Expressing **Heterogeneous** Parallelism in **C++** with Intel **Threading Building Blocks**

James Reinders
James Reinders Consulting LLC

Pablo Reble, Jim Cownie
Intel

Rafael Asenjo
Dept. of Computer Architecture
University of Malaga, Spain.

UNIVERSIDAD DE MÁLAGA

Agenda

- Part I: Motivation & Background (30’)
- Part II: TBB heterogeneous features (50’)
- Part III: Heterogen. Flow Graph node (80’)
- Part IV: Templates on top of TBB (20’)

UNIVERSIDAD DE MÁLAGA

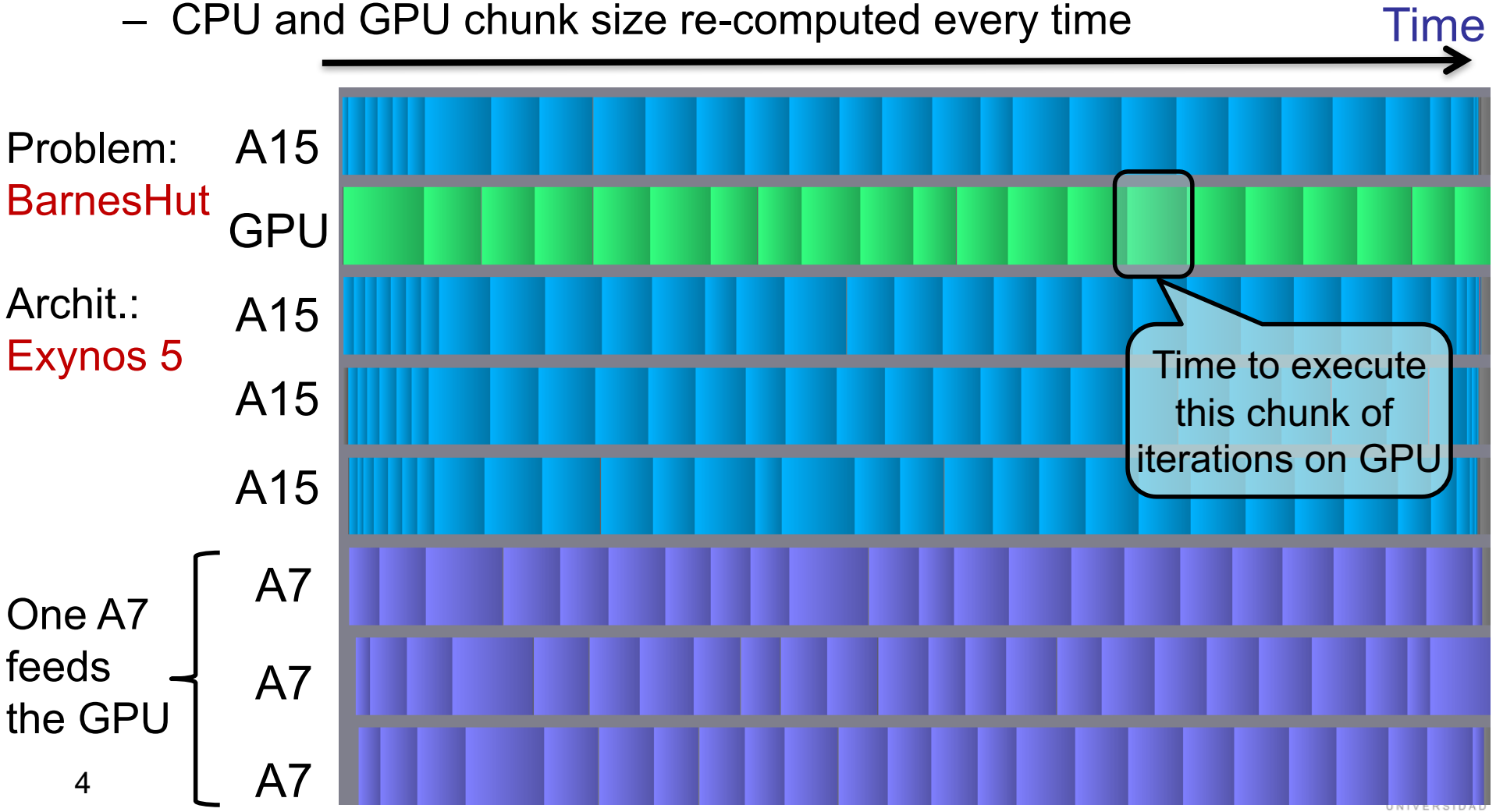
Part IV

- Our heterogeneous parallel_for template based on TBB
  - Results on GPU+CPU
  - Results on FPGA+CPU
- Our heterogeneous parallel pipeline template based on TBB
  - Results on GPU + CPU
  - Results on FPGA + CPU

UNIVERSIDAD DE MÁLAGA

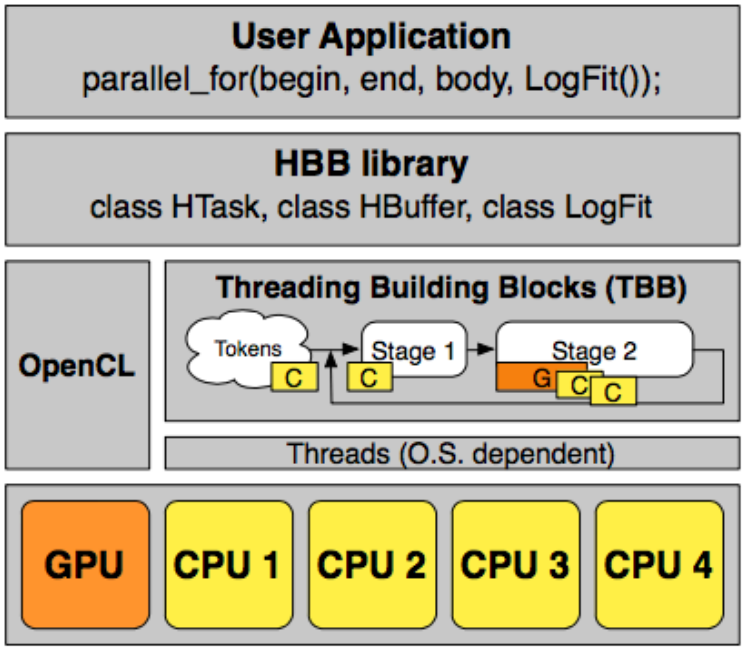
Our work: parallel_for

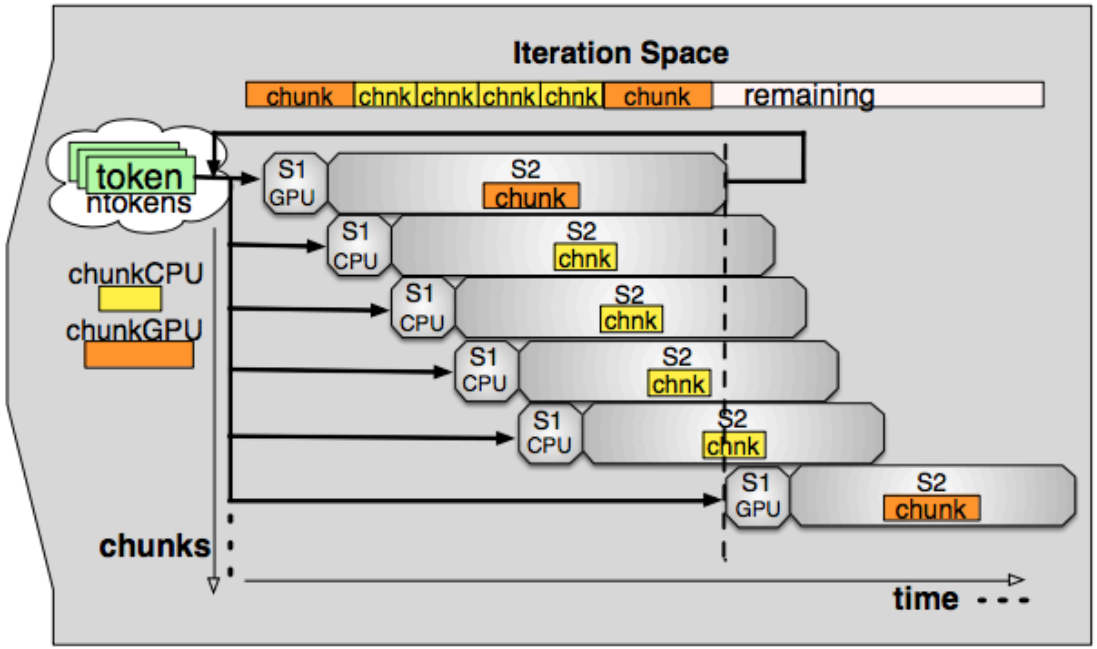
- Heterogeneous parallel for based on TBB
  - Dynamic scheduling / Adaptive partitioning for irregular codes
  - CPU and GPU chunk size re-computed every time



UNIVERSIDAD DE MÁLAGA

New scheduler and API





UNIVERSIDAD DE MÁLAGA

New API

```
1  #include ''parallel_for.h''
2  using namespace hbb;
3  int main(int argc, char* argv[]){
4    // Start task scheduler
5    HInit HInit(numcpus, true);
6    ...
7    KernelInfo k(kernelFile, kernelName);
8    HBuffer<int> * a = new HBuffer<int>(N,USE_ZCB);
9    Body body(k, a);
10   ...
11   parallel_for(begin, end, body, new LogFit());
12   ...
13 }
```

Use the GPU

Exploit Zero-Copy-Buffer

Scheduler class

UNIVERSIDAD DE MÁLAGA

```
1  #include ''HTask.h''
2  #include ''HBuffer.h''
3  using namespace hbb;
4
5  class Body : public HTask{
6     HBuffer<int> * buf_a;
7  public:
8    Body(KernelInfo k, HBuffer * buf_a) : HTask(k){...}
9    void operatorCPU(int begin, int end) {
10      int *a = buf_a->getHostPtr(BUF_RW);
11      for(i=begin; i!=end; i++) a[i] = a[i] + 1;
12   }
13   void operatorGPU() (int begin, int end){
14      //Setting kernel arguments
15      setKernelArg(0, sizeof(int), &begin);
16      setkernelArg(1, sizeof(int), &end);
17      setKernelBuf(2, sizeof(BUF), buf_a->getDevicePtr(BUF_RW));
18      //Launching kernel
19      launchKernel(begin, end);
20   }
21 };
```

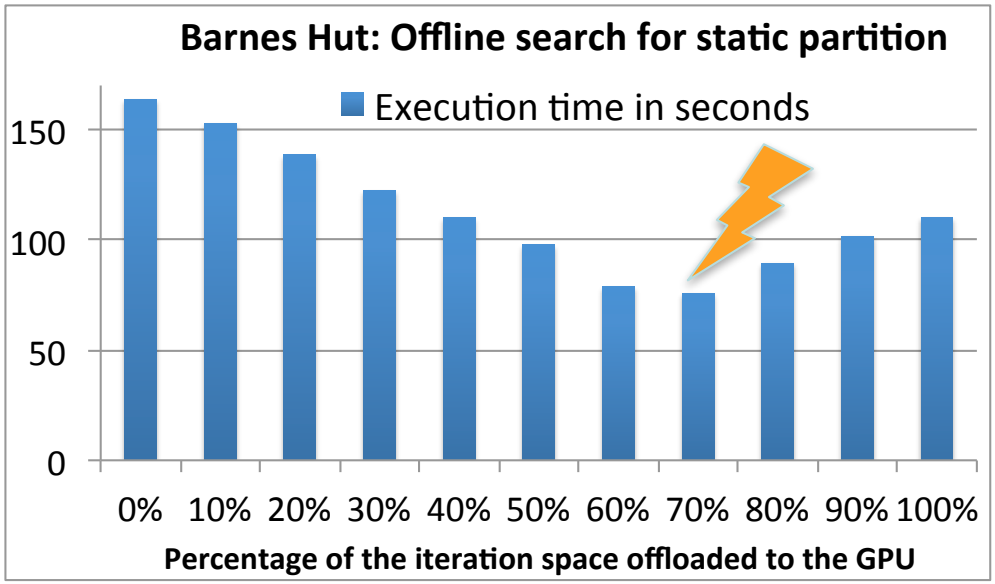

UNIVERSIDAD DE MÁLAGA

Related work: Qilin

- Static Approach: **Bulk-Oracle**

  Offline search (11 executions)

