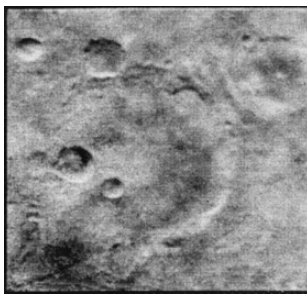

CODES IN DE RUIMTEVAART

Vanaf de jaren '60 van de vorige eeuw werden veel satellieten de ruimte in gestuurd om foto's te maken van de planeten van ons zonnestelsel. In een foto is het van belang dat alle pixels goed doorgezonden worden.

Stel dat er iets mis gaat bij het versturen vanuit een foto door een satelliet die langs een verre planeet zweeft. Je zou kunnen vragen of de satelliet de foto nog een keer wil sturen.

- Waarom is dit geen goede optie?



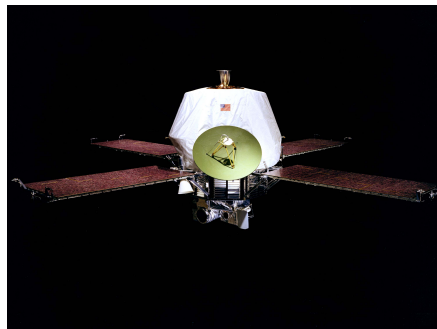
Deze foto werd in 1965 gemaakt van het oppervlak van Mars door de Mariner 4. Een foto is 200 bij 200 pixels. Iedere pixel van een foto werd gecodeerd door 6 bits: een rijtje van zes nullen/enen.

- Hoeveel verschillende grijs tinten kan je met deze 6 bits onderscheiden?

Omdat bij het versturen van berichten van Mars naar de Aarde nogal wat fouten konden optreden, werden de foto's héél langzaam verstuurd, met 25/3 bits per seconde. Zo waren de bits goed herkenbaar.

- Hoe lang duurde het om een foto in zijn geheel naar de Aarde te sturen? Hoeveel uur is dat?

De Mariner 4 maakte 21 foto's van Mars. Het versturen van deze foto's kostte dus een week! Bij de Mariner 9, die in 1971 gelanceerd werd en die de eerste satelliet was die in een baan rond Mars gebracht werd, had men iets slimmere bedacht.



De foto's die de Mariner 9 van Mars maakte, waren nog steeds 200 bij 200 pixels en iedere pixel werd gecodeerd met 6 bits. Maar bij elke 6 bits werden er nog 26 achter geplakt, zodat iedere pixel gecodeerd werd met 32 bits.

- Hoeveel bits moesten er nu doorgestuurd worden?

Omdat er gebruik gemaakt werd van een code, kon het versturen van de bits veel sneller: met 16.200 bits per seconde.

Coderingstheorie: meer enen en nullen voor een betere digitale communicatie

- Hoe lang duurde het nu voordat een foto naar de aarde gestuurd was?
- Hoeveel sneller is dit dan het versturen van de foto's die de Mariner 4 maakte?

De 64 codewoorden die de Mariner 9 gebruikte, zijn zo gekozen dat bij een verminking van maximaal 7 bits het nog steeds mogelijk is de juiste grijswaarde te achterhalen. Hier zie je de code:

0	00000000000000000000000000000000	32	10010110011010010110100110010110
1	00000000000000001111111111111111	33	10010110011010011001011001101001
2	00000000111111110000000011111111	34	10010110100101100110100101101001
3	00000000111111111111111100000000	35	10010110100101101001011010010110
4	00001111000011110000111100001111	36	10011001011001100110011010011001
5	00001111000011111111000011110000	37	10011001011001101001100101100110
6	00001111111100000000111111110000	38	10011001100110010110011001100110
7	00001111111100001111000000001111	39	10011001100110011001100110011001
8	00110011001100110011001100110011	40	10100101010110100101101010100101
9	00110011001100111100110011001100	41	10100101010110101010010101011010
10	00110011110011000011001111001100	42	10100101101001010101101001011010
11	00110011110011001100110000110011	43	10100101101001011010010110100101
12	00111100001111000011110000111100	44	10101010010101010101010101101010
13	00111100001111001100001111000011	45	10101010010101011010101001010101
14	00111100110000110011110011000011	46	10101010101010100101010101010101
15	00111100110000111100001100111100	47	10101010101010101010101010101010
16	01010101010101010101010101010101	48	11000011001111000011110011000011
17	01010101010101011010101010101010	49	11000011001111001100001100111100
18	01010101101010100101010110101010	50	11000011110000110011110000111100
19	01010101101010101010100101010101	51	11000011110000111100001111000011
20	01011010010110100101101001011010	52	11001100001100110011001111001100
21	01011010010110101010010110100101	53	11001100001100111100110000110011
22	010110101001010101101010100101	54	11001100110011000011001100110011
23	010110101001011010010101011010	55	11001100110011001100110011001100
24	01100110011001100110011001100110	56	11110000000011110000111111110000
25	01100110011001101001100110011001	57	11110000000011111111000000001111
26	01100110100110010110011010011001	58	11110000111100000000111100001111
27	01100110100110011001100101100110	59	11110000111100001111000011110000
28	01101001011010010110100101101001	60	11111111000000000000000011111111
29	01101001011010011001011010010110	61	11111111000000001111111100000000
30	01101001100101100110100110010110	62	11111111111111110000000000000000
31	01101001100101101001011001101001	63	11111111111111111111111111111111

