Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Laura Wenderoth

Reproduzierbarkeit Bit-genauer Anwendungen

Proseminar 2020 – Software in Entwicklung

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Designing Bit-Reproducible Portable High-Performance Applications

A. Arteaga et al, 2014 IEEE 28th International Parallel and Distributed Processing Symposium

Übersicht

- ❑ Definition: High performance computing
- ❑ Reproduzierbarkeit
- ❑ Problemdarstellung von Reproduzierbarkeit bei HPC
- ❑ Problemlösung durch reproduzierbare Reduktion
- ❑ Ausblick der Anwendung und Forschung
- ❑ Zusammenfassung

Definition: High performance computing (HPC)

- Deutsch: Hochleistungsrechen
- Nutzung verteilter Rechenressourcen zur Lösung komplexer Probleme mit großen Datenmengen
- „Rechenanwendungen, deren Komplexität oder Umfang eine Berechnung auf einfachen Arbeitsplatzrechnern unmöglich oder zumindest unsinnig macht“ [1]
- Möglich durch **parallele Verarbeitung**
- Verbreitung: Wissenschaft und Forschung

Definition: Reproduzierbarkeit Bit-genauer Anwendungen

- “A computation is reproducible if it achieves bitwise identical results from the same computation when given equivalent inputs.” [2]
- Das Erhalten von bitweise identischen Gleitkomma-Ergebnissen auch bei mehreren Durchläufen desselben Programms bei gleicher Eingabe

Reproduzierbarkeit Bit-genauer Anwendungen

- Nutzen & Ziele
  - Debugging
  - Simulation des Verhaltens komplexer und stark nichtlinearer Systeme (wissenschaftliche Anwendungen)
    - Molekulargenetische Forschung
    - Wetter
    - Klimasimulation
- Wichtige Faktoren
  - Entwurf des Algorithmus
  - Implementierung
  - Programmiersprache
  - Compilierung
  - Laufzeitumgebung
  - Hardware

Exkurs: IEEE-754 Standard und floating-point values

- Definition von Standarddarstellungen für binäre Gleichkommazahlen
- Festlegung für genaue Verfahren für die Durchführung mathematischer Operationen, besonders Rundungen
- verschiedene Grundformate:
  - 16 Bit (minifloat)
  - 32 Bit (single precision)
  - 64 Bit (double precision)

IEEE-754 Standard und floating-point values

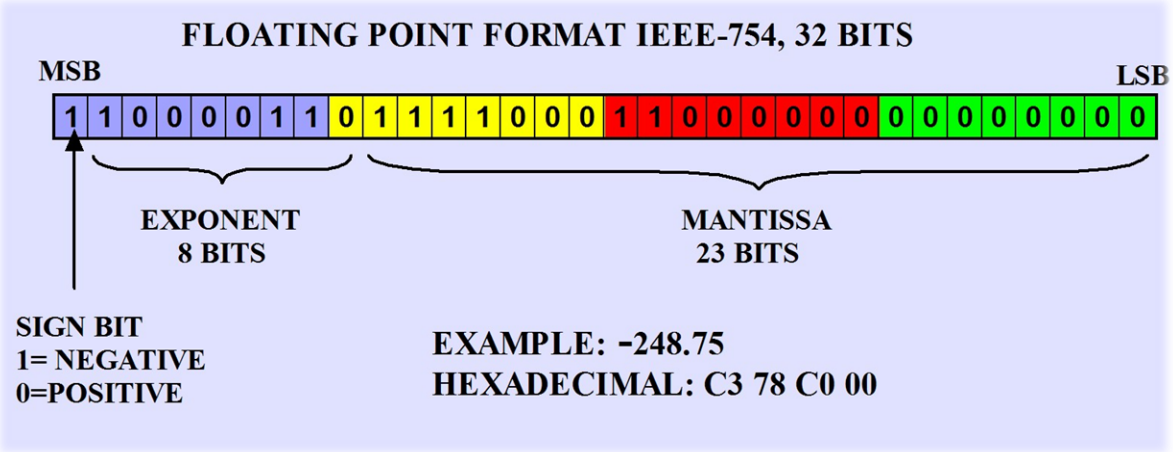
- Darstellung einer Gleitkommazahl

$$x = s \cdot m \cdot b^e$$

  - Vorzeichen s: 1 Bit (0 = positiv; 1 = negativ)
  - Mantisse m (p Bits): erstes Bit ist immer eine 1 und wird deswegen weggelassen
  - Basis b (beim IEEE 754 Standard immer b =2)
  - Exponent e (r Bits)
  - Darstellung undefinierter Werte: NaN: not a number

