Übung zur Vorlesung Thread Programmierung

Alle Programmieraufgaben sind auch im GitHub-Repository
`https://github.com/lukaswerner/JavaConcurrency` **zu finden.**

1 Thread-Beispiele

1.1 Erzeugen von Threads in Java

Schreiben Sie ein Programm in Java, welches acht Threads erstellt und startet. Jeder Thread soll dabei „Hello World from Thread i“ mit i als Index des Threads im Array ausgeben. Beobachten und interpretieren Sie das Ergebnis.

Lösung: `https://gist.github.com/lukaswerner/c1a53b8b03fe4d93e1e9dfcfc50c9cb5`

1.2 Erzeugen von Threads in Java (Ohne Synchronisierung von Strings)

Erweitern Sie das Programm aus 1.1 so, dass die Ausgabe des Strings „Hello World from Thread i“ nicht synchronisiert erfolgt. Schreiben Sie dafür eine Methode, welche einen String Zeichen für Zeichen ausgibt, rufen Sie diese Methode anstelle von System.out.println() auf.

Lösung: `https://gist.github.com/lukaswerner/61a25246aca3c9fc3fa7c57d86f42af5`

1.3 Race Condition Beispiel

Schreiben Sie ein Programm in Java, welches ≥ 10000 Threads startet. Jeder Thread soll dabei eine statische Variable (z.B. z) hochzählen mit dem $++$-Operator. Geben Sie am Ende diese statische Variable aus und beobachten, sowie interpretieren das Ergebnis.

Lösung: `https://gist.github.com/lukaswerner/32e344a4c1febc69724d08a60c56064c`

1.4 Race Condition Lösung

Erweitern Sie das Programm aus 1.3 so, dass die Race Condition nicht stattfindet. Implementieren Sie hierfür eine Methode, die gesperrt wird, sobald ein Thread sie ausführt

(Stichwort: *synchronized*). Diese Methode soll dann $z{+}{+}$ ausführen.

Lösung: `https://gist.github.com/lukaswerner/12825318f45e55ec2f3f502a0a49584a`

2 Amdahls Gesetz

a) Finden Sie heraus, welcher Anteil der Zähler-Aufgabe mit $k = 10000$ Threads parallelisierbar ist.

Lösung

$a :=$ Zeit bei sequentiellem Ablauf
$b :=$ Zeit bei parallelem Ablauf

$$
\begin{aligned}
& & b &= a\left(1 - p + \frac{p}{n}\right) \\
\Leftrightarrow & & \frac{b}{a} &= 1 - p + \frac{p}{n} \\
\Leftrightarrow & & \frac{b}{a} - 1 &= \frac{p}{n} - p = p\left(\frac{1}{n} - 1\right) \\
\Leftrightarrow & & \frac{\frac{b}{a} - 1}{\frac{1}{n} - 1} &= p \\
\Leftrightarrow & & p &= \frac{n\left(\frac{b}{a} - 1\right)}{n\left(\frac{1}{n} - 1\right)} = \frac{n\left(\frac{b}{a} - 1\right)}{1 - n} = \frac{na\left(\frac{b}{a} - 1\right)}{a\left(1 - n\right)} = \frac{n\left(b - a\right)}{a\left(1 - n\right)}
\end{aligned}
$$

Testwerte:
$a = 0,0000588089s$
$b = 0,6387376938s$

$$p = \frac{n\left(b - a\right)}{a\left(1 - n\right)} = \frac{10000\left(0,6387376938s - 0,0000588089s\right)}{0,0000588089s\left(1 - 10000\right)} = -10861,3281... \approx -1086132,8\%$$

Mit Parallelisierung erheblich langsamer!

b) Welchen Anteil erwarten Sie für $k = 20000$ Threads?
Zurück gestellt...

c) Wie weit kann man die Bearbeitung durch Parallelisierung beschleunigen, wenn man beliebig viele Prozessoren zur Verfügung hat?

$$\lim_{n\to\infty} \frac{1}{1 - p + \frac{p}{n}} = \frac{1}{1 - p} = \frac{1}{1 - \frac{n(b-a)}{a(1-n)}} = ?$$

3 Verschränkung (1)

Thread p habe m Schritte, Thread q habe n Schritte auszuführen.

a) Geben Sie eine rekursive Definition für die Anzahl $\text{anz}(m,n)$ der möglichen verschränkten Abläufe von p und q an.

