A NEW SEQUENTIAL CUTTING PLANE ALGORITHM FOR SOLVING MIXED INTEGER NONLINEAR PROGRAMMING PROBLEMS

Claus Still

Process Design Laboratory
Åbo Akademi University
Biskopsgatan 8, FIN-20500, Åbo, Finland
Email: claus.still@abo.fi
Web page: http://www.abo.fi/fak/tkf/at/

Abstract. This paper presents a new algorithm for solving mixed integer nonlinear programming problems. The algorithm uses a branch and bound strategy where a nonlinear programming problem is solved in each integer node of the tree.

The nonlinear programming problems, at each integer node of the tree, are not solved to optimality; rather one iteration step is taken at each node and then linearizations of the nonlinear constraints are added as cuts to the problem. A Sequential Cutting Plane (SCP) algorithm is used for solving the nonlinear programming problems by solving a sequence of linear programming problems.

We present numerical results for the algorithm on a set of block layout problems indicating that the described algorithm is a competitive alternative to other existing algorithms for these types of problems.

Key words: Mixed Integer Nonlinear Programming, Cutting Plane Methods, Sequential Cutting Plane Method, Extended Cutting Plane Method, Branch and Bound

1 INTRODUCTION

In this paper we study a new algorithm for efficiently solving difficult mixed integer nonlinear programming (MINLP) problems of the form

\[ \min f(x, y), \]

s.t. \( g_j(x, y) \leq 0, \ j = 1, \ldots, m, \]
\( x \in X, \ y \in Y. \) (1)

Here the set \( X \) is a bounded, box-constrained set of the form \( X = \{ x \in R^n | x^L \leq x \leq x^U \} \), the set \( Y \) is a finite bounded set \( Y = \{ y \in Z^m | y^L \leq y \leq y^U \} \) and the functions \( f \) and \( g \) are convex and continuously differentiable.
We particularly study performance on a difficult set of block optimization problems. In [2], the Extended Cutting Plane algorithm proved to be very efficient for solving these types of optimization problems. Here, the Sequential Cutting Plane algorithm is further enhanced by using the ideas from the Extended Cutting Plane algorithm in order to form a new efficient algorithm for solving these difficult MINLP problems. We note that the algorithm can, in fact, be used to solve any convex MINLP problem of the form (1) to the global optimum. It can also be applied on nonconvex MINLP problems, but in that case, global optimality of the solution cannot be guaranteed.

2 SEQUENTIAL CUTTING PLANE ALGORITHM OVERVIEW

Our algorithm solves the problem (1) by doing a normal branch and bound on a relaxed version of (1). In each integer node of the tree, we fix the integer variables and thus obtain a nonlinear programming (NLP) subproblem. We perform an NLP iteration on the NLP subproblem. If the iterate is optimal, the solution in the integer node is an upper bound on the optimal solution of the original problem (1) and can be used for dropping nodes from the tree. If not, we add linearizations of the nonlinear constraints in the current iterate as cuts to the relaxed version of (1) and continue the branching process.

2.1 Branch and Bound

We construct the root node $\bar{P}^1$ of the tree by relaxing (1) such that we drop the integer requirements and the nonlinear constraints from the problem formulation. Note that if there are any linear constraints in $g(x, y) \leq 0$, we can keep them in our relaxed problem.

We then do a normal branch and bound procedure on this linear relaxation until we obtain an integer feasible node $\bar{P}^k$. In this node, we fix the integer variables $y^k$ and do an NLP iteration in order to solve the NLP subproblem

$$\begin{align*}
\min & \quad f(x, y^k), \\
\text{s.t.} & \quad g(x, y^k) \leq 0, \\
& \quad x \in X.
\end{align*}$$

(2)

We note that it is not necessary to solve this problem to optimality. We may only perform one NLP iteration to get a good approximation of the optimal solution to the subproblem. Let $x^k$ be the iterate we obtain as the solution to (2).

