DNA Computing
Computer in de Reageerbuis

H.J. Hoogeboom G. Rozenberg

2 juli 2002

De PC die op ons bureau staat heeft een kloksnelheid die destijds bij aanschaf onwaarschijnlijk hoog leek. Inmiddels staan bij ALDI computers in de advertentie die tien maal zo snel rekenen. De vooruitgang laat zich niet zo makkelijk voorspellen. Toch moeten we er rekening mee houden dat in de nabije toekomst de grenzen van de huidige technologie bereikt worden. Immers, bij verregaande miniaturisatie gaan quantumeffecten een rol spelen, terwijl de communicatie tussen samenwerkende processoren begrensd wordt door de lichtsnelheid. Het is daarom nodig om nieuwe technieken te verkennen. *DNA computing* is zo'n voorgestelde drastisch nieuwe technologie, en het onderwerp van dit hoofdstuk. De gebruikte hardware bestaat uit DNA moleculen, de daarop uitgevoerde operaties komen uit de gereedschapskist van de biotechnologen. De verwachtingen die gewekt worden zijn enorm: een klein reageerbuisje DNA kan zo'n 10^{15} moleculen bevatten die we allemaal tegelijkertijd kunnen bewerken. Bovendien kunnen geschikt gekozen kleine strengen DNA samengevoegd in een reageerbuis langere strengen DNA vormen met geplande eigenschappen (*self assembly*). De New Scientist kopte 'the ultimate computer: a trillion calculations in parallel'. Dit ter ere van het door Adleman in 1994 in Science beschreven experiment: het oplossen van een eenvoudig soort handelsreizigersprobleem, zonder de computer aan te zetten, geheel in het biochemisch laboratorium.

Inhoudsopgave. Rekenen met DNA, en in het bijzonder het genoemde experiment van Adleman staan centraal in dit hoofdstuk. We proberen te kijken zowel naar de praktijk (de handelingen in het laboratorium) als naar de onderliggende theorie (het wiskundige model van een computer).

We beginnen in paragraaf 1 met het beschrijven van DNA, de gebruikte hardware. We besteden aandacht aan een aantal laboratoriumtechnieken die het mogelijk maken DNA te manipuleren. In paragraaf 2 beschrijven we het

experiment van Adleman. Hij liet zien dat een eenvoudig voorbeeld van een lastig probleem opgelost kan worden door berekeningen uit te voeren met behulp van DNA. Dat probleem is het vinden van een pad door een netwerk dat alle knopen in dat netwerk precies één keer aandoet. Het staat bekend als HPP, het Hamilton Pad Probleem.

De noodzaak tot het uitdenken van nieuwe rekentechnieken komt aan de orde in paragraaf 3. We bespreken daar complexiteit van problemen, en het bestaan van praktisch gemotiveerde problemen waarvoor geen efficiënte oplossingsmethoden bekend zijn. Eén van deze problemen is het genoemde HPP.

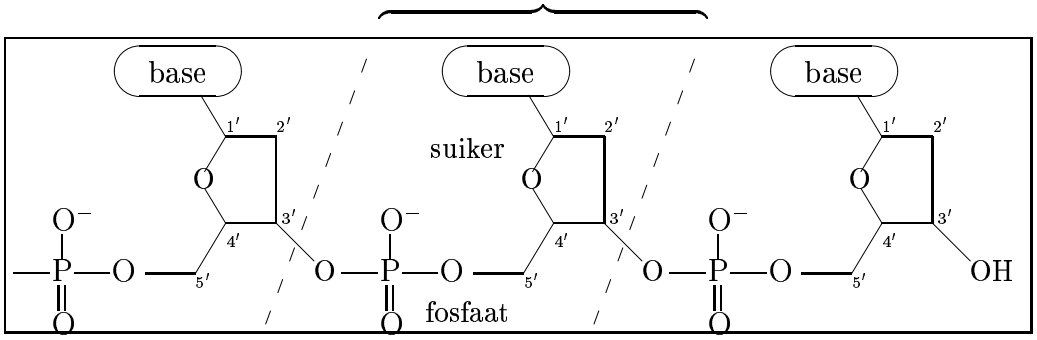
We sluiten af met paragraaf 4, waar we ingaan op *Natural Computing*, het gebied van de informatica waar op de natuur gebaseerde (eigenlijk van de natuur afgekeken) rekenmethoden worden bestudeerd.

