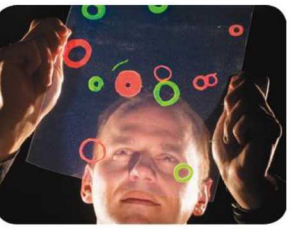
SAND2019-13041C



Sandia National Laboratories

Exceptional service in the national interest

Recent Developments in Pyomo

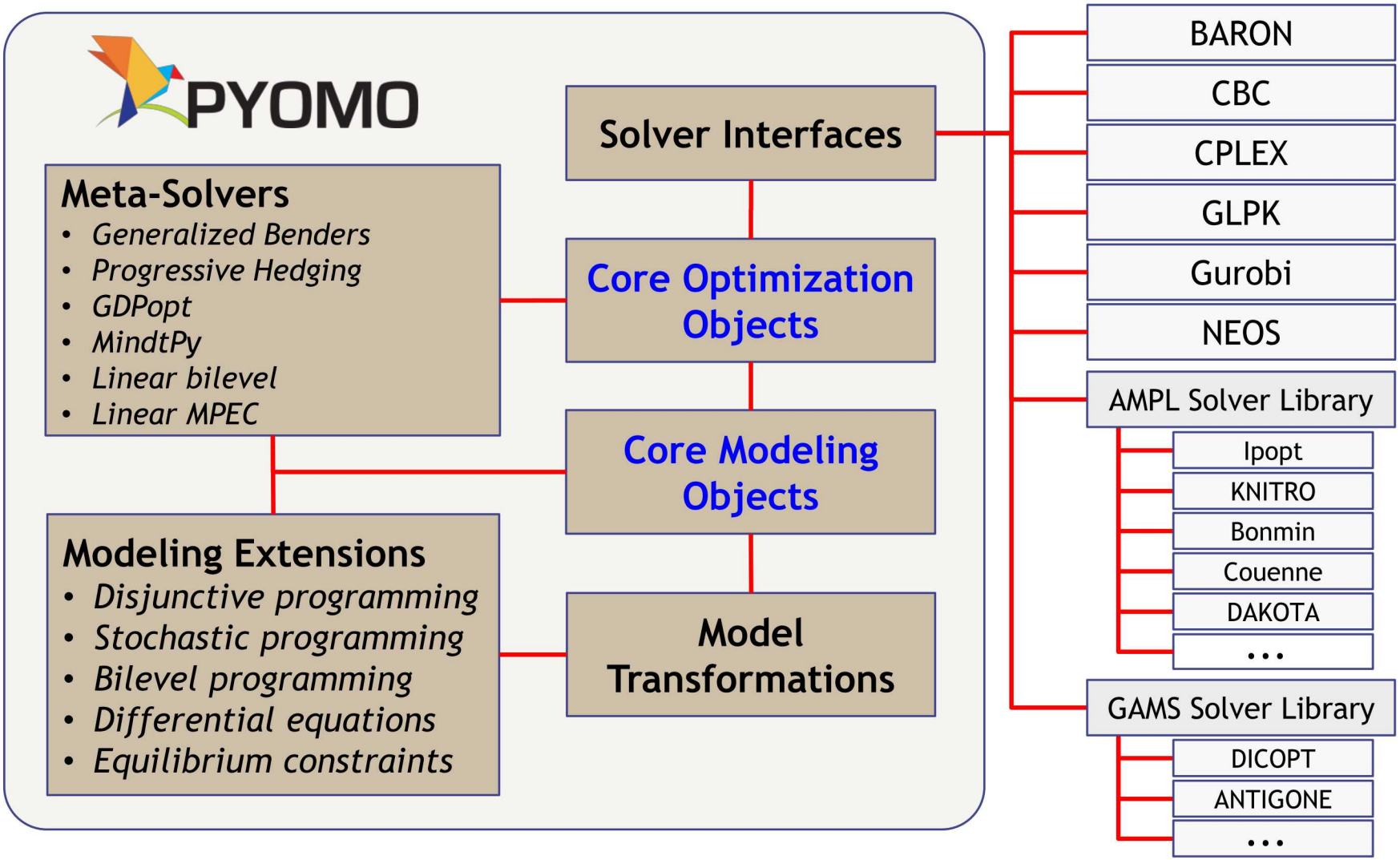
John D. Siirola and Bethany L. Nicholson

