CarnegieMellon ENGINEERING

Electrical & Computer ENGINEERING

A Software Performance Engineering Approach to Fast Transmission Probabilistic Load Flow Analysis

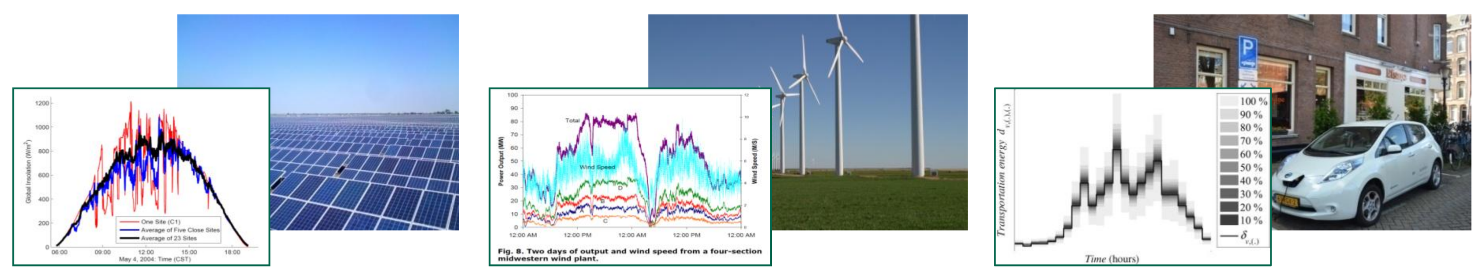
Tao Cui and Franz Franchetti

Email: tcui@ece.cmu.edu

Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA

Motivation and Background

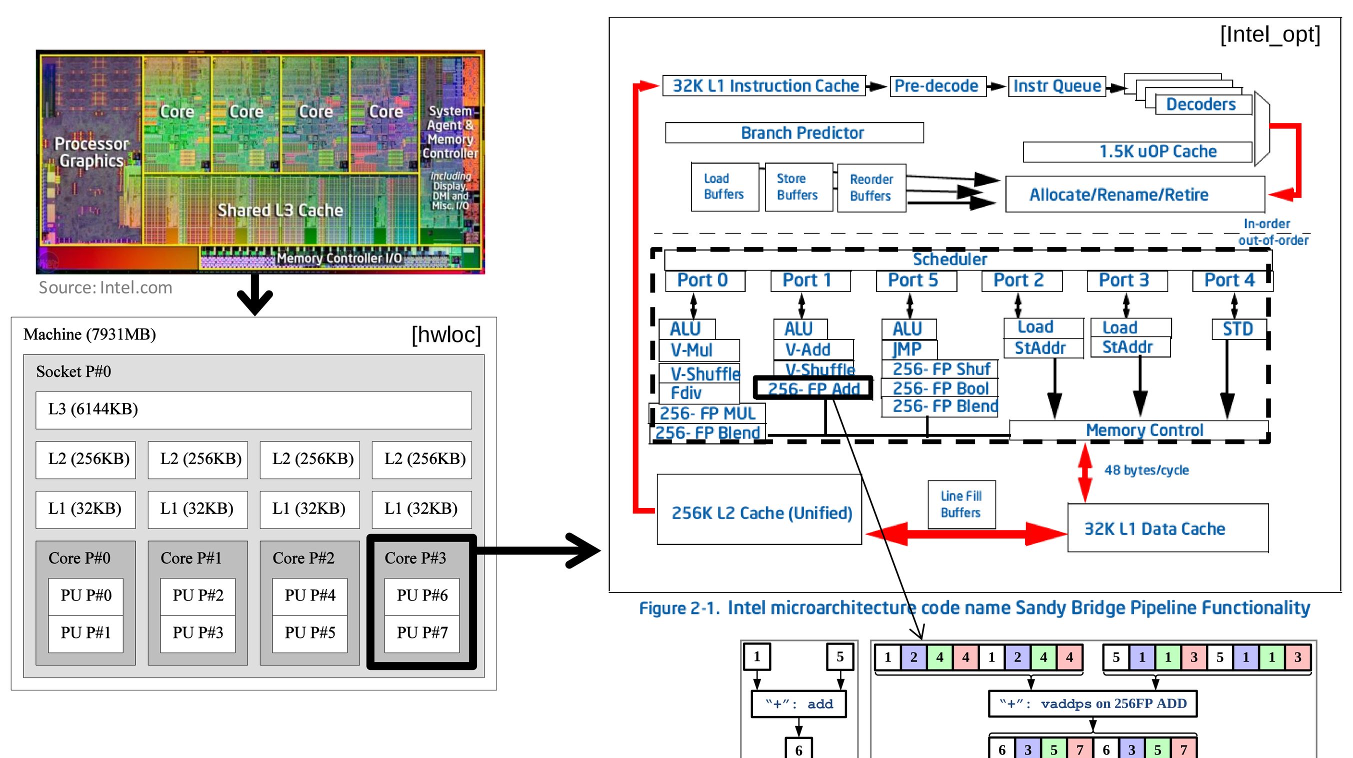
❖ Power System Probabilistic Analysis:



Source: LBNL-3884e | Source: ORNL/TM2004/291 | Source: Pantos 2011

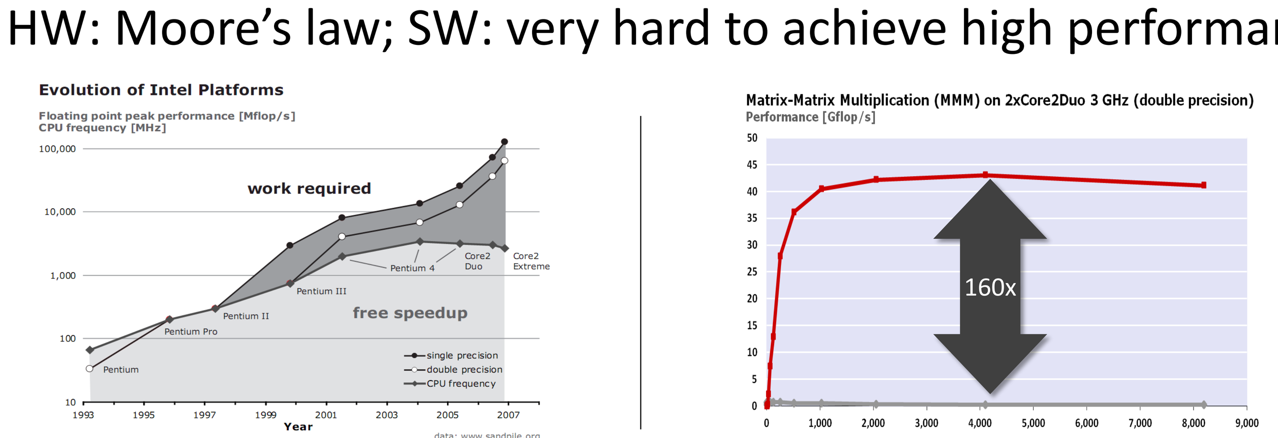
- Challenges: new players on the grid
  - Undispatchable, large variances, great impact on grid
  - Uncertainties & variations.
- New requirements
  - NERC: probabilistic analysis from distribution & transmission
- Online computation tool for the smart grid probabilistic analysis

❖ Modern Computer Architecture – Challenge for High Performance



- Memory hierarchy
- Multilevel parallelism: Data level (SIMD), instruction level, multithread

- HW: Moore's law; SW: very hard to achieve high performance



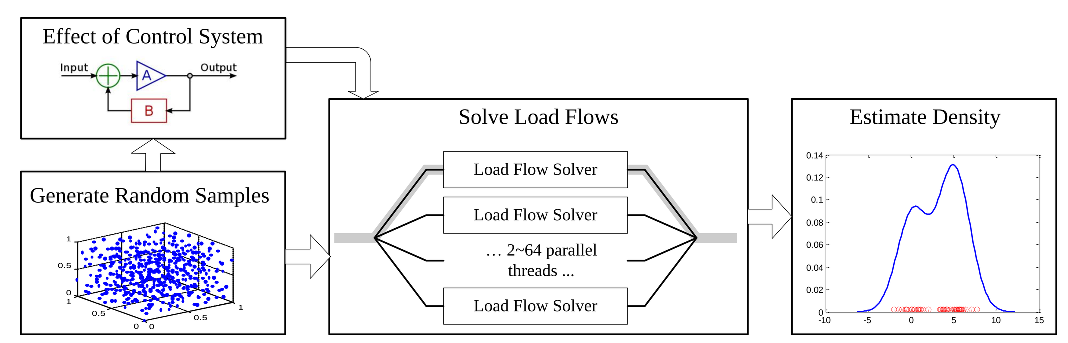
- Power system applications: algorithm & math library