  – Work partitioning between CPU and GPU



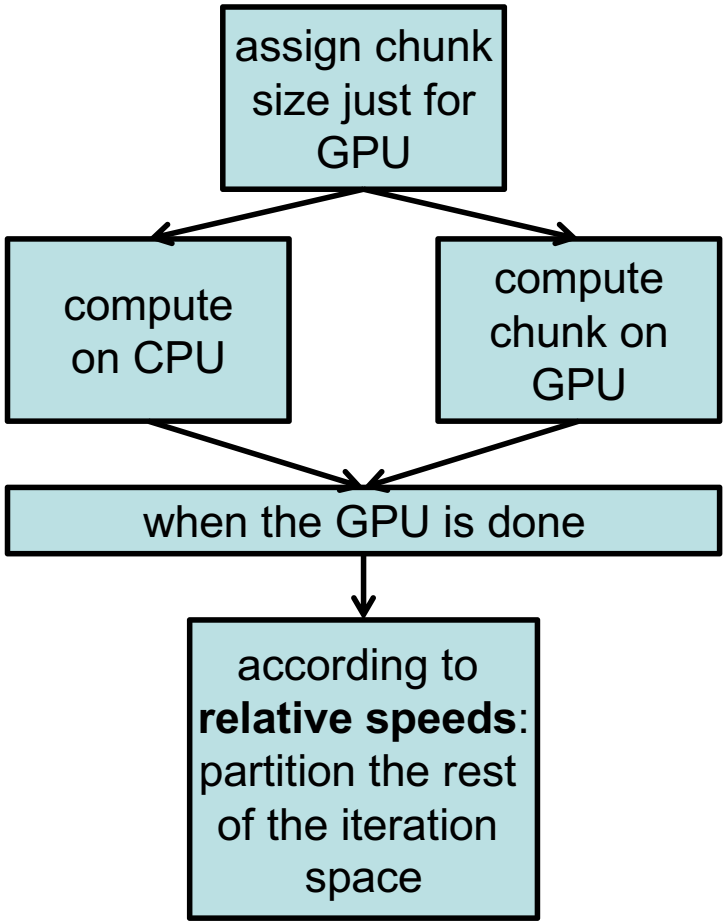
  – One big chunk with 70% of the iterations: on the GPU

  – The remaining 30% processed on the cores

UNIVERSIDAD DE MÁLAGA

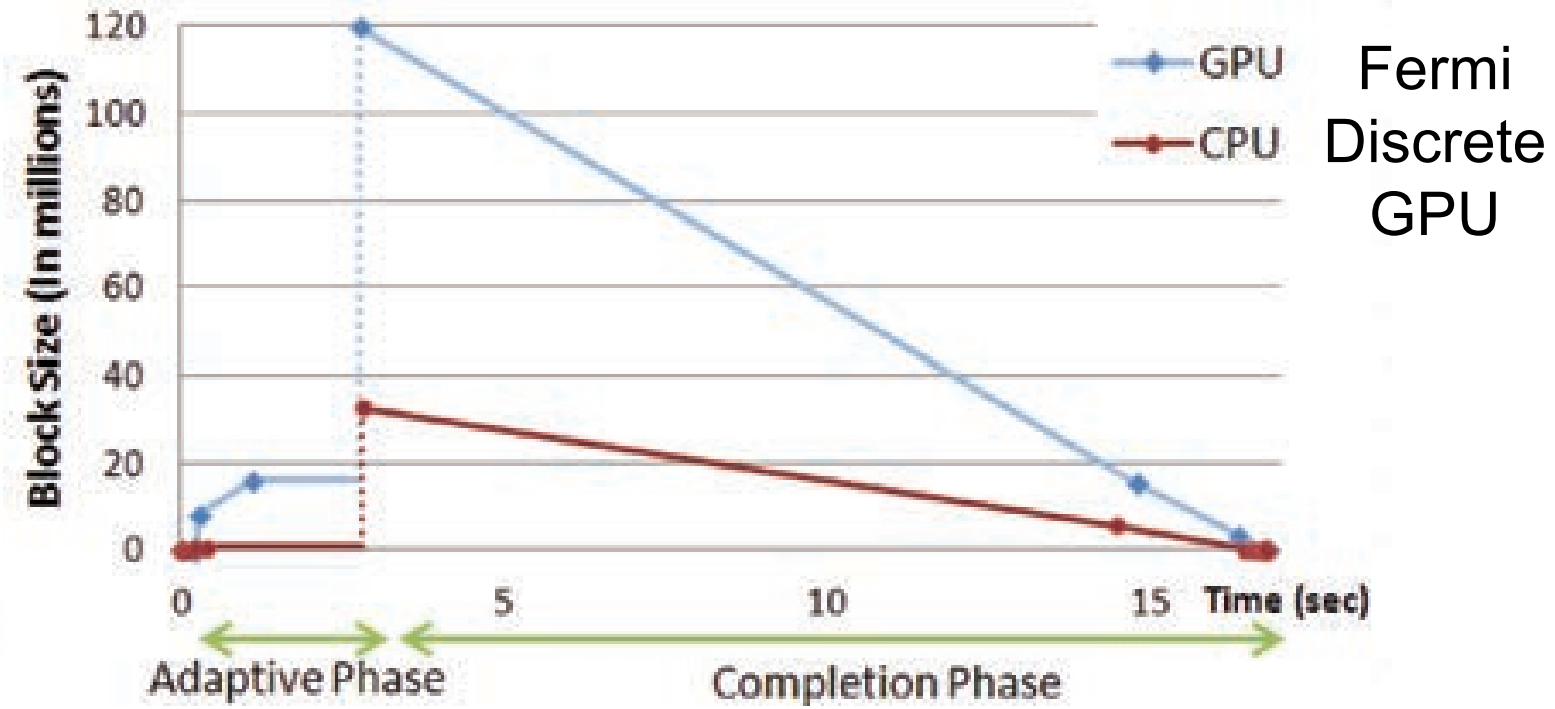
Related work: Intel Concord

- Papers:
  - Rashid Kaleem et al. Adaptive heterogeneous scheduling on integrated GPUs. **PACT 2014**.
  - Naila Farooqui et al. Affinity-aware work-stealing for integrated CPU-GPU processors. **PPoPP '16** (poster).



UNIVERSIDAD DE MÁLAGA

Related Work: HDSS

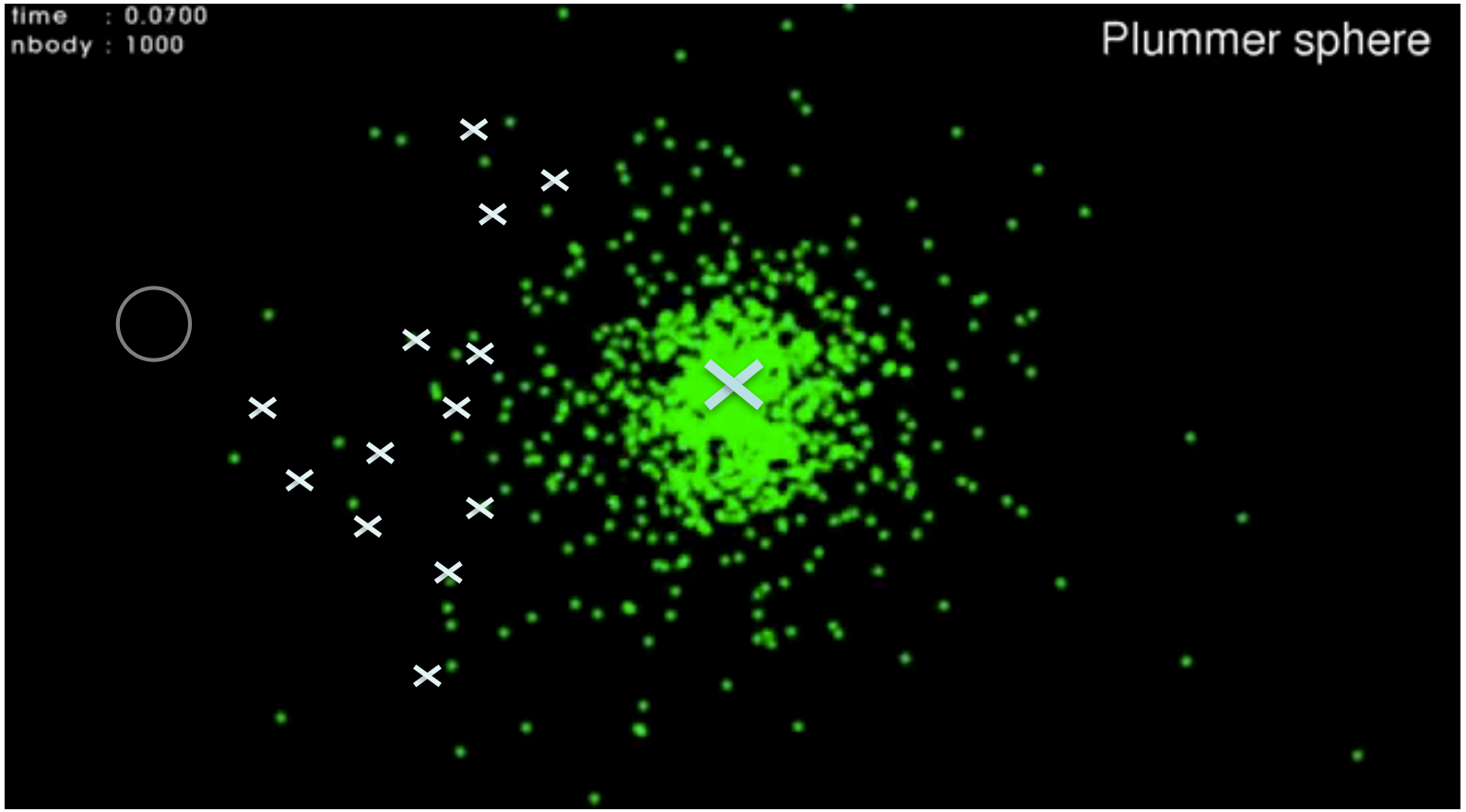


- Belviranli et al, A Dynamic Self-Scheduling Scheme for Heterogeneous Multiprocessor Architectures, **TACO 2013**

