

Coordinating the Use of GPU and CPU for Improving Performance of Compute Intensive Applications

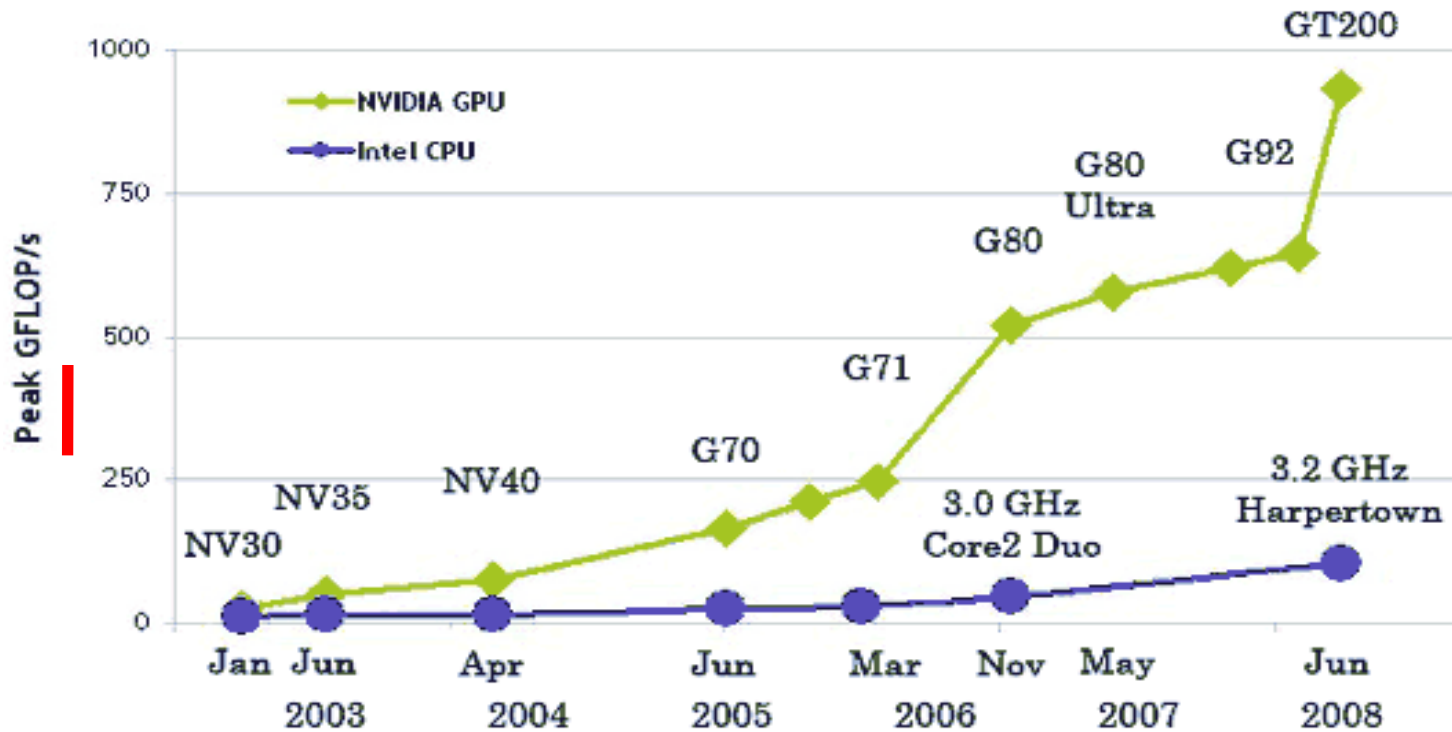
George Teodoro¹, Rafael Sachetto¹, Olcay Sertel²,
Metin Gurcan², Wagner Meira Jr. ¹, Umit Catalyurek²,
Renato Ferreira¹

1. Federal University of Minas Gerais, Brasil
2. The Ohio State University, US

Motivation

- High performance computing
 - Large cluster of-the-shelf components
 - Multi-core/Many-core
 - GPGPU
 - Massively parallel
 - High speedups compared to the CPU

Motivation



- But... GPU is not so fast in all scenarios...
- Current frameworks

– Assume exclusive use of GPU or CPU

Goal

- Target heterogeneous environments
 - Multiple CPU-cores/GPUs
 - Distributed environments
- Efficient coordination of the devices
 - Scheduling tasks according to their specificities
- High level programming abstraction

