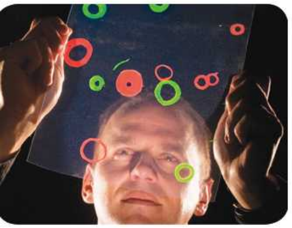
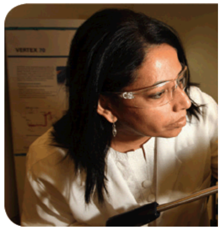
SAND2016-6825C





Accelerating and automatic tuning for Progressive Hedging

John D. Siirola[1], Jean-Paul Watson[1], and David L. Woodruff[2]

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

[2]Graduate School of Management
University of California, Davis
Davis, CA USA

International Conference on Stochastic Programming (ICSP)
27 June 2016

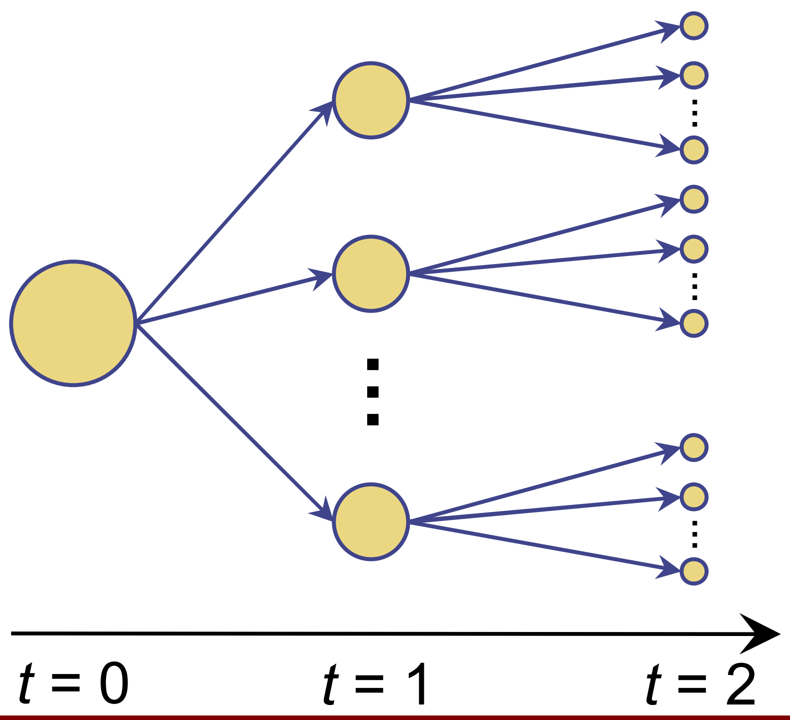
Sandia National Laboratories

Exceptional service in the national interest

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

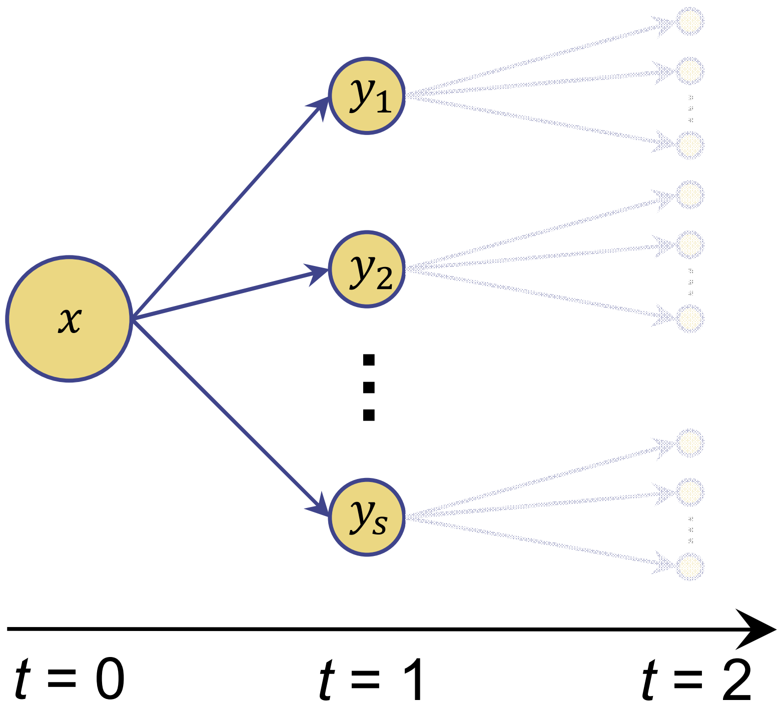
A graphical view of SP decomposition

- Stochastic programming is explicitly built on a tree of scenarios where nodes represent opportunities for decisions
  - The monolithic problem (extensive form) is typically intractable
  - Decomposition and iterative convergence the workhorse of SP



