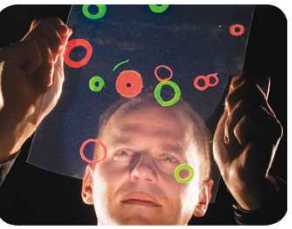
SAND2015-9231PE

Developing practical stochastic programming approaches to power grid operations and planning



John D. Siirola

Discrete Math & Optimization (1464)
Center for Computing Research
Sandia National Laboratories
Albuquerque, NM USA

Sandia National Laboratories

Exceptional service in the national interest

Texas A&M University
21 October 2015

U.S. DEPARTMENT OF ENERGY | NNSA National Nuclear Security Administration | CCR Center for Computing Research

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2011-XXXXP

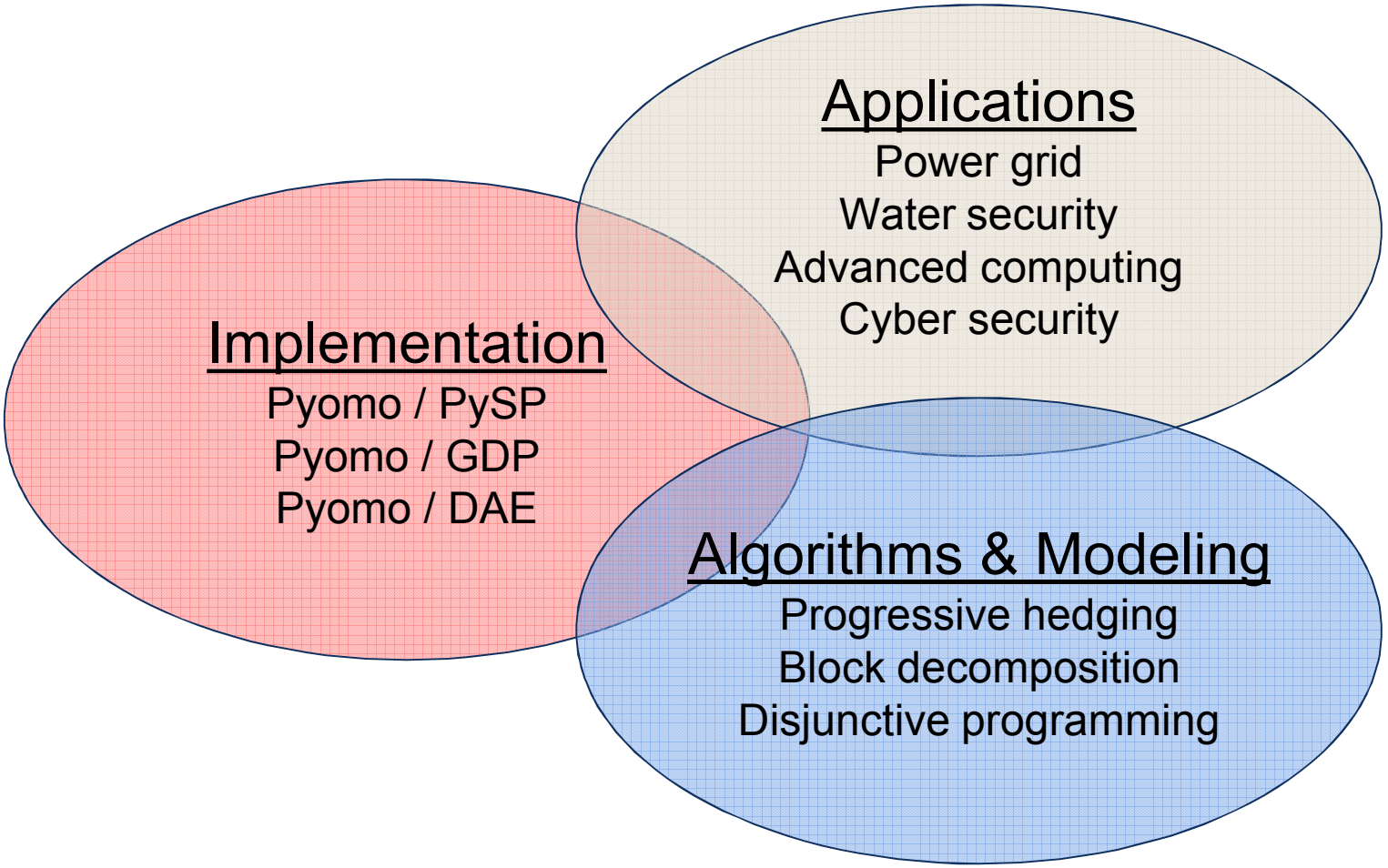
Acknowledgements

- Sandia National Laboratories
  - William Hart
  - Jean-Paul Watson
  - Francisco Munoz (now at Universidad Adolfo Ibáñez)
- University of California, Davis
  - Prof. David L. Woodruff
  - Prof. Roger Wets
- Purdue University
  - Prof. Carl D. Laird
- Oregon State University
  - Gabe Hackebeil
- Carnegie Mellon University
  - Bethany Nicholson
- Imperial College London
  - Mahdi Sharifzadeh

OR research at Sandia

- Sandia National Laboratories
  - Science & Technology Division
    - Center for Computing Research
      - Discrete Math and Optimization
- Sandia is a Federally-Funded Research and Development Center operated for the U.S. Department of Energy
  - Multimission national security laboratory
  - Historical focus on engineering applications
- Science & Technology Division is home to the Labs' *Research Foundations*
  - Pursue fundamental research driven by mission needs
- Discrete Math & Optimization Department
  - Conducts fundamental and applied OR/CS/Analytics research
  - Focus on algorithms, modeling approaches, and scalable tools
  - Partners with universities, other OR groups in Sandia "mission areas"

3 threads to our research (and this talk)



Operating the electric power grid

- Conceptually simple

$$\sum demand = \sum generation - \sum losses \quad \forall\, t \in T$$

  - This is just a single(*) component process flow network with fixed demands and controllable supplies

- In practice, this is complicated by
  - No (significant) storage
  - Dynamic constraints (ramp rates)
  - Transmission limitations
  - Security (reliability) requirements
  - Market constraints

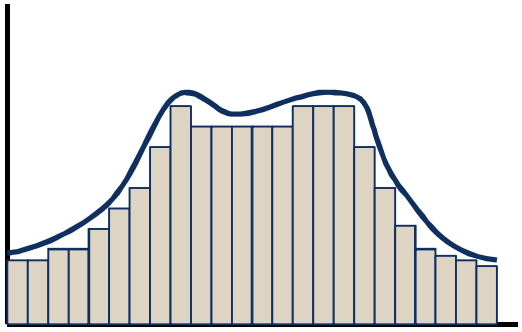
(*) Actually, it is a 2-component system (real and reactive power) with "reactors" at every node, but for the purposes of this talk we will follow industry's lead and allow a small angle assumption to only work with the "DC" optimal power flow model.

An ISO's view of operating the grid (1)

- [Unit Commitment]: Plan a day ahead (1600h)
  - Demand forecast
  - Generator bids
  - Produce:
    - Hourly (on/off) schedules for all participants
    - Hourly interconnect schedules
    - Hourly DAEM Locational Marginal Prices (LMPs)

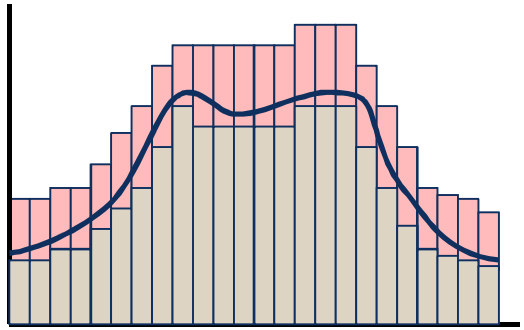
"Day Ahead Market" (DAM) / "Day Ahead Energy Market" (DAEM)

- So what's the problem?
  - *No one believes the forecast.*
  - *Things go wrong*



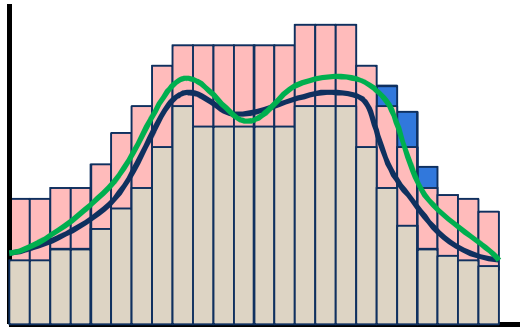
An ISO's view of operating the grid (2)

- [Reliability Unit Commitment]: Modify the plan
  - Reserve Adequacy Analysis – allocate reserves to meet load
    - Standard: 10% reserve requirement
  - Contingency analysis
    - *N-1*: survive loss of (1) generator / line
  - Produces:
    - Additional commitments (DAEM respected)
    - Updated generator dispatch points



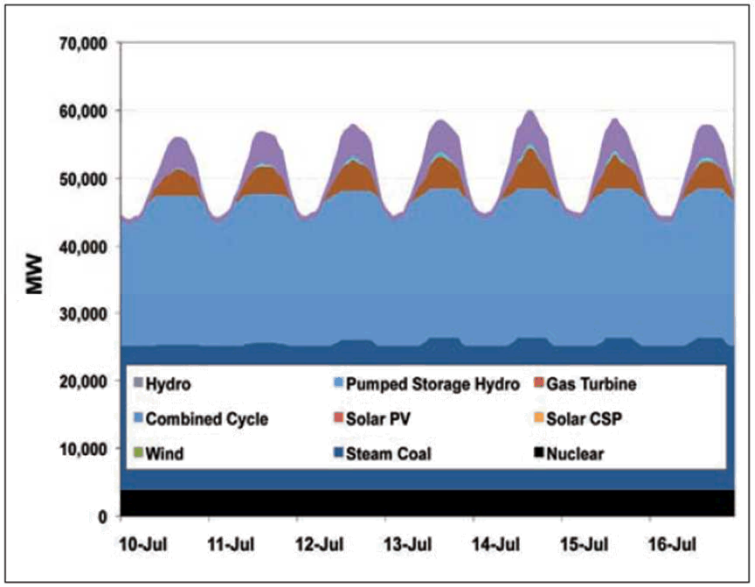
An ISO's view of operating the grid (3)

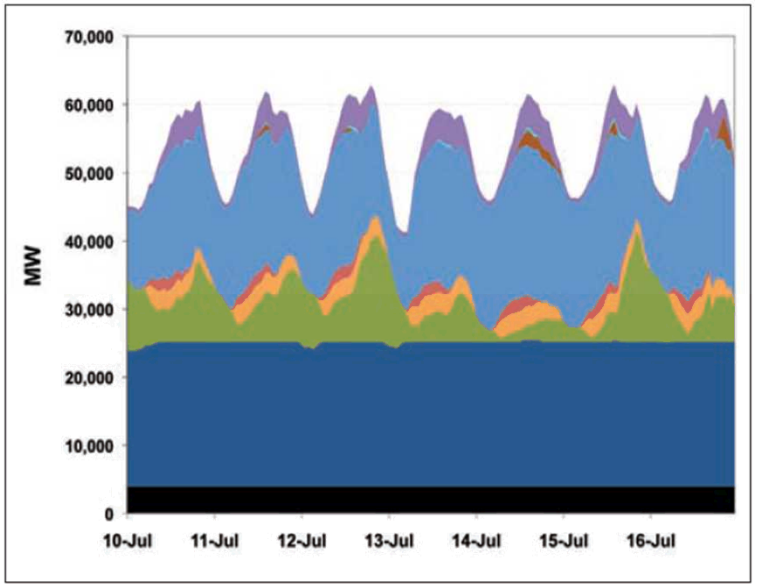
- [Economic Dispatch]: operate the grid / serve actual load
  - H – 1h: Look-ahead economic dispatch
  - H: Hourly economic dispatch
  - H+5*n*: 5-minute economic dispatch
  - Produces:
    - Updated dispatch points (generator output levels)
    - Additional commitments (fast-start units)



Given the lights are on, why care?

