Exceptional service in the national interest

Scalable, Parallel Stochastic Unit Commitment for Improved Day-Ahead and Reliability Operations

FERC Technical Conference: Increasing Real-Time and Day-Ahead Market Efficiency Through Improved Software

June 26, 2012

Project Team

- Sandia National Laboratories
  - Ross Guttromson, MS, MBA, PE
  - Jean-Paul Watson, PhD
  - Cesar Silva Monroy, PhD
- Iowa State University
  - Sarah Ryan, PhD
  - Leigh Tesfatsion, PhD
  - Dionysios Aliprantis, PhD
- Alstom Grid
  - Kwok Cheung, PhD
- UC Davis
  - Roger Wets, PhD
  - David Woodruff, PhD
- ISO New England
  - Eugene Litvinov, PhD

External Advisors

- Eugene Litvinov, ISO-NE
  - Chairs Advisory Team
- Richard O'Neill, FERC
- Ralph Masiello, KEMA
- David Morton, UT Austin

Sandia National Laboratories

Project Goals

- Execute _stochastic_ unit commitment (UC) ***at scale*, *on real-world data sets***
  - Stochastic UC state-of-the-art is very limited (tens to low hundreds of units)
  - Our solution must ultimately be useable by an ISO
- Produce solutions ***in tractable run-times***, ***with error bounds***
  - Parallel scenario-based decomposition
    - For both upper and lower bounding (Progressive Hedging and Dual Decomp.)
  - Quantification of uncertainty
    - Rigorous confidence intervals on solution cost
- Employ high-accuracy stochastic process models
  - Leveraged to achieve computational tractability while maintaining solution quality and robustness
- Demonstrate ***cost savings on an ISO-scale system*** at ***high renewables penetration levels***

Day-Ahead Unit Commitment (SCUC D-8h)

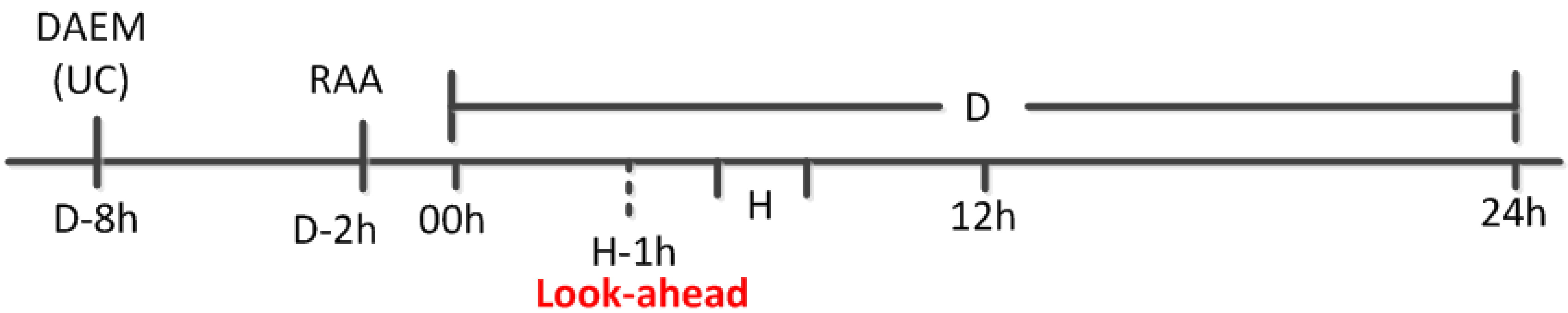
- Day-Ahead Energy Market (DAEM or DAM)
- Clears **demand bids** and **supply offers** at 1600h on the day prior to the operating day
- Produces:
  - Hourly schedules for the next operating day for market participants (i.e., generation and demand)
  - Hourly interchange schedules
  - Hourly day-ahead Locational Marginal Prices (LMPs)
- **No reserve requirements**

Reliability Unit Commitment - RUC (SCUC D-2h)

- Reliability Assessment (Reserve Adequacy Analysis - RAA)
- Minimize additional start-up and no load costs to provide sufficient capacity to satisfy the forecasted load plus the **operating and replacement reserve requirements**
- Clears ISO **forecasted load** at 2200h
- DAM commitments are respected
- Produces:
  - Additional commitments
  - Updated generator dispatch points