Can we fully utilize the modern commodity computing systems, build a fast, robust, & generally applicable solver for smart grid real time probabilistic analysis

Programming Model & Performance Tuning

❖ Monte Carlo Simulation Based Probabilistic Load Flow

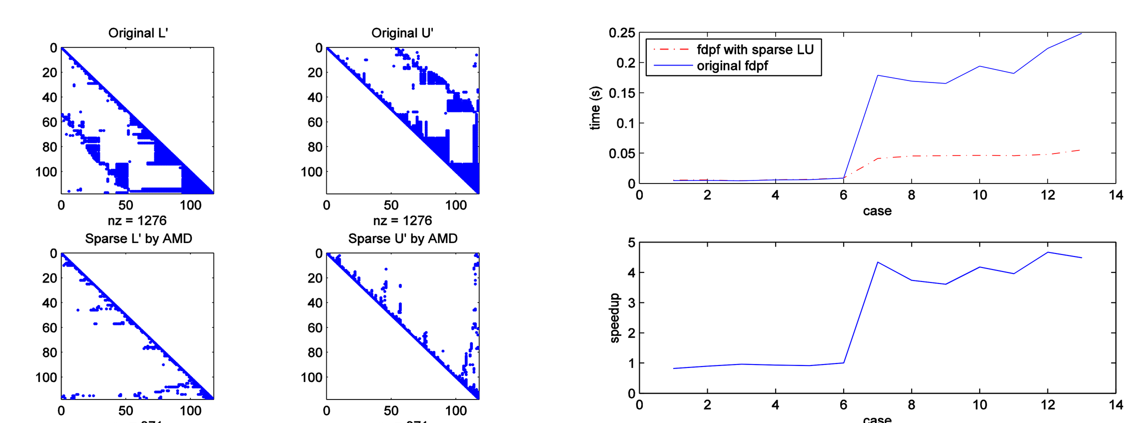
- "Gold standard"/accuracy reference for analytical methods
- Robust, generally-applicable, convergence in theory
- Heavy computational burden, impractical for online application?
- Well fitted problem on modern computing platform



❖ Algorithm Level Optimization:

$$-\mathrm{B}'\Delta\theta = \Delta\mathrm{P}/\mathrm{V}$$
$$-\mathrm{B}''\Delta\mathrm{V} = \Delta\mathrm{Q}/\mathrm{V}$$

- Base: Fast Decoupled Load Flow
- Sparse LU decomposition- Approximated Minimal Degree



IEEE 118 system

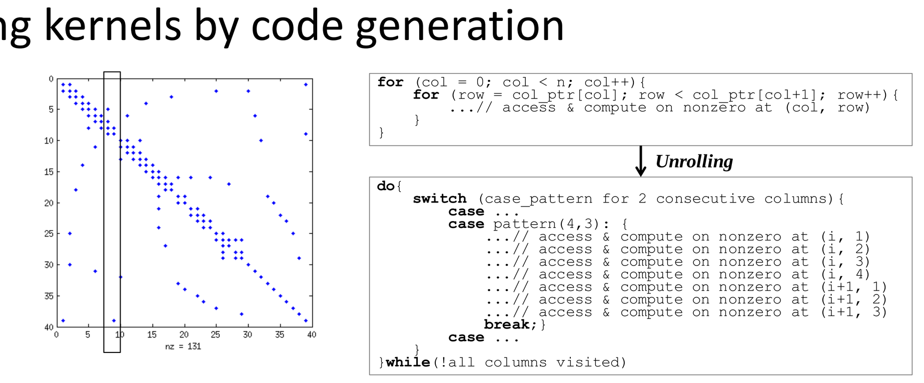
13 cases: 14, 30, 39, 57, 118, 300, 2383, 2736, 2737, 2746, 2746, 3012, 3120

❖ Computer Architecture Level Optimization:

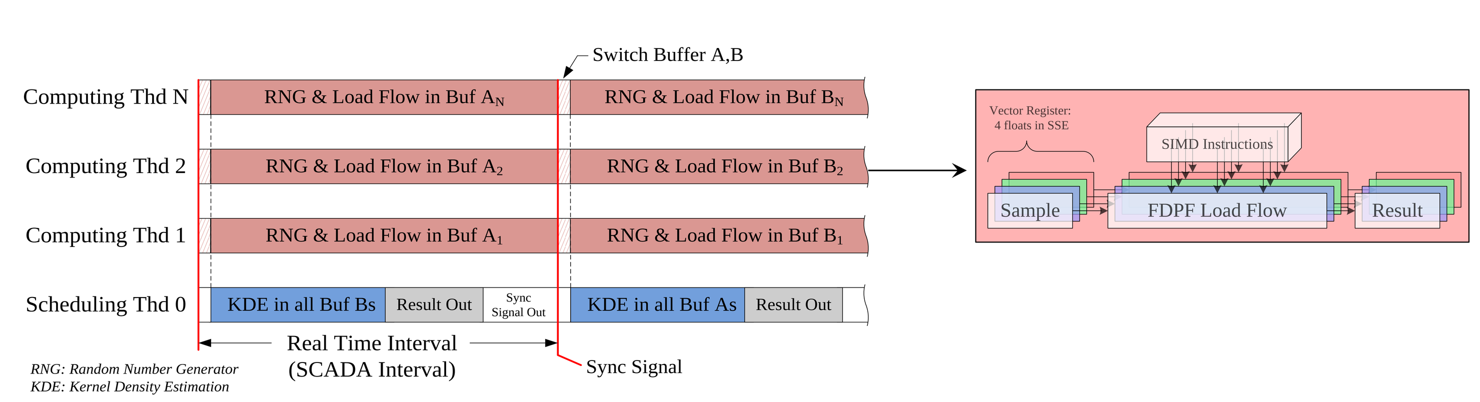
- Data structure optimization:

New CCS format for memory access | **Reduce trigonometric operations**

- Unrolling sparse computing kernels by code generation
  - Nonzeros' pattern
  - Pre-generated
  - More non-branch inst.
  - Superscalar processor

