What's New in Gurobi 11.0

December 2023

GUROBI OPTIMIZATION

Agenda

- LP Improvements
- MIP Improvements
- Global MINLP
- Additional features
- APIs: Java & Gurobipy Enhancements
- Dynamically Distributed Tuning
- Cluster Manager and Compute Server Enhancements

GUROBI OPTIMIZATION

Linear Programming

© 2023 Gurobi Optimization, LLC. Confidential, All Rights Reserved | 3

Performance Summary

Gurobi 10.0 vs. Gurobi 11.0 – Continuous Convex Models

GUROBI OPTIMIZATION

problem class	speed-up (>1s)	speed-up (>100s)
LP (concurrent)	0.7%	1.2%
LP (barrier)	2.0%	4.1%
LP (primal simplex)	3.1%	9.3%
LP (dual simplex)	0.3%	1.9%
QP	1.3%	—*
SOCP	40.6%**	—*

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

* Too few hard QP and SOCP models to measure performance
** Includes performance bug fix for dense cone handling

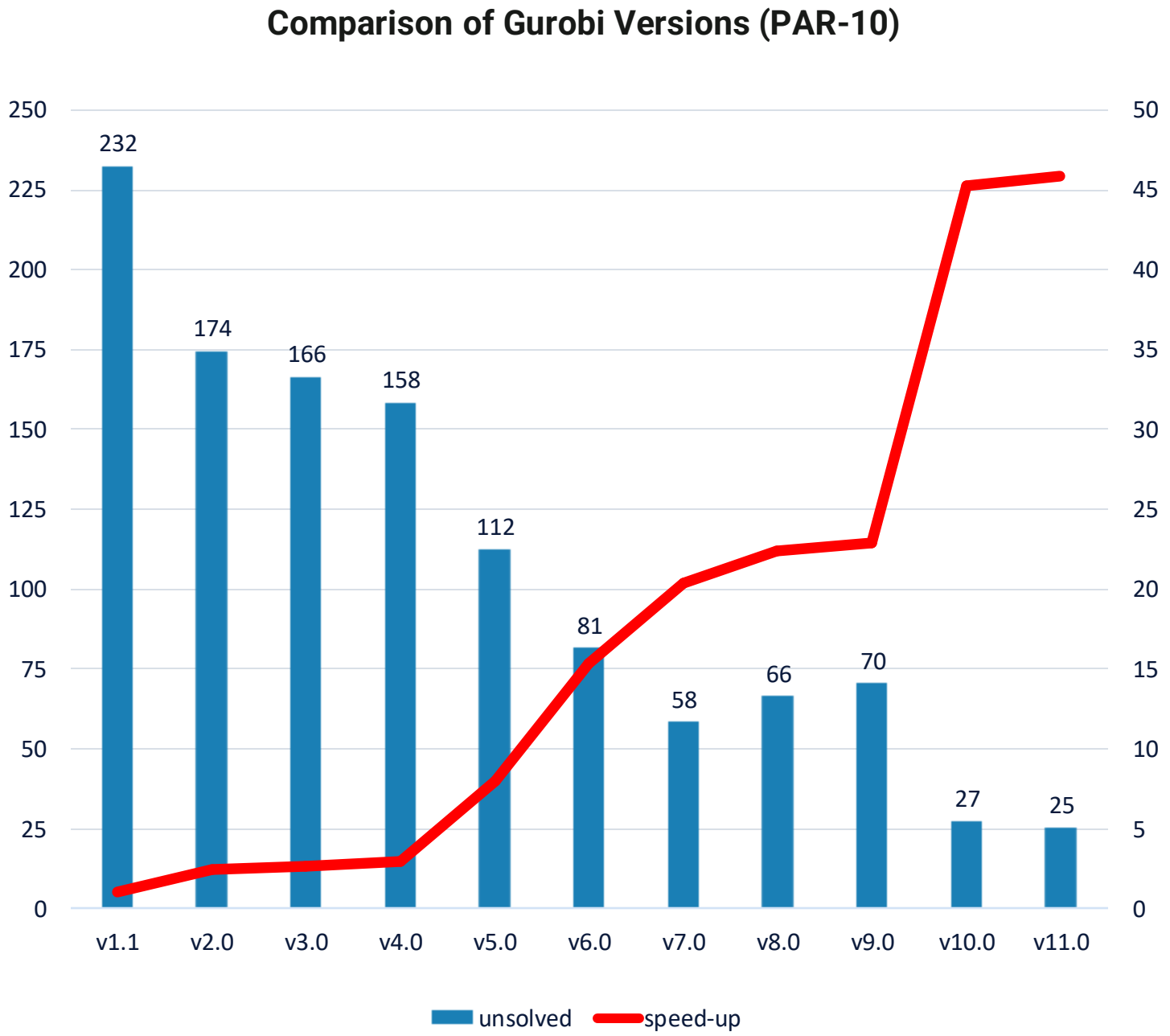
GUROBI OPTIMIZATION

LP

Performance Evolution

Default settings:

- Gurobi 1 – 4: dual simplex
- Gurobi 5+: concurrent LP



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

Test set has 2574 models:
- 225 discarded due to inconsistent answers
- 77 discarded that none of the versions can solve
- speed-up measured on >100s bracket: 596 models

SolutionTarget Parameter

GUROBI OPTIMIZATION

- Introduced in Gurobi 10.0.0 as undocumented parameter
- Fully documented and implemented in all APIs in Gurobi 11.0

- `SolutionTarget`:
  - `-1`: automatic (equal to SolutionTarget=0)
  - `0`: obtain optimal basic solution
  - `1`: obtain optimal (not necessarily basic) solution

- Better performance, as certain steps can be skipped:
  - Crossover after barrier solve
  - Unfolding after symmetry folding reduction

SolutionTarget Parameter

Performance

GUROBI OPTIMIZATION

- Gurobi 10.0:
  - `SolutionTarget=1` basically equivalent to `Method=2 Crossover=0`
- Gurobi 11.0:
  - Supports concurrent LP with barrier without crossover
  - Gradient Boosted Tree decides whether to use pure barrier or concurrent LP
- Performance improvement vs. Gurobi 10.0:
  - 23% faster overall (46% on >100sec models)*

* concurrent LP with SolutionTarget=1, 4 core machine

Concurrent LP Algorithm Parameters

Gurobi 10.0

- `Method`
  - `-1`: automatic
  - `0`: primal simplex
  - `1`: dual simplex
  - `2`: barrier
  - `3`: non-deterministic concurrent LP
  - `4`: deterministic concurrent LP
  - `5`: deterministic concurrent simplex

Gurobi 11.0

- `Method`
  - `-1`: automatic
  - `0`: primal simplex
  - `1`: dual simplex
  - `2`: barrier
  - `3`: non-deterministic concurrent LP
  - `4`: deterministic concurrent LP
  - `5`: deterministic concurrent simplex (deprecated)
- ConcurrentMethod
  - `-1`: automatic
  - `0`: barrier/dual/primal
  - `1`: barrier/dual
  - `2`: barrier/primal
  - `3`: dual/primal

GUROBI OPTIMIZATION

Mixed Integer Programming

© 2023 Gurobi Optimization, LLC. Confidential, All Rights Reserved | 9

Performance Summary

Gurobi 10.0 vs. Gurobi 11.0 – Mixed Integer Convex Models

GUROBI OPTIMIZATION

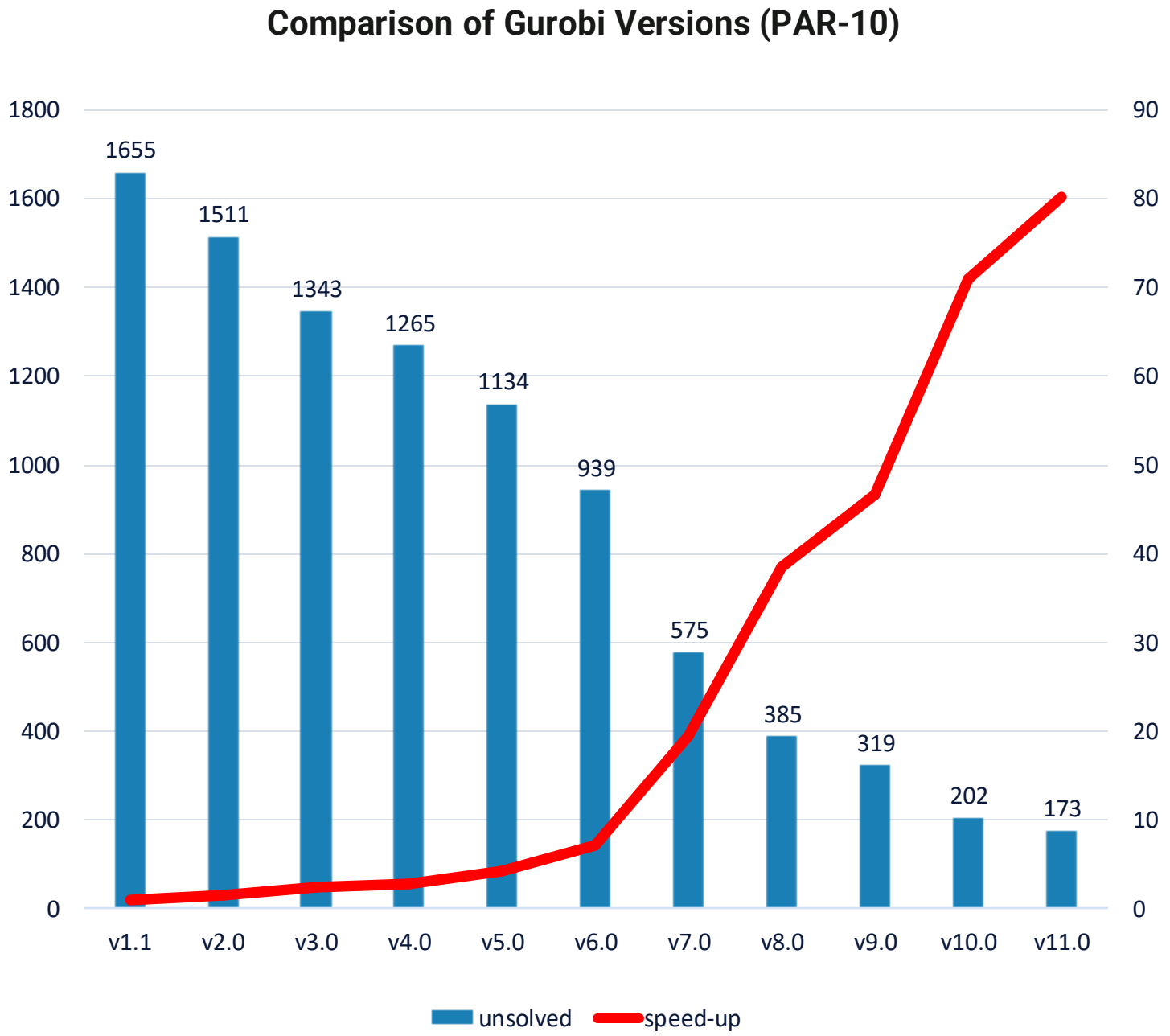
problem class	speed-up (>1s)	speed-up (>100s)
MILP	8.6%	12.4%
MIQP	12.8%	22.8%
MIQCP	9.2%	18.2%

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

GUROBI OPTIMIZATION

MILP

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 7766 models:
- 714 discarded due to inconsistent answers
- 2124 discarded that none of the versions can solve
- speed-up measured on >100s bracket: 2892 models

MIP Performance Improvements

Many small improvements that add up (1/2)

- Presolve:
  - Improved propagation in probing (0.6%)
  - Improved lifting sequence in probing (0.7%)
  - Improved work tracking in probing (0.3%)
  - Discard duplicates from clique table, collect substitutions (0.7%)
  - Better handling of significant model size reductions (1.3%)
- Node presolve:
  - Work limit adjustment in domain propagation (0.3%)
  - Better numerics in domain propagation (0.3%)
  - Earlier updates of global bounds from reduced cost fixing (2.1%)
- Branching:
  - GMI cut scores for branching variable selection (0.3%)
  - Propagated pseudo cost scores for branching variable selection (1.0%)

MIP Performance Improvements

Many small improvements that add up (2/2)

- Cuts:
  - Separate higher rank knapsack cover cuts (1.0%)
  - Improved clique cut separation (0.6%)
  - Aggressive constraint disaggregation cuts (0.8%)
  - Improvements in parallel root cut loops (0.4%)
  - More aggressive sub-MIP cuts (1.5%)
  - Mixing Path Cuts (1.2%)
- Symmetry:
  - Detect and exploit symmetry earlier at root node (0.9%)
  - Weak symmetry cuts (0.9%)
- Other:
  - Improved conflict constraint selection (0.8%)
  - Improved sorting methods (0.4%)

Mixing Path Cuts

GUROBI OPTIMIZATION

- References:
  - O. Gunluk, Y. Pochet: Mixing mixed-integer inequalities.
    Math. Program. 90, 429–457 (2001). https://doi.org/10.1007/PL00011430
  - P. Christophel: Separation algorithms for cutting planes based on mixed integer row relaxations: implementation and evaluation in the context of mixed integer programming solver software. (PhD thesis)
    University of Paderborn, 2009, pp. 1-222
- Integrated in our MIR aggregation procedure, using “U-cut procedure” from Christophel
- Parameter `MixingCuts` with values -1, 0, 1, 2
- Performance impact: 0.5% overall, 1.2% on >100sec models