- Nondispatchable generation (renewables)
  - Frequently treated as “must-take” resources
    - Appears as “negative demand”
  - Less predictable than consumer demand
    - Increased reserve requirement

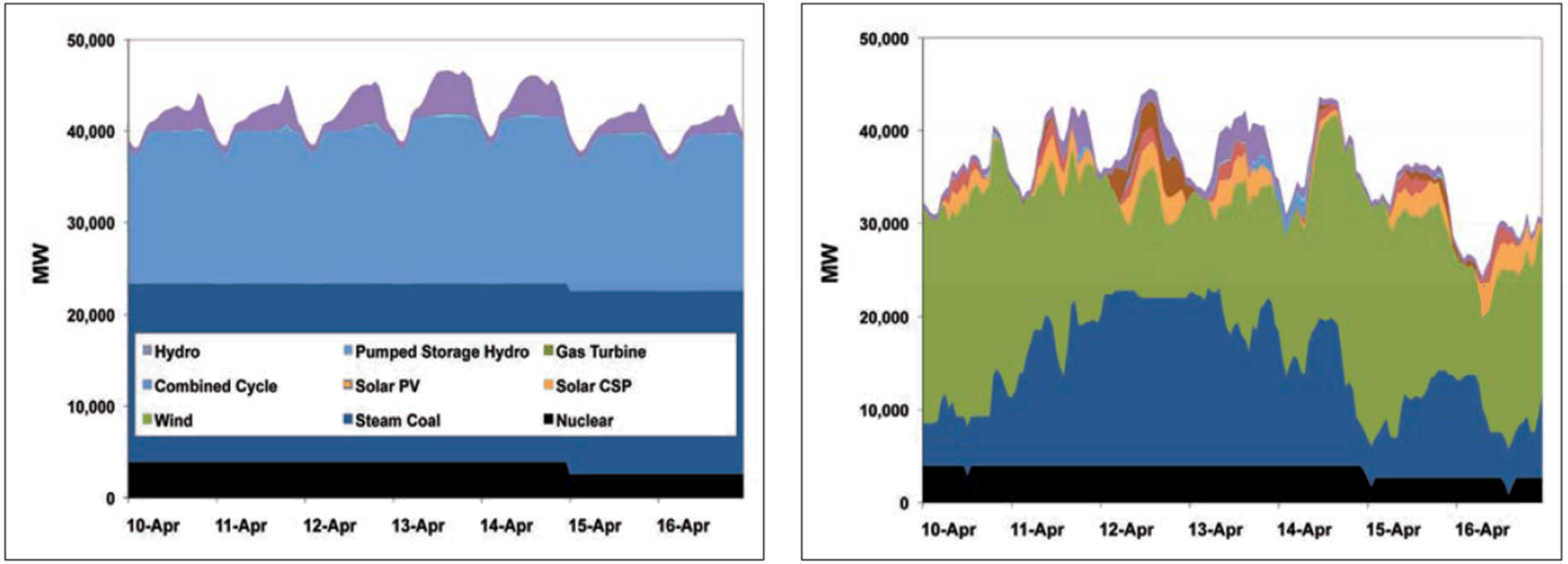




0% and 30% renewable penetration for an “easy week” in July.
Reproduced from NREL 2010 *Western Wind and Solar Integration Study*

Given the lights are on, why care?

- Nondispatchable generation (renewables)
  - Increased ramping of base-load generation
    - Results in increased O&M costs and higher forced outage rates
- Can we reduce cost by explicitly addressing uncertainty?

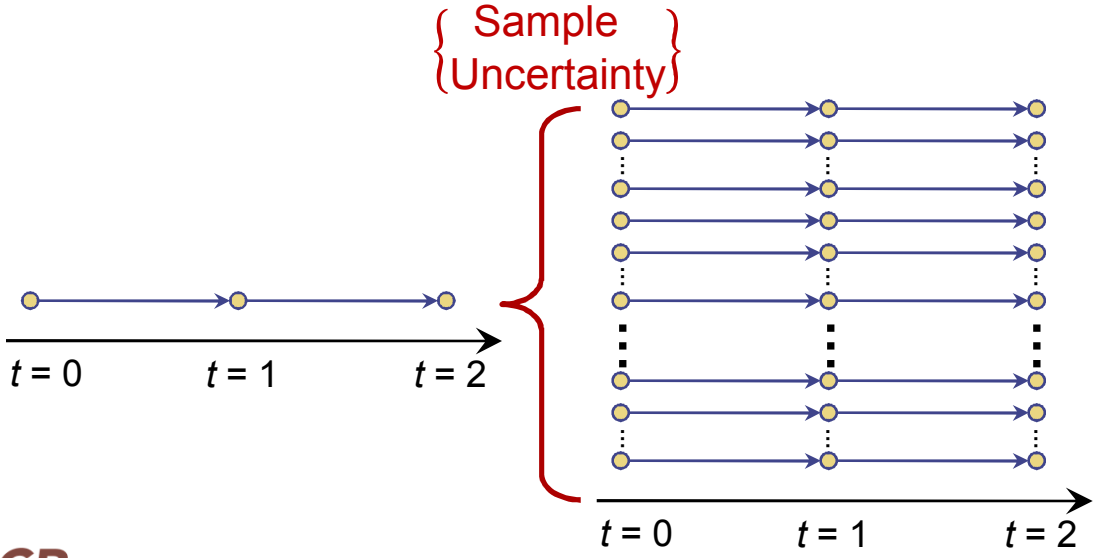


0% and 30% renewable penetration for an "challenging week" in April.
Reproduced from NREL 2010 *Western Wind and Solar Integration Study*

Stochastic Unit Commitment

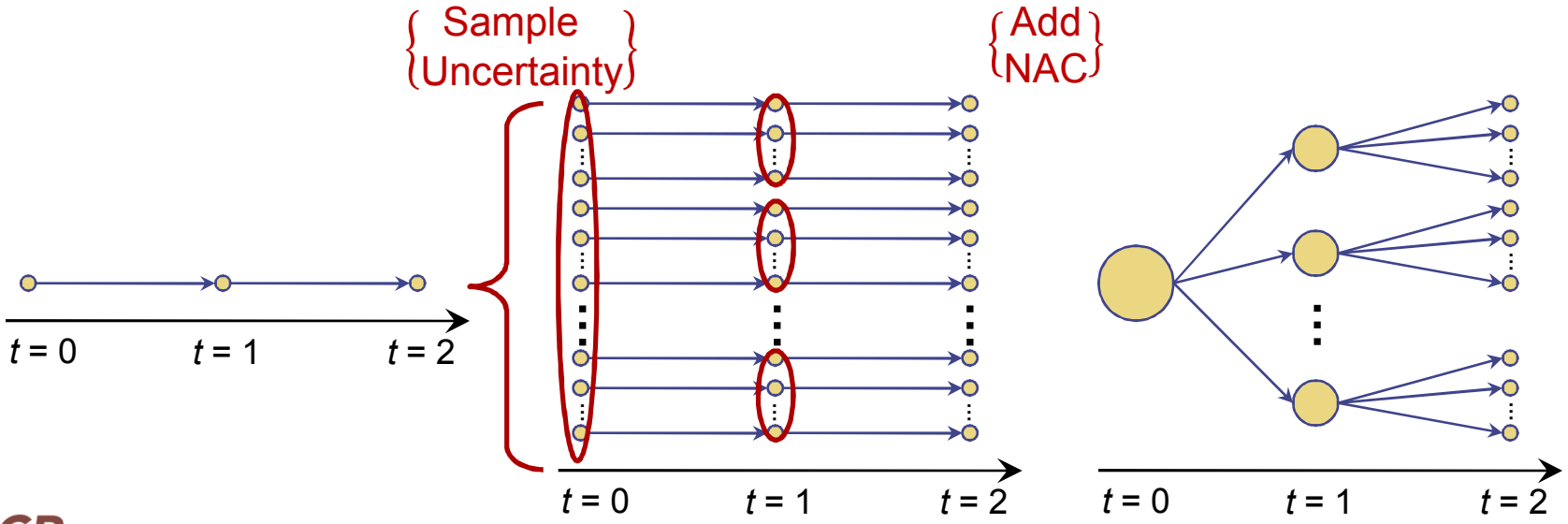
Optimization under Uncertainty

- Many options for capturing uncertainty
  - Sampling / Surrogate methods / Robust optimization / (Approximate) dynamic programming / Stochastic programming
  - *We focus on stochastic programming*
    - Capture problem uncertainty as a set of possible scenarios
    - Solve to select a single answer that optimizes across all scenarios



Optimization under Uncertainty

- Many options for capturing uncertainty
  - Sampling / Surrogate methods / Robust optimization / (Approximate) dynamic programming / Stochastic programming
  - *We focus on stochastic programming*
    - Capture problem uncertainty as a set of possible scenarios
    - Solve to select a single answer that optimizes across all scenarios



Pyomo: PYthon Optimization Modeling Objects

Sandia National Laboratories

- Pyomo is…
  - A library of classes for expressing optimization models in Python?
  - A collection of tools for importing and exporting data?
  - A set of interfaces to numerous LP, MIP, NLP, and MINLP solvers?
  - An executable that takes a model & data, sends it to a solver, and reports the final solution?
  - A collection of routines for manipulating optimization models?
  - A collection of optimization algorithms implemented in Python?
  - A software environment for teaching optimization?
  - An open-source community for applied OR research?
- But doesn't this sound a lot like AMPL, GAMS, AIMMS, gPROMS, …?
  - Why did we "reinvent the wheel"?

CCR Center for Computing Research

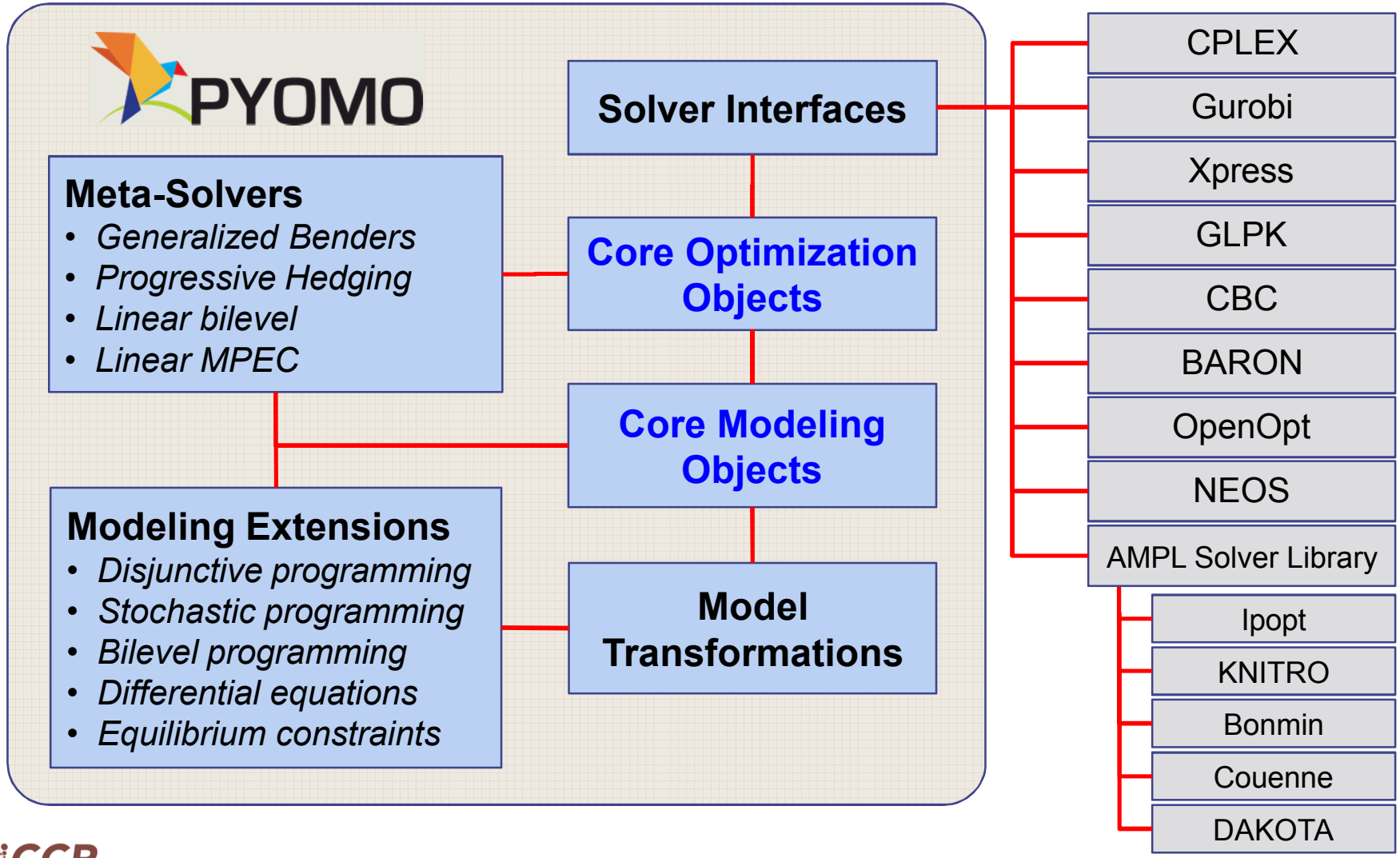
Why *we* needed Pyomo

Sandia National Laboratories

- 10 years ago…
  - Modeled primarily in AMPL (with some GAMS)
  - Developed solvers primarily in C++ (with MPI)
    - PICO, Coliny, DAKOTA, Opt++, APPSPACK
- This model worked, but…
  - Large code bases were unwieldly and not easily extensible
  - Commercial modeling environments lacked support for "higher level" modeling constructs, e.g. stochastic programming
  - No obvious path to implementing "meta algorithms"
    - Require frequent calls to optimization codes to solve subproblems
    - Difficult to implement in optimization modeling environments
    - Tedious to code directly against a solver's API (and then you are tied to that solver)
  - Difficult to transfer our results to other organizations
  - Difficult to incorporate new ideas developed by the broader OR community
  - Dampened our ability to rapidly prototype ideas and explore new areas
    - We increasingly found ourselves providing *optimization support* to larger projects

