Energy Efficient Big Data Processing at the Software Level

Da-Qi Ren, Zane Wei

Huawei US R&D Center
Santa Clara, CA 95050

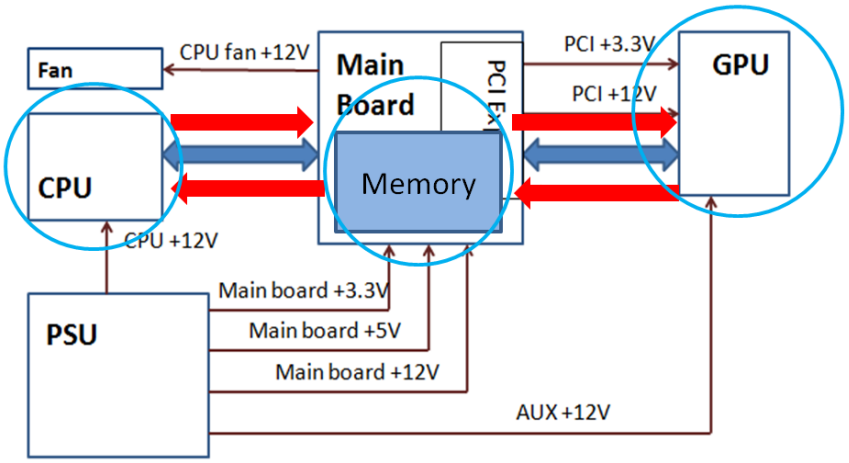
Power Measurement on Big Data Systems

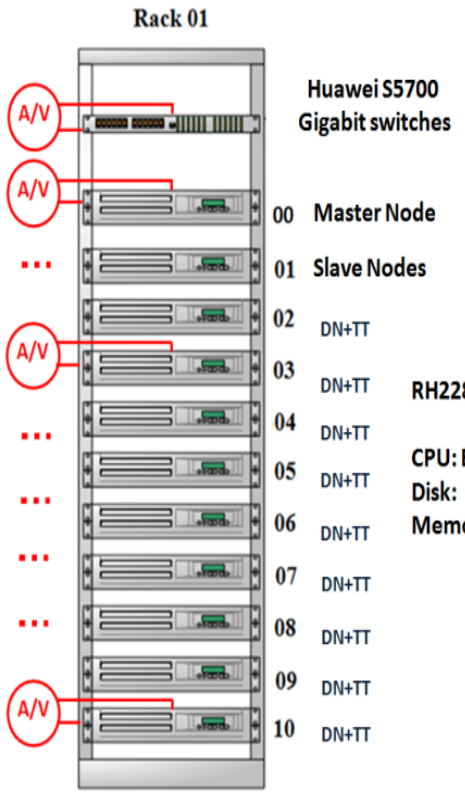
1. If the System Under Test (SUT) is physically standalone, ----- measure from the input of the rack/cabinet/case.

2. If devices are separately powered --- separately measured.

3. The total power ----- summation of each component.

$$P = \sum_{1 \le i \le m} p_i$$





Component level power need to be measured

Energy Metrics

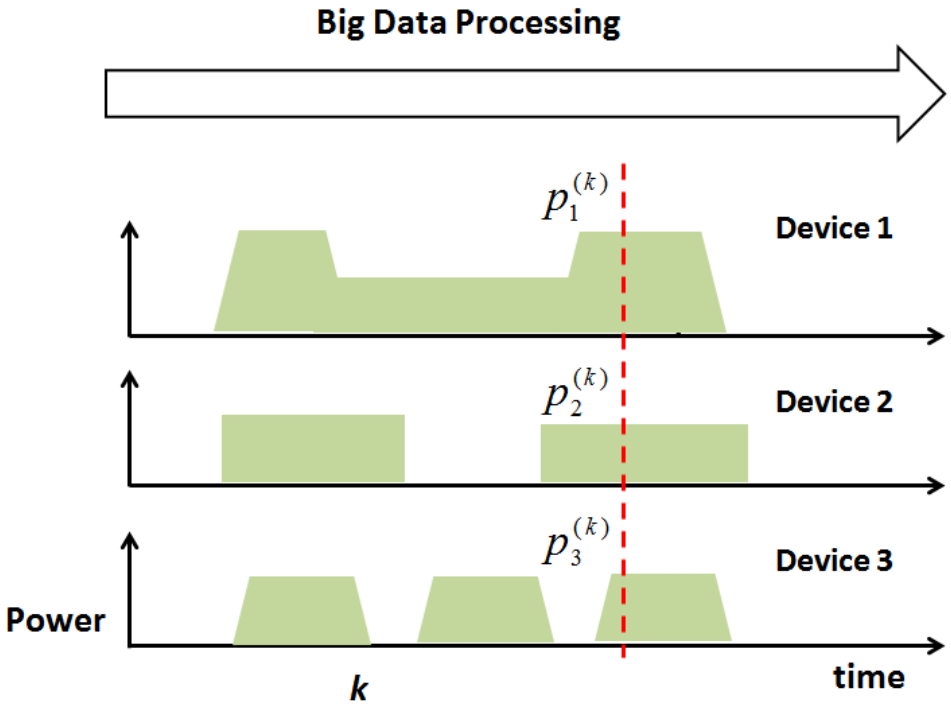
Energy Calculation Based on Measurement Results

