Stochastic Look Ahead Commitment

A Computational Perspective

Presented By: Bernard Knueven, National Renewable Energy Laboratory

2021 IEEE Power & Energy Society General Meeting, July 26, 2021

This work was authored in part by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by U.S. Department of Energy, Advanced Research Projects Agency - Energy. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

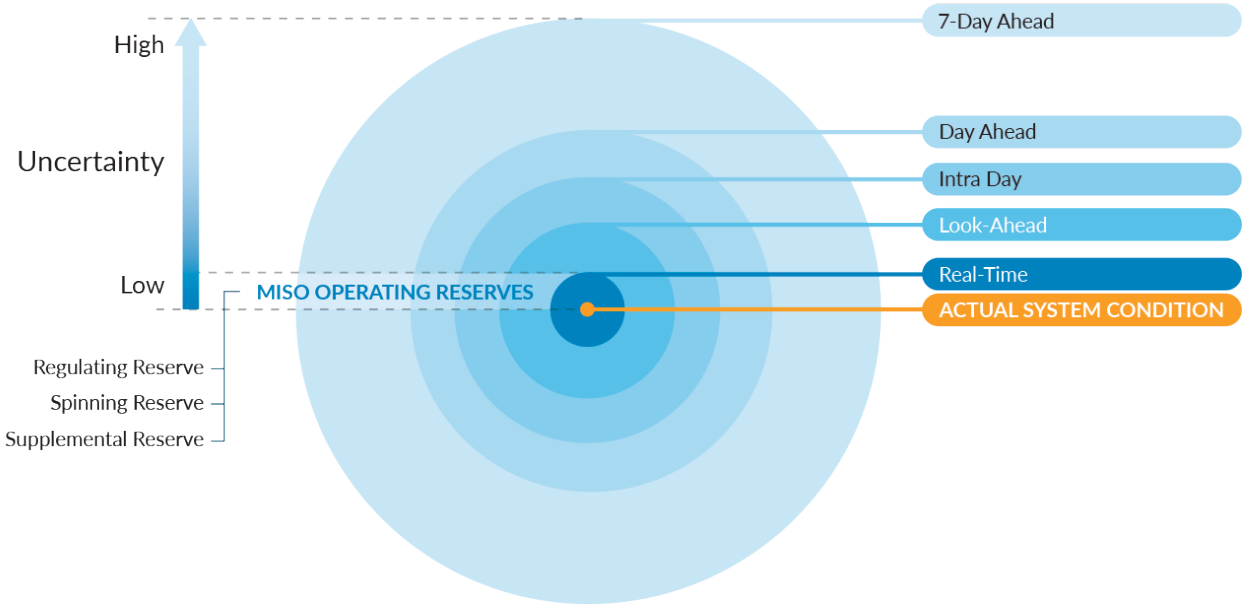
NREL/PR-2C00-80308

Acknowledgements

- This work was funded by ARPA-E to develop a stochastic look-ahead commitment (SLAC) advisory tool
- Tool would inform system operators managing existing and future grid resources and associated uncertainty

Partners & Team

- MISO
  - Yonghong Chen, Mohammad Faqiry, Anupam Thatte, Long Zhao, Shengfei Yin
- Nexant Inc
  - Roger Treinen, Herminio Pinto, Narsi Vempati
- Arizona State University
  - Junshan Zhang, Vijay Vittal, Trevor Werho, Kory Hedman
- Sandia National Labs
  - Manuel Garcia, Jean-Paul Watson



Uncertainty Management at MISO

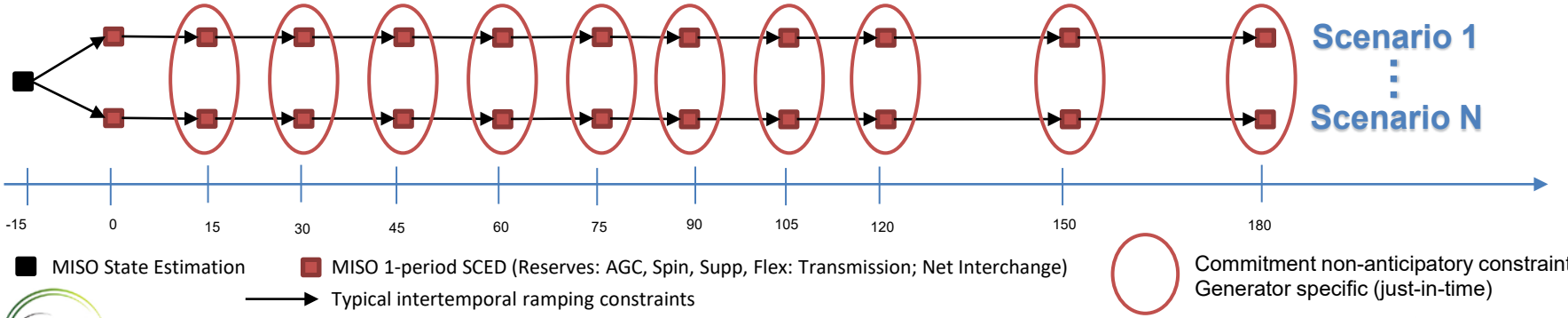
Look Ahead Commitment (LAC) commits intra-day fast-start units every 15 minutes with a 3-hour look-ahead and 15- to 30-minute intervals

