Computational Progress in Linear and Mixed Integer Programming

Gurobi
Optimization

Overview

- Linear Programming
  - Historical perspective
  - Computational progress
- Mixed Integer Programming
  - Introduction: what is MIP?
  - Solving MIPs: a bumpy landscape
  - Computational progress

The Early History

- 1947 – George Dantzig invents simplex method for LP
  - Introduced the idea of an objective function
  - Promoted the idea of actually using LP to make decisions
  - 4 Nobel Prizes in LP
  - First LP solved: Laderman (1947), 9 cons., 77 vars., 120 man-days.
- 1951 – First computer code for solving LPs
- 1960 – LP commercially viable
  - Used largely by oil companies
- 1970 – MIP commercially viable
  - MPSX/370, UMPIRE

Gurobi Optimization

The Decade of the 70's

- **Interest in optimization flowered**
  - Numerous new applications identified
    - Large scale planning applications particularly popular

- **Significant difficulties emerged**
  - Building application was very expensive and very risky
    - 3-4 year development cycles
    - Developers and application owners had to be multi-faceted experts: Computer, Data, Algorithm, and Modeling skills necessary.
      - "Deploying an application was virtually impossible"
    - Technology just wasn't ready: LPs were hard and MIP was a disaster
  - **Result:** *Disillusionment and much of that disillusionment persists to this day.*

Gurobi Optimization

The Decade of the 80’s

- Mid 80’s:
  - There was perception was that LP software had progressed about as far as it could go – MPSX/370 and MPSIII
  - BUT LP was definitely ***not a solved problem*** … example: “Unsolvable” airline LP model with 4420 constraints, 6711 variables
- There were several key developments
  - IBM PC introduced in 1981
  - Relational databases developed:
    - Separation of logical and physical allocation of data.
    - ERP systems introduced.
  - Karmarkar’s 1984 paper on interior-point methods

The Decade of the 90's

- LP performance takes off
  - LP software becomes embeddable and flexible
  - Algorithms
    - Primal-dual log-barrier algorithms completely reset the bar
    - Simplex algorithms unexpectedly kept pace

- Data became plentiful and accessible
  - ERP systems became commonplace

- Popular new applications begin to show that Optimization could work on difficult, real problems
  - Business: Airlines, Supply-Chain
  - Academic: Traveling Salesman Problem

Linear Programming

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz Pentium 4):

- Test: Went back to 1st CPLEX (1988)
- 1988 (CPLEX 1.0): Houston, 13 Nov 2002

Gurobi Optimization

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz Pentium 4):

- Test: Went back to 1st CPLEX (1988)
- 1988 (CPLEX 1.0): 8.0 days (Berlin, 21 Nov)

Gurobi Optimization

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz Pentium 4):

- Test: Went back to 1st CPLEX (1988)
- 1988 (CPLEX 1.0): 15.0 days (Dagstuhl, 28 Nov)

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz Pentium 4):

- Test: Went back to 1st CPLEX (1988)
- 1988 (CPLEX 1.0): 19.0 days (Amsterdam, 2 Dec)

Gurobi Optimization

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz Pentium 4):

- Test: Went back to 1st CPLEX (1988)
- 1988 (CPLEX 1.0): 23.0 days (Houston, 6 Dec)

Gurobi Optimization

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz Pentium 4):

		Speedup
◦ Test: Went back to 1st CPLEX (1988)		
◦ 1988 (CPLEX 1.0):	29.8 days	1x
◦ 1997 (CPLEX 5.0):	1.5 hours	480x
◦ 2003 (CPLEX 9.0):	59.1 seconds	43500x

The algorithm: Dantzig's primal simplex algorithm!