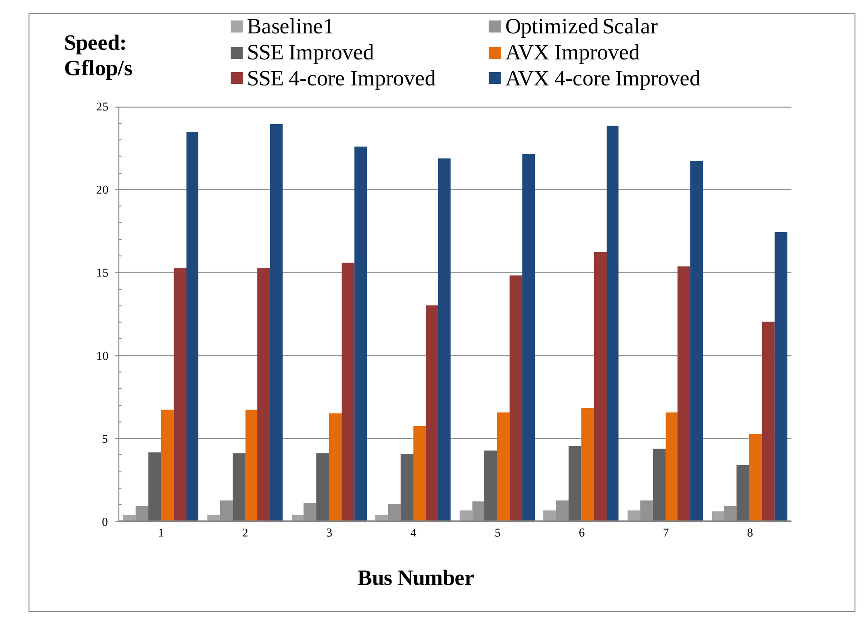


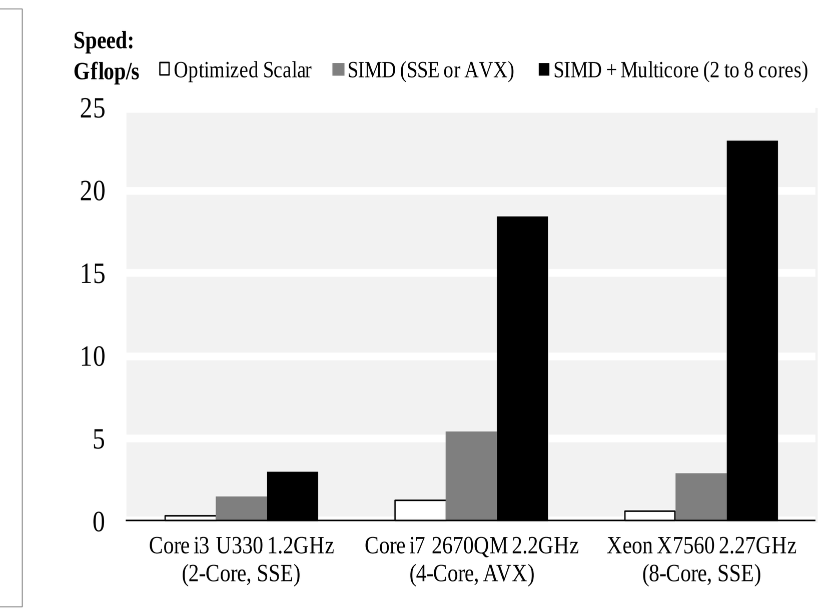
- Multiple level parallelizaion for real time MCS



Implementation & Demonstration

❖ Performance Result: High Performance Computing Engine





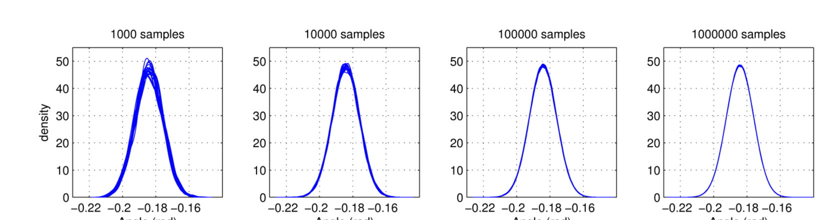
- ~50x speedup on Core i7 thanks to architecture level optimizations
- Performance increases with HW parallel capabilities

❖ Monte Carlo Results and Power Flow Throughput Performance

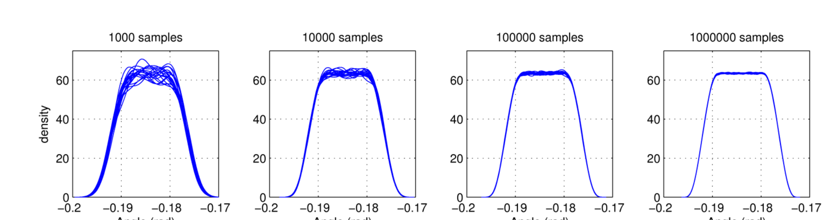
Approx. Speed: Load Flow Cases Solved per Second on Core i7

Test Cases		Approx. Speed [cases/s]	
System Size.	Flops/Iteration	Baseline[1]	AVX 4-Core
14	1,034	39,000	1,920,000
24	1,788	23,000	1,066,000
30	2,242	19,000	860,000
39	2,715	23,000	697,000
57	4,467	15,000	414,000
118	9,130	7,000	202,000
300	23,370	3,000	76,000
2,383	175,365	340	8,100

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



(a) Phase angle of Bus 59. Normal distribution P Injections



(b) Phase angle of Bus 59. Uniform distribution P Injections

- Left: How many load flow can be solved every second
- Right: Example phase angle results on IEEE118 system
  - (a)Normal(0,10)MW and (b)Uniform(-10,10)MW random active power on first three highest loading buses (Bus 59, 90, 116)

Conclusions

❖ Code optimization / parallelization on commodity CPUs

- Fully taking advantages of commodity computing system

❖ Performance scalable with the hardware parallel capacity:

- Tracking new development in CPU micro-architecture.

❖ A real time Monte Carlo solver for probabilistic load flow

- A novel, robust, generally applicable & fast solver for smart grids challenges & requirements by software performance engineering

Acknowledgement

- This work is supported by NSF through awards 0931978 and 0702386.

Electric Energy EESG Systems Group

SPIRAL www.spiral.net