Informatics within CSE
… and a few more considerations ☺

Hans-Joachim Bungartz

TUM – Department of Informatics, Chair of Scientific Computing
Leibniz Supercomputing Centre

Kick-off meeting of the new GAMM Activity Group “Computational Science and Engineering”

Garching, September 17, 2012

First Remarks

Yes, we need this – and maybe more – CSE-specific (sub-)organizations

It is good to have it within a broad & inherently cross-disciplinary society

It should be seen as a new ally & mouthpiece of and for our field

It should not be seen as something diminishing any existing activity

Therefore: thanks to our three “drivers”, and thanks to GAMM!

Notions and Names … Kind of a Mess?

Driven by history, domains, funding interests, …

Computational Engineering

Computational & Data-Enabled Science & Engineering

Computational Science & Engineering

Simulation Sciences

Simulation-Based Engineering

Computational X
(X ∈ {Mechanics, Physics, Biology, …},
X = my_field ➔ CX is mine, too)

Scientific Computing

Computational Engineering Science

Notions and Names … Let's Take the Umbrella!

Computational Engineering

Computational & Data-Enabled Science & Engineering

Computational Science & Engineering

Simulation Sciences

Simulation-Based Engineering

Computational X
(X = my_field ➔ CX is mine, too)

Scientific Computing

Computational Engineering Science

Challenges – defining the field

- multi-level
- multi-modal
- multi-scale
- multi-disciplinary
- multi-dimensional
- multi-core
- multi-physics

From data / images / numbers …
… to information / insight

From counting operations …
… to energy awareness

From qualitative descriptions …
… to quantitative prediction

From sequential algorithm design …
… to massive parallelism

From simulation …
… to optimisation

From parameter assumptions …
… to identification & estimation

From one-way batch jobs …
… to user interaction

From simple tools & codes …
… to 2x complex ones

From forward problems …
… to inverse problems

From heroic PhD codes …
… to large teams / SW

From deterministic models …
… to random & uncertainty

From hacker's delight …
… to complex workflows

From one (spatial/temporal) scale …
… to cascades of scales

From island fun …
… to embedding & integration

From single-physics problems …
… to coupled scenarios

From flat algorithms & data …
… to hierarchy

Multi-Disciplinarity ... Fascinating, but not Easy

There are, at least, six levels of multi-disciplinarity ☺ :

Level 0 – "we don't need anything, we are intrinsically interdisciplinary"

… the physics point of view

Level 1 – "let's buy and read a book from another discipline"

… the fallback position if level 0 does not apply at all

Level 2 – "let's hire a student assistant from another discipline"

… only if level 1 does not work at all

Level 3 – "let's hire a research assistant from another discipline"

… it starts to get painful

Level 4 – "let's ask a colleague from another discipline"

… only in case of complete despair

Level 5 – Insight: "we are unable to solve the problem on our own"

… almost never with a professor

My Understanding: Multi-Disciplinarity …

… means "no owners, but many contributors"

… implies that such a topic can never be a sub-field / sub-community of a single discipline

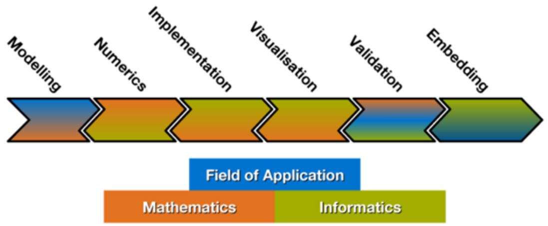
… comprises the curiosity to deal and get in contact with something new, but not to claim it for yourself

… should never be considered as a threat ("these XYZ people believe they can do everything"), but as a chance of broader support & expertise

… requires respect concerning the contributions of others (cf. the discussions on the importance of models – algorithms – software)

➔ **hence: don't focus on classical disciplines too much – CSE is more than "ApplMath + CS + Eng/Sci"**

CSE –
Key Technology for Science & Industry

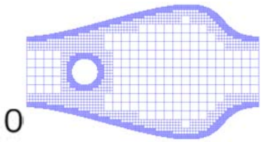


Mathematical model

$$\nabla \cdot \mathbf{u} = 0$$
$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\,\mathbf{u} + \frac{1}{\rho}\nabla p - \nu\,\Delta \mathbf{u} = \mathbf{0}$$

Discretization & solver

$$M\mathbf{u}_h = 0$$
$$A\dot{\mathbf{u}}_h + D\mathbf{u}_h + C(\mathbf{u}_h)\mathbf{u}_h - M^T p_h/\rho = \mathbf{0}$$



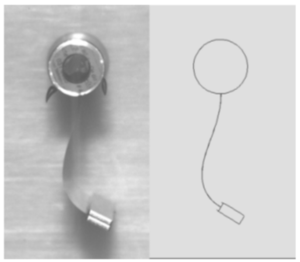
Impact of each step on all other steps!