CCR Center for Computing Research

Pyomo at a Glance



More than just mathematical modeling…

Sandia National Laboratories

Meta-solvers

- Integrate scripting and/or transformations into optimization solver
- Leverage Python's introspective nature to build "generic" capabilities
- e.g., progressive hedging, Benders decomposition

Model transformations (a.k.a. reformulations)

- Automate generation of one model from another
- Leverage Pyomo's object model to apply transformations sequentially
- e.g., DAE → NLP, GDP → Big M

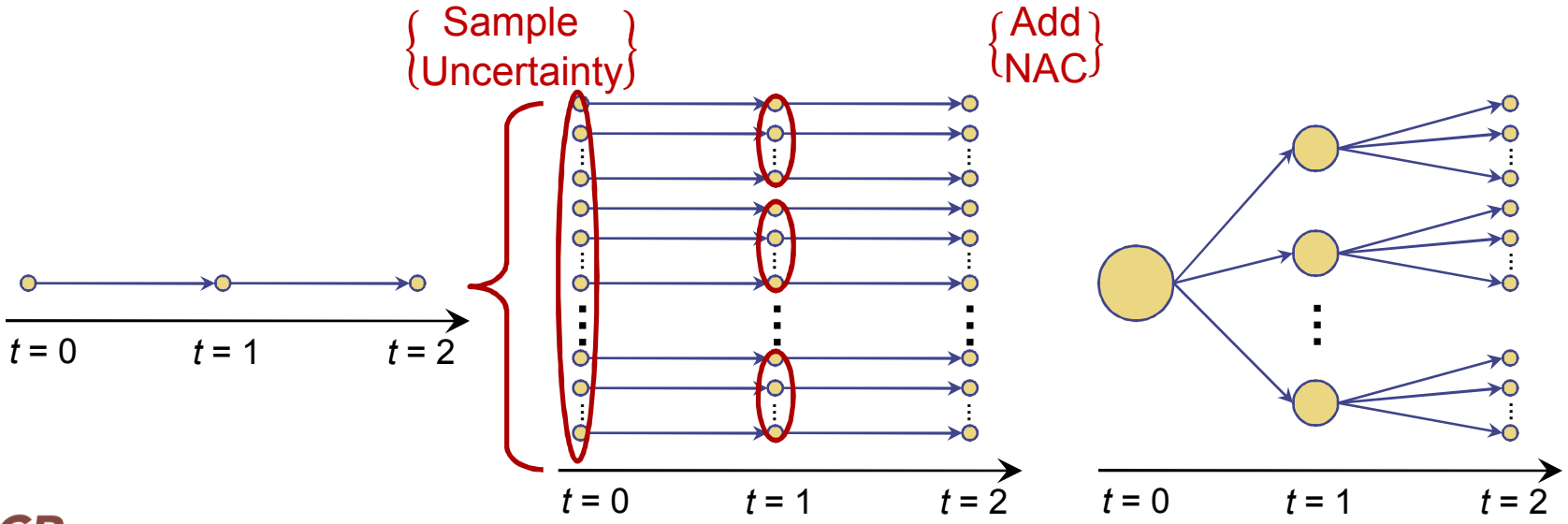
Scripting

- Construct models using native Python data
- Iterative analysis of models leveraging Python functionality
- Data analysis and visualization of optimization results

CCR Center for Computing Research

PySP: SP made Simple

- Step 1: write the deterministic model (in Pyomo)
  - You've (probably) already done this
- Step 2: generate scenario data
  - This is the hardest part (for UC, it took 2 years)
- Step 3: PySP expands the model & adds the NACs

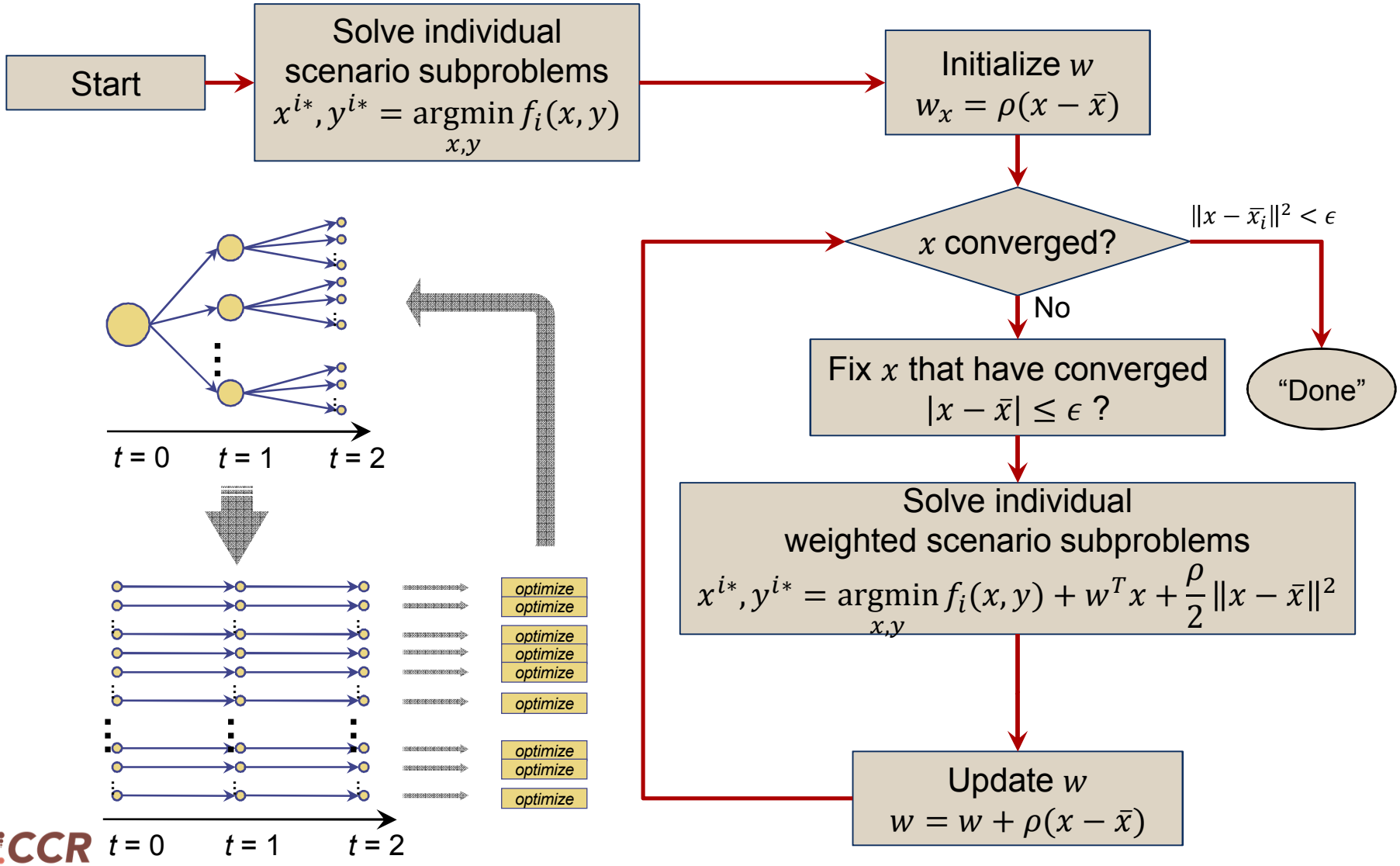


“So what?”

- The PySP process is conceptually no different than you would do in any other AML
  - Except in a traditional AML you would have to explicitly add and track the scenario index and add the NAC
  - If the Extensive Form is solvable, no significant difference between PySP and AMPL / GAMS / AIMMS…
    - But what if it’s not solvable?

- Structure: The real power of PySP
  - PySP explicitly understands the structure of the problem
  - We can *automate* decomposition strategies
    - Stage-wise decomposition (e.g., Benders decomposition)
    - Scenario-wise decomposition (e.g., Progressive hedging)

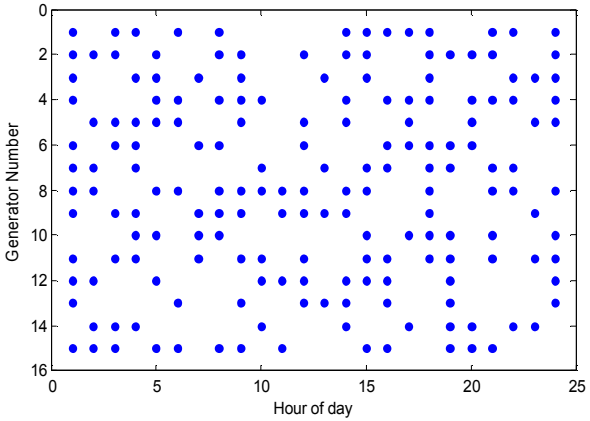
Progressive Hedging: the algorithm



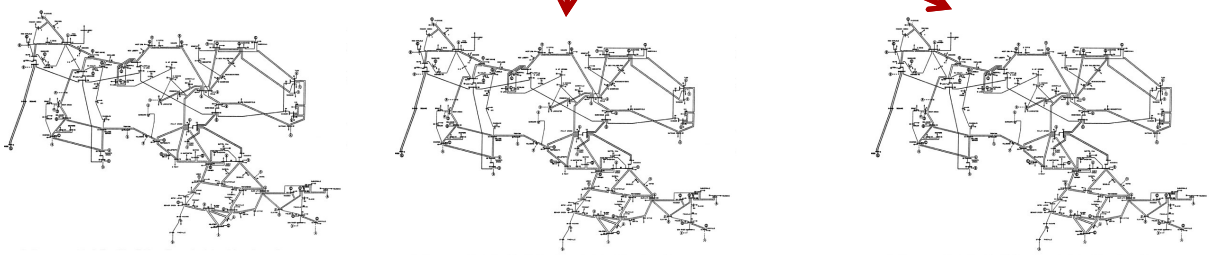
Stochastic Unit Commitment (at scale)

[J.-P. Watson, D. Woodruff]

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

- Load
- Renewables output
- Forced outages

Second stage variables (*per time period*):

- Generation levels
- Power flows
- Voltage angles
- …

Progressive Hedging Results: WECC-240++

Table 7 Solve time (in seconds) and solution quality statistics for PH executing on the *WECC-240-r1* instance, with $\alpha = 0.5$, $\mu = 6$, and $\gamma = 0.025$

# Scenarios	Convergence Metric	Obj. Value	PH L.B.	# Vars Fx.	Time	
		64-Core Workstation Results				Latest…
3	0.0 (20 iters)	64213.397	63235.381	4080	508	166
5	0.0 (in 18 iters)	62642.531	61767.253	4079	674	119
10	0.0 (in 35 iters)	61396.553	60476.604	4066	648	167
25	0.0 (in 22 iters)	60935.040	59992.622	4066	761	212
50	0.0 (in 15 iters)	60625.149	59631.839	4034	1076	280
100	0.0 (in 25 iters)	61155.387	60014.571	4080	1735	315
		Red Sky Results				
50	0.0 (in 16 iters)	60623.343	59779.813	4007	404	
100	0.0 (in 25 iters)	61120.943	60275.744	4080	549	

ISO-NE results are obtained on Red Sky on average in 10 minutes, 20 minutes in the worst case (with 100 scenarios)

Improved UC Formulations?

- Morales-Espana et al. (2013)
  - Extends prior tight formulation by Ostrowski et al.
- Shows off advantage of PH, in that improved deterministic models immediately impact stochastic solve times
- Results

Table 10 Solve time (in seconds) and solution quality statistics for PH executing on the *WECC-240-r1* instance, with $\alpha = 0.5$, $\mu = 3$, and the MTR deterministic UC model.

