A graph structure to encode bound implications in MINLP

Giacomo Nannicini  
*Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA*  
Email: nannicin@andrew.cmu.edu

Pietro Belotti  
*Dept. of Mathematics, Clemson University, Clemson, SC*  
Email: pbelott@clemson.edu

Jon Lee  
*IBM T. J. Watson, Yorktown Heights, NY*  
Email: jonlee@us.ibm.com

Jeff Linderoth  
*Dept. of Industrial and Systems Engineering, University of Wisconsin-Madison, Madison, WI*  
Email: linderoth@wisc.edu

François Margot  
*Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA*  
Email: fmargot@andrew.cmu.edu

Andreas Wächter  
*IBM T. J. Watson, Yorktown Heights, NY*  
Email: andreasw@watson.ibm.com

March 25, 2011

1 Motivation

Consider a Mixed-Integer Nonlinear Program (MINLP):

\[
\begin{align*}
\min \quad & f(x) \\
\text{s.t.} \quad & g_j(x) \leq 0 \quad \forall j \in M \\
& x_i^L \leq x_i \leq x_i^U \quad \forall i \in N \\
& x_i \in \mathbb{Z} \quad \forall i \in N_I,
\end{align*}
\]

where \( f \) and \( g_j \) are factorable [1] functions, \( N = \{1, \ldots, n\} \) is the set of variable indices, \( M = \{1, \ldots, m\} \) is the set of constraint indices, and \( x \in \mathbb{R}^n \) is the vector of variables with lower/upper bounds \( x^L \in (\mathbb{R} \cup \{-\infty\})^n \), \( x^U \in (\mathbb{R} \cup \{+\infty\})^n \). The variables with indices in \( N_I \subset N \) are constrained to take on integer values in the solution.
Bound tightening is an important component of Branch-and-Bound-based solution algorithms for \( \mathcal{P} \). In practice, we can assume that a solver for MINLPs invests at most a user-specified fraction of time for bound tightening. When using CPU-intensive bound tightening techniques, it is likely that only a few applications of the algorithm are possible before the time limit is hit. Therefore we face the problem of choosing which variables should be tightened first. This is our main motivation. In this abstract we propose to use a specially built graph in the task of selecting the most promising variables for bound tightening.

2  **Bound implications graph**

A bound for a variable \( x_i \) is a triple \((i, \circ, z) \in N \times \{\geq, \leq\} \times (\mathbb{R} \cup \{+\infty, -\infty\})\). The bounds of \( \mathcal{P} \) are \((i, \geq, x_i^L)\) and \((i, \leq, x_i^U)\) \( \forall i \in N \). Let \( F \) be the set of feasible solutions to \( \mathcal{P} \). A bound \((i, \circ, z) \) implies \((j, \circ', z')\) if \( \forall \bar{x} \in F : \bar{x}_i \circ z \) we have \( \bar{x}_j \circ' z' \).

Given \( \mathcal{P} \), consider a directed graph \( G(\mathcal{P}) = (V, A(\mathcal{P})) \) with \( |V| = 2n \), and an invertible mapping \( \phi : N \times \{\ell, u\} \mapsto V \). \( \phi \) establishes a 1-to-1 correspondence between the bounds of each variable and the nodes, i.e., each variable has two associated nodes: one for its lower bound, one for its upper bound. The arcs \( A(\mathcal{P}) \) are constructed as follows: there is an arc from \((i, \ell)\) to \((j, \ell)\) (resp. \((j, u)\)) if the variable associated with \( i \) is a neighbour of the variable associated with \( j \) in the expression tree used for constraint propagation (FBBT [2]), and \( \forall \epsilon > 0, \exists \epsilon' > 0 \) such that \((i, \geq, x_i^L + \epsilon)\) implies \((j, \geq, x_j^L + \epsilon')\) (resp. \((j, \leq, x_j^L - \epsilon')\)). Similarly, there is an arc from \((i, u)\) to \((j, \ell)\) (resp. \((j, u)\)) if the variable associated with \( i \) is a neighbour of the variable associated with \( j \) in the expression tree used for constraint propagation, and \( \forall \epsilon > 0, \exists \epsilon' > 0 \) such that \((i, \leq, x_i^U - \epsilon)\) implies \((j, \geq, x_j^U + \epsilon')\) (resp. \((j, \leq, x_j^U - \epsilon')\)). We call \( G(\mathcal{P}) \) the bound implications graph of \( \mathcal{P} \).

Observe that \( G(\mathcal{P}) \) depends on the bounds \( x_i^L, x_i^U \) in \( \mathcal{P} \). Therefore, in a Branch-and-Bound algorithm, each node of the enumeration tree could potentially be associated with a different bound implications graph.

3  **Utility**

In this section we briefly discuss how the bound implications graph could be used in the context of bound tightening and of branching.

The bound implications graph could be used to rank the variables when choosing which ones should be tightened first. Given \( G(\mathcal{P}) \), for \( v \in V \) let \( R(v) \) be the set of nodes that can be reached from \( v \). Define an equivalence relation \( \sim \) on \( V \): \( u \sim v \) if \( R(u) = R(v) \). If \( u \sim v \), tightening \( \phi^{-1}(u) \) has an impact on exactly the same bounds as tightening \( \phi^{-1}(v) \). Choose a representative \( v \) for each equivalence class in \( V/ \sim \) (how \( v \) is chosen will not be discussed here; for instance, it could be a random choice). Let \( C \) be the set of representatives of \( V/ \sim \).

While performing Branch-and-Bound with a time limit on the bound tightening time, we propose to rank nodes \( v \) in \( C \) by decreasing \( |R(v)| \), and tighten the bounds following this order. Note that the graph does not contain information on the difficulty of tightening \( \phi^{-1}(u) \), or on the amount by which bounds will be tightened.

The graph could also be used for branching. A variable \( x_i \) with large value of \( |R(\phi(i, \ell))| \) and \( |R(\phi(i, u))| \) is a good candidate for branching, since both children nodes in the Branch-and-Bound tree will have tighter bounds than their father on a large number of variables. Furthermore, it may be appealing to consider branching on variables
that have an impact on the objective function; this can be done by looking at nodes in the bound implications graph from which the node corresponding to the lower bound of the objective function $f(x)$ can be reached.

4 Ongoing work

In our talk, we will present ongoing work with the bound implications graph in the context of bound tightening. In particular we will discuss how to build and maintain the graph, and report computational experiments on its accuracy and its usefulness on benchmark MINLP instances.

References
