Modeling and Computation of Security-constrained Economic Dispatch with Multi-stage Rescheduling

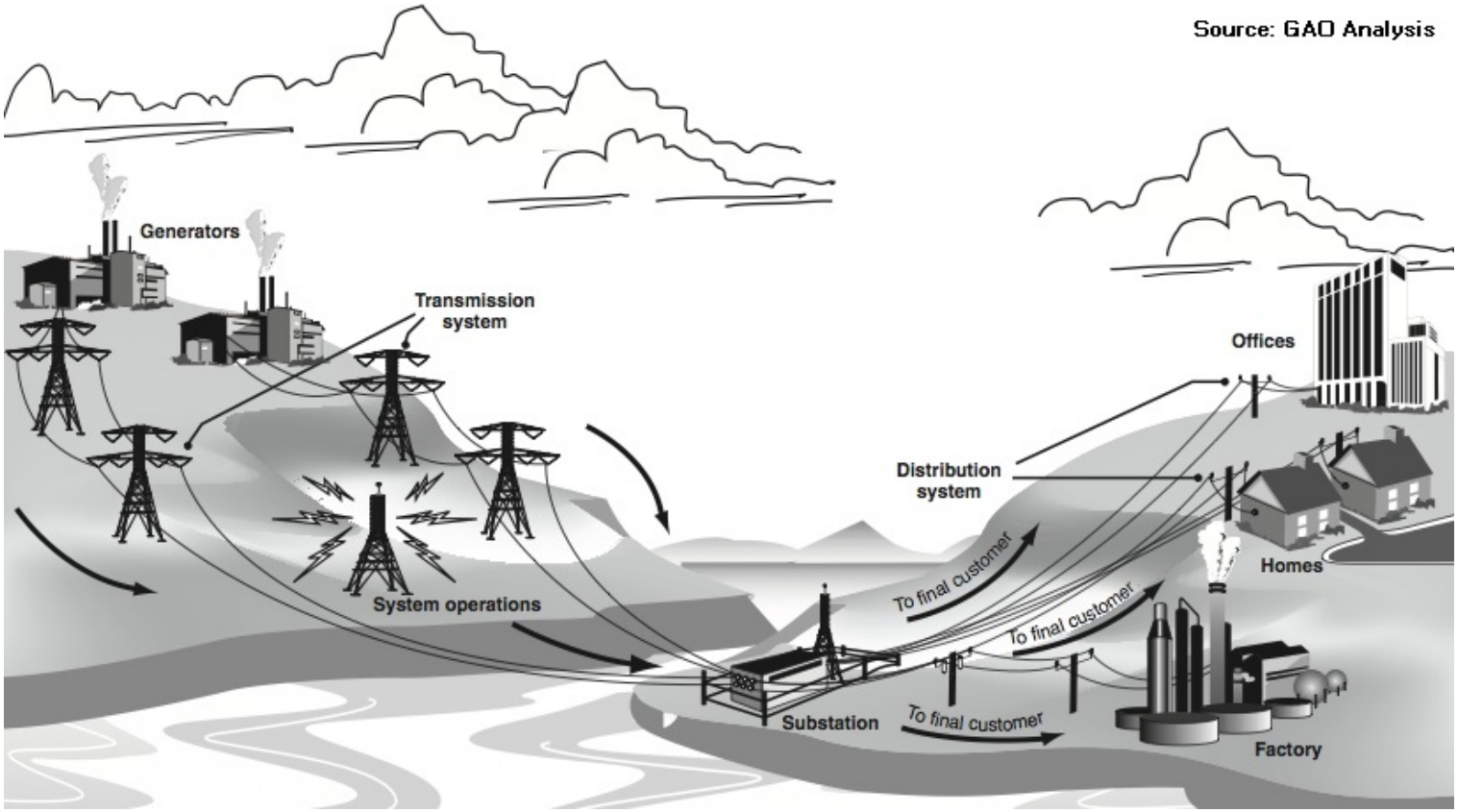
Michael Ferris

University of Wisconsin, Madison

Joint work with Yanchao Liu, University of Wisconsin-Madison, *yliu67@wisc.edu*
and Feng Zhao, ISO New England Inc.

