Sandia National Laboratories

Exceptional service in the national interest

Accelerating Optimization of Stochastic Programs via Multifidelity Modeling and Bundle-based Decomposition

Rachael M Alfant, PhD

Discrete Math & Optimization, Center for Computing Research

rmalfan@sandia.gov

ICERM - Asynchronous Methods for Numerical Linear Algebra

May 7 2026

U.S. DEPARTMENT of ENERGY | NNSA

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

SAND2026-20706C

Collaborators

LDRD LABORATORY DIRECTED RESEARCH & DEVELOPMENT

Bill Hart[1]

Zach Kilwein[1]

Matthew Viens[1,2]

J. Kyle Skolfield[1]

Georgia Stinchfield[3]

[1] Sandia National Labs, [2] University of Wisconsin-Madison, [3] Carnegie Mellon University

Outline

- Motivation: Power Systems Planning
- Multifidelity Scenario Bundling
- Stochastic Programming And Related Optimization Workflows (SPAROW)
- Summary & Future Research Directions

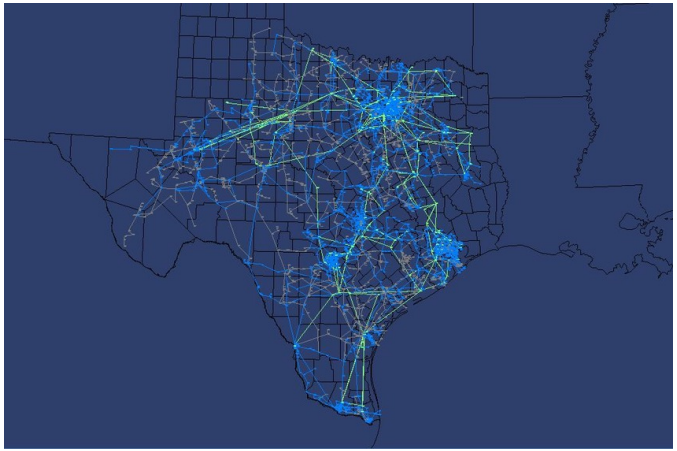
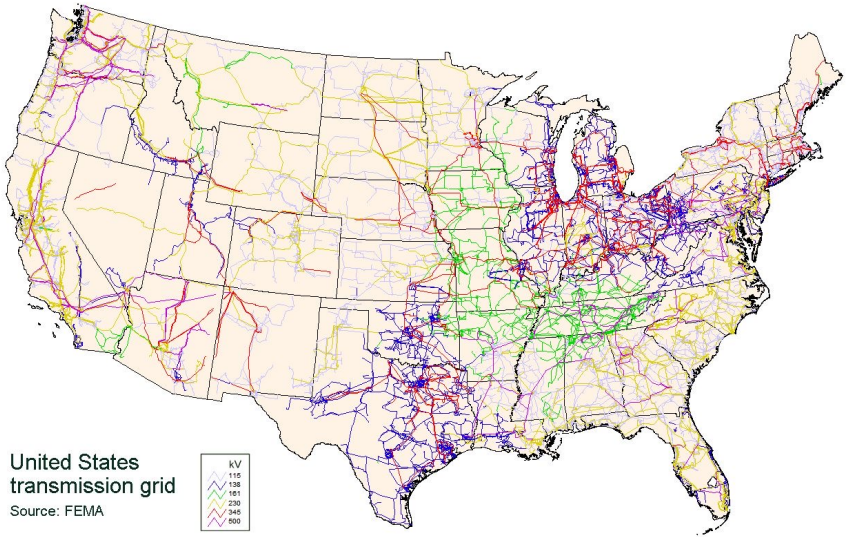


Image: Li, Nan, et al. "Optimal generation investment planning: Pt. 2: Application to the ERCOT system." 2012 North American Power Symposium. IEEE, 2012.

Power grids transport electricity from power plants to customers

*"The greatest engineering achievement of the 20th century"**

- ***>642,000 miles*** of high-voltage transmission lines
- ***>6.3 million miles*** of local distribution lines



Source: https://commons.wikimedia.org/wiki/File:UnitedStatesPowerGrid.jpg

* Constable, George, and Bob Somerville, eds. *A century of innovation: Twenty engineering achievements that transformed our lives.* Joseph Henry Press, 2003.

Recent events highlight vulnerabilities in critical infrastructure; importance of planning for resilience

L.A. Wildfires: Property Damage Estimated At Between $28B-$53.8B; Full Recovery Will Take At Least Until 2029

By Denise Petski, Tom Tapp
February 27, 2025 3:43pm

Hundreds of thousands without power in the U.S. after a powerful winter storm

UPDATED JANUARY 26, 2026 · 2:41 PM ET

Debbie Elliott

Capacity Expansion Planning: strategic long-term planning of investment decisions for power systems infrastructure

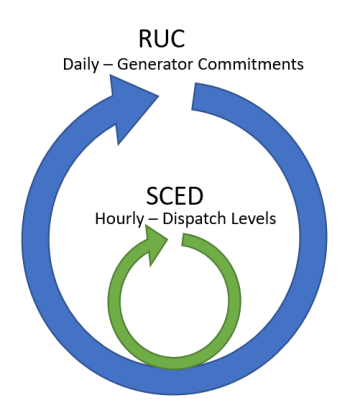
Capacity Expansion Planning: strategic long-term planning of investment decisions for power systems infrastructure

Investment Stage:

Quantity, type, timing of generator & line investments

Operational Stage:

Daily/hourly operating conditions



L. Rodriguez, "Challenges of integrating renewables into today's power grids". https://ratedpower.com/blog/challenges-integrating-renewables-ower-grid/
J.-P. Watson, B. Knueven, et. al. (2020). Prescient. https://doi.org/10.11578/dc.20201101.1

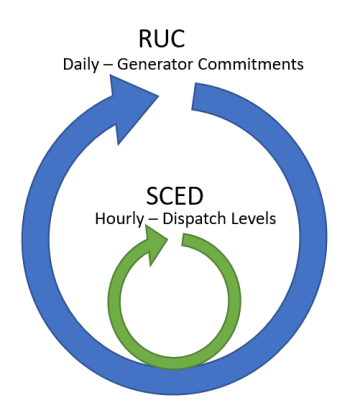
Capacity Expansion Planning: strategic long-term planning of investment decisions for power systems infrastructure

Investment Stage:

Quantity, type, timing of generator & line investments

Operational Stage:

Daily/hourly operating conditions



Solving full model to optimality is computationally challenging.

L. Rodriguez, "Challenges of integrating renewables into today's power grids". https://ratedpower.com/blog/challenges-integrating-renewables-ower-grid/
J.-P. Watson, B. Knueven, et. al. (2020). Prescient. https://doi.org/10.11578/dc.20201101.1

Why is power grid planning so complex?

- Multiple sources of uncertainty.
  - Load, renewable energy generation, etc.

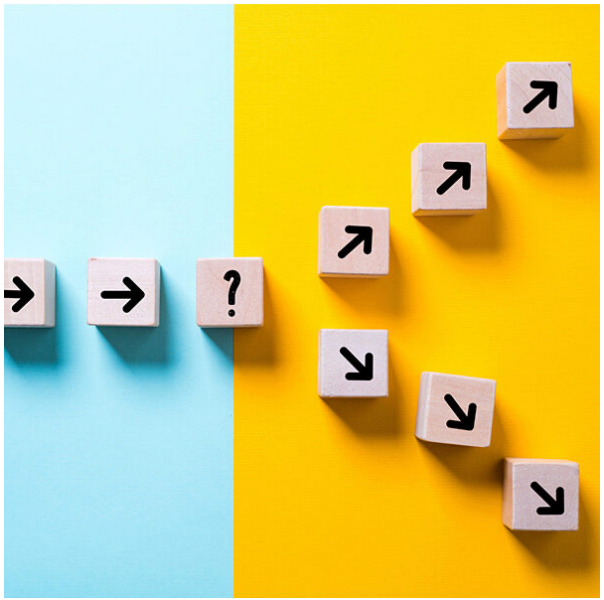
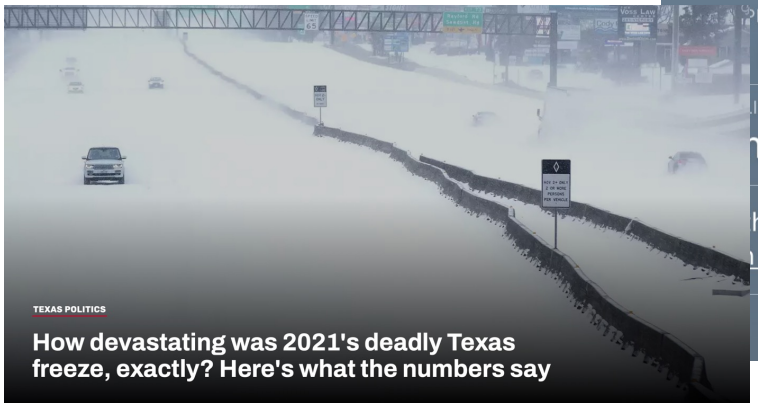


Image: https://executiveeducation.wharton.upenn.edu/thought-leadership/wharton-at-work/2021/08/decision-making-under-uncertainty/

Why is power grid planning so complex?

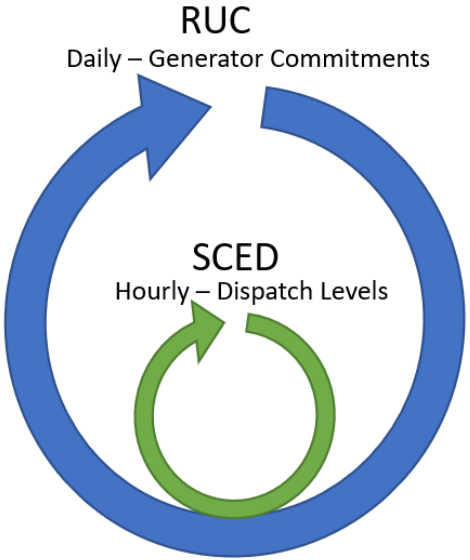
- Multiple sources of uncertainty.
  - Load, renewable energy generation, etc.
- Multiple potential vulnerabilities.
  - Deliberate cyber-attacks, extreme weather events.



https://www.houstonchronicle.com/politics/texas/article/How-Texas-leaders-let-winter-storm-freeze-state-16909408.php

Why is power grid planning so complex?

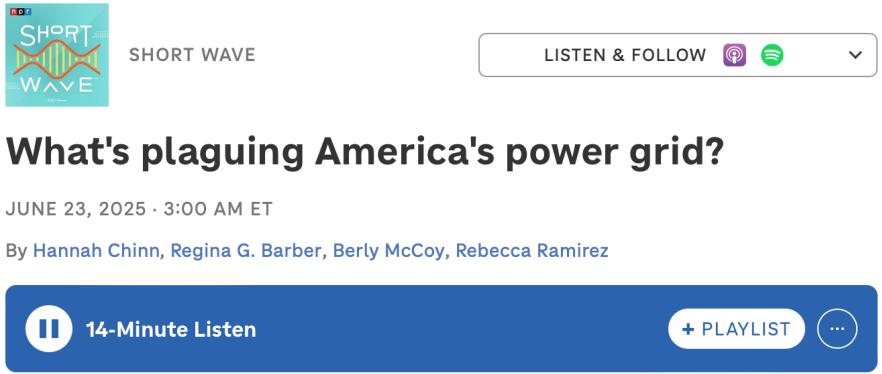
- Multiple sources of uncertainty.
  - Load, renewable energy generation, etc.
- Multiple potential vulnerabilities.
  - Deliberate cyber-attacks, extreme weather events.
- Multiple time scales.
  - Hourly, day-to-day, years out.



J.-P. Watson, B. Knueven, et. al. (2020). Prescient.
https://doi.org/10.11578/dc.20201101.1

Why is power grid planning so complex?

