Pacific Northwest
NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

Distributed Solution Algorithms for Security Constrained Unit Commitment in Evolving Day Ahead Electricity Markets

JESSE HOLZER (PRESENTER)

PNNL, MISO, GE, GUROBI

FERC TECHNICAL CONFERENCE, JUNE 27, 2018

U.S. DEPARTMENT OF ENERGY

SCUC trends and algorithms

- Security Constrained Unit Comment (SCUC) for day ahead wholesale energy and reserves
  - Large ISO
  - Many generators with nonconvex operating costs and constraints ~ 1300
  - Many security constraints – line flow bounds, contingencies ~ 250 per time period
  - Many other convex resources – virtual bids, dispatchable demand ~ 20000 per time period
- Market trends
  - Larger systems
  - More and smaller generators – e.g. natural gas turbines rather than coal
  - More virtuals and dispatchable demands
  - Larger problem, more difficult for MIP solvers, but less severely nonconvex
- Developing algorithms adapted to the difficult features of the problem in the present and the future
  - RINS-E (E: emulation/enhanced)
  - ADMM-subsystems

RINS-E

- RINS = relaxation induced neighborhood search, most good solutions found in Gurobi use RINS.
- RINS-E = RINS emulated/enhanced.
  - Integer solutions and relaxation solutions from main Gurobi run generate subproblems.
  - Subproblems solved in parallel, asynchronous to main Gurobi run.
  - Specialized techniques for solving the RINS subproblems. Set loose security constraints as lazy. Fix some virtual bid and dispatchable demand variables to decrease the subproblem size. Unfix some of the fixed variables and resolve.
  - Subproblem techniques specifically adapted to evolving market characteristics.
- Showed substantial improvement vs. Gurobi MIP solver
  - In best solution found by 1200 sec.
  - In time to 1% (resp. 0.5%, 0.2%, 0.1%, 0.05%) gap to true optimal value

RINS-E subproblem procedure

- RINS subproblem:
  - Given XREL and XINC, fix all discrete variables with matching values and reoptimize remaining variables, obtaining solution XSOL.
  - This subproblem is expensive.
  - We have developed heuristics to quickly obtain a good solution.
- RINS-E subproblem procedure:
  - Given XREL and XINC, fix all variables with matching values to a given tolerance and reoptimize remaining variables, with XINC as a MIP start, obtaining solution X1.
  - If no improvement from XINC to X1, set XINT = X1, go to END.
  - Fix matching variables among only discrete variables and VDD, use X1 to set SC with large slack as lazy, reoptimize with X1 as a MIP start, obtaining new solution X2.
  - If no improvement from X1 to X2, set XINT = X2, go to END.
  - Fix matching variables among only discrete variables and VDD with narrow bounds, use X2 to set SC with large slack as lazy, reoptimize with X2 as a MIP start, obtaining new solution X3, set XINT = X3, go to END.
  - END: Fix all discrete variables in XINT, reoptimize continuous variables, return solution XSOL.

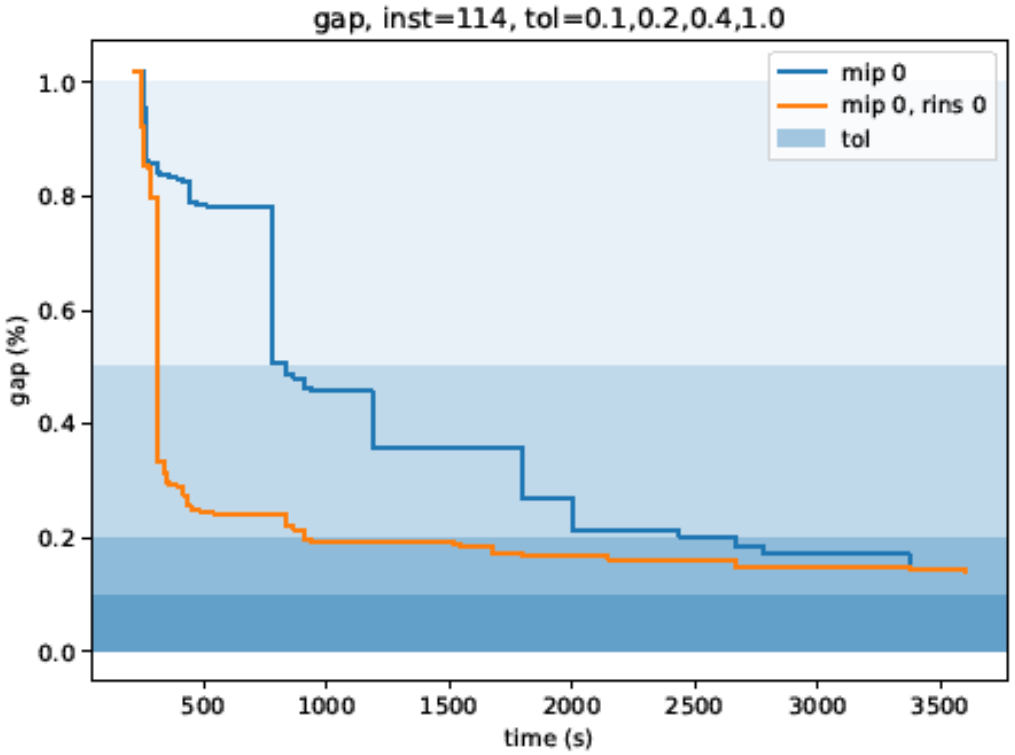
MIP vs RINS-E
Time to 0.05% gap with true opt

