

Aufgabenblatt 4

Simulation Virtueller Speicher

Speicherverwaltung / Virtueller Speicher

1 Simulation eines virtuellen Speichers

Unter https://github.com/chrlns/haw_bs_vmem befinden sich ein Java-Programm zur Simulation eines Hauptspeicherverwaltungssystems auf Basis eines "virtuellen Speichers" mit lokaler Seitenersatzungsstrategie und fester Hauptspeicherzuteilung pro Prozess.

Folgende Klassen stehen zur Verfügung:

SimulationEnv

Initialisieren des Betriebssystems inkl. Parametern für den aktuellen Simulationslauf, Start/Ende der Simulation durch Erzeugen/Beenden von beliebig vielen Prozessen und Ausgabe einer statistischen Auswertung.

Process

Simulation eines Anwendungsprogramms: Read-Operationen ausführen (pseudo-zufallsgesteuert) mit mehreren Operationen im selben Seitenbereich. Der „Lokalitätsfaktor“ bestimmt in der Simulation das Lokalitätsverhalten eines Prozesses, indem er festlegt, wie viele aufeinander folgende Zugriffsoperationen stark benachbarte Adressen verwenden, d.h. wie lange der aktuelle „Working Set“ nicht verlassen wird.

OperatingSystem

Basisfunktionen eines Betriebssystems mit System Calls

- **createProcess:** Prozess-Objekt erzeugen, in Prozesstabelle eintragen, Laden des Programmtextes in den Hauptspeicher und initialisieren der Datenbereiche (durch write-Operationen).
- **killAll:** Alle aktiven Prozesse in der Prozesstabelle beenden
- **write:** Datenwort auf eine virtuelle Adresse im virtuellen Speicher schreiben und ggf. neue Seite erzeugen (neuen Seitenrahmen anfordern, neuer Eintrag in der Seitentabelle)
- **read:** Datenwort von einer Adresse im virtuellen Speicher lesen

PageTable

Eine Seitentabelle eines Prozesses sowie eine zirkulare Liste (**pteRAMlist**) aller Seitentabelleneinträge, die sich zur Zeit im Hauptspeicher befinden.

PageTableEntry

Datenstruktur eines Seitentableneintrags

FreeListBlock

Datenstruktur eines Freibereichslisteneintrags

Statistics

Sammlung und Auswertung statistischer Daten eines Simulationslaufs

1.1 Code für Seitennummerberechnung

Ergänzen Sie den Code für die Methoden

```
private int getVirtualPageNum(int virtAdr)    und  
private int getOffset(int virtAdr)
```

in der Klasse `OperatingSystem`, die für eine übergebene virtuelle Adresse die virtuelle Seitennummer bzw. den berechneten Offset zurückgeben.

1.2 Code für Lesen

Ergänzen Sie den Code für die Methode

```
public synchronized int read(int pid, int virtAdr)
```

in der Klasse `OperatingSystem`, die für einen Prozess mit der angegebenen Prozess-ID (`pid`) ein Datenwort (`int`) von einer Adresse im virtuellen Speicher (`virtAdr`) liest und zurückgibt. Verwenden Sie dafür die vorgegebenen Methoden.

1.3 Code für Seitenersetzungsalgorithmen

Random

Der Seitenersetzungsalgorithmus **RANDOM** soll *zufällig* eine beliebige Seite des Prozesses, die sich im Hauptspeicher befindet, zum Verdrängen auswählen. Fügen Sie in der Klasse `PageTable` den entsprechenden Code hinzu:

```
private PageTableEntry randomAlgorithm (PageTableEntry newPte)
```

`randomAlgorithm` erhält als Parameter einen Zeiger auf die „neue“ Seite (als Seitentableneintrag), soll eine zu verdrängende Seite auswählen und als Ergebnis zurückgeben. Außerdem muss eine Aktualisierung der internen Liste der Hauptspeicher-Seiten vorgenommen werden.