UNIVERSIDAD DE MÁLAGA

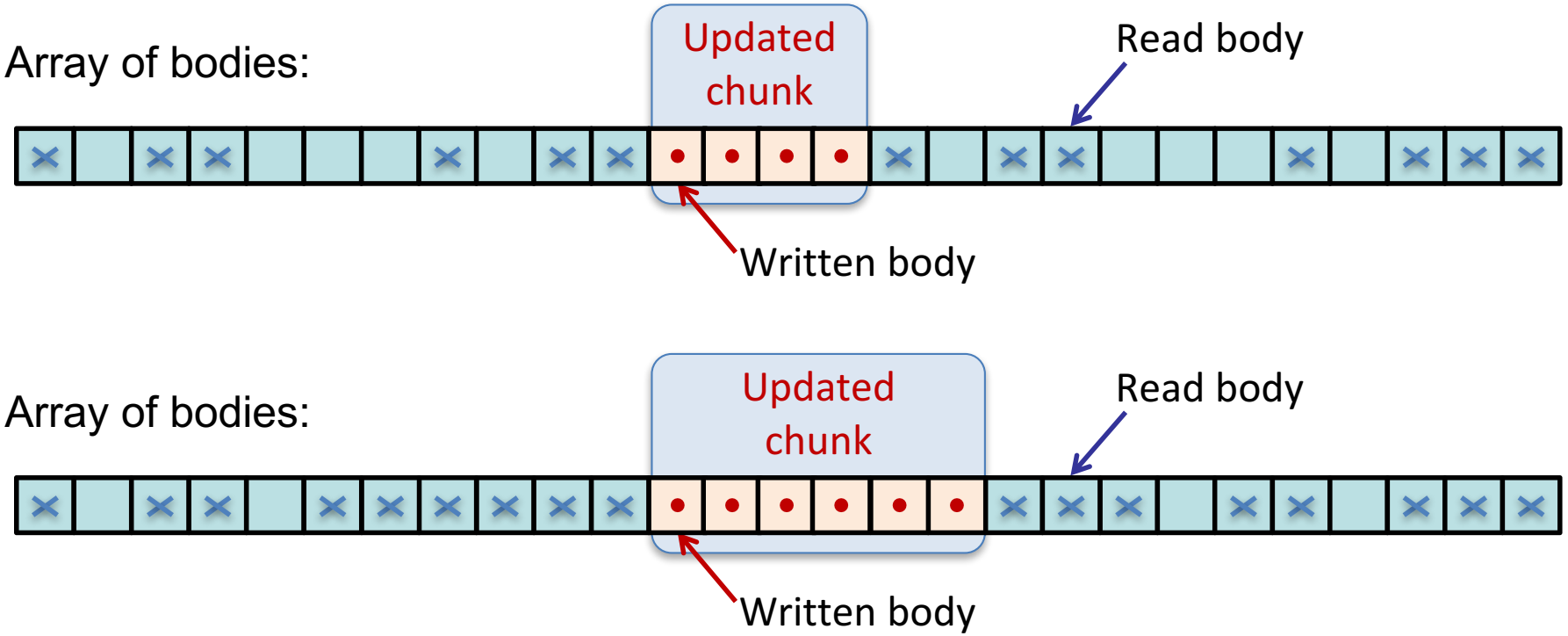
Debunking big chunks

- Example: irregular Barnes-Hut benchmark



UNIVERSIDAD DE MÁLAGA

Debunkign big chunks



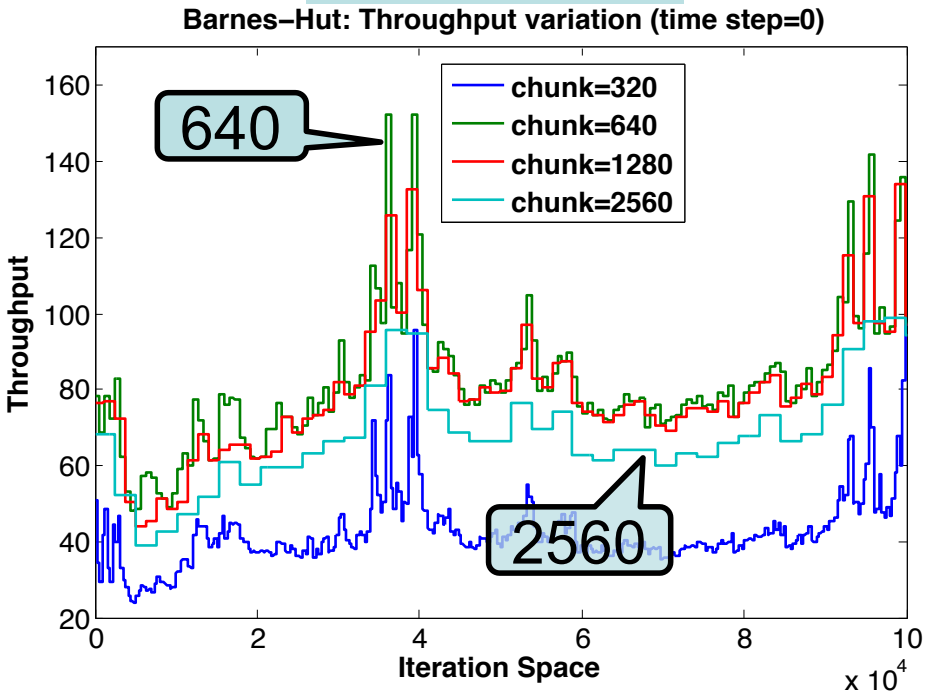
A larger chunk causes more memory traffic

UNIVERSIDAD DE MÁLAGA

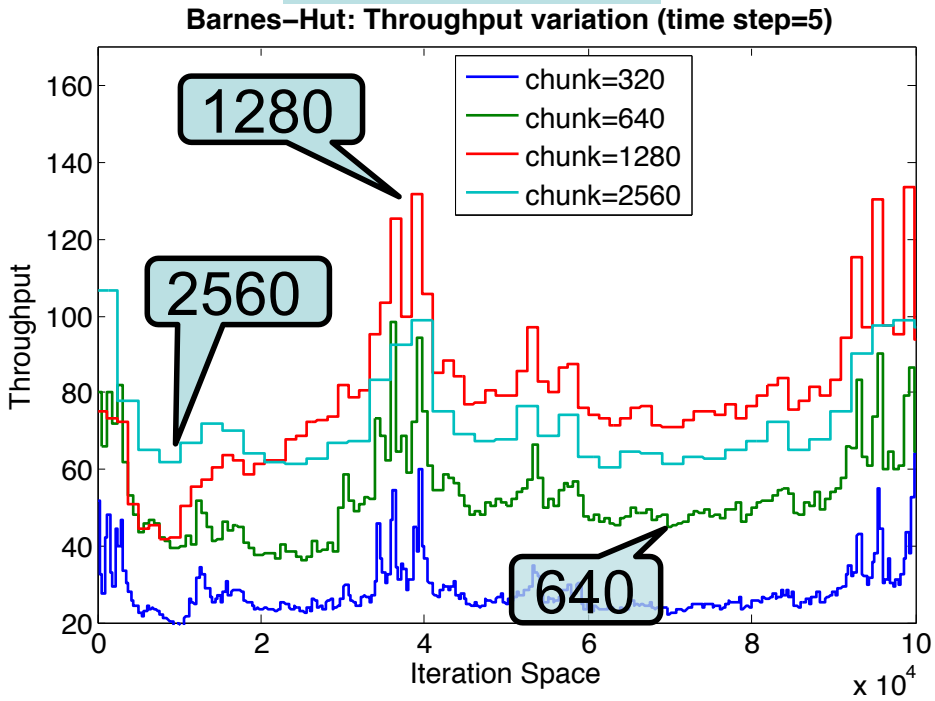
Debunking big chunks

- BarnesHut: GPU throughput along the iteration space
  - For two different time-steps
  - Different GPU chunk-sizes