For each device or subsystem, the calculation defined for Energy metrics is:

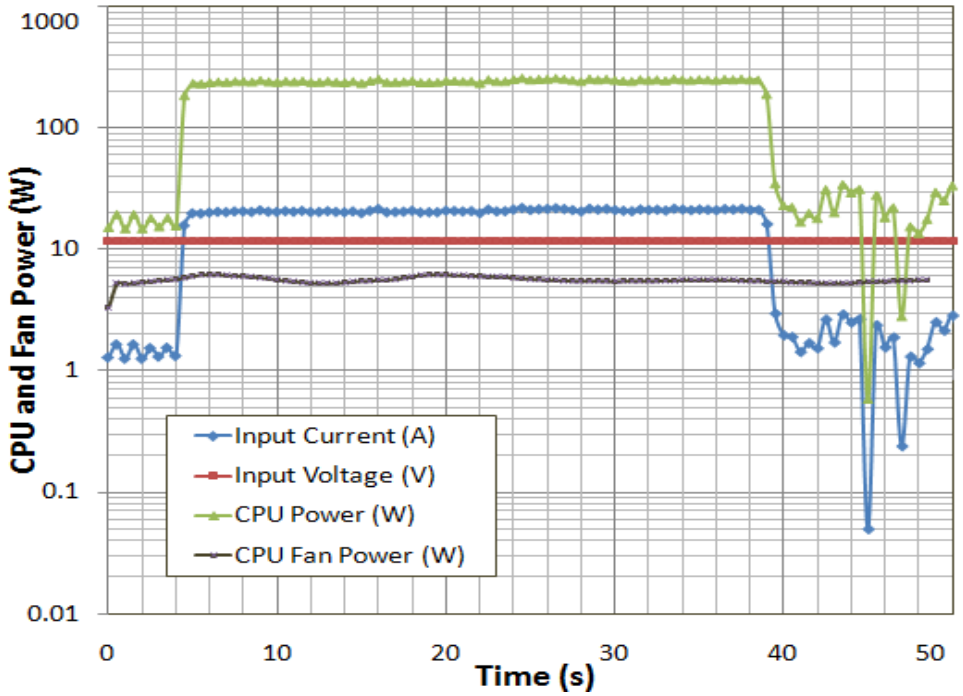
$$E = \int_0^T P(t)dt \qquad E = \sum_0^T P^{(k)}k$$

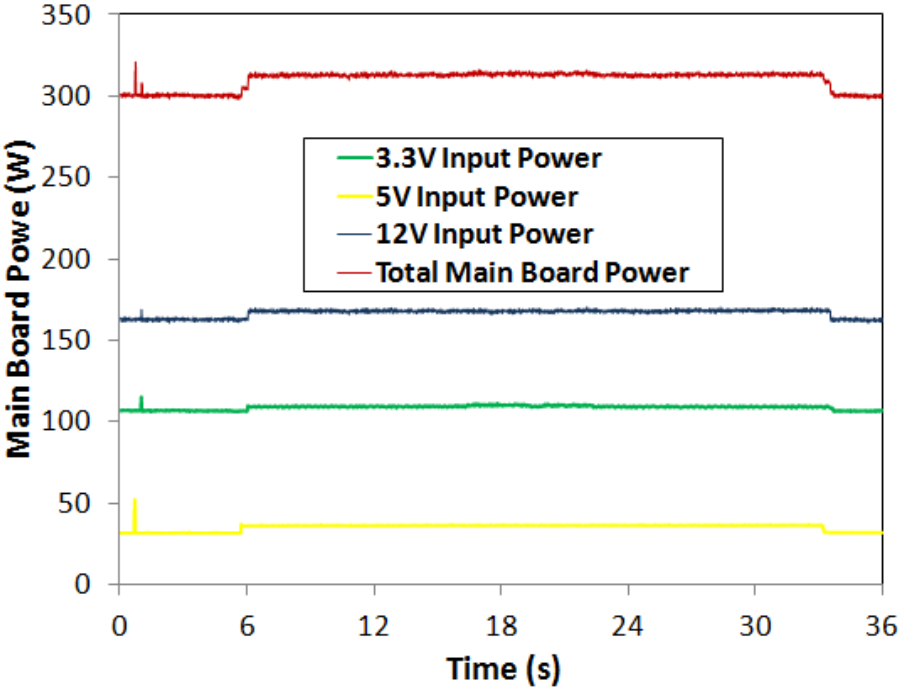
Equation (1) , *Energy* ***E*** is the elapsed time ***t*** times power ***P***.

