UNIVERSITY OF MINNESOTA

ASP-DAC 2010

Baktash Boghrati, Sachin Sapatnekar

Incremental Solution of Power Grids using Random Walks

Outline

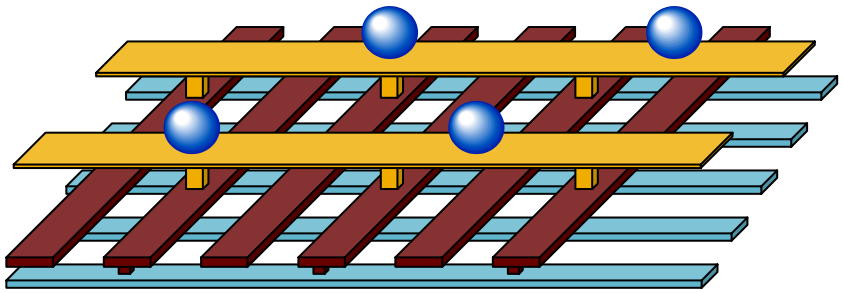
- Introduction
- Big picture
- Random walks
- Incremental solver algorithm
- Experimental results
- Conclusion

Introduction

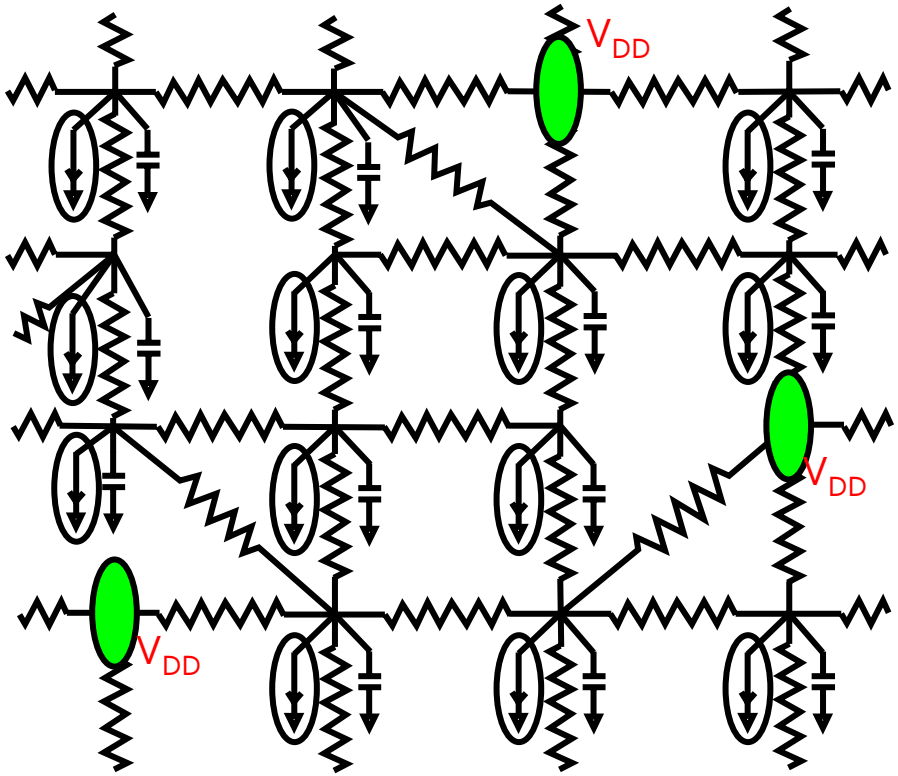
- Power grid modeled as RLC network
- Steady state:

$$G\mathbf{V} = \mathbf{E}$$

$$V_{DD}: \text{ fixed}$$



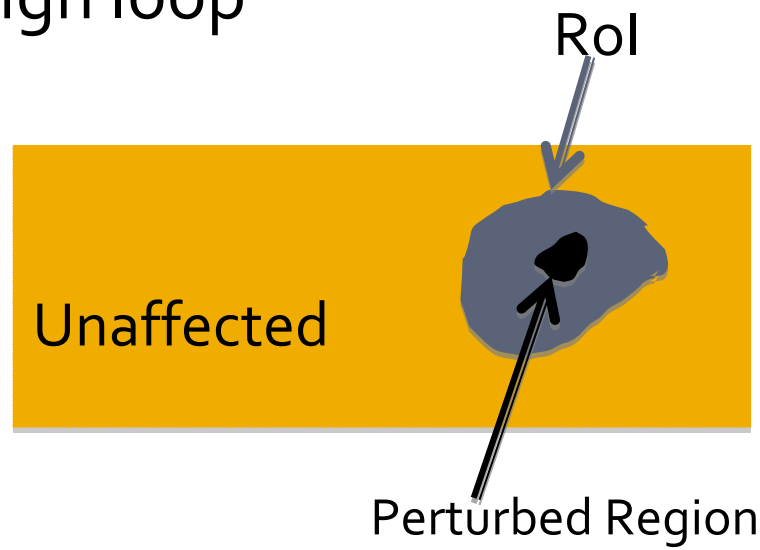
[Qian et al., DAC03]



[Qian et al., DAC03]

