PDAWL: Profile-based iterative Dynamic Adaptive WorkLoad Balance on Heterogeneous Architectures

Tongsheng Geng Marcos Amaris Stéphane Zuckerman
Alfredo Goldman Guang R. Gao Jean-Luc Gaudiot

June 22, 2020

Outline

Outline

Introduction and Motivation

- High Performance Computing systems : homogeneous/heterogeneous many-cores connected within a single or across multiple chips:
  - Hundreds of integrated Processing Elements (PEs)
  - Heterogeneous resources (GPUs, FPGAs, *etc.*)
  - High performance/low power consumption requirements
- How to exploit parallelism, utilize resources and deliver scalability ?
  - Shared-memory many-core systems: data locality, cache conflict, *etc.*
  - Heterogeneous many-core systems: workload balance, PCIe, *etc.*
- Applications: scientific computing
  - Data Regular: Stencil
  - Data Irregular: Sparse Matrix Vector Multiplication (SpMV)

Outline

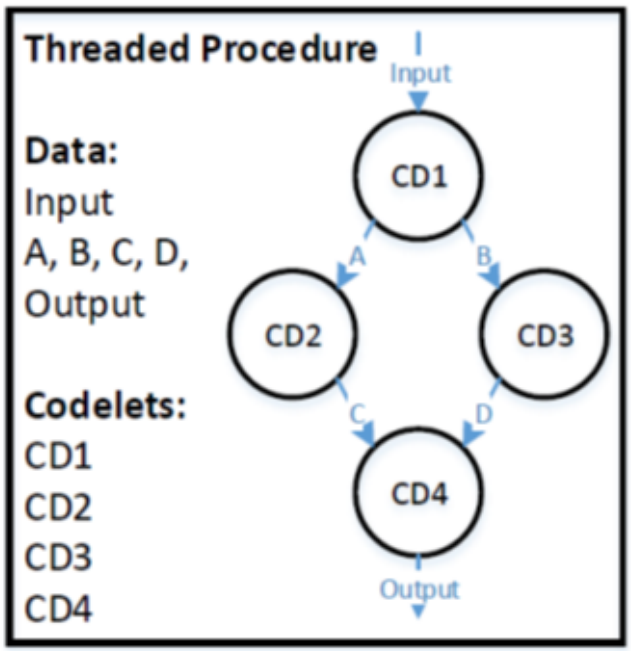
Codelet Model

Codelet Definition

A codelet is a sequence of machine instructions which act as an atomically-scheduled unit of computation

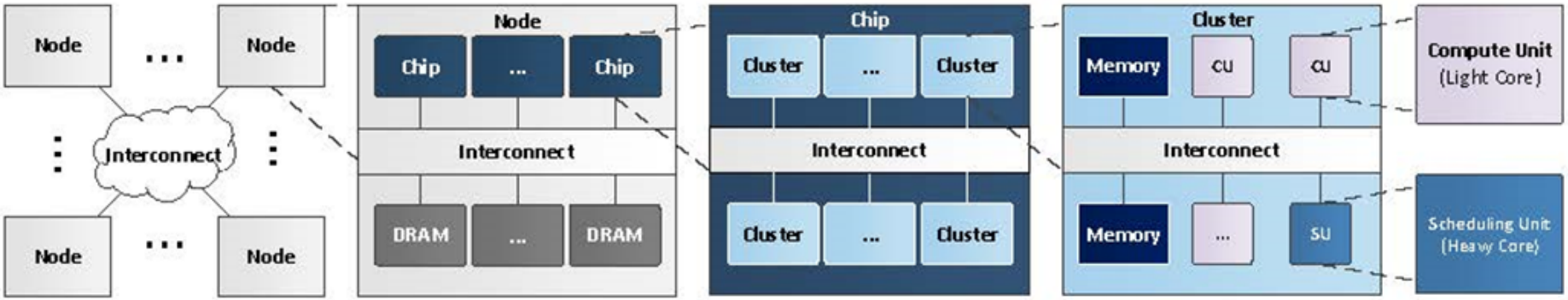
Codelet Properties

- Event-driven
- Communication through inputs and outputs
- Non-preemptive
- All data and code are local



