Het harde probleem van de handelsreiziger

Bart Demoen *

January 3, 2008

Een handelsreiziger lost het probleem dagelijks op: bezoek een aantal (potentiele) klanten, keer 's avonds terug waar je 's morgens vertrok en leg daarbij zo weinig mogelijk afstand af. De benaming *handelsreizigersprobleem* ligt dan ook voor de hand, maar ook niet-handelsreizigers hebben met het probleem te maken. Een meer algemene formulering van het probleem is:

> **Gegeven:** een eindig aantal knooppunten met daartussen verbindingen met hun lengte, t.t.z. een gewogen graaf
>
> **Gevraagd:** een kring die alle knopen juist één keer bevat en een minimale totale lengte heeft

Een kring die alle knopen juist één keer bevat, heet een Hamiltoniaanse kring (HK), omdat Sir W. Hamilton, een Ierse wiskundige uit de 19de eeuw, als eerste het concept formaliseerde, bestudeerde en zelfs commercieel exploiteerde.

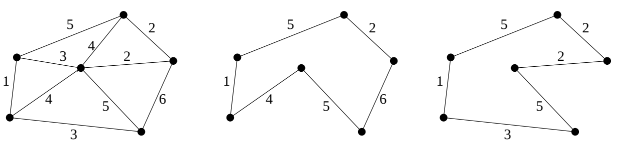

Figure 1: Links een graaf met 6 knopen en 10 verbindingen; in het midden een HK voor die graaf; rechts een kortste HK.

Het recente boek *The Traveling Salesman Problem - A Computational Study* van David Applegate et al. beschrijft waar het probleem zoal voorkomt in de praktijk. Soms gaat het gewoon om variaties op de handelsreiziger: toeristen die op één dag zoveel mogelijk attracties willen bezoeken, een schoolbus die zo snel mogelijk alle kinderen moet afzetten of een postbode die zijn toer zo kort mogelijk wil maken. Andere toepassingen komen uit de productie, bijvoorbeeld van printplaten: daarin wordt een groot aantal gaten geboord door één boorkop. Die boorkop beweegt boven de printplaat van boorgat naar boorgat.Het productieproces kan verkort worden door een goede volgorde van de boorgaten te kiezen. Het verband met het handelsreizigersprobleem is direct: modelleer de boorgaten als de knopen van een graaf, en de tijd die het neemt om de boorkop van het ene naar het andere boorgat te laten bewegen als de lengte van de verbinding tussen die twee knopen. Een kortste kring geeft nu een optimale volgorde waarin de boorgaten moeten gemaakt worden. Een gelijkaardige toepassing is te vinden in een sterrenobservatorium waar tijdens één nacht een aantal sterren geobserveerd worden, en waarbij de bewegingen van de telescoop van de ene naar de andere ster best geminimaliseerd worden. Toepassingen zijn echt overal te vinden, van het reduceren van verlies bij behangpapier, tot het organiseren van playlists op een mp3-speler. En toepassingen van TSP bestonden zeker al in de oudheid, want ook toen wilden reizigers hun reisweg optimaliseren.

In 1832 vinden we het probleem expliciet terug in een Duits handboek *Der Handlungsreisende ...* waarin een *alte Commis-Voyageur* raad geeft aan zijn collegas handelsreizigers o.a. over hoe een geschikte route te plannen. De eerste keer dat naar het geformaliseerde probleem verwezen werd als *het handelsreizigersprobleem*, of eigenlijk het *Traveling Sales-*

*Dept. of Computer Science, K.U. Leuven, Belgium, bmd@cs.kuleuven.ac.be

man Problem (TSP), is tijdens de jaren 1930 aan de universiteit van Princeton, en die benaming is blijven hangen.

Het TSP kan zo gemakkelijk in één zin geformuleerd worden en het is zo gemakkelijk te vatten, dat er toch zeker ook een gemakkelijke oplossing moet bestaan! Inderdaad: voor een eindig aantal steden bestaat maar een eindig aantal kringen; bereken de lengte van elke kring en kies de kortste. Dit geeft ons een algoritme dat TSP oplost ... althans in principe. Voor 5 steden zijn er maximaal 12 kringen en een TSP met 5 steden kan je dus gemakkelijk met de hand oplossen. Voor 10 steden zijn er hoogstens 181.440 kringen en dat is een klein getal voor onze computers. Maar voor 20 steden zijn er potentieel 60.822.550.204.416.000 kringen en die allemaal genereren ligt ver buiten onze rekencapaciteit. Dat hoeft ons niet af te schrikken: misschien kan bovenstaand *brute kracht* algoritme verbeterd worden. Maar wat betekent het eigenlijk dat één algoritme beter is dan een ander?

