Leistung von Computersystemen

Ernst Haunschmid, Christoph Überhuber
Institut für Angewandte und Numerische Mathematik, Technische Universität Wien

Die Fortschritte bei der Entwicklung immer leistungsfähigerer Computer haben für deren Anwender nicht nur Vorteile gebracht. Mit wachsender Hardware-Leistung wurde nämlich die Diskrepanz zwischen der theoretisch möglichen Maximalleistung eines Computers – dessen *Peak Performance* – und der bei der Ausführung von Anwendungsprogrammen tatsächlich erzielten Leistung immer größer. Für den Anwendungsprogrammierer wurde es zunehmend schwerer, die potentiell vorhandenen Hardware-Ressourcen zufriedenstellend nutzen zu können.

Beispiel (Matrizenmultiplikation) Die Matrizenmultiplikation ist ein Grundbaustein vieler Algorithmen der numerischen Linearen Algebra. Die von Computern bei dieser Operation erreichte Leistung ist daher für das *Scientific Computing* von besonderer Wichtigkeit.

Der nächstliegende Algorithmus für die Matrizenmultiplikation – „Zeilen mal Spalten" – ergibt sich aus der Definition des Matrizenproduktes. Zur Berechnung von

$$C = A \cdot B \quad \text{mit } A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times n},\ C \in \mathbb{R}^{n\times n}$$

erhält man auf diese Weise z. B. folgenden Programmabschnitt:

```
c = 0
DO i = 1, n
   DO j = 1, n
      DO k = 1, n
         c(i,j) = c(i,j) + a(i,k)*b(k,j)     (1)
      END DO
   END DO
END DO
```

Dieser „klassische" Algorithmus ist aber *nicht* geeignet, auf modernen Computersystemen eine zufriedenstellende Rechenleistung zu erzielen. In Experimenten auf einem Prozessor des SGI Power Challenge XL Serversystems des EDV-Zentrums der TU Wien mit einer Maximalleistung von 390 Mflop/s wurde damit für kleinere bis mittelgroße Matrizen ($n \le 700$) nur eine Leistung von nicht einmal einem Drittel der maximalen Gleitpunktleistung erreicht. Für größere Matrizen, mit $n \ge 800$, sinkt die Rechnerleistung auf 10 % der maximalen Gleitpunktleistung.

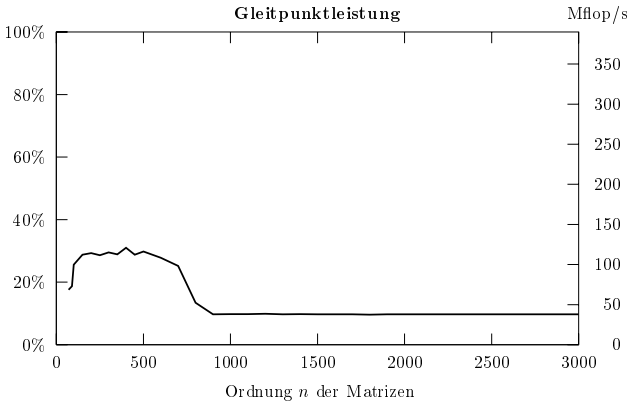

Abb. 1: Leistung und Wirkungsgrad eines Prozessors des SGI Power Challenge XL Servers bei der „klassischen" Matrizenmultiplikation (1)

Selbst bei einem derart einfachen Algorithmus ist offensichtlich die Ausnutzung des vorhandenen Hardware-Potentials nicht trivial.

Zeit = Arbeit/Leistung

Für den Benutzer eines Computersystems, der auf die Lösung einer konkreten Aufgabenstellung wartet, ist vor allem die dafür erforderliche Zeit von Interesse, die von zwei Einflußgrößen – Arbeit und Leistung – abhängt:

$$\text{Zeit} = \frac{\text{Arbeit}}{\text{Leistung}_{\text{effektiv}}} = \frac{\text{Arbeit}}{\text{Leistung}_{\text{maximal}} \cdot \text{Wirkungsgrad}}.$$

Für den Anwender sind daher folgende Kenngrößen von Interesse:

1. Die zu verrichtende *Arbeitsmenge*, die sowohl von der Art und Komplexität der Problemstellung als auch von Eigenschaften des Lösungsalgorithmus abhängt. Die Arbeitsmenge und damit der Zeitbedarf für die Problemlösung kann durch algorithmische Verbesserungen reduziert werden.

2. Die *Maximalleistung* ist ein Hardware-Charakteristikum des eingesetzten Computersystems, das von speziellen Anwendungsproblemen unabhängig ist. Die Anschaffung neuer Hardware mit größerer Maximalleistung führt (fast) immer zu einer Verringerung des Zeitbedarfs der Problemlösung.

