Power System Probabilistic and Security Analysis Using Commodity High Performance Computing Systems

Tao Cui (Presenter)
Franz Franchetti
Dept. of Electrical and Computer Engineering
Carnegie Mellon University
tao.cui@ieee.org, franzf@ece.cmu.edu

This work was supported by NSF through awards 0931978 and 1116802.

Smart Grid Challenges

The Source: Probabilistic

- **New players: stochastic, varying renewable**
- **Challenges**
  - Non-dispatchable, stochastic in nature, large variances
  - Great impact on the grids: present and future trend

The Grid: Security

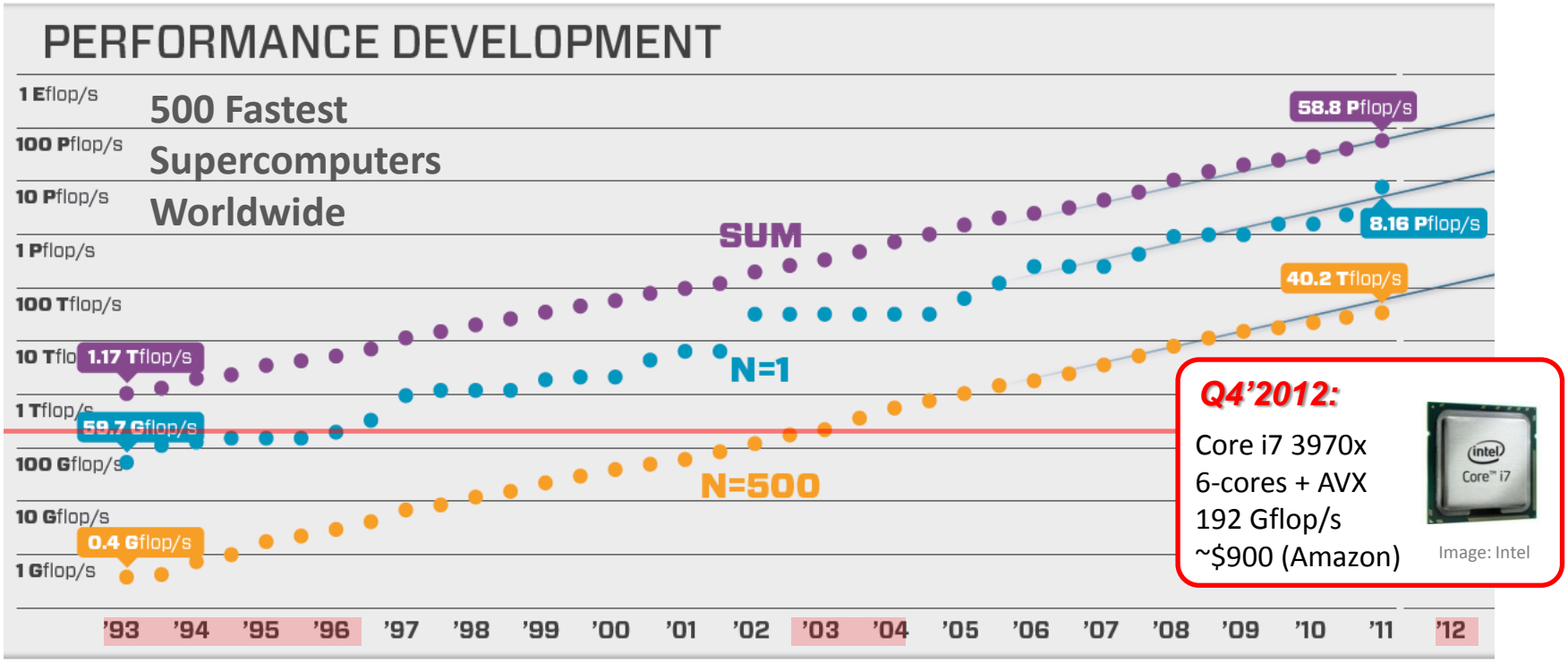
- **Aged, varying conditions & close to limit operations**
- **Challenges**
  - High contingency possibility w/ severe consequences
  - Varying grid conditions require merging offline to online

- **Requirements from regulators (NERC)**
  - Probabilistic analysis approaches from distribution feeder to bulk grid[N1]
  - Extensive contingency analysis for online / real time[N2]
- **Online computation tool for the smart grid challenges**

[N1] NERC Special Report: Accommodating High Levels of Variable Generation, 2009
[N2] NERC Transmission System Standards – Normal and Emergency Conditions

Commodity CPUs: No Free Speedup

- **Top 500 HPC in every substation?**



- Peak performance = fully utilize hardware capability
- Much more complicated architecture ~ lose performance easily

Performance in "flop/s": floating point operations per second, the higher the better

Commodity CPUs: Opportunity & Challenge

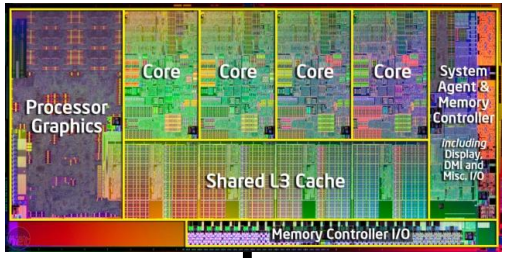
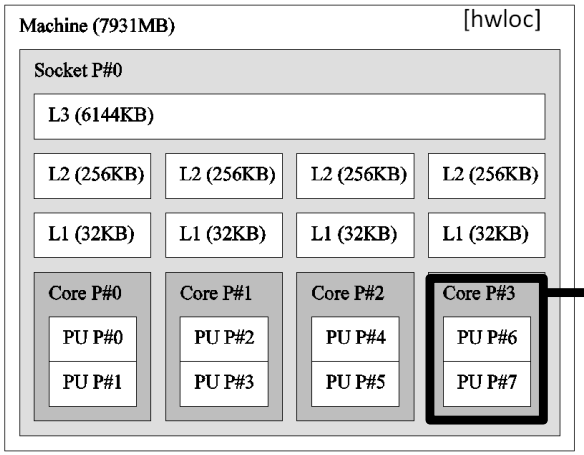


Image: Intel.com

Main Memory

#L1, L2, L3 Cache memory system



*Multi-core

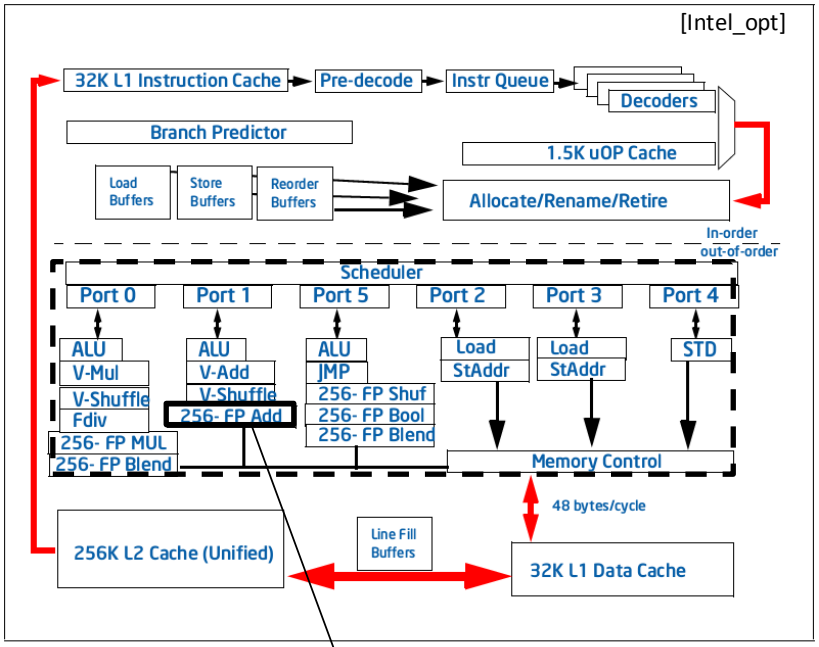
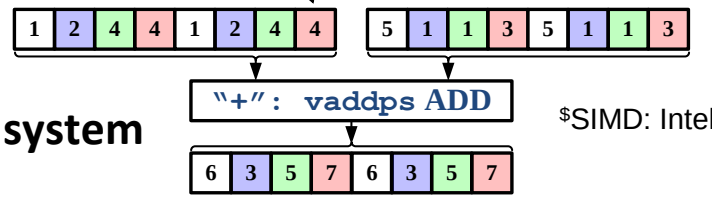


Figure 2-1. Intel microarchitecture code name Sandy Bridge Pipeline Functionality

^Superscalar instruction scheduler



$SIMD: Intel AVX

- **Memory hierarchy: #cache memory system**
- **Multilevel parallelism:**
  - *Multi-thread on multi-core CPU
  - ^Superscalar instruction scheduler: parallel scheduler inside each core
  - $SIMD (Single Instruction Multiple Data): Intel AVX/SSE instruction set

[hwloc] the hardware locality tool from open-mpi.org
[Intel_opt] Intel® 64 and IA-32 Architectures Optimization Reference Manual

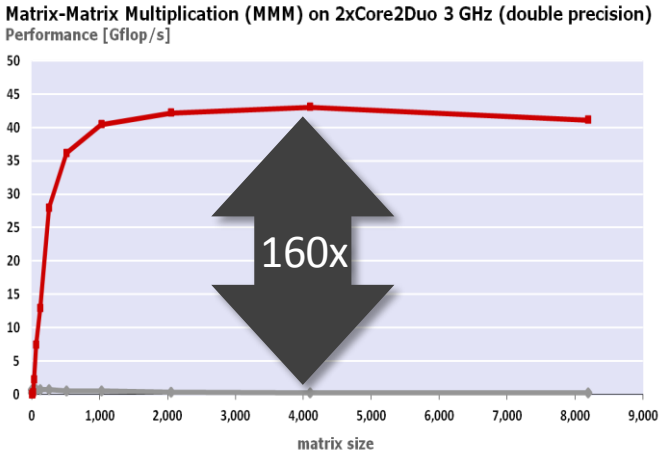
Challenges for Smart Grids Applications

- **Computing**
  - Exponential growth (Moore's law)
  - Peak perf: fully using all computing units
  - Lose performance easily
- **Numerical application development**
  - Example: MMM on Core 2 Duo
  - Gap between SW/HW for power system
    - Performance tuning for HW
    - Best math library? or best algorithm?

High-end desktop (Dell) 2012

?
≈

Supercomputer (Fujitsu NWT) 2000s



Goal: a win-win result

Fully utilize the modern commodity computing systems, make new important and difficult smart grid probabilistic and security analysis problems solvable… in real time.