Time step=0



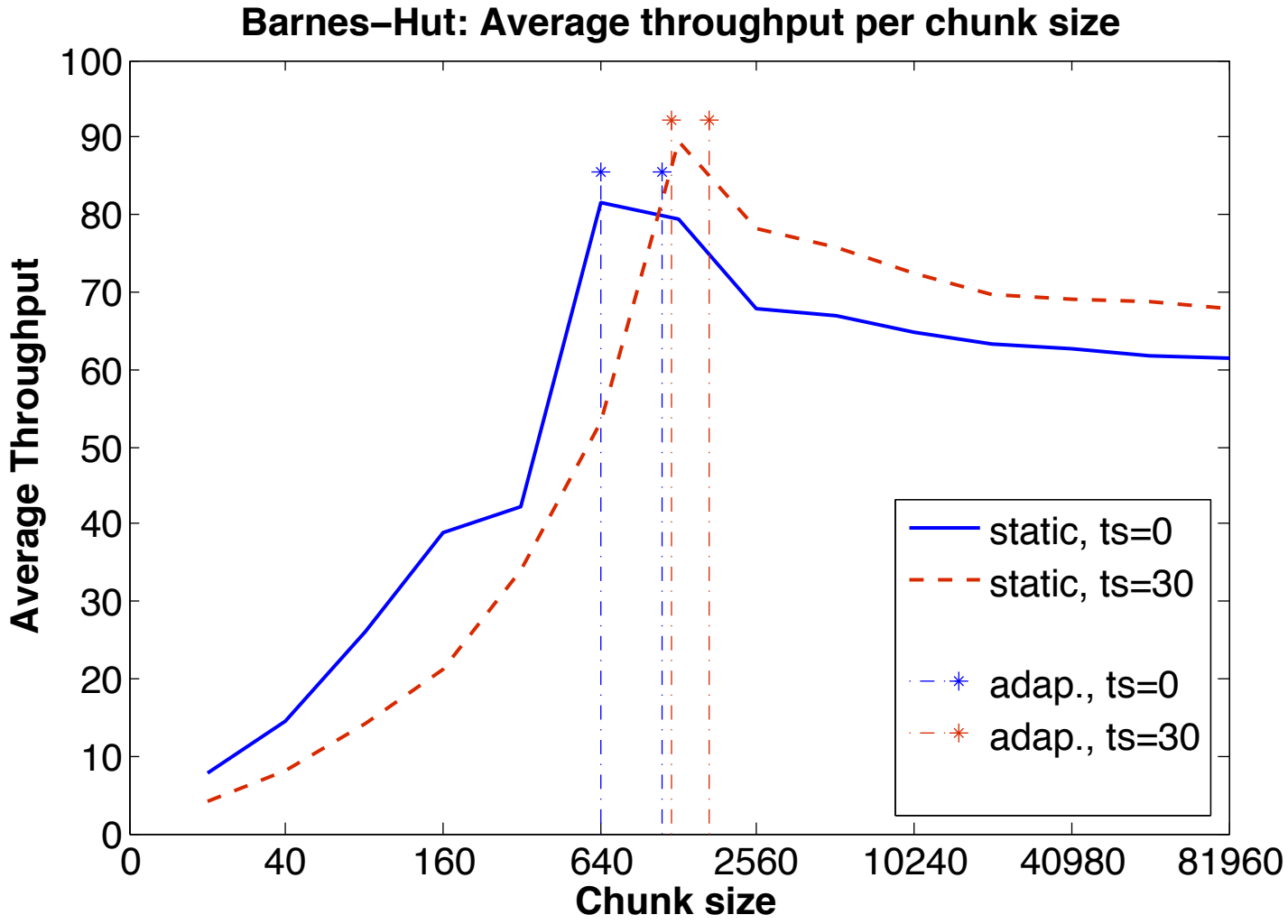
Time step=5



UNIVERSIDAD DE MÁLAGA

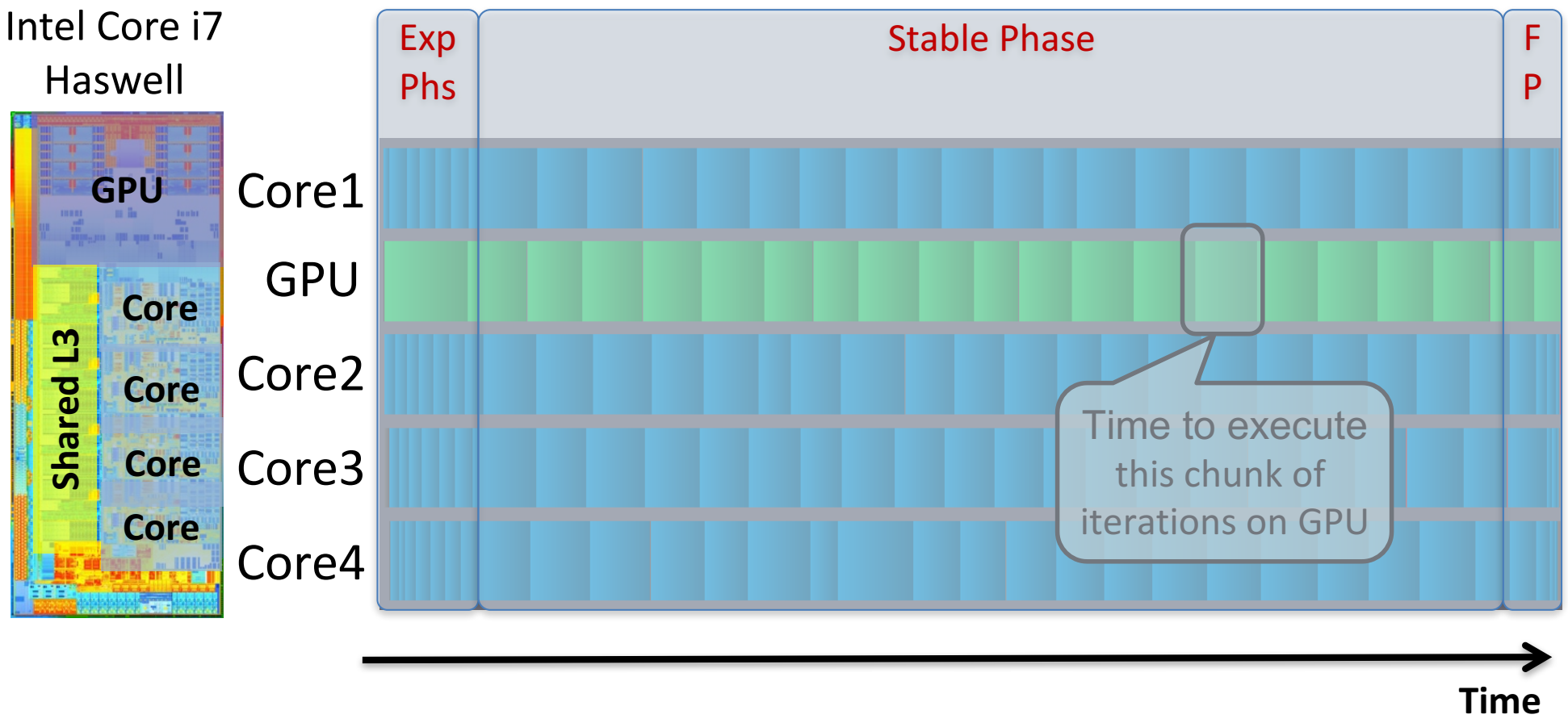
Adaptive GPU block size

- Adapt between time-steps and inside the time-step



UNIVERSIDAD DE MÁLAGA

LogFit main idea

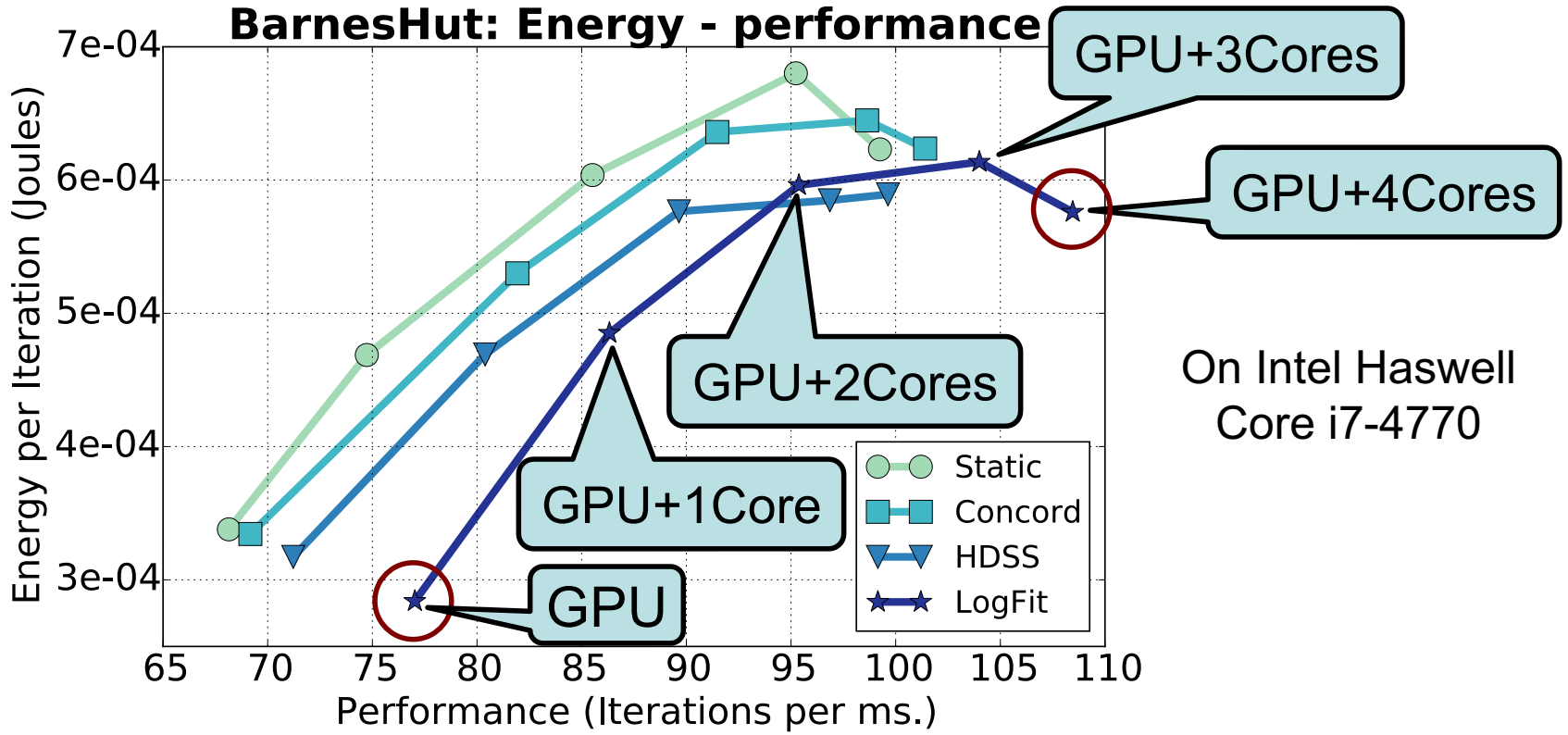


UNIVERSIDAD DE MÁLAGA

Experimental Results

- Test bed:
  - Processor Intel Core i7-4770 (Haswell)
    - 4 Cores at 3.4GHz (without hyperthreading)
    - GPU Intel HD 4600, with 20 execution units at 1.2 GHz
  - Software stack
    - Intel Threading Building Blocks (TBB) 4.2
    - Intel SDK OpenCL 2014
    - Intel C++ Compiler, -O3 optimization flag
    - Intel PCM (Performance Counter Monitors)
    - Windows 7
- Experiments run 10 times, report the average

Parallel_for: Performance per joule

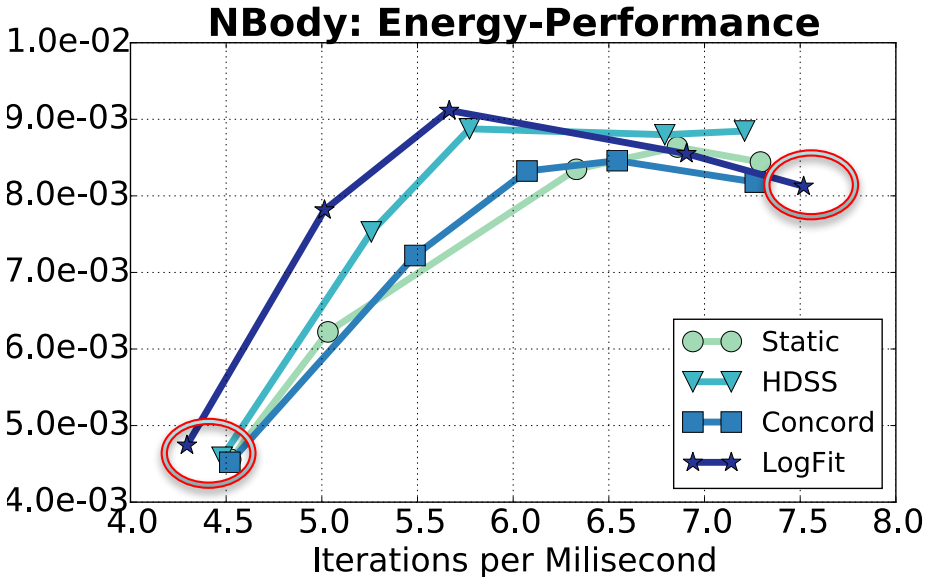


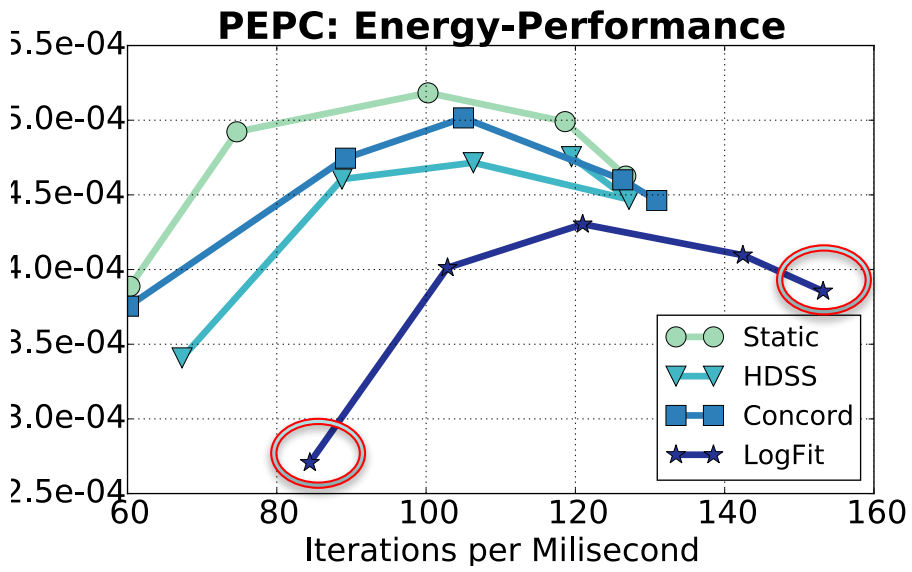
- **Static**: Oracle-Like static partition of the work based on profiling
- **Concord**: Intel approach: GPU size provided by user once
- **HDSS**: Belviranli et al. approach: GPU size computed once
- **LogFit**: our work

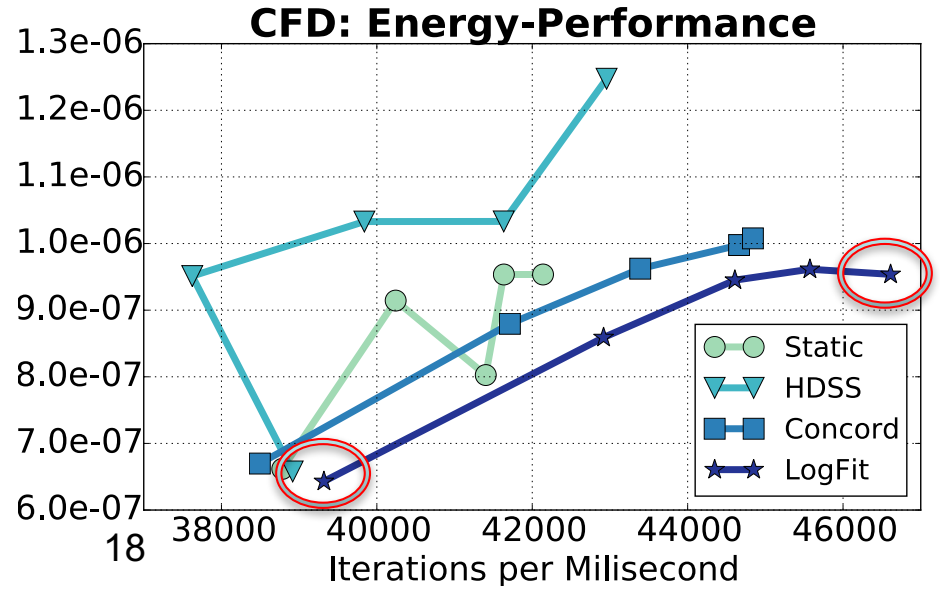
Antonio Vilches et al. *Adaptive Partitioning for Irregular Applications on Heterogeneous CPU-GPU Chips*, **Procedia Computer Science**, 2015.

