GUROBI OPTIMIZATION

Welcome

The webinar will begin shortly.

Speakers

GUROBI
OPTIMIZATION
What's New in
Gurobi 10.0
Tuesday, November 15
11 AM ET/5 PM CET
Tuesday, November 22
10 AM ET/4 PM CET
Tobias Achterberg
VP of Research &
Development
Michel Jaczynski
Sr Director of Cloud and
Platform R&D

Agenda

New Performance Techniques

Platform Features and WLS

API and Engine Features

Open-Source Github Repositories

New Performance Techniques

GUROBI OPTIMIZATION

Gurobi 10.0

Performance Summary

Performance improvements compared to Gurobi 9.5

Algorithm	Overall speed-up	On >100sec models
LP – default	10%	25%
LP – primal simplex	3%	10%
LP – dual simplex	3%	10%
MILP	13%	24%
Convex MIQP	57%	2.4x*
Convex MIQCP	28%	88%*
Non-convex MIQCP	51%	2.6x

* MIQP and MIQCP hard model test sets too small to give reliable benchmark results

GUROBI OPTIMIZATION

Gurobi 10.0

Performance Summary

Performance improvements compared to Gurobi 9.5

Algorithm	Overall speed-up	On >100sec models
LP – default	10%	25%
LP – primal simplex	3%	10%
LP – dual simplex	3%	10%
MILP	13%	24%
Convex MIQP	57%	2.4x*
Convex MIQCP	28%	88%*
Non-convex MIQCP	51%	2.6x

* MIQP and MIQCP hard model test sets too small to give reliable benchmark results

LP Performance

GUROBI OPTIMIZATION

- New network simplex algorithm
- Concurrent LP improvements: concurrent only on the final presolved model
- Crossover improvements
  - Parallel primal pushes
  - Barrier solution adjustment before pushes
- New and improved presolve reductions
  - Extend some MIP reductions to LP, like PreSparsify reduction
    - Handle dual and basis uncrush
  - New value 2 for parameter Aggregate

Network Simplex Algorithm

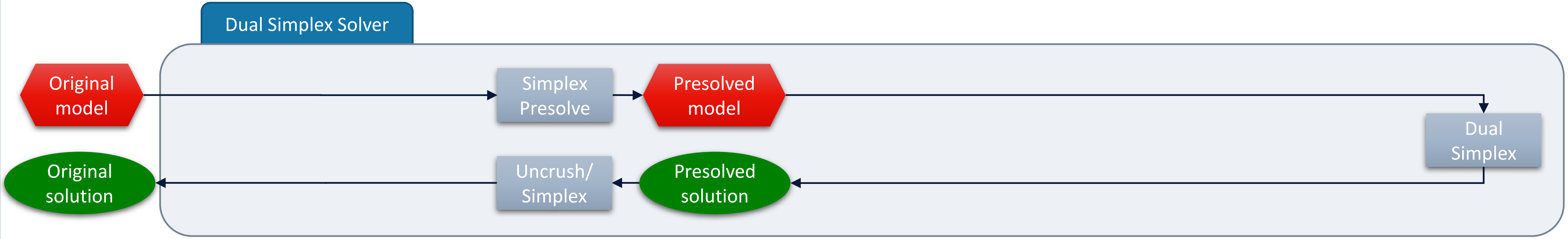
- Problem
  - Minimum cost flow
  - Can be formulated as an LP and solved by general LP solvers
- Motivation
  - Well-known: often taught at OR, CS and Math courses
  - Well studied: many different algorithms
    - Successive shortest path algorithm
    - Scaling algorithms, polynomial
      - Cost scaling, capacity scaling, double scaling, etc.
    - Network simplex algorithms
      - Primal and dual network simplex
    - Reference: Network Flows, R. Ahuja, T. Magnanti and J. Orlin

Network Simplex Algorithm

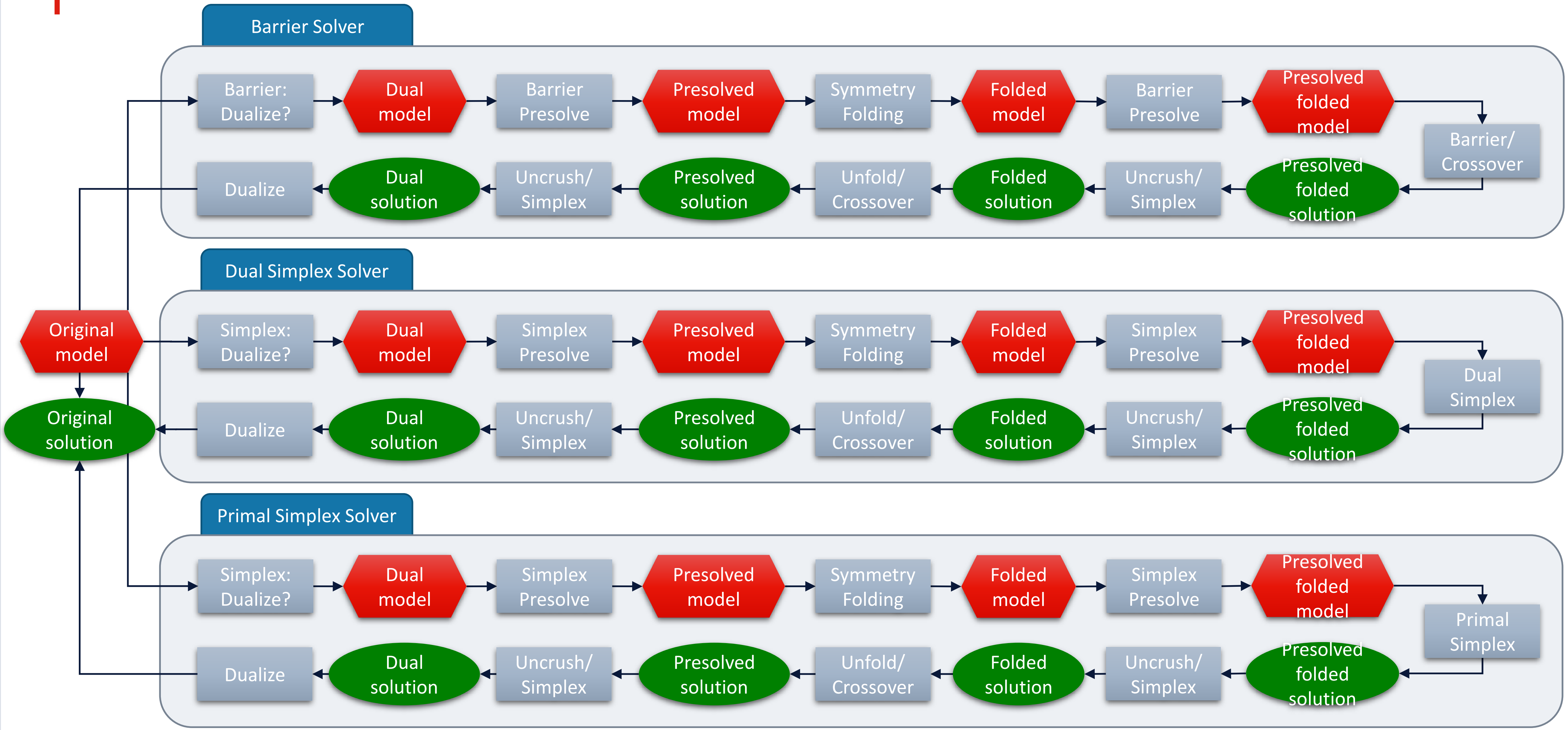
- Gurobi network simplex algorithm
  - Implemented only primal simplex
  - Most challenging parts for the implementation
    - Data structure for spanning tree, i.e., basis
    - Maintain/update spanning tree
- Advantages of primal network simplex over general LP primal simplex
  - Special structure makes computation much faster: about 5x
  - Strong feasible spanning tree: guarantee no cycling
  - Easier to construct special algorithms for initial good spanning tree (basis crash), etc.
  - Performance on our network set
    - Vs. general primal simplex: 36x, about 50% fewer iterations
    - Vs. general dual simplex: 3.9x, about 10x more iterations
- Dual network simplex
  - Not implemented: similar difficulty to implement, maybe a bit harder
  - Don't know any simple nice way to guarantee no cycling
  - Still expect to be faster than primal network algorithm

