Energy Innovation Summit
Modeling Meets Optimization

Enhancing Energy Planning with Gurobi

Bridging Historical Challenges and Modern Optimization Tools

Christine Tawfik
Optimization Engineer

Agenda

Optimization in the Energy Sector

- Early challenges
- MIP progress and real-world feasibility
- From scaling up to bottlenecks in modern energy systems

Relevant Gurobi features

- Multiple objectives
- Multiple scenarios
- Variable start & hint values
- No relaxation heuristic

Success story with Gurobi

Q&A

History of Optimization in the Energy Sector

70+ years of problem-solving innovation

1940s and 50s

Linear (LP) and mixed-integer programming (MIP) introduced

Production planning and power generation scheduling

1960s and 70s

Modeling of reservoirs

Optimization of plant operations

1980s and 90s

Maintenance scheduling

Early integration of renewables

Key event: California 7-day unit commitment problem solved in 22 minutes

2000s and 10s

Breakthroughs in solver capabilities

Scaling up Optimization applications in Energy

MIP streamlined in the electrical power industry

2020s

Investment planning

Carbon footprint reduction

Wider integration of distributed energy resources

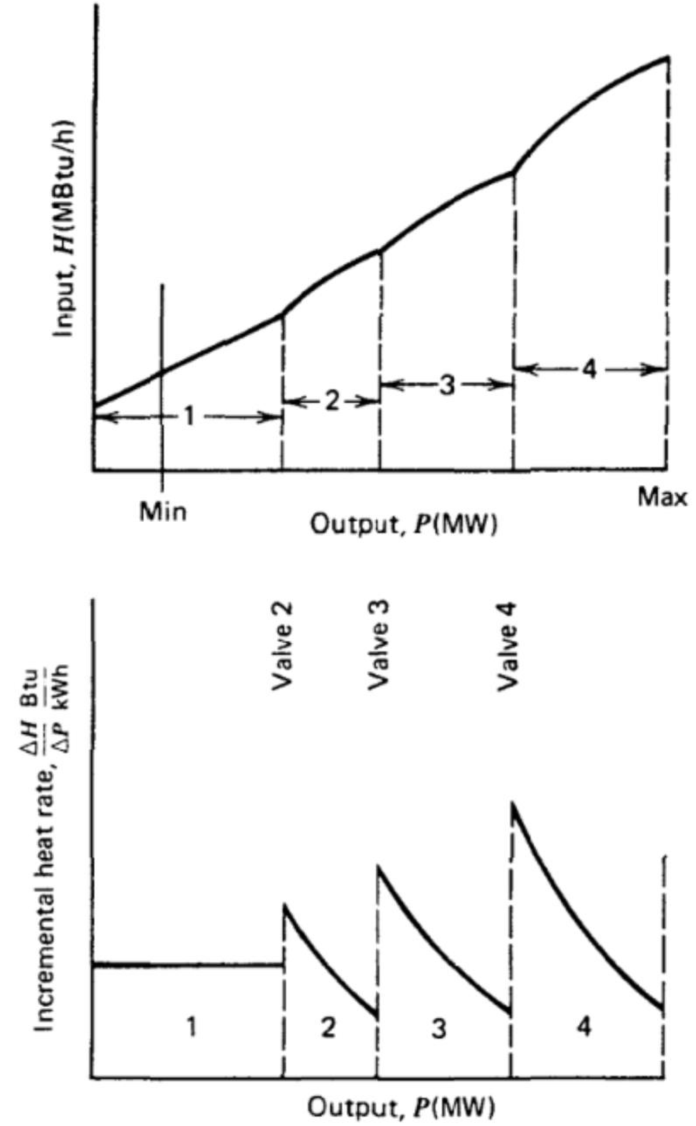
Early challenges (1940s-1970s)

Key problem classes

- Economic dispatch of generation systems and plant operations planning
- Energy transport optimization, e.g., pipelines, rail networks
- Early interest in storage systems and introduction of unit commitment
- LP, MIP applications, DP considerably exploited

Optimization limitations

- Scalability
- Heavy linearization and convexity assumptions
- Short horizon, determinism



Characteristics of a steam turbine generator with 4 steam admission valves
Source: A. Wood, B. Wollenberg, and G. Sheblé, Power Generation, Operation and Control. Wiley, 2013.

MIP progress and real-world feasibility
(1980s-1990s)

MIP transforming the electrical power industry

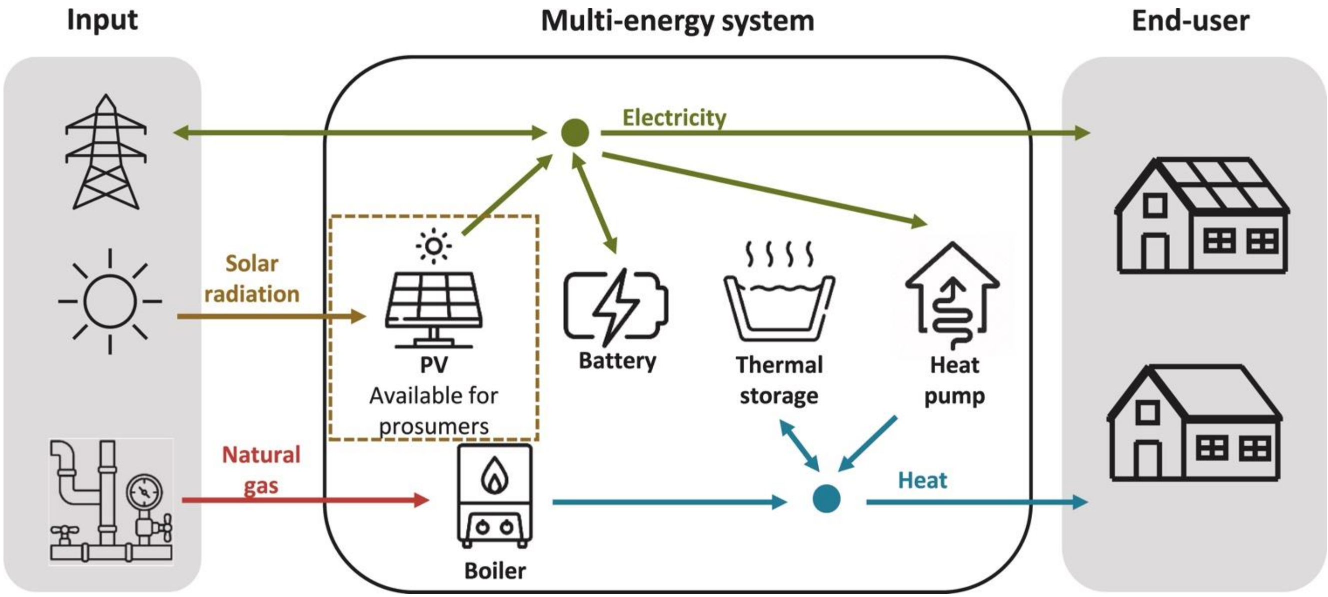
- EPRI report, June 1989:
  - *“Mixed-integer programming (MIP) is a powerful modeling tool…. ‘They are, however, theoretically complicated and computationally cumbersome’”*
  - 7-day unit commitment model could not be solved to optimality
- DIMACS meeting 1999:
  - Bob Bixby demonstrated that MIP had improved to the point where practical power models could be solved
  - *7-day unit commitment model solved to optimality in 22 minutes*
- Electrical power deregulated in the late 1990s
  - Need to create a market for electricity

… *Within a few years, nearly every grid operator in the world was using MIP to solve these models*