Complexiteitstheorie bestudeert de kwaliteit van een algoritme en de intrinsieke moeilijkheid om bepaalde problemen op te lossen. In oorsprong gaat het om beslissingsproblemen, t.t.z. vragen van de vorm *heeft een gegeven ding een bepaalde eigenschap?* Bijvoorbeeld: is dit getal deelbaar door 9? of, is deze rij getallen stijgend geordend? of, heeft deze graaf een Hamiltoniaanse kring? Er bestaat niet altijd een algoritme voor een beslissingsprobleem. Zo is de vraag *stopt dit programma* niet te beantwoorden voor een willekeurig gegeven programma, met behulp van een algoritme: het is het fameuze *Halting Probleem.* Maar als een beslissingsprobleem een algoritme heeft, dan heeft het altijd veel algoritmen: zo kan je voor de deelbaarheid door 9 de deling echt uitvoeren en nagaan of de rest 0 is; of je kan de som module 9 maken van de cijfers van het getal in decimale voorstelling zoals in de negenproef. Van één vast algoritme kunnen we nu bestuderen hoeveel elementaire stappen het nodig heeft in functie van de grootte van de invoer. In het geval van deelbaarheid door 9, nemen we het aantal cijfers N in de decimale voorstelling van het getal als grootte van de invoer. Het uitvoeren van de deling door 9 kost dan N keer een deling, een vermenigvuldiging en een aftrekking: het deel-door-9-algoritme noemen we dan ook *lineair* in N en we noteren dat door $O(N)$. Die notatie verbergt constante factoren, maar de *lineaire-versnellings-stelling* zowel als de verwachting dat onze volgende PC minstens 2 keer sneller zal zijn, maakt de constante factor minder interessant. Vermenigvulding van twee getallen met elk N cijfers zoals we het op school leerden gebruikt een kwadratisch aantal bewerkingen: elk van de N cijfers van het ene getal wordt met elk van de N cijfers van het andere getal vermenigvuldigd, en dan zijn er nog wat optellingen: in het totaal $O(N^2)$. We zeggen: het algoritme is van orde N-kwadraat, of ook nog het algoritme heeft een kwadratische complexiteit. Elk algoritme heeft zijn eigen complexiteit, en we vergelijken algoritmen door hun complexiteitsfunctie te vergelijken: zo vinden we een algoritme van orde N^3 slechter dan eentje van orde N^2, want vanaf een zekere N is N^3 groter dan N^2. Zo bestaat voor (bijna) elk beslissingsprobleem een *beste* algoritme, ruwweg het algoritme met de complexiteitsfunctie die het minst snel stijgt. Een algoritme met een complexiteitsfunctie die een veelterm is, heet een polynomiaal algoritme. Een probleem met een polynomiaal algoritme noemen we een polynomiaal probleem. Zo is vermenigvuldingen van twee getallen een polynomiaal probleem, en ook het

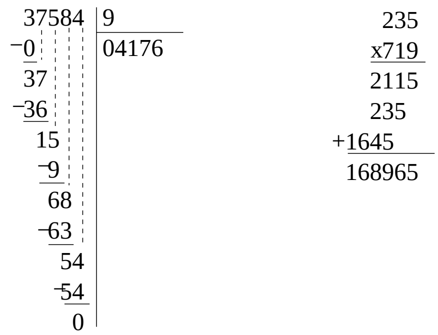

Figure 2: Links: testen of een getal met 5 cijfers deelbaar is door 9; 5 keer delen, vermenigvuldigen en aftrekken. Rechts: 2 getallen van 3 cijfers vermenigvuldigen; 9 elementaire vermenigvuldigingen.

sorteren van een rij getallen, of het bepalen van de kortste afstand tussen twee steden (iets dat ViaMichelin coortdurend doet). Polynomiale problemen hebben algoritmen die ook voor grote invoer door computers snel kunnen beslist worden - tenminste als de graad van de polynoom niet te hoog is. De verzameling van polynomiale problemen noteert men door P.

