schema-F

Einführung in die technischen und theoretischen Grundlagen der Informatik

Speicherverwaltung und Cache
Erläuterungen

Lernmaterial
zum Modul
- 31231 -
der Fernuniversität
Hagen

Inhaltsverzeichnis

Vorwort

Die Informatik ist eine mathematische Wissenschaft, die sich mit der strukturierten Verarbeitung von Informationen beschäftigt. Ziel ist es, strukturiert Lösungsansätze für verschiedene Probleme zu finden. Die Informationsverarbeitung und Strukturierung geschieht hierbei häufig im Hinblick auf die Automatisierung der Verarbeitung.
Um die Vorgänge zu verstehen, ist es unerlässlich die Hintergründe zu kennen. Ein wichtiges Mittel, dieses Verständis zu erlangen, ist die Abstraktion der Sachverhalte. Auf diese Weise können die Problemstellungen auf einer Ebene betrachtet werden, auf welcher ausschließlich die Informationen dargestellt werden, welche für die Problemstellung relevant sind. Diese Abstraktion wird im folgenden Skript - unmerklich - fortwährend angewandt, da bei Betrachtung des Gesamtsystems die Menge der Informationen unnötig und vollkommen unüberschaubar wäre.
Die theoretische Informatik beschäftigt sich mit verschiedenen Problemen auch ohne technische Aspekte. Hauptaugenmerk liegt hierbei auf der Abstraktion realer Probleme und deren Formalisierung. Nur auf diese Weise können im Endeffekt Probleme unserer Welt in eine Form übersetzt und strukturiert werden, in der eine Maschine wie ein Computer in der Lage ist, diese zu bearbeiten und zu lösen. Der hohe Grad an Abstraktion wird dabei besonders deutlich. Dieser ist allerdings häufig auch nötig, da es sich teilweise um äußerst komplexe Probleme handelt. Als Beispiel könnte man eine Landkarte nehmen, wie sie sich noch vor 15 Jahren in fast jedem Auto befand. Diese Karte gibt sehr detaillierte Informationen über den realen Zustand. Viele dieser Informationen sind allerdings nicht nötig. Wenn es nur um das Straßennetz geht, ist es komfortabler alle anderen Informationen daraus wegzulassen. Ein Beispiel wäre die Google Maps Karte (nicht Satellit). Möchte man nur den kürzesten Weg zwischen zwei Punkten finden, kann man weiter abstrahieren. Am Ende steht ein Graph mit Knoten (z.B. Kreuzungen), Kanten (Wege) und Kantengewichten (Wegstrecke). Wie viele Kurven die Straßen haben spielt keine Rolle. Auf solchen abstrakten Konstrukten mit minimalen Informationen kann dann bequem gearbeitet werden. Auch hier setzt die theoretische Informatik an, indem sie sich mit dem Finden und Beweisen schneller und effizienter Algorithmen beschäftigt, da Zeit und andere Ressourcen immer eine wichtige Rolle spielen. Als Hilfsmittel stehen hier verschiedenste Modelle wie Graphen, Automaten und Mengen zur Verfügung.
Die technische Informatik beschäftigt sich mit den technischen Möglichkeiten und Gegebenheiten zur Realisierung von meist elektrischen Schaltungen. Dies reicht von den elektrischen Grundbauteilen wie Transistoren und Schaltungen im Elektrotechnikbereich über Gatterlogiken, Gatterarrays und Datenpfade bis hin zu Prozessorrealisierung und Betriebsystemen. Auch hier wird mit zunehmender Komplexität immer weiter abstrahiert. Es ist bei der Befehlsverarbeitung im Prozessor nicht mehr wirklich relevant, wie die Transitoren aufgebaut und geschaltet sind. Ein weiterer wichtiger Teil der technischen Informatik ist der Aufbau des Betriebssystems, da dieses die Hardware gegenüber dem Benutzer und den Anwendungsprogrammen verwaltet.

Viele könnten sich an dieser Stelle fragen, wozu sie diese Details nun kennen müssen. Vor allem wenn man zum Beispiel als Programmierer tätig ist, scheinen einem diese Informationen für die Tätigkeit nicht relevant zu sein, da man nur den Computer bedient und in einem Anwendungsprogramm arbeitet. Tatsächlich hilft das Wissen über die Hintergründe und Zusammenhänge im Bereich der theoretischen und technischen Aspekte auch dem einfachen Programmierer. Es beginnt prinzipiell mit der folgenden Herangehensweise. Da man die Probleme erst einmal in mathematische Form bringen muss, kann man sich die gegebenen Techniken zur Formalisierung zu Nutze mache. Gleichzeitig

bringt einem die Bearbeitung derartiger Problemstellungen zu einer strukturierten Denkweise. Hat man diesen Schritt nun geschafft, kann man natürlich beginnen Programmcode zu schreiben und die Lösung des gestellten Problems zu erarbeiten. Doch auch hierbei ist nicht jede Lösung gleich und vor allem nicht gleich gut. Die Lösung soll optimalerweise mit möglichst geringem Einsatz von Ressourcen wie Speicher auskommen und möglichst schnell durch das Programm bearbeitet werden. Zusätzlich ist es eventuell notwendig zu zeigen, dass das Programm immer die richtige Lösung liefert. Feinheiten wie die Konstruktion von Schleifen, welche der dynamischen Sprungvorhersage eines Prozessors helfen und Berechnungen, welche trotz begrenzter Ressourcen, hohe Genauigkeiten bewirken und Rundungsfehler abschwächen, können zur Qualität des Programms beitragen.