1 DNA, Bioware

1.1 Structuur van DNA

Het DNA in ons lichaam wordt gebruikt om erfelijke informatie op te slaan. DNA is een polymeer, een lang en stabiel molecuul dat bestaat uit geschakelde kleinere elementen die elk een eenheid van informatie kunnen dragen. Deze eigenschap lijkt sterk op het computergeheugen, een lange reeks simpele bits rijgt zich samen tot een grote hoeveelheid data. Hierin ligt de inspiratie tot het gebruik van DNA als *hardware*, of misschien beter gezegd, als *bioware* In deze paragraaf beschrijven we de basiseigenschappen van DNA, niet in volledig chemisch detail, maar voldoende om het experiment van Adleman te kunnen volgen. We geven ook een aantal operaties die in het biochemisch laboratorium op DNA kunnen worden uitgevoerd. Ze vormen de gereedschappen waarmee we DNA kunnen manipuleren om er berekeningen mee uit te voeren.

De basiscomponent waaruit DNA (*desoxyribonucleïnezuur*) is opgebouwd is een nucleotide. Zo'n nucleotide bestaat uit drie componenten: suiker (de *desoxyribose* uit de naam), fosfaat en base. De koolstofatomen van de suiker worden genummerd van $1'$ tot $5'$, om ze van elkaar te kunnen onderscheiden. Zo is de fosfaatgroep verbonden aan de $5'$ koolstof van het suiker gedeelte, en de base is verbonden aan de $1'$ koolstof. Enkelstrengs DNA bestaat uit een reeks nucleotiden, onderling verbonden via de fosfaatgroepen. De fosfaatgroep van de ene nucleotide bindt aan de $3'$ koolstof van de volgende nucleotide; dit is een zgn. covalente binding die zeer sterk is. Een zo gevormde keten van afwisselend suiker en fosfaat vormt de ruggengraat, *backbone*, van

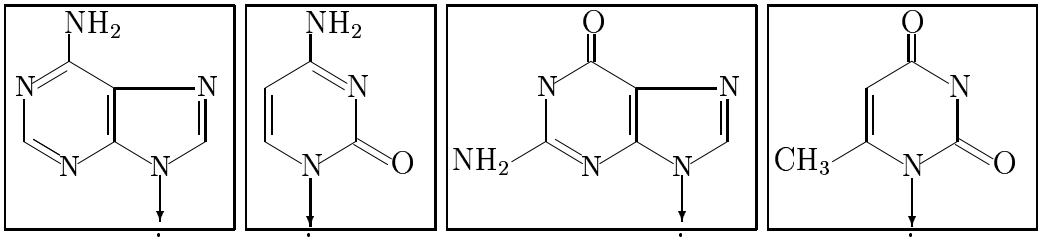


Figuur 1: Drie geschakelde nucleotiden in DNA. Getekend is de ruggengraat van suiker en fosfaat. De vier mogelijke basen staan in Figuur 2.

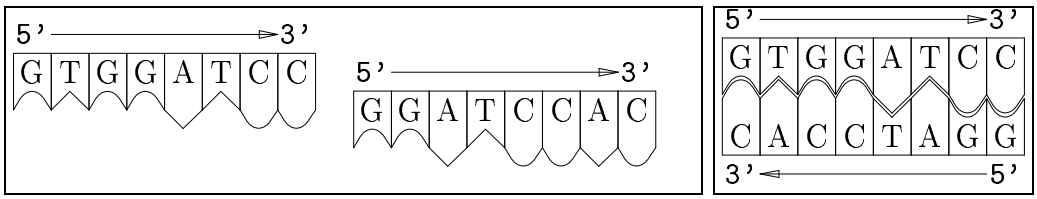
het DNA, zie Figuur 1. Merk op dat een streng DNA aan de ene zijde een $3'$ koolstof beschikbaar heeft voor verdere uitbreiding van het molecuul, en aan de andere zijde een fosfaatgroep aan de $5'$ koolstof. De chemische eigenschappen van deze beide uiteinden verschillen, waardoor een streng DNA een natuurlijke, chemisch herkenbare, oriëntatie heeft. We onderscheiden daarom het $5'$ en het $3'$ uiteinde.