When sampling interval is small enough, the energy calculation of equation (1) becomes the format in equation (2). All real measurements can be done follow it by using power analyzers.



Sample Power Chart





CPU-GPU Processing Element Power Model

1. **CPU power Measurement.** From CPU socket on main board ---- to measure the CPU input current and voltage at the 8-pin power plug. (Most of the onboard CPUs are powered only by this type of connector)

2. **GPU power measurement.**

3. **Memory and main board power estimation.** we can make an approximation on its power by measuring the power change on the main board.

$$P_{total}(w) = \sum_{i=1}^{N} P_{GPU}^{i}(w^{i}) + \sum_{j}^{M} P_{CPU}(w^{j}) + P_{mainboard}(w)$$

CPU-GPU PE Power Feature Determination

1. Measuring the power from each component of the PE;

2. Find out the energy consumption features
   - **FLOPS / Watt**
   - **Data Size / Watt**
   - **Workload / Watt**

   ……

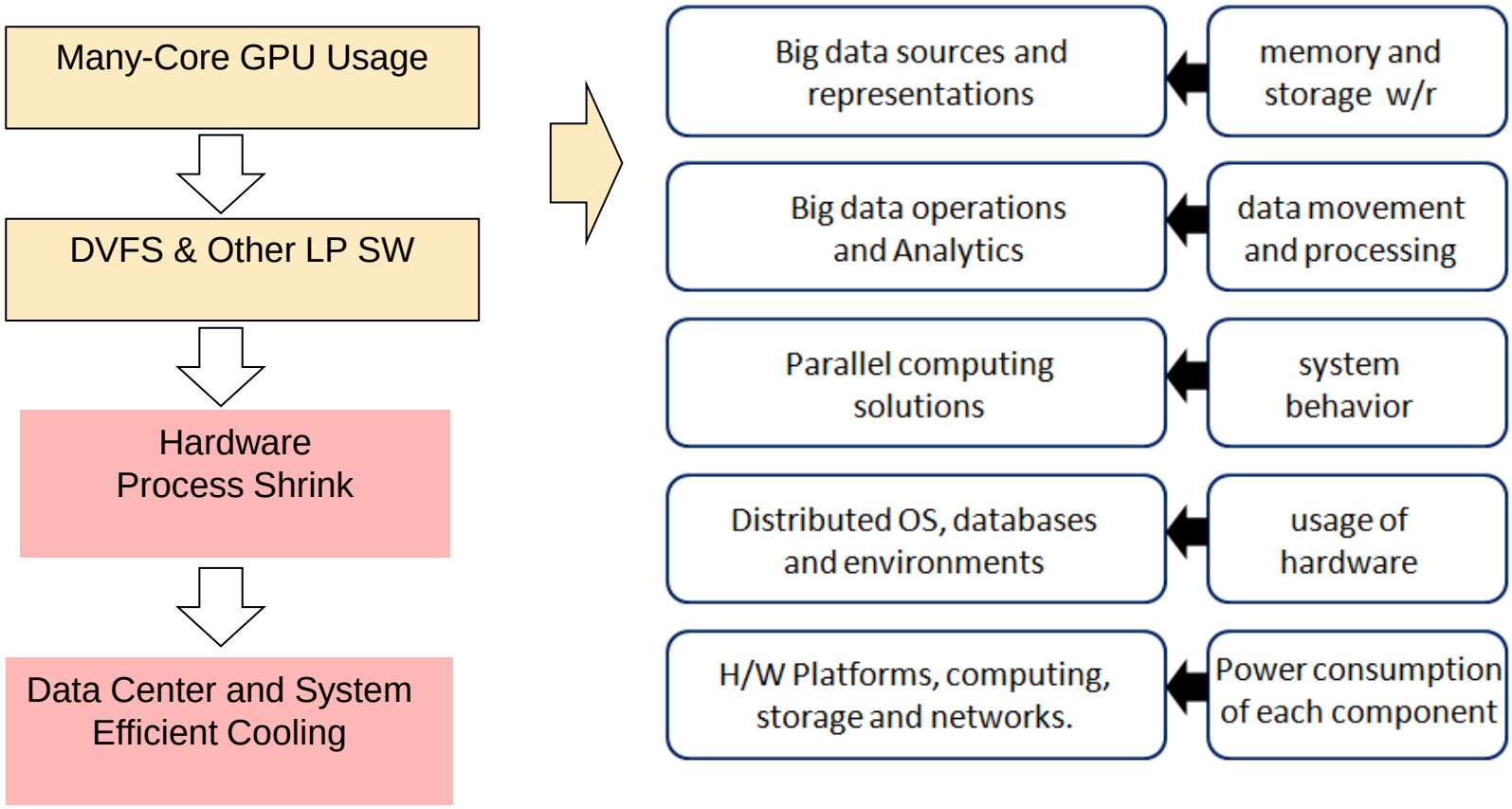
   to this application;