Problem Statement

- Formulate and solve a Stochastic Look Ahead Commitment (SLAC) problem analogous to existing deterministic LAC
- Commit units in real-time with 3-hour look ahead under uncertainty
  - Find a set of commitments consistent with different possible evolutions of the grid while minimizing cost in expectation

SLAC Formulation

- Non-anticipatory constraints only apply to generators and time periods we need to decide about “now”
  - (S)LAC is solved every 15 minutes, so some final commitment decisions can be delayed
  - First-stage: immediate commitment decisions; Second-stage: everything else
- This formulation gives a realistic accounting of system flexibility in future time periods, while allowing for decisions on slower-start units to be made now

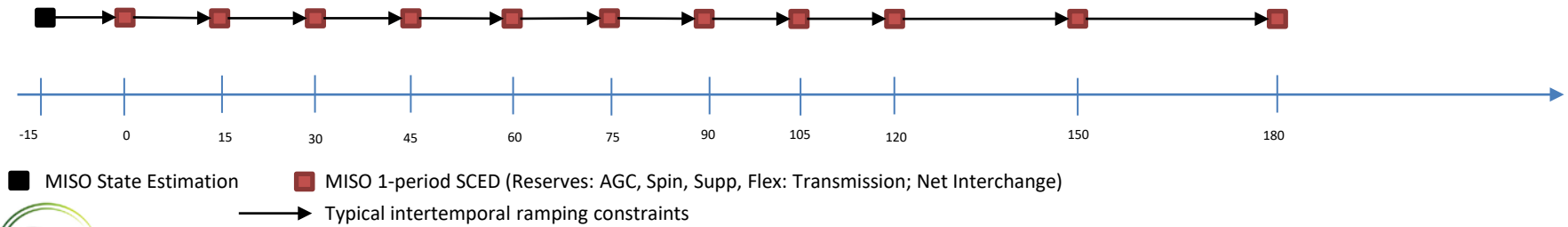


Scenario Generation

- Developed at Arizona State University (ASU)
- Three sources of uncertainty (wind, load, and NSI) are considered together
- Probabilistic forecasts are created for future time horizon
- The joint distribution between all elements and forecast horizons is constructed using copula theory – to capture spatial-temporal correlations
- Scenarios are sampled in a Monte Carlo fashion
- The algorithm is extremely efficient, can produce 1000 scenarios for MISO on the order of 10 seconds
- To obtain an adequate understanding of the future, 200 scenarios are sampled from the model
- The scenarios are then down-sampled to 40 scenarios using Backwards Reduction (distance-based reduction method)

