An Ecosystem for Combining Performance and Correctness for Many-Cores

Pieter Hijma
pieter@cs.vu.nl

Friday 16 May 2018

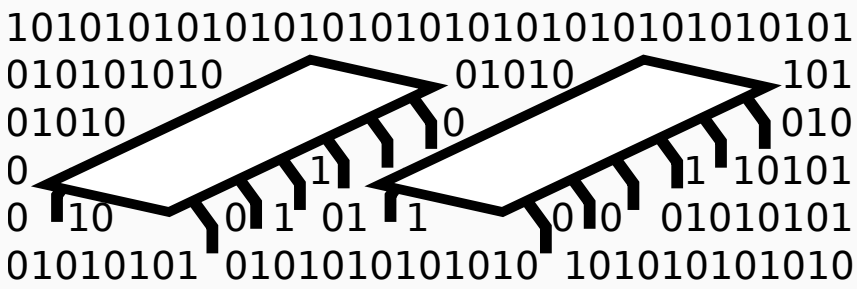
Fourth NIRICT GPGPU Reconnaissance Workshop

VU UNIVERSITY AMSTERDAM
UNIVERSITY OF AMSTERDAM
TU/e Technische Universiteit Eindhoven University of Technology
netherlands eScience center by SURF & NWO

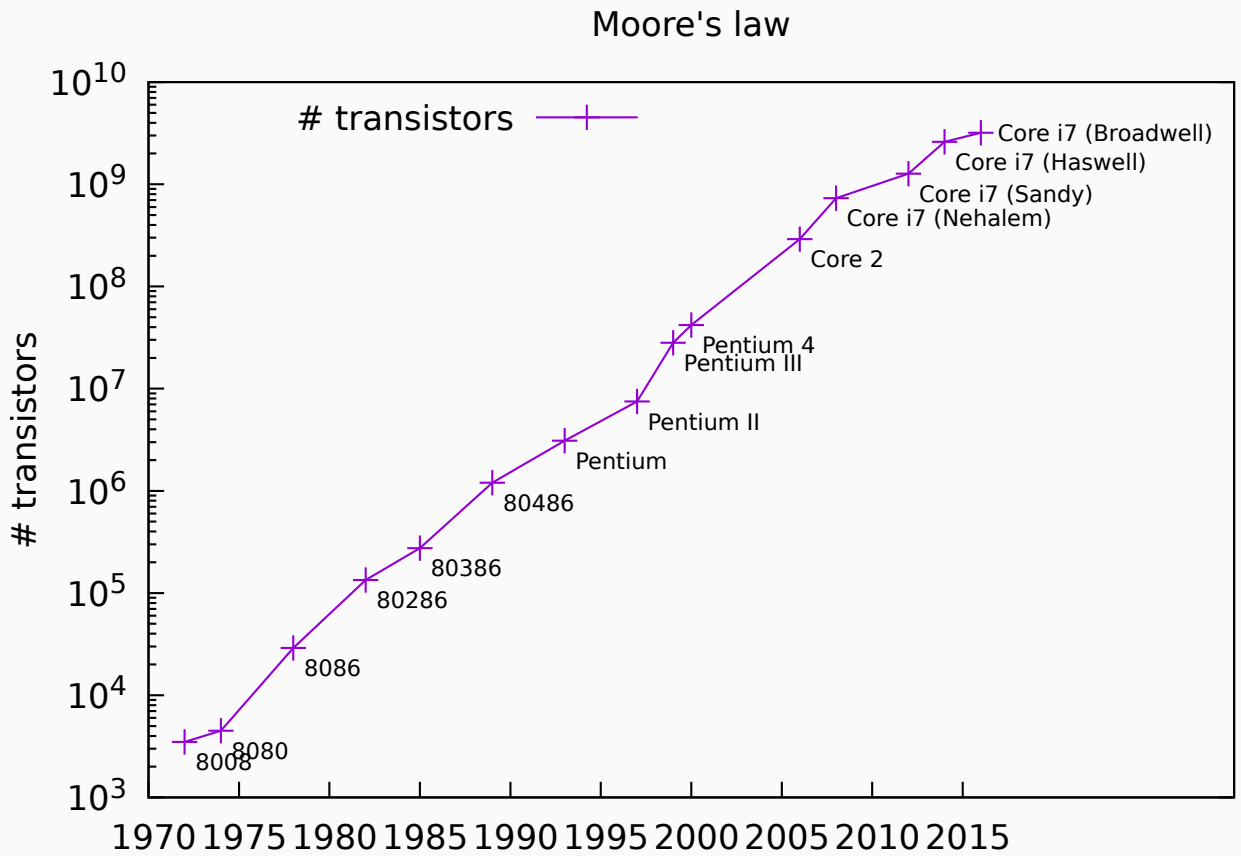
Message

In coming years, software will have to adapt to hardware more than previously.

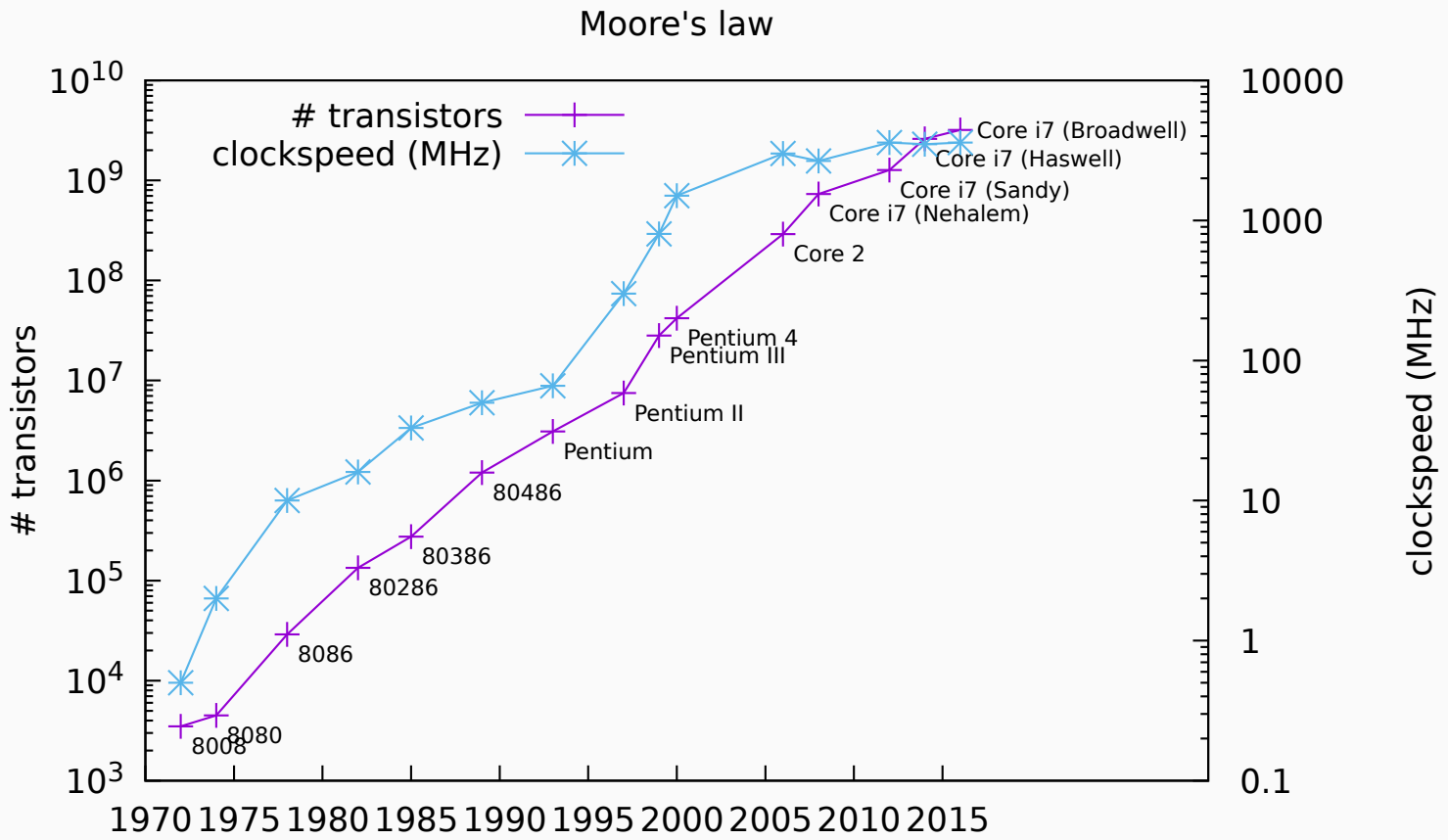
Our solution: An Ecosystem for Combining Performance and Correctness for Many-Cores



