Greg Glockner, Ph.D.
VP and Technical Fellow

March 2024

GUROBI
OPTIMIZATION

Using NVIDIA Grace

For better performance and energy efficiency

GUROBI OPTIMIZATION

Agenda

About Math Optimization
And why should you care!

Computational Optimization

Benchmarking on Grace

About Mathematical Optimization

Why it matters

GUROBI OPTIMIZATION

Planning, scheduling and allocation decisions made everywhere

- The flight you took … scheduled by mathematical optimization
- The hotel room you booked … priced by mathematical optimization
- The route you took by car or Uber … selected by mathematical optimization
- The GTC conference program (…ought to have been) scheduled by mathematical optimization
- The food you eat comes from farms … planned via mathematical optimization

These are just a few examples

Gurobi was founded in 2008 by the sharpest minds in computational mathematical optimization

Their vision was simple:

Endeavor to build the world's most powerful mathematical optimization solver.

About Mathematical Optimization

Gurobi everywhere

GUROBI OPTIMIZATION

2,500+
global customers

40+
industries

ABB	AIRFRANCE	AMD	AT&T	Bank of America
bhpbilliton	BM&F BOVESPA A Nova Bolsa	CSX	DAIMLER	Department of Defense United States of America
ESTĒE LAUDER	FCC Federal Communications Commission	Ferrari	Google	HUAWEI
LEANLOGISTICS	Lufthansa	Marriott	Microsoft	NFL
nielsen	Nokia Siemens Networks	New York ISO Independent System Operator	ORACLE	ProCom einschneidend besser
Prudential	SAP	SCHNEIDER	SIEMENS	Starbucks Coffee
TransUnion	Uber	Walmart	WM WASTE MANAGEMENT	YUTONG

About Mathematical Optimization

Solving the most complex problems across industries

Solving the NFL Season Schedule

Gurobi considers over 824 trillion possible combinations, with constraints around travel, broadcast rights, matchups, marathons, concerts, and more.

SAP Supply Chain Optimization

The Gurobi Optimizer solves some of the most complex problems for SAP Supply Chain and Advanced Planning customers around the world.

NYISO Wholesale Electrical Power Optimization

Relies on Gurobi to optimize 500 power-generation units and 11,000 miles of transmission lines to meet consumer demand in real-time.

About Mathematical Optimization

Computationally intensive

- Perpetual demand to solve faster
  - Increase planning horizon
  - Consider more scenarios

- Always looking for improvements from hardware and algorithms

About Mathematical Optimization

Linear Programming (LP)

GUROBI OPTIMIZATION

Solve for variables x:

$$
\begin{aligned}
\min c^T x \\
Ax &= b \\
x &\geq 0
\end{aligned}
$$

goal – “objective”

m x n matrix of activities – “coefficients”

n-dimensional array of choices – “decision variables”

m-dimensional array of requirements – “constraints”

About Mathematical Optimization

Mixed Integer Programming (MIP)

Solve for variables x:

