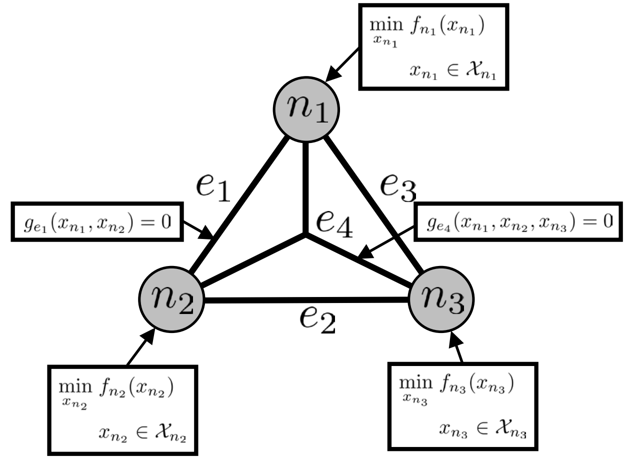
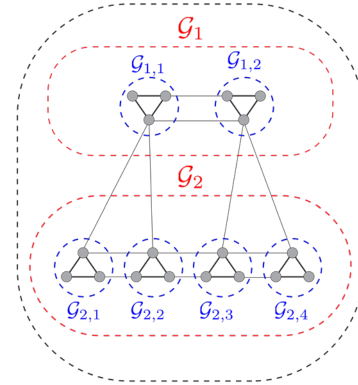




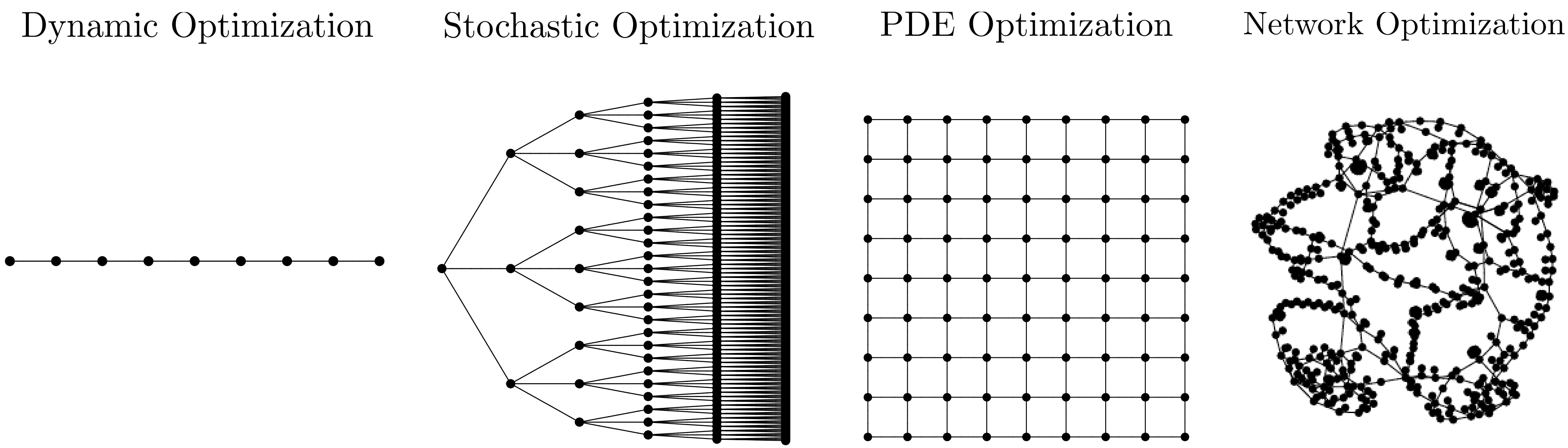
Graph-Based Decomposition Approaches through Plasmo.jl

David L. Cole and Victor M. Zavala

Department of Chemical and Biological Engineering
University of Wisconsin-Madison

JuMP-dev 2024
20 July 2024

Power of Abstraction



- Graphs can be used to represent different optimization formulations

Plasmo.jl Overview

$$\begin{aligned} \min \ & f(\boldsymbol{x}) \\ \text{s.t.} \ & \boldsymbol{x} \in \mathcal{X} \\ & g(\boldsymbol{x}) = 0 \end{aligned}$$

`Plasmo.jl` OptiGraph, $\mathcal{G}(\mathcal{N}, \mathcal{E})$

$$\begin{aligned} \min \ & \sum_{n \in \mathcal{N}(\mathcal{G})} f_x(x_n) \\ \text{s.t.} \ & x_n \in \mathcal{X}_n, \quad n \in \mathcal{N}(\mathcal{G}) \quad \text{node constraints} \\ & g_e(\{x_n\}_{n \in \mathcal{N}(e)}) = 0, \quad e \in \mathcal{E}(\mathcal{G}) \quad \text{edge/link constraints} \end{aligned}$$

- Uses a unifying abstraction called an OptiGraph, containing OptiNodes and OptiEdges
- OptiNodes contain variables, objectives, constraints; OptiEdges contain linking constraints

J. Jalving, S. Shin, V. Zavala. 2022. *Math. Prog. Comp. 14:* 699-747

Plasmo.jl Overview

$$\min \ f(\boldsymbol{x}) \quad \text{s.t. } \boldsymbol{x} \in \mathcal{X}, \quad g(\boldsymbol{x}) = 0$$