3. Der *Wirkungsgrad* ist der Prozentsatz der Maximalleistung, der bei der Ausführung einer bestimmten Arbeitsmenge erreicht wird. Er bringt zum Ausdruck, in welchem Ausmaß die potentiellen Möglichkeiten des Computersystems von einem Programm genutzt werden. Er ist also eine Maßzahl dafür, wie gut die Implementierung, also die Umsetzung des Algorithmus in ein Computerprogramm, gelungen ist. Eine Erhöhung des Wirkungsgrades läßt sich durch Optimieren der Programme herbeiführen.

Art der Zeitmessung

Für den Benutzer ist fast immer nur jene Zeitdauer von Bedeutung, die der Computer für die Erledigung einer ihm übertragenen Aufgabe braucht. Diese Zeit, die als *Antwortzeit* (*response time*, *elapsed time*, manchmal auch als *wall-clock time*) bezeichnet wird, liegt zwischen dem Absenden des Kommandos zum Starten einer bestimmten Datenverarbeitungsaktivität (der vom Benutzer gewünschten Problemlösung) und deren Beendigung, also der „Antwort" des Rechners.

Dieser aus der Sicht jedes Anwenders entscheidende Wert ist jedoch meist von Faktoren abhängig, die sich geplanten, reproduzierbaren Experimenten entziehen. So kann

z. B. *Multitasking* die Ursache sein, daß der betrachtete Prozeß nicht die ganze Zeit aktiv ist, sondern die CPU zeitweise auch andere Aufgaben bearbeitet.

Besonders groß wird der Unterschied zwischen der Antwortzeit für den einzelnen Benutzer und dem systemintern für dessen Aufgabenstellungen benötigten Zeitaufwand im *Multi-user-Betrieb*.

Dem Umstand, daß beim Multi-user-Betrieb meist nur ein kleiner Teil der potentiell verfügbaren Rechnerkapazität einem speziellen Auftrag (*job*) gewidmet wird, trägt die *CPU-Zeit* Rechnung. Sie gibt an, wie lange die CPU effektiv mit der gestellten Aufgabe beschäftigt war, ohne deren zwischenzeitliche Belegung durch andere Prozesse oder Ein-/Ausgabe-Wartezeiten zu berücksichtigen. Die CPU-Zeit enthält jedoch *nicht* die Zeiten, die ein Anwendungsprogramm inaktiv warten muß (um z. B. das Eintreffen von Daten aus dem Sekundärspeicher abzuwarten), auch dann, wenn es das einzige im Computersystem ist !

Bei der Bewertung der Leistung eines Computersystems muß a priori die Entscheidung getroffen werden, welche Art der Zeitmessung die zu untersuchende Leistung charakterisieren soll. Dementsprechend unterscheidet man zwischen der *Gesamtsystemleistung* (*System Performance*), falls die Antwortzeit zugrundegelegt wird, und der *CPU-Leistung* (*CPU Performance*), wenn die CPU-Zeit verwendet wird.

Arbeitsmenge

Die *Komplexität* eines Algorithmus ist ein Maß für die Arbeit, die bei dessen Abarbeitung zu verrichten ist. Sie ist nicht nur ein Charakteristikum des Lösungsverfahrens, sondern hängt auch vom Schwierigkeitsgrad (Umfang) des behandelten Problems ab. Meist verwendet man zur Charakterisierung des Schwierigkeitsgrades eines Problems nur eine einzige skalare Kennzahl. So wird z. B. der Arbeitsaufwand zur Lösung eines Systems von n linearen Gleichungen in n Unbekannten im allgemeinen in Abhängigkeit von der Dimension n als Parameter angegeben.

Aussagen über den Abarbeitungsaufwand eines Algorithmus werden immer in Abhängigkeit von der Schwierigkeitskennzahl des Problems gemacht. Dabei wird meist die Anzahl der *Berechnungsschritte*, die zur Durchführung des Algorithmus bei einem bestimmten Problemumfang benötigt werden, ermittelt.

Trotz der Abstraktion, die in der Verwendung von „Rechenschritten“ zum Ausdruck kommt, sind derartige Komplexitätsbetrachtungen nicht von speziellen Hardwareeigenschaften unabhängig. Je nachdem, was als gleichsam „unteilbare“ *Einheit der Arbeit* angesehen wird, ergeben sich signifikante Unterschiede bei der Bestimmung der Komplexität eines Algorithmus. Im Bereich der Numerik ist es üblich, eine Gleitpunktoperation als eine solche elementare Einheit der Arbeit anzusehen.

Beispiel (Gauß-Algorithmus) Der Eliminationsalgorithmus (LU-Zerlegung) zur Lösung linearer Gleichungssysteme benötigt, abhängig von der Anzahl n der Gleichungen,

$K(n) = 2n^3 / 3 + 3n^2 / 2 - 7n / 6$ Gleitpunktoperationen.