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

2019 INFORMS Annual Meeting

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and 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-NA-0003525

Pyomo at a Glance



Why *we* started Pyomo

Sandia National Laboratories

- 11 years ago…
  - Modeled primarily in AMPL (with some GAMS experience)
  - 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
  - Difficult to transfer our results to other organizations
  - Difficult to incorporate new ideas developed by the broader OR community
  - Recognized interdependence of modeling with solution approaches
    - Once-through process ("model + data → solver → solution") replaced by iterative approaches
      - Cut generation, outer approximation, Benders decomposition
    - New domain-specific modeling needs not supported by off-the-shelf AMLs
      - Stochastic programming, Dynamic optimization, Disjunctive programming, Bilevel optimization
    - Solutions increasingly required a combination of new modeling capabilities *and* manipulation / exploitation of the specific model structure
  - 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

So, how to discuss what's new this year…?

- 316 merged Pull Requests in the last 18 months
  - Smallest touched one file of documentation
  - Largest touched 574 source files

- If we really look at the changes,
  - 40% originated from outside the core development team
  - Primary emphasis:
    - Improved support for working with structured models
    - Development of *solvers* in Python/Pyomo
    - (never-ending) core enhancements

Key things not discussed here

- …because they will be covered by subsequent talks
  - Support for robust optimization
  - Mixed integer decomposition (MindtPy)
  - Specialized solvers (SHOT)
  - Tailored nonlinear algorithms in Python (Pynumero)
  - Flexibility analysis in Pyomo
  - Nested decomposition algorithms
  - Disjunctive programming (pyomo.gdp)

Core developments in Pyomo

- We have begun the process of revisiting much of the core of Pyomo with an eye toward performance and scalability
  - First target was the expression system
    - Rewritten to resolve several longstanding issues
      - Added support for the high performance interpreter PyPy
      - Resolved edge case where the following had quadratic runtime:

```
ans = 0
for i in model.INDEX:
    ans = ans + model.x[i]
```

  - Currently working to merge a rewrite of the Set component
  - Next up will be the core solver interfaces

- Goal is to maintain backwards compatibility with the 5.x release series (and the current edition of the Book)

Structured models & nonlinear math programming

Sandia National Laboratories

Qi Chen, Sunjeev Kale [CMU]

- Structured models (hierarchical, block-composed, disjunctive) can obfuscate the NLP through, e.g., a large number of $x = y$ constraints and domain inconsistencies
- Instead of writing a single "nonlinear presolve", Pyomo is adopting a menu-based approach
  - Variable Aggregator
    - Simplify $x = y$ constraints
  - Map Constraints to Variable Bounds
    - Move univariate linear constraints to bounds
  - Induced Linearity Reformulation
    - Detects implicitly discrete variables, linearizes bilinear terms involving a discrete variable
  - Deactivate Trivial Constraints
    - Remove constraints that are trivially feasible
  - Fixed Variable Detection
    - Detect variables that are fixed by $x = p$
  - Fixed Variable Equality Propagator
    - Propagates (tightens) bounds through $x = y$
  - Zero Sum Propagator
    - Fix x_n to 0 when $z = 0$ and $z = \sum x_n$ and $x_n \geq 0$

CCR Center for Computing Research

Composition of structured models

Sandia National Laboratories

Grant Seastream [CMU]

- Several years ago, Pyomo implemented "`Connectors`"
  - Collections of variables that could be manipulated as one object
    - Analogous to the Simulink MUX
  - Model components were "hooked up" by regular constraints over connectors
  - Key challenges:
    - Support for "extensive" variables in the connector (the syntax was awkward)
    - Identifying the network structure (connections were just regular Constraints)
- *Connectors* have been replaced by `pyomo.network`
  - Explicit components: `Port` and `Arc`
    - Ports replace Connectors, individual variables are marked with the expansion function (Equality and Extensive are provided by `pyomo.network`)
    - Arcs indicate (possibly directed) connections between two Ports.
  - With explicit representations of connectivity, we can build algorithms aware of the problem's network structure
    - Sequential modular simulation
      - MIP-based, heuristic-based, and manual tear selection
      - Direct substitution and Wegstein acceleration

