In het kader van *Twents Meesterschap* gaan medewerkers van de Universiteit Twente de scholen in om gastlessen te verzorgen. **Mariëlle Stoelinga** en **Mark Timmer** waren te gast op het Twickel College om lessen over zoekmethoden in de informatica te verzorgen. Ziehier hun bevindingen.

Efficiënt zoeken in grote tekstbestanden

Een gastles wiskunde voor HAVO/VWO 3 en 4, verzorgd door de Universiteit Twente

Inleiding

Google, Twitter, en Facebook doorzoeken in een mum van tijd miljarden tekstdocumenten: google je "wiskunde", dan krijg je zo'n vier miljoen resultaten binnen 0,1 seconde. Hoe doen applicaties als Google, Twitter en Facebook dit? Binnen de informatica is een aantal slimme methoden (ook wel algoritmen genoemd) ontwikkeld om snel te zoeken in tekstbestanden. Deze methoden zijn gebaseerd op zogenaamde *eindige automaten*: een speciaal soort grafen waarvan de pijlen gelabeld zijn met letters van het te zoeken woord.

Wij hebben begin dit jaar een gastcollege gegeven over deze methoden aan leerlingen van HAVO 3 en VWO 4. De gastles vond plaats op het Twickel College te Hengelo. In dit artikel beschrijven we deze gastles: we leggen uit hoe deze zoekmethoden werken, en welke opdrachten we gedaan hebben binnen de gastles. Ook gaan we dieper in op de wiskundige achtergronden van de methoden uit de les – dit hoeven de leerlingen zelf niet te weten, maar geeft aan dat onze eindige automaten geen ad hoc methoden zijn, maar deel uitmaken van een rijke en ook elegante wiskundige theorie, die een fundamentele rol speelt binnen de informatica.

Al het lesmateriaal dat we gebruikt hebben, stellen we ter beschikking, inclusief een docentenhandleiding; het kan gedownload worden via de UT-website, zoals hieronder verwezen wordt. We hopen dat dit materiaal wiskundedocenten zal inspireren om ook eens een les over dit onderwerp te geven, en zo leerlingen te laten zien waar wiskunde toe kan leiden.

Context van de gastles

Deze gastles is gegeven in het kader van Twents Meesterschap, een activiteit van de Universiteit Twente (UT) voor HAVO/VWO-docenten. Onder de titel "Op bezoek bij onderzoek" verzorgen onderzoekers aan de universiteit workshops over nieuwe ontwikkelingen in hun eigen onderzoek (wiskunde, informatica, natuurkunde, scheikunde, biologie, economie, en management). Daarnaast zijn er workshops over onderwijsactiviteiten die de UT aanbiedt aan middelbare scholen: profielwerkstukbegeleiding, online leeromgevingen in de klas en het leerlingenlab waar leerlingen proeven kunnen doen met geavanceerde proefopstellingen die op de meeste middelbare scholen niet aanwezig zijn. Meer informatie over het Twents Meesterschap is te vinden onder www.utwente.nl/lerarenconferentie.

Om lesuitval te voorkomen, worden de lessen van docenten die deelnemen aan het Twents Meesterschap overgenomen door studenten en docenten van de Universiteit Twente. Wij hebben, zoals gezegd, enkele lessen overgenomen aan het Twickel College te Hengelo.

Naïef zoeken in teksten

Het zoekprobleem dat we in de in de gastles hebben behandeld, luidt als volgt: gegeven een woord w (ook wel patroon genoemd) en een tekst T, willen we weten of w voorkomt in T. Een voor de hand liggende manier om dit te doen, is het een-voor-een vergelijken van de letters van w in T. Een voorbeeld: we zoeken het woord *ananas* in de tekst T = s a s a n a n a n a n a s s a a.