Unbeachtet der Qualitätssteigerung der eigenen Arbeit und der Möglichkeit, eventuell neue technische Wege zu beschreiten, wofür bekannt sein muss was es bereits gibt, macht es auch einfach Spaß dieses tiefere Verständnis um die Funktionsweisen und Zusammenhänge zu erlangen - mir zumindest.

1 Speicherverwaltung

Erklärung

Um den zur Verfügung stehenden physikalischen Speicher möglichst effizient nutzen zu können, werde durch das Betriebssystem verschiedene Techniken eingesetzt. Den einzelnen Prozessen werden meist eigene Adressräume zugeordnet, welche eine zusammenhängende Menge von Adressen darstellen. Der sog. virtuelle Speicher ist eine Art Adressraum. bei welcher die physikalischen und technischen Details für den Prozess verborgen sind. Hierbei ist es auch möglich, bestimmte Beschränkungen aufzuheben. Die Abbildung des physikalischen Speichers auf den virtuellen Speicher erfolgt zur Laufzeit des Prozesses durch die Memory Management Unit des Prozessors. In den bekannten Betriebssystemen ist die seitenbasierte Speicherbelegung das Standardverfahren. Bei der Seitenadressierung werden der virtuelle und der physikalische Speicher in Stücke fester Länge eingeteilt. Stücke im logischen Adressraum heißen Seiten, im physikalischen Adressraum Kacheln. Neben der Adresse enthält jeder Eintrag der Seitentabelle normalerweise noch weitere Informationen wie Modifikationsbit (dirty-bit), Präsenzbit (valid bit) und Referenzbit (wurde bereits zugegriffen). Ist bei einem Zugriff auf eine Seite diese nicht vorhanden, wird ein Seitenfehler ausgelöst. Das Laden der Seiten kann nach verschiedenen Techniken geschehen. Bei dem Pre-Paging werden die benötigten Seiten vor Verwendung bereits geladen. Nachteilig ist hierbei, dass das Prozessverhalten im Vorraus bekannt sein muss. Beim Demand-Paging werden Seiten nur auf Anfrage bzw. nach einem Seitenfehler geladen. In der Praxis wird häufig eine Kombination aus beiden Techniken angewendet.

1.1 Seitentabelle analysieren

Erklärung

Wie bereits erwähnt müssen die logischen bzw. virtuellen Adressen auf die physikalischen Adressen abgebildet werden. Diese Aufgabe wird durch ein Paging-System übernommen. Wie das gemacht wird, soll anhand der folgenden Beispielaufgabe erläutert werden.

Beispielaufgabe 1

Ein Paging-System hat einen maximalen logischen Speicher der Größe 16 GByte. Die Seitengröße beträgt 4 KByte und der maximale physische Speicher ist 4 GByte groß. Ein Abschnitt der Seitentabelle eines aktiven Prozesses hat die folgenden Einträge:

Seitennummer	Seitenrahmennummer
0	4
1	2
2	0
3	1
4	6
5	3

a) Wie viele Bit hat eine logische Adresse?
b) Wie viele Bit hat ein Eintrag in der Seitentabelle?
c) Welche physische Adresse hat die logische Adresse 4097?
d) Welche logische Adresse wird auf die physische Adresse 10020 abgebildet?

a) Um die Bits für die logischen Adresse zu ermitteln müssen wir uns die Größe des logischen Speichers betrachten.

$16GB = 2^4 * 2^{30} = 2^{34}$

Eine logische Adresse benötigt also 34 Bit

b) Der physikalische Speicher wird in gleich große Seitenrahmen aufgeteilt. Der 4 GB Speicher wird in 4 KB Seitenrahem aufgeteilt.

$4GB/4KB = 2^{32}/2^{12} = 2^{20}$

Wir benötigen also 20 Bit für einen Eintrag.

c) Da die Seitengröße 4 KB also 4096 Byte beträgt, sind die logischen Adressen auch so durchnum-

meriert.

Die Seitennummer ist also 1. Zur Seitennummer 1 gehört die Seitenrahmennummer 2. Beim Suchen der phyischen oder logischen Adresse wird immer durch die Seitengröße mit Rest geteilt. Die logische Adresse ist 4097, die Seitengröße 4096.

$\frac{4097}{4096} = 1 \; Rest \; 1$

Die 1 vor dem Rest bedeutet, dass die Adresse in der Seite mit der Nummer 1 (wie oben festgestellt) liegt. Die 1 hinter dem Rest ist das Offset. Diese wird für die weitere Adressierung verwendet: Da die Seitenrahmennummer der ersten Seite 2 ist, wird 2 mit der Seitengröße multipliziert und danach noch das Offset addiert.

$2 \cdot 4096 + 1 = 8193$

Die logische Adresse 4097 wird demnach auf die physische Adresse 8193 abgebildet.

d)Die physische Adresse wird durch die Seitengröße dividiert:

$10020/4096 = 2 \; Rest \; 1828$

Die physische Adresse 10020 liegt im Seitenrahmen mit der Nummer 2 und Offset 1828. Da die logische Seite 1 auf Seitenrahmen 2 abgebildet wird, wird 1 mit der Seitengröße multipliziert und das Offset dazu addiert:

$1 \cdot 4096 + 1828 = 5924$

Die physische Adresse 10020 wird demnach auf die logische Adresse 5924 abgebildet.