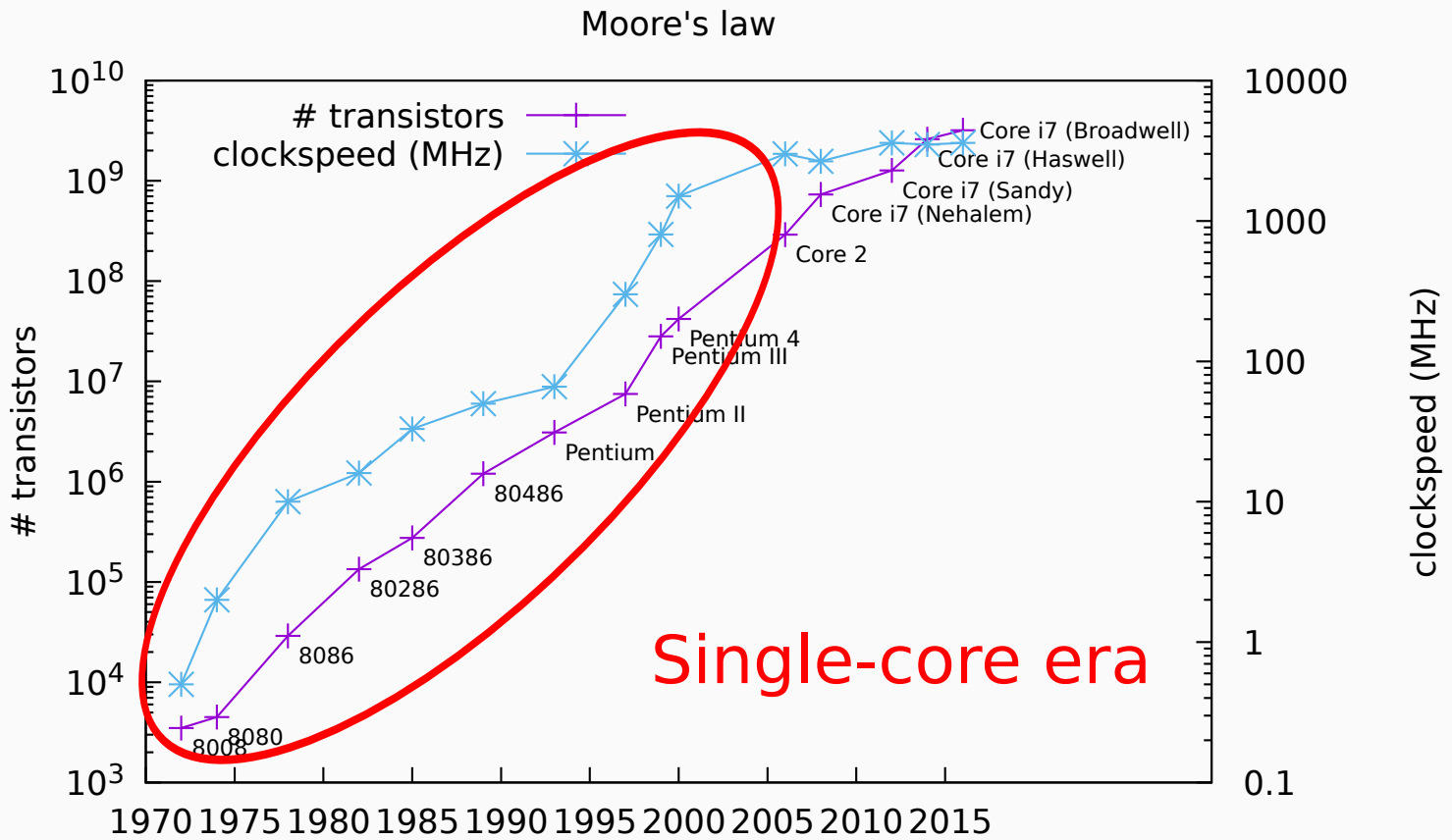
Look into the Past



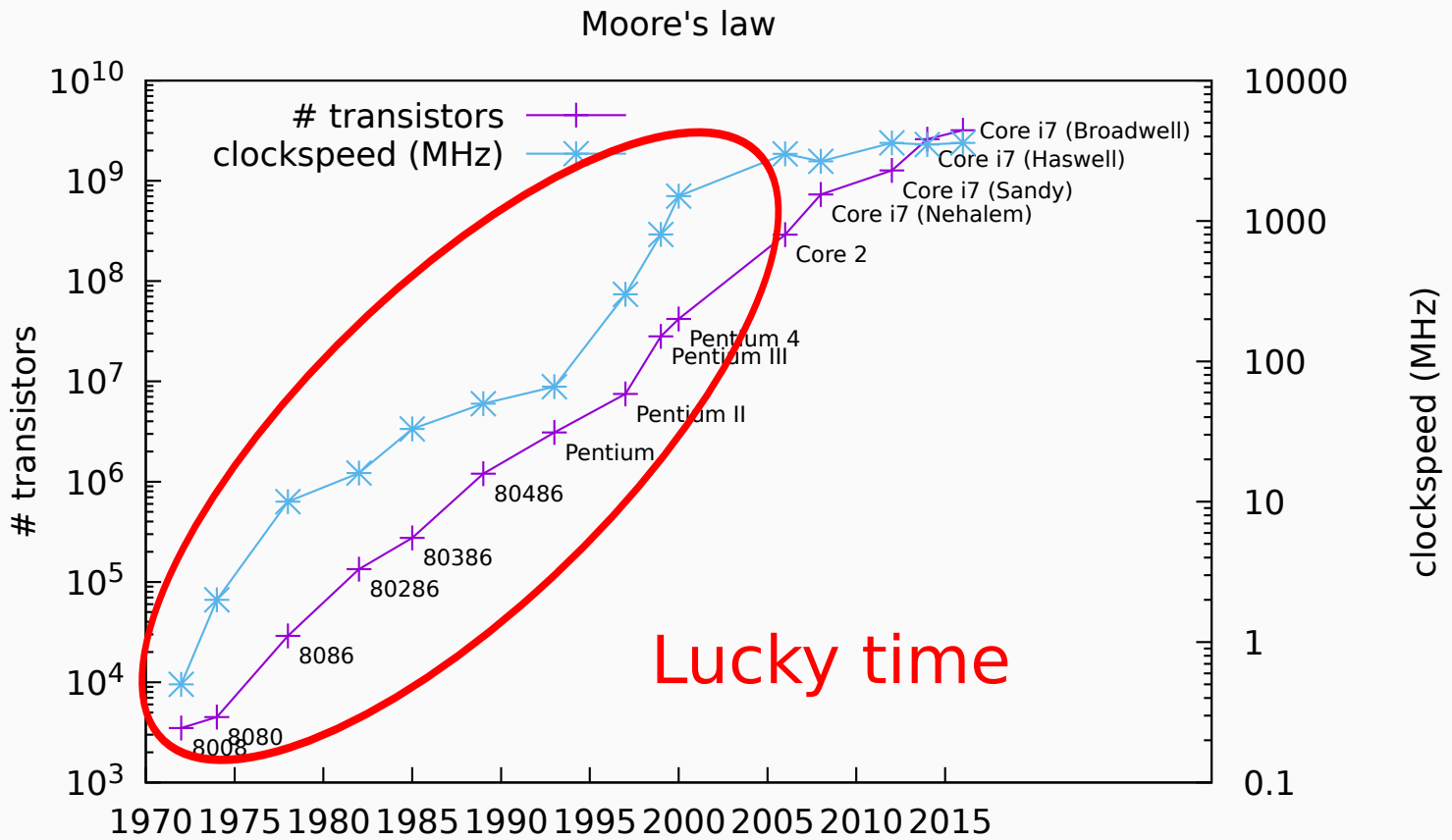
Look into the Past



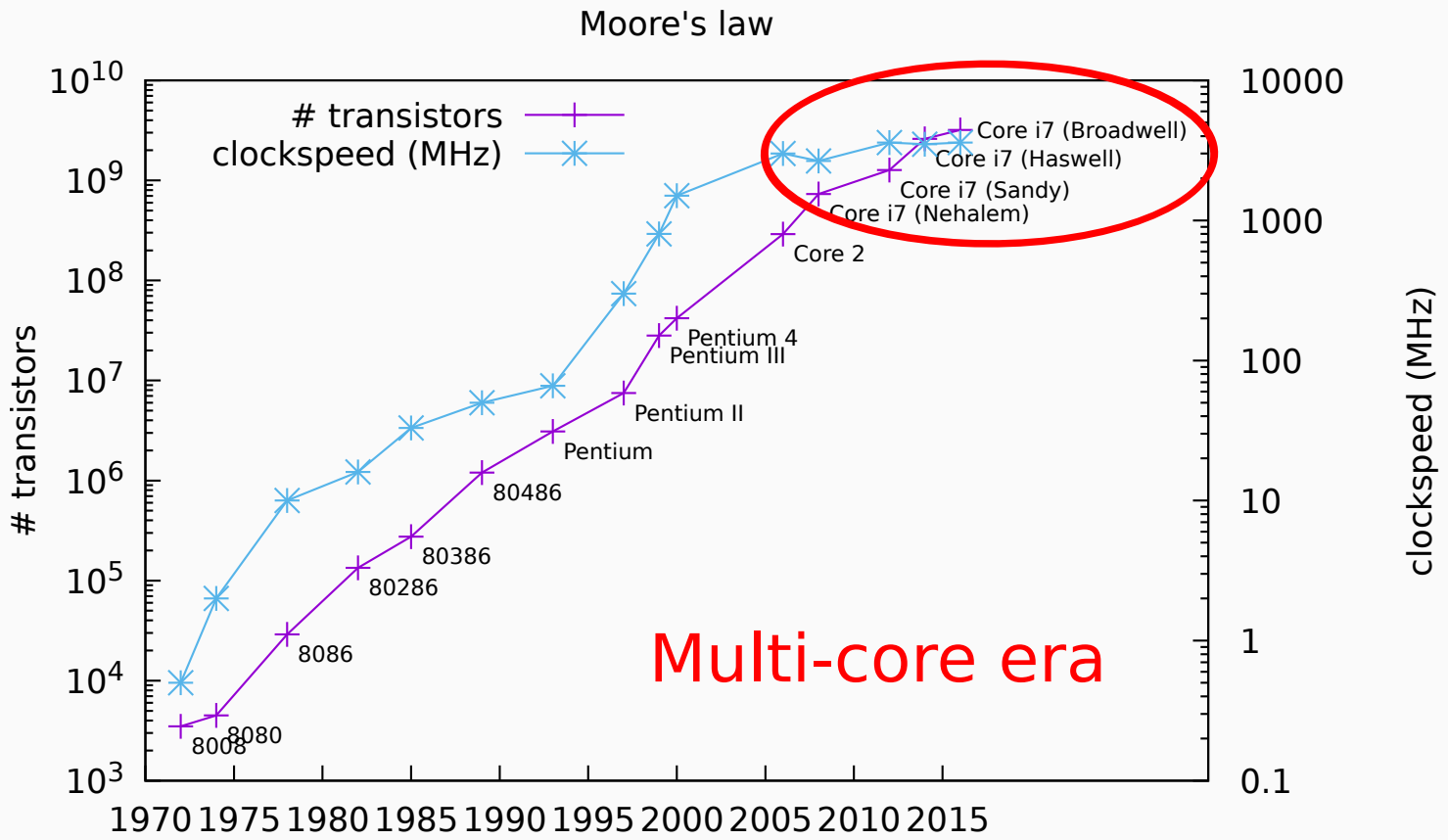
Look into the Past



Look into the Past

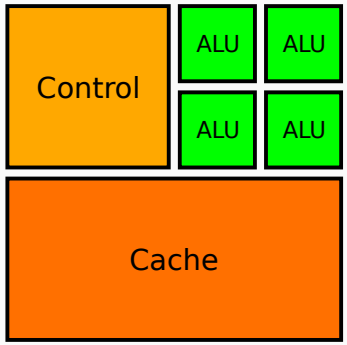


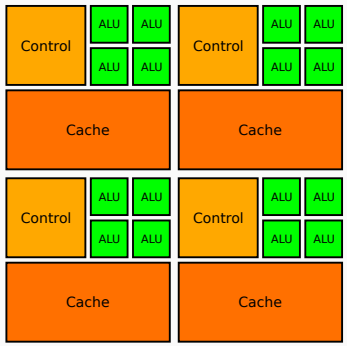
Look into the Past

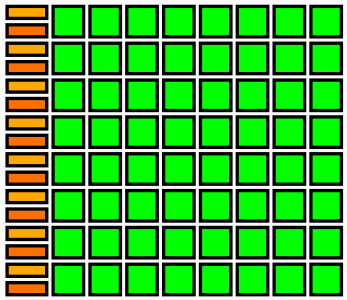


Processor types

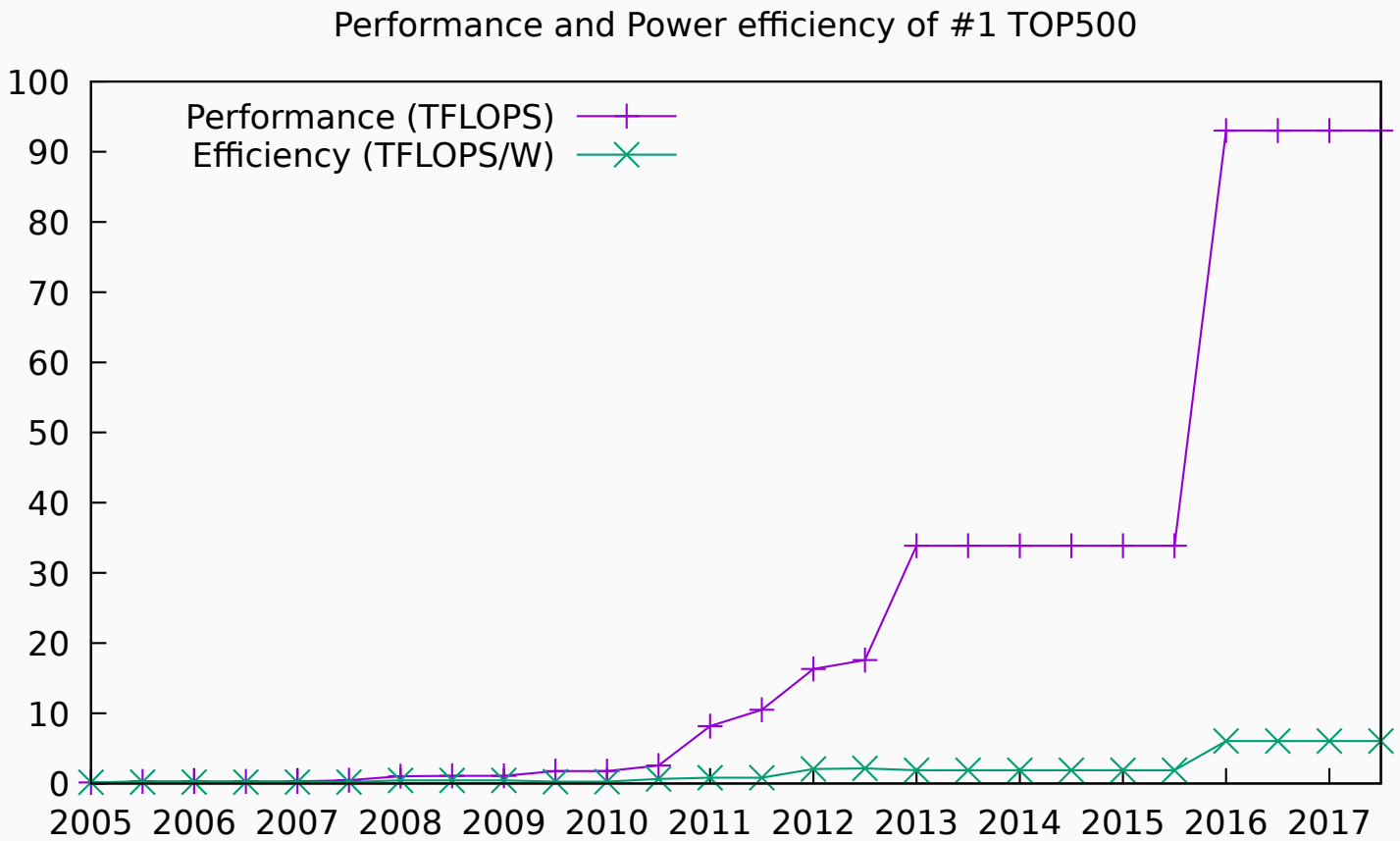
- Single-core
  - Optimized for latency
- Multi-core
