Proseminar Konzepte von Betriebssystem-Komponenten
Disk-Caches & Dateizugriff
von Athanasia Kaisa

Massenspeichermedien sind in der Regel gegenüber dem Hauptspeicher eines Rechners viel langsamer. Da Massenspeicher immer größer und billiger werden, können auch immer größere Datenmengen übertragen werden. Um Wartezeiten zu verkürzen, nutzt man schnelle Zwischenspeicher zum vorübergehenden Einlagern von Daten, so genannte Caches, bzw. in diesem Fall Disk-Caches.

Grundzüge eines Zwischenspeichers

Wenige Zugriffe auf große Datenmengen benötigen in der Regel weniger Zeit, als mehrere Zugriffe auf kleinere Datenmengen. Gelingt es nun, aus irgendwelchen Zusammenhängen zu erschließen, welche Daten in der Zukunft benötigt werden, kann man bei Anforderung einiger weniger Datenblöcke gleich eine große Zahl in den Zwischenspeicher einlesen. Lag man mit der Vorausschau richtig, kann dadurch die Leistung des Computers erhöht werden. War die Annahme falsch, muss man die anderen Datenblöcke erneut einlesen und verliert Leistung. Je besser die Methoden zur Vorausschau sind, desto mehr erhöht ein Zwischenspeicher die Gesamtleistung des Computers. Bei einem Plattenzwischenspeicher werden Zugriffe auf die Festplatte über einen Bereich im Hauptspeicher oder einen Zwischenspeicher oder im Laufwerk selbst gespeichert.

Zwischenspeichern ist ein Vorgang bei dem schneller Speicher dazu verwendet wird, einen Teil der Daten eines langsameren Speichers aufzunehmen. Es gibt verschiedene Arten von Zwischenspeichern, normalerweise zwei, manchmal auch drei:

1. Auf der Steuerplatine der Festplatte selbst ist ein Zwischenspeicher vorhanden, der dazu dient mechanische Schreib-/Lesevorgänge von Datenübertragungen durch den Datenbus abzutrennen. Er enthält Daten die kürzlich übertragen wurden. Je größer der Speicher ist, um so schneller erfolgt die Datenübertragung, weil mehr Daten eingelagert werden, und somit die zeitaufwendigen Blocksuchen auf dem Rotationskörper vermieden werden. Bei modernen IDE- und SATA-Festplatten ist der Zwischenspeicher bis zu 8 MB groß.

2. Bei teuren SCSI-Systemen befindet sich auf dem SCSI-Controller ein dritter Zwischenspeicher. Er befindet sich dann zwischen dem Zwischenspeicher der Festplatte und dem Zwischenspeicher der Software. Der Zwischenspeicher kann eine Größe von bis zu 256 MB haben. Werden vom Betriebssystem Daten angefordert die nicht im Software-Zwischenspeicher stehen, können diese im Controller zwischengespeichert sein und von da gelesen werden. Die Festplatte muss dazu nicht angesprochen werden, was auch den SCSI-Bus entlastet.

3. Zusätzlich zu diesen Hardware-Zwischenspeichern, gibt es auch oft einen Software-Zwischenspeicher, der vom Betriebssystem verwaltet wird. Es ist vom Betriebssystem abhängig, wieviele MB des Hauptspeichers bereitgestellt werden. Werden die gleichen Daten ein weiteres mal angefordert, können sie aus dem Hauptspeicher des Rechners gelesen werden, ohne die Festplatte ansprechen zu müssen.
   Zu den gebräuchlichsten Techniken gehören der Blockcache. Hier werden

Festplattenblöcke als Abbild im Hauptspeicher des Rechners gehalten. Es müssen dann die Leseaufträge überprüft werden, ob die gewünschten Daten schon im Zwischenspeicher sind. Wenn nicht, werden die Daten von der Festplatte eingelesen und in den Zwischenspeicher kopiert. Ist er voll, muss ein anderer Block entfernt, d.h. auf die Festplatte zurückgeschrieben, werden.
Wird ein im Zwischenspeicher befindlicher Block verändert, so muss er wieder auf die Festplatte zurückgeschrieben werden. Dies muss nicht immer sofort geschehen, meist aber in bestimmten Zeitintervallen. Dabei besteht jedoch die Gefahr, dass modifizierte Blöcke nicht auf die Platte zurückgeschrieben werden, wenn z.B. zu schreibende Blöcke zwischengespeichert werden und die Stromzufuhr zum Rechner unterbrochen wird. Das führt zu Datenverlust und im Extremfall ist das Dateisystem beschädigt.

