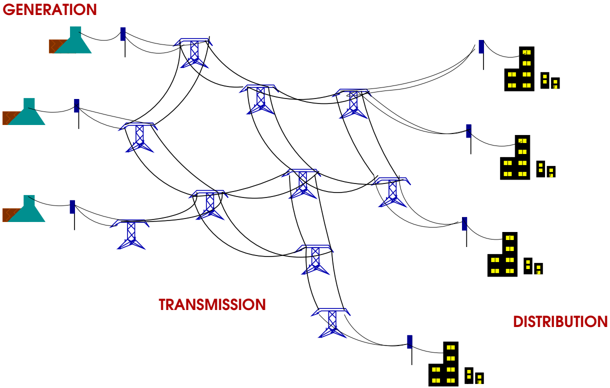
Power grid vulnerability analysis

Daniel Bienstock

Columbia University

Dimacs 2010

Background: a power grid is three systems



Challenges to analysis

- Power grids follow the laws of physics, characterized by nonlinear, nonconvex equations that make fast computation difficult.

Challenges to analysis

- Power grids follow the laws of physics, characterized by nonlinear, nonconvex equations that make fast computation difficult.
- Furthermore, direct control is difficult: we cannot dictate how power will flow.

Challenges to analysis

- Power grids follow the laws of physics, characterized by nonlinear, nonconvex equations that make fast computation difficult.
- Furthermore, direct control is difficult: we cannot dictate how power will flow.
- Power grids are subject to “noise” which is difficult to model accurately.

Challenges to analysis

- Power grids follow the laws of physics, characterized by nonlinear, nonconvex equations that make fast computation difficult.
- Furthermore, direct control is difficult: we cannot dictate how power will flow.
- Power grids are subject to “noise” which is difficult to model accurately.
- Power grids can exhibit non-monotone behavior as a result of control or adversarial actions.

Challenges to analysis

- Power grids follow the laws of physics, characterized by nonlinear, nonconvex equations that make fast computation difficult.
- Furthermore, direct control is difficult: we cannot dictate how power will flow.
- Power grids are subject to “noise” which is difficult to model accurately.
- Power grids can exhibit non-monotone behavior as a result of control or adversarial actions.
- Power grids can cascade.

AC power flows – polar coordinates

→ Voltage at a node ("bus") k is of the form $U_k\, e^{j\theta_k}$, where $j = \sqrt{-1}$

→ Power flowing on edge ("line") $\{k, m\}$ equals $p_{km} + jq_{km}$, where

$$p_{km} = U_k^2\, g_{km} - U_k\, U_m\, g_{km} \cos\theta_{km} - U_k\, U_m\, b_{km} \sin\theta_{km}$$

$$q_{km} = -U_k^2\, (b_{km} + b_{km}^{sh}) + U_k\, U_m\, b_{km} \cos\theta_{km} - U_k\, U_m\, g_{km} \sin\theta_{km}$$

Here, $\theta_{km} \doteq \theta_k - \theta_m$

g_{km}, b_{km}, b_{km}^{sh} are known *parameters* (series conductance, series reactance, shunt susceptance)

Voltage at $k = U_k e^{j\theta_k}$; power on line $\{k, m\} = p_{km} + jq_{km}$, where

$$p_{km} = U_k^2 \, g_{km} - U_k U_m \, g_{km} \cos\theta_{km} - U_k U_m \, b_{km} \sin\theta_{km}$$

$$q_{km} = -U_k^2 \, (b_{km} + b_{km}^{sh}) + U_k U_m \, b_{km} \cos\theta_{km} - U_k U_m \, g_{km} \sin\theta_{km}$$

$(\theta_{km} \doteq \theta_k - \theta_m)$

$P_k = \Sigma_{\{k,m\}} p_{km}$ (active power), $\quad Q_k = \Sigma_{\{k,m\}} q_{km}$ (reactive power)

Voltage at k = $U_k\, e^{j\theta_k}$; power on line $\{k, m\}$ = $p_{km} + jq_{km}$, where

$$p_{km} = U_k^2\ g_{km} - U_k\, U_m\ g_{km}\ \cos\theta_{km} - U_k\, U_m\ b_{km}\ \sin\theta_{km}$$

$$q_{km} = -U_k^2\ (b_{km} + b_{km}^{sh}) + U_k\, U_m\ b_{km}\ \cos\theta_{km} - U_k\, U_m\ g_{km}\ \sin\theta_{km}$$

$(\theta_{km} \doteq \theta_k - \theta_m)$

$P_k = \Sigma_{\{k,m\}} p_{km}$ (active power), $Q_k = \Sigma_{\{k,m\}} q_{km}$ (reactive power)

Power flow problem: Choose the vectors p, q, θ, P, Q so as to satisfy all equations above, and

Voltage at k = $U_k\, e^{j\theta_k}$; power on line $\{k, m\}$ = $p_{km} + jq_{km}$, where

$$p_{km} = U_k^2\, g_{km} - U_k\, U_m\, g_{km}\, \cos\theta_{km} - U_k\, U_m\, b_{km}\, \sin\theta_{km}$$

$$q_{km} = -U_k^2\, (b_{km} + b_{km}^{sh}) + U_k\, U_m\, b_{km}\, \cos\theta_{km} - U_k\, U_m\, g_{km}\, \sin\theta_{km}$$

$(\theta_{km} \doteq \theta_k - \theta_m)$

$P_k = \Sigma_{\{k,m\}} p_{km}$ (active power), $Q_k = \Sigma_{\{k,m\}} q_{km}$ (reactive power)

Power flow problem: Choose the vectors p, q, θ, P, Q so as to satisfy all equations above, and

meet demand requirements and generator constraints

Voltage at $k = U_k e^{j\theta_k}$; power on line $\{k, m\} = p_{km} + jq_{km}$, where

$$p_{km} = U_k^2 g_{km} - U_k U_m g_{km} \cos\theta_{km} - U_k U_m b_{km} \sin\theta_{km}$$

$$q_{km} = -U_k^2 (b_{km} + b_{km}^{sh}) + U_k U_m b_{km} \cos\theta_{km} - U_k U_m g_{km} \sin\theta_{km}$$

$(\theta_{km} \doteq \theta_k - \theta_m)$

$P_k = \Sigma_{\{k,m\}} p_{km}$ (active power), $Q_k = \Sigma_{\{k,m\}} q_{km}$ (reactive power)

Power flow problem: Choose the vectors p, q, θ, P, Q so as to satisfy all equations above, and

meet demand requirements and generator constraints

and, ideally, meet thermal constraints (flow limits) on the power lines

Research challenges

$\longrightarrow$ Do we have **fast** and **reliable** algorithm for the power flow problem?