De naïeve methode is weergegeven in de onderstaande tabel. In regel i checken we of het woord *ananas* begint op positie i in de tekst T. Zodra we een letter vinden die niet gelijk is, gaan we naar de volgende positie. We beginnen op positie 1 van de tekst. Aangezien de eerste letter niet klopt (het is een *s* maar moet een *a* zijn; zie regel 1 in tabel), gaan we kijken of het woord begint op positie 2. Nu is de eerste letter wel goed, maar de tweede letter niet (is een *s*, maar moet een *a* zijn, zie regel 2). Op naar positie 3, waar de eerste letter niet goed is. Dus naar positie 4. Hier zijn

de eerste vijf letters goed, maar helaas, de laatste letter is een *n* in plaats van een *s*, etcetera. Pas vanaf positie 8 vinden we het woord: alle letters zijn goed.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	s	a	s	a	n	a	n	a	n	a	n	a	s	s	a	a
1	a															
2		a	n													
3			a													
4				a	n	a	n	a								
5					a											
6						a	n	a	n	a	n					
7							a									
8								a	n	a	n	a	s			

Helaas is deze methode niet efficiënt, aangezien geen gebruik wordt gemaakt van herhalingen in het woord *ananas*: we zien bijvoorbeeld dat de letters op posities 7 en 8 wel drie keer bekeken worden. Dit is niet nodig, want door slim gebruik te maken van herhalingen in het woord hoeven alle letters uit de tekst maar eenmaal bekeken te worden. Dit is precies wat eindige automaten doen: gebruikmaken van herhalingen in een woord. Stel bijvoorbeeld dat we de letters *anana* al hebben gelezen, en de volgende letter is een *n* – dus we zien *ananan*. Dit is niet het woord *ananas*, maar we zijn wel al een eind op weg: we hebben al het beginstuk *anan* gezien, en hoeven dus alleen nog maar een *a* en een *s* te lezen. Eindige automaten onthouden dit door een toestand bij te houden, en zorgen zo voor efficiëntie.

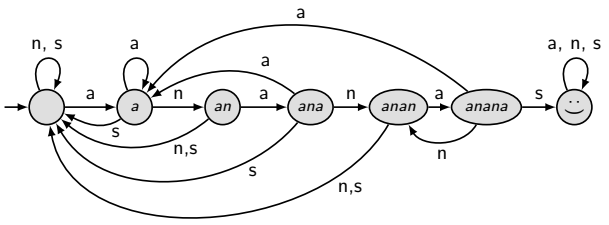

fig. 1 Voorbeeld van een eindige automaat.

Theorie van eindige automaten

Een eindige automaat (zie figuur 1 voor een voorbeeld) is een graaf waarvan de pijlen gelabeld zijn met letters uit een vooraf gegeven eindige verzameling A, die we het alfabet van de automaat noemen. Er is een begintoestand, die we aangeven met een inkomende pijl uit het niets, en een eindtoestand, die we in dit geval visueel aangeven met een smiley. Een woord w wordt gevonden in de tekst, als je door de letters van het woord w te volgen vanaf de begintoestand, in de eindtoestand terechtkomt; we zeggen in dat geval dat de automaat het woord w accepteert. De automaat uit figuur 1 accepteert dus het woord *ananas* (en ieder ander woord dat *ananas* bevat). Als we met deze automaat het woord *ananas* gaan zoeken in de tekst s a s a n a n a n a n a s s a a, dan zien we dat de tekst inderdaad geaccepteerd wordt. Bovendien hebben we iedere letter uit te tekst maar een keer bekeken, en dat is dus sneller dan bij de 'naïeve' methode van boven.

Een automaat accepteert in het algemeen meerdere woorden. De verzameling van alle woorden die door een automaat A geaccepteerd worden, heet de taal van A. De automaat uit figuur 2 accepteert oneindig veel woorden: ieder woord dat minimaal drie *a*'s bevat, wordt geaccepteerd (andere letters mogen willekeurig vaak voorkomen). De automaat uit figuur 3 accepteert alle woorden waarin het aantal *a*'s een drievoud is.

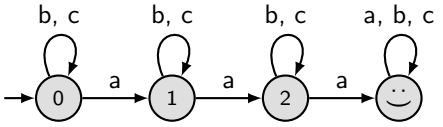

fig. 2 Een automaat die alle woorden accepteert waarin minimaal drie a's voorkomen.