Wenn man mit dieser Formel den Rechenaufwand des Eliminationsalgorithmus charakterisiert, so vernachlässigt man den Zeitaufwand, der für spezielle algorithmische Maßnahmen (Pivotsuche, Zeilenvertauschen etc.) erforderlich ist.

Jedoch ist es vor allem im Bereich der Parallelrechner möglich und sinnvoll, auch komplexere Operationen (Matrix-Vektor-Operationen, Matrix-Matrix-Operationen, Transponieren einer Matrix, eine schnelle Fourier-Transformation) zu einem elementaren „Rechenschritt“ zusammenzufassen.

Präzisierung des Begriffs „Operation“

Bei der praktischen Ermittlung des Arbeitsaufwandes eines bestimmten Algorithmus muß berücksichtigt werden, daß auf verschiedenen Computersystemen Gleitpunktoperationen in unterschiedlicher Form vorhanden sind. Beispielsweise sind auf vielen Mikroprozessoren neben den Instruktionen für die arithmetischen Operationen + - · / eigene Instruktionen für die Berechnung der Quadratwurzel-, Sinus- und Kosinusfunktion etc. vorhanden. Einige Mikroprozessoren besitzen eine Instruktion, die eine Multiplikation und eine davon abhängige Addition – die Operation $(a \cdot b) + c$ – in derselben Zeit wie eine einzelne Gleitpunktaddition bzw. -Multiplikation ausführen kann (*Floating-Point Multiply and Add*).

Die im Instruktionssatz eines Prozessors verfügbaren Gleitpunktoperationen sind *nicht* homogen bezüglich ihres Zeitaufwandes; so benötigen z. B. Gleitpunktadditionen und -Multiplikationen meist deutlich weniger Taktzyklen als Gleitpunktdivisionen. Diese Tatsache darf sowohl bei der Aufwandsbewertung von Algorithmen als auch bei der empirischen Leistungsbewertung von Computersystemen nicht vernachlässigt werden.

Maximale Gleitpunktleistung

Eine wichtige Hardware-Kennzahl ist die *Maximalleistung* (*peak performance*) P_{max} eines Computers. Sie entspricht der theoretisch möglichen Maximalanzahl von Gleitpunkt- (oder anderen) Operationen, die von diesem Computer pro Zeiteinheit (meist pro Sekunde) durchgeführt werden können. Falls sich P_{max} auf die pro Sekunde ausführbaren Gleitpunktoperationen bezieht, so erhält man die *maximale Gleitpunktleistung* mit der Einheit

flop/s (*floating-point operations per second*)

bzw. Mflop/s (10^6 flop/s) oder Gflop/s (10^9 flop/s). Es wird dabei oft irreführenderweise nicht zwischen den verschiedenen Arten von Gleitpunktoperationen und deren unterschiedlichen Ausführungszeiten unterschieden.

Manchmal wird anstelle von flop/s, Mflop/s etc. die Schreibweise flops, Mflops etc. verwendet. Diese bringt aber den Charakter der Leistungskennzahl *nicht* zum Ausdruck, sondern entspricht eher einer Mehrzahlform.

Eine Aussage, die man auf Grund einer Maximalleistungsangabe mit Sicherheit machen kann, ist, daß man sie mit keinem noch so guten Programm, das auf dem betreffenden Computer läuft, je übertreffen wird. Selbst in die Nähe der Maximalleistung kommen nur einzelne, besonders optimierte Programmabschnitte. Die Ursache dafür liegt unter anderem in der Vernachlässigung des Zeitaufwandes aller

unterstützenden Adreßberechnungen, Speicheroperationen etc., die nur indirekt zur Ermittlung des Resultates beitragen. Man kann daher die Maximalleistung als eine Art „Lichtgeschwindigkeit“ des jeweiligen Computers ansehen.

Beispiel (HP-Workstation) Das Modell K460 der HP-Workstations hat eine Zykluszeit von T_c = 5.556 ns. In einem Taktzyklus können maximal vier Gleitpunktoperationen – je eine Multiply-*and*-Add-Instruktion in den zwei parallelen Gleitpunkt-Pipelines – ausgeführt werden. Als maximale Gleitpunktleistung ergibt sich damit:

$$P_{\max} = \frac{N_c}{T_c} = \frac{4 \text{ Operationen}}{5.556 \cdot 10^{-9} \text{ Sekunden}} = 720 \text{ Mflop / s.}$$

Je geringer der Anteil der Multiply-and-Add-Instruktionen an der Gesamtheit der konkret auszuführenden Gleitpunktoperationen eines Programms ist, desto weiter entfernt sich die reale Leistung dieser Workstation von ihrer Maximalleistung.