Er zijn vier basen die van nature in DNA voorkomen: adenine, cytosine, guanine, en thymine, getekend in Figuur 2. De base is het enige waarin de nucleotiden verschillen; hierna gebruiken we dan ook de baseletters `A`, `C`, `G` en `T` om de vier soorten nucleotiden aan te geven. De belangrijkste eigenschap van de basen in deze context is hun *Watson-Crick complementariteit*: adenine bindt met thymine, en guanine met cytosine, door het vormen van waterstofbruggen. Deze waterstofbruggen zijn relatief zwak.

Twee enkele strengen DNA kunnen zich samenvoegen tot een dubbelstrengs molecuul doordat hun basen zich verbinden, vooropgesteld dat de opeenvolgende basen complementair zijn. Zo past de enkele streng `5'-GTGGATCC-3'` op de enkele streng `3'-CACCTAGG-5'`, de opeenvolgende basen zijn stuk voor stuk complementair en vormen dus paren `G-C` en paren `A-T`. Let op, de enkele strengen voegen zich 'anti-parallel' samen, de nucleotiden van de ene streng in $5'$-$3'$ richting rijgen zich aan de opeenvolgende nucleotiden van de andere



Figuur 2: De vier basen van DNA: adenine, cytosine, guanine, thymine



Figuur 3: Annealing: de enkele strengen `5'-GTGGATCC-3'` en `5'-GGATCCAC-3'` combineren tot één dubbele streng. De enkele strengen zijn complementair.

streng in $3'$-$5'$ richting, zie Figuur 3. De vorming van dubbelstrengs DNA uit complementair enkelstrengs DNA is een spontaan proces (bekend als *annealing*) wanneer de omstandigheden (oplosmiddel en temperatuur) geschikt zijn. Het feit dat de waterstofbrug die de complementaire strengen bindt veel zwakker is dan de koppeling die de opeenvolgende nucleotiden bijeenhoudt is een belangrijke eigenschap die wordt uitgebuit bij *genetic engineering*. Om die reden immers kan dubbelstrengs DNA gesplitst worden in enkele strengen (*denaturisatie*), chemisch of door verhitting; de enkele strengen vallen dan niet uiteen.

1.2 Notatie

Omdat de informatie, de reeks basen, van een DNA-streng in twee richtingen gelezen kan worden, zijn er twee schrijfwijzen voor hetzelfde molecuul. De enkele streng `5'-GTGGATCC-3'` is dezelfde als die beschreven wordt door `3'-CCTAGGTG-5'`. De natuurlijke afleesrichting, dat wil zeggen de richting waarin de genetische informatie afgelezen wordt, is de $5'$-$3'$ richting. Het is daarom gewoonte om die richting ook te gebruiken wanneer een streng DNA beschreven wordt, dus de notatie `GTGGATCC` beschrijft onze enkele streng. Dubbelstrengs DNA wordt dan beschreven door de twee strengen onder elkaar te schrijven. We moeten ons daarbij wel realizeren dat de twee strengen tegengestelde oriëntatie hebben. Afspraak is nu dat een van de strengen bovenaan in de natuurlijke $5'$-$3'$ richting genoteerd wordt, en de andere streng in de (complementaire) $3'$-$5'$ richting daaronder. Zo wordt het dubbelstrengs molecuul uit Figuur 3 genoteerd als $\begin{smallmatrix}\texttt{GTGGATCC}\\ \texttt{CACCTAGG}\end{smallmatrix}$, maar natuurlijk ook als $\begin{smallmatrix}\texttt{GGATCCAC}\\ \texttt{CCTAGGTG}\end{smallmatrix}$, omdat elk van de strengen van het DNA molecuul bovenaan geschreven kan worden. Het kan voorkomen dat beide schrijfwijzen overeenkomen, namelijk als beide enkele strengen (in dezelfde oriëntatie gelezen) aan elkaar gelijk zijn. In dat geval heet het molecuul een *palindroom*. Een voorbeeld hiervan is $\begin{smallmatrix}\texttt{GGATCC}\\ \texttt{CCTAGG}\end{smallmatrix}$.