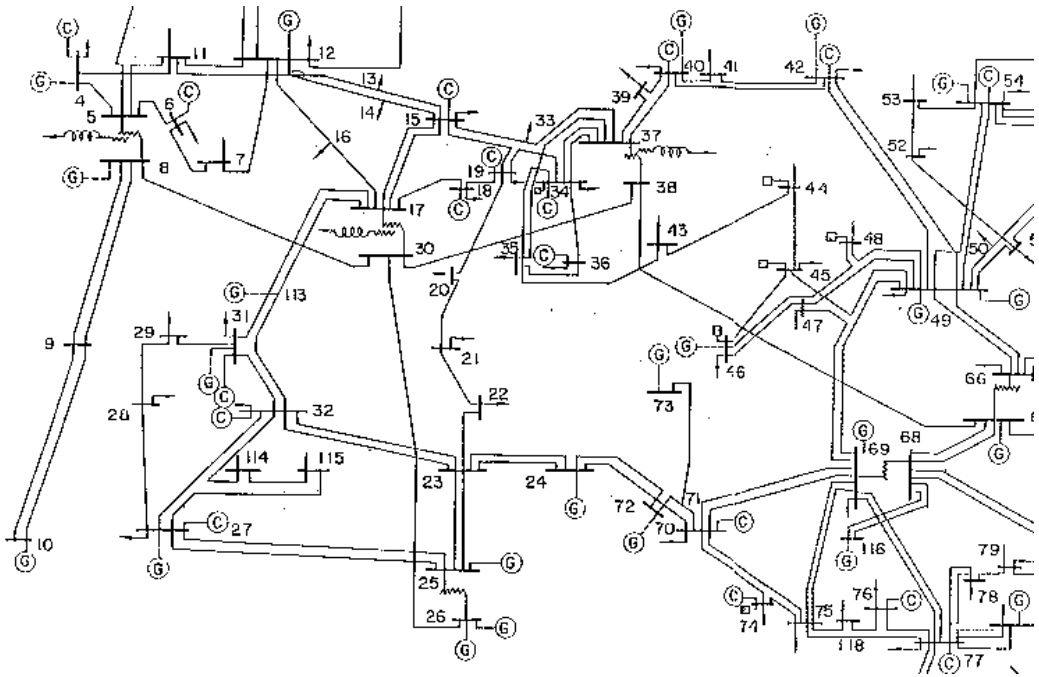
June 24, 2014

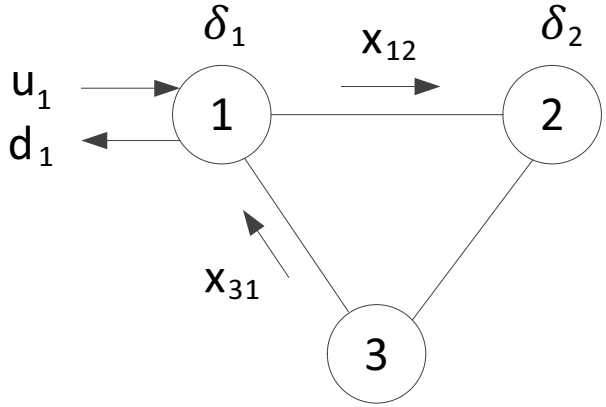
Power generation, transmission and distribution



- Determine generators' output to reliably meet the load
  - $\sum$ Gen MW $=$ $\sum$ Load MW, at all times.
  - Power flows cannot exceed lines' transfer capacity.

Economic dispatch (a linear program)





Nodal power balance:

$$u_1 - x_{12} + x_{31} = d_1$$

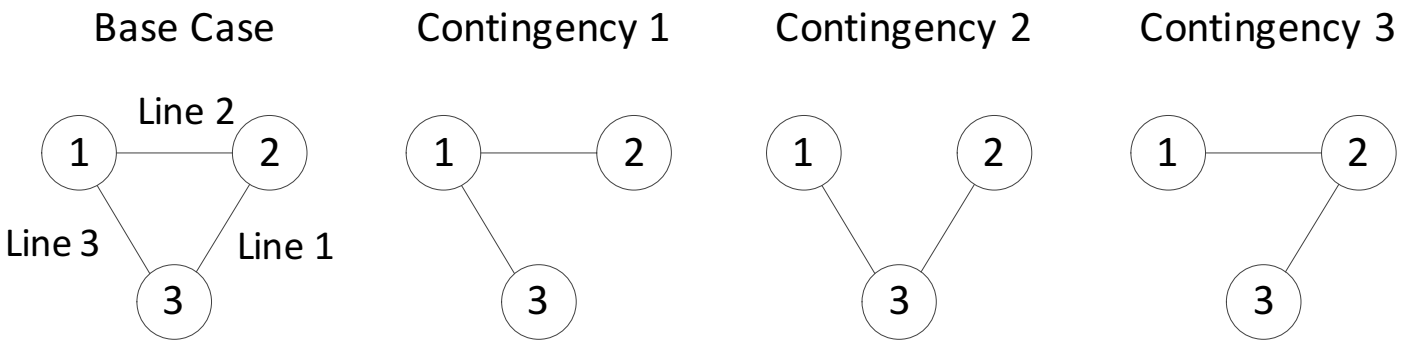
Flow definition:

$$x_{12} = B_{12}(\delta_2 - \delta_1)$$

Variables: Generators' output u; Power flows on lines x; Bus voltage angle δ
Objective: Minimize the total generation cost, $c^T u$
Constraints:

- Kirchhoff's laws: $g(x, u) = 0$, where g is a linear function, including:
  - Nodal balance equations, line flow equations.
- Variable bounds: $h(x, u) \leq 0$, including:
  - Line limit: $-\bar{x} \leq x \leq \bar{x}$; Generator capacity: $0 \leq u \leq \bar{u}$

Contingency: a single line failure



- A network with N lines can have up to N contingencies
- Each contingency case:
  - Corresponds to a different network topology
  - Requires a different set of equations g and h
  - E.g., equations g_k and h_k for the k-th contingency.

Control v.s. State variables

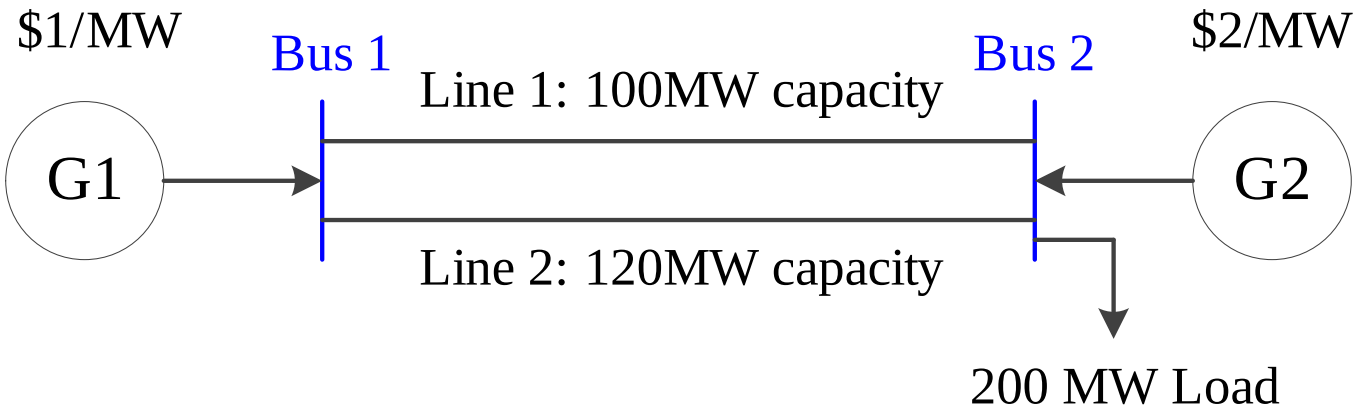
- Generator output u is a CONTROL variable:
  - System operator can directly set/adjust its level
  - No abrupt change, i.e., it takes time to ramp up/down a generator
- Line flow x is a STATE variable:
  - The level depends on u and the network topology
  - Automatically jumps to a new level when topology changes, e.g., when a line suddenly fails
- **Security requirement:** When a line fails, other lines should not overload.
- Change "base" state and control variables to achieve this.