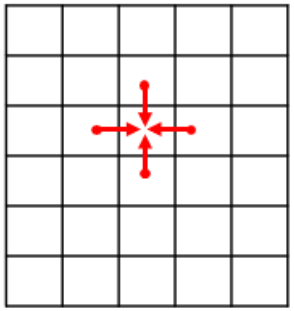
Codelet Abstract Machine(CAM) & Run-Time System (DARTS)

- CAM is a general purpose many-core architecture
  - Scheduling Unit
  - Computation Unit
- DARTS (Delaware Adaptive Run-Time System)
  - Invoke threaded procedures and map them on a given cluster of cores
  - Run the codelets contained within thread procedures

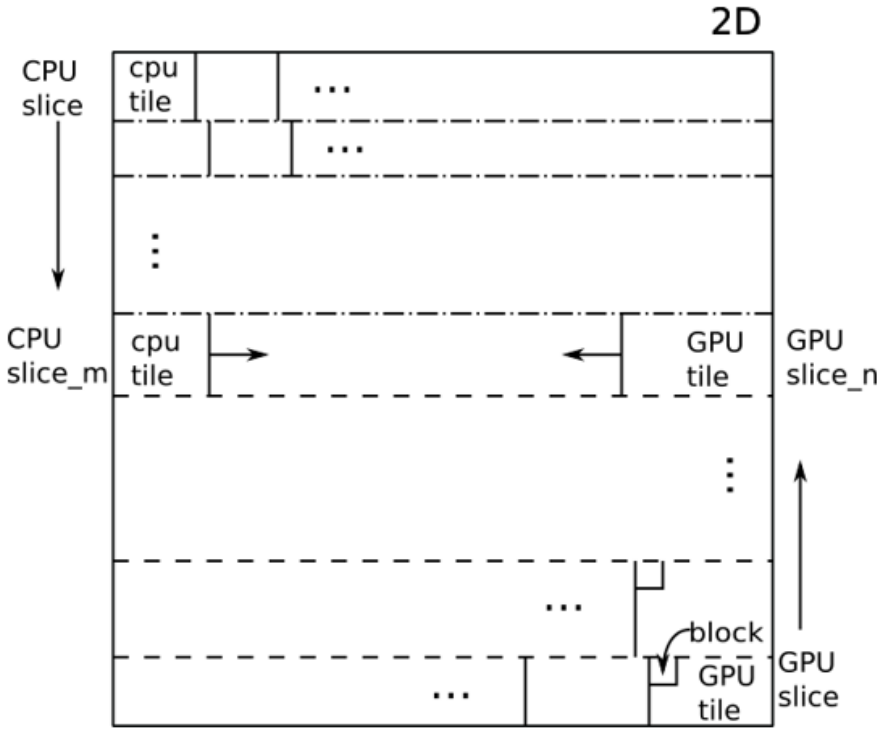


Stencil-based iterative computation

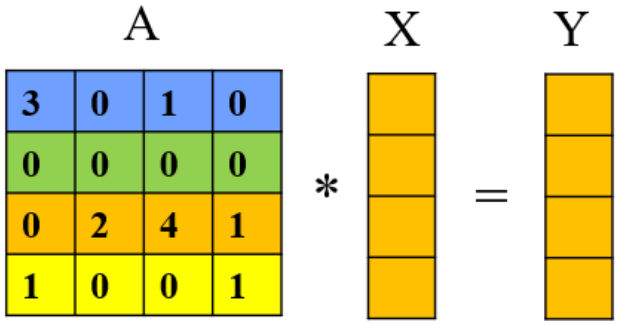
Stencil codes are a type of iterative kernel which update array elements according to some fixed pattern, called a stencil



```
While (--time_step >0){
    for ( size_t i = 0; i < n_rows-1; ++i)
        for ( size_t  j = 1; j < n_cols-1; ++j)
            DST[i][j]=(SRC[i-1][j]+SRC[i+1][j]
                    +SRC[i][j-1]+SRC[i][j+1]) / 4;
    SWAP(&DST, &SRC);
}
```