CCR Center for Computing Research

Working with structured model hierarchies

Sandia National Laboratories

- Pyomo's `Block` modeling object encourages a composition-based approach to building models:

```
model.plant[i].process[j].heat_exchanger[k].side[l].temperature
model.scenario[i].dispatch_time[j].generator[k].power
```

  - While convenient for constructing complex models and for managing decomposition, reporting becomes challenging
    - …most of the model variables are actually scalars!
  - We support a specialized form of slicing to make iteration easier

```
for p in model.scenario[:].dispatch_time[:].generator[:].power:
    print(p.name, value(p))
```

    - But when slicing you "lose" the indices / "matrix" structure
  - This year we extended that to produce "reference components":

```
model.power = Reference(model.scenario[:].dispatch_time[0].generator[:].power)
model.power[i,j] is model.scenario[i].dispatch_time[0].generator[j].power
model.power.pprint()
```

CCR Center for Computing Research

Creating custom "views" into the model

David Thierry [CMU]

- One of the biggest challenges working with Pyomo models is that our users can – and will – do *anything.*
  - Simply *finding* all the variables in a model can be complex
    - On the model? Sub-blocks? On "special" components (e.g., Disjuncts)?
- Algorithms based on model manipulation can be cumbersome
  - Consider Caprese: MHE+NMPC for

$$\frac{dx(t)}{dt} = F\big(x(t), y(t), u(t), w(t)\big)$$
$$0 = G\big(x(t), y(t), u(t), w(t)\big)$$

  - Find the "time indexed variables"
    - `model.x[t]`
    - `model.unit[t].x`
    - `model.unit[i,t].stream[j].x`

- Instead of working with the model directly, create a new "view" of the model that matches theory/paper

```
z, x = categorize_vars(model)
```

List of simple (unindexed) vars

List of Reference vars, indexed ONLY by t

Only 100 lines of code!

New "meta" (coordinating) solvers

Sandia National Laboratories

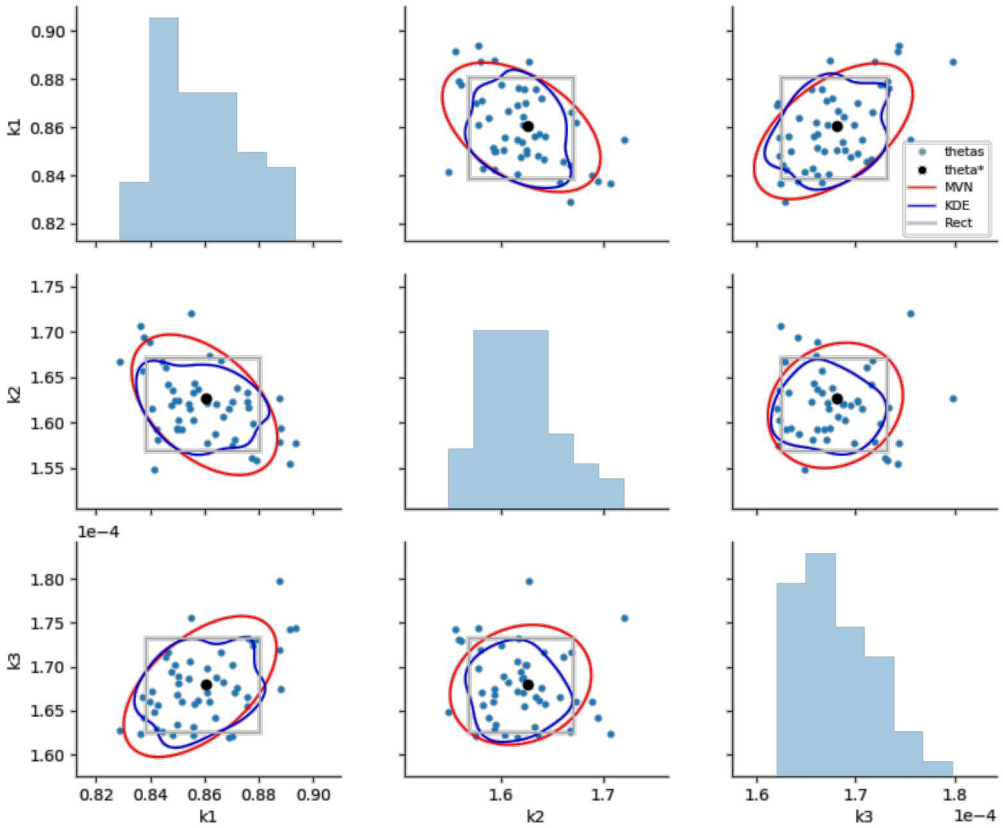
Sunjeev Kale, Qi Chen [CMU]