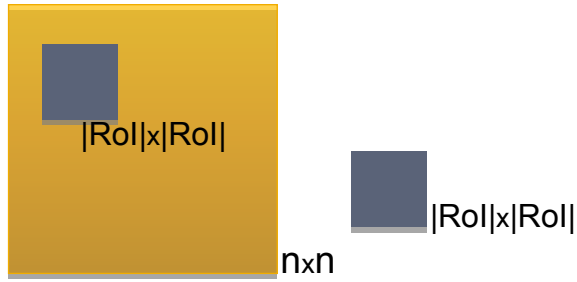
Incremental Solution

- Why incremental?
  - What-if scenarios within design loop
- Small perturbation
  - Solution *close* to original
  - *Locality* of change
- Define the concept of a *Region of Influence* (RoI)
  - RoI = set of nodes with "significant" change
  - Only solve for RoI



Big Picture

- The original solution is given
- Generate *bookkeeping info* from the random walks
  - Based on traditional random walk solver [Qian *et al.*, DAC03]
  - A one-time preprocessing phase
- Find the RoI for given perturbation
  - Use *bookkeeping info*
  - Approximate solution is found as well



- Solve for the nodes within RoI, keeping the original solution for others (Refinement phase)

Steady State Power Grid

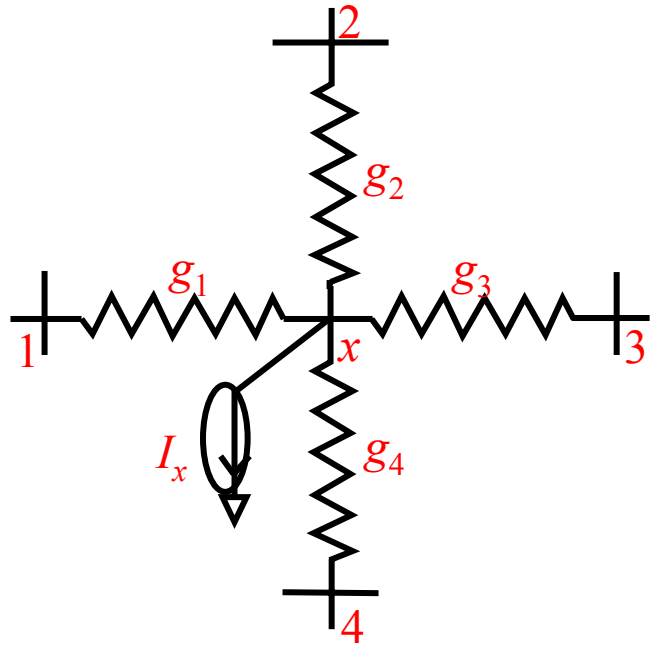
- Kirchhoff equations at node *x*

$$(V_1 - V_x)g_1 + (V_2 - V_x)g_2 + (V_3 - V_x)g_3 + (V_4 - V_x)g_4 = I_x$$

- Alternatively:

$$V_x - \frac{g_1}{\sum g} V_1 - \frac{g_2}{\sum g} V_2 - \frac{g_3}{\sum g} V_3$$

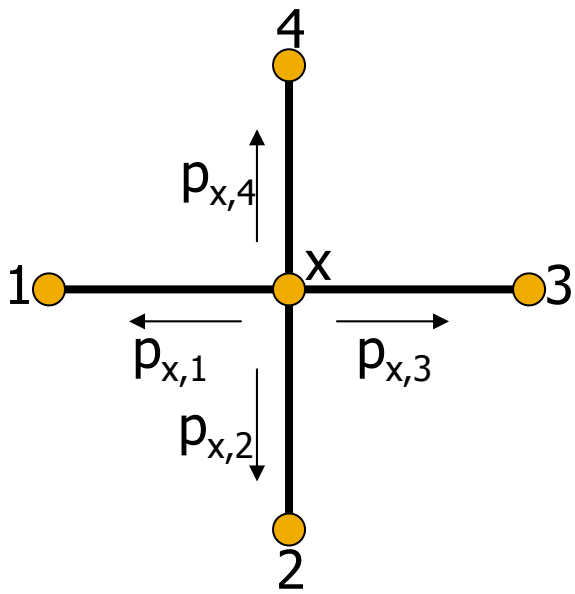
$$- \frac{g_4}{\sum g} V_4 = - \frac{I_x}{\sum g}$$



V: Node voltages, are unknown

Random Walks

- Start a walk from node x
- Randomly choose a road
- Pay for motel at the road (m_i)
- Continue until a home is reached and get a reward (m_o)



- Expected total money starting from node x: **f(x)**

$$f(x) = \sum_{i=1}^{\text{degree}(x)} p_{x,i} f(i) - m_x \quad \text{OR} \quad f(\text{home}) = m_0$$

Random Walks for Solving Circuits

- Remember power grid equation for node *x*:

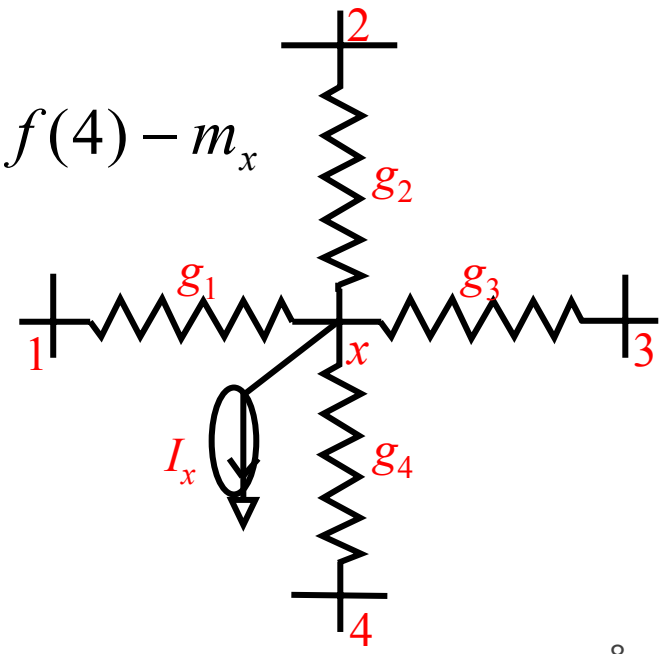
$$V_x = \frac{g_1}{\sum g} V_1 + \frac{g_2}{\sum g} V_2 + \frac{g_3}{\sum g} V_3 + \frac{g_4}{\sum g} V_4 - \frac{I_x}{\sum g}$$