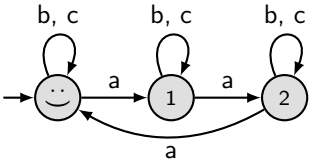

fig. 3 Een automaat die alle woorden accepteert waarin het aantal a's een drievoud is.

Formeel ziet het bovenstaande framework er als volgt uit – de onderstaande theorie hoeft uiteraard niet door de leerlingen begrepen te worden.
Definitie 1. Een eindige automaat A is een 5-tupel $(S, s_0, \Sigma, F, \Delta)$ waarbij

- S een eindige verzameling van *toestanden* is;
- $s_0 \in S$ de *initiële toestand* is;
- Σ het alfabet van A is;
- $F \subseteq S$ de verzameling van eindtoestanden is;
- $\Delta: S \times \Sigma \rightarrow S$ de toestandsovergangsfunctie is.

De betekenis van $\Delta(s, a) = s'$ is dat de automaat vanuit toestand s een a kan lezen, om vervolgens verder te gaan vanuit toestand s'. In de visualisatie (die we het toestandsdiagram noemen) tekenen we dan een pijl van s naar s' met label a.

Een pad in A is een rijtje $s_0 a_1 s_{1\ldots} a_n s_n$, waarbij $s_i \in S$, $a_i \in F$ en $\Delta(s_1, a_{i+1}) = s_{i+1}$. Het woord dat gelezen wordt met dit pad is $a_{1\ldots} a_n$. Woord w wordt geaccepteerd door A indien er een pad p in A bestaat zodanig dat p begint in de begintoestand, eindigt in een toestand uit F en het woord dat met p gelezen wordt gelijk is aan w. De taal van A is nu de verzameling van alle woorden die door A geaccepteerd worden.

Een interessante stelling in de theorie van eindige automaten is dat niet alle talen beschreven kunnen worden door automaten. Het is bijvoorbeeld onmogelijk om een automaat te construeren die de taal $L = \{a^k b^k \mid k \in \mathrm{N}\}$ accepteert. (Hierbij betekent a^k dat we k keer de letter *a* achter elkaar zetten; we bedoelen dus geen machtsverheffen. De taal L bevat in dit geval alle woorden met eerst een aantal *a*'s en dan precies hetzelfde aantal *b*'s.) Intuïtief is de reden voor deze onmogelijkheid dat de automaat die L zou accepteren moet onthouden hoeveel *a*'s hij van een woord al gelezen heeft. Echter, omdat een automaat eindig is (zeg dat hij m toestanden heeft), kan hij nooit meer dan m letters onthouden.

Stelling 1. Er bestaat geen eindige automaat die de taal $L = \{a^k b^k \mid k \in \mathrm{N}\}$ accepteert.

Bewijs. Stel, er bestaat een eindige automaat A die L accepteert. Laat m het aantal toestanden in A zijn. We bekijken het woord $w = a^m b^m$. Dit woord zit in L, dus moet er in A een pad $p = s_0 a_1 s_1 \ldots a_{2m} s_{2m}$ zijn van de begintoestand naar de eindtoestand dat woord w leest. Merk op dat dit pad $2m + 1$ toestanden heeft. Omdat A maar m toestanden heeft, moeten sommige van de toestanden van p gelijk zijn aan elkaar. Stel dat $s_i = s_j (i \neq j)$, dan bevat de eindige automaat dus een cycle vanuit de toestand s_i. Als alle letters op de overgangen binnen deze cycle de letter *a* lezen, en de cycle bijvoorbeeld n overgangen lang is, dan zou dat betekenen dat het woord $a^{m+n} b^m$ ook in de taal van A zit. Immers, we kunnen een nieuw pad construeren op basis van p, door de cycle tweemaal te doorlopen. Aangezien de aanname was dat A de taal L accepteert en $a^{m+n} b^m$ geen woord in L is, levert dit een tegenspraak op. Dezelfde redenatie gaat op voor het geval dat alle overgangen een *b* lezen. In geval dat er zowel *a*'s als *b*'s op de cycle voorkomen, zou een dubbele doorloop zelfs zorgen voor een woord waarin *a*'s en *b*'s door elkaar heen staan; ook dit soort woorden bevinden zich niet in A. Aangezien de eindige automaat die precies L accepteert in alle gevallen ook woorden accepteert die niet in L zitten, komen we altijd uit op een tegenspraak. Hieruit volgt dat er blijkbaar geen eindige automaat bestaat die de taal L accepteert.