Lösung

$$\text{anz}(m,n) = \begin{cases} 1 & m = 0 \vee n = 0 \\ \text{anz}(m-1,n) + anz(m,n-1) & \text{sonst} \end{cases}$$

b) Finden Sie einen geschlossenen Ausdruck für $\text{anz}(m,n)$.

Lösung

Wegbeschreibung ist ein Bitvektor mit m Nullen und n Einsen, wobei hier gilt:

$$0 \mathrel{\widehat{=}} \text{Schritt nach rechts}$$
$$1 \mathrel{\widehat{=}} \text{Schritt nach unten}$$

Die Länge des Bitvektors ist $n+m$. Isomorph zum Bitvektor mit m Nullen und n Einsen sind n Teilmengen einer $n+m$ Menge.

Beispiel:
Bitvektor: 011010, $(m+n)$-Menge sei $\{1,...,m+n\}$, dargestellte Teilmente ist $\{2,3,5\}$

Satz: Die Anzahl der k Teilmengen einer n-Menge ist $\binom{n}{k}$.

Behauptung: $\text{anz}(m,n) = \binom{m+n}{n}$

Beweis.

IA Sei $n = 0 \Rightarrow \text{anz}(m,0) = 1 = \binom{m}{0}$

IV Für ein festes $k = m+n$ gilt: $\text{anz}(m,n) = \binom{m+n}{n}$.

IS Annahme: $m+n = k+1$

$$\text{anz}(m,n) = \text{anz}(m-1,n) + \text{anz}(m,n-1) \overset{\text{IV}}{=} \binom{m-1+n}{n} + \binom{m+n-1}{n-1}$$
$$= \binom{k}{n} + \binom{k}{n-1} \overset{\text{def.}}{=} \binom{k+1}{n}$$

□

c) Schätzen Sie die Größenordnung von $\text{anz}(m, n)$.
(Korrektur):
Behauptung: $\text{anz}(n, n) \geq 2^n - 1$

Beweis.

$$1.\ n = 1 : \text{anz}(n, n) = \binom{2}{1} = 2$$

$$2.\ n > 1 : \text{anz}(n, n) = \binom{2n}{1} = \binom{2n-1}{n} + \binom{2n-1}{n-1}$$

$$2^n = 2 * 2^{n-1} \overset{\text{IV}}{\leq} 2 * \binom{2(n-2)}{n-1} \leq \binom{2n-1}{n} + \binom{2n-1}{n-1}$$

$$\text{weil } \binom{2n-1}{n-1} \geq \binom{2n-2}{n-1} \text{ und } \binom{2n-1}{n} \geq \binom{2n-2}{n-1}$$

$$\binom{2n-1}{n-1} = \binom{2n-2}{n-1} + \binom{2n-2}{n-2}$$

□

4 Verschränkung (2)

n Threads sollen verschränkt zueinander laufen. Jeder Thread habe 2 Schritte auszuführen.

a) Geben Sie eine rekursive Definition an für die Anzahl $\text{anz}^*(n)$ der möglochen Abläufe.

Lösung

Es gilt $\text{anz}^*(1) = 1$. Sei nun $n > 1$.
Nach IV liefert $\text{anz}^*(n-1)$ die Anzahl der möglichen Abläufe für $n-1$ Threads.

