TU/e TU Delft

UNIVERSITY OF TWENTE.

UNIVERSITY OF AMSTERDAM

© vectors

Put Dutch GPU research on the (road)map!

A Reconnaissance Project by:

NIRICT

Netherlands Institute for Research on ICT

What's in a name?

- GPUs (Graphical Processing Unit)
  - The most popular accelerators
  - Performance reports of 1-2 orders of magnitude larger than CPU
  - Mix-and-match in large-scale systems
  - Challenging to program with traditional programming models
  - Difficult to reason about correctness
  - Impossible to reason about performance bounds

Who are we?

UNIVERSITY OF TWENTE.

UNIVERSITY OF AMSTERDAM

- Marieke Huisman (UT, FMT)
- Gerard Smit, Jan Kuper,
  Marco Bekooij (UT, CAES)
- Hajo Broersma, Ruud van Damme (UT, FMT/MMS)
- Henk Sips, Dick Epema, Alexandru Iosup (TUD, PDS)
- Kees Vuik (TUD, NA)

- Ana-Lucia Varbanescu (UVA, SNE)
- Henk Corporaal (TU/e, ESA)
- Andrei Jalba (TU/e, A&V)
- Anton Wijs, Dragan Bosnacki (TU/e, SET, BME)

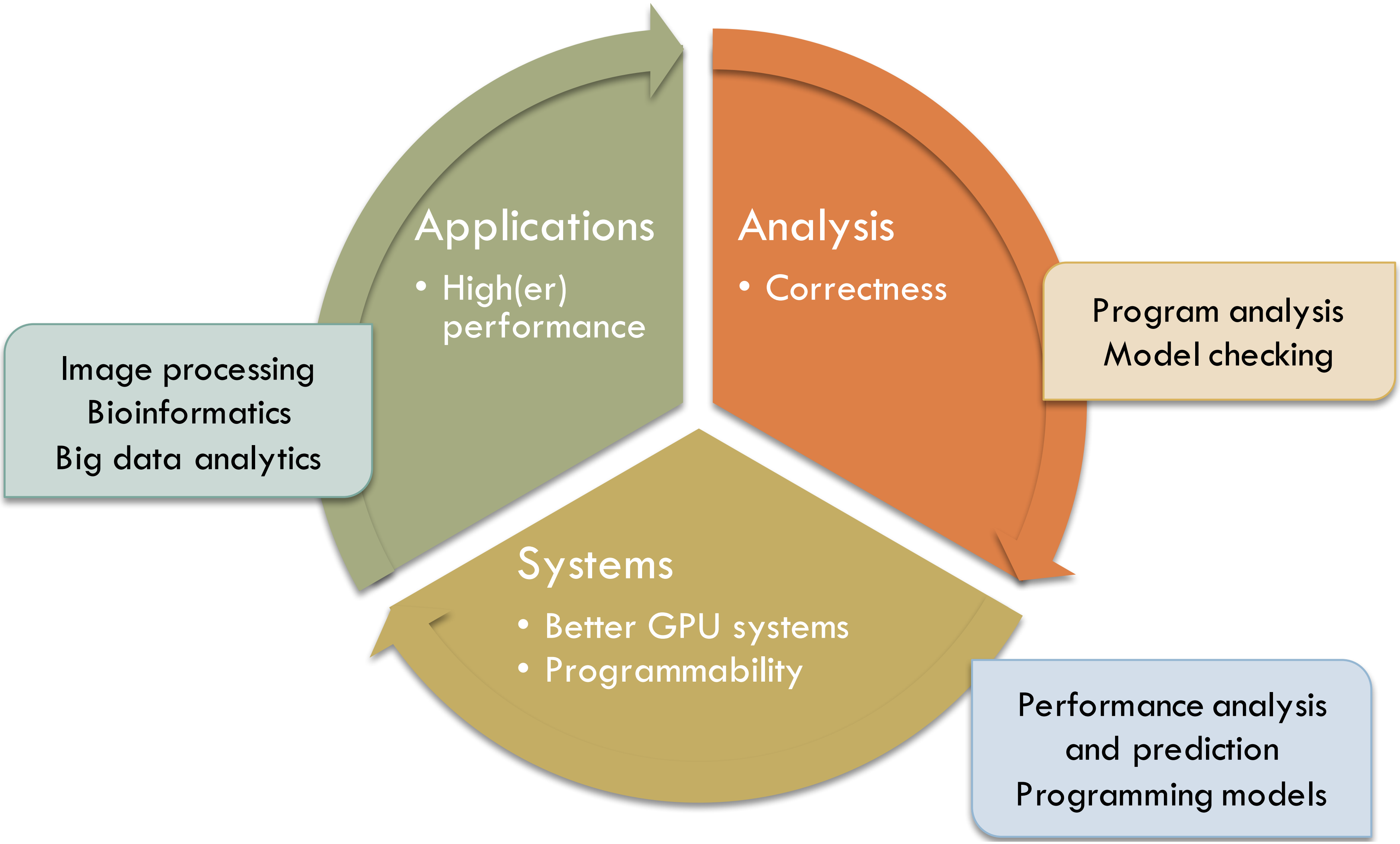
The goal of our collaboration

- To understand the landscape of GPU computing
- To map existing efforts in academia on this landscape
- To collect and map the efforts from industry
- To position ourselves as a strong participant in GPU research internationally

The Landscape of GPU research

- Applications
  - Most success stories come from numeric simulation, gaming, and scientific applications.
  - New-comers like graph processing are interesting targets, too.
  - Graphics and vizualisation remain a big consumer
- Analysis
  - Techniques to reason about correctness of applications
- Systems
  - First steps in performance analysis, modeling, and prediction
  - Building better GPUs and better systems with GPUs emerges as a necessity for GPU computing
  - Highly-programmable models for programming GPU-systems

Our Mission Statement



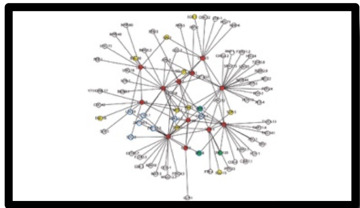
Outline

Andrei Jalba
Kees Vuik
Hajo Broersma, Ruud van Damme

Applications

- High(er) performance

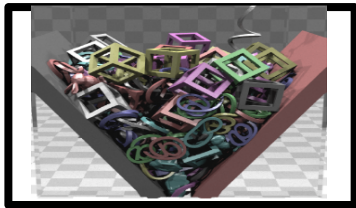
Applications (1/2)



