Controlling Cascading Failures in Interdependent Networks under Incomplete Knowledge

Diman Zad Tootaghaj[1]

Hana Khamfroush[1]

Novella Bartolini[2]

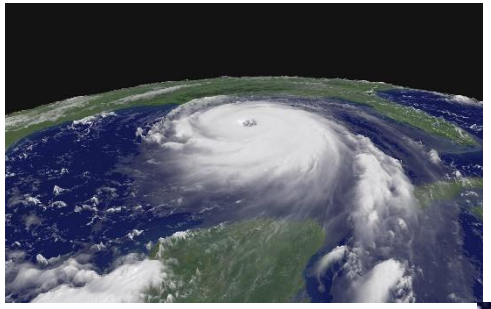
Thomas La Porta[1]

[1] The Pennsylvania State University (US)

[2] Sapienza University (Italy)

Outline

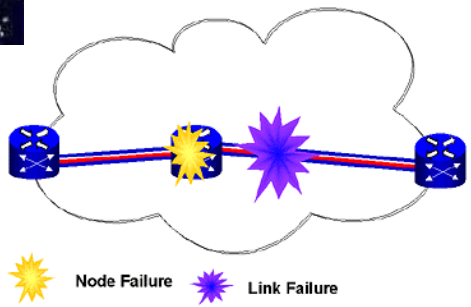
- Motivation
- Problem Definition
- Proposed Algorithms
- Evaluation
- Conclusion



Natural disaster

Malicious attacks

Hurricane



Random Failures

Natural

Recovery approaches may not work as they should due to lack of knowledge and uncertainty

Hurricane

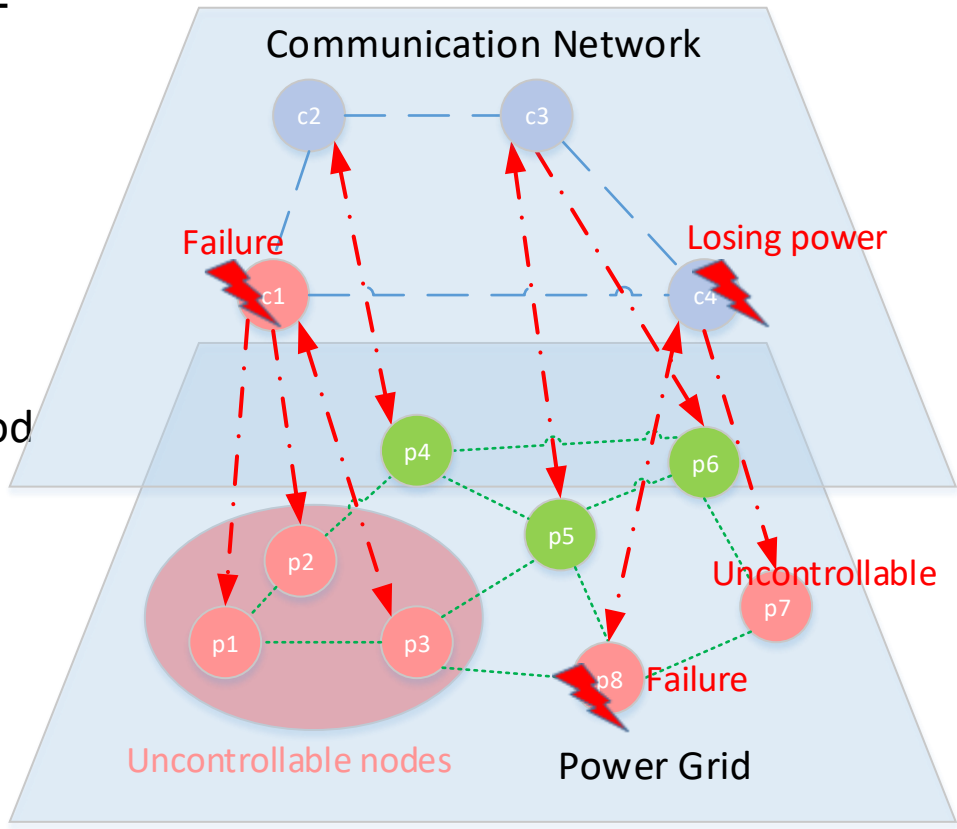
Random Failures

Examples of Major Blackouts:

People Affected (millions)	Date	Location
620	30–31 July 2012	India
230	2 January 2001	India
150	1 November 2014	Bangladesh
100	18 Aug 2005	Indonesia
97	11 March 1999	Brazil
87	10–11 Nov 2009	Brazil, Paraguay
70	31 March 2015	Turkey
55	14–15 August 2003	United States, Canada
55	28 September 2003	Italy, Switzerland
44	7 June 2016	Kenya

Disruption in Multiple Networks

- **Failures in multiple networks:**
  - Critical infrastructures are highly **correlated** and **dependent**
    - Power grid, Communication network.
    - Transportation network and food supply.
  - Failure in one network causes failure in the other network.
  - Recovery plan can be complicated without complete knowledge



Interdependent Networks

Observations:

The operation and reliability of **power grid** is highly dependent on the operation of the **communication network** that provides the necessary information needed by the **SCADA** system.