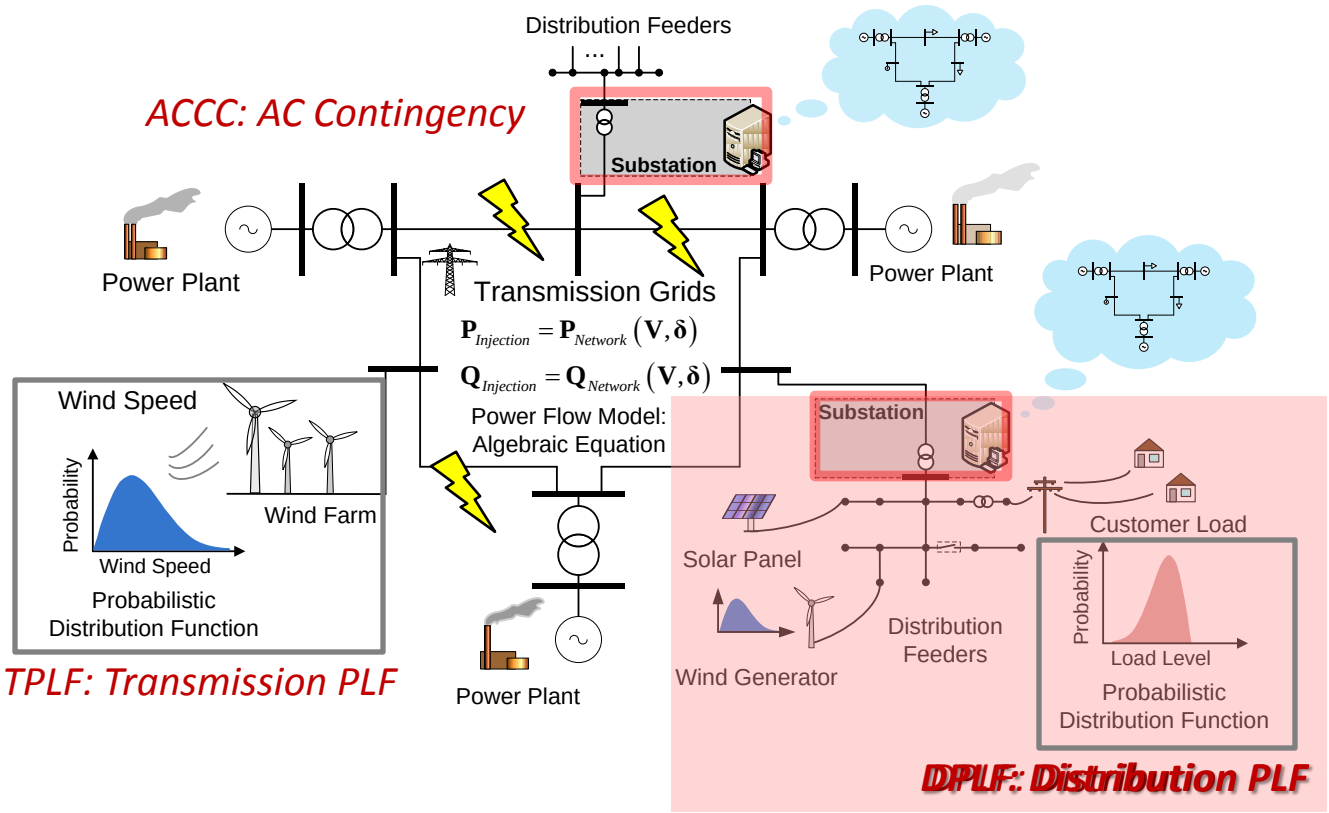
Content

- **Distribution feeder probabilistic load flow (DPLF)**
- **Transmission grid probabilistic load flow (TPLF)**
- **AC contingency calculation (ACCC)**

Goal: a win-win result

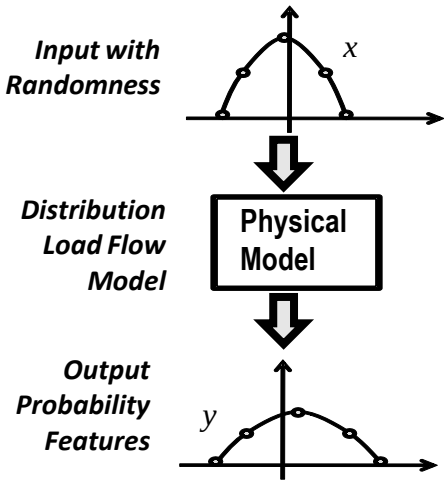
Fully utilize the modern commodity computing systems, make new important and difficult smart grid probabilistic and security analysis problems solvable… in real time.

Outline: HPC in Substation



- **Distribution feeder probabilistic load flow (DPLF)**
- Transmission grid probabilistic load flow (TPLF)
- AC contingency calculation (ACCC)

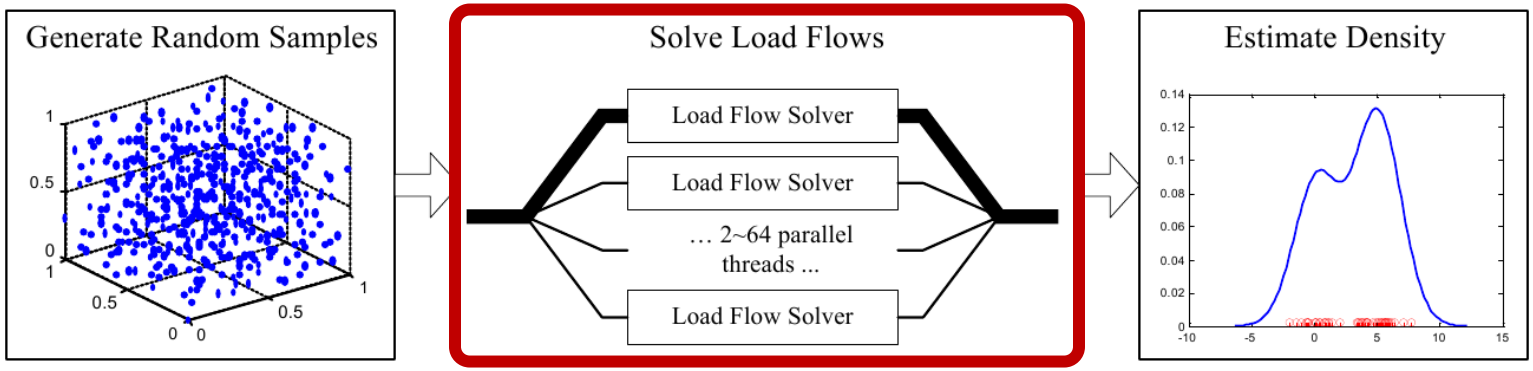
Distribution PLF: Background



- **Fundamental tool for probabilistic analysis**
  - Pseudo-measurements [Ghosh:1997]
  - Wind in distribution systems [Jorgensen:1998] [Caramia:2007]
  - Solar in distribution systems [Ruiz:2012]
  - EV in distribution systems [Li:2012]
  - Monte Carlo simulation as gold standard
- **Unique challenges in distribution systems**
  - Unbalanced, multi-phase, complicated device models
  - Nonlinearity: regulator, control devices
  - Monte Carlo simulation: robust, generally-applicable, UNFEASIBLE online?
- **Our approaches[Cui_PES:2012]**
  - Hardware / software performance engineering perspective
  - Pushing Monte Carlo simulation practical for online operations

[Cui_PES:2012] **Tao Cui** and Franz Franchetti.
"A Multi-Core High Performance Computing Framework for Probabilistic Solutions of Distribution Systems".
IEEE PES-GM 2012

Our Approach



- **Random number generator**
  - Basic uniform Pseudo RNG + transformation for different PDFs
  - Parallel strategy for multi-thread implementation
- **Optimized parallel distribution load flow solver**
  - Code optimizations
  - Highly parallel implementation for Monte Carlo applications
- **Density estimation & visualization**
  - Kernel density estimation

Load Flow Algorithm

- **Distribution system:**
  - Radial , high R/X ratio, varying Z, unbalanced
  - **NOT** suitable for transmission load flow
  - IV Forward / Backward Sweep (FBS)[Zimmer:1995]
    - Implicit Z-matrix, detail model, fewest flops
- **Generalized component model**[Kersting:2006]
  - One terminal node model:

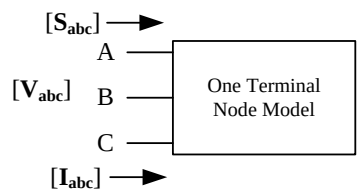
    Constant PQ:

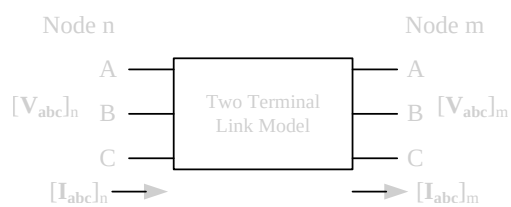
$$[S_{abc}] = [V_{abc}] \cdot [I_{abc}]^*$$

  - Two terminal link model:

Backward: $[I_{abc}]_n = [c] \cdot [V_{abc}]_m + [d] \cdot [I_{abc}]_m$

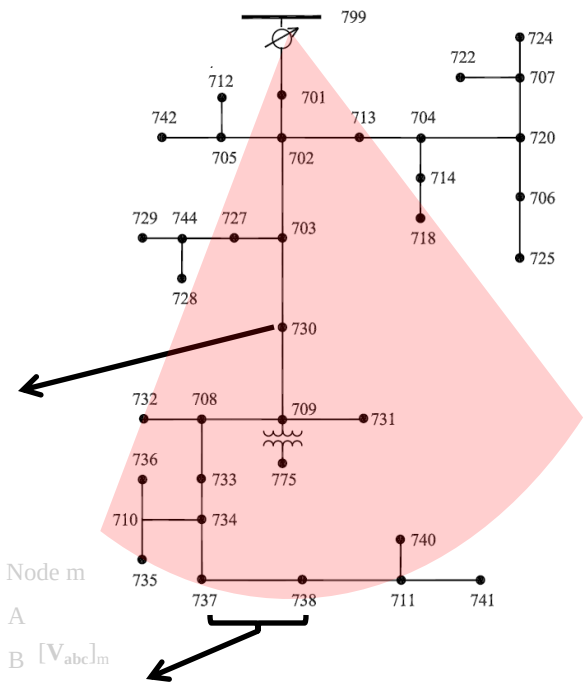
Forward: $[V_{abc}]_m = [A] \cdot [V_{abc}]_n - [B] \cdot [I_{abc}]_m$





IEEE 37 NodeTest Feeder:
Based on an actual feeder in California

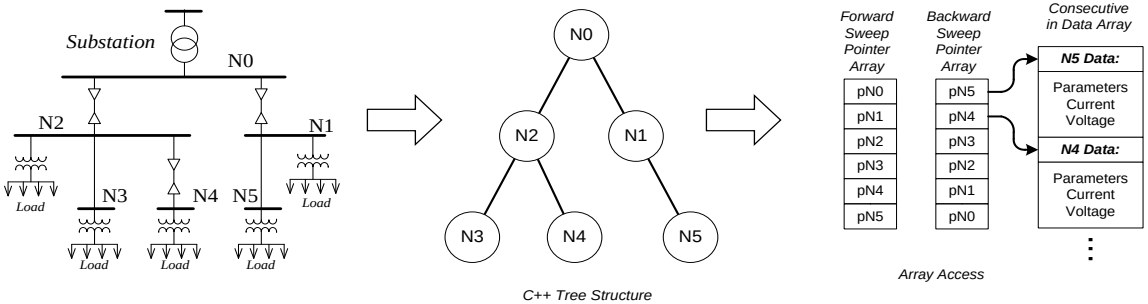
Source: IEEE PES Distribution System Analysis Subcommittee



[Zimmer:1995] R. Zimmerman, "Comprehensive distribution power flow: modeling, formulation, solution algorithms and analysis," Ph.D. dissertation, Cornell University, 1995.
[Kersting:2006] William Kersting, Distribution System Modeling and Analysis.CRC, 2006

Code Optimization

- **Data structure optimization**
  - Transform ***Baseline C++*** to array access, exploit spatial/temporal locality



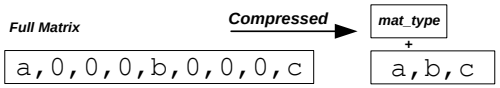
- **Algorithm-level optimization: pattern & unrolling**
  - Pattern based matrix-vector multiplication for Kersting's algorithm

$$[I_{abc}]_n = [c]\cdot[V_{abc}]_m + [d]\cdot[I_{abc}]_m$$

$$[V_{abc}]_m = [A]\cdot[V_{abc}]_n - [B]\cdot[I_{abc}]_m$$

  - Reduce unnecessary operations
  - Unrolling innermost loops
  - Similar to [Belgin:2009]: pattern SpMV

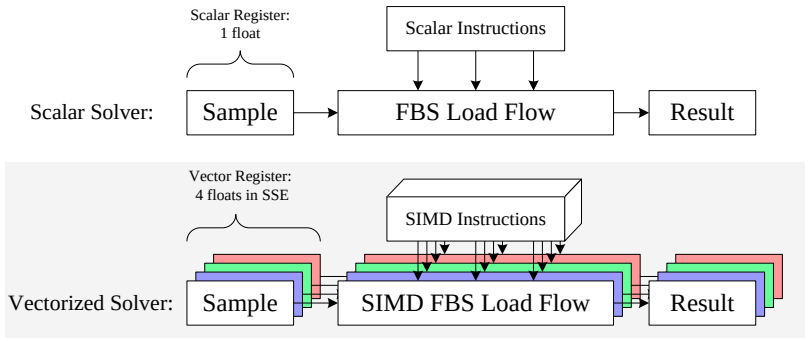
```
switch (mat_type){
    case real_diag_equal_mat:
        output[0] = *constant * input[0];
        ...
        output[5] = *constant * input[5];
        break;
    case imag_diag_equal_mat:
        output[0] = -*constant * input[3];
        output[1] = -*constant * input[4];
        output[2] = -*constant * input[5];
        output[3] = *constant * input[0];
        output[4] = *constant * input[1];
        output[5] = *constant * input[2];
        break;
        ...
}
```



