Literature Review

Yuvan Das - 11/07/24

Paper 1

A Computational Study for Piecewise Linear Relaxations of Mixed-Integer Nonlinear Programs

1. Introduction

- General MINLPs remain challenging to solve, with spatial branch-and-bound still at the core of most solvers.
- Over the last two decades, new methods have emerged using piecewise convex relaxations without directly branching on continuous variables.
- These approaches face two main challenges: constructing tight relaxations of nonlinear functions and incorporating them into MIP or convex NLP models.
- Various techniques exist for optimal linearization and polynomial relaxations, but they face the curse of dimensionality for higher-dimensional functions.
- Multiple MIP formulations for modeling piecewise linear (PWL) functions have been developed, including disaggregated and aggregated convex combination models, incremental methods, and more recent binary and integer Zig-Zag formulations.
- Despite existing comparisons, no comprehensive study has evaluated the performance of these MIP models across a wide range of MINLP problems, which this paper aims to address.

2. Preliminaries

- The MINLP problem (P) is defined with linear and nonlinear constraints, bounded variables, and a mix of continuous and integer variables.
- The formulation is general, allowing for equality constraints and nonlinear objective functions through simple transformations.
- The domain of nonlinear functions (Df) is a compact d-dimensional box, suitable for piecewise linear (PWL) relaxations in low dimensions.
- Many practical MINLPs have factorable nonlinear functions, which can be represented using elementary operators and univariate nonlinearities
- Bivariate product functions can be reformulated into one-dimensional representations using additional variables and constraints, including McCormick relaxations for strengthening.
- The focus is on one-dimensional MIP models for PWL relaxations of nonlinear functions, omitting adaptive refinement strategies for simplicity in comparison.

3. One-dimensional MIP models for piecewise linear relaxations

- The chapter describes various MIP representations of piecewise linear models, including disaggregated and aggregated convex combination models, and their logarithmic variants.
- The disaggregated convex combination model uses binary variables for each segment and represents points as convex combinations of neighboring breakpoints.
- The aggregated convex combination model reduces the number of continuous variables by combining some variables from the disaggregated model.
- Logarithmic formulations are introduced to reduce the number of binary variables, including a binary branching scheme and two new Zig-Zag formulations (binary and integer).

4. Implementation

- MINLP problems are formatted in OSIL, an XML-based language that organizes variables and nonlinearities in a tree structure, enhancing traceability of complex expressions.
- Problems are reformulated to simplify multivariable expressions into one-dimensional nonlinearities using a custom format (`OSILData`), preparing them for MIP relaxations.
- Nonlinear functions are approximated with PWL functions using error-bound-controlled segments to achieve accurate approximations.
- Reformulated instances are converted to Pyomo-based MIP models and solved with Gurobi on FAU cluster resources, allowing solver flexibility and efficiency.
- Key metrics like primal and dual bounds, optimization status, and time to the first feasible solution are recorded to analyze performance and solution quality.

5. Numerical Results

- Different error bounds are tested to measure the effect on segment counts and segment lengths in the piecewise linear (PWL) relaxations, with smaller bounds increasing both the segment count and complexity of the MIP model.
- Performance of eight PWL MIP models is analyzed across error bounds. Results show that higher error bounds lead to faster solutions and more problems solved to optimality, while tighter bounds increase runtime and reduce the number of solvable instances within the time limit.
- Solution quality, in terms of the gap between the optimal MINLP and MIP relaxation solutions, is assessed. For small error bounds, gaps are minimal, showing that PWL relaxations can yield accurate solutions with good dual bounds.
- Figures track the number of problems solved over time, with checkpoints marking the stages (e.g., MIP model creation, solving process completion, finding of primal/optimal solutions). Some models show significant variations in progression depending on error bounds.
- The study presents the shrinking of primal and dual gaps over time for each model, with shifted geometric means used to indicate overall improvement, particularly noticeable with smaller error bounds that yield closer approximations to the optimal solution.