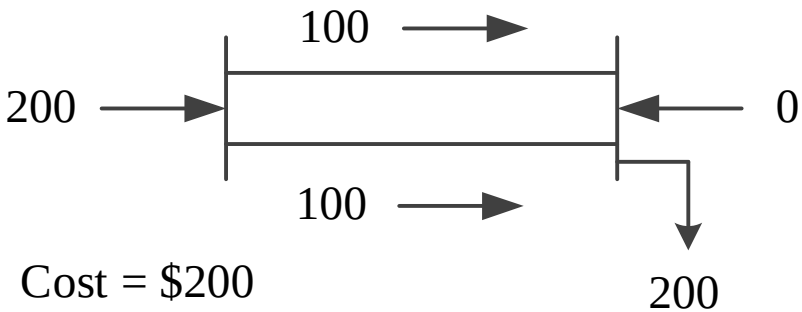
Security-constrained Economic Dispatch

- Base-case network topology g_0 and line flow x_0.
- If the k-th line fails, line flow jumps to x_k in new topology g_k.
- Ensure that x_k is within limit, for all k.
- SCED model:

$$
\begin{array}{lll}
\min\limits_{u,x_0,\ldots,x_k} & c^T u & \triangleright \text{ Total cost} \\
\text{s.t.} & 0 \le u \le \bar{u} & \triangleright \text{ GEN capacity const.} \\
& g_0(x_0, u) = 0 & \triangleright \text{Base-case network eqn.} \\
& -\bar{x} \le x_0 \le \bar{x} & \triangleright \text{Base-case flow limit} \\
& g_k(x_k, u) = 0, \quad k = 1, \ldots, K & \triangleright \text{Ctgcy network eqn.} \\
& -\bar{x} \le x_k \le \bar{x}, \quad k = 1, \ldots, K & \triangleright \text{Ctgcy flow limit}
\end{array}
$$

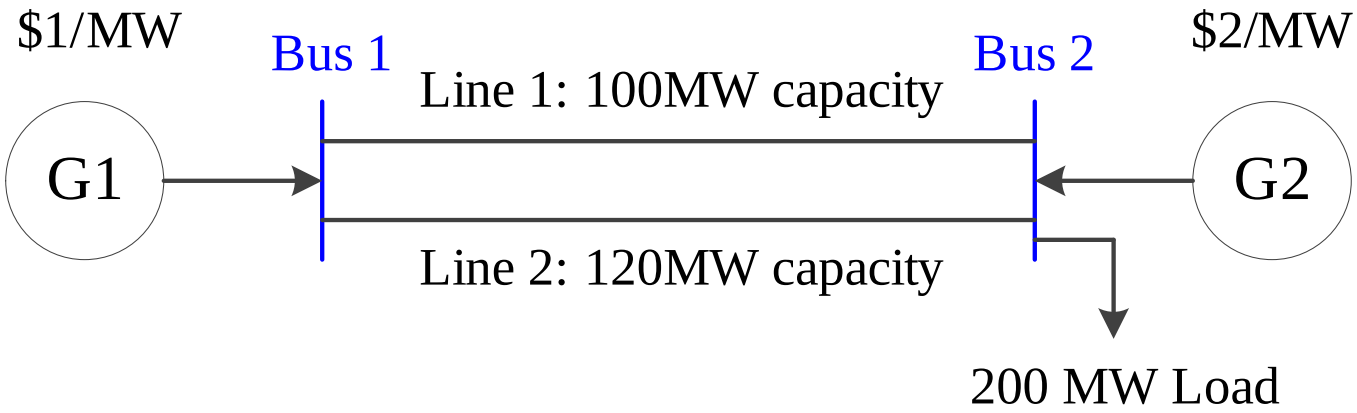
Security-constrained Economic Dispatch (SCED)

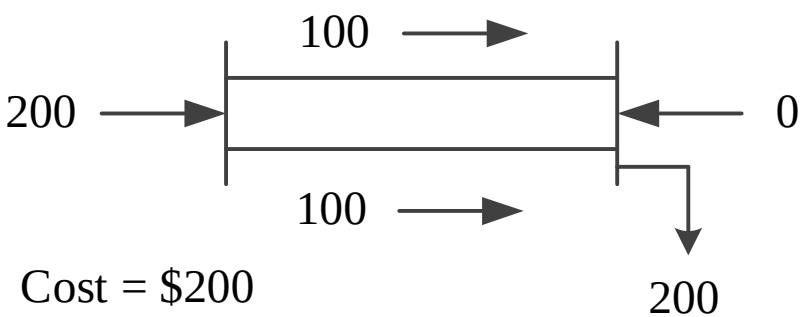




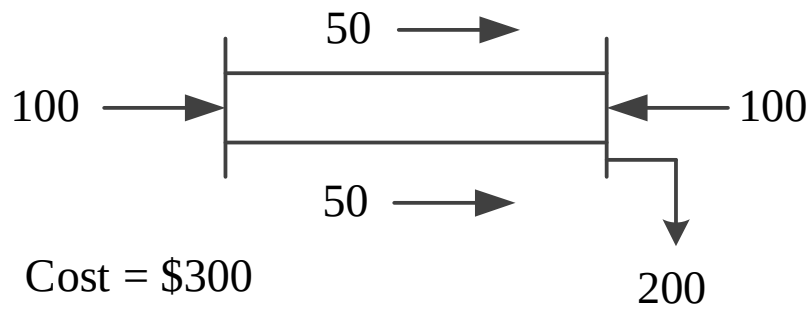
Economic dispatch

Security-constrained Economic Dispatch (SCED)