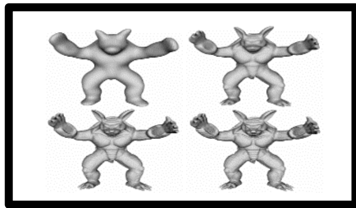
100 X
Biomedical applications

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix}$$

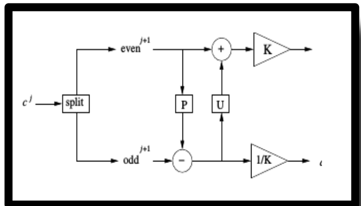
10 X
SpMV, linear system solvers



50 X
Elastic objects with contact

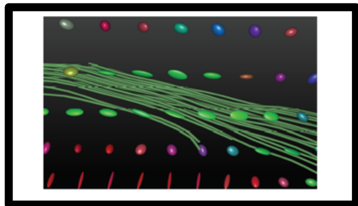


200 X
Level sets



25 X
Wavelets

15 X
HD video decoding



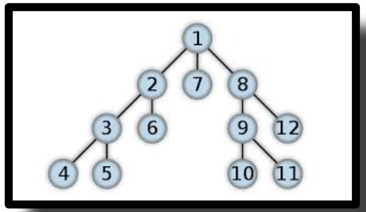
40 X
Geodesic fiber tracking

Applications (2/2)

TU Delft

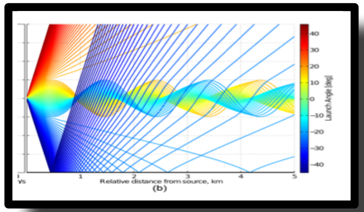
20-40 X

Numerical methods: ship simulator



2-50 X

Graph processing



10-12 X

Sound Ray-tracing

80 X

Stereo vision

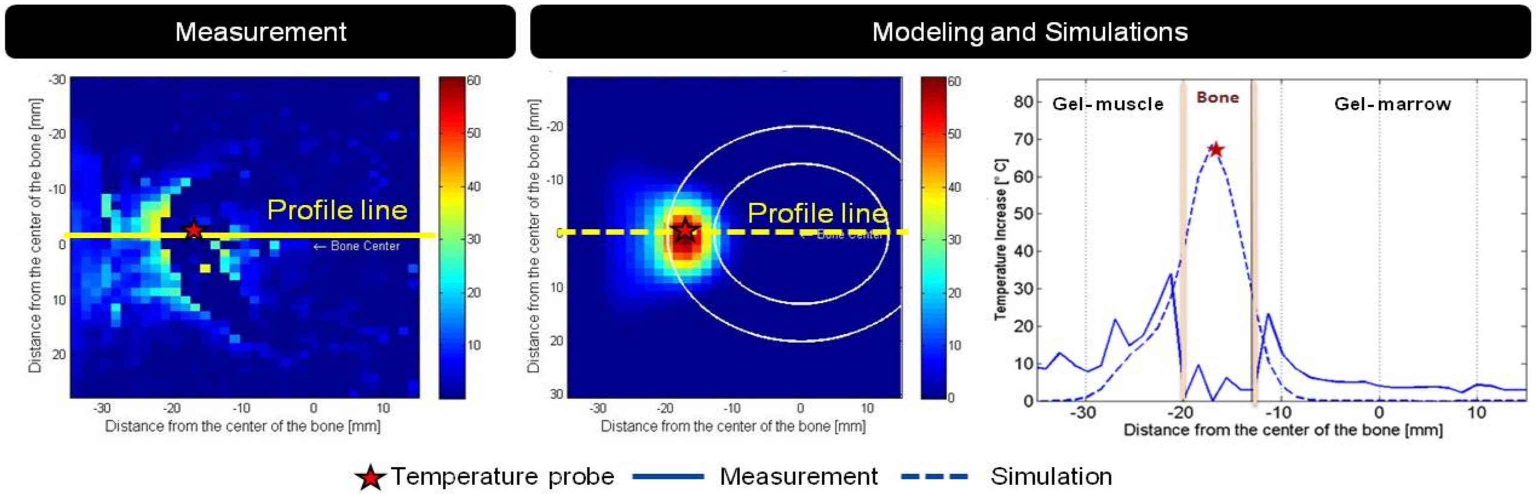
UNIVERSITY OF TWENTE.

Nano-particle networks

Biomedical:

Modeling MR-guided HIFU treatments for bone cancer

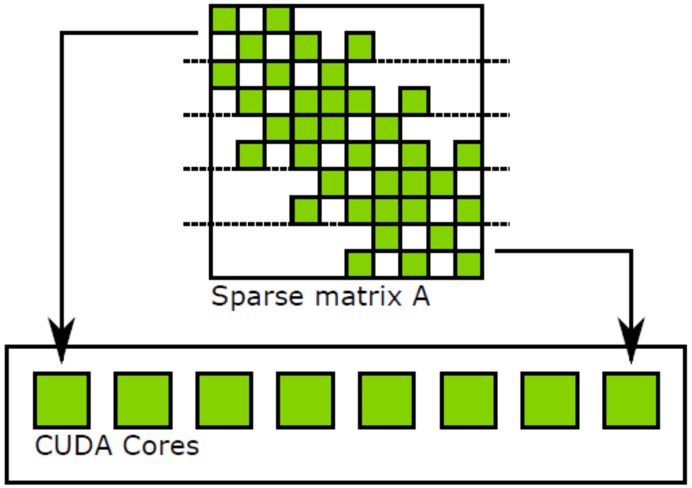
- Magnetic Resonance Guided High-Intensity Focused Ultrasound Treatments
  - Impossible to measure temperature with HIFU methods
  - Prediction of temperatures with mathematical models instead



GPU algorithms can speed up the methods by factor 1000
crucial since it makes the methods applicable in practice