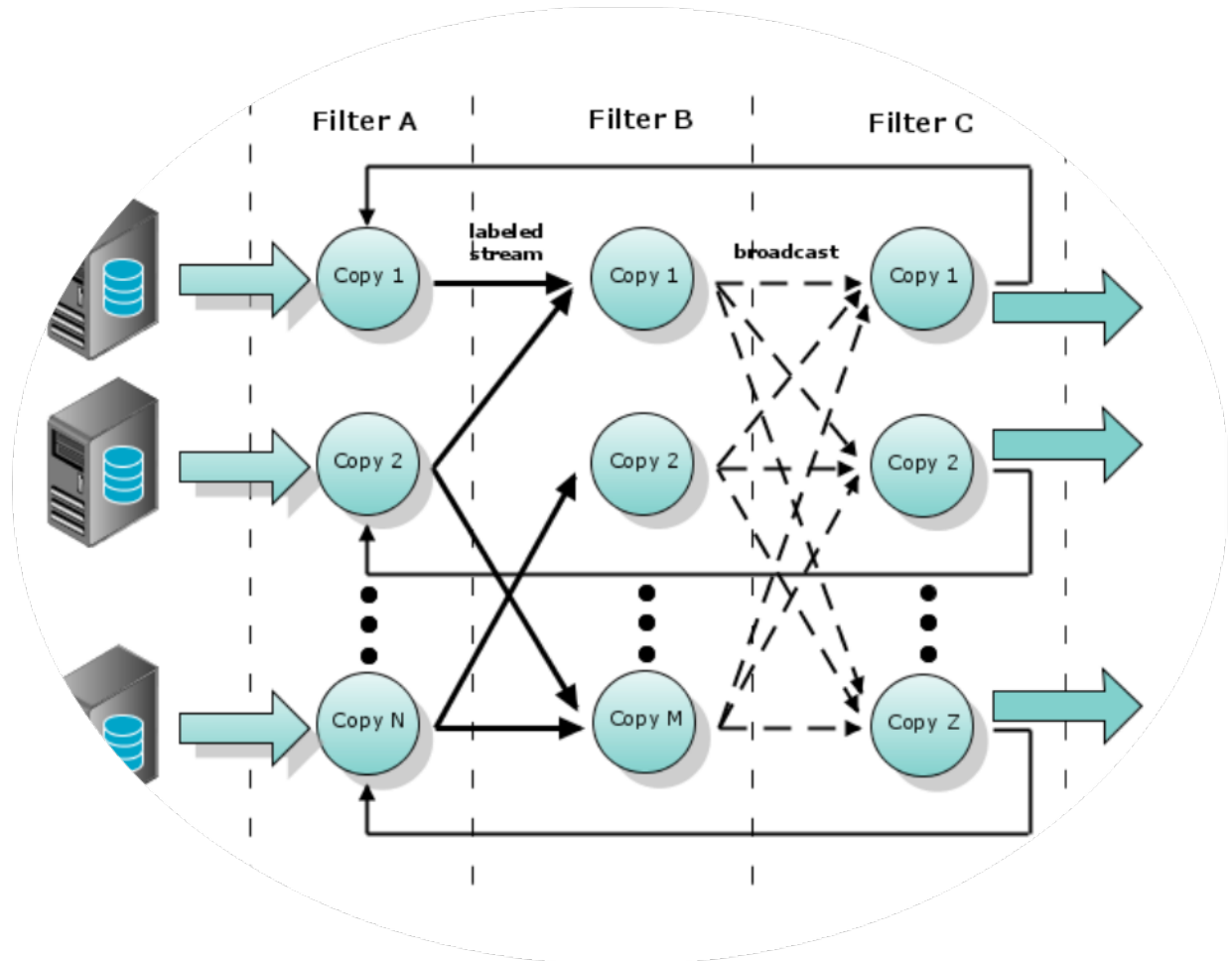
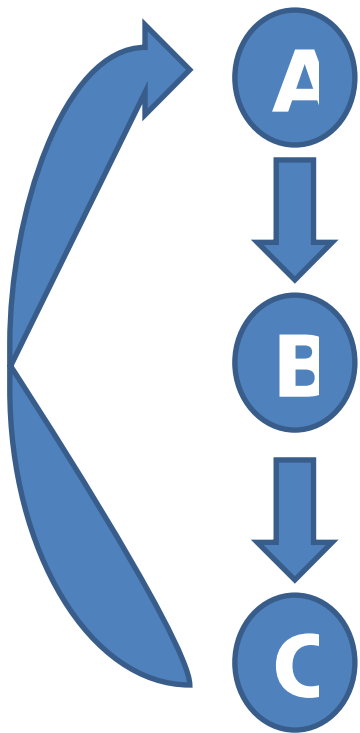
Outline

