Algorithms of Scientific Computing

General Remarks

Tobias Neckel, Dirk Pflüger

Summer Term 2011

Classification of the Lecture

Students of Informatics:

- Diplom/Hauptstudium: Wahlpflichtfach theoretische Informatik
- Master Informatik: Wahlfach im Fachgebiet „Algorithmen und Wissenschaftliches Rechnen“
- Bachelor Informatik/Wirtschaftsinformatik: Wahlfach
- Students of Computational Science and Engineering (CSE)

Students of Physics: „nichtphysikalisches Wahlfach“ (Anmeldung erforderlich, 4 SWS)

Students of Mech. Engin.: Modul Numerische Simulation

Students of Mathematics and all others:

Wahlfach, Nebenfach Informatik (mit Prüfungsschriftführer abklären)

Warm Welcome!

Algorithms of Scientific Computing

Scientific Computing

also: *Computational Science and Engineering, Wissenschaftliches Rechnen*

Attempt of a definition:

Scientific Computing is . . .

- (numerical) simulation of problems stemming from engineering or natural sciences using High Performance Computing (Bungartz, TUM)
- the interdisciplinary conjunction of mathematical and computer science methods as well as different applications of the natural sciences and engineering disciplines, e.g. (TU Darmstadt)
- an *interdisciplinary discipline*
- the focus at our chair SCCS
- part of the catalogue „Algorithmen und Wissenschaftliches Rechnen“

Algorithms in Scientific Computing?

Central Question: What do I get from this lecture?

- . . . in particular in the field of Scientific Computing?
- . . . in general in the field of Informatics?

$\Rightarrow$ What could/should/do i want to learn in Informatics?

Cross sectoral topics: What central . . .

- problem
- technique, method
- analytical question

. . . of Informatics play a (major) role?

Cross Sectoral Topics

Representation of Information

Assumption:

Informatics is the art to store information in such a way that it can be used efficiently.

What and how is info stored, resp.:

- tables (data bases of all kind)
- trees, graphs (path searching, ...)
- multi-dimensional fields (raster data, etc.)

Our topic:

How do we store *continuous* data (mathematical functions)?

For Comparison: Representation of Scalars

A brief history of the representation of numbers:

- bars: $|, ||, |||, ||||$
  (still successfully used as „Bierstriche“ bars/restaurants)
- number symbols such as $\mathrm{I}, \mathrm{V}, \mathrm{X}, \mathrm{MMIV}$:
  compact but tedious for computing
- denominational number system (decimal numbers, binary system, etc.):
  ease of arithmetics up to machine computing

Crucial ideas:

- Hierarchy (different „place value“ of digits)
- Structure (concept of 0 as placeholder!)

Representation of Mathematical Functions

Possibilities of representation (historical):

- *Analytically*: $f(x) = e^x \sin(x)$
- *Value tables*
  (z.B. logarithm tables, newly rastered data/sampling)
- *Interpolation* (also piecewise):
  ("Polygonal chain", polynomial interpolation, spline interpolation, trigonometrical interpolation, . . .)

Motivation: *make information efficiently usable!*

- More compact storage
- Identification of certain properties (information)
- generally: more efficient algorithms for processing/computations

Topics of Lecture 1 – Multi-Dimensional Data

Fast Fourier Transformation: DFT, FFT, applications of DFT/FFT

Wavelets: short introduction

Hierarchical and recursive methods: Archimedes' quadrature, adaptivity

Multigrid and multi-level methods

Space-filling curves

Multi-Dimensional Data

Examples for multi-dimensional data structures:

- Matrices
- Image data (images, tomography, movies, , . . .)
- Discretization based on grids (discretization of physical models / partial differential equations)
- Coordinates of any kind (often going along with graphs)
- Tables (relational databases)
- In financial mathematics: baskets of stocks/options/. . .

Multi-Dimensional Data

Core topic: linearization/sequentialization

- Storage of data structure in memory
- Data processing (traversal)

Demands on linearization ("efficiency"):

- Maintain neighborhood $\Rightarrow$ locality of data, "clustering"
- Simple, fast computation of indices
- "Continuity", regularity
- Symmetry w.r.t. single dimensions

Topics of Lecture 2 – Algorithms

Fast Fourier Transformation: DFT, FFT, applications of DFT/FFT

Hierarchical and recursive methods: Archimedes' quadrature, adaptivity

Space-filling curves: Definition and construction of space-filling curves; algorithms and applications

Algorithms in Scientific Computing!

In the past:

- FORTRAN-programs; mainly loops

Nowadays:

- *Recursive* and *hierarchical*:
  w.r.t. algorithms (partitioning of problem) and data structures (trees, object orientation)
- *Adaptive*: invest effort, where most benefit can be achieved
- *Distributed:* Computing on parallel and distributed systems
- *Hardware-oriented:* $\rightarrow$ High Performance Computing

$\Rightarrow$ generally applicable concepts and ideas

Schedule

Fast Fourier Transformation: (Neckel)
DFT, FFT, applications of DFT/FFT

Hierarchical and recursive methods: (Pflüger)
Archimedes' quadrature, hierarchical bases, adaptivity;
outlook on wavelets;
multigrid and multi-level methods

Space-filling curves: (Neckel)
Definition and construction of space-filling curves;
algorithms and applications