- From random walk equation:

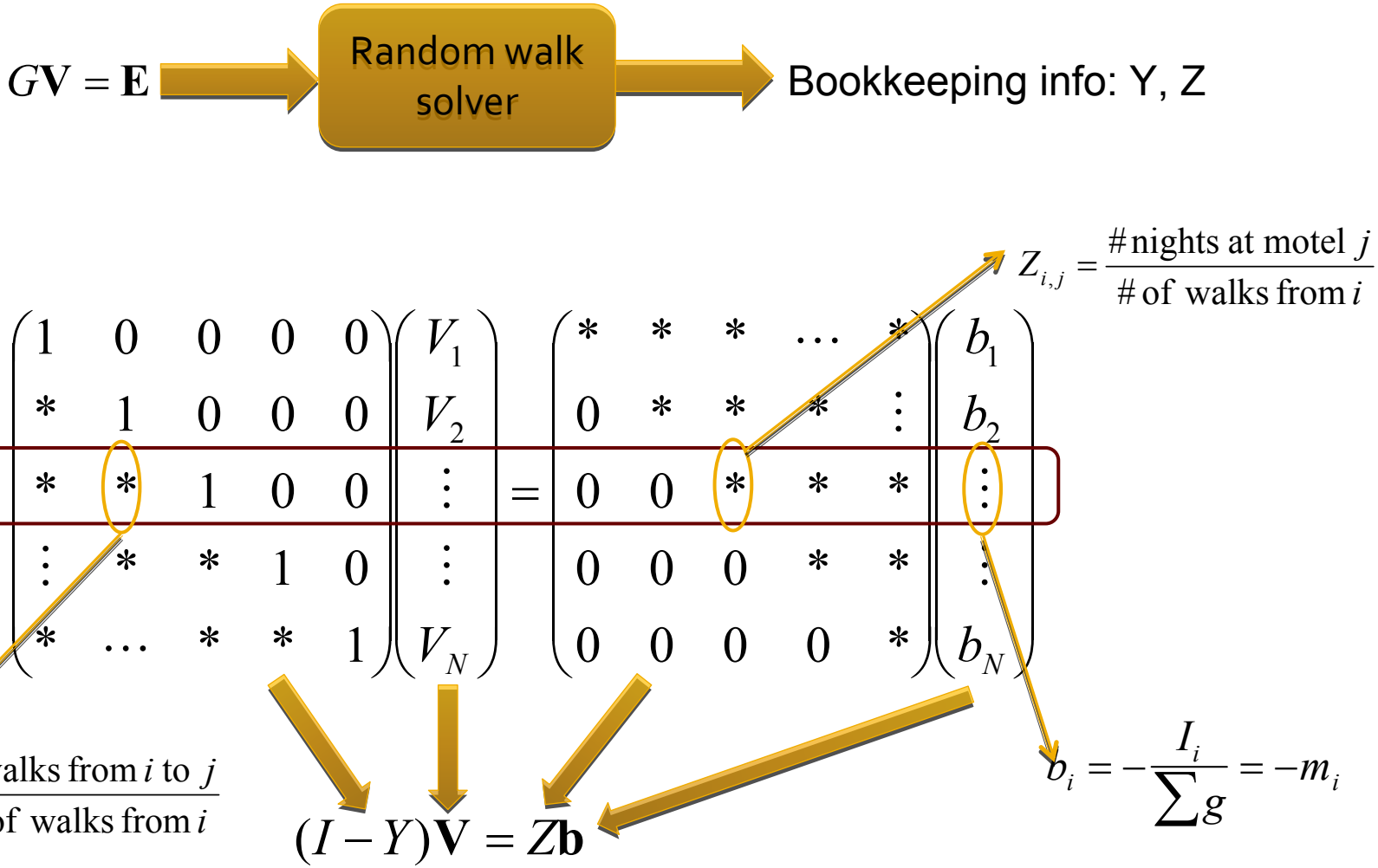
$$f(x) = p_{x,1} f(1) + p_{x,2} f(2) + p_{x,3} f(3) + p_{x,4} f(4) - m_x$$

$$m_0 = V_{pad} = V_{DD} \quad or \quad 0$$

DC analysis ⟺ Random walk



Bookkeeping Info



The Perturbed System

- Perturbed system equation:

$$(G + \Delta\mathbf{V})(\mathbf{V} + \Delta\mathbf{V}) = \mathbf{E} + \Delta\mathbf{E}$$