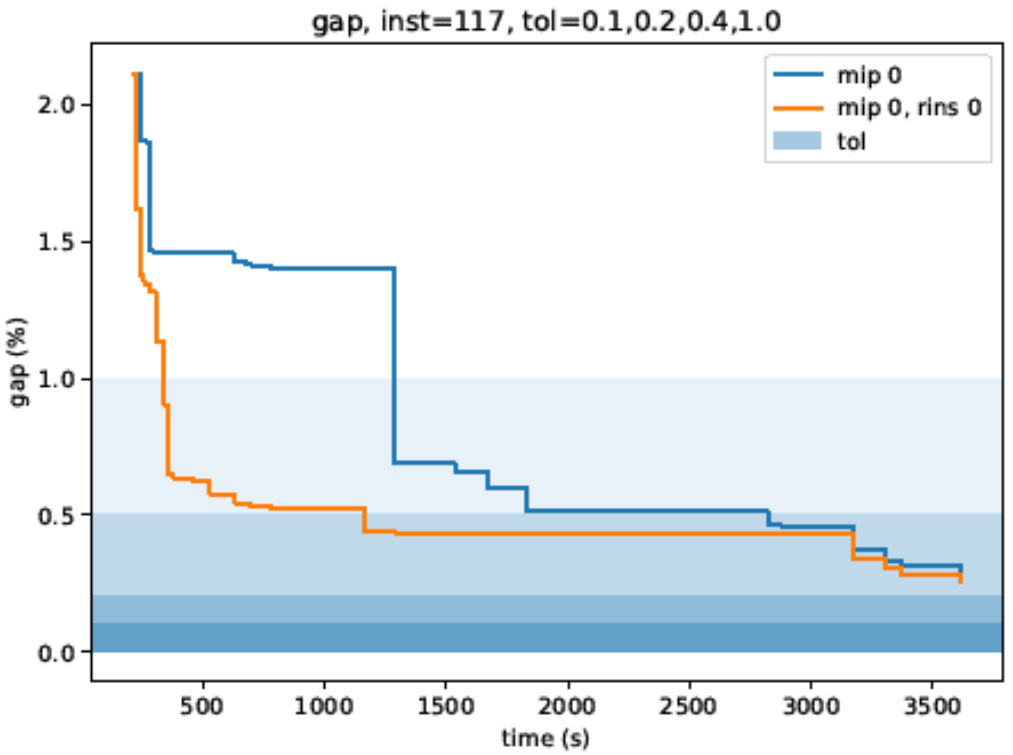
- RINS-E: large speedup in time to 0.05% gap to opt, often 2x.
- Note significant variation in performance of both methods with respect to formulation and seed.

form	6	6	0	0	6	6	0	0		
seed	0	1	0	1	0	1	0	1		
inst	mip 0	mip 1	mip 2	mip 3	rins 0	rins 1	rins 2	rins 3	rins/mip 0	rins/mip best
102	-	410	471	-	334	378	-	-	0	0.814634
114	2436	3008	2386	2078	435	729	638	777	0.178571	0.209336
115	1649	658	1135	1857	604	846	528	858	0.366283	0.802432
116	952	1697	2153	2367	1101	1683	2356	1700	1.156513	1.156513
117	2824	3492	2940	3703	1164	2006	748	732	0.412181	0.259207
119	1393	1407	1050	1089	881	611	1033	1162	0.632448	0.581905
121	1319	2148	2871	3649	2966	1783	1519	1696	2.248673	1.15163
130	1125	1381	1348	1525	538	694	824	1012	0.478222	0.478222
215	3537	3050	1079	-	3536	3335	2964	2340	0.999717	2.168675
220	2053	1592	-	1824	384	381	449	674	0.187043	0.239322

MIP vs RINS-E: time to mipgap=X%
Significant improvement for different X

- MIP/RINS-E makes rapid progress, hitting various mipgap values well before MIP.
- By the time mipgap=0.1% is achieved, MIP catches up.