6. Discussion

- With larger error bounds, the number of solved problems remains quite similar across models. However, as error bounds decrease, differences in model performance become pronounced. For moderate error bounds models perform similarly up to the first 1000 seconds.
- Overall, the incremental method consistently solves the most instances, especially for higher error bounds, making it the recommended choice for practical use. Non-logarithmic convex combination methods perform worse than logarithmic methods across most instances, both in terms of the number of problems solved and runtime efficiency.
- For instances where feasible solutions are found by all methods, Inc continues to excel, with logarithmic models trailing slightly behind. Logarithmic models are adaptable for high-accuracy requirements.,
- Analyzing primal and dual optimality gaps reveals that for larger error bounds, gap behavior is similar across methods. As error bounds tighten, differences become more apparent, with logarithmic models outperforming non-logarithmic methods in closing these gaps.
- For general use, Inc is recommended for its overall performance across different error bounds. Logarithmic models provide a viable alternative, especially for instances requiring high accuracy or for instances where Inc encounters performance limitations.
- The study demonstrates that MIP relaxation solutions approach the optimal MINLP solutions closely, with median gaps already negligible. This suggests that MIP relaxations are a practical approach for solving MINLPs, particularly when precise MINLP solutions are difficult to obtain directly.

7. Paper 1 - Overall Takeaways

- There are many ways to represent piecewise linear (PWL) functions as MIP models
- There are aggregated and disaggregated versions of these models. This is dependent on how variables are used to represent PWL functions, with the aggregate models typically more efficient
- Logarithmic variants of these models can be applied to reduce the number of variables.
- Zig-Zag methods can be applied to reduce the number of variables and simplify the branching, by ensuring that logarithmic models only branch in the feasible regions through extra constraints.
- Inc models are simpler to formulate but can be more time complex as they operate in O(n) variables. Log models are more complex to implement, but more efficient as they operate in O{log(n)) binary variables, which is more compact.
- The overall goal of this analysis is to identify the models with the tightest bounds on the solution to efficiently and correctly solve opti problems.

Paper 2

Mixed Integer Linear Programming
Formulation Techniques

1. Introduction

- Mixed integer linear programming (MIP) has become essential in business and engineering due to powerful LP-based solvers and MIP's modeling flexibility.
- While constructing basic MIP formulations is often straightforward, advanced techniques are sometimes necessary for optimal solver performance.
- This study focuses on MIP formulation techniques that involve both integer and continuous variables, with limited coverage of topics like combinatorial optimization.
- Effective MIP formulation is challenging to predict, but some properties consistently correlate with better computational results.
- Preliminary testing of multiple formulations can help identify the most effective approach for a given problem.

2. Preliminaries and Motivation

- MIP modeling often involves non-convex, continuous, piecewise linear functions. These functions are modeled using sets representing the function's graph, making it feasible to transform complex non-convex problems into MIP problems.
- The reformulation involves using auxiliary variables and constraints that represent the function's graph (such as breakpoints, slopes, and intercepts), which enable MIP formulations to handle piecewise linear functions within bounds.
- A sharp MIP formulation tightly bounds the solution space by ensuring its linear programming (LP) relaxation equals the convex hull of the feasible region. This provides an optimal LP relaxation bound, enhancing MIP solution quality.
- A stronger property, known as locally ideal, ensures all integer requirements are met automatically within the LP relaxation's feasible solutions, avoiding unnecessary branching and improving solver efficiency.
- Creating formulations involves balancing strength (for better LP relaxation) and formulation size, which can impact performance. Problems are often divided into portions to manage complexity.
- Sharp formulations provide strong LP relaxation bounds, while locally ideal formulations are even stronger. Traditional MIP formulations without auxiliary variables can be sharp if they meet certain integrality conditions.