- Anthill
- Supporting heterogeneous environments
- Experimental evaluation
- Conclusions

Anthill

- Based on the filter-stream model (DataCutter)
 - Application decomposed into a set of filters
 - Communication using streams
 - Transparent instance copy
 - Data flow
 - Multiple dimensions of parallelism
 - Task parallelism
 - Data parallelism

Anthill



Filter programming abstraction

- Event driven interface
 - Aligned with the data flow model
- User provide data processing functions to be invoked upon availability of data
- System controls invocation of user function
 - Dependency analysis
 - Parallelism

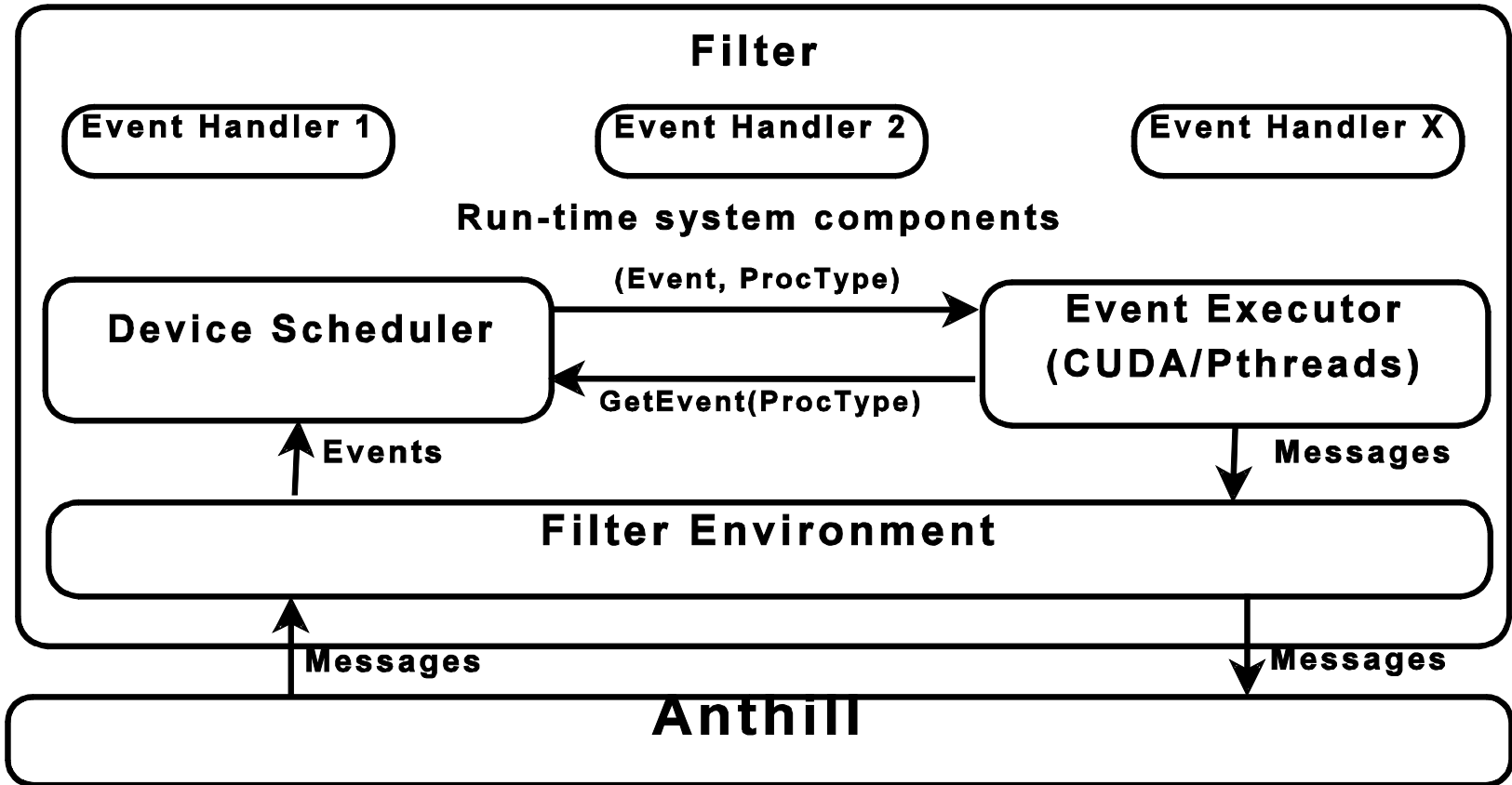
Event handlers

- User provided functions
- Operate on data objects
 - Updates filter state (global)