# Scenarios	Convergence Metric	Obj. Value	PH L.B.	# Vars Fx.	Time
		64-Core Workstation Results			
3	0.0 (in 36 iters)	64141.771	64109.021	4080	237
5	0.0 (in 23 iters)	62628.532	62499.212	4080	161
10	0.0 (in 26 iters)	61384.016	61327.734	4080	215
25	0.0 (in 41 iters)	60927.903	60850.717	4080	366
50	0.0 (in 11 iters)	60617.311	60470.956	4044	318

- ISO-NE results drop to 15 minutes maximum (10 average)

So what is the impact? ISO-NE analysis

Sandia National Laboratories

- Cost-savings analysis for ISO-NE
  - 2004 Eastern Wind data
  - 50 wind scenarios per day
    - Generated using our tool chain based on epi-splines
    - (Simulated) actual taken from NREL database
  - 1 load scenario per day
    - Expected load computed using our epi-spline tool chain
    - Models fit using historical ISO-NE 2011 data
    - Actual taken from actual ISO-NE 2011 data
    - "Platinum" standard simulation, i.e., rolling horizon
  - Wind is *not* modeled as must-take
    - Per advice from NREL
    - In practice, there are days at these penetration levels in which it is impossible to use net load formulations w/o shedding

CCR Center for Computing Research

Cutting to the Chase: Cost Savings

- Computed in terms of relative cost increase of deterministic (w/ 10% reserves) over stochastic (w/ 2% reserves)
  - Yes, this implies that stochastic does win (but)…
- Results in terms of percentages
  - Q1: 1.52%
  - Q2: 1.31%
  - Q3: 0.89%
  - Q4: 1.23%
- Not as significant as we would have anticipated, given the large wind penetration levels we simulated
  - For various reasons, we believe these results underestimate savings

Cutting to the Chase: Cost Savings

Sandia National Laboratories

- Computed in terms of relative cost increase of deterministic (w/ 10% reserves) over stochastic (w/ 2% reserves)
  - Yes, this implies that stochastic does win (but)…
- Results in terms of percentages
  - Q1: 1.52% → ~$ 4M per month
  - Q2: 1.31% → ~$ 3M per month
  - Q3: 0.89% → ~$12M per month
  - Q4: 1.23% → ~$2.5M per month

  $64.5M "estimated savings" for 2011
- Not as significant as we would have anticipated, given the large wind penetration levels we simulated
  - For various reasons, we believe these results underestimate savings

CCR Center for Computing Research

Reliability Results

- We did not report load shedding and/or reserve shortfalls in the previous cost savings statistics
  - Placing arbitrary penalty values on these quantities is not useful
  - Distinct reporting allows more insight into system behaviors
- Stochastic UC
  - One load shedding event – peak day in July
  - Incurred due to particularly bad load forecast
- Deterministic UC
  - Five load shedding events – including the peak day in July
  - Additionally incurs reserve margin shortfalls on approximately 10% of all days in 2011
- Summary
  - Stochastic UC, despite lower reserve margins, is more reliable

Power grid contingency analysis

Power grid contingency analysis

Sandia National Laboratories

[with J.-P. Watson]

- U.S. ISO's must operate with "*N-1*" reliability
  - System must be able to "survive" loss of 1 generator / (non-radial) line
  - Not explicitly included in Unit Commitment model
  - Practice is to include "proxy constraints" and post-solve verification
- Some studies indicate that intentionally switching lines could improve contingency response
  - UC + N-1 + Transmission switching (e.g. [Hedman, et al. 2010])
  - *"Just allowing processing* [of the RTS-96 test case] *at the root node typically takes 20h on a desktop workstation…"*
  - *"While reducing [the] optimality gap to zero is an interesting academic issue…"*
- Case study: RTS-96 test case
  - 73 busses, 115 non-radial lines, 99 generators
  - 214 contingencies

CCR Center for Computing Research

The solution: UC + Transmission Switching + *N-1*

Sandia National Laboratories

$$\text{Minimize}: \quad \sum_t \sum_g \left(c_g P_{g0t} + c_g^{SU} v_{gt} + c_g^{SD} w_{gt}\right) \tag{1}$$

$$\text{S.t.} \quad \theta^{\min} \le \theta_{nct} \le \theta^{\max}, \quad \forall\, n, c, t \tag{2}$$

$$\sum_{\forall\, k(n,.)} P_{kct} - \sum_{\forall\, k(.,n)} P_{kct} + \sum_{\forall\, g(n)} P_{g0t} = d_{nt},$$

$$\forall\, n, \quad c = 0, \text{ transmission contingency states } c, t \tag{3a}$$

$$\sum_{\forall\, k(n,.)} P_{kct} - \sum_{\forall\, k(.,n)} P_{kct} + \sum_{\forall\, g(n)} P_{gct} = d_{nt},$$

$$\forall\, n, \text{ generator contingency states } c, t \tag{3b}$$

$$P_{kc}^{\min} N1_{kc} z_{kt} \le P_{kct} \le P_{kc}^{\max} N1_{kc} z_{kt}, \quad \forall\, k, c, t \tag{4}$$

$$B_k(\theta_{nct} - \theta_{mct}) - P_{kct} + (2 - z_{kt} - N1_{kc}) M_k \ge 0, \quad \forall\, k, c, t \tag{5a}$$

$$B_k(\theta_{nct} - \theta_{mct}) - P_{kct} - (2 - z_{kt} - N1_{kc}) M_k \le 0, \quad \forall\, k, c, t \tag{5b}$$

$$P_g^{\min} N1_{gc} u_{gt} \le P_{gct} \le P_g^{\max} N1_{gc} u_{gt}, \quad \forall\, g, c, t \tag{6}$$

$$v_{g,t} - w_{g,t} = u_{g,t} - u_{g,t-1}, \quad \forall\, g, t \tag{7}$$

$$\sum_{q=t-UT_g+1}^{t} v_{g,q} \le u_{g,t}, \quad \forall\, g, t \in \{UT_g, \ldots, T\} \tag{8}$$

$$\sum_{q=t-DT_g+1}^{t} w_{g,q} \le 1 - u_{g,t}, \forall\, g, t \in \{DT_g, \ldots, T\} \tag{9}$$

$$P_{g0t} - P_{g0,t-1} \le R_g^{+} u_{g,t-1} + R_g^{SU} v_{g,t}, \quad \forall\, g, t \tag{10}$$

$$P_{g0,t-1} - P_{g0,t} \le R_g^{-} u_{g,t} + R_g^{SD} w_{g,t}, \quad \forall\, g, t \tag{11}$$

$$P_{gct} - P_{g0,t} \le R_g^{+}, \quad \forall\, g, c, t \tag{12}$$

$$P_{g0,t} N1_{gc} - P_{gct} \le R_g^{-}, \quad \forall\, g, c, t \tag{13}$$

$$0 \le v_{g,t} \le 1, \quad \forall\, g, t \tag{14}$$

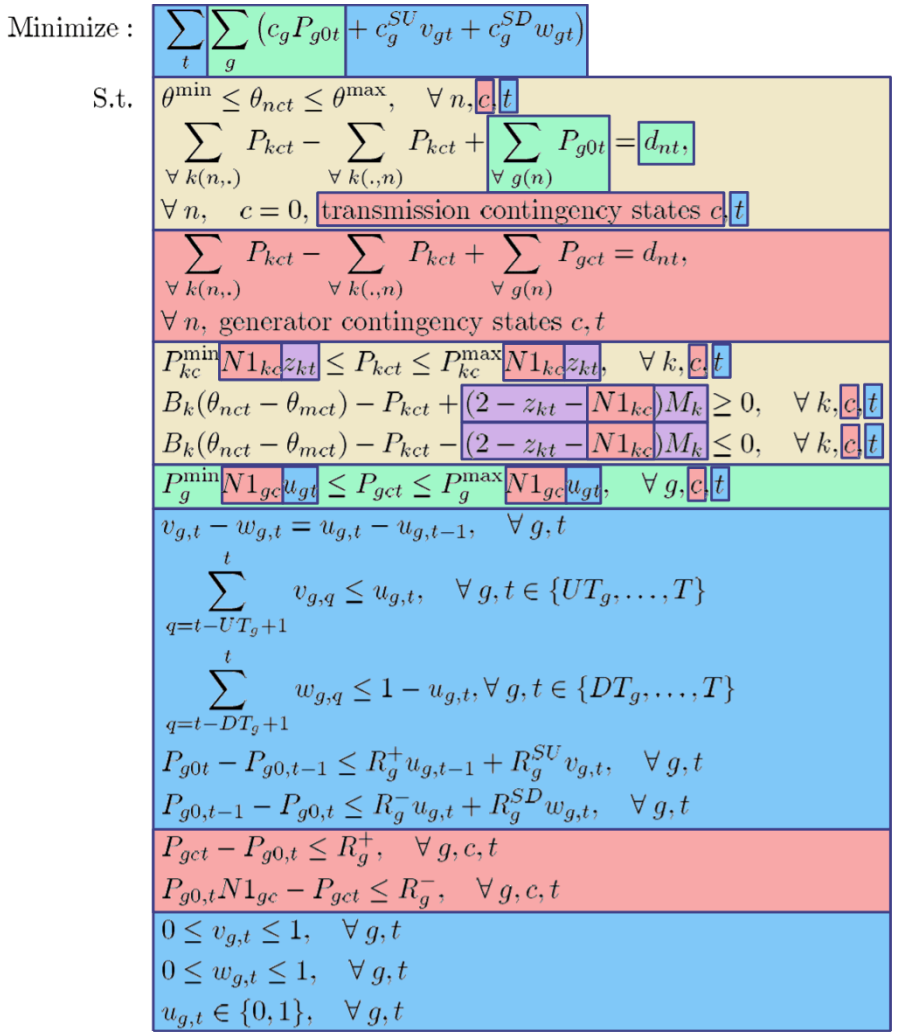
$$0 \le w_{g,t} \le 1, \quad \forall\, g, t \tag{15}$$

$$u_{g,t} \in \{0, 1\}, \quad \forall\, g, t \tag{16}$$

CCR Center for Computing Research

Hedman, et al., "Co-Optimization of Generation Unit Commitment and Transmission Switching With N-1 Reliability," *IEEE Trans Power Systems*, 25(2), pp.1052-1063, 2010

The challenge: MP is dense and subtle

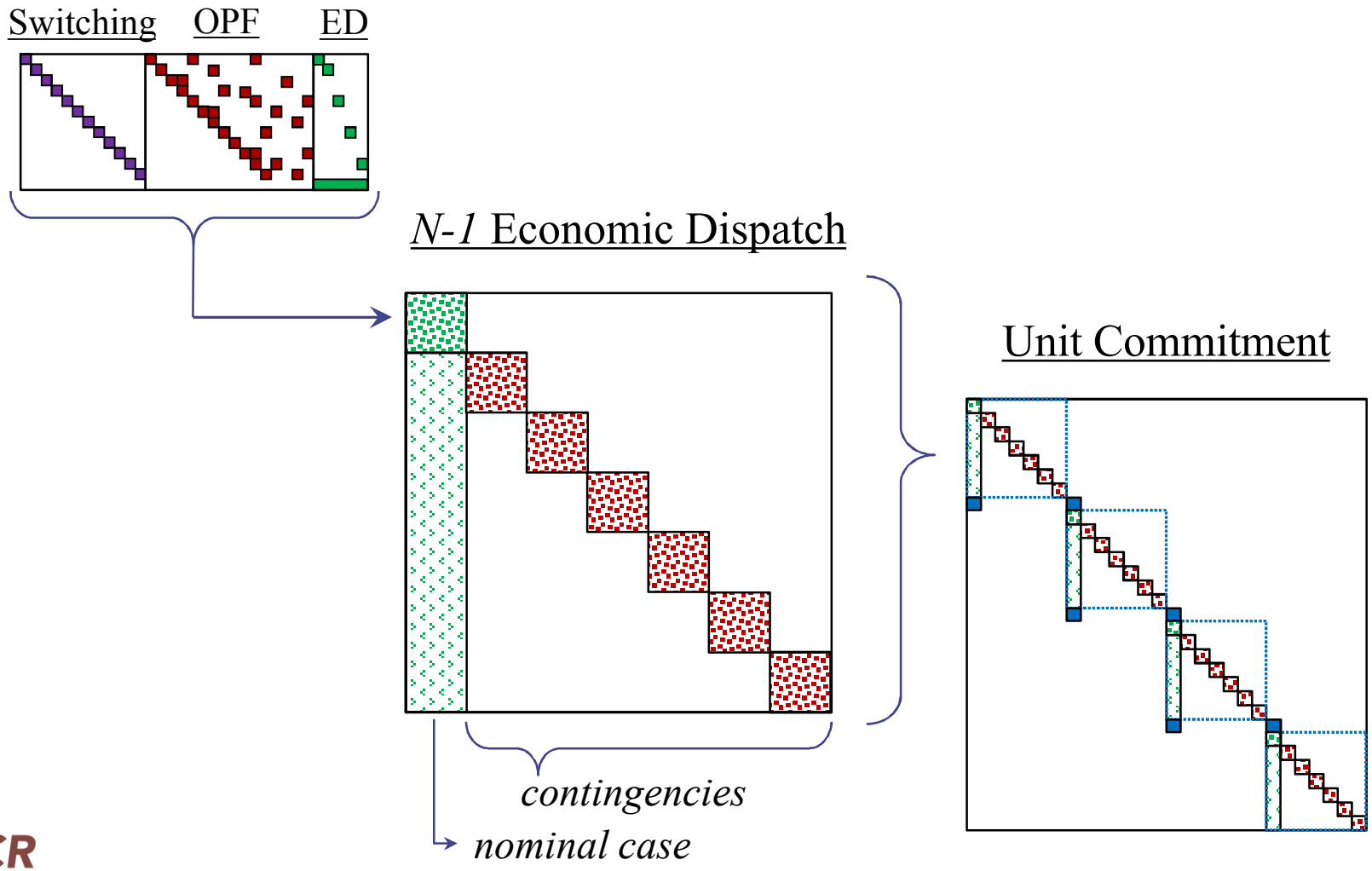


To a first approximation:

- DCOPF
- Economic dispatch
- Unit commitment
- Transmission switching
- N-1 contingency

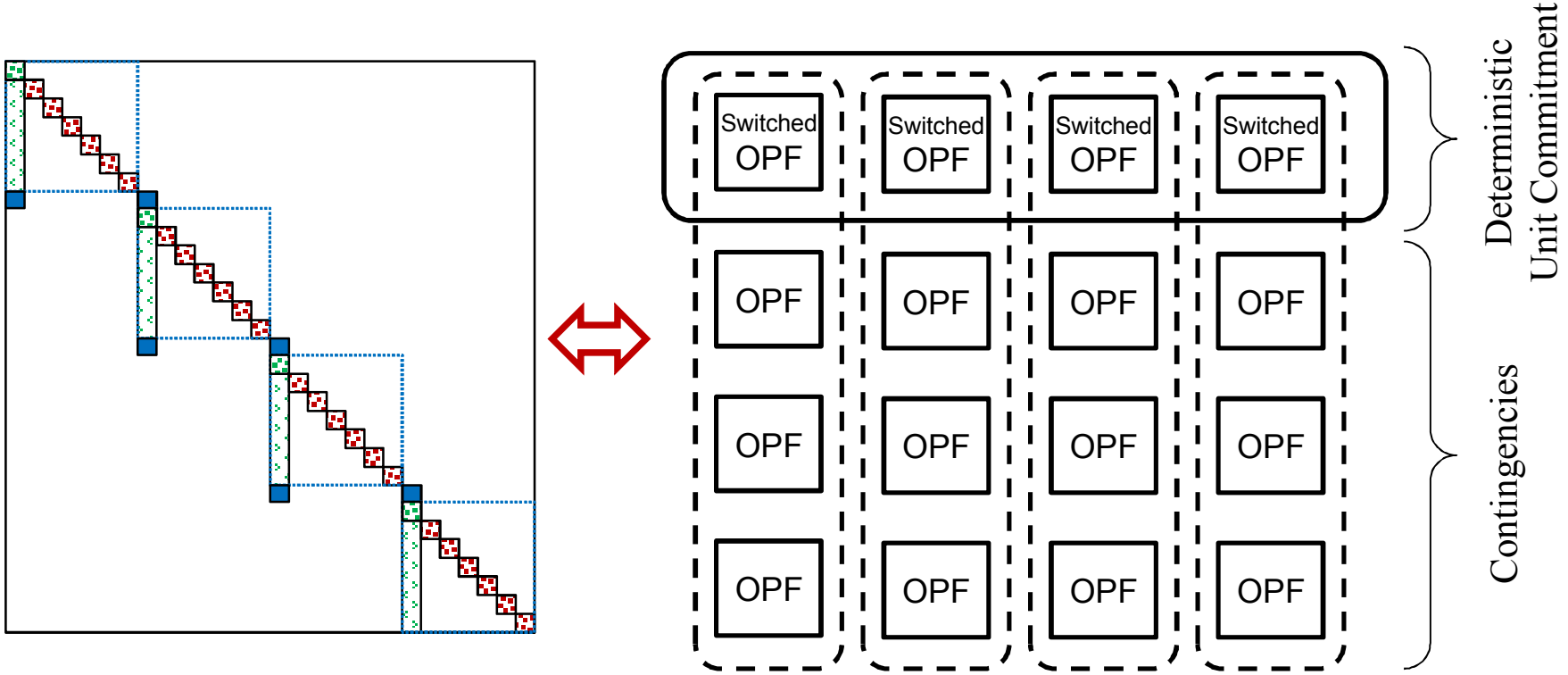
Hedman, et al., "Co-Optimization of Generation Unit Commitment and Transmission Switching With N-1 Reliability," *IEEE Trans Power Systems*, 25(2), pp.1052-1063, 2010

(Nonobvious) Inherent structure



UC + N-1 + Switching block structure

- "2-D" grid of linked optimal power flow models



Explicitly expose disjunctive decisions

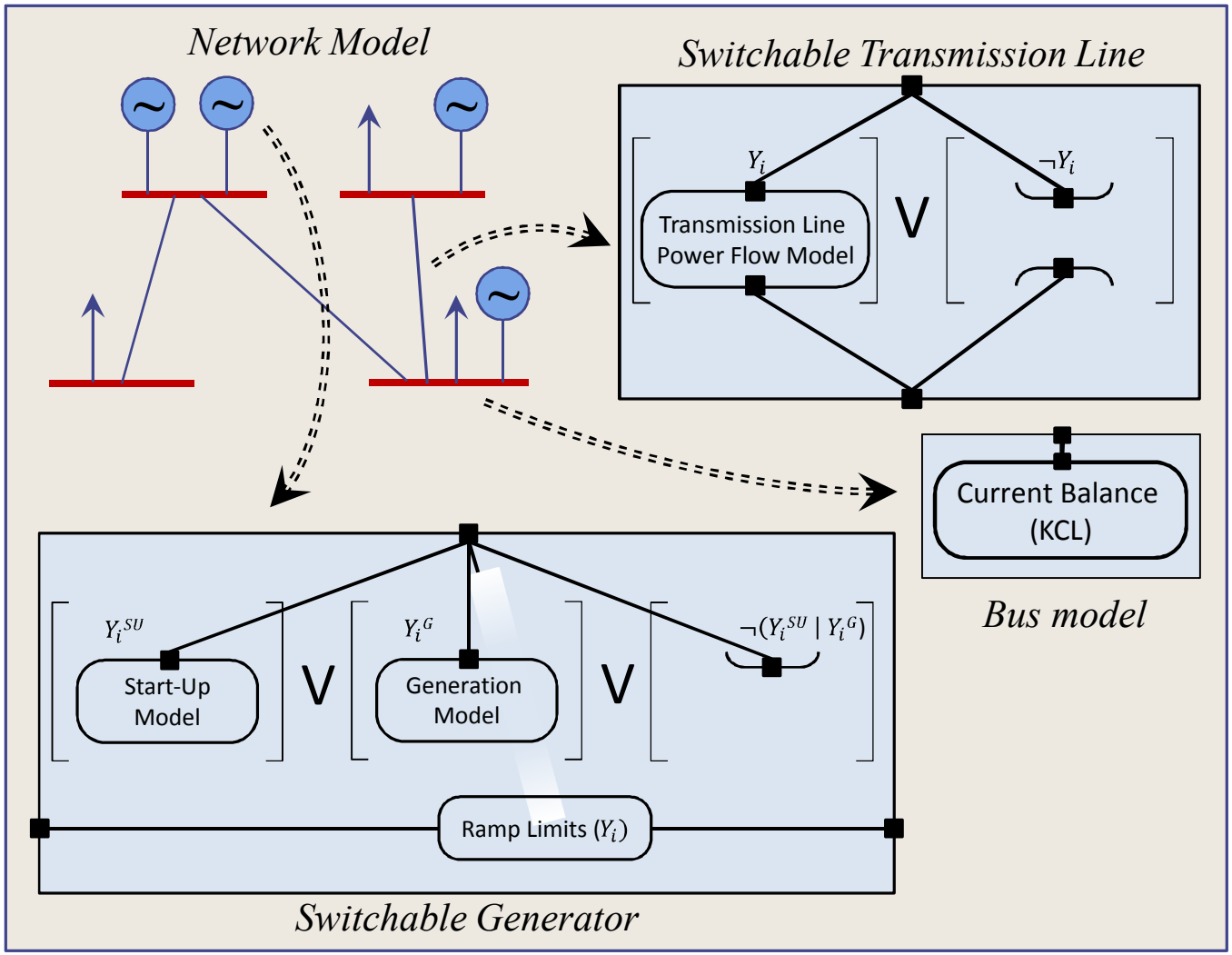
- Transmission switching:

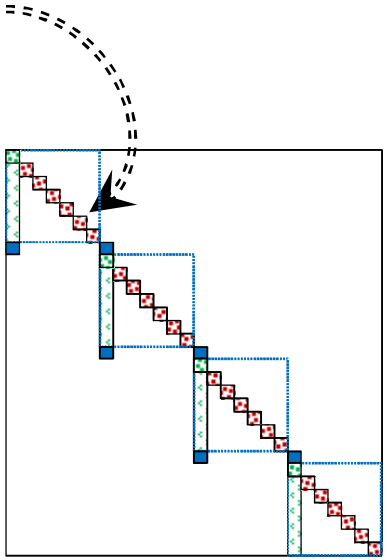
$$\begin{bmatrix} z_{kct} \\ P_{kct} = B_k(\theta_{k1} - \theta_{k2}) \end{bmatrix} \vee \begin{bmatrix} \neg z_{kct} \\ P_{kct} = 0 \end{bmatrix}$$

- Generation

$$\begin{bmatrix} u_{gt} \\ C_{gt} = P_{gt} c_g \\ R_g^+ \geq P_{gt} - P_{gt-1} \\ R_g^- \geq P_{gt-1} - P_{gt} \end{bmatrix} \vee \begin{bmatrix} v_{kt} \\ C_{gt} = P_{gt} c_g + c_g^{SU} \\ R_g^{SU} \geq P_{gt} - P_{gt-1} \end{bmatrix} \vee \begin{bmatrix} \neg(u_{kt} \mid v_{kt}) \\ C_{gt} = c_g^{SD} u_{kt-1} \\ R_g^{SD} \geq P_{gt-1} - P_{gt} \\ P_{gt} = 0 \end{bmatrix}$$

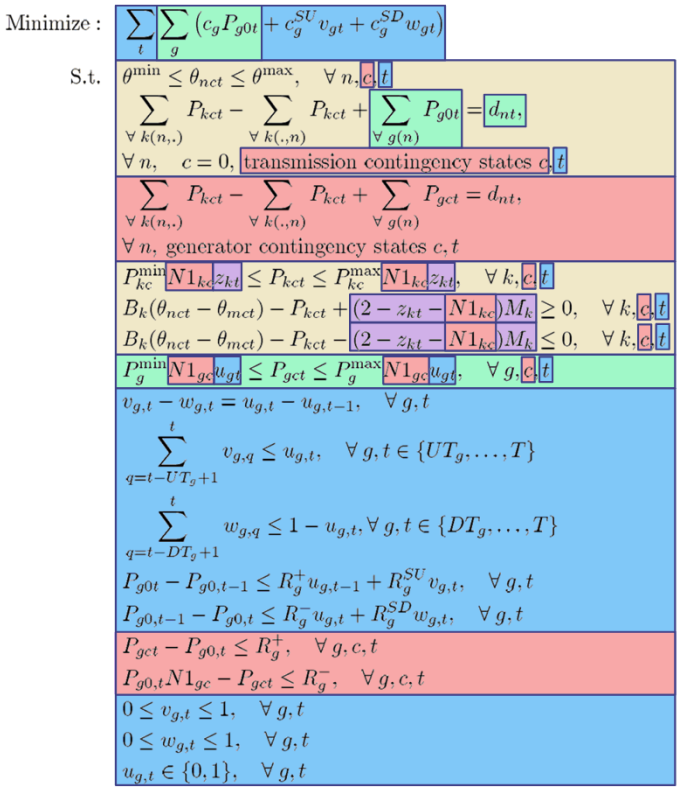
Embed within structured model





Expressing & preserving modeler intent

- If the *N-1* model has all this structure, why do we write it



- What I want:
  - Clear, structured syntax
  - Explicit disjunctions
  - Express block structure
    - Define meaningful components
    - “bus,” “line,” “generator”
  - Control how this gets mapped to the solver