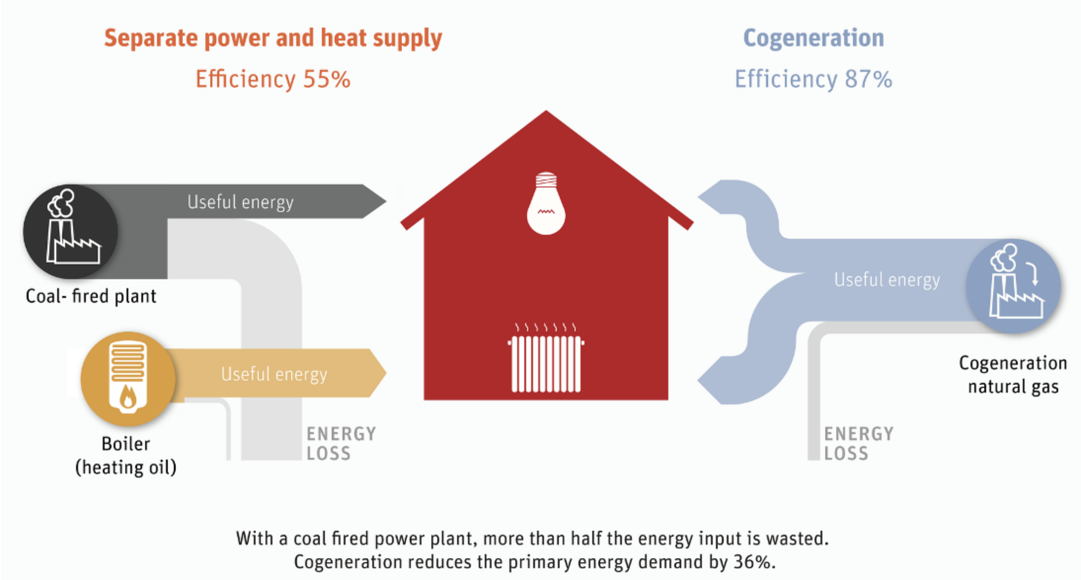
MIP progress and real-world feasibility
(1980s-1990s)

But …

- Co-generation and (implicit) sector coupling started gaining traction to reduce losses and facilitate the use of renewables
- Introduction of multi-energy systems
- More nonlinearities for coupling constraints
- Further expansion of problem sizes for integrated approaches, putting a strain on solvers



Source: Brodnicke, L., Gabrielli, P., & Sansavini, G. (2023). Impact of policies on residential multi-energy systems for consumers and prosumers. *Applied Energy*, *344*, 121276.



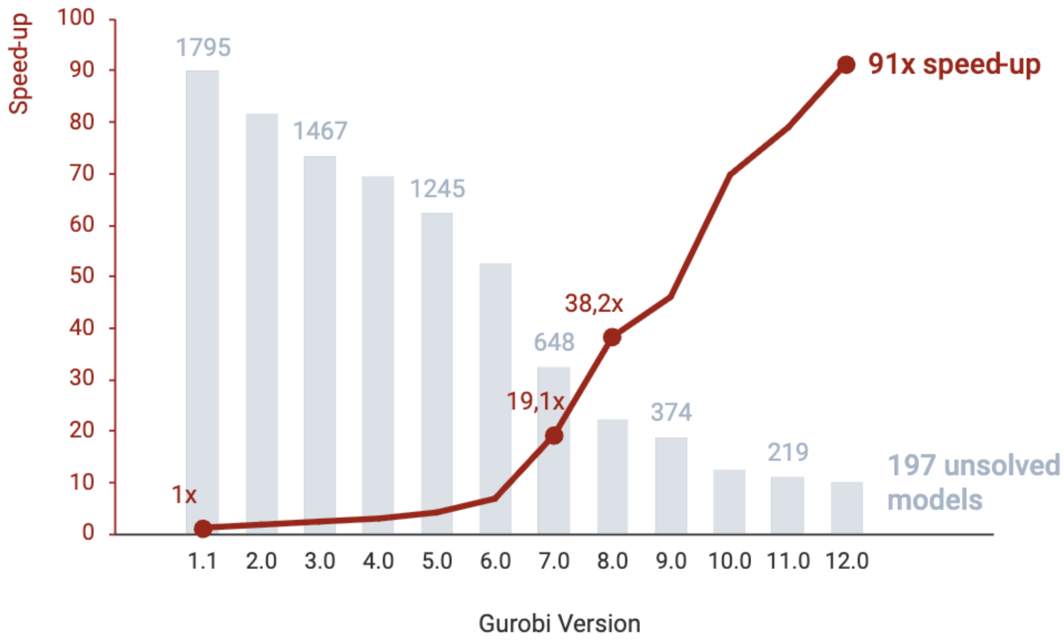
With a coal fired power plant, more than half the energy input is wasted.
Cogeneration reduces the primary energy demand by 36%.

Source: Wikipedia (wikipedia.org/wiki/Cogeneration)

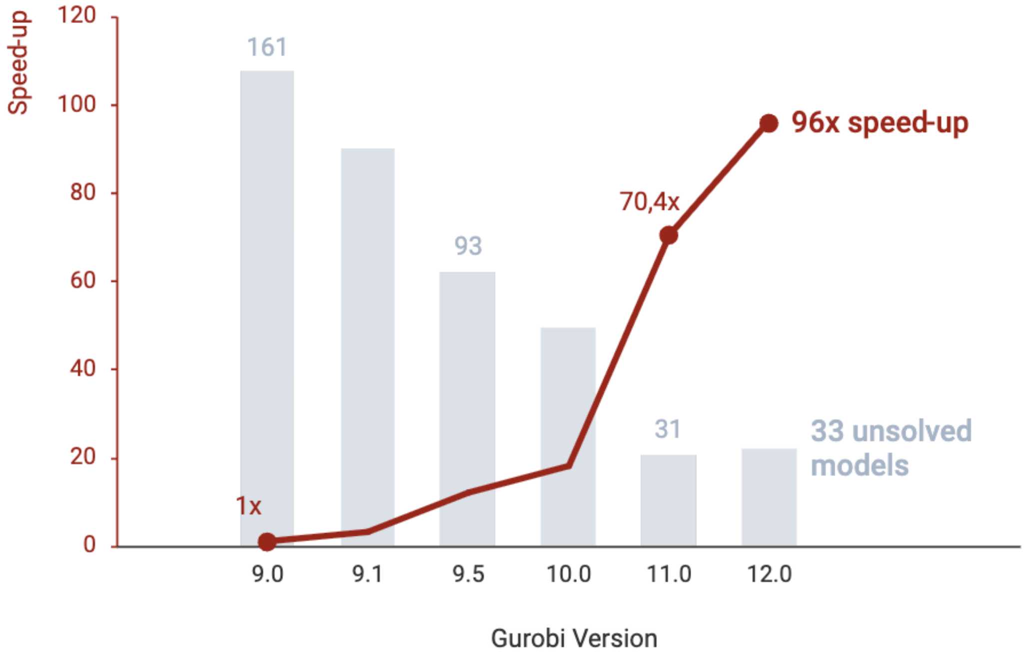
Scaling up and emerging system bottlenecks
2000s onwards

Technology advancements

- Breakthroughs in mathematical solving and hardware acceleration enabling large-scale, real-time models
- Wider adoption of special problem classes: e.g., Security-Constrained Unit Commitment (SCUC) and Security-Constrained Economic Dispatch (SCED)



Gurobi Performance Evolution on MILP: Speed and Solvability (PAR-10)