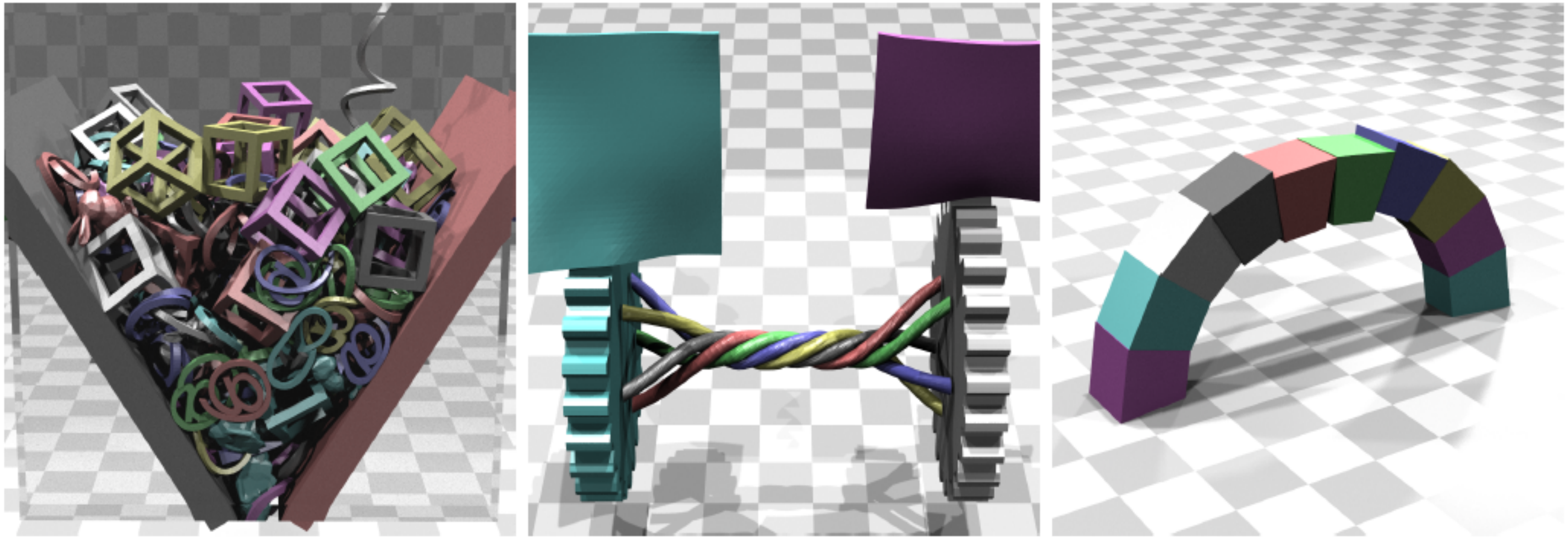
Numerical methods:
SpMVs

- Sparse matrices have relatively few non-zero entries
- Frequently $O(n)$ rather than $O(n^2)$
- Only store & compute non-zero entries
- Difficult to parallelize efficiently: low-arithmetic intensity
  - Bottleneck is memory throughput
  - Solution: block-compressed layout (BCSR)