What do these have in common?

$$a = \sqrt{(x-3)^2 + \epsilon}$$

$$\begin{aligned} a &= b + c \\ b &\leq M \cdot y \\ c &\leq M(1-y) \\ x - 3 &= c - b \\ b &\geq 0 \\ c &\geq 0 \\ y &\in \{0,1\} \end{aligned}$$

$$\begin{aligned} a &\geq x - 3 \\ a &\geq 3 - x \end{aligned}$$

$$\begin{aligned} a &= b + c \\ x - 3 &= c - b \\ b \geq 0 &\perp c \geq 0 \end{aligned}$$

$$a = \frac{2(x-3)}{1 + e^{-\frac{x-3}{h}}} - x + 3$$

What do these have in common?

$$a = \sqrt{(x-3)^2 + \epsilon}$$

$$\begin{aligned} a &= b + c \\ b &\le M \cdot y \\ c &\le M(1-y) \\ x - 3 &= c - b \\ b &\ge 0 \\ c &\ge 0 \\ y &\in \{0,1\} \end{aligned}$$

$$a = abs(x-3)$$

$$\begin{aligned} a &\ge x - 3 \\ a &\ge 3 - x \end{aligned}$$

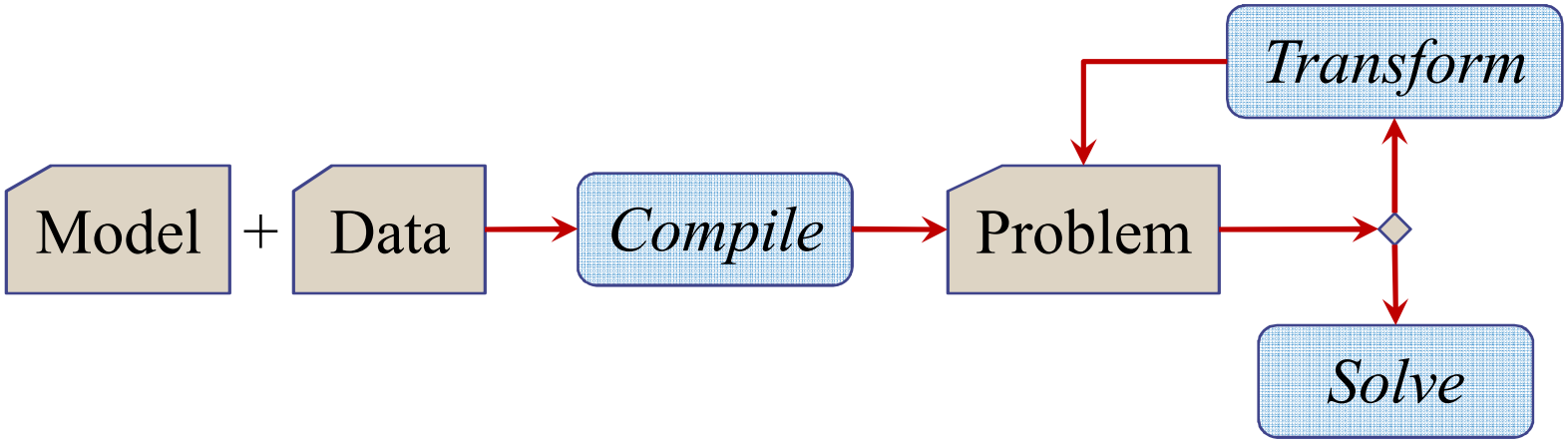
$$\begin{aligned} a &= b + c \\ x - 3 &= c - b \\ b \ge 0 &\perp c \ge 0 \end{aligned}$$

$$a = \frac{2(x-3)}{1 + e^{-\frac{x-3}{h}}} - x + 3$$

If we *mean* "$a = abs(x-3)$",
why don't we *write* that in our models???

A new solution workflow

- Model Transformations: *Projecting problems to problems*
  - Project from one problem space to another
  - Standardize common reformulations or approximations
  - Convert "unoptimizable" modeling constructs into equivalent optimizable forms



A transformation-centric view of *abs()*

- Chaining transformations

$$f = abs(x) \Rightarrow \begin{array}{c} f = x^+ + x^- \\ x = x^+ - x^- \\ x^+ \geq 0 \perp x^- \geq 0 \end{array} \Rightarrow \begin{array}{c} f = x^+ + x^- \\ x = x^+ - x^- \\ \begin{bmatrix} Y \\ x^- = 0 \end{bmatrix} \vee \begin{bmatrix} \neg Y \\ x^+ = 0 \end{bmatrix} \\ x^+ \geq 0, x^- \geq 0 \end{array} \Rightarrow \begin{array}{c} f = x^+ + x^- \\ x = x^+ - x^- \\ x^- \leq My \\ x^- \leq M(1-y) \\ x^+ \geq 0, x^- \geq 0 \end{array}$$

```
model = ConcreteModel()
# […]
TransformFactory("abs.complements").apply(model, inplace=True)
TransformFactory("mpec.disjunctive").apply(model, inplace=True)
TransformFactory("gdp.bigm").apply(model, inplace=True)
```

Why are we interested in transformations?

Sandia National Laboratories

- Separate model expression from how we intend to solve it
  - Defer decisions that improve tractability until solution time
  - Explore alternative reformulations or representations
  - Support *solver-specific* model customizations (e.g., `abs()`)
  - Support iterative methods that use different solvers requiring different representations (e.g., initializing NLP from MIP)
- Support "higher level" or non-algebraic modeling constructs
  - Express models that are "closer" to reality, e.g.:
    - Piecewise expressions
    - Disjunctive models (switching decisions & logic models)
    - Differential-algebraic models (dynamic models)
    - Bilevel models (game theory models)
- Reduce "mechanical" errors due to manual reformulation

CCR Center for Computing Research

Why transformations in Pyomo?

Sandia National Laboratories

- Pyomo is an *object model*
  - Extensions declare new object classes (*components*)
    - Supports annotating model components
  - Transformations can detect presence of relevant components
  - Core code (e.g., problem writers) can validate supported components
  - Whole model (including expressions) is transparent and manipulatable

- Pyomo natively supports hierarchical models
  - "Block": collection of modeling components (e.g., Sets, Prams, Vars)
    - Namespacing: component names must only be unique within a block
    - Blocks can contain blocks: hierarchical structure
  - Many modeling extensions derive from Block
  - Transformations can be "sandboxed" in transformation-specific Blocks

CCR Center for Computing Research

Expressing disjunctive programming

- Disjunctions: selectively enforce sets of constraints
  - Sequencing decisions: x ends before y or y ends before x
  - Switching decisions: a process unit is built or not
  - Alternative selection: selecting from a set of pricing policies

- Implementation: leverage Pyomo "blocks"
  - `Disjunct`:
    - Block of Pyomo components
      - (Var, Param, Constraint, etc.)
    - Boolean (binary) indicator variable determines if block is enforced
  - `Disjunction`:
    - Enforces logical XOR across a set of Disjunct indicator variables
  - (Logic constraints on indicator variables)

$$\bigvee_{i \in D_k} \begin{bmatrix} Y_{ik} \\ h_{ik}(x) \le o \\ c_k = \gamma_{ik} \end{bmatrix}$$
$$\Omega(Y) = true$$

Solving disjunctive models

- Few solvers "understand" disjunctive models
  - *Transform* model into standard math program
  - Big-M relaxation:
    - Convert logic variables to binary
    - Split equality constraints in disjuncts into pairs of inequality constraints
    - Relax all constraints in the disjuncts with "appropriate" M values
      - Automatically calculate M values for linear expressions
  - Convex hull relaxation (Balas, 1985; Lee and Grossmann, 2000)
    - Disaggregate variables in all disjuncts
    - Bound disaggregated variables with Big-M terms

Scripting complex transformations

[with M. Sharifzadeh, F. Trespalacious]

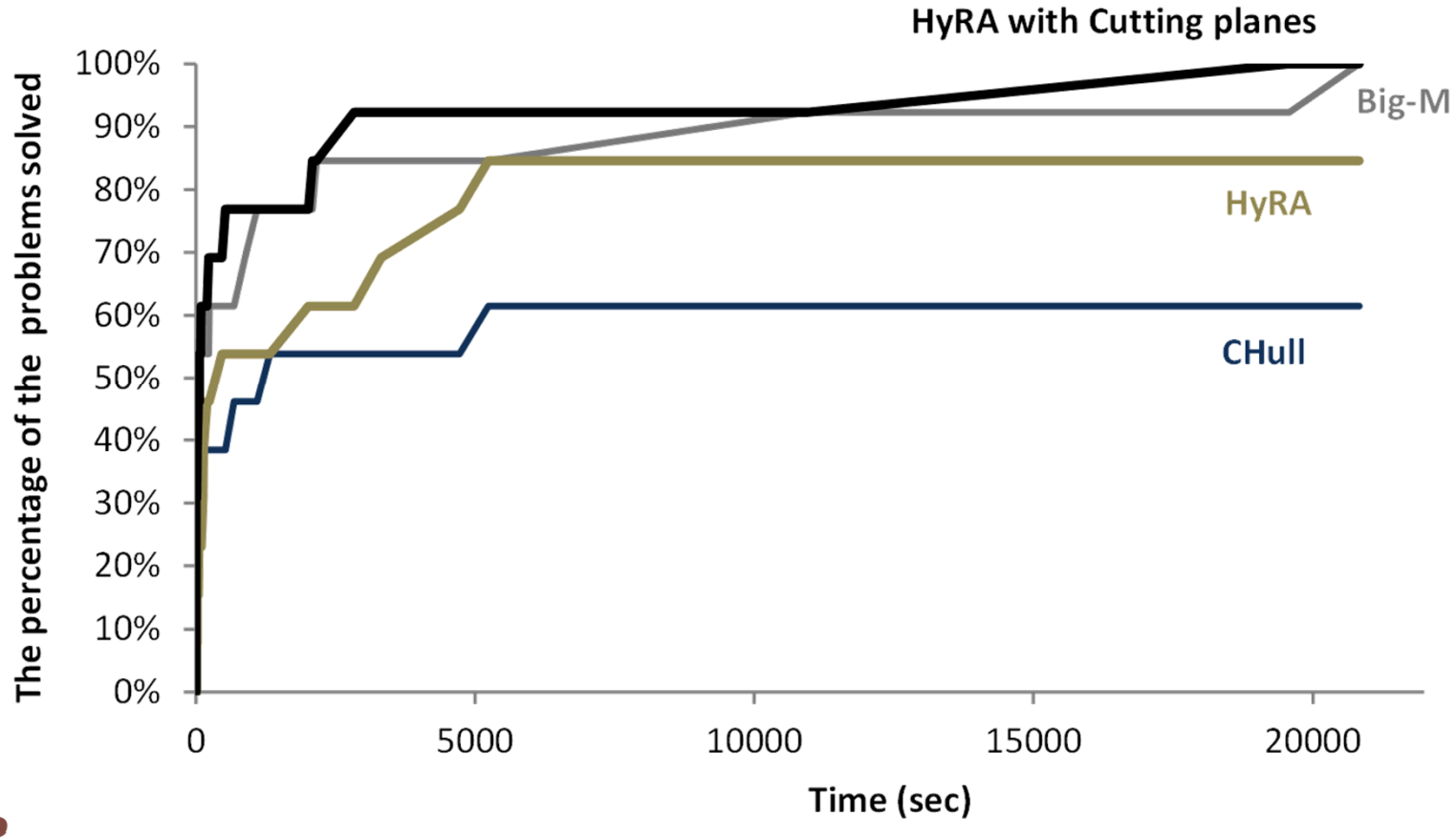
- What is the key challenge in disjunctive programming?
  - Big-M: small formulation ↔ weak LP relaxation
  - Convex Hull: tight(er) relaxation ↔ larger formulation
- Idea: apply preprocessing to probe the model and customize the transformation on a disjunction-by-disjunction basis
  - Key operation: *Basic Step* [Balas, 1985]

$$\begin{bmatrix} Y_1 \\ A_1x \le 0 \end{bmatrix} \veebar \begin{bmatrix} Y_2 \\ A_2x \le 0 \end{bmatrix} \\ \begin{bmatrix} Y_3 \\ A_1x \le 0 \end{bmatrix} \veebar \begin{bmatrix} Y_4 \\ A_1x \le 0 \end{bmatrix} \rightarrow \begin{bmatrix} Y_1 \wedge Y_3 \\ A_1x \le 0 \\ A_3x \le 0 \end{bmatrix} \veebar \begin{bmatrix} Y_1 \wedge Y_4 \\ A_1x \le 0 \\ A_4x \le 0 \end{bmatrix} \veebar \begin{bmatrix} Y_2 \wedge Y_3 \\ A_2x \le 0 \\ A_3x \le 0 \end{bmatrix} \veebar \begin{bmatrix} Y_2 \wedge Y_4 \\ A_2x \le 0 \\ A_4x \le 0 \end{bmatrix}$$