IEEE-754 Standard und floating-point values

- Darstellung einer Gleitkommazahl



http://blog.ruofeidu.com/exploration-ieee-754-floating-point-standard/

Problemdarstellung von Reproduzierbarkeit

Problemdarstellung von Reproduzierbarkeit

- Beispiel: $2^{57} + 1 - 2^{57}$
  - normalerweise Assoziativität gegeben
  - $2^{57} + 1 = 2^{57}$ (IEEE-754 Standard)

$$2^{57} + 1 - 2^{57} = 0$$

$$2^{57} - 2^{57} + 1 = 1$$

Lösungsansätze für Reproduzierbarkeit von floating point values

Lösungsansätze

- ❑ Konsequente, vollständige Definition
- ❑ Vermeiden von Rundungen
- ❑ Lookup tables (Bibliothek)
- ❑ Genaue Dokumentation von mathematischen Funktionen der verschiedenen Programmiersprachen

- ❑ NVIDIA (CUDA)

CUDA (Compute Unified Device Architecture)

"CUDA is a parallel computing platform and programming model designed to deliver the most flexibility and performance for GPU-accelerated applications.

To maximize performance and flexibility, get the most out of the GPU hardware by coding directly in CUDA C/C++ or CUDA Fortran." [6]

CUDA (Compute Unified Device Architecture)

- Co-Prozessor: GPU
- Programmiersprache: C for CUDA (mit Erweiterungen von Nvidia)
- Anwendungsbeispiele:

Problemdarstellung von Reproduzierbarkeit bei high performance computing

Problemdarstellung von Reproduzierbarkeit bei high performance computing

- ❑ Normalerweise: Berechnungen sind assoziativ und kommutativ
- ❑ Floating-point Arithmetik stellt keine Assoziativität und Kommutativität sicher.

→ Fixe Klammerung

→ Geringe Effizienz

Reproduzierbare Reduktion arithmetischer Funktionen

Definition: Reduktion

„Eine Reduktion des Problems P1 auf das Problem P2 ist ein Algorithmus, der das Problem P1 unter der Voraussetzung effizient löst, dass eine Black Box mit konstanten Kosten zur Lösung von P2 benutzt werden darf. […]Bezogen auf die betrachtete Komplexitätsklasse muss ein passender Reduktionstyp gewählt werden, z. B. polynomielle Zeitreduktion, Turing-Reduktion“ [7]

Deterministische Summenreduktion

Double-Sweep-Algorithmus

Deterministische Summenreduktion

- Prinzip: Vorheriges Runden der einzelnen Zahlen ermöglicht weiteres Rechnen ohne weiteres Runden. [nach Rump, Demmel and Nguyen]

- Extraktor M: größer als jede Partialsumme von allen Summanden
- Genauigkeit ϵ
- n Summanden v_i

$$M \geq n \cdot |v_i| \,/\, (1 - 2n\epsilon) \quad \forall\ 1 \leq i \leq n.$$

Deterministische Summenreduktion

- Berechnung der vorgerundeten Werte (constributions)

$$q_i := (v_i \oplus M) \ominus M$$

- Berechnung der Summe

$$T := \sum_{i=1}^{n} q_i$$

Deterministische Summenreduktion – Beispiel

$$M = 16 = 1 \cdot 2^4$$

$$v_1 = fl(3{,}16) = 1{,}1001010010 \cdot 2^1$$

$$v_2 = fl(1{,}73) = 1{,}1011101011 \cdot 2^0$$

$$3{,}16 + 1{,}73 = 4{,}89$$

$$4{,}89_{10} = 100.111000\ldots_2$$

Deterministische Summenreduktion – Beispiel

$$M = 16 = 1 \cdot 2^4$$

$v_1 =$ 

$v_2 =$ 

$$3{,}16 + 1{,}73 = 4{,}89$$

$$4{,}89_{10} = 100.111000\ldots_2$$

Deterministische Summenreduktion – Beispiel

$M = 16 = 1 \cdot 2^4$

$v_1 = fl(3{,}16) = 1{,}1001010010 \cdot 2^1$ | $v_1 + M = 1{,}0011001010 \cdot 2^4$

$q_1 = (v_1 + M) - M = 1{,}1001010000 \cdot 2^1$

$v_2 = fl(1{,}73) = 1{,}1011101011 \cdot 2^0$ | $v_2 + M = 1{,}0001101110 \cdot 2^4$