Terug naar onze handelsreiziger: het naieve algoritme dat van elke kring de lengte berekent, heeft complexiteit $N!$[1] waarbij N het aantal steden is. Bij het maken van printplaten is het aantal te boren gaten al snel een paar honderd ... het naieve algoritme is duidelijk niet adequaat voor zulke waarden van N. We hadden graag een *goed*, dus polynomiaal algoritme voor TSP. En daar wringt het schoentje: de enorme inspanning van hordes wetenschappers heeft eigenlijk niks opgeleverd, geen polynomiaal algoritme voor TSP, geen bewijs dat een polynomiaal algoritme voor TSP bestaat en ook geen bewijs dat het niet bestaat! Voor we dit uitspitten moeten we het verband tussen TSP en het geassocieerd TSP-beslissingsprobleem bekijken:

Gegeven: een gewogen graaf en een getal L

Gevraagd: bestaat een Hamiltoniaanse kring die kleiner is dan L?

Op deze vraag is enkel ja of nee een zinnig antwoord: het is dus een beslissingsprobleem. Stel dat je een goed algoritme had om het TSP-beslissingsprobleem op te lossen, dan zou je voor TSP als volgt een algoritme kunnen bouwen: los het TSP-beslissingsprobleem op voor $L = 1$, dan voor $L = 2$, dan voor $L = 3$ enzovoort tot je *ja* als antwoord krijgt. Nu heb je in ieder geval de lengte van de kleinste kring - hiermee is TSP nog niet opgelost, maar het is al een stap in de goede richting. Maar weerom: voor het TSP-beslissingsprobleem hebben we geen goed algoritme, t.t.z. men weet niet of het in P zit!

Maar het TSP-beslissingsprobleem heeft wel een zeer speciale eigenschap: als ik je wil overtuigen dat een gegeven gewogen graaf met N knopen een HK heeft die kleiner is dan L, dan geef ik je gewoon zulk een HK. Jij kan dan heel eenvoudig nakijken dat mijn kring inderdaad een HK is, en kleiner dan L. De HK die ik je gaf is een *certificaat* voor mijn ja-antwoord. Het certificaat is in dit geval een opeenvolging van N knopen, dus lineair in N, of $O(N)$. Jouw algoritme om mijn certificaat na te kijken is ook linear in N. Dus, voor het TSP-beslissingsprobleem bestaan polynomiale certificaten.[2] De verzameling van alle beslissingsproblemen met een polynomiaal certificaat noteren we met NP, hetgeen staat voor *niet-deterministisch polynomiaal*, een terminologie die vanuit een equivalente alternatieve beschrijving van die verzameling komt. Als verzameling is P duidelijk een deel van NP.

De moeilijkste problemen in NP worden NP-compleet genoemd en één van de meest bekende NP-complete problemen is ... het TSP-beslissingsprobleem. Het is relatief gemakkelijk aan te tonen dat indien het TSP-beslissingsprobleem tot P behoort, dat dan $P = NP$. De vraag of $P = NP$ houdt ons echter al vele jaren in de ban, en is zelfs één van de zeven *Millenium Problemen* waarvoor het Clay Mathematics Institute of Cambridge, Massachusetts een prijs uitloofde van 1 millioen US$. TSP zelf wordt *NP-hard* genoemd: elk NP-probleem kan ernaar gereduceerd worden.

Het lijkt er dus op dat er niet snel een goed, polynomiaal algoritme voor TSP zal komen, en toch is het in de praktijk belangrijk dat voor netwerken met honderden steden optimale kringen worden gevonden. Het begon natuurlijk wat kleiner ...

TSP werd waarschijnlijk bekend bij het grote publiek in het begin van de jaren 1960, toen Proctor & Gamble als reklamestunt een wedstrijd uitschreef waarbij een TSP voor 33 steden in de USA moest worden opgelost. De prijs voor de winnaar was 10.000 US$, een zeer hoog bedrag voor die tijd. Daarvoor hadden in 1954 een aantal wiskundigen met behulp van technieken uit lineair programmeren, al een 49 steden TSP opgelost. Het duurde 17 jaar voor dit aantal op 57 steden kwam, en het groeide gestaag naar 2.392 in 1987. Ondertussen heeft de gemeenschap van TSP-onderzoekers een hele testbatterij van gewogen grafen aangelegd, en die werden verzameld

[1] Die ! is de faculteitsfunctie: $1! = 1$, en $N! = N * (N - 1)!$

[2] Merk op dat we hier alleen polynomiale certificaten hebben om een *ja* te certifieren!