Gurobi Optimization

Progress in LP: 1988—2004

(<u>Operations Research</u>, Jan 2002, pp. 3—15, updated in 2004)

- Algorithms (*machine independent*):

  Primal *versus* best of Primal/Dual/Barrier 3,300x
- Machines (workstations →PCs): 1,600x
- NET: Algorithm × Machine 5,300,000x

  (2 months/5300000 ~= 1 second)

Gurobi Optimization

Progress in LP: 1988—2015

- All times relative to Dual Simplex (> 1.0x => Dual faster)

- Algorithm comparison – 2004 (CPLEX)
  - Dual simplex vs. primal: 2.70x
  - Dual simplex vs. barrier: 1.06x

- Algorithm comparison – 2015 (Gurobi)
  - Dual simplex vs. primal: 2.11x
  - Dual simplex vs. barrier: 0.51x
  - Dual simplex vs. concurrent: 0.40x

Gurobi Optimization

LP Today

- Practitioners consider LP a solved problem
- Large models can now be solved robustly and quickly
  - Regularly solve models with millions of variables and constraints

LP Today

- However, a word of warning …
  - Real applications still exist where LP performance is an issue
    - ~2% of MIPs are blocked by LP performance
    - Challenging pure-LP applications persist
      - Ex: Power industry (Financial Transmission Right Auctions)
  - **Challenge:** Further research in LP algorithms is needed (there has been little progress since 2004)

Gurobi Optimization

Mixed Integer Programming

A Definition

A *mixed-integer program* (MIP) is an optimization problem of the form

$$
\begin{aligned}
\textit{Minimize} \quad & c^T x \\
\textit{Subject to} \quad & Ax = b \\
& l \le x \le u \\
& \text{some or all } x_j \text{ integer}
\end{aligned}
$$

Customer Applications

(2012 Gurobi Sales – 200+ new customers)

