SUDOKU'S EN WISKUNDE

KLAAS PIETER HART

Inleiding

Hoewel ze reeds geruime tijd bestaan — sinds 1979 in de V.S. onder de naam *Number Place*, sinds 1986 in Japan onder the namen *Nanpure* en *Sudoku* — zijn de Sudoku-puzzels pas het afgelopen jaar in onze buurt een ware rage gaan vormen, eerst in Groot-Brittanië en nu ook in Nederland. Voor degenen die nog in zalige onwetendheid verkeren: hier is een Sudoku-puzzel (nummer 80 uit [7]):

	5			6				
		1					9	
					7			4
2	4		9					
		8				6		
					5		3	7
7			8					
	9					2		
				3			1	

De bedoeling is in elk vak een cijfer van 1 tot en met 9 te plaatsen zó dat in elke rij, elke kolom en in elk 3×3-blok (vetomlijnd) elk cijfer éénmaal voorkomt. Een goedgestelde Sudoku heeft precies één oplossing en dat is iets dat we in onze beschouwingen altijd mee zullen nemen.

De reactie op de Sudoku's verschilt van persoon tot persoon maar onder wiskundigen merk ik vooral de neiging tot uitzoeken-hoe-het-allemaal-in-elkaar-zit. Voor degenen die daar nog niet de tijd voor hebben gehad of genomen zal ik in deze bijdrage een paar wiskundige aspecten van de puzzels belichten. De keuze voor onderstaande onderwerpen is tot stand gekomen door rond te vragen en door eigen voorkeur.

(1) Sudoku's tellen.
(2) Moeilijkheisgraad vaststellen.
(3) Hoeveel cijfers moet men minimaal geven?
(4) Sudoku is moeilijk.

Voor de goede orde: ik beperk mij in dit artikel tot de gewone, 9×9, Sudoku's, voornamelijk om de omvang van het verhaal binnen bepaalde perken te houden en ook omdat de variaties *wiskundig* niet veel nieuws opleveren. Om bepaalde ideeën te illustreren gebruik ik Baby-Sudoku's, met een 4×4-speelveld.

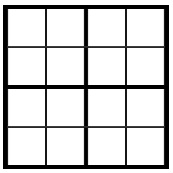
1. Sudoku's tellen

Een veel gestelde vraag, ook op `www.wisfaq.nl` opgeworpen, is "Hoeveel puzzels zijn er eigenlijk?". Die vraag is nog niet beantwoord; hij is ook nog niet helemaal goed gesteld omdat bepaalde aspecten van Sudoku's nog niet zijn opgehelderd.

Date: Monday 30-01-2006 at 13:05:37 (cet).

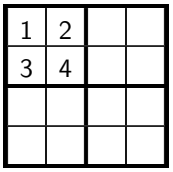
Een indicatie dat die telling een niet-triviale onderneming zal zijn krijgen we als we bekijken hoe de verzameling van alle ingevulde Sudoku-vierkanten geteld is en zien hoe groot die verzameling wel niet is.

1.1. **Baby-Sudoku's tellen.** Om te laten zien hoe zo'n telling in zijn werk gaat tellen we Baby-Sudoku-vierkanten.

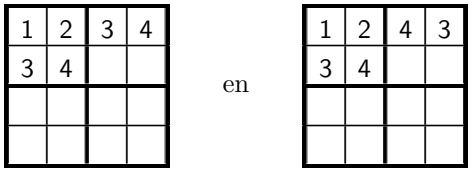


De bedoeling is dit diagram, net als in het 9×9-geval, te vullen met de cijfers 1, 2, 3 en 4 en wel zo dat elk cijfer in elke rij, kolom of blok precies één keer voorkomt.

