Christian-Albrechts-Universität zu Kiel
Institut für Informatik und Praktische Mathematik
Prof. Dr. W.-P. de Roever
Harald Fecher und Heiko Schmidt

Nebenläufige Programmierung

Wintersemester 2004/05 — **Serie 6** — 22. November 2004

Thema: Gegenseitiger Ausschluss

Ausgabetermin: 22. November 2004

Abgabe: 6. Dezember 2004 (12:00)

Am 1.12.04 findet keine Übung statt (man hat 2 Wochen Bearbeitungszeit).

Aufgabe 1 (4 Punkte) Exercise 3.2 aus Andrews auf S. 142.

Aufgabe 2 (6 Punkte)

1. Zeige dass die `number` Variable im Ticket Algorithmus (S. 108ff) unbeschränkt hohe Werte annehmen kann.
2. Modifiziere den "coarse-grained" Algorithmus (Figure 3.8) derart, dass das Wachstum von `number` (und `next`) durch eine Konstante beschränkt ist und der Algorithmus korrekt bleibt. Erläutere den Lösungsansatz und die Korrektheit der Lösung.
3. Modifiziere den "fine-grained" Algorithmus (Figure 3.9) so, dass `number` und `next` beschränkt sind und der Algorithmus korrekt bleibt. Die Fetch-and-Add Intruktion ist hier benutzbar. Erläutere Lösungsansatz und Korrektheit.

Aufgabe 3 (10 Punkte)

1. Zeige, dass die `turn[i]` Variable im Bakery Algorithmus (S. 111ff) unbeschränkt hohe Werte annehmen kann.
2. Modifiziere den Bakery Algorithmus (coarse grained, Fig. 3.10 auf S. 112) derart, dass die `turn[i]` Werte begrenzt werden.
   Hinweis: Die atomare Anweisung welche `turn[i]` setzt, darf nicht geändert werden, das await-Statement wohl. Neue Variablen, z.B. ein `next`, dürfen eingeführt werden.
3. Modifiziere ebenfalls die fine-grained Variante von Fig. 3.11 auf S. 114.
   Hinweis: Es darf hier kein Fetch-and-Add verwendet werden. Die Aufgabe ist schwer. Man kann versuchen, beim Überschreiten der Grenze wieder von vorne anzufangen. Wähle hierfür eine passende der Grenze in Abhängigkeit von der Anzahl der Prozesse. Mache dann eine Fallunterscheidung darüber, in welchem Bereich die `turn` Werte der Prozesse liegen können. Und zuletzt eleminiere weitere Fehlverhalten.