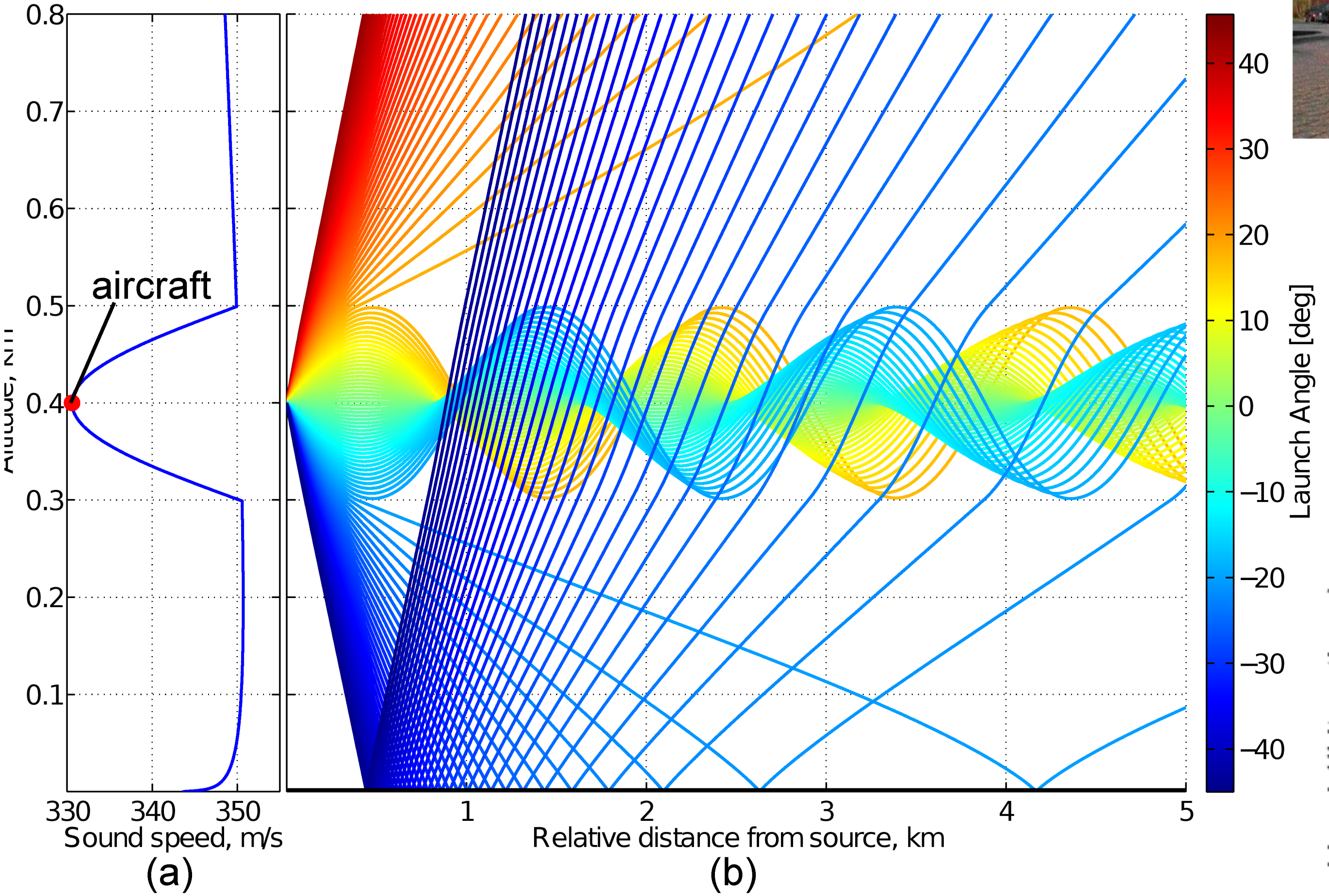


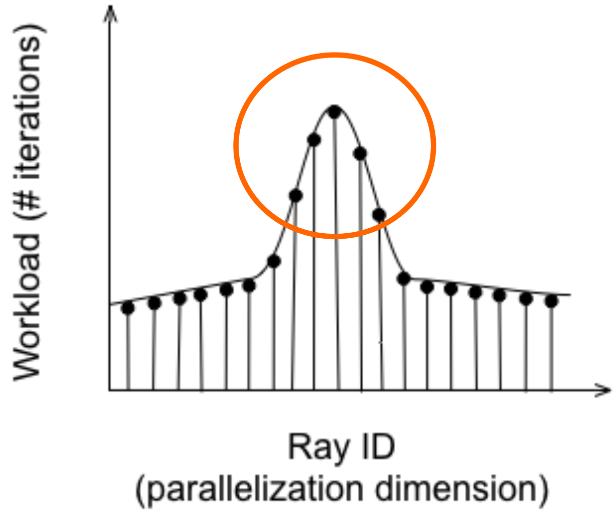
Elasticity with contact

- One order of magnitude faster than CPU version

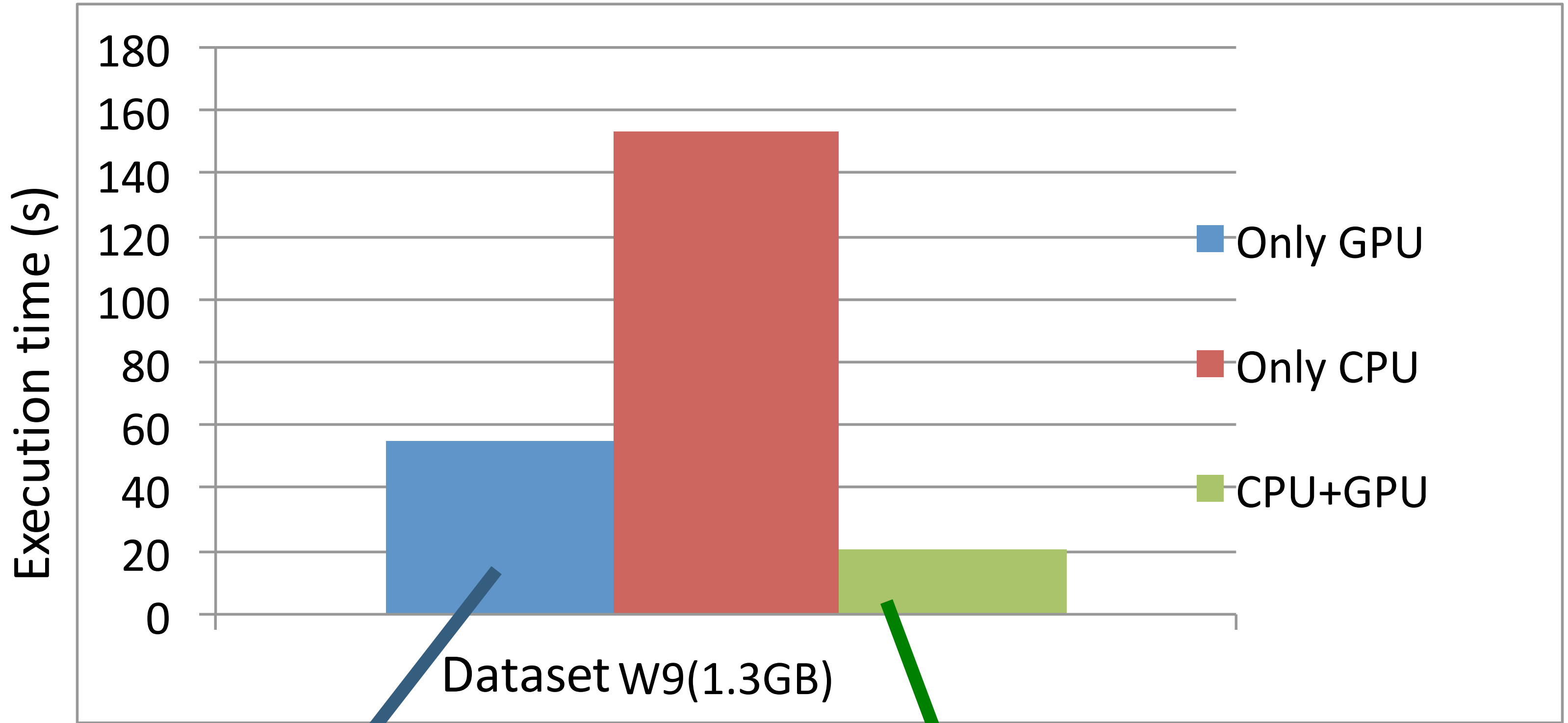


Numerical simulation: Sound ray tracing





Numerical simulation: Sound ray tracing

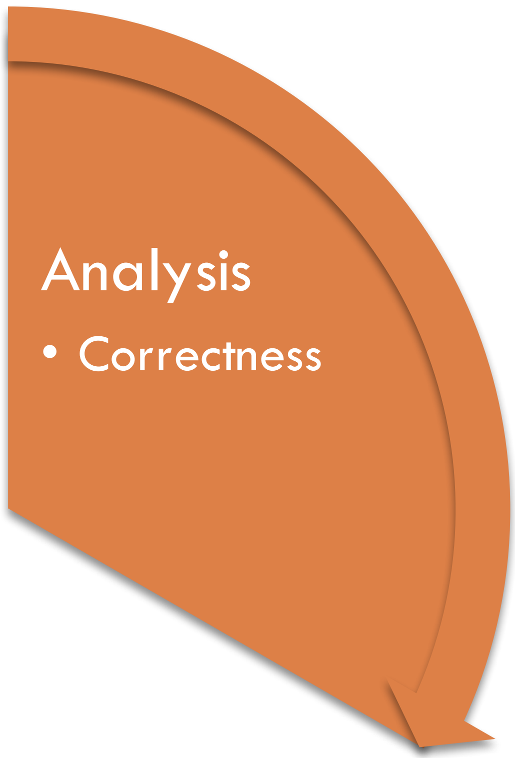


More than 2x performance improvement compared to CPU

62% performance improvement compared to "Only-GPU"

Outline

Marieke Huisman
Anton Wijs, Dragan Bosnacki



VerCors: Verification of Concurrent Programs

UNIVERSITY OF TWENTE.

HTTP://UTWENTE.NL/VERCORS/

- Basis for reasoning: Permission-based Separation Logic
- Java-like programs: thread creation, thread joining, reentrant locks
- OpenCL-like programs
- Permissions:
  - Write permission: exclusive access
  - Read permission: shared access
  - Read and write permissions can be exchanged
  - Permission specifications combined with functional properties

A logic for OpenCL kernels

UNIVERSITY OF TWENTE.

- Kernel specification
  - All permissions that a kernel needs for its execution
- Group specification
  - Permissions needed by single group
  - Should be a subset of kernel permissions
- Thread specification
  - Permissions needed by single thread
  - Should be a subset of group permissions
- Barrier specification
  - Each barrier allows redistribution of permissions

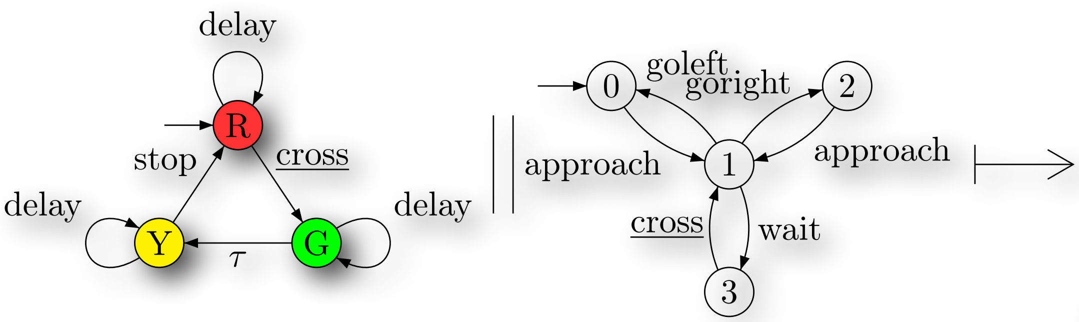
Plus:
functional specifications (pre- and postconditions)