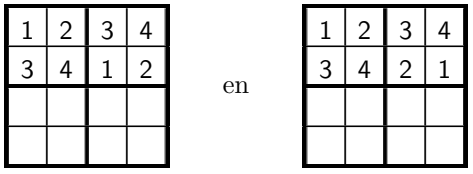
Om al deze invullingen te tellen heeft het zin eerst even na te denken of het niet handiger kan dan door alle vierkanten uit te schrijven en te tellen. Het grote probleem bij die aanpak is het risco van toch wel erg veel hooi op de vork en het gevaar van het missen van een paar vierkanten. Het is zaak het werk verstandig in te delen en zo de telling tot overzichtelijke werkjes te reduceren. De eerste stap geeft een forse reductie: tel alleen de vierkanten die als volgt beginnen:



Als we deze geteld hebben kunnen we het resultaat met 4! vermenigvuldigen om het eindresultaat te krijgen: elke permutatie van $\{1, 2, 3, 4\}$ levert een andere linkerbovenhoek met evenveel aanvullingen als het speciale voorbeeld. We gaan nog even door met verdelen; de eerste rij kan op twee manieren worden aangevuld:



Beide hebben evenveel completeringen: verwissel de laatste kolommen. We werken verder met het linker vierkant, dat representeert dus 48 groepen die allemaal even groot zijn. Door de tweede rij ook aan te vullen ontstaan weer twee mogelijkheden:



Nu is er geen overduidelijke reden waarom deze evenveel completeringen zouden hebben (het zou kunnen maar we weten het niet ...), daarom bekijken we ze apart. We kunnen de eerste kolom nog aanvullen, 'verwisseling van de laatste rijen' laat zien dat we, in beide gevallen het werk kunnen halveren. De posities van 2 en 4

liggen dan ook vast.

1	2	3	4
3	4	1	2
2		4	
4		2	

en

1	2	3	4
3	4	2	1
2		4	
4			2

Links hebben we nog twee mogelijkheden maar rechts nog maar één:

1	2	3	4
3	4	1	2
2	1	4	3
4	3	2	1

en

1	2	3	4
3	4	1	2
2	3	4	1
4	1	2	3

en

1	2	3	4
3	4	2	1
2	1	4	3
4	3	1	2

Nu terugrekenen: links krijgen we 4 completeringen en rechts 2. Dus elk van de 48 groepen heeft 6 elementen; dus in totaal zijn er $48 \times 6 = 288$ Baby-Sudoku-vierkanten.

Belangrijke punten uit deze telling: we verdeelden de vierkanten in groepen, eerst $4! = 24$ en elk van die nog in twee, en allemaal even groot. Eén zo'n groep verdeelden we in twee groepen die niet meer even groot waren maar wel redelijk snel te tellen.

1.2. **Sudoku-vierkanten tellen.** Dezelfde strategie is gebruikt om de Sudoku-vierkanten te tellen. Begin met

1	2	3						
4	5	6						
7	8	9						

dat reduceert het werk met een factor 9!. De eerste telling werkte met aanvullingen van de eerste drie rijen en verdeelde die in, uiteindelijk, 71 equivalentieklassen van aanvullingen met per klasse telkens evenveel completeringen. De verdeling kwam tot stand door allerlei symmetrieën uit te buiten: kolompermutaties binnen de blokken, permutaties van de blokken zelf, hernummeringen enzovoort. Die 71 tellingen werden, met bruut geweld, door de computer gedaan. De tellingen gaven 44 verschillende uitkomsten; enig speurwerk levered nog wat equivalenties op waardoor men inderdaad het aantal klassen tot die 44 kon reduceren.

Zie de slides voor een aantal voorbeelden van het soort reducties dat toegepast werd.

Het resultaat. Er zijn 6 670 903 752 021 072 936 960 Sudoku-vierkanten.

Een rechtgeaarde wiskundige vraagt zich natuurlijk af: zijn die 'echt' allemaal verschillend? Natuurlijk niet: permutatie van de cijfers geeft niet wezenlijk andere Sudoku's, net als permutatie van rijen/kolommen binnen de 'dikke' rijen/kolommen of permutatie van de dikke rijen/kolommen zelf. Verder kunnen we nog roteren en spiegelen in de diagonalen. De hierbij behorende Sudoku-symmetriegroep is ook

bepaald en het resultaat is dat er 'maar' 5 472 730 538 echt verschillende Sudokovierkanten zijn. Alle informatie over deze tellingen is te vinden via [5].