GUROBI OPTIMIZATION

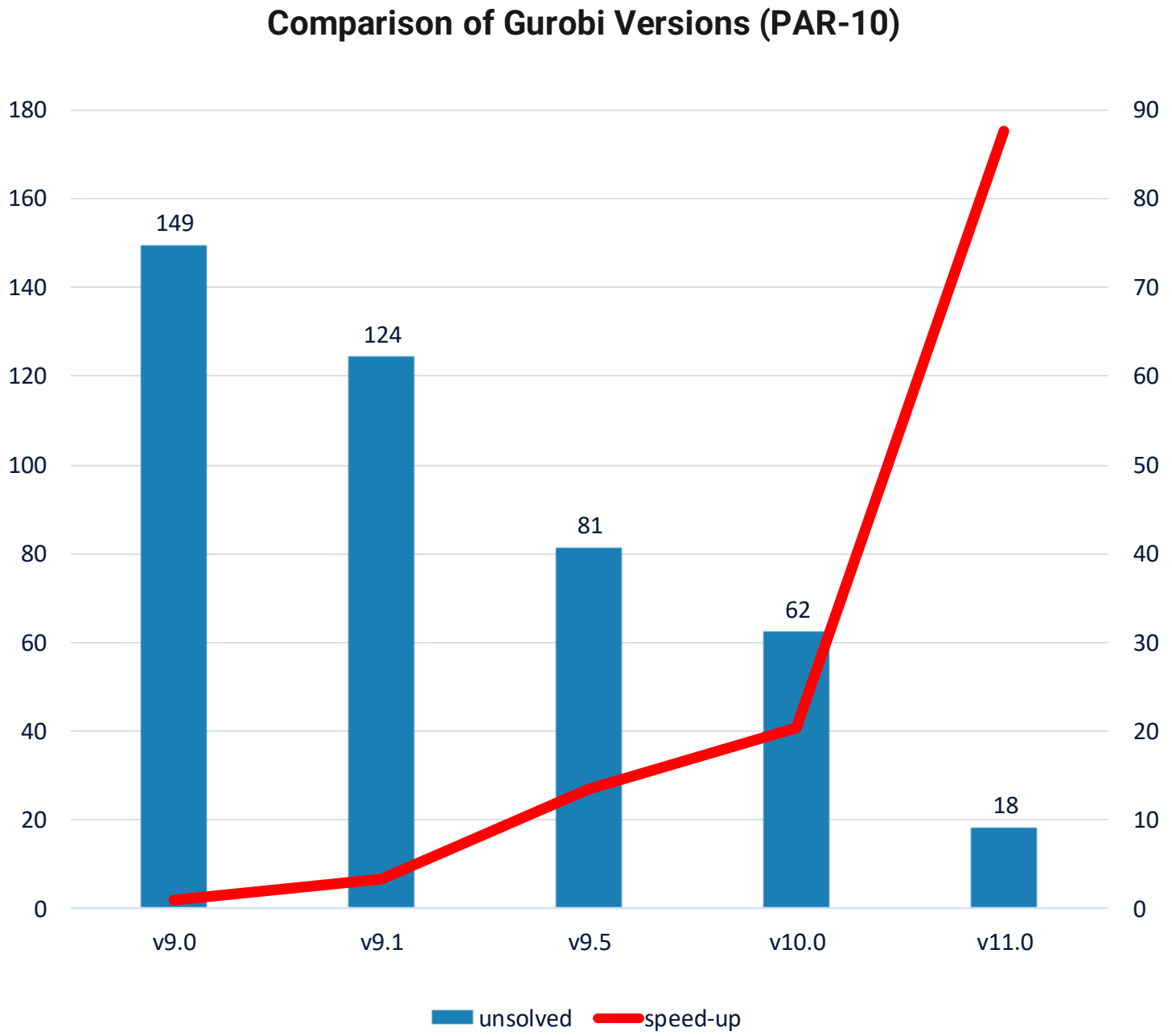
Nonlinear Programming

© 2023 Gurobi Optimization, LLC. Confidential, All Rights Reserved | 15

GUROBI OPTIMIZATION

Nonconvex 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 1064 models:
- 50 discarded due to inconsistent answers
- 344 discarded that none of the versions can solve
- speed-up measured on >100s bracket: 275 models

Performance Summary

Gurobi 10.0 vs. Gurobi 11.0 –Nonconvex Models

problem class	speed-up (>1s)	speed-up (>100s)
nonconvex MIQCP	2.3x	5.8x

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

Quadratic Objective and Constraints

NonConvex Parameter

GUROBI OPTIMIZATION

- `NonConvex`:
  - `-1`: automatic
  - `0`: return error if original model has nonconvex Q objective or constraints
  - `1`: return error if presolved model has nonconvex Q that cannot be linearized
  - `2`: accept nonconvex Q by using a bilinear transformation
- Gurobi 10.0 default (-1): equivalent to 1
- Gurobi 11.0 default (-1): essentially equivalent to 2

- Change can break user code or be at least surprising to users!
- Often, nonconvexity is still a sign of an error in model or data
  - Users now may want to explicitly set `NonConvex=1`

Nonlinear Constraints

- Gurobi 9.0 and later provide API to define nonlinear functions
  - e^x, a^x
  - $\ln(x), \log_a(x)$
  - $\sin(x), \cos(x), \tan(x)$
  - x^a
  - $ax^3 + bx^2 + cx + d$

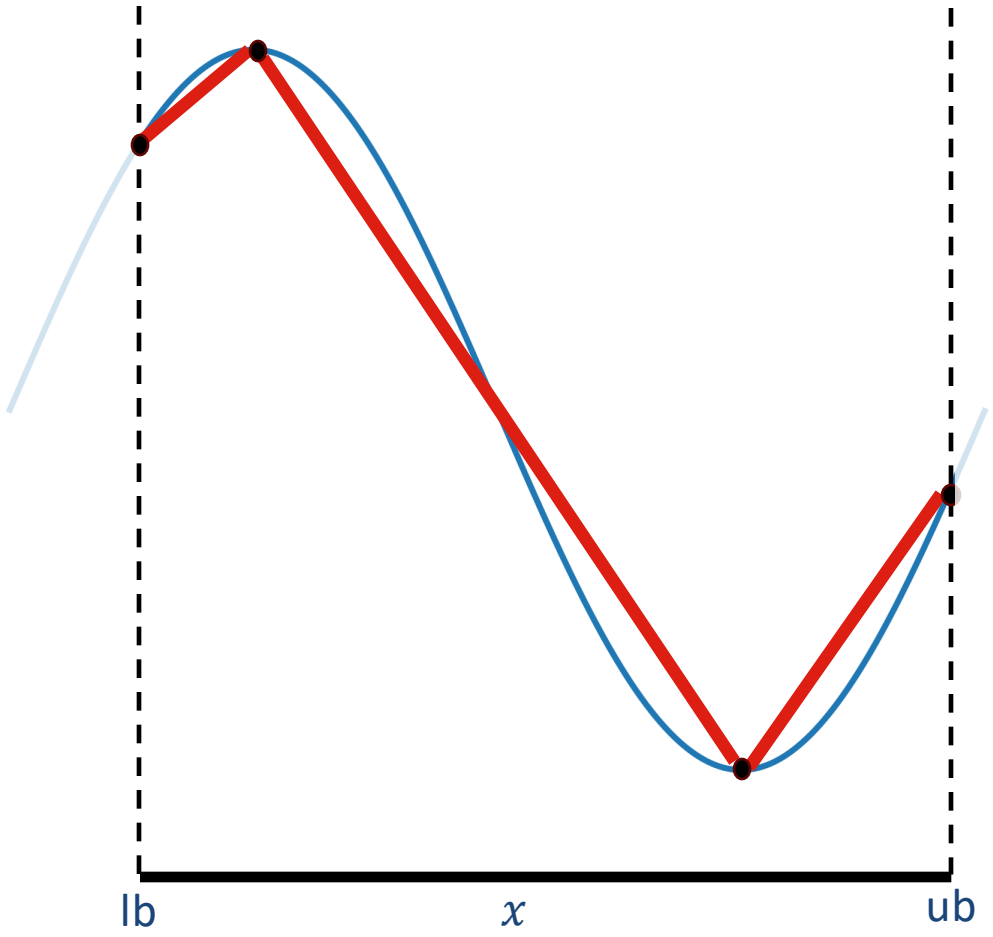
```
addGenConstrExp(), addGenConstrExpA()
addGenConstrLog(), addGenConstrLogA()
addGenConstrSin(), addGenConstrCos(), addGenConstrTan()
addGenConstrPow()
addGenConstrPoly()
```

- Gurobi 9.0 – 10.0:
  - Nonlinear functions are replaced during presolve by a piecewise-linear approximation
- Gurobi 11.0:
  - Can choose to treat nonlinear constraints exactly by a dynamic outer approximation

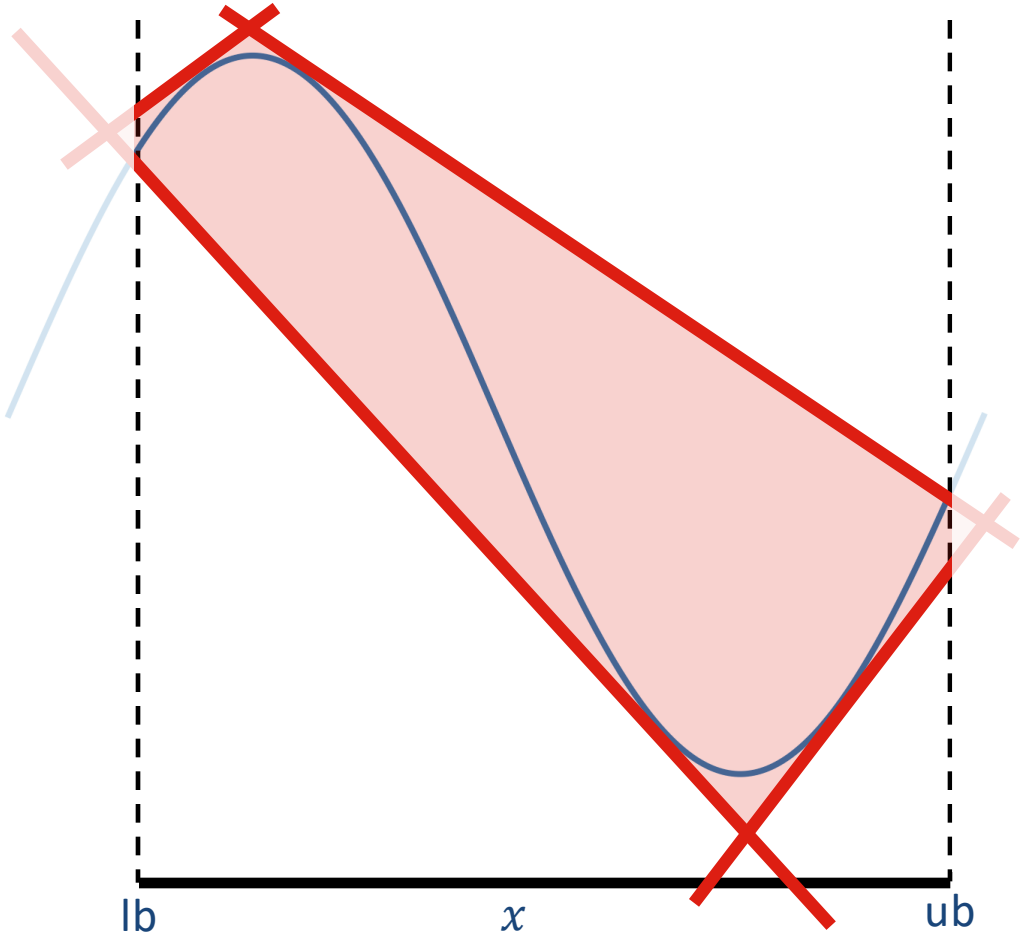
FuncNonlinear Parameter and Attributes

- Existing general function constraints attributes to control PWL approximation:
  - `FuncPieces`
  - `FuncPieceError`
  - `FuncPieceLength`
  - `FuncPieceRatio`
- Default behavior of `FuncPieces` is now to use relative error approach
  - Was mainly restricting total number of pieces in Gurobi 10.0
- New `FuncNonlinear` attribute to switch between PWL and outer approximation:
  - `-1`: behavior defined by FuncNonlinear parameter (default)
  - `0`: use static PWL approximation
  - `1`: use dynamic outer approximation
- New `FuncNonlinear` parameter to control default (-1) of attributes:
  - `0`: use static PWL approximation (default)
  - `1`: use dynamic outer approximation