Concurrent LP Algorithms: Gurobi 9.5

GUROBI OPTIMIZATION



Concurrent LP Algorithms: Gurobi 9.5



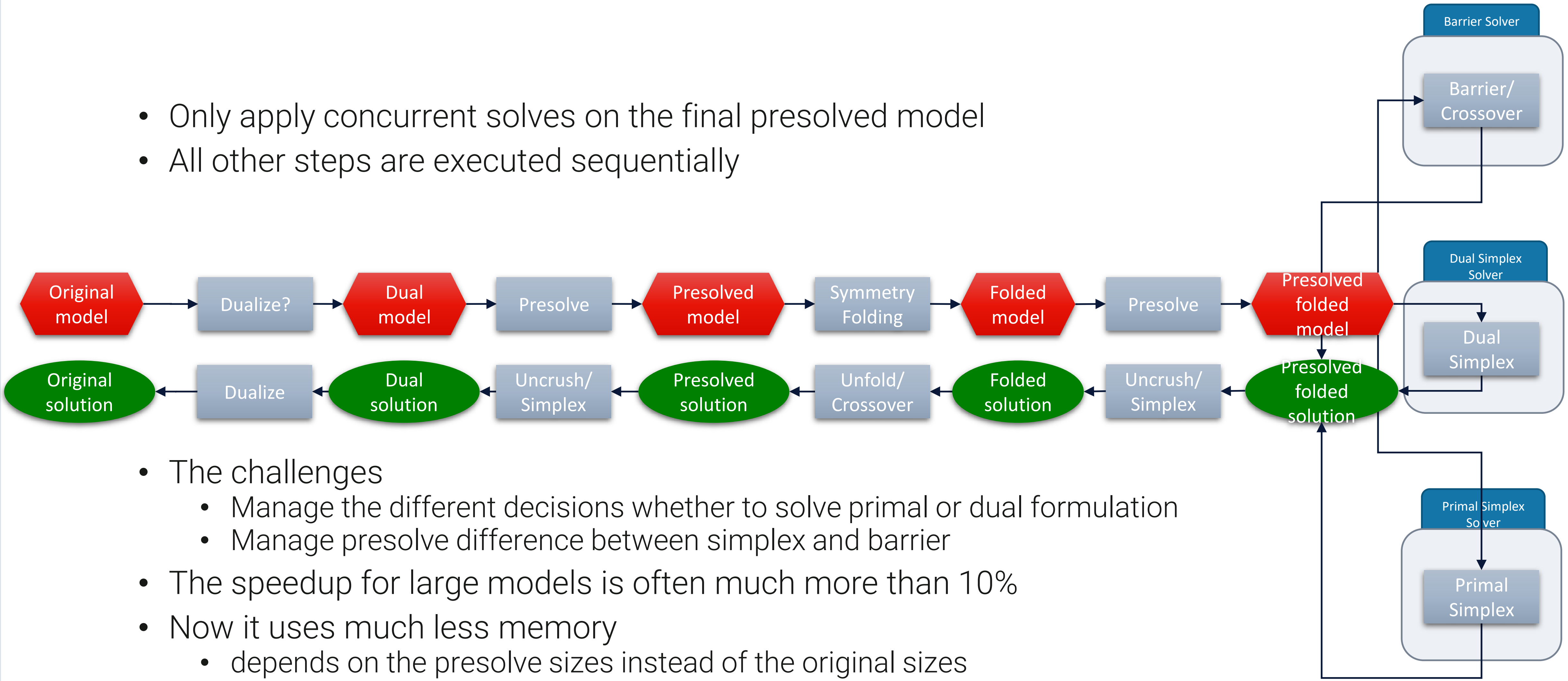
Concurrent LP Algorithms: Gurobi 9.5

- Make a model copy for each concurrent job
  - Often takes a lot of memory
- Each job, primal simplex, dual simplex or barrier, will apply all the steps independently
  - Each step is performed by a concurrent job concurrently
- Concurrently running jobs can slow down computation significantly
  - Depending on machines and the size of a model, it can be 30% - 60% slowdown

Concurrent LP Algorithms: Gurobi 10.0

GUROBI OPTIMIZATION

- Only apply concurrent solves on the final presolved model
- All other steps are executed sequentially



- The challenges
  - Manage the different decisions whether to solve primal or dual formulation
  - Manage presolve difference between simplex and barrier
- The speedup for large models is often much more than 10%
- Now it uses much less memory
  - depends on the presolve sizes instead of the original sizes

GUROBI
OPTIMIZATION

Gurobi 10.0

Performance Summary

Performance improvements compared to Gurobi 9.5

Algorithm	Overall speed-up	On >100sec models
LP – default	10%	25%
LP – primal simplex	3%	10%
LP – dual simplex	3%	10%
MILP	13%	24%
Convex MIQP	57%	2.4x*
Convex MIQCP	28%	88%*
Non-convex MIQCP	51%	2.6x

* MIQP and MIQCP hard model test sets too small to give reliable benchmark results

MIP Performance

- Various strong branching improvements
- Several symmetry improvements
- Disabling inactive cuts for relaxations while diving
- Aggressive settings for solving sub-MIPs
- New presolve reductions and improvements
- Concurrent LP improvement and tuning for relaxations
- Aggressive VUB merging using cliques
- Optimization-based bound tightening (OBBT)
    - Helps MIQP/MIQCP/MINLP more, will be discussed later
- Various improvements for machine learning models
    - Will be discussed in the part for open-source Github repositories

Strong Branching Improvements

- Strong branching
  - Select a set of fractional binary/integer variables
  - For each variable, perform certain number of dual iterations for down and up branches
    - Use the objective changes for both branches for selecting branching variable
- Improvements in Gurobi 10
  - Strong branching is very expensive, do less while keeping good quality
    - "Look ahead": abort after n successive candidates tried without new best candidate
    - Use symmetry to skip symmetric candidates
  - Combined with implications from branching down or up
    - Propagate implied bounds
    - Propagate cliques
    - Propagate SOS constraints
  - Tuned decision on how often to apply strong branching
  - Various other tweaks

GUROBI OPTIMIZATION

Gurobi 10.0

Performance Summary

Performance improvements compared to Gurobi 9.5

Algorithm	Overall speed-up	On >100sec models
LP – default	10%	25%
LP – primal simplex	3%	10%
LP – dual simplex	3%	10%
MILP	13%	24%
Convex MIQP	57%	2.4x*
Convex MIQCP	28%	88%*
Non-convex MIQCP	51%	2.6x

* MIQP and MIQCP hard model test sets too small to give reliable benchmark results

MIQP/MIQCP Performance

- New QUBO heuristic
- Perspective strengthening
- Move Q objective terms to constraints
- Work limit adjustment for QC fixing heuristics
- Strengthening coefficients of binary variables in quadratic constraints
- Fix binary in certain order for heuristics
- Solve set covering problem to select linearization
- Remove common variables
- Optimization-based bound tightening (OBBT)
- Many MIP improvements also apply

New QUBO Heuristic

- Two types of heuristics
  - Construction: create a new solution
  - Improvement: improve an existing solution
- QUBO heuristic in Gurobi 9.5
  - Tabu search – improvement heuristic
    - Start from a random start point
  - Local improvement is easy for QUBO
    - No constraints
- New QUBO heuristic in Gurobi 10.0
  - Rank-2 relaxation heuristic – construction heuristic
    - Burer, Monteiro, and Zhang, Rank-Two Relaxation Heuristics for Max-Cut and Other Binary Quadratic Programs

New QUBO Heuristic

- Good to have both
  - Neighborhood search can be defeated when good solutions are far apart
  - Particularly important when constraints are captured as penalties
- Our computational results
  - Able to find good solution for QUBO problems quickly
  - Also, able to reduce optimization time significantly, which is rare for heuristics

Gurobi 10.0

Performance Summary

Performance improvements compared to Gurobi 9.5

Algorithm	Overall speed-up	On >100sec models
LP – default	10%	25%
LP – primal simplex	3%	10%
LP – dual simplex	3%	10%
MILP	13%	24%
Convex MIQP	57%	2.4x*
Convex MIQCP	28%	88%*
Non-convex MIQCP	51%	2.6x

* MIQP and MIQCP hard model test sets too small to give reliable benchmark results

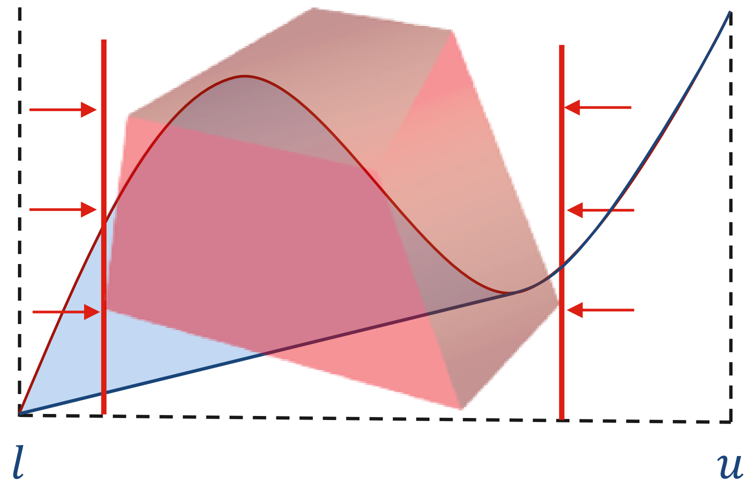
Non-convex MIQCP Performance

- Optimization-based bound tightening (OBBT)
- Dealing explicitly with bipartite graphs in the product term covering
- Improvement on NLP heuristic termination
- NLP heuristic multi-start
- Many MIP and convex MIQCP improvements also apply

Optimization Based Bound Tightening

- Common technique for MINLP solvers
- Given the LP relaxation of a (non-convex) MI(NL)P
- For each variable x
  - Minimize/maximize x value over relaxation
  - Use optimal value as lower/upper bound for x
  - Tighten coefficients of relaxation using new bounds
- Enhancements for OBBT (Gleixner et al. 2017)
  - Filter variables
  - Exploit warm starts
  - Use dual solution of OBBT LPs to tighten bounds in the tree.