In [4] wordt beschreven hoe die symmetriegroep kan helpen om wel heel snel een boekje met Sudoku-puzzels in elkaar te zetten.

2. Moeilijkheidsgraad

Een andere vraag die vaak terugkwam: "Hoe bepalen ze de moeilijkheidsgraad?" Hier is geen eenduidig *wiskundig* antwoord op te geven want moeilijkheid is een subjectief begrip. De gang van zaken is in het algemeen als volgt: men laat de puzzel door een programma oplossen en daarbij bijhouden wat voor oplosstappen worden gebruikt. De aard van de gebruikte stappen bepaalt de moeilijkheidsgraad.

2.1. **De NWD-classificatie van Sudoku's.** Bij het voorbereiden van mijn lezing heb ik een programmaatje geschreven dat een paar stappen die ikzelf veel gebruik implementeert. Die stappen zijn

wegstrepen: systematisch de rijen, kolommen en blokken aflopen en door gegeven cijfers uit andere cellen af te strepen zien wat overblijft

uniciteit controleren: zoek per cijfer uit of er in een rij, kolom of blok nog maar één mogelijke positie vrij is

rij/kolom in blok: als een cijfer binnen een blok in nog maar één rij kan zitten kun je het in de rest van die rij wegstrepen (evenzo voor kolommen)

Uitgaande van deze stappen komen we tot de volgende classificatie van Sudoku's, de *NWD-classificatie van Sudoku's*:

makkelijk: kan met wegstrepen

beetje moeilijk: kan niet alléén met wegstrepen maar heeft uniciteitscontroles nodig

moeilijk: heeft al onze stappen nodig

heel moeilijk: lukt niet met mijn programmaatje

Op de slides zijn voorbeelden van alle vier moeilijkheidsgraden te vinden.

Zie [1, 3] voor nog meer voorbeelden van oplosstappen.

3. Hoeveel zaadjes?

Een nog onopgelost probleem is: "hoeveel cijfers moeten minimaal gegeven worden om de puzzel uniek oplosbaar te doen zijn?"

Je moet zeker 8 *verschillende* cijfers geven anders kun je uit een oplossing een andere maken door de niet gegeven cijfers om te wisselen.

Verder is er een grote collectie van puzzels met 17 gegeven cijfers gevonden maar (nog) geen enkele met 16, zie [9].

Ik heb nog geen *bewijs* gezien dat er geen puzzel met 8 gegeven cijfers is ...

3.1. **Zaadjes voor Baby-Sudoku.** Voor Baby-Sudoku is met de hand eenvoudig vast te stellen dat 4 cijfers genoeg is:

1			
			3
		2	
	4		

Het is ook eenvoudig met de hand vast te stellen dat drie *niet* genoeg is. Met moet in ieder geval drie *verschillende* cijfers gebruiken; we proberen dus met 1, 2 en 3 een

puzzel te maken. Na diverse symmetrie-argumenten kom je uit op de de volgende mogelijkheden

1			
3	3	2	
3	3	3	3

met vaste posities voor 1 en 2 en zes mogelijkheden voor 3. Even spelen met de zes afzonderlijke diagrammetjes leert dat geen van deze een unieke oplossing heeft.

3.2. **Kan het met acht?** Net als bij het tellen wordt het werk bij de stap van Baby- naar echte Sudoku schier oneindig veel groter. Met symmetrie-overwegingen komen we tot

1						∅	∅	∅
						∅	∅	∅
						∅	∅	∅
			2					
						3		

waarbij '∅' inderdaad 'leeg' betekent: met acht cijfers zal tenminste één 3×3-blok leeg blijven. De overige cijfers 4, 5, 6, 7 en 8 moeten zó gezaaid worden dat in elke dikke rij/kolom zeker twee rijen/kolommen iets krijgen; anders kun je door verwisseling van die twee rijen/kolommen een andere oplossing creëren. Omdat de cijfers verschillend zijn zijn alleen de posities van belang: er zijn $\binom{69}{5} = 11\,238\,513$ willekeurige zaaiingen, daarvan valt een redelijk aantal af maar er blijven genoeg mogelijkheden over ...; het lijkt onbegonnen werk die met de hand na te lopen.