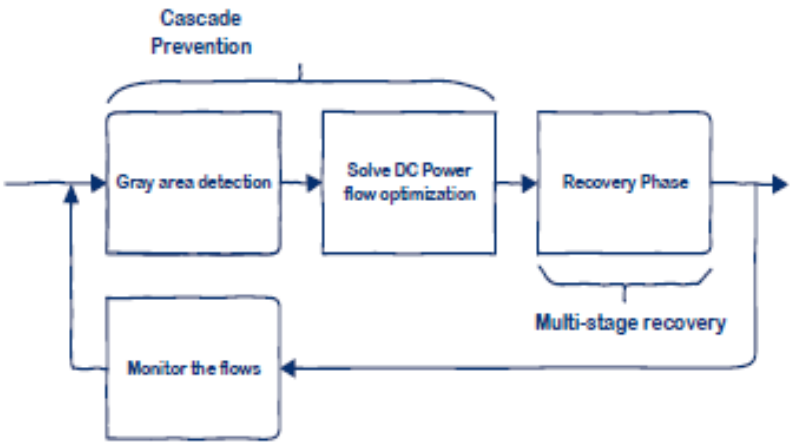
Restoring the power grid after a cascaded failure is not a **one shot** algorithm and requires **time** and **resources.**

Key Idea: 2-phase recovery approach: 1) Preventing the cascade (Detecting the failures, DC power flow optimization), 2) Recovery phase.

Results:

Cascade Prevention: Higher delivered power (**54.39%** delivered power when 60% of network is disrupted).

Recovery Phase: Higher delivered power (**20%** more in **backward algorithm** compared to **shadow-pricing approach**).



Outline

- Motivation
- Problem Definition
- Proposed Algorithms
- Evaluation
- Conclusion

(1) Preventing the Cascade

Key idea:

Minimum Cost Flow Assignment (Min-CFA): Re-distribute the power to find a minimum cost feasible solution without overloading other lines.

1) Solution is found using Linear programming (LP) optimization.
2) What if the exact location of failure is unknown or partially known?
    1) Consistent Failure set (**CFS**) algorithm
    2) Finds the exact location of failure if the unknown graph is **cycle-free**.
    3) If **multiple** consistent failure sets: choose one using **local inspection**.

$$\text{minimize} \quad \sum_{G_i, L_j \in V_p} w_{G_i}(P^0_{G_i} - P^t_{G_i}) - w_{L_j}(P^t_{L_j} - P^0_{L_j})$$

$$\text{subject to} \quad 0 \leqslant P^t_{G_i} \leqslant P^0_{G_i}, \ \forall G_i \in V^t_p$$

$$0 \leqslant P^t_{L_j} \leqslant P^{demand}_{L_j}, \ \forall L_j \in V^t_p$$

$$-F^{max}_{ij} \leqslant F^t_{ij} \leqslant F^{max}_{ij}, \ \forall (ij) \in E^t_p$$

$$\sum_{G_i, L_j \in V_p} P^t_{G_i} + P^t_{L_j} = 0.$$

$$P^t_{G_i} = \sum_j F^t_{ij}, \ \forall G_i \in V^t_p, (ij) \in E^t_p$$

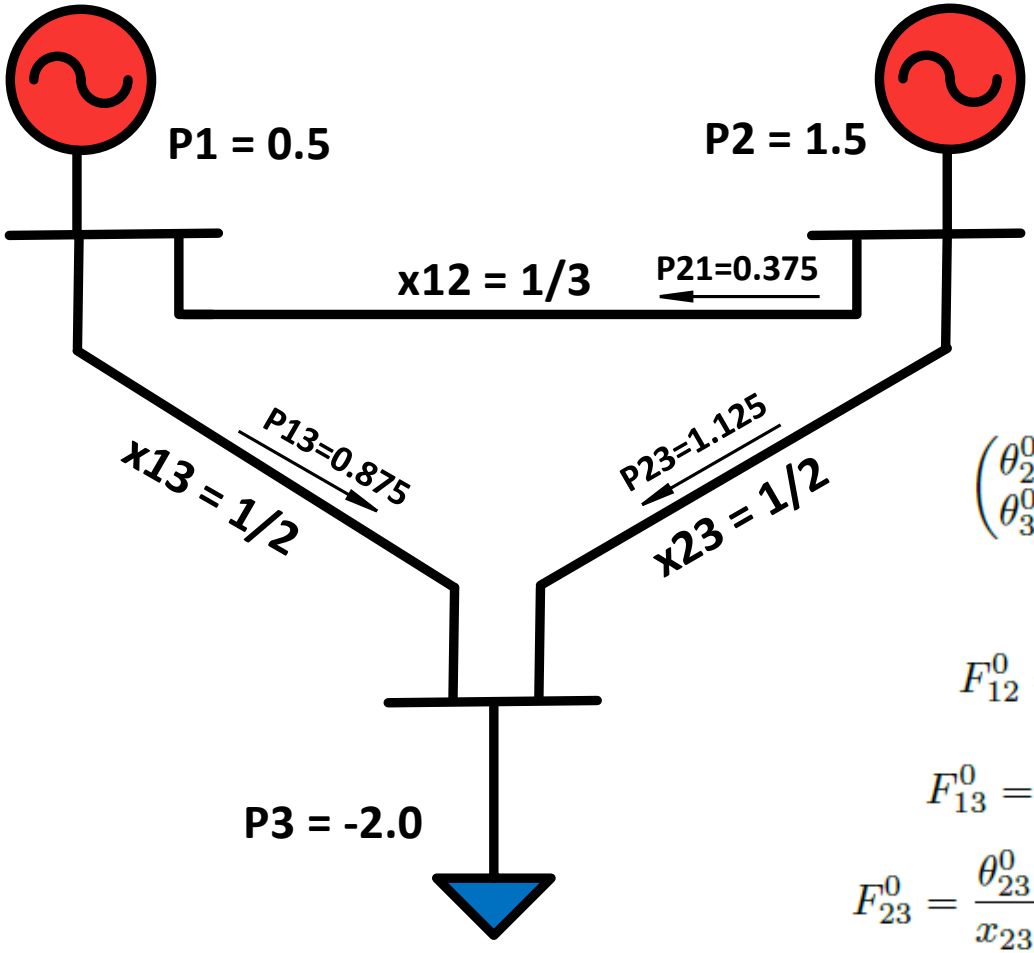
$$P^t_{L_i} = \sum_j F^t_{ij}, \ \forall L_i \in V^t_p, (ij) \in E^t_p$$