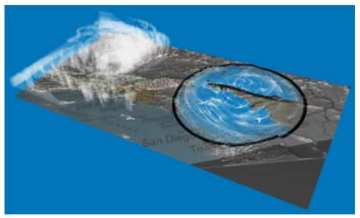
Hence #1: no pipeline any more, no cycle, but a complete graph

Hence #2: need for co-design – or a number of co-designs

Parallel implementation, HPC

Validation





Software

Insight, Design

Exploration

Why Informatics Should Be Involved

The hardware issue: *hardware is parallel, complex, not straightforward to exploit*

The algorithm issue: *"efficiency" has become multi-faceted – beyond O(…)*

The programming issue: *high-level coding is more than producing LoC …*

The software issue: *design & manage large systems; tool support*

The co-design issue: *systems – algorithms – applications are all interwoven*

The data issue – handling, fusion, exploration (mining, visualization), …:
cf. Richard Hamming's "The purpose of computing is insight, not numbers!"

The HPC issue: *only a minority of CSE needs HPC (cf. small-scale simulations, real-time simulations, …) – but the vast majority enjoys HPC's benefits*

Why Informatics Should Be Involved

The hardware issue: *hardware is* **parallel***, complex, not straightforward to exploit*

The **algorithm** issue: *"efficiency" has become multi-faceted – beyond O(…)*

The programming issue: *high-level coding is more than writing LoC …*

The **software** issue: *design & manage large systems; tool support*

The **co-design** issue: *systems – algorithms – applications are all interwoven*

The data issue – handling, fusion, exploration (mining, visualization), …: *cf. Richard Hamming's "The purpose of computing is insight, not numbers!"*

The HPC issue: *only a minority of CSE needs HPC (cf. small-scale simulations, real-time simulations, …) – but the vast majority enjoys HPC's benefits*

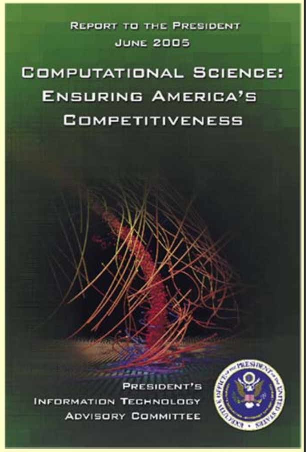
Software!

Software in CSE – State of the Art

"Today's CSE ecosystem is unbalanced, with a software base that is inadequate to keep pace with and support evolving HW and application needs."

"The crisis in CSE software is multifaceted and remediation will be difficult. The crisis stems from years of inadequate investments, a lack of useful tools, a near-absence of widely accepted standards and best practices, …, and a simple lack of perseverance by the community. This indictment is broad and deep, covering applications, programming models and tools, data analysis and visualization tools, and middleware."

PITAC report 2005



"The field has reached a threshold at which better organization becomes crucial. New methods of verifying and validating complex codes are mandatory if CSE is to fulfil its promise …."

"Verification, validation, and quality management, we found, are all crucial to the success of a large-scale code-writing project."

Post and Votta, *Computational Science Demands a New Paradigm*, Physics Today, 2005

"In many domains software engineering quality management processes like CMMI and ISO 9000 have been successful, but apparently less so in CSE, especially in HPC-related applications."

R. Kendall, *Advanced Computing Focus Day*, Informatik 2008, München

Software!

Studies on CSE-Related SW Development in the US

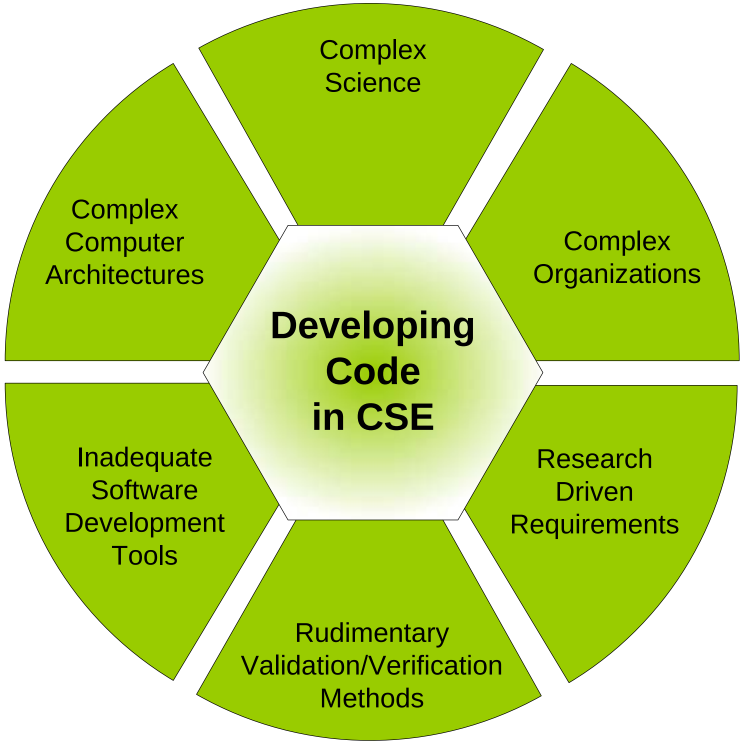
(DoD, DoE – ASCI and successors; study from 2004)

Development team size (median): 6	increasing
Code size (median): 300 k LoC	increasing
Number of users (median): 25	stable
Code project age (median): 17.5 years	increasing
Presence of Fortran: 58% (24% F77, 34% F90/95)	decreasing

Source: Kendall, Post, et al.