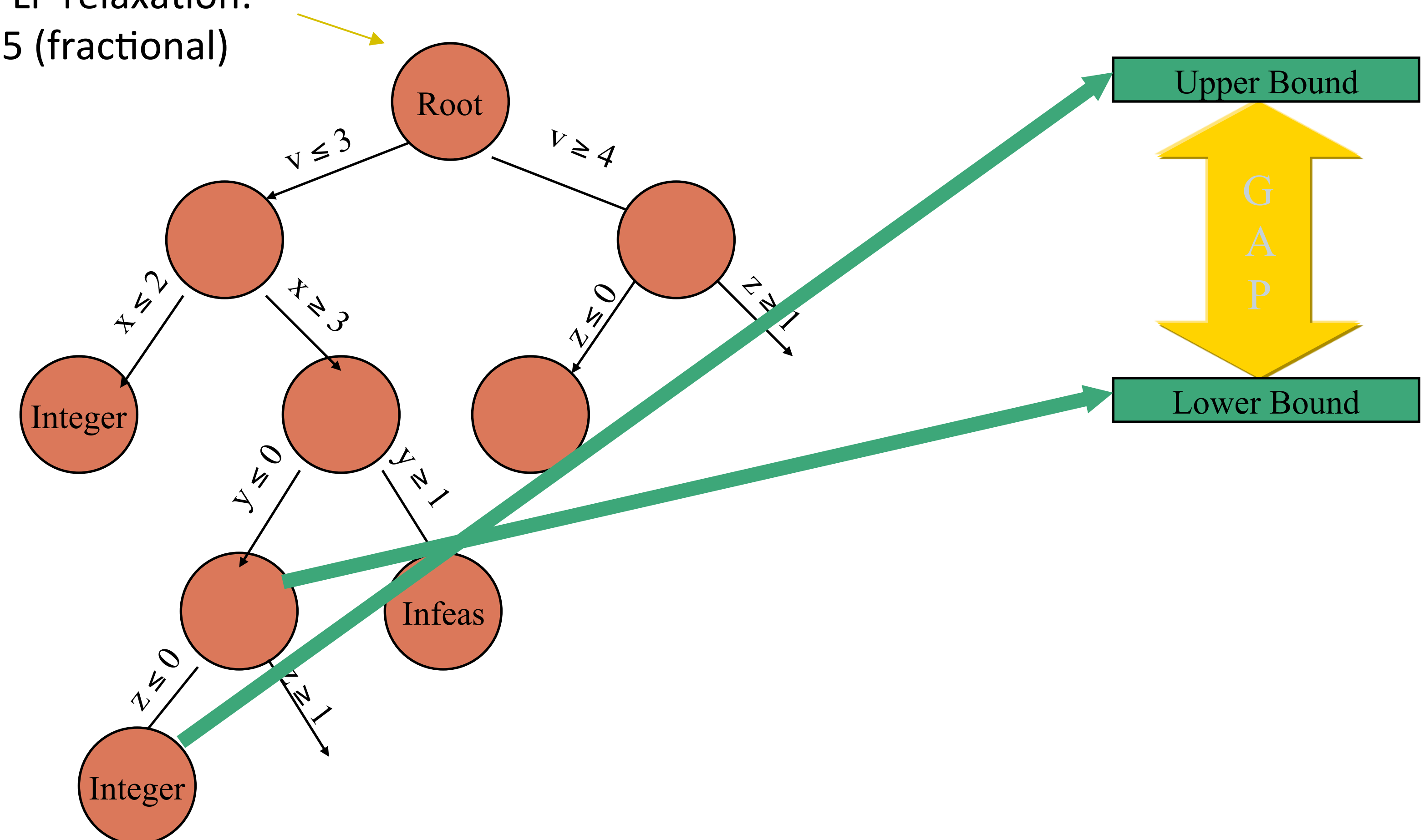
- Accounting
- Advertising
- Agriculture
- Airlines
- ATM provisioning
- Compilers
- Defense
- Electrical power
- Energy
- Finance
- Food service
- Forestry
- Gas distribution
- Government
- Internet applications
- Logistics/supply chain
- Medical
- Mining
- National research labs
- Online dating
- Portfolio management
- Railways
- Recycling
- Revenue management
- Semiconductor
- Shipping
- Social networking
- Sourcing
- Sports betting
- Sports scheduling
- Statistics
- Steel Manufacturing
- Telecommunications
- Transportation
- Utilities
- Workforce Management

Gurobi Optimization

Solving MIPs

MIP solution framework:
LP based Branch-and-Bound

Solve LP relaxation:
v=3.5 (fractional)



Remarks:

(1) GAP = 0 $\Rightarrow$ Proof of optimality

(2) In practice: Often good enough to have good Solution

A Bumpy Solution Landscape

Example 1: LP still can be HARD

SGM: Schedule Generation Model

157323 rows, 182812 columns

- LP relaxation at root node:
  - 18 hours
- Branch-and-bound
  - 1710 nodes, first feasible
  - 3.7% gap
  - Time: 92 days!!
- MIP does not appear to be difficult: *LP is a roadblock*

Gurobi Optimization

Example 2: MIP really is HARD

A customer model: 44 constraints, 51 variables, maximization
51 general integer variables (*and no bounds*)

Branch-and-bound:	Initial integer solution	-2186.0
	Initial upper bound	-1379.4

…after 1.4 days, 32,000,000 B&B nodes, 5.5 Gig tree

Integer solution and bound: UNCHANGED

What's wrong? Bad modeling. Free GIs chase each other off to infinity.

Gurobi Optimization

Example 2: Here's what's wrong

```
Maximize
  x + y + z
Subject To
  2 x + 2 y ≤ 1
  z <= 0
  x, y, z free
  x,y integer
```

Note: This problem can be solved in several ways

- Fix z to 0 and remove, objective is integral [*Presolve*]
- Euclidean reduction on the constraint [*Presolve*]

However: Branch-and-bound cannot solve!