- Should not require human input in order to terminate.
- When no "acceptable" solution exists, should produce a certificate that this is the case.

Research challenges

$\longrightarrow$ Do we have **fast** and **reliable** algorithm for the power flow problem?

- Should not require human input in order to terminate.
- When no "acceptable" solution exists, should produce a certificate that this is the case.

What about the cases where multiple solutions exist?

Research challenges

$\longrightarrow$ Do we have **fast** and **reliable** algorithm for the power flow problem?

- Should not require human input in order to terminate.
- When no "acceptable" solution exists, should produce a certificate that this is the case.

What about the cases where multiple solutions exist?

- After a contingency has take place, or a control has been applied: which solution should be instantiated?

Research challenges

→ Do we have **fast** and **reliable** algorithm for the power flow problem?

- Should not require human input in order to terminate.
- When no “acceptable” solution exists, should produce a certificate that this is the case.

What about the cases where multiple solutions exist?

- After a contingency has take place, or a control has been applied: which solution should be instantiated?
- What if all solutions are “bad”?

Solution methodologies

- Newton-Raphson (iterative) algorithms to solve system of equations

Solution methodologies

- Newton-Raphson (iterative) algorithms to solve system of equations

  The claim: this "always" works fast.

Solution methodologies

- Newton-Raphson (iterative) algorithms to solve system of equations

  The claim: this "always" works fast.   At least in the case of a "normal" system.

Solution methodologies

- Newton-Raphson (iterative) algorithms to solve system of equations

  The claim: this "always" works fast. At least in the case of a "normal" system.

- **New result:** Low et al (2010). Some (many?) optimal power flow problems can be solved using semidefinite programming.

Linearized ("DC") model: $sin(\theta_i - \theta_j) \approx (\theta_i - \theta_j)$ for $\theta_i \approx \theta_j$

A **power flow** is a solution f, θ to:

- $\sum_{ij} f_{ij} - \sum_{ij} f_{ji} = b_i$, for all i, where

  $b_i > 0$ for each generator i,

  $b_i < 0$ for demand node i,

Linearized ("DC") model: $sin(\theta_i - \theta_j) \approx (\theta_i - \theta_j)$ for $\theta_i \approx \theta_j$

A **power flow** is a solution f, θ to:

- $\sum_{ij} f_{ij} - \sum_{ij} f_{ji} = b_i$, for all i, where

  $b_i > 0$ for each generator i,

  $b_i < 0$ for demand node i,

- $x_{ij} f_{ij} - \theta_i + \theta_j = 0$ for all (i, j). (x_{ij} = "reactance")

Linearized ("DC") model: $sin(\theta_i - \theta_j) \approx (\theta_i - \theta_j)$ for $\theta_i \approx \theta_j$

A **power flow** is a solution f, θ to:

- $\sum_{ij} f_{ij} - \sum_{ij} f_{ji} = b_i$, for all i, where

  $b_i > 0$ for each generator i,

  $b_i < 0$ for demand node i,

- $x_{ij} f_{ij} - \theta_i + \theta_j = 0$ for all (i, j). (x_{ij} = "reactance")

Lemma: Given a choice for b with $\sum_i b_i = 0$ (a requirement),

Linearized ("DC") model: $sin(\theta_i - \theta_j) \approx (\theta_i - \theta_j)$ for $\theta_i \approx \theta_j$

A **power flow** is a solution f, θ to:

- $\sum_{ij} f_{ij} - \sum_{ij} f_{ji} = b_i$, for all i, where

  $b_i > 0$ for each generator i,

  $b_i < 0$ for demand node i,

- $x_{ij} f_{ij} - \theta_i + \theta_j = 0$ for all (i,j). (x_{ij} = "reactance")

Lemma: Given a choice for b with $\sum_i b_i = 0$ (a requirement), the system has a **unique** (in f) solution.

A quote from:

Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations, U.S.-Canada Power System Outage Task Force, April 5, 2004. (https://reports.energy.gov)

A quote from:

Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations, U.S.-Canada Power System Outage Task Force, April 5, 2004. (https://reports.energy.gov)

Cause 1 was “inadequate system understanding”

A quote from:

Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations, U.S.-Canada Power System Outage Task Force, April 5, 2004. (https://reports.energy.gov)

Cause 1 was "inadequate system understanding" – stated 20 times

A quote from:

Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations, U.S.-Canada Power System Outage Task Force, April 5, 2004. (https://reports.energy.gov)

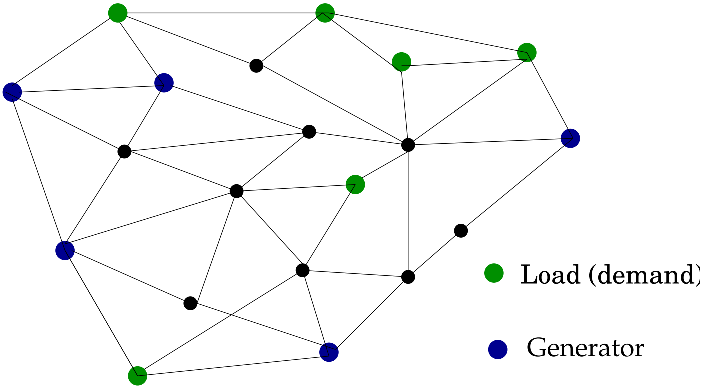
Cause 1 was "inadequate system understanding" – stated 20 times