3. Estimated execution time --- the total workload FLOP to be computed divides by the computational speed -------- the CPU-GPU processing element can support;

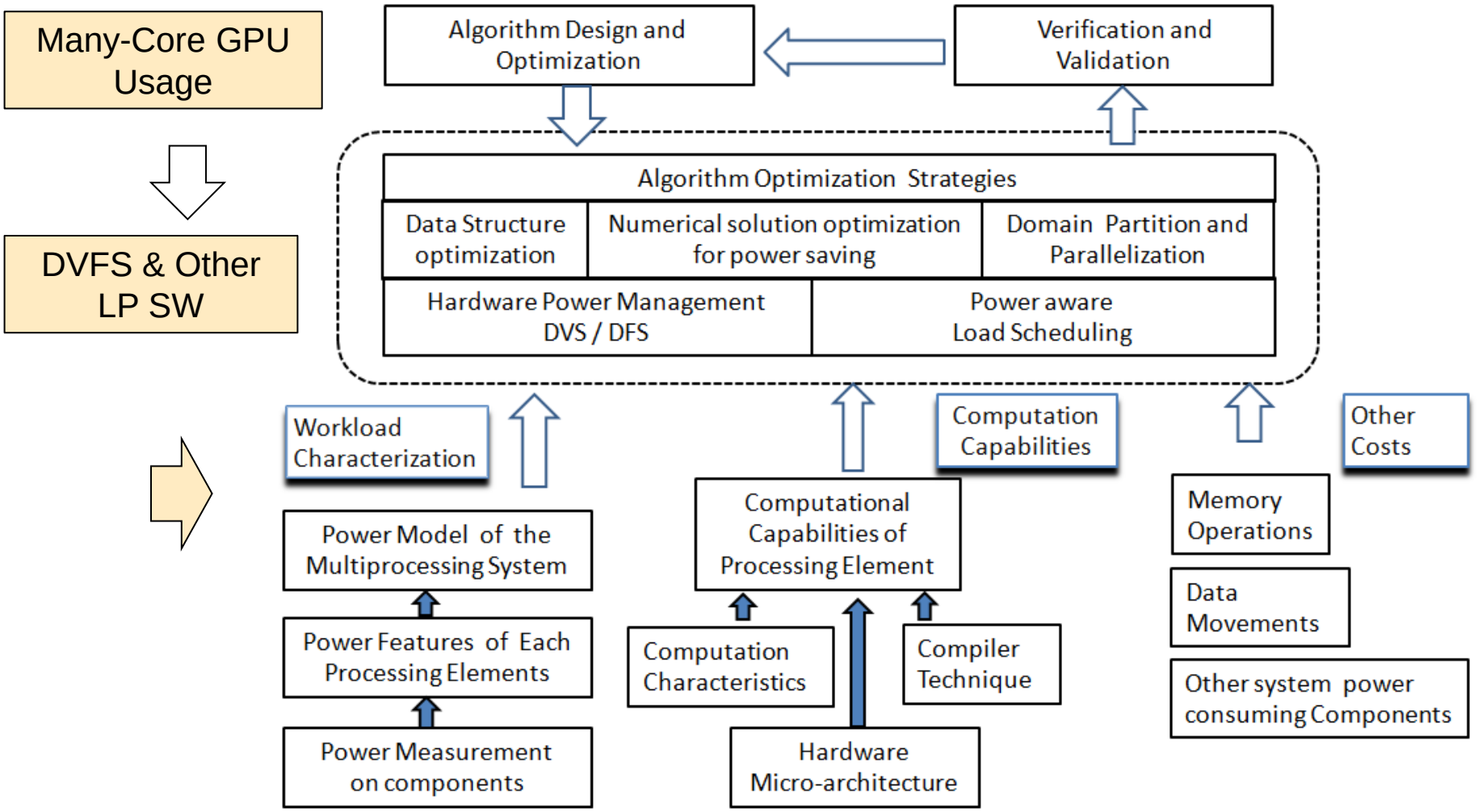
4. Estimated energy consumption to complete the program --------- the summation of products of the component powers and the execution times.