$$P^t_{G_i} = B^t \theta^t, \ \forall G_i \in V^t_p$$

$$P^t_{L_j} = B^t \theta^t, \ \forall L_j \in V^t_p$$

$$F^t_{ij} = \frac{(\theta^t_i - \theta^t_j)}{x_{ij}}, \ \forall (ij) \in E^t_p$$

(1) Preventing the Cascade



All lines working:

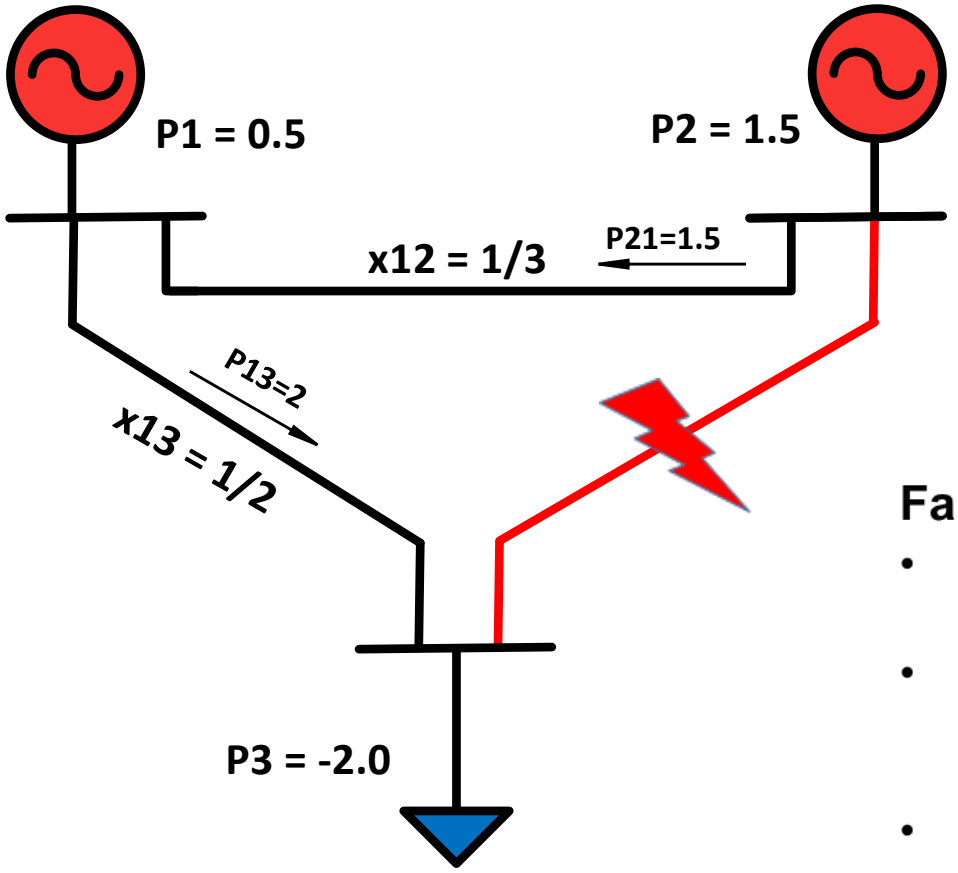
$$\begin{pmatrix}\theta_2^0\\ \theta_3^0\end{pmatrix} = \begin{pmatrix}5 & -2\\ -2 & 4\end{pmatrix}^{-1}\begin{pmatrix}1.5\\ -2\end{pmatrix} = \begin{pmatrix}0.125\\ -0.4375\end{pmatrix}$$