Wanneer we straks ter illustratie stukjes DNA geven, tekenen we daarbij aan dat deze het principe illustreren, maar vanuit realistisch oogpunt in het algemeen te kort zijn.

1.3 Operaties

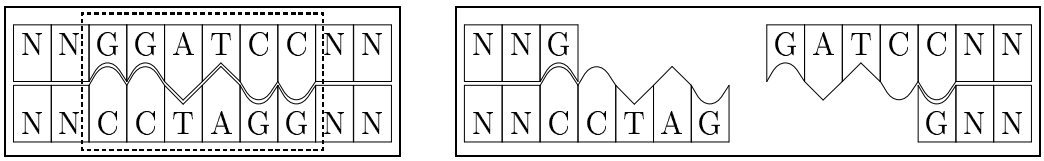
De hierboven voorgestelde bioware moet gemanipuleerd worden om met de gerepresenteerde gegevens berekeningen te kunnen uitvoeren. Daarvoor zijn een groot aantal operaties beschikbaar, afkomstig uit de DNA-gereedschapskist van de biochemicus. De industriële belangstelling voor *genetic engineering* (inclusief het *human genome project*) lijkt garant te staan voor een gestage uitbreiding en perfectionering van deze methoden en technieken.

Annealing en denaturisatie. Hierboven beschreven we al annealing, of hybridisatie, het spontaan samengaan van twee complementaire enkelstrengs DNA moleculen. Dit hoeft niet (zoals hierboven gesuggereerd) per se te leiden tot een perfect dubbelstrengs molecuul. Ook `5'-GTGGATCC-3'` en `5'-ATCC-3'` paren samen tot $\begin{smallmatrix}\texttt{GTGGATCC}\\ \texttt{CCTA}\end{smallmatrix}$. Het omgekeerde proces denaturisatie splitst een dubbelstrengs molecuul in twee enkelstrengs moleculen, zonder dat deze enkele strengen in losse nucleotiden uiteen vallen, zoals gezegd doordat de waterstofbruggen tussen de twee strengen zwakker zijn dan de bindingen in de ruggengraat. Denaturisatie kan in het laboratorium uitgevoerd worden door chemische stoffen aan de oplossing toe te voegen, of door de temperatuur van de oplossing voldoende te verhogen.

Knippen. Er zijn in de natuur veel enzymen die operaties op DNA kunnen uitvoeren. Deze worden ook in het laboratorium gebruikt. Een bepaald soort, *restrictie endonucleases*, wordt gebruikt om DNA te knippen[1]. In het algemeen herkent zo'n enzym een specifiek stuk dubbelstrengs DNA aan zijn reeks baseparen en knipt binnen deze herkenningsreeks het DNA molecuul; andere enzymen knippen juist weer buiten de herkenningsreeks. De meeste enzymen knippen niet 'recht' maar laten aan beide zijden een kenmerkend stuk enkelstrengs DNA achter aan de dubbelstrengs helften: *sticky ends*, zo genoemd omdat ze graag weer combineren met hun complementaire wederhelft. Zo herkent het enzym *BamHI*[2] de reeks $\begin{smallmatrix}\texttt{GGATCC}\\ \texttt{CCTAGG}\end{smallmatrix}$ en klieft het dubbelstrengs DNA aldaar aan beide zijden tussen de twee `G` aan de $5'$ zijde. Zo ontstaan twee

[1] restrictie is zo genoemd omdat met deze operatie vijandig DNA de toegang ontzegd wordt.

[2] *BamHI* is genoemd naar de "*Bacillus amyloliquefaciens H*" het organisme dat dit enzym produceert. Uitgebreide informatie over enzymen, hun herkenningsreeks en hun knipgedrag is te vinden op `http://rebase.neb.com/`.



