Tweede Toets Concurrency

28 januari 2016, 8.30 – 10.30, Educ-β.

Motiveer je antwoorden *kort*! Zet je mobiel uit. Stel geen vragen over deze toets; als je een vraag niet duidelijk vindt, schrijf dan op hoe je de vraag interpreteert en beantwoord de vraag zoals je hem begrijpt.
Cijfer: Maak vragen 1 en 2 op pagina 1, 3 en 4 op pagina 2, en 5 en 6 op pagina 3. Vragen 1, 4 en 6 zijn 3pt, vragen 2, 3 en 5 zijn 2pt. Cijfer T2 is totaal plus 0,15, gedeeld door 1,5.

1. **Permutaties en Combinaties:** (a) Schrijf $P(n,\, k)$ als quotient van twee faculteiten en $C(n,\, k)$ als breuk met drie faculteiten.
(b) Bewijs dat voor elke positieve n en $0 \leq k \leq n$ geldt: $C(n,\, k) \leq 2^n$.
(c) Hoeveel is $\sum_{k=0}^{n}[2^k + (\frac{1}{2})^k] \cdot C(n,\, k)$?

   Oplossing: (a) $P(n,\, k) = \frac{n!}{(n-k)!}$ en $C(n,\, k) = \frac{n!}{k!(n-k)!}$.
(b) Dit kan met inductie: Voor $n = 1$ geldt $C(1,\, 0) = C(1,\, 1) = 1$, dus beide kleiner dan $2^1 = 2$. Voor $n > 1$ geldt $C(n,\, k) = C(n-1,\, k) + C(n-1,\, k-1)$. De InductieHypothese stelt dat beide termen kleinergelijk 2^{n-1} zijn, hun som is dus kleinergelijk 2^n.
Het kan ook simpeler met het Binomium: omdat $\sum_k C(n,\, k) = 2^n$, en geen van deze termen is negatief (dat is ook weer te bewijzen, maar mocht je hier voor zoete koek aannemen), is elke term op zich begrensd door 2^n.
(c)

$$\begin{array}{lll} & \sum_{k=0}^{n}[2^k + (\frac{1}{2})^k] \cdot C(n,\, k) & \\ = & \sum_{k=0}^{n}[2^k] \cdot C(n,\, k) + \sum_{k=0}^{n}[(\frac{1}{2})^k] \cdot C(n,\, k) & \text{Termsplitsing} \\ = & 3^n + (1\frac{1}{2})^n & \text{want } \sum_k C(n,\, k)a^k = (a+1)^n \end{array}$$

   Dit laatste kun je niet verder vereenvoudigen.

   Beoordeling: Tot 3pt, eentje per deelvraag. Codes:

   B = Alleen de basis van de inductie is goed, 0pt.
C = Iets bewijzen is **niet**: Conclusies trekken uit je bewijsdoel! Bv zo: $C(n,\, k) \leq 2^n$, pijltje, $n!/k!(n-k)! \leq 2^n$, pijltje, $n! \leq 2^n k!(n-k)!$, pijltje, en dan doorpruttelen tot je op $1 \leq 1$ bent aangekomen. Dit is geen bewijs! In ieder geval *niet* van de stelling $C(n,\, k) \leq 2^n$. Met enige fantasie kun je het zien als een bewijs van de (niet interessante) stelling $1 \leq 1$, maar dan een fout bewijs want gebaseerd op een onbewezen aanname! Conclusies trekken is namelijk niet omkeerbaar. Als je uit A, B kan concluderen, kun je nog niet zeggen dat A uit B volgt. Een bewijs begint met bekende of bewezen dingen, en trekt daaruit (gerechtvaardigde) conclusies, totdat het bewijsdoel (de beoogde stelling) is geconcludeerd.
I = Je probeert Inductie maar met de multiplicatieve definitie van C, dat werkt niet.
K = Voor P moet je delen door $(n-k)!$, niet door $k!$.
M = De som van Machten $2^k + (\frac{1}{2})^k$ is *niet* hetzelfde als $(2\frac{1}{2})^k$.
M = $C(n,\, k)$ is niet Monotoon in k dus (b) volgt niet uit $C(n,\, 0) \leq 2^n$ en $C(n,\, n) \leq 2^n$.
Q = Inderdaad, een Quotient is hetzelfde als een breuk.
V = Te Vaag Verhaal om bewijs te heten.

