Optimierung von Programmen

Einführung

Christian Pape

Hochschule Karlsruhe

Wintersemester 2017/18

Inhalt

Begriff Optimierung

Optimierung nach Merriam-Webster-Wörterbuch:

> *an act, process, or methodology of making something (such as a design, system, or decision) as fully perfect, functional, or effective as possible;*

Begriff Optimierung

Optimierung nach Merriam-Webster-Wörterbuch:

> *an act, process, or methodology of making something (such as a design, system, or decision) as fully perfect, functional, or effective as possible;*

Nach Cambridge-Dictionary:

> *the act of making something as good as possible*

Begriff Optimierung

Optimierung nach Merriam-Webster-Wörterbuch:

> *an act, process, or methodology of making something (such as a design, system, or decision) as fully perfect, functional, or effective as possible;*

Nach Cambridge-Dictionary:

> *the act of making something as good as possible*

Etwas perfekt oder so gut wie möglich zu machen ist

- ziemlich schwierig und
- in der Praxis oft unerreichbar.

Begriff Optimierung

Optimierung nach Merriam-Webster-Wörterbuch:

> *an act, process, or methodology of making something (such as a design, system, or decision) as fully perfect, functional, or effective as possible;*

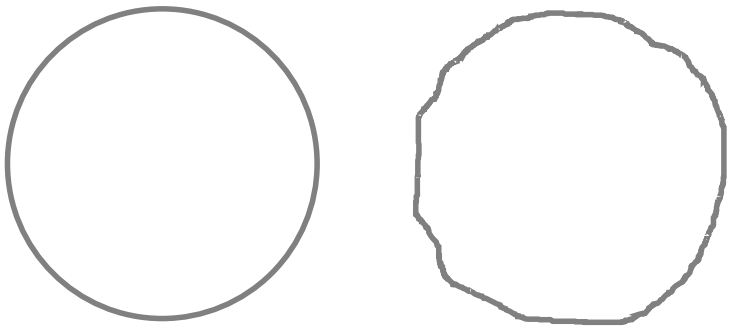
Nach Cambridge-Dictionary:

> *the act of making something as good as possible*

Etwas perfekt oder so gut wie möglich zu machen ist

- ziemlich schwierig und
- in der Praxis oft unerreichbar.

Beispiel: Nahezu perfekten Kreis frei zeichnen.



Begriffe

Optimierung von Programmen *oder Software bedeutet Systemresourcen wie* Zeit *und* Speicher *so wenig wie möglich zu verbrauchen.*

- In der Regel kein perfekter Resourcenverbrauch möglich.

Begriffe

Optimierung von Programmen *oder Software bedeutet Systemresourcen wie* Zeit *und* Speicher *so wenig wie möglich zu verbrauchen.*

- In der Regel kein perfekter Resourcenverbrauch möglich.
- Nur Teile des Programms können realistischerweise optimiert werden.
- Unvollständiges Wissen über komplexe Ablaufumgebung verhindert Optimierung in der Regel.
- **Verbesserung von Programmen** wäre ein treffenderer Begriff.

Begriffe

Optimierung von Programmen *oder Software bedeutet Systemresourcen wie* Zeit *und* Speicher *so wenig wie möglich zu verbrauchen.*

- In der Regel kein perfekter Resourcenverbrauch möglich.
- Nur Teile des Programms können realistischerweise optimiert werden.
- Unvollständiges Wissen über komplexe Ablaufumgebung verhindert Optimierung in der Regel.
- **Verbesserung von Programmen** wäre ein treffenderer Begriff.
- **Softwareoptimierung**: synonym für Programmoptimierung.

Warum optimieren?

Mooresche Gesetz.

- Hardware wird doch immer leistungsfähiger.
- Bis die Software realisiert, ist die Hardware vielfach schneller geworden.
- Ungelöste Probleme werden lösbar: Gesichtserkennung, Go.