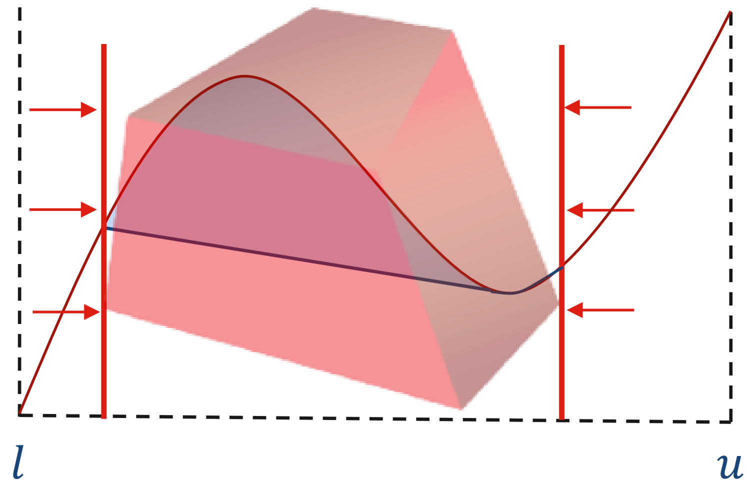
e.g.: $\mathrm{conv}(y \geq f(x): l \leq x \leq u) \cap X$



Optimization Based Bound Tightening

- Common technique for MINLP solvers
- Given the LP relaxation of a (non-convex) MI(NL)P
- For each variable x
  - Minimize/maximize x value over relaxation
  - Use optimal value as lower/upper bound for x
  - Tighten coefficients of relaxation using new bounds
- Enhancements for OBBT (Gleixner et al. 2017)
  - Filter variables
  - Exploit warm starts
  - Use dual solution of OBBT LPs to tighten bounds in the tree

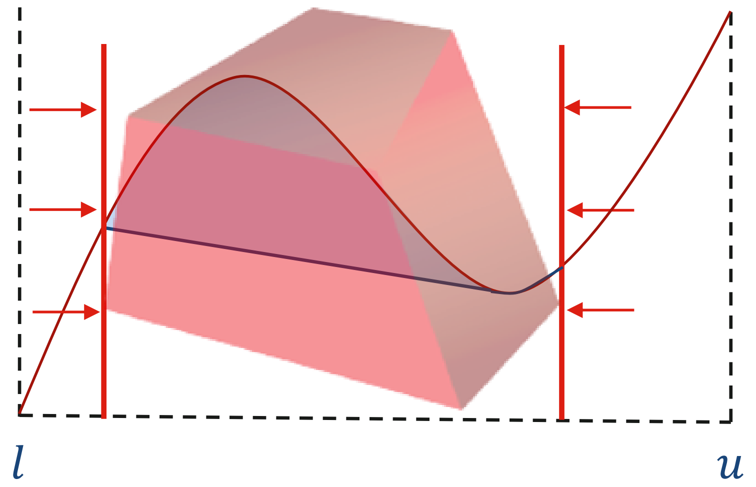
e.g.: $\mathrm{conv}(y \geq f(x) : l \leq x \leq u) \cap X$



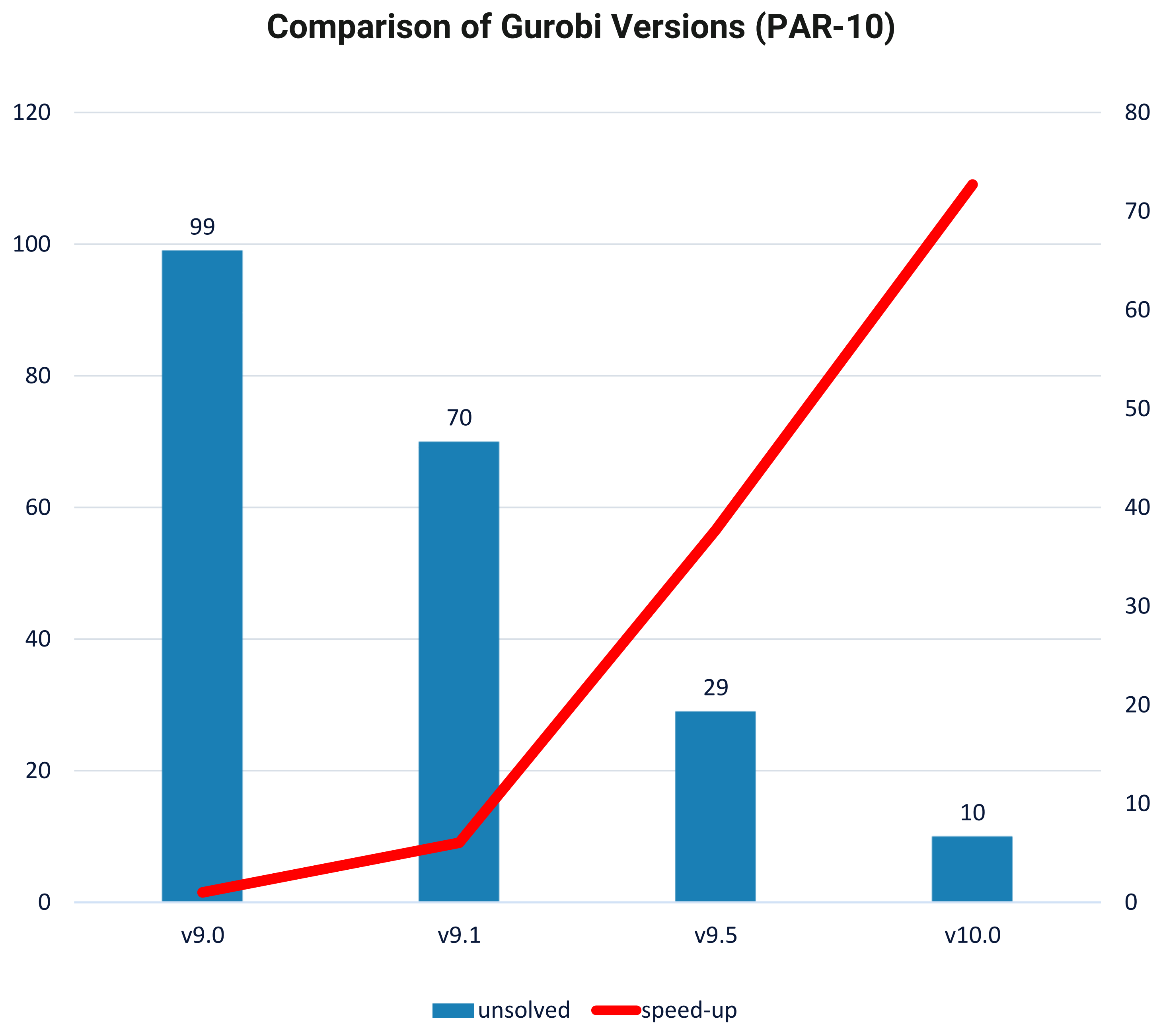
Optimization Based Bound Tightening

- For non-convex MIQCP:
  - 14% improvement overall
  - 33% improvement on models solved in ≥ 100 sec.
- For MIP, additional improvements:
  - Detect variables that influence big-M coefficients
  - Group those in clusters
  - Do OBBT, within each cluster and propagate
  - Aimed at Neural network with ReLU structures (inspired by Fischetti, Jo 2017)
  - Modest average improvement MIP/MIQP/MIQCP: 1%
  - But big improvement on certain of models (NN with ReLU)

e.g.: $\text{conv}(y \geq f(x): l \leq x \leq u) \cap X$



Non-convex MIQCP
Performance Evolution



Time limit: 10000 sec.
Intel Xeon CPU E3-1240 v5 @ 3.50GHz
4 cores, 8 hyper-threads
32 GB RAM