Example 3: A typical situation today – Supply-chain scheduling

- Model description:
  - Weekly model, daily buckets: Objective to minimize end-of-day inventory.
  - Production (single facility), inventory, shipping (trucks), wholesalers (demand known)
- Initial modeling phase
  - Simplified prototype + complicating constraints (production run grouping req't, min truck constraints)
  - RESULT: Couldn't get good feasible solutions.
- Decomposition approach
  - Talk to current scheduling team: They first decide on "producibles" schedule. Simulate using heuristics.
  - Fixed model: Fix variables and run MIP

Gurobi Optimization

Supply-chain scheduling (continued): Solving the fixed model

CPLEX 5.0 (1997):

```
Integer optimal solution (0.0001/0):  Objective =    1.5091900536e+05
Current MIP best bound =    1.5090391809e+05 (gap = 15.0873)
Solution time = 3465.73 sec.  Iterations = 7885711  Nodes = 489870 (2268)
```

CPLEX 11.0 (2007):

```
Implied bound cuts applied:  60
Flow cuts applied:  85
Mixed integer rounding cuts applied:  41
Gomory fractional cuts applied:  29

MIP - Integer optimal solution:  Objective =  1.5091900536e+05
Solution time =    0.63 sec.  Iterations = 2906  Nodes = 12
```

Original model: Now solvable to optimality in ~100 seconds (20% improvement in solution quality)

Gurobi Optimization

Computational History: 1950 -1998

- 1954 Dantzig, Fulkerson, S. Johnson: 42 city TSP
  - Solved to optimality using LP and cutting planes
- 1957 Gomory
  - Cutting plane algorithms
- 1960 Land, Doig; 1965 Dakin
  - B&B
- 1969 LP/90/94
  - First commercial application (British Pet, ref: Max Shaw)
- IBM 360 computer
  - 1974 MPSX/370
  - 1976 Sciconic
    - LP-based B&B
    - MIP became commercially viable

- 1975 – 1998 Good B&B remained the state-of-the-art in commercial codes, in spite of ….
  - Edmonds, polyhedral combinatorics
  - 1973 Padberg, cutting planes
  - 1973 Chvátal, revisited Gomory
  - 1974 Balas, disjunctive programming
  - 1983 Crowder, Johnson, Padberg: PIPX, pure 0/1 MIP
  - 1987 Van Roy and Wolsey: MPSARX, mixed 0/1 MIP
  - TSP, Grötschel, Padberg, …

Gurobi Optimization

1998 … A New Generation of MIP Codes

- Linear programming
  - Stable, robust dual simplex
- Variable/node selection
  - Influenced by traveling salesman problem
- Primal heuristics
  - 12 different tried at root
  - Retried based upon success
- Node presolve
  - Fast, incremental bound strengthening (very similar to Constraint Programming)
- Presolve – numerous small ideas
  - Probing in constraints:

    $\sum x_j \leq (\sum u_j)\, y, \quad y = 0/1$

    $\Rightarrow x_j \leq u_j y$ (for all j)
- Cutting planes
  - Gomory, mixed-integer rounding (MIR), knapsack covers, flow covers, cliques, GUB covers, implied bounds, zero-half cuts, path cuts