- Multi-start (local search) solver
  - Multiple strategies for selecting starting points
  - Confidence-based stopping criterion
- GDPbb: "branch and bound" approach for GDP
  - Remove all Disjunctions from the model
  - Walk a search tree, adding a single disjunction at each level in the tree
- GDPopt: Logic-based outer approximation
  - MILP master problem
  - Disjunctive indicator variable-free subproblems
    - (subproblem is LP/NLP if the model contains no other discrete variables)

Parameter estimation toolbox

Andrea Staid, Katherine Klise [SNL], David Woodruff [UCDavis]

- Toolbox for model-based parameter estimation
  - Builds on the PySP stochastic programming package
  - General interface to construct and solve estimation problems
    - Parameter estimation using experimental data
    - Support for bootstrap resampling and likelihood ratio analysis
    - Generate single- and multivariate confidence intervals
  - Key parts of the analysis can be parallelized



Pyomo is growing beyond a single repository

Sandia National Laboratories

- The Pyomo Community continues to grow
  - Significant developments are in external packages built on Pyomo
  - Modeling and solver development
    - CORAMIN – nonlinear relaxation toolbox
    - SUSPECT- special structure detection
  - Domain-specific modeling environments
    - EGRET – electric grid modeling and optimization
    - IDAES – process systems design, control, optimization

Tools for MINLP algorithms

Sandia National Laboratories

F. Ceccon, R. Misener [Imperial]; M. Bynum, C. Laird [SNL]

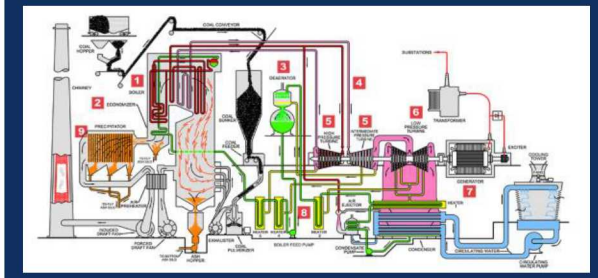
- SUSPECT (https://github.com/cog-imperial/suspect)
  - Special structure detection for Pyomo
  - Converts complete Pyomo model to a DAG
    - Common subexpression identification
  - Extensible tree walker for reasoning on the DAG
    - Feasibility based bounds tightening
    - Convexity detection
- CORAMIN (https://github.com/Coramin/Coramin)
  - Toolbox for constructing MINLP algorithms
  - Standard approaches for constructing convex relaxations
    - Piecewise univariate, McCormick, sin, cos, arctan, etc.
  - Utilities for refining the relaxations
  - Implementation of optimization based bounds tightening

IDAES: Institute for the Design of Advanced Energy Systems

Sandia National Laboratories

Next generation process modeling and optimization platform

Challenge: Develop and utilize multi-scale models and computational tools to accelerate innovation through the design, analysis, optimization, operation and troubleshooting of advanced fossil energy systems