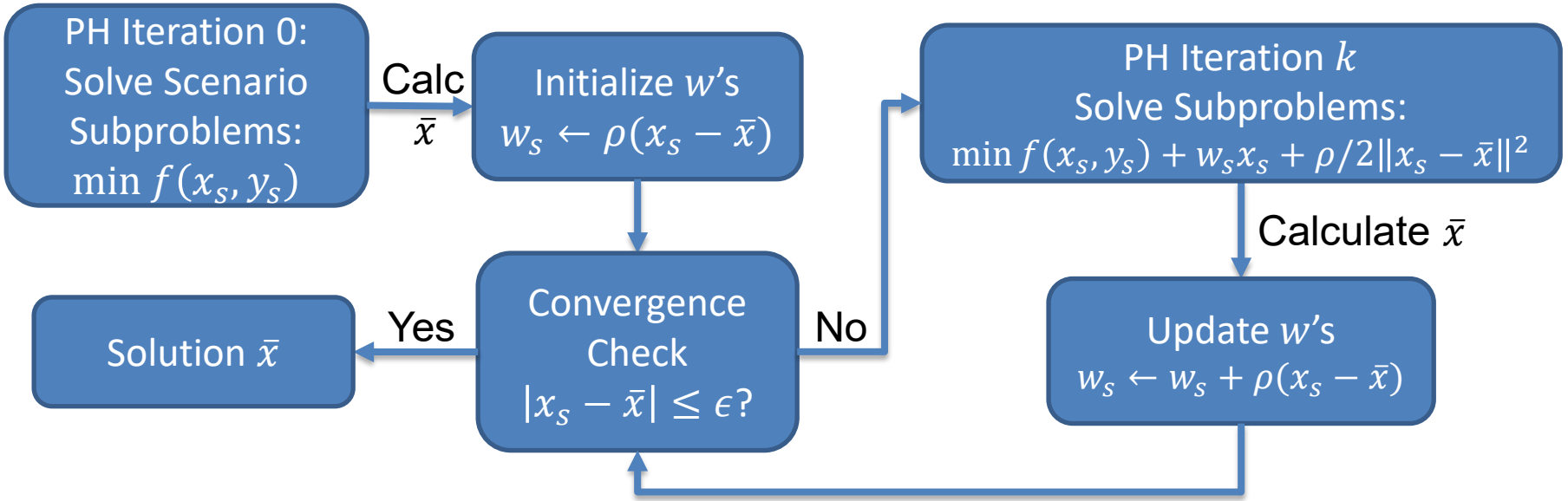
Deterministic Equivalent Validation

- The team customized, benchmarked, and verified MISO-customized Pyomo models (based on EGRET) against existing LAC results with MISO data
- This deterministic LAC model is the deterministic-equivalent basis for the SLAC model



Solution Methodology: Key Ingredients

- Customized Progressive Hedging Algorithm (PH)
  - Implemented using *mpi-sppy*, which automates stochastic formulation from deterministic equivalent; allows for deep customization and composition of decomposition algorithms
  - Generator-specific values for *rho*
    - dual-update step & regularization
  - Cleanup / finalization heuristic
    - No need to take PH to convergence
  - MIP solver tuning
  - Scenario-bundled subproblems
    - Stronger dual bounds; fewer iterations for convergence