3. Polyhedra

- Polyhedra are defined in two forms: H-polyhedra (defined by linear inequalities) and V-polyhedra (defined by convex hulls of points and cones). These representations are equivalent for rational polyhedra.
- A polyhedron is bounded if its recession cone is {0}, and it is unbounded if the recession cone is nontrivial.
- Extreme points are points that cannot be expressed as a convex combination of other points and extreme rays are directions of unboundedness.
- This theorem establishes that any nonempty pointed rational polyhedron has both an H- and V-representation, where the set of extreme points and rays can describe the polyhedron’s structure.
- The Fundamental Theorem of Integer Programming is introduced, showing that the convex hull of integer points in a polyhedron is also a polyhedron, providing insights into the structure of MIP formulations. Linear transformations are also discussed, demonstrating how extreme points are preserved under transformations.

4. Size and Extended MIP Formation

- The passage highlights an issue where binary variables used in certain formulations might not be enough to construct a polynomial-sized sharp formulation.
- A lemma shows that the number of facets of SSS grows exponentially in nnn, making it impossible to construct a sharp formulation of SSS with a polynomial number of variables and constraints.
- A proposition provides a method for creating polynomial-sized formulations for a wide range of disjunctive constraints by using a growing number of auxiliary variables, leading to "locally ideal" formulations.
- A polynomial-sized sharp formulation can be obtained using the technique from Proposition 4.2, resulting in a formulation with a linear number of auxiliary variables.
- For certain combinatorial optimization problems, such as the perfect matching problem on a complete graph, no polynomial-sized sharp extended formulation exists.

5. Basic Extended Formulations

- There are two primary mixed integer linear programming (MILP) formulations for disjunctive constraints: the H-formulation (inequality-based) and the V-formulation (extreme point/ray-based), each useful for different scenarios.
- Although V-formulation can be derived from H-formulation, their structural differences influence computational efficiency, especially in applications with disjunctive constraints.
- For constraints involving absolute values, H-formulation applies direct inequalities, while V-formulation decomposes variables to represent absolute values, which may lead to performance differences in LP relaxation.
- Both formulations are shown to produce sharp LP relaxations for their respective feasible regions, impacting optimization outcomes.
- Additional references suggest methods for expanding these formulations to handle more complex cases with added parameters, allowing for broader application across MILP problems.

6. Projected Formulations

- Even the strongest traditional Big-M formulation can fail to be sharp, meaning it doesn't always yield tight constraints or optimal solutions in certain complex cases.
- Balas, Blair, and Jeroslow proposed a hybrid Big-M formulation, combining elements of projected formulations and traditional Big-M, aimed at enhancing formulation strength for problems with common left-hand side matrices across constraints.
- The hybrid formulation can be applied to a broader range of problems than traditional Big-M and can exploit structural commonalities in constraints to reduce problem size and improve efficiency.
- The hybrid approach is demonstrated in cases like SOS (Special Ordered Set) constraints, where the formulation can reduce complexity and improve computational performance.
- The hybrid Big-M formulation tends to be smaller in size than traditional Big-M, especially when partial common left-hand structures are present, without significantly compromising strength.
- Although the hybrid formulation is stronger, it still may not be locally ideal or sharp in all cases, and recognizing sharpness in some hybrid cases is NP-hard, meaning it's computationally challenging.

7. Large Formulations

- Hybrid Big-M formulations generally offer theoretical and computational advantages over traditional Big-M approaches, but specific problem characteristics can sometimes make traditional formulations more effective.
- While compact MIP formulations are preferred, certain problems only allow for large formulations; specialized methods like branch-and-cut and branch-and-price can effectively manage these cases.
- Proposition 7.1 provides a technique to create smaller, extended formulations from large ones, making certain constraints more manageable in optimization problems.
- Adding facet-defining inequalities can make formulations “locally ideal,” though this approach can increase complexity due to the potential exponential growth in the number of constraints.
- Large constraints in MIP models can sometimes be replaced by a set of smaller, derived constraints using separation LP techniques, simplifying the model without sacrificing structure.
- When both large and small formulations are available, modelers must weigh the benefits of computationally simpler large formulations against the precision of smaller, more complex ones.