Sparse Matrix Vector Multiplication - CSR



3	1	2	4	1	1	1

Values

0	2	1	2	3	0	3

Col indices

0	2	2	5	7

Row indices

M: number of rows in the matrix
N: number of columns in the matrix
S: sparsity level [0-1], 1 being fully-dense
Storage requirement: 2MS + M +1

```
Void SpMV_CSR (array A, float *x, float *y){

    for(int row =0; row < A.M ; ++ row){
        float vals = 0;
          const int row_start = A.row_indices[row];
          const int row_end  = A.row_indies[row+1];
          for (idx=row_start; idx<row_end; ++idx){
              vals += A.values[idx]*X[A.col_indices[idx]]
          }
          Y[row]  = vals;
    }
}
```

Note: To obtain high performance, we first extract irregular rows and allocate them onto CPUS, the left regular rows can be allocate on both GPU and CPUs

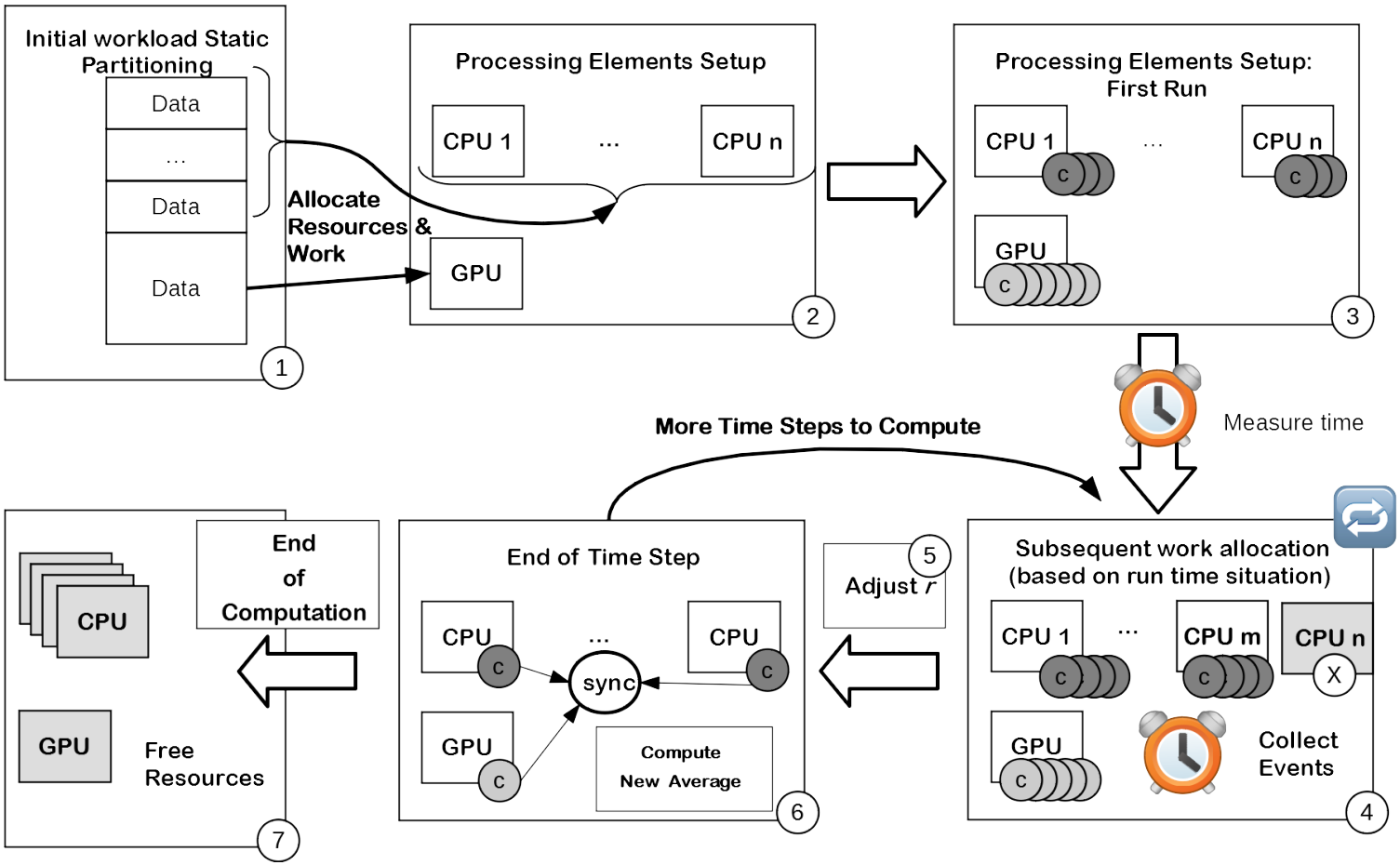
Outline

Methodology

- Build hardware (CPUs or GPU) models (mathematical model or others)
- Allocate initial workloads onto CPUs or GPU based on hardware models
- Adjust workload dynamically based on real time situation.