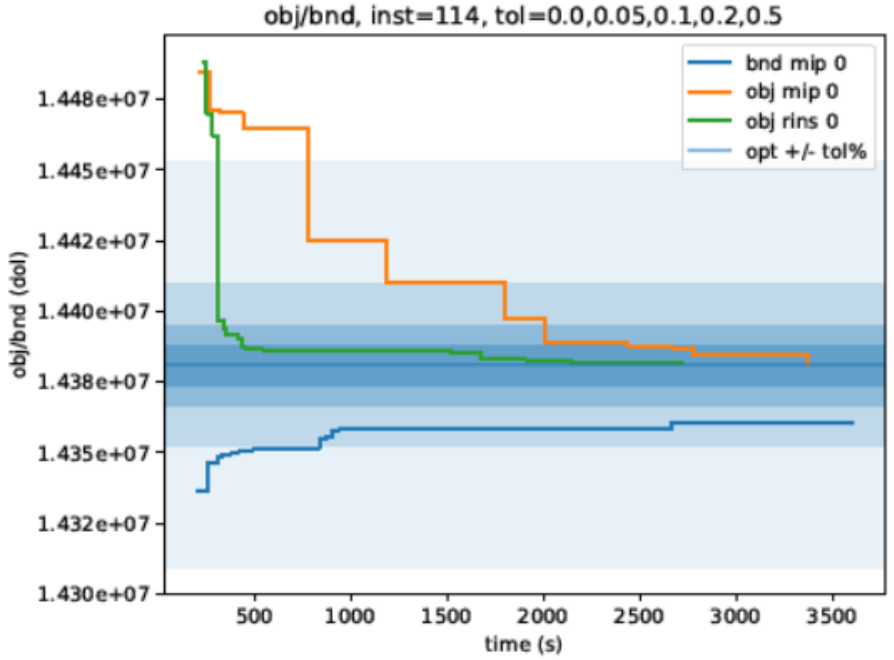


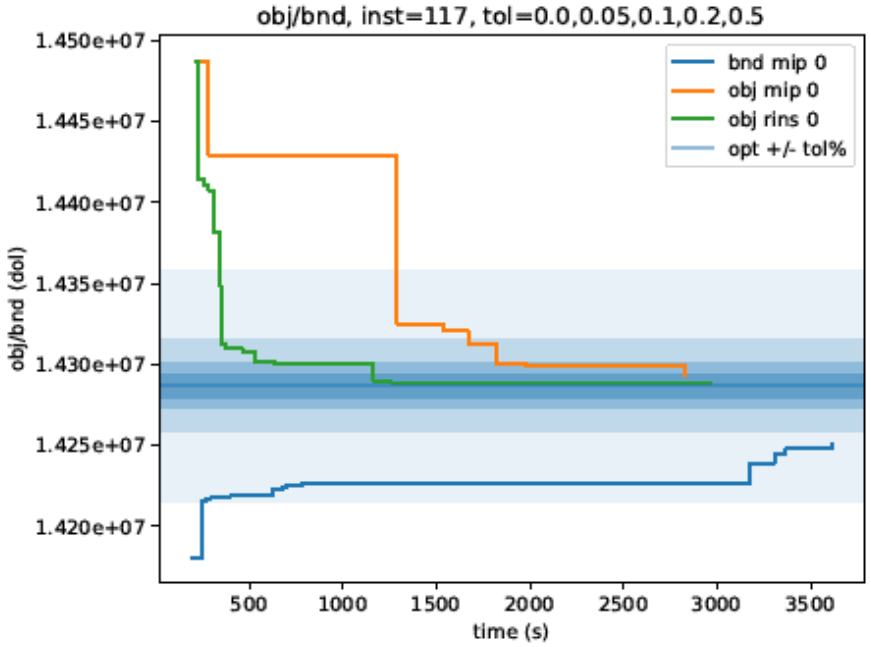


MIP vs RINS-E: time to ubgap=0.05%
Significant improvement

- RINS-E makes rapid progress in the upper bound.
- Hits 0.05% gap to opt well before MIP.
- Progress stalls because it is not possible to get a significantly better solution (already at 0.05% to opt).
- Lower bound (from MIP) progresses slowly.

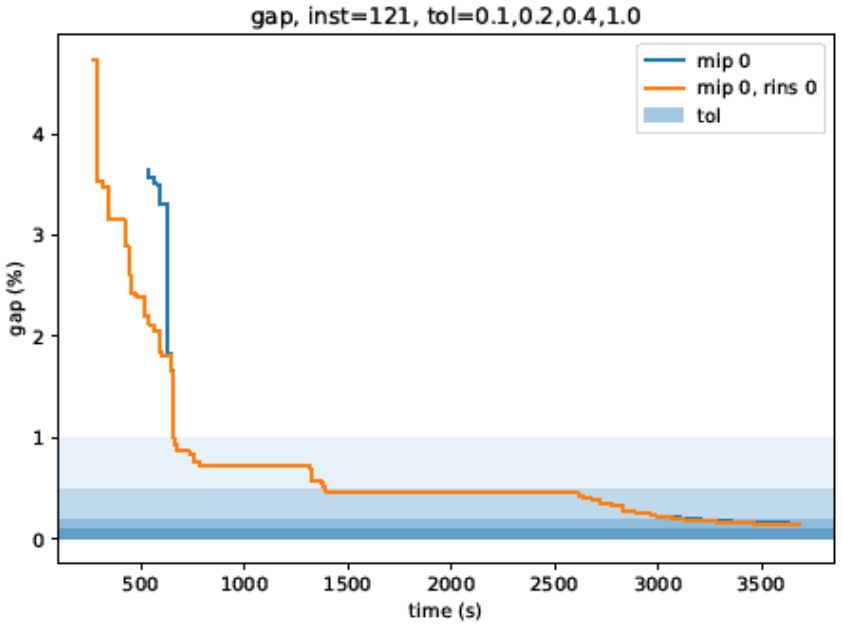
- Before the lower bound reaches 0.1% gap to opt, MIP upper bound catches up to RINS-E.
- Thus for time to 0.1% mip gap (ub – lb), RINS-E shows no improvement over MIP.
- We need a faster/stronger lower bound
- Clearly RINS-E can be of value in a business context

