
Concurrent Programming

Blatt 4

Abgabe: 27.5.2002

1. [10 Punkte] Die Programmiersprache ID-90 bietet sogenannte I- und M-Strukturen zur Synchronisation zwischen Threads an. Bei einer solchen Struktur handelt es sich um einen Platzhalter für einen Wert, auf den synchronisiert mit `put`- und `get`-Operationen zugegriffen wird.

Beide Datenstrukturen haben Gemeinsamkeiten: Die `put`-Operation füllt die Speicherzelle mit einem bestimmten Wert. Ist die Speicherzelle bereits gefüllt, signalisiert `put` einen Fehler. Die `get`-Operation blockiert solange, bis die Speicherzelle mit einem Wert gefüllt ist

Die `get`-Operationen für I- und M-Strukturen unterscheiden sich in einer Eigenschaft: In einer M-Strukturen leert `get` den Inhalt des Speichers (dieser kann also wieder mit `put` neu gefüllt werden) während die `get`-Operation der I-Strukture den Inhalt der Speicherzelle nicht verändert.

Programmiere die beiden Datenstrukturen mit den dazugehörigen Operationen in Scheme!

2. [20 Punkte] Der Sortieralgorithmus `merge-sort` ist ein Divide-and-Conquer-Algorithmus und kann auf abstraktem Niveau so beschrieben werden:

```
(define (merge-sort sequence)
  (if (= (length sequence) 1)
      sequence
      (let ((two-sequences (split sequence)))
        (merge (merge-sort (car two-sequences))
               (merge-sort (cdr two-sequences))))))
```

Schreibe eine Version von `merge-sort`, die für jeden rekursiven Aufruf einen neuen Thread startet. Die Threads sollen untereinander mit synchronem Message-Passing kommunizieren.

`Merge-sort` soll keine Argumente akzeptieren und einen Kanal zurückliefern. Auf diesem muß der Klient von `merge-sort` zunächst eine Folge der zu sortierenden Zahlen senden, terminiert durch ein abschließendes `#f`. Alsdann kann der Klient auf demselben Kanal die sortierte Folge abholen, wieder terminiert durch `#f`.

- (a) [2 Punkte] Behindert die Verwendung ein- und desselben Kanals für Ein- und Ausgabe irgendwie die Eleganz der Programmierung des Problems?
- (b) [2 Punkte] Zeichne ein Prozeßnetzwerk, das illustriert, wie `merge-sort` arbeitet.
- (c) [16 Punkte] Programmiere `merge-sort`!

Anleitung:

- [2 Punkte] Schreibe eine Prozedur `split`, die als Argumente drei Kanäle akzeptiert. `split` soll den ersten Kanal bis zum ersten `#f` auslesen und die Werte gleichmäßig verteilt auf dem zweiten und dritten Kanal senden.

- [8 Punkte] Schreibe eine Prozedur `merge`, die als Argumente drei Kanäle akzeptiert: zwei Eingabe-Kanäle und einen Ausgabekanal. Die Eingabekanäle produzieren jeweils eine sortierte Zahlenfolge, terminiert durch `#f`. `Merge` soll nun die Zahlen aus den beiden Eingabekanälen sortiert auf den Ausgabekanal schicken.
 - [6 Punkte] Schreibe mit Hilfe von `split` und `merge` eine Prozedur `merge-sort`, die für jeden rekursiven Aufruf Threads startet.
3. [30 Punkte] Der k -te Koeffizient des Produktes zweier Potenzreihen F und G ist durch

$$P_k = \sum_{i=0}^k (F_i \cdot G_{k-i})$$

gegeben. Für die Berechnung des k -ten Koeffizienten müssen also die Koeffizienten kleiner k bekannt sein. Für eine Implementierung als Prozeßnetzwerk ist diese Darstellung des Produktes daher ungeeignet. Das ändert sich, wenn Potenzreihen in einer Form mit Restglied ($F = F_0 + x\bar{F}$) dargestellt werden:

$$P = P_0 + x\bar{P} = F_0G_0 + x(G_0\bar{F} + F_0\bar{G}) + x^2(\bar{F} \cdot \bar{G})$$

Daraus ergibt sich in allgemeiner Form:

$$P_0 = F_0G_0$$

$$\bar{P} = G_0\bar{F} + F_0\bar{G} + x(\bar{F} \cdot \bar{G})$$

Programmiere eine Version der Potenzreihenmultiplikation, bei der die Potenzreihen als synchrone Kanäle repräsentiert sind, welche die Koeffizienten liefern.

- (a) Schreibe zuerst einige Prozeduren für einfache Operationen auf Potenzreihen:
- [3 Punkte] Schreibe eine Prozedur `power-series-add`, die zwei Potenzreihen addiert.
 - [2 Punkt] Schreibe eine Prozedur `power-series-multiply-by-constant`, die eine Potenzreihe mit einer Konstante multipliziert.
 - [3 Punkte] Schreibe eine Prozedur, die eine Potenzreihe mit ihrer Variable x multipliziert, also

$$xF = \sum_{i=0}^{\infty} F_i x^{i+1}$$

berechnet.

- (b) [8 Punkte] Aus der Multiplikationsformel ergibt sich, daß Werte aus den Kanälen immer doppelt gebraucht werden: Die Koeffizienten von \bar{F} werden mit G_0 multipliziert und werden auch für die Berechnung von $\bar{F} \cdot \bar{G}$ gebraucht. Wie kann dieses Problem einfach gelöst werden?
- (c) [6 Punkte] Fertige eine Skizze des Prozeßnetzwerks für die Potenzreihenmultiplikation an. Diese Skizze soll zeigen, wie die einzelnen Operationen für Potenzreihen durch Kanäle verbunden sind.
- (d) [8 Punkte] Implementiere eine Multiplikation von Potenzreihen nach dem oben angegebenen Verfahren!