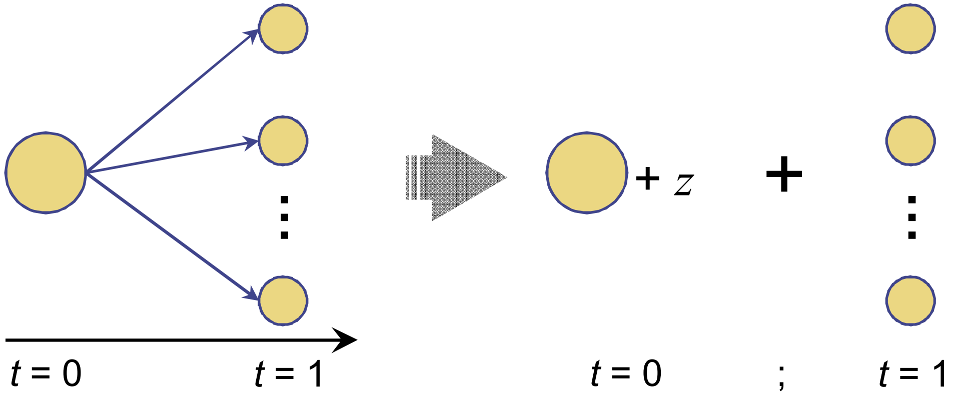
A graphical view of SP decomposition

- Stochastic programming is explicitly built on a tree of scenarios where nodes represent opportunities for decisions
  - The monolithic problem (extensive form) is typically intractable
  - Decomposition and iterative convergence the workhorse of SP
  - For convenience, let us restrict ourselves to 2-stage problems



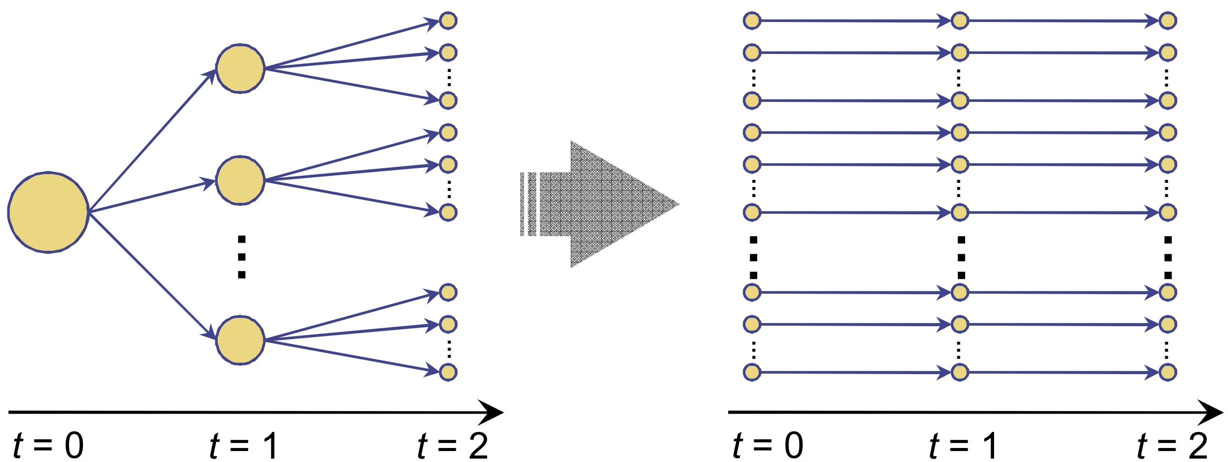
Benders Decomposition: splitting time

- Benders Decomposition splits the problem “in time” into
  - A *master* problem that (initially) has the 1st-stage variables
    - (Plus any discrete variables from the 2nd stage)
    - Plus a proxy variable *z* to capture impact of subproblems
  - Independent *subproblems*, 1 per scenario, for the 2nd-stage variables
    - Must be continuous
- Iteratively solve master to obtain upper bound (incumbent)
  - Solve subproblems to obtain lower bound + cuts for the master
- Challenges
  - Discrete recourse decisions
  - Master problem “bloat”
  - Multistage problems
    - (bookkeeping)



Progressive Hedging: splitting anticipativity

- Progressive Hedging splits the problem by scenarios
  - "No" master problem
  - One subproblem per scenario
  - Relax nonanticipativity constraints
- Iteratively converge the stage nonanticipativity constraints
  - Penalize decision variable value by weight $w^T x$
  - Penalize deviation from average, $\rho \, ||x - \bar{x}||^2$
  - Update w using ρ