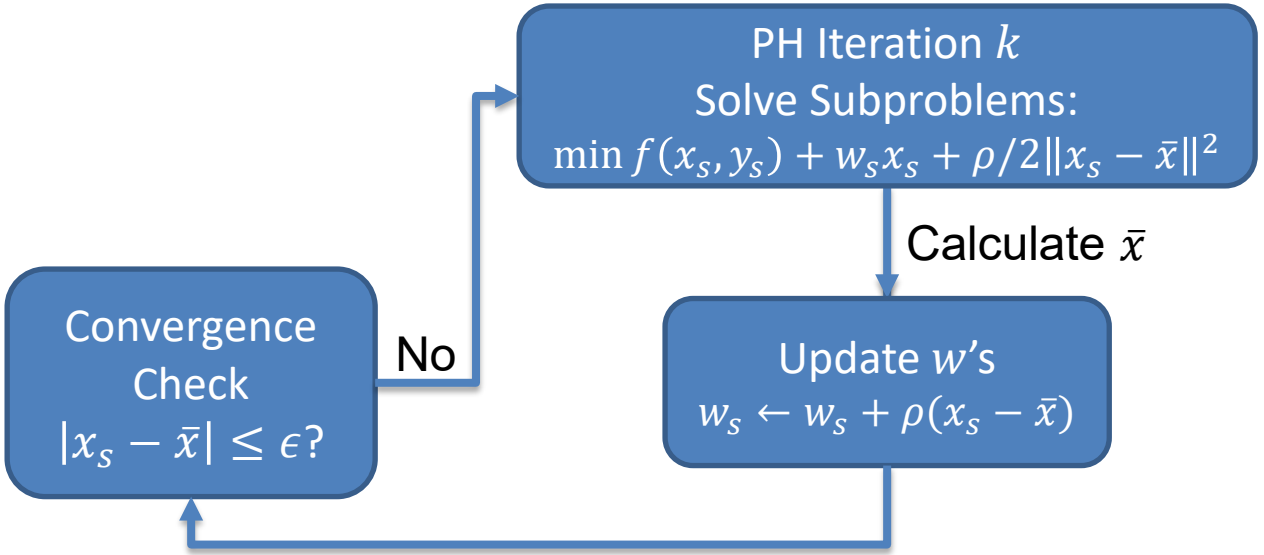


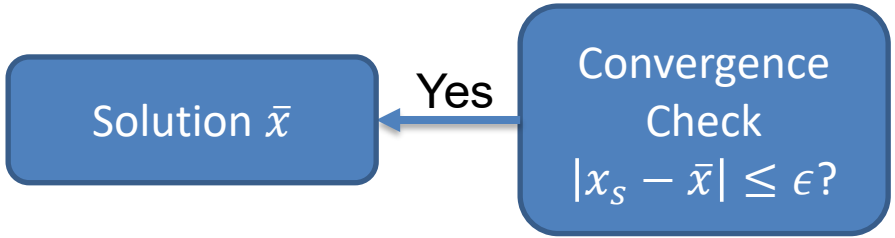
Progressive Hedging Algorithm

- Embarrassingly parallel – need communication to calculate $\bar{x}$ and check convergence, both can be easily and efficiently done via MPI reductions
- Not guaranteed to converge for non-convex problems, but can be leveraged as a heuristic
- Lagrangian dual bounds can be calculated using w_s: $\min f(x_s, y_s) + w_s x_s$
  - Trivial dual bound is calculated "for free" at iteration 0

Generator-specific ρ

- The chosen value(s) of ρ drive the entire PH algorithm, serving as both regularization penalty and dual update weight
- While classical PH uses a single value for ρ, in general ρ can be set on a per-variable basis
- For SLAC, we chose ρ_g to be proportional to generator g's production cost, which has proven effective on academic test cases





Cleanup and Finalization

- In practice, PH can often not be run to convergence
  - Time limitations
  - Non-convex problems have no convergence guarantees
  - Problems with first-stage integer variables are very prone to cycling
- Using PH in practice means setting an iteration limit, and creating a solution out of the result
  - For problems with relatively complete recourse, we can pick an arbitrary subproblem's solution
- For SLAC, we created a heuristic for reliability
  - If any subproblem, at termination which wants to commit (or not decommit) a generator, and the decision cannot be delayed, we commit that unit
  - Sometimes subproblems are hold-outs due to potential transmission or reserve violations

PH Iteration 0:
Solve Scenario Subproblems:
$\min f(x_s, y_s)$

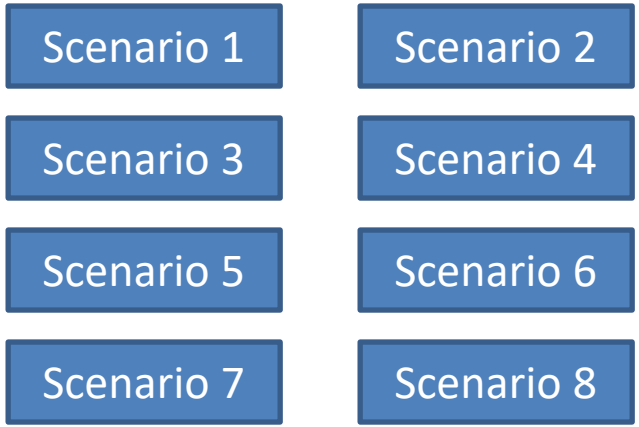
MIP Solver Tuning

PH Iteration k
Solve Subproblems:
$\min f(x_s, y_s) + w_s x_s + \rho/2 \|x_s - \bar{x}\|^2$

- Most of the computational time in PH is spend solving subproblems or waiting for other subproblems to solve
- For SLAC we turned CPLEX as follows:
  - Leverage Pyomo "persistent" solvers to maintain solver state
  - Tight optimality gap (<0.01%)
  - 30 second time limit; single-thread limit (parallel instances)
  - For iteration 0, set "MIP emphasis" to focus on bound
    - used to calculate a good "trivial" lower bound
  - For all other iterations, set "MIP emphasis" to focus on finding feasible solutions
    - Keeps PH progressing

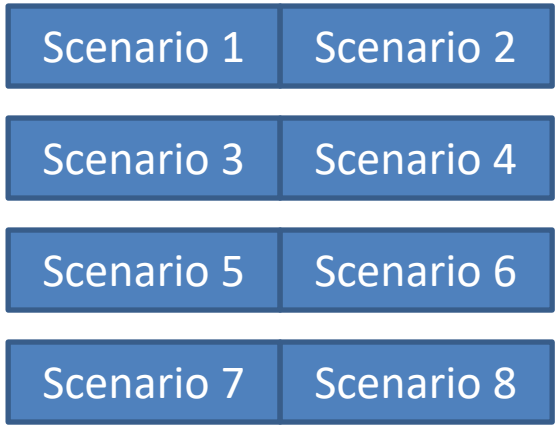
Subproblem Bundles

Classical Progressive Hedging



- Each scenario is its own subproblem
- Maximum parallelization
  - up to one thread per scenario