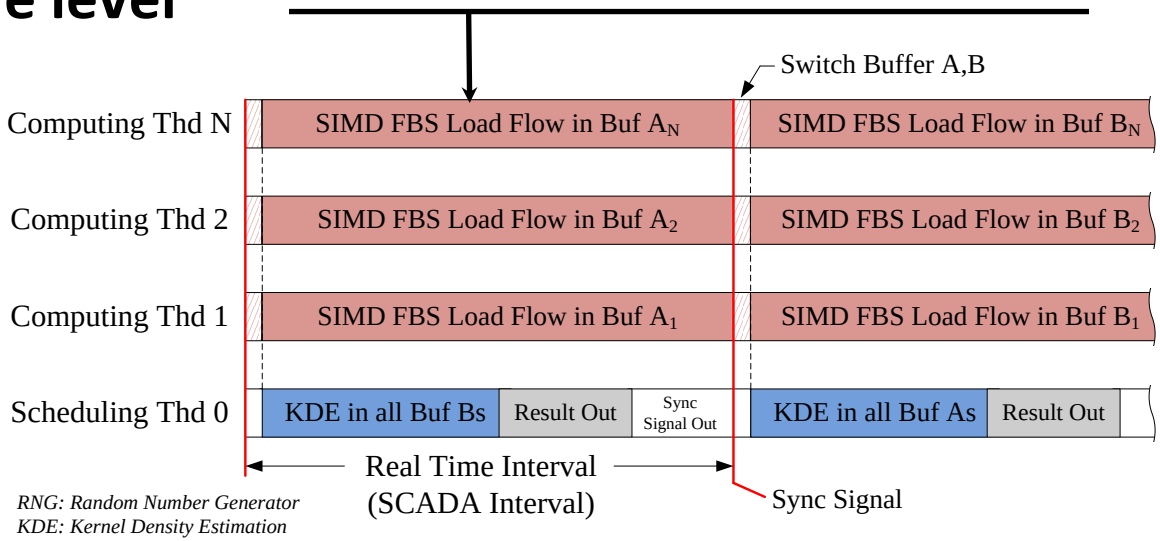
[Belgin:2009]. M. Belgin, G. Back, and C. J. Ribbens,
"Pattern-based sparse matrix representation for memory-efficient SMVM kernels,"
in Proceedings of the 23rd international conference on Supercomputing, ser. ICS '09. New York, NY, USA: ACM, 2009, pp. 100–109.

Parallel MCS with Real Time Considerations

- **Data level (SIMD)**

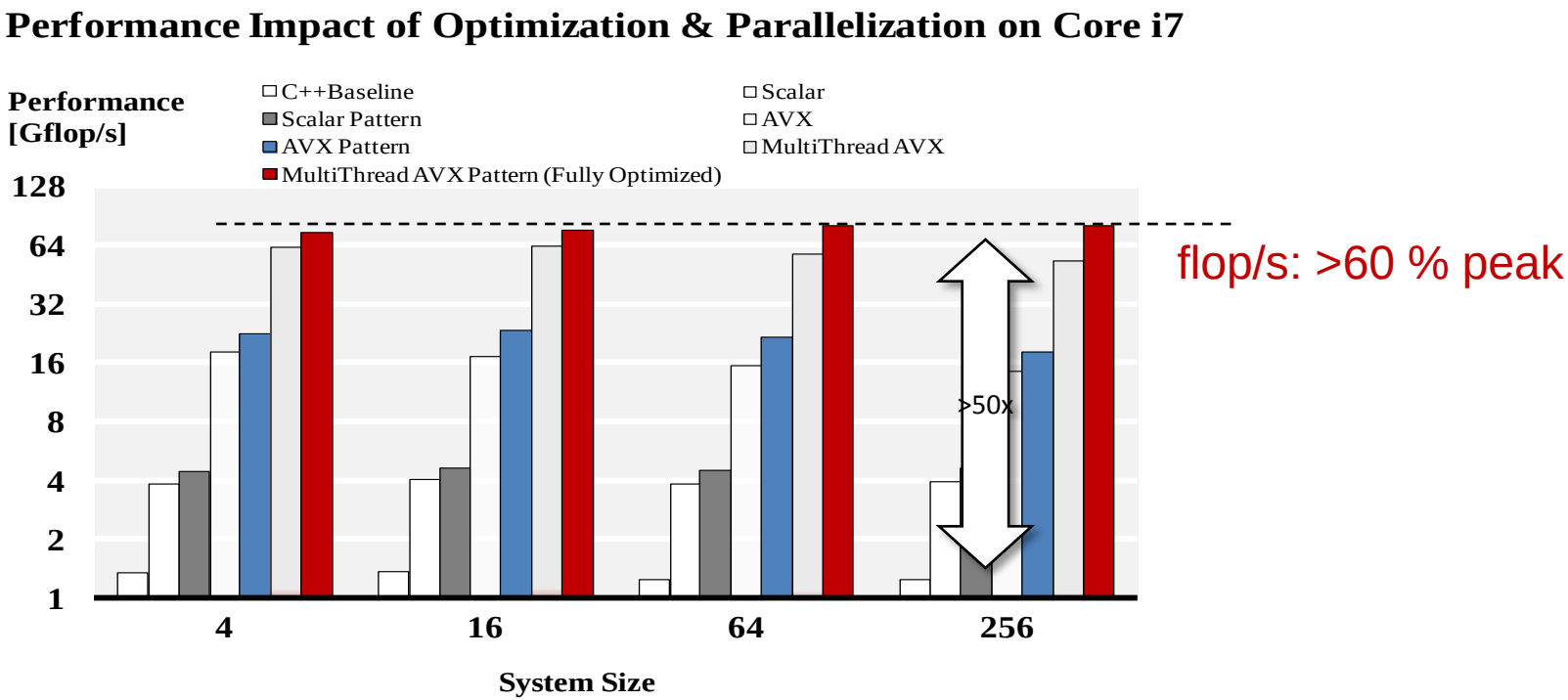


- **Task/core level**



- Realtime MCS scheduler: task decomposition, double buffer, auto-balancing)
- Fully utilize computation power of multi-core CPUs

Details: Performance Gains

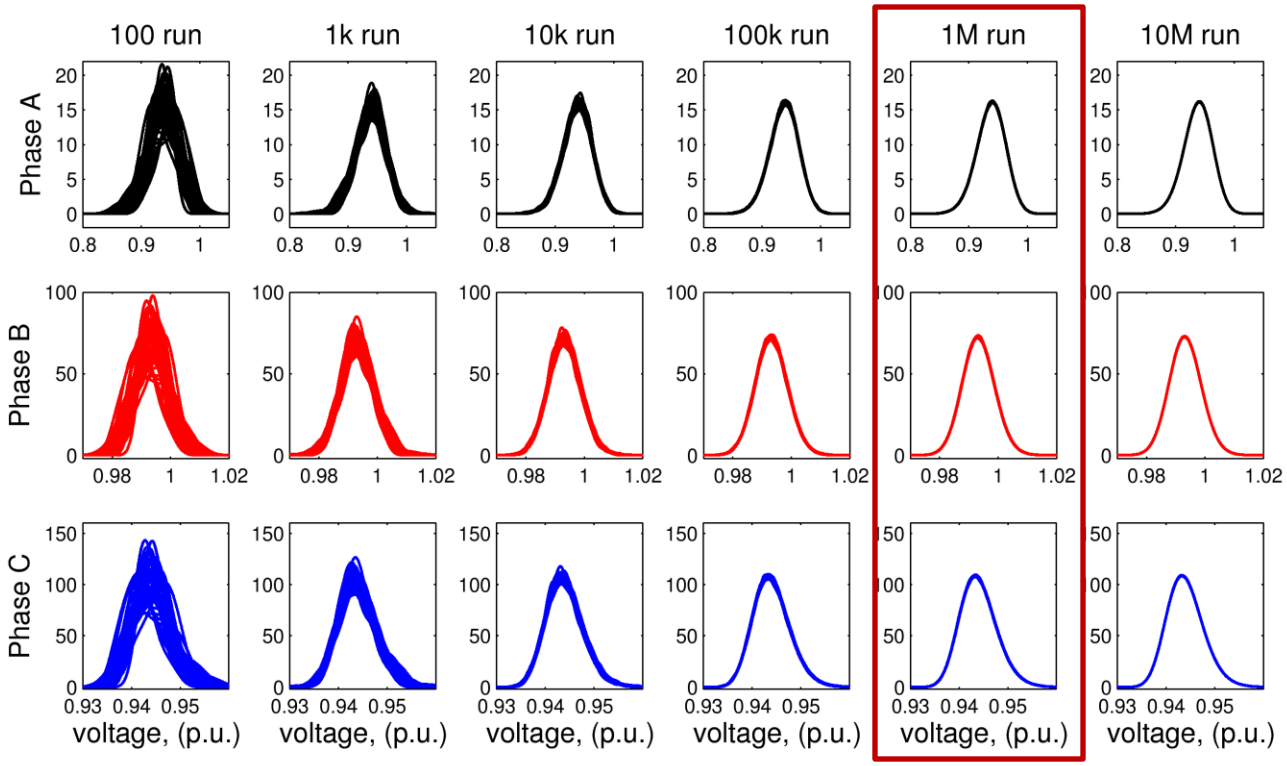


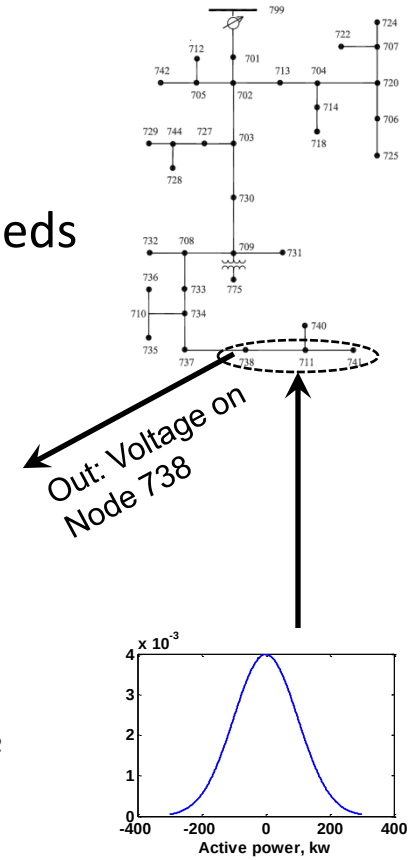
Problem Size (IEEE Test Feeders)	Approx. flops	Approx. Time / Core2 Extreme	Approx. Time / Core i7	Baseline. C++ ICC –o3 (~300x faster then pure Matlab scripts)	Comments
IEEE37: one iteration	12 K	~ 0.3 us	~ 0.3 us		0.01 kVA mismatch
IEEE37: one load flow (5 Iter)	60 K	~ 1.5 us	~ 1.5 us		
IEEE37: 1 million load flow	60 G	~ < 2 s	~ < 1 s	~ 60 s (>5 hrs Matlab)	SCADA Interval: 4 seconds
IEEE123: 1 million load flow	200 G	~ < 10 s	~ < 3.5 s	~ 200 s (>15 hrs Matlab)	

- **Core i7 2670QM, quad-core, 2.2GHZ, AVX, Machine Peak: 140 Gflop/s**
- **Millions of load flow cases solved within SCADA interval**

Accuracy

- **Convergence of Monte Carlo**
  - Crude MCS: MCG59+ICDF, 50 trials with “time(NULL)” seeds