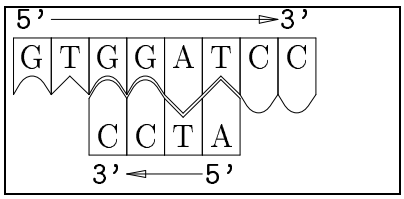
Figuur 4: Knippen. Enzym *BamH*I herkent `5'-GGATCC-3'` en klieft de dubbele streng aldaar met *sticky end* `5'-GATC-3'`. De 'N' staan voor ongespecificeerde nucleotiden.

(gelijke) sticky ends $\overset{\texttt{G}}{\underset{\texttt{CCTAG}}{}}$ en $\overset{\texttt{GATCC}}{\underset{\texttt{G}}{}}$, zie Figuur 4. Merk op dat de herkende reeks en de manier van knippen symmetrisch zijn. De herkenningsreeks vormt een palindroom; dit is het geval bij de meerderheid van de bekende restrictie enzymen.

Ligatie. Een ander enzym, ligase, is in staat om de fosfaatkoppeling tussen opeenvolgende nucleotiden in een streng DNA te herstellen wanneer deze niet meer aanwezig is. De sticky ends van bijvoorbeeld $\overset{\texttt{ATCA}}{\underset{\texttt{TA}}{}}$ en $\overset{\texttt{T}}{\underset{\texttt{GTA}}{}}$ zullen zich samenvoegen tot het molecuul $\overset{\texttt{ATCAT}}{\underset{\texttt{TAGTA}}{}}$. Maar de twee samenstellende delen van dit molecuul worden alleen samengehouden door de relatief zwakke waterstofbruggen tussen de basen. De fosfaatbanden in de ruggengraat van het DNA molecuul op de plek waar de twee delen werden samengevoegd, aangegeven door de driehoekjes in $\overset{\texttt{ATCA}\triangledown\texttt{T}}{\texttt{TA}\triangle\texttt{GTA}}$, vormen zich niet spontaan. Het enzym ligase herstelt de missende fosfaatbanden en vormt zo een stabiel dubbelstrengs DNA-molecuul.

Selectie op een patroon. Een veel gebruikte operatie op DNA is de selectie van strengen DNA die een bepaalde reeks basen bevatten. Het principe hiervoor is eenvoudig. Wanneer we bijvoorbeeld willen selecteren op de reeks `5'-GGAT-3'` wordt de complementaire reeks `5'-ATCC-3'` gesynthetiseerd. Deze complementaire strengen worden dan vastgehecht op microscopisch kleine glazen balletjes, die vervolgens in een buis dicht opeen worden gepakt. Wanneer we een oplossing DNA door deze buis laten stromen binden de moleculen met de reeks `5'-GGAT-3'` zich aan het complementaire stukje DNA `5'-ATCC-3'` op de balletjes. We hebben nu de oorspronkelijke oplossing DNA gesplitst in twee delen: het deel zonder `5'-GGAT-3'` is doorgestroomd, en het deel met `5'-GGAT-3'` is in de buis achtergebleven, en kan daaruit losgemaakt worden door te spoelen met denaturatiechemicaliën.

Selectie op lengte. Een van de meest uitgevoerde operaties op DNA is de selectie op lengte, waarbij de lengte van een molecuul het aantal nucleotiden (enkelstrengs) of nucleotideparen (dubbelstrengs) is. De selectie wordt ge-

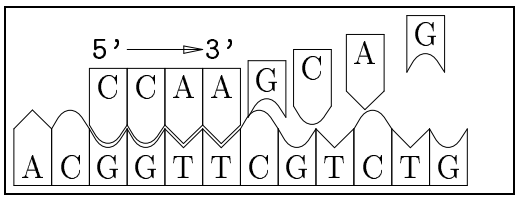


Figuur 5: Selectie: de reeks `5'-GGAT-3'` wordt gebonden door de complementaire reeks `5'-ATCC-3'`.