$$F_{12}^0 = \frac{\theta_{12}^0}{x_{12}} = 3\times(0-0.125) = -0.3750,$$

$$F_{13}^0 = \frac{\theta_{13}^0}{x_{13}} = 2\times(0-(-0.4375)) = 0.875,$$

$$F_{23}^0 = \frac{\theta_{23}^0}{x_{23}} = 2\times(0.125-(-0.4375)) = 1.125.$$

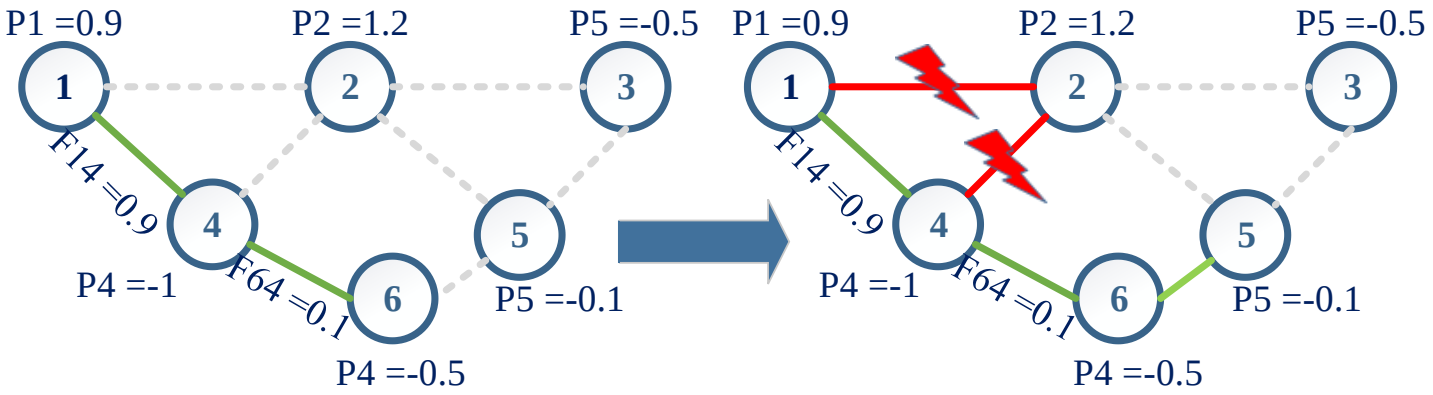
(1) Preventing the Cascade



Failure in line (2-3):

- If Threshold = 1.3 -> The whole system collapses,
- However if we knew the failure location the cascaded failure could be avoided using Min-CFA
- One trivial solution of Min-CFA is to reduce the first generator's power to $P_2^1 = 0.8$ and load $P_3^1 = -1.3$.

(1) Detecting the failures



Key idea:

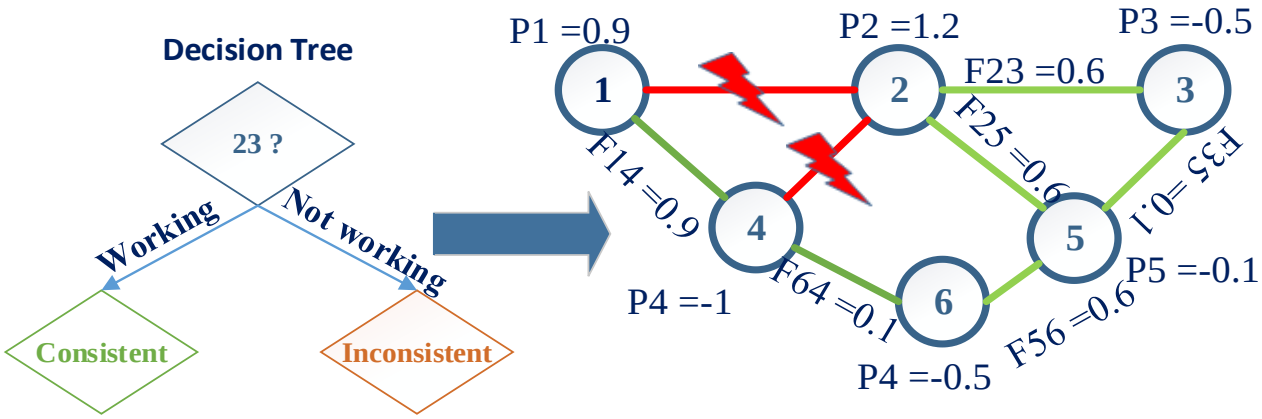
Detecting the gray area by finding consistent failure sets.

If more than one consistent failure set: Use local inspection

1) If the gray area does not contain any cycles, the detection is easy
2) If the gray area consists of cycles:

   Use a decision tree and solve the DC power flow model to find consistent sets.

(1) Detecting the failures



Key idea:

Detecting the gray area by finding consistent failure sets.

If more than one consistent failure set: Use local inspection

1) If the gray area does not contain any cycles, the detection is easy
2) If the gray area consists of cycles:

   Use a decision tree and solve the DC power flow model to find consistent sets.

(2) Recovery phase

Key idea:

Maximum Recovery (Max-R): Recover the lines in k steps such the total delivered power during k steps is maximized.

Max-R is **NP-Hard** and intractable.

$$\text{maximize} \quad \sum_{k=1}^{K} \sum_{L_j \in V_p} P_{L_j}^{k}(Rep_k) \ ,$$

$$\text{subject to} \quad \sum_{m=1}^{k} \sum_{(ij) \in E_k^R} \delta_{(ij),m}.r_{ij} \leq \sum_{m=1}^{k} R_m \quad k = 1, ..., K \ ,$$

$$\sum_{k=1}^{K} \delta_{(ij),k} \leq 1, \ \forall (ij) \in E_k^R \quad k = 1, ..., K \ ,$$

$$\delta_{(ij),k} \in \{0, 1\}, \ \forall (ij) \in E_k^R \quad k = 1, ..., K \ ,$$

Recovery Heuristics:

1. **Shadow pricing approach:** Greedily recover power lines that add more to the total delivered power per unit of cost using Min-CFA.
2. **Backward approach:** Solve a single-stage assuming $R_1 + R_2 + \ ... + R_k$ resources are available, and then solve for $R_1 + \ ... + R_{k-1}$ to find the solution for step k

Outline

- Motivation
- Problem Definition
- Proposed Algorithms
- Evaluation
- Conclusion

Evaluation (Methodology)

Datasets used in our evaluation:

Italian high voltage power grid (hviet) [1],
Communication network (garr) [1].