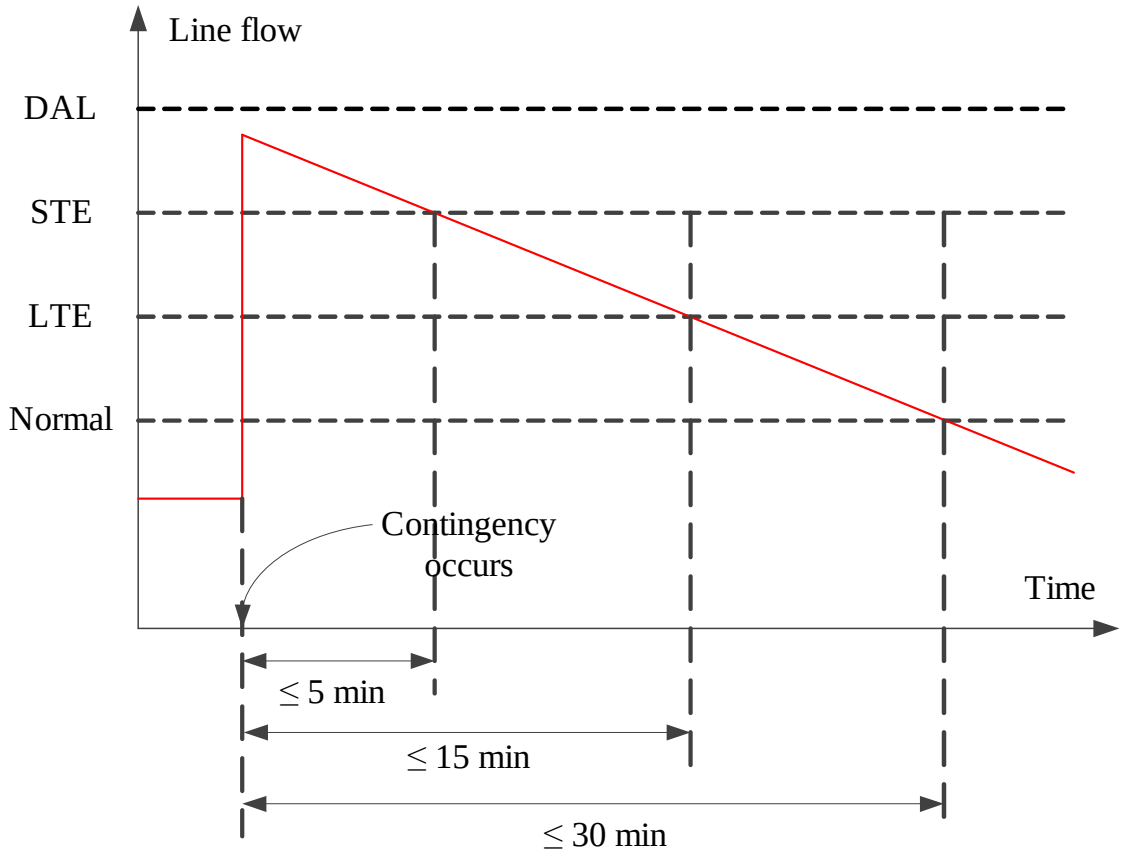


Economic dispatch



Security-constrained Economic Dispatch

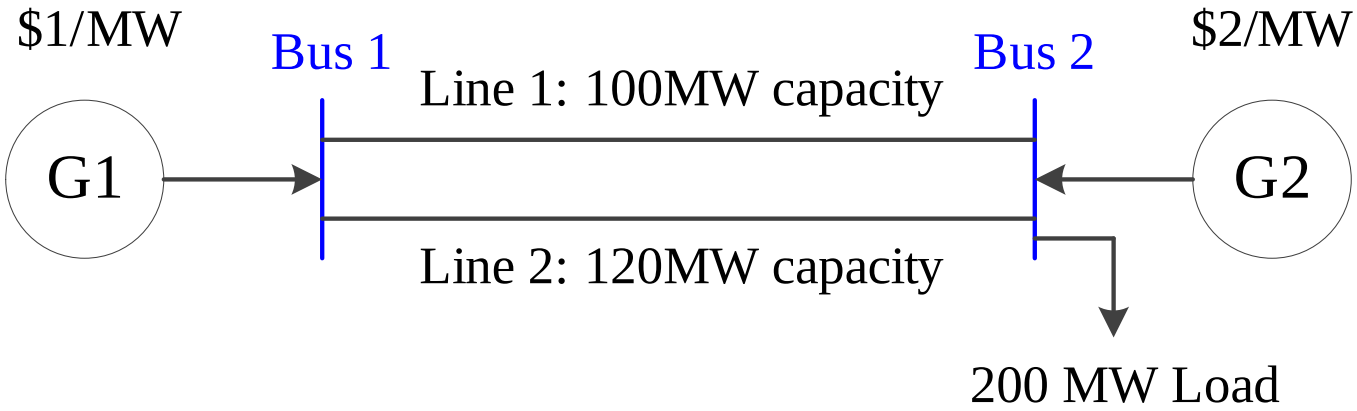
Reality offers a sweeter deal...



Operating procedure of ISO New England requires the post-contingency line loadings be:

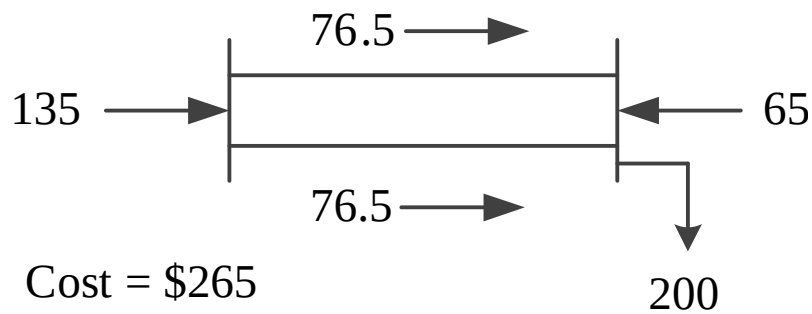
- $\leq$ STE (short time emergency) rating in 5 minutes;
- $\leq$ LTE (long time emergency) rating in 15 minutes;
- $\leq$ Normal rating in 30 minutes.

The cost can be lower...



If:

- The lines can withstand 40% overload for 5 minutes.
- G1 and G2 have a 5-min ramping radius of 40 MW and 35 MW, respectively.



SCED with Corrective rescheduling.
Lower Cost!

What we will contribute

Research issues:

- Corrective actions are not modeled in ISO's dispatch software.
- Because it was "insolvable" due to its large size ($\geq$ 10GB LP).
  - "We looked into SCED with corrective actions before, and were hindered by the computational challenge." – Feng Zhao, senior analyst at ISO-NE, via private correspondence.

Our contributions:

- We **model** the *multi-period* corrective rescheduling in SCED.
- **Enhance** the Benders' **algorithm** to solve the problem faster.
- **Achieve** about 50$\times$ **speedup** compared to traditional approaches.