Cause 2 was "inadequate situational awareness" – stated 14 times

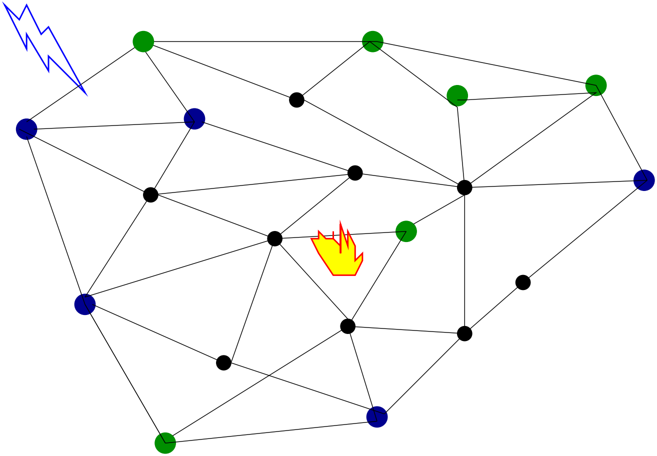
Cause 3 was "inadequate tree trimming" – stated 4 times

Cause 4 was "inadequate RC diagnostic support" – stated 5 times

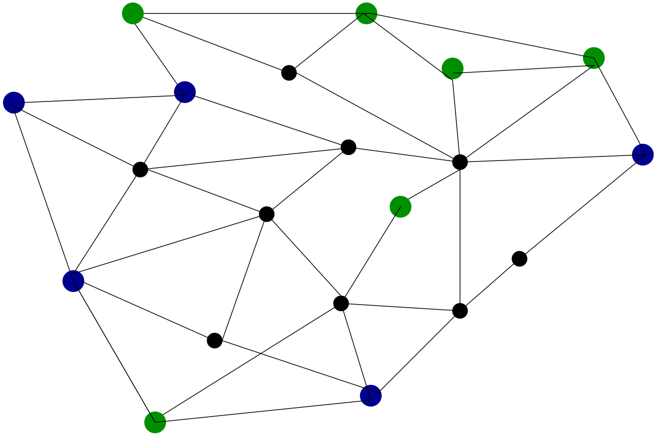
Cascades



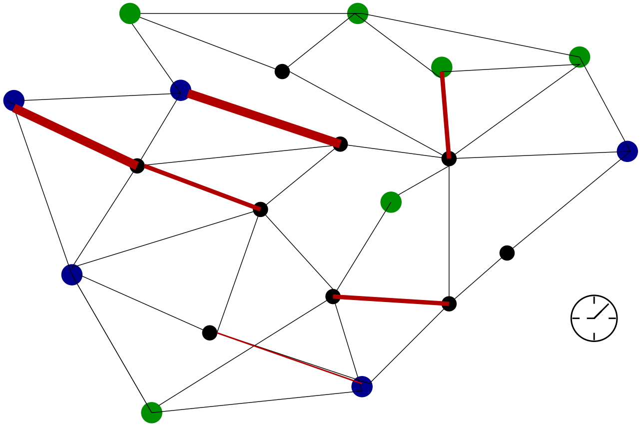
Cascades



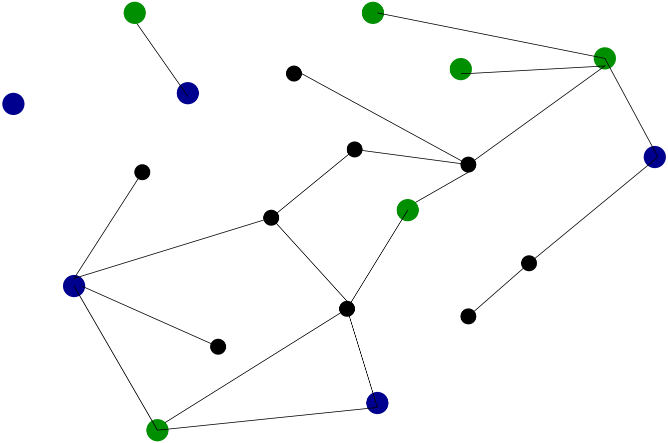
Cascades



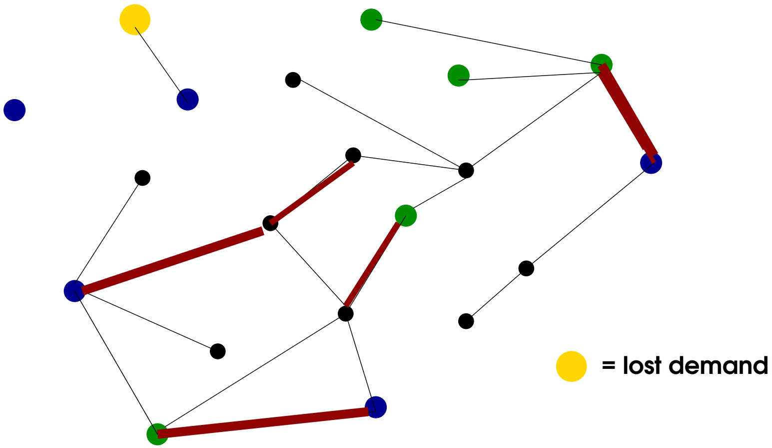
Cascades



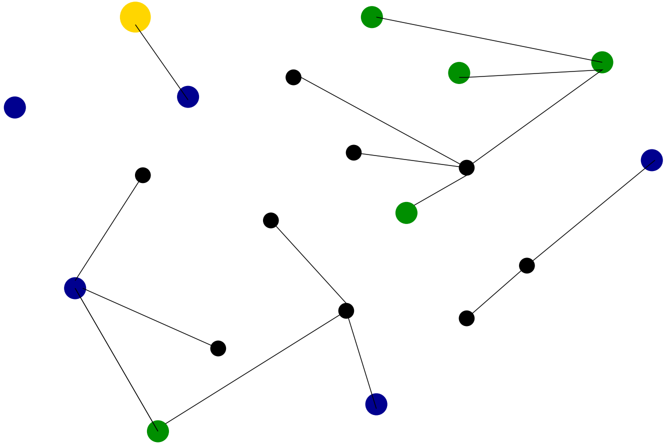
Cascades



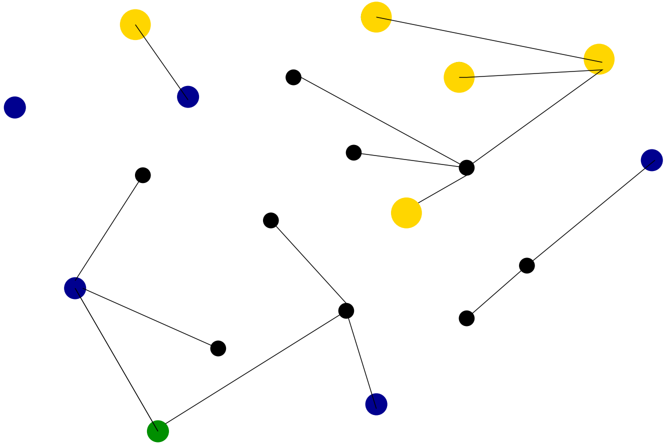
Cascades



Cascades



Cascades



Formal cascade model (Dobson et al)

$\longrightarrow$ Initial fault event takes place (an "act of God").

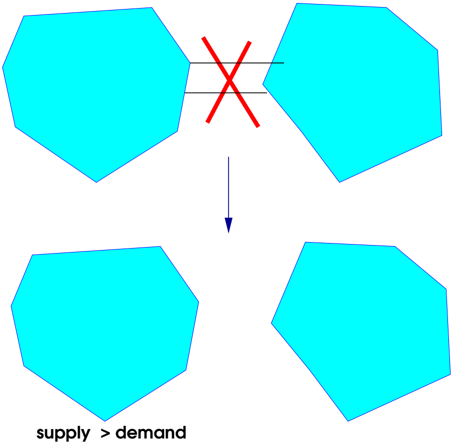
Formal cascade model (Dobson et al)

$\rightarrow$ Initial fault event takes place (an “act of God”).

For $r = 1, 2, \ldots,$

1. Reconfigure demands and generator output levels.

Islanding



Formal cascade model (Dobson et al)

$\rightarrow$ Initial fault event takes place (an “act of God”).

For $r = 1, 2, \ldots,$

1. Reconfigure demands and generator output levels.

Formal cascade model (Dobson et al)

$\longrightarrow$ Initial fault event takes place (an “act of God”).

For $r = 1, 2, \ldots,$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.

Formal cascade model (Dobson et al)

$\longrightarrow$ Initial fault event takes place (an “act of God”).

For $r = 1, 2, \ldots,$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.
3. The next set of faults takes place.

Formal cascade model (Dobson et al)

$\longrightarrow$ Initial fault event takes place (an "act of God").

For $r = 1, 2, \ldots,$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.
3. The next set of faults takes place.
   (Stochastic or history-dependent criterion)

Outage mechanism

f_e = flow on line e

u_e = flow "limit" (threshold) on e

Outage mechanism

f_e = flow on line e

u_e = flow "limit" (threshold) on e

- Prob(e fails) = $F(|f_e|/u_e)$, where $F(x) \to 1$ as $x \to +\infty$.

Outage mechanism

f_e = flow on line e

u_e = flow “limit” (threshold) on e