Power grid (hviet)

Communication network (garr)

Implementation:

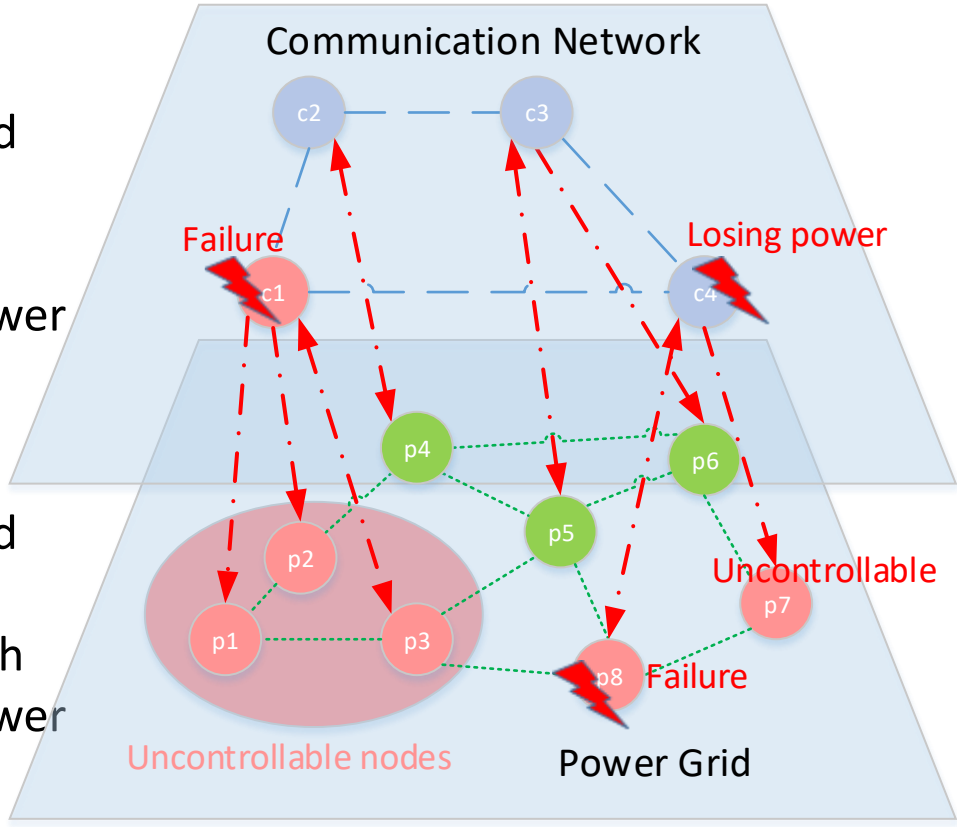
- **Python, Networkx, Gurobi optimization toolkit**

Solving LP optimization: gurobi optimization toolkit.
Detection: CFS algorithm.
Multiple consistent failure sets: local inspection.

[1] V. Rosato et al. Modelling interdependent infrastructures using interacting dynamical models. International Journal of Critical Infrastructures, 2008.

Impact of Dependency Model

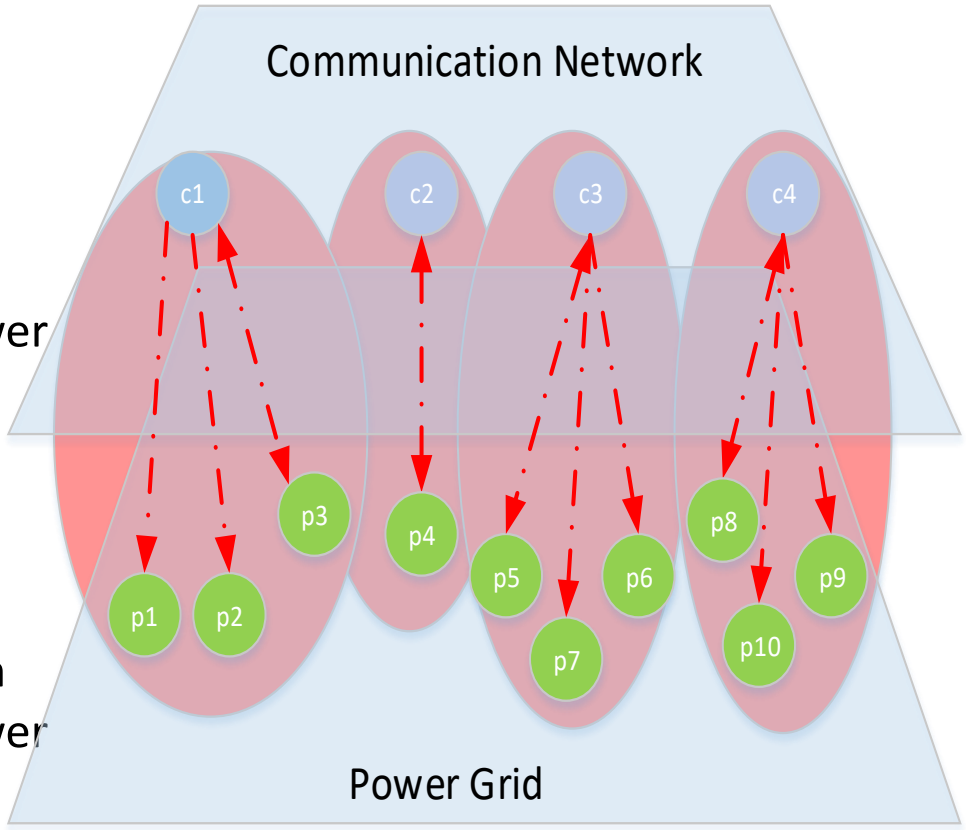
- **Three types of dependency:**
  - **Location-based**
    - Each power node is monitored and controlled by the closest power node and each communication node gets power from the closest power node.
  - **Random**
    - Each power node is monitored and controlled by a random communication node and each communication node gets power from a random power node
  - **One-way**
    - Communication nodes get power from an external source.