Test set has 874 models:
- 38 discarded due to inconsistent answers
- 308 discarded that none of the versions can solve
- speed-up measured on >100s bracket: 205 models

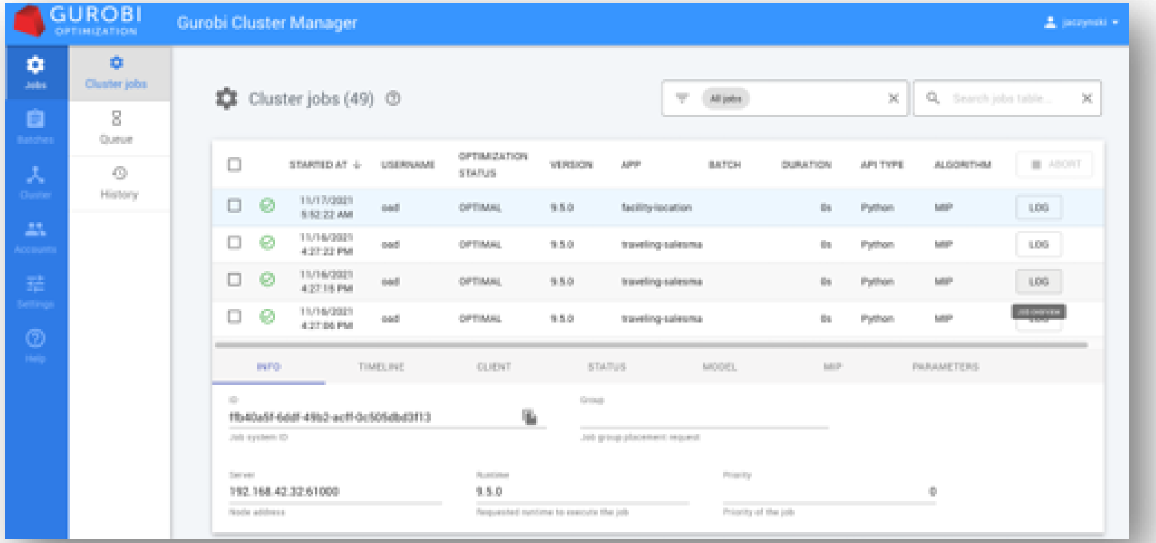
New Features

Platform Components and WLS

New Platform Features

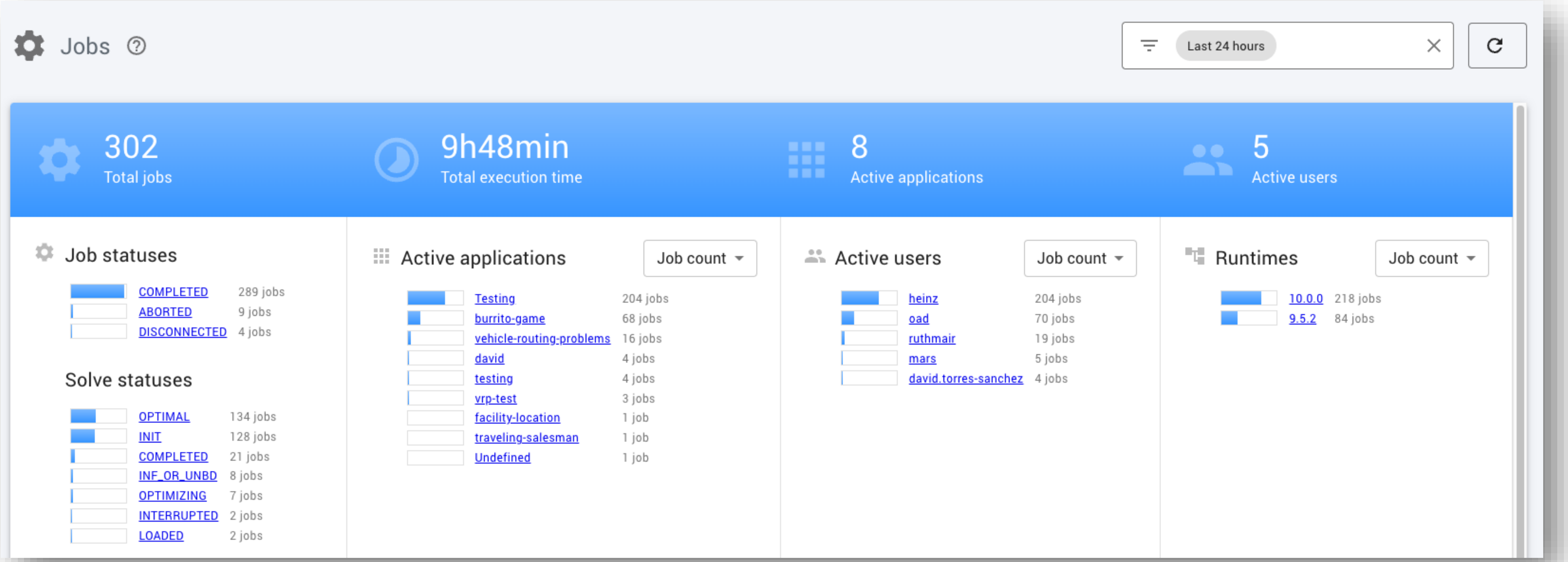
Gurobi Cluster Manager 10.0

- Cluster Manager/Compute Server
  - Client-server architecture
  - Web UI, security, optimization nodes
- New dashboards in Cluster Manager
  - The job dashboard
  - The node dashboard
- Easier to understand application behavior and node usage



Job Dashboard

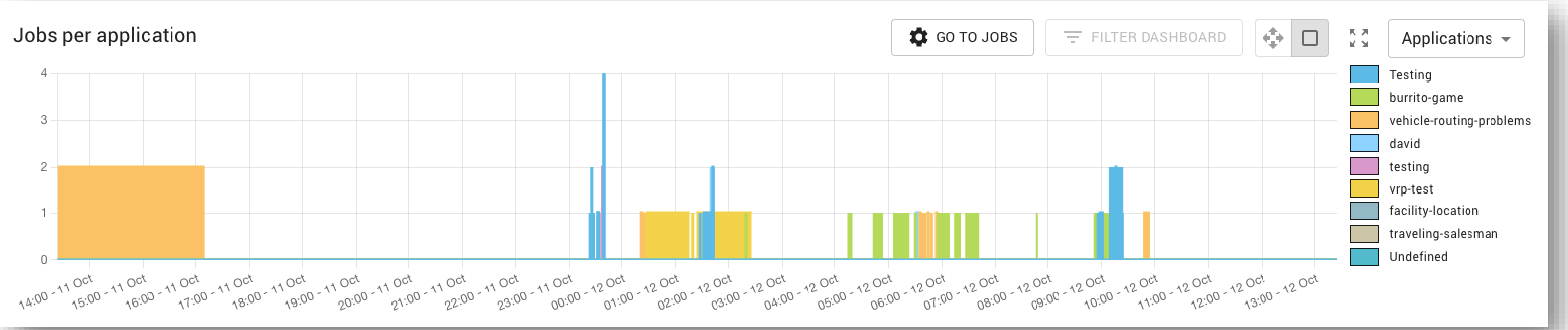
Gurobi Cluster Manager 10.0



- Predefined filters for last 24h, 7 days or 30 days
  - More filtering available
- Global metrics
  - number of jobs, execution time, active application, active users
- Distribution by several dimensions
  - Job and solve statuses, applications, users, runtimes
  - Drill down to job list