- Prob(e fails) = $F(|f_e|/u_e)$, where $F(x) \to 1$ as $x \to +\infty$.
- Set $\tilde{f}_e^r = \alpha_e|f_e^r| + (1 - \alpha_e)\tilde{f}_e^{r-1}$, where $0 \leq \alpha_e \leq 1$ is given.

Outage mechanism

f_e = flow on line e

u_e = flow "limit" (threshold) on e

- Prob(e fails) = $F(|f_e|/u_e)$, where $F(x) \to 1$ as $x \to +\infty$.
- Set $\tilde{f}_e^r = \alpha_e |f_e^r| + (1 - \alpha_e)\tilde{f}_e^{r-1}$, where $0 \leq \alpha_e \leq 1$ is given.

  $\to$ $\tilde{f}_e^r$ = running average of $|f_e|$.

  $\to$ r = round (time).

Outage mechanism

f_e = flow on line e

u_e = flow "limit" (threshold) on e

- Prob(e fails) = $F(|f_e|/u_e)$, where $F(x) \to 1$ as $x \to +\infty$.
- Set $\tilde{f}_e^r = \alpha_e |f_e^r| + (1 - \alpha_e)\tilde{f}_e^{r-1}$, where $0 \leq \alpha_e \leq 1$ is given.

$\rightarrow$ $\tilde{f}_e^r$ = running average of $|f_e|$.

$\rightarrow$ r = round (time).

$\rightarrow$ e fails if $\tilde{f}_e > u_e$.

Outage mechanism

f_e = flow on line e

u_e = flow "limit" (threshold) on e

- Prob(e fails) = $F(|f_e|/u_e)$, where $F(x) \to 1$ as $x \to +\infty$.
- Set $\tilde{f}_e^r = \alpha_e |f_e^r| + (1 - \alpha_e)\tilde{f}_e^{r-1}$, where $0 \le \alpha_e \le 1$ is given.

$\to$ $\tilde{f}_e^r$ = running average of $|f_e|$.

$\to$ r = round (time).

$\to$ e fails if $\tilde{f}_e > u_e$. **or:** e fails if $\tilde{f}_e \ge u_e$

Outage mechanism

f_e = flow on line e

u_e = flow "limit" (threshold) on e

- Prob(e fails) = $F(|f_e|/u_e)$, where $F(x) \rightarrow 1$ as $x \rightarrow +\infty$.
- Set $\tilde{f}_e^r = \alpha_e |f_e^r| + (1 - \alpha_e)|f_e^{r-1}|$, where $0 \leq \alpha_e \leq 1$ is given.

  $\rightarrow$ $\tilde{f}_e^r$ = two-round average of $|f_e|$.

  $\rightarrow$ r = round (time).

  $\rightarrow$ e fails if $\tilde{f}_e > u_e$,

Outage mechanism

f_e = flow on line e

u_e = flow "limit" (threshold) on e

- Prob(e fails) = $F(|f_e|/u_e)$, where $F(x) \rightarrow 1$ as $x \rightarrow +\infty$.
- Set $\tilde{f}^r_e = \alpha_e |f^r_e| + (1 - \alpha_e)|f^{r-1}_e|$, where $0 \le \alpha_e \le 1$ is given.