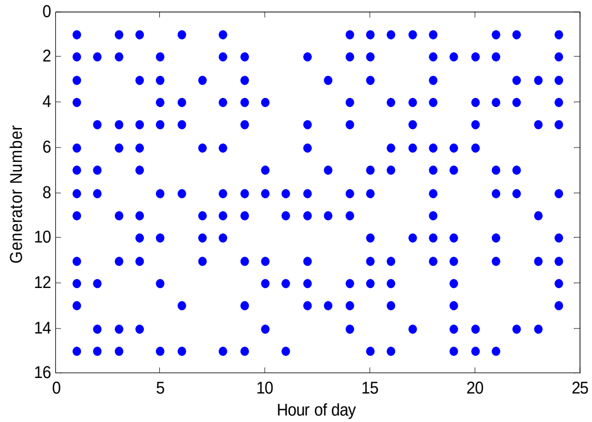
Look-Ahead Process – LA-SCUC (SCUC H-1h)

- RAA with ability to bring online fast start resources
- Also known as Look-Ahead SCED
- Intended to meet intra-hour reserve requirements
- Uses updated load and variable generation forecasts
- Produces:
  - Generator set points
  - Commitment of fast start units

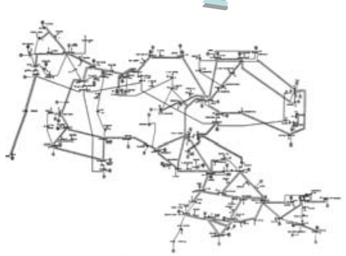
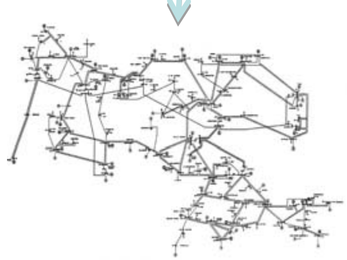


The General Structure of a Stochastic Unit Commitment Optimization Model

Objective: Minimize expected cost



p_1 p_2 … p_N

Scenario 1 Scenario 2 … Scenario N

First stage variables:
- •Unit On / Off

Nature resolves uncertainty
- •Renewables output
- •Forced outages

Second stage variables (*per time period*):
- •Generation levels
- •Power flows
- •Voltage angles
- •…

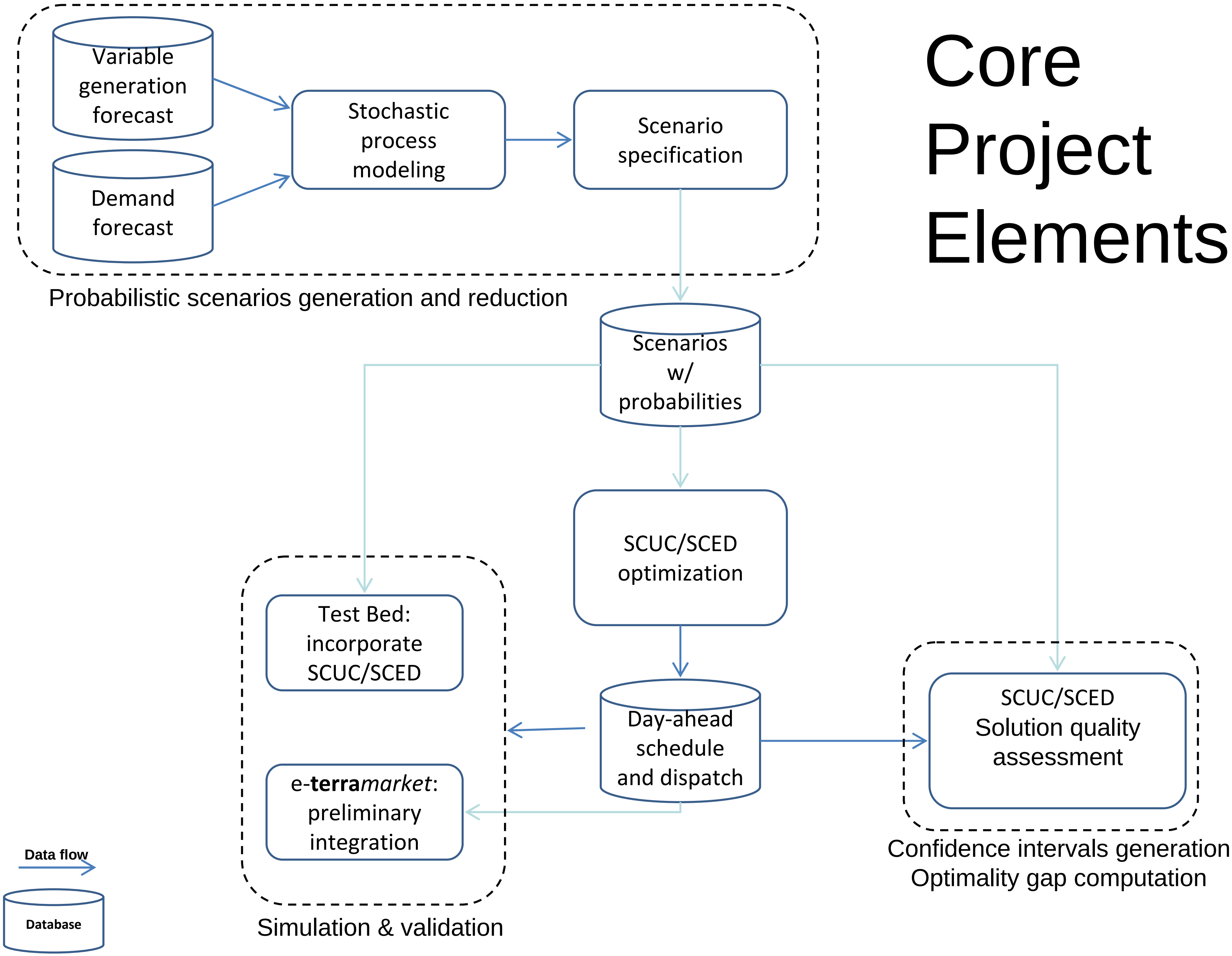
Uncertainty in DAM, RUC, and LA-SCUC Stochastic Programming Models