woonlijk uitgevoerd door een monster DNA te plaatsen op een uiteinde van een gel, en deze gel vervolgens in een electrisch spanningsveld te plaatsen. De (negatief) geladen DNA moleculen zullen zich verplaatsen naar de positieve electrode, waarbij de langere moleculen zich langzamer bewegen vanwege de grotere weerstand die ze ondervinden bij het doorlopen van de gel. Wanneer de stroom wordt uitgeschakeld als de eerste moleculen het andere uiteinde van de gel hebben bereikt, zullen de moleculen zich naar hun lengte over de gel hebben gesorteerd. Normaler wijze is DNA onzichtbaar. De moleculen kunnen zichtbaar worden gemaakt door ze met ethidium bromide te kleuren; op die manier vormen zich 'bandjes' wanneer de gel onder ultraviolet licht bekeken wordt. Om de lengtes van de moleculen in elk van de bandjes te bepalen kunnen we parallel aan het te testen monster een hoeveelheid DNA mee te laten lopen waarvan de lengteverdeling bekend is. Zulke DNA linialen zijn commercieel verkrijgbaar. Wanneer we niet alleen een indruk willen krijgen van de aanwezige lengtes, maar de moleculen van een bepaalde lengte willen selecteren, kunnen deze uit de gel worden vrijgemaakt door het betreffende bandje te verwijderen en te spoelen.

Vermeerdering. Dubbelstrengs DNA kan gekopieerd worden door het molecuul te splitsen (denaturiseren) in twee enkele strengen DNA en elk der helften te gebruiken als een mal om de oorspronkelijke dubbele streng op te bouwen. Deze verdubbeling wordt uitgevoerd door het enzym polymerase. Uit de oplossing worden aanwezige losse nucleotiden aangetrokken en complementair aan de enkele streng gehecht. Omdat de verdubbeling weer de regels voor complementariteit volgt zijn de nieuwe dubbele strengen kopieën van het origineel. Polymerase kan zijn werk niet doen op een geheel enkelstrengs molecuul, maar start zijn werk bij een aangehecht complementair stuk, *primer* genoemd. Figuur 6 geeft dit proces schematisch weer, waarbij aangetekend is dat in de praktijk langere primers gebruikt worden.

Een belangrijke vooruitgang in de DNA technologie werd geboekt door de ontdekking dat dit procédé geautomatiseerd kan worden. Zo kunnen we in het laboratorium in korte tijd onwaarschijnlijk grote hoeveelheden van het oorspronkelijke DNA fabriceren. Dit geautomatiseerde proces heet de *polymerase*



Figuur 6: Verdubbeling. Polymerase vult een enkele streng aan tot een dubbele in de 5′-3′ richting, beginnend bij een primer, hier `5'-CCAA-3'`.

chain reaction, PCR, en was destijds een Nobelprijs waard.

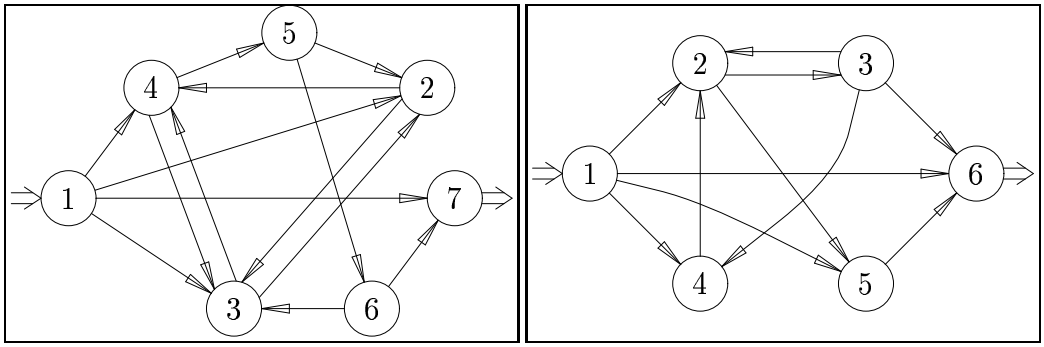
In grote lijnen laat PCR zich als volgt beschrijven. Neem het dubbelstrengs DNA dat we willen vermeerderen en voeg daaraan polymerase toe met een grote hoeveelheid losse nucleotiden om nieuwe moleculen te kunnen bouwen. Verder bepalen we twee geschikte primers; deze bepalen het stuk van het oorspronkelijke DNA dat vermenigvuldigd zal gaan worden.