$\longrightarrow$ $\tilde{f}^r_e$ = two-round average of $|f_e|$.

$\longrightarrow$ r = round (time).

$\longrightarrow$ e fails if $\tilde{f}_e > u_e$, (or, e fails if $\tilde{f}_e \ge u_e$)

Stochastic faults

e **fails** if $u_e \ < \ \tilde{f}^r_e$,

Stochastic faults

e **fails** if $u_e < \tilde{f}_e^r$,

e **does not fail** if $(1 - \gamma)u_e > \tilde{f}_e^r$, ($\gamma$ = tolerance)

Stochastic faults

e **fails** if $u_e < \tilde{f}^r_e$,

e **does not fail** if $(1-\gamma)u_e > \tilde{f}^r_e$, ($\gamma$ = tolerance)

if $(1-\gamma)u_e \leq \tilde{f}^r_e \leq u_e$ then e fails **with probability 1/2**

Formal cascade model (Dobson et al)

$\longrightarrow$ Initial outage event takes place (an “act of God”).

For $r = 1, 2, \ldots,$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.
3. The next set of outages takes place.
   (Stochastic or history-dependent criterion)

Formal cascade model (Dobson et al)

→ Initial outage event takes place (an “act of God”).

For $r = 1, 2, \ldots,$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.
3. The next set of outages takes place.
   (Stochastic or history-dependent criterion)

→ If no more faults occur or too much demand has been lost, STOP

Online control

$\longrightarrow$ Initial outage event takes place.

Online control

$\longrightarrow$ Initial outage event takes place. **Compute control algorithm.**

Online control

$\longrightarrow$ Initial outage event takes place. **Compute control algorithm.**

For $r = 1, 2, \ldots, R - 1$

1. Reconfigure demands and generator output levels.

Online control

$\longrightarrow$ Initial outage event takes place. **Compute control algorithm.**

For $r = 1, 2, \ldots, R - 1$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.

Online control

$\longrightarrow$ Initial outage event takes place. **Compute control algorithm.**

For $r = 1, 2, \ldots, R - 1$

1. Reconfigure demands and generator output levels.

2. New power flows are instantiated.

3*a*. Take measurements and apply control to shed demand.

Online control

$\longrightarrow$ Initial outage event takes place. **Compute control algorithm.**

For $r = 1, 2, \ldots, R - 1$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.

3*a*. **Take measurements and apply control to shed demand.**

3*b*. **Reconfigure generator outputs;**

Online control

$\longrightarrow$ Initial outage event takes place. **Compute control algorithm.**

For $r = 1, 2, \ldots, R - 1$

1. Reconfigure demands and generator output levels.
2. New power flows are instantiated.

3*a*. Take measurements and apply control to shed demand.

3*b*. Reconfigure generator outputs; get new power flows.

Online control

$\longrightarrow$ Initial outage event takes place. **Compute control algorithm.**

For $r = 1, 2, \ldots, R - 1$

1. Reconfigure demands and generator output levels.

2. New power flows are instantiated.

3*a*. **Take measurements and apply control to shed demand.**

3*b*. **Reconfigure generator outputs; get new power flows.**

4. The next set of outages takes place.

Online control

$\longrightarrow$ Initial outage event takes place. **Compute control algorithm.**

For $r = 1, 2, \ldots, R - 1$

1. Reconfigure demands and generator output levels.

2. New power flows are instantiated.

3*a*. Take measurements and apply control to shed demand.

3*b*. Reconfigure generator outputs; get new power flows.

4. The next set of outages takes place.

At round R, reduce demands so as to remove any line overloads.

Deterministic, no history model

"Optimal" control via integer programming formulation

Deterministic, no history model

“Optimal” control via integer programming formulation ?

Deterministic, no history model

"Optimal" control via integer programming formulation ?

- f_j^r = flow on arc j at round r
- y_j^r = $\mathbf{1}$, if arc j fails in round r, $\mathbf{0}$ otherwise
- d_j^r = demand at node i in round r
- and many other variables

$$\max \sum_{i \in \mathcal{D}} d_i^R$$

Subject to:

$$\sum_{j \in \delta^+(i)} f_j^r - \sum_{j \in \delta^-(i)} f_j^r = \begin{cases} s_i^r & i \in \mathcal{G} \\ -d_i^r & i \in \mathcal{D} \\ 0 & \text{otherwise} \end{cases} \quad \forall\, 1 \le r \le R \tag{1}$$

$$f_j^r = \pi_j^r - \nu_j^r \quad \forall\, j \in \mathcal{A} \text{ and } 1 \le r \le R \tag{2}$$

$$\pi_j^r \le \tilde{D} p_j^r, \quad \nu_j^r \le \tilde{D} n_j^r, \quad \forall\, j \in \mathcal{A} \text{ and } 1 \le r \le R \tag{3}$$

$$p_j^r + n_j^r = 1 - \sum_{h=1}^{r-1} y_j^h, \quad \forall\, j \in \mathcal{A} \text{ and } 1 \le r \le R \tag{4}$$

$$\pi_j^r + \nu_j^r - u_j \le \tilde{D} y_j^r \quad \forall\, j \in \mathcal{A} \text{ and } 1 \le r \le R \tag{5}$$

$$\pi_j^r + \nu_j^r \ge u_j y_j^r \quad \forall\, j \in \mathcal{A} \text{ and } 1 \le r \le R-1 \tag{6}$$

$$\pi_j^R + \nu_j^R \le u_j \quad \forall\, j \in \mathcal{A} \tag{7}$$

$$|\phi_i^r - \phi_j^r - x_j f_j^r| \le M_j \sum_{h=1}^{r-1} y_j^h \quad \forall j \in \mathcal{A} \tag{8}$$

$$0 \le s_i^r \le \tilde{s}_i \quad \forall\, i \in \mathcal{G}, \quad 0 \le d_i^r \le \tilde{d}_i \quad \forall\, i \in \mathcal{D}, \tag{9}$$

$$p_j^r,\ n_{ij}^r,\ y_j^r = 0 \text{ or } 1, \quad \forall\, j \in \mathcal{A} \text{ and } 1 \le r \le R \tag{10}$$

$$0 \le \pi_j^r,\ 0 \le \nu_j^r, \quad \forall\, j \in \mathcal{A} \text{ and } 1 \le r \le R. \tag{11}$$