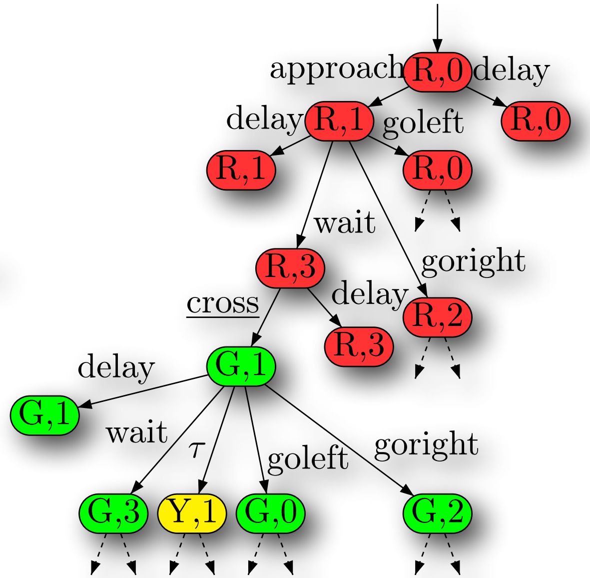
Challenges

- High-level sequential programs compiled with parallelising compiler
  - Ongoing work: verification of compiler directives
- Correctness of compiler optimisations and other program transformations
- Scaling of the approach
- Annotation generation

Efficient Multi-core model checking

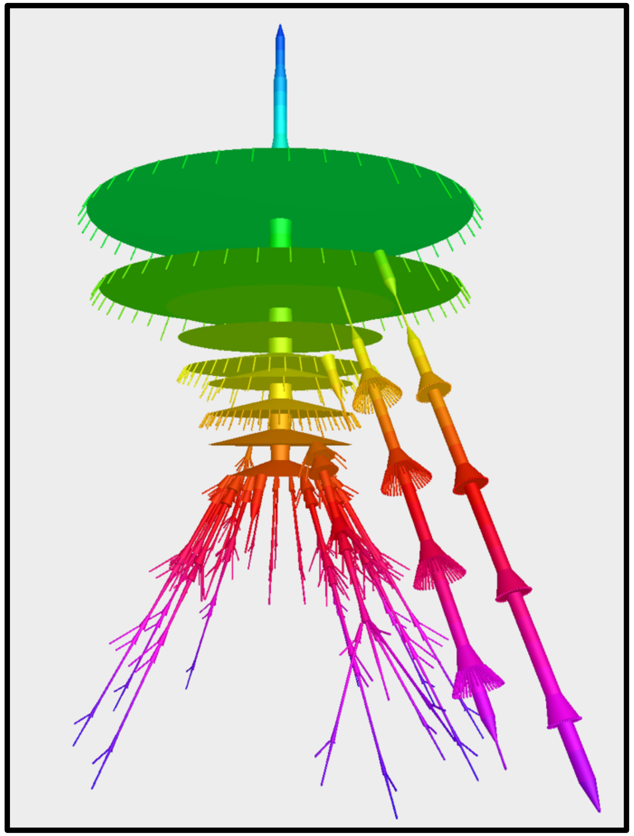
- Technique to exhaustively check (parallel) software specifications by exploring state space: Model Checking
- Push-button approach, but scales badly
- A GPU-accelerated model checker: GPUexplore (10-100x speedup)





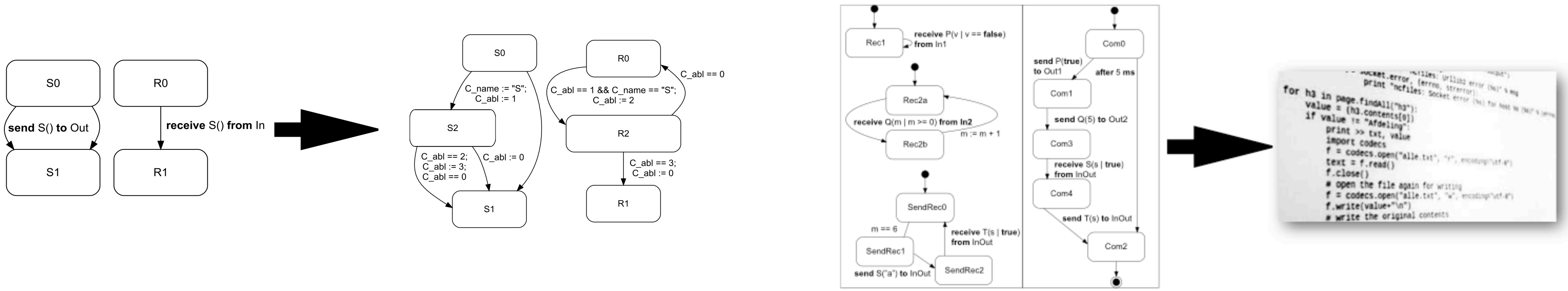
Efficient Multi-core model checking

- Other model checking operations performed on a GPU
- State space minimisation: reducing a state space to allow faster inspection (10x speedup)
- Component detection: relevant for property checking (80x speedup)
- Probabilistic verification: check quantitative properties (35x speedup)



Model-driven code engineering

- Approach: first design the application through modelling, using a Domain Specific Language
- Model transformations are used to prepare the model for the (parallel) platform
- Verifying property preservation of model-to-model transformations (are functional properties of the system preserved?)



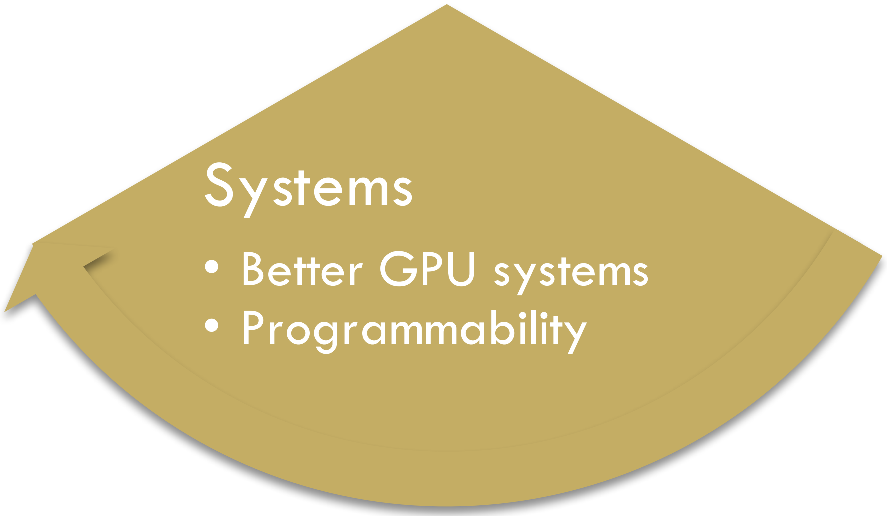
- Then, generate parallel code implementing the specified behaviour
- Verify the relation between code and model using separation logic (VeriFast tool)

Challenges

- Support for GPUexplore of more expressive modelling language
- Model transformations: express code optimisations
- Code generation: support for platform model specifying the specifics of the targeted hardware

