

Übungen zu „Parallelität in funktionalen Sprachen“

Nr. 3, Abgabe: 12. November in der Vorlesung

Abgabe: Die Lösungen sollten grundsätzlich schriftlich, Programme zusätzlich auf Diskette oder per E-Mail an eden@mathematik.uni-marburg.de abgegeben werden.

Die Abgabe ist in Gruppen bis zu zwei Personen erlaubt.

Parallelisierung mit `par` und `seq`

1. Parallelisieren und testen Sie das Programm `Wortsuche.hs` aus der ersten Übung.

12 Punkte

- (a) Erstellen Sie zwei unterschiedliche parallele Varianten von `Wortsuche.hs`. Kommentieren Sie die verwendete Parallelisierung hinsichtlich ihrer Granularität.
- (b) Testen Sie Ihre parallelen Programme mit `GranSim`. Lassen Sie dazu jede Version mit 4 und 8 Prozessoren laufen (legen Sie die erzeugten Aktivitätsprofile bei).

2. Inhärente Parallelität im Springerproblem

8 Punkte

- (a) Analysieren Sie Ihr Programm `Springerproblem.hs`¹ auf inhärente Parallelität.
- (b) Parallelisieren Sie das sequenzielle Programm nach Ihren Überlegungen. Erklären Sie, wie Sie die “Größe” der parallelen Teilaufgaben kontrollieren.

Hinweis: Auch mit `GpH` lassen sich Aktivitätsprofile wie in `GranSim` erstellen. Dazu dient ähnlich wie bei `GranSim` die `RTS`-Option `-qP`, mit der `*.gr`-Dateien erzeugt werden. Die erzeugten Dateien (eine je Prozessor) müssen vor der Bearbeitung mit `gr2ps` noch mit dem Script `grs2gr` zusammengeführt werden:

```
berthold@windhoek:parfp/bsp> parFac1 5000 +RTS -qp8 -qP > out
==== Starting parallel execution on 8 processors ...
berthold@windhoek:parfp/bsp>
berthold@windhoek:parfp/bsp> ls *.gr
=parFac1.001.gr =parFac1.003.gr =parFac1.005.gr =parFac1.007.gr
=parFac1.002.gr =parFac1.004.gr =parFac1.006.gr =parFac1.008.gr
berthold@windhoek:parfp/bsp>
berthold@windhoek:parfp/bsp> grs2gr =parFac1.???.gr > grfile.gr
berthold@windhoek:parfp/bsp>
berthold@windhoek:parfp/bsp> gr2ps grfile.gr
```

¹Sie können auch eine Musterlösung von der Vorlesungsseite benutzen.