  - Still optimized for latency, but just more than one
- Many-core
  - Optimized for throughput
  - High performance/Watt







Performance per Watt



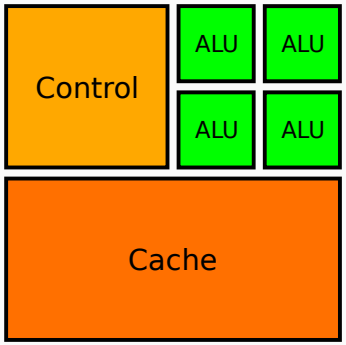
Many-core processors

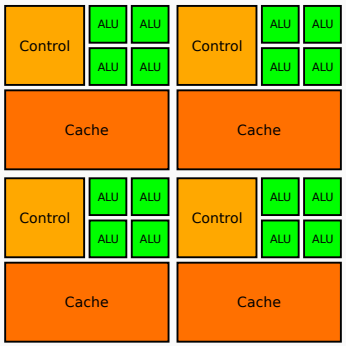
features

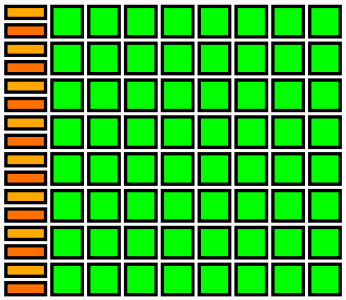
- throughput oriented
- fast evolution of the architecture
- architectural features for high performance

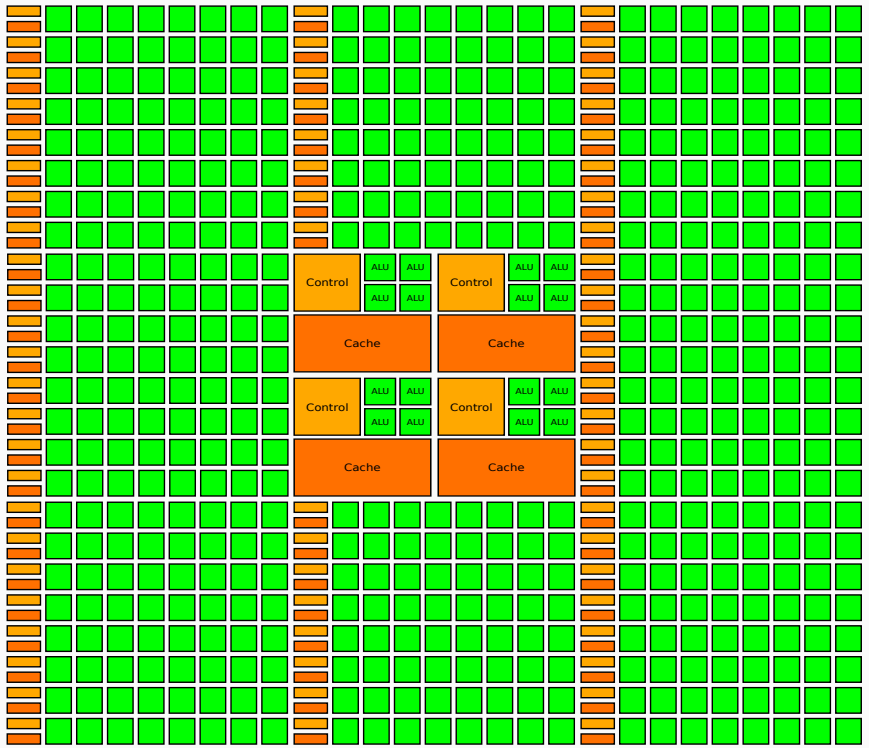
Difficult to program, especially for high-performance