Software!

Some Observations on SW Development in CSE



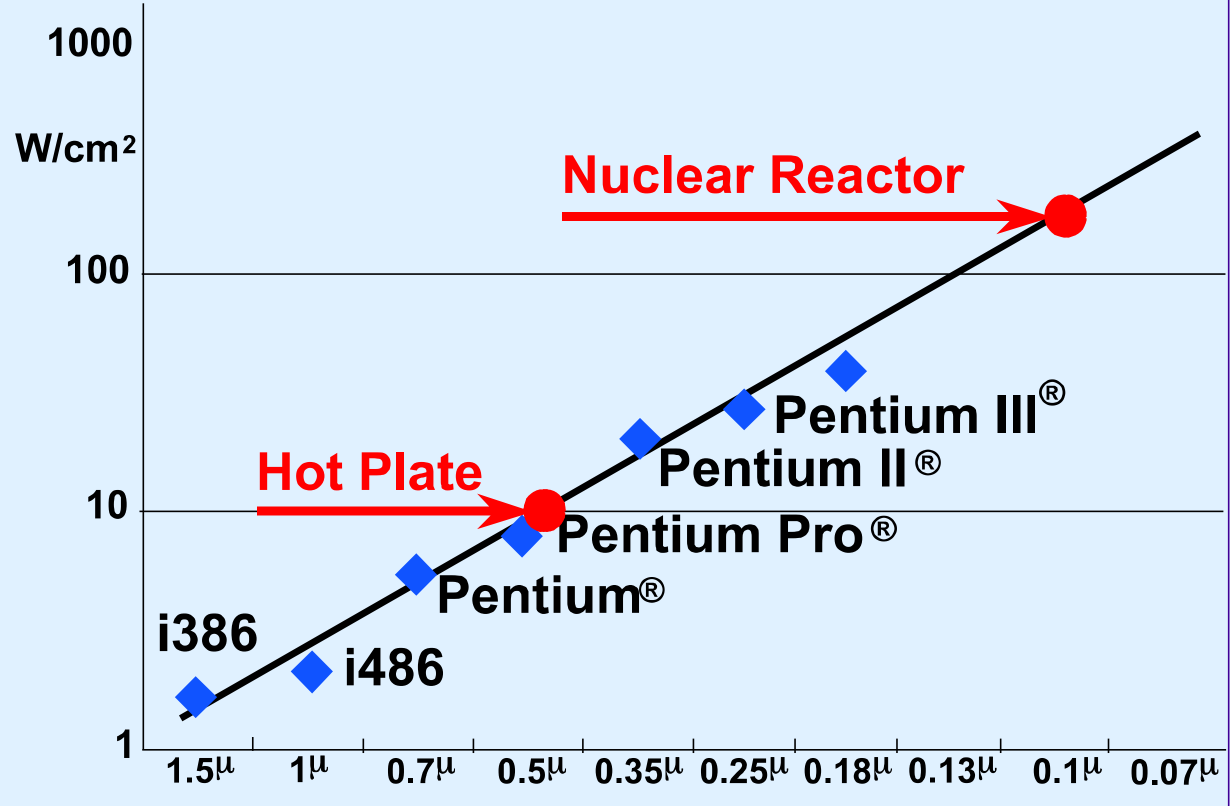
"Software development is the principal bottleneck in CSE" (R. Kendall)

Roadblocks:

- Most developers are domain scientists and engineers, not computer scientists
- Typical priority: science >> code performance >> software quality
- Intellectual level assigned:
  models >> algorithms >> programs
- No "software engineering mainstreaming": design, process models, workflow models, …
- No "team understanding": co-operative work, trans-disciplinary, project management, …
- Instead, still the lonely heroes with their heroic codes (and sometimes accumulated heroism)
- No systematic testing culture
- No formal support – verification
- No best practices
- Many groups working on PSE – in general with limited success