- Multiple sources of uncertainty.
  - Load, renewable energy generation, etc.
- Multiple potential vulnerabilities.
  - Deliberate cyber-attacks, extreme weather events.
- Multiple time scales.
  - Hourly, day-to-day, years out.
- Enormous number of components, interconnections, etc.



https://www.npr.org/2025/06/23/1254614614/weather-electricity-outage-electric-grid

How can power systems be optimized for efficiency and resiliency?

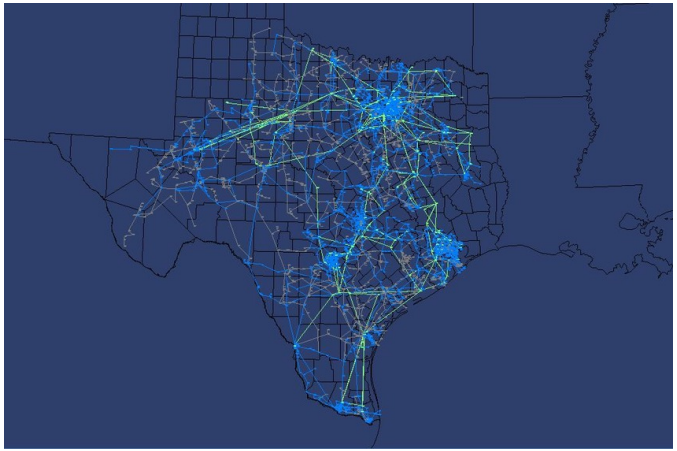


Image: Li, Nan, et al. "Optimal generation investment planning: Pt. 2: Application to the ERCOT system." 2012 North American Power Symposium. IEEE, 2012.

How can power systems be optimized for efficiency and resiliency?

rapid, high-quality, cost-minimizing solution

remain operational through high-impact, low frequency events

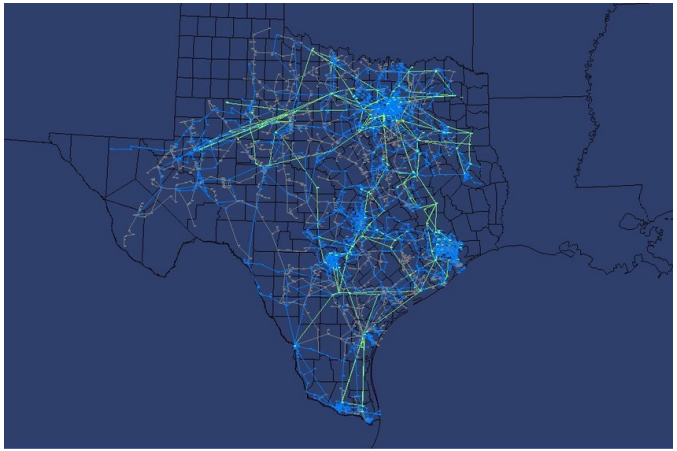
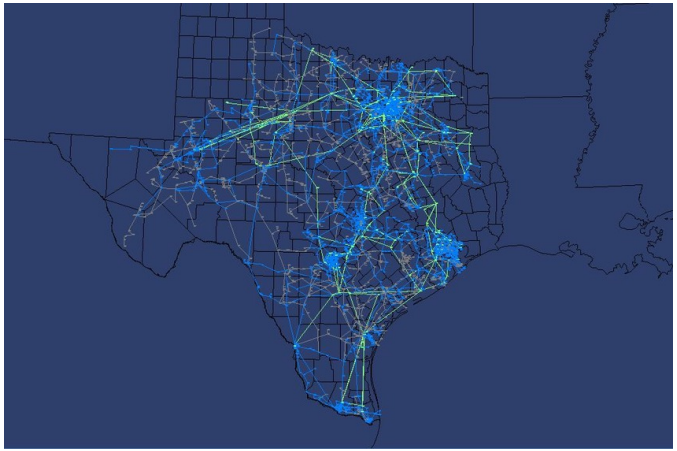


Image: Li, Nan, et al. "Optimal generation investment planning: Pt. 2: Application to the ERCOT system." 2012 North American Power Symposium. IEEE, 2012.

How can power systems be optimized for efficiency and resiliency?

rapid, high-quality, cost-minimizing solution

remain operational through high-impact, low frequency events



Hypothesis: Leveraging multifidelity modeling and scenario bundling can accelerate solutions to stochastic programs.

Image: Li, Nan, et al. "Optimal generation investment planning: Pt. 2: Application to the ERCOT system." 2012 North American Power Symposium. IEEE, 2012.

How can power systems be optimized for efficiency and resiliency?

- Motivation: Power Systems Planning
- Multifidelity Scenario Bundling
- Stochastic Programming And Related Optimization Workflows (SPAROW)
- Summary & Future Research Directions

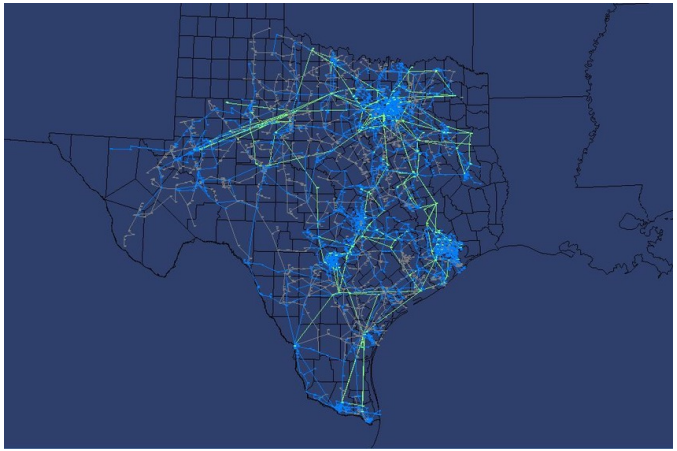
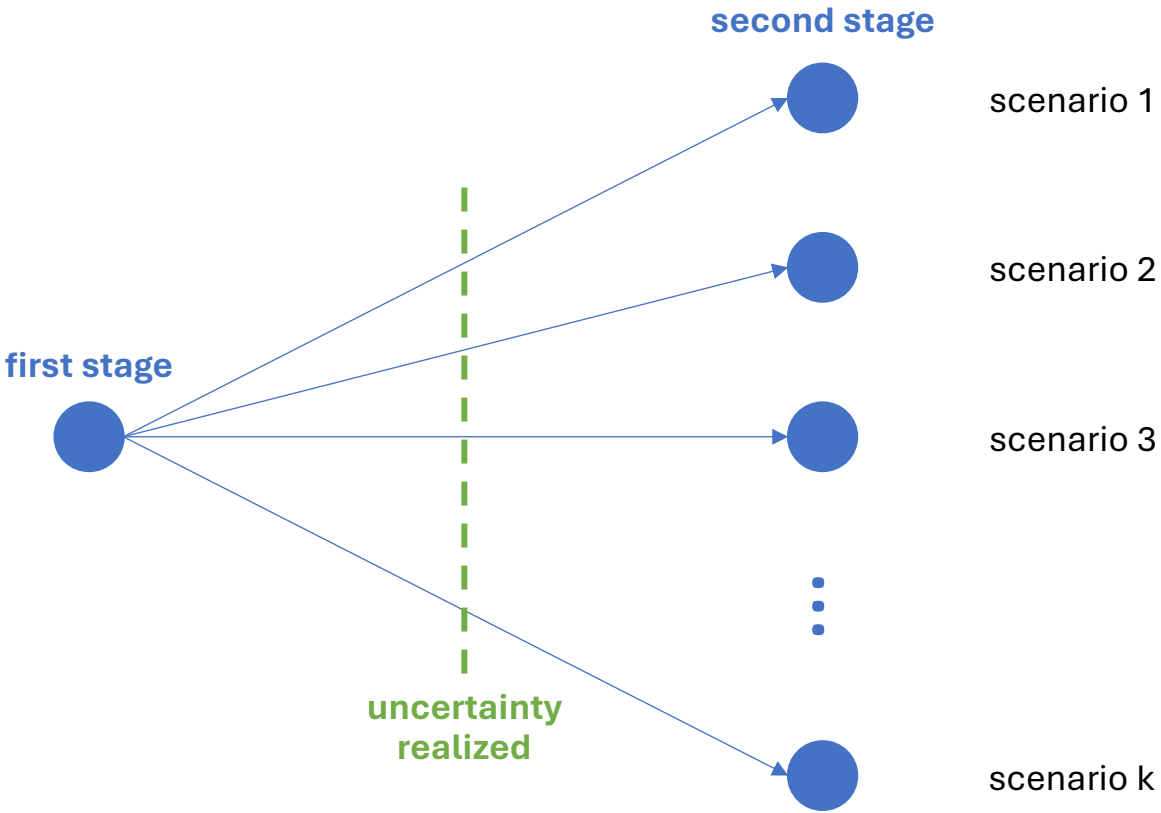
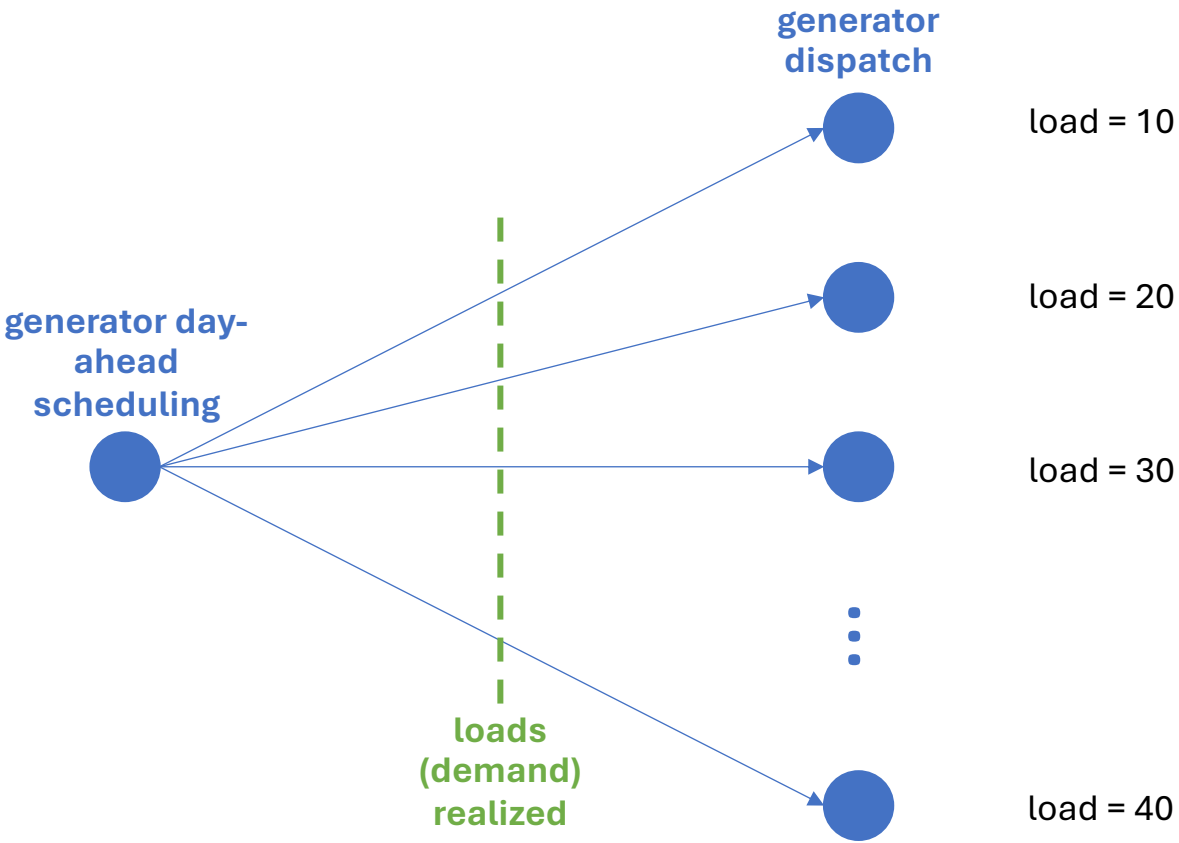


Image: Li, Nan, et al. "Optimal generation investment planning: Pt. 2: Application to the ERCOT system." 2012 North American Power Symposium. IEEE, 2012.