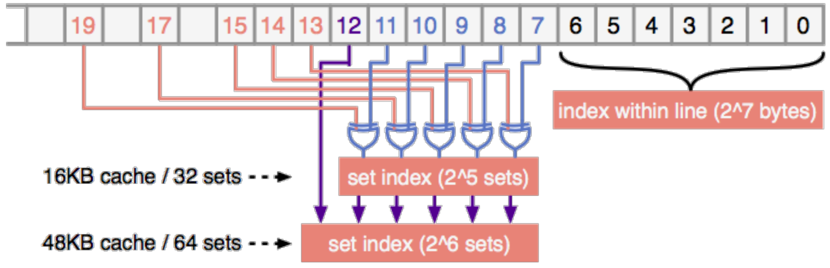
Outline

Henk Sips, Dick Epema, Alexandru Iosup
Ana Lucia Varbanescu
Gerard Smit, Marco Bekooij, Jan Kuper
Henk Corporaal

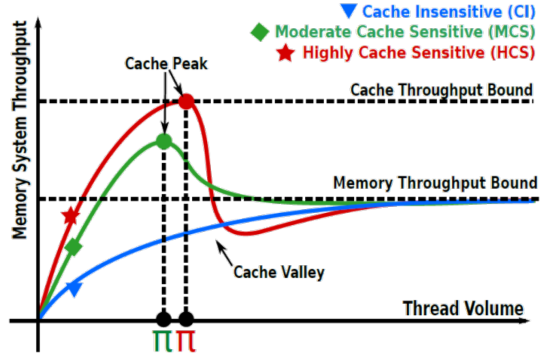


Understanding GPUs

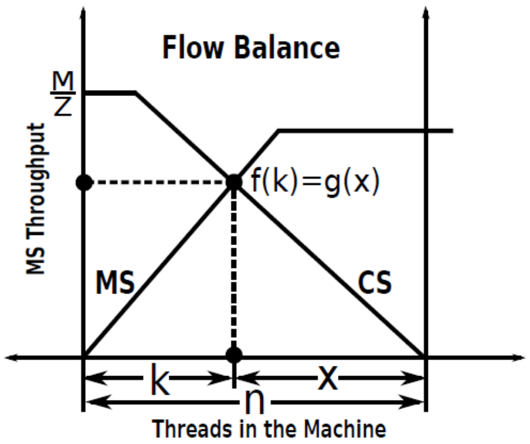
- Modeling of GPU L1 cache



- Cache bypassing

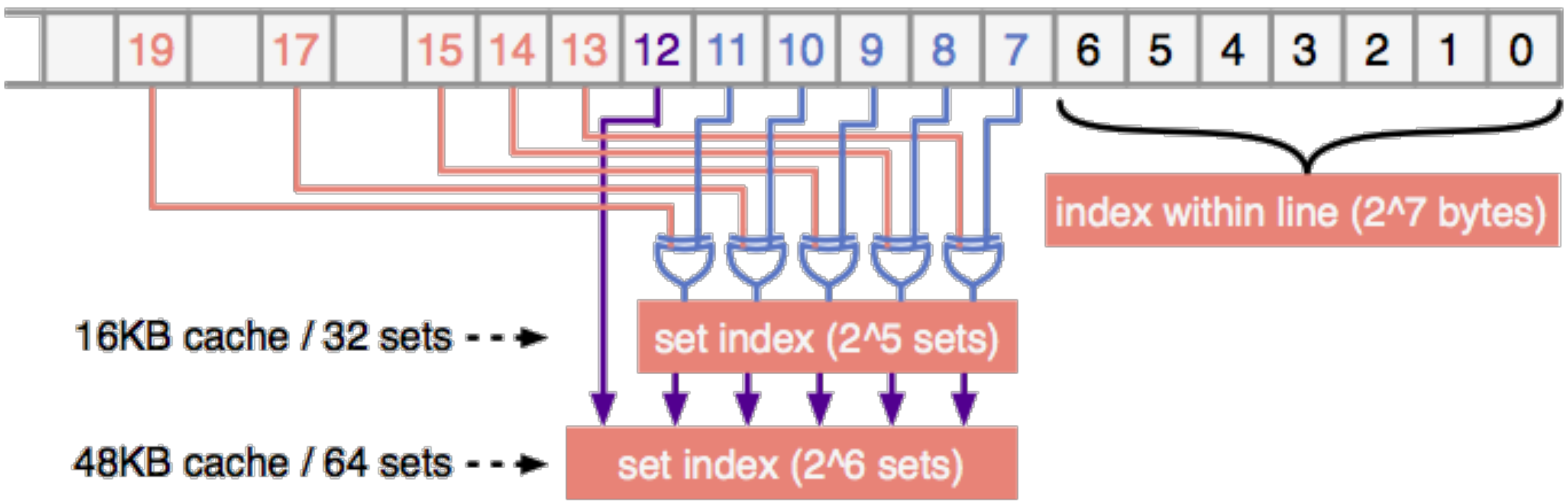


- Transit model

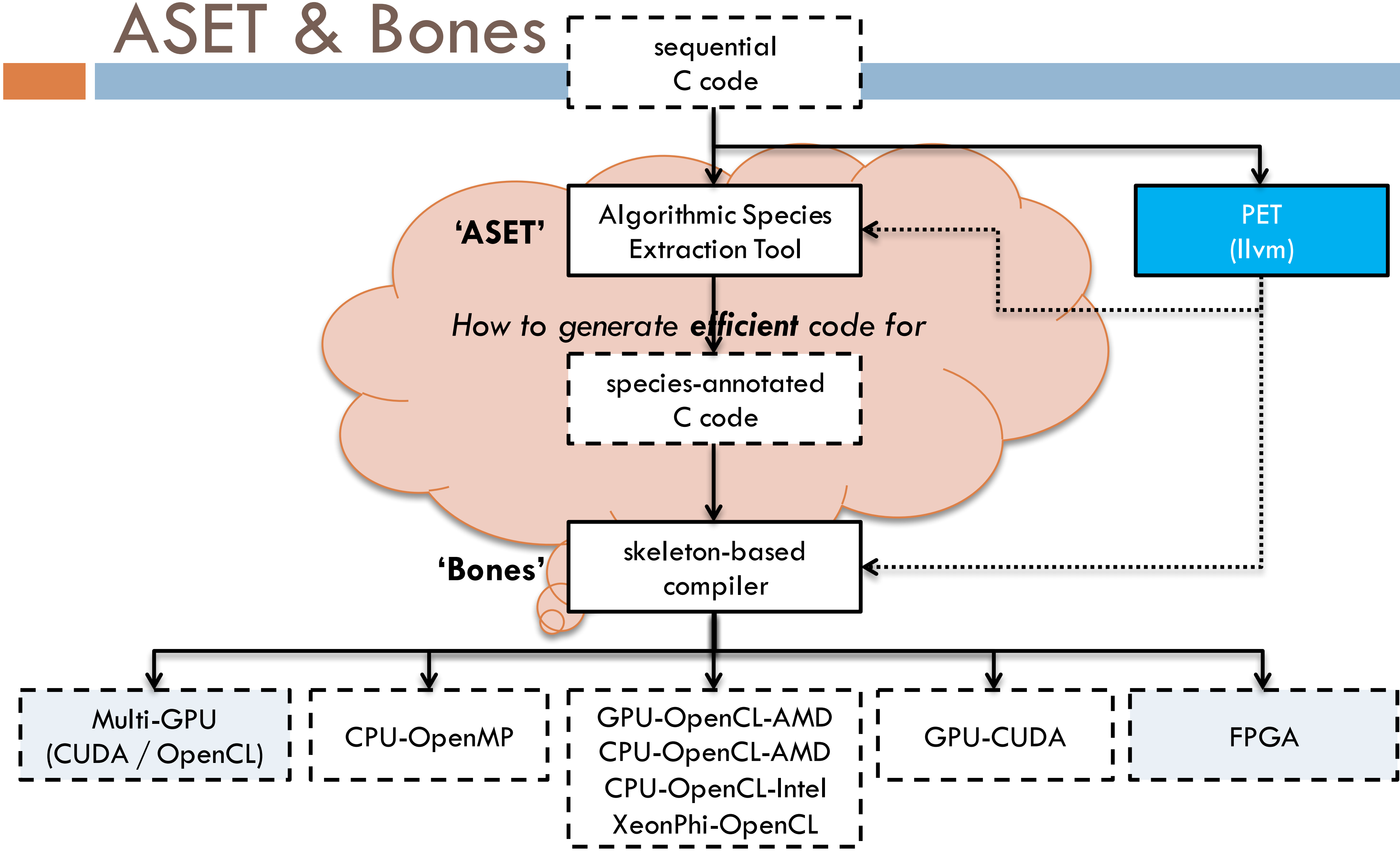


Understanding GPUs: L1 cache modeling

- GPU Cache model:
  - Execution model (threads, thread blocks)
  - Memory latencies
  - MSHRs (pending memory requests)