Energy Density – the Fundamental Problem

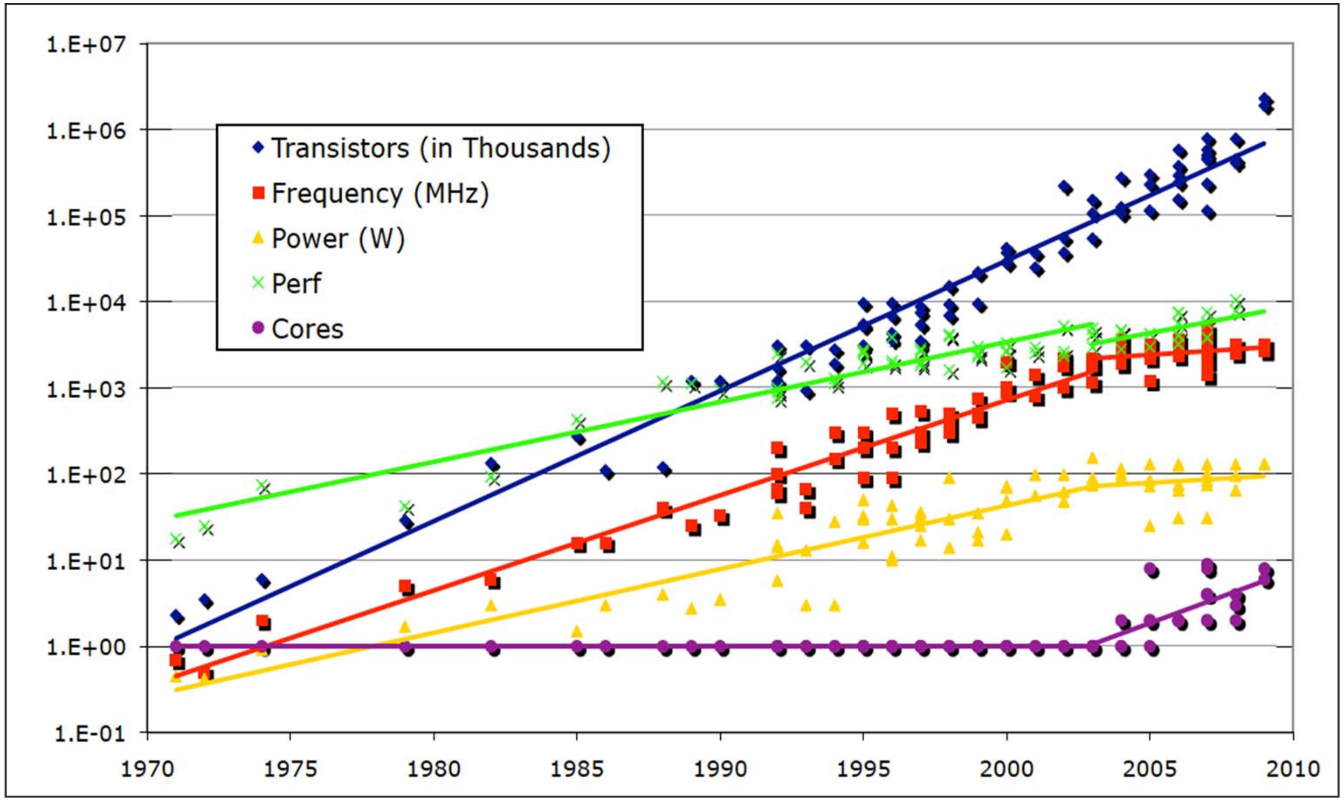
Parallel!



Source: Fred Pollack, Intel. New Microprocessor Challenges in the Coming Generations of CMOS Technologies, Micro32

We‘re no longer getting faster – we‘re getting more!

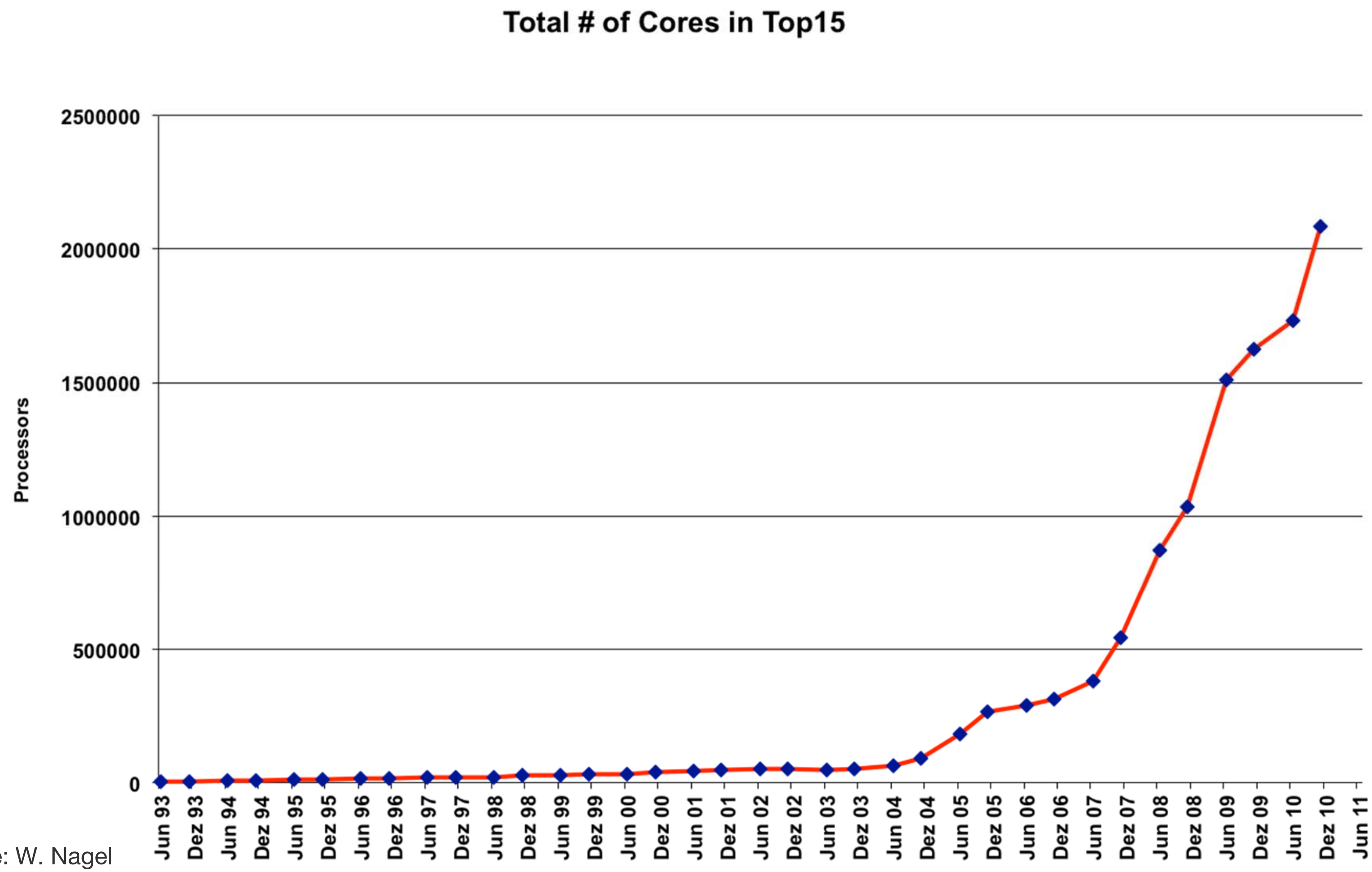
Parallel!



