The Gurobi Solver V1.0

Robert E. Bixby
Gurobi Optimization & Rice University

Ed Rothberg, Zonghao Gu
Gurobi Optimization
Overview

• Background
• Rethinking the MIP solver
  – Introduction
  – Tree-of-Trees
  – Parallel MIP
  – Other
    • Heuristics
    • Cutting Planes

• Where we are now
  – Performance
Gurobi Optimization

- Gurobi Optimization, Inc. founded July, 2008
  - Founders: Gu, Rothberg, Bixby
- Building a algorithmic base
  - LP simplex and MIP in Version 1.0
- Development timeline
  - March 2008
    - Development began
  - October 2008
    - Version 1.0 development completed
  - 18 May 2009
    - Version 1.0 released
  - 2 October 2009
    - Version 2.0 to be released
Redesigning the MIP Solver
MIP problem

- Mixed Integer Programming

Minimize \( c^T x \)

Subject To \( A x = b \)

\( l \leq x \leq u \)

Some \( x_j \) must be integral
MIP Solution Approach

• Branch and bound [Land & Doig, 1960]
  – Explore a tree of *relaxations* (ignore integrality)

![Diagram of MIP solution approach](attachment://mip_solution_diagram.png)
Key Ingredients

• For an effective MIP solver, you need ...
  – Heuristics
    • Explore solutions near the relaxation quickly
    • Find feasible solutions at nodes other than leaf nodes
  – Parallelism
  – Presolve
    • Tighten formulation before starting branch & bound
    • Once you start branching, mistakes replicate
  – Branch-variable selection
  – Cutting planes
  – LP dual-simplex solver
A Fresh Look
A Fresh Look

• Things have changed:
  – Two examples:
    • “Sub-MIP” as a pervasive approach
    • Ubiquitous parallel processing
Sub-MIP As A Paradigm

• Key recent insight for heuristics:
  – Can use MIP solver recursively as a heuristic
  – Solve a related model:
    • Hopefully smaller and simpler
  – Examples:
    • Local cuts [Applegate, Bixby, Chvátal & Cook, 2001]
    • Local branching [Fischetti & Lodi, 2003]
    • RINS [Danna, Rothberg, Le Pape, 2005]
    • Solution polishing [Rothberg, 2007]
RINS

• Relaxation Induced Neighborhood Search
  – Given two “solutions”:
    • $x^*$: any integer feasible solution (not optimal)
    • $x^R$: optimal relaxation solution (not integer feasible)
  – Fix variables that agree
  – Solve the result as a MIP
    • Possibly requiring early termination

• Extremely effective heuristic
  – Often finds solutions that no other technique finds
Why Is RINS So Effective?

• MIP models often involve a hierarchy of decisions
  – Some much more important than others
• Fixing variables doesn’t just make the problem smaller
  – Often changes the nature of the problem
    • Extreme case:
      – Problem decomposes into multiple, simple problems
    • More general case:
      – Resolving few key decisions can have a dramatic effect
  – Strategies that worked well for the whole problem may not work well for RINS sub-MIP
    • More effective to treat it as a brand new MIP
Rethinking MIP Tree Search
Tree-of-Trees

• Gurobi MIP search tree manager built to handle multiple related trees
  – Can transform any node into the root node of a new tree

• Maintains a pool of nodes from all trees
  – No need to dedicate the search to a single sub-tree
Tree-of-Trees

Node pool:

Search tree:

x=0

y=0

y=1

x=1
Tree-of-Trees

• Each tree has its own relaxation and its own strategies...
  – Presolved model for each subtree
  – Cuts specific to that subtree
  – Pseudo-costs for that subtree only
  – Symmetry detection on that submodel
  – Etc.

• Captures structure that is often not visible in the original model
Parallel MIP
Why Parallel?