PWL Approximation vs. Outer Approximation

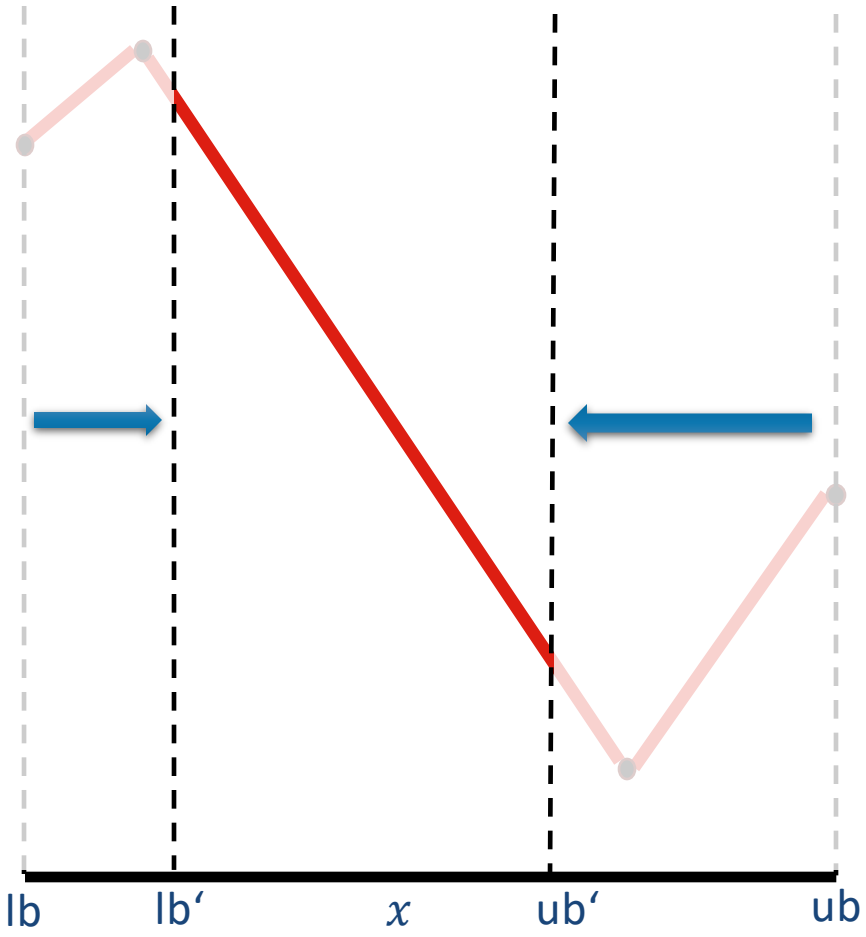


static PWL approximation

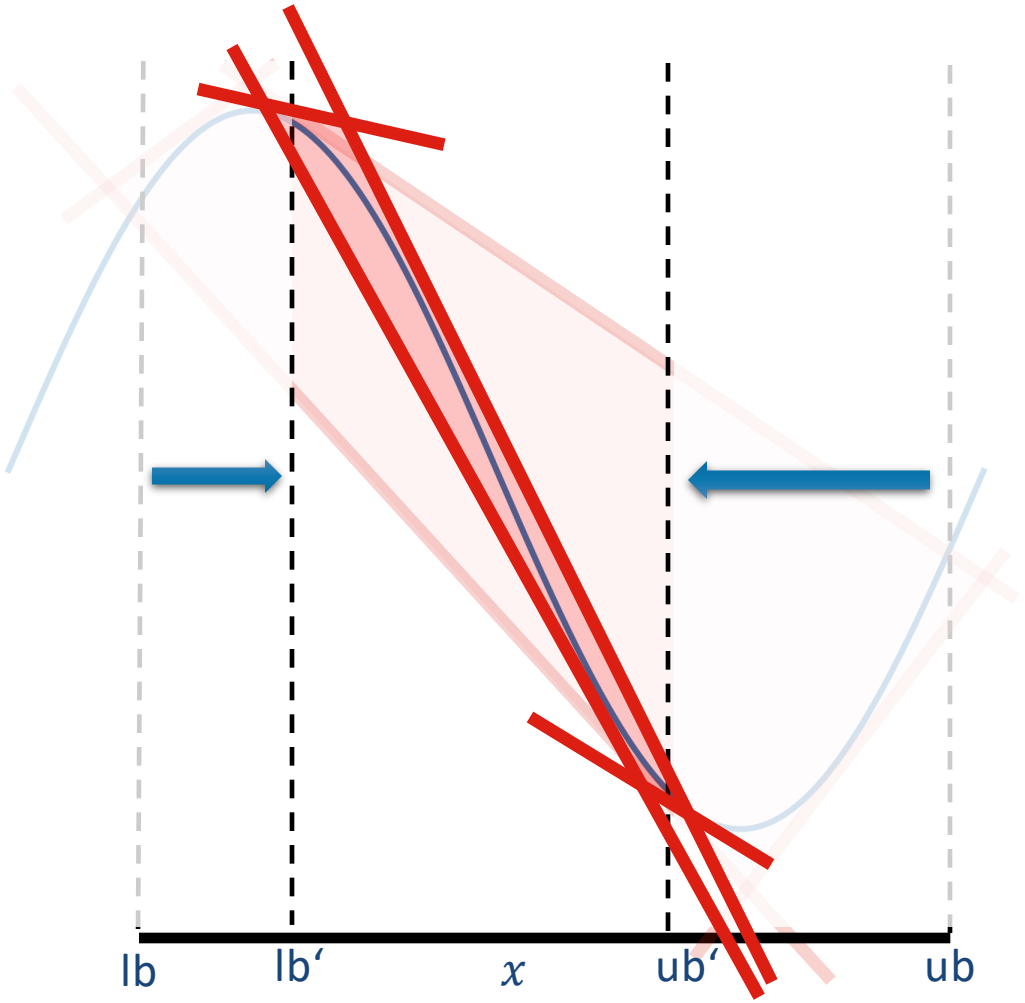


dynamic outer approximation

PWL Approximation vs. Outer Approximation



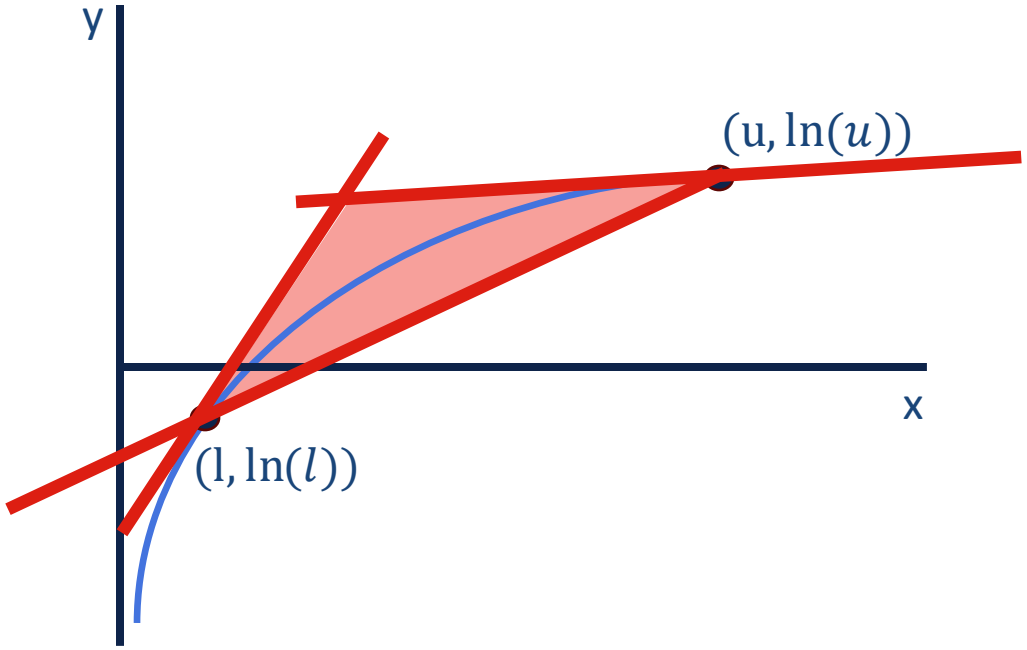
static PWL approximation

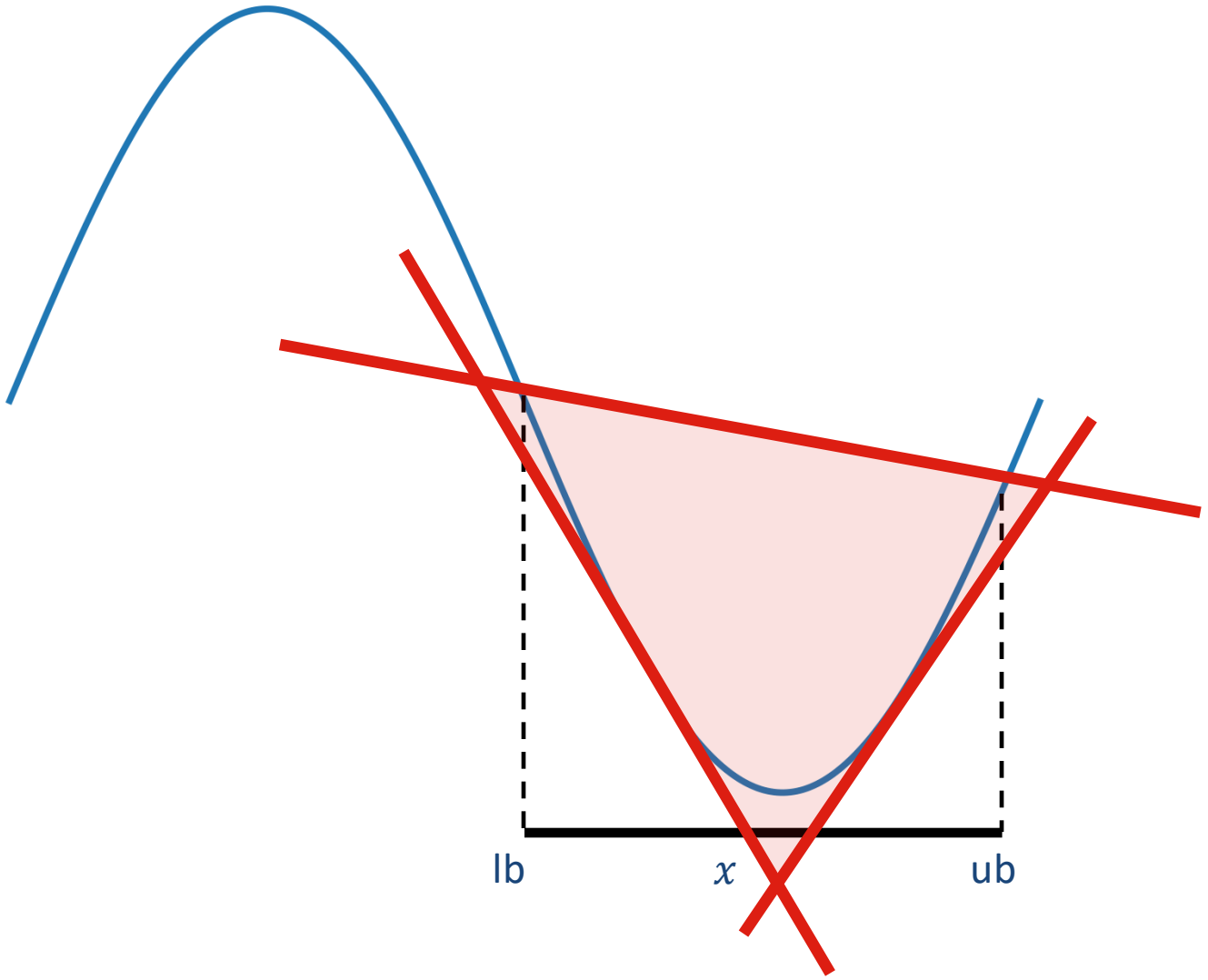


dynamic outer approximation

Relaxation of Nonlinear Functions

- Some convex envelopes are easier to approximate than others
  - Example: $y = \ln(x)$, a concave function, $l \leq x \leq u$
  - Lower envelope is given by secant through $\ln(l)$ and $\ln(u)$
  - Upper envelope is constructed by tangents