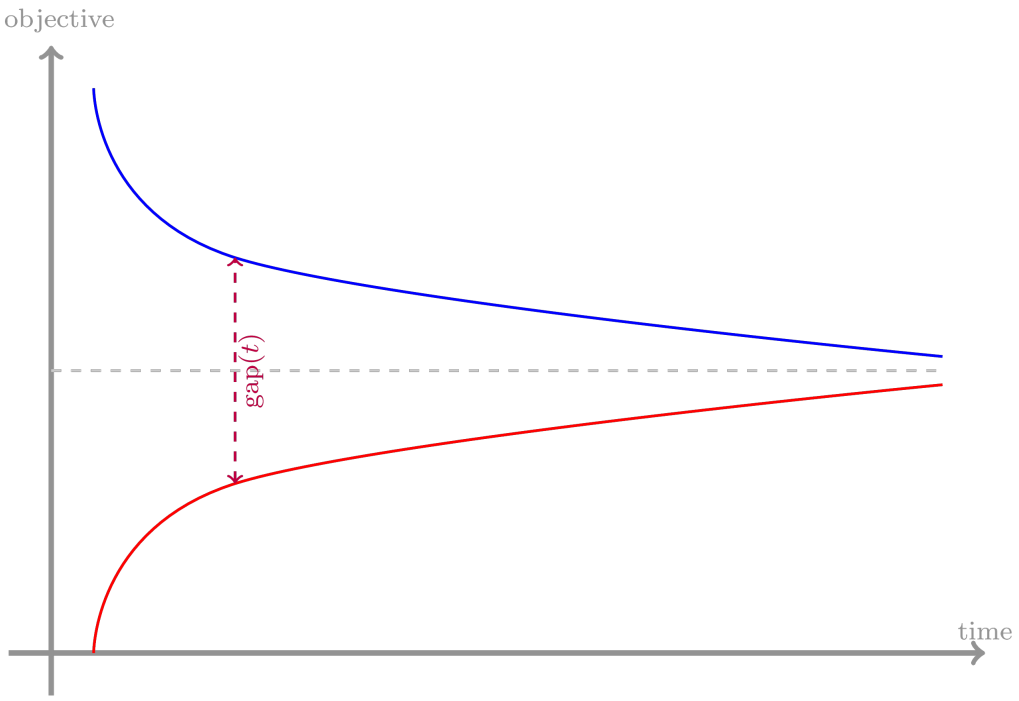
$$\begin{aligned} \min c^T x & \\ Ax &= b \\ x &\geq 0 \\ \text{Some } x_i \text{ integer} & \end{aligned}$$

- Most real-world applications require integer variables due to alternate choices
- Much harder computationally
- Can generalize to quadratic and nonlinear functions
  - Can also cast or approximate (“reformulate”) any function in this form

About Mathematical Optimization

Algorithm basics

- Two types of information
  - Upper bound: Feasible solutions that meet constraints (requirements)
  - Lower bound: Theoretical best solution possible
- Difference is known as the “gap”
- We are optimal when these converge through any combination of
  - Upper bound: Improving the best known solution
  - Lower bound: Tightening proof of optimality