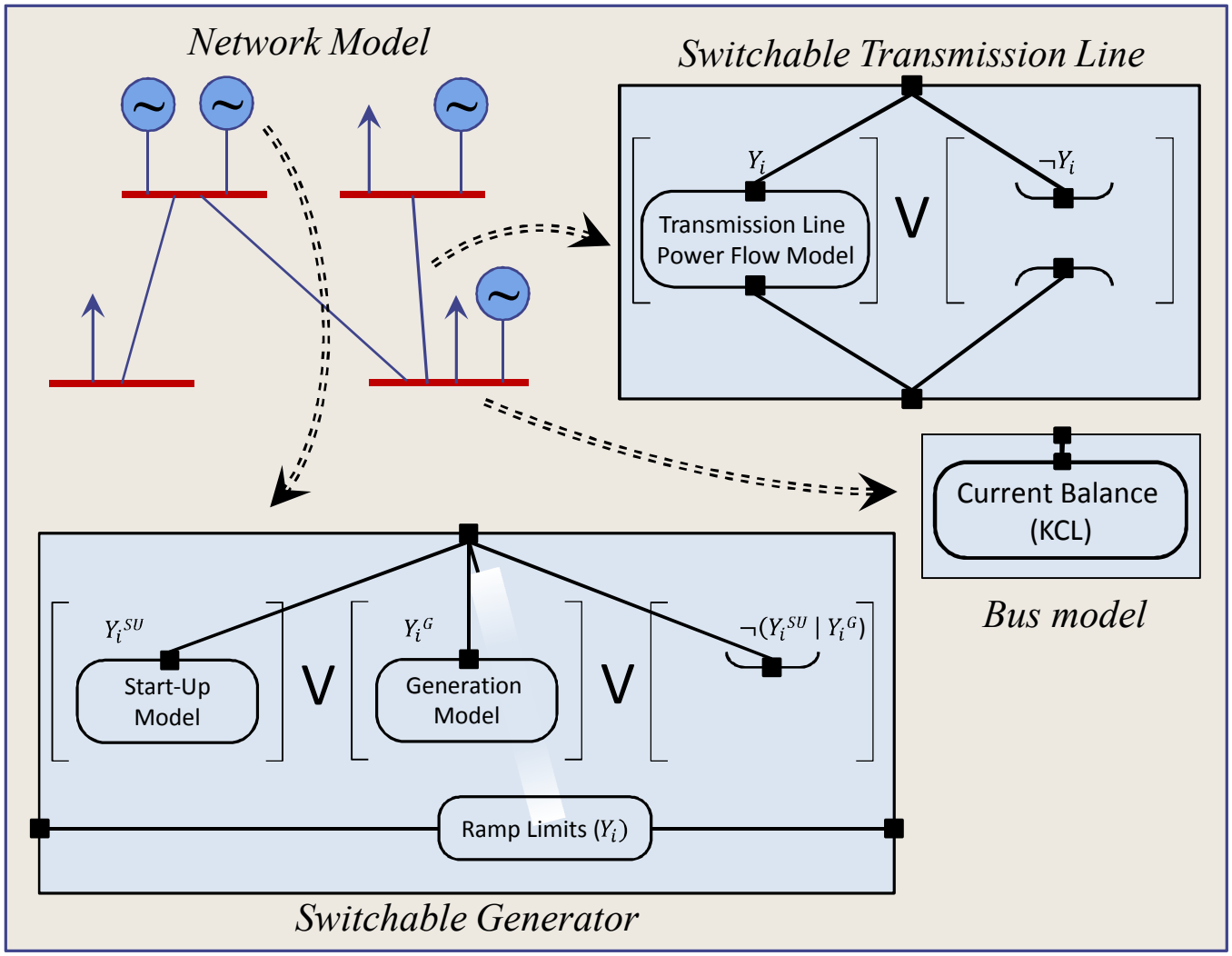
  - Approach 1: [Trespalacios and Grossmann, 2014]
    - Identify *key disjunctions* to combine using basic steps, relax with convex hull
    - Relax the balance of the disjunctions with big-M
  - Approach 2: [Trespalacios and Grossmann, *In Press*]
    - Identify *key disjunctions* as before, but apply basic steps to a *copy* of the model
    - Solve a series of separation problems between the big-M relaxation of the original model the convex hull relaxation of the modified model + basic steps.
      - Add separation cuts to the original model

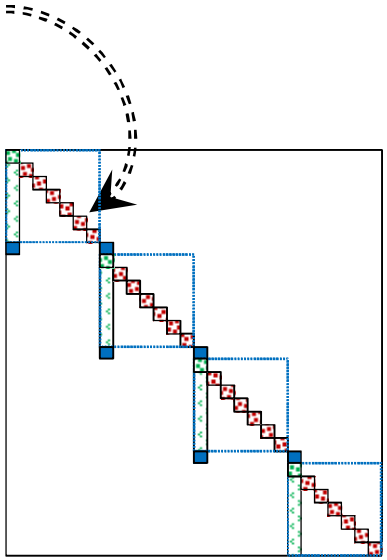
Impact on solution time

- Performance profiles for a family of constrained layout, strip packing, and process synthesis test problems from minlp.org



UC + N-1 + Switching model





Optimal Solution of RTS-96

Sandia National Laboratories

- From Hedman, et al. 2010
  - N-1 UC solution: 3,245,997
  - N-1 UC w/ Switching: 3,125,185 (2 pass UC+switching heuristic)

	Rows	Columns	Binaries
Raw model	5,118,760	1,501,177	5,184
After presolve	2,634,851	1,062,290	4,476

- Restructured problem (complete N-1 UC w/ switching):

	Rows	Columns	Binaries
Raw model	21,232,224	13,129,692	3,796,830
After presolve	2,471,714	1,249,976	187,194

  - Solution (1e-4 gap): 2,990,004 (60,000 sec)
  - Automated Big-M relaxation (including automatic M calculation)
  - Default CPLEX settings

CCR Center for Computing Research

Improving PH: borrowing from Benders

- Consider the Benders "feasibility" cut:
  - Given x^* computed by the RMP, if subproblem is unbounded, add a cut determined by an extreme ray in the dual space to the RMP
- In PH, a similar operation would be fix the values of x in subproblem f_j to the values computed by subproblem f_i:

$$\min_y f_j(x^{i*}, y)$$

  - If the problem is infeasible, then we can solve a separation problem (in the primal space) to determine a valid cut in the 1st-stage variables:

$$\min_{x,y} \|x - x^{i*}\|^2$$
$$s.t. \quad \hat{f}_j(x, y)$$

  - Notes:
    - $\hat{f}_j$ is the continuous relaxation of f_j → not guaranteed to generate a cut
    - The resulting cut is valid for *all* scenario subproblems
    - $\sum p_i f_i(x^{i*}, y^{i*})$ for initial scenario solves ($w = 0$) gives Lagrangian bound

Case study: UC + *N-1* analysis

- 2-stage unit commitment model for the electric power grid
  - 24-hour horizon, 1-hour commitment intervals
  - Explicitly include *N-1* analysis (loss of any 1 generator / non-radial line)
    - Each contingency modeled as a no-cost recourse scenario

- Case 1: 5 busses, 7 generators (13 scenarios):
  - Optimal solution (extensive form): 19.9756
  - Default PH ($\rho = 1$):
    - 17 iterations, objective = 22.9997, total time 123 seconds
  - PH ($\rho = 1$) + Feasibility cuts:
    - 3 feasibility cut iterations at PH iteration 0
      - Improved Lagrangian bound from 19.7 to 19.909
    - 13 iterations, objective = 23.14, total time 1300 seconds

Improving PH: setting ρ

- How do we get "good" values of ρ?
  - Currently: experimentation
  - Challenge: ρ is problem dependent
    - Too low and PH never converges
    - Too high and PH rapidly converges to suboptimal solution
  - Hint: scale relative to cost of each variable [Watson & Woodruff, 2011]

- We can get good cost estimates from the subproblem duals
  - When we evaluate $f_j(x^{i*}, y)$, record the duals for $x = x^{i*}$
  - Compute average duals weighted relative to scenario probability
- Adds two new scalar tuning parameters:
  - Dual scaling factor and a change damping factor

Case studies: UC + *N-1* analysis

- Recall:
  - Optimal solution (extensive form): 19.9756
  - Default PH ($\rho = 1$):
    - 17 iterations, objective = 22.9997, total time 123 seconds
  - PH ($\rho = 1$) + Feasibility cuts:
    - 3 feasibility cut iterations at PH iteration 0
      - Improved Lagrangian bound from 19.7 to 19.909
    - 13 iterations, objective = 23.14, total time 1300 seconds

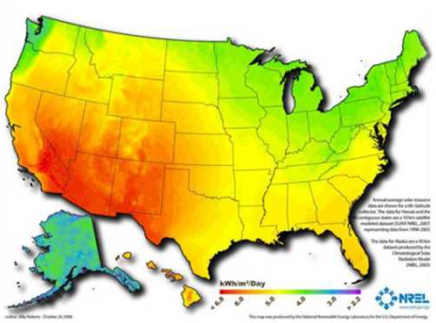
- Now:
  - PH + Feasibility cuts + ρ setter (rhoScale = 0.5)
    - 12 iterations, objective = 19.9814, total time 1060 seconds

Power system expansion planning

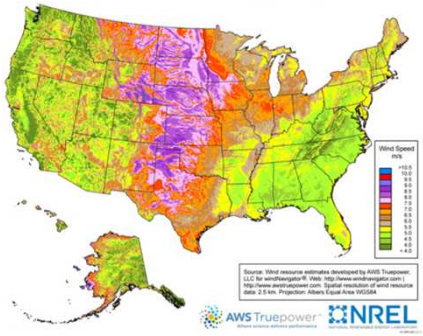
Power system expansion planning

[F. Munoz, J.-P. Watson]

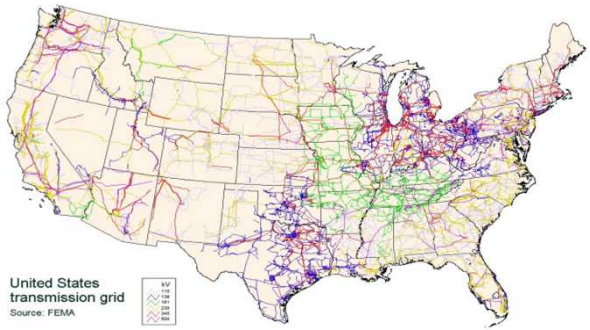
Solar Resources (NREL)



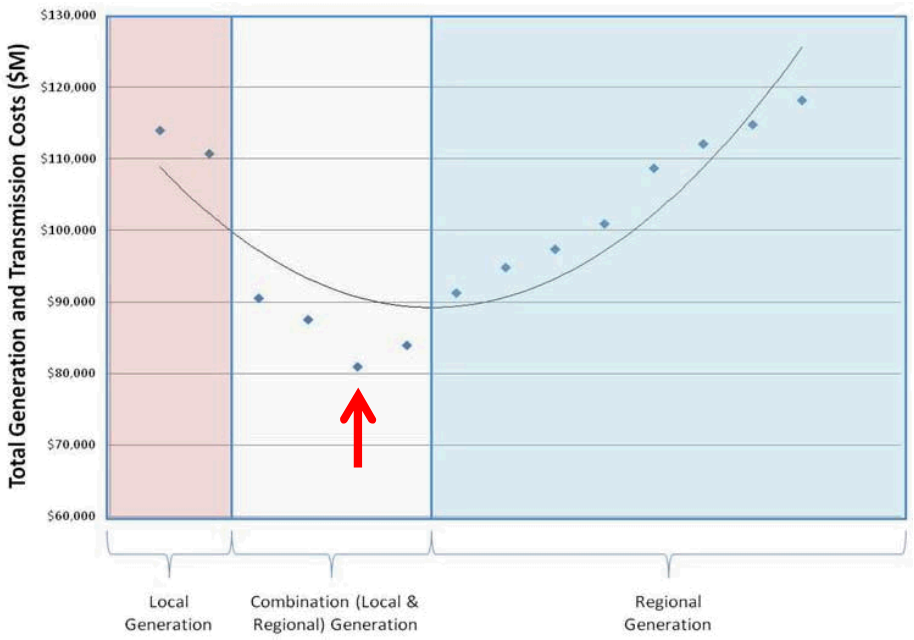
Wind Resources (NREL)



U.S. Transmission System (FEMA)