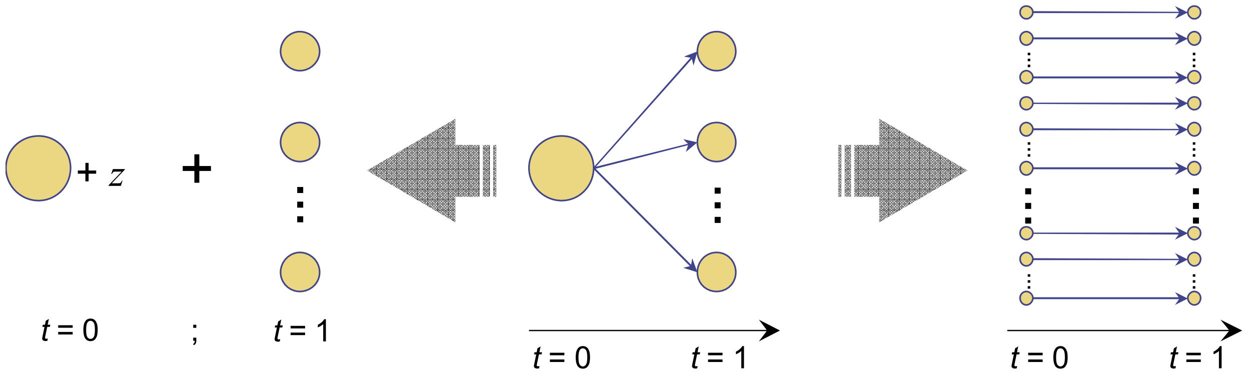
PH: Benefits and open challenges

- PH avoids many of the challenges experienced by Benders
  - No restriction on stage variable domains (discrete 2^{nd} stage OK)
  - "Trivially" extends to multistage problems
  - No master problem, no cuts generated: no problem "bloat"

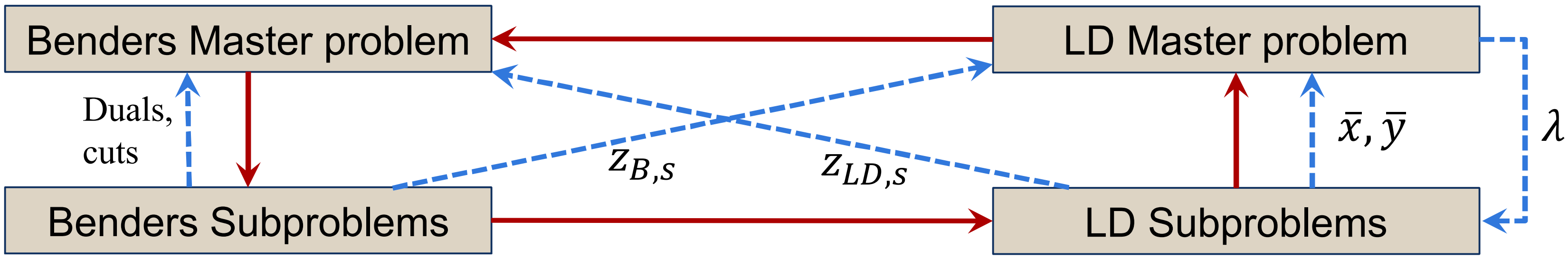
- BUT…
  - Not provably convergent for the discrete case
    - …although bounds exist on the quality of solution you get
  - "Infeasible path" algorithm
  - Discrete variable cycling
  - Slow convergence due to "holdout" scenarios
  - How to choose ρ???

We will attempt to address these challenges

PH + Benders: orthogonal decompositions

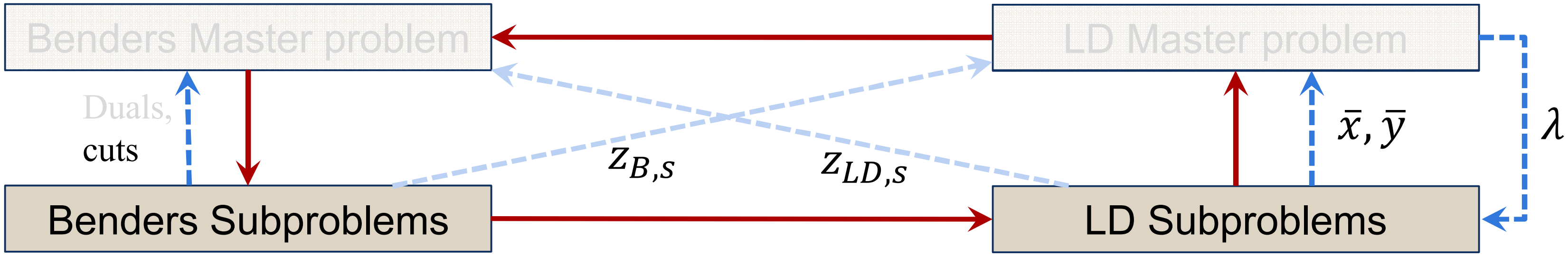


- Can we leverage ideas from each to accelerate the other?
  - Related: Cross Decomposition (Lagrangean Decomposition + Benders)
    - [Van Roy 1983; Holmberg, 1990; Mitra,et al. 2016]