Location-based Dependency Model

Impact of Dependency Model

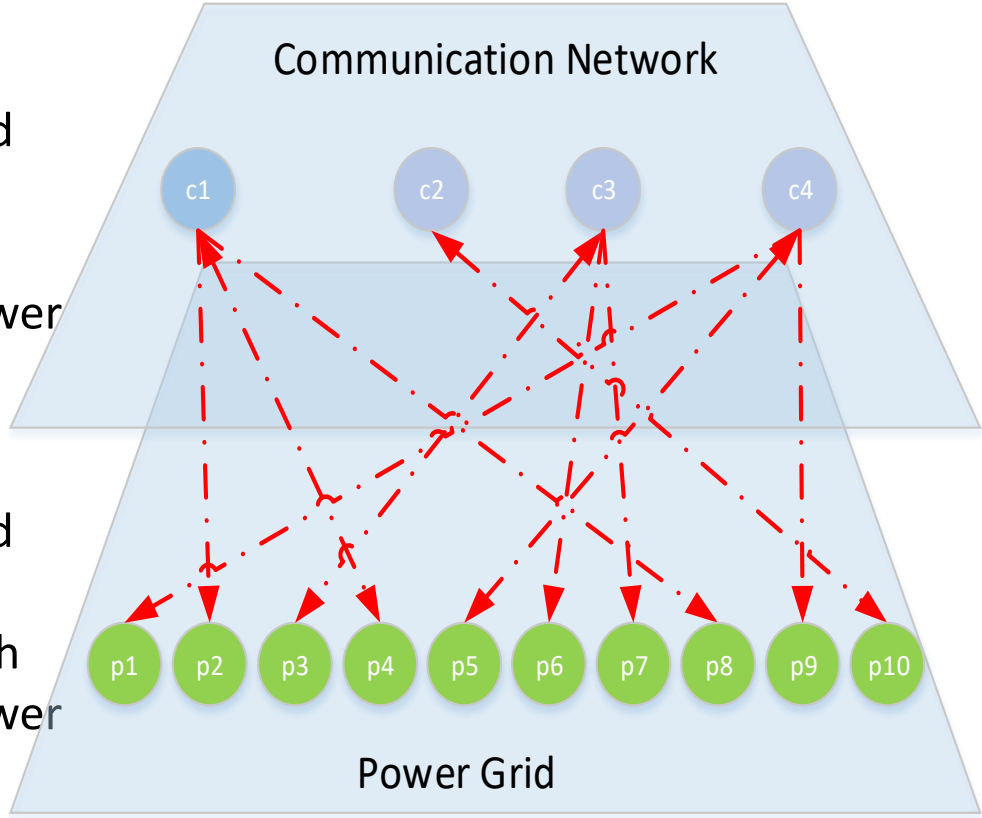
- **Three types of dependency:**
  - **Location-based**
    - Each power node is monitored and controlled by the closest power node and each communication node gets power from the closest power node.
  - **Random**
    - Each power node is monitored and controlled by a random communication node and each communication node gets power from a random power node
  - **One-way**
    - Communication nodes get power from an external source.



Location-based Dependency Model

Impact of Dependency Model

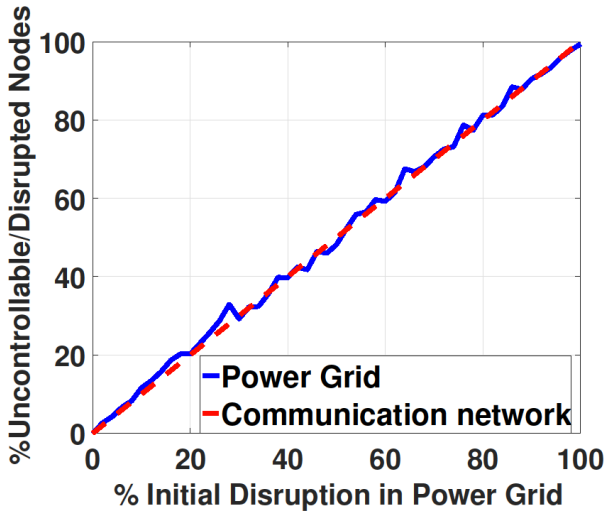
- **Three types of dependency:**
  - **Location-based**
    - Each power node is monitored and controlled by the closest power node and each communication node gets power from the closest power node.
  - **Random**
    - Each power node is monitored and controlled by a random communication node and each communication node gets power from a random power node
  - **One-way**
    - Communication nodes get power from an external source.