Dynamic Adaptive Work-Load (DAWL) Scheduling

DAWL was created to decide which tasks should be scheduled and where to schedule workload to minimize the load imbalance between heterogeneous processing elements.



DAWL : same tasks run on both CPUs and GPU

Naive Mathematical Model

$$GPU_{naive} = \text{memcpy}_{Host \rightarrow Device} + \frac{\text{Compute}_{Device}}{\text{NumThreads}_{Device}} + \text{memcpy}_{Device \rightarrow Host} \tag{1}$$

$$CPU_{naive} = \frac{\text{Compute}_{Host}}{\text{NumThreads}_{Host}} \tag{2}$$

$$r = \frac{\text{CPU}_{naive}}{\text{GPU}_{naive}} \tag{3}$$

Hardware selection

- if $r \gg 1$, GPU(s)
- if $r \ll 1$, CPU(s)
- if $r \approx 1$, CPU(s)-GPU(s) co-running

Problems with Mathematical model and ML solution

Problems

- Complexity: the more varied the hardware devices involved, the more complicated the mathematical model
- Inflexibility: any change of hardware configuration results in a rebuilding of the mathematical model
- Static: cannot capture the run time situation

Solution

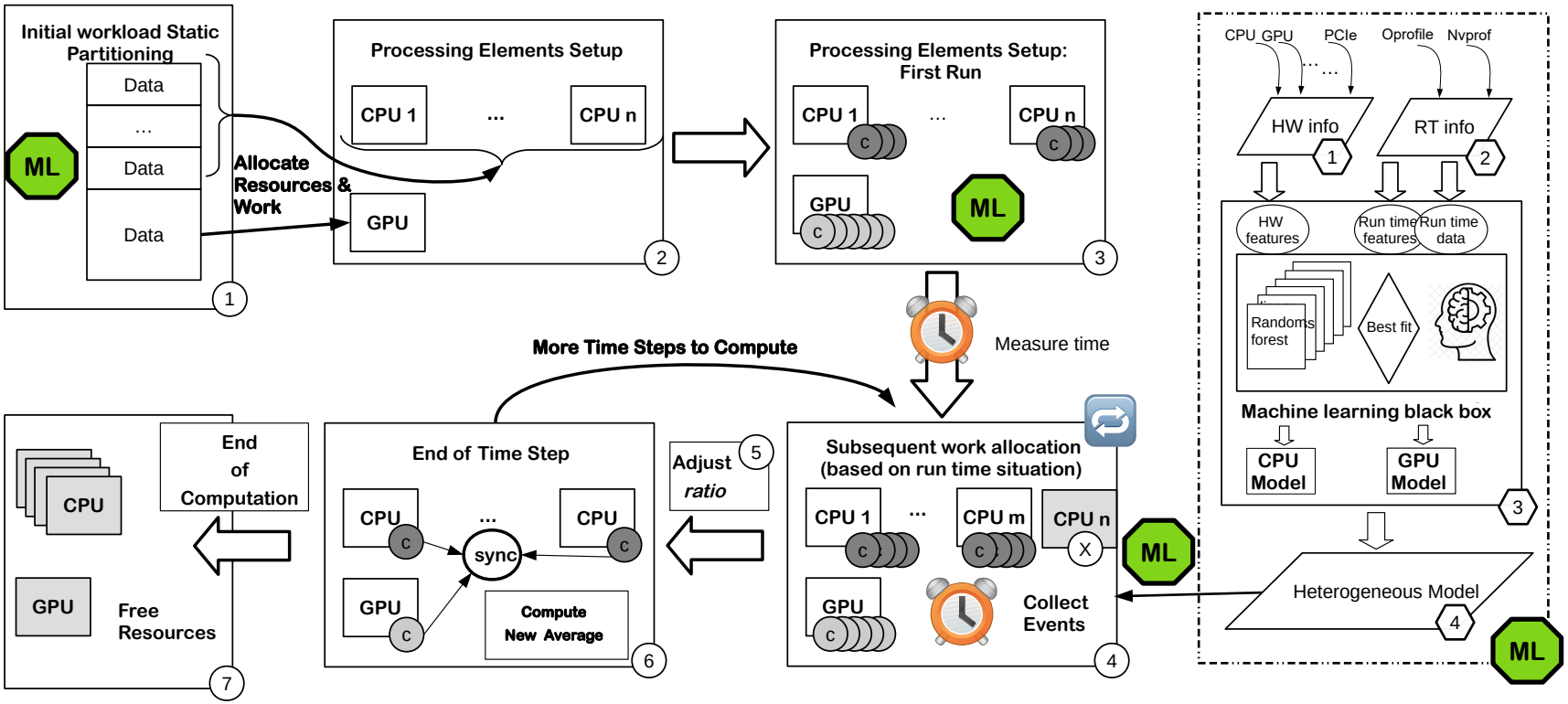
profile-based Machine learning (ML)-assisted Hardware model

