Approximate Computing

1. Introduction

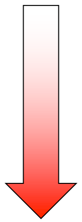
Hassan Ghasemzadeh Mohammadi and Marco Platzner
Computer Engineering Group

Chapter Overview

Motivation

- Try to quickly answer these questions:

1. Is 47.2 divided by 1.3 greater than 1?
2. Is 47.2 divided by 1.3 greater than 35?
3. What is 47.2 divided by 1.3?



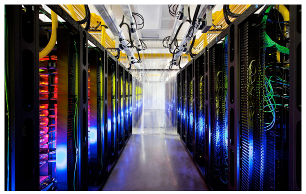
required accuracy (effort)

- Computational accuracy should be task-dependent …
  … yet, computers always (mostly) use the same accuracy

Approximate Computing (AxC)

- The central idea of AxC is to trade-off computational accuracy for a reduction in effort, i.e.,
  - energy and/or
  - execution time and/or
  - hardware area.
- These are key concerns for today's computing devices

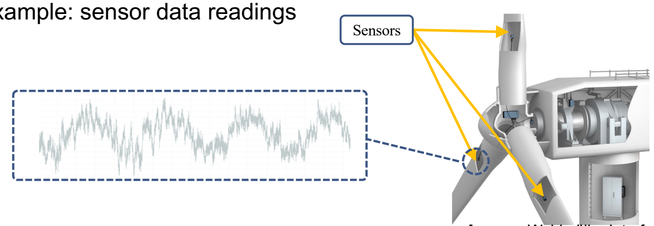
embedded / mobile systems



server farms

Suitable Application Domains

- Redundant, noisy, imprecise or incomplete input data
  - example: sensor data readings



[source: Weidmüller Interface]

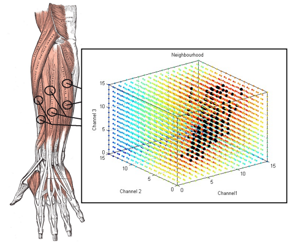
- Output perceived by humans with limited perception
  - example: JPEG lossy image compression

PSNR = 31.16 dB

PSNR = 28.90 dB

Suitable Application Domains

- Time/energy constraints prohibit the computation of an exact or optimal result
  - examples: big data, machine learning

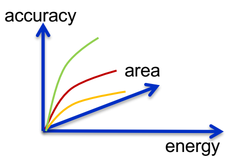


- No unique or golden answer exists
  - examples: search machines, recommender systems

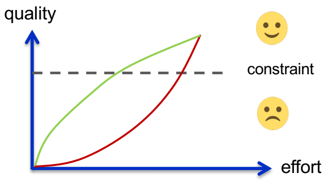
Google Booking.com yahoo! search NETFLIX

Quality-Effort Trade-Off

- 'Quality' as a new characteristic of an implementation
  - quality metric is application-dependent
    - examples: PSNR for image compression, classification accuracy for neural network, worst-case error for an arithmetic computation, user rating for a search engine …
  - quality can be a constraint or an optimization objective
- New challenges for design and optimization
  - exploit novel trade-offs between quality and effort
  - define, measure and guarantee quality
  - adjust trade-off at design time or even at runtime



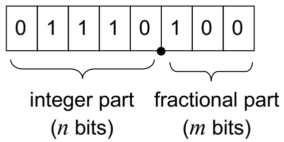
[source: A. Raghunathan]



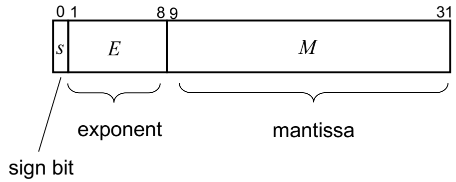
Is AxC Really New? – Related Forms of AxC

- Finite machine precision
  - floating point data types always have limited precision
  - a floating point variable is an approximation of a real number
  - extensive work exists on studying quantization and rounding effects, error analysis, error propagation in longer computations, ...