Our model (K contingencies, T periods)

$$
\begin{aligned}
\min_{x_0,\ldots,x_k,u_0,\ldots,u_k} \quad & c^T u_0 \\
\text{s.t.} \quad & g_0(x_0, u_0) = 0 \\
& h_0(x_0, u_0) \leq 0 \\
& g_k(x_k^t, u_k^t) = 0 && k = 1,\ldots,K,\ t = 0,\ldots,T \\
& h_k(x_k^t, u_k^t) \leq 0 && k = 1,\ldots,K,\ t = 0,\ldots,T \\
& |u_k^t - u_k^{t-1}| \leq \Delta_t && k = 1,\ldots,K,\ t = 1,\ldots,T \\
& u_k^0 - u_0 = 0 && k = 1,\ldots,K
\end{aligned}
$$

- Subscript 0 indicates a quantity in the base-case network topology.
- This is a large-scale linear program.
- What special structure does it have?

Model structure

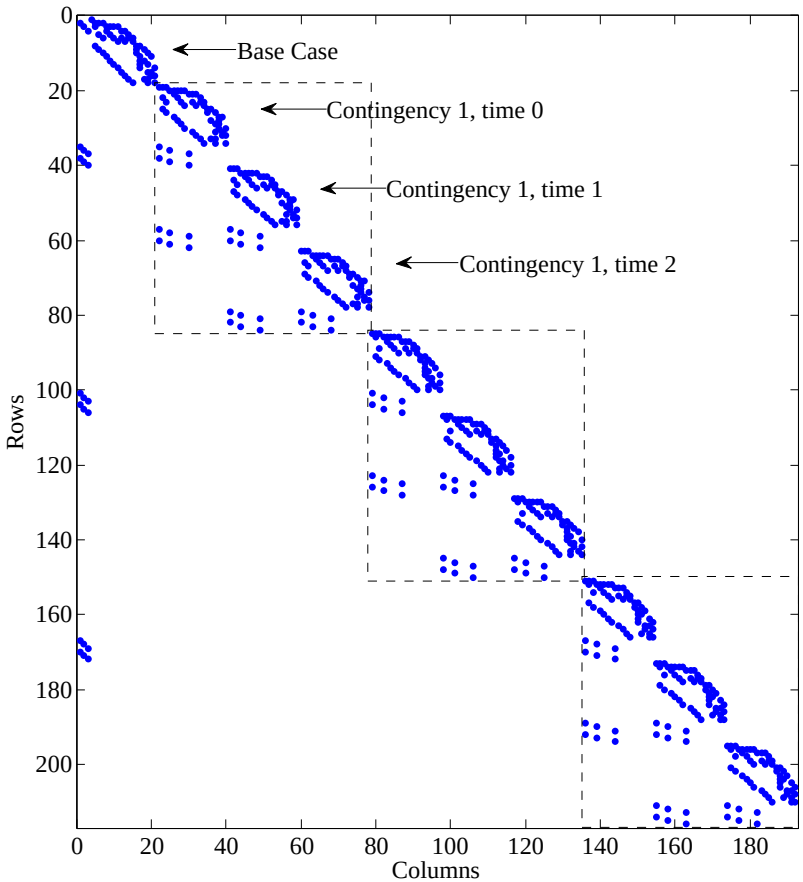


Figure : Sparsity structure of the Jacobian matrix of a 6-bus case, considering 3 contingencies and 3 post-contingency checkpoints.

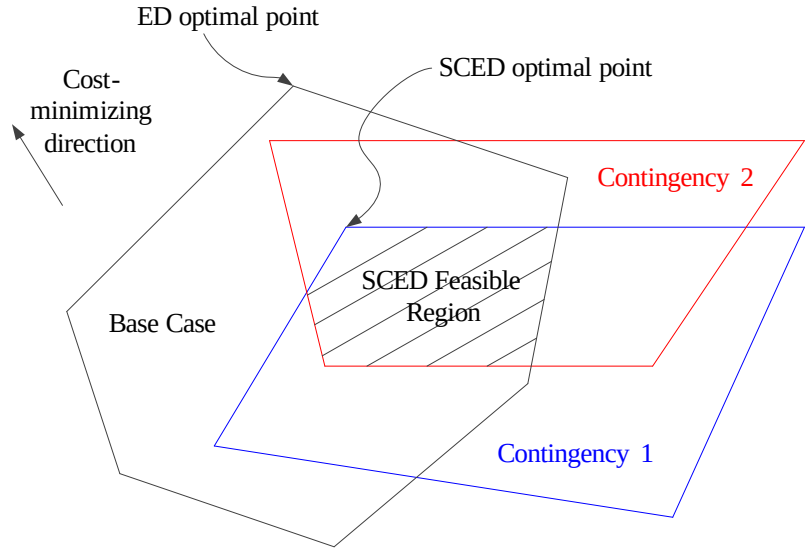


Figure : On the u_0 plane, the feasible region of a SCED is the intersection of K+1 polyhedra.

Basic idea of Benders' decomposition

- Approximate the feasible region using cuts
- 1 Master problem + N subproblems
- Iterate:
  1. (If not converged) Solve the Master problem
  2. Given Master solution, solve each subproblem and generate cuts
  3. Add the cuts to Master problem; goto step 1.
- Solution time $\approx$ (Time to solve each subproblem) $\times$ (# of subproblem per iteration) $\times$ (# of iterations needed for convergence).

Benders' decomposition

The model is equivalent to

$$
\begin{aligned}
\min_{x_0, u_0} \quad & c^T u_0 \\
\text{s.t.} \quad & g_0(x_0, u_0) = 0 \\
& h_0(x_0, u_0) \leq 0 \\
& w_k(u_0) \leq 0 \qquad\qquad k = 1, \ldots, c
\end{aligned} \tag{1}
$$

where $w_k(u_0)$ is given by the subproblem

$$
\begin{aligned}
w_k(u_0) = \quad \min_{x_k, u_k, s_k} \quad & ||s_k^t|| \\
\text{s.t.} \quad & g_k(x_k^t, u_k^t) = 0 && t = 0, \ldots, T \\
& h_k(x_k^t, u_k^t) - s_k^t \leq 0 && t = 0, \ldots, T \\
& |u_k^t - u_k^{t-1}| - s_k^t \leq \Delta_t && t = 1, \ldots, T \\
& u_k^0 - u_0 = 0 \\
& s_k^t \geq 0
\end{aligned} \tag{2}
$$

Benders' cut for $w_k(u_0) \leq 0$

Any point $\bar{u}_0$ would provide a subgradient inequality:

$$w_k(u_0) \geq \bar{w}_k + \bar{\lambda}_k(u_0 - \bar{u}_0)$$

where $\bar{w}_k = w_k(\bar{u}_0)$ and $\bar{\lambda}_k$ is the Lagrangian multiplier of the constraint (2) at the subproblem solution.

$$\bar{w}_k + \bar{\lambda}_k(u_0 - \bar{u}_0) \leq 0$$

is a valid inequality for the master problem and will cut off the point $\bar{u}_0$ if $\bar{w}_k$ is positive.