Profile-based Machine Learning Estimation Model

- Step 1: Collect hardware architecture information
- Step 2: Collect application's profile information
- Step 3: Pre-process data: normalize data
- Step 4: Correlation and cluster analysis (Hierarchical Agglomerative Clustering algorithm (HAC))
- Step 5: Build a pure CPU and pure GPU profile-based estimation model
- Step 6: Build a heterogeneous prediction model based on the pure CPU and GPU model

Profile-Based Dynamic Adaptive Work-Load (PDAWL) Scheduling

PDAWL scheduling algorithm coupled with Machine Learning. Machine learning occurs in steps 1, 3, and 4 (see the bold polygon ML)



Profile-based Machine Learning Estimation Model - details

ML black box algorithms

- Linear Regressions: (poly, log, exponential, etc):
- Random Forests
- Support Vector Machines (SVM): polynomial and Gaussian kernels.

Overfitting

- 10-fold cross validation
- L2 regularity

Evaluation

- $R^2 = \frac{\text{expected variation}}{\text{total variation}} = [0, 1]$: how well the data fits the model
- MAPE: measures the fitness of ML model
- $|t - static|$: obtains the impact factor of parameters (only for analysis)

Outline

Testbed

Table: Hardware Platforms

Param. / Machines	CPU Parameters					GPU Parameters				PCIe
	Cores	Clock	Socket	L3 Size	Mem	SM	Clock	L2 Size	Mem	
Machine1 (K20)	32	2.6 GHz	2	20 MB	64 GB	13	0.71 GHz	1.25 MB	4.8 GB	6.1 GB/s
Machine2 (K20)	40	3 GHz	2	25 MB	256 GB	13	0.71 GHz	1.25 MB	4.8 GB	6.1 GB/s
Machine3 (k40)	8	3.4 GHz	1	8 MB	16 GB	15	0.75 GHz	1.5 MB	12 GB	10.3 GB/s
Machine4 (Titan)	12	3.4 GHz	1	12 MB	31 GB	14	0.88 GHz	1.5 MB	6 GB	11.5 GB/s

Table: Software Environment

Machines	GCC	CUDA
`Machine1`	v6.2/v8.1	v8.0
`Machine2`	v4.85/v6.2	v8.0
`Machine3`	v5.4	v9.0
`Machine4`	v4.92	v9.1