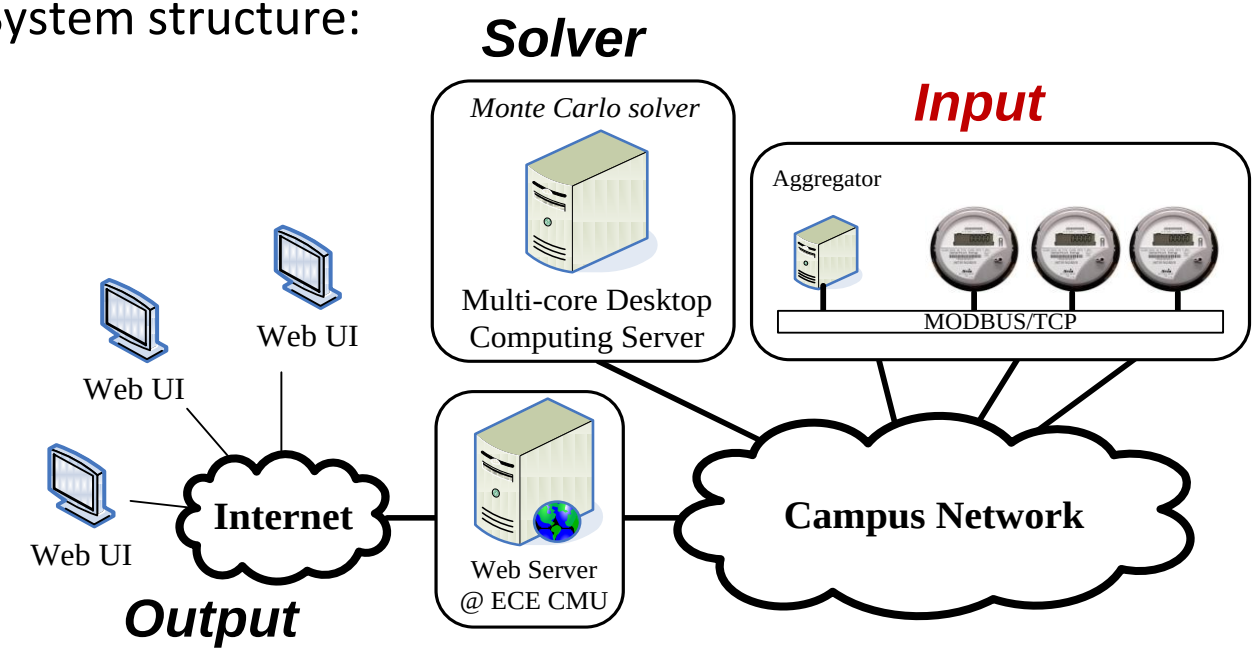


In: Active Power P~ u=0,std=100kw on Phase A of Node 738,711,741

  - Converged MCS solution to PLF within SCADA Interval
    - Accurate, generally-applicable, real time

Applications

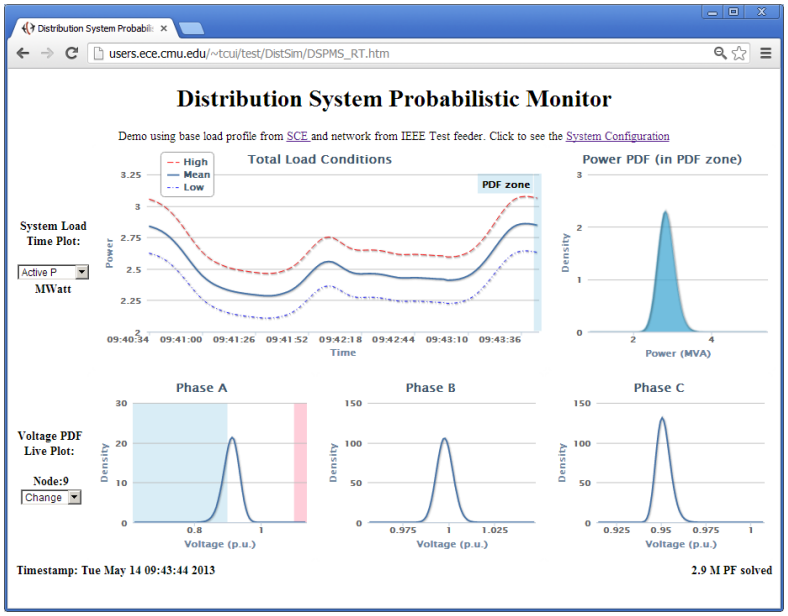
- **Distribution System Probabilistic Monitoring System (DSPMS)**
  - System structure:

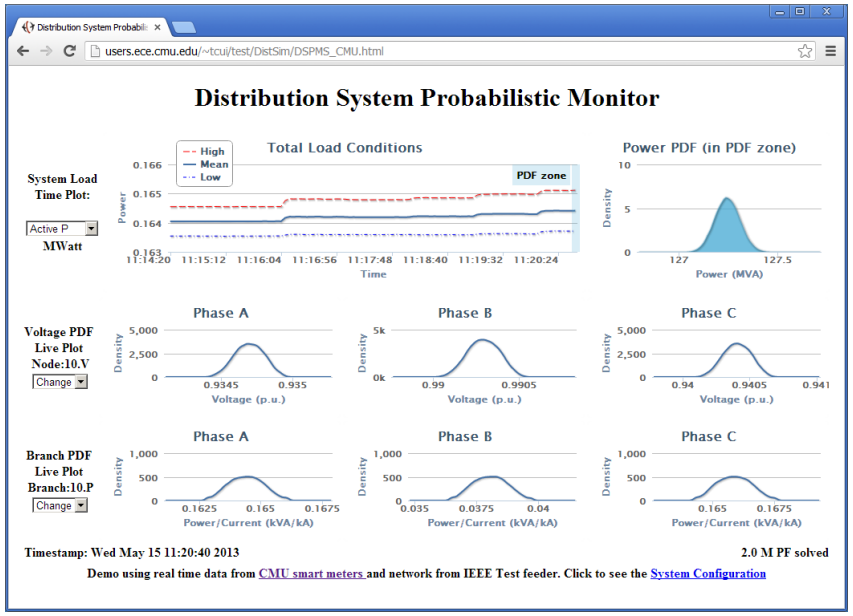


  - Input: smart meters in campus building (MODBUS/TCP)
  - Solver: MCS solver on multi-core desktop server (code optimization)
  - Output: dynamic web interface via ECE web server (TCP/IP, JavaScript)

User Interfaces

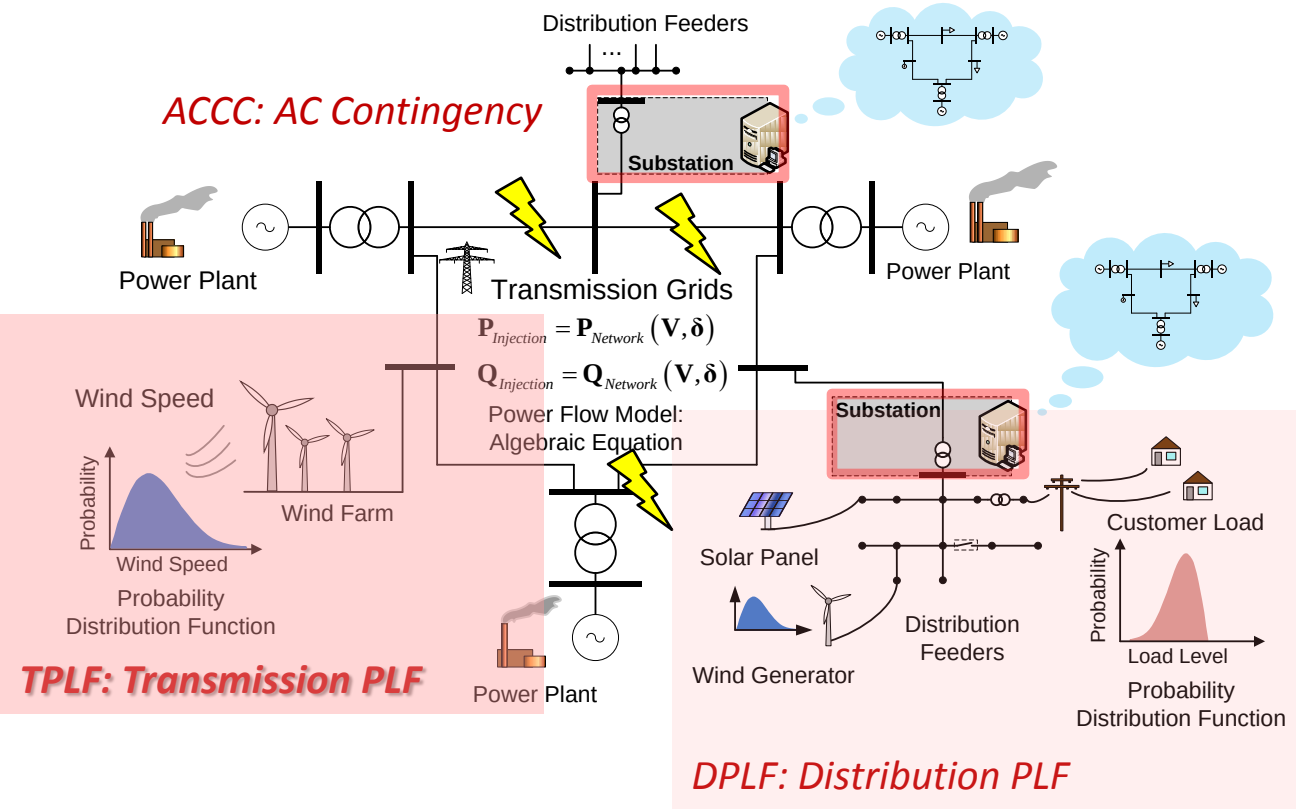
- **Left: application 1 -- online forecasting (e.g. Gaussian error)**
- **Right: application 2 -- online impact assessment**





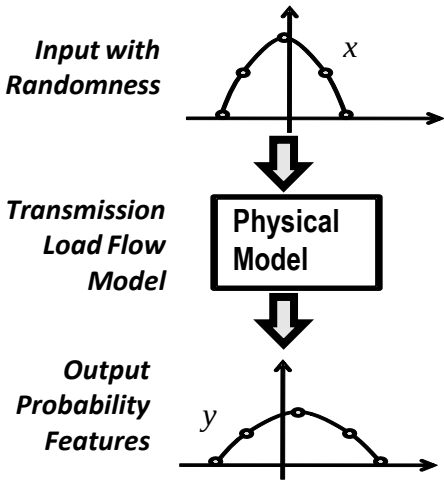
  - Live update, every 4 seconds (SCADA interval)
- **Novel SCADA extension with full real time probabilistic analysis**

Outline: HPC in Substation



- Distribution feeder probabilistic load flow (DPLF)
- **Transmission grid probabilistic load flow (TPLF)**
- AC contingency calculation (ACCC)

Transmission PLF: Background



- **Related work on improving PLF solutions**
  - Simplified power system model:
    - linearized, multi-linearized[Allan:1981][Silva:1990]
  - Simplified probabilistic model:
    - Interval[Wang:1992], point[Su:2005], cumulants[Zhang:2004]
  - Improving Monte Carlo
    - Variance reduction[Zhang:2009], deterministic sample[Liao:2007]
  - Monte Carlo simulation as accuracy references

- **Opportunities from computing hardware perspective**
  - Analytical method: not robust, not generally-applicable…
  - Fully take advantages of the computing system
  - Can we push the gold standard (MCS) into real time?