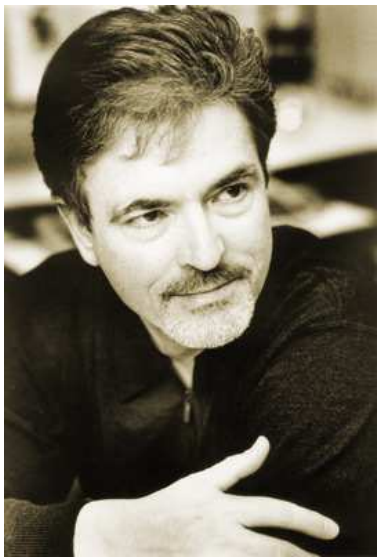
Gordon E. Moore [Moore, 1965]:

> *The complexity for minimum component costs has increased at a rate of roughly a factor of two per year*

Von links: Robert Noyce, Andy Grove und Gordon Moore, 1978 (Intel Free Press)

Warum optimieren?

- Software wird durch unwesentliche Features in kürzeren Produktzyklen immer komplexer und langsamer.
- Auch bekannt als: Wirth's Law, Page's law, Gates' law, May's law ...



Martin Reiser (nach Niklaus Wirth) [Wirth, 1995]:

> *Software is getting slower more rapidly than hardware becomes faster.*

Foto von `http://r-e-i-s-e-r.de`.

Warum optimieren?

Im Ingenieurbereich ist Effizienzsteigerung fast immer erstrebenswert:

- Wirkungsgrad von Kraftwerken erhöhen,
- Kraftstoffverbrauch von motorbetriebenen Fahrzeugen senken oder
- Produkte bei gleicher Belastung und Gebrauch haltbarer machen.
- Mikroprozessoren: Bei gleicher Schaltungsdichte und Taktfrequenz Verarbeitungsgeschwindigkeit erhöhen.

Wenn man nicht optimiert, dann tut es die Konkurrenz!

Warum optimieren?

Programmoptimierungen ist überall dort sinnvoll, wo

- eine Effizienzsteigerung durch Hardware nicht mehr möglich ist,
- die technische oder
- finanzielle Grenze der Optimierung erreicht ist.

Beispiele aus der Informatik:

- Videospiele: Bildwiederholfrequenz zu gering für ein flüssigen Spielablauf.
- Maschinelles Lernen (z.B. Künstliche Neuronale Netzen): Bisher ungelöste Probleme nur lösbar mit optimierte Software.
- Physikalische Simulationen (z.B. Klimamodelle): Rechenzeit kann auch auf Supercomputern teilweise Monate dauern.

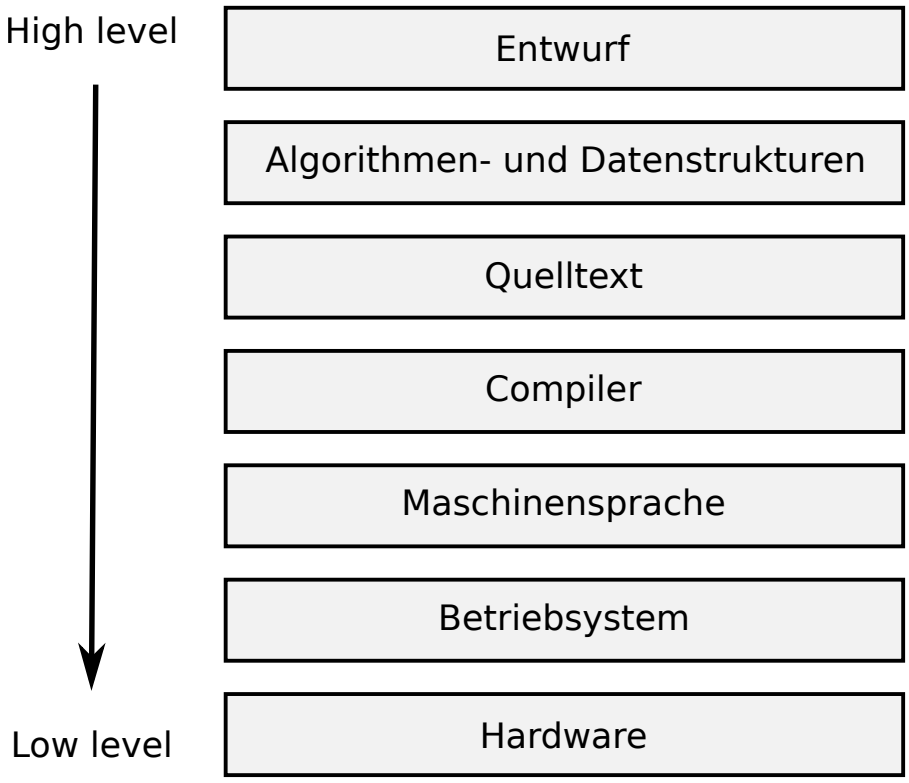
Warum optimieren?