What's bad about the formulation

- probably can't solve it for medium to large networks
- stochastic variant probably needed, harder

What's bad about the formulation

- probably can't solve it for medium to large networks
- stochastic variant probably needed, harder
- optimal solutions = complex policies

Adaptive affine controls

For each demand v, and round r, control c_v^r, b_v^r, s_v^r to be computed

Adaptive affine controls

For each demand v, and round r, control c_v^r, b_v^r, s_v^r to be computed

$\rightarrow$ Parameterized by integer $r > 0$.

Adaptive affine controls

For each demand $\boldsymbol{v}$, and round $\boldsymbol{r}$, control $\boldsymbol{c_v^r}$, $\boldsymbol{b_v^r}$, $\boldsymbol{s_v^r}$ to be computed

$\rightarrow$ Parameterized by integer $\boldsymbol{r} > \mathbf{0}$.

At round $\boldsymbol{r}$,

- Let κ = maximum **overload** of any line within radius $\boldsymbol{r}$ of $\boldsymbol{v}$

Adaptive affine controls

For each demand v, and round r, control c_v^r, b_v^r, s_v^r to be computed

$\rightarrow$ Parameterized by integer $r > 0$.

At round r,

- Let κ = maximum **overload** of any line within radius r of v
- If $\kappa > c_v^r$, demand at v reduced (scaled) by a factor

$$\max\left\{1,\ s_v^r\,(c_v^r - \kappa)\ +\ b_v^r\right\}.$$

Adaptive affine controls

For each demand v, and round r, control c_v^r, b_v^r, s_v^r to be computed

$\rightarrow$ Parameterized by integer $r > 0$.

At round r,

- Let κ = maximum **overload** of any line within radius r of v
- If $\kappa > c_v^r$, demand at v reduced (scaled) by a factor

$$\max\left\{1,\ s_v^r\,(c_v^r - \kappa)\ +\ b_v^r\right\}.$$

The goal: pick control to maximize demand being served at the end of round R.

For each demand v, and round r, control $c_v^r,\ b_v^r,\ s_v^r$

At round r, if $\kappa > c_v^r$, demand at v reduced (scaled) by a factor

$$\min \left\{1,\ [\, s_v^r\,(c_v^r - \kappa)\ +\ b_v^r\,]^+ \right\}.$$

For each demand v, and round r, control $c_v^r,\ b_v^r,\ s_v^r$

At round r, if $\kappa > c_v^r$, demand at v reduced (scaled) by a factor

$$\min\left\{1,\ [\, s_v^r\,(c_v^r - \kappa)\ +\ b_v^r\,]^+\ \right\}.$$

This talk: $r = n$ (number of nodes)

For each demand v, and round r, control $c_v^r,\ b_v^r,\ s_v^r$

At round r, if $\kappa > c_v^r$, demand at v reduced (scaled) by a factor

$$\min\left\{1,\ [\,s_v^r\,(c_v^r - \kappa)\ +\ b_v^r\,]^+\ \right\}.$$

This talk: $r = n$ (number of nodes)

Special case: (optimal scaling problem)

Insist that for each r, $(c_v^r,\ b_v^r,\ s_v^r) = (c^r,\ b^r,\ s^r)$ for every v

For each demand v, and round r, control $c_v^r,\ b_v^r,\ s_v^r$

At round r, if $\kappa > c_v^r$, demand at v reduced (scaled) by a factor

$$\min\left\{1,\ [\, s_v^r\,(c_v^r - \kappa)\ +\ b_v^r\,]^+\ \right\}.$$

This talk: $r = n$ (number of nodes)

Special case: (optimal scaling problem)

Insist that for each r, $(c_v^r,\ b_v^r,\ s_v^r) = (c^r,\ b^r,\ s^r)$ for every v

Then, equivalent problem:

- In round r, let $\alpha^r(K) \leq 1$ be chosen for each *component* of the network in round r
- If node $v \in$ component K, then its demand is scaled by $\alpha^r(K)$

Notation:

- $\hat{\beta}$ = supply/demand vector at time 0
- $\hat{f}$ = corresponding power flows at time 0
- $\Theta^R(t, \beta) : \mathcal{R}_+ \rightarrow \mathcal{R}_+$ = total demand, at the end of round R, using optimal control, if the supply/demand vector is $t\ \beta$

Notation:

- $\hat{\beta}$ = supply/demand vector at time 0
- $\hat{f}$ = corresponding power flows at time 0
- $\Theta^R(t, \beta) : \mathcal{R}_+ \rightarrow \mathcal{R}_+$ = total demand, at the end of round R, using optimal control, if the supply/demand vector is $t\,\beta$

Note: supply/demand = $t\beta$ means flow = $t\,\hat{f}$

Notation:

- $\hat{\beta}$ = supply/demand vector at time 0
- $\hat{f}$ = corresponding power flows at time 0
- $\Theta^R(t, \beta) : \mathcal{R}_+ \rightarrow \mathcal{R}_+$ = total demand, at the end of round R, using optimal control, if the supply/demand vector is $t\,\beta$

Note: supply/demand = $t\beta$ means flow = $t\,\hat{f}$

Theorem:

- $\Theta^R(t, \hat{\beta})$ is nondecreasing piecewise-linear with at most $m^R/R! \ + \ O(m^{R-1})$ breakpoints. $m =$ no. of arcs

Notation:

- $\hat{\beta}$ = supply/demand vector at time 0
- $\hat{f}$ = corresponding power flows at time 0
- $\Theta^R(t, \beta) : \mathcal{R}_+ \to \mathcal{R}_+$ = total demand, at the end of round R, using optimal control, if the supply/demand vector is $t\ \beta$

Note: supply/demand = $t\beta$ means flow = $t\ \hat{f}$

Theorem:

- $\Theta^R(t, \hat{\beta})$ is nondecreasing piecewise-linear with at most $m^R/R!\ +\ O(m^{R-1})$ breakpoints. $m =$ no. of arcs
- In round 1, the optimal scale is equal to $u_j/(t\ \hat{f}_j)$ for some arc j.