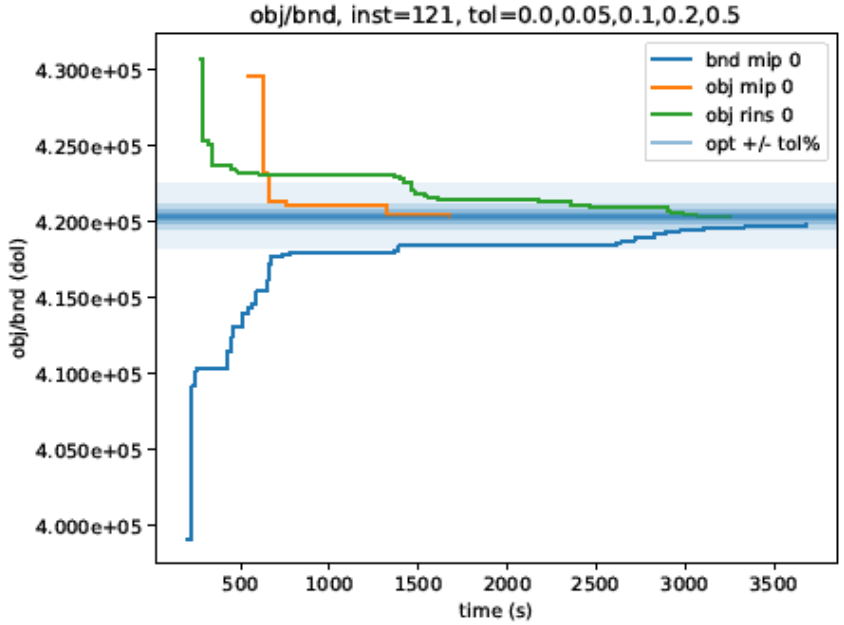




MIP vs RINS-E
Not all cases show improvement

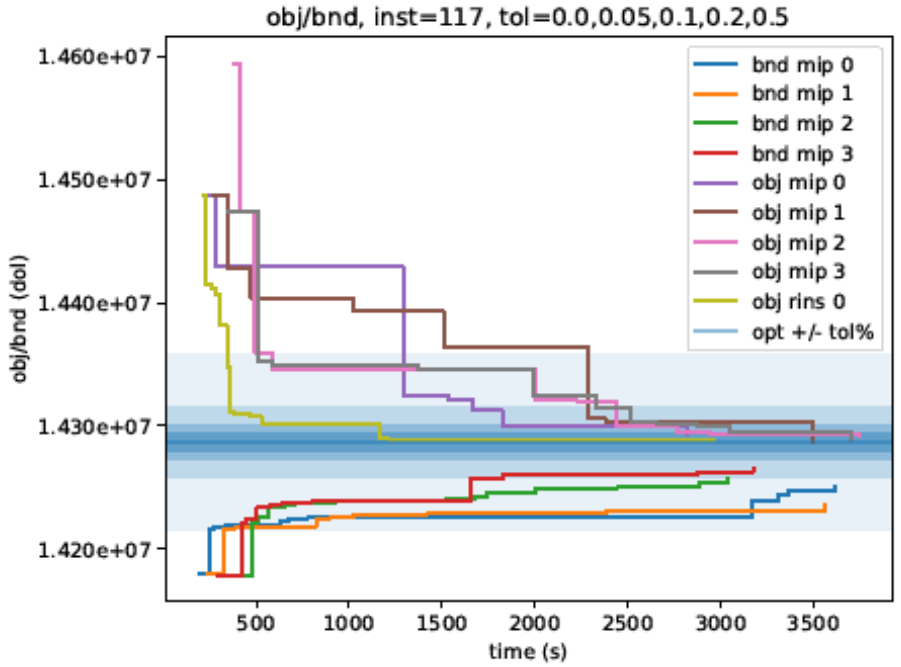
- An unfavorable result for RINS-E. RINS-E uses a Gurobi MIP run to generate its subproblems. How can the result of a standalone MIP run be so different (and better)?
  - The MIP run in RINS-E is limited to 8 threads, while the standalone run is 16.
  - Externalize the MIP run through HIPPO concurrent solver framework.
  - At worst concurrent will have the upper bounds from both MIP and RINS-E.