- Reliability Unit Commitment
  - Renewables generator output, load, forced (unplanned) outages
  - Fewer binaries than DAM, long time horizon, many scenarios

- Look-Ahead Unit Commitment
  - Similar to Reliability Unit Commitment
  - Fewer binaries than RUC, short time horizon, few scenarios

- Day-Ahead Unit Commitment
  - In contrast to RUC and LA-SCUC, an ISO can't really make direct use of a stochastic UC in the DAM without changing DAM procedures
  - With our partners, we are exploring alternative models and experimenting with procedures that incorporate stochastic models
  - We are eager to discuss ideas offline

Core Project Elements



Scenario-Based Decomposition via Progressive Hedging (PH)

1. $k := 0$
2. For all $s \in \mathcal{S}$, $x_s^{(k)} := \text{argmin}_x(c \cdot x + f_s \cdot y_s) : (x, y_s) \in \mathcal{Q}_s$
3. $\bar{x}^k := (\sum_{s \in \mathcal{S}} p_s\, d_s x_s^{(k)}) / \sum_{s \in \mathcal{S}} p_s\, d_s$
4. For all $s \in \mathcal{S}$, $w_s^{(k)} := \rho(x_s^{(k)} - \bar{x}^{(k)})$
5. $k := k + 1$
6. For all $s \in \mathcal{S}$, $x_s^{(k)} := \text{argmin}_x(c \cdot x + w_s^{(k-1)} x + \rho/2 \left\| x - \bar{x}^{(k-1)} \right\|^2 + f_s \cdot y_s)$ $: (x, y_s) \in \mathcal{Q}_s$
7. $\bar{x}^{(k)} := (\sum_{s \in \mathcal{S}} p_s\, d_s x_s^{(k)}) / \sum_{s \in \mathcal{S}} p_s\, d_s$
8. For all $s \in \mathcal{S}$, $w_s^{(k)} := w_s^{(k-1)} + \rho\left(x_s^{(k)} - \bar{x}^{(k)}\right)$
9. $g^{(k)} := \frac{(1-\alpha)|\mathcal{S}|}{\sum_{s \in \mathcal{S}} p_s\, d_s} \sum_{s \in \mathcal{S}} \left\| x^{(k)} - \bar{x}^{(k)} \right\|$
10. If $g^{(k)} < \epsilon$, then go to step 5. Otherwise, terminate.