Fast Decoupled Power Flow

- **Fast, fewest floating point operations**[Stott:1974][Monticelli:1990]

$$\begin{cases} \Delta \mathbf{V}^{(k)} = -\mathbf{B}''^{-1} \Delta \mathbf{Q}(\theta^{(k)}, \mathbf{V}^{(k)}) / \mathbf{V}^{(k)} \\ \mathbf{V}^{(k+1)} = \mathbf{V}^{(k)} + \Delta \mathbf{V}^{(k)} \end{cases}$$

$$\begin{cases} \Delta \theta^{(k)} = -\mathbf{B}'^{-1} \Delta \mathbf{P}(\theta^{(k)}, \mathbf{V}^{(k+1)}) / \mathbf{V}^{(k+1)} \\ \theta^{(k+1)} = \theta^{(k)} + \Delta \theta^{(k)} \end{cases}$$

Linear solver

Mismatch

$$\Delta P_i = P_i^{INJ} - V_i \sum_{j=1}^{n} V_j \left(G_{ij} \cos \theta_{ij} + B_{ij} \sin \theta_{ij} \right)$$

$$\Delta Q_i = Q_i^{INJ} - V_i \sum_{j=1}^{n} V_j \left(G_{ij} \sin \theta_{ij} - B_{ij} \cos \theta_{ij} \right)$$

[Stott:1974] Stott, B.
Review of load-flow calculation methods
Proceedings of the IEEE, 1974, 62, 916 - 929

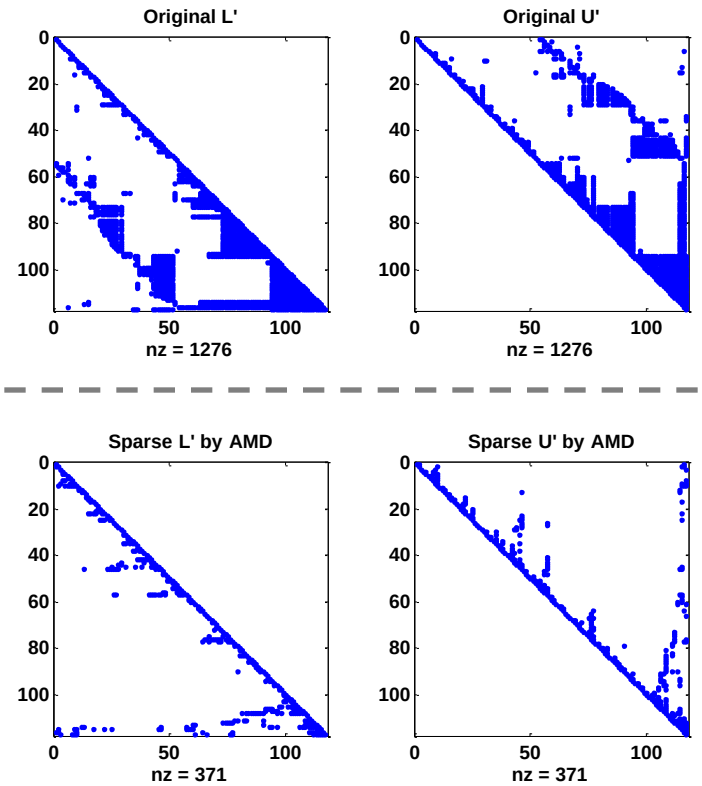
[Monticelli:1990] Monticelli, A.; Garcia, A.; Saavedra, O.R.,
"Fast decoupled load flow: hypothesis, derivations, and testing,"
Power Systems, IEEE Transactions on , vol.5, no.4, pp.1425,1431, Nov 1990

Algorithm Optimization: Sparsity

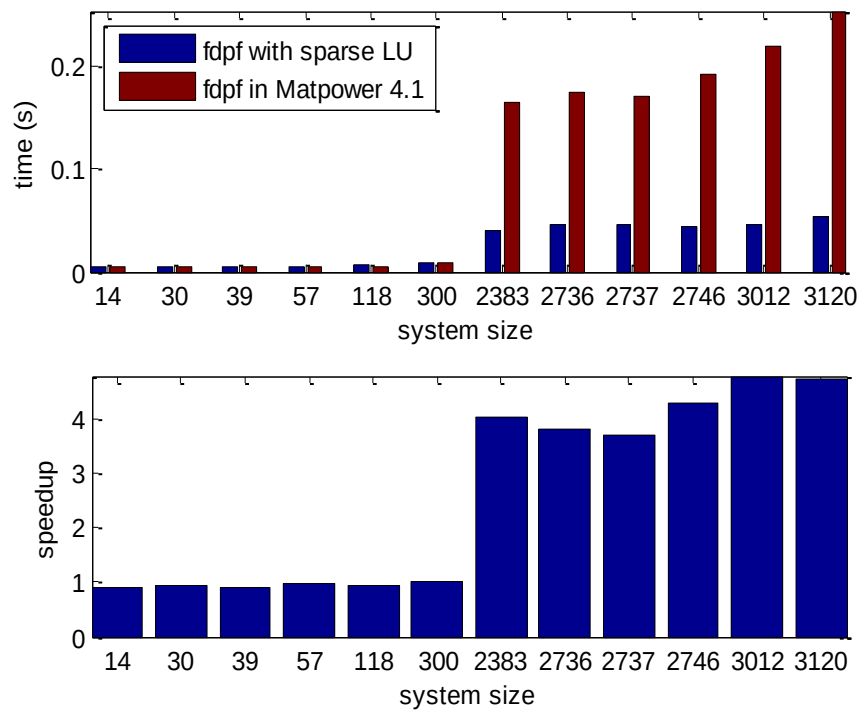
- **Linear solver: exploit sparsity -- Baseline**
  - Sparse LU factors by using AMD[Davis:2004]
  - Speedup Matpower 4.1's FDPF module

Original

Improved



LU Factors of B' of IEEE 118 system



Speed Comparison

[Davis:2004] P. Amestoy, T. Davis, and I. Duff,
"Algorithm 837: AMD, an approximate minimum degree ordering algorithm,"
ACM Transactions on Mathematical Software (TOMS), vol. 30, no. 3, pp. 381–388, 2004

Data Structure/Math Func Optimization

- **Linear solver**
  - New mixed compress column storage (CCS) format

Original CCS:

col ptr	c1	c2	c3			
row idx	r1	r2	r3	r4	r5	r6
value:	v1	v2	v3	v4	v5	v6

Mixed CCS:

col ptr	c1	c2	c3									
mixed:	v1	r1	v2	r2	v3	r3	v4	r4	v5	r5	v6	r6

- **Mismatch**

Original θ array:

θ_1	θ_2	...	θ_N

Mixed θ array:

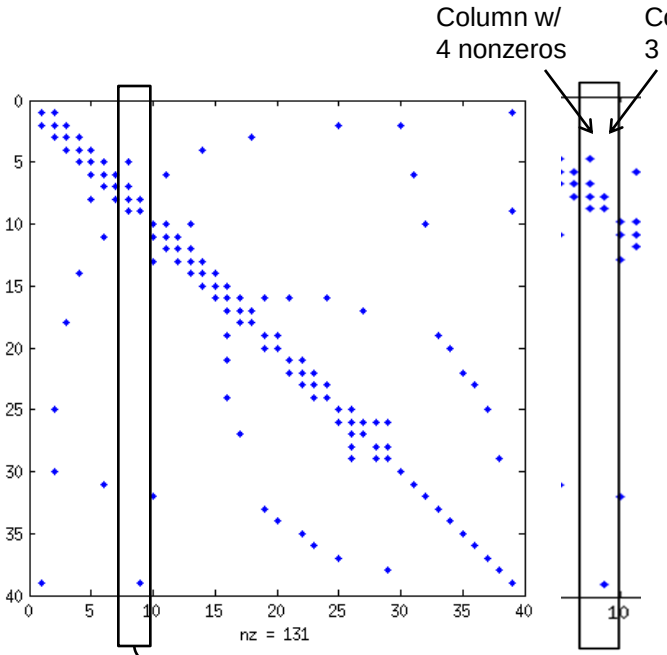
θ_1	$\sin\theta_1$	$\cos\theta_1$	θ_2	$\sin\theta_2$	$\cos\theta_2$	...	θ_N	$\sin\theta_N$	$\cos\theta_N$

  - **`sin/cos`**: >100 cycles v.s. **`mul/add`**: 1 cycle
  - Efficient utilization: reduce **`sin cos`** operations e.g. **`sin(a-b)`**
  - Efficient implementation: alternative version for SIMD datatype[Shibata:2010]

[Shibata:2010] Naoki Shibata.
"Efficient evaluation methods of elementary functions suitable for SIMD Computation".
Computer Science-Research and Development, 25(1-2):25–32, 2010.

Unrolling for Sparse Kernels

- **General sparse kernel: traversal of the CCS sparse matrix**
  - Original: branch on every non-zero element
  - Unrolled: pre-generate bigger code blocks for multiple columns
  - Bigger code blocks -> better instruction level parallelism



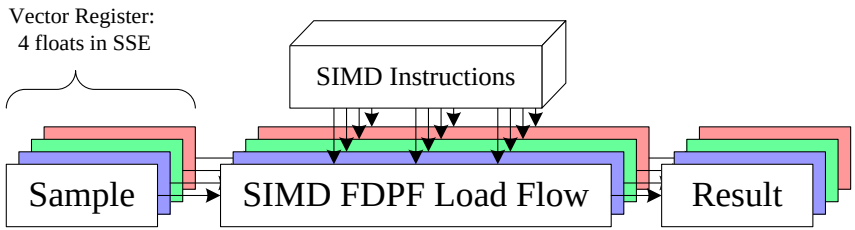

```
for (col = 0; col < n; col++){
    for (row = col_ptr[col]; row < col_ptr[col+1]; row++){
        ...// access & compute on nonzero at (col, row)
    }
}
```

Unrolling

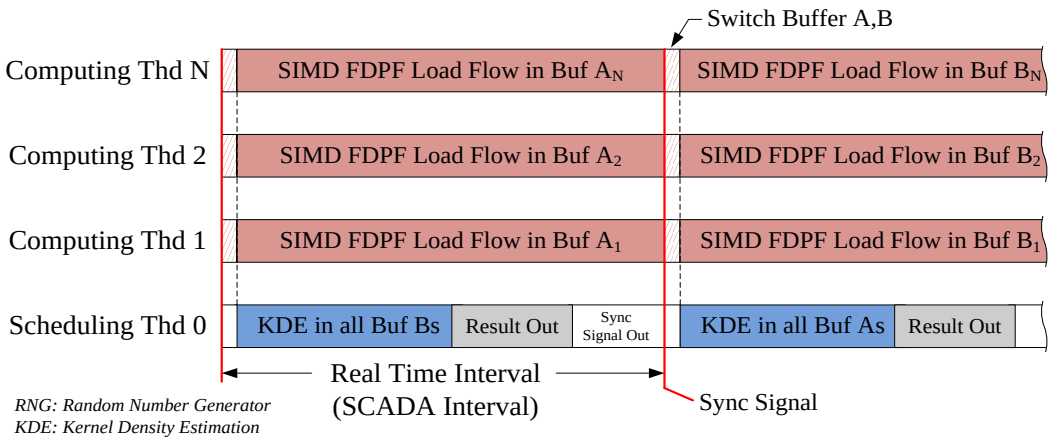
```
do{
    switch (case_pattern for 2 consecutive columns){
        case
        case pattern(4,3): {
            ...// access & compute on nonzero at (i, 1)
            ...// access & compute on nonzero at (i, 2)
            ...// access & compute on nonzero at (i, 3)
            ...// access & compute on nonzero at (i, 4)
            ...// access & compute on nonzero at (i+1, 1)
            ...// access & compute on nonzero at (i+1, 2)
            ...// access & compute on nonzero at (i+1, 3)
        break;}
        case ...
    }
}while(!all columns visited)
```