- Ignore second order term:

$$G\Delta\mathbf{V} + \Delta G\mathbf{V} = \Delta\mathbf{E}$$

$$i.e.\ \ G\Delta\mathbf{V} = \Delta\hat{\mathbf{E}}$$

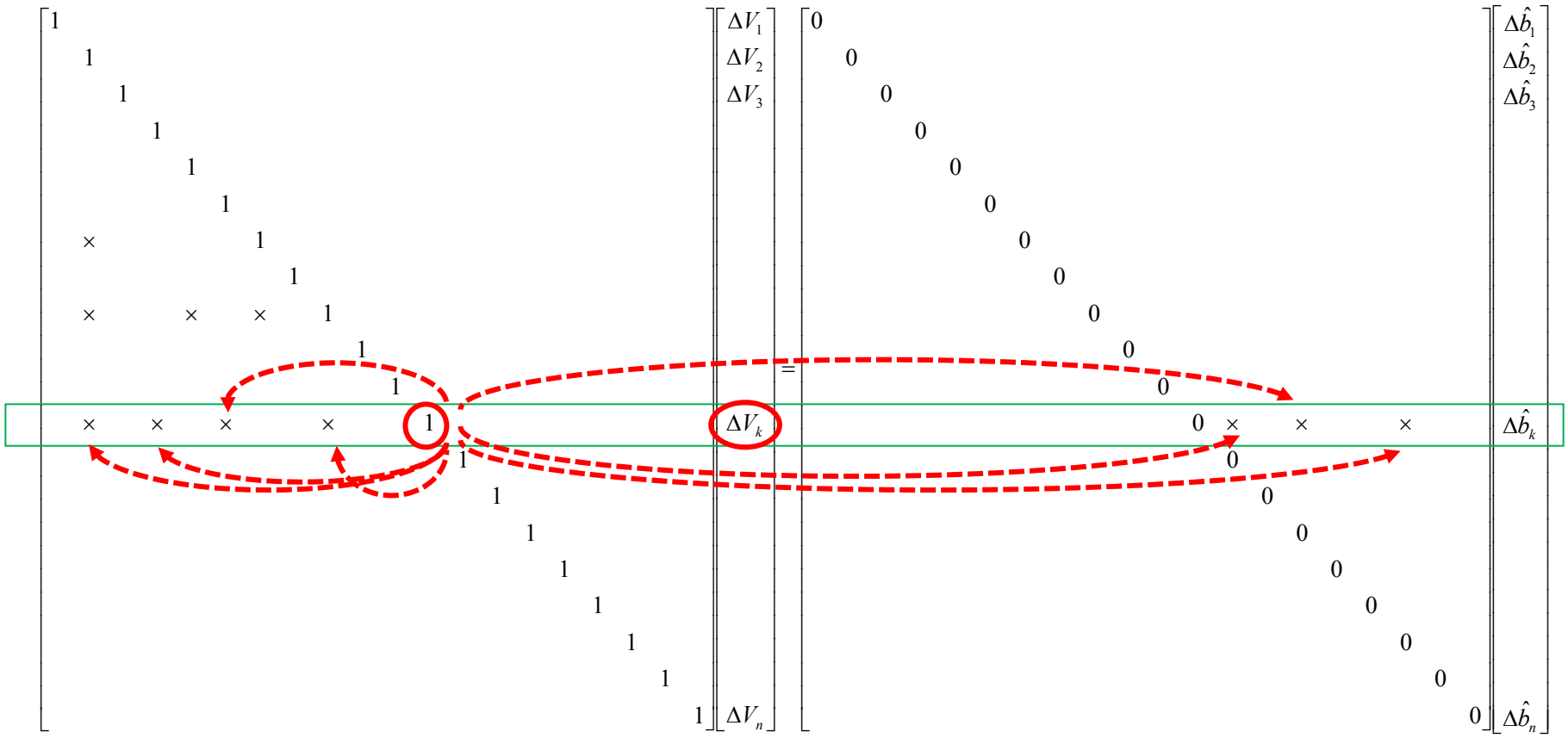
Any Perturbation modeled in RHS

- Bookkeeping info gives the voltage change for *any* perturbation with RHS defined as:

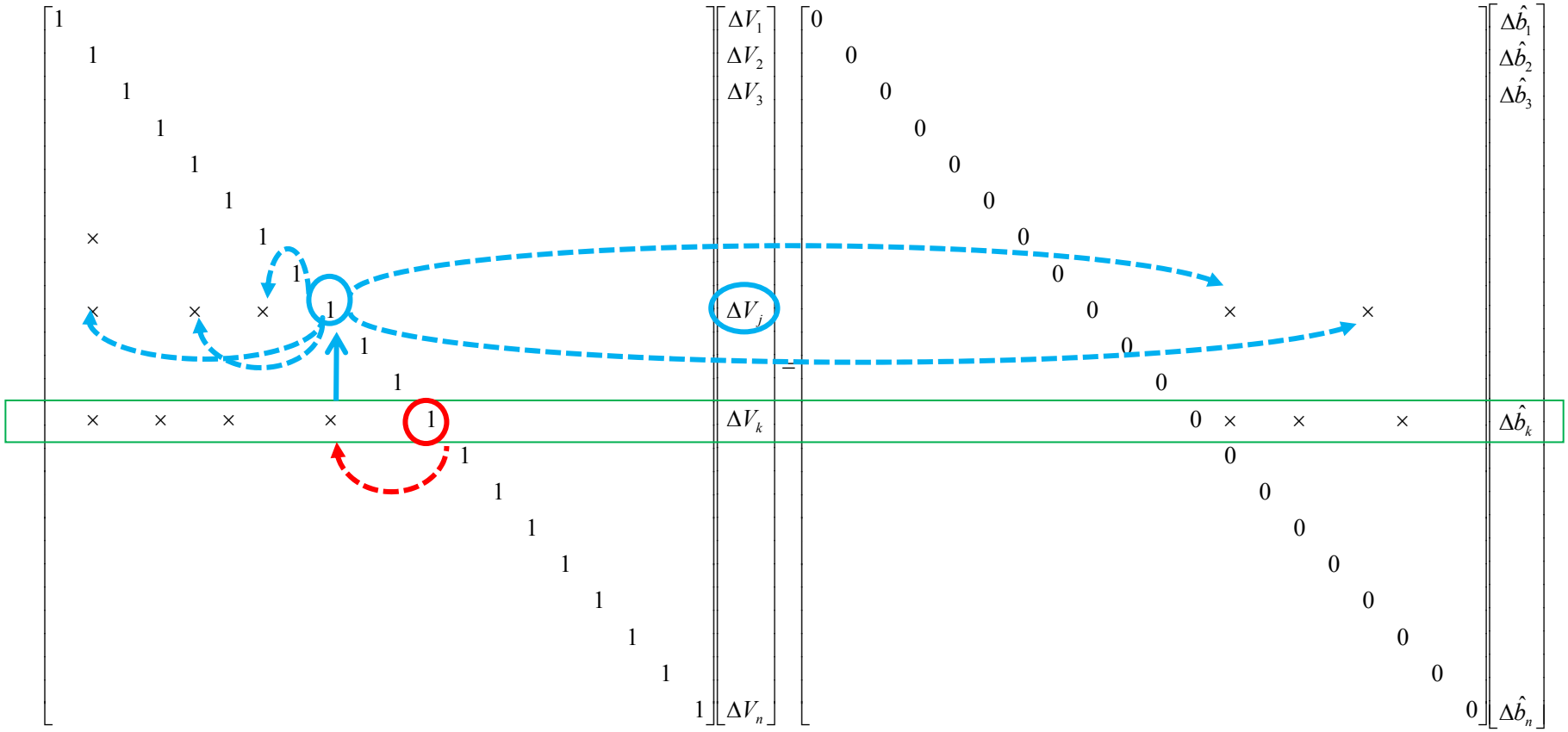
$$(\Delta\hat{\mathbf{b}})_i = \Delta\hat{b}_i = \frac{\Delta\hat{E}_i}{\sum_j g_{i,j}}$$

- Voltage changes: $\Delta\mathbf{V} = (I - Y)^{-1} Z\Delta\hat{\mathbf{b}}$

Incremental Analysis



Incremental Analysis (Cont.)



In Other Words…

$$(I - Y)\Delta\mathbf{V} = Z\Delta\hat{\mathbf{b}}$$

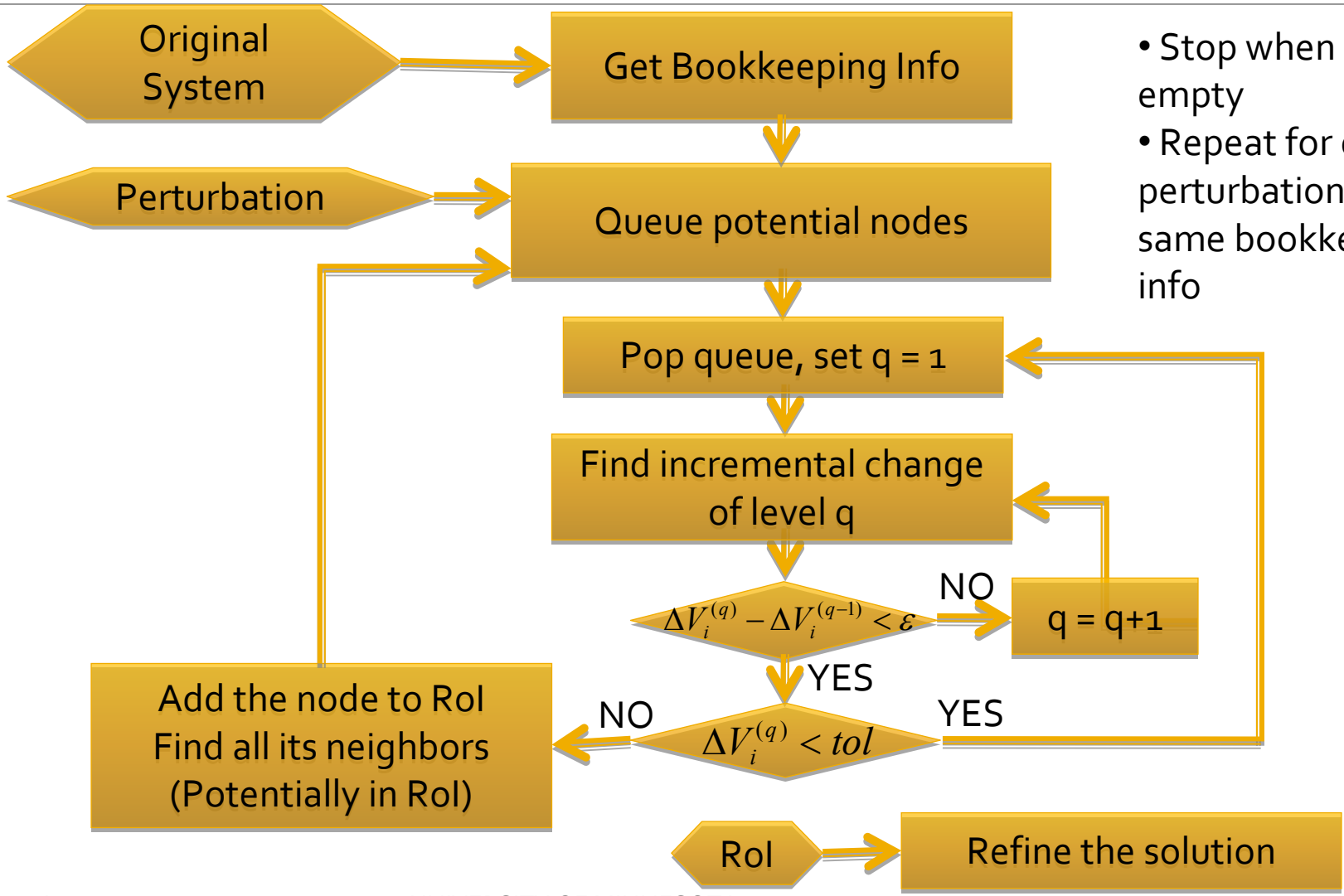
$$\Delta\mathbf{V} = (I - Y)^{-1} Z\Delta\hat{\mathbf{b}}$$