in wat nu bekend staat als TSPLIB: elk nieuw idee om TSP sneller te implementeren wordt daarop uitgetest. Sinds de jaren 1990 steekt één systeem boven alle andere uit: Concorde. De auteurs van het boek zijn ook de auteurs van dit systeem dat overigens vrij kan gedownload worden voor academisch onderzoek. Ondertussen loste Concorde in 2004 TSP op voor Zweden, t.t.z. een kortste kring door de 24.978 steden in Zweden: de berekening gebruikte bijna 85 jaar computertijd, verdeeld over 96 machines. In 2006 werd zelfs een TSP met 85.900 *steden* in een VLSI-toepassing opgelost. Zulke aantallen steden liggen ver buiten het bereik van het naieve algoritme, en zelfs van het beste bekende algoritme van Held en Karp, dat nog altijd van de orde $N^2 2^N$ is. Concorde combineert dan ook een aantal methoden en heuristieken, en maakt bovendien gebruik van ondergrenzen op de gezochte korste kring. Het concept van ondergrens is interessant, want die geeft aan dat geen enkele kring korter kan zijn en geeft daardoor greep op hoeveel een reeds gevonden oplossing nog van de optimale kan verschillen. Het is ook interessant hoe eenvoudig die krachtige ondergrenstechniek kan geillustreerd worden. Figuur 3 laat zien hoe op de knopen van de graaf schijfjes zijn gelegd zodanig dat de schijfjes niet overlappen; de som van de diameters van de schijfjes is nu een ondergrens voor de lengte van elke Hamiltoniaanse kring. Als we rond een groep schijfjes nu ook nog een gracht leggen - zoals in de rechterkant van Figuur 3 - dan krijgen we in vele gevallen al een optimale ondergrens!

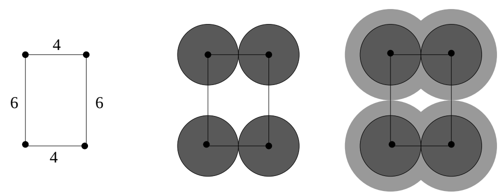

Figure 3: Links een gewogen graaf; in het midden liggen er schijfjes op de knopen; rechts liggen er nog grachtjes rond.

Ondertussen zijn er voor verschillende landen optimale HK's berekend: ze zijn te vinden op `http://www.tsp.gatech.edu/world/countries.html`. De volgende uitdaging is een wereldtoer: een wereldkaart met 1.904.711 steden werd aan Concorde gegeven. Zelfs de auteurs van Concorde verwachten niet dat hun technieken voldoende zijn om een optimale HK te vinden, maar het bestuderen van zulke grote problemen leidt tot nieuwe inzichten en algorithmen. Ondertussen is er een benadering voor de wereldtoer tot op minder dan 0,06% gevonden!

TSP is zo moeilijk in het algemeen op te lossen, dat het zin heeft om varianten van TSP te onderzoeken, in de hoop dat die gemakkelijker zijn dan het oorspronkelijke probleem. Een eerste voor de hand liggende variant is die waarbij de steden meer dan eens mogen bezocht worden: die variante is nog altijd NP-hard. Misschien wordt het eenvoudiger als we veronderstellen dat de graaf in het vlak ligt en de afstanden Euclidisch zijn? Dat helpt zeker om sommige kringen uit te sluiten, want met de driehoeksongelijkheid kan je aantonen dat een minimale HK geen kruisende wegen heeft. Maar deze variante is nog altijd NP-hard. Misschien kan er efficient een willekeurig dichte benadering van een optimale kring berekend worden: voor heel wat andere NP-complete problemen bestaan inderdaad polynomiale benaderingsalgoritmen, maar spijtig genoeg niet voor TSP. Voor de Euclidische variant bestaan zulke benaderingsalgoritmen dan weer wel.

TSP is ook bestudeerd in andere contexten dan comlexiteitstheorie: zo hebben psychologen het gebruikt om de evolutie te bestuderen van perceptuele en cognitieve vaardigheden van kinderen tot volwassenen. Artiesten hebben het gebruikt om interessante tekeningen te produceren. En zelfs chimpansees en duiven hebben in experimenten mogen tonen hoe goed ze TSP kunnen oplossen.

TSP blijkt een universele uitdagende inspiratie, en lijkt dat nog lang te blijven.