Progressive Hedging w/ Bundles



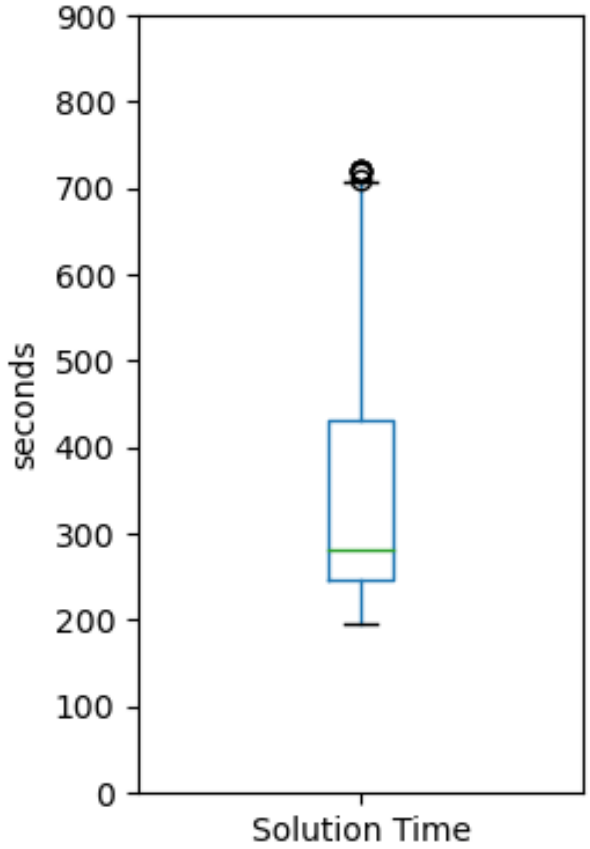
- Each subproblem has multiple scenarios
- Better LD bounds & convergence
  - Individual scenarios tend to “over optimize” for their particular realization

Computational Performance Evaluation

- To assess the impact of SLAC on grid operations, rolling horizon studies were conducted on historical days selected by MISO
  - Some days were “normal operations” and others were “conservative operations” to evaluate SLAC’s potential economic and reliability impact
- As a result, we solved over >1,400 SLAC problems, giving a broad computational perspective
- Hardware: 32-VCPU Linux VM with 256GB RAM
  - 20 threads concurrently, 1 per subproblem
- All times are wall-clock times

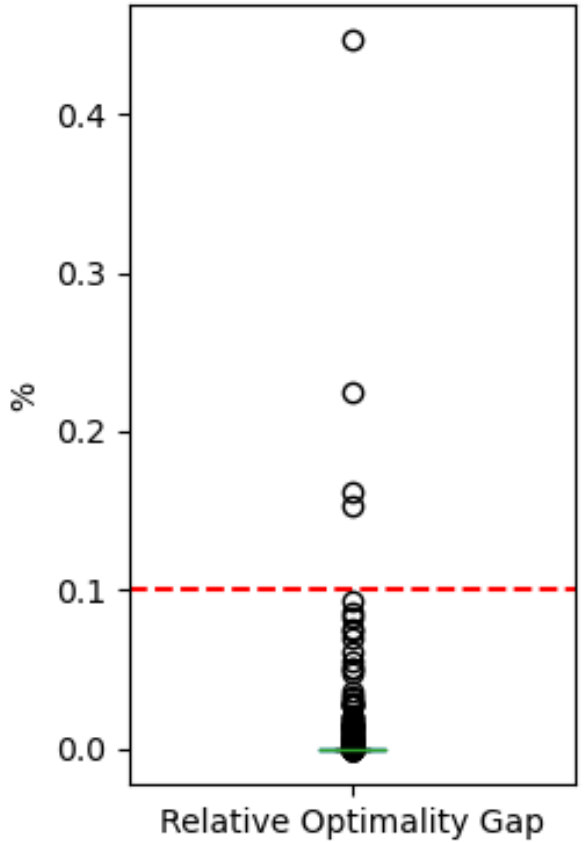
SLAC Solution Times

- All wall-clock times are comfortably within 900 seconds
  - In practice LAC is solved every 15 minutes
- Includes reading data from disk, setting up Pyomo models, computing objective value from heuristic, and writing full scenario solutions to disk
  - Pyomo models could be more highly optimized for build-speed
  - Full scenario solution and objective value are not strictly necessary to implement the SLAC solution
  - Further returns to parallelism would be expected with more compute resources