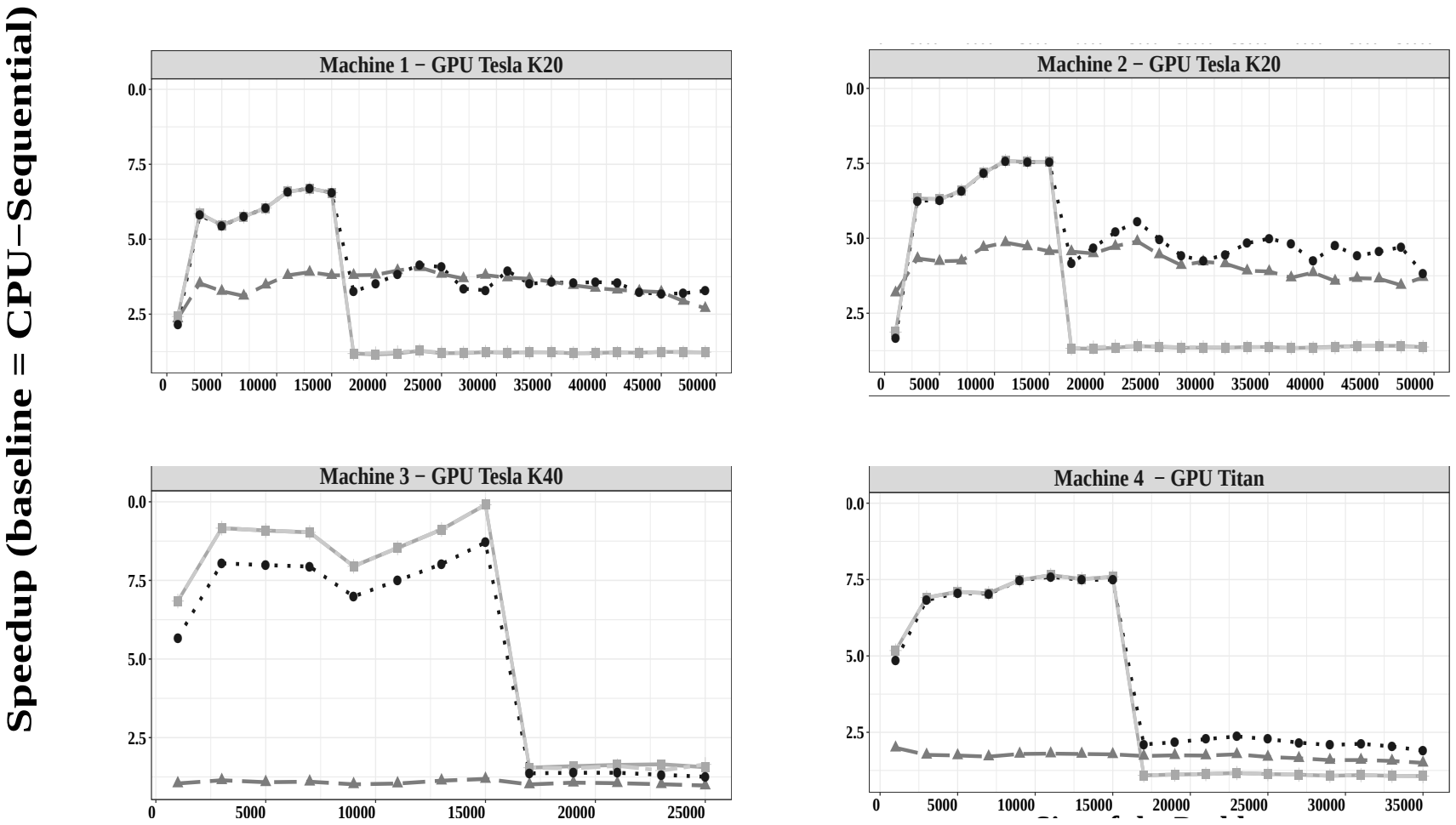
Table: Matrices for SpMV[1]

Name	Dimension	NNZ	μ	σ	cv	MAX
`circuit5M`	5.56 M	59.52 M	10.71	1356.62	126.68	1290501
`eu-2005`	0.86 M	19.24 M	22.30	29.33	1.32	6985
`in-2004`	1.38 M	16.92 M	12.23	37.23	3.04	7753
`FullChip`	2.99 M	26.62 M	8.91	1806.80	202.73	2312481
`kmer_U1a`	67.7 M	138.8 M	2.05	0.37	0.18	35