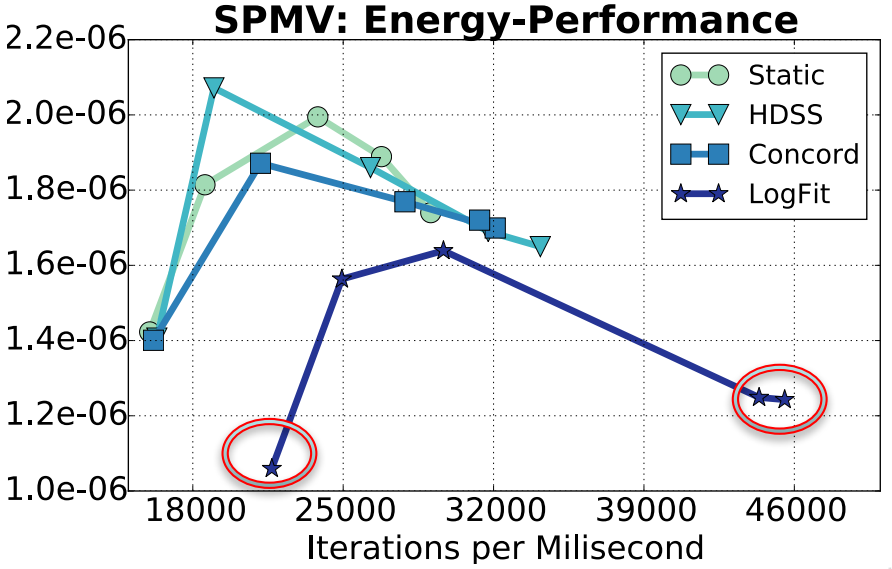
UNIVERSIDAD DE MÁLAGA

LogFit: better for irregular benchmarks



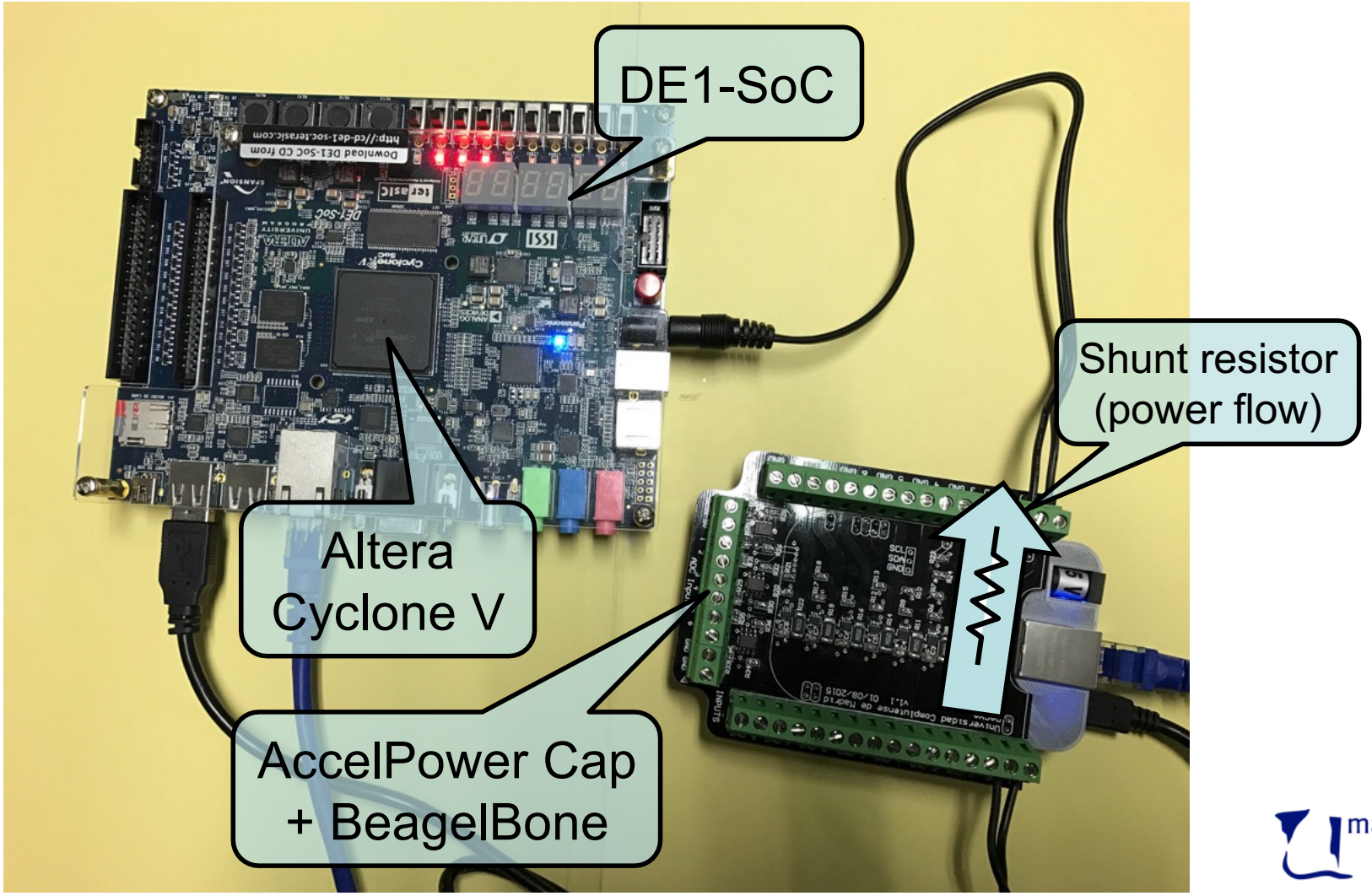






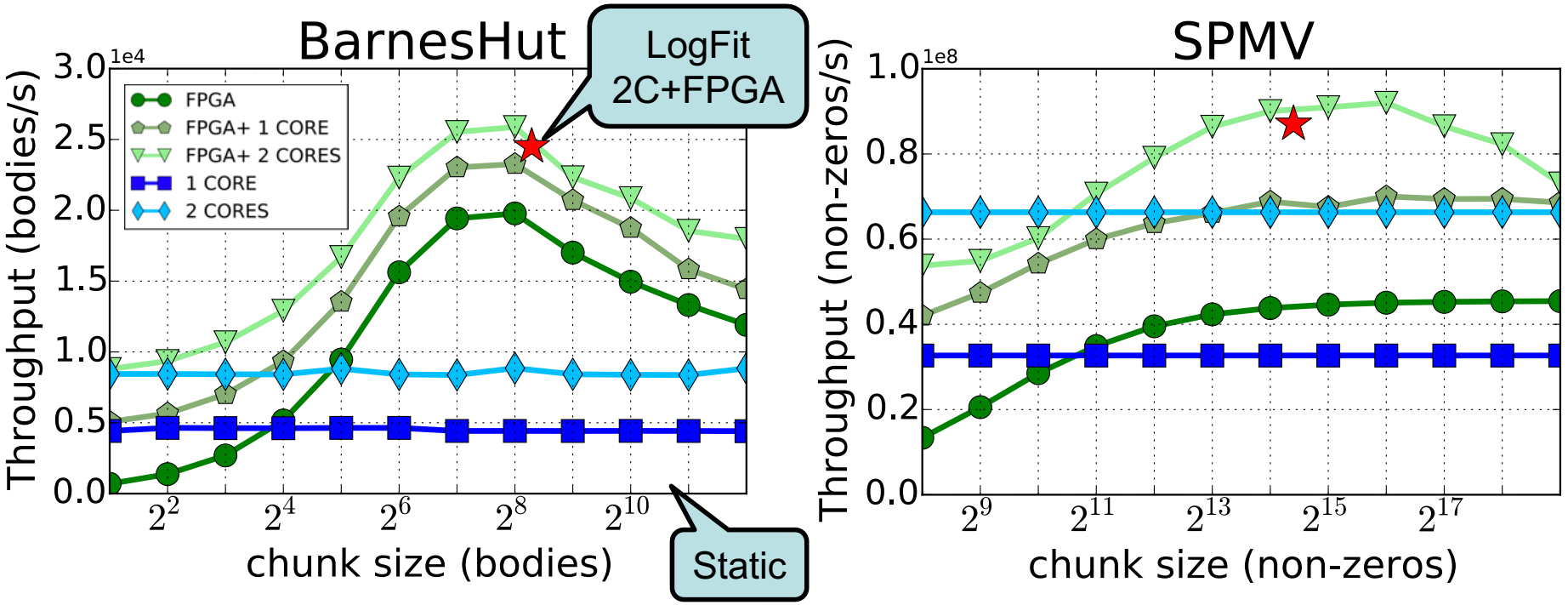
LogFit working on FPGA

- Energy probe: AccelPower Cap (Luis Piñuel, U. Complut.)



UNIVERSIDAD DE MÁLAGA

LogFit working on FPGA



- Terasic DE1: Altera Cyclone V (FPGA+2 Cortex A9)
- Energy probe: AccelPower Cap (Luis Piñuel, U. Complut.)

UNIVERSIDAD DE MÁLAGA

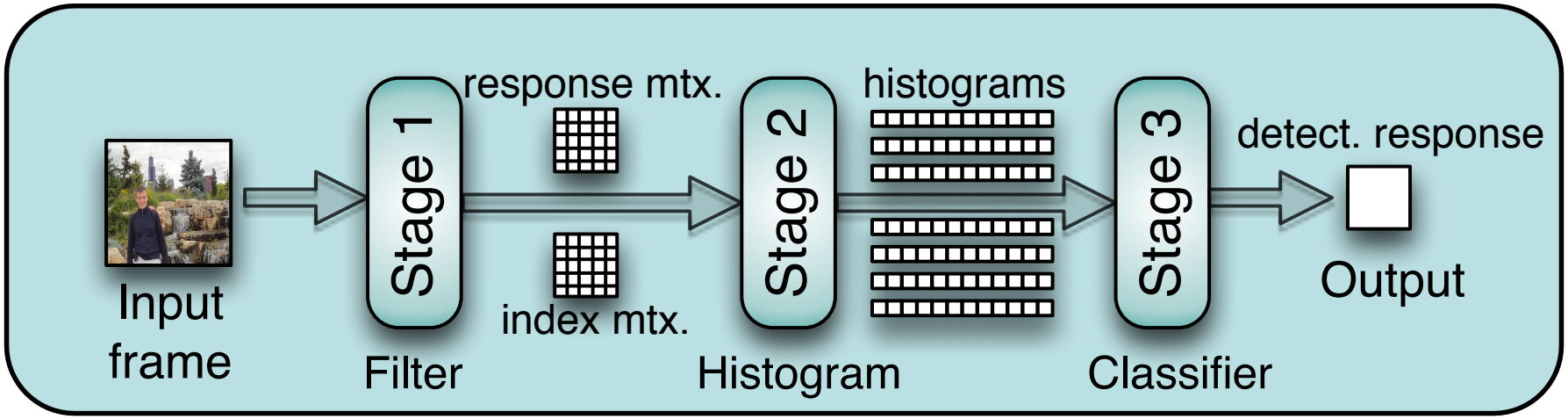
Part IV

- Our heterogeneous parallel_for template based on TBB
  - Results on GPU+CPU
  - Results on FPGA+CPU
- Our heterogeneous parallel pipeline template based on TBB
  - Results on GPU + CPU
  - Results on FPGA + CPU

UNIVERSIDAD DE MÁLAGA

Our work: pipeline

- ViVid, an object detection application
- Contains three main kernels that form a pipeline

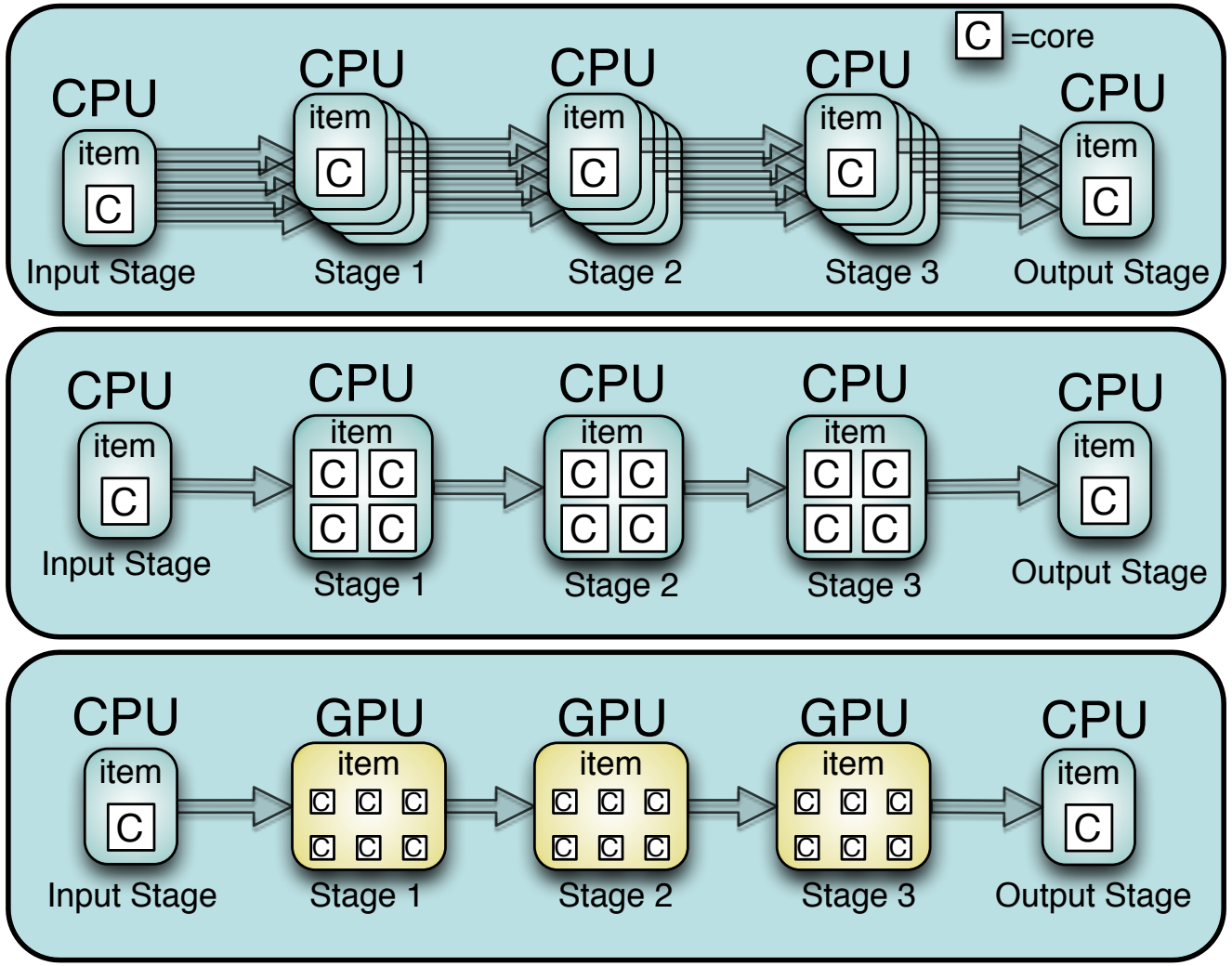


- Would like to answer the following questions:
  - **Granularity**: coarse or fine grained parallelism?
  - **Mapping of stages**: where do we run them (CPU/GPU)?
  - **Number of cores**: how many of them when running on CPU?
  - **Optimum**: what metric do we optimize (time, energy, both)?