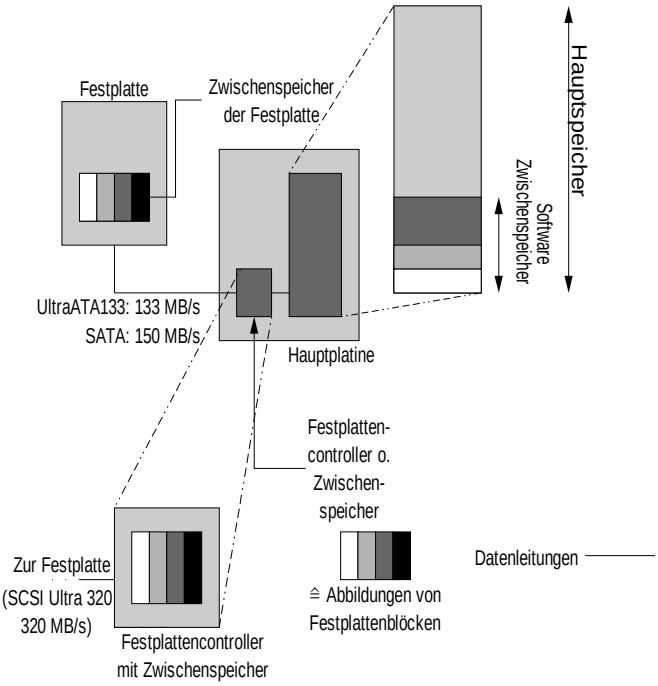

Abbildung 1: hier wird die Anordnung der verschiedenen Zwischenspeicher dargestellt

Plattenzwischenspeicher in LINUX

LINUX nutzt, wie die meisten Betriebssysteme, einen Blockcache. Die Blöcke einer Festplatte werden im Hauptspeicher abgebildet und durch so genannte Pufferköpfe referenziert. Ein Puffer ist die Abbildung eines Festplattenblocks im Zwischenspeicher. Die Blockgröße beträgt ein Vielfaches von 512 Byte. Ein Block ist zu laden, wenn er noch nicht im Zwischenspeicher vorhanden ist. Er ist zu schreiben, wenn der Inhalt dieses zwischengespeicherten Blocks verändert wurde. Dieser Block wird dann als verändert gekennzeichnet.
Oft angeforderte Daten verbleiben länger im Zwischenspeicher, als selten angeforderte.

Die Datenstruktur der Pufferköpfe

Zu jedem Puffer gehört ein Pufferkopf. Der Pufferkopf enthält u.a. die Speicheradresse an der die Daten des Puffers zu finden sind. Es muss aber nicht jeder Pufferkopf verwendet werden, da sich der Kernel immer einen Vorrat von Pufferköpfe frei hält, um nicht ständig Speicher für neue Pufferköpfe allokieren zu müssen. Dadurch kann Rechenzeit eingespart werden!

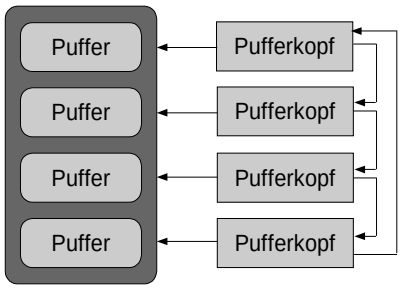

Abbildung 2: Funktionsweise der Pufferköpfe

Pufferköpfe können in verschiedene Zustände gebracht werden:

- **unbenutzter Pufferkopf**
  - ist verfügbar, aber Inhalt ist bedeutungslos
- **Pufferkopf für einen freien Puffer**
  - zeigt auf einen leeren Puffer; Nur der Puffer ist verfügbar, nicht aber der Pufferkopf!
- **Pufferkopf für einen verwendeten Puffer**
  - zeigt auf einen Puffer im Zwischenspeicher
- **asynchroner Pufferkopf**
  - zeigt auf einen temporär verwendeten Puffer, der für Ein-/Ausgabe Operationen verwendet wird

Bestimmte Kernfäden, die die Puffer wieder auf die Festplatte schreiben, sind bdflush und kupdated im 2.4 Kernel. Man kann diese Prozesse mit ps anzeigen lassen.

kupdate
kupdate schreibt modifizierte Puffer die lang nicht mehr benutzt wurden, sowie Superblock- und Inode Informationen im Zeitintervall von 5 Sekunden auf die Platte zurück. Alle 30 Sekunden werden modifizierte Puffer zurückgeschrieben. Es werden aber keine "jungen" Puffer zurückgeschrieben.

bdflush
bdflush wird standardmäßig alle 5 Sekunden ausgeführt. Befinden sich zu diesem Zeitpunkt 30% der genutzten Puffer, die verändert wurden, im Zwischenspeicher, werden max. 64 Puffer zurückgeschrieben. Sind 60% der momentanen Pufferköpfe modifiziert, wird nicht auf den 5 sekündigen Aufruf von bdflush gewartet.

Wie schon oben erwähnt, schreibt LINUX nicht sofort auf die Platte zurück, denn wenige große Datenübertragungen nehmen weniger Zeit in Anspruch, wie mehrere kleine.

Die Listenstrukturen des Zwischenspeichers

LINUX verwaltet die Pufferköpfe in einer Vielzahl von verschiedenen Listen. Pufferköpfe freier Puffer werden in doppelt verketteten Ringlisten verwaltet. Zu jeder Blockgröße gehört eine extra Liste. Zusätzlich wird im Pufferkopf eingetragen, dass der Puffer gerade nicht benutzt wird.