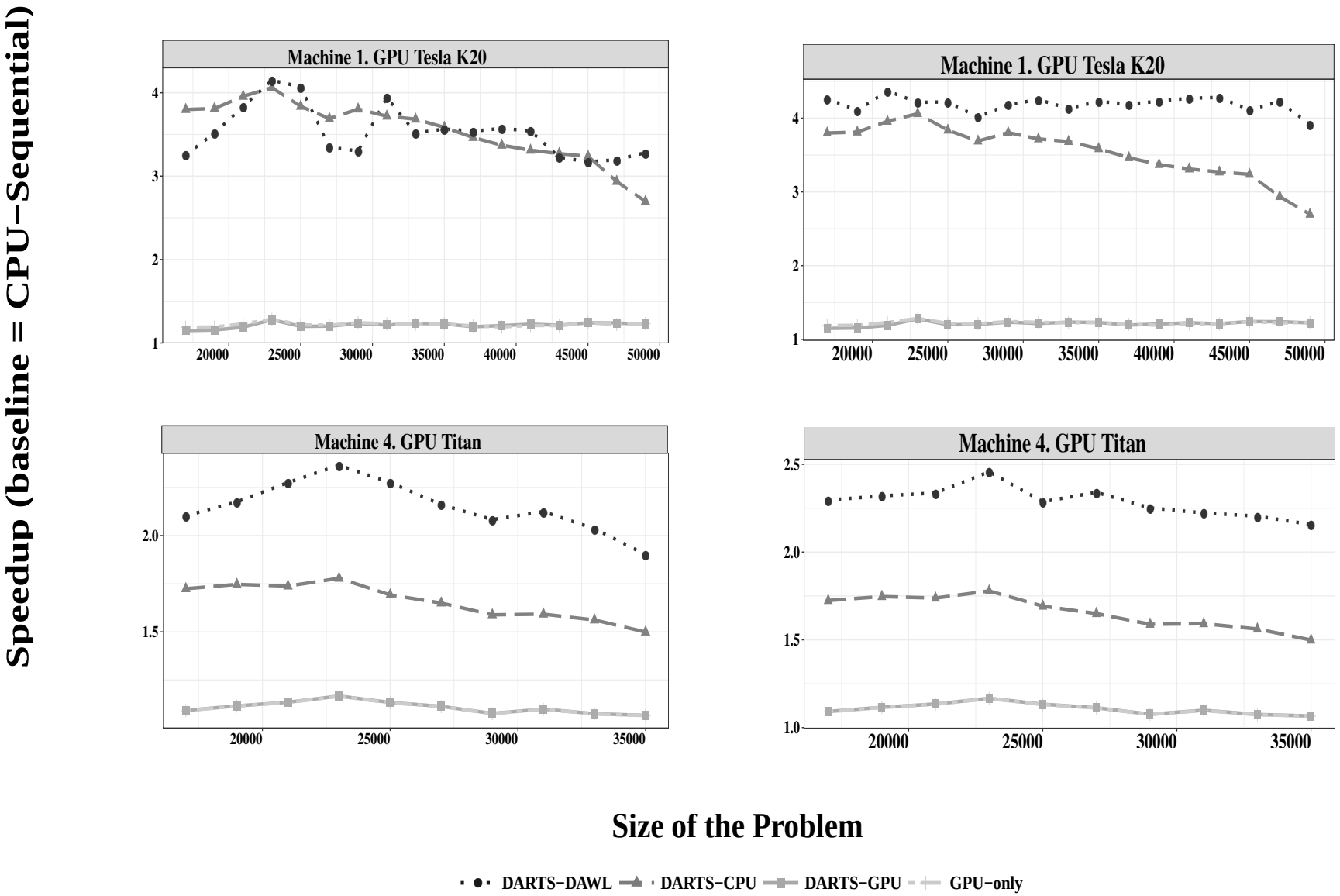
1

[1]Sparse Matrices are from the University of Florida Sparse Matrix Collection(UFSMC)