 - May trigger communication
 - Returns after processing the data element
- Gets invoked automatically when data is available
 - And dependencies are met

Supporting heterogeneous resources

- Event handler implemented to multiple devices
 - Each filter may be implemented targeting the appropriate device
- Multiple devices used in parallel
- Anthill run-time chooses the device for each event

Heterogeneous support overview



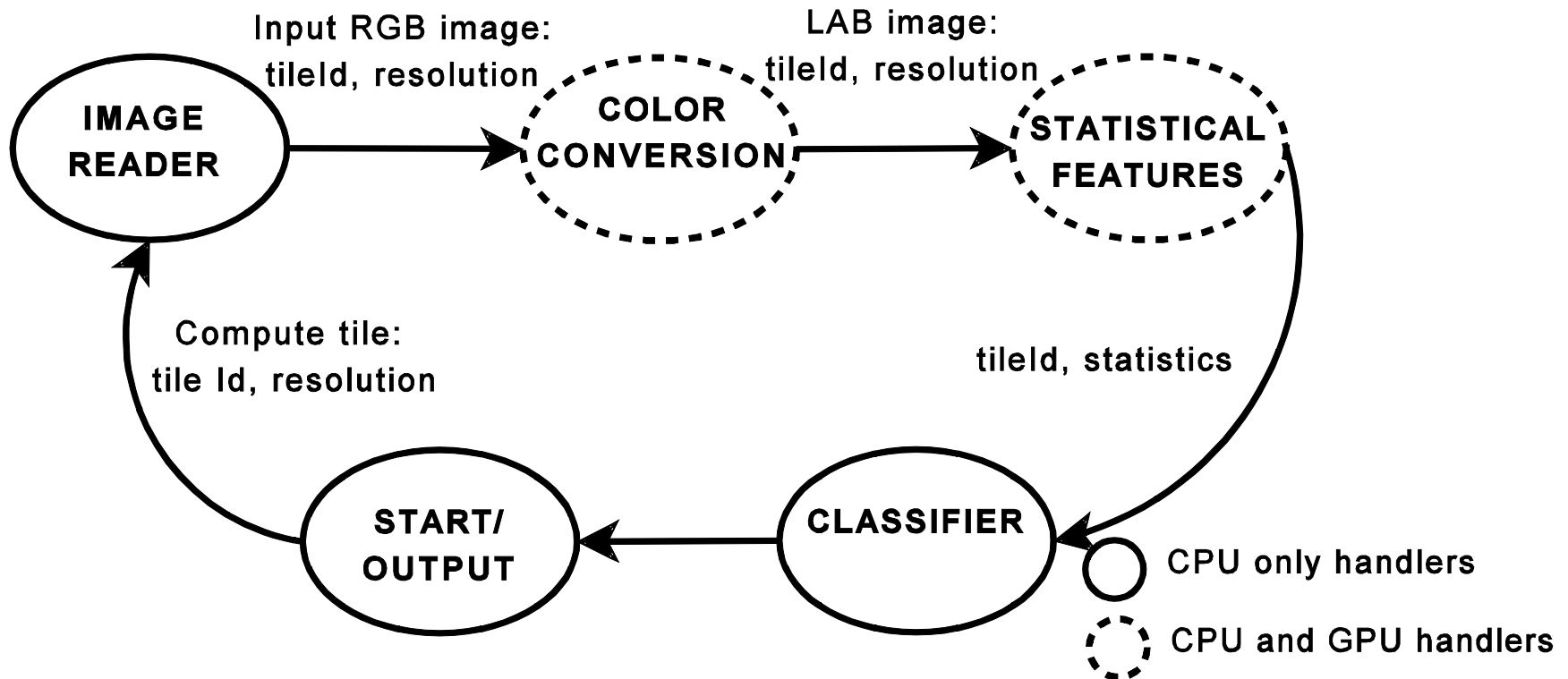
Device scheduler

- Assumes
 - Events are independent
 - Out-of-order execution
- Scheduling policies
 - FCFS – first -come, first-served
 - DWDR – dynamic weighted round robin
 - Orders events according to its performance to each device
 - Selects the event with the highest speedup
 - User given function