Kathy Yelick: *Ten Ways to Waste a Parallel Computer*
Keynote *ISCA 2009*. The 36th International Symposium on Computer Architecture (mit Daten von Kunle Olukotun, Lance Hammond, Herb Sutter, Burton Smith, Chris Batten und Krste Asanoviç)

Parallel!

Increase of Numbers of Cores



Source: W. Nagel

Co-Design!

There is something between applications & systems

(My partial) experience: application codes frequently* …

[* everyone invited to feel not addressed/meant/included!!! ☺]

- … reside, have been developed, and are maintained in groups representing the respective application domain
- … are inspired and driven by researchers standing for concrete simulation scenarios and expertise in the underlying models
- … are not putting too much emphasis on algorithmic aspects ("we scanned the literature, found that method, and implemented it")
- … prefer minimal-invasive code surgeries ("my poor baby …")
- … look for painless performance gains far away from the code:
  - "buy a bigger machine" preferred to "syntax changes"
  - "syntax changes" preferred to "semantics changes"

True also for experienced groups & leading application codes

➔ Really an application-system co-design only?

Co-Design!

CSE Challenges

From data / images / numbers …
… to information / insight

From qualitative descriptions …
… to quantitative prediction

From simulation …
… to optimisation

From parameter assumptions …
… to identification
& estimation

From forward problems …
… to inverse problems

From deterministic models …
… to random & uncertainty

From one (spatial/temporal) scale …
… to cascades of scales

From single-physics problems …
… to coupled scenarios

multi-level
➔ was a new algorithmic pattern

From counting operations …
… to energy awareness

From sequential algorithm design …
… to massive parallelism

From one-way batch jobs …
… to user interaction

From simple tools & codes …
… to 2x complex ones

From heroic PhD codes …
… to large teams / SW

From hacker's delight …
… to complex workflows

From island fun …
… to embedding & integration

From flat algorithms & data …
… to hierarchy

Co-Design!

CSE Challenges

From data / images / numbers …
… to information / insight

From qualitative descriptions …
… to quantitative prediction

From counting operations …
… to energy awareness

From simulation …
… to optimisation

From sequential algorithm design …
… to massive parallelism

From parameter assumptions …
… to identification
& estimation

From one-way batch jobs …
… to user interaction

From simple tools & codes …
… to 2x complex ones

multi-dimensional
➔ is a new algorithmic pattern

From forward problems …
… to inverse problems

From heroic PhD codes …
… to large teams / SW

From deterministic models …
… to random & uncertainty

From hacker's delight …
… to complex workflows

From one (spatial/temporal) scale …
… to cascades of scales

From island fun …
… to embedding & integration

From single-physics problems …
… to coupled scenarios

From flat algorithms & data …
… to hierarchy

Co-Design!

Therefore: Instead of HW-Appl Co-Design …

Co-design of applications and *algorithms*

PLUS

Co-design of *algorithms* and system SW / systems

… in a tightly interwoven way!

Abbreviation *"applications & systems"* is often not equivalent!

➔ *what about new/future algorithms?*

➔ *"100 hours of 80% peak vs. 10 hours of 8% peak"*

Co-Design!

First example

FHI-aims code (Max-Planck-/FHI-hosted electron structure code)

- **Bottleneck**: symmetric eigenvalue solvers (scalability)
- **Path**: BMBF-funded project ELPA
- **Team**: mathematics (BUW), informatics (TUM), HPC (RZG) + application
- **Algorithm**: 2-step tridiagonalization, block reduction; full & sparse
- **Parallel**: BG/P, Power6, hybrid MPI-OpenMP
- **Strategy**: most of the development outside the application code, but frequent transfer/integration
- **Performance**: outperforms ScaLAPACK & Co., (D)PLASMA (published)
- **Results**: one improved code, publicly available library routines, wide scope

➔ gain by algorithmic improvement / change

➔ driven by one code/application, but general algorithmic result

Co-Design!