Benders' algorithm

Alternately solving the master problem and the subproblems, approaching a better and better satisfaction of (1) until $\max_k ||s_k|| \leq \epsilon$. We take $\epsilon = 10^{-6}$.

Current state of the art (unsatisfactory)

Table : CPLEX v.s. Vanilla Benders Algorithm

Case	Ctgcy	Big LP (time)		Vanilla Benders		
		Simplex	Barrier[1]	Iter	LPs	Time
118-bus	183	207.8	13.8	8	1464	123.5
2383-bus	20	175.0	205.5	52	1040	1281.2
2383-bus	50	1403.2	123.1	49	2450	2799.3
2383-bus	100	3621.8	240.6	32	3200	3688.6
2383-bus	400	-	2354.5	-	-	-

- Three time-periods: 5-min STE, 15-min LTE and 30-min Normal.
- Vanilla Benders' algorithm is inferior to the big LP formulation.
- Big LP cannot handle large instances.

[1] Barrier method without crossover. Crossover may take even more time.

How we enhanced the Benders' algorithm ...

1. Reduce the number of LPs
2. Solve subproblem LPs faster
3. Parallel computing
4. Add difficult contingencies to master model

Case	Ctgcy	Big LP (time)		Vanilla Benders			**Enhanced Benders**		
		Simplex	Barrier	Iter	LPs	Time	Iter	LPs	Time
118-bus	183	207.8	13.8	8	1464	123.5	12	755	13.5
2383-bus	20	175.0	205.5	52	1040	1281.2	11	60	41.5
2383-bus	50	1403	123.1	49	2450	2799.3	11	135	46.5
2383-bus	100	3621	240.6	32	3200	3688.6	12	245	79.4
2383-bus	400	-	2354.5	-	-	-	13	879	197.8
2383 wp	2349						21	9529	515.7
2736 sp	2749						4	5500	220.9
2737 sop	2753						1	2753	100.5
2746 wop	2794						1	2794	118.5
2746 wp	2719						14	5558	333.5

Enhancement #1: Reduce the number of LPs

- Most contingencies are actually feasible (i.e., subproblem has obj. value 0) given the base-case solution.
- Once a contingency becomes feasible, it is likely to remain feasible in subsequent iterations.
- **Improvements:**
  - Once a contingency is feasible, remove it from the *Active List*.
  - When the Active List is empty, add back all contingencies and re-scan.
  - Done if all are feasible; continue otherwise.
- This will reduce the number of subproblem LPs to solve.

Illustration

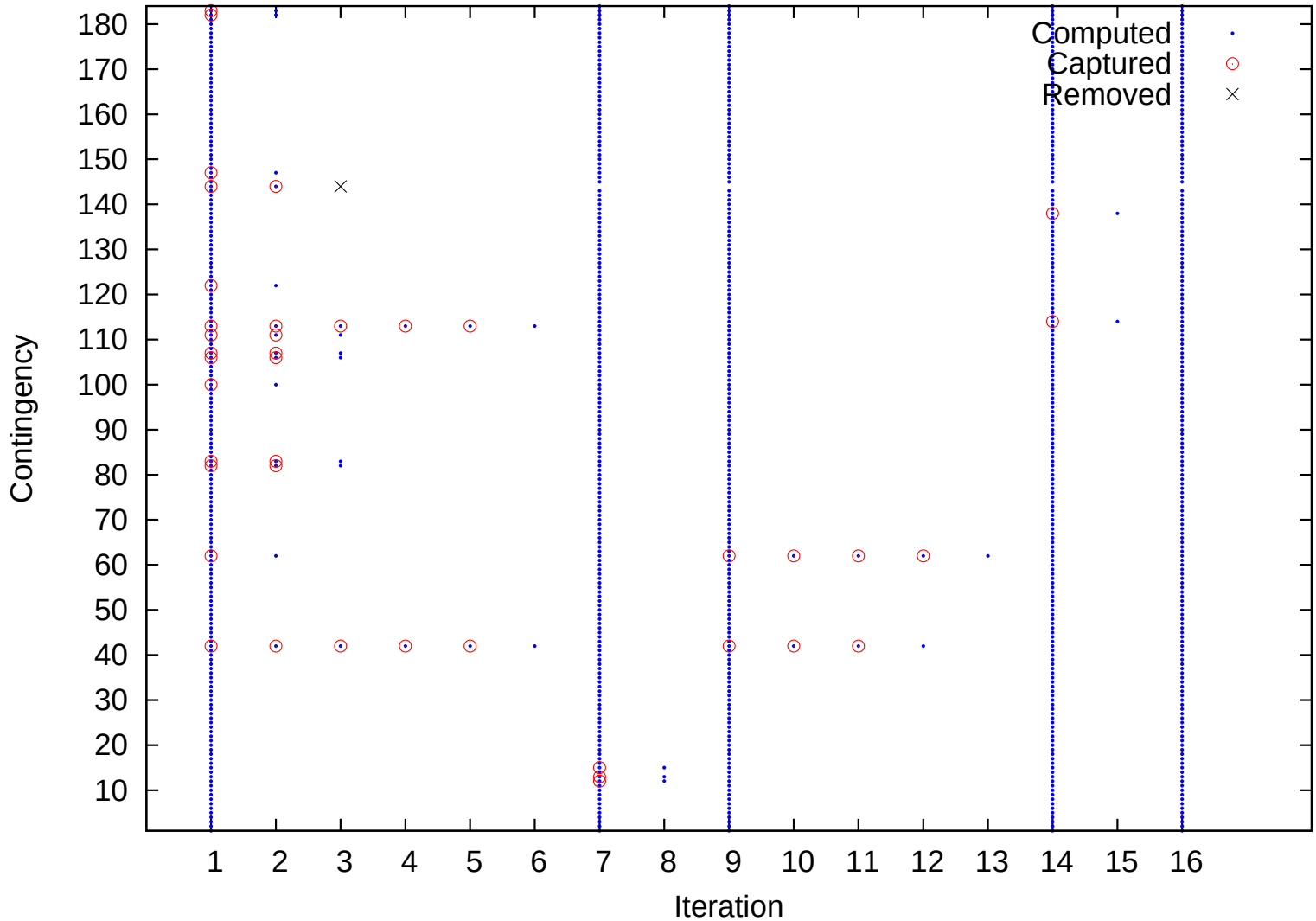


Figure : Benders' algorithm with reduced number of subproblem LPs, 118-bus case

Enhancement #2: Solve subproblem LPs faster

Table : Time (seconds) spent in sequentially solving 100 subproblems using different LP methods