Gurobi Performance Evolution on Non-convex MIQCP: Speed and Solvability (PAR-10)

Scaling up and emerging system bottlenecks

2000s onwards

+1.22°C

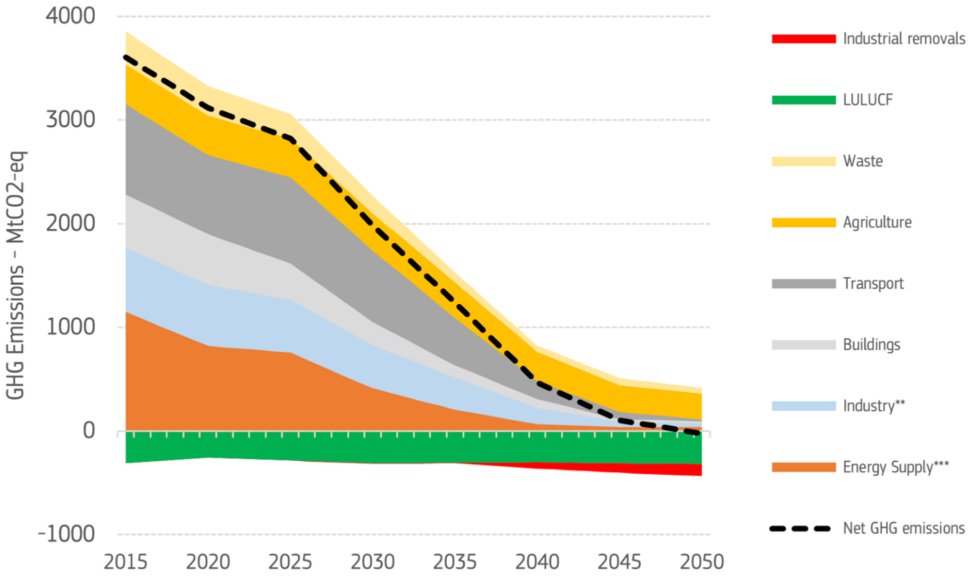
was the increase in global average temperature in 2013-2023

3

planets

would be needed by 2050, if we continue using resources as we do today

Greenhouse gas emissions in the period 2015-2050*



Sources:

Council of the European Union (www.consilium.europa.eu/en/policies/european-green-deal/)

European Commission (www.climate.ec.europa.eu/eu-action/climate-strategies-targets/2040-climate-target_en)

On the road to Energy Transition….

1. Large-scale, long-term investment planning
   - **Multi-decade** horizons
   - **Thousands** of variables and constraints per year
   - Decomposition and heuristic approaches often explored, e.g., rolling-horizon algorithms
2. Overcoming uncertainty
   - Acceleration in deploying renewables
   - Higher grid volatility and complexity (e.g., variability in supply/demand, reserve and ramping requirements etc..)
   - Market operators now rely on **multi-scenario simulations** and **probabilistic reliability metrics**
3. Decarbonization goals
   - Trade-offs between economic and environmental objectives
   - CO_2 caps, carbon pricing, emissions penalties embedded in models, triggering numerical instability

Relevant Gurobi features

Relevant Gurobi features

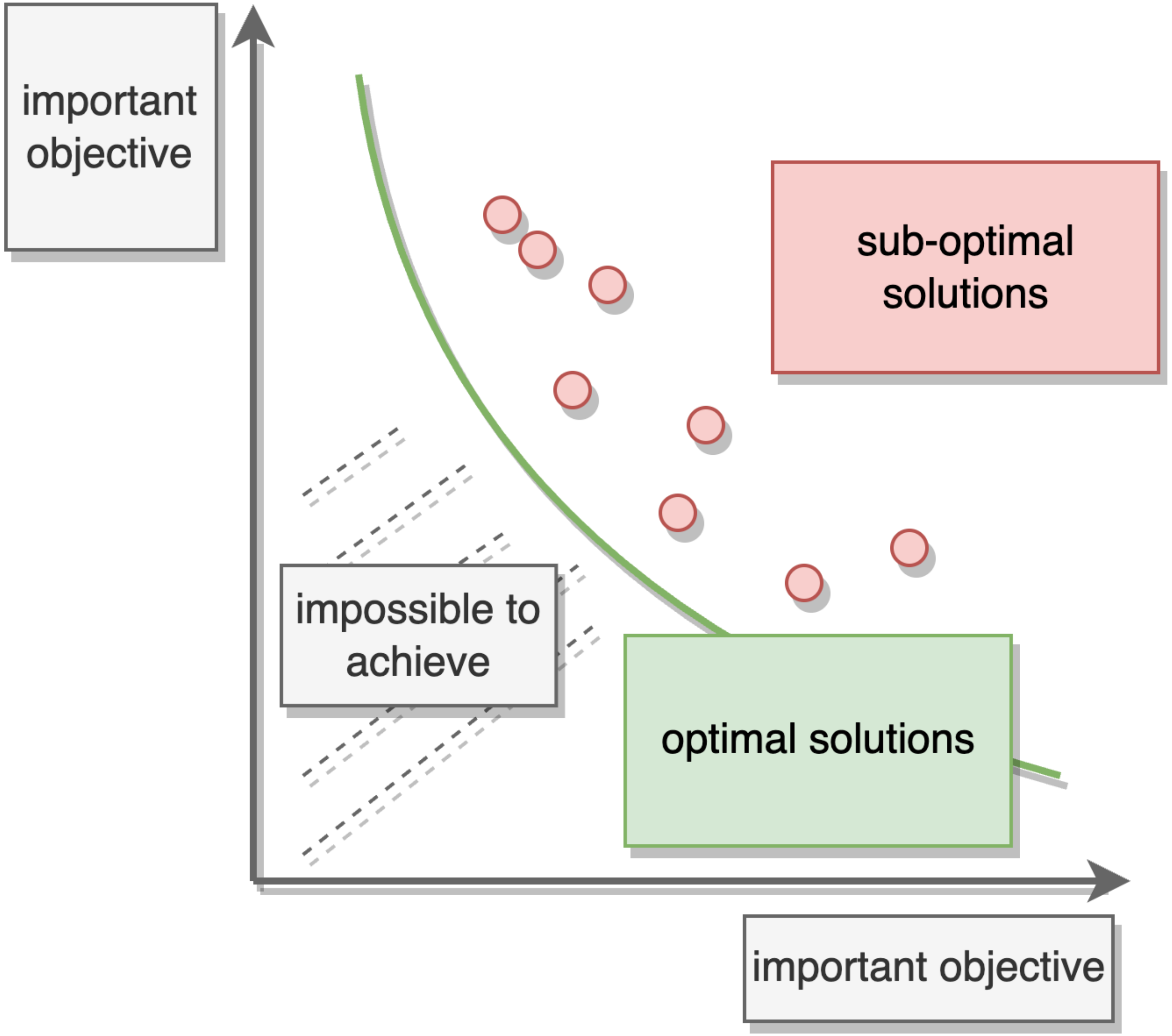
How can we better address energy optimization bottlenecks by leveraging Gurobi?