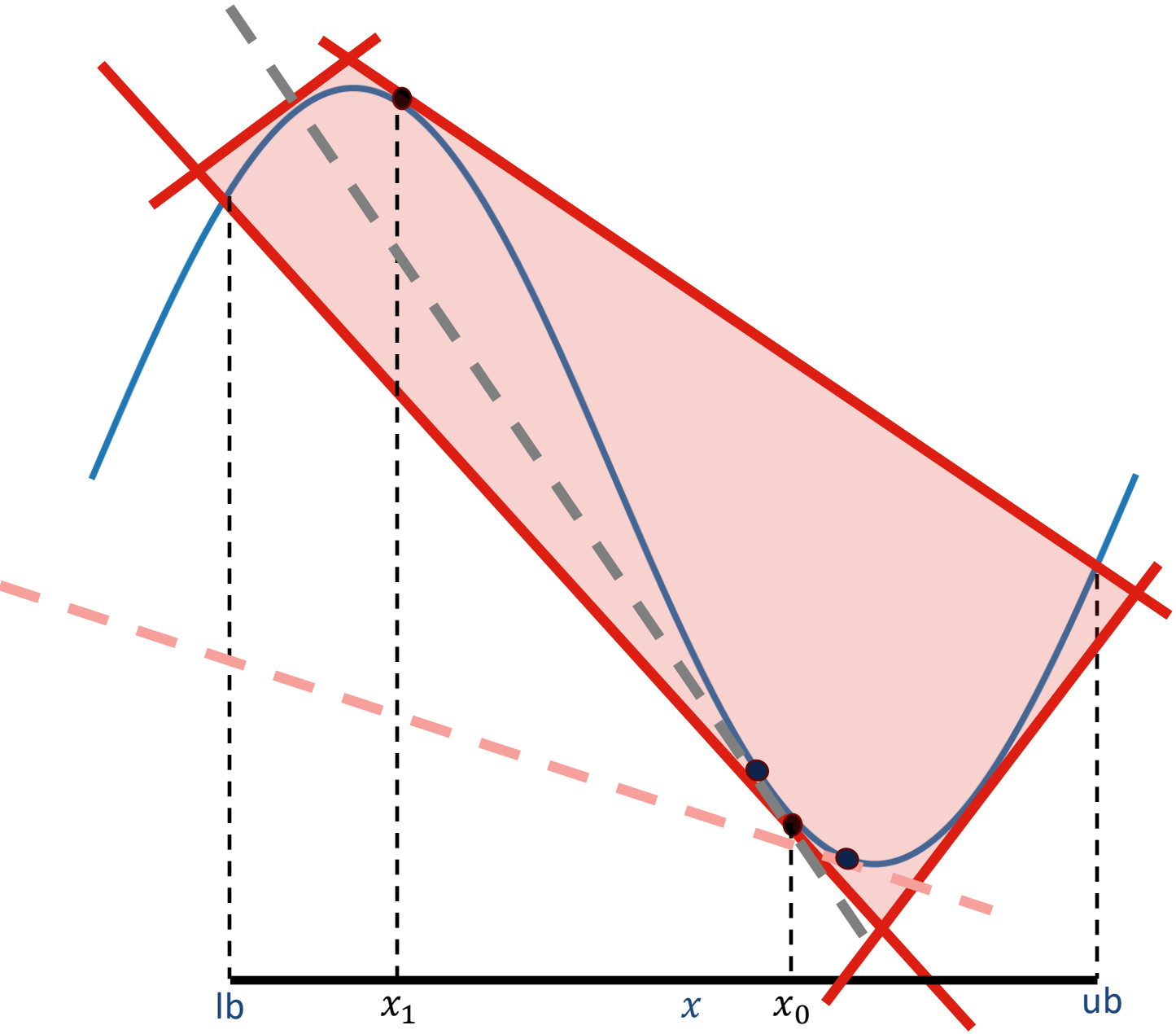
GUROBI OPTIMIZATION

Extend to More Complex Functions

If sin is convex within the bounds of x …

- Upper envelope is given by secant through $f(\mathrm{lb})$ and $f(\mathrm{ub})$
- Lower envelope constructed by tangents to $\sin$
- Resulting hyperplanes added to LP
- Shaded in red: Relaxation of $y = f(x)$

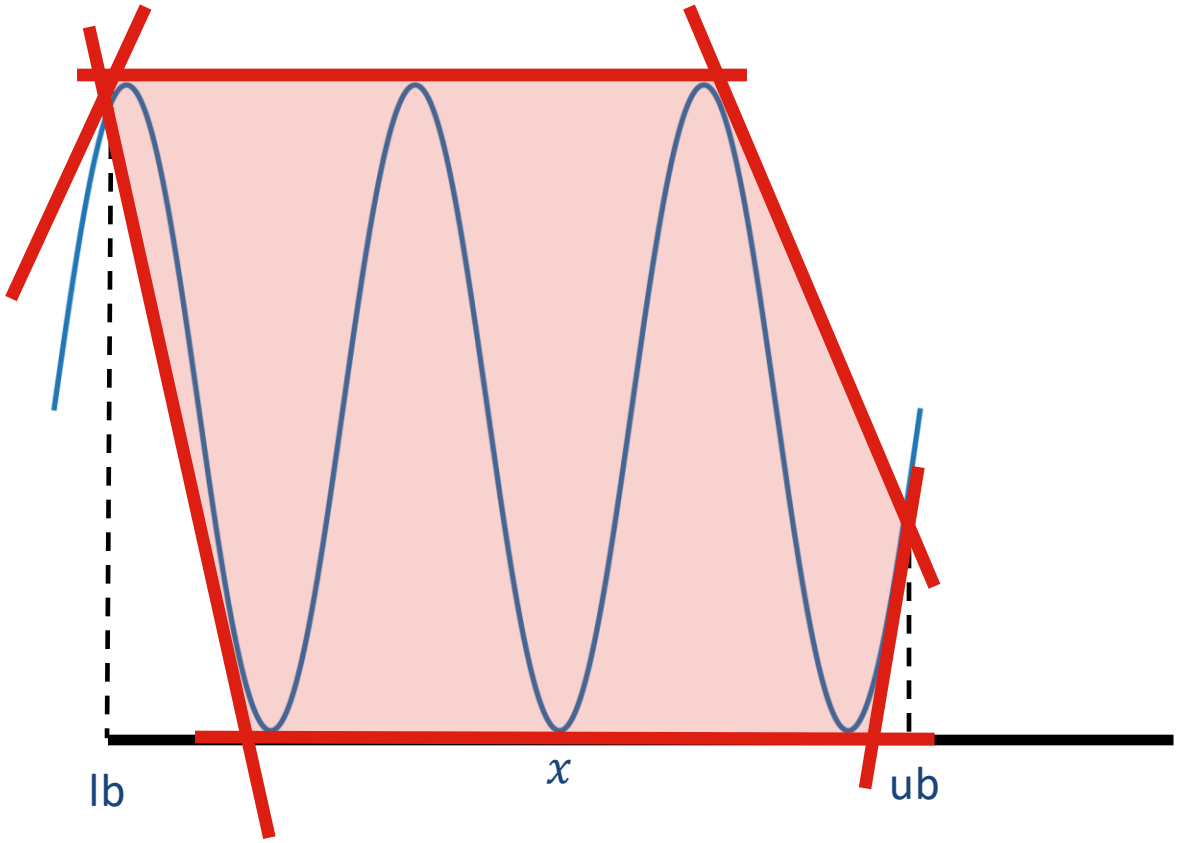
- Similar if $\sin$ is *concave* on the domain of x
- Adding more tangents at various points improves the relaxation



GUROBI OPTIMIZATION

Neither Convex Nor Concave

- If sin is neither convex nor concave on the domain of x…
- Lower envelope
  - Compute leftmost solution x_0 to $\frac{d}{dx}\sin(x) = \frac{\sin(x) - \sin(\mathrm{lb})}{x - \mathrm{lb}}$
  - Computed x_0 defines one tangent
  - Remaining part is convex, use some tangent(s)
- Upper envelope
  - Compute rightmost solution x_1 to $\frac{d}{dx}\sin(x) = \frac{\sin(\mathrm{ub}) - \sin(x)}{\mathrm{ub} - x}$
  - Computed x_1 defines one tangent
  - Remaining part is concave, use some tangent(s)

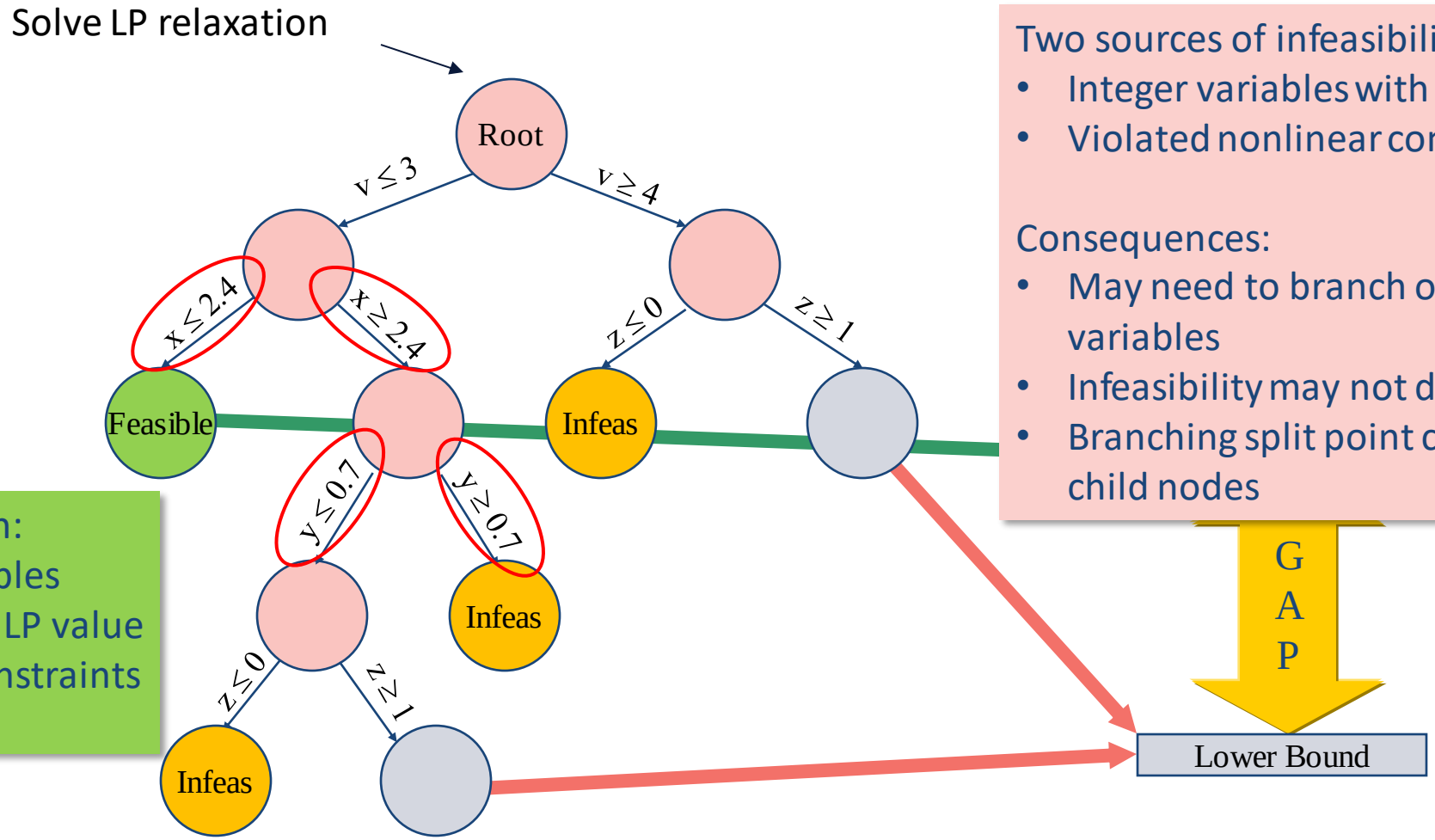


GUROBI OPTIMIZATION

"Large" Domains

- Not much to get from the relaxation if domain of x is large
- Branching on x tightens the relaxation quickly!
- Tighter initial bounds will speed up performance

Spatial Branch and Bound for MINLPs



Two sources of infeasibility:

- Integer variables with fractional LP value
- Violated nonlinear constraints

Consequences:

- May need to branch on continuous variables
- Infeasibility may not directly be resolved
- Branching split point contained in both child nodes

Feasible solution:

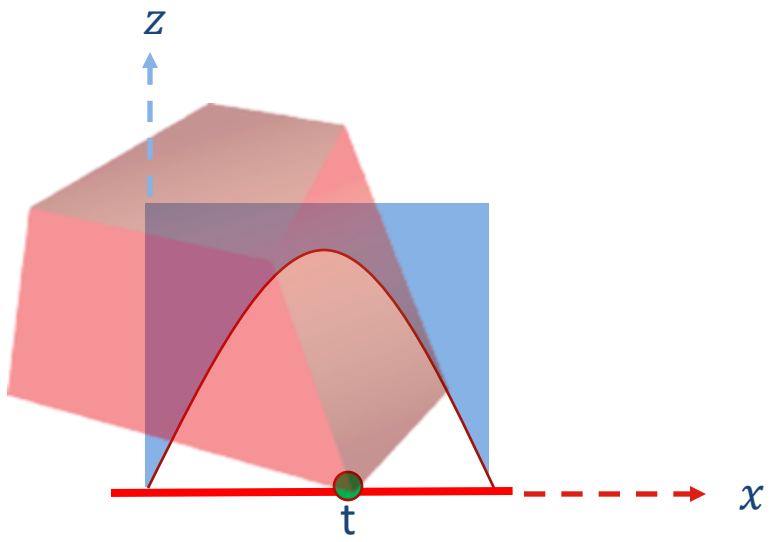
- Integer variables have integral LP value
- Nonlinear constraints are satisfied

Branching for MINLPs

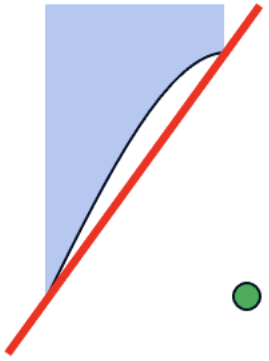
Similar to Nonconvex MIQCP

- After solving the convex relaxation, how do we branch on the violated nonconvexities?