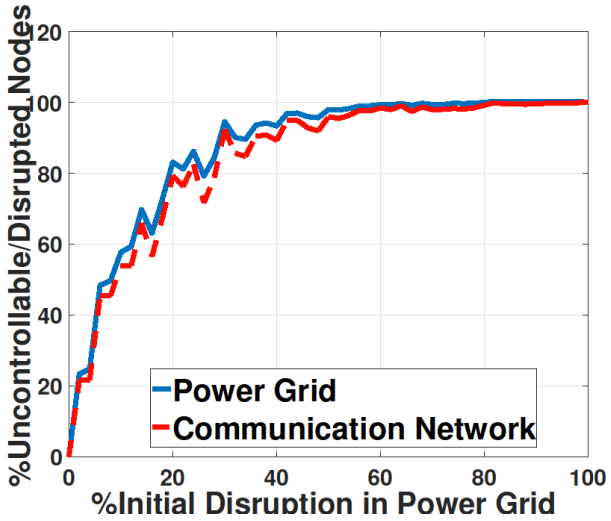
Random Dependency Model

Impact of Dependency Model

Percentage of uncontrollable/disrupted nodes



Location-based

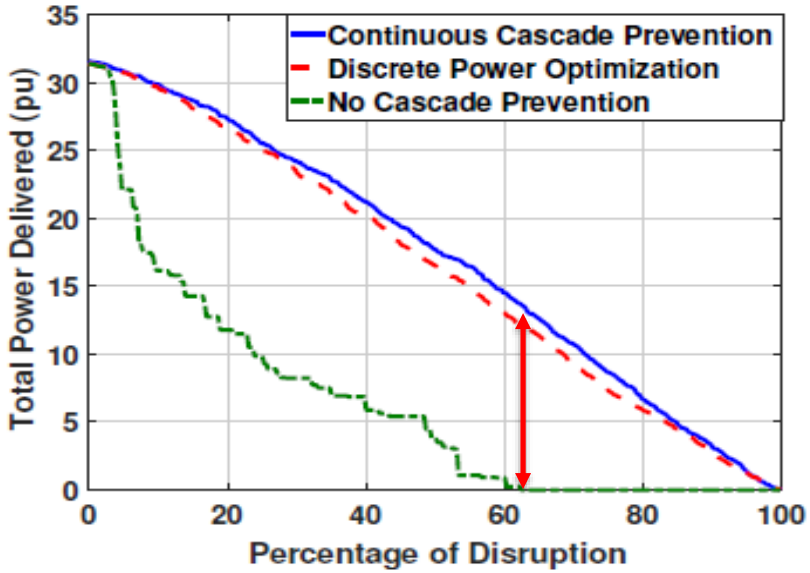


Random

In a **random** dependency model the disruption **spreads** more in the two networks while in a **location-based** dependency model the disruption is limited to the initial failed area

Evaluation (Total Delivered Power)

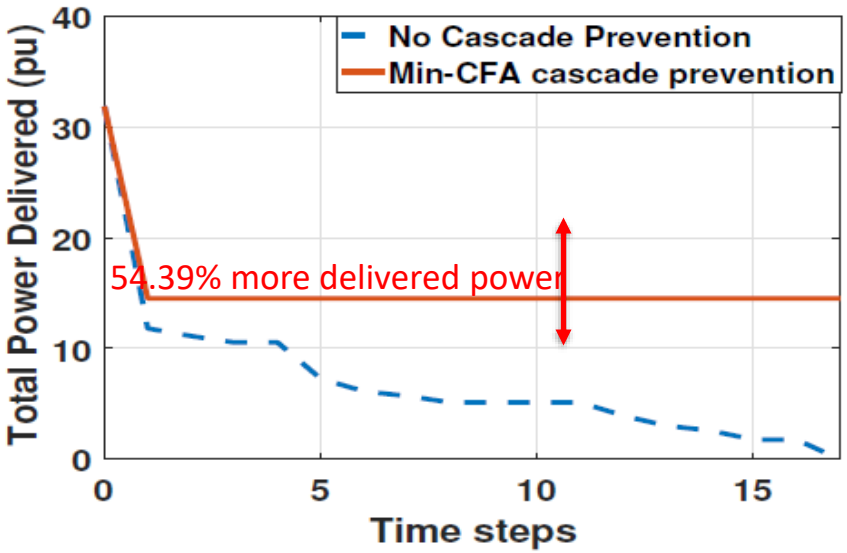
Total Delivered Power vs Percentage of Disruption



Without a cascade prevention approach the whole system fails when we **60%** of the lines are disrupted, while **our approach** can save **50%** of the power.