De eigenlijke PCR wordt nu automatisch uitgevoerd door cyclisch de temperatuur te variëren. Bij hoge temperatuur splitst het DNA zich in enkele strengen. Bij lage temperatuur hechten de primers zich op de losse strengen. Bij een oplopende temperatuur doet polymerase zijn werk en worden de enkele strengen verdubbeld vanaf de primers. Hierna herstart de temperatuur cyclus. Er hoeft bij deze methode niet steeds nieuw polymerase toegevoegd te worden dankzij de ontdekking van hitte resistent polymerase (uit bacteriën die leven in ondergrondse geisers en bestand zijn tegen hoge temperaturen).

2 Hamilton Pad Probleem

2.1 Grafen

Veel praktische problemen uit de zogenaamde *operations research* laten zich beschrijven met behulp van grafen. Een *graaf* is de wiskundige abstractie van een netwerk van steden met hun tussenliggende verbindingen. De steden heten dan knopen, de verbindingen takken. We kunnen de verbindingen waarden meegegeven, die dan een eigenschap van die verbinding weergeven. Zo'n eigenschap kan zijn de kosten van de verbinding (bijvoorbeeld uitgedrukt in het aantal af te leggen kilometers langs een weg) of de capaciteit van de verbinding (bijvoorbeeld het maximaal aantal bierflessen per minuut langs een transportband). Gegeven een graaf met haar verbindingen is dan het probleem om een optimale goederenstroom te bepalen, waarbij capaciteiten gemaximaliseerd worden en kosten geminimaliseerd. Denk bijvoorbeeld aan een vertegenwoordiger (handelsreiziger) die in een zo efficiënt mogelijke



Figuur 7: Twee grafen. De linker graaf heeft een Hamilton pad, de rechter niet.

volgorde zeven steden moet bezoeken die op gegeven afstand van elkaar liggen (vandaar dat dit type problemen ook wel handelsreizigersproblemen wordt genoemd.)

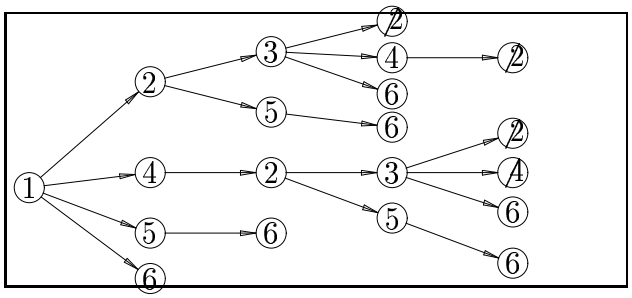
Ook zonder waarden op de verbindingen kunnen eenvoudige graafproblemen onverwacht algoritmisch inefficiënte oplossingen hebben, in de zin dat er geen snelle algoritmen voor bekend zijn. We beschrijven een voorbeeld, het Hamilon Pad Probleem, hier afgekort tot HPP.

Formeel is een graaf een paar $G = (V, E)$, waarbij V een verzameling is van *knopen* en E een verzameling *takken*. Elke tak is een paar (x, y) van knopen. Takken zijn gericht, dat wil zeggen de tak kan doorlopen worden van x naar y, en niet andersom.

Grafen kunnen op een natuurlijke manier als plaatje worden weergegeven. Bijvoorbeeld, de graaf (V, E) met knopen $V = \{1, 2, 3, 4, 5, 6, 7\}$ en takken $E = \{\ (1,2), (1,3), (1,4), (1,7), (2,3), (2,4), (3,2), (3,4), (4,3), (4,5), (5,2), (5,6), (6,3), (6,7)\ \}$ is getekend in Figuur 7 (links). Een *pad* in een graaf is een reeks knopen die achterenvolgens door takken zijn verbonden. In de zojuist gegeven graaf is $6, 3, 4, 5, 2, 4, 3$ een pad in de graaf vanuit knoop 6 naar knoop 3.