Case	Subproblem Size			Default Simplex	Barrier	Barrier \Xover
	Row	Col	NZ			
118-bus	1070	2668	8545	14.9	14.4	13.4
2383-bus	16814	37129	115006	453.6	139.0	79.8

- Subproblem does not need a basic solution to generate a cut, any optimal solution (mid-face or vertex) can give a cut.
- So we use barrier method and disable the crossover (*barcrossalg=-1*) for speedup.

Effect of Enhancement #1 and #2

Case	Ctgcy	Big LP		Vanilla Benders			RedLP+Opt		
		Splx	Bar	Iter	LPs	Time	Iter	LPs	Time
118-bus	183	207.8	13.8	8	1464	123.5	10	764	72.6
2383 wp	20	175.0	205.5	52	1040	1281.2	46	115	99.8
2383 wp	50	1403	123.1	49	2450	2799.3	48	193	160.3
2383 wp	100	3621	240.6	32	3200	3688.6	33	289	226.0
2383 wp	400	-	2354.5	-	-	-	35	953	913.3

Enhancement #3: Parallel computing

- In each iteration, divide the Active List in N batches, process them in parallel.
- For each batch of LPs, use GUSS facility of GAMS (build model once, update data and solve for different LPs).

Case	Ctgcy	Vanilla Benders			RedLP+Opt			Paraguss (8)		
		Iter	LPs	Time	Iter	LPs	Time	Iter	LPs	Time
118-bus	183	8	1464	123.5	10	764	72.6	14	776	15.1
2383 wp	20	52	1040	1281.2	46	115	99.8	48	117	95.4
2383 wp	50	49	2450	2799.3	48	193	160.3	48	193	101.7
2383 wp	100	32	3200	3688.6	33	289	226.0	32	288	96.3
2383 wp	400	-	-	-	35	953	913.3	38	956	218.0

Enhancement #4: Add difficult contingencies to master model

- Observation: A few contingencies remain in the Active List for many iterations.
  - A small-sized Active List means low efficiency of parallelism (due to overhead).
  - Too many iterations means much time needed for convergence.
- Improvement:
  - When the size of the Active List drops to a threshold level L^{fc}, add the remaining active contingencies to the master problem.
  - The added contingencies remain in the master problem in all future iterations and their subproblems is removed from the List for good.
- The level of L^{fc} reflects tradeoff between "a harder master problem" and "more iterations".
- Empirically, $L^{\text{fc}} = 3$ or 5 is good for large instances.

Computational Results

Case	Ctgcy	RedLP+Opt			Paraguss (8)			Fatmaster (5)		
		Iter	LPs	Time	Iter	LPs	Time	Iter	LPs	Time
118-bus	183	10	764	72.6	14	776	15.1	12	755	13.5
2383 wp	20	46	115	99.8	48	117	95.4	11	60	41.5
2383 wp	50	48	193	160.3	48	193	101.7	11	135	46.5
2383 wp	100	33	289	226.0	32	288	96.3	12	245	79.4
2383 wp	400	35	953	913.3	38	956	218.0	13	879	197.8

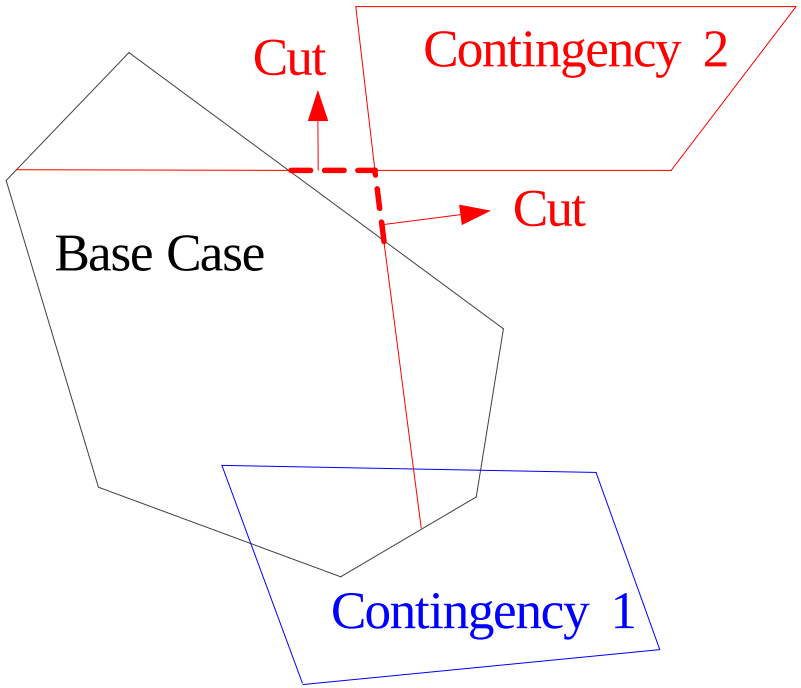
Case	Ctgcy	RedLP+Opt			Paraguss (40)			Fatmaster (5)		
		Iter	LPs	Time	Iter	LPs	Time	Iter	LPs	Time
2383wp	2349	106	12123	12165	104	9788	769.5	21	9529	515.7
2736sp	2749	45	5543	5836	44	5542	366.2	4	5500	220.9
2737sop	2753	1	2753	2801	1	2753	100.1	1	2753	100.5
2746wop	2794	1	2794	3046	1	2794	118.3	1	2794	118.5
2746wp	2719	262	8646	9738	278	8622	1427.7	14	5558	333.5

- Big LP for 2383-bus 2349-contingency case generates a 18GB LP. CPLEX could not solve it in 3 hours.
- Computer used for the lower table: Dell R710 (opt-a006) 2 3.46G Chips 12 Cores, 288G Memory.