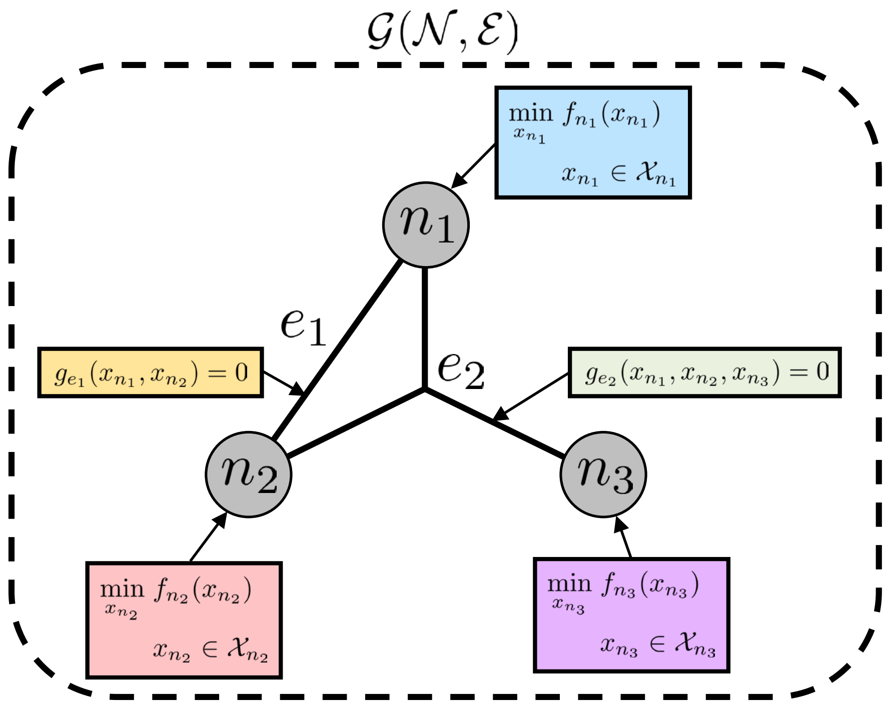
→

`Plasmo.jl` OptiGraph, $\mathcal{G}(\mathcal{N}, \mathcal{E})$

$$\begin{aligned} \min \ & \sum_{n \in \mathcal{N}(\mathcal{G})} f_x(x_n) & & \\ \text{s.t. } & x_n \in \mathcal{X}_n, \quad n \in \mathcal{N}(\mathcal{G}) & & \text{node constraints} \\ & g_e(\{x_n\}_{n \in \mathcal{N}(e)}) = 0, \quad e \in \mathcal{E}(\mathcal{G}) & & \text{edge/link constraints} \end{aligned}$$

$$\begin{aligned} \min \ & f_{n_1}(x_{n_1}) + f_{n_2}(x_{n_2}) + f_{n_3}(x_{n_3}) \\ \text{s.t. } & g_{e_1}(x_{n_1}, x_{n_2}) = 0 \\ & g_{e_2}(x_{n_1}, x_{n_2}, x_{n_3}) = 0 \\ & x_{n_1} \in \mathcal{X}_{n_1} \\ & x_{n_2} \in \mathcal{X}_{n_2} \\ & x_{n_3} \in \mathcal{X}_{n_3} \end{aligned}$$

→



- Uses a unifying abstraction called an OptiGraph, containing OptiNodes and OptiEdges
- OptiNodes contain variables, objectives, constraints; OptiEdges contain linking constraints

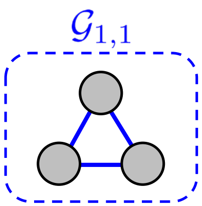
J. Jalving, S. Shin, V. Zavala. 2022. *Math. Prog. Comp. 14:* 699-747

Plasmo.jl Syntax

```
using Plasmo, Ipopt
```

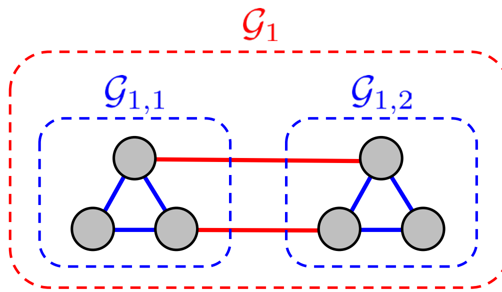
- Plasmo extends JuMP.jl and provides easy, user-friendly interface

Modular and Hierarchical Model Building

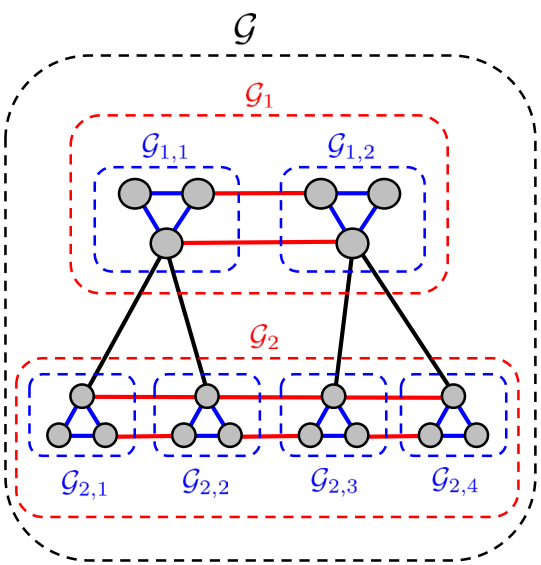



```
g11 = OptiGraph()

@optinode(g11, ....)
@variable(....)
@constraint(....)
@linkconstraint(g11, ....)
```




```
g1 = OptiGraph()
g11 = OptiGraph()
.....
g12 = OptiGraph()
.....
add_subgraph!(g1, g11)
add_subgraph!(g1, g12)
add_linkconstraint(g1, ....)
```



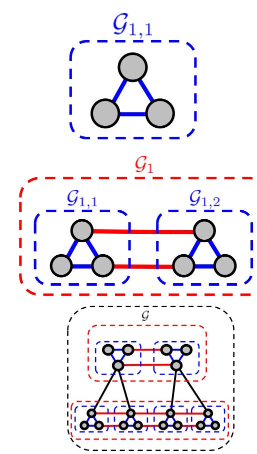

```
g = OptiGraph()

g1 = OptiGraph()
.....
add_subgraph!(g1, g11)
add_subgraph!(g1, g12)

g2 = OptiGraph()
.....
add_subgraph!(g2, g21)
add_subgraph!(g2, g22)
add_subgraph!(g2, g23)
add_subgraph!(g2, g24)

add_subgraph!(g, g1)
add_subgraph!(g, g2)
add_linkconstraint(g, ....)
```