Accounting for Uncertainty: Stochastic Programming



Accounting for Uncertainty: Stochastic Programming



Modeling Operating Stage as a Stochastic Program

Minimize: $\sum_s \Pr(s) * cost$ (expected operating costs)
subject to:

Modeling Operating Stage as a Stochastic Program

Minimize: $\sum_s \Pr(s) * cost$ (expected operating costs)
subject to: *non-anticipativity,*
physical generator/line constraints,
balance load and generation.

Modeling Operating Stage as a Stochastic Program

Minimize: $\sum_s \Pr(s) * cost$ (expected operating costs)
subject to: *non-anticipativity,*
physical generator/line constraints,
balance load and generation.

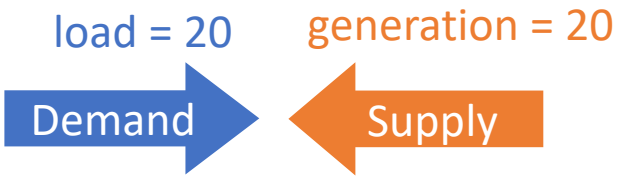
Deterministic

load = 20

Modeling Operating Stage as a Stochastic Program

Minimize: $\sum_s \Pr(s) * cost$ (expected operating costs)
subject to: *non-anticipativity,*
physical generator/line constraints,
balance load and generation.

Deterministic



Modeling Operating Stage as a Stochastic Program

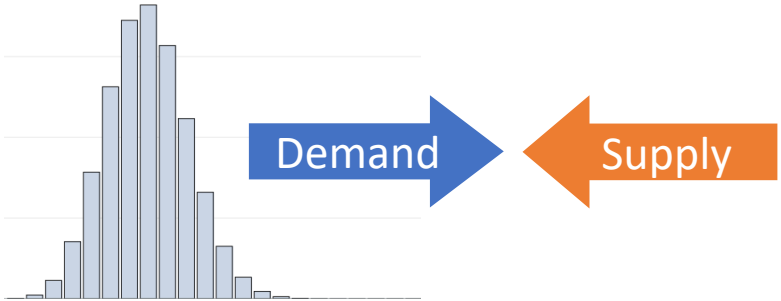
Minimize: $\sum_s \Pr(s) * cost$ (expected operating costs)
subject to: *non-anticipativity,*
physical generator/line constraints,
balance load and generation.

Deterministic

load = 20 → Demand

generation = 20 → Supply

Stochastic

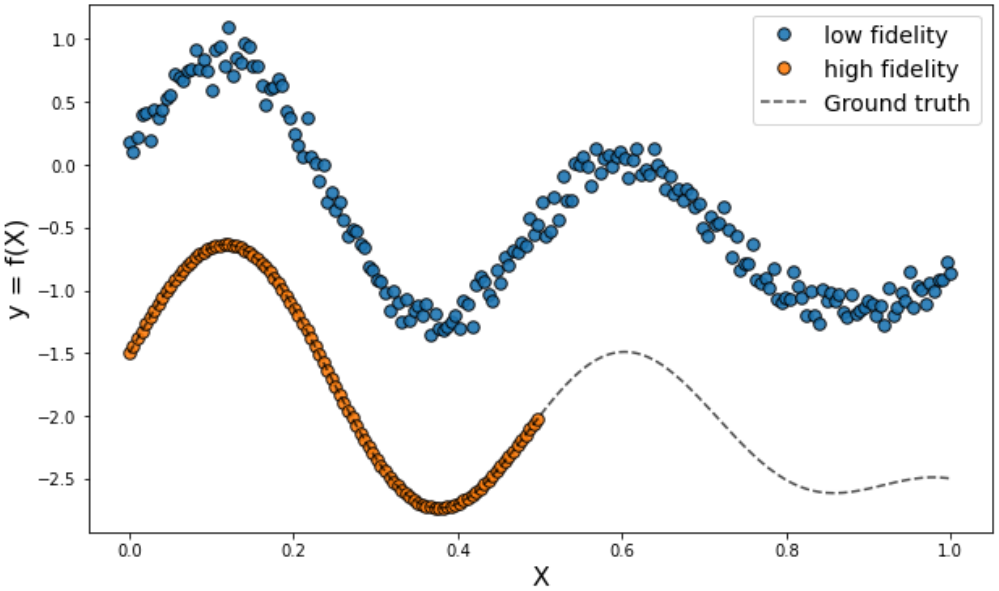


❖ Can be computationally expensive!

Image: https://blogs.sas.com/content/iml/2022/03/23/discrete-distributions.html

Multifidelity Modeling for Solving Stochastic Programs Efficiently

- Multiple models (varying fidelities) of the same problem.
- Balance tradeoff between accuracy and tractability.



https://adapt-python.github.io/adapt/examples/Multi_fidelity.html

Multifidelity Modeling for Stochastic Programming Applied to Power Systems

Visualization of Multi-Fidelity Approximations of Stochastic Economic Dispatch

Kinshuk Panda
kinshuk.panda@nrel.gov
Computational Science Center
National Renewable Energy
Laboratory
Golden, Colorado, USA

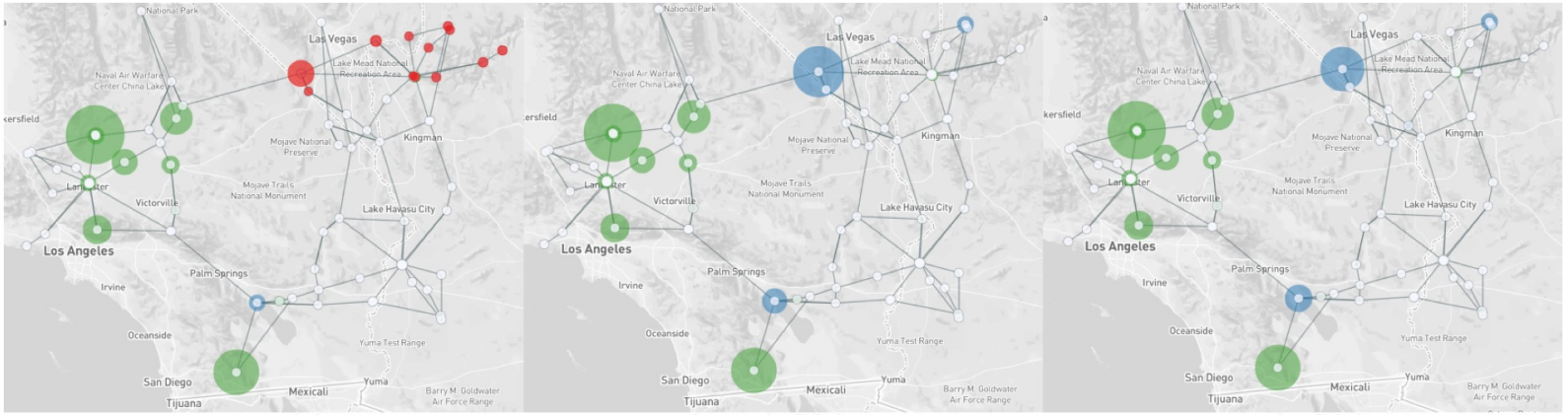
Ryan King
ryan.king@nrel.gov
Computational Science Center
National Renewable Energy
Laboratory
Golden, Colorado, USA

Jonathan Maack
jonathan.maack@nrel.gov
Computational Science Center
National Renewable Energy
Laboratory
Golden, Colorado, USA

Ignas Satkauskas
ignas.satkauskas@nrel.gov
Computational Science Center
National Renewable Energy
Laboratory
Golden, Colorado, USA

Kristi Potter
kristi.potter@nrel.gov
Computational Science Center
National Renewable Energy
Laboratory
Golden, Colorado, USA

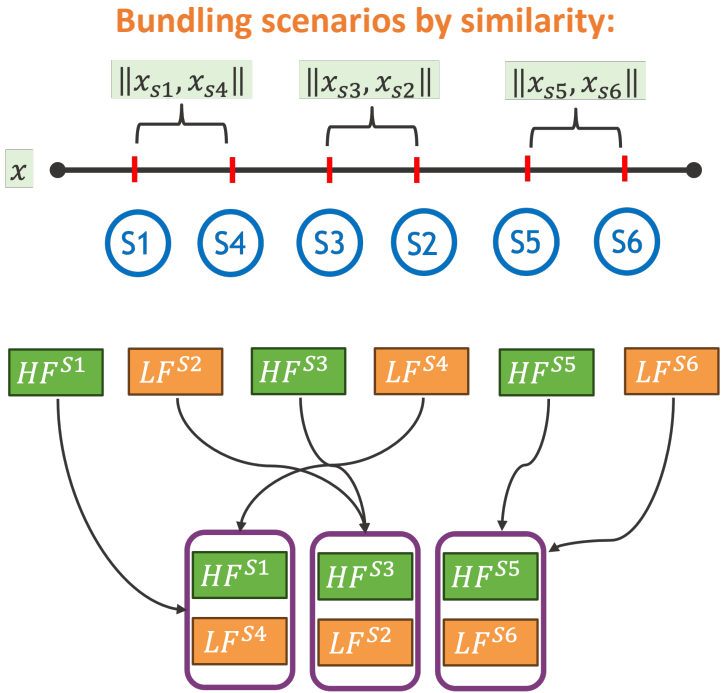
Figure 4. A single time-step of the (left to right) low-, high-, and multi- runs. Renewable generation is shown in green, thermal generation in blue, and loss of load in red.



a) Low-Fidelity b) High-fidelity c) Multi-fidelity

Scenario Bundling can increase tractability of stochastic programs*

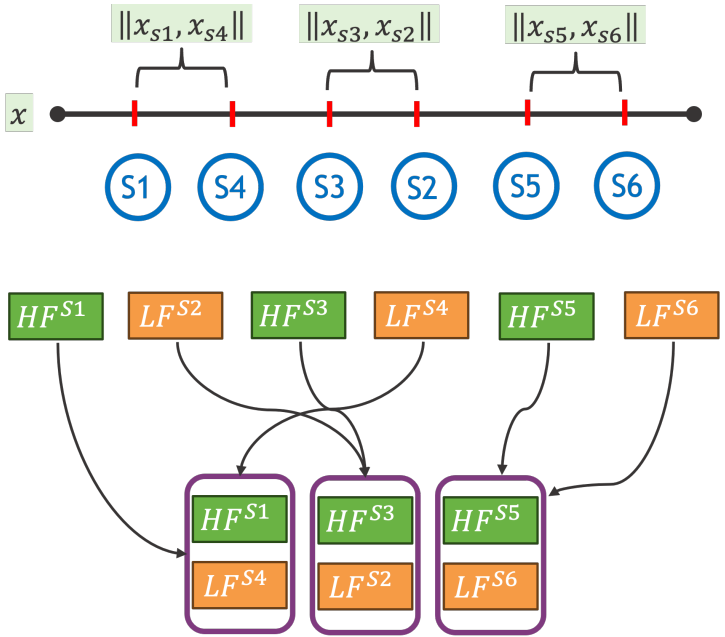
- **Decompose main problem into bundle-subproblems.**
  - Group scenarios into “bundles”.
  - Each subproblem more complex.
  - Decreases total number of subproblems.



* Escudero et. al. (2013), Crainic et. al. (2014), Jiang et. al. (2021) (see reference slides).