Example: unsigned fixed-point number with precision (accuracy) of 2^{-m}



Example: floating-point number in single precision IEEE 754 standard format, precision is not constant over value range



Is AxC Something New? – Related Forms of AxC

- Heuristics and approximation algorithms
  - a heuristic is an efficient algorithm for a problem without a guarantee that the solution is accurate / exact / optimal
  - an approximation algorithm is an efficient algorithm with a proven distance between returned and optimal solution (approximation factor)
- Anytime (iterative) algorithms
  - an anytime algorithm can return a valid solution to a problem even if it is interrupted before it ends
  - typically realized by iterative improvements of an initial valid solution
  - if running longer, the quality of the output increases
- Heuristics, approximation algorithms, and anytime algorithms compute approximated solutions

Is AxC Something New? – Related Forms of AxC

- Unconventional computing paradigms that approximate solutions
  - Stochastic Computing
    - represent numbers by streams of random bits: with n as number of bits in the stream, and m as the number of '1's, the value is m/n

0	0	1	1	0	1	0	0

3/8 = 0.375

    - requires little hardware for many operations and is tolerant against bit flips, but also slow and requires bit streams to be uncorrelated
  - (Electronic) Analog Computing
    - computers built from electronic components, such as diodes, resistors, operational amplifiers, etc.
    - can excel in speed and energy-efficiency, but suffer from low reliability, limited precision, errors, and cumbersome programming
  - Probabilistic Computing
    - analog computing where signals express probabilities
    - basically, same pros and cons as analog computing

The (Re-)Emergence of AxC

- Current focus: Aggressively apply approximations on and across all levels of the computer design hierarchy, including hardware

Design hierarchy	Approaches for AxC (selection)
Application/Algorithm	Approximate DNNs, Approximate SVM, Neural algorithmic transformation, Precision scaling
Language/Compiler	Loop perforation for global patterns, HDL extensions to support AxC
Microarchitecture/RTL	AxC-based accelerators, AxC-based circuit synthesis, partial memory access elimination
Components/Logic	Logic optimization and pruning for AxC, Approximate arithmetic components (e.g., adder, multiplier, and divider), AxC FPUs
Technology	Dynamic voltage over-scaling, Memory refresh period enlargement

The (Re-)Emergence of AxC

- DARPA/ISAT Workshop 2012 on "Advancing computer systems without technology progress"
  - identified AxC as one approach for future performance gains

  [C. Kozyrakis: "Advancing computer systems without technology progress", IEEE International Symposium on Performance Analysis of Systems and Software, 2013]
- AxC workshop series
  - Workshop on Probabilistic and Approximate Computing (APPROX)
  - Workshop on Approximate Computing (WAPCO)
  - Workshop on Approximate Computing Across the Stack (WACAS)
  - Workshop on Approximate Computing (AC)
    - AC 2015 in Paderborn

Gesellschaft für Informatik

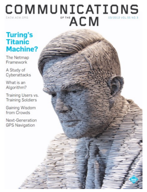
- AxC sessions in several other conferences

The (Re-)Emergence of AxC

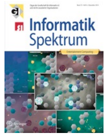
- Increasing number of publications, also in popular magazines

Sampson et al.: "EnerJ, the Language of Good-Enough Computing", *IEEE Spectrum*, Oct. 2013

R. Nair: "Big Data Needs Approximate Computing". *Comm. ACM*, Jan 2015



L. Kugler: "Is 'good enough' computing good enough?" *Comm. ACM*, May 2015



C. Plessl, M. Platzner, P. Schreier: "Aktuelles Schlagwort: Approximate Computing", *Informatik Spektrum*, 2015

Special Issue on Approximate Computing in *IEEE Design & Test*, 2016

Dependability vs. Approximation

- Semiconductor technology progresses into low nm domain
  - fault-free operation difficult to achieve due to increasing densities and shrinking supply voltages
  - dependability of chips becomes a huge concern
    - guardbanding leads to diminishing returns
    - redundancy techniques show excessive overheads in hardware and/or software
  - currently unclear how long technology scaling will continue and when (and which) post-CMOS technologies will be ready
- AxC does not combat 'faults', but intentionally insert them
  - yet, same techniques could be useful for both approaches