Job Dashboard

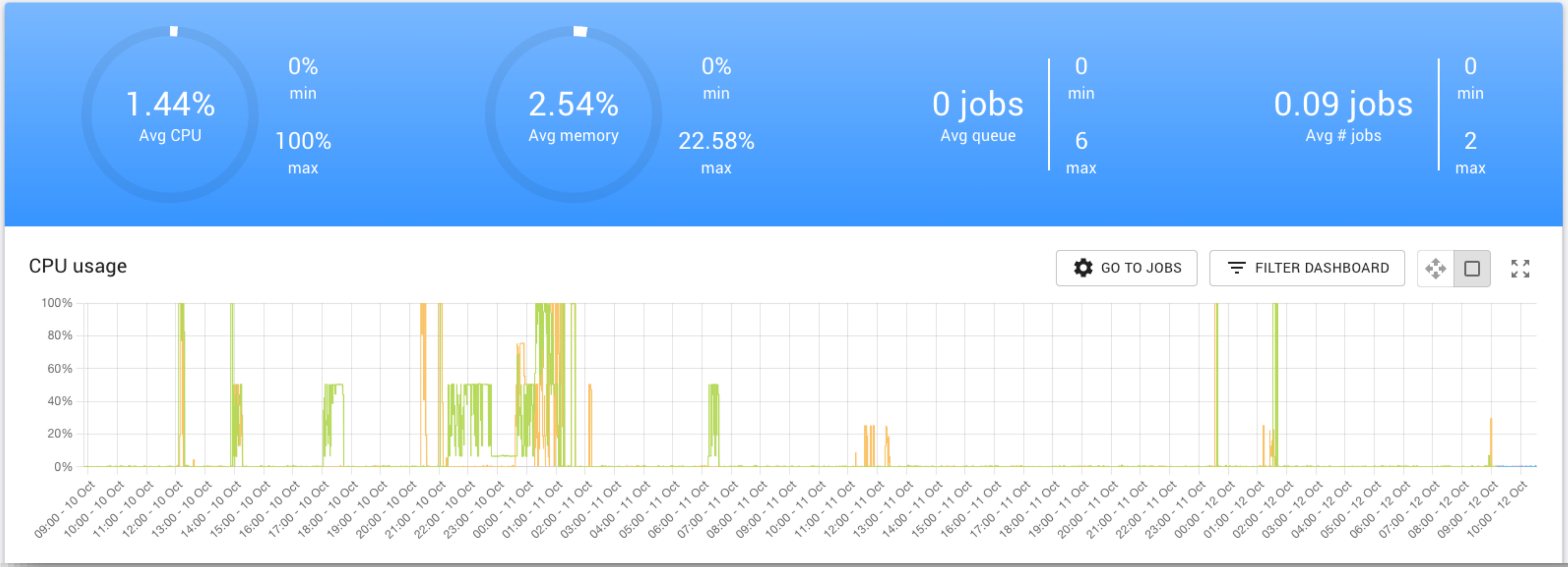
Gurobi Cluster Manager 10.0



- Timeline by several dimensions
  - Applications, Job/Solve statuses, Users, Runtime and solve times
  - Zoom and pan over time
  - Legend and colors to differentiate values
- Drilldown
  - Go to the job list of the selected period
  - Filter the dashboard with the selected period

Node Dashboard

Gurobi Cluster Manager 10.0

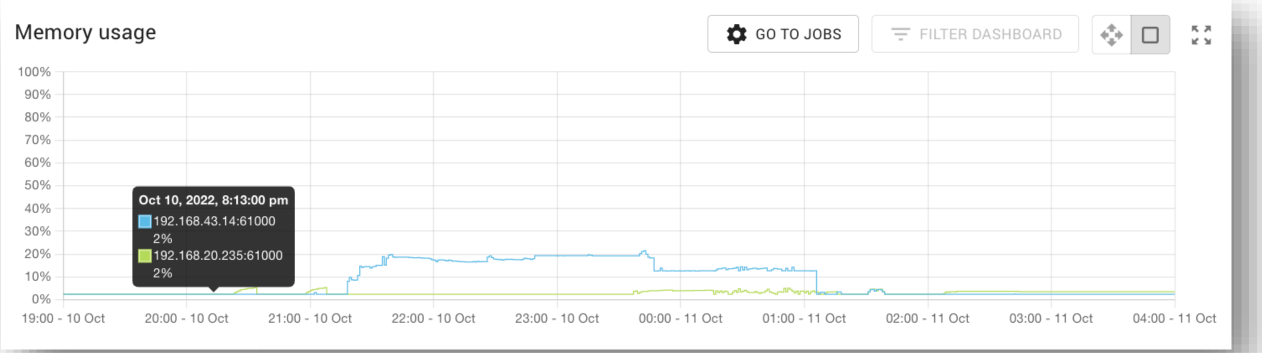


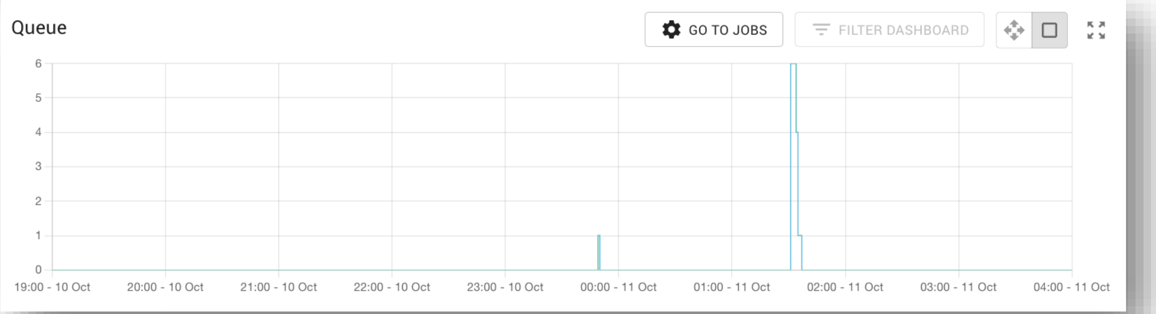
- Predefined filters for last 24h, 7 days or 30 days
- Global metrics
  - CPU, Memory,
  - Job in queue and running

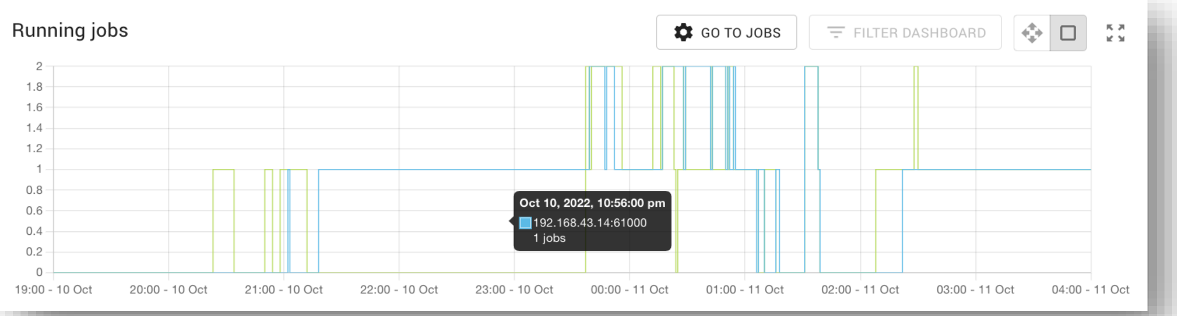
Node Dashboard

Gurobi Cluster Manager 10.0

- Timeline
  - CPU usage
  - Memory usage
  - Job in queue
  - Running jobs
- Drilldown
  - Zoom, pan
  - Node selection