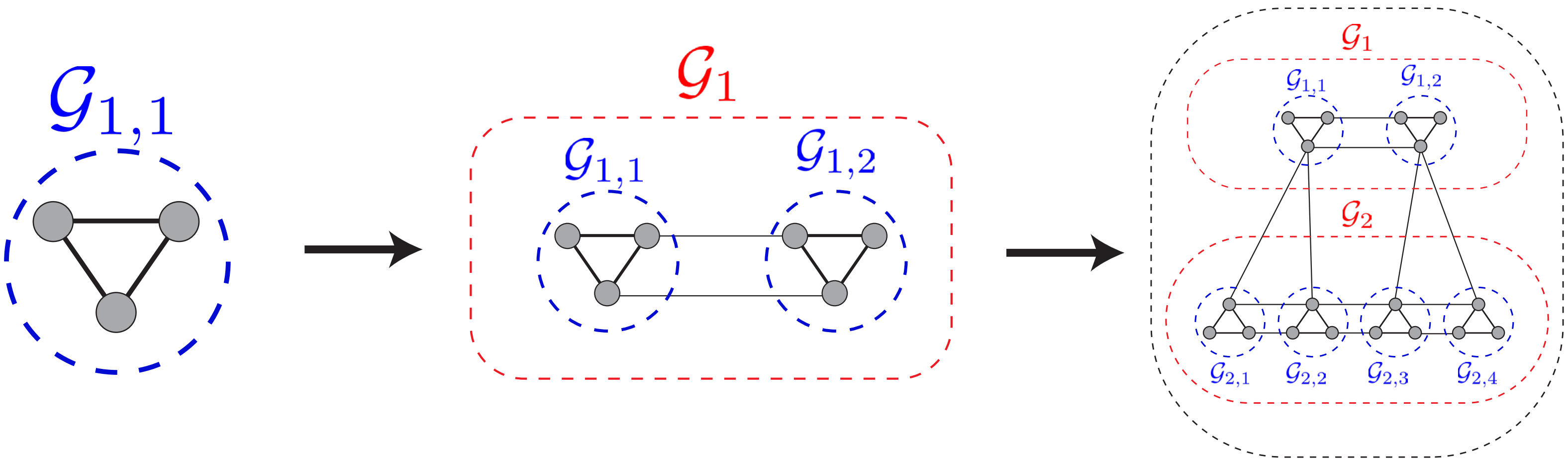
```
optimize!(g11)
optimize!(g1)
optimize!(g)
```



- Graphs can be “nested” as subgraphs within other graphs
- Subgraphs enable modular and hierarchical construction of models
- Each subgraph can be treated as independent optimization problems

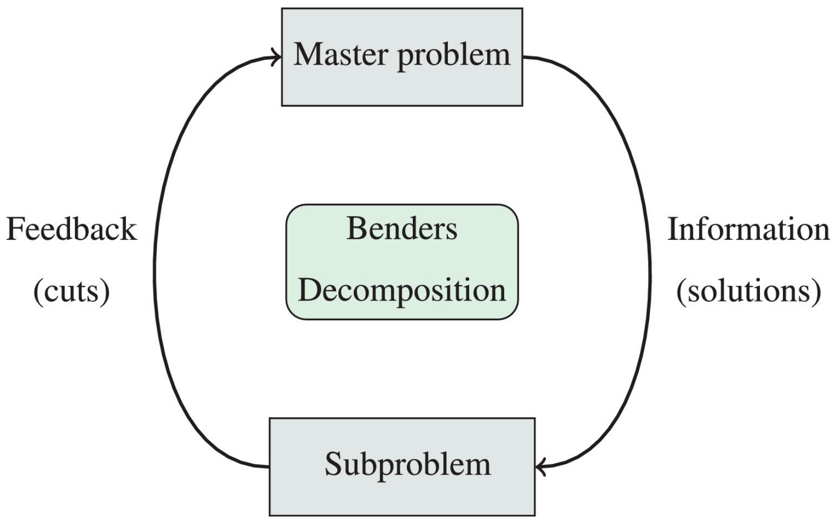
Plasmo.jl v0.6

- More subgraph-centric, better memory usage
- No longer uses `@NLconstraint`, `@NLobjective`, `@NLexpression`
- `LinkConstraintRef` is replaced by a standard MOI constraint
- Hypergraph structure no longer stored internally

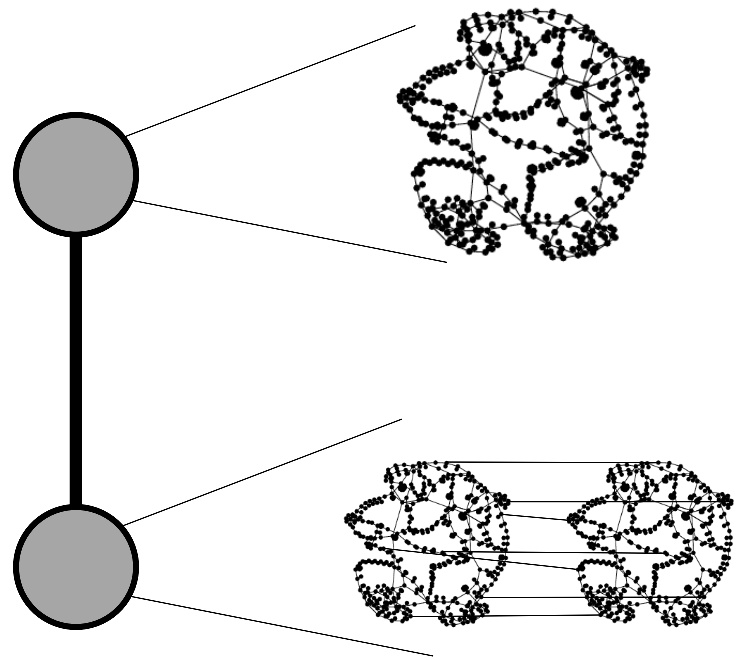


Plasmo.jl Creates Decomposable Structures

Benders Decomposition



Graph Representation

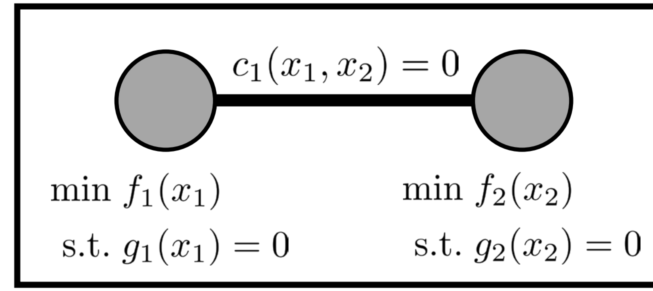


- Benders decomposition (left) can be generalized to a graph structure
- Graphs can provide a basis for applying decomposition schemes

