Oplossingen examen Discrete Wiskunde

Eerste zittijd 2011-2012

31 januari 2011

Oefening 1. (a) Bewijs dat er getallen $a, b < 512$ zijn met $2^a \equiv 2^b \pmod{2012}$.

(b) Schrijf een algoritme op om dergelijke a, b te vinden, op een machine die alle standaard rekenkundige en logische bewerkingen kan, maar slechts 1 KiB (=1024 bytes) werkgeheugen[1] heeft. Doe dit in zo weinig mogelijk rekentijd.

Oplossing. (a) De oplossingstechniek van Oefening 73c overschrijven en 2010 vervangen door 2012, levert direct het bestaan voor dergelijke a, b, maar dan zonder de voorwaarde dat $a, b < 512$ (in de plaats $a, b \leq 2012$). Dit opmerken leverde al een deel van de punten op. Er zijn verschillende manieren om van hier verder te gaan. De eenvoudigste is om te zien dat $\text{ggd}(2012, 2^a) = 4$ voor $a \geq 2$, dus dat dit feitelijk neerkomt op $a, b < 510$ vinden met $2^a \equiv 2^b \pmod{503}$. En dat lukt wel met de techniek van Oefening 73c.

(b) Wie de theorie goed beheerste, kon meteen zeggen dat $2^{\phi(503)} \equiv 1 \pmod{503}$. Dus bijvoorbeeld $a = 2$ en $b = 2 + \phi(503) = 504$ zijn goede voorbeelden. Voor de minder theoretische geesten was een efficiënt algoritme ook toegelaten.

De naiëve manier om dit te doen (hoe je het wellicht zou doen vóór je de cursus discrete wiskunde gevolgd hebt) is als volgt: `for (i=0,...,512) for (j=0,...,512) if (2^a%2012==2^b%2012) return a,b;` maar dat is uiteraard absoluut niet efficient. De eenvoudigste 'grote' verbetering is de truc van Oefening 77 toepassen: niet alle koppels vergelijken, maar bijhouden welke restklassen al opgetreden zijn. Idealiter zonder elke keer de machten te herberekenen, maar hun resten modulo 2012 op te slaan en effenaan verdubbelen. Dit staat in Java-stijl uitwerkt hieronder. Maar uiteraard zijn er nog tal van andere algoritmes mogelijk die beter presteren dan de inefficiënte dubbele lus hierboven. □

```
mem=new short[503];
for (a=2, x=1; a<=504; a++) {
  if (mem[x]==0) { mem[x]=a; x=(2*x)%503; }
  else return (a,mem[x]);
}
```

[1] Het algoritme zelf is niet beperkt in grootte, maar je moet het wel volledig opschrijven. Je hoeft het werkgeheugen voor ons zelfs niet te gebruiken, als je direct via theoretische overwegingen twee zo'n getallen kan vinden is dat natuurlijk nog beter (en dat is zeker mogelijk met wat je gezien hebt). Maar een goed (optimaal geschreven) algoritme zal ook alle punten opleveren.

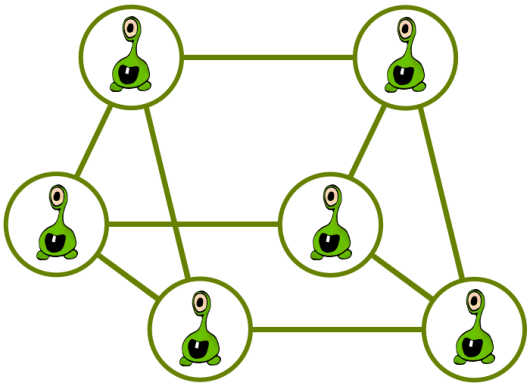
Oefening 2. Op de planeet Xzorg wonen groene, eenslachtige wezentjes. De wetten op Xzorg bepalen dat elke Xzorgiaan hoogstens 3 partners[2] mag hebben. Dit wordt dan ook zorgvuldig uitgebuit: elke twee Xzorgianen die geen partner van elkaar zijn hebben minstens één partner gemeenschappelijk[3].

(a) Hoeveel wezentjes kunnen er maximaal op Xzorg wonen?

(b) Als je bovendien weet dat er op Xzorg al een triootje heeft plaatsgevonden (wat enkel met partners mag volgens de wet), hoeveel kunnen er dan nog maximaal wonen?