Second example

GENE code (Max-Planck-/IPP-hosted plasma physics code)

- **Pros:** wide HPC experience of the team, large-scale scalability
- **Bottleneck**: restrictions in problem size ("Numerical ITER" far away)
- **Algorithmic diagnosis**: direct solvers, no adaptivity, dimensionality (5/6)
- **Paths**:
  - Solvers: master's thesis + G8-funded exascale project (preconditioning)
  - Adaptivity: HEPP-funded project (block adaptivity, then local adaptivity)
  - Dimensionality: TUM Institute for Advanced Study (sparse grids)
- **Team**: maths (ANU), CS (TUM, U-S), HPC (LRZ, RZG), physics (IPP)
- **Strategy**: very strict algorithm-application co-design
- **Results**: first results on solvers and dimensionalities very promising

➔ gain by algorithmic improvement / change

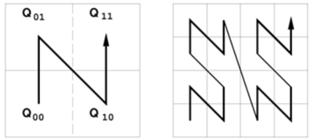
➔ driven by one code/application, but general algorithmic result

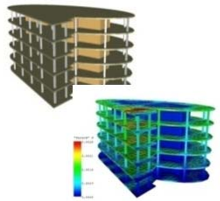
Algorithms!

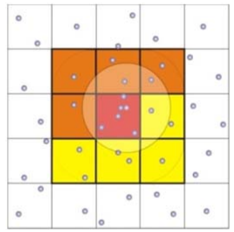
Overview of Space-Filling Curves

Lebesgue

- organizing simulation tasks

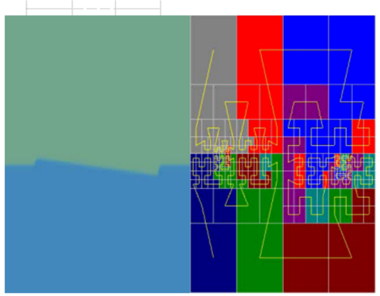


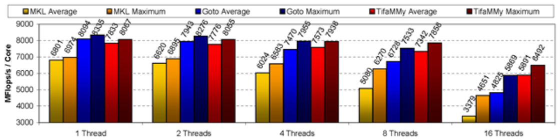




Hilbert

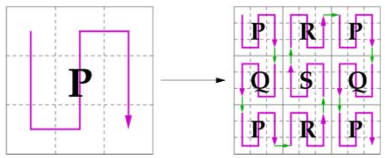
- dynamic load balancing





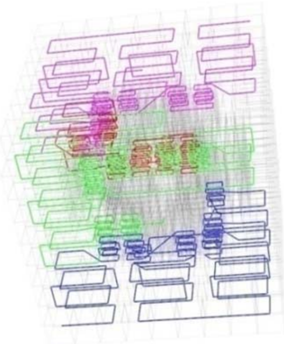
Peano

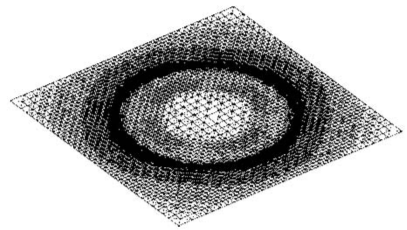
- matrix-matrix multiplication



Peano

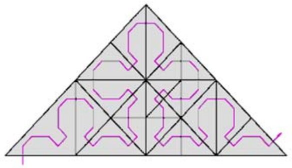
- PDE solvers

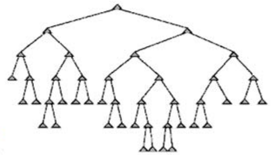




Sierpinski

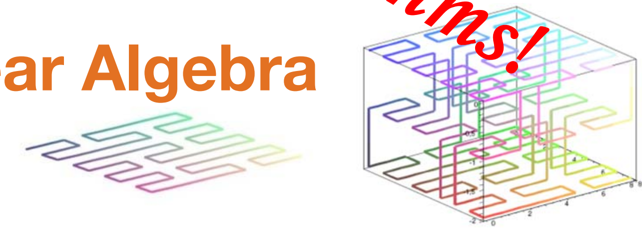
- Tsunami simulation





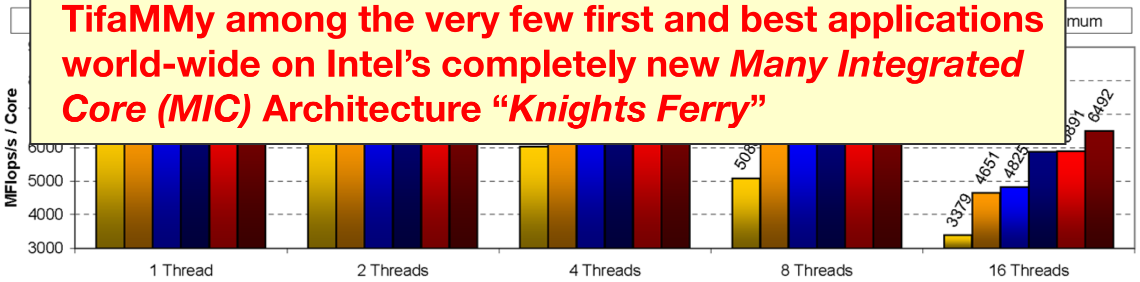
Algorithms!