UNIVERSIDAD DE MÁLAGA

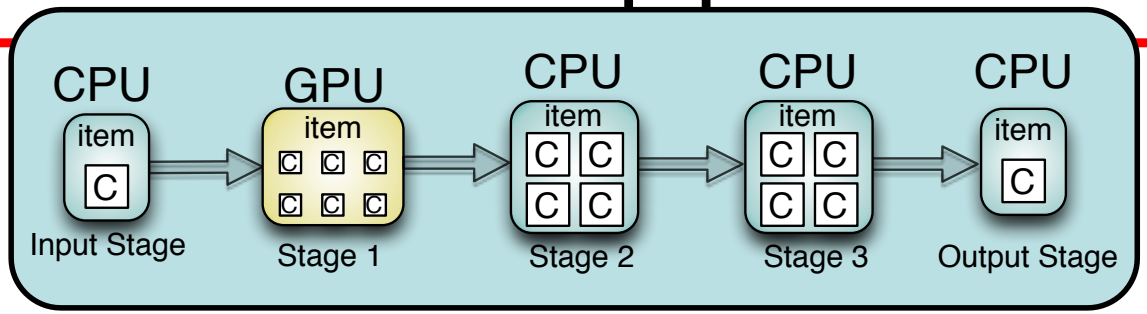
Granularity

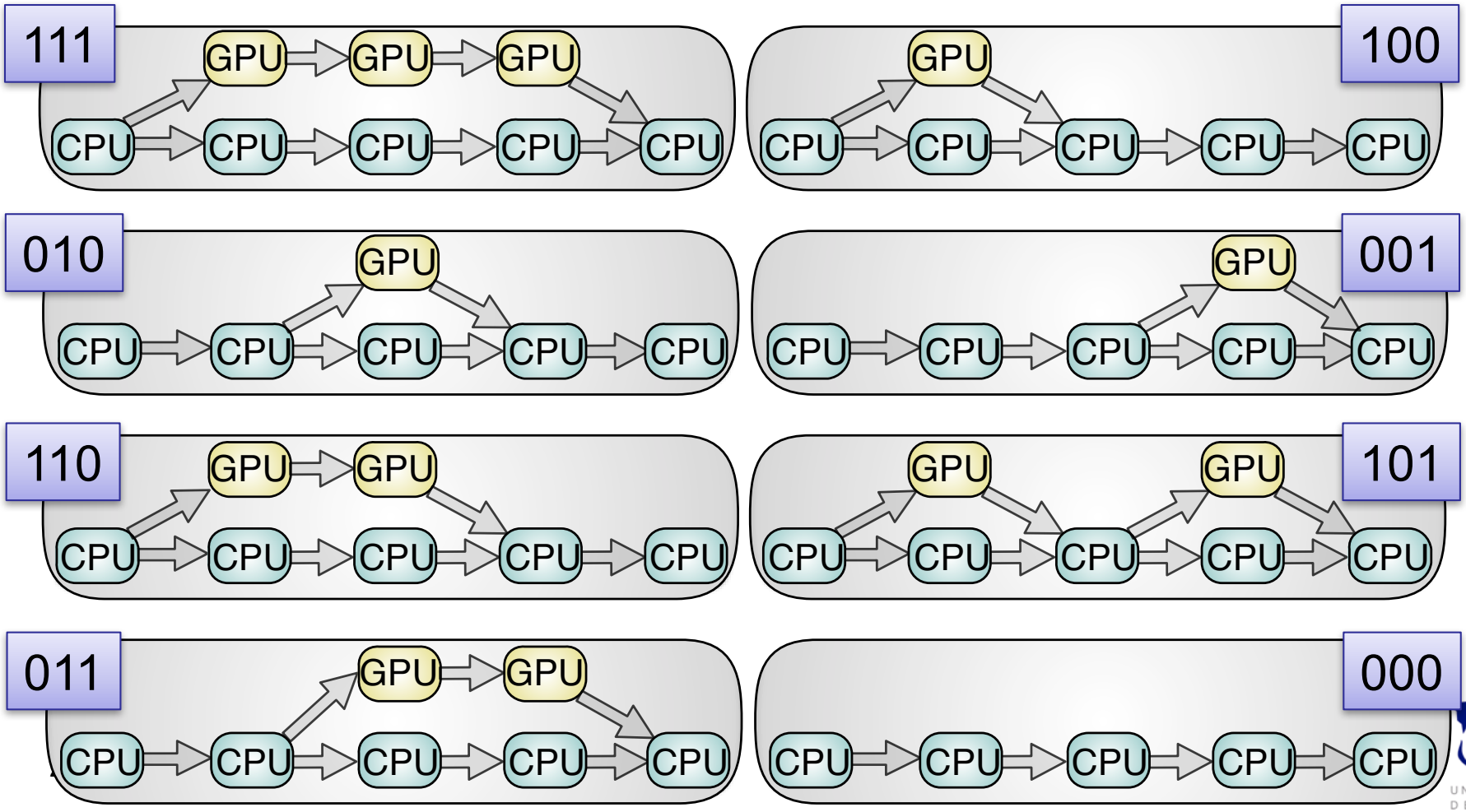
- Coarse grain:
  - CG
- Medium grain:
  - MG
- Fine grain:
  - Fine grain also in the CPU via AVX intrinsics



UNIVERSIDAD DE MÁLAGA

Our work: pipeline

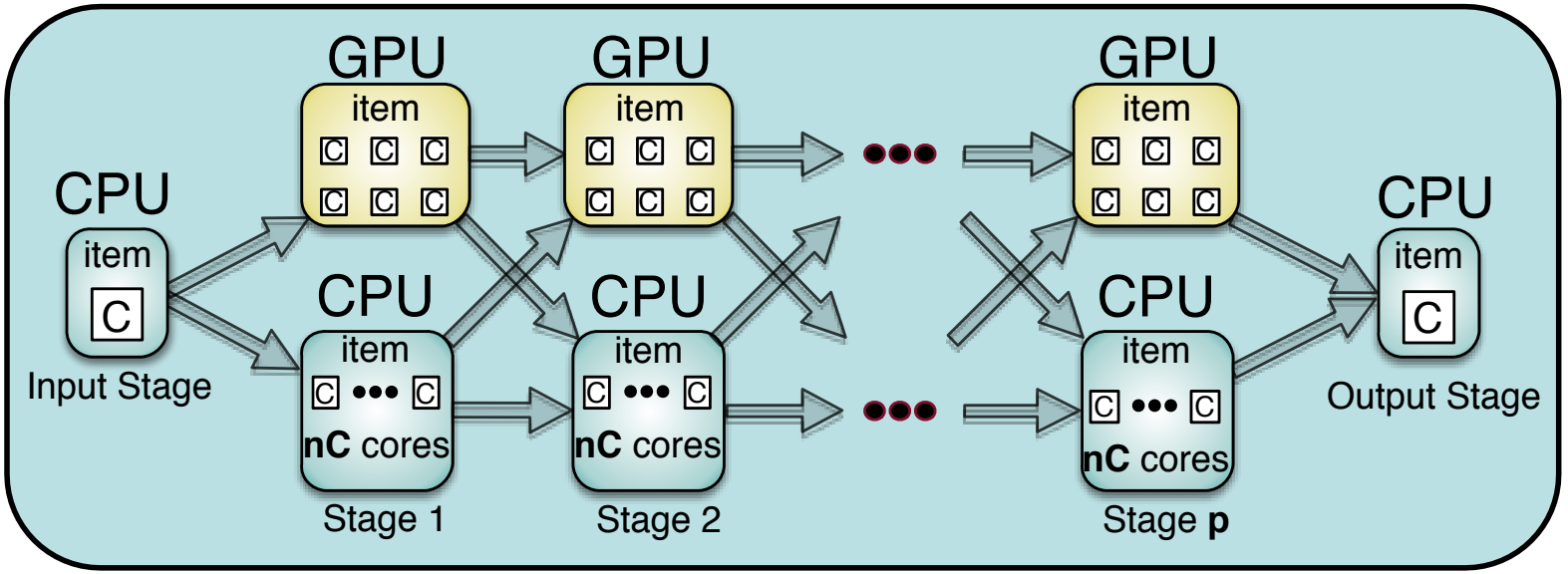




UNIVERSIDAD DE MÁLAGA

Accounting for all alternatives

- In general: **nC** CPU cores, **1** GPU and **p** pipeline stages



$$\#\ \text{alternatives} = 2^{p} \times (nC + 2)$$

- For Rodinia's SRAD benchmark (p=6,nC=4) → **384** alternatives

Mapping streaming applications on commodity multi-CPU and GPU on-chip processors, **IEEE Tran. Parallel and Distributed Systems**, 2016.

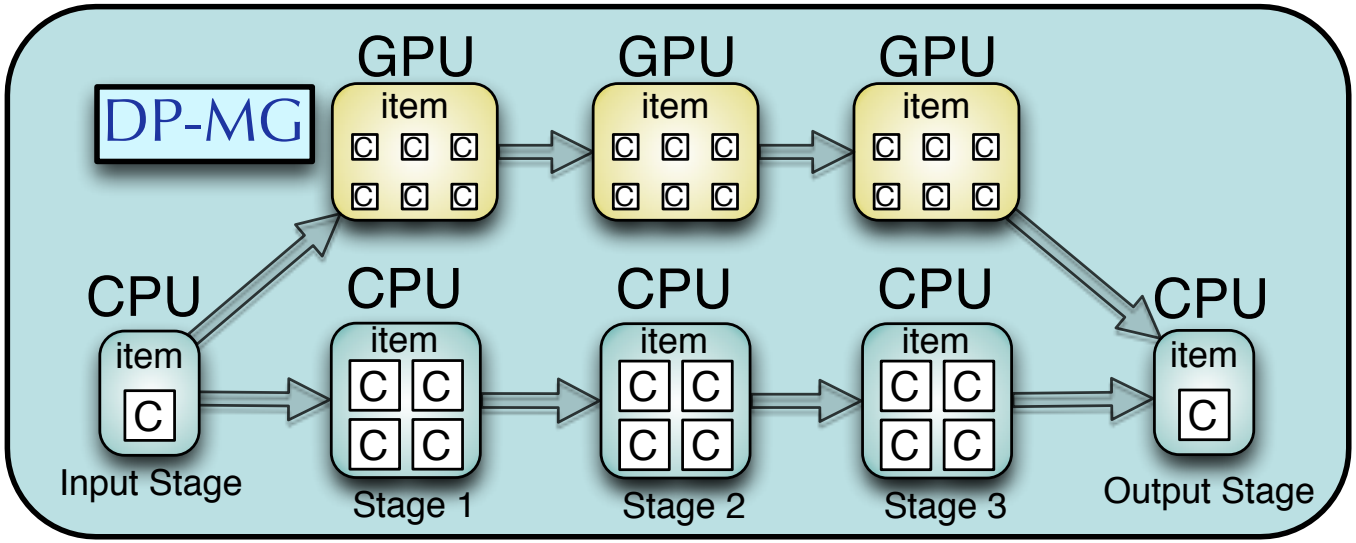
UNIVERSIDAD DE MÁLAGA

Framework and Model

- Key idea:
  1. Run only on GPU
  2. Run only on CPU

  } collect λ and E (homogeneous values)

  3. Analytically extrapolate for heterogeneous execution
  4. Find out the best configuration → RUN

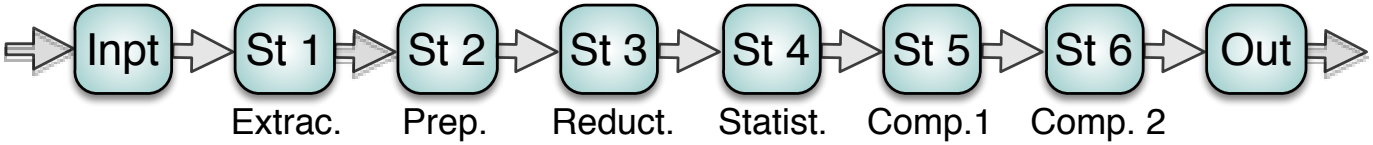


UNIVERSIDAD DE MÁLAGA

Environmental Setup: Benchmarks

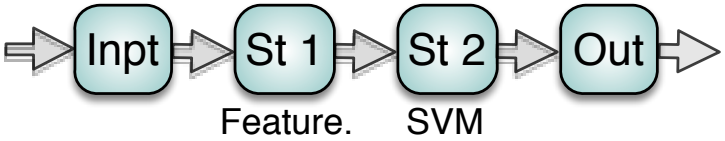
- Four Benchmarks
  - ViVid (Low and High Definition inputs)

