Eerste deeltoets Concurrency

3 oktober 2014, 8.30 – 10.30, Educ-α.

Motiveer je antwoorden *kort* en schrijf *netjes*! Maak op pagina 1 de vragen 1 en 2, op pagina 2 de vragen 3 en 4, op pagina 3 de vragen 5 en 6. Gebruik het achterblad als je ergens ruimte te kort komt. Vraag 1 is 4pt, vraag 2 is 2pt, de rest elk 3pt. DT1 is totaal plus 1 gedeeld door 1,7.

1. **Terminatie van Threads:** Twee threads delen variabele s, die door thread 1 steeds wordt opgehoogd. Thread 0 leest s en termineert beide threads als s *even* is:

```
Thread0                  Thread1
  s = 0; t = 0;            b = True;
  while (s%2 == 1)         while (b)
    { t++ }                  { s++ }
  b = False;               print(t);
```

   (a) Wanneer is een methode wait-free?
   (b) Is het programma van Thread0 waitfree? Is dat van Thread1 wait-free?
   (c) Is het programma terminerend onder de aannamen van *read-write atomicity* en *fairness*?

2. **Carré:** Je trekt vijf kaarten uit een goed geschud pak van 52. Hoe groot is de kans op een *carré*? (Bij een carré heb je *allevier* de tweeën, of drieën, etc.)

3. **Gloeilampen:** Doos A bevat 80 lampen; het is bekend dat er 20 kapot zijn. Doos B bevat ook 80 lampen; elke lamp heeft een kans van $\frac{1}{4}$ om kapot te zijn. Uit beide dozen pakken we 12 lampen.
   (a) Wat is de kans dat er van de lampen uit doos A, precies 3 kapot zijn?
   (b) Wat is de kans dat er van de lampen uit doos B, precies 3 kapot zijn?
   (c) Wat is het verwachte aantal kapotte lampen in de trekking uit doos A?
   (d) Wat is het verwachte aantal kapotte lampen in de trekking uit doos B?

4. **Bakery labels:** In het Bakery algoritme kiest een thread i een label:

```
l = 0;
for (j=0; j<n; j++) if (label[j] > l) l = label[j];
label[i] = l+1
```

   (a) Geef een voorbeeld waarin thread *a eerder klaar is* met kiezen dan thread b, maar toch een groter label krijgt.
   (b) Is het een probleem dat twee threads een *gelijk* label kunnen krijgen? Licht toe!

5. **Atomaire en Regular Registers:** Deze vraag gaat over Single Reader, Single Writer Registers.
   (a) Geef een voorbeeld van gedrag dat *wel* Regular is, maar *niet* Atomic.
   (b) Hoe kun je met een Regular register een Atomic register maken?

6. **Ticket met Maxer:** In haar softwareproject heeft Anneke behoefte aan een *wachtvrij*, atomair *ticket* object (door meerdere threads te gebruiken). Het heeft methode `rinc` (read-and-increment) die bij meerdere aanroepen, opeenvolgende getallen teruggeeft.
   In haar vorige project heeft Anneke al een wachtvrij *maxer* object gebouwd, met methoden `update(x)` en `curmax`. De maxer houdt per thread de laatst met `update` ingevoerde `x` bij, en de methode `curmax` geeft atomair de hoogste huidige waarde.
   Kan het Ticket object met behulp van een Maxer (en registers) worden gebouwd? Leg uit!