Kansrekening. Het totaal aantal puzzels met acht verschillende zaadjes is snel geteld: kies acht posities, kies acht cijfers en verdeel die. Het resultaat is $\binom{81}{8} \times \binom{9}{8} \times 8! = 11\,671\,764\,328\,224\,000$ puzzels. De verhouding tussen het aantal zaaiingen, $1{,}167 \times 10^{16}$, en het aantal Sudoku-vierkanten, $6{,}67 \times 10^{21}$, is ongeveer $1 : 571\,542$. Die verhouding wordt alleen maar kleiner als we alleen 'goede' zaaiingen bekijken. Het lijkt dus nogal onwaarschijnlijk dat er een puzzel met acht zaadjes is maar helemaal sluitend is dit argument natuurlijk niet.

3.3. **Hoe zit het met negen (of meer)?** De situatie bij acht zaadjes, zoals boven geschetst, is nog redelijk overzichtelijk maar bij negen of meer wordt de zaak aanzienlijk gecompliceerder. Omdat er cijfers meer dan één keer voor kunnen komen moet men echt op de plaatsing gaan letten en zo wordt de zoekruimte aanzienlijk groter.

4. Sudoku is moeilijk

Wie een Sudoku-oplosser schrijft zal merken dat er altijd wel een puzzel is die het programma nog niet aankan en wie de fora op, bijvoorbeeld, `www.sudoku.com` bezoekt ziet dat er nog steeds over nieuwe oplosstappen wordt nagedacht. Dat is niet geheel verwonderlijk want Sudoku's behoren tot een klasse van problemen die, met de huidige stand van de kennis, als 'moeilijk' worden betiteld.

Het verschil tussen 'makkelijke' en 'moeilijke' problemen zal ik hieronder aan de hand van twee voorbeelden duidelijk proberen te maken.

4.1. **Makkelijke problemen.** Het uitrekenen van een $n \times n$ determinant is 'makkelijk'. Dat wil zeggen, na enig denkwerk. Wie de officiële recursieve definite door middel van 'ontwikkelen naar de eerste rij' implementeert overleeft het uitrekenen van een 25×25-determinant niet. De ontwikkelformule komt namelijk naar op het uitvoeren van veel meer dan $n!$ vermenigvuldigingen en bij $n = 25$ wordt dat $25! \approx 1{,}55 \times 10^{25}$ (een computer die 10^{12} vermenigvuldigingen per seconde kan uitvoeren doet er dus zeker 490 513 jaar over).

Door de matrix eerst op driehoeksvorm te brengen en dan de diagonaalelementen te vermenigvuldigen reduceert men het werk tot $n+(n-1)+\cdots+1+(n-1) = \frac{1}{2}n^2+\frac{3}{2}n-1$ vermenigvuldigingen. Bij $n = 25$ wordt dat 349, een redelijk tijdsbesparing.

Het uitrekenen van determinanten is iets dat in *polynomiale* tijd gedaan kan worden; dat wil zeggen: er is een polynoom p met de eigenschap dat de rekentijd bij een invoer ter grootte n niet langer duurt dan $p(n)$ tijdseenheden. Voor het determinant-uitrekenen hebben we dus $p(n) = \frac{1}{2}n^2 + \frac{3}{2}n - 1$ gevonden.

De letter P staat voor de klasse van dergelijke problemen; bepalen of een getal n priem is is ook in P *als functie van het aantal cijfers van* n, zie [11]. Dit zijn de 'makkelijke' problemen; de rest is 'moeilijk'.

4.2. **Moeilijke problemen.** Een bekend combinatorisch probleem is het volgende: *Gegeven een eindige verzameling en een overdekking van die verzameling, dun die overdekking uit tot een* disjuncte *overdekking.* Zo'n probleem is met behulp van een matrix met nullen en enen te coderen. De punten van de verzameling komen overeen met de kolommen en de rijen geven de verzamelingen uit de overdekking weer. Een 1 op de positie (i, j) in de matrix geeft aan dat punt j tot de i-de verzameling behoort.