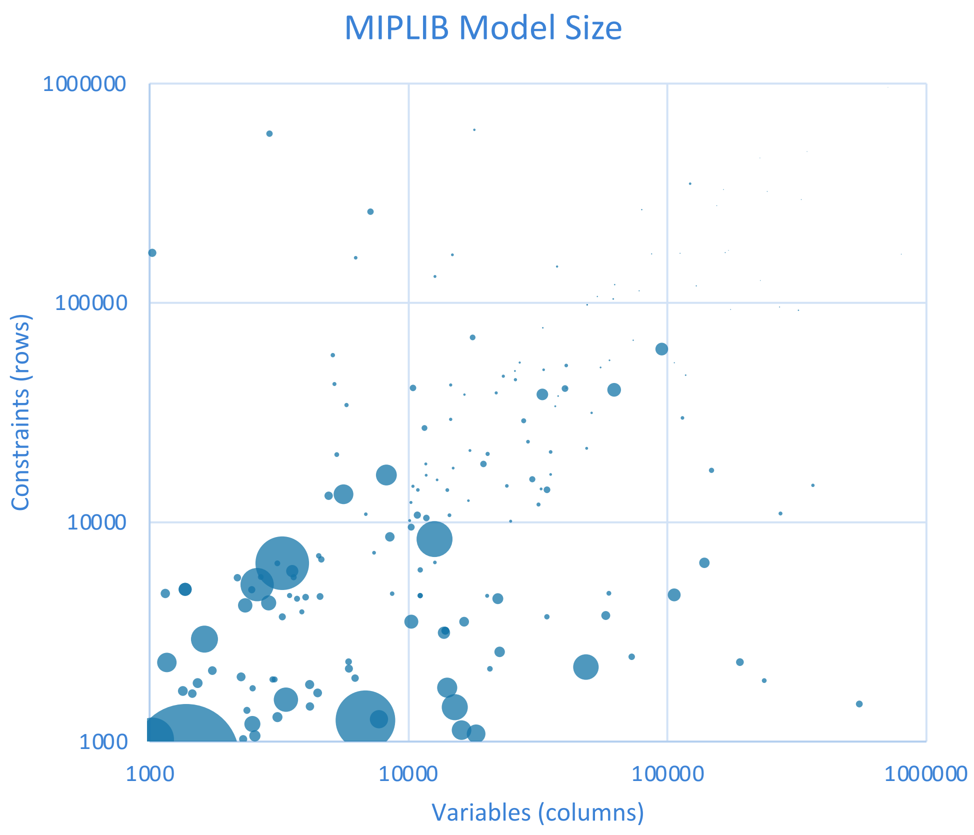


Computational Optimization

Sparsity

- In most applications, nearly every element in the coefficient matrix is zero
  - Max density (largest dot): 32.65%
  - Median density: 0.14%
  - Min density: 0.00062%

- Correct handling of sparsity is key to scaling to very large applications with millions of variables and constraints
  - Only store the nonzero elements
  - Want computation to preserve sparsity



Computational Optimization

Key algorithm 1: Simplex method

- Partition feasibility condition $Ax = b \rightarrow Bx_B + Nx_N = b$
  - B is square matrix called the basis
- Set $x_N = 0$ and solve for x_B via LU-factorization
  - Don't compute B^{-1} directly
- Swap columns between B and N matrices
  - Select column that improves the objective function $\min c^T x$
  - Stop when there is no improving direction
    - This is optimality condition
  - Since only 1 column changes, we make incremental updates

Computational Optimization

Key algorithm 2: Barrier method

- Write conditions of optimal solution: want $\mu = 0$ in:

$$\begin{aligned} Ax &= b \\ A^T y + z &= c \\ XZe - \mu e &= 0 \end{aligned}$$

- From initial point (x_0, y_0, z_0), use Newton method to find improving direction $(\Delta_x, \Delta_y, \Delta_z)$ plus a step size on μ:

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ Z_0 & 0 & X_0 \end{pmatrix} \begin{pmatrix} \Delta_x \\ \Delta_y \\ \Delta_z \end{pmatrix} = \begin{pmatrix} Ax_0 - b \\ A^T y_0 + z_0 - c \\ X_0 Z_0 e - \mu e \end{pmatrix}$$

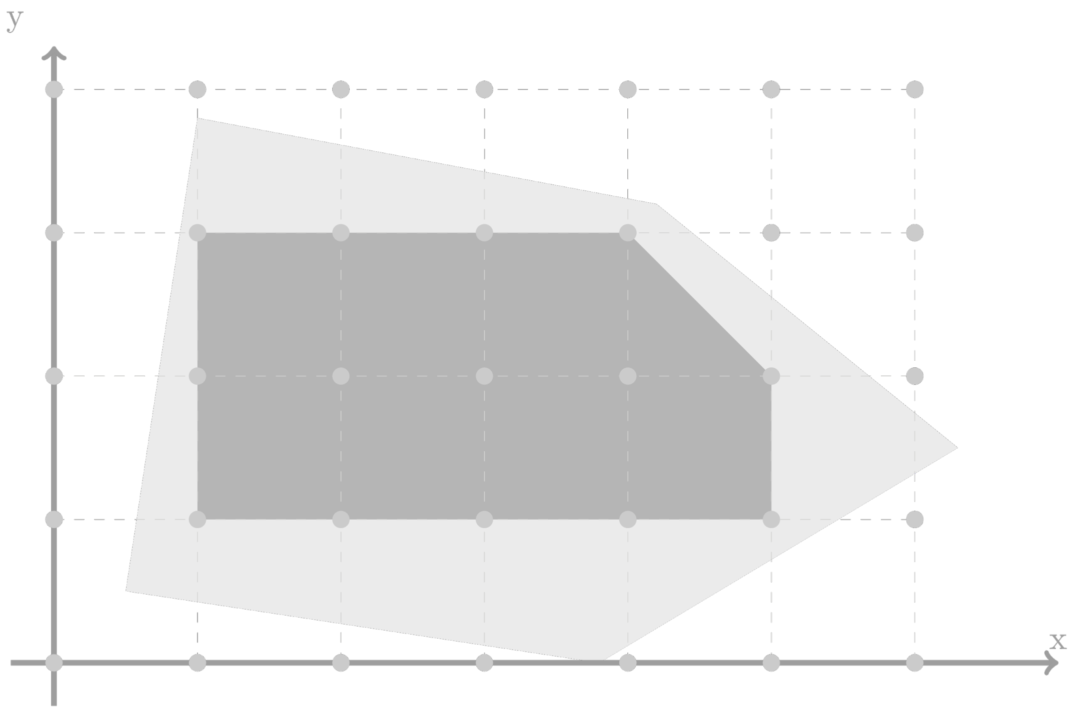
- Most computational work involves solving Cholesky factorizations in:

$$AZ_0^{-1} X_0 A^T \, \Delta_y = b - Z_0^{-1} \mu e - AZ_0^{-1} X_0 (A^T y_0 + z_0 - c)$$

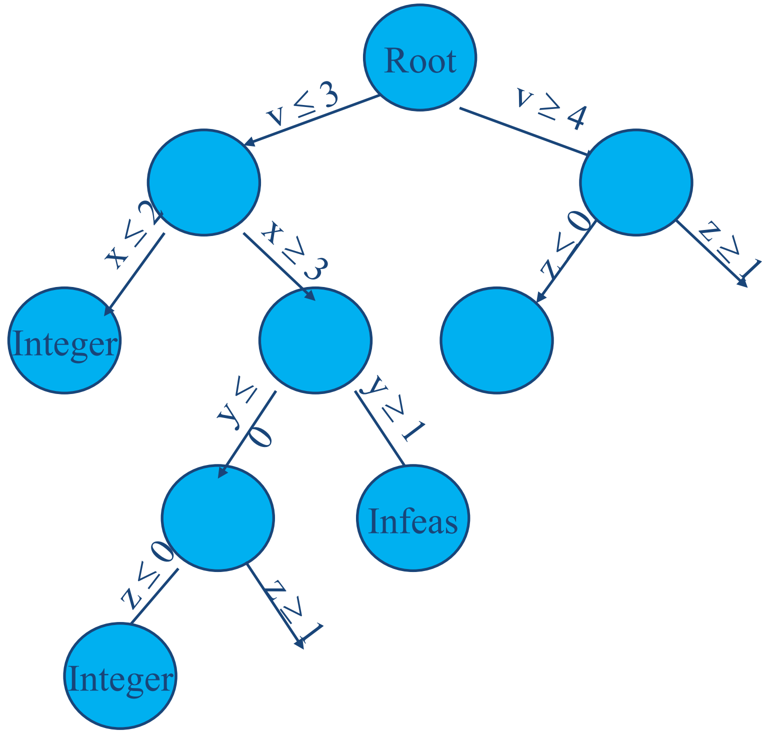
Computational Optimization

Key algorithm 3: MIP branch-and-cut

- Cut generation
  - Use geometry to find valid inequalities that cutoff part of fractional region



- Branch and bound: Relax integer restrictions
  - Pick a fractional value
  - Create 2 new models; solve them incrementally
  - Iterate until gap between best solution and proof is tight



Computational Optimization

Parallelization

- Challenges for parallelization
  - Solution algorithms are sequential
  - Fast performance depends heavily on incremental updates

- Top solvers like Gurobi do Symmetric Multi-Processing
  - MIP search tree
  - Barrier Cholesky factorization
- GPU computing not used historically but this may be changing

- Distributed parallelization across multiple computers is possible but only helpful in a few specific cases

Benchmark results are preliminary

Disclaimer

- Gurobi just received a computer with NVIDIA Grace Hopper Superchip in February
- Gurobi Optimizer is not yet highly tuned for
  - ARM processors
  - Large core count – most customers have 4-16 CPU cores
  - GPU computing