Future processors

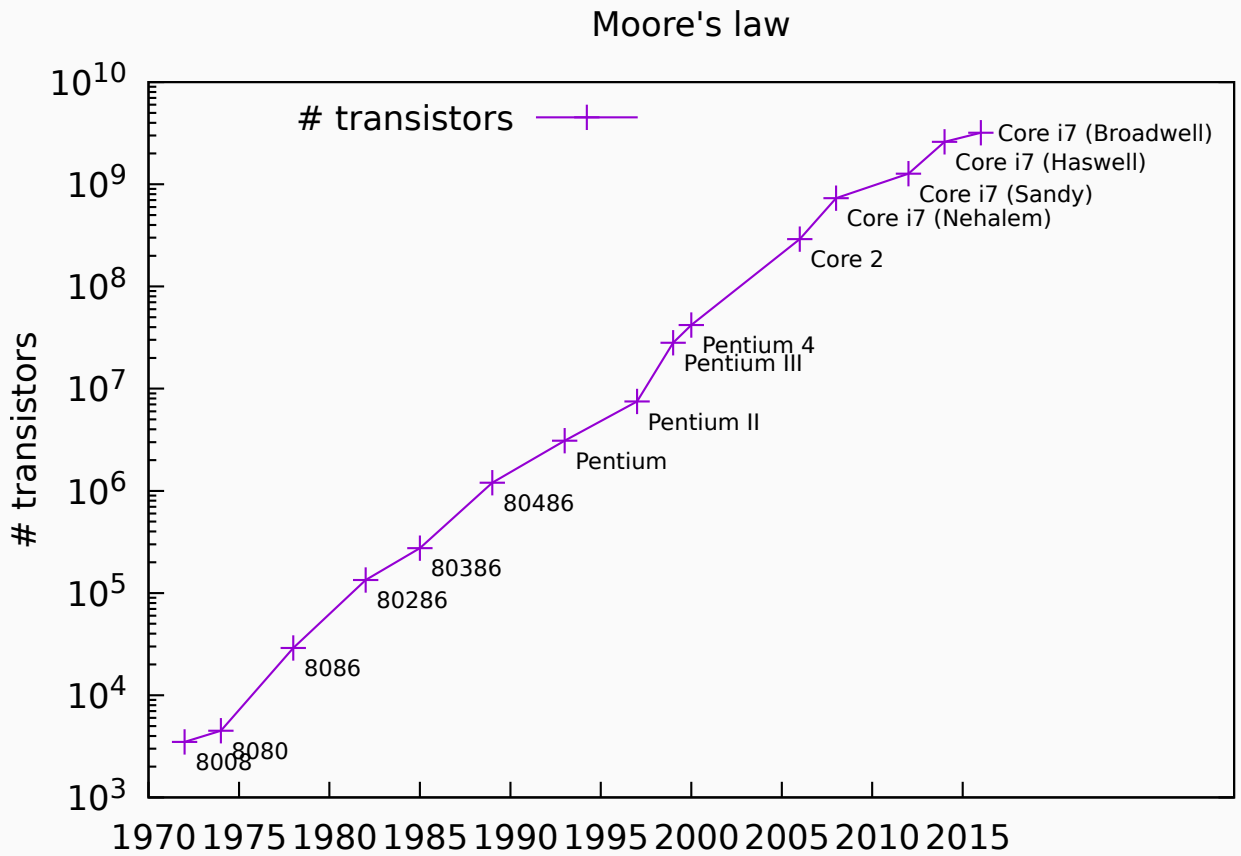




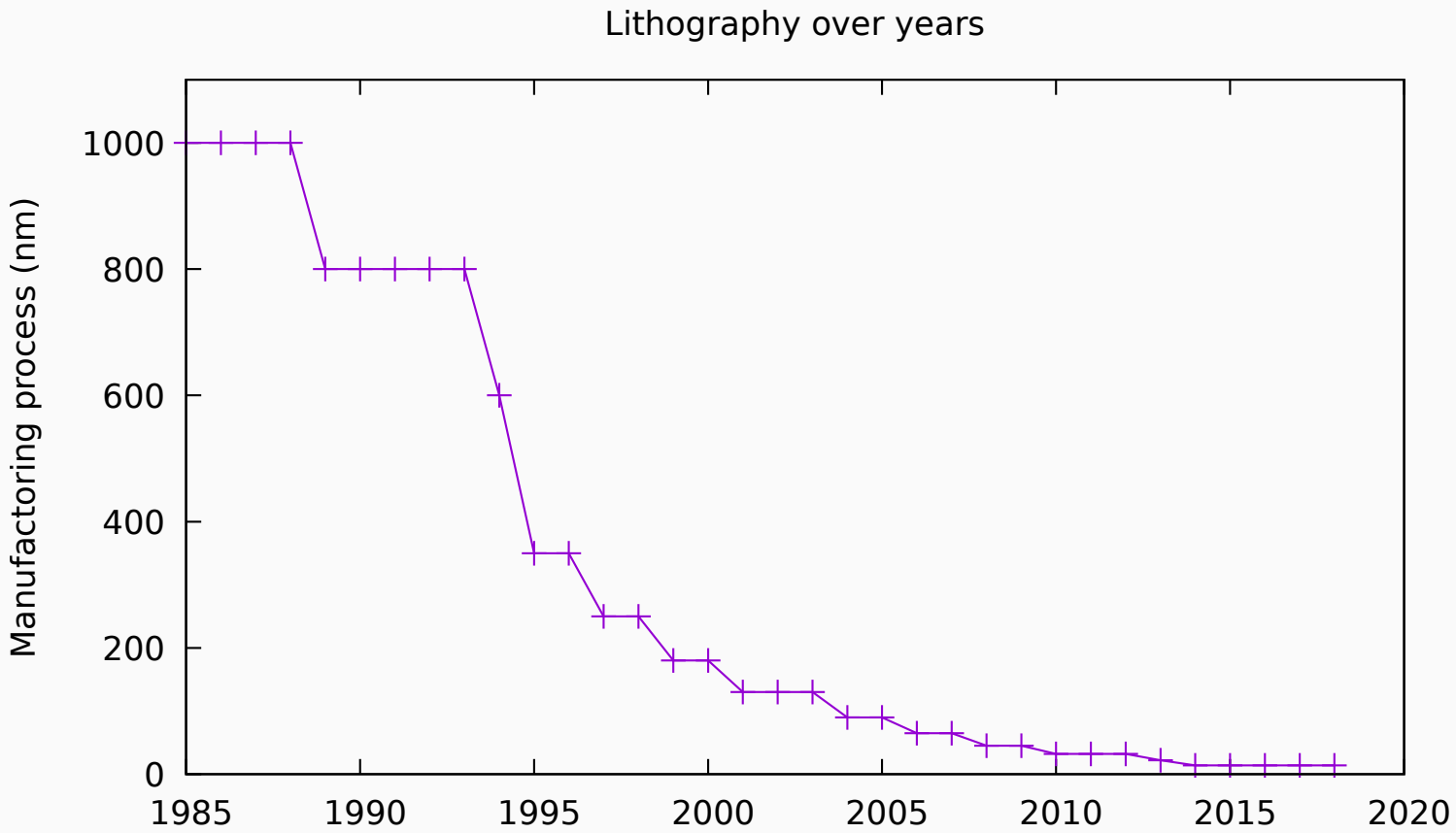




Moore's Law


Moore's law
transistors
10^{10}
10^{9}
10^{8}
10^{7}
10^{6}
10^{5}
10^{4}
10^{3}
1970 1975 1980 1985 1990 1995 2000 2005 2010 2015
Core i7 (Broadwell)
Core i7 (Haswell)
Core i7 (Sandy)
Core i7 (Nehalem)
Core 2
Pentium 4
Pentium III
Pentium II
Pentium
80486
80386
80286
8086
8080
8008

Moore's Law ending



Walls

- energy wall
- memory wall
- Moore's law $\rightarrow$ Moore's wall

result
hardware without compromises to the interface to programmers $\rightarrow$ difficult to program $\rightarrow$