- Extra: Kies een woord uit deze tabel en verander 7 bits. Wat moet de ontvanger doen om te achterhalen welk woord dit was? Kan je daar een handige manier voor verzinnen?

Coderingstheorie: meer enen en nullen voor een betere digitale communicatie

DE ISBN - CODE

Om het opzoeken van boeken in de bibliotheek en in boekwinkels makkelijker te maken, heeft ieder boek dat wordt uitgegeven een International Standard Book Number, kortweg ook wel ISBN. Dit bestaat uit 10 cijfers (oude systeem) of 13 cijfers (nieuwe systeem). Vaak staan er ook wat streepjes in het ISBN, maar die kun je voor nu negeren. Het laatste cijfer van een ISBN-10 code kan behalve nul tot en met negen ook 'X' zijn; dit stelt dan het 'cijfer' 10 voor.

Hoe werkt deze code?

Als we een ISBN-10 code noteren als $c_1c_2c_3c_4c_5c_6c_7c_8c_9c_{10}$, dan kunnen we als volgt controleren of het nummer een correct ISBN is: vermenigvuldig de positie van het cijfer met het cijfer zelf, en tel alle producten op. Het linkse cijfer heeft positie 1, het rechtse positie 10. Als het resultaat deelbaar is door 11 is het nummer een geldig ISBN; anders niet.

Laten we de volgende ISBN controleren: 0-34-539180-2. We berekenen eerst het controlegetal, ook wel de 'checksum' geheten:

$$1 \cdot 0 + 2 \cdot 3 + 3 \cdot 4 + 4 \cdot 5 + 5 \cdot 3 + 6 \cdot 9 + 7 \cdot 1 + 8 \cdot 8 + 9 \cdot 0 + 10 \cdot 2.$$

Dit komt uit op 198 en dat is inderdaad deelbaar door 11. Dit is dus een geldig ISBN!

Opgaven

- Controleer de volgende ISBN: 9-06-069088-5, 9-02-586844-X, 9-06-068540-9.
- Het laatste cijfer van een ISBN-10 nummer is het zogenaamde 'check digit'. Dit kan altijd zo gekozen worden dat het ISBN nummer voldoet aan de hierboven omschreven voorwaarde. (Waarom?) Wat is bijvoorbeeld het laatste cijfer van het ISBN 0-631-18385-□ ?
- Hoe kan je in het algemeen de check digit vinden?
- Bepaal het missende cijfer in het volgende ISBN: 0-78-030□34-9.

Waarom een ISBN-code?

Je kunt je voorstellen dat het sommetje dat nodig is voor het controleren van een ISBN voor een computer een fluitje van een cent is. Het is in elk geval heel veel makkelijker dan het bijhouden van een grote lijst met alle mogelijke ISBN codes. Zo'n lijst vereist veel opslagcapaciteit, is niet makkelijk up-to-date te houden, en het doorzoeken is langzamer dan het simpelweg berekenen van het controlegetal.

Behalve voor het detecteren van één fout, en het corrigeren van één fout wanneer we weten om welke positie het gaat, is de ISBN-code zo ontworpen dat als je twee tekens in een geldig ISBN per ongeluk verwisselt, je nooit een ander geldig ISBN kunt krijgen. Dit is belangrijk omdat dit een veel gemaakte fout is, bijvoorbeeld bij het oplezen en overtypen van een ISBN.