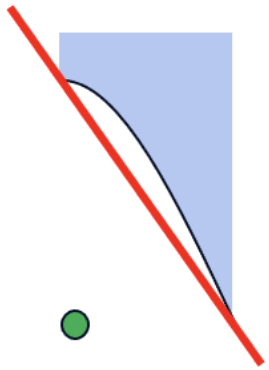
nonconvex
$-z - x^2 \leq 0$



$$
\begin{array}{lccccc}
\min & c^T x & + & d^T z & & \\
\text{s.t.} & Ax & + & Dz & \leq & b \\
 & -x_i x_j & + & z_{ij} & = & 0 \quad \text{for all } (i,j) \in S \\
 & l & \leq & x & \leq & u \\
 & & & x_j & \in & \mathbb{Z} \quad \text{for all } j \in I
\end{array}
$$



branching
$x \leq t$ or $x \geq t$



update relaxation bounds and the associated McCormick envelopes locally

Composite Nonlinear Functions

- Gurobi 11.0 can handle selected univariate constraints $f(x) = y$
  - Trigonometric, power functions, logarithms, exponentials, etc.
  - Use them as building blocks for more elaborate functions
- Example: Suppose we want to model

$$f(\mathrm{x}) = \underbrace{\sqrt{\underbrace{1+\mathrm{x}^2}_{u=1+\mathrm{x}^2}}}_{v=\sqrt{u}} + \ln\Big(\underbrace{\mathrm{x}+\underbrace{\sqrt{1+\mathrm{x}^2}}_{v}}_{w=\mathrm{x}+\mathrm{v}}\Big)}^{z=\ln w} \leq 2,\ \mathrm{x} \geq 0$$

- We introduce auxiliary variables $u, v, w, z \geq 0$ and constraints as follows:
  - $\mathrm{u} = 1 + \mathrm{x}^2,\ u = v^2,\ w = \mathrm{x} + v,\ z = \ln w$
  - Then $f(\mathrm{x}) \leq 2$ can be represented as $v + z \leq 2$

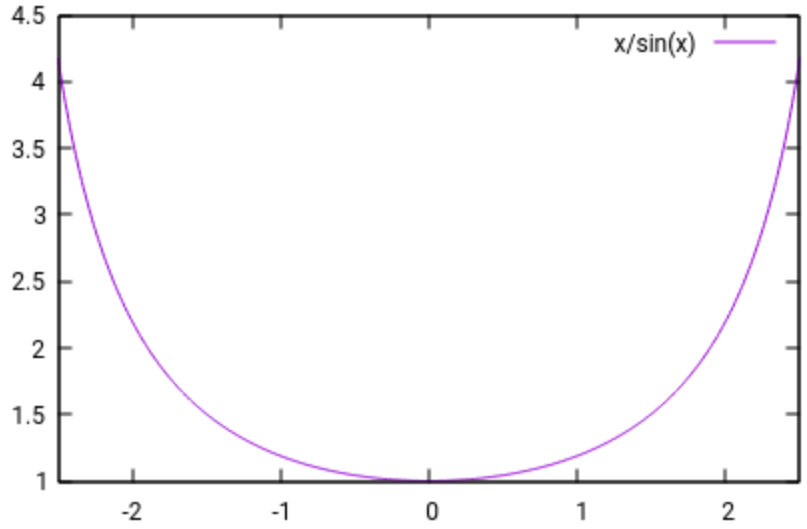
Limitations of Gurobi 11 Approach

- Need to model composite nonlinear constraint as a set of
  - linear,
  - quadratic, and
  - univariate nonlinear constraints
- Each individual constraint is subject to feasibility tolerance
- Result could be that composite constraint is violated much more

- Note that most solvers do the same decomposition under the hood
  - Gurobi 11 is only missing the control on the overall composite constraint violation

Limitations of Gurobi 11 Approach

- Example: $\mathrm{y} = \frac{x}{\sin x}$
- One solution:
  - $x' = 0.0001$
  - $y' = 1.00000000016666666$



- Gurobi model: $u = \sin x, v = u^{-1}, y = x \cdot v$
- One solution:
  - $x' = 0.0001$
  - $u' = 0.000099999999833333343$
  - $v' = 10000.000016666666$
  - $y' = 1.00000000016666666$

- A solution with a violation within a tolerance of 10^{-6}:
  - $x'' = 0.0001$
  - $u'' = 0.000098999999833333343$ $\quad u'' = \sin x'' - 10^{-6}$
  - $v'' = 10101.010118015167$
  - $y'' = 1.0101010118015167$

Violation of 10^{-6} in auxiliary constraint leads to violation of 10^{-2} in composite constraint

Composite function behaves very nicely for $x \in [-2.5, 2.5]$: no numerical issues expected

MINLP Roadmap for Gurobi 12

- Add API to state composite nonlinear functions directly
  - Use composite nonlinear functions for feasibility checks
  - Use composite nonlinear functions for interior point NLP solver
  - Exploit knowledge about composite nonlinear in presolve and for outer approximation
- Improve global MINLP performance
  - Presolve reductions
  - Cutting planes
  - Improve heuristics to better work with nonlinear constraints
  - Better branching variable and split point selection
- Interior point local NLP solver
  - Expose our internal local NLP solver to the user
    - Provides a locally optimal solution
  - Improve performance and robustness of local NLP solver
- Improve numerics

GUROBI OPTIMIZATION

Additional Features

Supported Software Versions

- Added support for Xcode 14 on macOS
- Added support for Python 3.12
- Added support for MATLAB R2023b

Copying Models Between Environments

- Copy model from one environment to another

```
c = m.copy()      # regular copy: c in same environment as m
c = m.copy(env)   # new: c is created in environment env
```

- Use case: parallel execution of two optimization runs

- Caveat:
  - Can copy to a remote (Compute Server) model but not from a remote model

Interrupt and Resume with Change of Threads

- Interrupting a solve and then calling optimize() again:
  - Gurobi 10: changes to `Threads` parameter in between are ignored
  - Gurobi 11: changes to `Threads` parameter will be obeyed when resuming

- Example use case:

```
m.Params.Threads = 8
m.Params.SoftMemLimit = 4
m.optimize()
if m.status == gp.GRB.MEM_LIMIT:
  m.Params.Threads = 1
  m.optimize()
```


GUROBI OPTIMIZATION

APIs: Java & Gurobipy Enhancements

Java API

GUROBI OPTIMIZATION

- Java package name is now `com.gurobi.gurobi` instead of `gurobi`
  - To follow Java standard naming scheme
- Java package now distributed on Maven Central
  - Most popular Java package repository
  - Similar to PyPI for Python
  - Helps build and deployment processes for Java users

gurobipy: Installation Changes

- Type hints now included in gurobipy
  - No more `gurobipy-stubs`
- `setup.py install` is no more
  - Offline installs are possible with pip
  - Hash verification is possible with pip
- conda and pip play nicely together
  - No more duplicate installs
  - Cleaner install for our open-source packages on conda

```
The conflict is caused by:
    The user requested gurobipy==11.0.0b1
    gurobipy-stubs 2.0.0 depends on gurobipy==10.*
    The user requested gurobipy==11.0.0b1
    gurobipy-stubs 1.0.1.post0 depends on gurobipy==9.5.*
```

```
# Name                    Version                   Build  Channel
blas                      1.0                         mkl
bottleneck                1.3.5           py311hb9e55a9_0
bzip2                     1.0.8                h1de35cc_0
ca-certificates           2023.08.22           hecd8cb5_0
gurobi                    10.0.3                  py311_0  gurobi
gurobipy                  10.0.3                  pypi_0   pypi
gurobipy-pandas           1.0.0                   pypi_0   pypi
intel-openmp              2023.1.0        ha357a0b_43547
libcxx                    14.0.6               h9765a3e_0
```

gurobipy: Matrix-friendly API Integration

GUROBI OPTIMIZATION

- Callback functions now accept matrix-friendly API objects

```
x_sol = model.cbGetSolution(x)
model.cbLazy(A @ x <= b)
```

- Numpy-style concatenation (hstack, vstack, concatenate)

```
X = model.addMVar((n, m))
Y = model.addMVar((n, k))
XY = gp.concatenate((X, Y), axis=1)  # (n, m+k) MVar
```

- Matrix-friendly indicator constraints (vectorized, broadcastable)

```
z = model.addVar(vtype=GRB.BINARY)
x = model.addMVar(n)
model.addGenConstrIndicator(z, True, A @ x <= b)  # MGenConstr ...
```

gurobipy: Other Notable Mentions

- Any callable object can be a callback
  - Makes callable classes an option for callbacks
  - Avoids the `model._attribute` workaround
  - Check out the refreshed tsp.py and callback.py examples

- Performance improvements
  - `addConstr(A @ x == b)` ~2x faster for sparse data
  - ~10-20% faster term-based modeling patterns (credit to the Cython developers for that one!)

- Check out the Detailed Release Notes for a complete list

```
52 class TSPCallback:
53     """Callback class implementing lazy constraint
54     callbacks, solutions are checked for subtours
55     constraints are added if needed."""
56
57     def __init__(self, nodes, x):
58         self.nodes = nodes
59         self.x = x
60
61     def __call__(self, model, where):
62         """Callback entry point: call lazy constra
63         solutions are found. Stop the optimization
64         user code."""
65         if where == GRB.Callback.MIPSOL:
66             try:
67                 self.eliminate_subtours(model)
68             except Exception:
69                 logging.exception("Exception occur
70                 model.terminate()
71
```

GUROBI OPTIMIZATION

Dynamically Distributed Tuning

Gurobi Tuner Basics

- Automatic way to handle the huge number of possible parameter configurations when faster performance is needed
- Quickstart Guide
  - Interactive run, either from grbtune executable, or via Python API

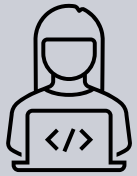
```
>>> m = gp.read("mymodelneedstorunfaster.mps")
>>> m.tune()

>>> m = gp.read("mymodelneedstorunfaster.mps")
>>> m.setParam("Heuristics", 0)          # Heuristics disabled for all non default tuning runs
>>> m.tune()
```

  - More elaborate configurations possible
    - Tuning time limit, number of tuning runs, etc.
- Gurobi runs multiple tuning configurations, recommends best set of parameters

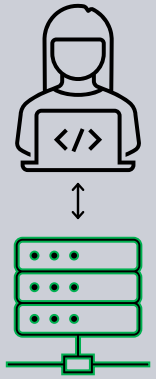
Three Ways of Tuning

GUROBI OPTIMIZATION

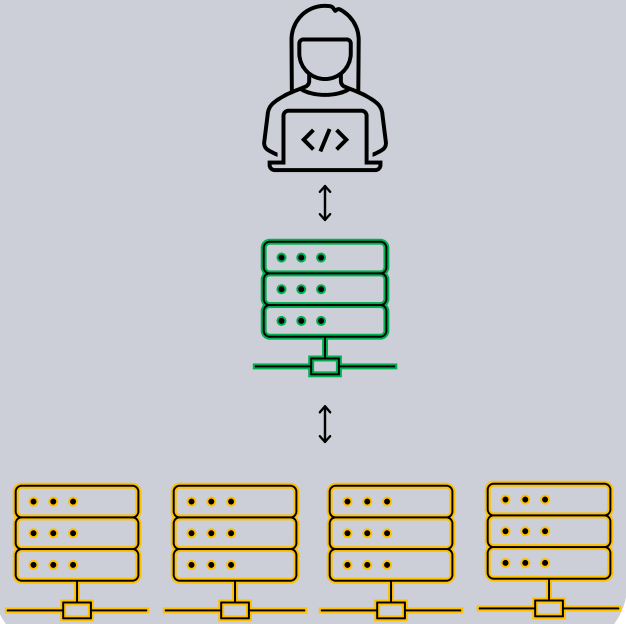
Local



Remote



Distributed

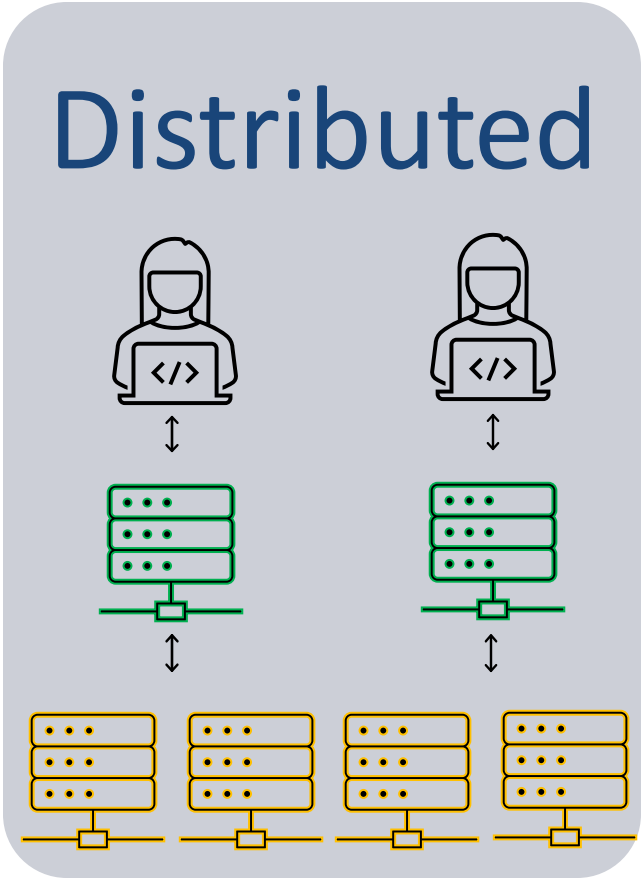


Distributed Tuning

- Significantly increase the performance of the tuning tool
- Previously, used a static set of remote workers

- Now, can handle a dynamic set of remote workers
- Remote workers are used for limited amount of time
- Returned such that they are available for other remote jobs
- Number of used remote workers are scaled down and up automatically.