Clock

Ergänzen Sie den Code für den Seitenersetzungsalgorithmus **CLOCK** (Uhrzeigeralgorithmus, siehe Vorlesungsfolien) in der Methode `clockAlgorithm()`.

1.4 Testen der Simulation

Testen Sie Ihren Code durch Ausführung der Klasse `SimulationEnv`.

Tipps:

- Über einen Aufruf von `setTestMode(boolean testMode)` in der Klasse `OperatingSystem` können Sie eingebaute Testausgaben aktivieren (`true`) und abschalten (`false`). Ein Aufruf ist bereits in der Main-Methode von `SimulationEnv` enthalten.
- Sie können unter Eclipse den Consolenpuffer für Ausgaben auf folgende Weise vergrößern: Window – Preferences – Run/Debug – Console – Console buffer size. Ähnliche Einstellungen sollte es auch in anderen IDEs geben.

1.5 Simulationsexperiment

Verwenden Sie das nun fertige Simulationssystem, um den Einfluss verschiedener Parameter auf die Seitenfehlerrate (Anzahl Seitenfehler / Anzahl Zugriffe auf den virtuellen Speicher) zu untersuchen. Die Parameter können als Werte in der Main-Methode von `SimulationEnv` gesetzt werden (hier mit Beispielwerten):

- Max. Anzahl Seiten pro Prozess im Hauptspeicher (sonst Verdrängung eigener Seiten)
 - `os.setMAX_RAM_PAGES_PER_PROCESS(10);`
- Seitenersetzungsalgorithmus (hier: RANDOM oder CLOCK oder FIFO)
 - `os.setREPLACEMENT_ALGORITHM(OperatingSystem.ImplementedReplacementAlgorithms.CLOCK);`
- Lokitätsfaktor zur Steuerung des Verhaltens von Programmen (Prozessen): Anzahl aufeinander folgender Operationen, bei denen sich der Zugriff auf den virtuellen Speicher nur innerhalb weniger Seiten bewegt (nur kleiner Teil des Adressraums wird genutzt).
 - `os.setDEFAULT_LOCALITY_FACTOR(100);`

Ermitteln Sie die Seitenfehlerrate für folgende Parameterkombinationen, wobei Sie die Dauer der Simulation so wählen sollten, dass mindestens 50.000 Zugriffe gezählt werden können. Die Werte für die Programmgröße (5.120 Byte, entspricht 20 Seiten) und die maschinenabhängigen Parameter sollen dabei nicht verändert werden.

MAX_RAM_PAGES_ PER_PROCESS (Hauptspeicherzuteilung)	DEFAULT_ LOCALITY_FACTOR (Lokalitätsfaktor)	Seitenfehlerrate RANDOM	Seitenfehlerrate CLOCK	Seitenfehlerrate FIFO
10	1	0.3574	0.0002	0.3595
10	10	0.2235	0.0009	0.2286
10	100	0.0708	0.00008	0.0553
10	1000	0.0105	0.00006	0.0187
15	10	0.0805	0.00004	0.0815
20	10	0	0	0

Hinweis: *Schalten Sie für die Simulationsexperimente die Testausgaben ab: `setTestMode(false)`!*

1.6 Beantworten Sie folgende Fragen

- Ist der Wert bei absolut zufälligen Zugriffsfolgen (Lokalitätsfaktor = 1) Ihrer Ansicht nach plausibel? Wenn ja, aufgrund welcher Überlegung?
Tipp: Berücksichtigen Sie die Hauptspeicherzuteilung und die Programmgröße!
- In welcher Größenordnung liegt (bei diesem einfachen Simulationsmodell) der Leistungsunterschied zwischen CLOCK-, FIFO- und RANDOM-Algorithmus (in %)?
- Welche Maßnahmen zur Leistungssteigerung Ihres Computers können Sie ergreifen, wenn Sie große Programme mit schlechtem Lokalitätsverhalten ablaufen lassen wollen? Sie können zur Beantwortung auch die Ergebnisse Ihrer Messungen nutzen.

Viel Spaß!