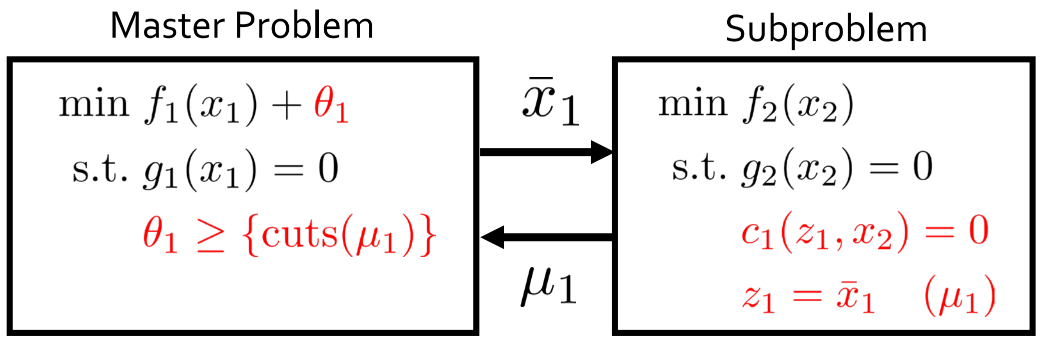
Rahmaniani, Crainic, Gendreau, Rei, 2017. The Benders decomposition algorithm: A literature review. *Eur. J. Oper. Res. 259(3):801-817*

Brunaud, B. 2019. Models and Algorithms for Multilevel Supply Chain Optimization. PhD Dissertation, Carnegie Mellon University.

Benders Decomposition Strategy



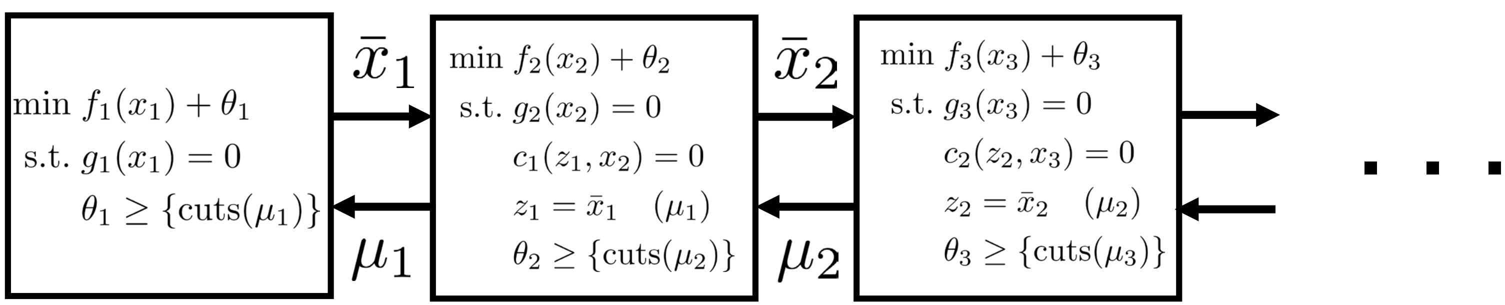
Benders



$$\text{Upper Bound} = \sum_{i=1}^{N} f_i(\bar{x}_i)$$

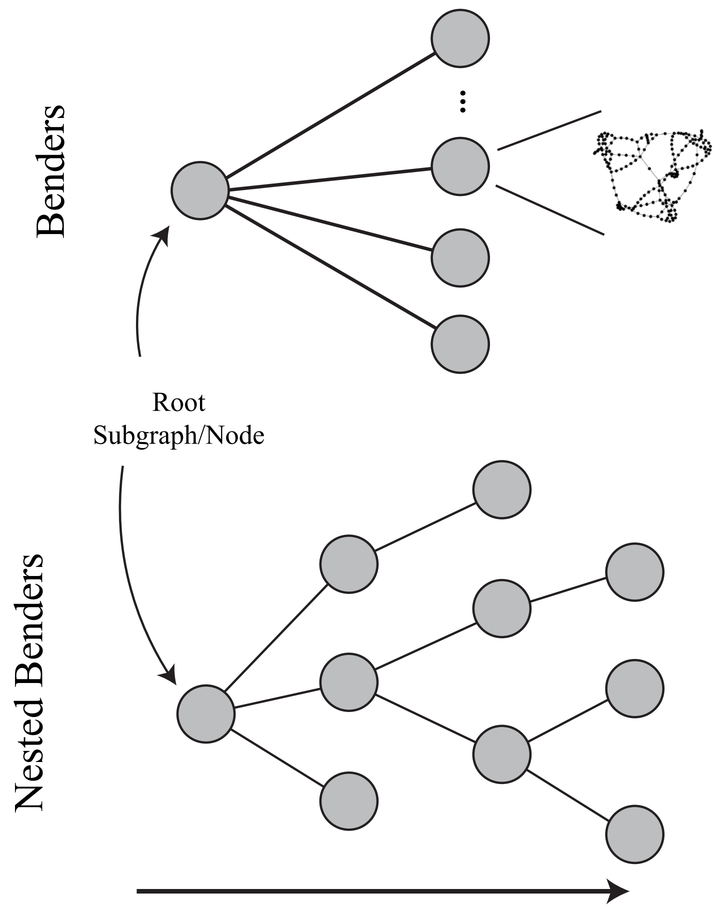
$$\text{Lower Bound} = f_1(\bar{x}_1) + \theta_1$$