Power issues in each layer of Big Data Computing

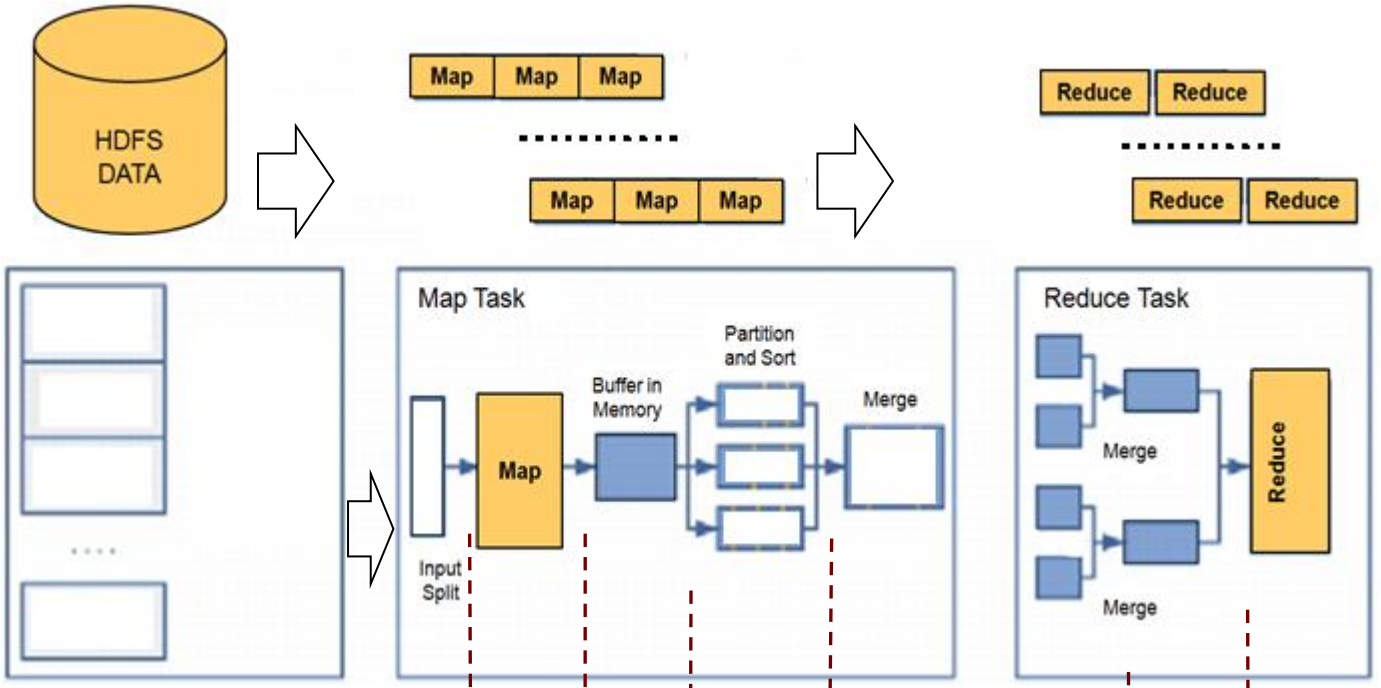
Power Effective Factors, when original power performance is x