Modeling

Multiple Objectives

Competing goals

- Multiple Objectives
  - Minimal costs
  - Minimal CO_2 emissions
  - Maximum energy efficiency
- Gurobi Multiple Objectives
  - Define multiple objective terms
  - Define weights, priorities and tolerances
  - Warm start MIPs in hierarchical multi objective



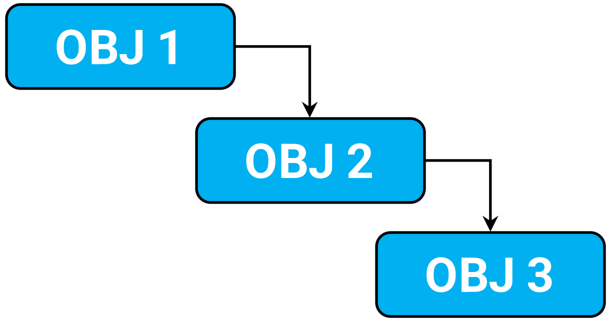
How does Gurobi handle the trade-offs?

- **Weighted:** Optimize a weighted combination of the individual objectives

OBJ 1 + **OBJ 2** + **OBJ 3**

$$\begin{aligned} \min \quad & w_1 f_1(x) + w_2 f_2(x) + w_3 f_3(x) \\ \text{s.t:} \quad & x \in \mathcal{C} \end{aligned}$$

- **Hierarchical (Lexicographical):** Optimize each objective in a priority order given while limiting the degradation of the higher-priority objectives



$$\begin{aligned} \min \quad & f_1(x) \\ \text{s.t:} \quad & x \in \mathcal{C} \end{aligned}$$

$$\begin{aligned} \min \quad & f_2(x) \\ \text{s.t:} \quad & x \in \mathcal{C} \\ & f_1(x) \leq \epsilon_1 \end{aligned}$$

$$\begin{aligned} \min \quad & f_3(x) \\ \text{s.t:} \quad & x \in \mathcal{C} \\ & f_1(x) \leq \epsilon_1 \\ & f_2(x) \leq \epsilon_2 \end{aligned}$$

- **Weighted + Hierarchical**

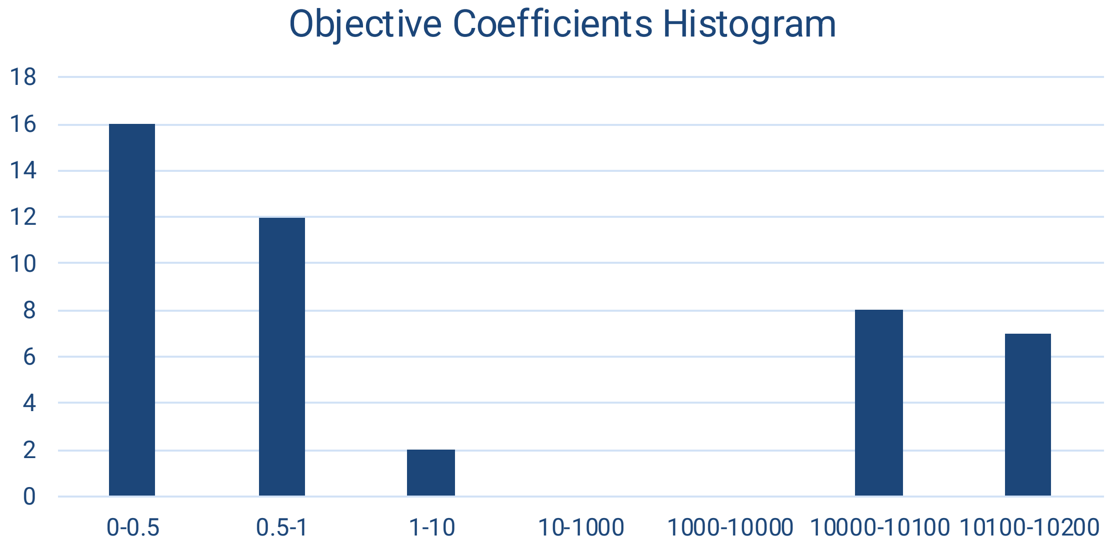
Multiple Objectives API

```
model.setObjectiveN(LinExpr, index, priority=0, weight=1,
                    abstol=1e-6, reltol=0, name="" )
```

- LinExpr: The objective expressions should be linear
- index: Index for new objective (Used to set different parameters/query the solution per objective)
- priority: Objectives' priority (ObjNPriority attribute)
- weight: Objectives' weight (ObjNWeight attribute)
- abstol: Absolute tolerance used in calculating the allowable degradation (ObjNAbsTol attribute)
- reltol: Relative tolerance used in calculating the allowable degradation (ObjNRelTol attribute)

Why use the Multiple Objectives API?

- Make the objective functions easy to understand and maintain
- Get faster performance with warm starts for hierarchical objectives
- Multiple objectives can help avoiding numerical issues with large objective coefficients
  - Soft constraints with large penalty variables
  - Combining monetary values with environmental assessment metrics



There are 45 coefficients in 2 distinct groups.
Is this a multi-objective case in hiding?

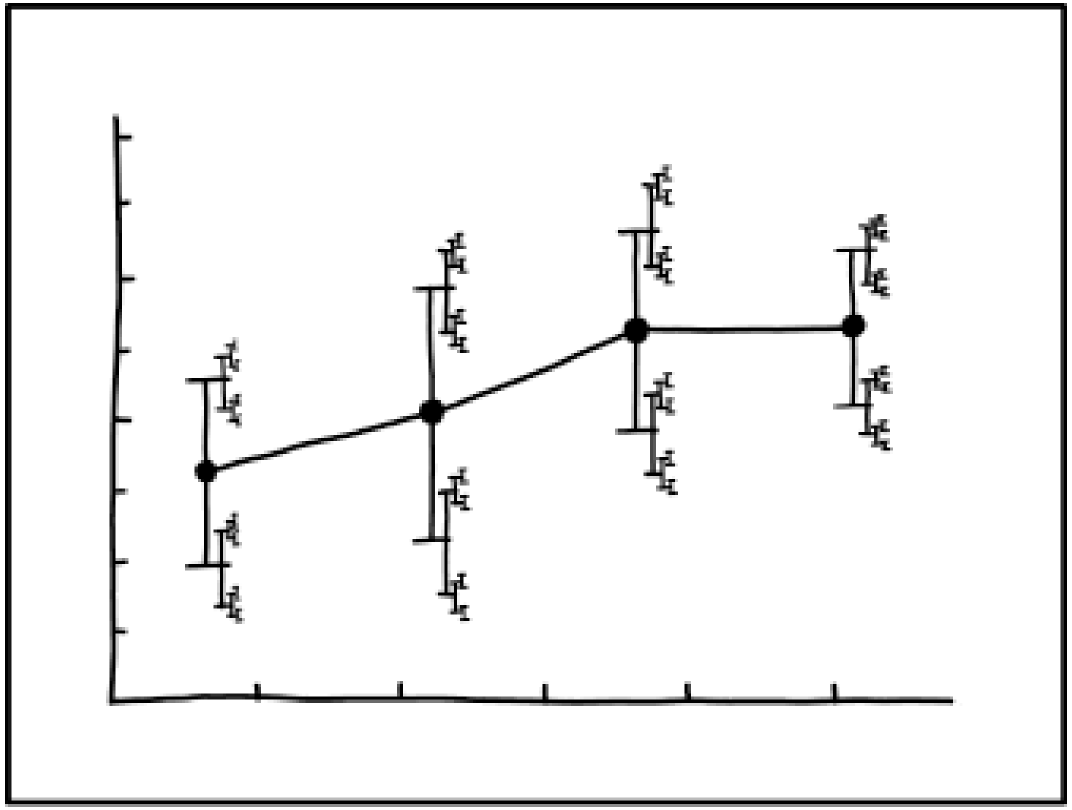
Modeling

Multiple Scenarios

Handling uncertainty

- Input data may be far into the future, based on rough estimates, or are known to fluctuate
- Math optimization offers some fundamental properties to express sensitivity
- Gurobi Multiple Scenarios
  - Define several scenarios for different input data
  - Solve all scenarios as one MIP
  - Retrieve and compare solutions