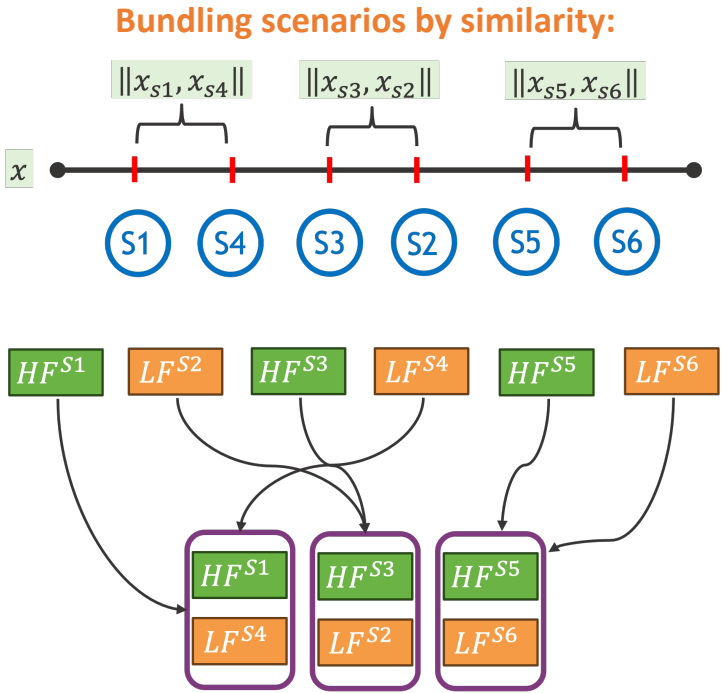
Idea: Leverage multifidelity scenario bundling to accelerate solutions

Bundling scenarios by similarity:

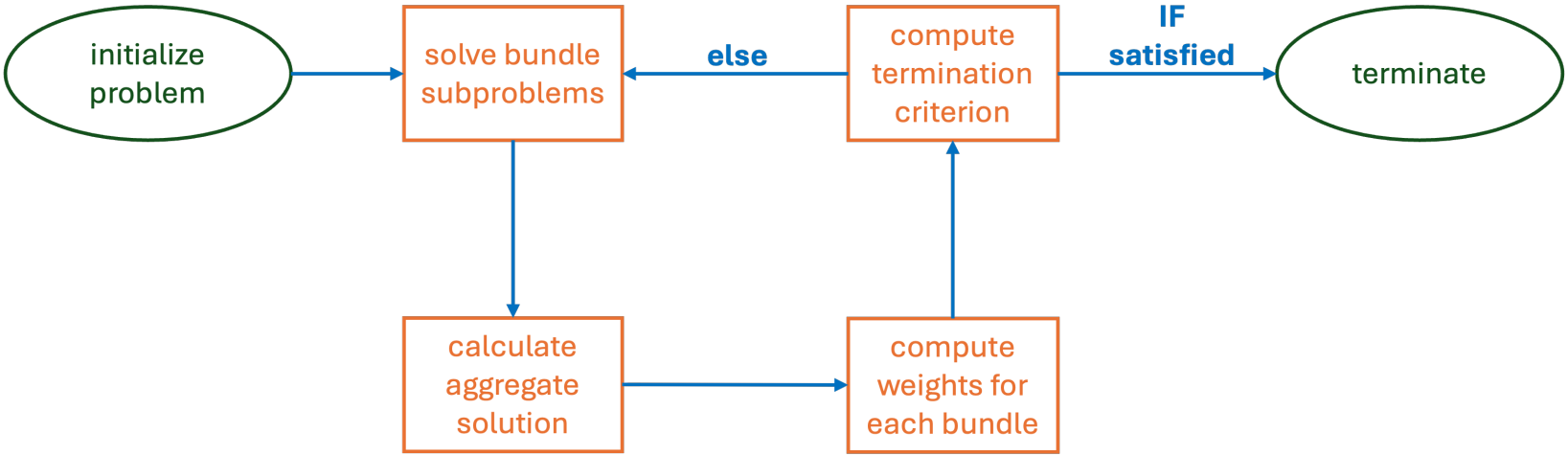


Idea: Leverage multifidelity scenario bundling to accelerate solutions

- Power systems planning problems have multiple models with varying fidelities.
- Sharing information across subproblems may make subproblem computations cheaper.



Progressive Hedging: SP algorithm used to decompose problem

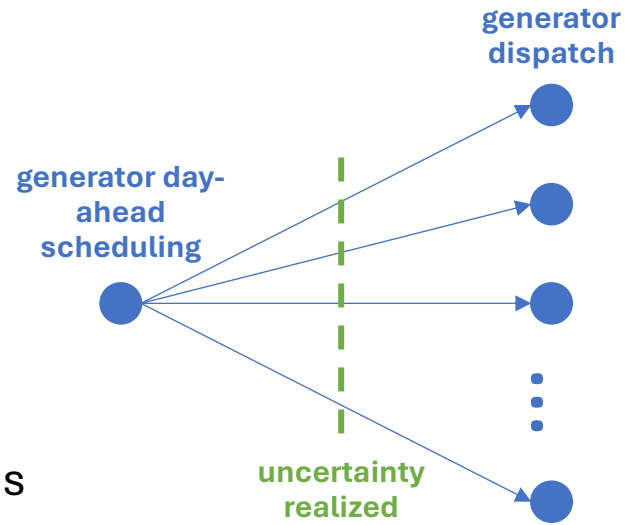


- Can integrate multifidelity scenario bundling into Progressive Hedging

Exemplar: Solving Operating Stage Problem

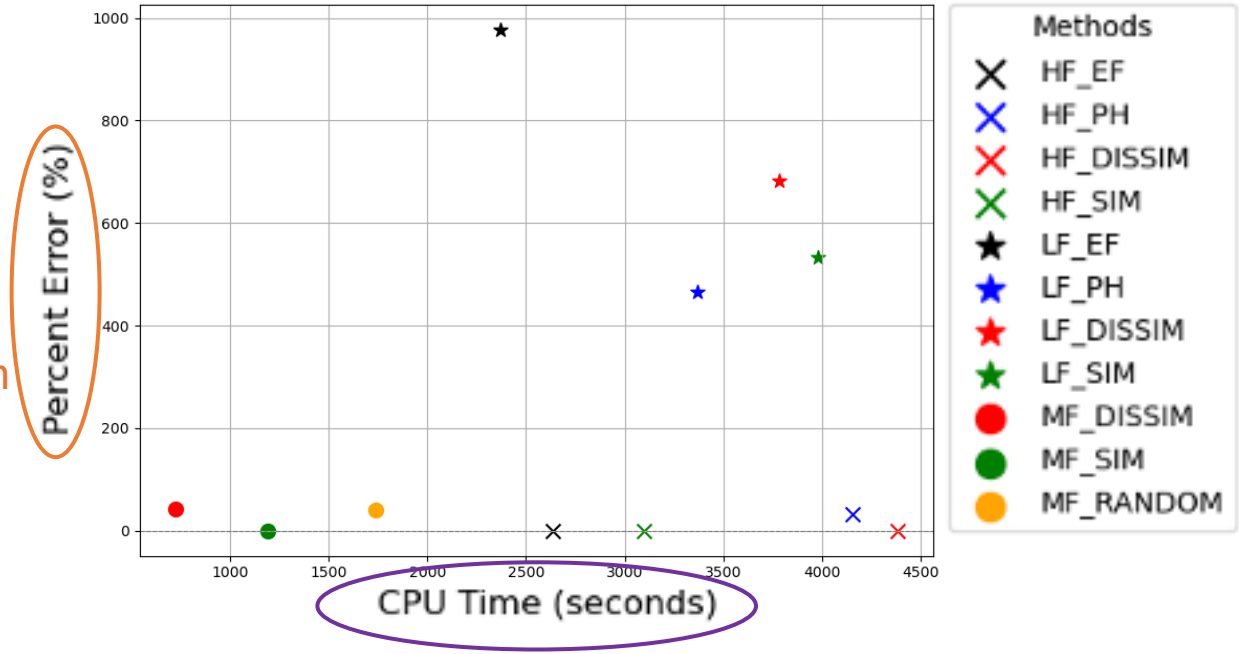
RTS-GMLC:

- 73 buses
- 158 generators
- 120 lines
- 48 hourly time periods
- 11275 binary, 25396 continuous variables



MF bundling outperforms all HF and LF schemes for November scenarios

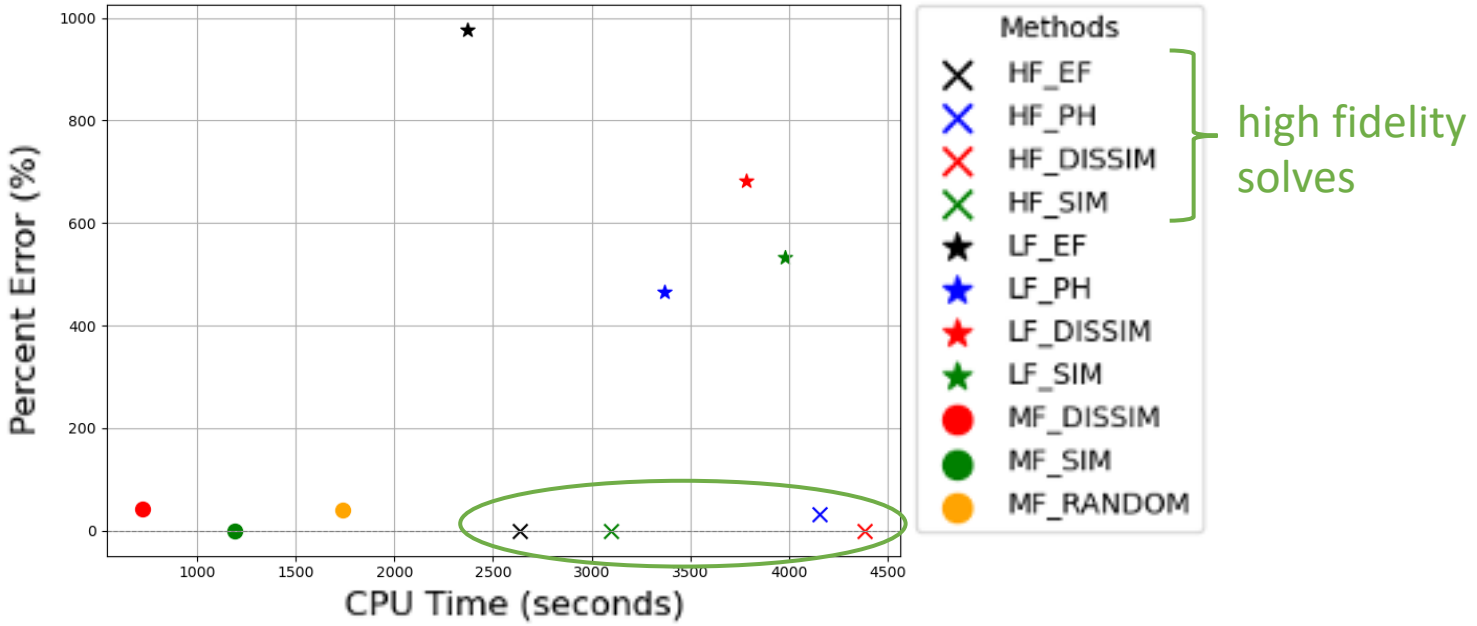
% difference from HF objective



solution time

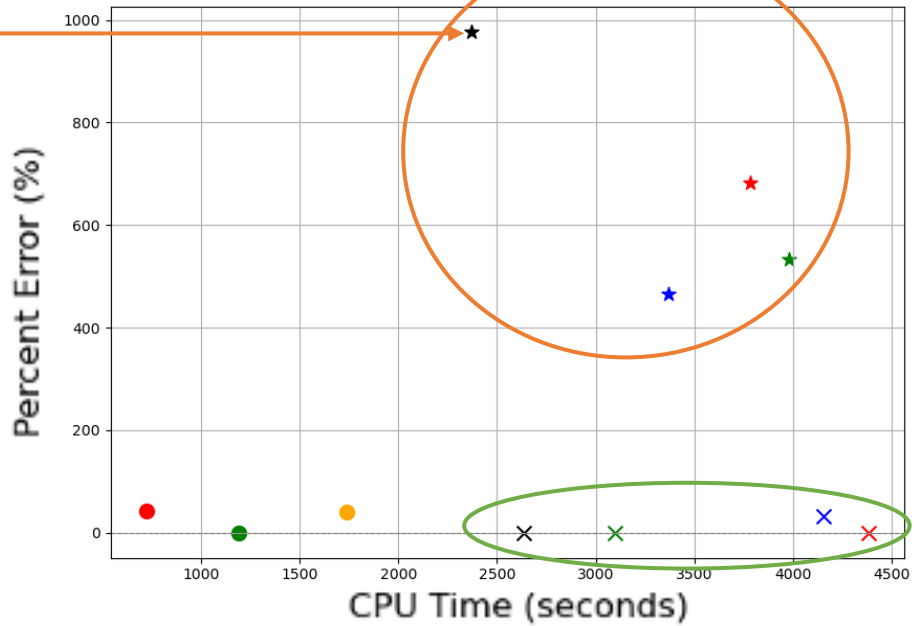
Z. Kilwein, **R. M. Alfant**, W. Hart. Accelerating Stochastic Unit Commitment with Multifidelity Scenario Bundling. *2025 57th North American Power Symposium (NAPS)*, p.1-6.