We then use the iterate $(x^k, y^k)$ to add linearizations of the violated and active nonlinear constraints in $(x^k, y^k)$,

$$g_j(x^k, y^k) + \nabla g_j(x^k, y^k)^T(x - x^k, y - y^k) \leq 0, \quad j \in \{j : g_j(x^k, y^k) \geq 0\},$$

(3)

to the set of cuts $\Omega_k$ and call this new set of cuts $\Omega_{k+1}$.

If $x^k$ is the optimal solution to (2), then $f(x^k, y^k)$ is an upper bound of the optimal solution to (1) and can be used to drop unexplored nodes from the branch and bound tree whenever the lower bounds of the nodes are greater than this upper bound.
2.2 NLP Iteration

We solve (2) by performing an NLP iteration. In an NLP iteration, a sequence of linear programming (LP) subiterations is performed. In each subiteration \( i \) within the NLP iteration, an LP problem \( LP(x^{(i)}) \) is generated in the current iterate \( (x^{(i)}, y^k) \). The LP problem \( LP(x^{(i)}) \) is of the form

\[
\begin{align*}
\min & \quad \nabla f(x^{(i)}, y^k)^T d + C t, \\
\text{s.t.} & \quad g_j(x^{(i)}, y^k) + \nabla g_j(x^{(i)}, y^k)^T d - t \leq 0, \ j = 1, \ldots, m, \\
& \quad (d^{(r)})^T H^{(i)} d = 0, \ r = 1, \ldots, i - 1; \ i > 1, \\
& \quad t \geq 0, \\
& \quad x^{(i)} + d_x \in X, d_y = 0,
\end{align*}
\]

where \( d = (d_x, d_y) \), and \( d^{(r)}, r = 1, \ldots, i - 1 \), are the previously obtained search directions within the NLP iteration. Also, \( H^{(i)} \) is the current estimate of the Hessian of the Lagrangian

\[
L(x, \lambda) = f(x, y^k) + \sum_{j=1}^{m} \lambda_j g_j(x, y^k).
\]

The BFGS update formula was used in our implementation of the algorithm to approximate \( H^{(i)} \).

The dual optimal solution to the LP problem \( LP(x^{(i)}) \) provides a Lagrange multiplier estimate, which can further be used for estimating the Hessian of the Lagrangian. Thus, the Hessian approximation can be updated in each LP subiteration.

The solution to each LP problem \( LP(x^{(i)}) \) provides a search direction \( d^{(i)} = (d_x^{(i)}, 0) \). A line search is then performed in the obtained search direction minimizing a modified function based on the Lagrangian of (2),

\[
\tilde{L}(x, \lambda) = f(x, y^k) + \sum_{j=1}^{m} \lambda_j g_j(x, y^k)^+ + \rho \sum_{j=1}^{m} (g_j(x, y^k)^+)^2,
\]

where \( g_j(x, y^k)^+ = \max(g_j(x, y^k), 0) \) and \( \rho (> 0) \) is a penalty parameter.

The current iterate \( (x^{(i)}, y^k) \) is then updated using the solution \( \alpha^{(i)} \) of the line search such that \( x^{(i+1)} = x^{(i)} + \alpha^{(i)} d_x^{(i)} \), where \( \alpha^{(i)} \) is the step length found in the line search.

A new LP problem is then constructed in the updated iterate \( (x^{(i+1)}, y^k) \). The new LP problem is constructed in a similar way as the previous LP problem with equality constraints requiring the new search direction to be a conjugate direction to the previously obtained search directions with respect to the current estimate of the Hessian of the Lagrangian.

The linear equality constraints

\[
(d^{(r)})^T H^{(i)} d = 0, \ r = 1, \ldots, i - 1,
\]

are cutting hyperplanes ensuring that \( d \) will be a conjugate direction to the old directions \( d^{(r)} \).

The new LP problem is then solved, a new line search performed and the iterate updated again. The procedure is repeated until the LP problem becomes infeasible,
Algorithm 1  Pseudo-code for the SCP algorithm

Initialize: Do a number of NLP iterations on the continuous relaxation of (1) to obtain the initial iterate for the problem. Construct the relaxed node $\bar{P}^1$ as described in Section 2.1 and insert it into the branch and bound tree as the top node and set the upper bound $U = \infty$. Add cutting planes for the initial iterate to the set of cutting planes $\Omega_1$. Let $k = 1$.