  - Cache associativity



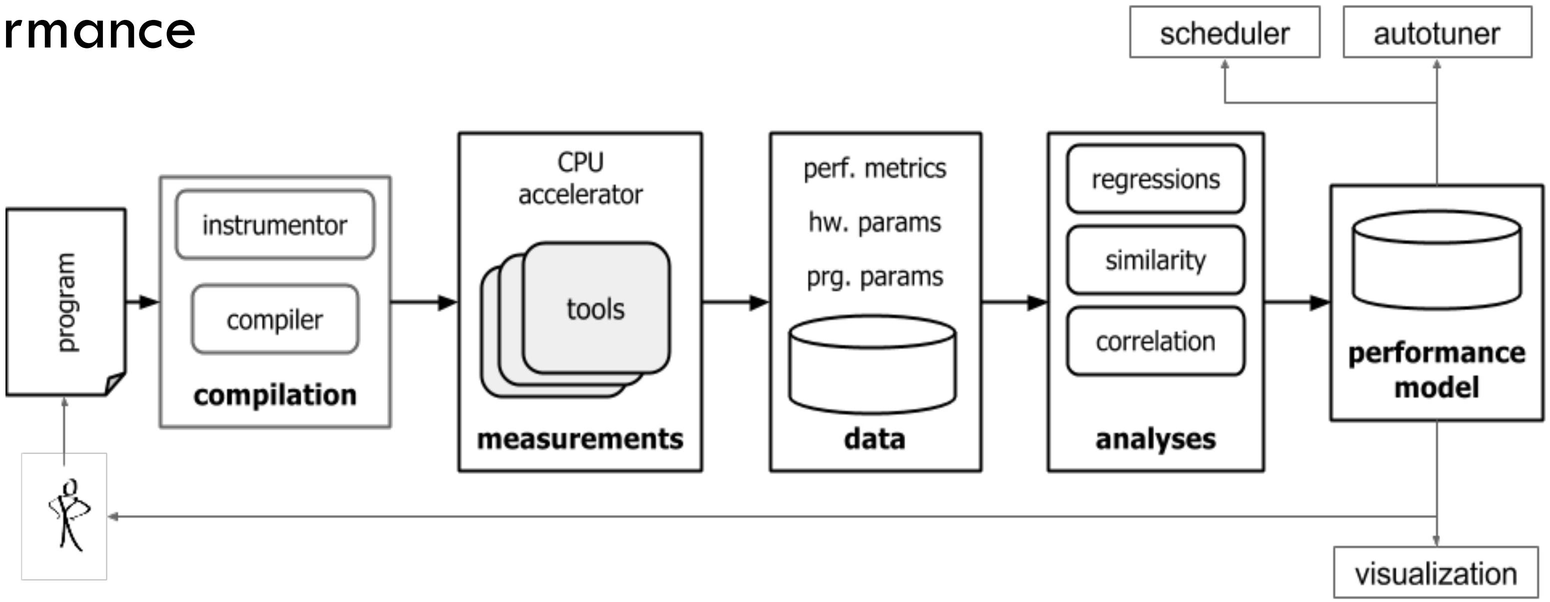
Code generation: ASET & Bones



Performance modeling: the BlackForest framework

UNIVERSITY OF AMSTERDAM

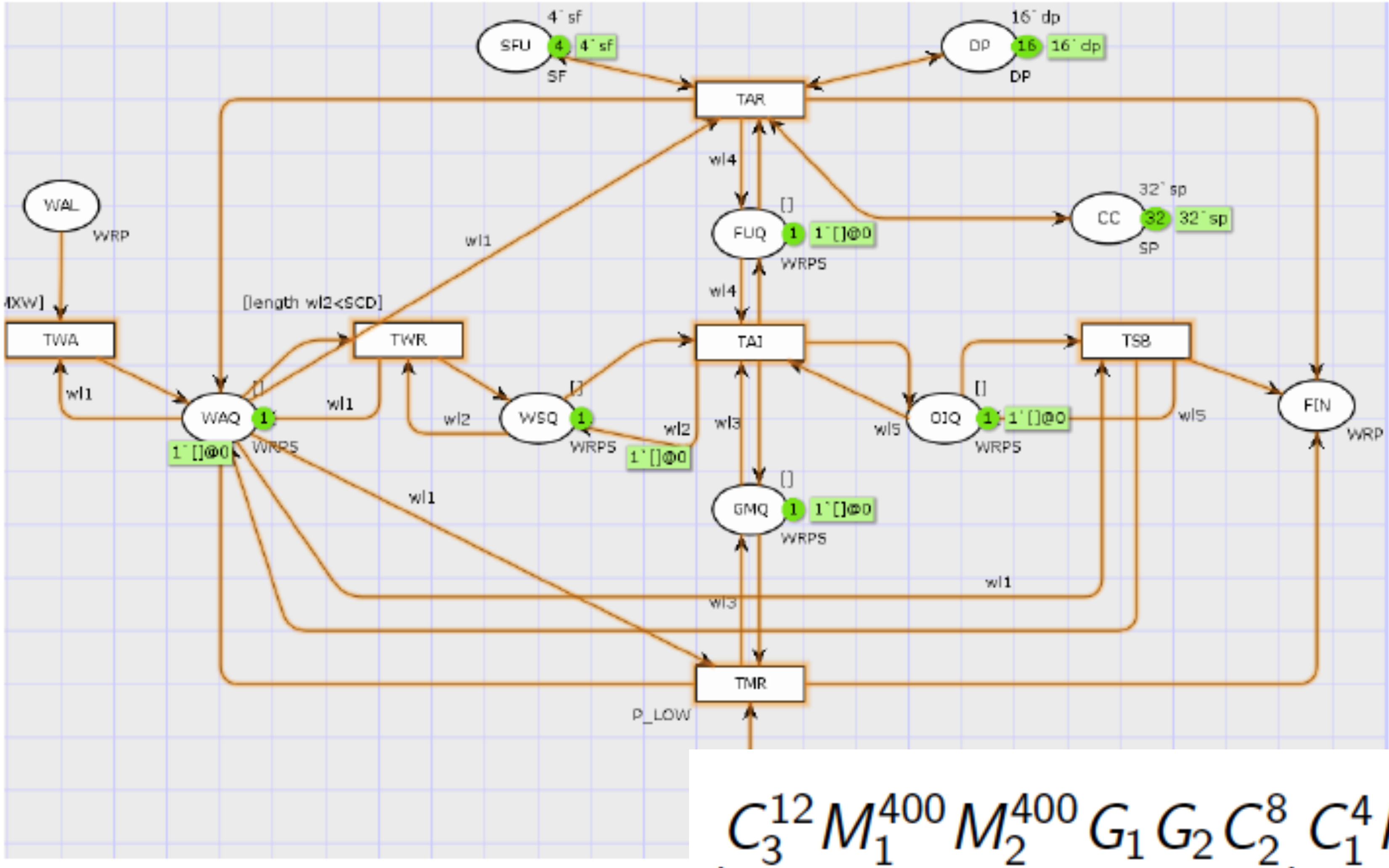
- Build a model based on statistical analysis using performance counters.
  - Compilation: optional, scope limitation by instrumentation
  - Measurements: performance data collection via hardware performance counters
  - Data: repository, file system, database
  - Analyses: reveal correlation between counter behavior and performance