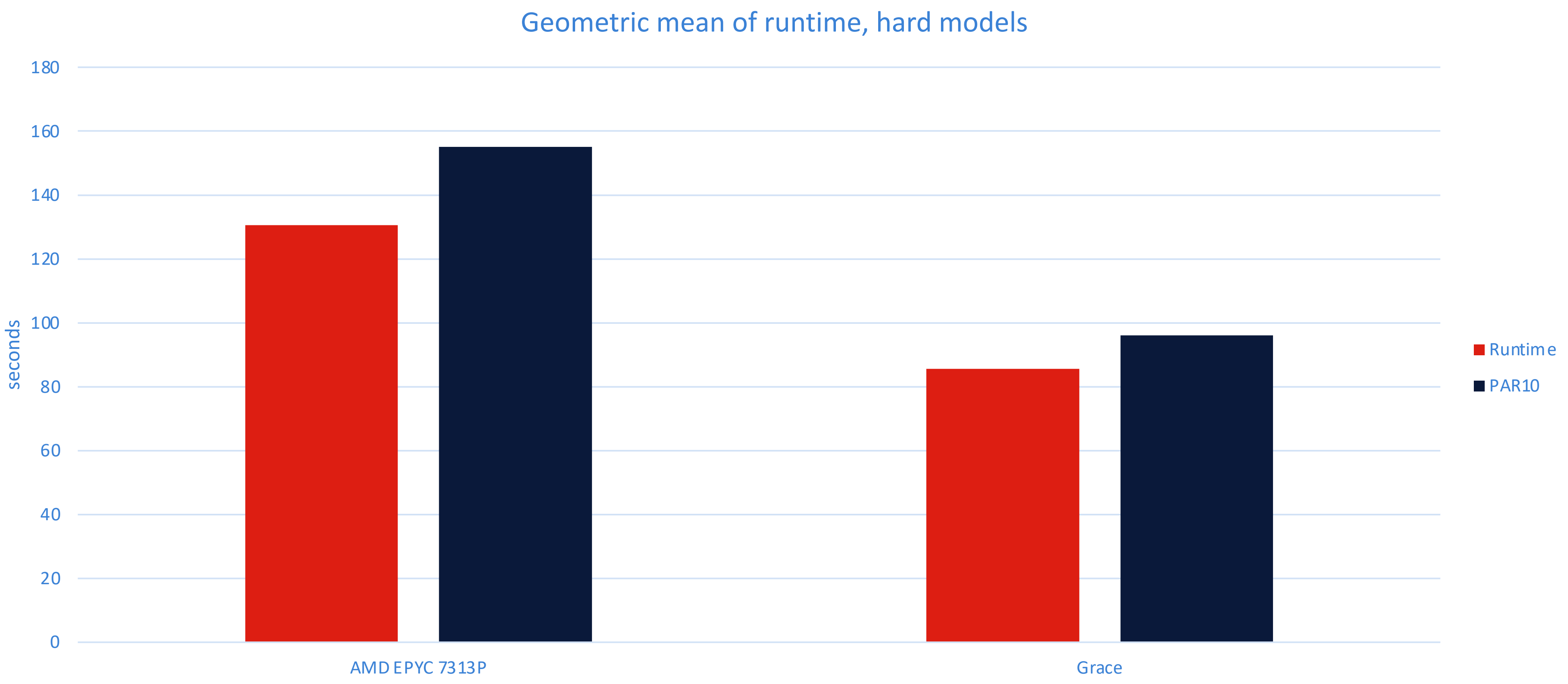
Benchmarking on NVIDIA Grace Hopper

Test platform

- Data
  - MIPLIB: https://miplib.zib.de
    - MIPLIB 2017 is the latest – the 6th edition
  - Applications collated by committee of 20+ from academia and industry
  - Benchmark set contains 240 optimization instances
  - Goal: to be representative of real-world applications
- Systems
  - Single NVIDIA Grace Hopper Superchip (72 cores, 480GB RAM)
  - Cluster with 3rd generation AMD EPYC ("Milan") (7313P, 16 cores, 256GB DDR4 RAM)
  - Gurobi Optimizer 11.0 on Ubuntu 22.04
  - Network and storage have no significant impact on Gurobi performance