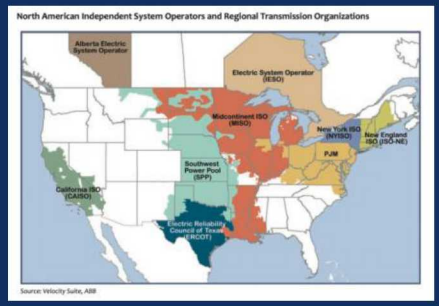


Rigorous Process Modeling and Optimization

- Model library of steady-state and dynamic unit models
- Full flowsheet models to support design and operation improvements of existing fleet (efficiency, flexibility, load)

Grid-integration and infrastructure planning

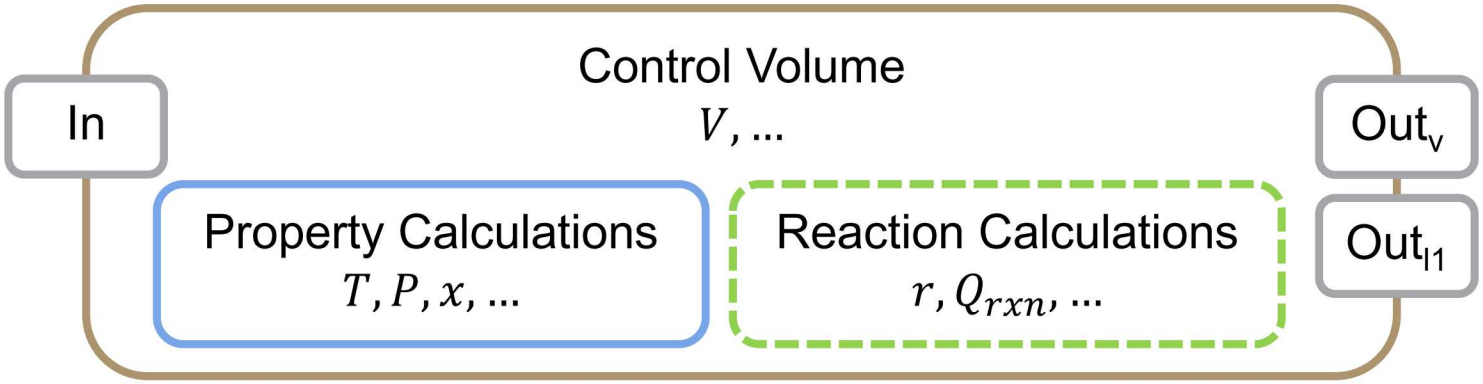
- Grid-level operational models and production cost modeling
- Targets for existing fleet improvements
- Multi-scale infrastructure planning

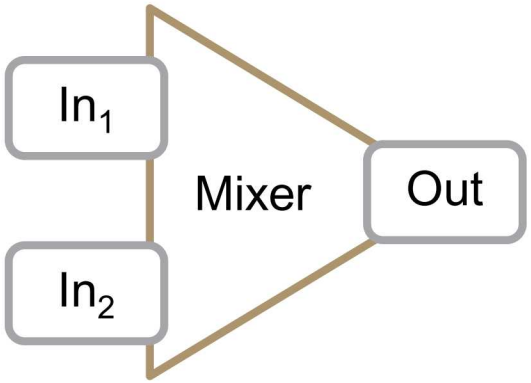


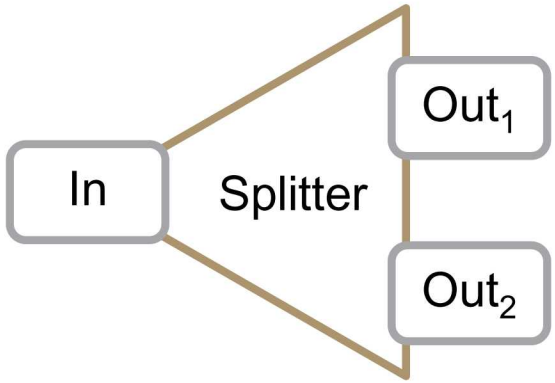
CCR Center for Computing Research

Building blocks of process models

- IDAES makes heavy use of the Pyomo "Block" component
  - Encapsulation, not inheritance