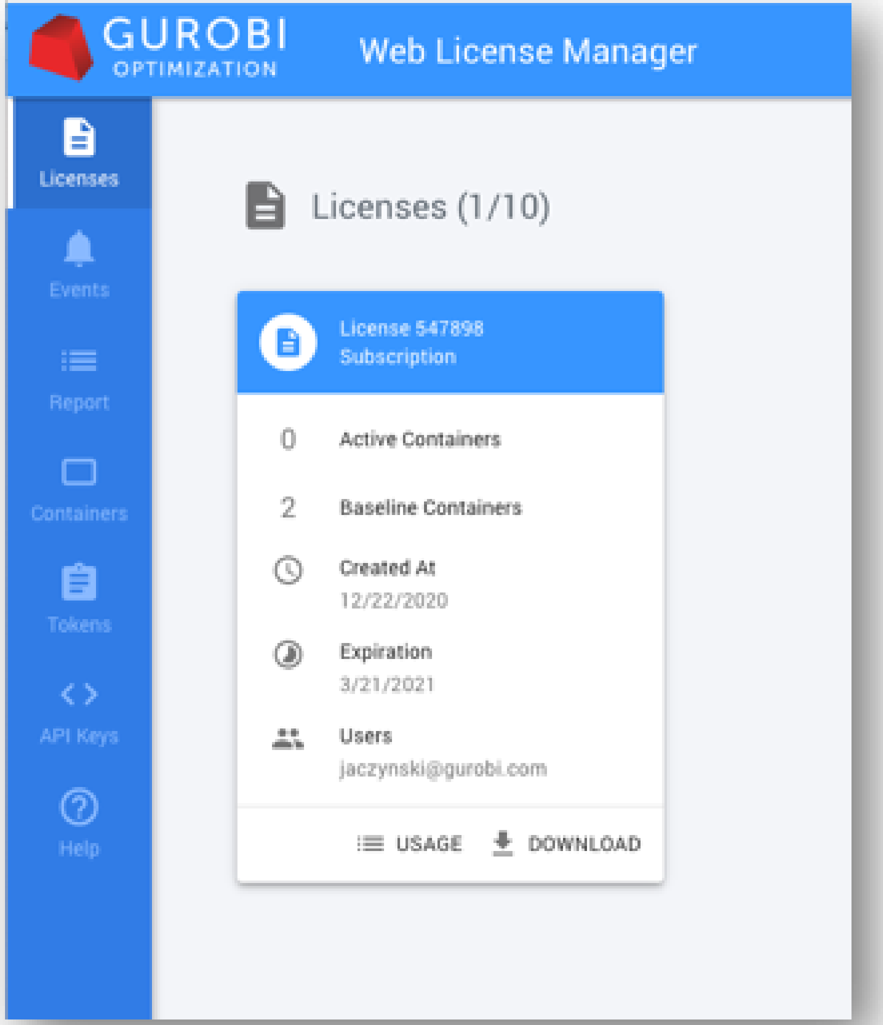




New WLS Deployment Types

Gurobi 10.0 – Web License Service

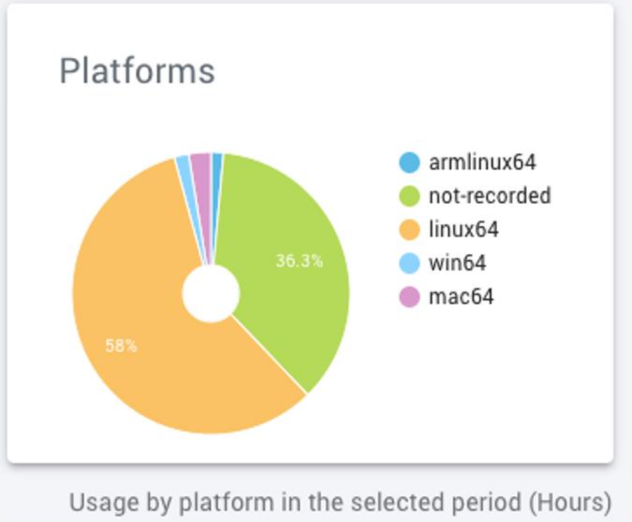
- Licensing service introduced with Gurobi 9.5
  - Servers are running in several regions worldwide
  - Dynamically activates the use of Gurobi
- Gurobi 10 supports different deployment types:
  - Containers only (Docker, Kubernetes) as in 9.5
  - Machines only (Linux, Windows, Mac)
  - Containers and Machines
- WLS licenses can now be used for any deployment scenario

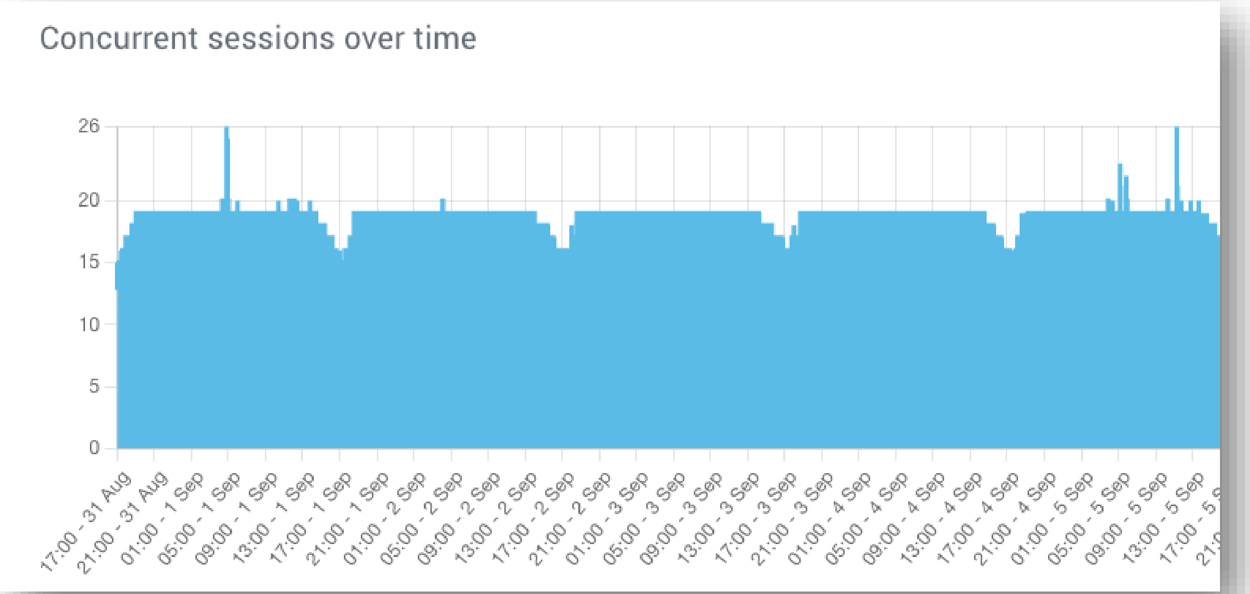


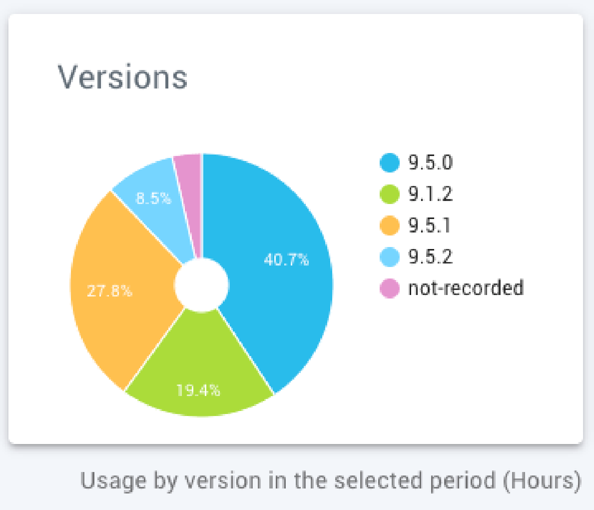
New WLS Reports and Features

Gurobi 10.0 – Web License Service

- The WLS manager reports new metrics:
  - Platforms
  - Versions
  - Sessions over time
- Explicit user control on token refresh intervals







GUROBI OPTIMIZATION

New Features

API and Engine

Gurobi 10.0 – Engine

Product Features

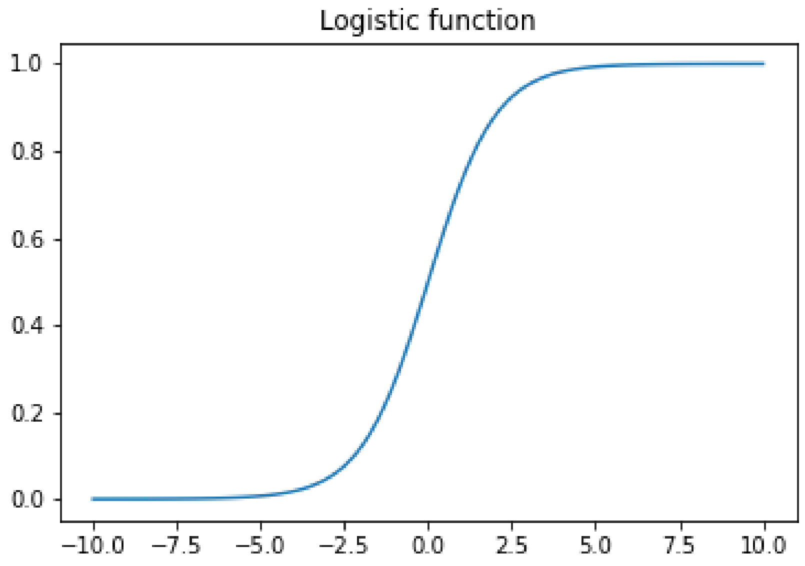
GUROBI OPTIMIZATION

- New logistic general constraint
  - Makes it easy to incorporate a constraint in MIP that models the logistic function
  - Logistic function has various applications, including ecology, statistics, machine learning, medicine, chemistry, and others
- Greatly improved the matrix-friendly API of gurobipy
  - All modeling objects now support multiple dimensions
  - Dimension handling leans consistently on NumPy, including broadcasting
- NuGet package for .NET
  - Allows .NET users to download Gurobi directly from NuGet server
- Memory limit parameter that allows graceful exit
  - User can set a memory limit and still get best solution and resume optimization after limit was hit

Logistic General Constraint

- Function constraints in Gurobi
  - Allow to state $y = f(x)$
    - f is a predefined function
    - y and x are one-dimensional variables
  - Gurobi automatically performs a piecewise-linear approximation of f in the domain of x.
- Added logistic function to our set of predefined f.