MF bundling outperforms all HF and LF schemes for November scenarios



MF bundling outperforms all HF and LF schemes for November scenarios

~1000% error



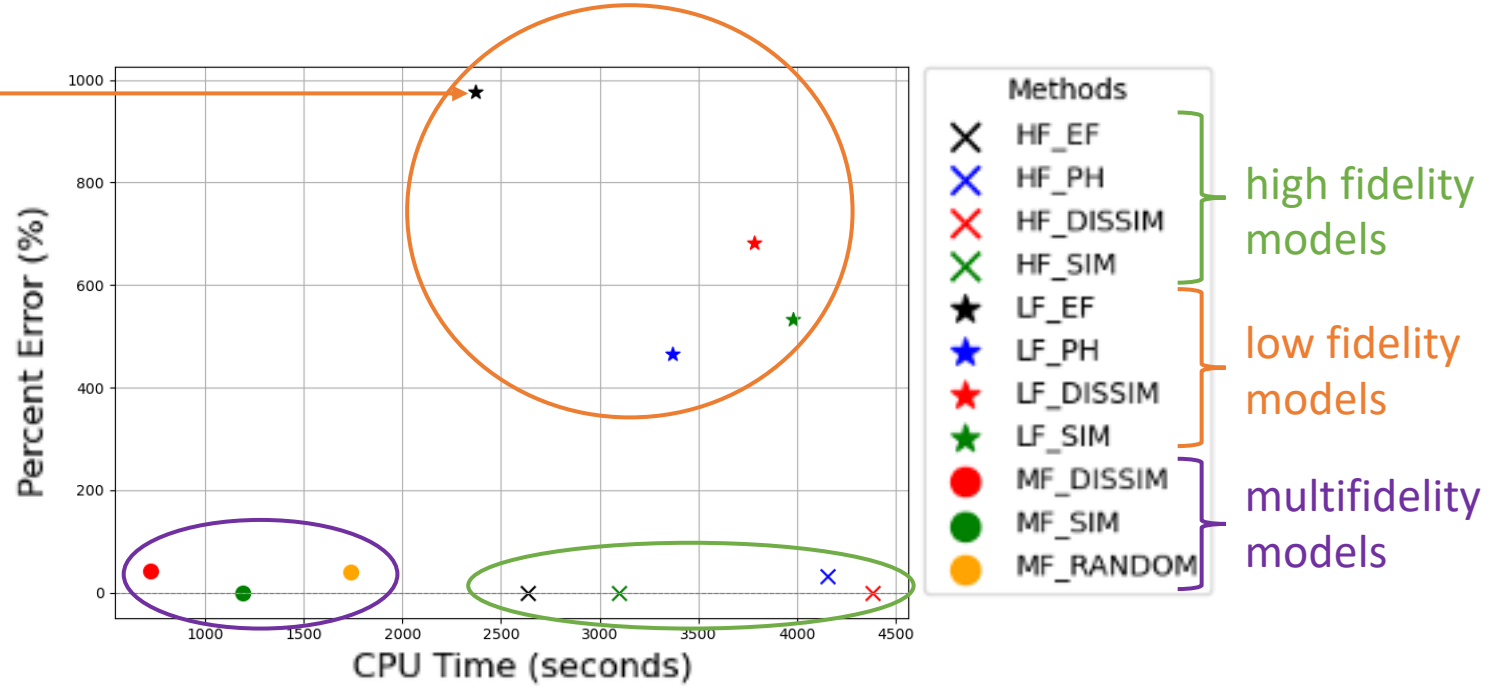
Methods
- HF_EF
- HF_PH
- HF_DISSIM
- HF_SIM
- LF_EF
- LF_PH
- LF_DISSIM
- LF_SIM
- MF_DISSIM
- MF_SIM
- MF_RANDOM

high fidelity models

low fidelity models

MF bundling outperforms all HF and LF schemes for November scenarios

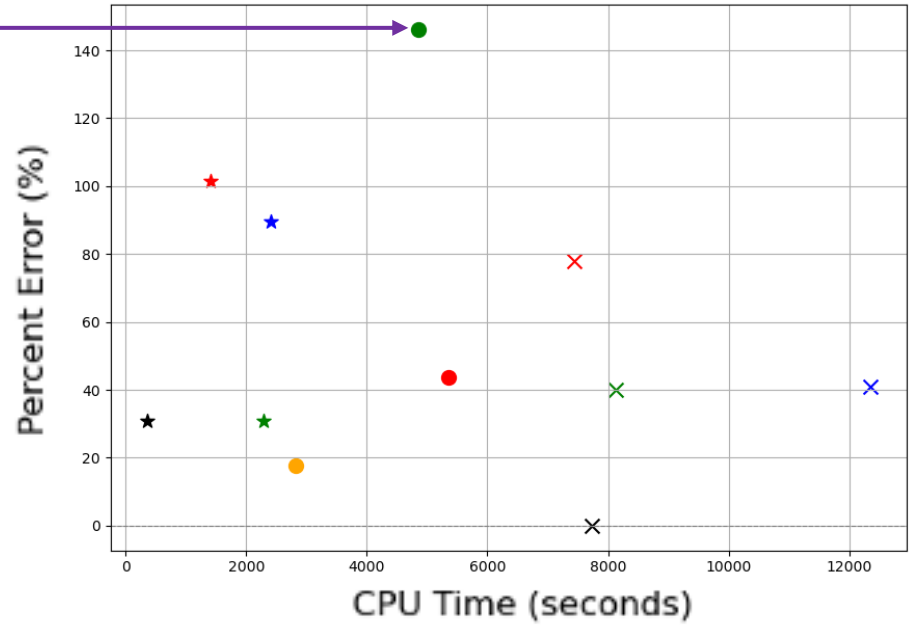
~1000% error



Z. Kilwein, **R. M. Alfant**, W. Hart. Accelerating Stochastic Unit Commitment with Multifidelity Scenario Bundling. *2025 57th North American Power Symposium (NAPS)*, p.1-6.

March scenarios: Some MF schemes are more accurate than others

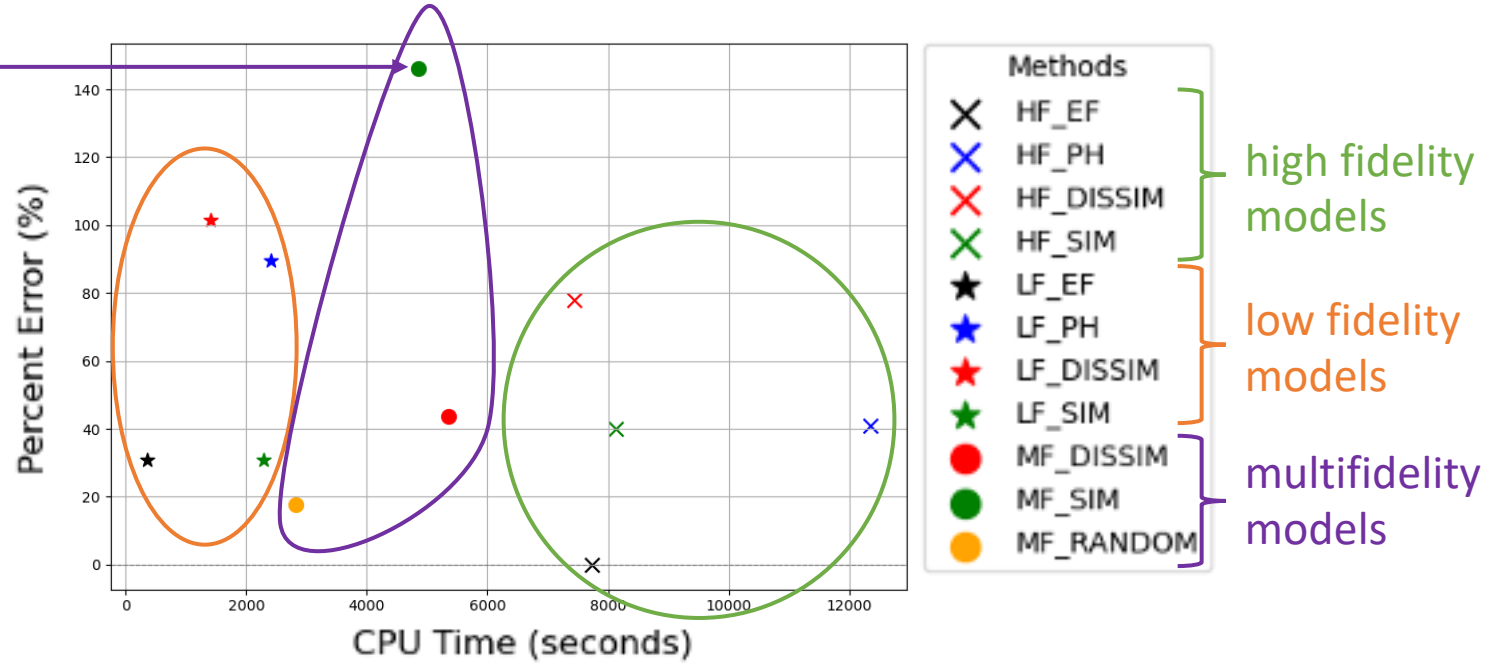
~150% error



Z. Kilwein, **R. M. Alfant**, W. Hart. Accelerating Stochastic Unit Commitment with Multifidelity Scenario Bundling. *2025 57th North American Power Symposium (NAPS)*, p.1-6.