Neuroblastoma Image Analysis System

- Classify tissues in different subtypes of prognostic significance
- Very high resolution slides
 - Divided in smaller tiles
- Multi-resolution image analysis
 - Mimics the way pathologists examine them

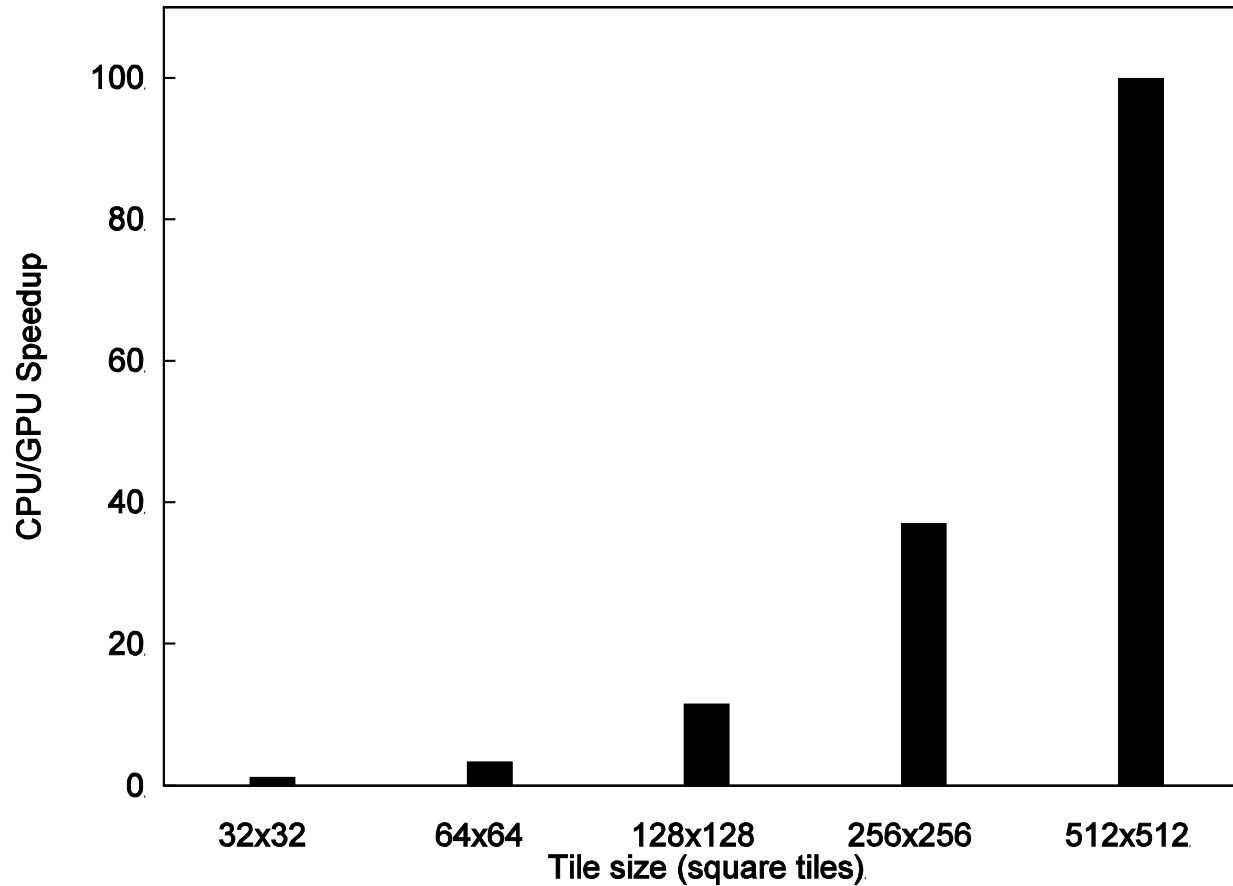
Anthill implementation