$$p(x) = \frac{1}{1 + e^{-(x)}}$$



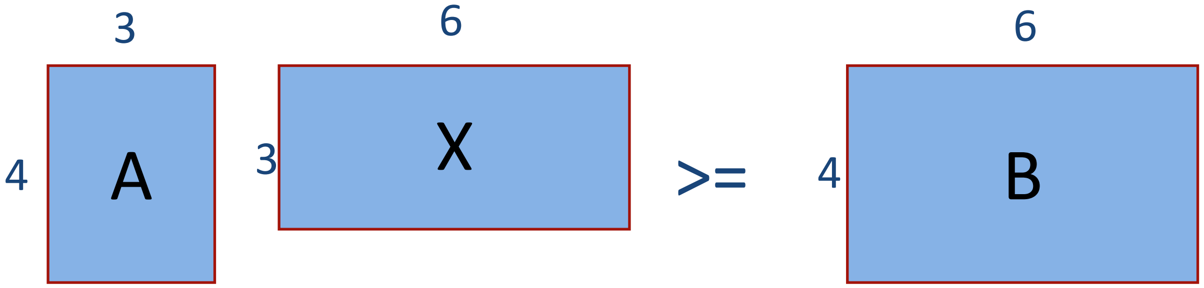
Gurobipy

Multi-dimensional modeling

- Up to version 9.5 support for multi-dimensional modeling was limited
- With version 10.0:
  - All of MVar, MLinExpr and MQuadExpr support arbitrary dimensions
  - Adding constraints from such expressions yield multidimensional MConstr/MQConstr

2-D linear constraint

```
A = np.random.rand(4, 3)
B = np.random.rand(4, 6)
X = model.addMVar((3, 6))
model.setObjective(X.sum())
# Add 4*6=24 linear constraints
mc = model.addConstr(A @ X >= B)
```



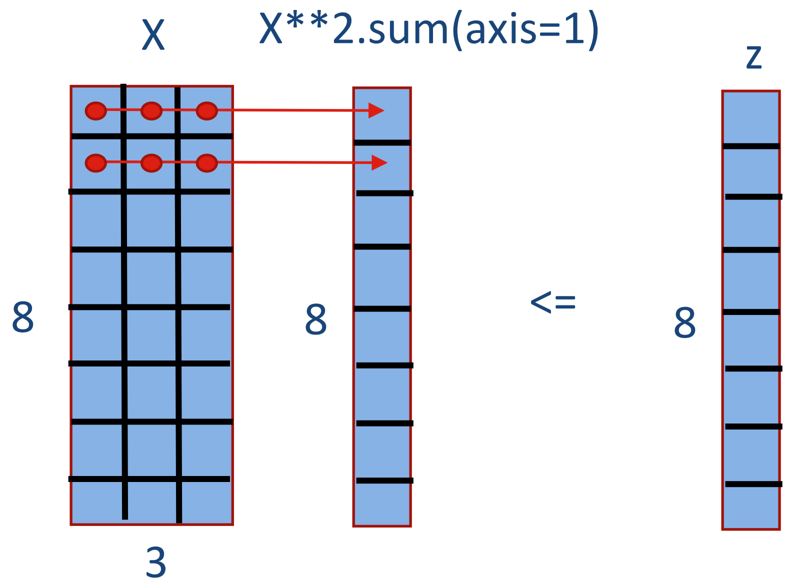
Gurobipy

Multi-dimensional modeling

- Up to version 9.5 support for multi-dimensional modeling was limited
- With version 10.0:
  - All of MVar, MLinExpr and MQuadExpr support arbitrary dimensions
  - Adding constraints from such expressions yield multidimensional MConstr/MQConstr

1-D quadratic constraint

```
X = model.addMVar((8, 3), lb=-np.inf)
z = model.addMVar(8)
# Add eight standard cones
model.addConstr((X**2).sum(axis=1) <= z**2)
```



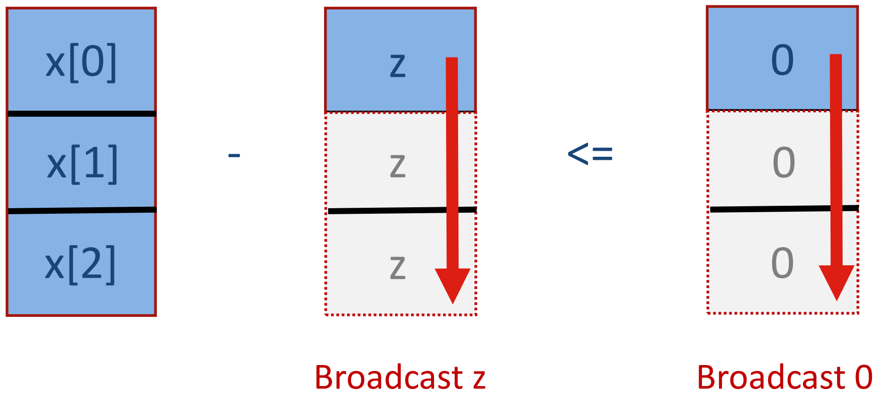
Gurobipy

Broadcasting

- Up to version 9.5: Dimensions had to agree for most operations
- With version 10.0: Embrace NumPy's broadcasting
  - All of MVar, MLinExpr and MQuadExpr can be broadcast
  - Operations with scalars, ndarrays and scipy.sparse matrices support broadcasting

Vectorized VUB constraints

```
x = model.addMVar(3, ub=1.0)
z= model.addMVar((), vtype='B')
# three VUB constraints x[i] - z <= 0
model.addConstr(x - z <= 0)
```



Gurobipy

Other new matrix-friendly features/methods

- General
  - Fewer surprises for experienced NumPy users wrt shapes of operation results
  - Support both matrix and element-wise multiplication
- MVar
  - Extract a diagonal from an MVar X : X.diagonal(offset).
  - Convert a list of Var objects to an MVar: x = MVar.fromlist(varlist)
  - Sum along an axis of an MVar X: X.sum(axis=…)
  - Elementwise squaring of an Mvar X: pow(X, 2), X**2
- MLinExpr
  - All-zero expression: MLinExpr.zeros(shape)
  - Sum along an axis of an MLinExpr mle: mle.sum(axis=…)
- New class MQuadExpr
  - For modeling multidimensional quadratic constraints
  - Similar features/methods as MLinExpr
- New class MQConstr
  - Multi-dimensional constraint handle returned from model.addConstr(…) for quadratic expressions
  - Similar features/methods as MConstr

Open-Source GitHub Repositories

Gurobi 10.0 – Open-Source GitHub Repositories

GUROBI OPTIMIZATION

- gurobipy-pandas
  - Enables convenient gurobipy model building patterns with pandas
- Gurobi Machine Learning
  - Allows users to add a trained machine learning model as constraint to a MIP
- Later this year or next year
  - Gurobi OptiMods
    - Collection of simple to use optimization modules for specific applications
    - Targets users who do not understand math modeling and just want to get solution to their problem
  - Numerical issues assessment tool*
    - Allows users to analyze models with numerical issues to find out root cause of such issues
- Gurobi GitHub projects: https://github.com/Gurobi/
  - Distributed as open-source under Apache License 2.0

*final name to be determined

Gurobipy and pandas

Easier model building with the popular Python data analytics package

- Create pandas Series and DataFrames of Gurobi variables
- Use pandas operations to combine variables and data into constraints
- Extract solution data as pandas Series
- No need to manually translate between pandas and gurobipy!