Hier is een voorbeeld, waarin de vetgedrukte rijen een disjuncte overdekking vormen.

$$\begin{array}{ccccccc}
1 & 0 & 0 & 1 & 0 & 0 & 1 \\
\mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
0 & 0 & 0 & 1 & 1 & 0 & 1 \\
\mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} \\
0 & 1 & 1 & 0 & 0 & 1 & 1 \\
\mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1}
\end{array}$$

Voor dit probleem is (nog) geen polynomiaal algoritme bekend, dus we weten (nog) niet of het tot P behoort. Op dit moment is het dus een 'moeilijk' probleem.

Het behoort wel tot de klasse NP, dat zijn problemen waarvan het verifiëren dat een oplossing inderdaad correct is wel in P zit. De N staat voor 'Non-deterministic': het oplossen is polynomiaal als je maar een orakel tot je beschikking hebt dat op cruciale momenten tijdens het oplossen duidelijke informatie in de goede richting geeft (dat kan dus ook een oplossing zijn).

Het overdekkingsprobleem is wat NP-*volledig* heet: als men kan bewijzen dat dit probleem toch in P zit dan heeft men gelijk bewezen dat P = NP. De geldigheid van die laatste gelijkheid is één van de Clay Millenium-problemen; het vaststellen of ontkrachten daarvan levert een miljoen dollar op.

Andere NP-volledige problemen: het handelsreizigerprobleem en Minesweeper. Voor meer over P = NP zie [2, 11].

4.3. **Sudoku is** NP**-volledig.** We zullen zo dadelijk zien dat het oplossen van Sudoku's een speciaal geval van het oplossen van overdekkingsproblemen is. Dit laat zien dat het oplossen van willekeurige Sudoku's tot de klasse NP behoort.

Aan de andere kant: het completeren van partiële Latijnse vierkanten, ook een bekend NP-volledig probleem, kan men oplossen door Sudoku's op te lossen; dit maakt het algemene Sudoku-probleem dus ook NP-volledig.

Een (partieel) $n \times n$ Latijns vierkant wordt hierbij gerelateerd aan een $n^2 \times n^2$ Sudoku. Dat gaat eigenlijk redelijk eenvoudig, we construeren eerst een speciale Sudoku:

	1	2		4	5		7	8
	4	5		7	8		1	2
	7	8		1	2		4	5
1	2	3	4	5	6	7	8	9
4	5	6	7	8	9	1	2	3
7	8	9	1	2	3	4	5	6
2	3	4	5	6	7	8	9	1
5	6	7	8	9	1	2	3	4
8	9	1	2	3	4	5	6	7

elke completering van deze Sudoku hoort bij een 3×3 Latijns vierkant en omgekeerd:

3a	1	2	3d	4	5	3g	7	8
3b	4	5	3e	7	8	3h	1	2
3c	7	8	3f	1	2	3i	4	5
1	2	3	4	5	6	7	8	9
4	5	6	7	8	9	1	2	3
7	8	9	1	2	3	4	5	6
2	3	4	5	6	7	8	9	1
5	6	7	8	9	1	2	3	4
8	9	1	2	3	4	5	6	7

$\Leftrightarrow$

a	d	g
b	e	h
c	f	i

In het algemeen relateert men $n\times n$-Latijnse vierkanten aan $n^2\times n^2$-Sudoku's. Dus elk algoritme dat (willekeurige) Sudoku's oplost kan ook aangewend worden om partiële Latijnse vierkanten te completeren.

Dat Sudoku inderdaad de status van NP-volledig heeft is ook te zien aan de oplosstappen die op de bovengenoemde fora besproken worden: die komen vaak neer op het onderzoeken van elk tweetal/drietal/... in iedere rij, iedere kolom of ieder blok. Dat komt uiteindelijk neer op het onderzoeken van *elke* deelverzameling van zo'n rij, kolom of blok. Dit betekent dat, dankzij die varianten mee, een doorsnee gericht oplosprogramma voor een $n^2 \times n^2$-Sudoku al gauw (veel meer dan) 2^{n^2} stappen zal moeten doen. Een polynomiaal algoritme voor Sudoku zal er derhalve heel anders uit moeten zien dan de standaard oplossers die voor het 9×9-geval zijn geschreven.