De gastles

Hoewel de gastles gebaseerd was op de bovenstaande theorie, hebben we uiteraard niet als doelstelling gehad om een dergelijke formele definitie en redenatie over te brengen. Wat we wel wilden bereiken was het volgende:

- De leerling weet dat zoeken in tekstbestanden op meerdere manieren kan, en is zich ervan bewust dat deze manieren verschillen in efficiëntie.
- De leerling kan op een intuïtieve wijze uitleggen wat eindige automaten zijn en waar ze voor dienen.
- De leerling kan een toestandsdiagram van een eindigde automaat herkennen en interpreteren. In eenvoudige gevallen kan de leerling bepalen wat de taal is die door de automaat wordt geaccepteerd.
- De leerling kan, gegeven een of meerdere woorden, een eindige automaat construeren die deze woorden accepteert. Vervolgens kan de leerling deze woorden in een gegeven tekst zoeken met behulp van de eindige automaat.

Om deze leerdoelen te bereiken moeten we vanzelfsprekend uitgaan van een bepaalde voorkennis. Aangezien er 'gerekend' wordt met woorden, is het van belang dat leerlingen al enigszins kennis hebben gemaakt met varianten van wiskunde die niet alleen over het rekenen met getallen gaan. Met name ervaring met letterrekenen lijkt een goede basis voor dit onderwerp te zijn, aangezien leerlingen dan al enig abstractieniveau hebben bereikt en niet al te vreemd op zullen kijken van een wiskundige aanpak van problemen die niet over getallen gaan.

De gastles die we gegeven hebben, bevatte vier onderdelen:

1. Motivatie van het onderwerp (3-5 minuten);
2. Uitleg van de stof (8-10 minuten);
3. Maken en bespreken van een aantal opdrachten (30-35 minuten);
4. Terugblik (3-5 minuten).

Motivatie van het onderwerp

Het eerste gedeelte van de les bestond uit een informele introductie op het onderwerp: waarom is efficiënt zoeken belangrijk? Hierbij hebben we de bovengenoemde voorbeelden van Google, Facebook en Twitter genoemd. Door concrete statistieken te noemen over het aantal zoektermen per dag bij Google werd duidelijk hoe belangrijk efficiëntie hier is.

Uitleg van de stof

Na de introductie hebben we twee technieken behandeld om te zoeken in tekstbestanden. Eerst hebben we de naïeve methode behandeld, waarbij vanaf iedere positie gekeken wordt of het woord op die positie begint (zoals uitgelegd aan het begin van dit artikel). Door duidelijk te maken dat hierbij dubbel werk verricht wordt, kwamen we uit op een slimme methode die bijhoudt wat je al gezien hebt: eindige automaten. Van deze methode hebben we grofweg de bovenstaande theorie behandeld tot aan de formele definitie; het voorbeeld in figuur 2 maakte onderdeel uit van de opdrachten die de leerlingen hebben uitgevoerd.

Opdrachten

We hebben de leerlingen (in tweetallen) laten werken aan vier opdrachten:

- In *Opdracht 1* moesten de leerlingen de automaat die we tijdens de les hadden gemaakt (figuur 1) gebruiken om het woord *ananas* te zoeken in een gegeven tekst. Bij iedere letter uit de tekst moesten zij de positie uit de automaat noteren die ze tegenkwamen bij het zoeken. Deze opdracht laat leerlingen werken met de theorie, en stelt hen in staat om de theorie beter te begrijpen.
- *Opdracht 2* vroeg de leerlingen een automaat te maken die het woord *cacao* accepteert. Dit gaat op dezelfde manier als de automaat voor het woord *ananas*, en leverde naar verwachting geen problemen op.
- In *Opdracht 3* werd gevraagd om een automaat te construeren die uitvindt of een tekst het woord *pen*, het woord *nep* of beide woorden bevat. Hiervoor moesten leerlingen creativiteit gebruiken om op de goede automaat uit te komen. Ongeveer de helft van de leerlingen kwam zelf tot het juiste antwoord, de andere helft had een hint nodig, maar uiteindelijk kwam men er wel uit.
- *Opdracht 4* vroeg om een automaat te maken die uitzoekt of een tekst minimaal drie *a*'s bevat. Aangezien deze *a*'s niet achter elkaar hoeven te staan, werd dit lastig gevonden. Een aantal leerlingen zag toch onmiddellijk wat de bedoeling was, een aantal anderen kwam er met een hint wederom uit.