Simulationen: Earth Simulator

- Betriebsstart: März 2002
- Hardware-Erneuerungen: 2009, 2015
- Derzeit: 5120 NEC SX-ACE (64 in einem Rack)
- Insgesamt: 1,3 PF; 320 TB Hauptspeicher

Infos aus Broschüre [Ear, 2017], Foto aus Earth Simulator Photo Gallery.

Ebenen der Optimierung



- keine klare Trennung zwischen benachbarten Ebenen
- Auswirkungen nehmen von oben nach unten ab
- Fehlentscheidungen auf oberer Ebene schwerer zu korrigieren als auf unterer
- von oben nach unten beginnen zu optimieren

Ebenen der Optimierung

Entwurf

Entwurf (design) [IEEE, 1984, Seite 26]:

> *process of defining the architecture, components, interfaces, and other characteristics of a system or component*

- Auch die Entwurfsergebnisse werden als Entwurf bezeichnet.
- **Softwarearchitektur** ist Entwurf.
- Entwurfsergebnisse sollen verbessert werden.

Ebenen der Optimierung

Entwurf

Entwurf (design) [IEEE, 1984, Seite 26]:

> *process of defining the architecture, components, interfaces, and other characteristics of a system or component*

- Auch die Entwurfsergebnisse werden als Entwurf bezeichnet.
- **Softwarearchitektur** ist Entwurf.
- Entwurfsergebnisse sollen verbessert werden.

Beispiele:

- Software-Architektur, die die Anzahl Netzwerkzugriffe verringert.

Ebenen der Optimierung

Entwurf

Entwurf (design) [IEEE, 1984, Seite 26]:

> *process of defining the architecture, components, interfaces, and other characteristics of a system or component*

- Auch die Entwurfsergebnisse werden als Entwurf bezeichnet.
- **Softwarearchitektur** ist Entwurf.
- Entwurfsergebnisse sollen verbessert werden.

Beispiele:

- Software-Architektur, die die Anzahl Netzwerkzugriffe verringert.
- Schnittstelle mit geringerer Anzahl Unterprogrammaufrufe.

Ebenen der Optimierung

Algorithmen und Datenstrukturen

Algorithmen und Datenstrukturen sind kleinere Teile des Schnittstellen-Entwurfs.

- Resourcenverbrauch mit der Gross-O-Notation hinsichtlich einer definierten Eingabekomplexität n abschätzbar.
- Algorithmus mit z.B. $\Theta(n^2)$ Zeitaufwand durch Algorithmus mit $\Theta(n \log n)$ Zeitaufwand ersetzen.
- Algorithmus bei gleichem Zeitaufwand ersetzen durch einen mit geringeren konstantem Faktor.
- Verzicht auf Rekursion.
- Vermeidung von Sonderfälle.
- Hybridisierung von Algorithmen.

Ebenen der Optimierung

Quelltexte

Quelltexte beinhalten die Programmanweisungen in einer **Hochsprache**.

- Programmiersprache beeinflußt Resourcenverbrauch.
- Mehr Optimierungsspielraum bei systemnahen Programmiersprachen
- Moderne Programmiersprachen: Automatische Parallelisierung oder Speicherverwaltung beschleunigt Programme.
- Teile der Software zur Verbesserung in anderer Programmiersprache implementieren.
- Langsame Programmbefehle durch äquivalente schnellere Datentypen oder Befehle ersetzen. `i >> 1` bei positiven ganzzahligen Werten statt `i / 2` in C.

Ebenen der Optimierung

Compiler

Ein **Compiler** übersetzt den Quelltext in ein ausführbares Programm: Maschinensprache oder Zwischencode.
Ein **Cross-Compiler** kann ausführbare Programm für unterschiedliche Plattformen erzeugen.