Preliminary Gurobi tests: NVIDIA Grace Hopper

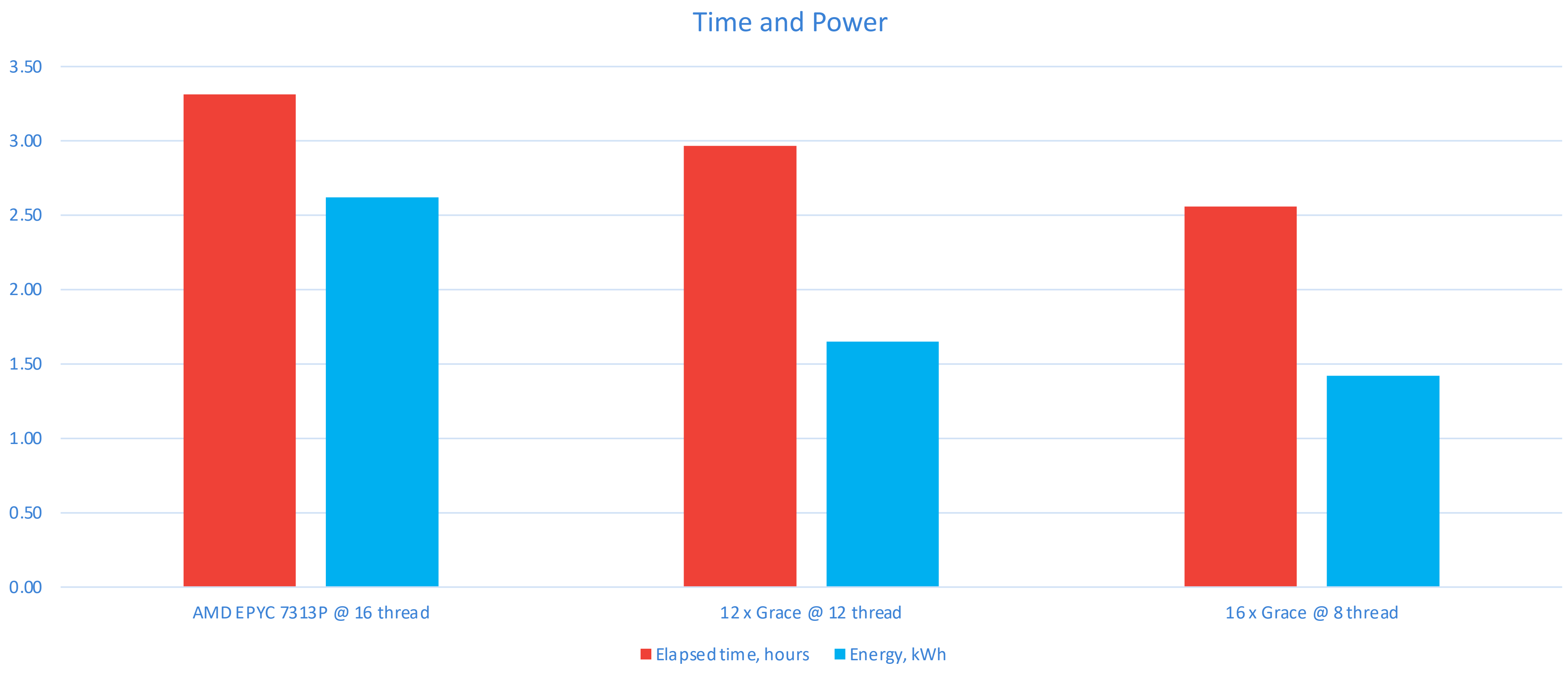
Runtime for hard models in MIPLIB Benchmark set