Evaluation (Cascade Prevention)

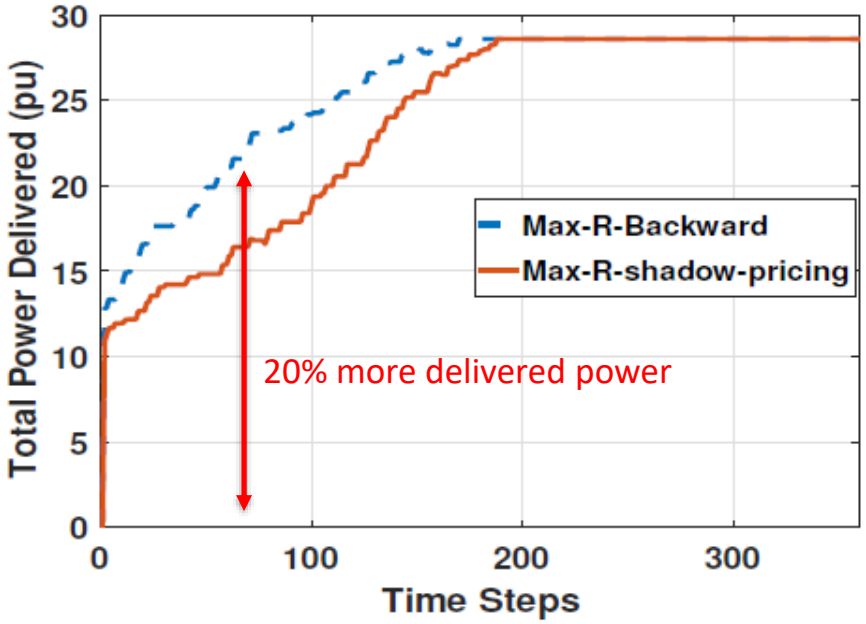
Re-distribution of Load to prevent the cascade:



Cascade Prevention (during time, 60% disruption),
54.39% more power **delivered**

Evaluation (Recovery Phase)

Comparison between backward and shadow-pricing



Recovery Phase: **Backward** Algorithm restores **20%** more power with respect to **shadow-pricing** approach

Conclusion

- **Observations**:
  - Large-scale **failures** in due to natural **disasters** or **malicious attacks** can severely affect complex networks and threaten lives of people.
  - In real-world scenarios, the failure pattern might be **unknown** or only **partially known**.
- **Key Idea**: Use a failure detection algorithm and re-distribute the power to prevent the cascade.
- **Results:**

  Cascade Prevention: Higher delivered power (**54.39%** delivered power when 60% of network is disrupted).

  Recovery Phase: Higher delivered power (**20%** more in **backward algorithm** compared to **shadow-pricing approach**).

Questions?

Hidden Slides

Evaluation

Dataset:
Italian high voltage power grid (hviet), Communication network (garr) [1].

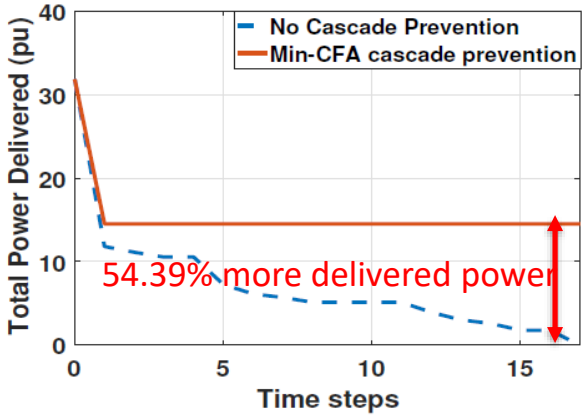
Solving LP optimization: gurobi optimization toolkit.

Detection: CFS algorithm.

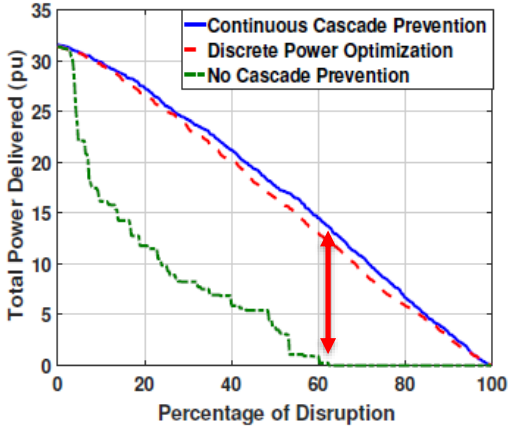
Multiple consistent failure sets: local inspection.

Power grid (hviet)

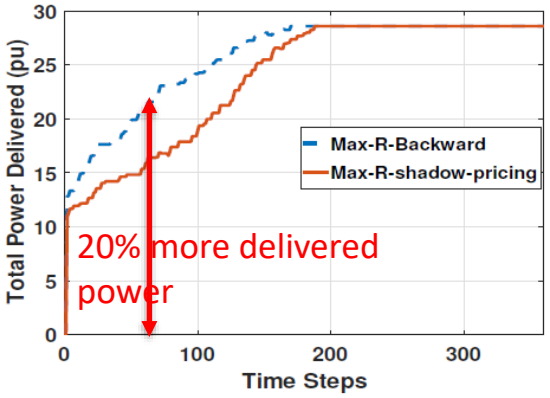
Communication network (garr)



Cascade Prevention (during time, 60% disruption)



Cascade Prevention (Percentage of disruption)



Recovery Phase

[1] V. Rosato et al. Modelling interdependent infrastructures using interacting dynamical models. International Journal of Critical Infrastructures, 2008.