We hadden een aantal antwoordbladen uitgedeeld, waarop de opgaven vermeld stonden en leerlingen hun automaten konden tekenen. Vervolgens werden deze klassikaal besproken, en werden uiteindelijk de juiste antwoorden door middel van een beamer geprojecteerd.

Terugkijken/Reflectie

Juist omdat het een gastles vanuit de universiteit betrof, vonden we reflectie belangrijk: wat hebben we geleerd? Werkt de methode echt? En: is dit wel wiskunde?

Evaluatie. De leerlingen hadden geen enkele moeite om de theorie te volgen en kwamen – hier en daar met enige hulp – goed uit de opdrachten. Omdat de stof niet tot de standaard examenstof behoort, waren echter niet alle leerlingen gemotiveerd om er ook mee aan de slag te gaan. Leerlingen die meteen meededen, vonden het een grappig onderwerp, vooral omdat het laat zien dat je ook zonder getallen leuke wiskunde kunt doen.

Uiteindelijk denken we dat de leerdoelen behaald zijn. De leerlingen hebben inderdaad kennisgemaakt met verschillende zoekstrategieën, en leken begrepen te hebben dat het zoeken door middel van eindige automaten efficiënter is dan de naïeve methode. Bovendien hadden leerlingen na de uitleg geen moeite met het hanteren van een eindige automaat om te zoeken in een tekst. Het construeren van een automaat die een aantal gegeven woorden accepteert, is uiteindelijk ook iedereen gelukt. Het oefenen met het bepalen van de taal van een gegeven automaat is niet uitgebreid aan de orde gekomen; hier zou eventueel in een vervolgles nog aandacht aan besteed kunnen worden.

We denken dat de leerlingen na deze les een interessant kijkje hebben kunnen nemen in de keuken van de geavanceerdere wiskunde. Hoewel de uitleg uiteraard nog niet heel ingewikkeld was, hebben leerlingen zo al wel eens kennisgemaakt met een vorm van wiskunde die ze normaal gesproken op de middelbare school nog niet zien.

Waar kom je eindige automaten nog meer tegen?
Zoals gezegd zijn eindige automaten een van de meest fundamentele wiskundige modellen binnen de informatica. Ze worden onder andere gebruikt bij parseren, dat wil zeggen het omzetten van computerprogramma's (bijvoorbeeld in C, PHP of Java) naar instructies voor de computer. Daarnaast zijn ze belangrijk om processen en protocollen te modelleren. Als je bijvoorbeeld een bericht over het internet wilt versturen, dan moeten de instructies in een bepaalde volgorde worden aangeroepen. In welke volgorde dat moet, wordt beschreven door een automaat. De populaire modelleertaal UML bevat bijvoorbeeld dit soort automaten (State Charts geheten).

Beschikbaar materiaal. Wij hebben de gastlessen gegeven aan de hand van een presentatie in PowerPoint. Voor het maken van de opdrachten waren antwoordvellen beschikbaar; deze waren van te voren geprint. Dit materiaal[1] mag gebruikt worden tijdens de lessen, mits het copyright notice (ontwikkeld door Mariëlle Stoelinga aan de Universiteit Twente) wordt vermeld.

Mariëlle Stoelinga,
Universitair docent informatica, Universiteit Twente
Mark Timmer,
promovendus, Universiteit Twente

Noot

[1] Presentatie:
www.cs.utwente.nl/~marielle/papers/gastcollege.ppt
Antwoordvellen:
www.cs.uwente.nl/~marielle/papers/antwoordvellen-gastcollege.pdf