Notation:

- $\hat{\beta}$ = supply/demand vector at time 0
- $\hat{f}$ = corresponding power flows at time 0
- $\Theta^R(t, \beta) : \mathcal{R}_+ \rightarrow \mathcal{R}_+$ = total demand, at the end of round R, using optimal control, if the supply/demand vector is $t\,\beta$

Note: supply/demand = $t\beta$ means flow = $t\,\hat{f}$

Theorem:

- $\Theta^R(t, \hat{\beta})$ is nondecreasing piecewise-linear with at most $m^R/R! + O(m^{R-1})$ breakpoints. $m =$ no. of arcs
- In round 1, the optimal scale is equal to $u_j/(t\,\hat{f}_j)$ for some arc j. So arc j will become *critical*

Notation:

- $\hat{\beta}$ = supply/demand vector at time 0
- $\hat{f}$ = corresponding power flows at time 0
- $\Theta^R(t, \beta) : \mathcal{R}_+ \rightarrow \mathcal{R}_+$ = total demand, at the end of round R, using optimal control, if the supply/demand vector is $t\ \beta$

Note: supply/demand = $t\beta$ means flow = $t\ \hat{f}$

Theorem:

- $\Theta^R(t, \hat{\beta})$ is nondecreasing piecewise-linear with at most $m^R/R!\ +\ O(m^{R-1})$ breakpoints. $m =$ no. of arcs
- In round 1, the optimal scale is equal to $u_j/(t\ \hat{f}_j)$ for some arc j. So arc j will become *critical*
- And recursively ...

Notation:

- $\hat{\beta}$ = supply/demand vector at time 0
- $\hat{f}$ = corresponding power flows at time 0
- $\Theta^R(t, \beta) : \mathcal{R}_+ \rightarrow \mathcal{R}_+$ = total demand, at the end of round R, using optimal control, if the supply/demand vector is $t\,\beta$

Note: supply/demand = $t\beta$ means flow = $t\,\hat{f}$

Theorem:

- $\Theta^R(t, \hat{\beta})$ is nondecreasing piecewise-linear with at most $m^R/R! \; + \; O(m^{R-1})$ breakpoints. $m =$ no. of arcs
- In round 1, the optimal scale is equal to $u_j/(t\,\hat{f}_j)$ for some arc j. So arc j will become *critical*
- And recursively ...
- Robust/stochastic version?

General case: simulation-based optimization

Given a control vector $\tilde{u} = (c_v^r, b_v^r, s_v^r)$ (over all v and r),

$\Theta(\tilde{u})$ = throughput (total demand) satisfied at end of cascade

- Maximization of $\Theta(\tilde{u})$ should be (very?) fast

General case: simulation-based optimization

Given a control vector $\tilde{u} = (c_v^r, b_v^r, s_v^r)$ (over all v and r),

$\Theta(\tilde{u})$ = throughput (total demand) satisfied at end of cascade

- Maximization of $\Theta(\tilde{u})$ should be (very?) fast
- Optimization should be robust (noisy process)

General case: simulation-based optimization

Given a control vector $\tilde{u} = (c_v^r, b_v^r, s_v^r)$ (over all v and r),

$\Theta(\tilde{u})$ = throughput (total demand) satisfied at end of cascade

- Maximization of $\Theta(\tilde{u})$ should be (very?) fast
- Optimization should be robust (noisy process)
- From a strict perspective, $\Theta(\tilde{u})$ is not even continuous

General case: simulation-based optimization

Given a control vector $\tilde{u} = (c_v^r, b_v^r, s_v^r)$ (over all v and r),

$\Theta(\tilde{u})$ = throughput (total demand) satisfied at end of cascade

- Maximization of $\Theta(\tilde{u})$ should be (very?) fast
- Optimization should be robust (noisy process)
- From a strict perspective, $\Theta(\tilde{u})$ is not even continuous

$\Theta(\tilde{u})$ is obtained through a simulation

Derivative-free optimization

Conn, Scheinberg, Vicente, others

Rough description:

- *Sample* a number of control vectors $\tilde{\boldsymbol{u}}$
- Use the sample points to construct a convex approximation to $\tilde{\boldsymbol{\Theta}}$
- Optimize this approximation; this yields a new sample point

Scalability to large dimensionality?

"First order" method

Given a control vector $\tilde{u}$

1. Estimate the "gradient" $g = \nabla\tilde{\Theta}(\tilde{u})$ through finite differences.

"First order" method

Given a control vector $\tilde{u}$

1. Estimate the "gradient" $g = \nabla\tilde{\Theta}(\tilde{u})$ through finite differences.

   Requires $O(1)$ simulations per demand node.

"First order" method

Given a control vector $\tilde{u}$

1. Estimate the "gradient" $g = \nabla \tilde{\Theta}(\tilde{u})$ through finite differences.
   Requires $O(1)$ simulations per demand node.
2. Estimate step size $\mathrm{argmax}\ \Theta(\tilde{u} + \sigma g)$

"First order" method

Given a control vector $\tilde{u}$

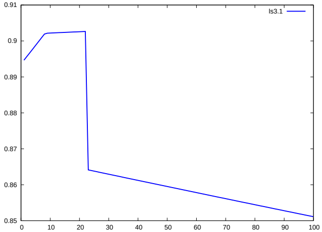
1. Estimate the "gradient" $g = \nabla\tilde{\Theta}(\tilde{u})$ through finite differences.

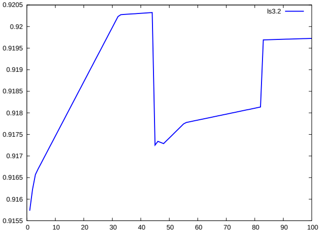
   Requires $O(1)$ simulations per demand node.