xkcd.com

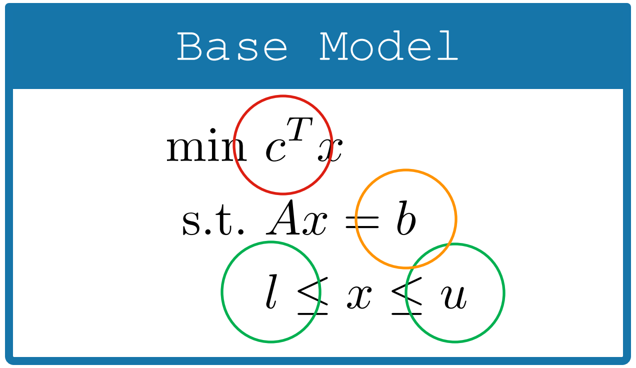


I DON'T KNOW HOW TO PROPAGATE ERROR CORRECTLY, SO I JUST PUT ERROR BARS ON ALL MY ERROR BARS.

Multiple Scenarios

- In many real-world situations, the following may occur:
  - The input data is not accurate
  - The input data is not known in advance and can take multiple values in real time
  - The input data is seasonal or periodic
  - The input data has a range of possible values

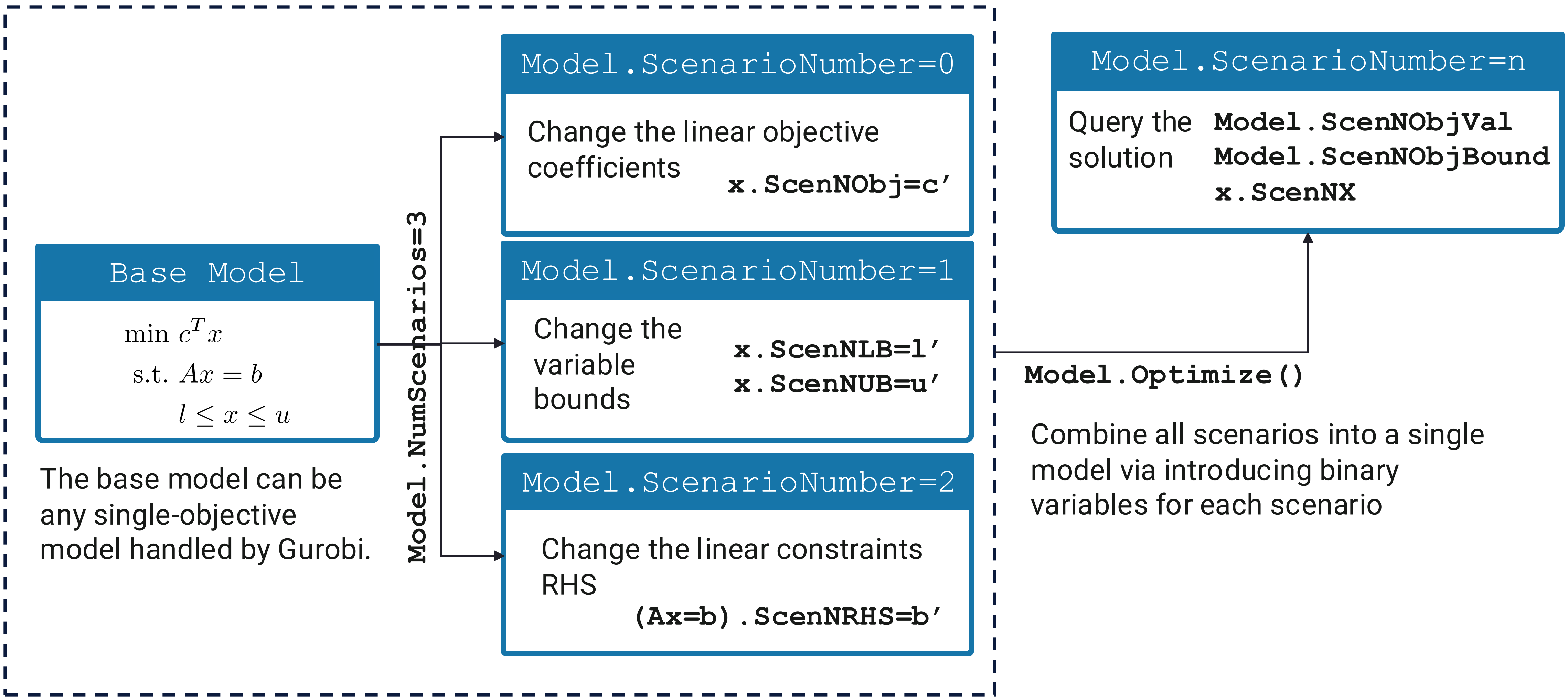
- The Gurobi Optimizer includes scenario analysis features which are useful to understand the sensitivity of the computed solution with respect to changes in the inputs:
  - Linear objective function coefficients
  - Variable lower and upper bounds
  - Linear constraint right-hand side values



Base Model

$$\begin{aligned} \min\ & c^T x \\ \text{s.t.}\ & Ax = b \\ & l \le x \le u \end{aligned}$$

Multiple Scenarios API



`Base Model`

$$\begin{aligned} \min\ & c^T x \\ \text{s.t.}\ & Ax = b \\ & l \le x \le u \end{aligned}$$

The base model can be any single-objective model handled by Gurobi.

`Model.NumScenarios=3`

`Model.ScenarioNumber=0`

Change the linear objective coefficients `x.ScenNObj=c′`

`Model.ScenarioNumber=1`

Change the variable bounds

`x.ScenNLB=l′`
`x.ScenNUB=u′`

`Model.ScenarioNumber=2`

Change the linear constraints RHS `(Ax=b).ScenNRHS=b′`

`Model.Optimize()`

Combine all scenarios into a single model via introducing binary variables for each scenario

`Model.ScenarioNumber=n`

Query the solution

`Model.ScenNObjVal`
`Model.ScenNObjBound`
`x.ScenNX`

Multiple Scenarios (Tips & Tricks)

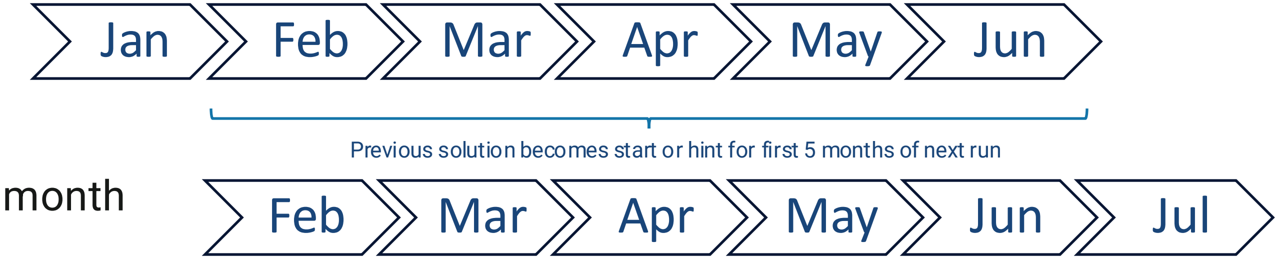
- The multiple scenarios API is restricted. For example, it is not possible to explicitly
  - Add/remove variables or constraints
  - Change the variable types
  - Change the sense of constraints
  - …
- However, we can circumvent some of the restrictions using useful tricks
  - To remove a variable, set its bounds to zero
  - To add a variable to a scenario, add it to the base model with bounds set to zero and then change the bounds accordingly
  - To remove a constraint, change its RHS values to GRB.INFINITY/-GRB.INFINITY
  - To add a constraint to a scenario or change its sense, add it as a pair of inequalities to the base model and change its RHS values accordingly

