time, for some problems there exist Approximation Algorithms (AAs) which can provide feasible solutions in as fast as polynomial-time, by sacrificing guaranteed optimality. In such cases, AAs provide a guarantee on the *quality* of the solution, within some factor.

An α-approximation is an AA that guarantees that the solution value will always be within a constant factor $\alpha < 1$ of the optimal solution; i.e. for an approximate solution value z_A, it must be that $\alpha z \leq z_A \leq z$. As a result, for a maximisation problem, we can analytically derive an upper bound on the optimal value from the lower bound guarantee on the optimal solution. In this way, we can use AAs in lieu of an LP relaxation to obtain valid dual bounds on our optimal solution. Further, if we use information from a found approximate solution, we can derive improved dual bounds *a posteriori* on our optimal solution. Because we are leveraging guarantees on the optimal value, the dual bounds we obtain are also guaranteed to be within a level of accuracy. This is distinct from LP relaxations, which can provide arbitrarily poor dual bounds.

When conducting a Branch and Bound with AAs, an important observation is that the general strategies which have contributed to the success of the standard Branch and Bound with LPs must be addressed. In particular, methods of branching and warm-starting solutions do not have a clear translation with respect to AAs. In this project, we investigate methods of effectively using dual bounds provided by AAs for the 0,1 Knapsack. To do this, we investigate known approximation schemes and methods to construct high quality dual bounds from them, as well as devising branching strategies within its Branch and Bound.

(**Put warm starting in future work**)