- programming wall

Large demand for computational power

Chemistry

- in vitro $\rightarrow$ in silico

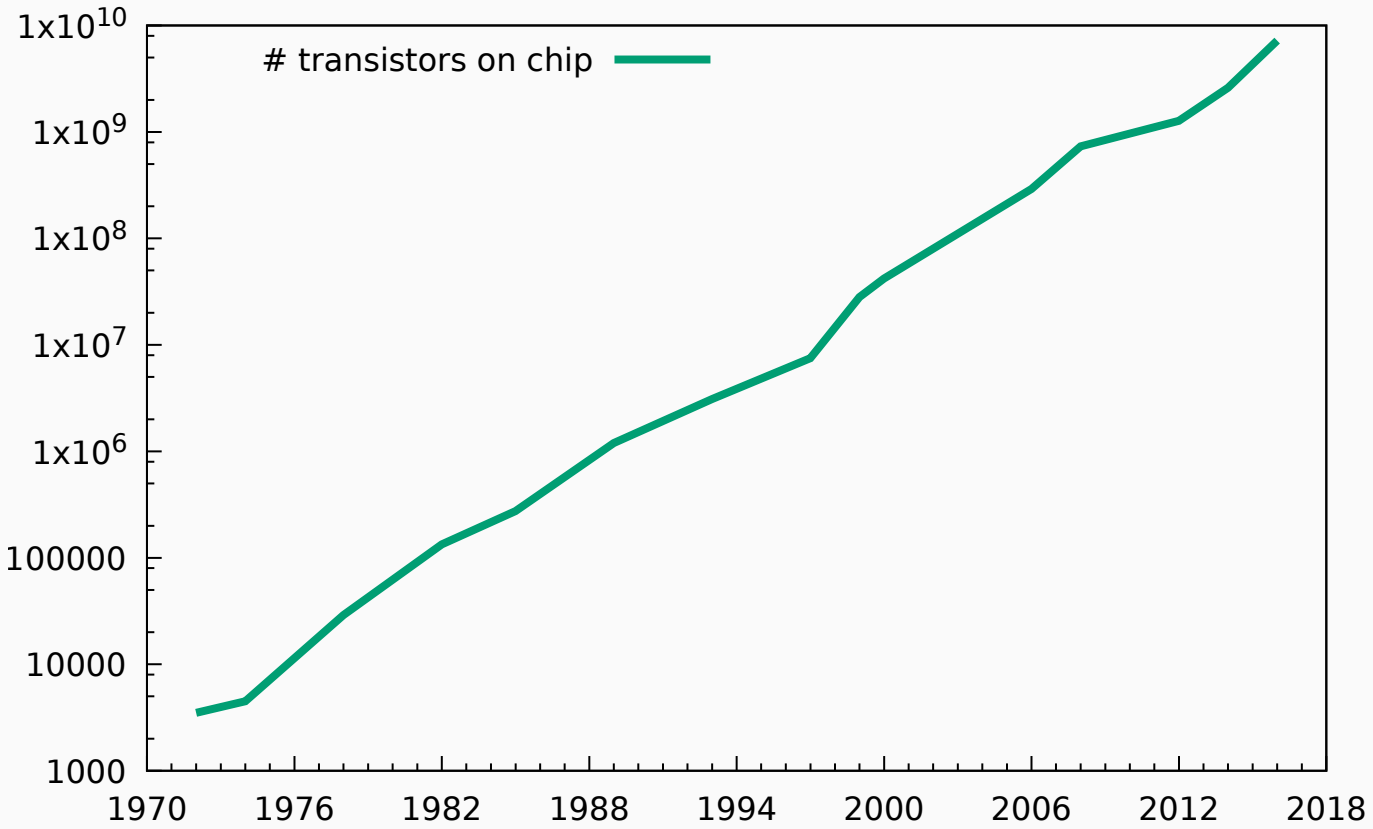
Machine Learning

- Shooting with a computational cannon

Increase in data to process

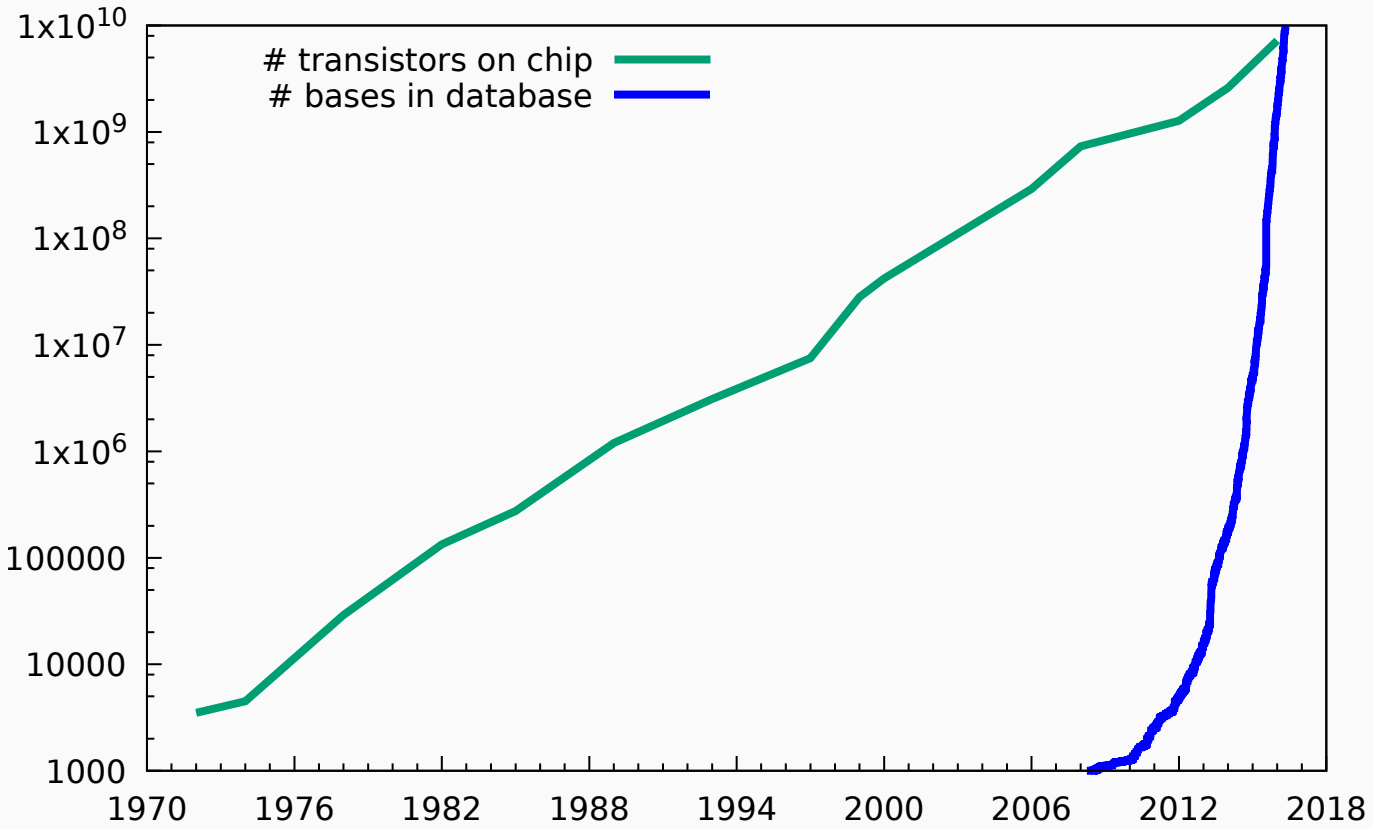
- For example gene-sequence alignment

Increase in data



Moore's law

Increase in data



Moore's law against the SRA genetic database.

Many-core era

- window of 5-10 years to figure out:
  - what hardware is going to look like
  - how to program for performance well

Recap

To deal with energy problems hardware will be:

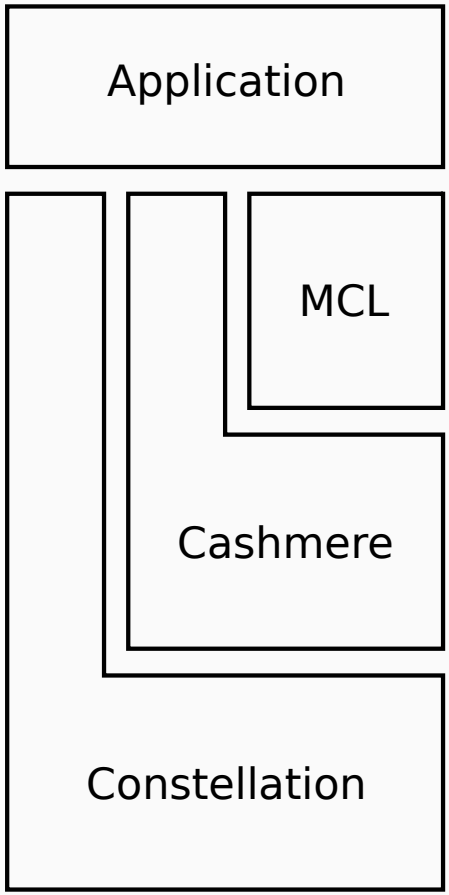
- highly parallel
- throughput oriented
- architectural details for performance
- difficult to program

Result

- More responsibility for software developers
- Increase in performance relies on software

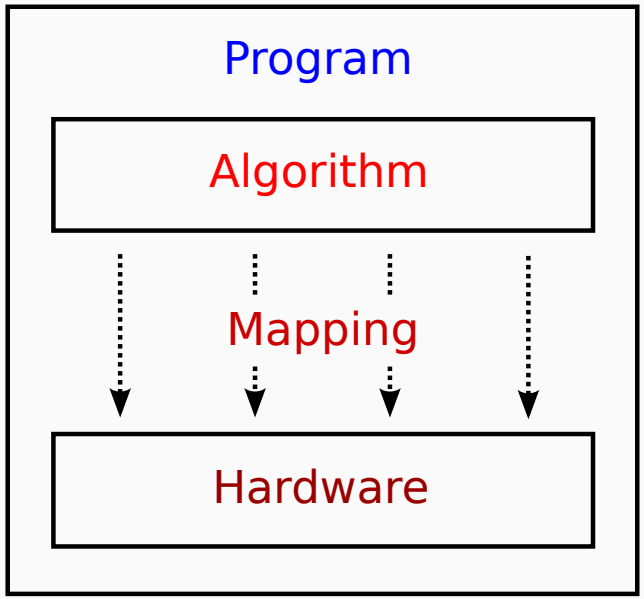
Ecosystem for Performance and Correctness

- clusters of many-cores
- obtain high performance
- understanding performance
- correctness with model checking