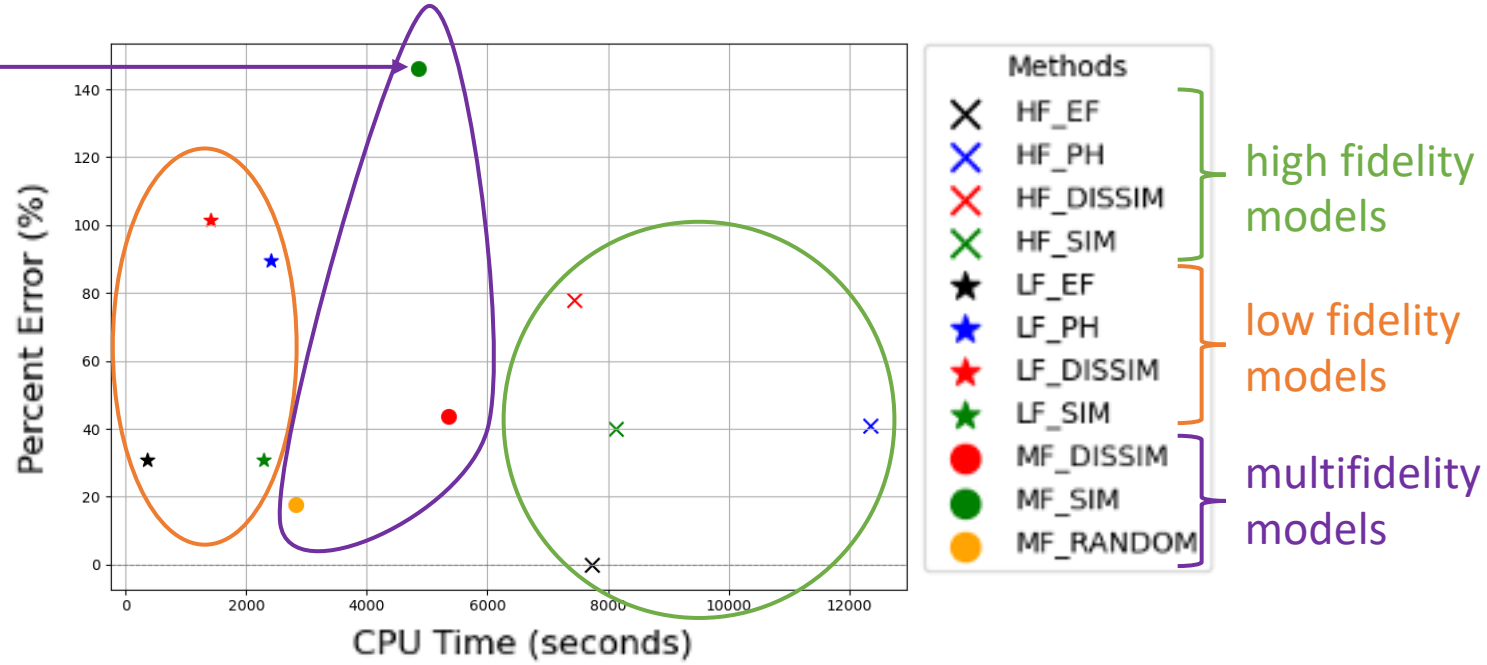
March scenarios: Some MF schemes are more accurate than others

~150% error



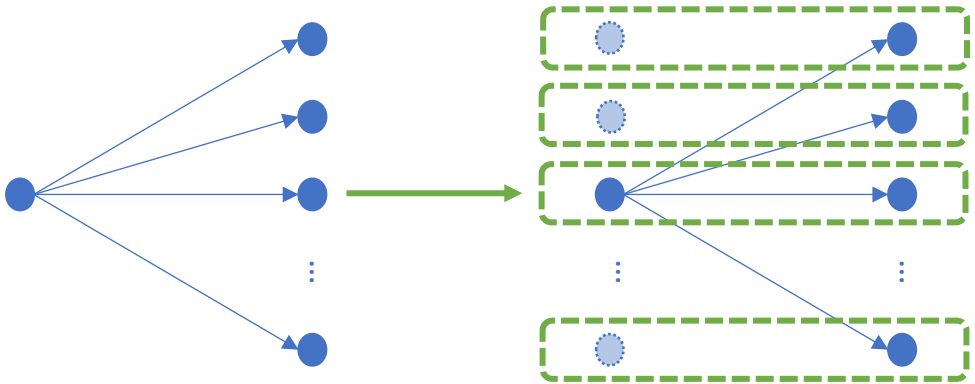
March scenarios: Some MF schemes are more accurate than others

~150% error



Multifidelity Scenario Bundling: Ongoing Work

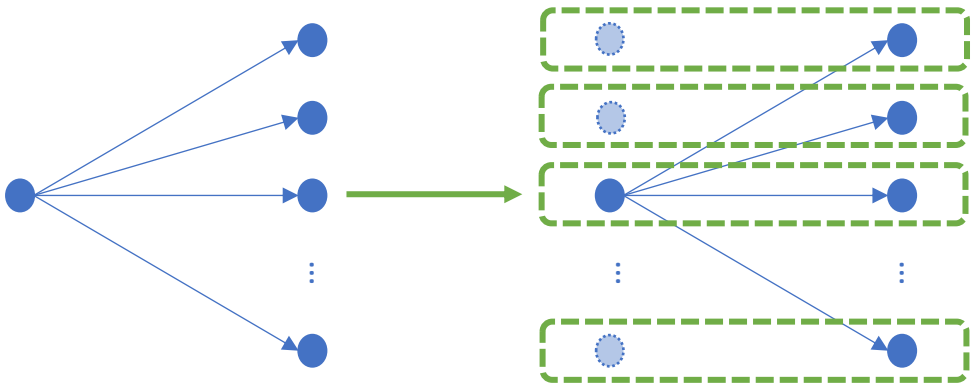
- Re-running experiments in parallel.



Stochastic Programs are parallelizable

Multifidelity Scenario Bundling: Ongoing Work

- Re-running experiments in parallel.
- Increasing maximum number of iterations.
- Extending model to capacity expansion planning.



Stochastic Programs are parallelizable

Z. Kilwein, **R. M. Alfant**, W. Hart. Performant Optimization Strategies for Multifidelity Stochastic Programs on Power Grid Applications. *In prep*.

How can power systems be optimized for efficiency and resiliency?

- Motivation: Power Systems Planning
- Multifidelity Scenario Bundling
- Stochastic Programming And Related Optimization Workflows (SPAROW)
- Summary & Future Research Directions

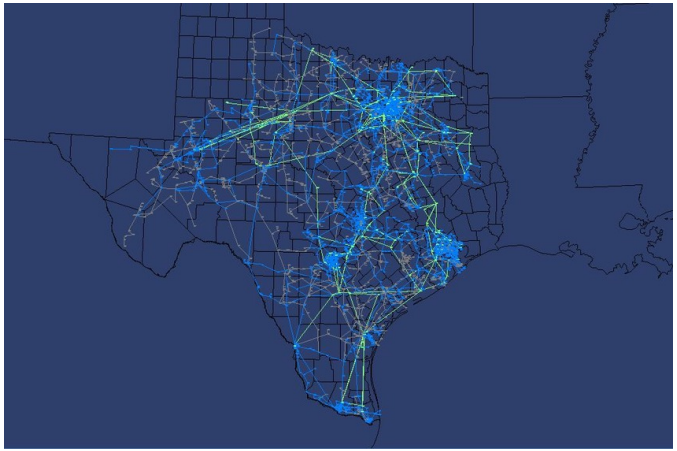


Image: Li, Nan, et al. "Optimal generation investment planning: Pt. 2: Application to the ERCOT system." 2012 North American Power Symposium. IEEE, 2012.

Standardize workflow with Python library

- **Goal:** simple interface for specifying stochastic programming models.

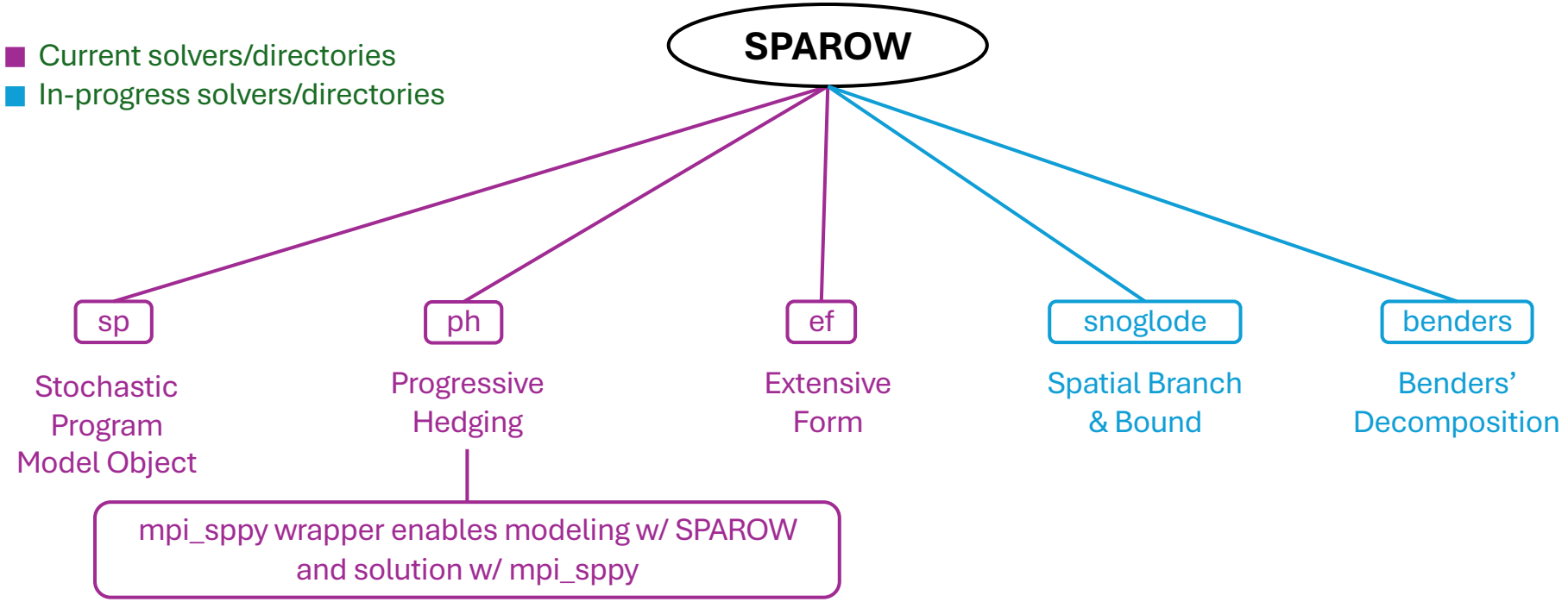
W. E. Hart, C. D. Laird, J. P. Watson, D. L. Woodruff, G. A. Hackebeil, B. L. Nicholson, & J. D. Siirola (2017). Pyomo-optimization modeling in Python. Vol. 67, *Berlin: Springer*.

Standardize workflow with Python library

- **Goal:** simple interface for specifying stochastic programming models.
- **Features:**
  - Explicit support for ***multifidelity modeling***.
  - Capabilities for sophisticated ***scenario bundling***.
  - Interfaces with variety of solvers.

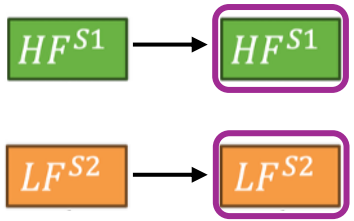
W. E. Hart, C. D. Laird, J. P. Watson, D. L. Woodruff, G. A. Hackebeil, B. L. Nicholson, & J. D. Siirola (2017). Pyomo-optimization modeling in Python. Vol. 67, *Berlin: Springer*.

SPAROW: Stochastic Programming And Related Optimization Workflows

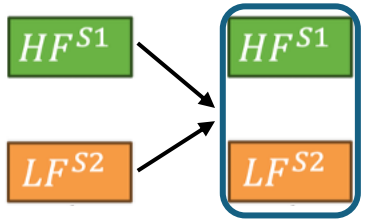


R. M. Alfant, Z. Kilwein, M. Viens, G. Stinchfield, W. Hart. Stochastic Programming And Related Optimization Workflows (SPAROW): A Python Package for Modeling and Solving Stochastic Programs with Pyomo. *Working Paper*.

SPAROW supports rich set of automated bundling schemes



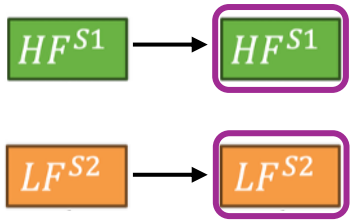
Default



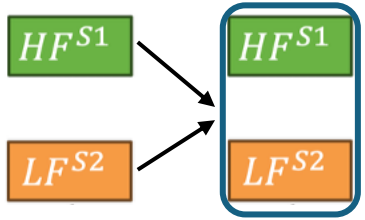
Extensive Form

R. M. Alfant, Z. Kilwein, M. Viens, G. Stinchfield, W. Hart. Stochastic Programming And Related Optimization Workflows (SPAROW): A Python Package for Modeling and Solving Stochastic Programs with Pyomo. *Working Paper*.

