Betriebsprogrammierung II - SS 2000

Inhalt

Teil 1 Pufferspeicher (Caches)

1 Vorstellungen und Motivation

2 In Hardware realisierte Pufferspeicher

C. Schimmel: "UNIX Systems for Modern Architectures", Addison-Wesley Publishing Company, 1994.

2.1 Monoprozessoren einschließlich E/A

2.1.1 Grundlagen

2.1.2 Virtuelle Caches

2.1.3 Virtuelle Caches mit Prozeß-Schlüsseln

2.1.4 Virtuelle Caches mit physikalischen Adressen als Tags

2.1.5 Physikalische Caches

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann
Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors unzulässig

2.1.6 Techniken zur effizienten Cache-Verwaltung

2.2 Multiprozessoren (einschl. Koordinierung)

2.2.1 Grundlagen

2.2.2 Konsistenzprotokolle: Das MESI-Protokoll

2.2.3 Implementierung von Spinlocks und Barrieren

3 In Software realisierte Pufferspeicher

3.1 Virtueller Speicher

3.2 Verteilter, gemeinsam genutzter Speicher

3.3 Verteilte Dateisysteme

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann
Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors unzulässig

Teil 2 Prozessorvergabestrategien

F. Hofmann: "Betriebssysteme: Grundkonzepte und Modellvorstellungen", 2. Auflage, Leitfäden der angewandten Informatik, B. G. Teubner Stuttgart, 1991.

4 Prozessorvergabe in Monoprozessoren

4.1 Die operationelle Methode

4.2 Die analytische Methode

4.2.1 Wahrscheinlichkeitstheoretische Grundlagen

4.2.2 Einfache Modelle

4.2.3 Wartezeiten in Abhängigkeit von der geforderten Bedienzeit

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann
Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors unzulässig

5 Pozessorvergabe in Multiprozessoren

5.1 Aufgabenstellung

5.2 Prozessorvergabe für Kern-Fäden

6 Prozessorvergabe in verteilten Systemen

6.1 Systemmodelle

6.2 Lastverteilung

6.3 Komponenten eines Lastverteilungsalgorithmus

6.4 Beispiele

6.5 Zuordnung bei nebenläufigen abhängigen Aufträgen

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann
Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors unzulässig

7 Prozessorvergabe in Echtzeitsystemen
7.1 Systemmodelle
7.2 Zyklische Ausführungsplan
7.3 Prozeßbasierte Zuordnung
7.4 Nutzungsbasierte Tests
7.5 Antwortzeitanalyse
7.6 Worst-case- Ausführungszeit
7.7 Sporadische und aperiodische Prozesse
7.8 Zuordnung nach *deadline monotonic priority*
7.9 Interaktion und Blockierung
7.10 Priority ceiling

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors unzulässig

Teil 3 Unity: Grundlage für Entwurf und Verifikation nebenläufiger Programme

Chandy, K. M.; Misra, J.: Parallel Program Design - A Foundation. Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.

8 Unity: Grundlage für den Entwurf nebenläufiger Programme
8.1 Grundidee, Syntax und Semantik
8.2 Basisrelationen zur Beschreibung des Programmverhaltens
8.3 Erfassung der Systemarchitektur
8.4 Anwendungsbeispiel: Verteilte Terminierungsentdeckung
8.5 Beschreibung (Spezifikation) eines Problems durch Gleichungen: Byzantinische Fehler

Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors unzulässig