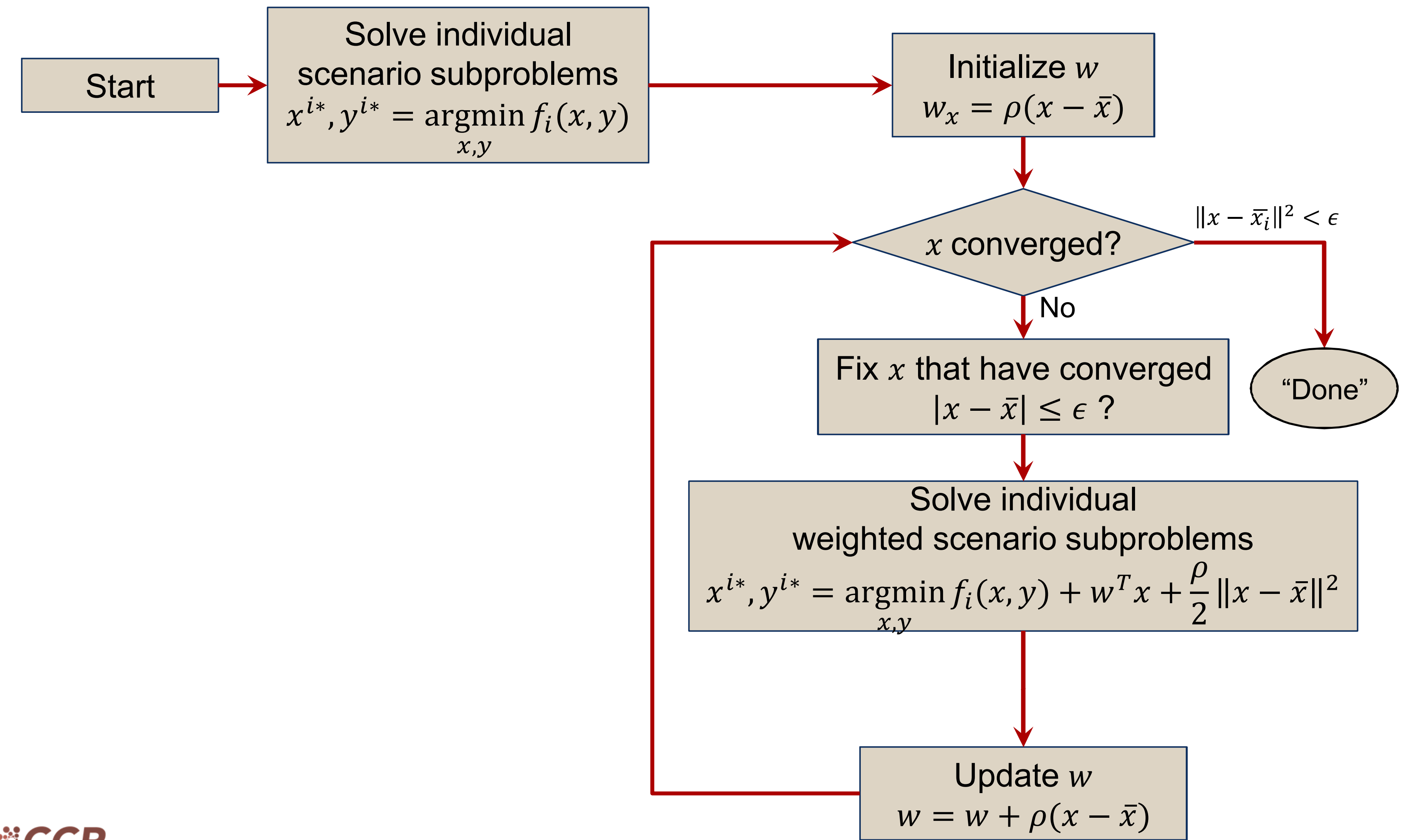


Cross Decomposition for PH?

- Challenges with naïve Cross Decomposition for PH
  - No LD Master problem
  - Discrete second stage variables
  - PH subproblems do not provide proper Lagrangean bounds
    - Except at *iteration 0*



Progressive Hedging: the algorithm



Improving PH: borrowing from Benders

- Consider the Benders "feasibility" cut:
  - Given x^* computed by the RMP, if (dual) 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:

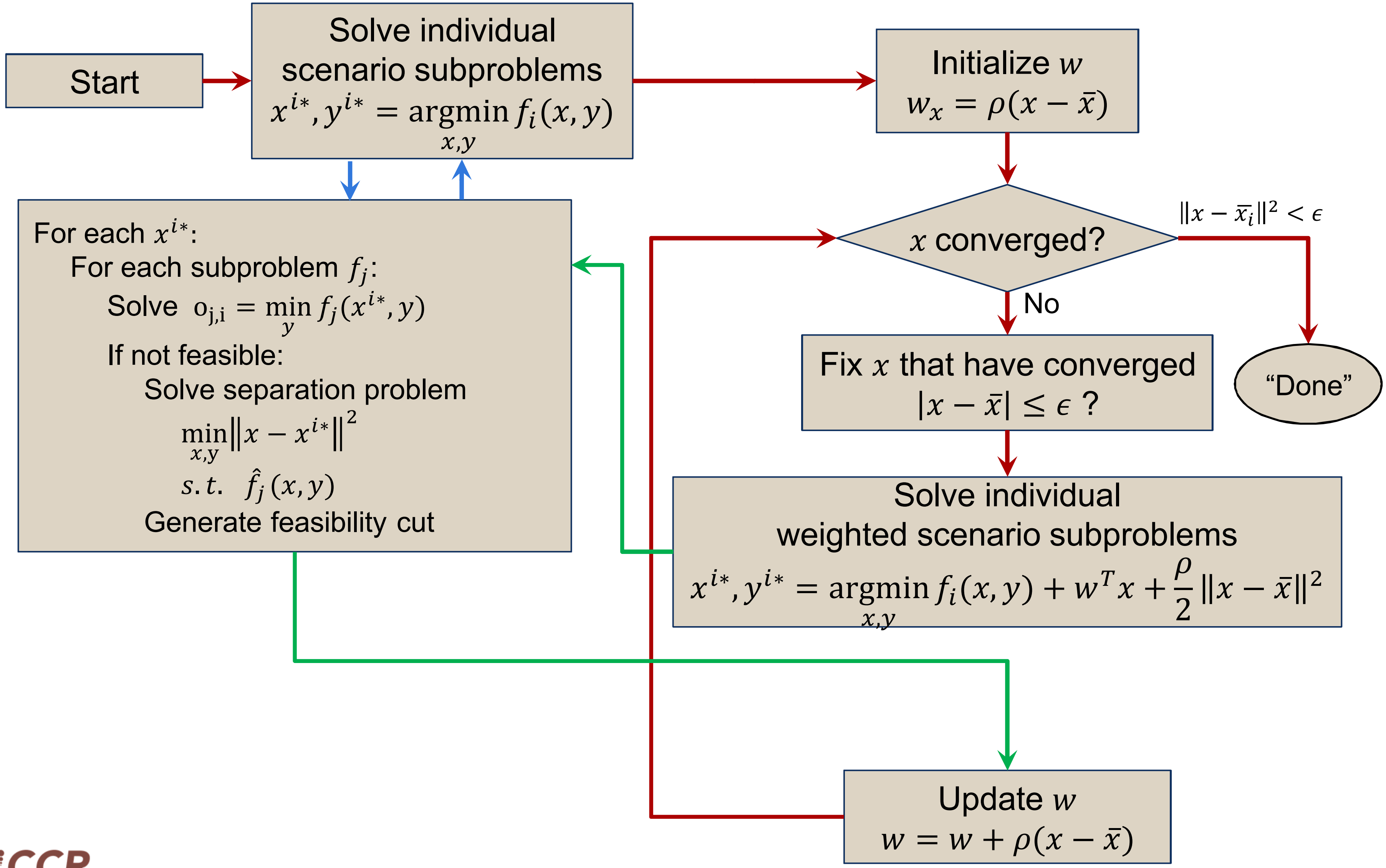
$$\min_{y} f_j(x^{i*}, y)$$

  - 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:

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

  - 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

PH + feasibility



Case study: stochastic network flow

- 2-stage stochastic network flow from [Watson & Woodruff, 2011]:
  - 1st stage variables:
    - $x_a \; \forall \; a \in Arcs$; $\{x_a | \; x_a \in R, 0 \le x_a \le x_a^{UB}\}$ Capacity of Arc a
    - $b_a^0 \; \forall \; a \in Arcs$; $\mathrm{b}_\mathrm{a}^0 \in \{0,1\}$ Arc a is available
  - 2nd stage variables:
    - $y_a^s \; \forall \; a \in Arcs, s \in Scenarios$; $\{y_a^s \mid y_a^s \in R, 0 \le y_a^s \le x_a\}$ Flow across Arc a in scenario s
    - $b_a^s \; \forall \; a \in Arcs, s \in Scenarios$; $\mathrm{b}_\mathrm{a}^s \in \{0, b_a^s\}$ Arc a is in use for scenario s
- PH ($\rho = 100$, with "Watson-Woodruff" extensions)
  - Significant cycling (no convergence after 1000 iterations)
- PH ($\rho = 100$, with WW extensions, with feasibility cuts)
  - Converges in 42 iterations (objective: 164426, 2726 seconds)
  - 6 preliminary cut passes: raises Lagrangian LB 135085 → 148656

Case study: UC + *N-1* + switching

- 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
    - Line switching (opening /closing a line) in 1st and 2nd stages

- 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 Lagrangean bound from 19.7 to 19.88
    - 12 iterations, objective = 20.11, total time 568 seconds