Gurobi Optimization

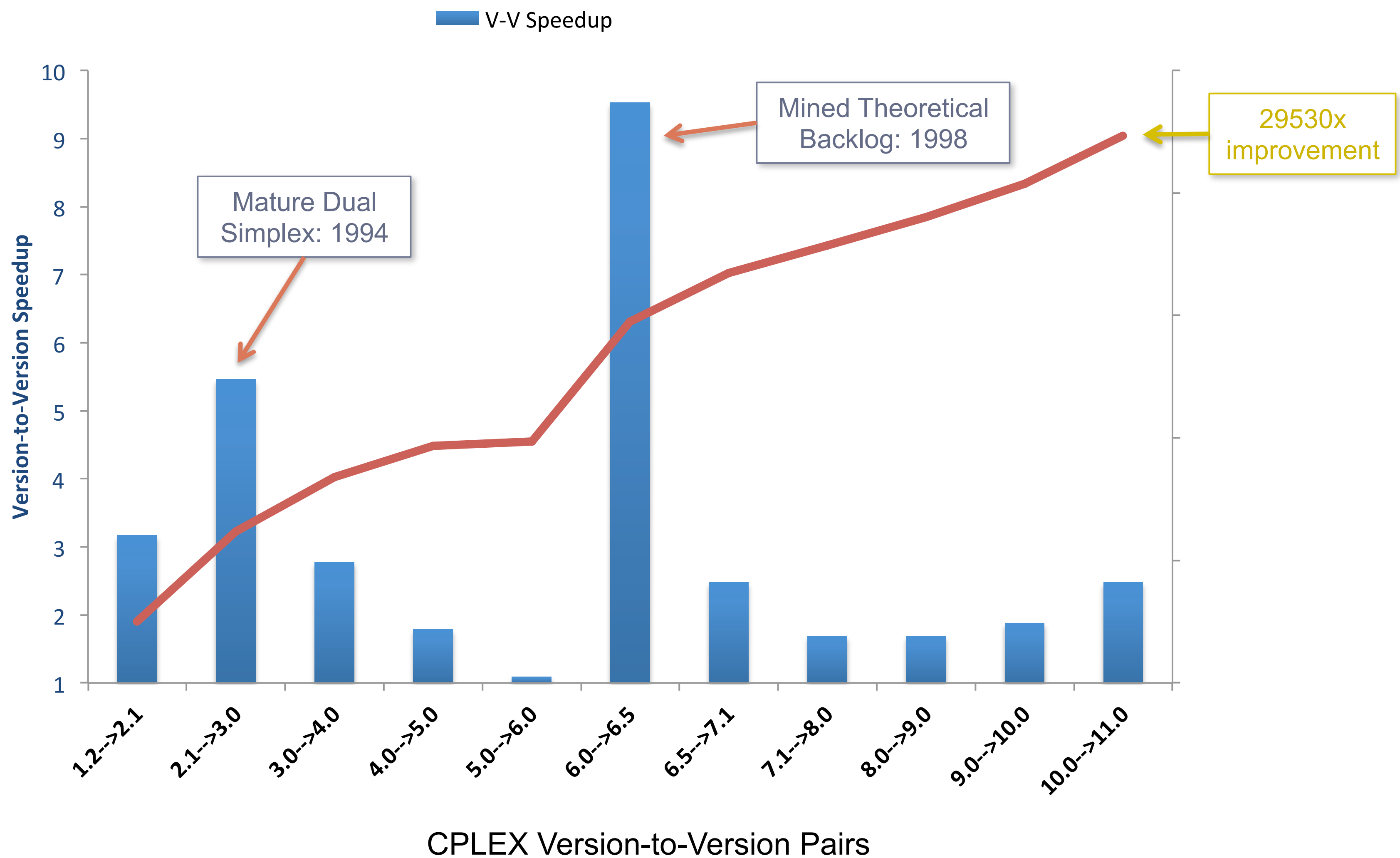
MIP Speedups

Some Test Results

- Test set: **1852 real-world MIPs**
  - Full library
    - 2791 MIPs
  - Removed:
    - 559 "Easy" MIPs
    - 348 "Duplicates"
    - 22 "Hard" LPs (0.8%)
- Parameter settings
  - Pure defaults
  - 30000 second time limit
- Versions Run
  - CPLEX 1.2 (1991) -- CPLEX 11.0 (2007)