Multilevel Parallelism

- **SIMD**

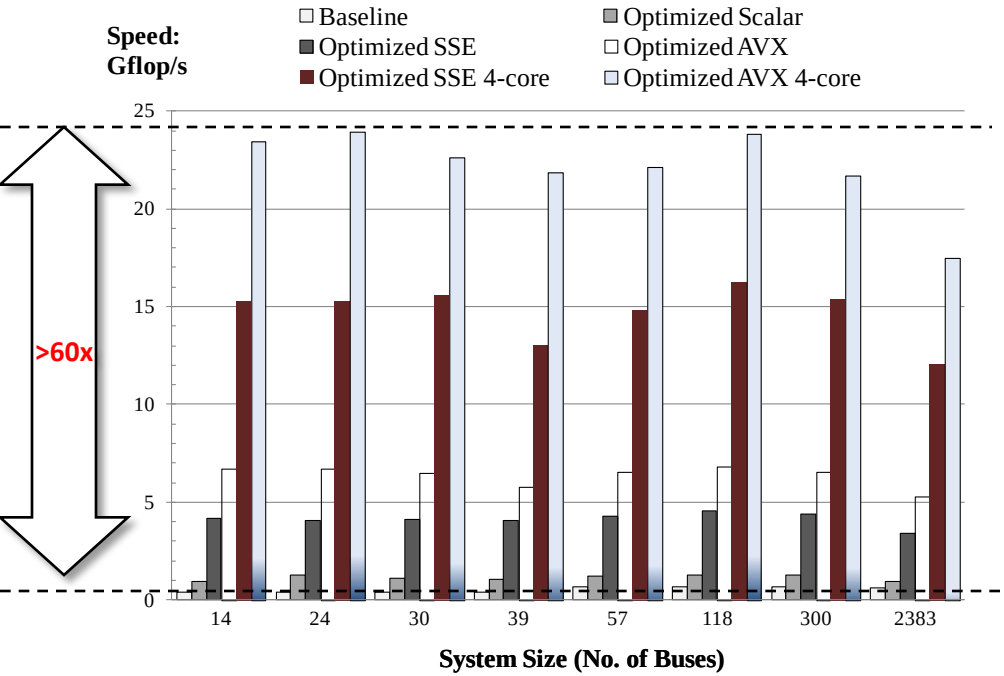


- **Multithread on multicore**



Result: Performance Breakdown

- **Optimizing iteration speed (on Core i7 2670QM)**



Approx. Speed: Load Flow Cases Solved per Second on Core i7

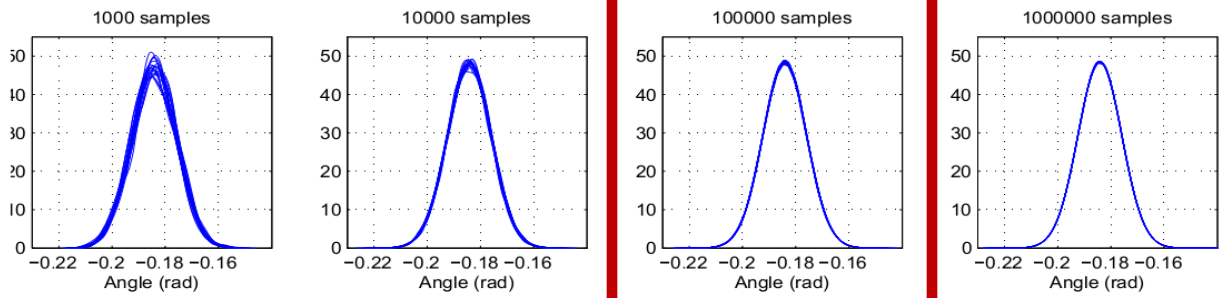
Test Cases		Approx. Speed [cases/s]	
System Size.	Flops/Iteration	Baseline[1]	AVX 4-Core
14	1,034	39,000	2,270,000
24	1,788	23,000	1,340,000
30	2,242	19,000	1,010,000
39	2,715	23,000	805,000
57	4,467	15,000	495,000
118	9,130	7,000	261,000
300	23,370	3,000	92,900
2,383	175,365	340	9,960

1. **Baseline** is compiler optimized (Intel C Compiler & O3).

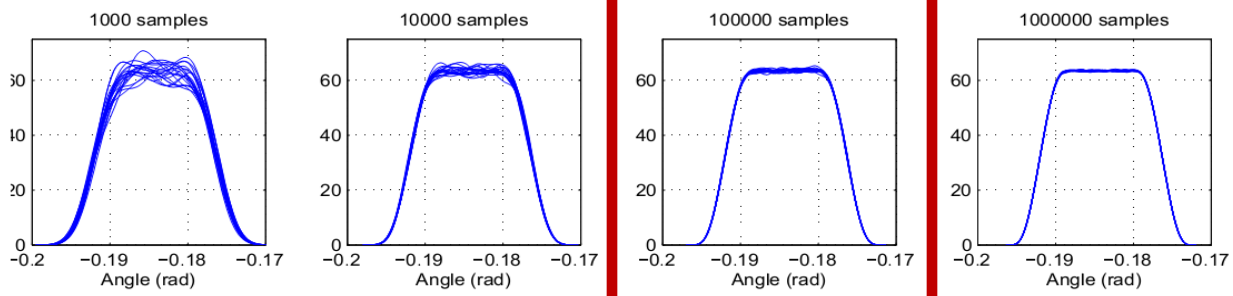
- **Upto 60x speedup comparing to scalar baseline (CXSparse)**
- **Converged MCS PLF solution every second**

Core i7 2670QM, quad-core, 2.2GHZ, AVX, 6MB Cache, Gaming Laptop

TPLF Result Example



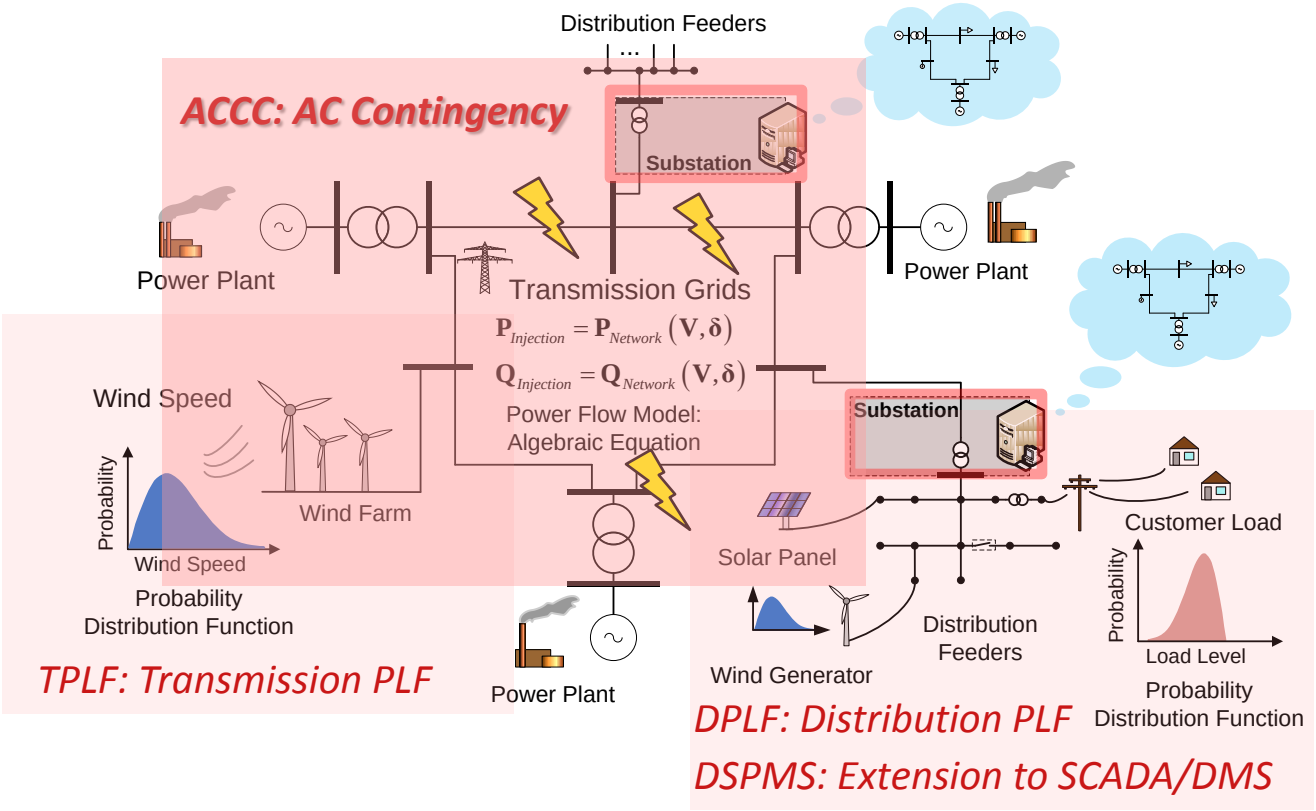
(a) Phase angle of Bus 59. Normal distributed P injections



(b) Phase angle of Bus 59. Uniform distributed P injections

- **Example on IEEE118 system**
  - Normal(0,10MW) and Uniform(-10MW,10MW) random active power plugged into first three highest loading buses (Bus 59: 277MW, Bus 90: 163MW, Bus 116: 184MW)
- **Converged MCS PLF solution every second**

Outline: HPC in Substation



- Distribution feeder probabilistic load flow (DPLF)
- Transmission grid probabilistic load flow (TPLF)
- **AC contingency calculation (ACCC)**

Related Work

- **Active joint field between power system & computing**
  - PNNL's work on supercomputer approach
    - Counter based load balancing[PNL_Massive:2009]
    - Hybrid: Cray-XMT (selection) + cluster (ACCC) [PNL_Hyprid:2009]
  - Multicore version ACCC in PTI PSS/E 33 & GE-PSLF SSTOOLS[PSSE][PSLF]
  - Drexel's GPU and FPGA approaches
    - Iterative linear solver on GPU, DC power flow[Gopal_GPU:2010]
    - FPGA accelerated for sparse linear solver[Johnson:2008]

Task Level

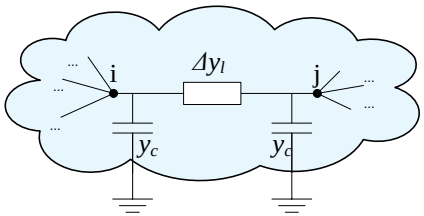
Hardware

- **Unique opportunities on commodity CPUs**
  - Accelerate AC Contingency Calculation by fully utilizing CPU hardware?

Our Approach

- **What we have: optimized FDPF computing kernel from TPLF**
  - Efficient linear solver, mismatch computation /w multilevel parallelism
- **What needed: deal with contingency, topology changes**
  - Goal: fine grain data parallelism for network of different outages?
- **Date level: on single core, SIMD + compensation**
  - Preserves most parts of computations: **compensation method**
  - Fully utilize the computing power of modern CPU core
- **Task level: on multi-core, thread pool**
  - Convergence at different steps
  - Dynamically balancing workload among cores

Topology Change



$$\begin{array}{c} \begin{bmatrix} \Delta y_l + y_c & -\Delta y_l \\ -\Delta y_l & \Delta y_l + y_c \end{bmatrix}_{n\times n} \begin{array}{c} \mathrm{i} \\ \mathrm{j} \end{array} \\ \begin{array}{cc} \mathrm{i} & \mathrm{j} \end{array} \end{array}$$

- **Handle topology change**
  - Contingency analysis with line trip
  - May change the problem formulation
- **Branch outage representation in admittance matrix:**
  - Given admittance matrix Y, branch i j change Δy

$$M = \begin{bmatrix} 0, ..., \underset{i}{1}, 0...0, 0, ..., 0 \\ 0, ..., 0, 0...0, \underset{j}{1}, ..., 0 \end{bmatrix}^T \quad \Delta y = -\begin{bmatrix} y_{ij} + b_{ij} & -y_{ij} \\ -y_{ij} & y_{ij} + b_{ij} \end{bmatrix} \quad \tilde{Y} = Y + M\Delta y M^T$$

- **In FDPF, B' and B" change accordingly**
- **In general, two approaches to handle topology changes**
  - LU re-factorization
  - Compensation method based on Woodbury Identity[Stott_Compensation:1983]

[Stott_Compensation:1983] Alsac, O.; Stott, B.; Tinney, W.F.; ,
"Sparsity-Oriented Compensation Methods for Modified Network Solutions,"
Power Apparatus and Systems, IEEE Transactions on , vol.PAS-102, no.5, pp.1050-1060, May 1983

Solution: Compensation

- **Handle topology change based on compensation**
  - One possible compensation method:

    Given: $\tilde{A} = A + M\Delta y M^T$

    Goal: Solve $\tilde{A}\tilde{x} = b$

    Compute using the lemma:

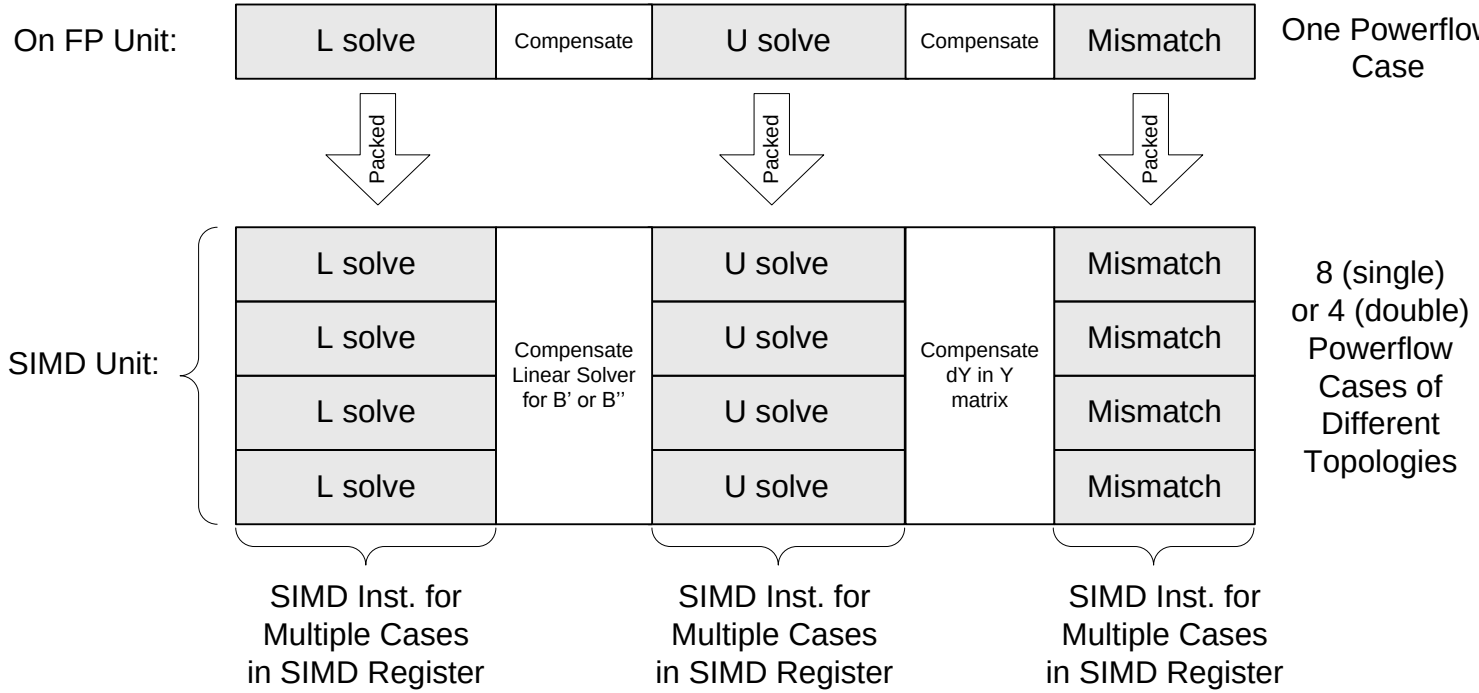
$$\tilde{x} = (A^{-1} - A^{-1}M(\Delta y^{-1} + M^T A^{-1} M)^{-1} M^T) A^{-1} b$$

  - Steps:
    - Forward: $\boxed{F = L^{-1}b}$ FS
    - Compensate: $\boxed{\begin{array}{l} W = L^{-1}M \quad \tilde{W}^T = M^T U^{-1} \\ c = (\Delta y^{-1} + \tilde{W}^T W)^{-1} \end{array}}$ Compensate Pre-compute

      $\Delta F = -W c \tilde{W}^T F \qquad \tilde{F} = F + \Delta F$
    - Backward: $\boxed{\tilde{x} = U^{-1}\tilde{F}}$ BS

Single Core: SIMD Implementation

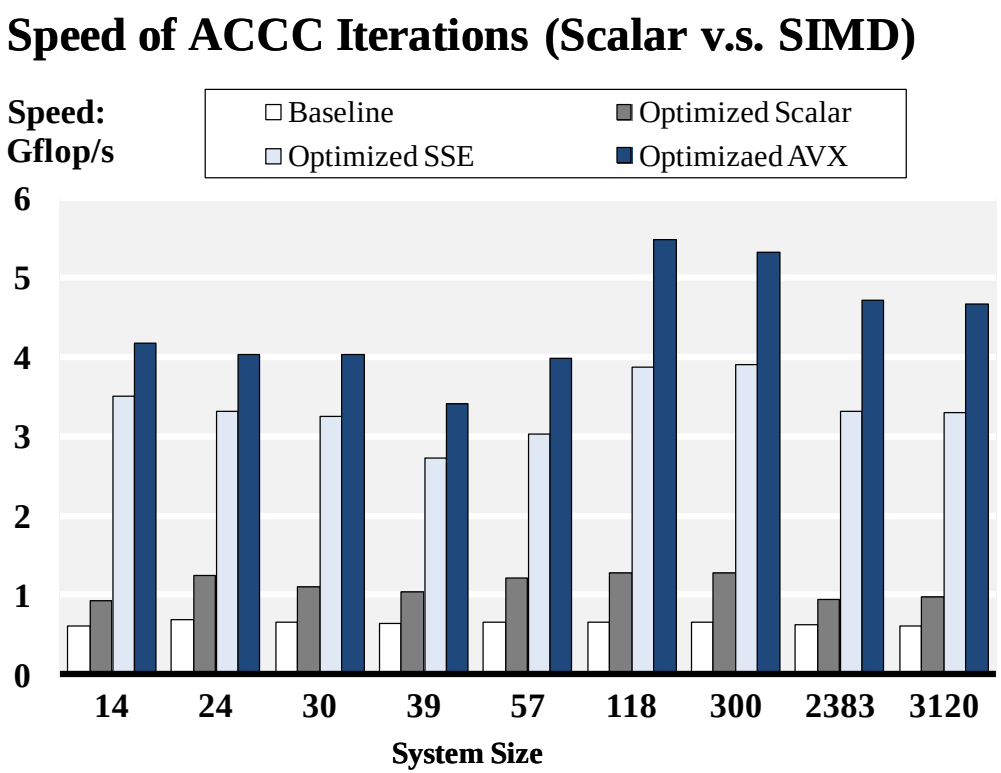
- **Mapping to fine grain parallel hardware**



- In L, U and mismatch: 4 (SSE) or 8 (AVX) cases computed together
- In compensation: serialized and compensate every SIMD slot
- Iteration num determined by the largest iteration num of a SIMD pack

Single Core Results:

- **Scalar speed and AVX speedup on *single* core in Gflop/s**

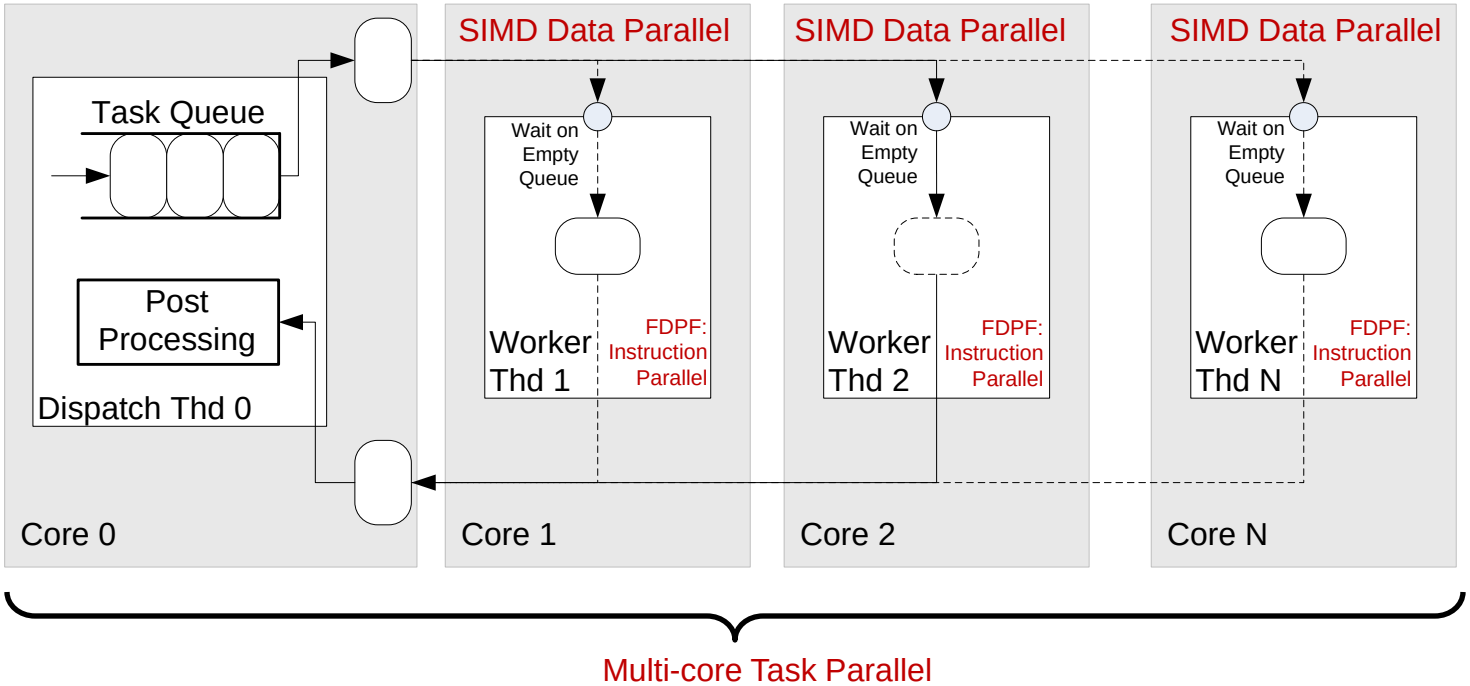


  - Fully utilize single core with SIMD
  - **Better** speedup on **larger** system

Core i7 2670QM, quad-core, 2.2GHZ, AVX, 6MB Cache, Gaming Laptop

Multi-Core: Thread Pool

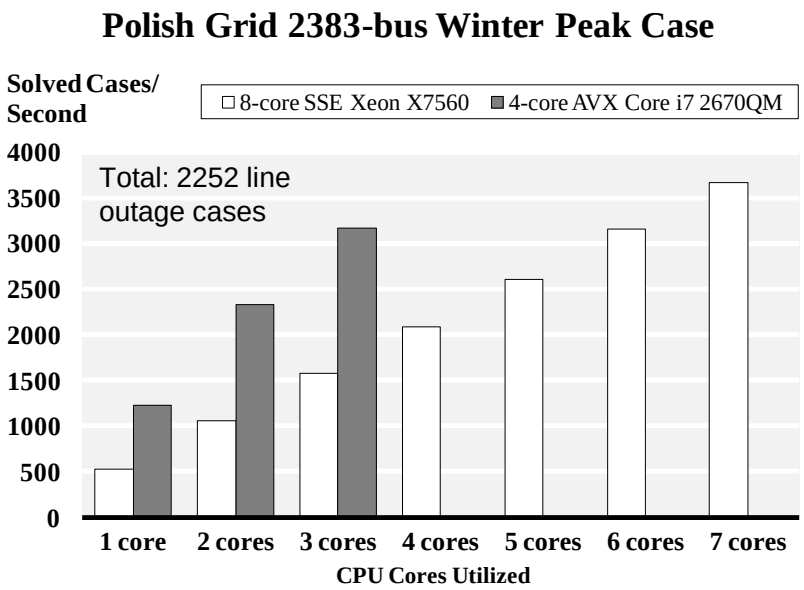
- **Main concern: balancing load among cores**
  - Thread pool scheduler