While (there are unexamined nodes in the tree) do
1. Do normal branching on the branch and bound tree including the cutting planes $\Omega_k$ as linear cuts until integer solution $(\tilde{x}^k, y^k)$ found.
2. Fix $y^k$ and perform an NLP iteration on (2).
   Let the iterate after the NLP iteration be $x^k$.
3. If solution $x^k$ optimal for (2)
   Then update upper bound $U := \min\{U, f(x^k, y^k)\}$.
4. If (2) is found to be infeasible
   Then let $x^k$ be the last iterate from the NLP iteration.
5. Let $\Omega_{k+1} = \Omega_k$. Add cutting planes generated in $(x^k, y^k)$ for the active and violated constraints and add them to $\Omega_{k+1}$. Let $k := k + 1$.

End While

a sufficient number of steps have been taken or the current solution to the LP problem is sufficiently close to zero.

Normally, the NLP iteration would then be repeated until we find an optimal point. However, in this version of the algorithm we perform only one NLP iteration in each integer node of the branch and bound tree in order to improve the performance of the algorithm.

Convergence properties of the NLP version of the SCP algorithm have been analyzed in earlier papers.\textsuperscript{6,8}

2.3 Algorithm Pseudo-Code

We summarize the algorithm in Algorithm 1.

3 NUMERICAL RESULTS

We tested this new algorithm on a set of difficult block optimization problems. The problems concerned optimizing the arrangement of a number of departments with unequal area requirements. The problems can be formed as mixed integer nonlinear programming problems where the constraints are department and floor area requirements as well as department locational restrictions. The target is to optimize the cost associated with the projected interactions between departments.\textsuperscript{2} In [2], the Extended Cutting Plane method was compared to a number of commercial algorithms and proved to be superior to the other solvers.

In Table 1 the results in [2] are summarized for the solvers in terms of the number of problems solved to optimality, number of problems for which a feasible solution was obtained and number of problems for which no solution was obtained within 12 hours of CPU time. More information about the solvers used can be found in [1].

The results are excellent for the $\alpha$-ECP algorithm on these test problems. It
only failed to solve one problem to global optimality and obtained the best integer feasible solution for the problem it could not solve to global optimality.

In Figure 1 we compare our new SCP algorithm with the $\alpha$-ECP algorithm using a performance profile.\textsuperscript{3} We also compare the results with a version where we solve each integer node to optimality by repeating the NLP iterations until an optimal solution to (2) is found. This version of the SCP algorithm is denoted SCP-NLP. Note that if you solve the integer nodes to optimality, the procedure is similar to the method described in [5]. CPLEX was used as the MILP solver and the NLP part of the algorithm was implemented in MATLAB.

As we can see from the results, the SCP algorithm performed very well on the test problems. In more than half of the cases it was the fastest solver. Considering that some of these problems can take several hours to solve, the performance improvement is significant.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
Solution & BARON & DICOPT & MINLPbb & SBB & $\alpha$-ECP \\
\hline
Optimal & 13 & 5 & 5 & 5 & 31 \\
Feasible & 16 & 2 & 27 & 23 & 1 \\
No solution & 3 & 25 & 0 & 4 & 0 \\
\hline
\end{tabular}
\caption{Performance of the solvers on the block layout problem as reported in [2]}
\end{table}

![Performance Profile](image.png)

Figure 1: Performance profile comparing CPU times for solving the block layout problems
4 CONCLUSIONS

We have presented a new cutting plane-based method for solving mixed integer nonlinear programming problems. The numerical results clearly show that it can be used to solve challenging block layout problems. One advantage of the algorithm is that it will efficiently find a good feasible solution to a problem, even if the problem is not solved to optimality. Thus, the SCP algorithm can be combined with metaheuristics to solve difficult combinatorial optimization problems in real-world applications.

REFERENCES