Performance modeling: Colored Petri nets

TU Delft

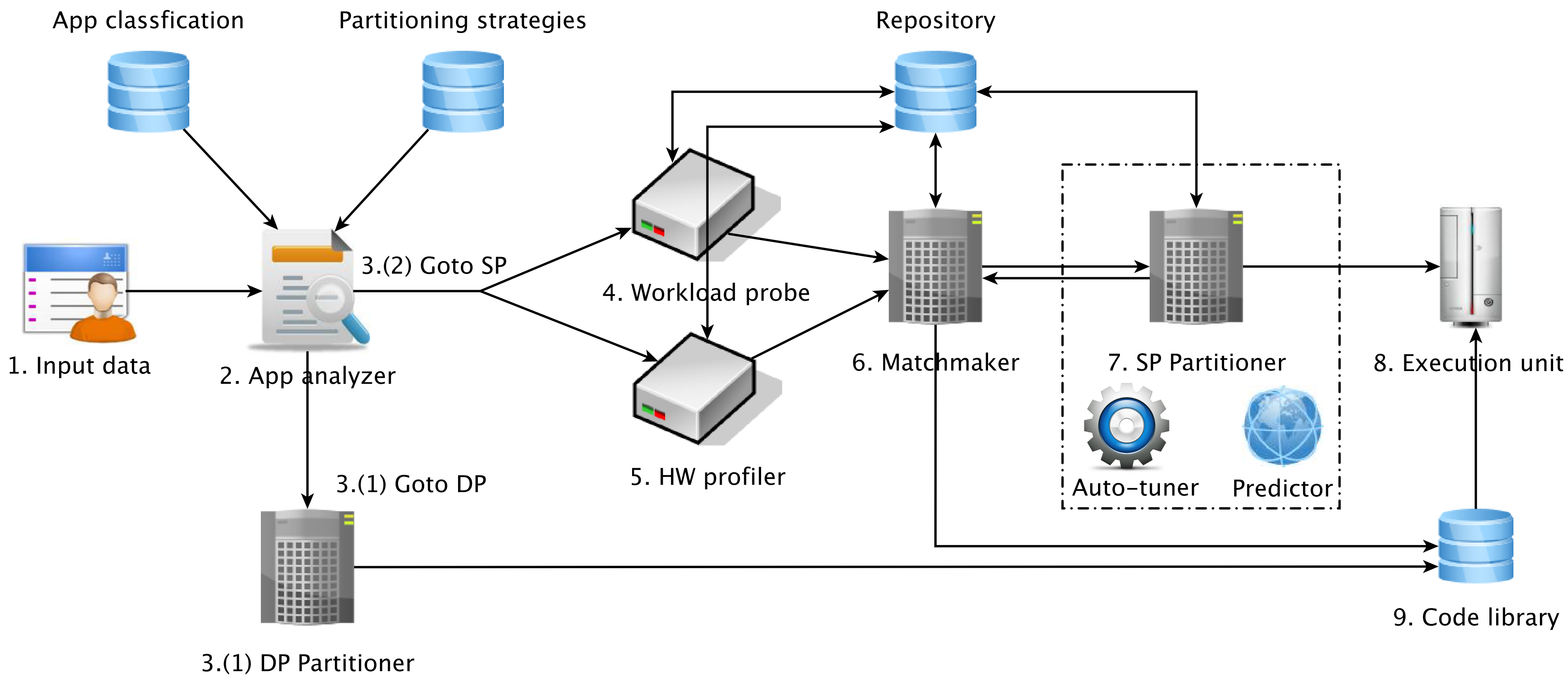
UNIVERSITY OF AMSTERDAM



$$\underbrace{C_3^{12} M_1^{400} M_2^{400} G_1 G_2 C_2^8}_{\text{Aw times}} C_1^4 \bar{M}^{400} C_2^8$$

Heterogeneous computing: the Glinda framework

- A framework for running applications on heterogeneous CPU+GPUs hardware
  - Static workload partitioning and heterogeneous execution.



Outline

What next?

Next steps

- Inventory of existing and near-future GPU-related research
  - Academia AND industry
- Focus on mapping the existing research on these three topics
  - … and add more topics!
- Understand collaboration potential between academia and industry
  - National and international level
- Go international !

First …

- We will organize 3+1 call for presentations
  - Systems and performance – June/July
  - Analysis – September/October
  - Applications – November/December
  - Education !!!
- All interested partners are invited to give a talk about their GPU-research and submit a 1-page description of the research.
  - Focus on potential collaborations
  - Focus on both *offer* and *demand*
- We will summarize the findings in a 3-volume report: "The Landscape of GPU computing in NL".

… and then…

- We will analyze correlations between topics
  - For potential collaboration
  - For potential partnerships
- We will compare with existing work internationally
- We will draft a “GPU Computing Research Roadmap” for the near future.

How can YOU contribute?

- Are you doing GPU research?
  - Let us know! Respond to our call for presentations!
- You need GPU-like performance?
  - Let us know! Come and talk about your application and challenges!
- Are you active in GPU-related education:
  - Let us know! E-mail and let us know if you want to meet other educators like you!
- You want to do GPU research?
  - Join our meetings! See our website: http://fmt.ewi.utwente.nl/Workshops/NIRICT_GPGPU/index.html