Gurobi Optimization

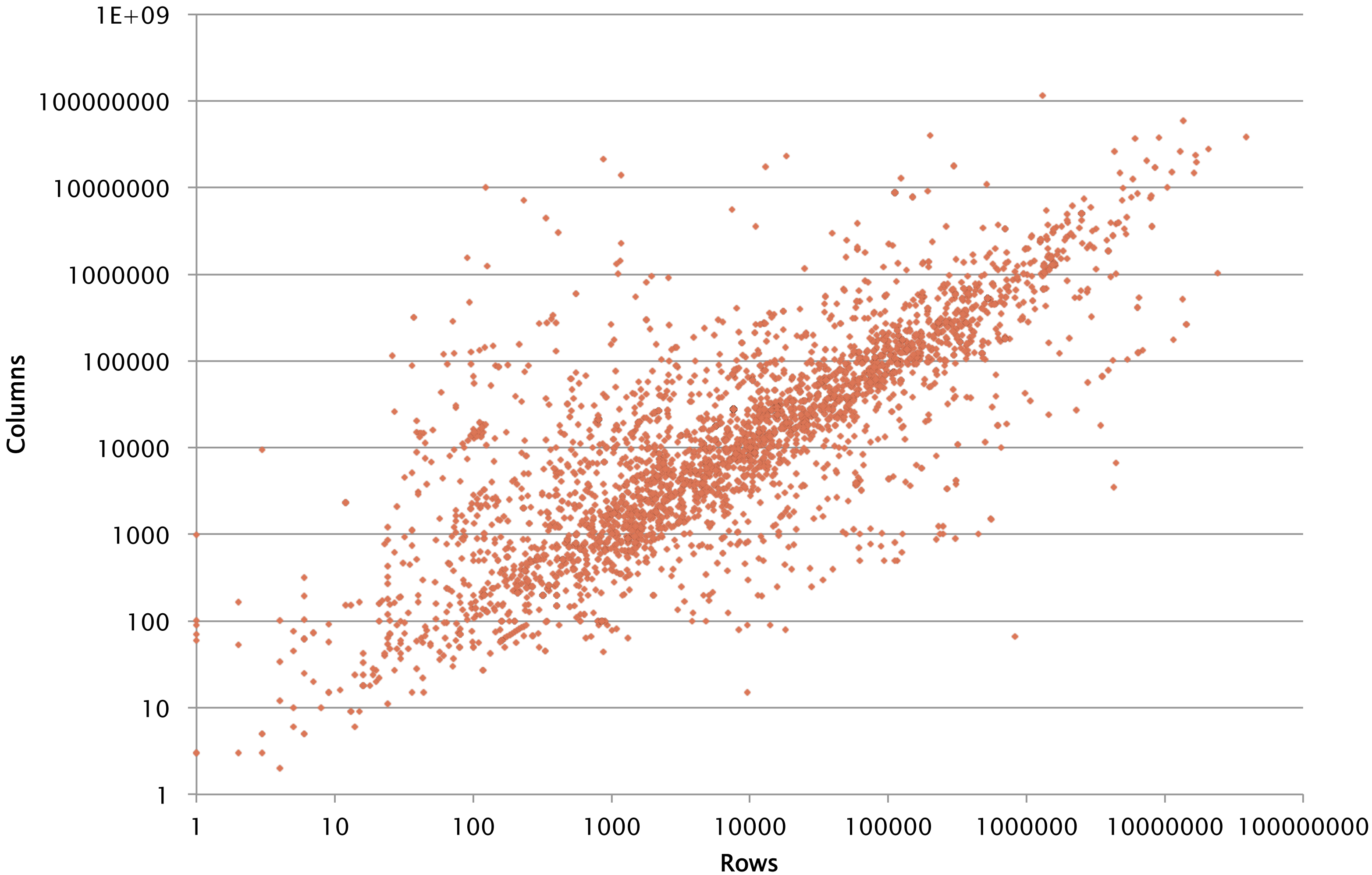
CPLEX Version Performance Improvements
(1991–2008)



Progress: 2009 – Present

Gurobi MIP Library

(3550 models)