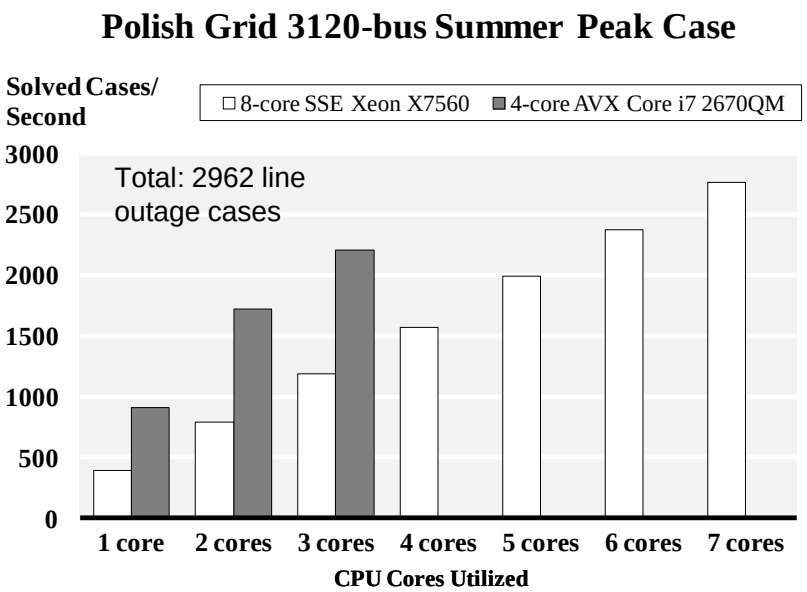


  - Dynamically balancing large amount of small tasks of different convergence steps

Multi-Core Results

- **Multi-core speed in Solved Cases / Second**





- **Fully utilize computing resources of multi-core CPUs**
- **Polish Grid N-1 line outage ACCC every second on 4-core Core i7 (~30x speedup over full compiler optimized baseline)**

Tolerance: 1kW, Maximal Iteration: 20

Xeon X7560: 8-core, 2.26GHz, SSE, 24MB Cache, Server

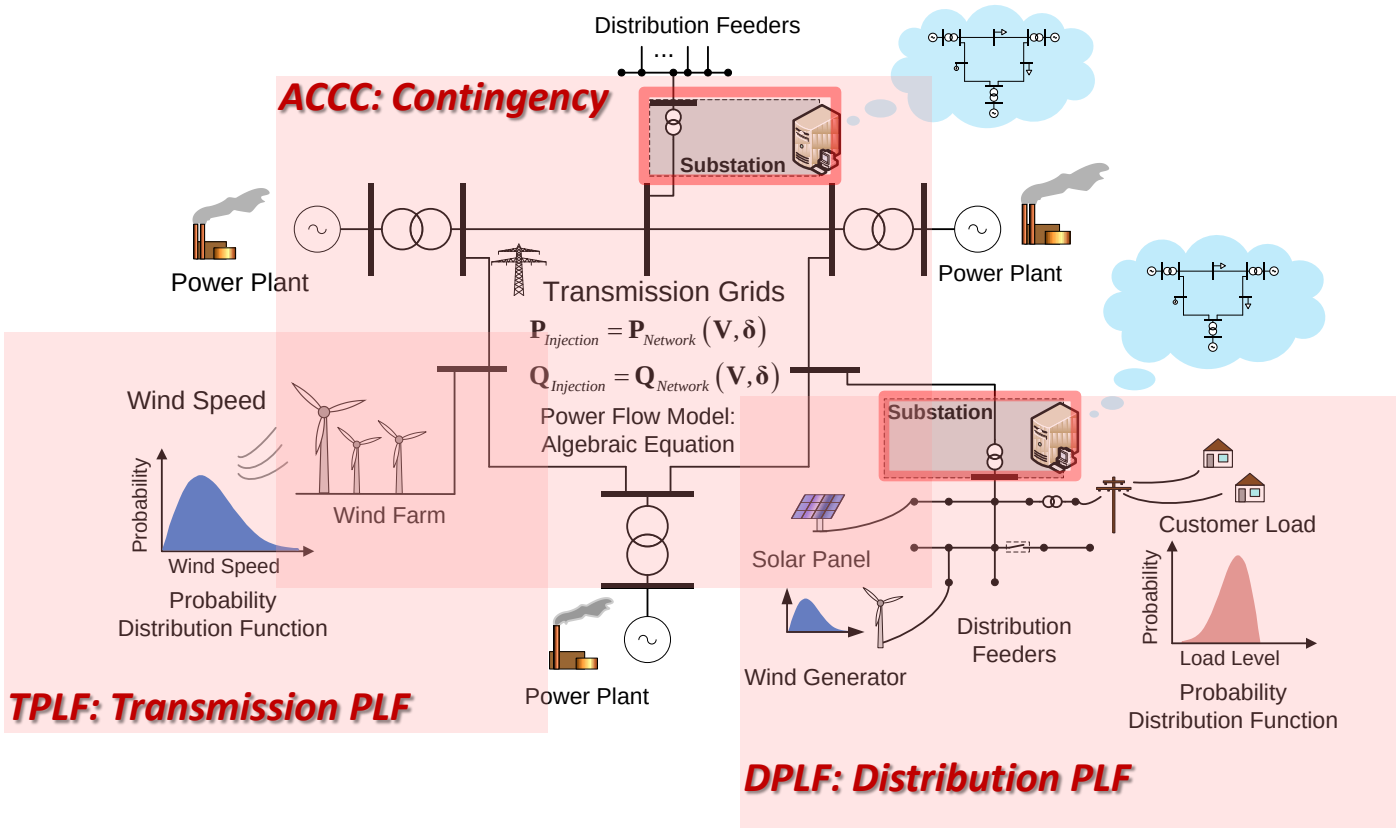
Core i7 2670QM, quad-core, 2.2GHZ, AVX, 6MB Cache, Gaming Laptop

Summary

- **DPLF: distribution PLF solver**
  - Millions of load flow cases within SCADA interval
  - Real time accurate, generally-applicable PLF solution
  - Novel full probabilistic SCADA extension
- **TPLF: transmission PLF solver**
  - Optimized solver for fast decoupled power flow
  - Converged accurate MCS PLF results every second for real time apps
- **ACCC: accelerated AC contingency calculation**
  - Fully utilize HW computing capability: cache, instruction level, SIMD, multi-core
  - Complete N-1 for entire national level network (Polish grid) every second

- **HW/SW performance engineering leads to unique solution: Computing enabled situational awareness in substations, control center capability for real time applications targeting critical smart grid challenges**

Summary



- **HW/SW performance engineering leads to unique solution:**
  Computing enabled situational awareness in substations, control center capability for real time applications targeting critical smart grid challenges

References

- [LBNL-3884E]. Mills, A. Implications of Wide-Area Geographic Diversity for Short-Term Variability of Solar Power, LBNL-3884E. Lawrence Berkeley National Laboratory, Berkeley,
- [ORNL/TM2004/291]. B. Kirby, "Frequency Regulation Basics and Trends," ORNL/TM 2004/291, Oak Ridge National Laboratory, December 2004.
- [Invest:2010] Kwok, Peter Jordan, "Electricity transmission investment in the United States : an investigation of adequacy", M.Sc Thesis MIT 2010.
- [Jorgensen:1998] P. Jorgensen, J. S. Christensen and J. O. Tande, "Probabilistic load flow calculation using Monte Carlo techniques for distribution network with wind turbines," 8th International Conference on Harmonics and Quality of Power, vol. 2, pp.1146-1151, 1998
- [Ruiz:2012] F. Ruiz-Rodriguez, J. Herna andndez, and F. Jurado, "Probabilistic load flow for radial distribution networks withphotovoltaic generators," Renewable Power Generation, IET, vol. 6, no. 2, pp. 110 –121, 2012
- [Li:2012] G. Li and X.-P. Zhang, "Modeling of plug-in hybrid electric vehicle charging demand in probabilistic power flow calculations," Smart Grid, IEEE Transactions on, vol. 3, no. 1, pp. 492 –499, march 2012
- [Ghosh:1997] A. Ghosh, D. Lubkeman, M. Downey, and R. Jones, "Distribution circuit state estimation using a probabilistic approach," IEEE Transactions on Power Systems, vol. 12, no. 1, pp. 45 – 51, Feb 1997.
- [Caramia:2007] P. Caramia, G. Carpinelli, M. Pagano, and P. Varilone, "Probabilistic three-phase load flow for unbalanced electrical distribution systems with wind farms," Renewable Power Generation, IET, vol. 1, no. 2, pp. 115–122, 2007
- [Allan:1981] R. Allan, A. Leite da Silva, and R. Burchett, "Evaluation methods and accuracy in probabilistic load flow solutions," IEEE Transactions on Power Apparatus and Systems , vol. PAS-100, no. 5, pp. 2539 –2546, May 1981
- [Silva:1990] Leite Da Silva, A.M.; Arienti, V. L., "Probabilistic load flow by a multilinear simulation algorithm," Generation, Transmission and Distribution, IEE Proceedings C , vol.137, no.4, pp.276,282, Jul 1990
- [Wang:1992] Wang, Z.; Alvarado, F.L., "Interval arithmetic in power flow analysis," Power Systems, IEEE Transactions on , vol.7, no.3, pp.1341,1349, Aug 1992
- [Su:2005] C. L. Su, "Probabilistic load-flow computation using point estimate method," IEEE Trans. Power Systems, vol. 20, no. 4, pp. 1843-1851, Nov. 2005
- [Zhang:2004] P. Zhang and S.T. Lee, "Probabilistic load flow computation using the method of combined and Gram-Charlier expansion," IEEE Trans. Power Systems, vol.19, no.1, pp. 676-682, Feb. 2004
- [Zhang:2009] G. Zhang, B. Zhang, H. Sun, and W. Wu, "Ultra-short Term Probabilistic Transmission Congestion Forecasting Considering Wind Power Integra-tion," in Advances in Power System Control, Operation and Management (APSCOM 2009), 8th International Conference on, Nov. 2009, pp. 1 –6
- [Liao:2007] H. Liao, "Fast Deterministic Sampling for Mean and Covariance Esti-mation in Stochastic Load Flow," in Power Engineering Society General Meeting, 2007. IEEE, Jun 2007, pp. 1 –6.
- [Hajian:2012] M. Hajian, W. D. Rosehart, and H. Zareipour, "Probabilistic Power Flow by Monte Carlo Simulation with Latin Supercube Sampling," Power Systems, IEEE Transactions on, vol. PP, no. 99, p. 1, 2012.
- [PNL_Hybrid:2009] I. Gorton, Z. Huang, Y. Chen, B. Kalahar, S. Jin, D. Chavarria-Miranda, D. Baxter, and J. Feo, "A high-performance hybrid computing approach to massive contingency analysis in the power grid," in e-Science, 2009. e-Science '09. Fifth IEEE International Conference on, dec. 2009, pp. 277 –283
- [PNL_Massive:2009] Z. Huang, Y. Chen, and J. Nieplocha, "Massive contingency analysis with high performance computing," in Power Energy Society General Meeting, 2009. PES '09. IEEE, july 2009, pp. 1 –8.
- [Gopal_GPU:2010] A. Gopal, "DC power flow based contingency analysis using graphics processing units," Master Thesis, Drexel ECE, 2010.
- [PSSE] Siemens-PTI, "PSS/E: Power System Simulator for Engineering."
- [Johnson:2008] J. Johnson, T. Chagnon, P. Vachranukunkiet, P. Nagvajara, and C. Nwankpa, "Sparse LU decomposition using FPGA," in International Workshop on State-of-the-Art in Scientific and Parallel Computing (PARA), 2008.
- [Sanja:2011] Sanja Cvijic and Marija Ilic. Contingency screening in multi-control area system using coordinated DC power flow. ISGT Europe Manchester, UK, 2011.
- [Sanja:2012] Sanja Cvijic and Marija Ilic, "Optimal Clustering for Efficient Computations of Contingency Effects in Large Regional Power Systems", PES General Meeting in San Diego, July 2012
- [Johnson:2008] Jeremy Johnson, Tim Chagnon, Petya Vachranukunkiet, Prawat Nagvajara, and Chika Nwankpa. Sparse lu decomposition using fpga. In International Workshop on State-ofthe-Art in Scientific and Parallel Computing (PARA), 2008.

The End

Thank you!

Q&A