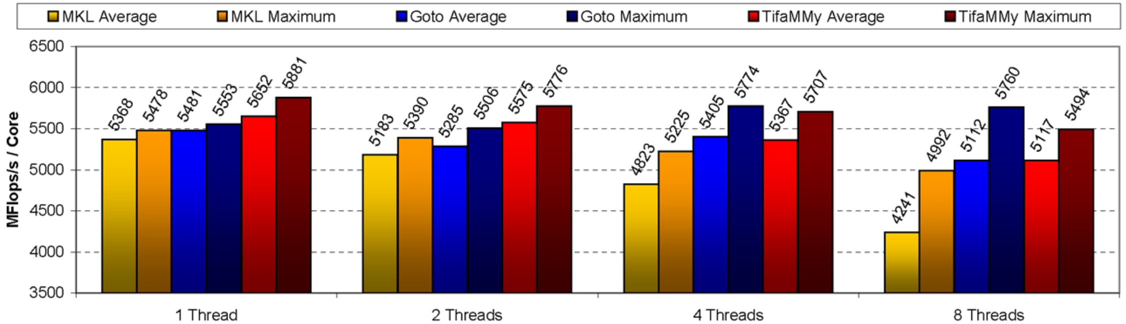
Example: Peano for Numerical Linear Algebra

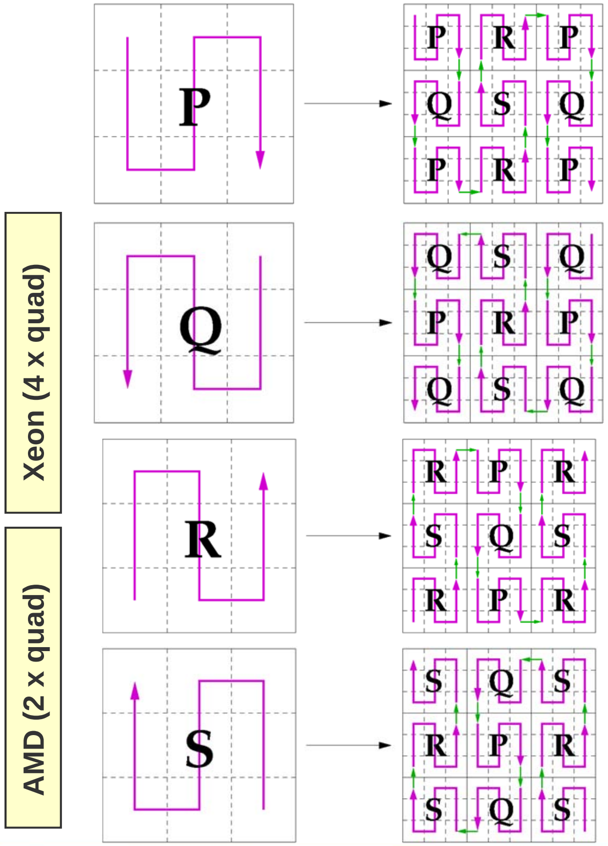


TifaMMy – cache-efficient matrix multiplication

- Peano-based traversal with high locality (dense or sparse matrices)
- block-structured data structure and algorithm
- parallel @ multicore: HW-conscious kernel, OpenMP
- parallel @ clusters: distributed caches, MPI
- application: quantum control (states via matrices)
- **ISC '11, Hamburg, June 20, 2011: Intel announces TifaMMy among the very few first and best applications world-wide on Intel's completely new *Many Integrated Core (MIC)* Architecture "*Knights Ferry*"**







Algorithms!

A Hot Topic: High-Dimensional Numerics

- **High:** not 2, not 3, not 3 plus time, but 10 … 100 ... 1000

  note: in literature, sometimes “high” *means higher than* “higher” ☺

- **Where?**
  - quantum mechanics
  - finance
  - parameter identification, optimisation (search in high-dim parameter spaces)
  - data mining, classification, information extraction

- **Why a problem?**
  - FEM: think of 11-dimensional hyper-tetrahedra and their adaptive refinement … ☺
  - Computational demand – the **curse of dimension**:
    - the cheapest 1-D discretisation … … and in 100-D?

➔ $1^{100} = 1$ ➔ cost ☺, benefit ☹

    - the second cheapest 1-D discretisation … … and in 100-D?

➔ 2^{100} approx. 10^{30} ➔ benefit ?, cost ☹

Algorithms!

Sparse Grids in a Nutshell

Starting point: finite element thinking!

How to choose grids, elements, basis functions?