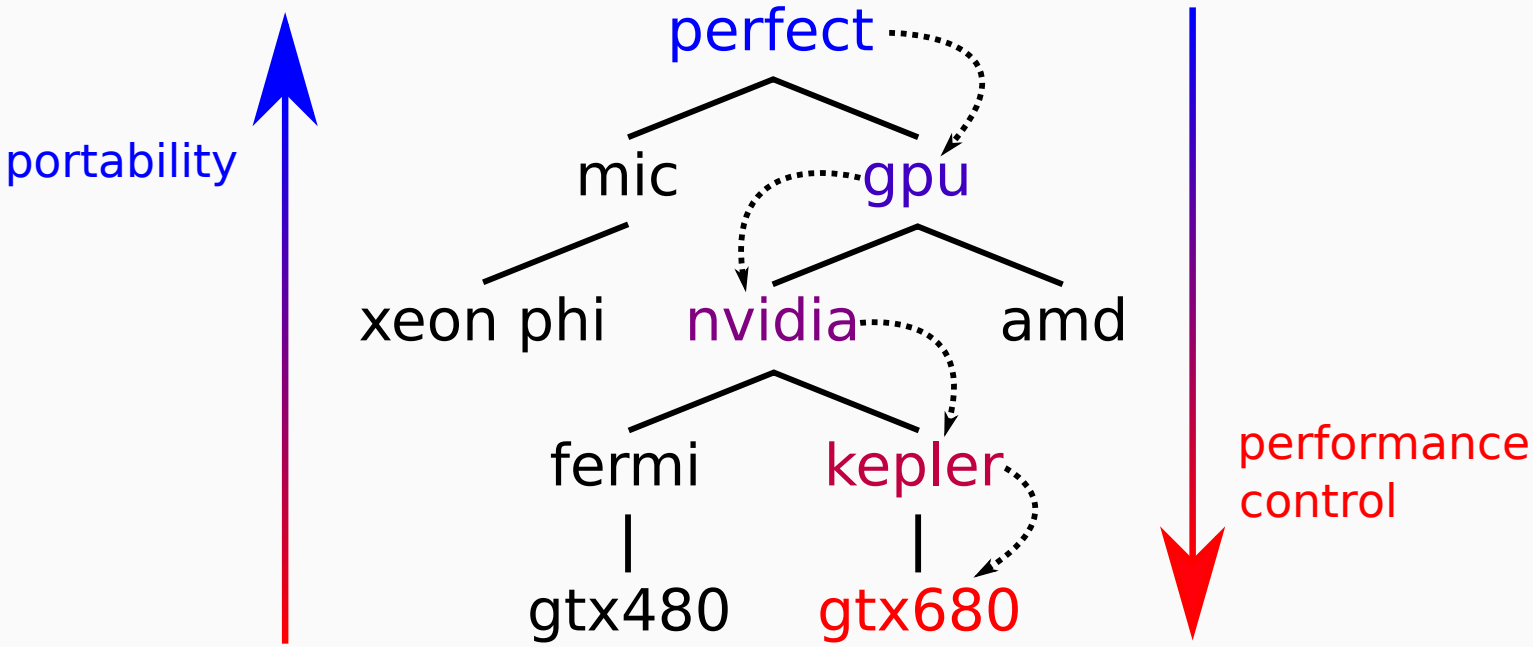
Programming in MCL

A program is an algorithm mapped to hardware

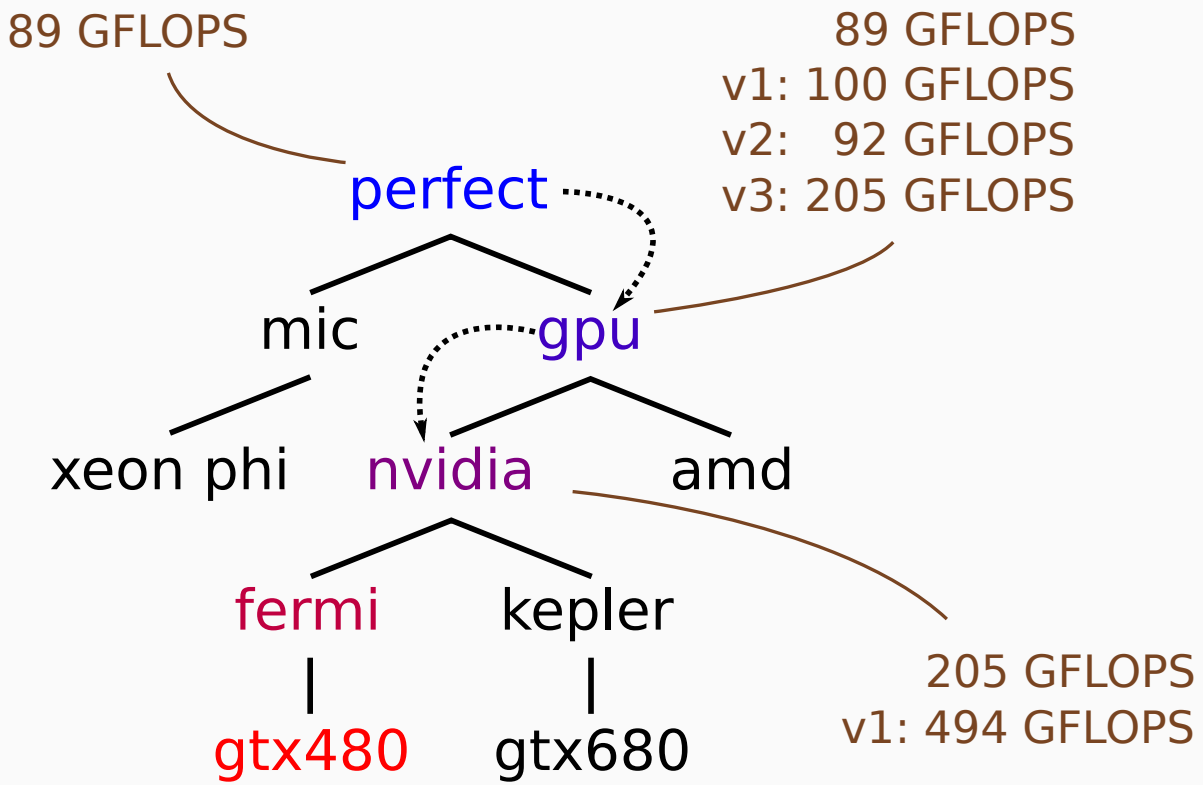


Solution
Incorporate hardware descriptions in the programming model

Hierarchy of hardware descriptions



Stepwise-refinement for performance



Feedback

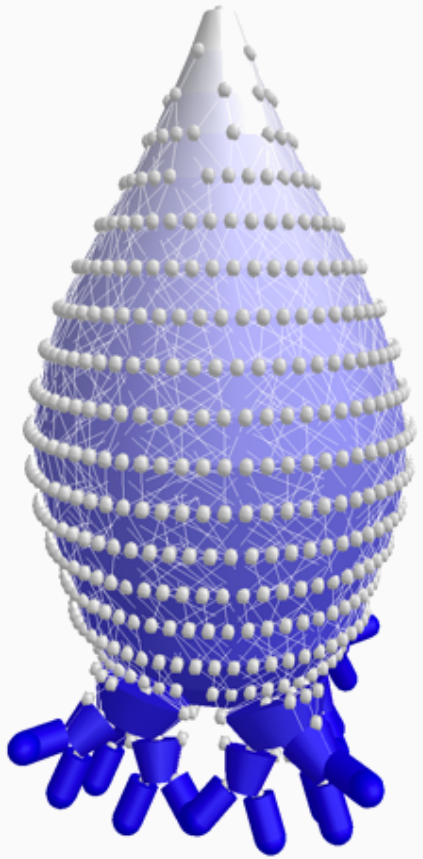
Using 1/8 *blocks* per *smp*. Reduce the amount of *shared* memory used by storing/loading shared memory in phases

Model checking: mCRL2

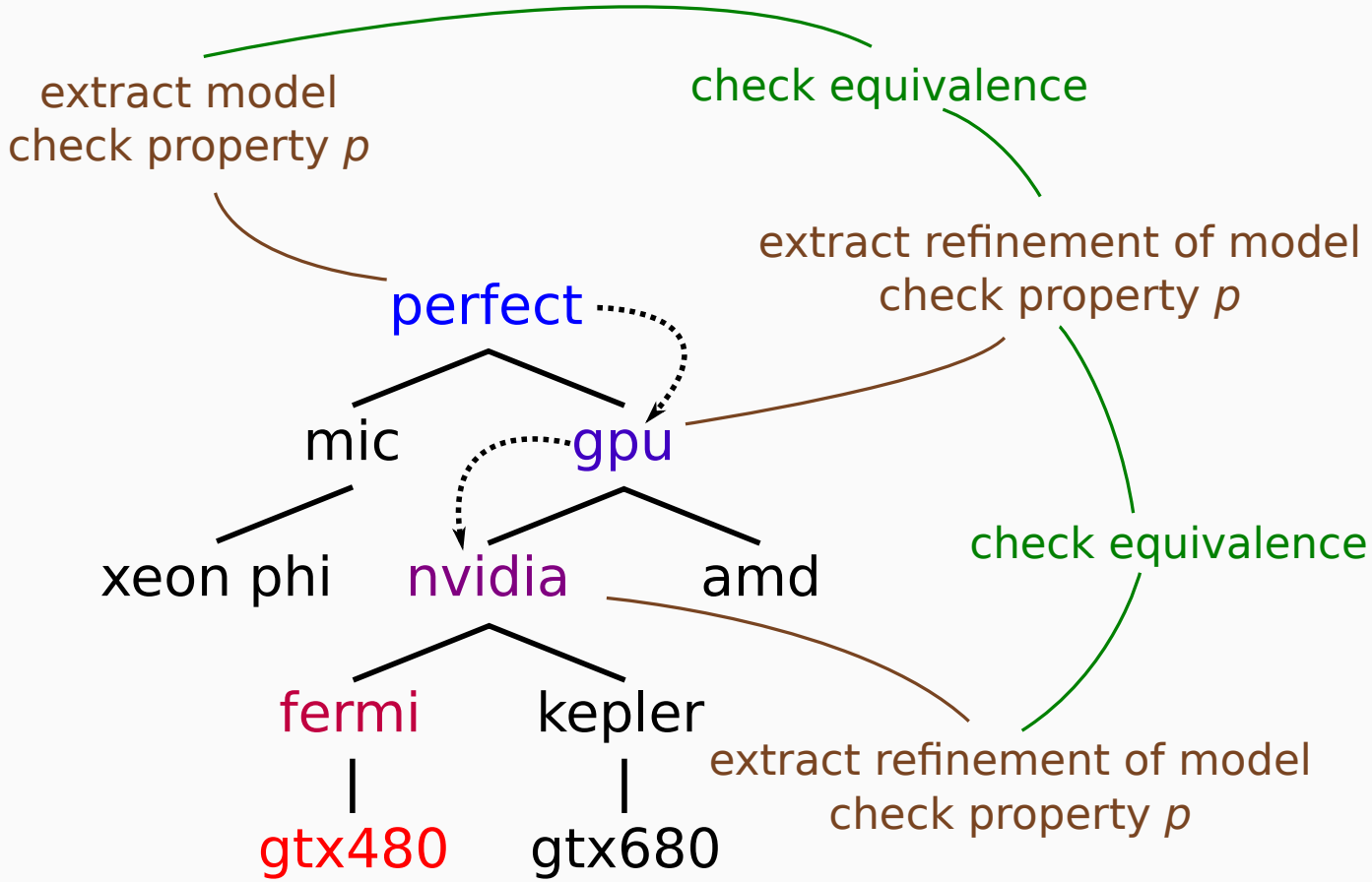
- effective tool for software flaws
- support rich data structure
- versatile
  - memory access problems
  - correctness of optimizations

Goals

- non-intrusive
- feed back verified properties into the compiler for optimization

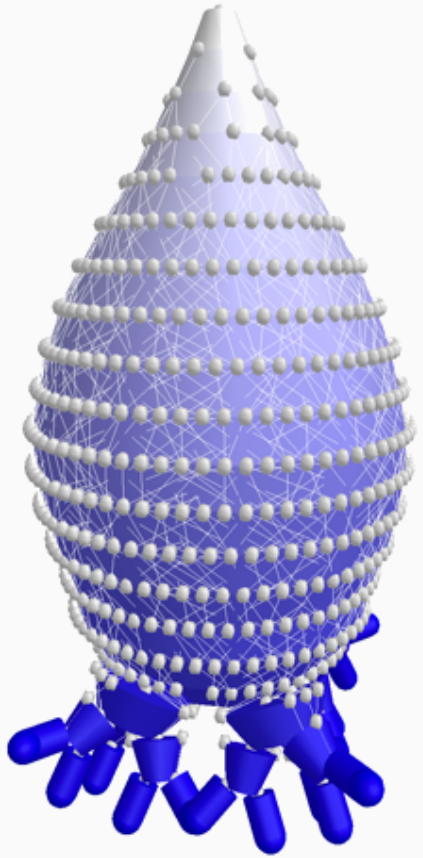


Performance-correctness co-refinement



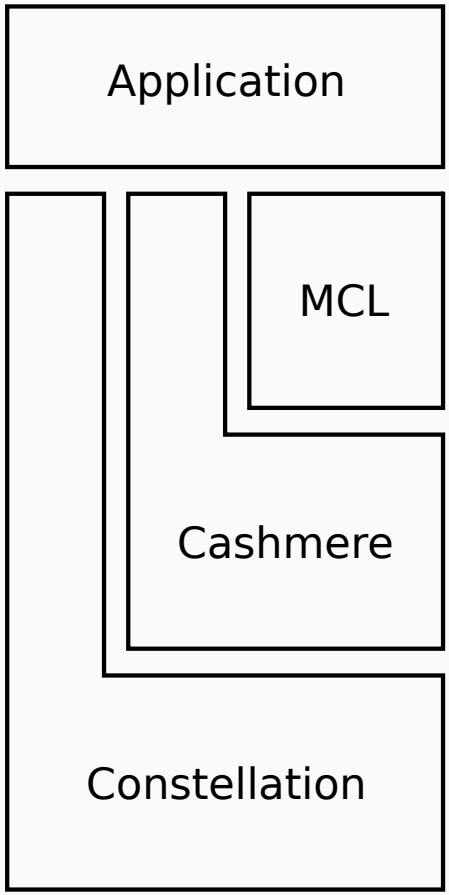
Accelerating Verification

- exploit symmetry in many-core programs
- use many-cores to accelerate model checking
    - accelerate the term-rewriting core in mCRL2