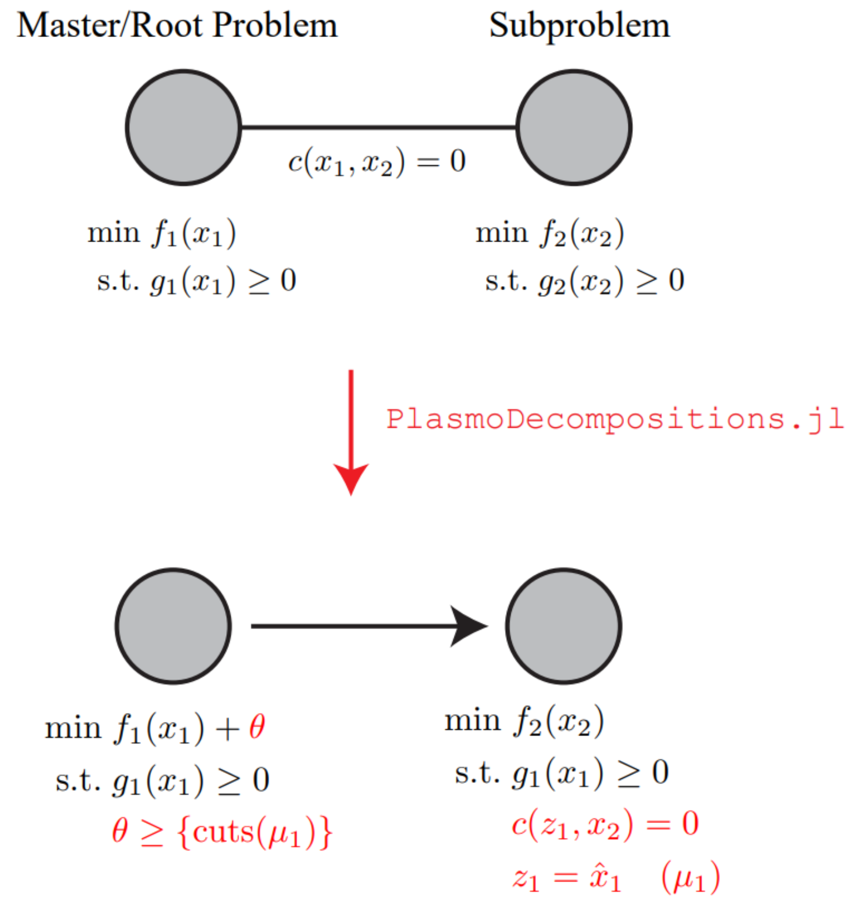
Nested Benders



- θ = cost-to-go – lower bound on subproblem objective
- Problems solved sequentially, duals passed backwards for cuts

PlasmoDecompositions.jl

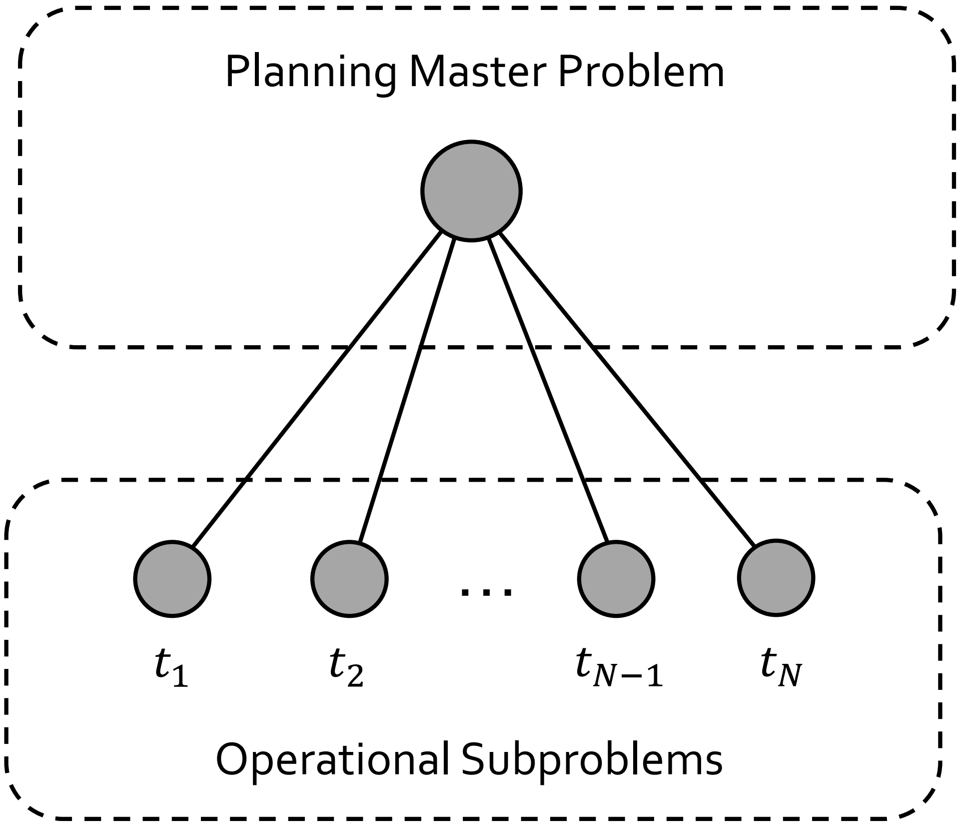




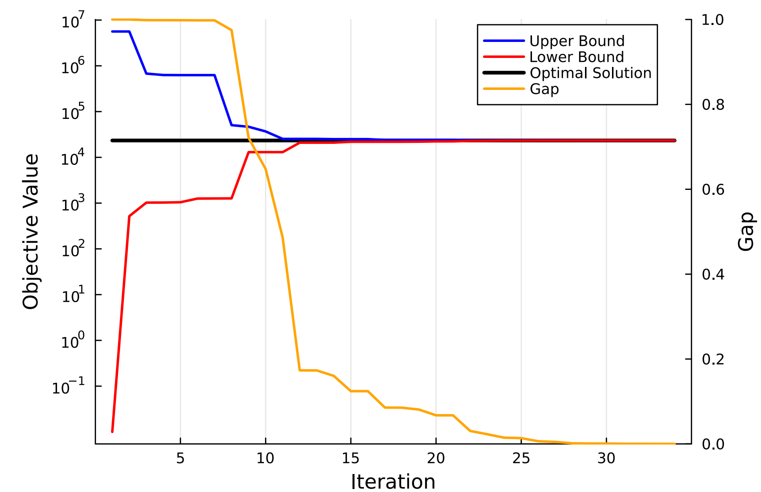
- PlasmoDecompositions.jl takes overall graph and root/master problem as arguments
- Detects links, adds cost-to-go/cuts, performs forward/backward passes
- Graphs are *agnostic* to a domain (e.g., temporal/spatial)