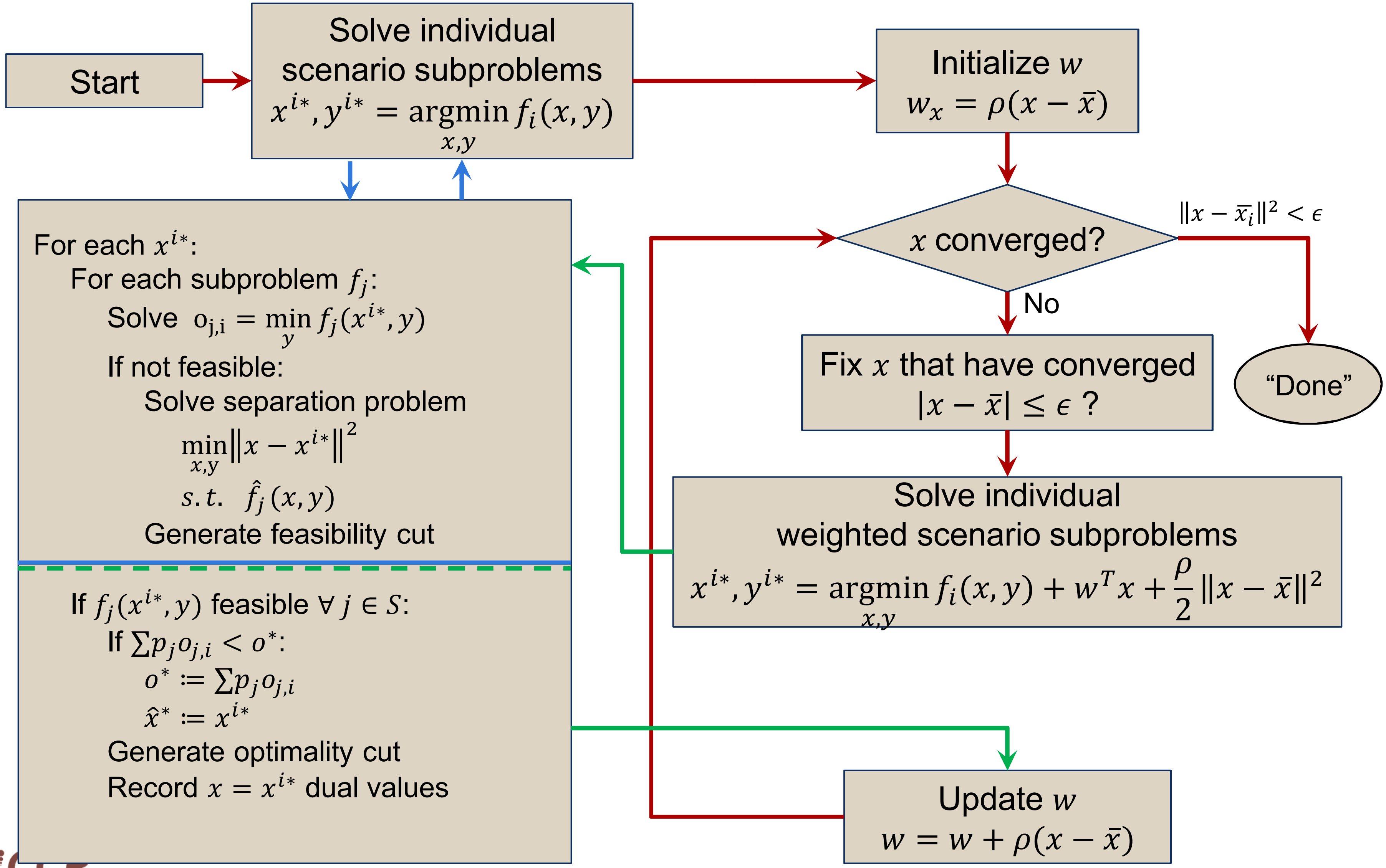
Improving PH: borrowing from Benders

- What happens if the subproblem $\min_{y} f_i(x^{j*}, y)$ is feasible?
  - The 1st stage decisions are valid in this scenario
  - If x^{j*} is valid in ALL scenarios, then x^{j*} satisfies nonanticipativity and the expectation of the subproblems forms a valid upper bound

$$E[f(x^*, y)] \leq E\left[f\left(x^{j*}, y\right)\right]$$

- If the first stage variables are all discrete
  - Repeating this process for all x^{i*} and identify additional nonanticipative solutions, then the upper bound can be used to generate *optimality cuts* to exclude sub-optimal solutions
  - These cuts are also valid on all scenario subproblems

PH + feasibility + optimality cuts



Case Studies: *farmer*

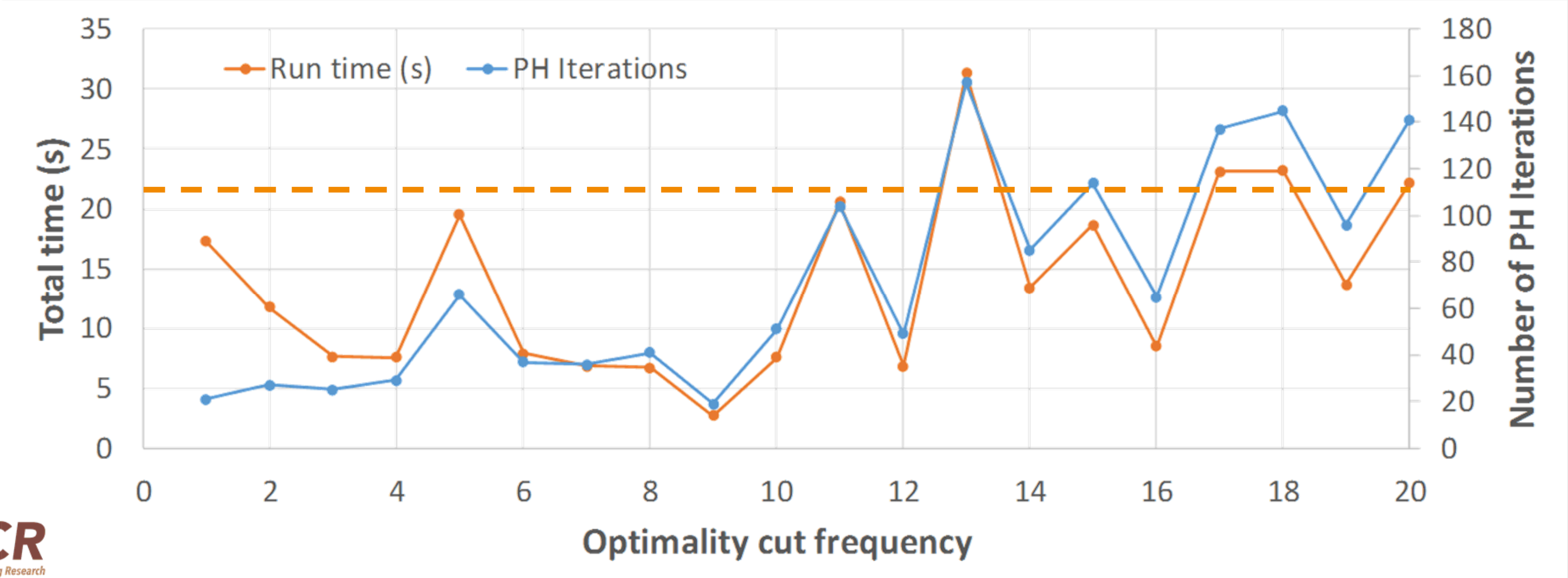
- 3-scenario farmer problem [Birge & Louveaux]
  - Integer acreage allocations
  - Cplex 12.5
    - PH ($\rho = 1$): 297 iterations, 21.55 seconds: objective = -108390
    - PH + optimality: 47 iterations, 8.75 seconds: objective = -108390