8. Incremental Formulations

- Traditional constraints on disjunctive formulations can lead to unbalanced branch-and-bound trees, which slow down the solution process.
- SOS1 branching, developed by Beale and Tomlin, helps avoid unbalanced trees by creating branches that fix about half of the integer variables, leading to a more efficient branching process.
- Transforming the binary variables yyy into an ordered set of variables www results in more balanced branches, where each branch effectively fixes multiple variables simultaneously, known as double contracting.
- The chapter demonstrates this with a transformed MIP formulation where branching on the www-variables allows the problem to be solved with fewer branch-and-bound nodes.
- For certain cases, a transformation of continuous variables can yield ad-hoc incremental formulations that are especially efficient for handling piecewise linear functions.
- Incremental formulations, especially for piecewise linear functions, often reduce the number of branch-and-bound nodes and enhance computational performance.

9. Logarithmic Formulations

- Traditional formulations for the union of kkk polyhedra use kkk binary variables, but logarithmic formulations aim to reduce this to log2(k)log_2(k)log2(k) binary variables for more efficient encoding.
- A binary encoding can replace the unary encoding for a variable leading to simpler formulations. This has been applied in certain MIP models, though it's generally ineffective for modeling general integer variables.
- This approach offers a structure to formulate constraints that need only log2(k)log_2(k)log2(k) binary variables, minimizing the binary variable count while retaining model efficiency.
- The formulation (9.7) builds on the polyhedral union approach, using binary encodings to represent the polyhedra. This method is beneficial for piecewise linear functions and polyhedral unions with common recession cones.
- Proposition 9.4 uses Big-M constraints, which omit auxiliary variables but require careful choice of the Big-M constant to ensure valid constraints, though it may lack sharpness and local optimality.
- Various logarithmic formulations have been developed for specific constraints, and comparisons with incremental formulations reveal benefits for each approach depending on problem structure and size.

10. Combining Formulations and Propositional Logic

- One approach to control formulation size is by independently constructing MIP formulations for each part of the problem and then intersecting them. Known as "model linkage", this approach can sometimes weaken the final model but can substantially reduce complexity.
- For cases where a set SSS can be represented as an intersection of unions of polytopes, one can represent SSS using a single disjunctive constraint, combining the unions directly. This produces stronger formulations but results in larger models.
- By using propositional logic, it's possible to construct variations that balance the extremes of intersecting and fully combining constraints. Propositional calculus techniques allow for custom combinations that achieve an optimal trade-off between formulation strength and size.
- When transforming a fully combined formulation (e.g., a disjunctive constraint) back to an intersection-based approach, auxiliary variables may be necessary. For example, in joint probabilistic constraints, binary variables can track which constraints are violated, allowing for a feasible combined formulation.
- The chapter references additional literature on propositional logic, logic constraints, and other combinatorial techniques used in MIP and constraint programming, which are useful for constructing robust, scalable models.

11. MIP Representability

- A set is bounded MIP representable if it can be formulated as a union of polyhedra with a common recession cone, where the integer variables are bounded or binary.
- This extension of bounded MIP representability allows for unbounded integer variables but restricts them to being part of the original variables, which increases the modeling power beyond just bounded integers.
- The most general form of MIP representability involves replacing continuous recession directions with discrete ones from a finitely generated integral monoid. This allows for the modeling of a wider variety of sets, including those with integer-based recession directions.
- For a set to be MIP representable, it must be closed, and its convex hull must be a polyhedron. However, these conditions alone are not sufficient, as demonstrated in counterexamples.

12. Other Topics

- LP and polyhedral methods in combinatorial optimization often involve creating sharp or locally ideal formulations. Extended formulations are actively researched in this area for problems like the Traveling Salesman Problem (TSP) and perfect matchings.
- The construction of extended formulations is linked to approximation problems in convex geometry. Recent works combine techniques for both polyhedral approximations of convex sets and formulations for combinatorial optimization.
- Linearization techniques for products of binary or bounded integer variables with continuous variables are key MIP formulation tools. These techniques, which transform nonlinear terms into linear ones, are foundational in MIP and continue to evolve.
- The linearization of products has been extensively studied, especially in the context of mathematical programming reformulations. A notable method for achieving this is the reformulation-linearization technique.

13. Paper 2 - Overall Takeaways

-