Naast de ISBN-10 code zijn er nog verschillende andere voorbeelden van codes die bijna hetzelfde werken, namelijk bankrekeningnummers, het burgerservicenummer, en serienummers van bankbiljetten.

Extra opgaven

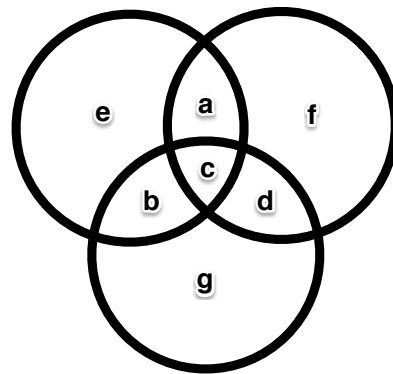
- Leg uit waarom je bij het controleren van het ISBN rekent modulo 11.
- Kan je een fout in een ISBN corrigeren als je niet weet op welke plek de fout is gemaakt? Waarom?
- Toon aan dat als je bij een geldig ISBN twee cijfers omdraait, je nooit een ander geldig ISBN krijgt.

DE HAMMINGCODE

Deze code van nullen en enen kan één fout opsporen en verbeteren.

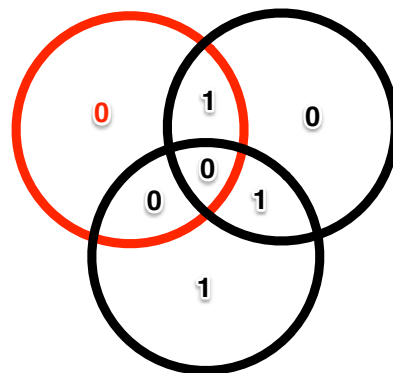
Coderen

- Kies een bericht van vier bits ('bits' zijn nullen of enen).
- Zet het bericht in het midden van de cirkels, op posities *a*, *b*, *c*, *d*.
- Vul de posities *e*, *f*, *g* in, zodanig dat in elke cirkel een even aantal enen staat.
- Schrijf de zeven bits achter elkaar in de volgorde *abcdefg*: dit is het gecodeerde woord.



Decoderen

- Schrijf de zeven bits van het ontvangen woord op de juiste posities in de cirkels.
- Tel het aantal enen in iedere cirkel. Een cirkel is 'fout' als er een oneven aantal enen in staat, anders 'goed'.
- Zoek de positie die in de foute cirkel(s) ligt, maar niet in de goede cirkel(s). Verander de bit op die positie.
- Schrijf de zeven bits weer achter elkaar in de volgorde *abcdefg*.
- Laat de laatste drie cijfers weg: je houdt het oorspronkelijke bericht over.

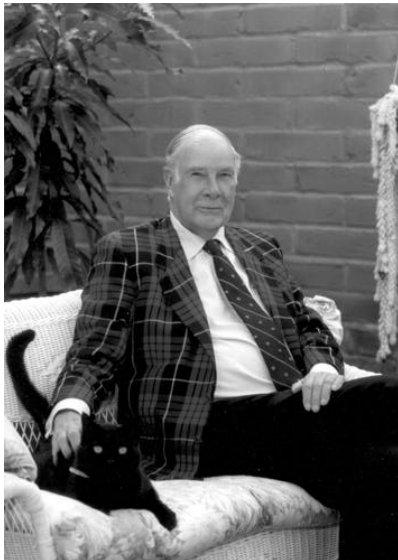


Coderingstheorie: meer enen en nullen voor een betere digitale communicatie

Opgaven

- Codeer de berichten 1101, 0000 en 0011.
- Decodeer de volgende ontvangen woorden: 1001001, 1011010 en 1001110.
- Kies je eigen bericht en codeer dit. Verander nu maximaal één van de zeven bits. Laat vervolgens iemand anders dit decoderen. (Ondertussen kan je zelf zijn/haar bericht decoderen.)
- Codeer een zelfgekozen bericht en verander twee bits. Decodeer het veranderde woord. Krijg je het goede bericht terug?
- Kan een gecodeerd bericht bestaan uit één 1 en zes nullen? Waarom wel/niet?
- Hoeveel rijtjes van zeven nullen/enen zijn er? Hoeveel mogelijke berichten van vier nullen/enen zijn er? Welk percentage van de rijtjes van zeven nullen/enen zijn codewoorden van de Hamming code?