Performance

Variable Start & Hint Values

Variable Start & Hint Values

- Take advantage of previous solutions & model insight to improve performance
  - Knowledge of some variable values may be available from previous solves
  - Example: Rolling horizon planning application
    - Run 1: 6mo plan
    - Run 2:
      Redo plan starting in 2nd month



Previous solution becomes start or hint for first 5 months of next run

- Idea: Reduce solve times by specifying these values in the solver
  - There are 2 options for how to provide this information
    - **Start values**: to generate an initial solution. (Full or partial MIP starts can be used)
    - **Variable hints**: to influence the MIP search

Variable Start & Hint Values – Candidates

- Values from prior solves are most common

- Other candidates
    - Preferences: Use the most efficient resource
    - Heuristics: Apply use case insight
    - Penalties: Avoid an expensive penalty resource
    - Symmetry: Pick one value as a start
    - Only the objective changes
    - Only new variables are added

- Values are specific to the model

```
# Guess at the starting point: close the plant with
the highest fixed costs;
# open all others

# First open all plants
for p in plants:
    open[p].Start  = 1.0

# Now close the plant with the highest fixed cost
print('Initial guess:’)
maxFixed = max(fixedCosts)
for p in plants:
    if fixedCosts[p] == maxFixed:
        open[p].Start = 0.0
        print('Closing plant %s' % p)
        break
```

Variable Start & Hint Values – Comparison

Start Values

- **Generate initial integer solution**, which is improved via MIP search
- **Can specify partial solution**
- **Controlled** via `Start` variable attribute (or load a .mst MIP start file)
- **Supports multiple start values** via `NumStart` model attribute and `StartNumber` parameter

Variable Hints

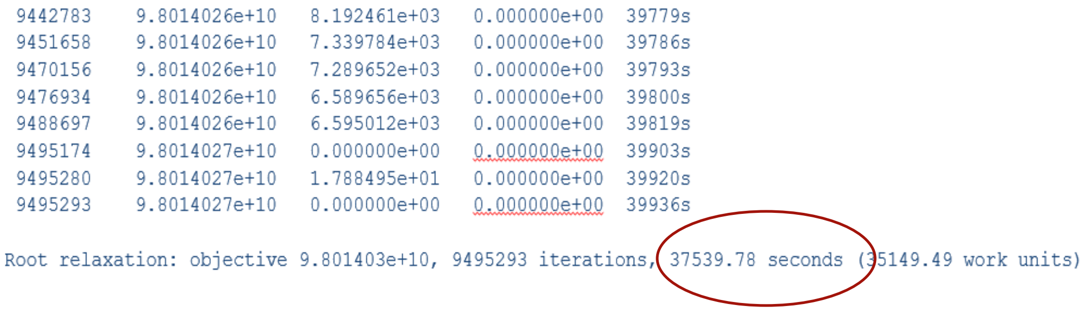
- **Guide MIP search** toward anticipated values
- **Can specify hints for subset of integer variables**
- **Controlled** via `VarHintVal` variable attribute, while **expressing your confidence for each hint** via `VarHintPri` variable attribute
- **Supports only one hint per variable**

Performance

No Relaxation Heuristic

The primal challenge

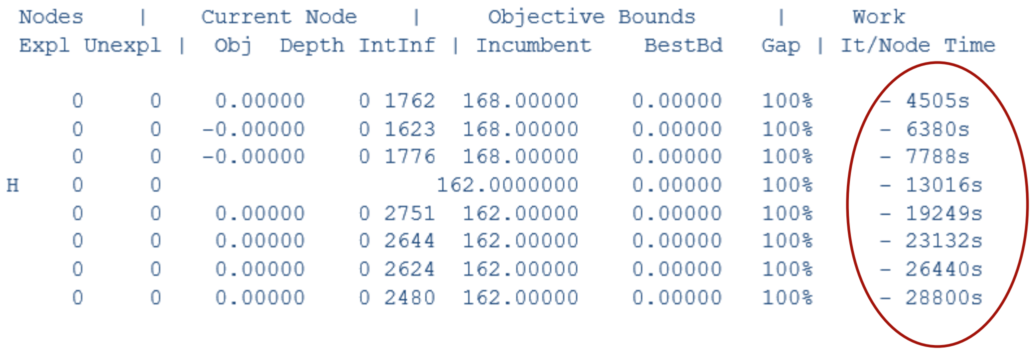
Slow root relaxation



```
9442783    9.8014026e+10   8.192461e+03   0.000000e+00  39779s
9451658    9.8014026e+10   7.339784e+03   0.000000e+00  39786s
9470156    9.8014026e+10   7.289652e+03   0.000000e+00  39793s
9476934    9.8014026e+10   6.589656e+03   0.000000e+00  39800s
9488697    9.8014026e+10   6.595012e+03   0.000000e+00  39819s
9495174    9.8014027e+10   0.000000e+00   0.000000e+00  39903s
9495280    9.8014027e+10   1.788495e+01   0.000000e+00  39920s
9495293    9.8014027e+10   0.000000e+00   0.000000e+00  39936s

Root relaxation: objective 9.801403e+10, 9495293 iterations, 37539.78 seconds (35149.49 work units)
```

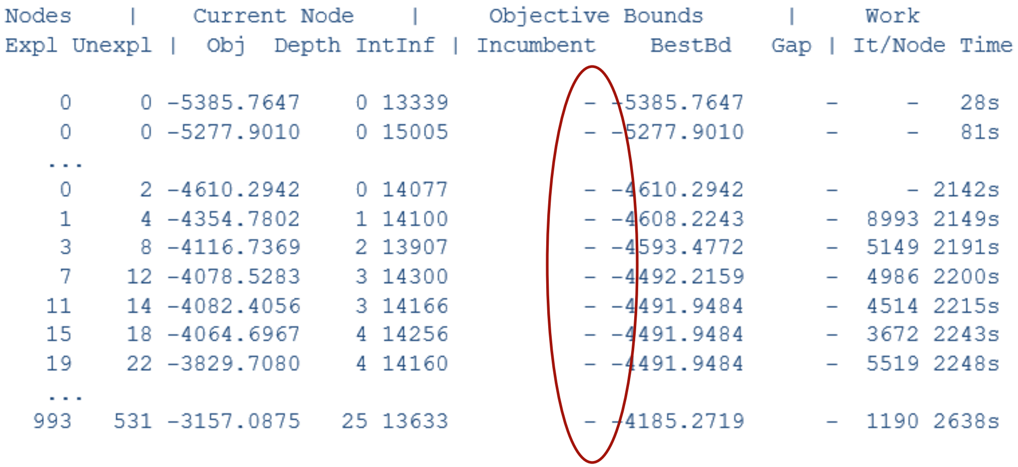
Slow node progress / Weak LP relaxation



```
    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    0.00000    0 1762  168.00000    0.00000   100%     -  4505s
     0     0   -0.00000    0 1623  168.00000    0.00000   100%     -  6380s
     0     0   -0.00000    0 1776  168.00000    0.00000   100%     -  7788s
H    0     0                      162.0000000    0.00000   100%     - 13016s
     0     0    0.00000    0 2751  162.00000    0.00000   100%     - 19249s
     0     0    0.00000    0 2644  162.00000    0.00000   100%     - 23132s
     0     0    0.00000    0 2624  162.00000    0.00000   100%     - 26440s
     0     0    0.00000    0 2480  162.00000    0.00000   100%     - 28800s
```