2. **Auwliebol:** Een groep van 20 studenten, waarvan 6 vrouwen (en dus 14 mannen), eten oliebollen. Helaas zijn 8 oliebollen bedorven, waardoor 8 random studenten ziek worden. Wat is de kans dat er precies 2 vrouwen (en dus 6 mannen) ziek worden?

   Oplossing: Neem als elementaire gebeurtenis een combinatie van 8-uit-20 studenten.
   Het aantal elementaire gebeurtenissen is $C(20,\ 8) = 125970$.
   Twee vrouwen kun je op $C(6,\ 2) = 15$ manieren kiezen en zes mannen kun je op $C(14,\ 6) = 3003$ manieren kiezen, dus het aantal elementaire gebeurtenissen dat aan de vraag voldoet is 15 keer 3003 ofwel 45045.
   De kans is $\frac{\#E}{\#S} = \frac{45045}{12597} = 0,3576$ dus 36%.

   Beoordeling: Voor een goed antwoord 2pt.

   A = Het aantal zieke mannen en vrouwen is zeker niet onAfhankelijk.
   B = De Breuk $\frac{231}{646}$ wordt ook goedgerekend, de breuk $\frac{45045}{125970}$ is als eindantwoord onvoldoende vereenvoudigd, min $\frac{195}{390}$pt.
   E = Hoeveel oliebollen Eet iedereen? Je hoefde eigenlijk geen extra aannamen te maken, omdat de vraag heel duidelijk stelt dat er acht studenten ziek worden.
   P = In de teller van de breuk een plus ipv Maal, 1pt.

3. **Letters:** Leo heeft een zak met letterballen: het zijn er 26, een voor elk van de letters `A` t/m `Z`. Leo trekt steeds willekeurig een bal, bekijkt hem en legt hem terug.
   (a) Hoevaak moet Leo verwacht trekken totdat hij alle letters gezien heeft?
   (b) Hoevaak moet Leo verwacht trekken totdat hij 15 verschillende letters gezien heeft?

   Oplossing: (a) Volgens de theorie van de Coupon Collector is dit $26 * H_{26}$. Dit H-getal is 3,8544 dus er komt 100,21 uit, maar je mag de H ook benaderen met $\ln 26$ met antwoord 84,71.
   (b) Volgens de theorie van de Coupon Collector is dit $26 * (H_{26} - H_{11})$ omdat Leo nu de duurste 11 letters niet af hoeft te wachten. Hier komt 21,70 uit, maar als je het verschil van de H getallen benadert met $\ln(\frac{26}{11})$ krijg je 22,37 wat ook goed is vandaag.

   Beoordeling: Tot 2pt, eentje per deelvraag.

   A = Je moet de verwachting niet Afronden, de verwachting hoeft niet geheel te zijn. Halve strafpunt!
   H = Tot mijn verrassing waren er meer met de exacte waarde van H_n dan met de benadering $\lg n$. Zit het harmonisch getal soms in de TI84 zonder dat ik het weet, hebben jullie daar een programmaatje voor geschreven, schrijf je zo'n optellusje ter plekke, of doen jullie dat met heel veel knopjes indrukken?
   R = De Range 11–26 heeft 16 getallen. Voor 15 getallen moet je sommeren vanaf 12, min 1/2pt.

4. **Collectives:** Onderstaande code is een voorbeeld van een stencil operatie. Beschrijf kort drie mogelijke bronnen van inefficiency, en stel voor elk daarvan een verbetering voor:

```
for(int x=0; x<8192; x++)
  for(int y=0;y<8192; y++)
    set(x,y,(get(x-1,y)+get(x+1,y)+get(x,y-1)+get(x,y+1))/4);
```

Hierbij implementeert `get` toegang tot array A en `set` toegang tot array B. Beide arrays hebben een grootte van 8192x8192 floats. Method `get` levert 0 voor coordinaten buiten array A.