Also gilt:

$$\text{anz}^*(n) = \begin{cases} 1 & \text{falls } n = 1 \\ \text{anz}(n-1)(2n-1)(n-1) & \text{sonst} \end{cases}$$

b) Finden Sie einen geschlossenen Ausdruck für $\text{anz}^*(n)$.

c) Schätzen Sie die Größenordnung von $\text{anz}^*(n)$. b) sowie c) kann auch Prof. Geser nicht lösen, wächst aber in etwa im Quadrat der Fakultät

5 zeitliche Abläufe

a) Geben Sie zu dem folgenden zeitlichen Ablauf den passenden seriellen Ablauf (= Aktionsfolge) an: $\{(a, 5.3), (b, 3.7), (c, 3.2), (a, 1.7), (a, 2.1), (b, 2.5)\}$

Lösung

$$f(x) = \begin{cases} a & \text{falls } x \in 1, 2, 6 \\ b & \text{falls } x \in 3, 5 \\ c & \text{sonst} \end{cases}$$

b) Welche der folgenden Mengen von Ereignissen sind diskrete zeitliche Abläufe? Dazu sei $A = \{a, b\}$.

Lösung

c) Geben Sie zu jedem diskreten zeitlichen Ablauf E aus Teilaufgabe (b) die Ereignisse E^1, E^2, E^3 an.

Lösung

6 Einigkeit

Beweisen Sie, dass für alle Threads p_1, p_2 folgende Aussagen äquivalent sind:

$$\alpha)\ \pi_{A_1 \cap A_2}(E_1 \cup E_2) = E_1 \cap E_2$$
$$\beta)\ \pi_{A_1 \cap A_2}(E_1 \oplus E_2) = \emptyset$$
$$\gamma)\ \pi_{A_1}(E_2) = \pi_{A_2}(E_1)$$

Dabei sei A_i das Prozessalphabet von Thread p_i und E_i der zeitliche Ablauf von Thread $p_i, i \in \{1, 2\}$.
Voraussetzung: Für alle $e \in E_i$ gilt $aktion(e) \in A_i, i \in \{1, 2\}$.
Dann gilt: $\pi_{A_i}(E_i) = E_i$

Lösung

zz. $\beta \Rightarrow \alpha$

Beweis.

$$
\begin{aligned}
&\pi_{A_1 \cap A_2}(E_1 \cup E_2) = \\
&\pi_{A_1 \cap A_2}((E_1 \oplus E_2) \cup (E_1 \cap E_2)) = \\
&\pi_{A_1 \cap A_2}(E_1 \oplus E_2) \cup \pi_{A_1 \cap A_2}(E_1 \cap E_2) = \\
&\emptyset \cup \pi_{A_1 \cap A_2}(E_1 \cap E_2) = \\
&\pi_{A_1}(\pi_{A_2}(E_1 \cap E_2)) = \\
&\pi_{A_1}(E_1 \cap E_2) = E_1 \cap E_2
\end{aligned}
$$

□

zz. $\alpha \Rightarrow \beta$

Beweis.

$$
\begin{aligned}
&\pi_{A_1 \cap A_2}(E_1 \oplus E_2) = \\
&\pi_{A_1 \cap A_2}((E_1 \cup E_2) \setminus (E_1 \cap E_2)) = \\
&\pi_{A_1 \cap A_2}(E_1 \cup E_2) \setminus \pi_{A_1 \cap A_2}(E_1 \cap E_2) \stackrel{\alpha}{=} \\
&(E_1 \cap E_2) \setminus (E_1 \cap E_2) = \emptyset
\end{aligned}
$$

□

zz. $\beta \Rightarrow \gamma$

Beweis.