Pufferköpfe benutzter Puffer werden in einer Reihe spezieller LRU-Listen (least recently used, zuletzt verwendet) verwaltet. Auch hier gibt es zu jeder Blockgröße eine separate Liste.

BUF_CLEAN
- Nimmt Pufferköpfe nicht modifizierter Puffer auf. Das muss nicht gleichzeitig heißen, dass die Puffer dem Inhalt der Platte entsprechen! Die Liste dient also nur dazu, um ihre Puffer vor Zugriffen von Funktionen zu schützen, die das Zurückschreiben von Puffern übernehmen

BUF_DIRTY
- Puffer, deren Inhalt nicht mit den zugehörigen Blöcken auf der Festplatte übereinstimmen

BUF_LOCKED
- Enthält Pufferköpfe, deren zugehörige Puffer modifiziert sind und gewählt wurden, um auf die Platte zurückgeschrieben zu werden.

Benutzte Pufferköpfe werden in doppelt verkettete Hashlisten gespeichert. Anhand der Geräte- und Blocknummer werden die Puffer ausfindig gemacht. Das Verfahren ist in diesem Fall offenes Hashing. Durch Hashing werden die Puffer möglichst schnell verwaltet.

Verdrängungsstrategien

Wenn ein Block in einen vollen Cache geladen werden muss, ist ein Block zu entfernen und auf die Platte zu schreiben wenn er veränderdert wurde. Dazu gibt es Verdrängungsstrategien, die denen der Seitenauslagerung ähnlich sind. Es sind zum Beispiel FIFO, Second-Chance und LRU einsetzbar.
FIFO: Hier wird der älteste im Cache befindliche Block entfernt.
Second-Chance: Ist eine FIFO Variation.
LRU: Hier wird der am längsten unbenutzte Block entfernt.

LINUX benutzt die Strategie des LRU-Verdrängungsalgorithmus.

Verwendung des Zwischenspeichers

Soll ein Block von einem Gerät gelesen werden, wird zu erst überprüft, ob für den Block nicht schon ein entsprechender Block im Zwischenspeicher vorhanden ist. Ist das nicht der Fall muss der Puffer durch das Lesen von einem externen Medium angefordert werden.

Der Prozess, der die Daten angefordert hat, muss allerdings so lange warten, bis die gewünschten Daten im Puffer vorhanden sind.
Zum Zurückschreiben modifizierter Blöcke, incl. Inodes und Superblöcke gibt es die Funtkion sync. Bei dieser Funktion, werden Prozesse nicht angehalten, bis ein Schreibvorgang erfolgreich beendet wurde. Durch die Funktion wird der gesamte Zwischenspeicher nach modifizierten Puffern durchsucht und für jeden Gefundenen eine Schreibanforderung an einen entsprechenden Gerätetreiber geschickt.

Dateizugriff unter LINUX

Unter LINUX wird der Dateizugriff durch den Verzeichniscache beschleunigt. Er ist Teil des Virtuellen Dateisystems und kann daher durch alle Dateisystemimplementierungen genutzt werden.

Das Problem, dass Benutzer mit Dateinamen arbeiten, der Kernel aber mit Inodes arbeitet, wird durch den Verzeichniscache gelöst. Ohne diesen Verzeichniscache müsste der Kernel, bei jedem Zugriff auf die gleiche Datei, erneut die Inodes ermitteln, was Zeit kosten würde.

Sobald eine Datei angefordert wird, werden in den Verzeichniscache die Informationen des Ganzen Verzeichnisses, in der sich die angeforderte Datei befindet, geladen. Solche Einträge in den Verzeichniscache werden DEntry genannt.

Gespeichert wird u.a. der Dateiname, die zugehörige Inode, der zugehörige Superblock, Unterverzeichnisse usw.

Der Verzeichniscache ist eine globale Hashliste in der doppelt verketteten Listen eingetragen sind. Die Position der Unterliste in der globalen Hashliste bestimmt sich aus dem Hashwert des Namens und der Adresse des DEntry-Eintrages des übergeordneten Verzeichnisses.

Zusammenfassung

Anhand von Zwischenspeichern lässt sich die Gemessenheit der Festplatte zwar ansatzweise kompensieren, doch Bussysteme wie SATA und SCSI könnten einen weit größeren Datendurchsatz bewältigen.
Zwischenspeicher lassen sich nicht nur sinnvoll in einem Rechner einsetzen, sondern bringen auch im Internet oder Netzwerken einen ernormen Leistungszuwachs.

Literaturverzeichnis:

LINUX Kernelprogrammierung, Algorithmen und Strukturen der Version 2.4 (6. Auflage)
Michael Beck, Harald Böhme, Mirko Dziadzka Ulrich Kunitz, Robert Magnus, Claus Schröter, Dirk Verworner, 2001, Addison Wesley Verlag

Betriebssysteme
Tanenbaum Andrew S., 1990, Hanser Verlag

Architektur von Betriebssystemen (3. Auflage)
H. Wettstein, 1987, Hanser Verlag