• Microprocessor trends have changed
• Transistors are:
  – Still getting smaller
  – But not faster
• Implications:
  – New math for CPUs: more transistors = more cores
    • 4 cores now, 8 cores in late 2009, ...
  – *Sequential software won’t be getting significantly faster in the foreseeable future*
• Gurobi MIP solver built for parallel from the ground up
  – Sequential is just a special case
Need Deterministic Behavior

• Non-deterministic parallel behavior:
  – Multiple runs with the same inputs can give different results

• “Insanity: doing the same thing over and over again and expecting different results.”
  – Albert Einstein

• Conclusion: non-deterministic parallel behavior will drive you insane
Building Blocks
Building Blocks

• Parallel MIP is parallel branch-and-bound:

Available for simultaneous processing
Deterministic Parallel MIP

• Multiple phases
• In each phase, on each processor:
  – Explore nodes assigned to processor
  – Report back results
    • New active nodes
    • New solutions
    • New cuts
    • Etc.
• One approach to node assignment:
  – Assign a subtree to each processor
  – Limit amount of exploration in each phase
Subtree Partitioning

• Problem:
  – Subtree may quickly prove to be uninteresting
    • Poor relaxation objectives
      – May want to abandon it
    • Pruned quickly
      – Leaves processor idle
More Global Partitioning

• Node coloring: assign a color to every node

- Processor can only process nodes of the appropriate color
- New child node same color as parent node
- Perform periodic re-coloring
More Dynamic Node Processing

• Allows much more flexibility
  – Processor can choose from among many nodes of the appropriate color

• *Deterministic priority queue* data structure required to support node coloring
  – Single global view of active nodes
  – Support notion of node color
    • Processor only receives node of the appropriate color
  – Efficient, frequent node reallocation
Heuristics
Summary of Heuristics

• 5 heuristics prior to solving root LP
  – 5 different variable orders, fix variables in this order

• 15 heuristics within tree (9 primary, several variations)
  – RINS, rounding, fix and dive (LP), fix and dive (Presolve), Lagrangian approach, pseudo costs, Hail Mary (set objective to 0)

• 3 solution improvement heuristics
  – Applied whenever a new integer feasible is found
Cutting Planes
Cutting Planes

- Gomory
  - Gu ISMP 2006, strengthen by lifting in GUBs
- Flow covers
  - Strengthened by lifting before un-transforming
- MIR
  - Different aggregation for MIR and flow covers
- Knapsack covers
- Clique
- Implied bound
- Flow paths
  - Results fed into flow covers
- GUB covers
Performance
Performance Benchmarks

• Performance test sets:
  – Mittelmann optimality test set:
    • 55 models, varying degrees of difficulty
    • http://plato.asu.edu/ftp/milpc.html
  – Mittelmann feasibility test set:
    • 34 models, difficult to find feasible solutions
    • http://plato.asu.edu/ftp/feas_bench.html
  – Our own broader test set:
    • A set of 263 models that require between one minute and one hour to solve on one core
      – Publicly available models, plus a few customer models

• Test platform:
  – Q9450 (2.66 GHz, quad-core system)
Performance Results

• Two basic questions:
  – Is P=1 efficient?
  – How much does performance improve with P>1?

• P=1 efficiency (geometric means):
  – Mittelmann optimality test set
    • Gurobi 1.1 is 1.18X slower than CPLEX 12.0
  – Mittelmann feasibility test set
    • Gurobi 1.1 is 2.3X faster than CPLEX 12.0
Parallel Performance

Comparisons:

- Gurobi
  - 1.45X faster than CPLEX 12.0 for p=4 on Mittelmann
  - Gurobi p=1 versus p=4
    - 1.73X speedup on Mittelmann
    - 1.61X speedup on broader set
- CPLEX p=1 versus p=4
  - 1.21X speedup for CPLEX 12 on Mittelmann (non-deterministic)
  - 1.29X speedup for CPLEX 11 on their broader set
Speedups for P=4 (By Model)
Gurobi 2.0

• Improve simplex solver
• Cutting planes
  – Add missing cutting planes
• Add missing presolve reductions
Thank You