$$\Delta\mathbf{V} = (I - Y)^{-1} Z\Delta\hat{\mathbf{b}} = (1 + Y + Y^2 + \cdots)Z\Delta\hat{\mathbf{b}}$$

- Turns out that all entries of Y are less than 1 in magnitude (since they can be interpreted as probabilities)
- Net result – this is *guaranteed* to converge
- Can be smarter: avoid unnecessary multiplications – different "paths" converge at different rates, can stop when done.

$$\Delta\mathbf{V}^{(q)} = (1 + Y + Y^2 + \cdots + Y^q)Z\Delta\hat{\mathbf{b}}$$

Algorithm



- Stop when queue is empty
- Repeat for each perturbation with the same bookkeeping info

Efficient Computation

- Observation:
  - In computation of incremental change of level q, many of the lower level incremental changes are *insignificant*

$$\Delta V_i^{(q)} - \Delta V_i^{(q-1)} < \varepsilon$$

  - Simply ignore insignificant lower level changes
  - Saves a lot of computation by increasing levels as needed only
  - Error is upper-bounded by accumulated error in ignored neighbors

- Safety factor (s<1):
  - To compensate for truncation error and efficient computation technique
  - Empirically chosen: s = 0.1

$$tol = s \times tol_{given}$$

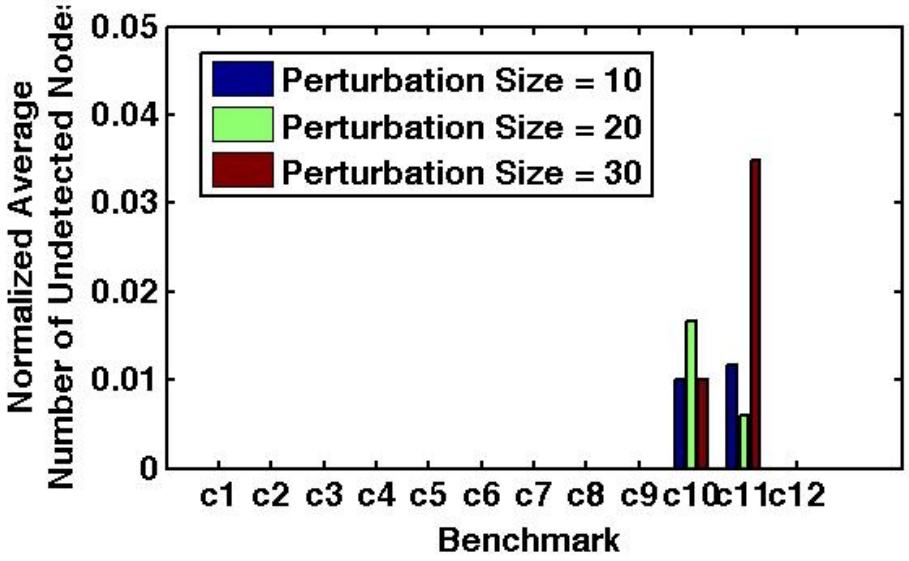
Experimental Results

- Unix machine:
  - 2GHz, 2GB RAM
- V_{DD}=1.2V
- 20 random perturbation scenarios to RHS and LHS
- Perturbation amount uniformly distributed in (0%, 10%)
- Tol = 1%V_{DD}