MIP Speedup 2009-Present

- Starting point
  - Gurobi 1.0 & CPLEX 11.0 ~equivalent on 4-core machine

- Gurobi Version-to-version improvements
  - Gurobi 1.0 -> 2.0: 2.4X
  - Gurobi 2.0 -> 3.0: 2.2X (5.1X)
  - Gurobi 3.0 -> 4.0: 1.3X (6.6X)
  - Gurobi 4.0 -> 5.0: 2.0X (12.8X)
  - Gurobi 5.0 -> 6.0: 2.2X (27.6X)
  - Gurobi 6.0 -> (6.5): 1.4X (38.6X)

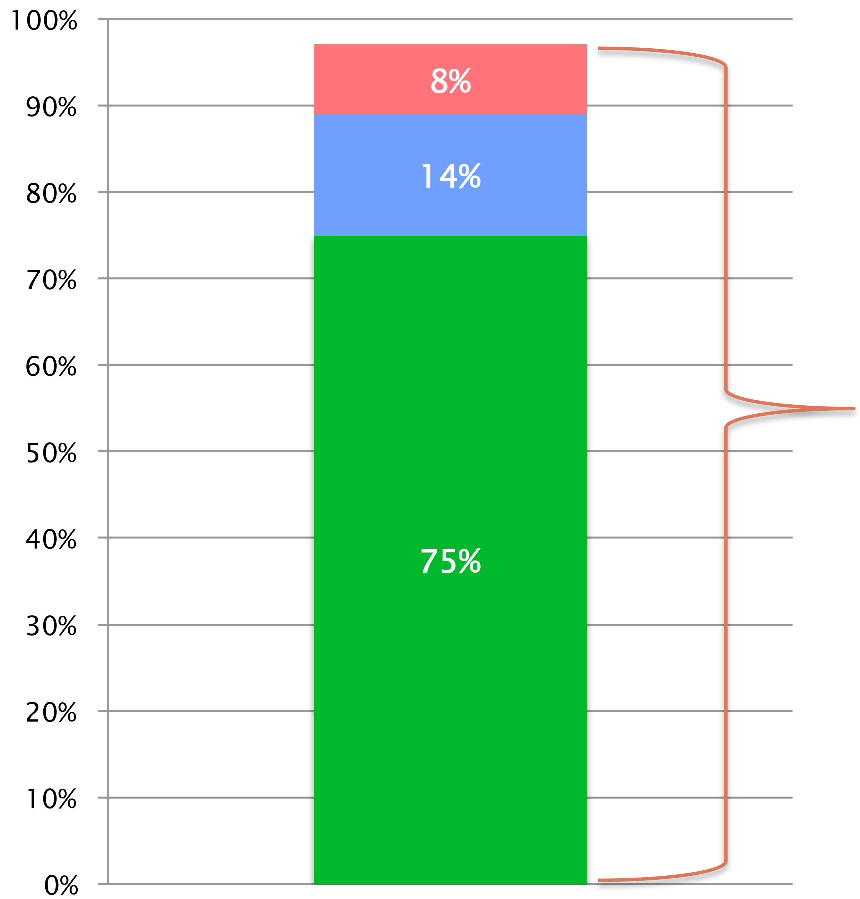
- Machine-independent IMPROVEMENT since 1991
  - Over 1.1M X -- 1.8X/year

Gurobi Optimization

MIP Solvability

Solvability of MIPs – Gurobi (6.5)

3550 MIPs, 30000 second time limit, run with defaults



97% (107 MIPs found no solution)

- 54 blocked by LP, ~1.5%
- 16 tunable
- 37 remain, ~1%

Suppose you were given the following choices:

- **Option 1**: Solve a MIP with today's solution technology on a machine from 1991
- **Option 2**: Solve a MIP with 1991 solution technology on a machine from today

Which option should you choose?

- **Answer**: Option 1 would be faster by a factor of approximately ~300.

Questions?