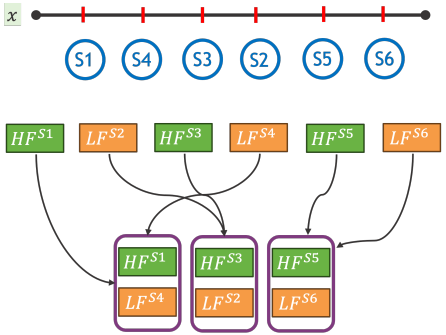
SPAROW supports rich set of automated bundling schemes



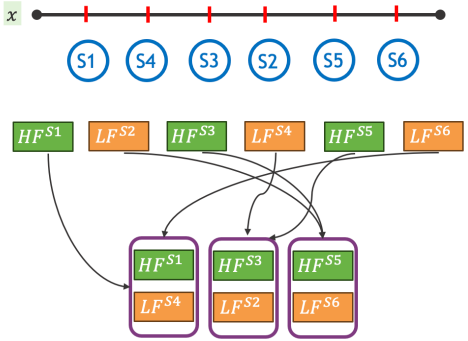
Default



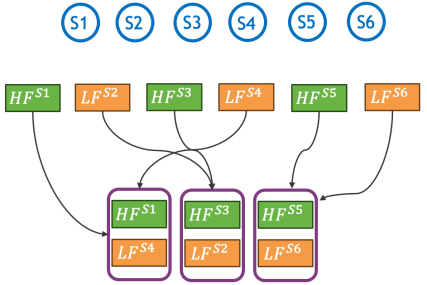
Extensive Form



Similarity

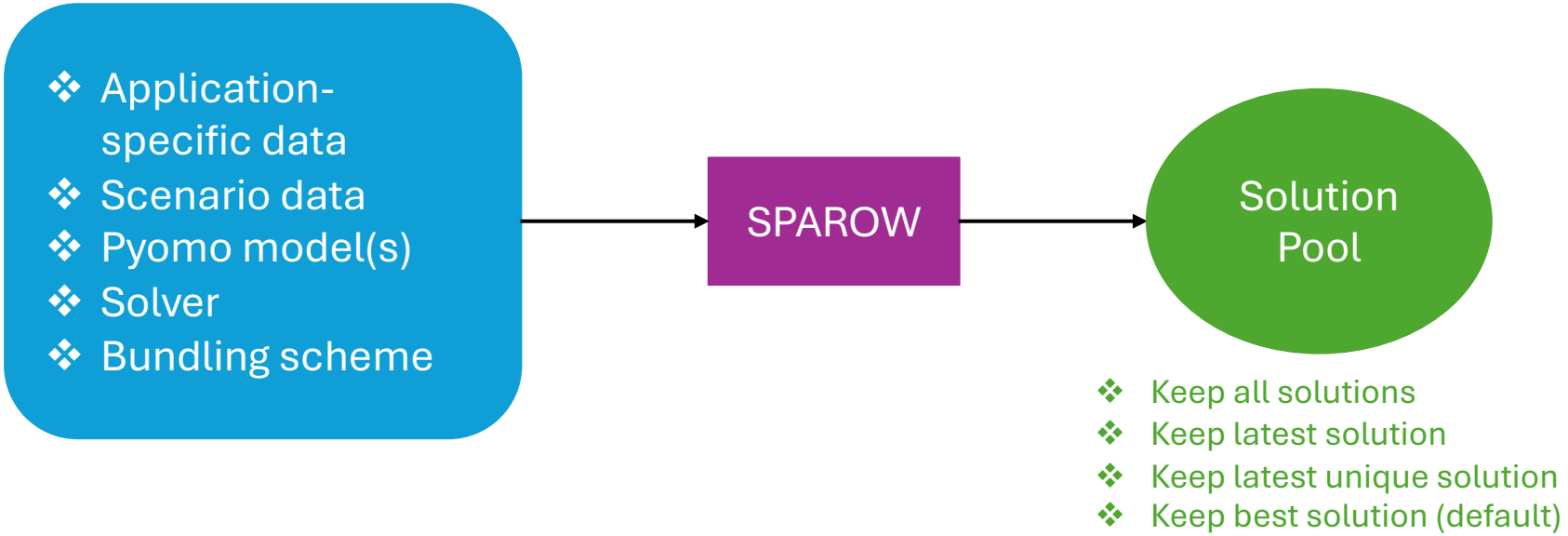


Dissimilarity



Random

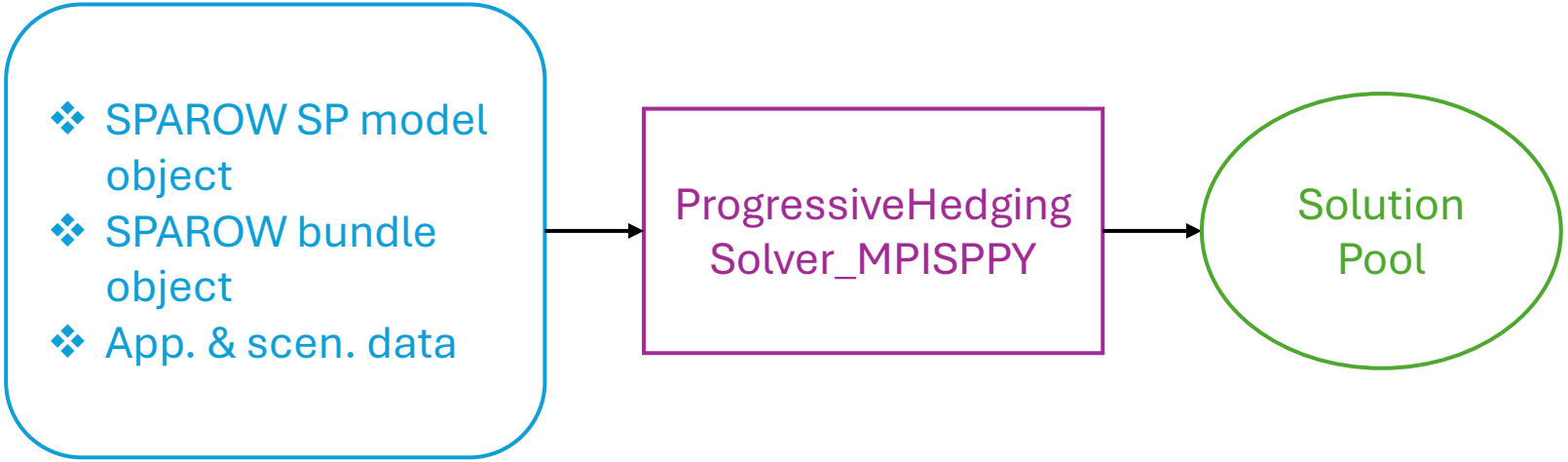
SPAROW enables straightforward specification of SP models



R. M. Alfant, Z. Kilwein, M. Viens, G. Stinchfield, W. Hart. Stochastic Programming And Related Optimization Workflows (SPAROW): A Python Package for Modeling and Solving Stochastic Programs with Pyomo. *Working Paper*.

mpi-sppy has generic configuration that accepts bundled-subproblems

❖**Specify SP model with SPAROW, solve w/ mpi_sppy capabilities**



R. M. Alfant, Z. Kilwein, M. Viens, G. Stinchfield, W. Hart. Stochastic Programming And Related Optimization Workflows (SPAROW): A Python Package for Modeling and Solving Stochastic Programs with Pyomo. *Working Paper*.

SPAROW repository contains well-documented exemplars

❖**Long-term goal: increase adoption of stochastic programming**

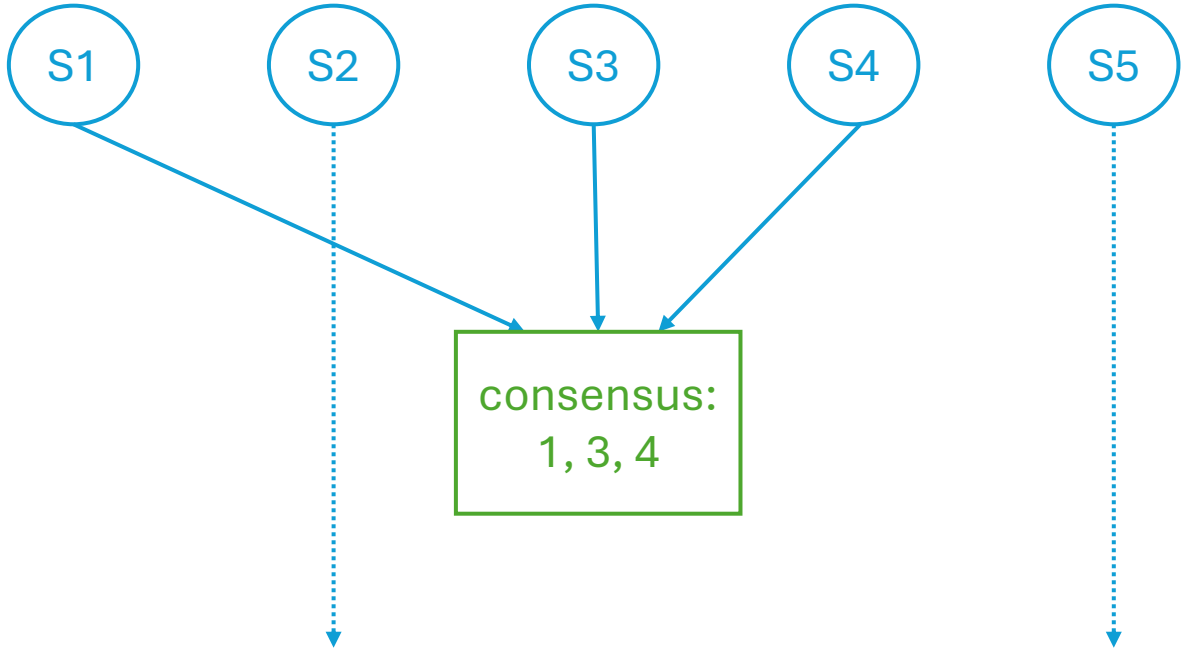
Solving optimization problems in sparow_examples with Serial PH

In [2]:
```
### Import serial PH solver
from sparow.ph.ph import ProgressiveHedgingSolver

### Import pprint for solution readability
import pprint
```

In [3]:
```
sp = random_HF_LF1_grid_facilityloc()   # SP model object imported from sparow_examples
solver = ProgressiveHedgingSolver()     # solving with serial PH
solver.set_options(
    solver="gurobi",   # solving with gurobi
    max_iterations=2,  # this will default to 100
    loglevel="INFO",   # can replace with DEBUG, VERBOSE, etc.
    rho_updates=True,  # rho parameter will update at each iteration
)

results = solver.solve(sp, solver="gurobi")  # returns the solution object
pprint.pprint(results.to_dict())             # pretty-prints results
```

Next Steps: Asynchronous PH with Bundling?