Benchmarks

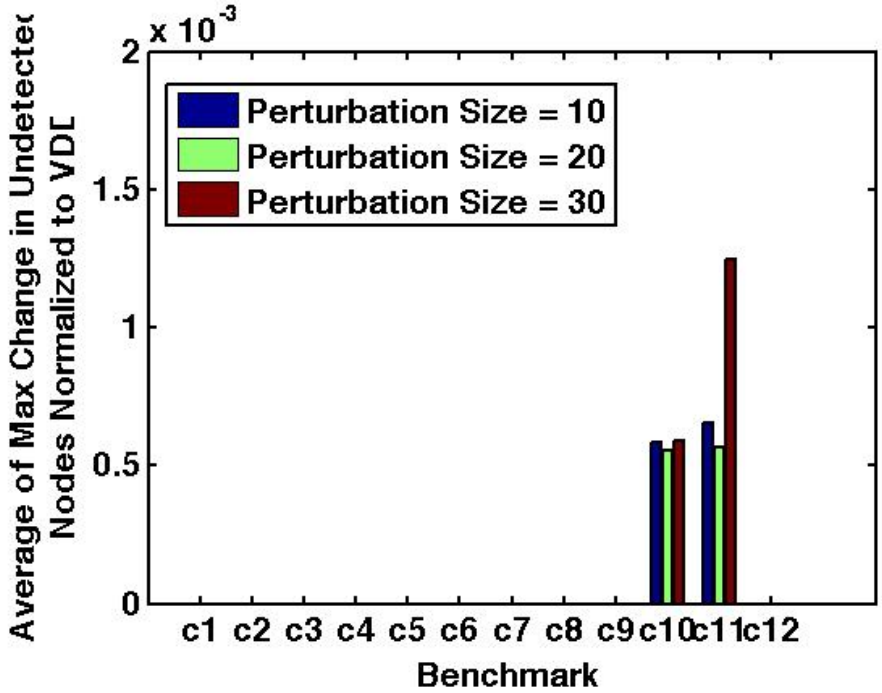
Name	Size	NNZ	Average NNZ per row
c1	16194	98030	6.1
c2	26300	165810	6.3
c3	29551	178345	6.0
c4	34784	203322	5.8
c5	37403	251535	6.7
c6	42409	255747	6.0
c7	58866	352310	6.0
c8	65938	412448	6.3
c9	69351	438985	6.3
c10	93194	593908	6.4
c11	92327	553245	6.0
c12	95303	635727	6.7

Undetected Nodes

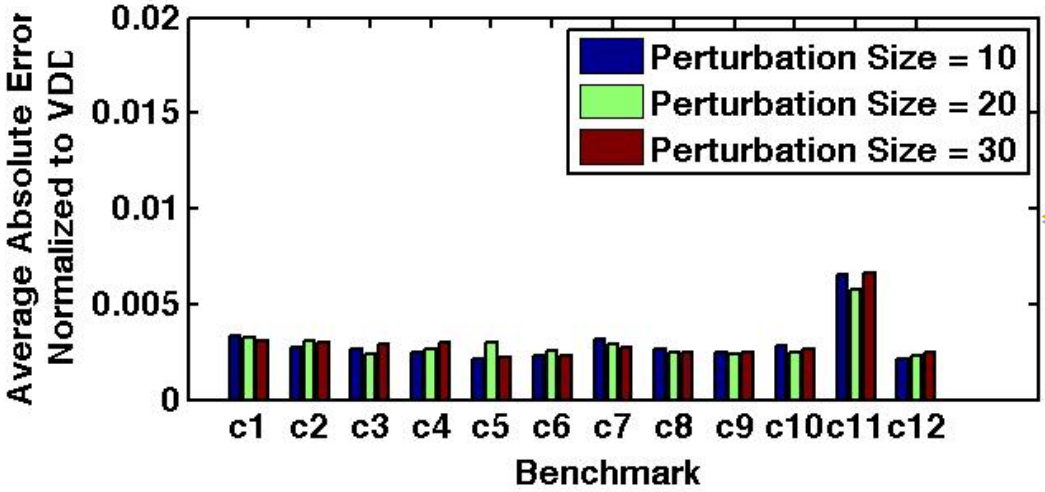


Average number of undetected nodes for different perturbation sizes normalized to exact RoI size averaged over different perturbation scenarios

Max change over all the undetected nodes for different perturbation sizes normalized to V_{DD} averaged over different perturbation scenarios

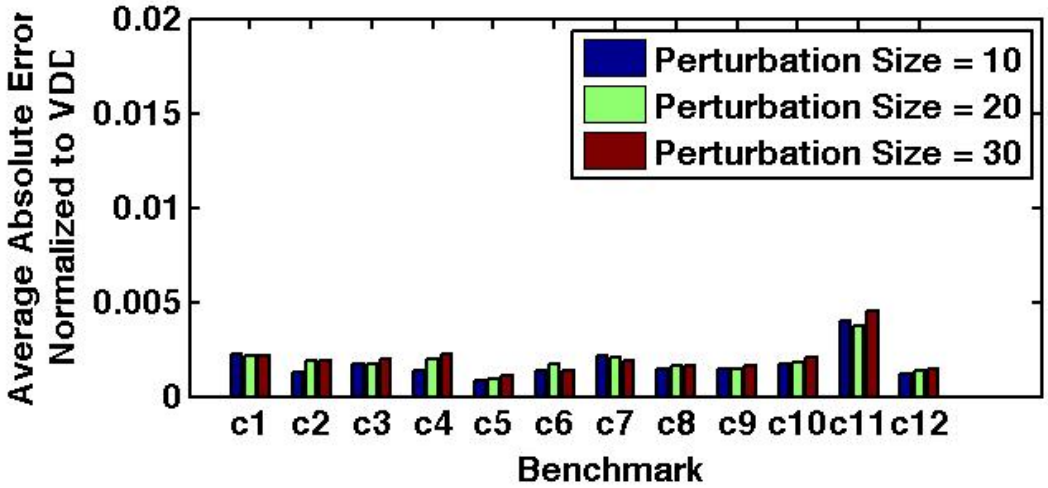


Average Error (RHS Perturbed)