Experimental results

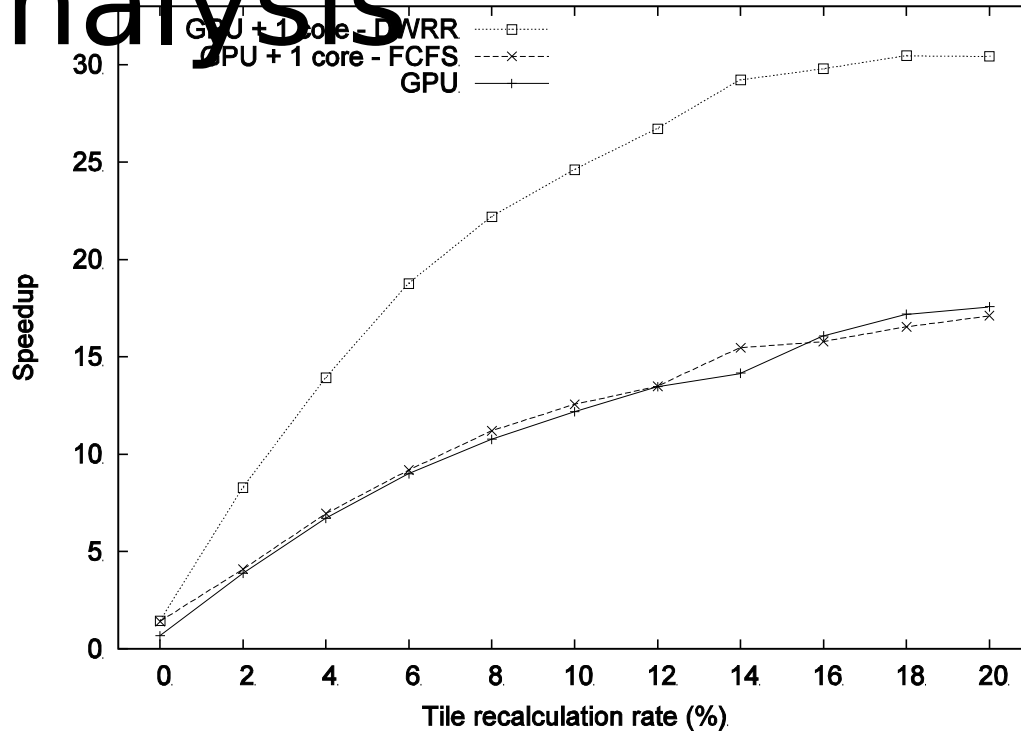
- Setup
 - 10 PCs with an Intel Core 2 Duo CPU 2.13GHz /NVIDIA GeForce 8800GT GPU
 - 4 PCs with a dual quad-core AMD Opteron 2.00GHz processor/ NVIDIA GeForce GTX 260
 - Input data: images of 26,742 tiles using two resolution levels: 32x32 and 512x512