1.2 For Whom is this Course

- Goals
  - research-oriented advanced Master-level course
  - introduce to the emerging field of Approximate Computing
  - provide an overview over approximation approaches at different levels
  - serve as a starting point for research activities
- Addressed study programs
  - Computer Science (CS) master students
    - elective module in focus area "Computer Systems"
  - Computer Engineering (CE) master students
    - elective module in focus areas "Computer Systems" and "Embedded Systems"
- Prerequisites
  - no formal prerequisites with respect to other Master-level courses
  - HOWEVER: solid background in micro/nanoelectronics, digital design, computer architecture, and algorithms/applications is extremely helpful

1.3 Course Content & Organization

- Lecture – Topics (tentatively)
  1. Introduction
  2. Application / algorithm level
  3. Language / compiler level
  4. Microarchitecture / register-transfer level
  5. Components / logic level
  6. Technology level

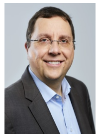
Design hierarchy

- Application/Algorithm
- Language/Compiler
- Microarchitecture/RTL
- Components/Logic
- Technology

Course Content & Organization

- Lecturers

Hassan Ghasemzadeh Mohammadi, O3.134, ☎ 60 4344, hgm@mail.upb.de



Marco Platzner, O3.207, ☎ 60 5250, platzner@upb.de

- Lecture sessions
  - Thursday 11:15 - 13:45

- Course uses the “inverted classroom model”
  - elements of lecture and post-processing are swapped (inverted)
  - learning activities that students can do well on their own are shifted to a preparation phase
  - the common attendance time is used for an active discussion

“Inverted Classroom” Model

- Advantage
  - instead of frontal teaching, a time window is created for joint discussion and deepening of understanding
  - the lecture material can be studied at any time, as often as desired, and from anywhere

- Implementation
  - The lecturers make material available in PANDA (slides, screencast+audio)
  - You prepare independently for the classroom sessions
  - We use the common attendance times for
    - clarification of specific questions, discussions
    - examples, possibly small exercises, quizzes
    - reflection of the learning process

Course Content & Organization

- Lab sessions
  - two modules (successful participation of a module earns a bonus)
    - AxC for machine learning (ML)
      - AxC at the level of application/algorithm
      - get familiar with widely used ML libraries such as Scikit-learn and Tensor-flow
      - apply approximation on different groups of ML algorithms, e.g., clustering and classification
      - evaluate performance and quality of ML models using error analysis
    - AxC for digital signal processing (DSP) circuits
      - AxC at the levels of microarchitecture/RTL and components/logic
      - get familiar with a circuit AxC synthesis process
      - work with approximate component libraries
      - study quality evaluation of approximate DSP circuits
      - learn to work with commercial synthesis tools, e.g., Synopsys Design Compiler
  - start of lab sessions will be announced

Course Content & Organization

- Course materials
  - all information is made available on PANDA:
    https://panda.uni-paderborn.de/course/view.php?id=27940

- Grading
  - successful participation in the lab improves grade by 1 or 2 grade steps (if exam is passed)
  - oral exam (~45’) covering lecture + lab

Positional and Survey Papers (Selection)

- J. Han and M. Orhansky. Approximate Computing: An Emerging Paradigm for Energy-Efficient Design. IEEE European Test Symposium, 2013.
- A. Sampson, L. Ceze and D. Grossman. Enerj, the Language of Good-Enough Computing. IEEE Spectrum, Oct 2013.
- R. Nair. Big Data Needs Approximate Computing. Communications of the ACM, Dec 2014.
- L. Kugler. Is "Good Enough" Computing Good Enough? Communications of the ACM, Apr 2015.
- S. Mittal. A Survey of Techniques for Approximate Computing. ACM Computing Surveys, Nr. 62, 2016.
- S. Davidson. Good Enough Computing. IEEE Design & Test, 33(1), 2016.
- Q. Xu, T. Mytkowicz, and N.S. Kim. Approximate Computing: A Survey. IEEE Design & Test, 33(1), 2016.
- G. Rodrigues, F.L. Kastensmidt, and A. Bosio. Survey on Approximate Computing and Its Intrinsic Fault Tolerance. Electronics, 9(4):557, 2020.