Case Studies: *farmer*

- 3-scenario farmer problem [Birge & Louveaux]
  - Integer acreage allocations
  - Cplex 12.5
    - PH ($\rho = 1$): 297 iterations, 21.55 seconds: objective = -108390
    - PH + optimality: 47 iterations, 8.75 seconds: objective = -108390
- Very sensitive to the solver!
  - Gurobi 6.0.4
    - PH ($\rho = 1$): 39 iterations, 1.09 seconds: objective = -108390
    - PH + optimality: 154 iterations, 16.4 seconds: objective = -108390

Case Studies: *farmer*

- 3-scenario farmer problem [Birge & Louveaux]
  - Integer acreage allocations
  - Cplex 12.5
    - PH ($\rho = 1$): 297 iterations, 21.55 seconds: objective = -108390
    - PH + optimality: 47 iterations, 8.75 seconds: objective = -108390
- Very sensitive to the strategy!



Case study: Stochastic Unit Commitment

- Unit commitment under demand uncertainty
  - 24 hour horizon, 1-hour intervals

- Case 1: 5 busses, 7 generators, 3 load scenarios
  - Optimal solution (extensive form): 348.98
  - Default PH ($\rho = 10$):
    - Significant cycling (no convergence after 100 iterations)
  - PH ($\rho = 10$) + Optimality cuts:
    - 20 iterations, objective = 348.9835

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]