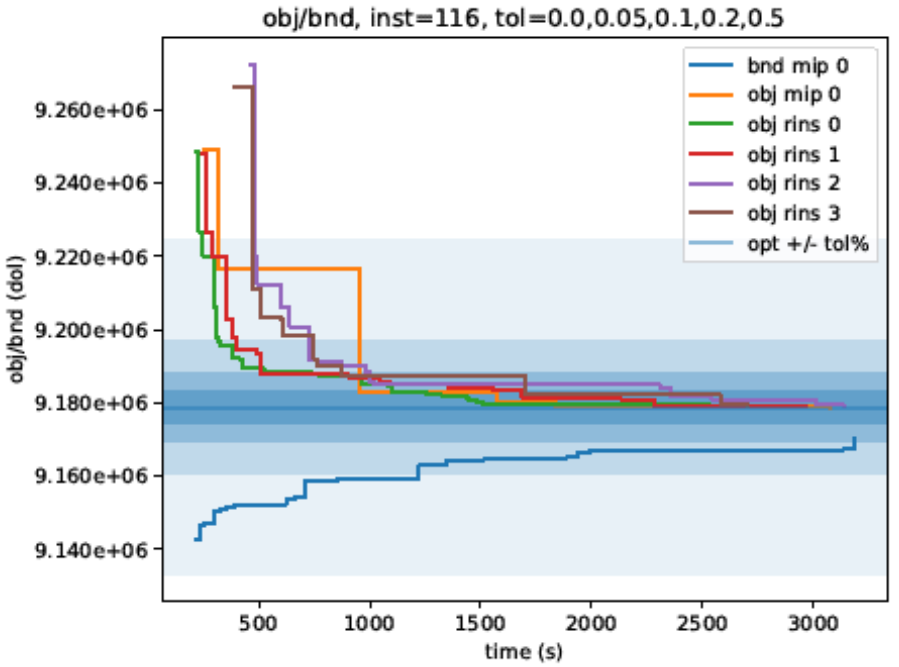




MIP and RINS-E: Performance variation with respect to seed and formulation

- Needs more study – these observations are not systematic.
- MIP: better lower bound from dense formulation (2,3)
- MIP: More consistent upper bound from sparse formulation (0,1)
- RINS-E: Generally less variation, sparse formulation looks better.





ADMM Subsystem Decomposition

- Partition resources (generators, virtuals, dispatchable demands) into subsystems *s*.
- Subsystem *s* solves a subproblem trying to meet a subsystem-specific demand Y_s for energy and reserves.
- Express system-wide constraints (energy and reserve balance, security) in terms of Y_s, not the individual resource dispatch values. This may be an approximation.
- Solve the subsystem subproblems in alternation with the system-wide constraints (ADMM). Some virtuals and dispatchable demands may be included in the system constraint subproblem.
- At each iteration of ADMM, extract the values of the integer variables and quickly solve a SCED to determine a feasible solution of SCUC.
- Construct subsystems by clustering resources based on their coefficients in the system constraints.
- Larger subsystems are more difficult to solve. But the behavior of Y_s is less erratic, closer to convex case, enabling better convergence of the overall method.
- Even single generator subsystem decomposition can be expected to perform better in future power systems with more and smaller resources.

ADMM Subsystem Decomposition Standard Formulation

Notation

- R – resources (each individual generator or virtual or dispatchable demand bid)
- T – time periods
- C – (system-wide) constraints (energy balance and security constraints)
- X_{rt} – energy dispatch from resource r in time t
- A_{crt} – coefficient of resource r in constraint c in time t
- H_{ct} – penalty function for constraint c in time t
- F_{rt} – cost function for resource r in time t (energy, no load, etc.)
- W_r – resource r feasible set (gen bounds, ramping, etc)

Basic model

Minimize

$$\sum_{rt} F_{rt}(X_{rt}) + \sum_{ct} H_{ct}\left(\sum_{r} A_{crt} X_{rt}\right)$$

Subject to

$$X_r \in W_r \text{ for all } r \in R$$

ADMM Subsystem Decomposition
Subsystem Formulation

Notation

- S – subsystems
- R_s – resources contained in subsystem s
- R_0 – global resources, not contained in any subsystem, no nonconvex or intertemporal constraints or costs
- B_{cst} – coefficient of subsystem s in constraint c in time t
- Y_{st} – total energy dispatch from resources in subsystem s in time t
- Z_{st} – copy of Y
- W_{rt} – feasible set for X_{rt}, $r \in Q$

Assumptions

- The sets R_0 and R_s, $s \in S$, form a partition of R
- $W_r = \Pi_{t \in T} W_{rt}$ for all $r \in R_0$

Subsystem formulation

- Minimize
  - $\sum_{rt} F_{rt}(X_{rt}) + \sum_{ct} H_{ct}(\sum_s B_{cst} Z_{st} + \sum_{r \in Q} A_{crt} X_{rt})$
- Subject to
  - $X_r \in W_r$ for all $s \in S$, $r \in R_s$
  - $X_{rt} \in W_{rt}$ for all $r \in R_0$, $t \in T$
  - $Y_{st} = \sum_{r \in Rs} X_{rt}$ for all $s \in S$, $t \in T$
  - $Y_{st} = Z_{st}$ for all $s \in S$, $t \in T$

ADMM Subsystem Decomposition
Augmented Lagrangian Relaxation

Notation

P_{st} – multiplier on linking constraints $Y_{st} = Z_{st}$ for subsystem s in time t