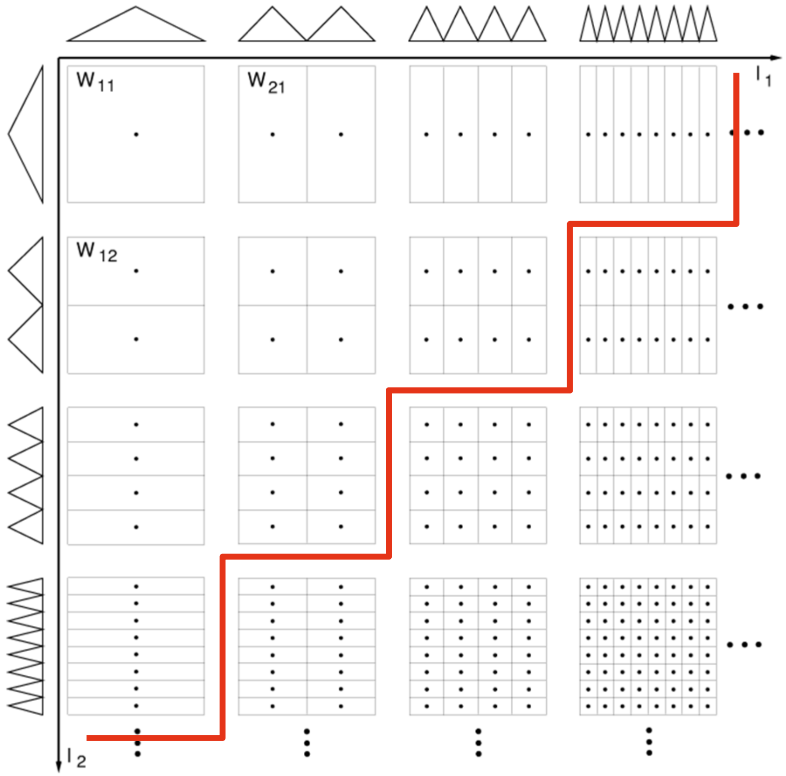
- *d=1*: **hierarchical bases** (here linear)
- *d>1*: **tensor product** approach
- **hierarchical subspaces**: collect all basis functions with support of same aspect ratio
- **regularity**: spaces *X(Ω)* of bounded mixed derivatives
- discretization / approximation as an **optimisation problem**: find optimum choice of subspaces (cf. **knapsack problem**)

$$\max_{u \in X(\Omega):\ |u|=1} \|u - u_{V(\text{opt})}\|$$

$$= \min_{U:\ |U|=N} \ \max_{u \in X(\Omega):\ |u|=1} \|u - u_U\|$$

- result: **sparse grids**

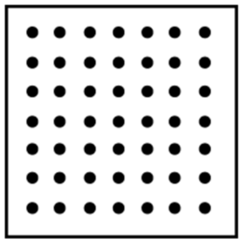
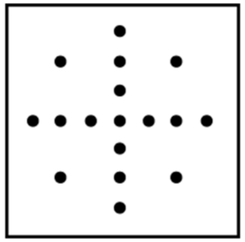
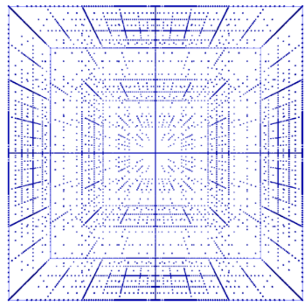
  [Zenger & co-workers, 1990]



Algorithms!

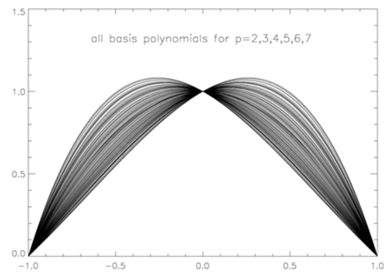
Sparse Grids in a Nutshell (cont'd)

- **Appearance**:

- **Cost** (number of grid points) vs. **benefit** (contribution to interpolant):
  (finest mesh width $h_n=2^{-n}$, hierarchical bases of piecewise degree p)

	sparse	full
# grid points	$O(h_n^{-1}\, n^{d-1})$	$O(h_n^{-d})$
error (max, L_2)	$O(h_n^{p+1}\, n^{d-1})$	$O(h_n^{p+1})$
error (energy)	$O(h_n^{p})$	$O(h_n^{p})$



- Extensions: straightforward access to **adaptive refinement**, generalisation to **piecewise polynomial** hierarchical bases, energy-optimal sparse grids

Blueprint of a Possible Informatics Contribution to CSE

New algorithmic patterns

Link those new patterns to hungry applications

Implement these (frequently "HW-nasty") patterns on current parallel systems

Try to improve performance (with its many meanings)

Contribute to the ultimate goal

"the best systems for the best algorithms for the best models"

Further Issues

Education is crucial!

- cf. Michael Hanke's talk & former minisymposia at SIAM CS&E conferences
- Specialization/new program? B.Sc./M.Sc./Ph.D.? Attached where?

Interaction with other initiatives is important!

- GACM – representing Computational Mechanics
- …?

Interact with funding agencies and advisory boards!

- German Research Foundation – DFG
- Science Council – Wissenschaftsrat: initiative to establish CSE as a field of its own, hearing tomorrow in Cologne
- BMBF, …

Thanks for your attention!