Software Solutions for Saving the Energy



Power Analysis in Big Data Computing



Big Data Process

$$E_{total} = \sum_{k=1}^{components} E_k$$

Data Model by Categories

$t_{Storage} p_{Storage}$

$E_{storage}$

$t_0 p_0$ $t_1 p_1$ $t_2 p_2$ $t_3 p_3$ $t_4 p_4$

$$E_1 = \sum_{i=1}^{steps} t_i p_i$$

$t_5 p_5$ $t_6 p_6$ $t_7 p_7$

$$E_2 = \sum_{j=1}^{steps} t_j p_j$$

Dynamic Data Flow

Power Efficient Implementation by Algorithm Optimization

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} \end{bmatrix}, \mathbf{B} = \begin{bmatrix} \mathbf{B}_{1,1} & \mathbf{B}_{1,2} \\ \mathbf{B}_{2,1} & \mathbf{B}_{2,2} \end{bmatrix}, \mathbf{C} = \begin{bmatrix} \mathbf{C}_{1,1} & \mathbf{C}_{1,2} \\ \mathbf{C}_{2,1} & \mathbf{C}_{2,2} \end{bmatrix}$$

$$\begin{aligned} \mathbf{C}_{1,1} &= \mathbf{A}_{1,1}\mathbf{B}_{1,1} + \mathbf{A}_{1,2}\mathbf{B}_{2,1} \\ \mathbf{C}_{1,2} &= \mathbf{A}_{1,1}\mathbf{B}_{1,2} + \mathbf{A}_{1,2}\mathbf{B}_{2,2} \\ \mathbf{C}_{2,1} &= \mathbf{A}_{2,1}\mathbf{B}_{1,1} + \mathbf{A}_{2,2}\mathbf{B}_{2,1} \\ \mathbf{C}_{2,2} &= \mathbf{A}_{2,1}\mathbf{B}_{1,2} + \mathbf{A}_{2,2}\mathbf{B}_{2,2} \end{aligned}$$

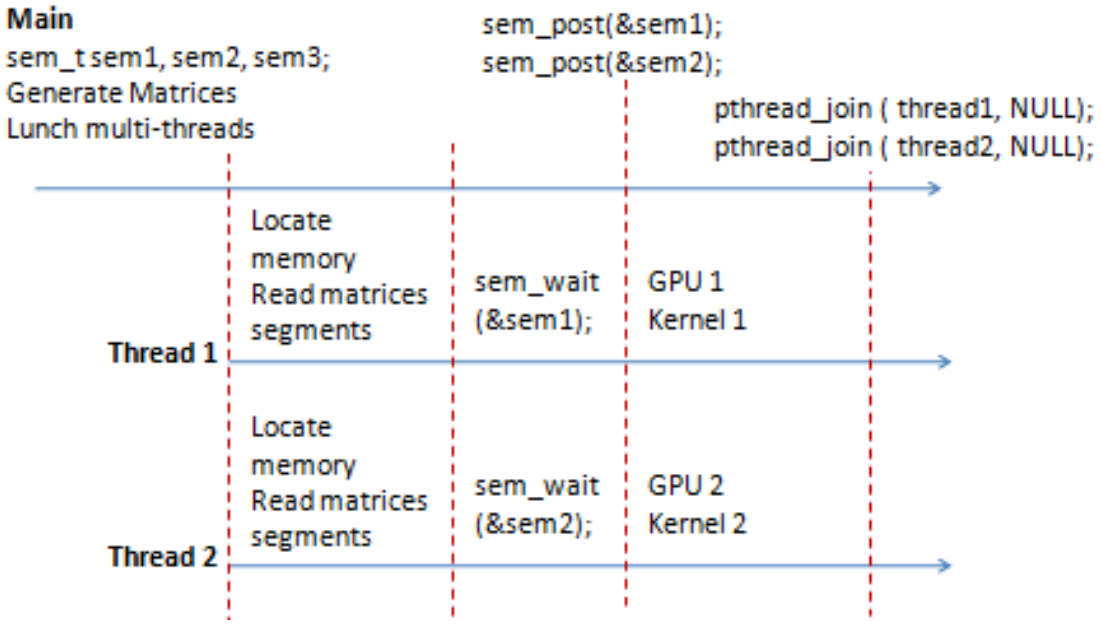
	Simple Algorithm	Enhance Algorithm
CPU Energy Consumption	0.24 wh	0.019 wh
GPU Energy Consumption	0.654 wh	0.055 wh
Main Board (Main Memory) Energy Consumption	0.47 wh	0.0402 wh
Overall Time Consumption	7.5 s	0.6s
Overall Energy Consumption	1.364 wh	0.1142 wh

Build up a power model incorporates physical power constrains of hardware;

Partition of smaller matrix-blocks to fill the shared memories of GPU.