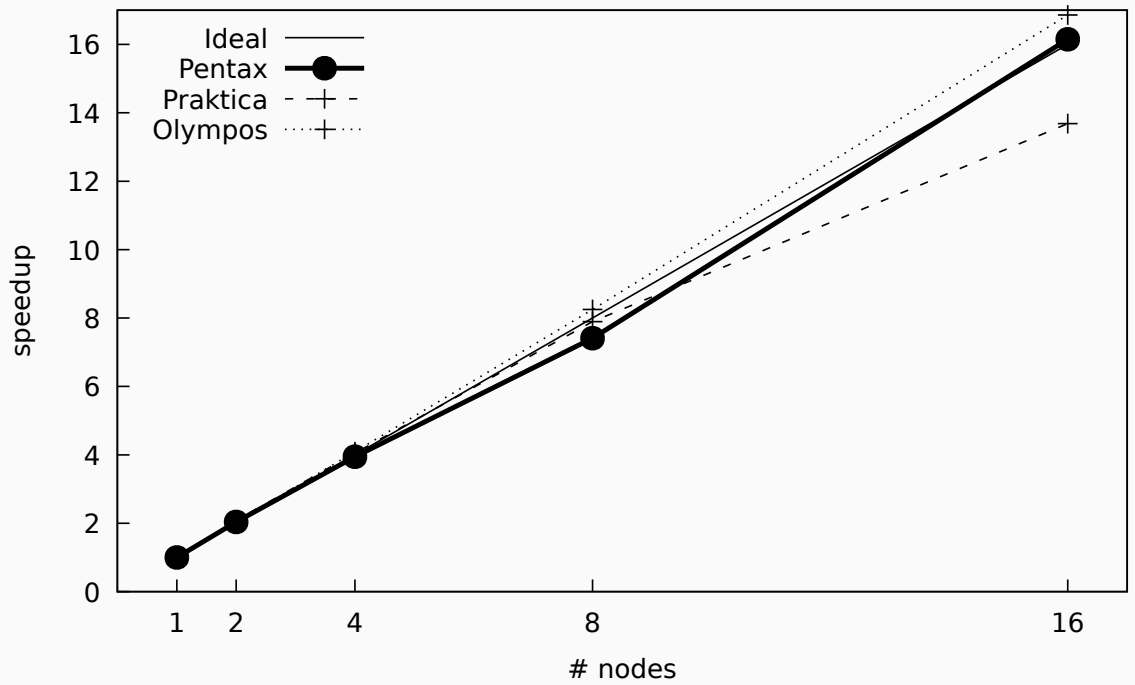


Many-core cluster computers

- Supports heterogeneous many-core clusters
- Can handle large-scale applications
- Excellent load balancing and scalability

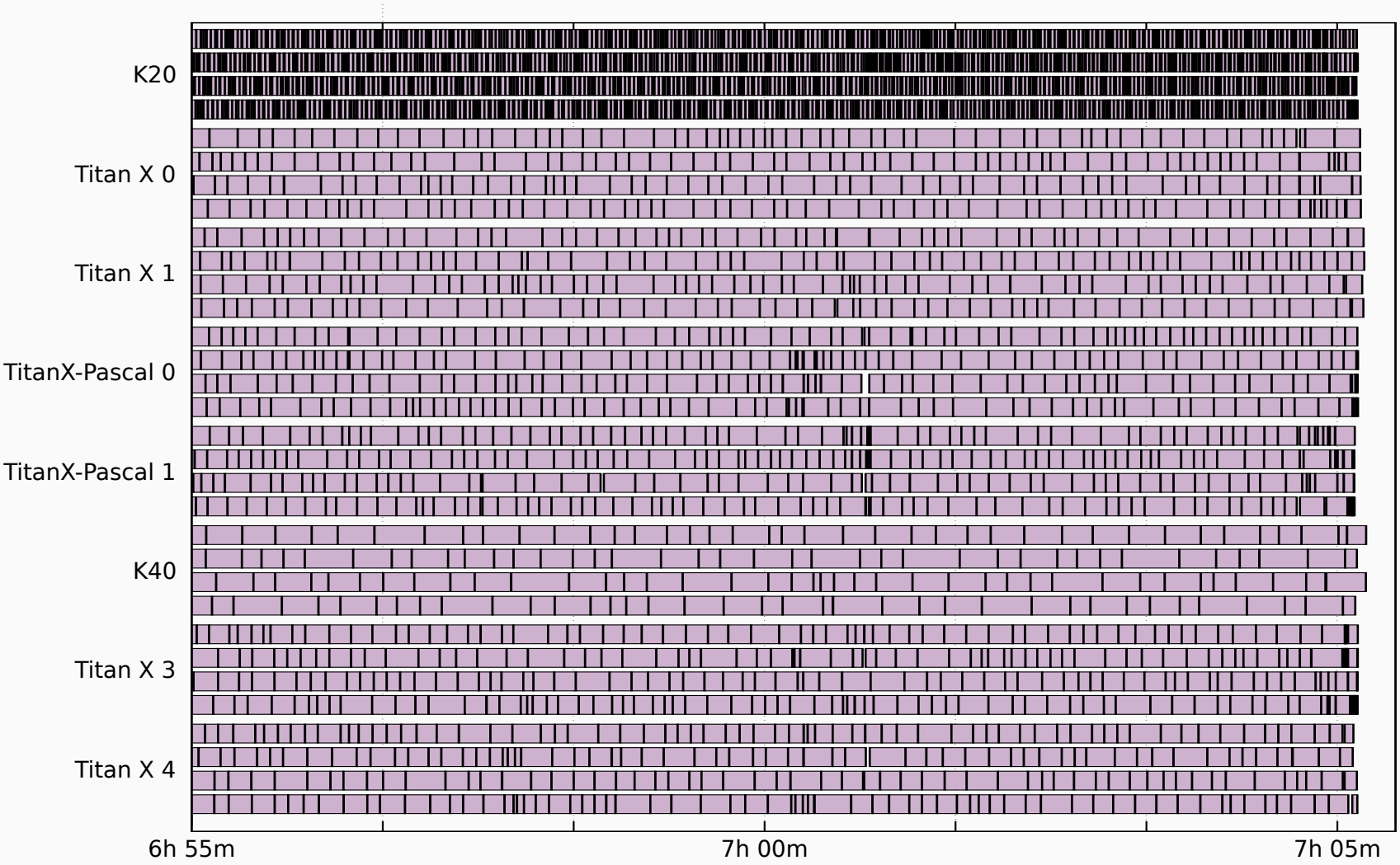


Scalability results forensics application



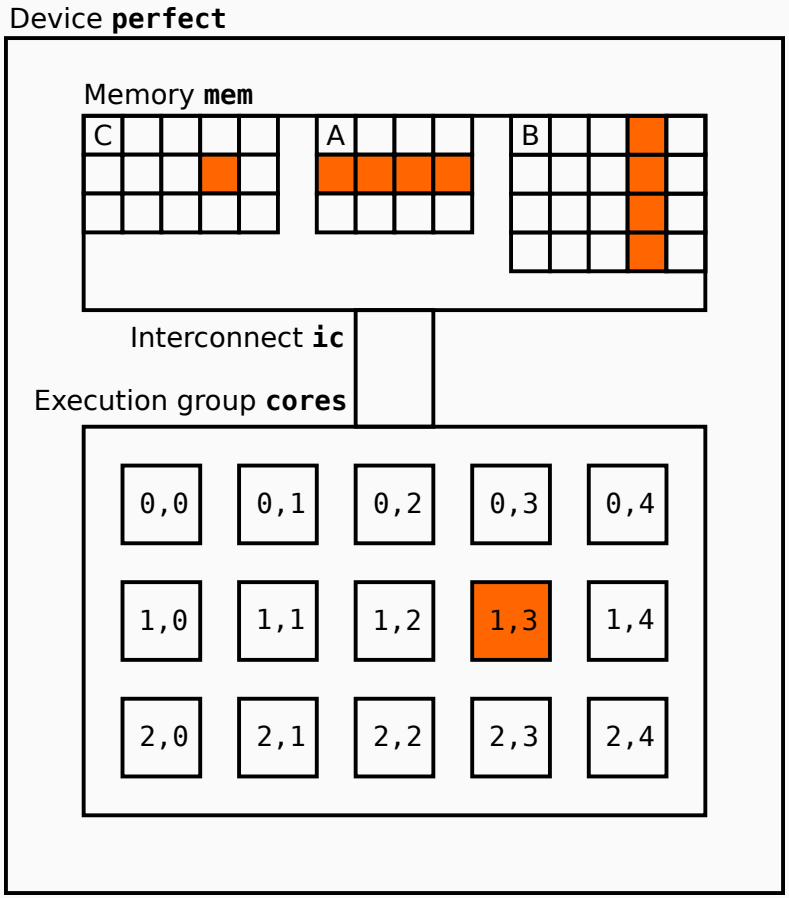
name data set	Pentax	Praktica	Olympos
number of images	638	1095	4980
#jobs	2075	1128	73920
time 1 node	47m 14s	44m 44s	53h 25m
time 16 nodes	2m 55s	3m 16s	3h 10m