ρ – regularization parameter, > 0

Augmented Lagrangian relaxation

Minimize

$$\sum_{rt} F_{rt}(X_{rt}) + \sum_{ct} H_{ct}(\sum_{s} B_{cst} Z_{st} + \sum_{r \epsilon Q} A_{crt} X_{rt}) + \sum_{st} (P_{st}(Y_{st} - Z_{st}) + \rho/2 \, |Y_{st} - Z_{st}|^2)$$

Subject to

$X_r \in W_r$ for all $s \in S$, $r \in R_s$

$X_{rt} \in W_{rt}$ for all $r \in R_0$, $t \in T$

$Y_{st} = \sum_{r \epsilon Rs} X_{rt}$ for all $s \in S$, $t \in T$

ADMM Subsystem Decomposition Algorithm

Given (X^k, Y^k, Z^k, P^k), compute $(X^{k+1}, Y^{k+1}, Z^{k+1}, P^{k+1})$ by:

Step 1 subproblem:

Fix: $((X_r^k)_{r\epsilon R0}, Z^k) \rightarrow ((X_r)_{r\epsilon R0}, Z)$

Optimize: $((X_r)_{r\epsilon Rs}, Y_s)_{s\epsilon S} \rightarrow ((X_r^{k+1})_{r\epsilon Rs}, Y_s^{k+1})_{s\epsilon S}$

Step 2 subproblem:

Fix: $((X_r^{k+1})_{r\epsilon Rs}, Y_s^{k+1})_{s\epsilon S} \rightarrow ((X_r)_{r\epsilon Rs}, Y_s)_{s\epsilon S}$

Optimize: $((X_r)_{r\epsilon R0}, Z) \rightarrow ((X_r^{k+1})_{r\epsilon R0}, Z^{k+1})$

Multiplier update:

$P^{k+1} = P^k - \rho\,(Y^{k+1} - Z^{k+1})$

ADMM Subsystem Decomposition
Subproblem components

The step 1 subproblem is decomposable with respect to s in S. Component s is:

Minimize

$$\sum_{r \epsilon Rs,t} F_{rt}(X_{rt}) + \sum_t (P_{st}(Y_{st} - Z_{st}) + \rho/2 \, |Y_{st} - Z_{st}|^2)$$

With respect to

$$X_r, r \in R_s, Y_s$$

Subject to

$$X_r \in W_r \text{ for all } r \in R_s$$

$$Y_{st} = \sum_{r \epsilon Rs} X_{rt} \text{ for all } t \in T$$

The step 2 subproblem is decomposable with respect to t in T. Component t is:

Minimize

$$\sum_{r \epsilon R0} F_{rt}(X_{rt}) + \sum_c H_{ct}(\sum_s B_{cst} Z_{st} + \sum_{r \epsilon R0} A_{crt} X_{rt}) + \sum_s (P_{st}(Y_{st} - Z_{st}) + \rho/2 \, |Y_{st} - Z_{st}|^2)$$

With respect to

$$X_r, r \in R_0, Z_s, s \in S$$

Subject to

$$X_{rt} \in W_{rt} \text{ for all } r \in R_0$$

ADMM Subsystem Decomposition
Subsystem Construction

- For r in R, consider the point $A_r = (A_{crt}, c \in C, t \in T)$.
- Cluster the points A_r. Each cluster defines a set R_s of resources.
- Define B by
  - $B_{cst} = 1/|R_s| \sum_{r \in R_S} A_{crt}$
- Clustering techniques
  - Round A to a given precision, cluster identical columns
  - For each resource, try to find unmatched resources with nearby A to a given tolerance, subject to a limit on subsystem size.
  - Given a set of subsystems on generator resources, allocate remaining virtual and dispatchable demand resources with preference to increasing the smaller subsystems to make them less severely nonconvex.
- Subsystem formulation is not exactly equivalent to basic formulation. But we just need the sequence of integer points out of ADMM, and we then run SCED on each using the basic formulation.
- Reserve products can also be treated in this method.

ADMM Single Generator Subsystem Performance

- Some improvements on ADMM with single generator decomposition.
  - Skip randomly selected subproblems (50%) in each iteration. Counteracts tendency for two generators to act to fix a system-wide constraint when only one is needed. Shows improvement, but still not as good as MIP or RINS-E
  - Tried adding derived bounds on linking variables in step 2 subproblems. Generally not helpful.

