Parallel Cut-and-Solve: A Method for Solving Mixed-Integer Programs Utilizing Distributed Computational Power

Sharlee Climer and Michael Chan
University of Missouri-St. Louis
Mixed-integer linear programs (MIPs)

- A particular format for defining a problem
  - Many combinatorial problems can be cast as MIPs
- NP-hard in general
  - Some relatively small instances have not been solved
  - Yet, great progress has been made in field
    - Instances with billions of variables have been solved
  - Success due to ability to prune away most of the solution space, while preserving optimality

Only certain types of well-studied problems have experienced these successes
Solving NP-Hard Problems

Approximate methods
• Most common
• Many approaches e.g.:
  • Statistical-based
  • Machine learning
  • Bayesian modeling
  • Network-based

Exact methods
• Exhaustive enumeration
• MIP search strategies
  • Cutting Planes
  • Branch & Bound
  • Branch & Cut
  • Cut & Solve
Exhaustive Enumeration

- Number of combinations increases exponentially
- Possible to test every pair
  - Given efficient algorithm and adequate computational resources

<table>
<thead>
<tr>
<th>Size</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td># Combinations</td>
<td>$n$</td>
<td>$\binom{n}{2} = \frac{n^2 - n}{2}$</td>
<td>$\binom{n}{3} = \frac{n^3 - 3n^2 + 2n}{6}$</td>
</tr>
<tr>
<td>$n = 1,000,000$</td>
<td>1,000,000</td>
<td>499,999,500,000</td>
<td>$1.7 \times 10^{17}$</td>
</tr>
</tbody>
</table>

$\binom{n}{k} = \frac{n!}{k!(n-k)!}$
Traveling Salesman Problem (TSP)

- Well-studied combinatorial problem
- Given a set of cities and the cost of travel between them, what is the least costly way to travel to all of them and return home?
120-city TSP

5.6 x 10^196 possible routes!

Estimated 3.3 x 10^80 subatomic particles in the known universe

85,900-city TSP
3,689,362,050 variables (integrated circuit)

24,978-city TSP
311,937,753 variables (Sweden)
MIP formulation

- MIP format includes:
  - Objective function
  - Constraints

- All must be linear!
  - \( y = mx + b \)
- May be inequality
  - \( y \leq mx + b \)
Example with 2 variables

Minimize
Z = -11x + 4y

Subject to:
3x + 8y <= 40
11x - 8y <= 16
x, y >= 0

Integrality not required, so this is just a linear program (LP)
• Easy to solve
Objective function:

Minimize $Z = -11x + 4y$

$y = \frac{11}{4} x + \frac{Z}{4}$

Family of parallel lines with slope of $\frac{11}{4}$ and unknown $y$-intercept
Optimal solution

\begin{align*}
x &= 4 \\
y &= 7/2 \\
Z &= -30
\end{align*}

Optimal solution is always on a vertex for LPs
Minimize
\[ Z = -11x + 4y \]

Subject to:
\[ 3x + 8y \leq 40 \]
\[ 11x - 8y \leq 16 \]
\[ x, y \geq 0 \]
\[ x, y \text{ integer} \]

Optimal solution
\[ x = 3 \]
\[ y = 3 \]
\[ Z = -21 \]
Increase number of variables

- 3 variables
  - Each constraint is a plane
  - All feasible integral solutions are contained within (or on) polyhedron
  - Objective function is also a plane

- Many variables
  - High-dimensional space
  - Convex polyhedron
    - Defined by hyperplanes
  - Optimal solution where objective hyperplane intersects last integral solution
Relaxations solved during search

- Common to relax integrality
  - Solve LP

- Simplex method
  - Move along edges in direction that improves the objective function value
  - Reach optimal solution
Cutting Planes

Relaxed solution removed
- No integral solutions are removed
- Optimality is ensured

Search path can be prohibitively long
- Cutting planes remove tiny slice of search space
  - Must avoid removing integral solutions
  - Effective cutting planes can be difficult to identify
Incumbent
Branch & bound

- Maintain incumbent solution
  - Feasible solution
  - Not necessarily optimal
  - Update when a better feasible solution is encountered
- Spawn child nodes
  - Each valid value for variable
- Tree is explored
- At each node a relaxation is solved
  - If relaxation is smaller than incumbent, prune subtree
    - (Assuming maximization objective)
    - Optimality is not compromised
- Sensitive to early decisions of variables to spawn and search directions
Incumbent

BRANCH & CUT
Branch & Cut

- Combination strategy
  - Branch & bound tree with cutting planes applied
  - Tightens bounds and improves pruning
- Padberg & Renaldi introduced in the 1980's
  - Ability to solve large-scale MIPs was born
- Primary research focus on separation algorithms
  - Determine cutting planes
- State-of-the-art commercial solvers
  - IBM's Cplex, Gurobi
    - Free academic/nonprofit research license
- Not suitable for massive parallelization
  - Application of cutting planes is sequential process
  - Memory exhaustion frequently occurs for difficult problems
Incumbent
Cut & solve

- Solve relaxation
- Cut away a chunk of the solution space
  - Include integral solutions
    - Use *piercing* cut
  - Solve this small problem optimally
  - Provides *anytime* solution
    - Update incumbent
- Solve relaxation for remaining solution space
- Repeat until limits cross
  - Current incumbent is optimal
Cut & solve

- Cut & solve was featured in Boris Goldengorin's plenary lecture
  - 2010 American Conference on Applied Mathematics

- Has outperformed IBM's Cplex in more than a dozen publications
  - All used Cplex to solve subproblems

- Straight Cplex is faster for easier problems

- Reasons for cut & solve's success for tough problems:
  - Never applies branch & cut to entire problem
  - Exploits branch & cut for moderate problems
  - Amplifies pruning power

Bonus: Independent sparse problems can be spawned as quickly as relaxations are solved
• Utilization of massive resources available
• Branch-and-Bound can be parallelized
  • Doesn't leverage cutting plane technology

• Cut-and-Solve holds promise
  • Never before attempted massive parallelization