Load balancing



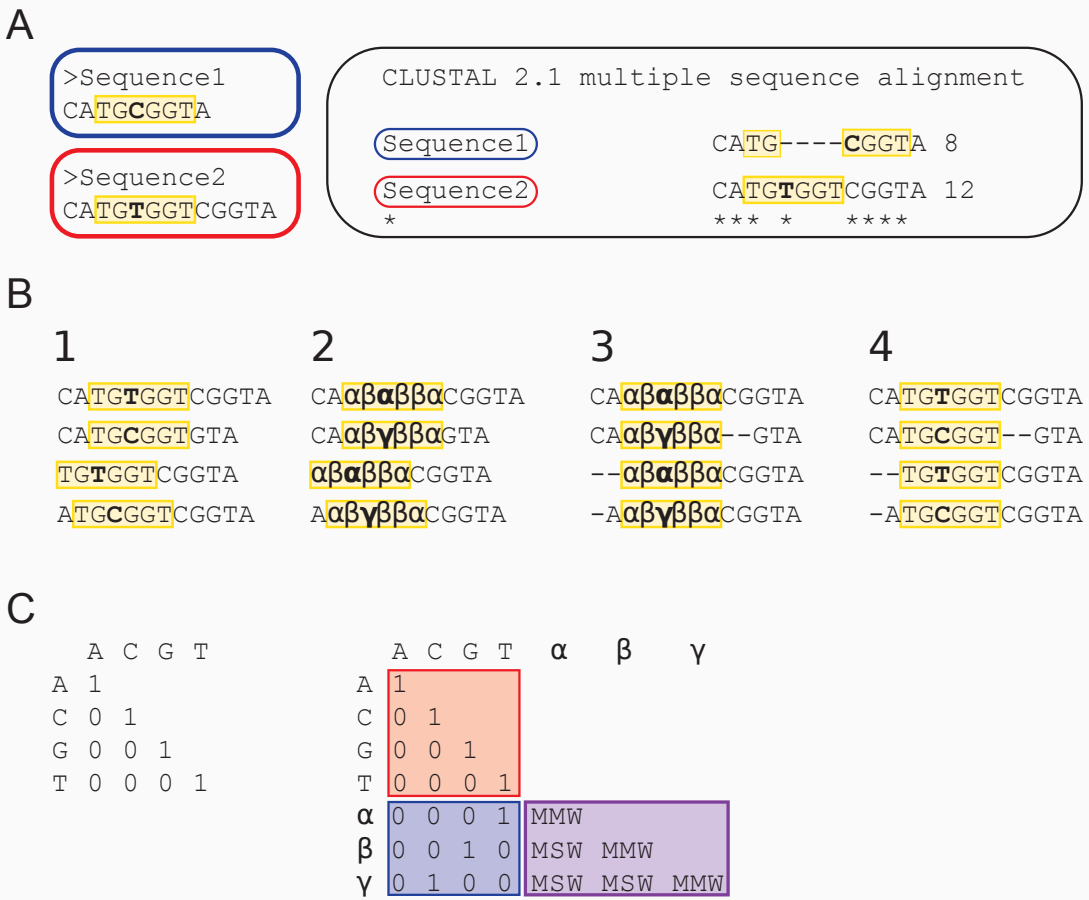
Visualizing kernel execution

Hardware descriptions designed such that they can be drawn:



Bioinformatics application

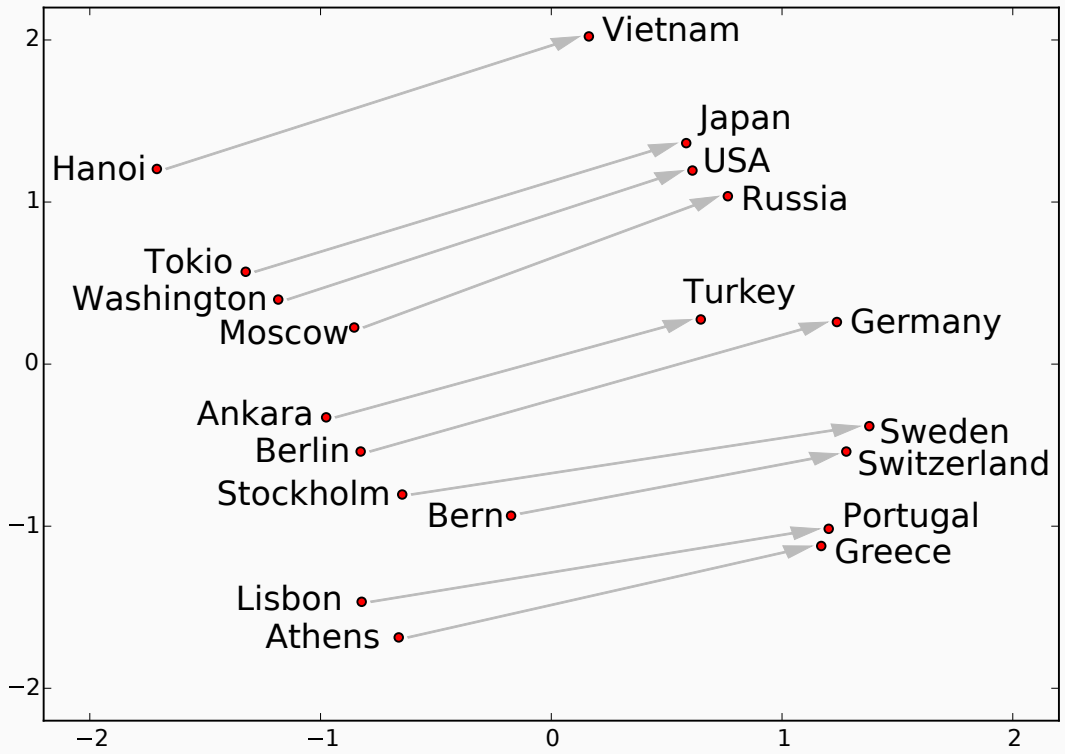
Motif-aware multiple sequence alignment



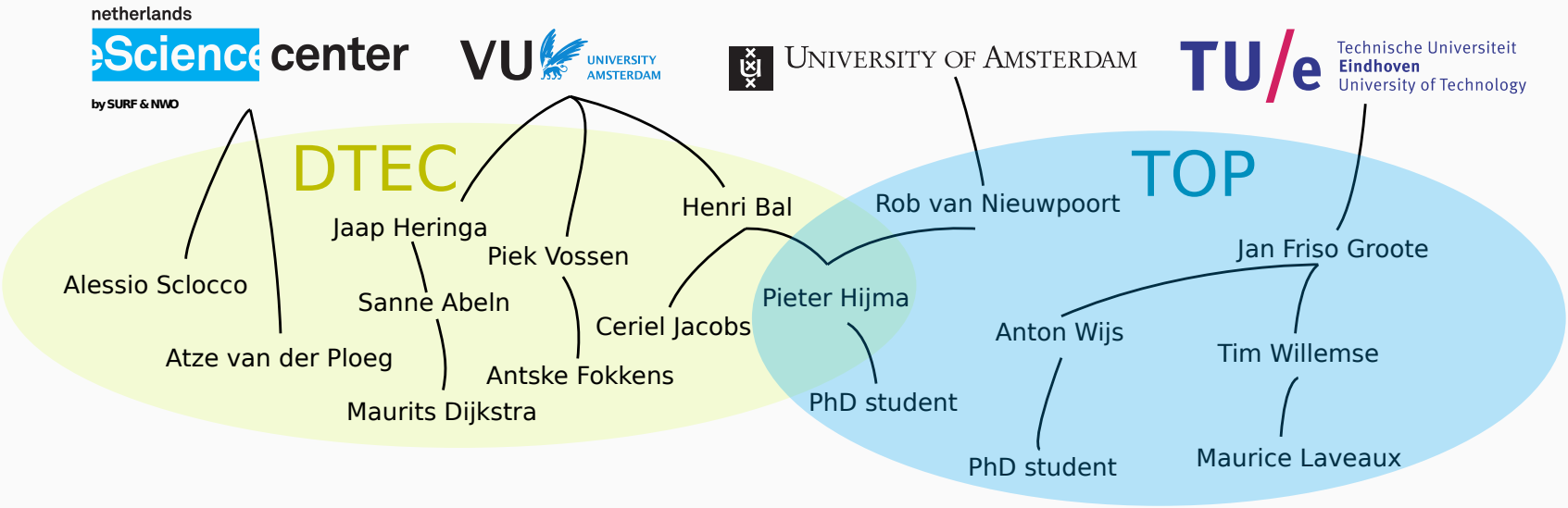
Natural Language Processing application

Word embeddings

- Map words to vectors or real numbers (word2vec)
- Take large corpus, create large multi-dimensional vector space



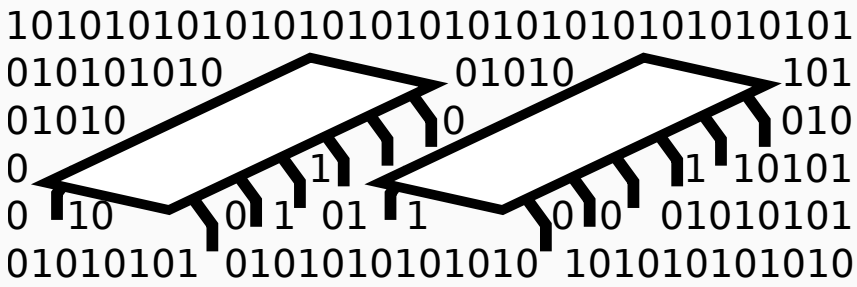
Overview people



Conclusion

In coming years, software will have to adapt to hardware more than previously.

Our solution: An Ecosystem for Combining Performance and Correctness for Many-Cores



FPGAs

- high-level synthesis: as successful as automagically parallelizing compilers
- only for:
  - extremely low latency applications
  - extremely power efficient
  - prototyping hardware
- tools are of low quality