$q_2 = (v_1 + M) - M = 1{,}1011100000 \cdot 2^0$

Deterministische Summenreduktion – Beispiel

$M = 16 = 1 \cdot 2^4$

$v_1 =$ 

$| v_1 + M$ 

$q_1 = (v_1 + M) - M = \ 1{,}1001010000 \cdot 2^1$

$v_2 =$ 

$| v_2 + M$ 

$q_2 = (v_1 + M) - M = 1{,}1011100000 \cdot 2^0$

Deterministische Summenreduktion – Beispiel

$M = 16 = 1 \cdot 2^4$

$v_1 = fl(3{,}16) = 1{,}1001010010 \cdot 2^1$

$v_2 = fl(1{,}73) = 1{,}1011101011 \cdot 2^0$

$q_1 = 1{,}1001010000 \cdot 2^1$

$q_2 = 1{,}1011100000 \cdot 2^0$

$$\begin{array}{r} 11,001010000 \\ +\quad 1,101110000 \\ \hline 100,111000000 \end{array}$$

$q_1 + q_2 = 1{,}001110000 \cdot 2^2$

$3{,}16 + 1{,}73 = 4{,}89$

$4{,}89_{10} = 100.111000\ldots_2$

Deterministische Summenreduktion

- Ausgleich des Informationsverlustes mit fehlerfreier Subtraktion

$$r_i = v_i - q_i$$

- Wiederholung des Algorithmus mit r_i

- first-level extraction: Ergebnis der Rechnung mit den originalen floiting point values
- second-level extraction: Ergebnis der Rechnung mit den Resten
- …

Zusammenfassung - Double-Sweep-Algorithmus

- ❑ Zahlen werden mit einem geeignet großen Extraktor vorgerundet
- ❑ Vorgerundete Zahlen können in beliebiger Reihenfolge summiert werden
- ❑ Summe semantisch äquivalent zu fehlerfreier Integer-Summe
- ❑ Verlust von Information - Ausgleich durch beliebig häufiges Wiederholen
- ❑ Zwei Kommunikationsschritte → schlecht für Performance

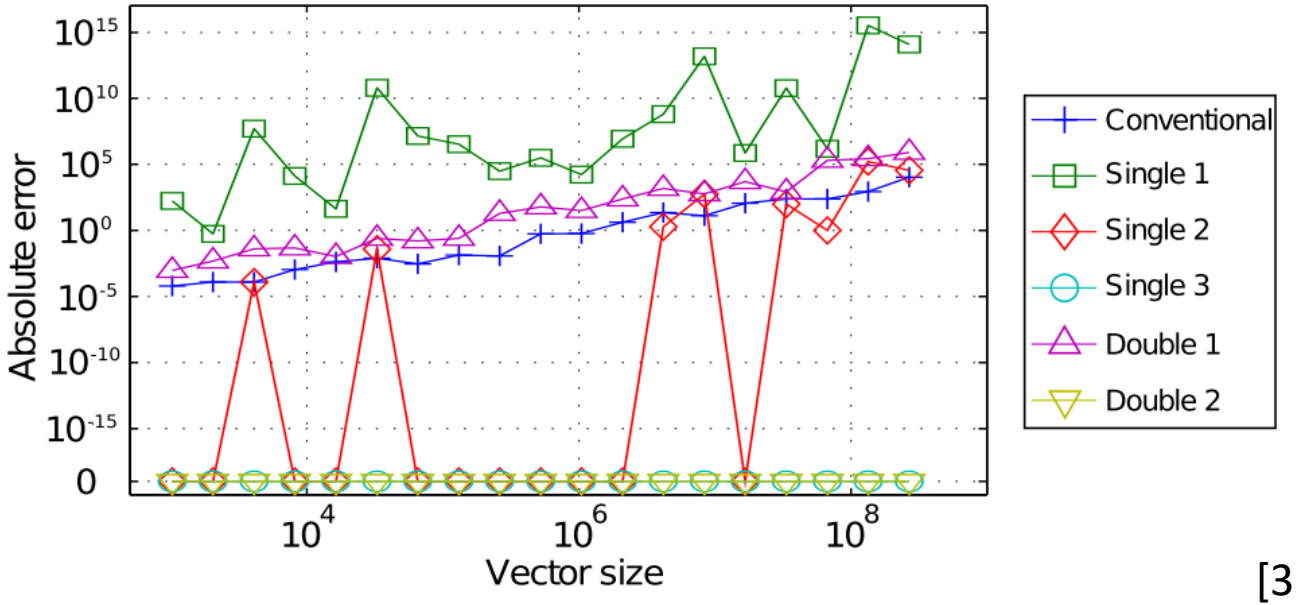
Deterministische Summenreduktion

Single-Sweep-Algorithmus

Deterministische Summenreduktion

- Idee: nur noch ein Kommunikationsschritt zur Performanceverbesserung
- Prinzip: Es wird kein gemeinsamer Extraktor am Anfang gebildet, sondern mit dem größtmöglichen Extraktor begonnen. Dadurch kommt es zu einem größerem Informationsverlust [aufgrund des Formates ist die maximale Größe des Extraktor bestimmt]