Example: Benders for Capacity Expansion

GenX.jl



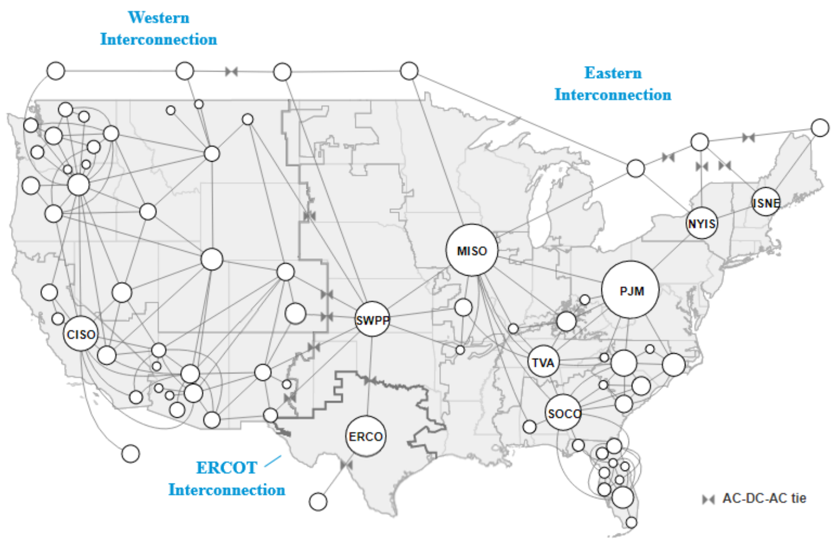

```
using PlasmoDecompositions
ddpopt = DDPOptimizer(
    g,
    master_graph,
    regularize = true,
    multicut = true,
    add_slacks = false,
    parallelize_benders = true,
    strengthened = true,
    tol = 1e-4
)
JuMP.optimize!(ddpopt)
```



- Proof of Concept Example: ~120k variables, 11 subproblem graphs
- Subproblems are independent and can be solved in parallel
- Overall solution time was the same as w/o decomposition when using HiGHS

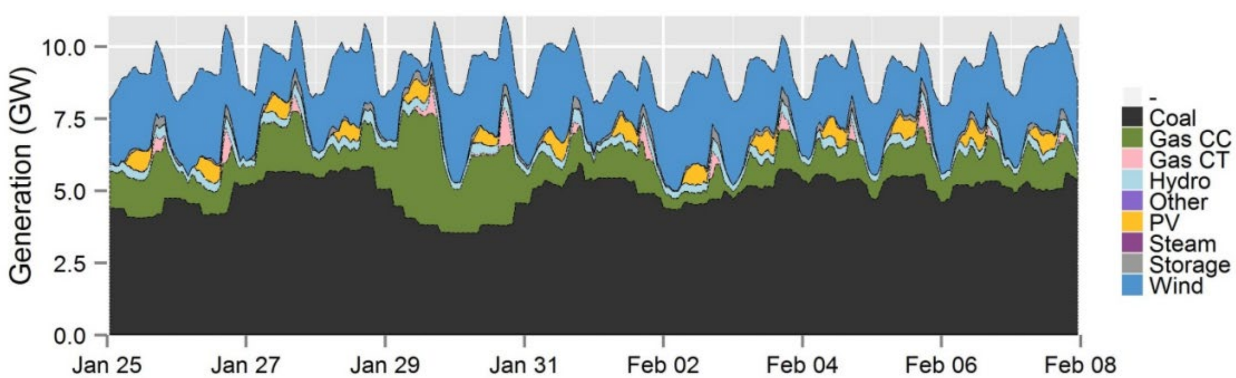
Pecci, F. and Jenkins, J.D. "Regularized Benders Decomposition for High Performance Capacity Expansion Models." *arXiv:2403.02559*

Example: Production Cost Modeling (PCM)



US Energy Information Administration, Hourly Electric Grid Monitor

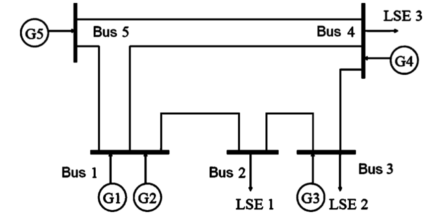
min	Start-Up/Shut-Down Costs + Operation Costs
s.t.	Start-Up Constraints
	Shut-Down Constraints
	Ramp Up/Down Constraints
	Capacity Constraints
	Transmission Constraints
	Battery Constraints

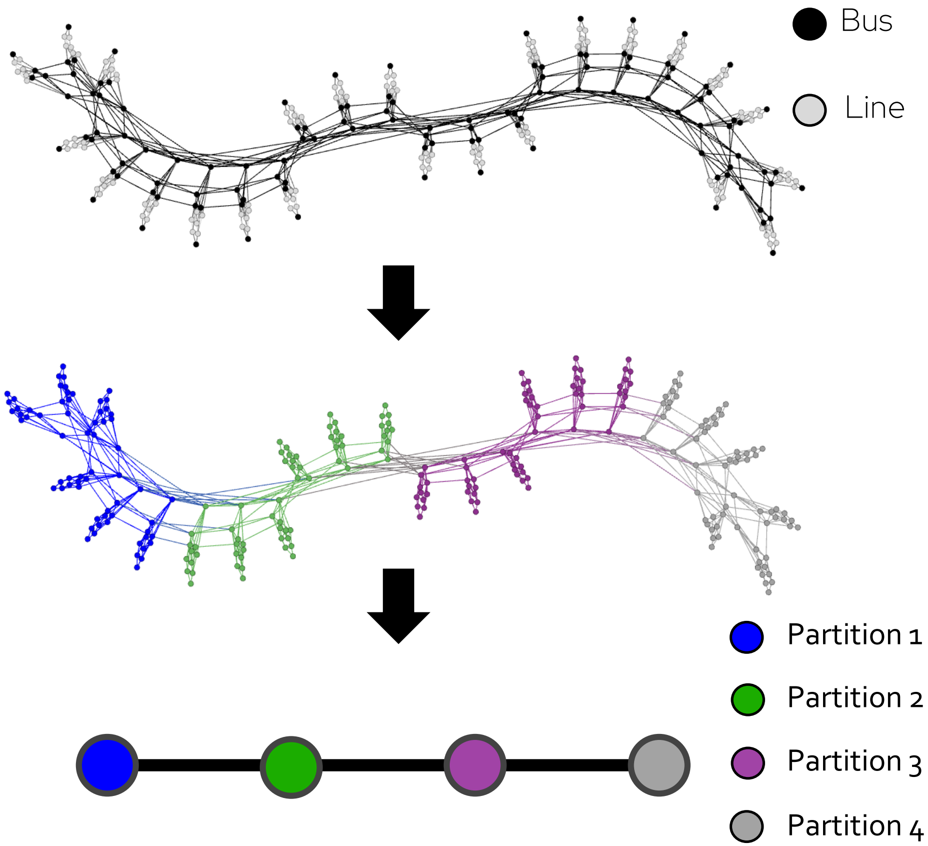


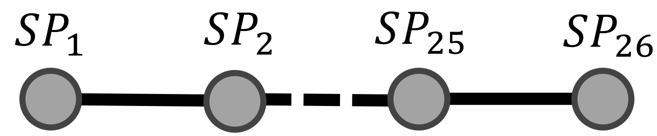
Barrows et al., 2014. *NREL/TP-6A20-60969*