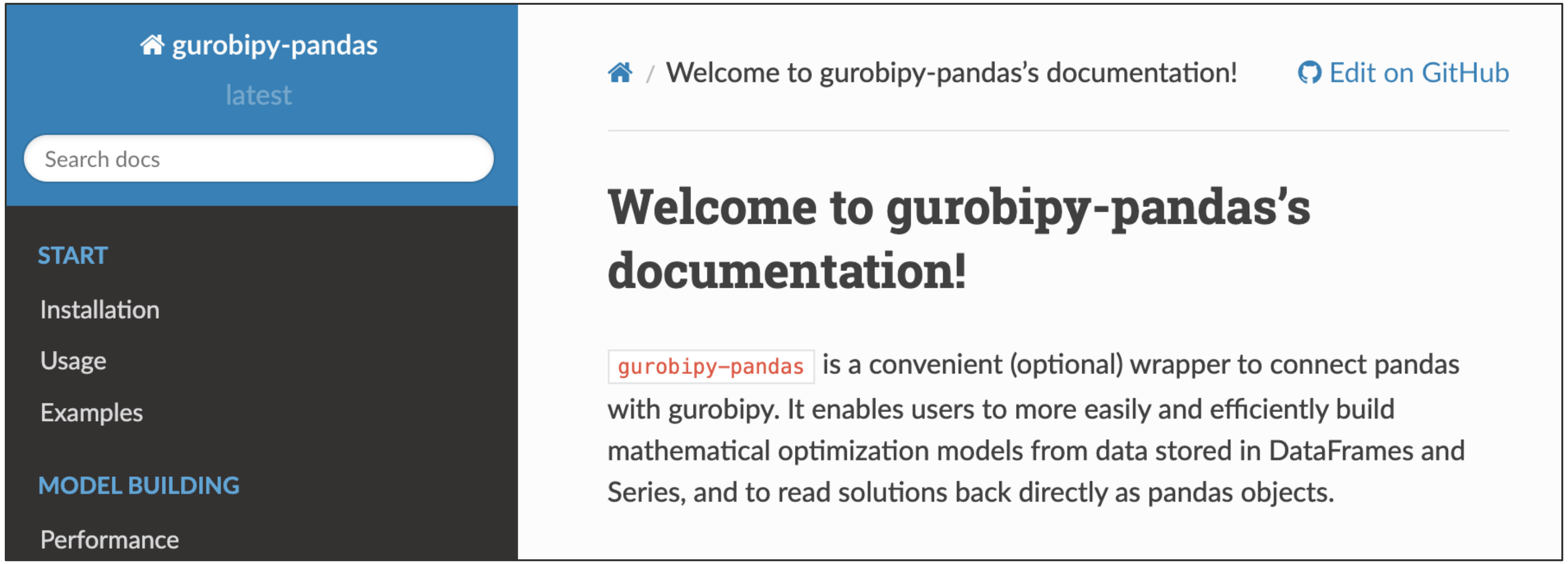
```
In [14]: x = gppd.add_vars(model, allowed_pairs, vtype=GRB.BINARY, obj="value", name="x")
         x.head()

Out[14]: project  team
         0        4       <gurobi.Var x[0,4]>
         1        4       <gurobi.Var x[1,4]>
         2        0       <gurobi.Var x[2,0]>
                  1       <gurobi.Var x[2,1]>
                  2       <gurobi.Var x[2,2]>
         Name: x, dtype: object
```


Gurobipy and pandas

Documentation and examples for users of the PyData stack

- Open source, with documentation available on readthedocs.com
  - Github repository: https://github.com/gurobi/gurobipy-pandas
  - Documentation: https://gurobi-optimization-gurobipy-pandas.readthedocs-hosted.com
- Complete model building examples as Jupyter notebooks
- Guidance for writing performant gurobipy-pandas code



Gurobi Machine Learning

Our Goals

- Simplify the process of importing a trained machine learning model built with a popular ML package into an optimization model.
- Improve algorithmic performance to enable the optimization model to explore a sizable space of solutions that satisfy the variable relationships captured in the ML model.
- Make it easier for optimization models to mix explicit and implicit constraints.

Other similar packages:

- Janos (Bergman et. al, 2019)
- ReLU_MIP (Lueg et. al, 2021)
- OptiCL (Maragno et.al, 2021)
- OMLT (Ceccon et. al, 2022)

Gurobi Machine Learning

Regression Models Understood

scikit learn

- Linear/Logistic regression
- Decision trees
- Neural network with ReLU activation
- Random Forests
- Gradient Boosting trees
- Transformations:
  - Simple scaling of features
  - Polynomial features of degree 2
  - One Hot encoder
- Pipelines to combine them

Keras

- Dense layers
- ReLU layers
- Object Oriented, functional or sequential

PyTorch

- Dense layers
- ReLU layers
- Only torch.nn.Sequential models

Gurobi Machine Learning

Example Usage

GUROBI OPTIMIZATION

- Say have trained the following regression with scikit-learn:

```
pipeline = make_pipeline(StandardScaler(), MLPRegressor([10]*2))
pipeline.fit(X_train, y_train)
```

- Embedding into a Gurobi model

```
m = gp.Model()
# Add matrix variables for the regression
input = m.addMVar((n_constr, X_train.shape[1]), lb=-GRB.INFINITY)
output = m.addMVar(n_constr, lb=-gp.GRB.INFINITY)
# Add predictor constraint
pred_constr = add_predictor_constr(m, pipeline, input, output)
```


Gurobi Machine Learning

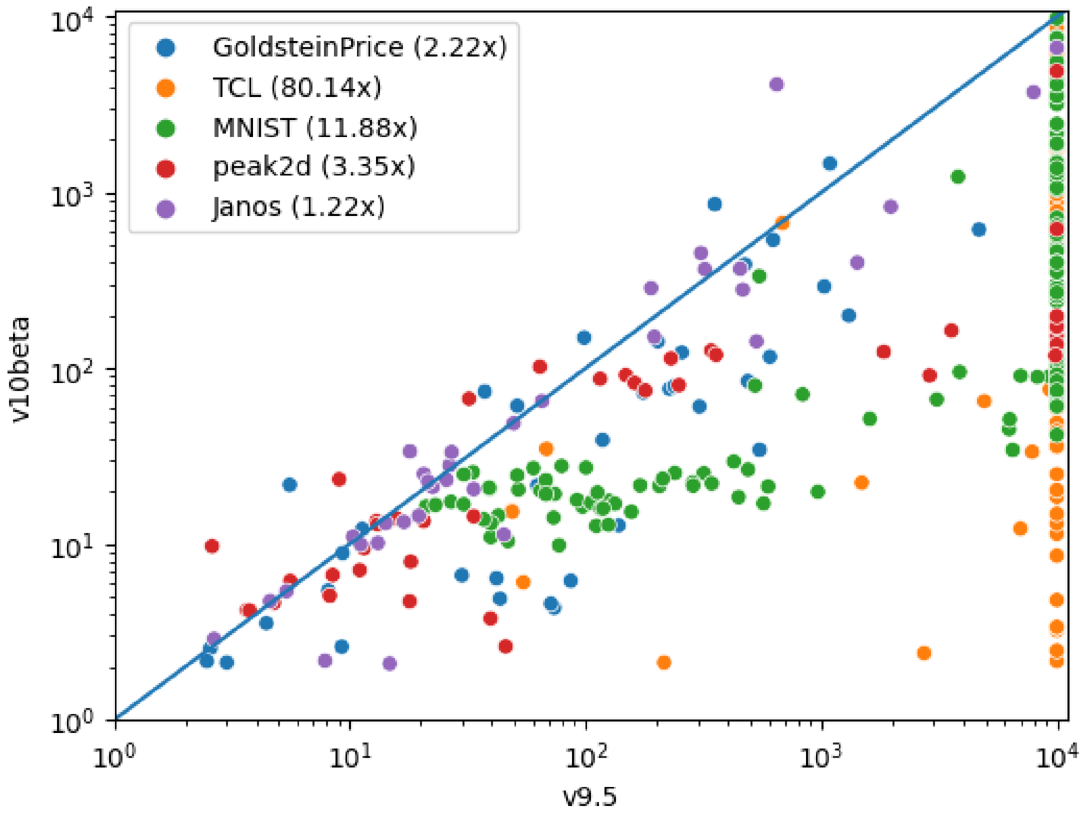
Benchmarks

GUROBI OPTIMIZATION

- Test Set
  - Function approximation:
    - Goldstein-Price function (60 instances)
    - Peak function (60 instances)
  - Janos (Bergman et.al. 2019): 500 predictor constraints with 3 features
  - TCL (Amasyali et.al. 2022): Application in electrical engineering find valid input/output within bounds minimizing costs
  - Adversarial machine learning on MNIST: 119 instance trained by tensorflow and 90 trained by scikit-learn
- Setup
  - Models solved on Intel(R) Xeon(R) CPU E3-1240 CPUs, 4 cores, 4 threads
  - Time limit 10,000 seconds
  - Models with logistic regression excluded
  - Models not solved by any in the time limit excluded

Gurobi Machine Learning

Gurobi 9.5 vs Gurobi 10.0



Gurobi Machine Learning

GUROBI OPTIMIZATION

- Github repository: https://github.com/Gurobi/gurobi-machinelearning
- Documentation: https://gurobi-machinelearning.readthedocs.io

build and test passing | build wheel passing | python 3.9 | 3.10 | code style black

pypi v1.0.0b4 | docs passing | Help Gurobi-Forum

Gurobi Machine Learning

Gurobi Machine Learning is an open-source python package to embed trained regression models in a `gurobipy` model to be solved with the Gurobi solver.

Q&A

© 2022 Gurobi Optimization, LLC. Confidential, All Rights Reserved | 52

10.0 Webinar Series

Deep Dives into Features and Enhancements

Nov

10.0 Overview
Nov 15 / Nov 22, 2022

Dec

Matrix-Friendly API
Dec 7 / Dec 8, 2022

Jan

Gurobi Machine Learning
Jan 25 / Feb 2, 2023

Gurobipy Pandas Accessors
Jan 31 / Feb 7, 2023

Feb

Mar

Numerical Issues Assessment Tool
Mar 2023

GUROBI
OPTIMIZATION

Thank You