Reduce the data transmission between GPU and main memory, will significantly enhance the GPU performance and power efficiency.

Parallel GPUs



Parallel GPU approach with signal synchronization.

Multithread kernel control to save the use of CPU cores.

Partition jobs and distribute them to each GPU device;

Design synchronization signal to synchronize each CUDA kernel.

Remove CUDA Overhead

- A GPU/CUDA overhead is on kernel initialization, memory copy and kernel launch, before start real kernel computation.
- Remove CUDA overhead by calling C function to compute small workload on CPU.
- Save the time and energy of CUDA kernel and GPU.

A threshold can be determined by experiment by analyzing the follows:

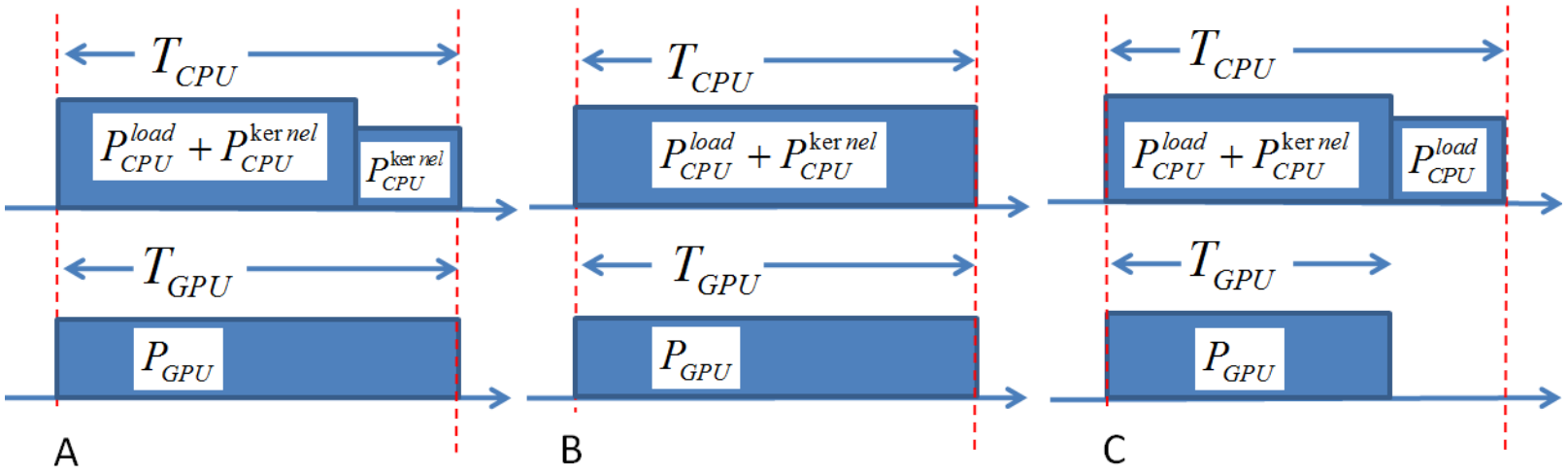
$$T_{CPU}^{k} \le T_{GPU}^{k} = T_{GPUoverhead}^{k} + T_{CUDA\,\mathrm{ker}\,nel}^{k}$$

$$E_{CPU}^{k} = P_{CPU} \times T_{CPU}^{k}$$

$$E_{GPU}^{k} = P_{CPU-GPU-PE} \times T_{GPU}^{k}$$

C function will be slected when matrix size less than k where $E_{CPU}^{k} \le E_{GPU}^{k}$.

CPU Shares GPU's Workload



$$A)\ E_{total} = T_{total}(P_{GPU} + P_{CPU}^{kernel}) + T_{CPU}^{load} P_{CPU}^{load} \quad \text{when } T_{GPU} > T_{CPU}$$

$$B)\ E_{total} = T_{total}(P_{GPU} + P_{CPU}^{load} + P_{CPU}^{kernel}) \quad \text{when } T_{GPU} = T_{CPU}$$

$$C)\ E_{total} = T_{GPU}(P_{GPU} + P_{CPU}^{load} + P_{CPU}^{kernel}) + (T_{total} - T_{GPU}) P_{CPU}^{load} \quad \text{when } T_{GPU} < T_{CPU}$$

CPU Frequency Scaling

- CPU frequency scaling can save CPU power, without decreasing the computation performance.
- CPU frequency should match the requirement of GPU/CUDA kernel calls.
- CPU frequency can be scaled down without compromising with the PE's performance however to save the CPU's power.