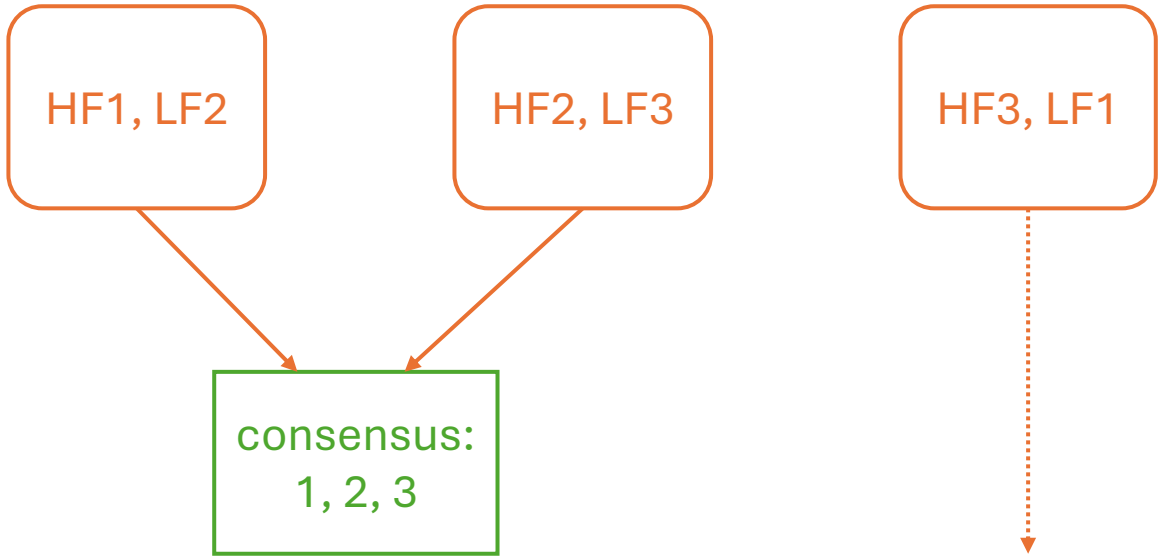
- Asynchronous PH (APH) has not been well studied for mixed-integer problems.



Only a subset of scenarios are included in consensus solution

Next Steps: Asynchronous PH with Bundling?

- Asynchronous PH (APH) has not been well studied for mixed-integer problems.
- Multifidelity APH with bundling may be effective heuristic.



More scenarios can be represented in each iteration using multiple fidelities

How can power systems be optimized for efficiency and resiliency?

- Motivation: Power Systems Planning
- Multifidelity Scenario Bundling
- Stochastic Programming And Related Optimization Workflows (SPAROW)
- Summary & Future Research Directions

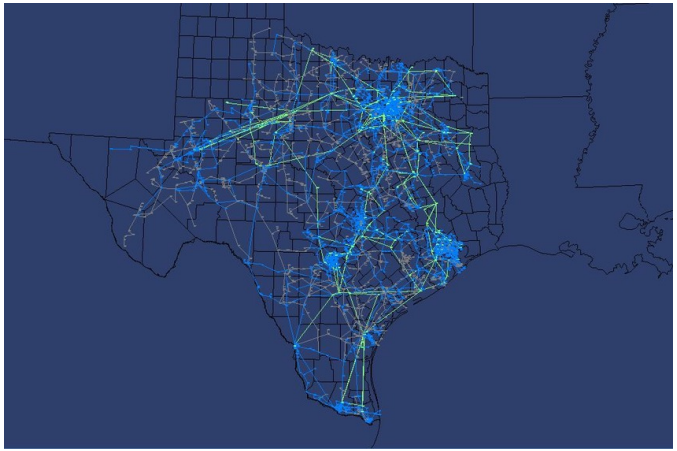


Image: Li, Nan, et al. "Optimal generation investment planning: Pt. 2: Application to the ERCOT system." 2012 North American Power Symposium. IEEE, 2012.

Key Takeaways

- Stochastic programming accounts for uncertainty via discrete scenarios; well-suited for power grid planning.

Contact: rmalfan@sandia.gov | https://rachaelalfant.github.io

Key Takeaways

- Stochastic programming accounts for uncertainty via discrete scenarios; well-suited for power grid planning.
- Multifidelity PH with scenario bundling can accelerate optimization for power grid planning problems.

Contact: rmalfan@sandia.gov | https://rachaelalfant.github.io

Key Takeaways

- Stochastic programming accounts for uncertainty via discrete scenarios; well-suited for power grid planning.
- Multifidelity PH with scenario bundling can accelerate optimization for power grid planning problems.
- SPAROW 1.0 release!

https://github.com/sandialabs/sparow/

Contact: rmalfan@sandia.gov | https://rachaelalfant.github.io

Key Takeaways

- Stochastic programming accounts for uncertainty via discrete scenarios; well-suited for power grid planning.
- Multifidelity PH with scenario bundling can accelerate optimization for power grid planning problems.
- SPAROW 1.0 release!
- Future work:
  - APH with bundling for distributed optimization under uncertainty.
  - Uncertainty quantification.

https://github.com/sandialabs/sparow/

Contact: rmalfan@sandia.gov | https://rachaelalfant.github.io

Extra Slides

Progressive Hedging

Algorithm 1 PH Algorithm with Scenario Bundling

1: **Initialization:** Let $i \leftarrow 0, w_b^i \leftarrow 0 \ \forall b \in \mathcal{B}$. For each $b \in \mathcal{B}$, compute:

$$(x_b^{i+1}, y_b^{i+1}) \in \arg \min_{x,y \in \mathcal{Q}_b} c^\top x + h_b^\top y$$

2: **Iteration Update:** $i \leftarrow i + 1$

3: **Aggregation:** $\bar{x}^i = \sum_{b \in \mathcal{B}} p_b x_b^i$

4: **Price Update:** $w_b^i \leftarrow w_b^{i-1} + \rho(x_b^i - \bar{x}^i)$

5: **Decomposition:** For each $b \in \mathcal{B}$, compute:

$$(x_b^{i+1}, y_b^{i+1}) \in \arg \min_{x,y \in \mathcal{Q}_b} \{c^\top x + h_b^\top y + w_b^{i\top} x + \frac{\rho}{2} ||x - \bar{x}^i||^2\}$$

6: If all bundle solutions x_b are equal, stop.

7: Else, go to Step 2.

app_data Examples

Dictionary containing application-specific data

- E.g., number of customers, time periods, etc.

Newsvendor Problem

```
#
# Data for a simple newsvendor example
#
app_data = dict(c=1.0, b=1.5, h=0.1)
```

Capacitated Facility Location

```
app_data = {"n": 3, "t": 4}  # number of facilities & customers
app_data["f"] = [400000, 200000, 600000]  # fixed costs for opening facilities
app_data["c"] = [
    [5739.725, 6539.725, 8650.40, 22372.1125],
    [6055.05, 6739.055, 8050.40, 21014.225],
    [8650.40, 7539.055, 4539.72, 15024.325],
]  # servicing costs
app_data["k"] = [1550, 650, 1750]  # facility capacity
```

model_data Example: Newsvendor

Keys of model_data dictionary are model fidelities

Each sub-dictionary requires “scenarios” key, whose value is a list containing a dictionaries that represent scenarios (one dict per scenario). Other keys are optional.

```
model_data = {
    "LF": {
        "scenarios": [
            {"ID": "1", "d": 15, "Probability": 0.1},
            {"ID": "2", "d": 60, "Probability": 0.3},
            {"ID": "3", "d": 72, "Probability": 0.15},
            {"ID": "4", "d": 78, "Probability": 0.25},
            {"ID": "5", "d": 82, "Probability": 0.2},
        ]
    },
    "HF": {
        "data": {"B": 0.9},
        "scenarios": [
            {"ID": "1", "d": 15, "C": 1.4, "Probability": 0.05},
            {"ID": "2", "d": 60, "C": 1.3, "Probability": 0.4},
            {"ID": "3", "d": 72, "C": 1.2, "Probability": 0.1},
            {"ID": "4", "d": 78, "C": 1.1, "Probability": 0.35},
            {"ID": "5", "d": 82, "C": 1.0, "Probability": 0.1},
        ],
    },
}
```

model_data Example: Newsvendor

Each scenario requires an ***"ID"***, ***probability***, and ***demand***

- ***"ID"*** value is a string
- ***Demand*** and ***probability*** are assumed to be floats
  - ***Demand*** can also be a list of floats
  - Key names can also be common abbreviations (e.g., "d", "Pr", "p", etc.), but less common naming conventions must be specified in the bundle_args dictionary with keys "demand_key" and/or "probability_key"
    - E.g., bundle_args = {"probability_key": "pb"}
  - Error returned if any demand values are missing (for scenarios that are used)
  - Error returned if some, but not all, probability values are missing
  - If no scenario probabilities are given, uniform distribution is assumed

```
model_data = {
    "LF": {
        "scenarios": [
            {"ID": "1", "d": 15, "Probability": 0.1},
            {"ID": "2", "d": 60, "Probability": 0.3},
            {"ID": "3", "d": 72, "Probability": 0.15},
            {"ID": "4", "d": 78, "Probability": 0.25},
            {"ID": "5", "d": 82, "Probability": 0.2},
        ]
    },
    "HF": {
        "data": {"B": 0.9},
        "scenarios": [
            {"ID": "1", "d": 15, "C": 1.4, "Probability": 0.05},
            {"ID": "2", "d": 60, "C": 1.3, "Probability": 0.4},
            {"ID": "3", "d": 72, "C": 1.2, "Probability": 0.1},
            {"ID": "4", "d": 78, "C": 1.1, "Probability": 0.35},
            {"ID": "5", "d": 82, "C": 1.0, "Probability": 0.1},
        ],
    },
}
```

model_builder Example: Newsvendor

```
#
# Function that constructs a newsvendor model
# including a single second stage
#
def LF_builder(data, args):
    b = data["b"]
    c = data["c"]
    h = data["h"]
    d = data["d"]

    M = pyo.ConcreteModel(data["ID"])

    M.x = pyo.Var(within=pyo.NonNegativeReals)

    M.y = pyo.Var()
    M.greater = pyo.Constraint(expr=M.y >= (c - b) * M.x + b * d)
    M.less = pyo.Constraint(expr=M.y >= (c + h) * M.x - h * d)

    M.o = pyo.Objective(expr=M.y)

    return M
```

```
def HF_builder(data, args):
    b = data["b"]
    B = data["B"]
    c = data["c"]
    C = data["C"]
    h = data["h"]
    d = data["d"]

    M = pyo.ConcreteModel(data["ID"])

    M.x = pyo.Var(within=pyo.NonNegativeReals)

    M.y = pyo.Var()
    M.greater = pyo.Constraint(expr=M.y >= (c - b) * M.x + b * d)
    M.greaterX = pyo.Constraint(expr=M.y >= (C - B) * M.x + B * d)
    M.less = pyo.Constraint(expr=M.y >= (c + h) * M.x - h * d)

    M.o = pyo.Objective(expr=M.y)

    return M
```

model_builder contains Pyomo model – need a builder function for each model fidelity being used

Solving the Extensive Form of the Model

create the stochastic_program object with a list strings of first-stage variable names

initialize application using app_data

initialize model by specifying the name, model_data, and model_builder

specify the solver (EF, PH, etc.) and relevant solver options

solve the model and save optimal solution results as a json file

```
sp = stochastic_program(first_stage_variables=["x"])
sp.initialize_application(app_data=app_data)
sp.initialize_model(
    name="HF", model_data=model_data["HF"], model_builder=HF_builder
)
solver = ExtensiveFormSolver()
solver.set_options(solver="gurobi")
results = solver.solve(sp)
results.write("results.json", indent=4)
```

Solving the Model with PH

create SP object, initialize application and model

call the PH solver

can cache the model after the first iteration of PH so that it doesn't have to be recreated in subsequent iterations

can specify max. number of iterations before PH terminates

method by which consensus solution is obtained (for integer first-stage, will round)

will iteratively update penalty rho parameter if switched on (helps accelerate convergence)

solve the model and save results

```
sp = stochastic_program(first_stage_variables=["x"])
sp.initialize_application(app_data=app_data)
sp.initialize_model(
    name="LF", model_data=model_data["LF"], model_builder=LF_builder
)

solver = ProgressiveHedgingSolver()
solver.set_options(
    solver="gurobi",
    loglevel=loglevel,
    cached_model_generation=cache,
    max_iterations=max_iter,
    finalize_all_xbar=finalize_all_iters,
    rho_updates=True,
)
results = solver.solve(sp)
results.write("results.json", indent=4)
```