$$\rho_i \propto \frac{C_i}{\left|x_i^{min} - x_i^{max} + 1\right|}$$

- 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

Case Studies: *farmer*

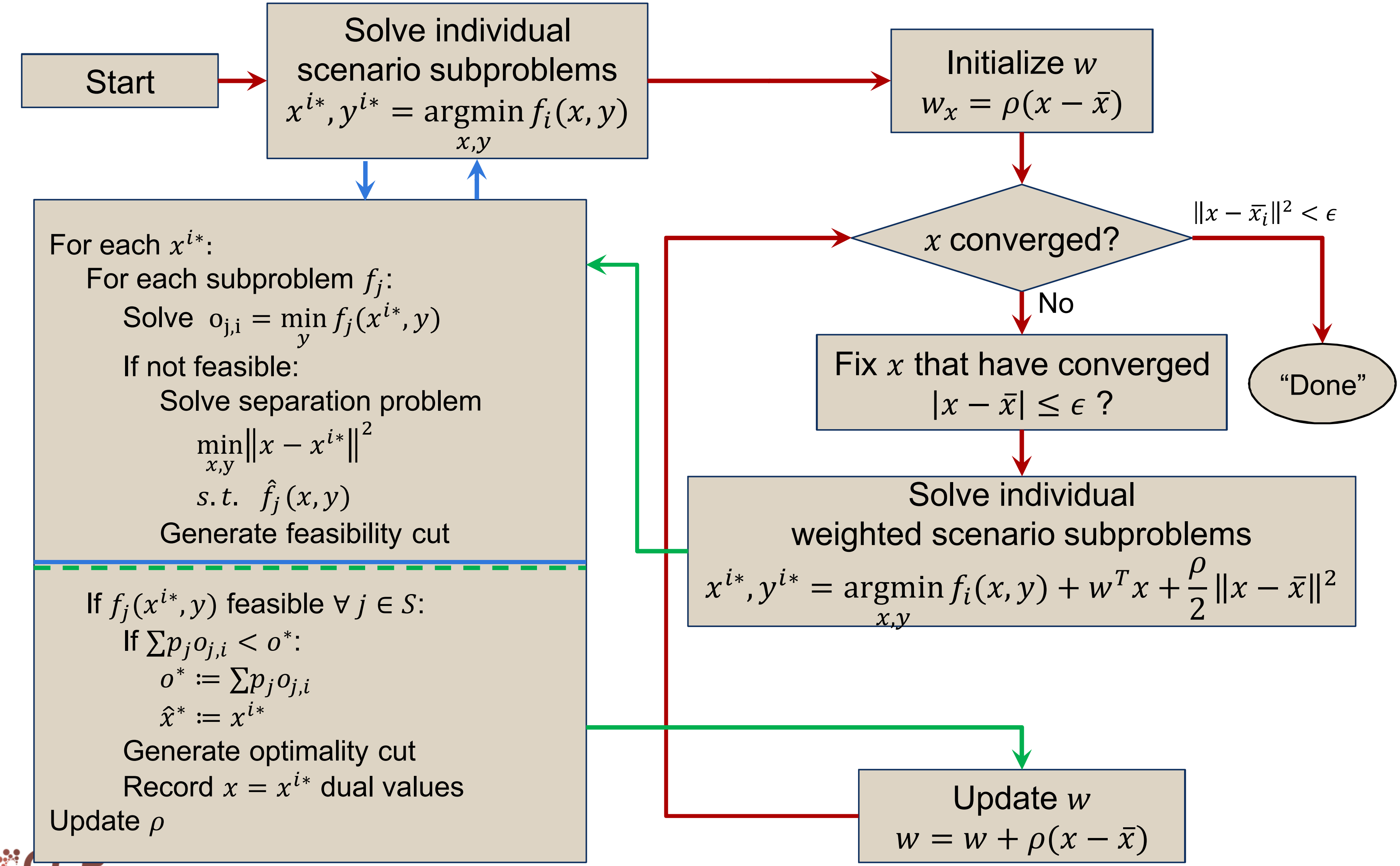
- 3-scenario farmer problem [Birge & Louveaux]
  - Continuous acreage allocations
    - PH ($\rho = 1$):
      - 33 iterations, 1.06 seconds: objective = -108388.7726
    - PH + ρ setter:
      - 34 iterations, 1.34 seconds: objective = -108389.7811
  - Discrete acreage allocations
    - PH ($\rho = 1$):
      - 39 iterations, 1.09 seconds: objective = -108390
    - PH + optimality cuts:
      - 154 iterations, 16.4 seconds: objective = -108390
    - PH + optimality cuts + ρ setter:
      - 40 iterations, 4.42 seconds: objective = -108390

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 19.88
    - 12 iterations, objective = 20.11, total time 568 seconds

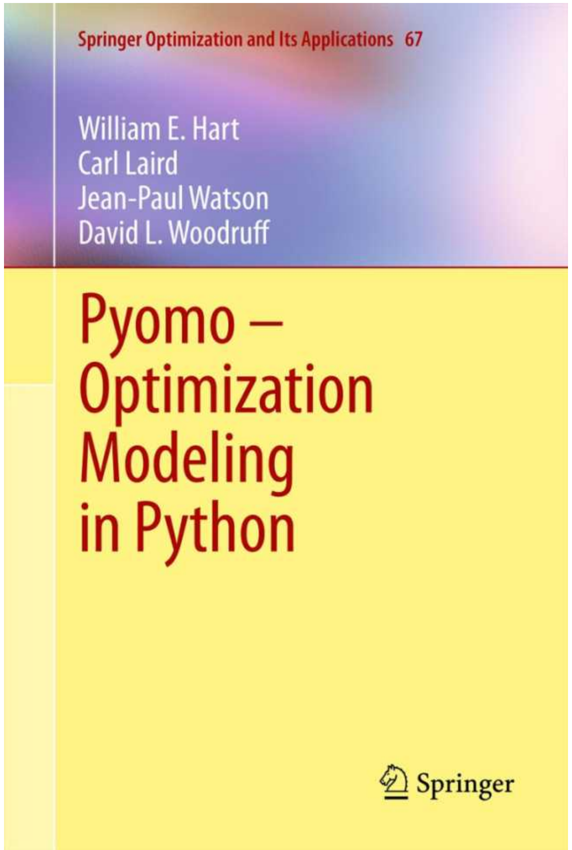
- Now:
  - PH + Feasibility cuts + ρ setter
    - 20 iterations, objective = 19.9808, total time 494 seconds

PH + feasibility + optimality cuts+ ρ



Our software environment: Pyomo

- Open-source optimization modeling environment written in Python
- All experiments performed using Pyomo's PySP PH implementation
  - Cuts and ρ updates implemented using PH callbacks
- Project homepage
  - http://www.pyomo.org
  - Development recently moved to GitHub
- "The Book"
  - Second edition going to press in O(weeks)



- 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)

Conclusions and future directions

- Significant benefits from using "Cross-scenario" information
  - Lagrangian bound improvement
  - Cycle breaking
  - Convergence acceleration
  - Automatic tuning
  - "Fewer" problem-specific tuning parameters
- Open questions
  - $|S|^2$ subproblem solves to evaluate solutions is expensive
    - Although it is "trivially" parallelizable
    - Can we gain most of the benefit using only $c|S|$ subproblems?
  - New (scalar) tuning parameters
    - How frequently to evaluate cross-scenario information?
    - How aggressively to update ρ?
    - what is the most robust ρ update formula?