Rockafellar and Wets (1991)

Progressive Hedging: Some Algorithmic Issues and their Resolution

- We are dealing with mixed-integer programs
  - So we have to deal with the possibility of cycling and other manifestations of non-convergence
  - See: *Progressive Hedging Innovations for a Class of Stochastic Mixed-Integer Resource Allocation Problems*, J.P. Watson and D.L. Woodruff, Computational Management Science, Vol. 8, No. 4, 2011

- What about good values for that pesky ρ parameter?
  - Poor or ad-hoc values of ρ can lead to atrocious performance
  - The good news in unit commitment
    - We have a lot of information concerning the cost of using a generator
    - Cost-proportional rho is a known, effective strategy in Progressive Hedging
  - Also see Computational Management Science paper indicated above

Progressive Hedging: Parallelization and Bundling

- Progressive Hedging is, at least conceptually, easily parallelized
  - Scenario sub-problem solves are clearly independent
  - Advantage over Benders, in that “bloat” is distributed
    - Critical in low-memory-per-node cluster environments
  - Parallel efficiency drops rapidly as the number of processors increases
    - But: *Relaxing barrier synchronization does not impact PH convergence*

- Why just one scenario per processor?
  - Bundling: Creating miniature “extensive forms” from multiple scenarios
    - Diverse or homogeneous scenario bundles?
  - Empirically results in a large reduction in total number of PH iterations
    - Growth in sub-problem cost *must* be mitigated by drop in iteration count
    - In practice, mitigation is enabled by cross-iteration warm starts

Scenario Sampling: How Many is Enough?

- Discretization of the scenario space is "standard" in stochastic programming
  - Often, no mention of solution or objective stability
  - Let alone rigorous statistical hypothesis-testing of stability
  - *Don't trust anyone who doesn't show you a confidence interval*

- Various approaches / alternatives in the literature
  - We like the Multiple Replication Procedure (MRP) introduced by Mak, Morton, and Wood (1999)

- Formal question we are concerned with
  - What is the probability that $\widehat{x}$'s objective function value is suboptimal by more than α%?
  - But making due with a fixed set or "universe" or scenarios

The Multiple Replication Procedure (Mak, Morton, Wood 1999)

MRP:

Input: Value $\alpha \in (0,1)$ (e.g., $\alpha = 0.05$), sample size n, replication size n_g, and a candidate solution $\hat{x} \in X$.

Output: Approximate $(1-\alpha)$-level confidence interval on $\mu_{\hat{x}}$.

1. For $k = 1, 2, \ldots, n_g$:
   1.1. Sample i.i.d. observations $\xi^{k1}, \xi^{k2}, \ldots, \xi^{kn}$ from the distribution of ξ.
   1.2. Solve (SP$_n$) using $\xi^{k1}, \xi^{k2}, \ldots, \xi^{kn}$ to obtain x_n^{k*}.
   1.3. Calculate $G_n^k(\hat{x}) = n^{-1} \sum_{j=1}^{n} (f(\hat{x}, \xi^{kj}) - f(x_n^{k*}, \xi^{kj}))$.
2. Calculate gap estimate and sample variance by

$$\bar{G}_n(n_g) = \frac{1}{n_g} \sum_{k=1}^{n_g} G_n^k(\hat{x}) \quad \text{and} \quad s_G^2(n_g) = \frac{1}{n_g - 1} \sum_{k=1}^{n_g} (G_n^k(\hat{x}) - \bar{G}_n(n_g))^2.$$

3. Let $\epsilon_g = t_{n_g-1,\alpha} s_G(n_g)/\sqrt{n_g}$, and output the one-sided CI on $\mu_{\hat{x}}$,

$$[0, \bar{G}_n(n_g) + \epsilon_g].$$

From Bayraksan and Morton (2009) – Assessing Solution Quality in Stochastic Programs Via Sampling

Illustrative MRP Results: Wind Farm Network Design

1000 scenarios, randomly sampled from a universe of 8760 scenarios

$\hat{n}$	n_g	n	Obj	E(Obj)	Gap(0.05)
70	2	465	89956	90639	828
70	5	186	89934	90639	764
70	10	93	89941	90639	870
70	20	46	89929	90639	1127
70	40	23	89929	90639	1356
140	2	430	89734	89779	354
140	5	172	89721	89779	272
140	10	86	89721	89779	462
140	20	43	89721	89779	792
140	40	21	89657	89779	1178
280	2	360	89755	89648	198
280	5	144	89744	89648	435
280	10	72	89750	89648	628
280	20	36	89750	89648	956
280	40	18	89750	89648	1403
420	2	290	90324	88832	251
420	5	116	90333	88832	555
420	10	58	90328	88832	718
420	20	29	90331	88832	996
420	40	14	90284	88832	1664
560	2	220	90577	89108	431
560	5	88	90587	89108	456
560	10	44	90583	89108	800
560	20	22	90585	89108	1252
560	40	11	90584	89108	2042

Key results:

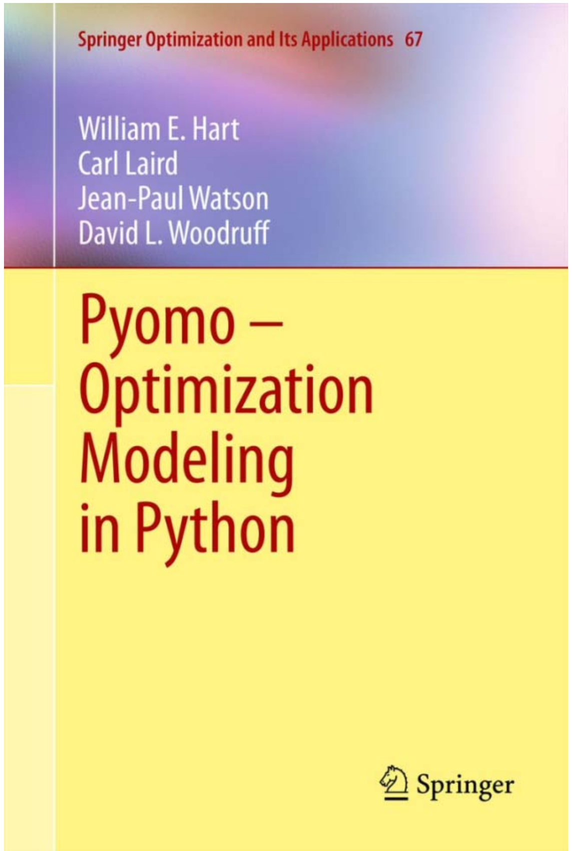
- Objective function value is remarkably stable across different parameterizations of the procedure
- Confidence interval widths are relatively small for a planning problem
- Results are stable across replications of the same parameterization of the MRP procedure

Practical impact: We don't need 8760 scenarios!

Looking to stochastic UC: We need more scenarios, not less…

Our Software Environment: Coopr

- Already discussed at this conference!
  - MD-1 (John Siirola)
- Project homepage
  - http://software.sandia.gov/coopr
- "The Book"



- Mathematical Programming Computation papers
  - Pyomo: Modeling and Solving Mathematical Programs in Python (Vol. 3, No. 3, 2011)
  - PySP: Modeling and Solving Stochastic Programs in Python (Vol. 4, No. 2, 2012)

Our Hardware Environments

- Our objective is to run on commodity clusters
  - Utilities don’t have, and don’t want, supercomputers
  - But they do or might have multi-hundred node clusters

- Sandia Red Sky (Unclassified Segment) – 39^{th} fastest on TOP500
  - Sun X6275 blades
  - 2816 dual socket / quad core nodes (22,528 cores)
    - 2.93 GHz Nehalem X5570 processors
    - 12 GB RAM per compute node (1.5 GB per core) << IMPORTANT!
  - For us, the interconnection is largely irrelevant
  - Red Hat Linux (RHEL 5)

- Sandia Red Mesa (with NREL)
  - Similar to Red Sky, but dedicated for energy research

A Few Words on UC Test Instances

- From the academic literature
  - Hand-constructed instances (Silva Monroy)
  - Textbook instances (Wood and Wollenberg)
  - RUC test literature – 10 and 100 generator instances
  - Simplified CAISO+WECC 240-bus test case
- From FERC
  - PJM-inspired / anonymized large-scale DAM UC and RUC instance
- From Alstom Grid
  - 70-bus test instance, used in development and testing of e-*terra*market
- From ISO-NE
  - Eastern Interconnection Planning Model-based instances

More Words on UC Test Instances

- What baseline deterministic UC model is best?
  - Carrion and Arroyo
  - Traditional three-binary generator state representation
  - Ostrowski et al.
  - …
- Lesson Learned #1
  - Not all models are correct, and all papers have unreported bugs
- Lesson Learned #2
  - Performance is *highly* dependent upon the test case
- Lesson Learned #3
  - Existing UC test cases are *really* bad
- Lesson Learned #4
  - Validating UC models is a highly non-trivial exercise

Conclusions

- Stochastic unit commitment has been studied in the literature
  - Indications are that it holds promise
  - Computational challenges have prevented industrial adoption
  - Far easier on paper and in academia than in practice…

- Our objective is to develop scalable solutions to stochastic unit commitment
  - In tractable run-times
  - On ISO-scale systems
  - To demonstrate (or not) both practical deployment ability and cost savings
  - Using reasonable, high-accuracy stochastic process models

- We are happy to talk to:
  - ISOs, vendors, and academics working toward related goals

QUESTIONS