Slow/no feasible solutions found



```
    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0 -5385.7647    0 13339          - -5385.7647      -     -   28s
     0     0 -5277.9010    0 15005          - -5277.9010      -     -   81s
 ...
     0     2 -4610.2942    0 14077          - -4610.2942      -     - 2142s
     1     4 -4354.7802    1 14100          - -4608.2243      -  8993 2149s
     3     8 -4116.7369    2 13907          - -4593.4772      -  5149 2191s
     7    12 -4078.5283    3 14300          - -4492.2159      -  4986 2200s
    11    14 -4082.4056    3 14166          - -4491.9484      -  4514 2215s
    15    18 -4064.6967    4 14256          - -4491.9484      -  3672 2243s
    19    22 -3829.7080    4 14160          - -4491.9484      -  5519 2248s
 ...
   993   531 -3157.0875   25 13633          - -4185.2719      -  1190 2638s
```

No Relaxation Heuristic

(Meta)heuristics offer a different approach

- Metaheuristics can be very effective to find high-quality feasible solutions

Examples of metaheuristics

- Ant Colony Optimization
- Genetic Algorithms
- Very Large-Neighborhood Search
- Simulated Annealing

- In practice, implementing an inhouse (meta)heuristic can be very difficult to extend/maintain

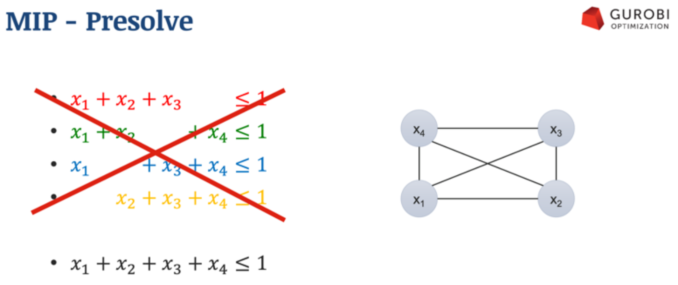
We developed our own!

✓ The NoRel heuristic works on a broad set of MIP models

✓ NoRel is massively using sub-MIP solves on smaller (partial) models that have easier LP relaxations than the main model

The traditional MIP solving process

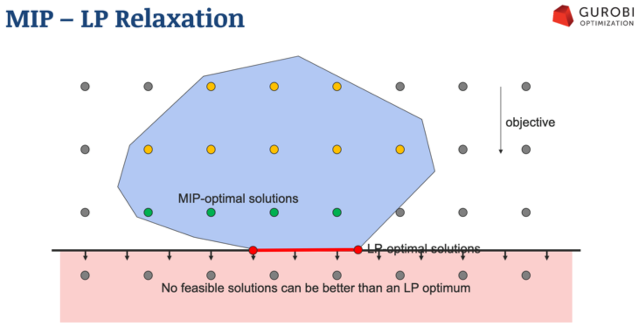
NoRel



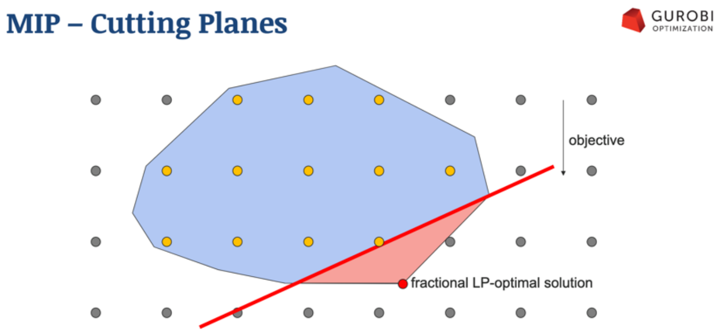
MIP - Presolve

- ~~$x_1 + x_2 + x_3 \le 1$~~
- ~~$x_1 + x_2 + x_4 \le 1$~~
- ~~$x_1 + x_3 + x_4 \le 1$~~
- ~~$x_2 + x_3 + x_4 \le 1$~~

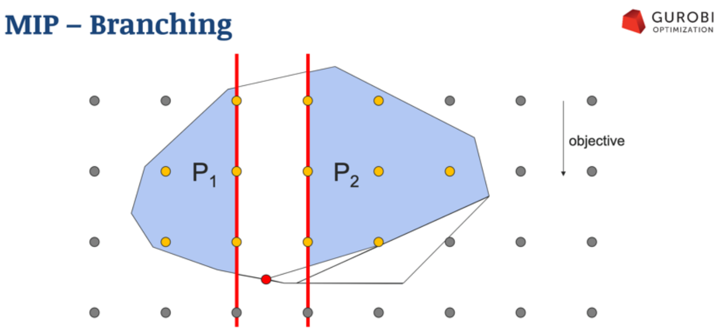
- $x_1 + x_2 + x_3 + x_4 \le 1$



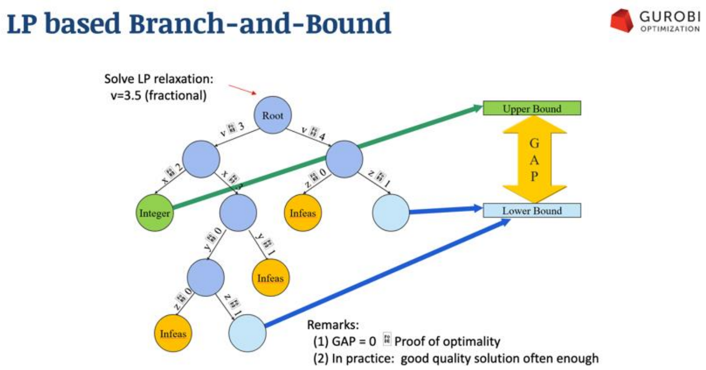
MIP – LP Relaxation



MIP – Cutting Planes



MIP – Branching



LP based Branch-and-Bound

Solve LP relaxation:
v=3.5 (fractional)

Remarks:
(1) GAP = 0 ▯ Proof of optimality
(2) In practice: good quality solution often enough

No Relaxation (NoRel) Heuristic

Not enabled by default!

- Controlled by two parameters:
  - **NoRelHeurTime** – limit time spent on NoRel Heuristic
  - **NoRelHeurWork** – limit deterministic work of NoRel Heuristic

  Coming soon in Gurobi 13.0:
  - **NoRelHeurSolutions** – to instruct the NoRel Heuristic to stop when it has found a specific number of solutions
- **Phase 1 – Find a feasible solution**
  - Keep track of infeasible solutions
  - Minimize the number of constraints that are violated

- **Phase 2 – Improve solution pool**
  - Keep track of feasible solution pool
  - Local search in neighborhoods around the solution