$$
\begin{aligned}
&(1)\pi_{A_1}(E_2) \setminus \pi_{A_2}(E_1) = \emptyset \iff \pi_{A_1}(E_2) \subseteq \pi_{A_2}(E_1) \\
&(2)\pi_{A_2}(E_1) \setminus \pi_{A_1}(E_2) = \emptyset \iff \pi_{A_2}(E_1) \subseteq \pi_{A_1}(E_2)
\end{aligned}
$$

$$
\begin{aligned}
&(1) \wedge (2) \\
&\iff \pi_{A_1}(E_2) \setminus \pi_{A_2}(E_1) \cup \pi_{A_2}(E_1) \setminus \pi_{A_1}(E_2) = \emptyset \\
&\iff \pi_{A_2}(E_2) \oplus \pi_{A_2}(E_1) = \emptyset
\end{aligned}
$$

$$
\begin{aligned}
&zz.\pi_{A_1 \cap A_2}(E_1 \oplus E_2) = \pi_{A_1}(E_2) \oplus \pi_{A_2}(E_1) \\
&\pi_{A_1 \cap A_2}(E_1 \oplus E_2) = \pi_{A_1}(\pi_{A_2}(E_1 \oplus E_2)) = \\
&\pi_{A_1}(\pi_{A_2}(E_1) \oplus E_2) = \pi_{A_1}(E_2) \oplus \pi_{A_1}(\pi_{A_2}(E_1)) = \\
&\pi_{A_2}(E_1) \oplus \pi_{A_1}(\pi_{A_1}(E_1)) = \\
&\pi_{A_2}(E_1) \oplus \pi_{A_1}(E_2)
\end{aligned}
$$

□

zz. $\gamma \Rightarrow \beta$

Beweis.

$$\pi_{A_1 \cap A_2}(E_1 \oplus E_2) = \pi_{A_1}(\pi_{A_2}(E_1 \oplus E_2)) =$$
$$\pi_{A_1}(\pi_{A_2}(E_1) \oplus E_2) = \pi_{A_1}(\pi_{A_1}(E_2) \oplus E_2) \stackrel{\gamma}{=}$$
$$\pi_{A_1}(E_2) \oplus \pi_{A_1}(E_2) = \emptyset$$

□

6.1 Aufgabe:

a) Beweisen Sie, dass „→“ eine Wohlordnung (irreflexiv, transitiv, total, fundiert) ist.

6.2 Lösung

→ **irreflexiv** Beweis durch Wiederspruch
Annahme: e → e gilt. Nach Definition von „→“ gilt zeit(e) < zeit(e).
Widerspruch zu < irreflexiv.

→ **transitiv** a → b und b → c, dann a → c
zeit(a) < zeit(b) und zeit(b) < zeit(c)
⇒ zeit(a) < zeit(b) < zeit(c) ⟺ zeit(a) < zeit(c)
⇒ a → c, wegen Transitivität <

→ **total** Für Ereignisse $e_1 \neq e_2$ gilt $e_1 \rightarrow e_2$ oder $e_2 \rightarrow e_1$
Seien $e_1, e_2; e_1 \neq e_2$ Ereignisse
Dann zeit(e_1) < zeit(e_2) oder zeit(e_2) < zeit(e_1) oder zeit(e_1) = zeit(e_2)
„<“ ist total.
Wenn zeit(e_1) = zeit(e_2), dann $e_1 = e_2$ nach Vorraussetzung (idealisierende Annahme 3).
Widerspruch zu $e_1 \neq e_2$. Also entweder zeit(e_1) < zeit(e_2) damit $e_1 \rightarrow e_2$ oder zeit(e_2) < zeit(e_1) damit $e_2 \rightarrow e_1$.

→ **fundiert** zz. gibt keine unendliche „←“-Kette. Beweis durch Widerspruch.
Annahme: $(e_i)_{i \in \mathbb{N}}$ sei unendliche „←“-Kette.
Damit $e_i \leftarrow e_{i+1}$ für alle $i \in \mathbb{N}$.
Damit zeit(e_i) > zeit(e_{i+1}) für alle $i \in \mathbb{N}$ wegen Definition von →.
Damit ist $(zeit(e_i))_{i \in \mathbb{N}}$ eine unendliche „←“-Kette.
Widerspruch zu: < ist fundiert auf $\mathbb{N}$.