Hint: gebruik bij zowel de constructie van je voorbeelden als de redenering voor je bovengrens, de notatie uit het voorbeeld hieronder: bolletjes voor de Xzorgianen en verbindingslijnen wanneer ze elkaars partner zijn.

Hint2: het kan helpen (maar hoeft niet) de verzameling $S = \{(a, b, c) | a \sim b \text{ en } b \sim c\}$ *te beschouwen, waarbij* $\sim$ *staat voor 'is partner met'.*



Oplossing. (a) Er zijn tal van manieren om deze oefening op te lossen. De eenvoudigste manier is (en daarvoor was Hint 2 ook gegeven) $|S|$ op twee manieren te tellen. Noteer met n het aantal Xzorgianen.

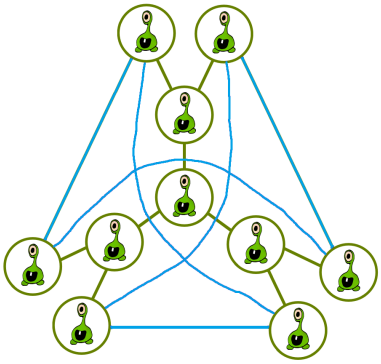
- Enerzijds is elk wezentje de b van hoogstens $3 \cdot 2 = 6$ zo'n drietallen, dus is $|S| \leq 6n$.
- Anderzijds is elk wezentje de a van minstens $n - 4$ zo'n drietallen, want buiten dat wezentje a en zijn hoogstens drie partners b zegt de eigenschap precies dat alle andere wezentjes de c uit zo'n drietal zijn, dus is $|S| \geq n(n - 4)$.

Samen hebben we dus dat $n(n - 4) \leq |S| \leq 6n$, dus $n \leq 10$.

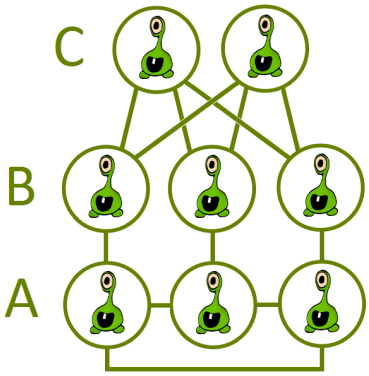
[2] Je mag hierbij onderstellen dat partnerschap een symmetrische relatie is.

[3] Bijgevoegd is een voorbeeld van een geldige configuratie met 6 Xzorgianen. Je krijgt dus al kado dat beide aantallen ≥ 6 zijn. Probeer voor beide delen een zo groot mogelijk voorbeeld te construeren (dit is de helft van de punten) en te bewijzen dat een groter voorbeeld niet kan (dit is de andere helft van de punten). Als het echte antwoord bijvoorbeeld $n = 20$ is, en je kunt een voorbeeld van 15 geven en een bewijs dat $n \leq 25$, dan zal je daar een substantieel deel van de punten voor krijgen.

Om een voorbeeld te kunnen construeren met $n = 10$ moeten we dus gelijkheid hebben in beide bovenstaande afschattingen. Dus moet elke Xzorg precies 3 partners hebben, en moeten elke twee niet-partners precies één gemeenschappelijk partner hebben (ook nooit twee). Als je met die twee extra regeltjes naar een voorbeeld begint te zoeken, kom je al heel snel tot een werkend voorbeeld, zoals het onderstaande.



(b) Dit onderdeel was al meer een doordenkertje. Neem zo'n trio Xzorgs. Zij A de verzameling Xzorgs in het trio, B hun partners buiten het trio en C de overige Xzorgs. Duidelijk is $|A| = 3$. Aangezien elke Xzorg uit A nog maar één andere partner buien A kan hebben, is $|B| \leq 3$, en dus $|C| \geq n-6$. Anderzijds moet elke Xzorg in C een partner gemeen hebben met elke Xzorg in A, dus hij moet minstens drie partners in B hebben. Maar een Xzorg uit B heeft nog hoogstens 2 niet-A-partners, dus $|C| \leq 2$. Samen krijgen we dus $n \leq 8$ en de constructie brengt ons direct bij onderstaand voorbeeld.



□