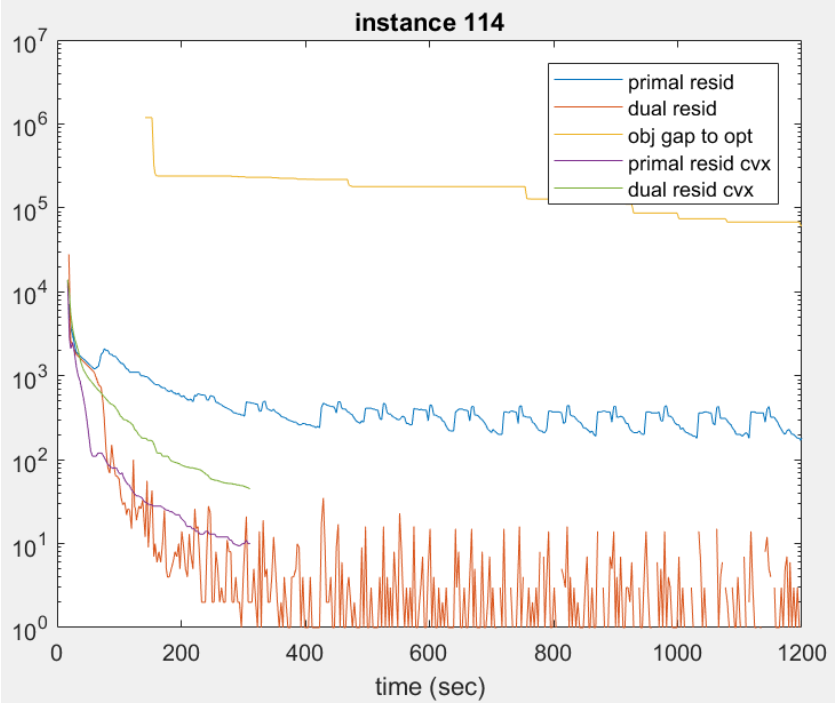
instance	admm_gap orig	admm_gap skip_subs	admm_gap q_bnd_2	rins_gap mp	mip_gap mp
102	0.237%	0.061%	0.065%	0.005%	0.004%
114	0.429%	0.117%	0.127%	0.018%	0.033%
115	0.323%	0.155%	0.119%	0.015%	0.001%
116	0.339%	0.302%	0.092%	0.029%	0.010%
117	0.541%	0.222%	0.517%	0.016%	0.081%
119	1.365%	0.365%	0.348%	0.007%	0.001%
121	7.435%	1.986%	2.587%	0.001%	0.001%
130	1.950%	0.482%	0.421%	0.026%	0.002%
215	6.492%	1.747%	3.237%	0.044%	1.041%
220	0.318%	0.096%	0.095%	0.013%	0.054%

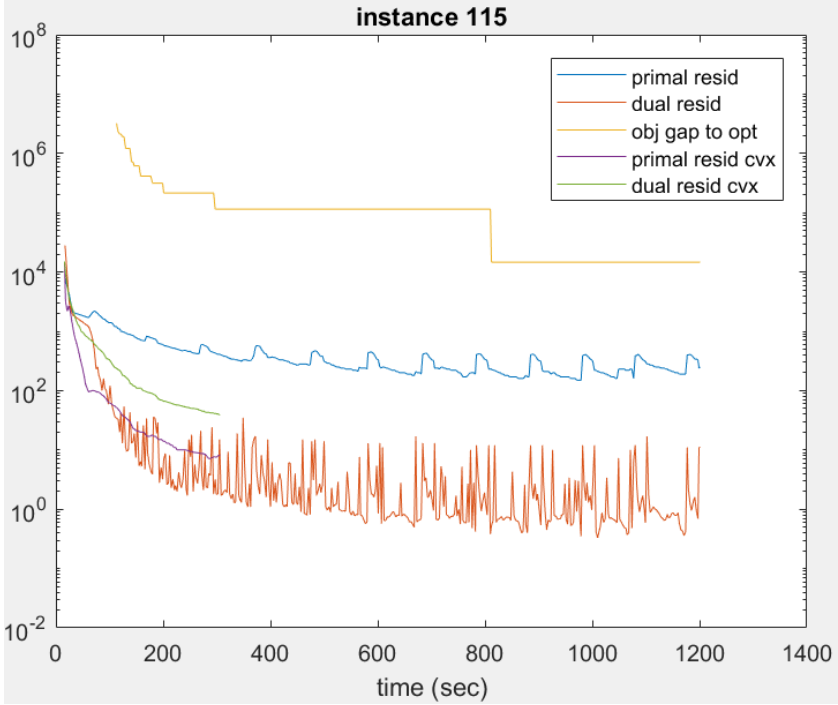
Results on multiple algorithms, best solution obtained in 1200 seconds, gap to true optimal value.

Note all the instances showed improvement in ADMM from skipping random subproblem solves, particularly those where the gap was very large.

ADMM convergence, MIP vs relaxed MIP instances 114 and 115

- On SCUC MIP problem, progress in primal and dual residuals and in feasible objective tends to stall.
- Converges with relaxed integer variables.
- Nonconvexity is the cause of stalling.