- Optimierungen des Compilers ausnutzen.
- Anderen Compiler ausprobieren.
- Andere virtuelle Maschine ausprobieren.
- Ein Compiler, der für eine sehr spezielle Hardware entwickelt wurde, zum Beispiel für eine Videospielkonsole, liefert im Gegensatz zu Cross-Compiler oft bessere Ergebnisse.

Ebenen der Optimierung

Maschinensprache

Mikroprozessor-Architektur und Betriebssystem müssen fest vorgegeben sein.

- Spezialbefehle etwa für SIMD nutzen, um gleichartige Berechnungen zu beschleunigen.
- Befehle umstellen oder ersetzt werden, um die Ausführung in der Pipeline des Mikroprozessors zu beschleunigen.
- Microcode der CPU abändern.

Teilweise durch Compiler-Erweiterungen (intrinsics) noch in Hochsprache möglich.

Ebenen der Optimierung

Betriebssystem

- Ein Betriebssystem steuert die Programmausführung.
- Es stellt Dienste wie Speicherverwaltung, Dateiverwaltung oder Prozessverwaltung zur Verfügung.
- 64-Bit-Betriebssysteme können in der Regel mehr Resourcen als 32-Bit-Betriebssystem zur Verfügung stellen.
- Auch die Anzahl Hardwareregister gehört dazu.
- Die Wahl des Betriebssystem kann großen Einfluss auf die Leistung eines Programms haben.

Ebenen der Optimierung

Hardware

Austauschen der Hardware mit

- schnelleren Mikroprozessoren,
- leistungsfähigereren GPU,
- mehr Haupt-, Festplatten- oder Cachespeicher,
- schnellerem Netzwerk.

Beschleunigung um konstanten Faktor
Die Grenzen dieser Optimierung sind durch die Technik und die betriebswirtschaftlichen Kosten meist vorgegeben.

Wann optimieren?

Die meisten Resourcen werden an nur wenigen Stellen in einem Programm verbraucht.

- Programm(teile) haben in der Regel keinerlei Optimierung nötig.
- Optimieren ist zeit- und arbeitsintensiv, das Ergebnis ungewiss.
- Aufgrund langsamen Hardware wurde vor Jahrzehnten oft frühzeitig optimiert,
- Wartbarkeit leidet unter Optimierung.

Wann optimieren?

Donald Knuth [Knuth, 1974, Seite 268]:

We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil. Yet we should not pass up our opportunities in that critical 3%. A good programmer will not be lulled into complacency by such reasoning, he will be wise to look carefully at the critical code; but only after that code has been identified.

Foto von Knuths Homepage `http://www-cs-faculty.stanford.edu/~knuth/dek-14May10-2.jpeg`

Wann optimieren?

Fazit: Ein Programm nicht zu früh und nur in den *nachweisbar* kritischen Teilen verbessern!

- Vorher ein lauffähiges, getestetes und korrektes Programm entwickeln.
- Auch notwendig um kritische Teile zu identifizieren.
- Resourcenverbrauch gezielt prüfen (to profile).

Profiler:

- Teile des Compiler, z.B. gcov und gprof bei der GCC.
- Eigenständige Programme, z.B. Valgrind, Linux Perf, OProfile.

Literatur

- Viel Literatur über High Performance (Scientific) Computing aber mit Focus auf Parallelrechner.
- Nicht viel Literatur über Optimierung von Programmen für Desktop-Rechnern.
- Agner Fog [Fog, 2017]
- Spezifische Optimierung ist meist Teil bestehender Literatur, z.B. Algorithmen, Compilerbau, C++.

Literaturverzeichnis

(2017).
Earth simulator.

Fog, A. (2017).
Optimizing software in C++. An optimization guide for Windows Linux and Mac platforms.
http://www.agner.org/optimize/optimizing_cpp.pdf.

IEEE (1984).
Software engineering standards : ANSI/IEEE Std 729-1983, glossary of software engineering terminology ...
Inst. of Electrical and Electronics Engineers, New York.

Knuth, D. E. (1974).
Structured programming with go to statements.
Computing Surveys, 6:261–301.

Moore, G. E. (1965).
Cramming more components onto integrated circuits.
Electronics, 38(8).