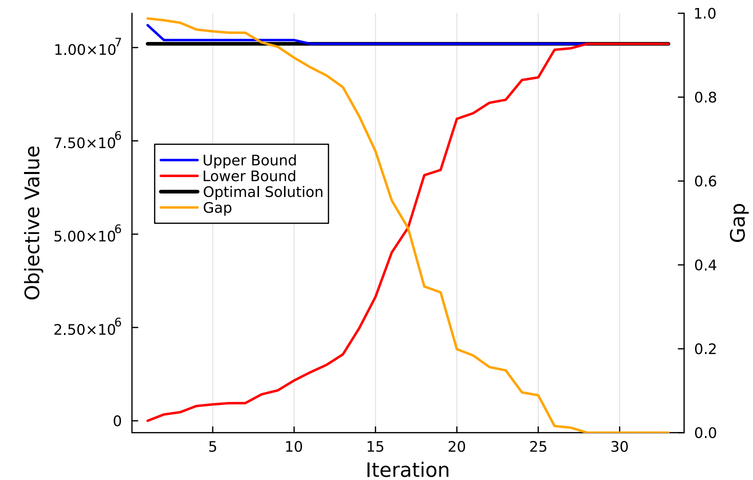
- MIP for determining cost of operating a power network
- PCM minimizes costs of generator startup, shutdown, and operation

PCM – 5-Bus System as an OptiGraph







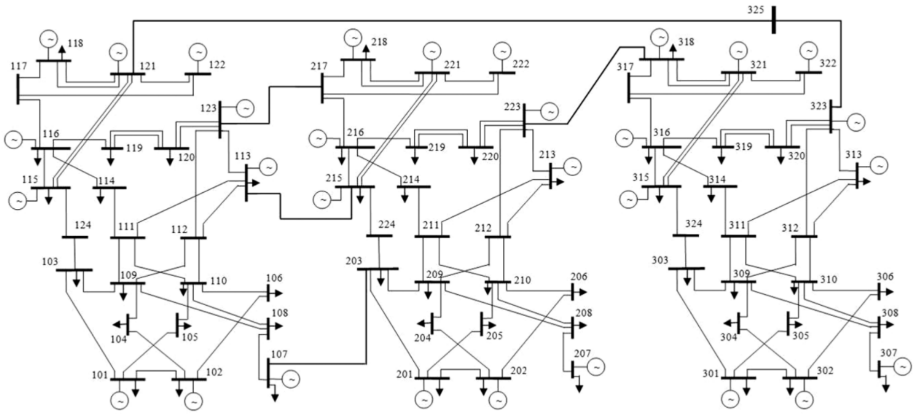


	NBD	Gurobi	
Gap	0.20%	0.95%	0.50 %
Solve Time (min)	58	43	240
Max Memory (GB)	14.4	42.3	44.2

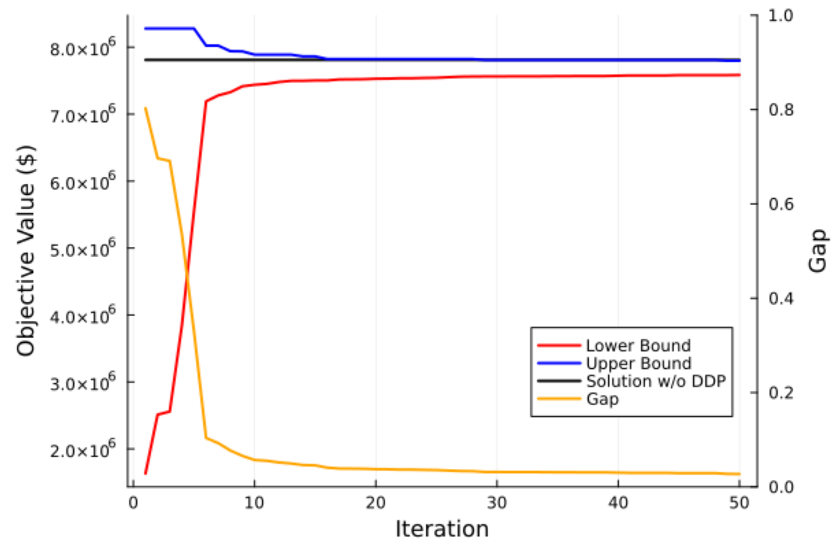
- Graph can be aggregated to create tree structure
- 52-week problem solved with Nested Benders (NBD) using 26 subproblems (2 weeks each)
- NBD used less than half the memory of Gurobi, and required less time to solve

Li, H., Sun, J., & Tesfatsion, L. In *Computational Methods in Economic Dynamics*, vol 13, 135-158

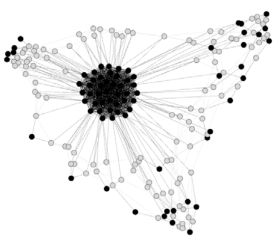
PCM – 73-Bus System as an OptiGraph



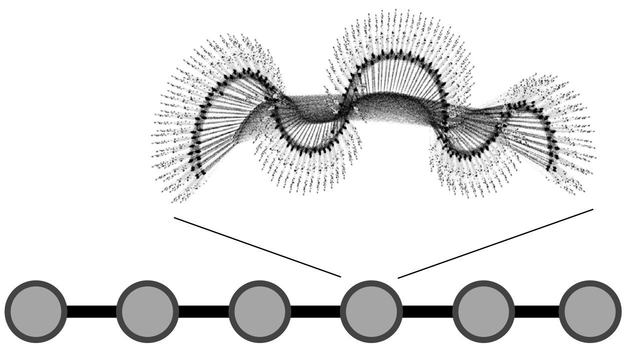
Gonzalez-Fernandez et al., *IEEE Trans. Power Syst.* 28, 4598–4606 (2013).