Oplossing: Probleem 1: de array wordt kolomsgewijs benaderd. De data wordt dus niet sequentieel gelezen; dit is ongunstig voor de cache. Dit kan opgelost worden door de x en y loop te verwisselen.
Probleem 2: de get method bevat conditional code om te voorkomen dat er buiten de array gelezen wordt. Dit is alleen nodig voor de cellen langs de rand; opsplitsen van de code voorkomt dat de condities voor alle cellen geevalueerd moeten worden.
Probleem 3: de array is zo groot dat bij de verwerking van een volgende kolom of rij waardes van de vorige wellicht al uit de cache zijn verdwenen. Data locality kan verbeterd worden met tiling of een Morton curve.

Beoordeling: Voor elke bron en oplossing een punt. Ook als geldige bron van inefficiency meegenomen: seriële uitvoering, met als oplossing parallelle executie of vectorizatie (maximaal één van deze opties). Geen punten voor het gebruik van ints i.p.v. floats (dit verandert de functionaliteit of precisie), het vervangen van de deling door een vermenigvuldiging met 0.25, en het voorstel om het lezen van de vier buren te herordenen.

5. **GPGPU:** Op een GPU kan conditional code leiden tot een verlaging van occupancy.
(a) Wat is occupancy in dit verband?
(b) Hoe kan conditional code leiden tot een verlaging van occupancy?

Oplossing: (a) Occupancy is het aantal actieve threads in een warp gedeeld door het maximale aantal threads in een warp (gewoonlijk 32). Als dit minder dan 1 is zijn een aantal threads inactief totdat de conditionele code is afgerond.
(b)Bij conditionele code worden sommige executiepaden niet door alle threads doorlopen. Deze worden dan tijdelijk op non-actief gezet.

Beoordeling: Voor antwoord a en b elk een punt.
Deel (a): Een accuraat antwoord is hier van belang. Occupancy is niet "het aantal actieve threads", of "de mate waarin de compute capaciteit benut wordt", of "het aantal threads dat niet gemasked wordt" maar echt "aantal actieve threads (in een warp) gedeeld door het totaal aantal threads (=32)". Occupancy kan ook iets anders opgevat worden, namelijk "het percentage CUDA cores / workitems (AMD) dat actief is" (globaal, dus device-breed). Ook goed is daarom eenvoudigweg "het percentage actieve threads / cores / workitems".
Deel (b): Niet correct: sommige threads doen meer werk dan andere (dit suggereert dat threads niet dezelfde code uitvoeren, wat juist wel zo is).

6. **Terminologie:** Wat is het verschil tussen:
   (a) Scan en reduction?
   (b) Inclusive en exclusive scan?
   (c) Stencil en recurrence?

   Oplossing: (a) Reduction levert 1 waarde op door het herhaaldelijk toepassen van een binaire operator; scan levert als resultaat ook de deelreducties voor ieder input element.
   (b) Bij de inclusive scan worden elementen 0..N meegenomen voor resultaat N, bij de exlusive scan alleen 0..N-1.
   (c) Een stencil gebruikt alleen de input data voor bewerkingen; een recurrence gebruikt daarnaast resultaten van bewerkingen.

   Beoordeling: Voor antwoord a, b en c elk een punt.
   Deel (a): Meestal wel goed beantwoord.
   Deel (b): Niet correct: het eerste element wordt niet meegenomen (dat wordt het wel, het heeft alleen pas effect op het tweede output element). Wel goed: het laatste element wordt niet meegenomen.
   Deel (c): Niet correct: stencil kijkt naar elementen in een lokaal gebied, recurrence neemt alles tot de linkerbovenhoek mee (of iets van die strekking). Dit laatste is misschien technisch wel waar, maar de kern van een efficiënte recurrence is toch echt de data dependency tussen elementen, dus het gebruik van de resultaten van eerdere bewerkingen.