Smaller is faster

Preliminary Gurobi tests: NVIDIA Grace Hopper

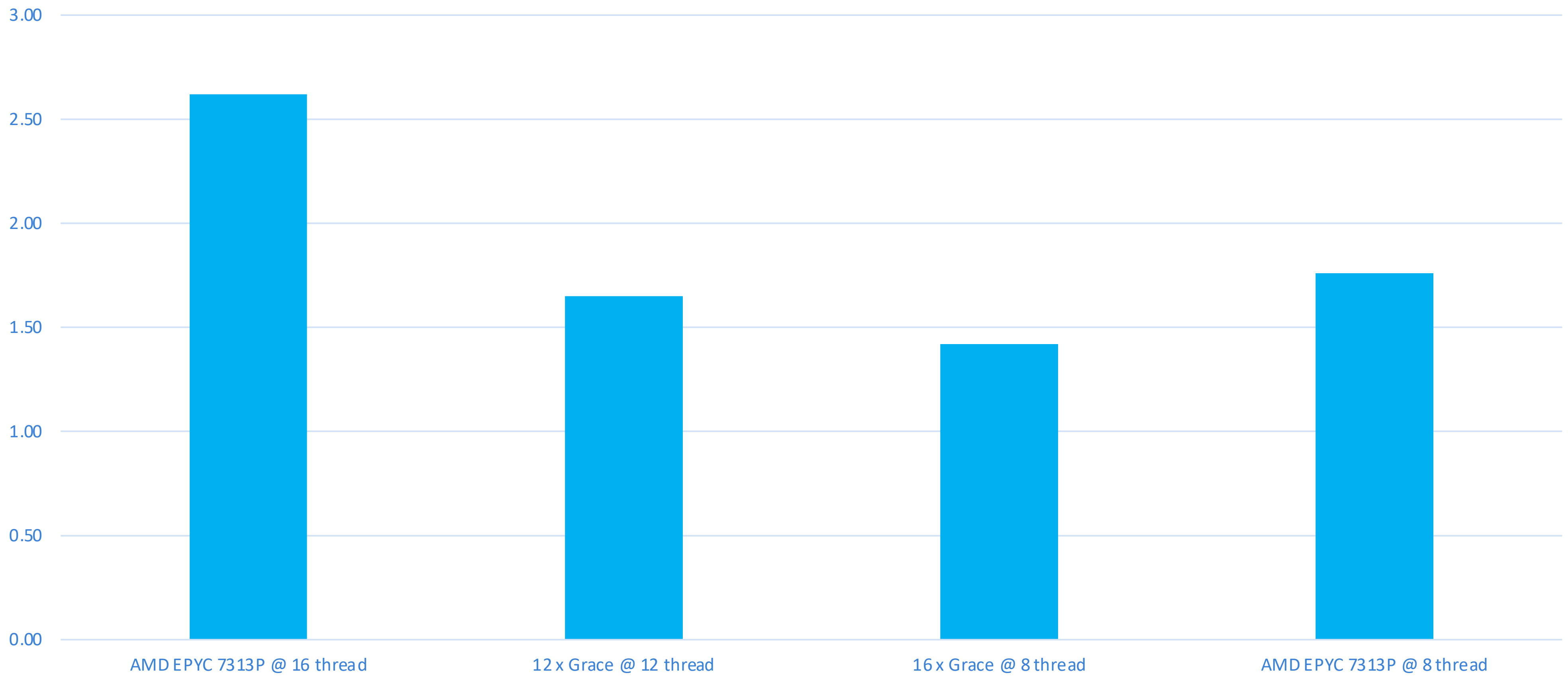
Throughput and energy for MIPLIB Benchmark set



Smaller is better

Preliminary Gurobi tests: NVIDIA Grace Hopper

Energy for MIPLIB Benchmark set, in kWh



Smaller is lower power

Improving the results for NVIDIA Grace Hopper

Next steps

- Tune Gurobi MIP performance for large number of cores
  - Concurrent algorithm strategies
  - Identify and address bottlenecks with increased parallelization
- Grace-specific tuning
  - ARM port of Gurobi Optimizer has not been tuned as much as our x86-64 port
- Testing the Hopper GPU
  - Preview of cuDSS GPU-accelerated Direct Sparse Solver for Grace Hopper just became available earlier this month

Greg Glockner, Ph.D.
VP and Technical Fellow

March 2024

GUROBI OPTIMIZATION

Thank You

For more information: gurobi.com