Achtergrond



De Hamming code is vernoemd naar Richard Hamming, die de code bedacht. In de jaren '50 van de vorige eeuw werkte hij in Bell Labs, waar ze de eerste computers maakten. Hamming was erg gefrustreerd dat één fout beetje in zijn invoer meteen de computer deed vastlopen. Daarom verzong hij de Hamming code.

De Hamming code is ook de reden dat het Hamming spelletje, dat je al hebt gedaan of nog gaat doen, werkt. Kan je de overeenkomsten zien?

Coderingstheorie: meer enen en nullen voor een betere digitale communicatie

HAMMING SPEL

Het spel

Vraag aan je tegenspeler een getal te bedenken tussen 1 en 16 (beiden inbegrepen) en het op te schrijven zonder het aan jou te laten zien. Je zal hem of haar nu 7 vragen stellen, eentje voor elk kaartje, en op elke vraag moet hij of zij 'ja' of 'nee' antwoorden. Hij of zij mag zelf kiezen of hij of zij de waarheid spreekt, of één keer liegt (maar niet vaker!). De vraag die je telkens stelt is heel eenvoudig: als je hem of haar een kaartje toont (het maakt niet uit welke kant je aan de bovenkant houdt) vraag je of het geheime getal op de bovenkant van het kaartje staat. Als het antwoord 'nee' is, dan draai je het kaartje om en leg je het neer. Als het antwoord 'ja' is, dan leg je het kaartje gewoon neer zoals je het vasthield. Leg op deze manier alle beantwoorde kaartjes op een stapeltje.

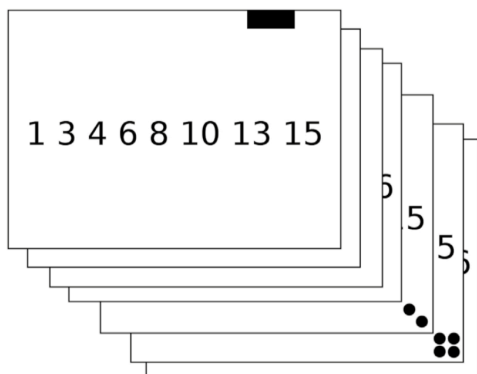
Hoe vind je dat getal nu?

1. Schuif alle kaartjes een beetje uit elkaar, zoals op deze figuur:



2. Aan de bovenkant zie je zwarte of witte rechthoekjes in 3 groepjes: links, midden en rechts. In elk van de 3 groepen zou het aantal zwarte rechthoekjes even moeten zijn. Als dat klopt, dan heeft je tegenspeler geen enkele keer gelogen. Als het niet klopt, vind dan elk van de 3 groepen waar het niet klopt en bepaal het kaartje dat rechthoekjes heeft in precies die groep(en).

Dat kaartje draai je om en steek je opnieuw in de stapel. Het kaartje dat je hebt moeten omdraaien is het kaartje waarbij je tegenspeler gelogen heeft. In het voorbeeld is dat het bovenste kaartje.



3. Schuif nu de kaartjes weer uit elkaar zodat de onderkanten zichtbaar zijn. Tel nu het aantal zichtbare bolletjes. Wanneer je 0 bolletjes telt, had je tegenspeler het getal 16 gekozen. Zo niet, dan komt het aantal bolletjes exact overeen met het gekozen getal. In het voorbeeld is dat het getal 6.

Opgaven


- Knip het spelletje uit en speel het een met een collega. Neem allebei een keer een getal in gedachten en probeer allebei een keer het getal van de ander te raden.
- Wat doe je in stap 2 als je tegenspeler nooit gelogen heeft?
- Probeer eens meer dan 1 keer te liegen. Kan je tegenspeler jouw getal raden?
- Extra: Kan je het verband met de Hammingcode omschrijven?

Hoezo coderingstheorie?



Coderingstheorie beschermt informatie tegen mogelijke fouten die kunnen optreden tijdens transport over een onbetrouwbaar kanaal. Denk hierbij aan het spel waarbij de tegenspeler mag liegen. Toch wil je graag de juiste informatie verkrijgen. Om die reden bedacht men foutverbeterende codes.

De code die in het spelletje verborgen zit is de zogenaamde Hammingcode die aan 4 bits 3 extra bits toevoegt om 1 fout te kunnen verbeteren. Als je je afvraagt waarom de getallen tussen 1 en 16 kunnen voorgesteld worden door 4 bits, dan moet je ze maar eens proberen binair te schrijven! De code is trouwens genoemd naar zijn uitvinder, Richard Wesley Hamming (1915-1998). Hij was een Amerikaans wiskundige.