NBIA tasks analysis - performance variation



Dual quad-core AMD Opteron 2.00GHz/NVIDIA GeForce GTX260

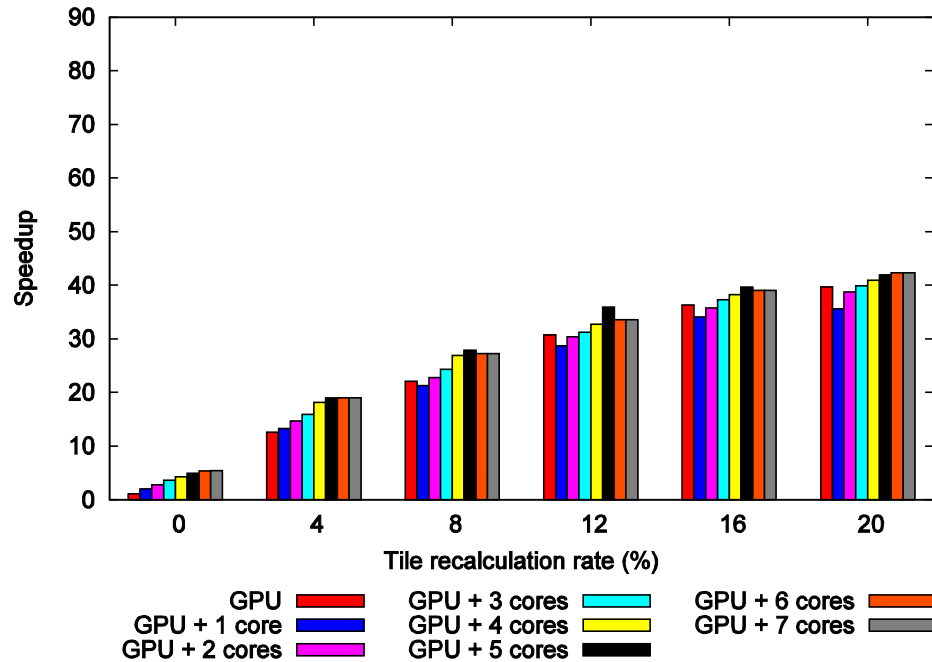
Heterogeneous scheduling analysis



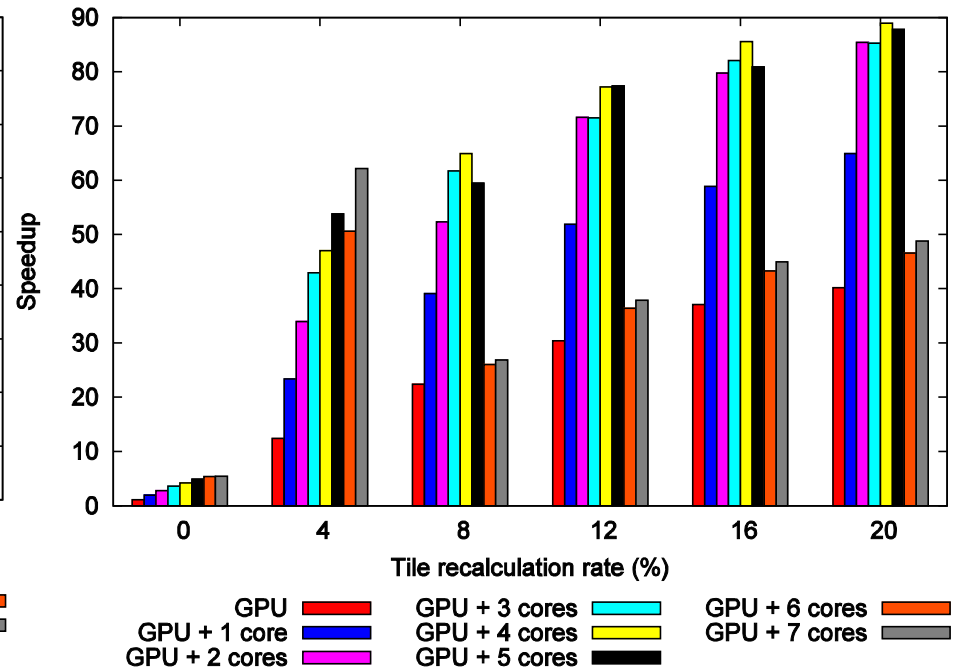
**16 + 1 =
30 ??**

Recalc (%)	12	
Resolution	Low	High
1 CPU core-	263	215
1 CPU core -	21592	4

Heterogeneous scheduling analysis



FCFS

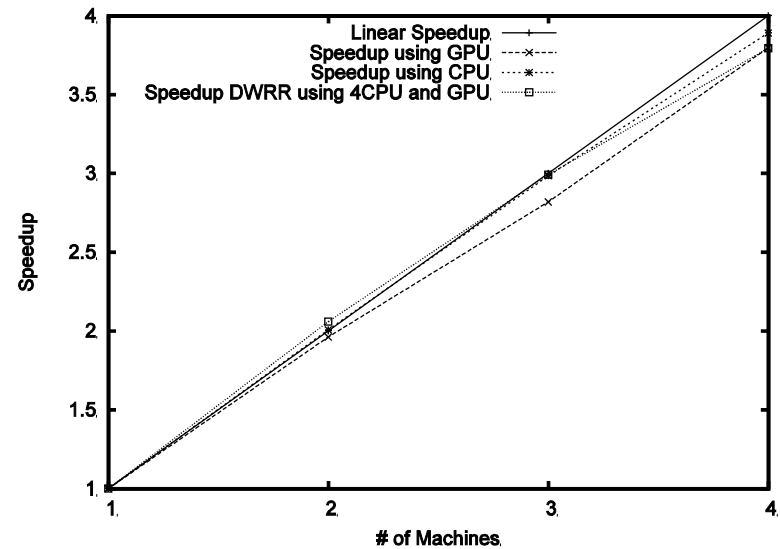
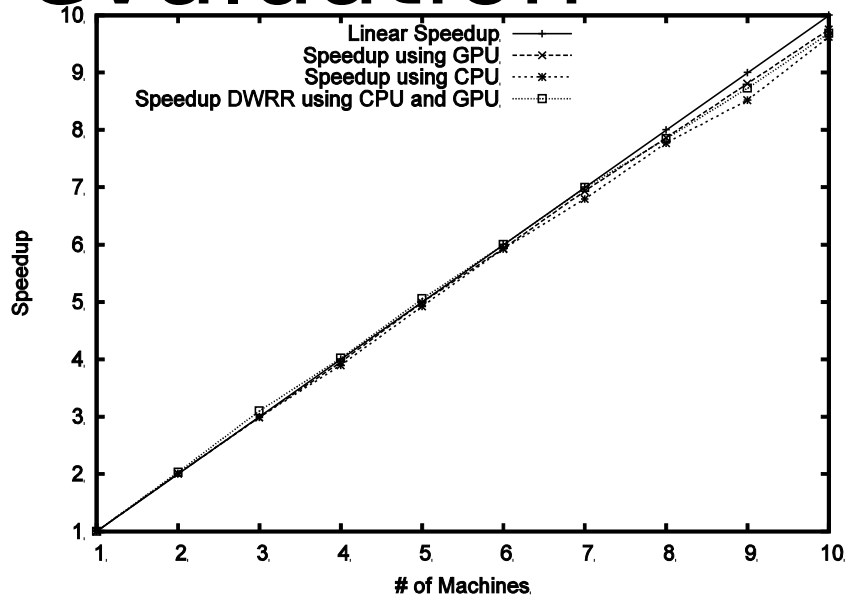


DWDR

Heterogeneous scheduling analysis

# of CPU cores	FCFS		DWRR	
	Low	High	Low	High
1	637	58	10714	1
2	117	133	15748	2
3	1925	173	18614	5
4	2090	219	18634	28
5	2872	286	20070	40
6	3819	393	20147	76