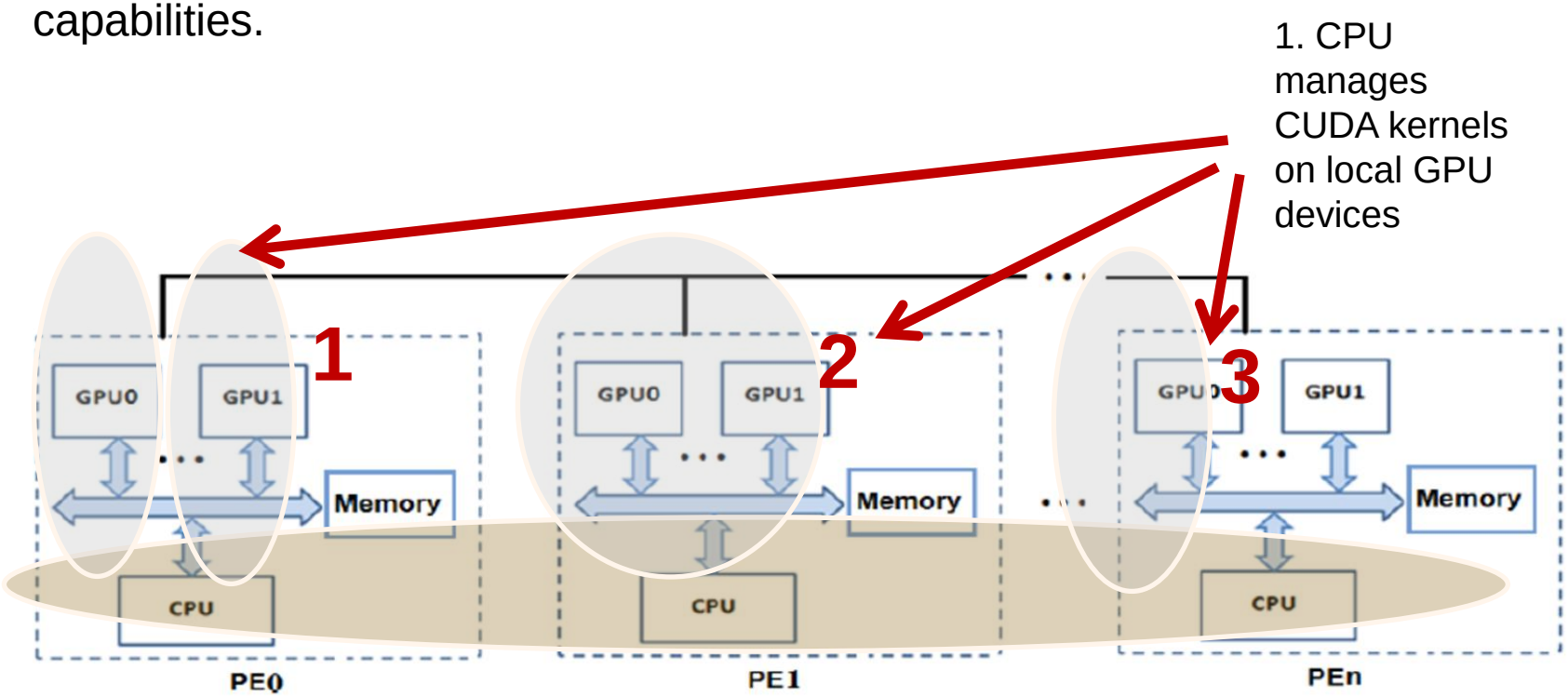
A rough estimation for the minimum CPU frequency value is :

$$F_{CPU} \geq F_{CudaCalls} \text{ (most of the cases)}$$

$$F_{CPU} \geq F_{GPUMemory} \text{ (if } F_{CudaCalls} \geq F_{GPUMemory} \text{)}$$

Energy Aware Load Scheduling with CUDA/MPI

- SIMD/SPMD (Single Instruction/Program Multiple Data) split up tasks to run simultaneously
- CUDA Processing Element (PE)
- CUDA/MPI An important choice in data intensive HPC in various applications.
- Scheduling the workload based on the PE's power feature and computing capabilities.



Conclusion

- Introducing energy efficient software design methodologies for big data processing, power performance metrics and measurements
- Modeling and evaluating large scale computer architectures with multi-core and GPU
- Global optimization for choosing the best power-efficient alternative based on data characteristics and quantitative performance analysis
- Validation of the energy efficiency improvements from the new designed approach

Thank you

www.huawei.com