Deterministische Summenreduktion - Vergleich



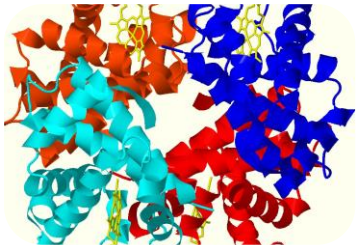
[3]

Verwendung bitgenauer Reproduzierbarkeit bei nicht-linearer Anwendungen

Astrophysik

Bioinformatik

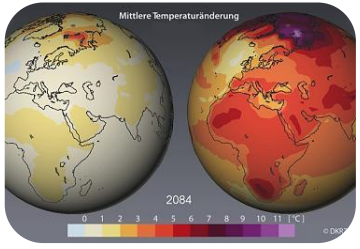
Künstliche Intelligenz



Molekulardynamik

Big Data – Analysen

Finanzwesen



Wetter- und Klimaberechnungen

Cyber Security

Zusammenfassung

Zusammenfassung

- Verwendung: Reproduzierbare Reduktion von Berechnungen in nicht-linearen Systemen zur beispielsweise Vorhersage von Klima/ Wetter
- Verschiedene Ansätze auf verschiedenen Ebenen von Hardware bis Algorithmus
- Interessante Ansätze auf Ebene der Software (Deterministische Summenreduktion)
- Forschung auf dem Feld ist aktuell und vielfältig
- Weitere Paper bei Interesse

Quellen

[1] https://de.wikipedia.org/wiki/Hochleistungsrechnen [letzter Zugriff 21.06.20]

[2] Peter Ahrens, Hong Diep Nguyen, James W. Demmel: Efficient Reproducible Floating Point Summation and BLAS; 2015

[3] A. Arteaga et al “Designing Bit-Reproducible Portable High-Performance Applications”; IEEE 28th International Parallesl and Distributed Processing Symposium, 2014

[4]S. M. Rump, “Ultimately fast accurate summation,” SIAM Journal on Scientific Comp., vol. 31, no. 5, p. 3466, 2009.

[5]J. Demmel and H. D. Nguyen, “Fast Reproducible FloatingPoint Summation,” in 21st IEEE Symp. on Computer Arithmetic, ser. ARITH’13, 2013, pp. 163–172.

[6] https://developer.nvidia.com/cuda-zone [letzter Zugriff 29.06.20]

[7] https://www.spektrum.de/lexikon/mathematik/reduktion/8419 [letzter Zugriff 29.06.20]

https://live.staticflickr.com/4042/34923045104_7388e6d074_b.jpg [letzter Zugriff 23.06.20]

https://en.wikipedia.org/wiki/CUDA [letzter Zugriff 23.06.20]

https://developer.nvidia.com/hpc [letzter Zugriff 23.06.20]

Bildquellen

https://pixabay.com/de/photos/wissenschaft-science-channel-5350595/
https://www.grenzwissenschaft-aktuell.de/genetiker-wollen-kuenstliches-menschliches-genom-erstellen20160617/
https://pixabay.com/de/photos/dna-dns-biologie-erbgut-1388696/
https://pixabay.com/de/photos/künstliche-intelligenz-roboter-ai-2167835/
http://www.chemgapedia.de/vsengine/vlu/vsc/de/ch/8/bc/vlu/proteindynamik/haemoglobin.vlu/Page/vsc/de/ch/8/bc/proteindynamik/haemoglobin.vscml.html
https://pixabay.com/de/photos/daten-tastatur-maus-big-data-4151152/
https://pixabay.com/de/photos/unternehmer-idee-kompetenz-vision-1340649/
https://www.focus.de/wissen/klima/klimaerwaermung/neueste-prognosen-zum-klimawandel-erde-erwaermt-sich-nicht-so-schnell-wie-befuerchtet_aid_994582.html
https://pixabay.com/de/photos/internet-sicherheit-online-computer-2296269/

Interessante Paper

Peter Ahrens, Hong Diep Nguyen, James W. Demmel „Efficient Reproducible Floating Point Summation and BLAS“; 2015

D. Goldberg, “What every computer scientist should know about floating-point arithmetic,” ACM Computing Surveys (CSUR), vol. 23, no. 1, pp. 5–48, 1991.

A. Arteaga et al “Designing Bit-Reproducible Portable High-Performance Applications”; IEEE 28th International Parallesl and Distributed Processing Symposium, 2014