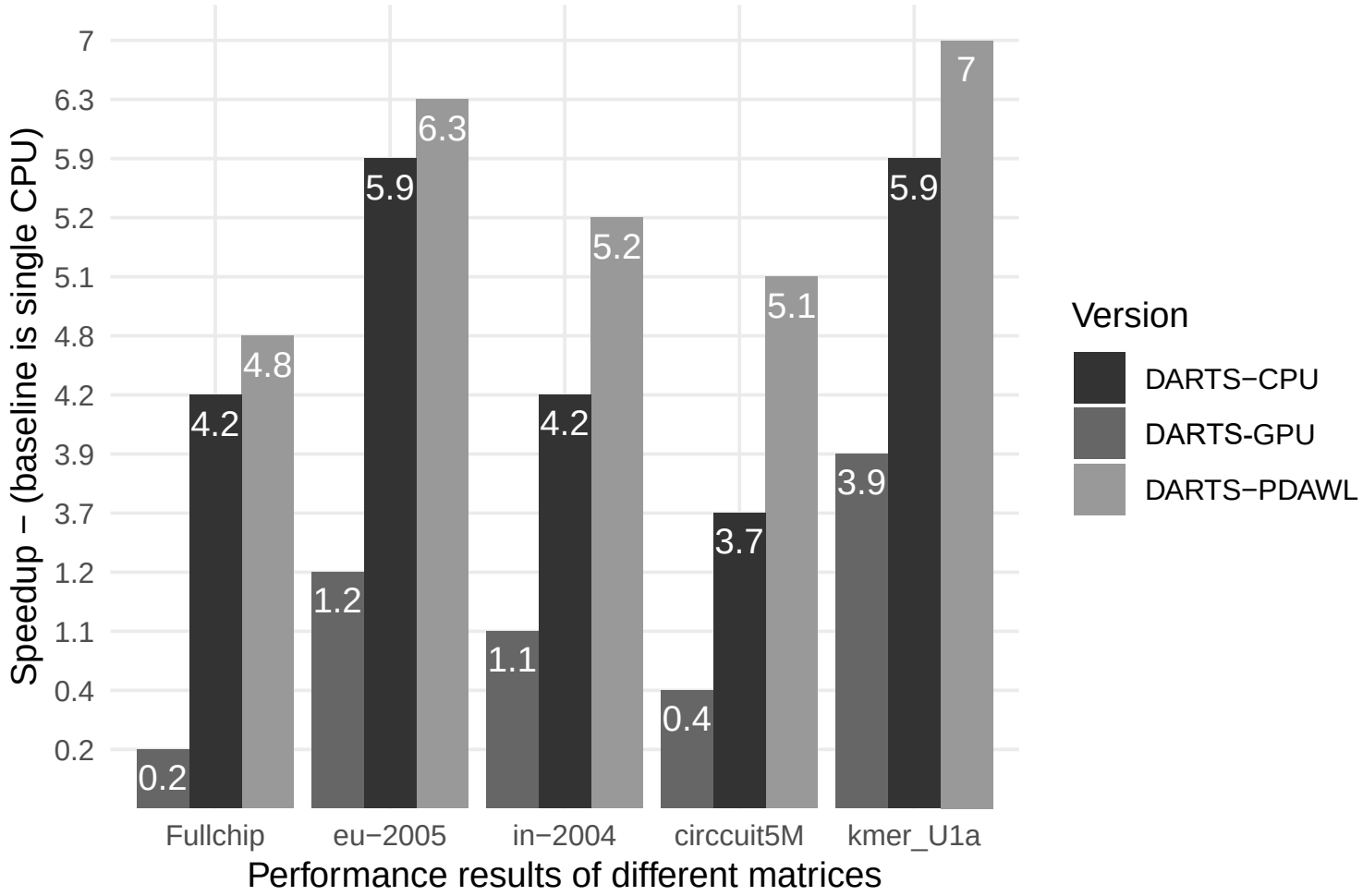
Results: DAWL - Stencil 2D



Results: DAWL VS PDAWL - Stencil 2D



Results: PDAWL - SpMV



Outline

Conclusions and Future work

Conclusion - PDAWL

- A profile-based ML assisted event-driven heterogeneous resource management approach
- Leverage offline ML and online scheduling
- Scientific computing applications: Stencil and SpMV
  - Stencil: 1.6× (pure CPUs), 4.8× (pure GPU)
  - SpMV: 1.37× (pure CPUs), 30.5× (pure GPU)

Future work

- Add ML algorithms, *e.g.*, Deep Learning
- Reduce training time: few-shots/meta learning
- Add more hardware, *e.g.*, FPGAs

Thank you!

Questions!