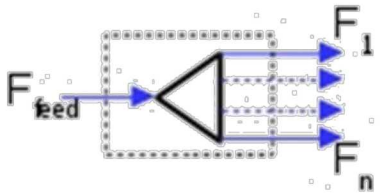




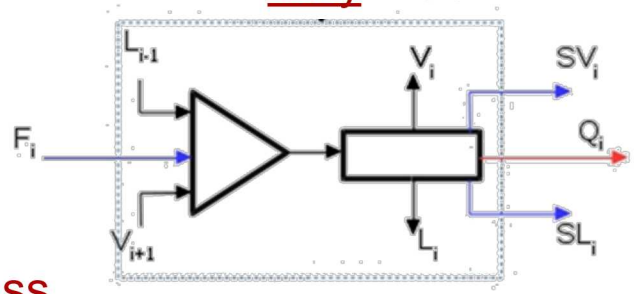
Building blocks of process models

- Basic blocks are combined into conceptual units (also blocks)

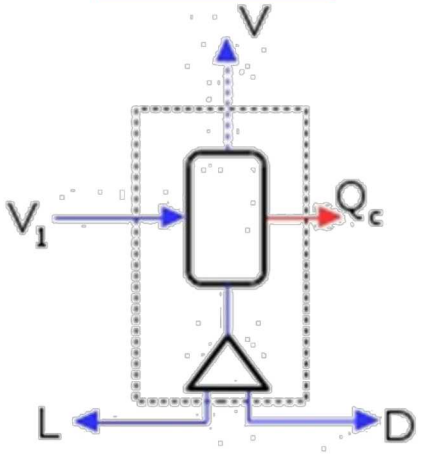
Feed splitter



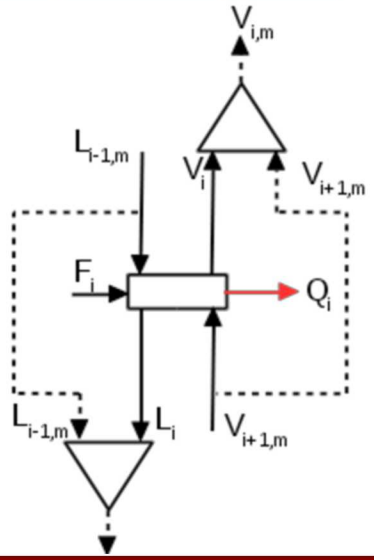
Tray



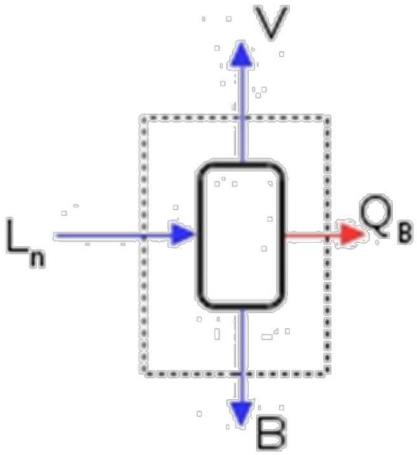
Condenser



Tray with bypass

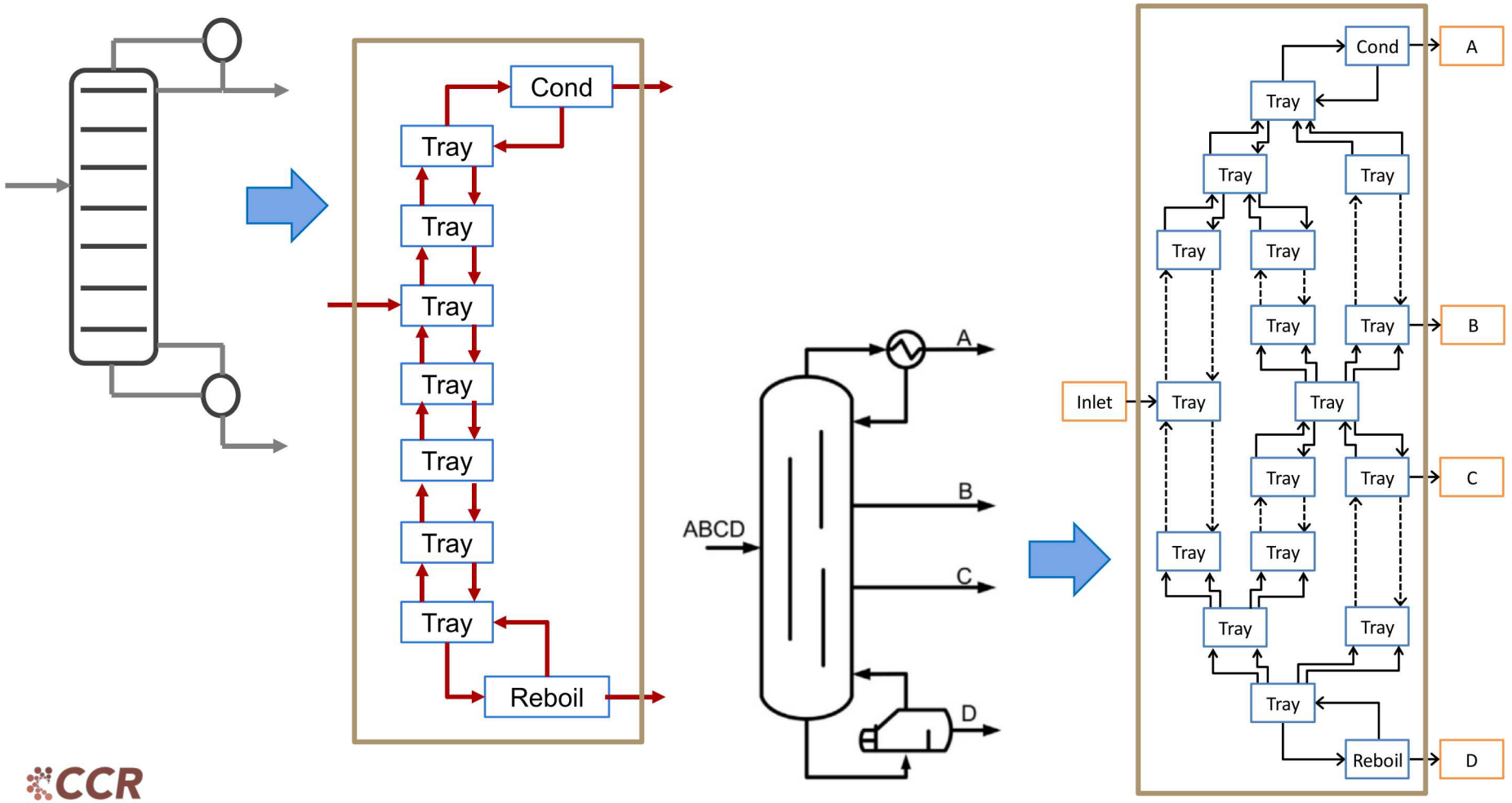


Reboiler



Building blocks of process models

- Complex units then assembled from basic units



Summary

- Pyomo development pace is actually increasing
  - Heavily driven by IDAES, grid modeling and external users
  - Over the last year, work has focused on
    - Improved support for working with structured models
    - Development of solvers / algorithms in Python/Pyomo
    - Core enhancements (which I didn't cover)

- What's next (for a year from now)?
  - I expect work on general algorithms / solvers to continue
  - Improved support for logic modeling
  - PySP rewrite
  - (more) core enhancements

Thank you!

- For more information:
  - www.pyomo.org
  - http://github.com/Pyomo/pyomo
  - pyomo-forum@googlegroups.com

Acknowledgements:

This work was conducted as part of the Institute for the Design of Advanced Energy Systems (IDAES) with funding from the Office of Fossil Energy, Cross-Cutting Research, U.S. Department of Energy