7	4726	478	20266	57

Distributed environment evaluation



Conclusions

- Relative performance between CPU/GPU is data dependent
- Adequate scheduling among heterogeneous processors doubled the performance of the application
- Neglect the CPU is a mistake
- Data-flow is an interesting model to exploit parallelism

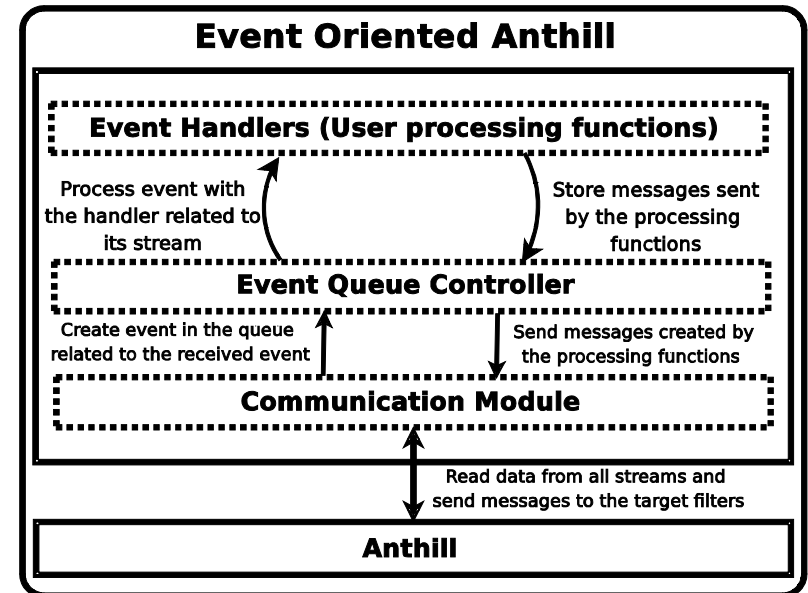
Future work

- New scheduling techniques
- Execution in cluster with heterogeneity among the computing nodes

Questions?

Filter architecture

- Programmer focus on the data transformations and explicit dependencies
- System handles scheduling and management of resources



Application performance