  - SRAD



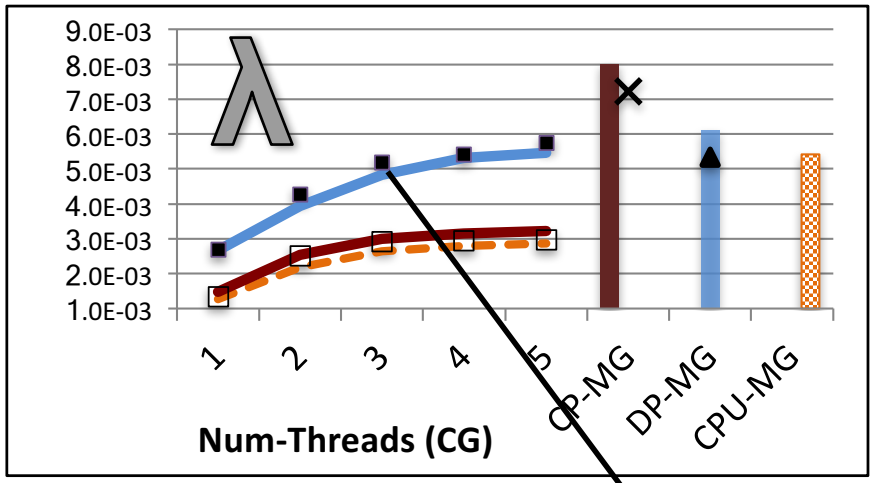
  - Tracking

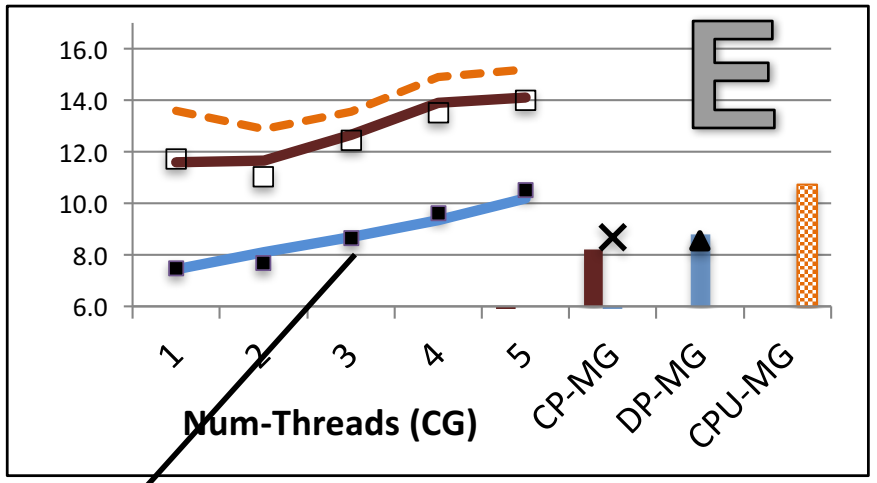
  - Scene Recognition



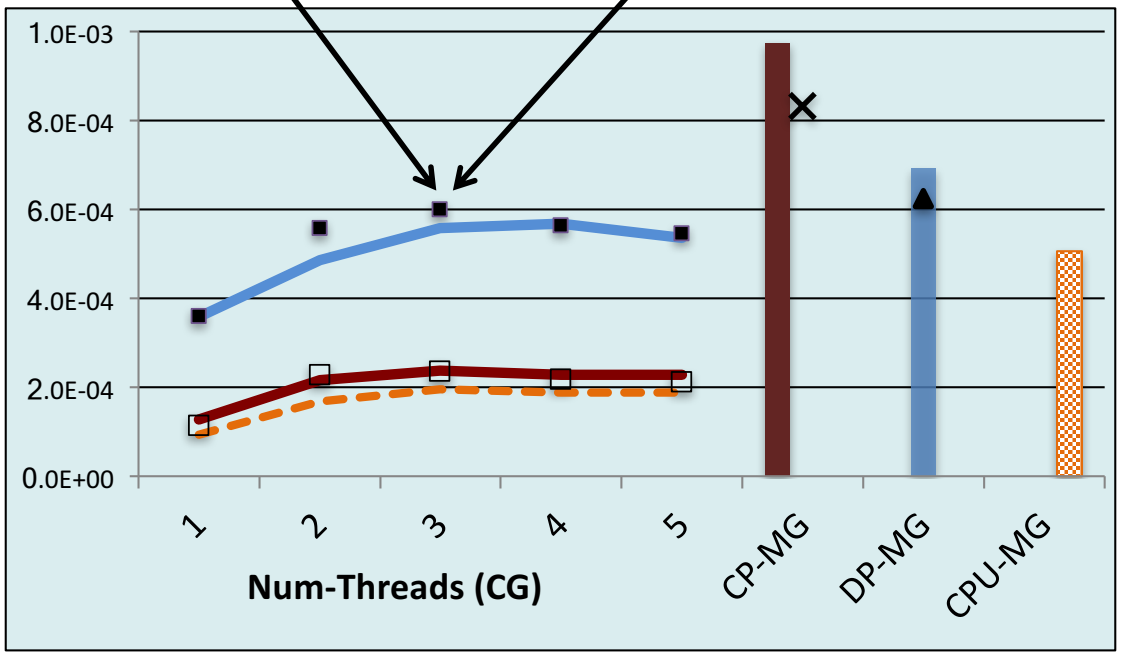
UNIVERSIDAD DE MÁLAGA

Does Higher Throughput imply Lower Energy?