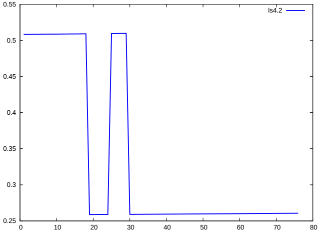
2. Estimate step size $\text{argmax}\ \Theta(\tilde{u} + \sigma g)$

$\longrightarrow$ Easily parallelizable

Line searches







Current parallel implementation: boss-nerd

- Boss carries out search algorithm
- Nerds simulate cascades with given control
- Communication using Unix sockets

Scaling

Example: 10000 nodes, 19309 lines

5 gradient steps

8-core i7 CPUs (3 machines total)

cores	wall-clock sec
2	94379
4	47592
8	28136
16	14618
24	9918

Initial experiments with Eastern Interconnect

- 15023 nodes, 23769 lines.
- 2122 generator nodes, 6261 demand nodes
- “Equivalent” DC flow version

Initial experiments with Eastern Interconnect

- 15023 nodes, 23769 lines.
- 2122 generator nodes, 6261 demand nodes
- “Equivalent” DC flow version
- Methodology for experiments
    1. Generate an interdiction of the grid (“initial event”)
    2. Compute control and simulate

Initial experiments with Eastern Interconnect

- 15023 nodes, 23769 lines.
- 2122 generator nodes, 6261 demand nodes
- “Equivalent” DC flow version
- Methodology for experiments
  1. Generate an interdiction of the grid (“initial event”)
  2. Compute control and simulate
  3. At least three rounds of cascade after initial event

Computing a control

(1) Solve scaling problem – let $(\boldsymbol{c}^*, \boldsymbol{b}^*, \boldsymbol{s}^*)$ be optimal

Computing a control

(1) Solve scaling problem – let $(\boldsymbol{c}^*, \boldsymbol{b}^*, \boldsymbol{s}^*)$ be optimal

(2) Partition demand nodes into “small” number of segments $\Sigma_1, \ldots, \Sigma_k$.

Computing a control

(1) Solve scaling problem – let $(\boldsymbol{c}^*, \boldsymbol{b}^*, \boldsymbol{s}^*)$ be optimal

(2) Partition demand nodes into "small" number of segments $\Sigma_1, \ldots, \Sigma_k$. Example = demand quantiles.

Computing a control

(1) Solve scaling problem – let $(\boldsymbol{c}^*, \boldsymbol{b}^*, \boldsymbol{s}^*)$ be optimal

(2) Partition demand nodes into "small" number of segments $\Sigma_1, \ldots, \Sigma_k$. Example = demand quantiles.

Perform segmented gradient search starting from $(\boldsymbol{c}^*, \boldsymbol{b}^*, \boldsymbol{s}^*)$.

Look for a control with $(\boldsymbol{c}^r_v, \boldsymbol{b}^r_v, \boldsymbol{s}^r_v)$ = constant for each given $\boldsymbol{r}$ and all $\boldsymbol{v}$ in a common $\boldsymbol{\Sigma_i}$.

Computing a control

(1) Solve scaling problem – let (c^*, b^*, s^*) be optimal

(2) Partition demand nodes into "small" number of segments $\Sigma_1, \ldots, \Sigma_k$. Example = demand quantiles.

Perform segmented gradient search starting from (c^*, b^*, s^*).

Look for a control with (c^r_v, b^r_v, s^r_v) = constant for each given r and all v in a common Σ_i.

(3) Perform full gradient search starting from the output in (2).

Experiments

- **K** random lines taken out
- highly loaded lines more likely to be taken out; connectivity preserved

Experiments

- **K** random lines taken out
- highly loaded lines more likely to be taken out; connectivity preserved

K	yield, (%) no control	yield, control	wallclock (sec)
1	90.04	95.03	134
2	12.54	50.13	87
5	32.94	81.05	107
10	2.02	36.97	97
20	1.64	27.84	159
50	0.83	16.96	209

Conjectures

- It is best to stop the cascade in the first round
- It is best to apply control in the first round only, and ride out the cascade

Conjectures

- It is best to stop the cascade in the first round
- It is best to apply control in the first round only, and ride out the cascade

(Answer: both wrong)

Details: cascade with 50 (highly loaded) random lines taken out

- No control $\Rightarrow$ yield = **0%**
- Optimal round 1 only constant control $\Rightarrow$ yield = **38%**
- Optimal scaling control $\Rightarrow$ yield = **45%**
- Plus segmented gradient seach $\Rightarrow$ yield = **50%**

Load distribution at time zero (load of arc $j = \frac{|f_j|}{u_j}$)

load	no. of arcs
1505	1
58	1
48	2
32	1
22	2
19	1
11	1
7	2
6	2
5	4
4	6
3	18
2	181

Load distribution at time zero

(load of arc $j = \frac{|f_j|}{u_j}$)

load	no. of arcs
1505	1
58	1
48	2
32	1
22	2
19	1
11	1
7	2
6	2
5	4
4	6
3	18
2	181

Optimal round 1 scale = **0.51**,

Load distribution at time zero

(load of arc $j = \frac{|f_j|}{u_j}$)

load	no. of arcs
1505	1
58	1
48	2
32	1
22	2
19	1
11	1
7	2
6	2
5	4
4	6
3	18
2	181

Optimal round 1 scale = **0.51**, so **44** faults

Out-of-sample testing: use stochastic faults

at round r,

e **fails** if $u_e < \tilde{f}_e^r$,

e **does not fail** if $(1 - \gamma) u_e > \tilde{f}_e^r$, ($\gamma$ = tolerance)

if $(1 - \gamma) u_e \leq \tilde{f}_e^r \leq u_e$ then e fails **with probability 1/2**

Out-of-sample testing: use stochastic faults

at round r,

e **fails** if $u_e < \tilde{f}^r_e$,

e **does not fail** if $(1-\gamma)u_e > \tilde{f}^r_e$, ($\gamma$ = tolerance)

if $(1-\gamma)u_e \le \tilde{f}^r_e \le u_e$ then e fails **with probability 1/2**

What is the impact of γ?

$\gamma = 0.03, 0.10, 0.20,$

10000 runs