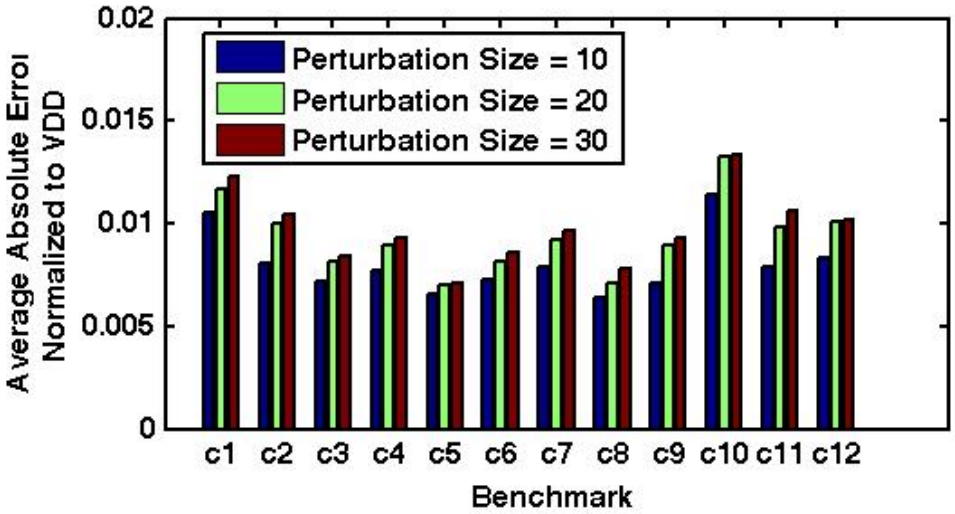


Average of absolute solution error for different perturbation sizes *before* refinement normalized to V_{DD} averaged over different perturbation scenarios

Average of absolute solution error for different perturbation sizes *after* refinement normalized to V_{DD} averaged over different perturbation scenarios

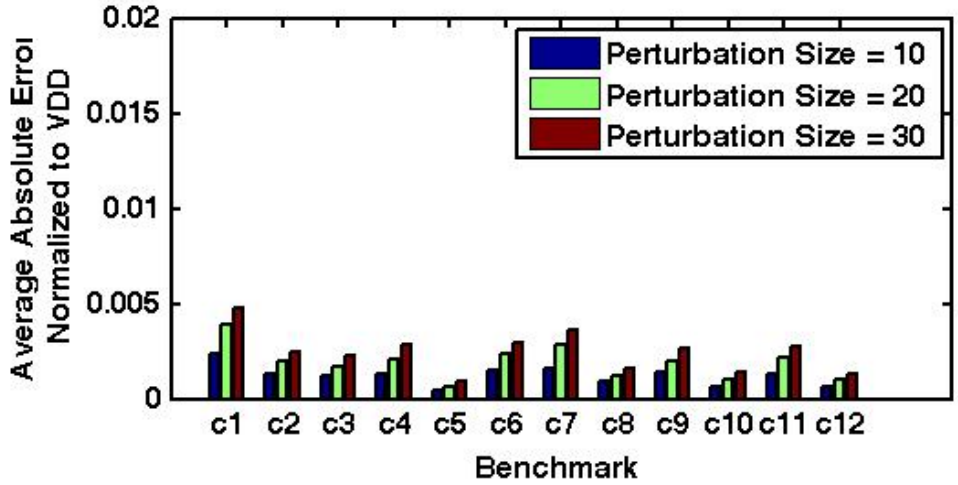


Average Error (LHS Perturbed)



Average of absolute solution error for different perturbation sizes *before* refinement normalized to V_{DD} averaged over different perturbation scenarios

Average of absolute solution error for different perturbation sizes *after* refinement normalized to V_{DD} averaged over different perturbation scenarios



Runtimes (Sec)

	Hybrid solver [3]	Book-keeping	RoI	Refine-ment	10 runs	20 runs
c1	0.29	0.99	0.08	0.01	1.87 (1.6x)	2.75 (2.1x)
c2	0.57	2.04	0.17	0.01	3.85 (1.5x)	5.67 (2.0x)
c3	0.59	2.23	0.19	0.01	4.28 (1.4x)	6.32 (1.9x)
c4	0.7	2.98	0.21	0.01	5.19 (1.4x)	7.41 (1.9x)
c5	0.84	2.86	0.23	0.02	5.27 (1.6x)	7.69 (2.2x)
c6	1.23	4.68	0.31	0.02	7.98 (1.5x)	11.28 (2.2x)
c7	1.88	7.02	0.4	0.02	11.25 (1.7x)	15.48 (2.4x)
c8	2.72	8.29	0.48	0.03	13.31 (2.0x)	18.33 (3.0x)
c9	2.92	9.16	0.44	0.03	13.77 (2.1x)	18.39 (3.2x)
c10	3.83	11.9	0.59	0.04	18.14 (2.1x)	24.38 (3.1x)
c11	3.79	12.97	0.72	0.04	20.53 (1.8x)	28.09 (2.7x)
c12	4.16	13.16	0.68	0.04	20.35 (2.0x)	27.53 (3.0x)
				Average	1.7x	2.5x

Conclusion

- Incremental solution of power grids
  - What-if scenarios
  - Locality of change
- Random walks with bookkeeping
  - Locality property
- RoI found efficiently
- Solution of RoI refined if needed
- Runtime of 20 runs 2.5x less than hybrid solver [Qian et al., ICCAD05]
  - Hybrid solver is 5x - 10x faster than state of the art solver