1 Time Point



31-day problem (with aggregation)

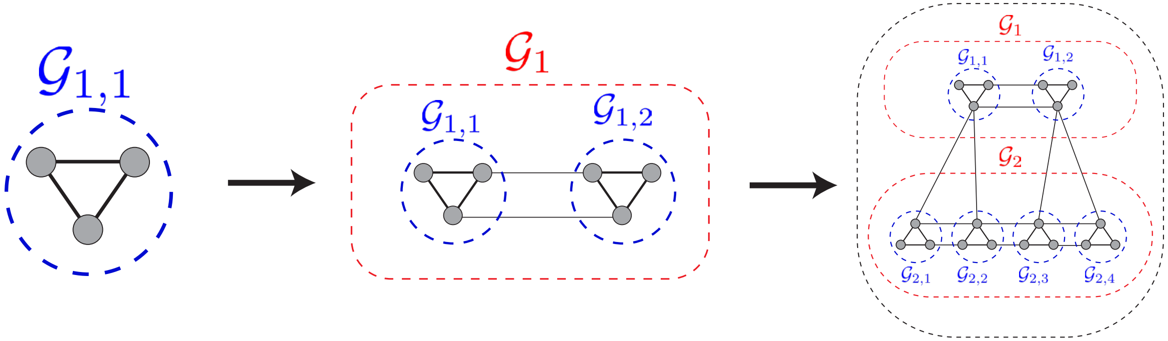


	NBD	Xpress
Gap	2.7%	1.2%
Objective Value	$7.798e6	$7.810e6
Solve Time (hr)	34.8	11.5
Max Memory (GB)	74	180

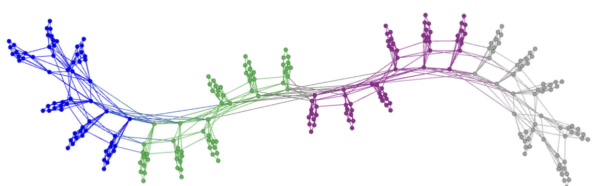
- 73-bus system has higher complexity of optimization and structure
- Solving with NBD over 31-day total horizon required less overall memory
- Decomposition enables use of non-commercial solvers

Conclusions and Future Work

- Graphs/Plasmo.jl give a general framework for structured optimization

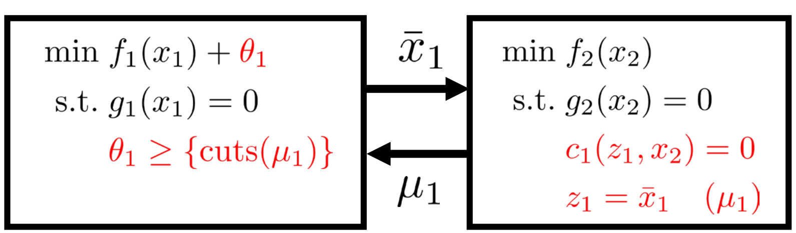


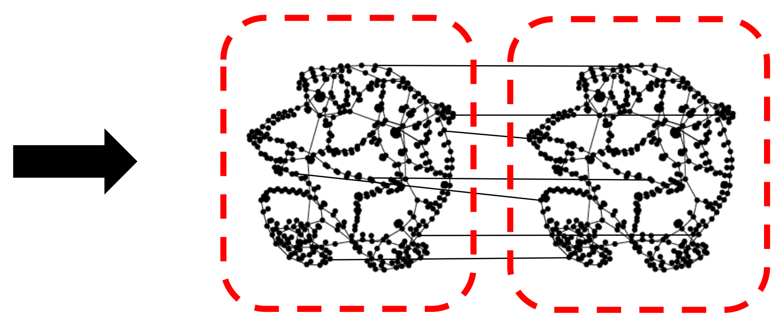
- PlasmoDecompositions.jl facilitates scalable decomposition strategies



	DDP	Gurobi	
Gap	0.20%	0.95%	0.50 %
Solve Time (min)	58	43	240
Max Memory (GB)	14.4	42.3	44.2

- Graphs can help identifying application of decomposition schemes





Acknowledgements

Dr. Victor Zavala

Dr. Jordan Jalving

Dr. Harsha Gangammanavar

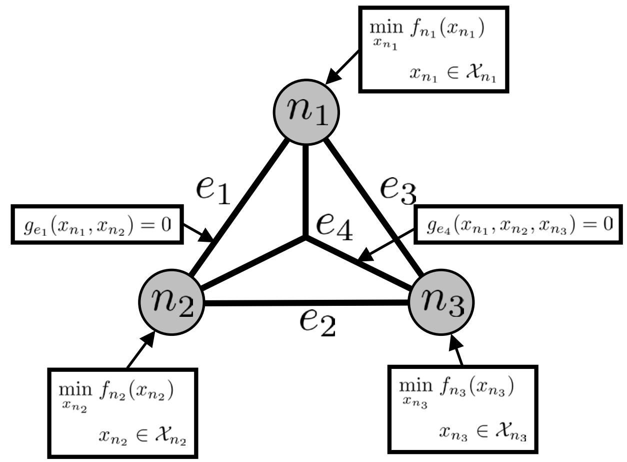
Dr. Omar José Guerra-Fernandez

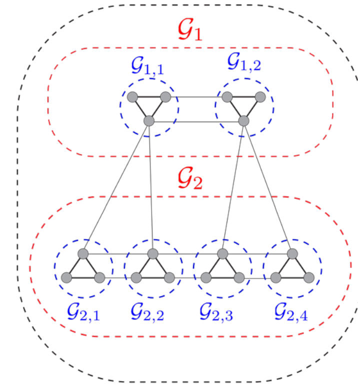
- Cole, D.L., Gangammanavar, H., and Zavala, V.M. 2023. Hierarchical graph modeling for multi-scale optimization of power systems. arXiv:2309.10568

- Cole, D.L. and Zavala, V.M., 2024. PlasmoData.jl – a Julia framework for modeling a and analyzing complex data as graphs. arXiv:2401.11404

- Github: dlcole3
- Email: dlcole3@wisc.edu

This material is based upon work supported by the National Science Foundation under award CBET-1748516.





Graph-Based Decomposition Approaches through Plasmo.jl

David L. Cole and Victor M. Zavala

Department of Chemical and Biological Engineering
University of Wisconsin-Madison

JuMP-dev 2024
20 July 2024