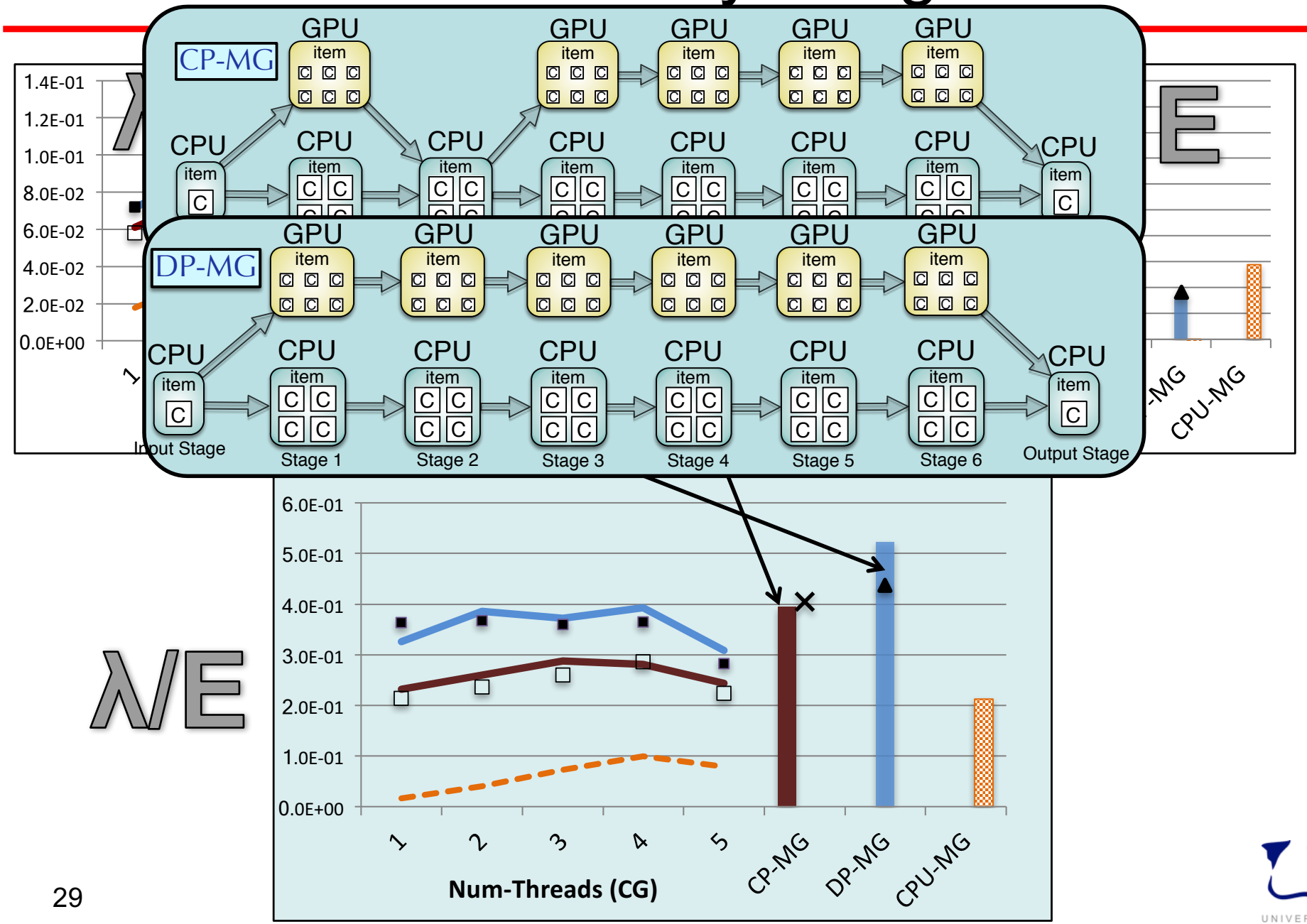
λ/E



Haswell
HD

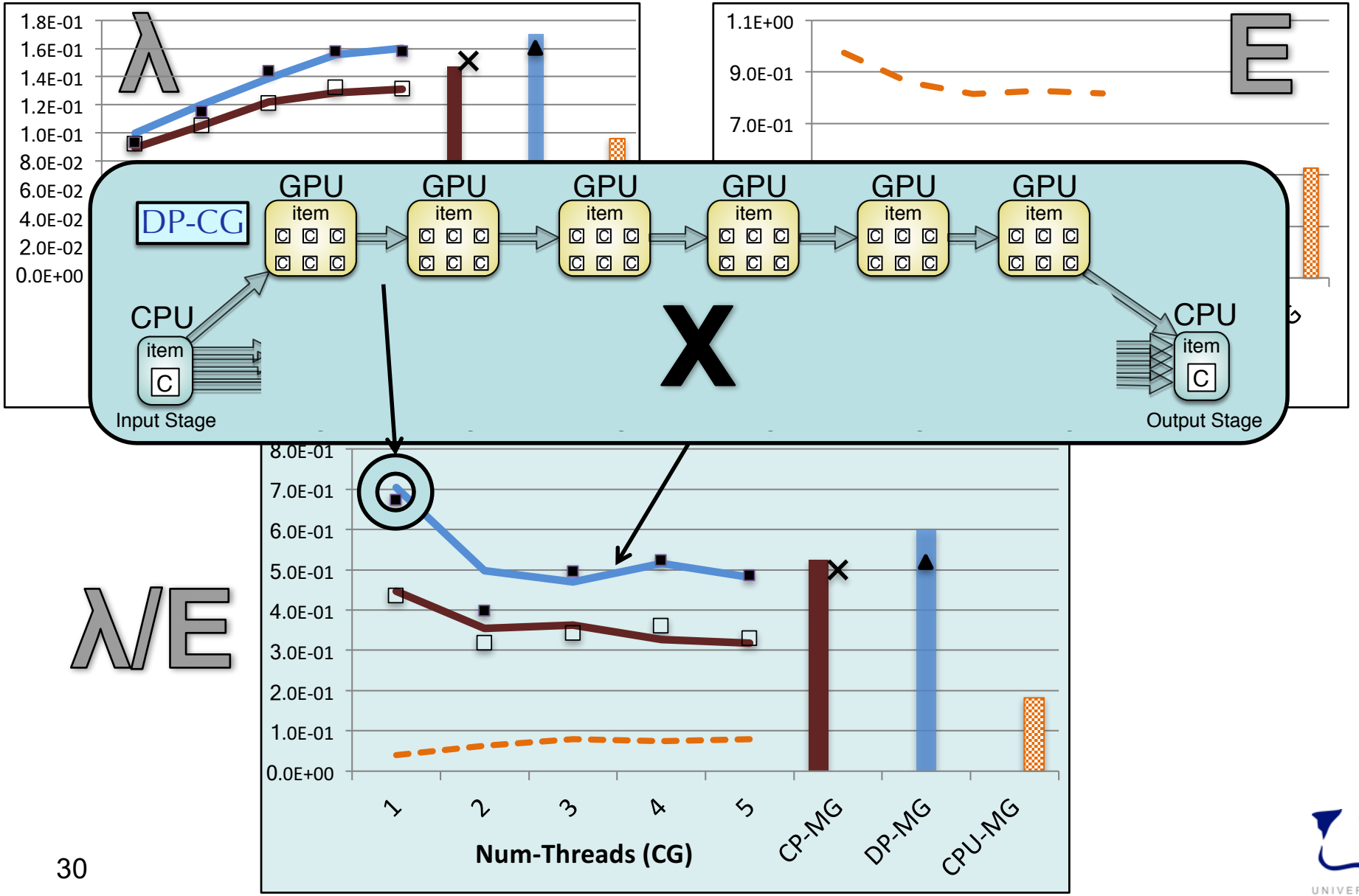
UNIVERSIDAD DE MÁLAGA

SRAD on Ivy Bridge



UNIVERSIDAD DE MÁLAGA

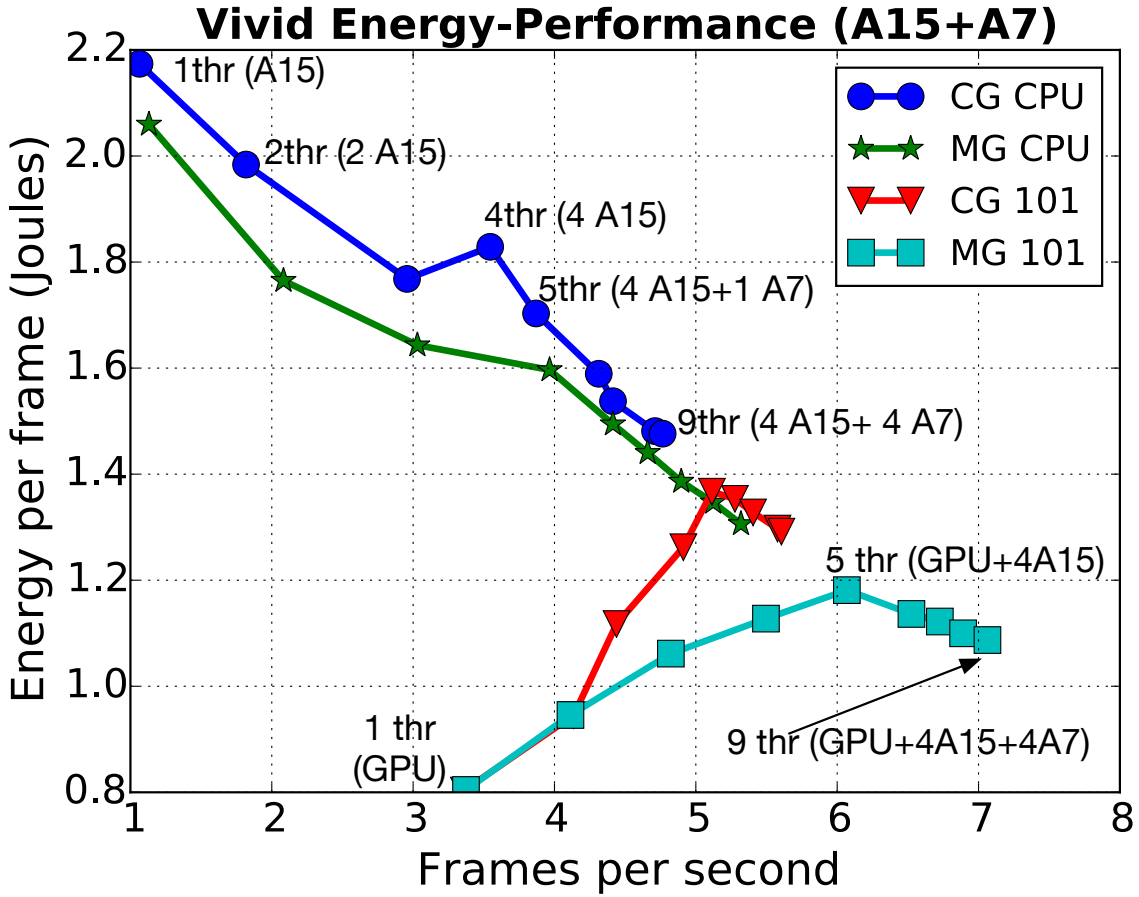
SRAD on Haswell

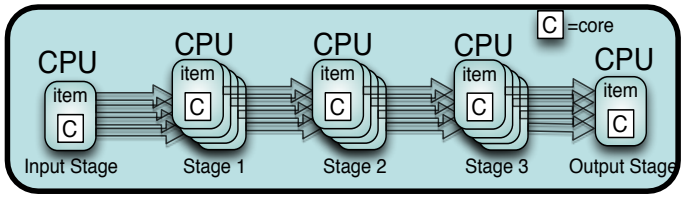


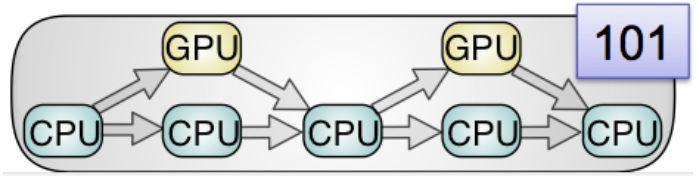
UNIVERSIDAD DE MÁLAGA

Results on big.LITTLE architecture

- On Odroid XU3: 4 Cortex A15 + 4 Cortex A7 + GPU Mali







UNIVERSIDAD DE MÁLAGA

Pipeline on CPU-FPGAs

- Change the GPU by a FPGA
  - Core i7 4770k 3.5 GHz Haswell quad-core CPU
  - Altera DE5-Net FPGA from Terasic
  - Intel HD6000 embedded GPU
  - Nvidia Quadro K600 discrete GPU (192 cuda cores)

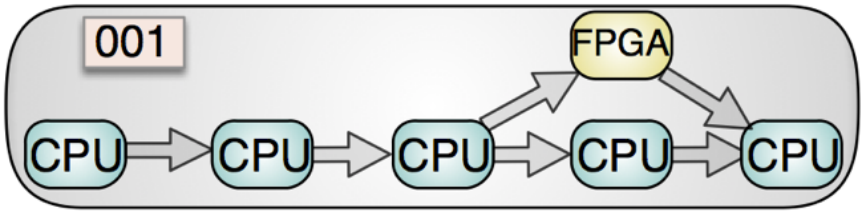
- Homogeneous results:

Device	Time (sec.)	Throughput (fps)	Power (watts)	Energy (Jules)
CPU (1 core)	167.610	0.596	37	6201
CPU (4 cores)	50.310	1.987	92	4628
FPGA	13.480	7.418	**5**	**67**
On-chip GPU	**7.855**	**12.730**	16	125
Discrete GPU	13.941	7.173	47	655

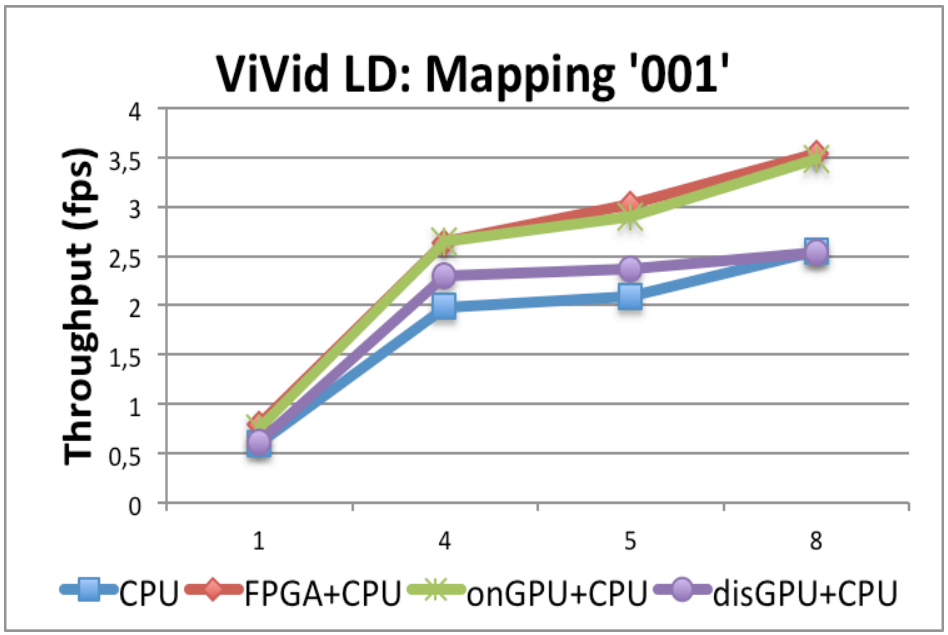
UNIVERSIDAD DE MÁLAGA

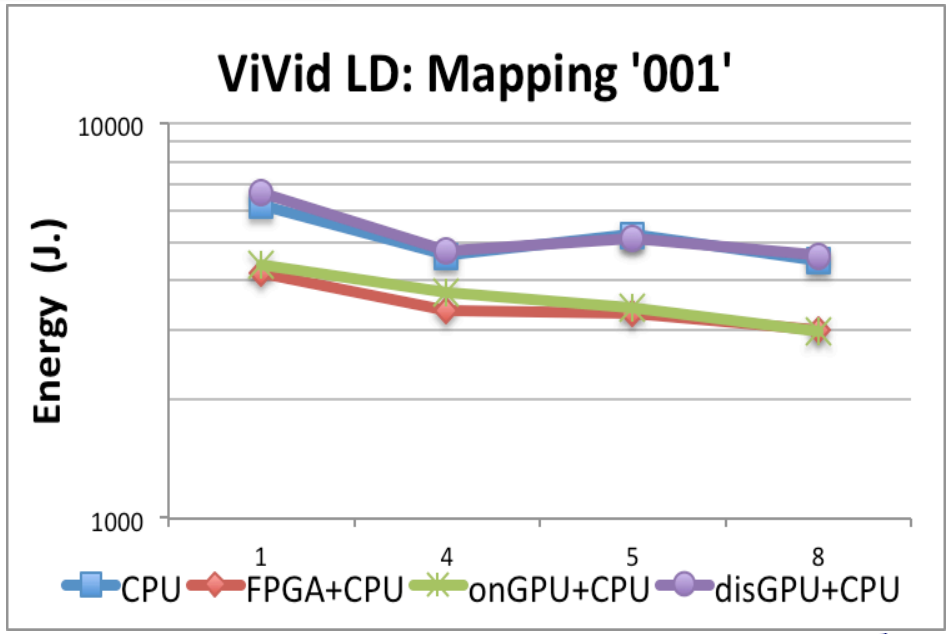
Pipeline on CPU-FPGA

- Heterogeneous results based on our TBB scheduler:
  - Accelerator + CPU
  - Mapping 001



onGPU+CPU ~ FPGA+CPU





UNIVERSIDAD DE MÁLAGA

Conclusions

- Plenty of heterogeneous on-chip architectures out there
- Why not use all devices?
  - Need to find the best mapping/distribution/scheduling out of the many possible alternatives.
- Programming models and runtimes aimed at this goal are in their infancy: striving to solve this.
- Challenges:
  - Hide HW complexity providing a productive programming model
  - Consider energy in the partition/scheduling decisions
  - Minimize overhead of adaptation policies

Future Work

- **Predict energy** consumption on heterogeneous CPU-GPU-FPGA chips
- **Consider energy** consumption in the partitioning heuristic
- Rebuild the scheduler engine on top of the **new TBB library (OpenCL node)**
- Check LogFit on **CPU-GPU-FPGA chip**

Collaborators

- Mª Ángeles González Navarro (UMA, Spain)
- Andrés Rodríguez (UMA, Spain)
- Francisco Corbera (UMA, Spain)
- Jose Luis Nunez-Yanez (University of Bristol)
- Antonio Vilches (UMA, Spain)
- Alejandro Villegas (UMA, Spain)
- Juan Gómez Luna (U. Cordoba, Spain)
- Rubén Gran (U. Zaragoza, Spain)
- Darío Suárez (U. Zaragoza, Spain)
- Maria Jesús Garzarán (Intel and UIUC, USA)
- Mert Dikmen (UIUC, USA)
- Kurt Fellows (UIUC, USA)
- Ehsan Totoni (UIUC, USA)
- Luis Remis (Intel, USA)

Questions

james@jamesreinders.com

pablo.reble@intel.com

james.h.cownie@intel.com

asenjo@uma.es

UNIVERSIDAD DE MÁLAGA