Tuning tool can make use of idle remote workers!

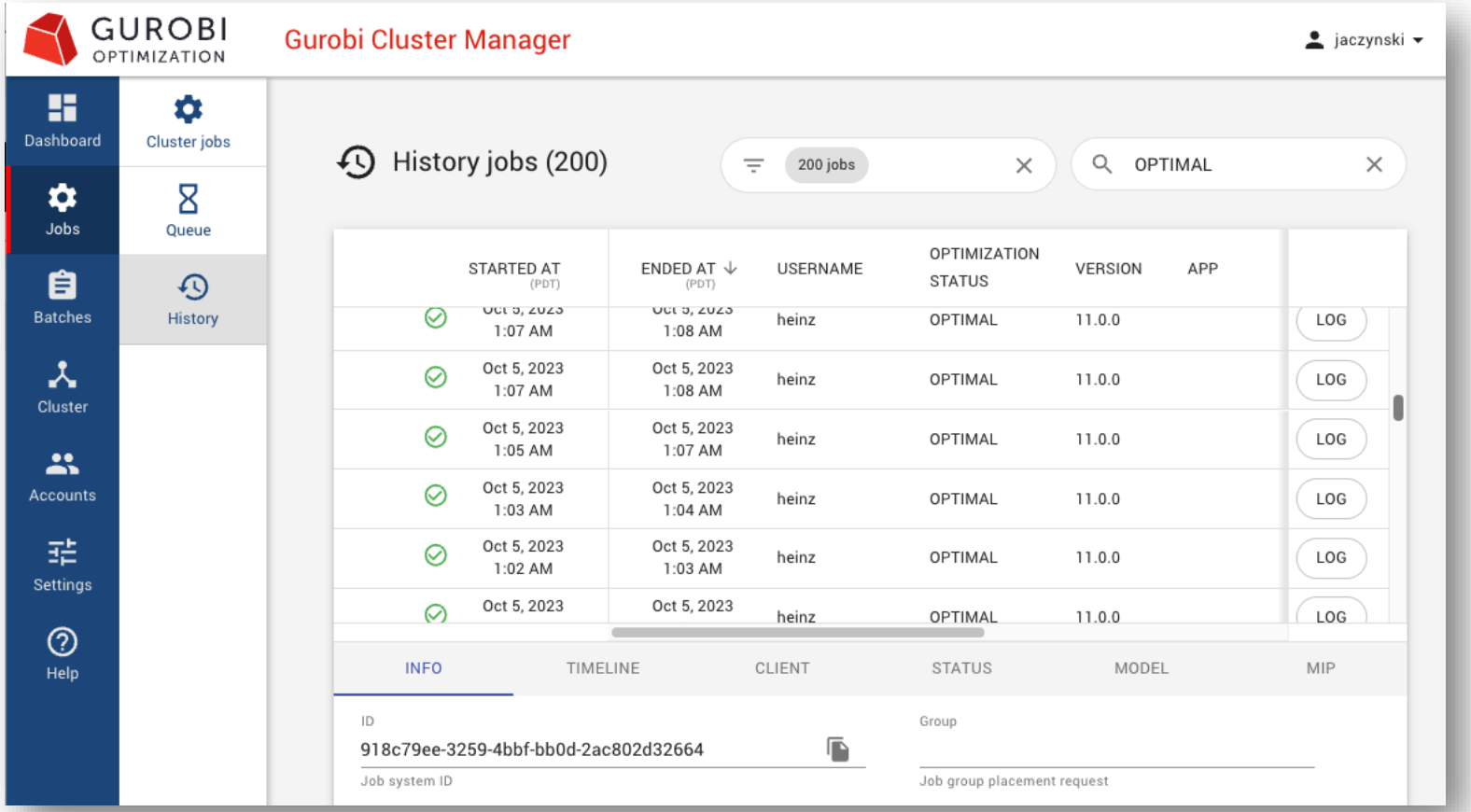


GUROBI OPTIMIZATION

Cluster Manager and Compute Server Enhancements

Improved Cluster Manager/Compute Server

Compute Server/Cluster Manager facilitates the deployment and use of optimization services on-premises or on private cloud.

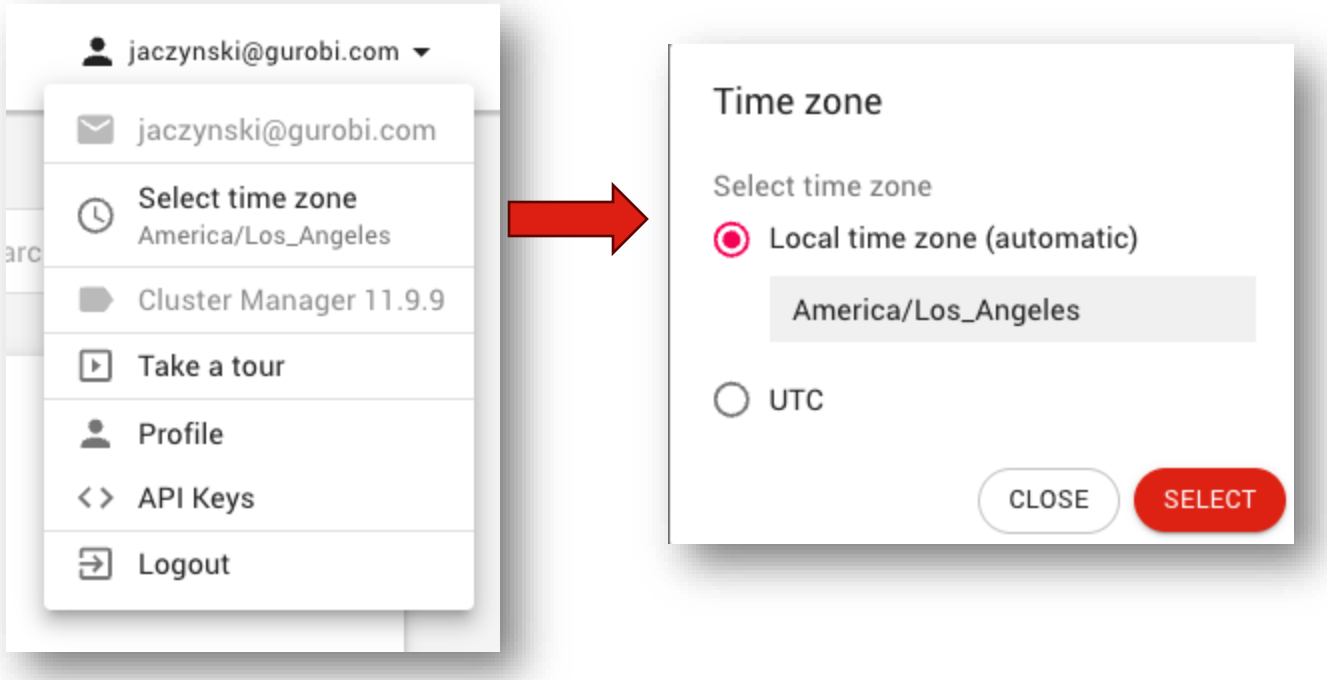


New Look and Feel

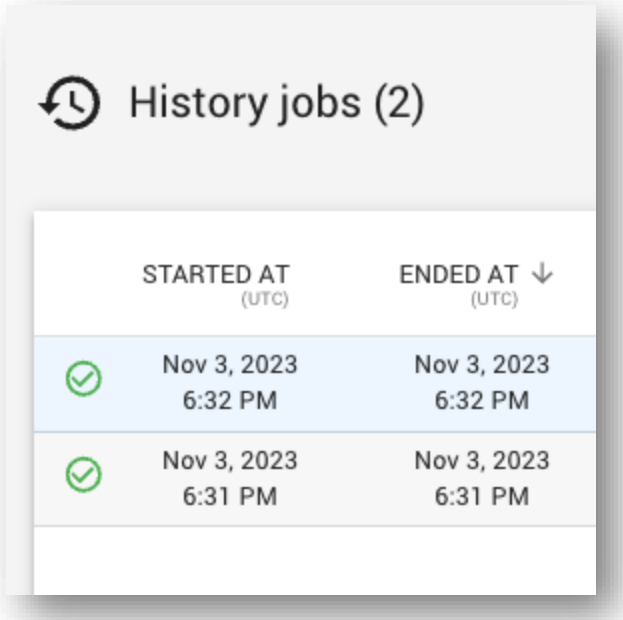
- User Management
- Cluster Monitoring
- Optimization Job Monitoring

Grade from https://securityheaders.com/

Time Zone Selection and Formatting



From the user menu, select your local time zone or UTC to display date/time



Time zone is indicated in column headers and property tabs

SAML 2.0 Authentication

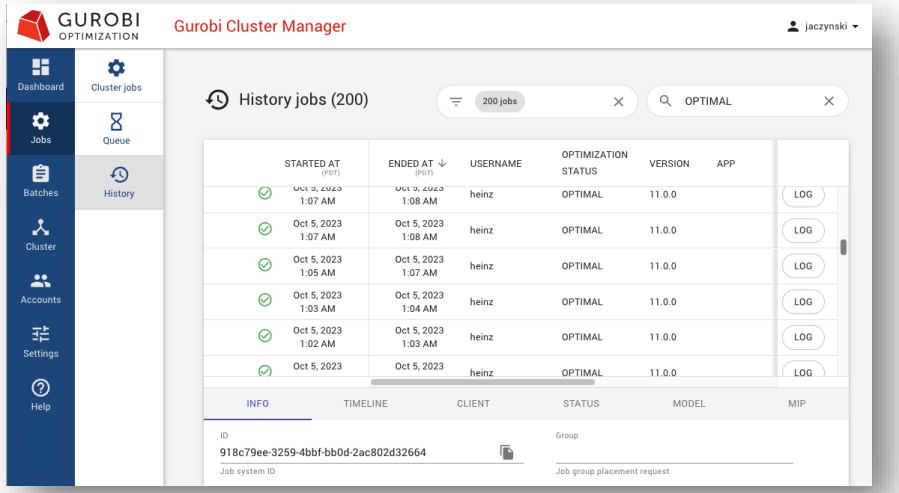
Azure Active Directory

okta

Google Workspace

jumpcloud

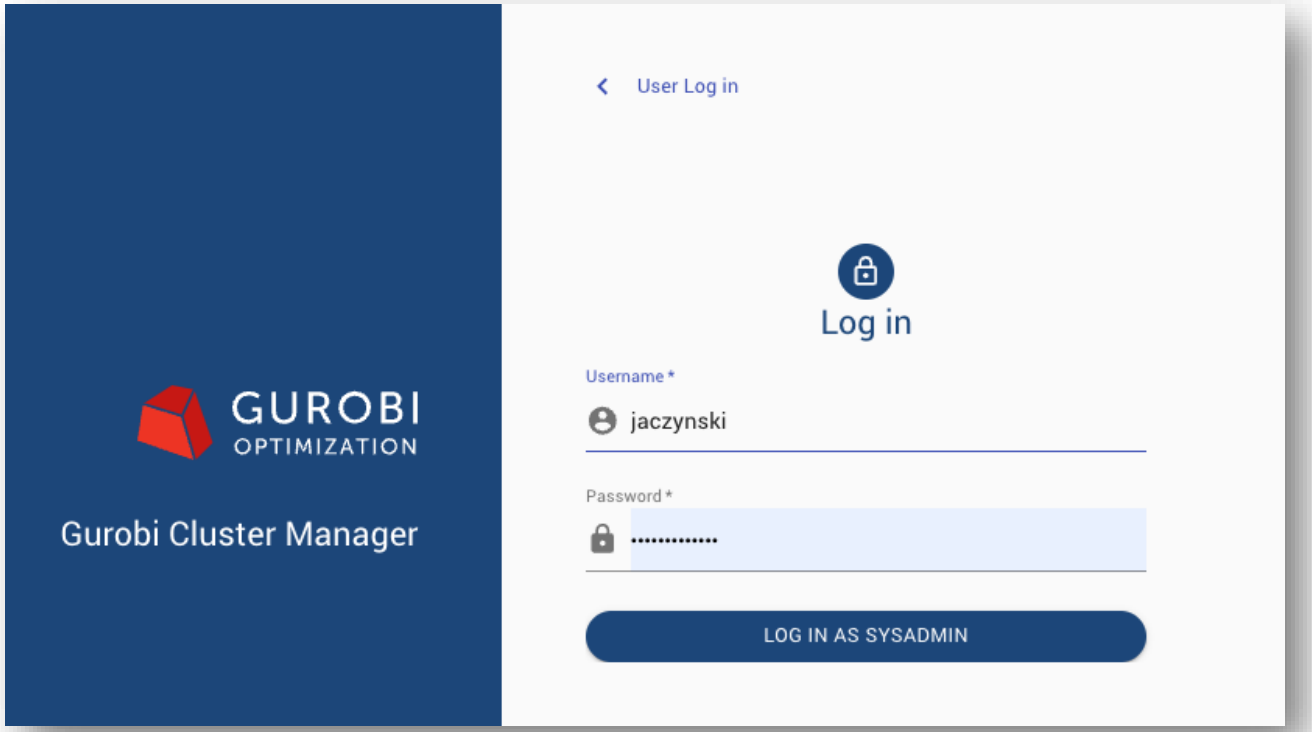
In addition to LDAP, v11 adds support for SAML 2.0 user directories for flexible enterprise integration