Zone Scenario Generation and Transmission Cost (MISO, 2010)



Goal:

Identify most cost effective combination of transmission and generation investments to meet:

1) Forecasted demand
2) Renewable and environmental goals

CCR Center for Computing Research

Tackling planning *at scale*

Sandia National Laboratories

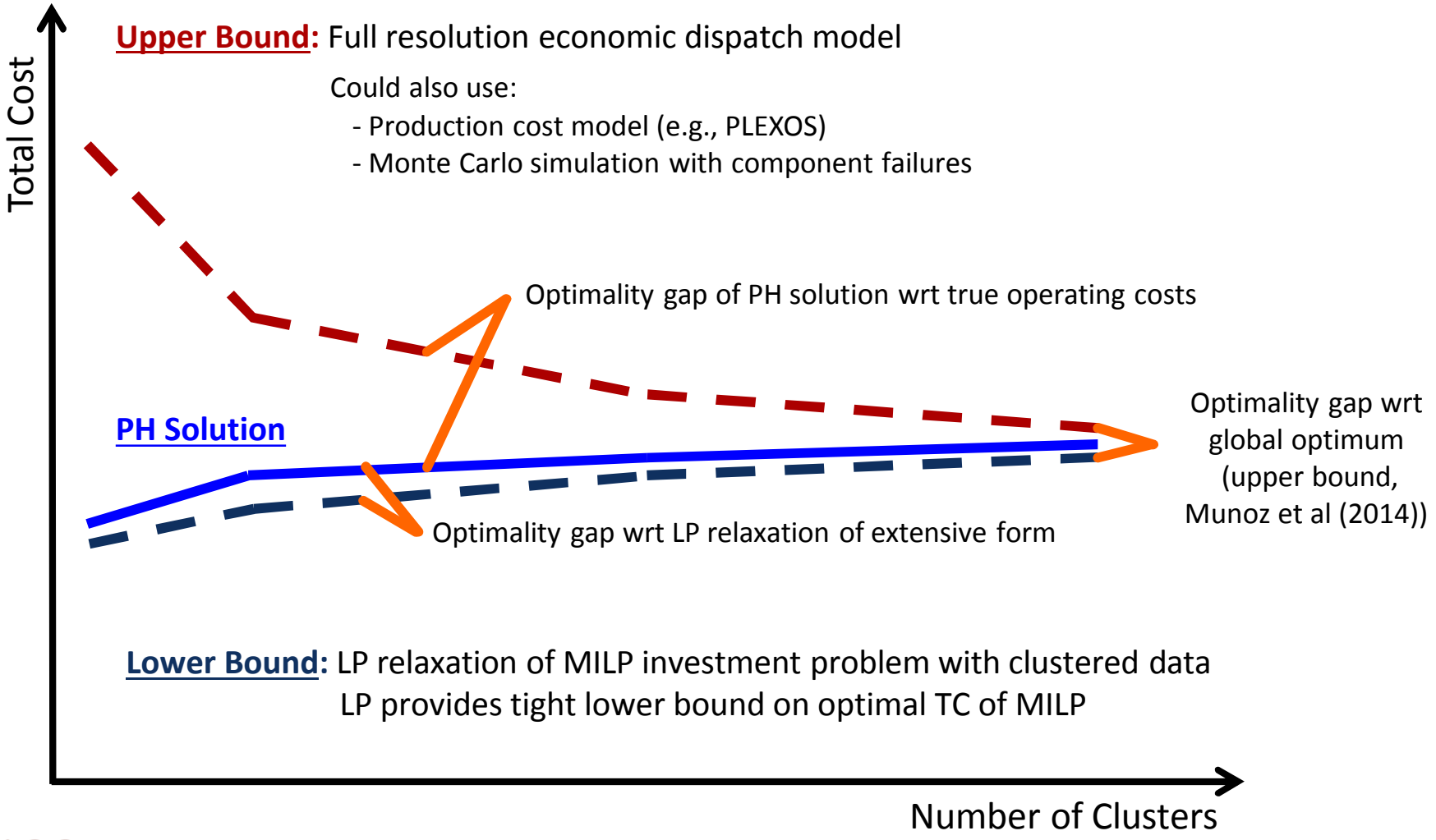
[F. Munoz, J.-P. Watson]

- Reported literature results
  - Often applied to **small test cases**
  - Usually consider only a **few scenarios** (often just one)
  - Exception: Munoz et al (2014) solved WECC 240-bus system using Benders decomposition.
    - Considered 8,736 scenarios, **87 hours to attain a 2.4% optimality gap**.

- Investigate scenario reduction and progressive hedging
  - 2-stage planning model:
    - Investments: Generation (continuous) + Transmission (binary)
    - Operations: Economic dispatch (DCOPF) + Soft constraint enforcing RPS
  - Scenario reduction
    - k-means clustering
    - How many clusters (scenarios) do we really need?

CCR Center for Computing Research

Assessing Solution Quality



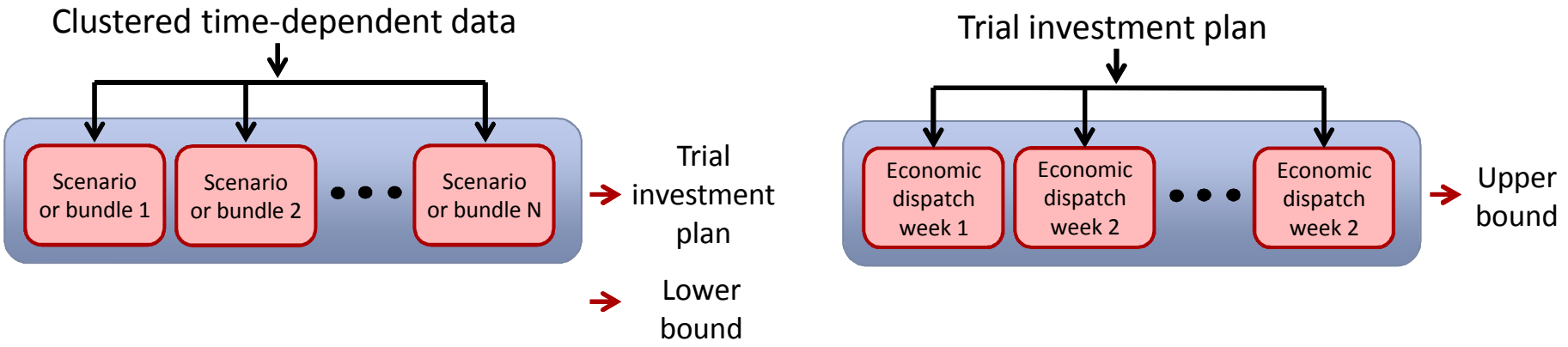
Experiments: WECC-240 system

Description

- Dataset of 8,736 historical observations of load, wind, solar, and hydro levels for year 2004
  - Results in ~15M variables and ~35M constraints
- 257 generation investment variables (continuous)
- 339 variables for transmission backbones (binary)
- 31 variables for interconnections to renewable hubs (integer)

Our Hardware Environments

- Red Sky/Red Mesa HPC: 43,440 cores of Intel Xeon series processors, 64TB of RAM (12 GB per node)
- Multi-Core SMP Workstation: 48-core Intel Xeon, 2.3 GHz, 512 GB RAM (~$20K)



Computational Performance

Sandia National Laboratories

Extensive form, 100 scenarios

- CPLEX, no feasible solution after 1 day on a 48-core workstation

Progressive Hedging, 100 scenarios

- **Red Mesa:** ~15 minutes, 186 iterations until full convergence of investment variables
- **Workstation:** ~31 minutes, 180 iterations until full convergence of investment variables

(1) UB from investment cost PH + true operating cost : $582.7B

(2) Expected cost from PH : $565.7B

(3) LB from solving extensive form LP relaxation : $555.4B

Gap UB = 2.9%

Gap LP = 1.82%

Gap LP => How suboptimal is the solution found using PH w.r.t. LP relaxation (not zero!!)
Gap UB => Difference between operating costs using clustered vs. full dataset

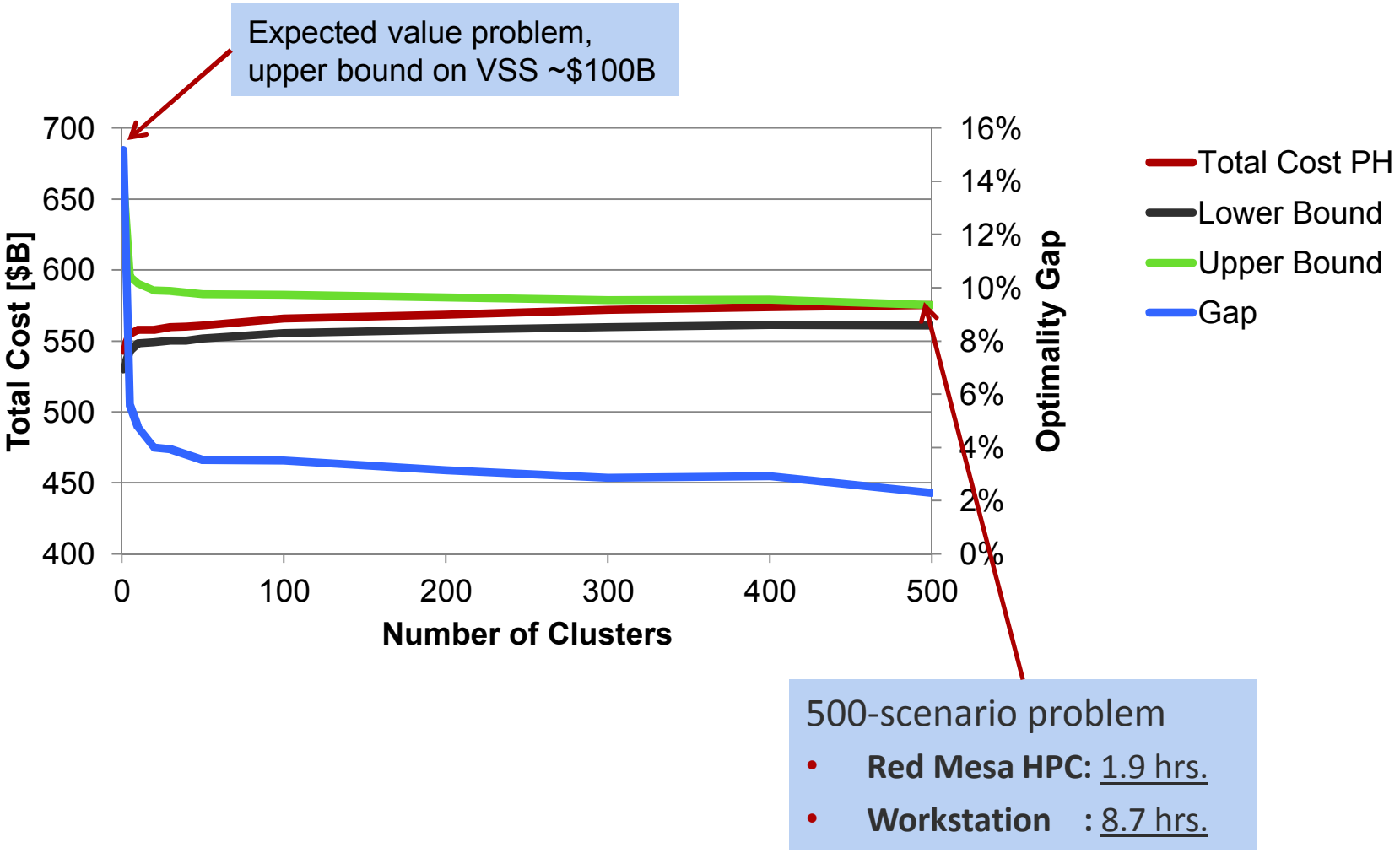
Total Gap => 3.5% (w.r.t. best lower bound)

CCR Center for Computing Research

Computational Performance

Sandia National Laboratories

Convergence of upper and lower bounds



CCR Center for Computing Research

Summary

- Pyomo is a full-featured, extensible, open source AML
  - …and we always welcome new users
- Pyomo enables our applied OR research
  - Transformation-centric approach to modeling
  - Extensions to new (or non-algebraic) modeling paradigms
    - …stochastic, dynamic, disjunctive
  - Development of new solution approaches
    - Decomposition-based algorithms
    - Problem analysis, cut generation, hybrid algorithms
- Pyomo is the vehicle through which we impact applications
  - Stochastic unit commitment at scale
  - Operations with explicit contingency analysis
  - System expansion planning

Thank you!

- For more information…
- Project homepages
  - http://www.pyomo.org
  - http://software.sandia.gov/pyomo
- User mailing lists
  - pyomo-forum@googlegroups.com
- "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)