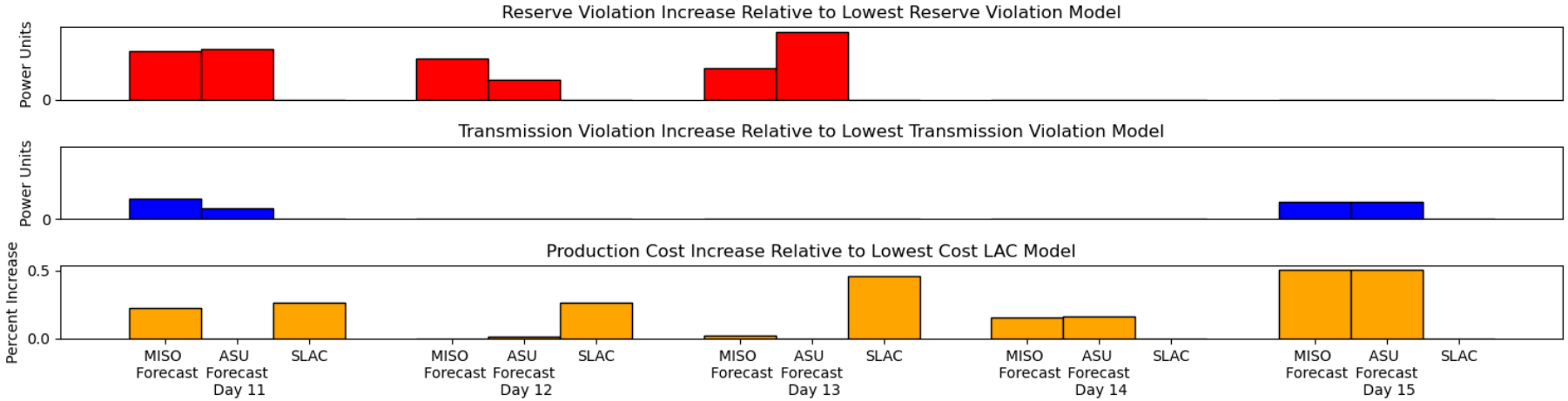
SLAC Solution Quality

- All but 4 of the >1,400 models solved fall within a 0.1% relative optimality gap requirement
- Most (>96%) meet a 0.01% relative optimality gap requirement
- No effort is expended in improving the lower bound given by the initial PH iteration



Intra-Day Rolling LAC Simulation Platform

- The team evaluated Stochastic LAC on a rolling horizon simulation for several test days selected by MISO
- At every 15-minute interval for the day, we solve a variant of the Look-Ahead Commitment (with 3-hour window) and a real-time SCED (with no look ahead)
  - Variants: SLAC; LAC with MISO-Forecast; LAC with ASU-Forecast
- The Production Cost, Reserve Violations, and Transmission Violations for a day are reported by the sum of the SCED values over the day



Days with Production Cost Differences (>0.2%)

- SLAC has significantly increased production costs for three of these dates
  - Day 11, Day 12, and Day 13 see significantly improved reserve violations
  - Day 11 also sees improved transmission violations
- SLAC has significantly decreased production costs for two of these dates
  - Day 14 and Day 15 see 0.25-0.5% production cost savings

Rolling-Horizon Study Summary

- SLAC showed a *reliability benefit* over both MISO- and ASU-Forecast LAC, on most study days, decreasing to eliminating reserve or transmission violation, or both.
- SLAC demonstrated an *economic benefit* over both MISO- and ASU-Forecast LAC on days 14 and 15, decreasing costs 0.25 – 0.5%
- On a few study days, SLAC has similar performance to MISO- and ASU-Forecast LAC (no reliability issues these days)
- MISO-Forecast LAC and ASU-Forecast LAC seem to perform similarly
  - Hypothesis: Not much more to gain from improved/different point forecasting methods

Conclusion

- Stochastic Optimization is a powerful tool for managing uncertainty in grid operations, demonstrating potential economic and reliability improvements over existing practice.
- Decomposition techniques, such as PH, can be customized to be reliable for specific applications. This allows operators to consider many potential future operational scenarios simultaneously; efficiently leveraging increasingly parallel computing resources.