Empirische Gleitpunktleistung

Wenn man die in einem Zeitraum T verrichtete Arbeit durch die Anzahl W_F der in T ausgeführten Gleitpunktoperationen charakterisiert, so erhält man die *Gleitpunktleistung* (*floating-point performance*)

$$P_F[\text{flop / s}] = \frac{W_F}{T} = \frac{\text{Anzahl der ausgeführten Gleitpunktoperationen}}{\text{Zeitdauer in Sekunden}}.$$

Im Gegensatz zu der rechnerisch ermittelten Maximalleistung gewinnt man diesen empirischen Wert durch Messungen an laufenden Programmen. Zahlenangaben erfolgen wie bei der analytischen Leistungsbewertung in Mflop/s oder Gflop/s.

Interpretation empirischer Leistungswerte

Die empirische Ermittlung der Gleitpunktleistung eines Computersystems kann nur mit Hilfe konkreter Algorithmen – in Form von Programmen – erfolgen. Man darf aber nicht den Fehler machen, die Gleitpunktleistung als absoluten Qualitätsmaßstab zur Beurteilung von *Algorithmen* heranzuziehen.

Ein Programm, das eine größere Gleitpunktleistung erzielt, ermöglicht nicht notwendigerweise auch eine kürzere CPU-Zeit. Trotz besserer (höherer) flop/s-Werte kann ein Programm unter Umständen *länger* für die Lösung des Problems brauchen, wenn bei seiner Verwendung mehr Arbeit zu verrichten ist. Nur bei Programmen mit demselben Arbeitsaufwand liefern die Werte der empirischen Gleitpunktleistung eine Vergleichsbasis für die Qualität der *Implementierungen*.

Beispiel (Leistung und Zeit) Bei der Lösung eines speziellen Diffusionsproblems auf einem Parallelrechner wurden mit einem iterativen, lokalen Jakobi-Verfahren 5.6-fach bessere Mflop/s-Raten erzielt als mit einem Multigrid-Verfahren. Das Jacobi-Verfahren benötigte jedoch – auf Grund des wesentlich höheren Arbeitsaufwandes – die 317-fache Gesamtrechenzeit des Multigrid-Verfahrens. Eine sinnvolle Beurteilung der beiden Verfahren ist hier nur mit Hilfe der CPU-Zeit möglich.

Zur Bewertung verschiedener Computersysteme mit ein und demselben Programm (*Benchmark*-Bewertung) ist die empirische Gleitpunktleistung ebenfalls geeignet und wird auch häufig verwendet (z. B. LINPACK-Benchmark, SPECfp92).

Empirischer Wirkungsgrad

Setzt man eine empirisch ermittelte Gleitpunktleistung in Relation zur Maximalleistung des verwendeten Computers, so erhält man Aufschluß darüber, in welchem Ausmaß das untersuchte Programm und der involvierte Compiler imstande sind, die potentielle Leistungsfähigkeit des Computers zu nutzen. Der so erhaltene *empirische Wirkungsgrad* liegt meist deutlich unter 100 % (siehe Abb. 1).

Beispiel (LINPACK-Benchmark) Beim LINPACK-Benchmark (Dongarra [1]) ergeben sich z. B. die in Tabelle 1 zusammengestellten Leistungen und empirischen Wirkungsgrade. Die Werte für n = 100 Gleichungen sind auf Grund von Zeitmessungen mit den unveränderten „Originalprogrammen“ `LINPACK/sgefa` und `LINPACK/sgesl` ermittelt worden.

Für n = 1000 wurden äquivalente Programme entwickelt, die eine möglichst gute Ausnutzung der Ressourcen ermöglichen. Wie man sieht, führt eine derartige (maschinenabhängige) Programmoptimierung zu erheblich besseren Wirkungsgraden.

Tabelle 1: Empirische Leistungen und Wirkungsgrade beim LINPACK-Benchmark

Computer		LINPACK-Benchmark			
Type	Maximal-leistung	n = 100		n = 1000	
	[Mflop/s]	[Mflop/s]	[%]	[Mflop/s]	[%]
NEC SX-4/2B (1 proc.)	1800	505	28	1750	97
HP 9000/K460	720	158	22	510	71
SGI Power Challenge XL	390	86	22	344	88
DEC 2100 4/275	275	52	19	208	76
IBM RS/6000-390	268	53	20	181	68

Literatur

[1] J. J. Dongarra: *The* LINPACK *Benchmark - An Explanation*, in „Evaluating Supercomputers“ (A. J. Van der Steen, Ed.). Chapman and Hall, London 1990, pp. 1-21.

[2] K. Dowd: *High Performance Computing*. O'Reilly & Associates, Sebastopol 1993.

[3] C. W. Ueberhuber: *Numerical Computation I*. Springer-Verlag, Berlin Heidelberg New York Tokyo 1997.