4.4. **Back-tracking.** Sudoku's zijn ook met bruut geweld op te lossen: probeer gewoon één voor één de vakken te vullen en ga telkens terug als je vastloopt. Deze methode werkt altijd maar kost nogal wat tijd en ruimte: zo'n programma werkt normaal gesproken recursief en dat betekent dat er voortdurend een heleboel uitgesteld werk op de stapel blijft staan. Zie [6] voor een eenvoudige beschrijving van dit proces. Voor alle duidelijkheid: back-tracking werkt zeker *niet* in polynomiale tijd (het zou in polynomiale tijd werken als iedere keer het *juiste* cijfer gekozen wordt).

Nu is Sudoku te herformuleren tot een overdekkingsprobleem. Voor 9×9-Sudoku een verzameling met 324 punten en een overdekking met 729 deelverzamelingen (die allen precies vier punten hebben). De zaadjes komen dat overeen met verzamelingen uit de overdekking die zeker in de te construeren uitdunning moeten zitten. In [12] wordt de vertaling netjes uitgelegd; het is de moeite waard die pagina even met pen en papier bij de hand door te lezen om wat schetsjes te kunnen maken.

Voor het overdekkingsprobleem is een algemeen algoritme opgesteld dat, vertaald naar de Sudoku-situatie, neerkomt op back-tracken, waarbij telkens een cel met een minimaal aantal mogelijkheden wordt gekozen. Het is daarmee een mix van wegstrepen en back-tracken.

Zie [12] voor een volledige beschrijving van het algoritme, links voor meer informatie en ook programma's die het algoritme implementeren. Waarschuwing: zelfs een Baby-Sudoku geeft aanleiding tot een matrix met 64 kolommen; men kan de uitvoering van het algoritme beter aan de computer overlaten.

Referenties

[1] Andries Brouwer, *Solving sudokus*, `http://homepages.cwi.nl/~aeb/games/sudoku/solving0.html`

[2] Clay Mathematics Institute, *Millennium Prize Problems*, `http://www.claymath.org/millennium/`

[3] Matthijs Coster, *Sudoku wiskundig bekeken*, Pythagoras, januari 2006

[4] Michel Dekking, *Meta Sudoku: hoe produceer je een puzzelboekje voor een habbekrats*, Volkskrant. Origineel via `http://dutiaw37.twi.tudelft.nl/~kp/nwd`

[5] Bertram Felgenhauer en Frazer Jarvis, *There are 6670903752021072936960 Sudoku grids*, `http://www.shef.ac.uk/~pm1afj/sudoku/`

[6] Jos Groot, *Sudoku met de computer*, Pythagoras, februari 2006

[7] The Guardian, *Sudoku, 100 original puzzles*, Guardian Books, 2005

[8] The Guardian, *Sudoku classic*, `http://www.guardian.co.uk/sudoku`

[9] Gordon Royle, *Minimum Sudoku*, `http://www.csse.uwa.edu.au/~gordon/sudokumin.php`

[10] Neil J. A. Sloane, *The On-Line Encyclopedia of Integer Sequences*, `http://www.research.att.com/~njas/sequences/Seis.html`, zie vooral A002860 (Latijnse vierkanten) en A107739 (Sudoku-vierkanten)

[11] Wikipedia, *Complexity classes P and NP*, `http://en.wikipedia.org/wiki/Complexity_classes_P_and_NP`

[12] Wikipedia, *Dancing Links*, `http://en.wikipedia.org/wiki/Dancing_Links`

[13] Wikipedia, *Sudoku*, `http://en.wikipedia.org/wiki/Sudoku`

Faculteit EWI, TU Delft, Postbus 5031, 2600 GA Delft

E-mail address: `K.P.Hart@TUDelft.NL`

URL: `http://fa.its.tudelft.nl/~hart`