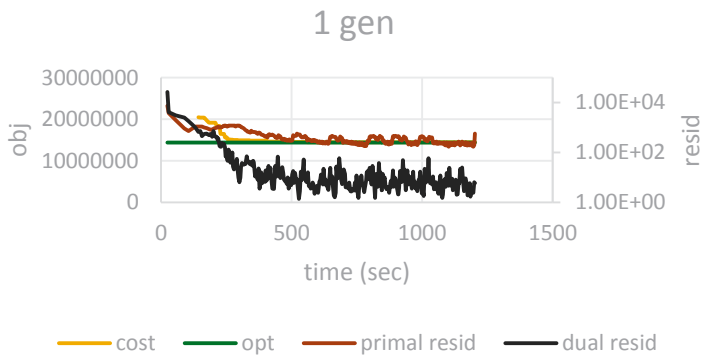


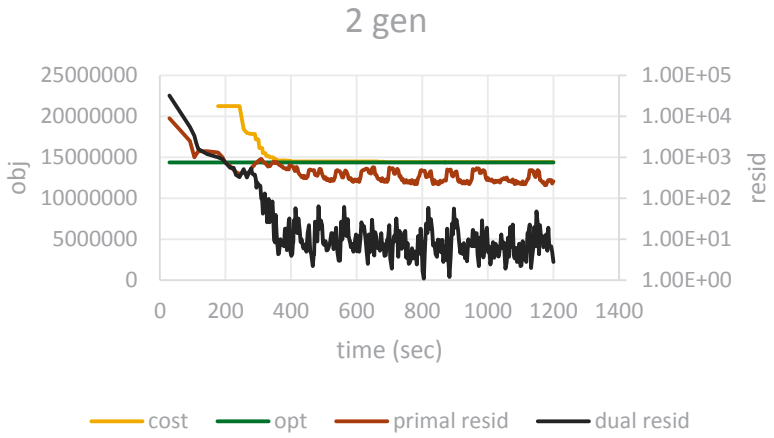
How nonconvexity causes stalling in ADMM with single generator subsystems

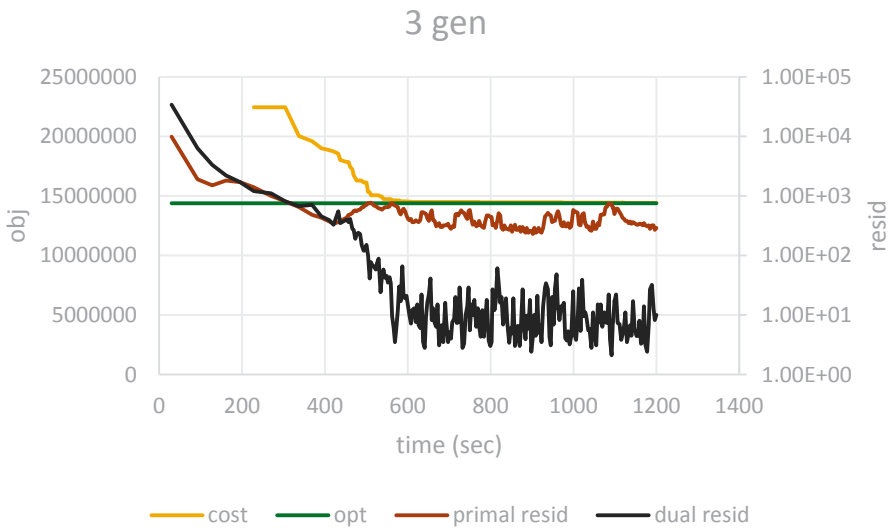
- 2 identical generators at a node, p_min=1, p_max=1, no_load_cost=1
- load=1
- Solution: p1=1, p2=0
- In ADMM, the step 2 subproblem will find p1=p2=0.5 (or some other common value), but step 1 can ONLY find p1=p2=0 or p1=p2=1 because they are in separate step 1 components
- Need step 1 subproblem components containing multiple generators, hence subsystem decomposition.

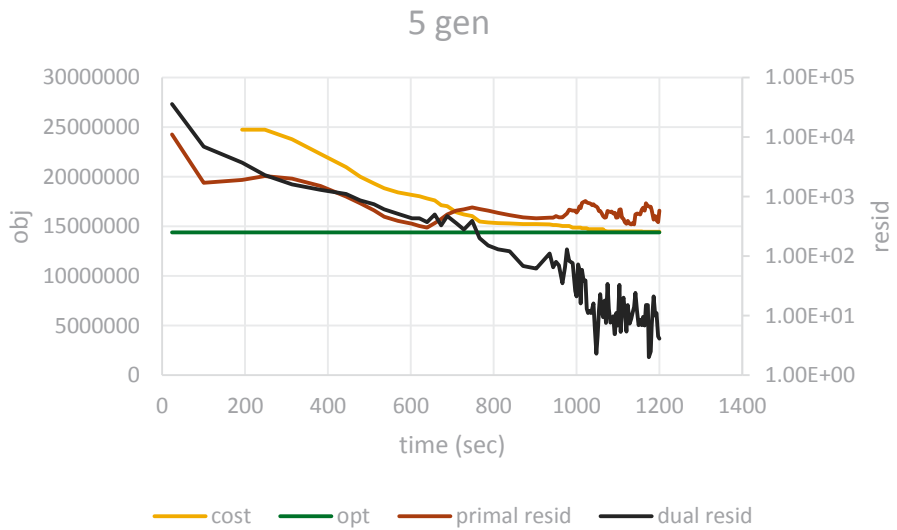
ADMM subsystem preliminary results
Instance 114, 1200 sec

- True optimal value: 14380892
- 1 gen subsys: 14443978
- 2 gen subsys: 14408697
- 3 gen subsys: 14411838
- 5 gen subsys: 14461945









Thank you

Pacific Northwest
NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965