Log Example

```
Optimize a model with 1874523 rows, 548926 columns and 11351663 nonzeros
...
Presolve removed 1328056 rows and 395707 columns (presolve time = 5s) ...
Presolve removed 1392822 rows and 396820 columns (presolve time = 10s) ...
...
Presolve removed 1416115 rows and 400727 columns
Presolve time: 88.30s
Presolved: 458408 rows, 148199 columns, 4130523 nonzeros
Variable types: 68874 continuous, 79325 integer (79243 binary)
Starting NoRel heuristic
Found phase-1 solution: relaxation 165814
Found phase-1 solution: relaxation 107491
Found phase-1 solution: relaxation 18536.5
Elapsed time for NoRel heuristic: 9s (best bound -0.507812)
Found phase-1 solution: relaxation 1019.14
Found phase-1 solution: relaxation 70
Found phase-1 solution: relaxation 67
Found phase-1 solution: relaxation 66
Found phase-1 solution: relaxation 54
Found phase-1 solution: relaxation 50
Found phase-1 solution: relaxation 40
Found phase-1 solution: relaxation 39
Elapsed time for NoRel heuristic: 16s (best bound -0.507812)
Found phase-1 solution: relaxation 25
Found phase-1 solution: relaxation 14
Found phase-1 solution: relaxation 10
Elapsed time for NoRel heuristic: 23s (best bound -0.507812)
Found phase-1 solution: relaxation 0
Transition to phase 2
Elapsed time for NoRel heuristic: 29s (best bound -0.507812)
Elapsed time for NoRel heuristic: 35s (best bound -0.507812)
Found heuristic solution: objective 1.282771e+12
Elapsed time for NoRel heuristic: 40s (best bound -0.507812)
Elapsed time for NoRel heuristic: 50s (best bound -0.507812)
Found heuristic solution: objective 1.282771e+12
...

NoRel heuristic complete
```

Strengths / Limitations of NoRel

- ✓ Initially motivated by models where the LP Relaxation is too slow
- ✓ Can find and rapidly improve feasible solutions (sometimes >10x faster in finding an initial feasible solution)
- ✓ Great for models with slow node throughput
- ✓ Parallelizes very nicely so good for large core machines

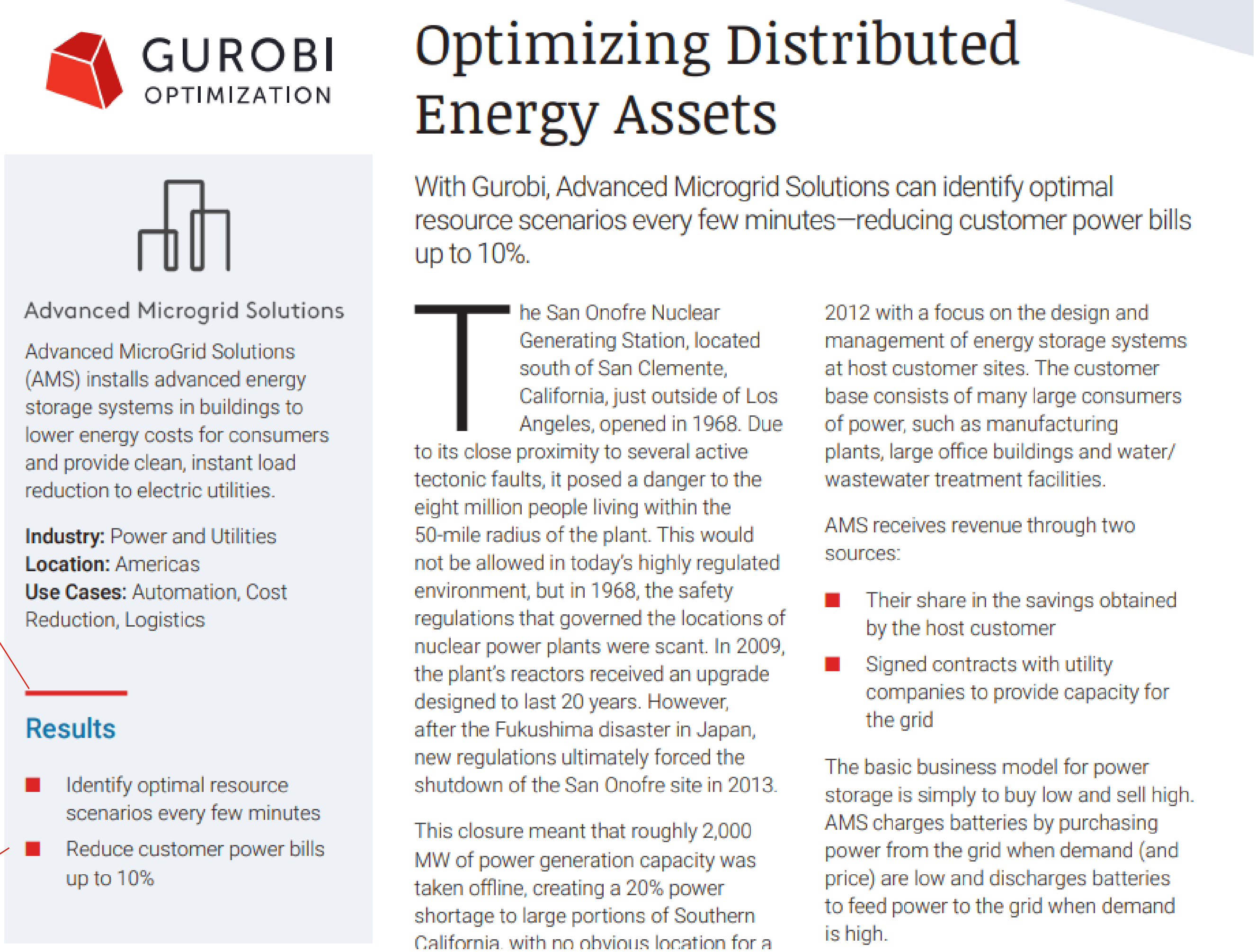
- Only a very weak bound is calculated
- It can be time consuming: sometimes it struggles to create an initial solution or improve upon the current best one

Success story with Gurobi

At Gurobi, the energy sector isn't just a use case—it's a priority!

Results

- Identify optimal resource scenarios every few minutes
- Reduce customer power bills up to 10%



Advanced Microgrid Solutions

Advanced MicroGrid Solutions (AMS) installs advanced energy storage systems in buildings to lower energy costs for consumers and provide clean, instant load reduction to electric utilities.

Industry: Power and Utilities
Location: Americas
Use Cases: Automation, Cost Reduction, Logistics

Results

- Identify optimal resource scenarios every few minutes
- Reduce customer power bills up to 10%

Optimizing Distributed Energy Assets

With Gurobi, Advanced Microgrid Solutions can identify optimal resource scenarios every few minutes—reducing customer power bills up to 10%.

The San Onofre Nuclear Generating Station, located south of San Clemente, California, just outside of Los Angeles, opened in 1968. Due to its close proximity to several active tectonic faults, it posed a danger to the eight million people living within the 50-mile radius of the plant. This would not be allowed in today's highly regulated environment, but in 1968, the safety regulations that governed the locations of nuclear power plants were scant. In 2009, the plant's reactors received an upgrade designed to last 20 years. However, after the Fukushima disaster in Japan, new regulations ultimately forced the shutdown of the San Onofre site in 2013.

This closure meant that roughly 2,000 MW of power generation capacity was taken offline, creating a 20% power shortage to large portions of Southern California, with no obvious location for a

2012 with a focus on the design and management of energy storage systems at host customer sites. The customer base consists of many large consumers of power, such as manufacturing plants, large office buildings and water/wastewater treatment facilities.

AMS receives revenue through two sources:

- Their share in the savings obtained by the host customer
- Signed contracts with utility companies to provide capacity for the grid

The basic business model for power storage is simply to buy low and sell high. AMS charges batteries by purchasing power from the grid when demand (and price) are low and discharges batteries to feed power to the grid when demand is high.

Energy Innovation Summit
Modeling Meets Optimization

Questions?!

Christine Tawfik
christine.tawfik@gurobi.com