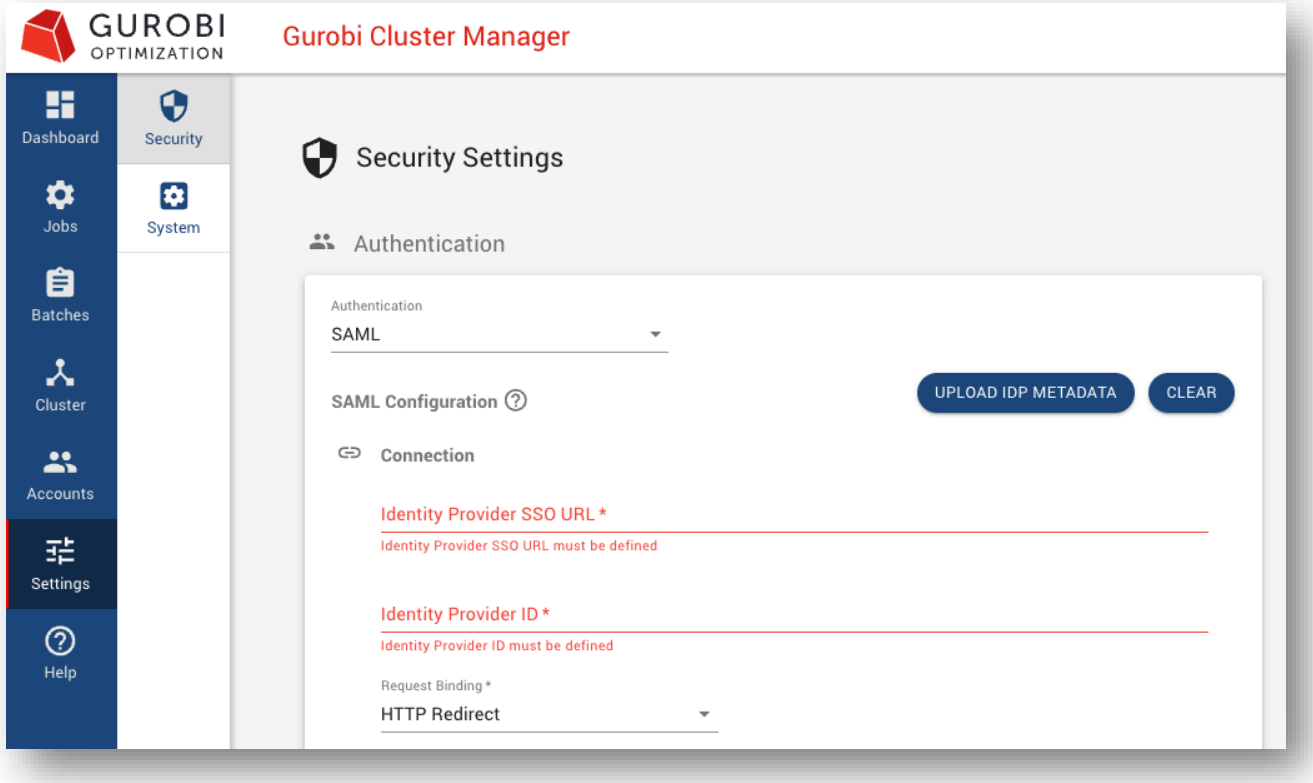
SAML 2.0 Sysadmin Login

Sysadmin user can still login locally, so that access can be managed even in case of problem with SAML 2.0 configuration

http://clustermanager/login/admin



SAML 2.0 Configuration



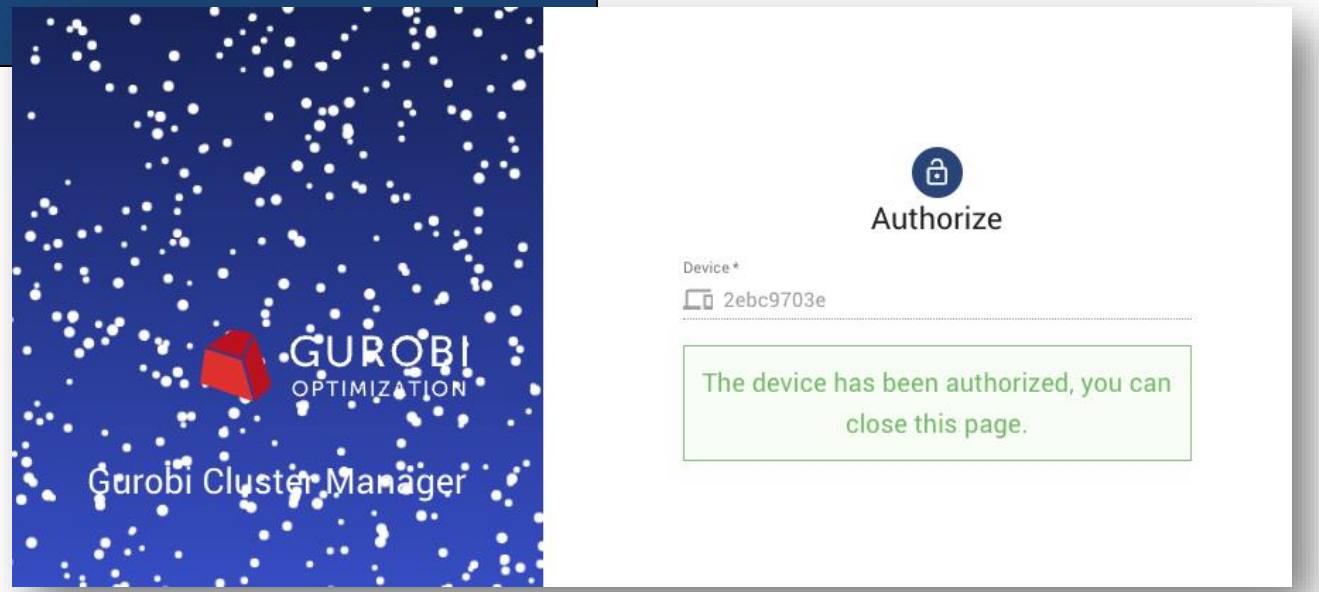
- Import Identity Provider XML metadata
- Import certificate to verify authentication
- Request binding (POST, Redirect)
- Custom mapping of user attributes (name, email)
- Case sensitivity settings
- Download Service Provider XML metadata

SAML 2.0 Command Line Authentication

```
~ $ export GRB_LICENSE_FILE=gurobi.lic
~ $ grbcluster login --manager https://mycluster --username user@gurobi.com
info : Using client license file 'gurobi.lic'
---------------
A new authorization request has been created for the user: user@gurobi.com
If the browser did not automatically open, use a web browser to open the page:
https://mycluster/authorize?device=2ebc9703e
If requested, please enter the following device ID code to authenticate: 2ebc9703e
---------------
```

(1) Login request is made on the command line for a specific username

(2) Authentication is done in the browser

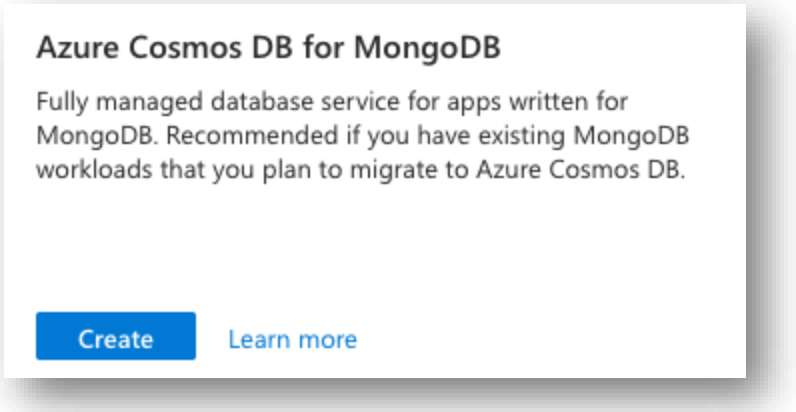


(3) command line returns when authentication is done.

Improved Database Support

New support of Cosmos DB 4.2 on Microsoft Azure

Azure CosmosDB



Existing database support of MongoDB and Amazon DocumentDB

mongoDB.

Amazon DocumentDB

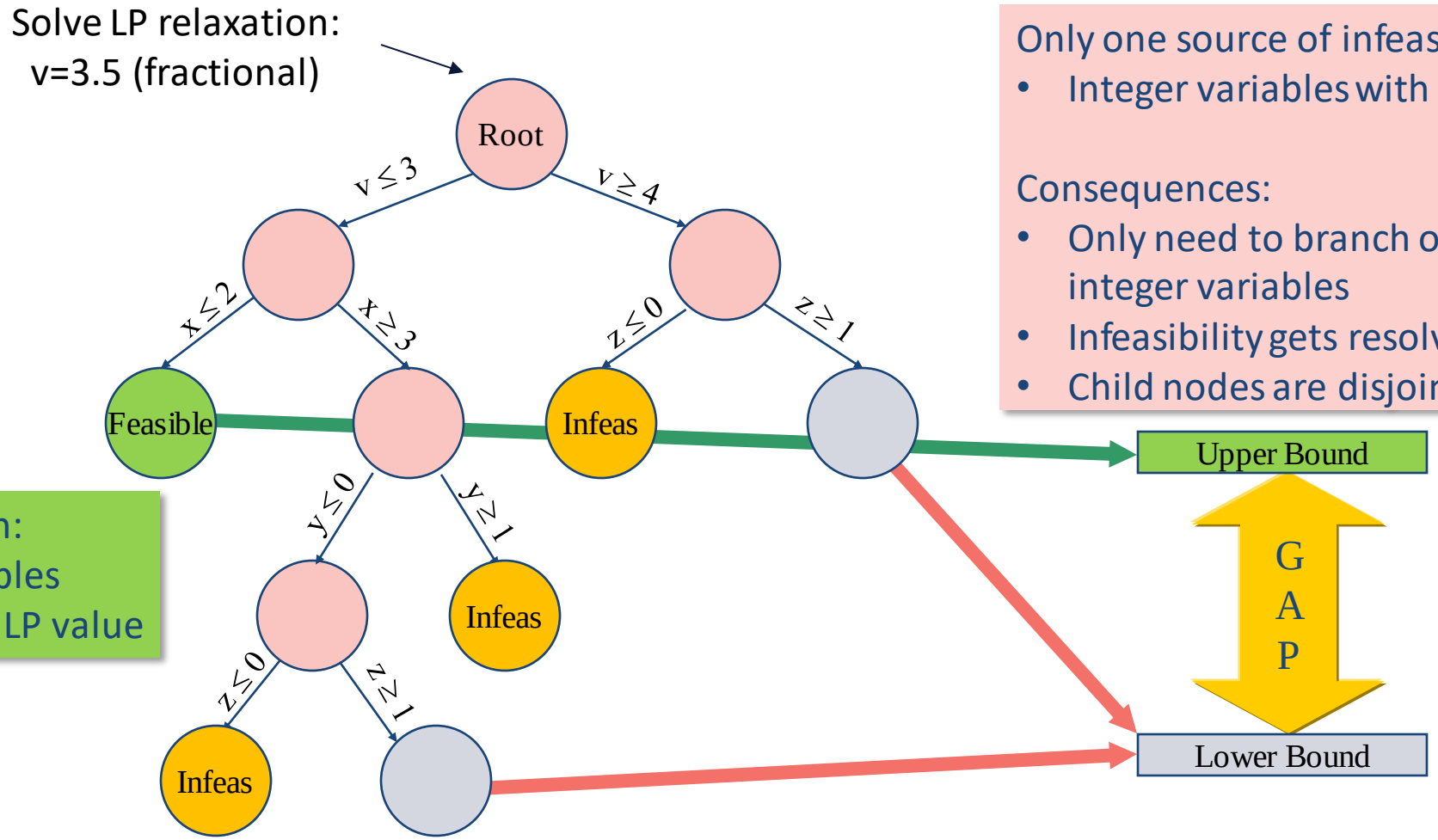
Better Server-Side Limits

- Server-side limits can be enforced by IT as a policy, clients cannot override these limits
- MEMLIMIT:
  - Limits the total amount of memory. If more is needed, Gurobi will fail with an OUT_OF_MEMORY error. Note that it is not possible to retrieve solution information after this termination error.
- SOFTMEMLIMIT
  - Limits the total amount of memory available to Gurobi. If more is needed, Gurobi will terminate with a MEM_LIMIT status code, leading to a graceful exit of the optimization, such that it is possible to retrieve solution information afterwards or (in the case of a MIP solve) resume the optimization.
- TIMELIMIT
  - Limits the total time expended during optimization. If the limit is reached, the optimization will return the TIME_LIMIT status.