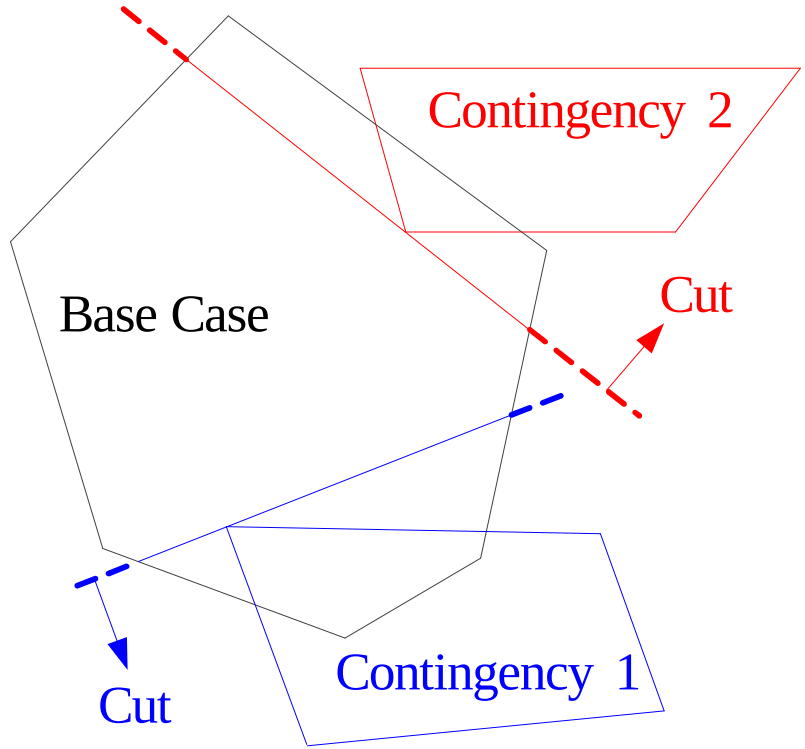
Dealing with Infeasibility

- Up to now, we implicitly assumed that the problem is feasible.
- What if it is not feasible?
  - Big LP formulation will report "infeasible".
  - Traditional Benders' algorithm will encounter infeasible master or subproblem and halt.
- Need to know the cause and the nearest feasible solution.
- **Existing method:** Pre-screen each contingency beforehand
  - Takes too much time to pre-screen
  - Can not catch all infeasible cases; not effective.
- **Our method:** Diagnose and remove in the algorithm. Faster and effective.

Causes of infeasibility



(a) Contingency 2 is intrinsically infeasible. Either the corresponding subproblem is infeasible or its Benders' cuts will render the master problem infeasible.



(b) Each individual contingency is feasible, but they are not simultaneously feasible. Their Benders' cuts will render the master problem infeasible.

Figure : Two cases of infeasibility.

Pre-screening is not a good idea

Pre-screening the contingencies 1 by 1

Solve an LP consisting of the base-case and a single contingency. If the LP is infeasible, it means the given contingency is intrinsically infeasible. Remove it from the Contingency List.

- Takes much time to pre-screen a large list of contingencies.
- Cannot catch individually feasible but simultaneously infeasible contingencies (pairs or triples, etc.).

Table : Time (in second) spent to pre-screen for different cases. The LPs are solved sequentially.

Case	# Lines	# Feasible	# Removed	Time
2383 wp	2896	2353	543	49670.8
2736 sp	3269	2749	520	76068.8
2737 sop	3269	2753	516	13069.2
2746 wop	3307	2794	513	20160.2
2746 wp	3279	2719	560	43618.7

Identifying infeasible contingencies in Benders' algorithm

- If a subproblem is infeasible (in the first iteration), the corresponding contingency is intrinsically infeasible. Remove (tabu) it.
  - Typically line failure results in an islanded load node or sub-network.
- Master problem infeasible: solve a modified master model to find the "minimal" set of problematic contingencies using sparse optimization.

$$\begin{aligned} \min_{x_0, u_0} \quad & f_0(x_0, u_0) + \sum_{(k,i)\in \text{CUT}} M v_k^i \\ \text{s.t.} \quad & g_0(x_0, u_0) = 0, h_0(x_0, u_0) \le 0 \\ & \bar{w}_k^i + \bar{\lambda}_k^i (u_0 - \bar{u}_0^i) - v_k^i \le 0, v_k^i \ge 0 \qquad \forall (k,i) \in \text{CUT} \end{aligned}$$

  - Solution of this model indicates the violated cut.
  - Tabu the contingency that has contributed one or more violated cuts.
- Start a pre-screening daemon in parallel when the Active List size is smaller than L^{fc}.
  - Tabu infeasible ones, and add feasible ones to the master problem.

Computational Results

Table : Solution for big cases on opt-a006, 80 threads, $L^{fc} = 5$

Case	Ctgcy	Iter	LPs	Time	Added	Tabu
2383 wp	2896	15	7694	522.1	6	547
2736 sp	3269	4	6020	252.9	1	520
2737 sop	3269	4	6023	242.2	0	516
2746 wop	3307	4	6102	280.2	0	513
2746 wp	3279	8	6053	334.3	4	560
2383 wp	2353	16	7156	460.6	6	4
2736 sp	2749	4	5498	245.9	1	0
2737 sop	2753	1	2753	110.8	0	0
2746 wop	2794	1	2794	131.7	0	0
2746 wp	2719	14	5558	354.4	4	0

- Upper: all lines are in the Contingency List (N-1 security).
- Lower: all pre-screened lines are in the Contingency List.

Summary

1. SCED is a million-dollar problem for system operators.
2. SCED with corrective actions can save money, but is hard to solve.
   - Too big for CPLEX
   - Original Benders' decomposition algorithm is slow.
3. Our algorithmic enhancements yield significant speedup.
4. Potential for practical deployment.

Extension

1. Decomposition approach is useful in many applications.
2. Currently in collaboration with ISO-NE to deploy our algorithm.