Coderingstheorie: meer enen en nullen voor een betere digitale communicatie





1 3 5 7 9 11 13 15




4 5 6 7 12 13 14 15





2 3 6 7 10 11 14 15




1 2 4 7 9 10 12 15



8 9 10 11 12 13 14 15



1 2 5 6 8 11 12 15



1 3 4 6 8 10 13 15

1 2 3 8 9 10 11 16

2 4 6 8 10 12 14 16

3 5 6 8 11 13 14 16

1 4 5 8 9 12 13 16

3 4 7 9 10 13 14 16

1 2 3 4 5 6 7 16

2 5 7 9 11 12 14 16

DE QR-CODE

Je kent ongetwijfeld de streepjescode of barcode, die we op al onze producten in de winkel terugvinden. Iets minder bekend is de QR-code, alhoewel we die ook steeds vaker tegenkomen in het dagelijkse leven. Deze QR-code, of voluit de 'Quick Response Code', is een tweedimensionale streepjescode. Hij werd in 1994 ontwikkeld door Denso Wave, een bedrijf dat gespecialiseerd is in de productie en ontwikkeling van auto-onderdelen. Het oorspronkelijke doel van deze QR-code was om de voertuigen te volgen tijdens de fabricage. Sindsdien is deze code echter uitgegroeid tot een van de meest populaire tweedimensionale streepjescodes. De meeste smartphones kunnen tegenwoordig QR-codes lezen.

Opgaven

- Je krijgt enkele QR-codes op een apart vel papier. Zoek eerst uit hoe je ze met je smartphone kan lezen (soms kan dat rechtstreeks via de camera-app).
- Probeer zelf eens na te gaan in hoeverre de QR-code leesbaar blijft bij het beschadigen van de code. Wees creatief met het beschadigen: kleur witte vakjes zwart, maak een scheur in het papier, zet er gekleurde stippen op...
- Zijn er bepaalde delen van de code die niet beschadigd mogen worden?
- Merk je verschillen op tussen de codes? Is er een code die langer leesbaar blijft als je de codes verder blijft beschadigen, terwijl een andere code onleesbaar wordt?

Wat is dat nu, zo'n QR-code?

Een QR-code codeert berichten met een lengte van 8 bits (1 byte). De lengte van de codewoorden in onze code hangt af van hoeveel fouten we willen kunnen verbeteren. Als we meer fouten willen kunnen verbeteren, dan zullen we ook minder informatie kunnen opslaan in onze code. Algemeen kunnen we echter vrij veel informatie in een QR-code opslaan. Dankzij een aantal coderingen zijn we in staat om verschillende foutverbeterende niveaus te bereiken. We onderscheiden bij QR-codes vier verschillende niveaus van foutverbetering, die tussen de 7% en 30% informatieverlies toestaan.

Coderingstheorie: meer enen en nullen voor een betere digitale communicatie

QR-code of streepjescode?



Bij het omzetten van langere teksten in een QR-code, wordt de tekst opgedeeld in verschillende blokken. Dit heeft een aantal voordelen. Ten eerste beperken we binnen elk blok het aantal fouten dat we kunnen verbeteren, zodat het gemakkelijker wordt om het oorspronkelijke bericht te lezen met onze QR-codescanner. Ten tweede kunnen we zo ook artistieke QR-codes maken, met bijvoorbeeld een logo erin, terwijl je de data toch kan blijven lezen.

Om terug te komen op de gewone streepjescode die we dagelijks op al onze producten tegenkomen, geven we nog enkele verschillen tussen de QR-code en de streepjescode.

Een eerste verschil vinden we bij de opslag van de data. Bij een gewone barcode wordt de informatie enkel in de breedte (horizontaal) opgeslagen, terwijl bij een QR-code de informatie zowel in de breedte (horizontaal) als in de hoogte (verticaal) wordt opgeslagen. Een QR-code gebruikt dus twee dimensies voor de informatie.

Een tweede verschil is de hoeveelheid informatie die we kunnen opslaan. Een gewone streepjescode kan ongeveer 20 cijfers bevatten. Als we in een QR-code binaire data opslaan, dan kunnen we maximaal 2953 bytes opslaan, wat dus heel wat meer informatie is dan bij een gewone streepjescode!

Tot slot is het dankzij het gebruik van positiedetectie ook mogelijk om een QR-code te scannen vanuit elke hoek. Je kan een QR-code dus altijd lezen, ook al staat die op zijn kop of scan je de code schreef.