GUROBI
OPTIMIZATION

Questions?

Branch and Bound for MILPs



Only one source of infeasibility:

- Integer variables with fractional LP value

Consequences:

- Only need to branch on fractional integer variables
- Infeasibility gets resolved in child nodes
- Child nodes are disjoint

Feasible solution:

- Integer variables have integral LP value

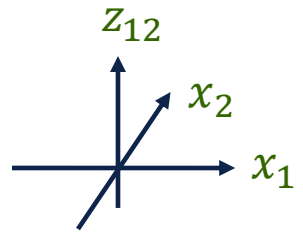
Recap: Nonconvex MIQCP

- Introduce auxiliary variables

$$z_{ij} := x_i x_j$$

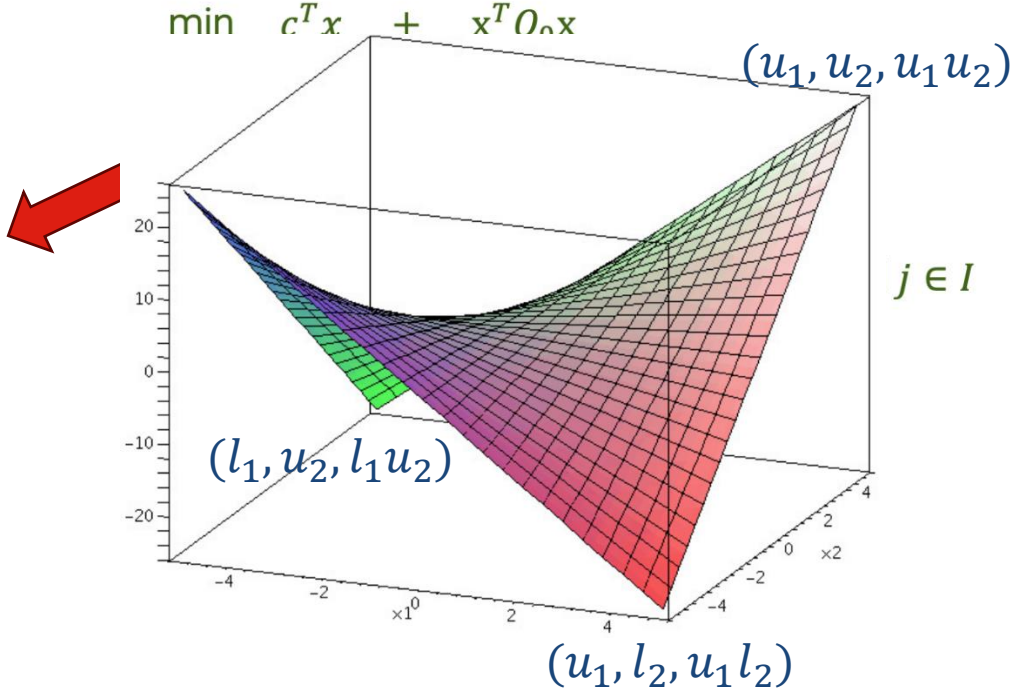
for each product term $x_i x_j$ that appears in some (indefinite) $\mathrm{Q} = Q_k$ with $Q_{ij} \neq 0$

Let S index all such (i,j) pairs



- A Mixed Integer Bilinear Program is defined as

$$\begin{array}{lllllll} \min & c^T x & + & d^T \mathrm{z} & & & \\ \text{s.t.} & Ax & + & D\mathrm{z} & \leq & b & \\ & -x_i x_j & + & \mathrm{z}_{ij} & = & 0 & \text{for all } (i,j) \in S \\ & l & \leq & x & \leq & \mathrm{u} & \\ & & & x_j & \in & \mathbb{Z} & \text{for all } j \in I \end{array}$$



McCormick Relaxation of Bilinear Constraints

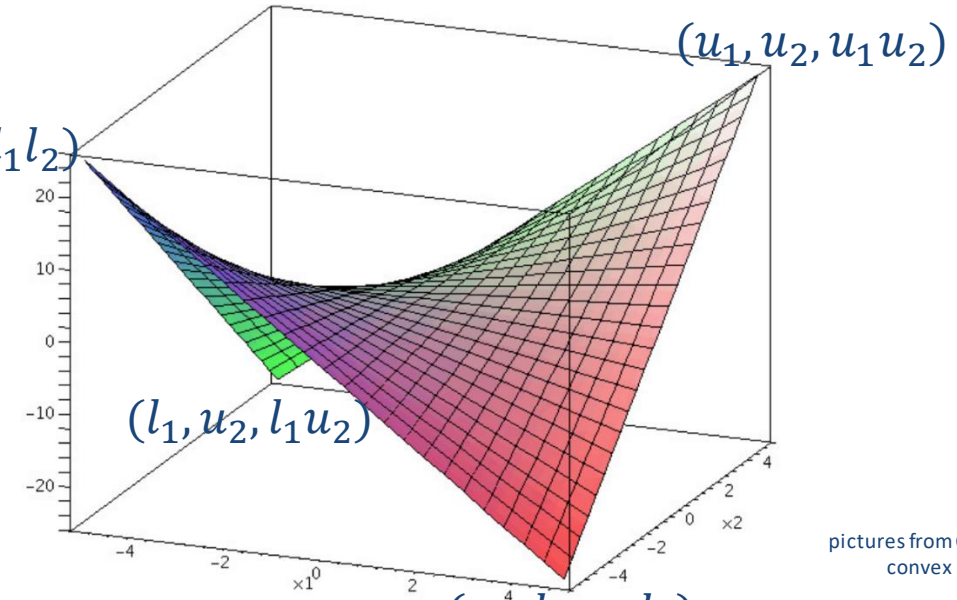
- Mixed product case: $-z_{ij} + x_i x_j = 0$

- McCormick lower and upper envelopes:

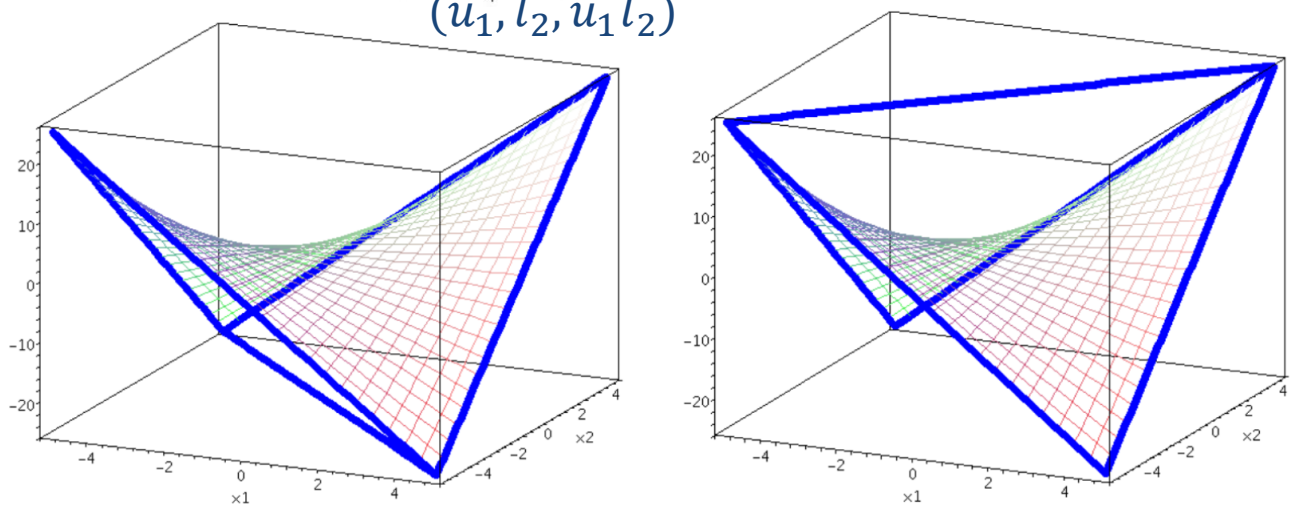
$$-z_{ij} + l_j x_i + l_i x_j \leq l_j l_i$$
$$-z_{ij} + u_j x_i + u_i x_j \leq u_j u_i$$

$$-z_{ij} + u_j x_i + l_i x_j \geq u_j l_i$$
$$-z_{ij} + l_j x_i + u_i x_j \geq l_j u_i$$

coefficients depend on local bounds



pictures from Costa and Liberti: "Relaxations of multilinear convex envelopes: dual is better than primal"



Feasibility Tolerance for Nonlinear Constraints

As Defined in Gurobi 11.0

- Linear and quadratic constraints

  $$y - f(x) = 0$$

  are defined as **feasible within tolerance** $\varepsilon > 0$ if

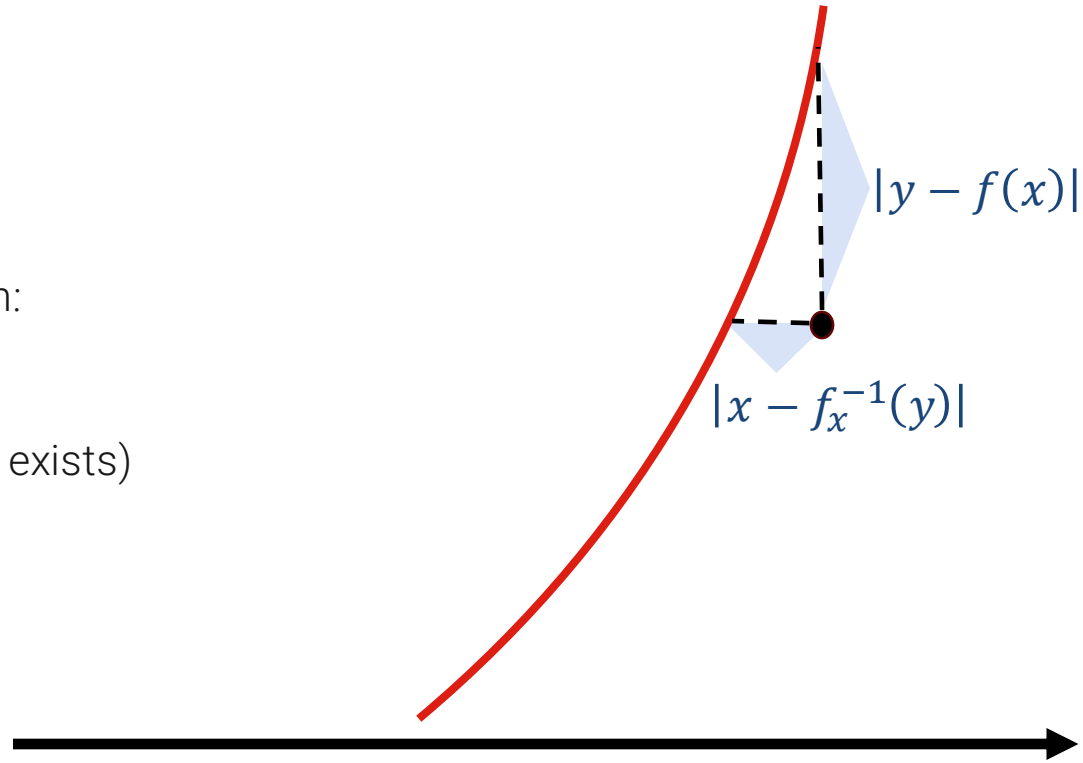
  $$|y - f(x)| \leq \varepsilon$$

- For nonlinear constraints we use a different definition:

  $$\min\{|y - f(x)|, |x - f_x^{-1}(y)|\} \leq \varepsilon$$

  with $f_x^{-1}(y)$ being the preimage of y closest to x (if it exists)

- Example: $y = 10^x$
  - $x = 10 + 10^{-6}, y = 10^{10}$
  - $|y - f(x)| = 23025.877$
  - $|x - f_x^{-1}(y)| = 10^{-6}$



gurobipy: Debugging Assists

GUROBI OPTIMIZATION

- Disable the default environment (opt-in feature)
  - Set environment variable `GUROBIPY_ALLOW_DEFAULTENV=0`
  - Helps with debugging token & remote job leaks, thread safety

```
>>> import gurobipy as gp
>>> model = gp.Model()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "src/gurobipy/model.pxi", line 80, in gurobipy.Model.__init__
  File "src/gurobipy/gurobi.pxi", line 55, in gurobipy.gurobi._getdefaultenv
gurobipy.GurobiError: Tried to start the default environment, but GUROBIPY_ALLOW_DEFAULTENV=0 is set
```

- Less silent failures
  - `Env.setParam` raises an exception for unknown parameters
  - `Model.getAttr/setAttr` raise an exception for variables not in a model
  - Too many keys passed to `select/sum/prod` raises an exception