Voor het HPP kijken we alleen naar grafen met speciaal gemerkte begin- en eindknoop. We nemen daarbij aan dat de beginknoop geen inkomende takken heeft (een zgn. bron) en de eindknoop geen uitgaande takken (een zgn. put). Het HPP stelt de vraag of de graaf een pad heeft van begin- naar eindknoop dat elke knoop uit de graaf precies één keer aandoet. Zo'n pad noemen we een *Hamilton pad*. Merk op dat de speciale beginknoop alleen als eerste knoop op een pad kan voorkomen, omdat die knoop geen inkomende takken heeft. Evenzo kan de speciale eindknoop alleen als laatste knoop op en pad liggen.

In de linker graaf uit Figuur 7 bestaat inderdaad zo'n Hamilton pad,



Figuur 8: Systematische opsomming van alle mogelijke paden vanuit 1 die geen knopen tweemaal aandoen.

en wel de reeks $1, 2, 3, 4, 5, 6, 7$; immers, $(1, 2), (2, 3), \ldots, (6, 7)$ zijn allemaal takken van de graaf; 1 en 7 zijn de gemarkeerde begin- en eindknopen.

De rechter graaf uit Figuur 7 heeft echter geen Hamilton pad zoals met enig proberen snel duidelijk wordt. In het algemeen echter zijn er geen eenvoudige structurele eigenschappen bekend van grafen die een Hamilton pad bezitten. De enige manier om afdoende aan te tonen dat zo'n pad niet bestaat is door in principe alle mogelijkheden na gaan. Zie hiervoor Figuur 8, waar we alle mogelijke paden opsommen in een boomdiagram, tot het moment waar de pad een knoop voor de tweede keer bezoekt. Geen van deze paden doorloopt alle knopen van de graaf.

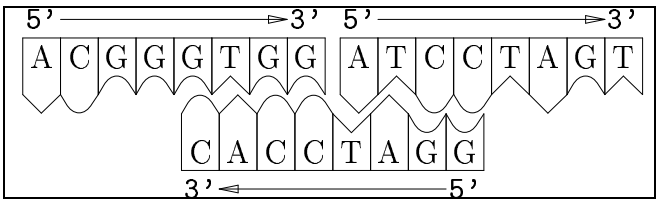
2.2 Het experiment van Adleman

Adleman nam een kleine graaf en beantwoordde hiervoor het Hamilton Pad Probleem: is er een pad van begin- naar eindknoop dat elk der knopen precies één keer aandoet? Hij implementeerde algoritme en invoer in DNA. Het HPP is een zogenaamd NP-compleet probleem (zie later in Paragraaf 3): er zijn geen efficiënte klassieke oplossingsmethoden voor HPP bekend, en het is onwaarschijnlijk dat die bestaan. Elk algoritme dat het HPP oplost moet in essentie elk pad door de graaf bekijken en verifiëren of dit alle knopen precies één maal aandoet. Alleen dit genereren van alle paden is op zich al een taak die erg veel[3] tijd kost.

Dit is het algoritme waar Adleman zich op baseert:

1. genereer een pad door de graaf
2. controleer of elke knoop van de graaf precies één keer voorkomt op het pad

[3]Namelijk exponentieel veel tijd, zie Paragraaf 3.



Figuur 9: **Adleman.** Knopen `5'-ACGGGTGG-3'` en `5'-ATCCTAGT-3'` verbonden door tak `5'-GGATCCAC-3'`.

3. indien het pad voldoet aan 2. heeft de graaf een Hamilton pad, anders terug naar 1.

Het inzicht van Adleman is nu om de paden niet een voor een te genereren, maar tegelijkertijd, gebruikmakend van complementariteit en *self assembly*, het vermogen van DNA om spontaan lange ketens te vormen opgebouwd uit een groot aantal passende kleine segmenten. Elk van de lange ketens dubbelstrengs DNA stelt een pad in de graaf voor. De tests die een Hamilton pad moeten aantonen worden daarna parallel op alle paden uitgevoerd. Omdat een oplossing met DNA werkelijk een zeer groot aantal moleculen kan bevatten is het alsof we een zeer groot aantal computers parallel aan het werk hebben gezet om het Hamilton pad te vinden.

De implementatie van Adleman werkt als volgt. Elke knoop krijgt een kenmerkende streng DNA toegekend, en een pad is dan een streng van deze knoopcodes. Takken krijgen evenzo elk een DNA-representatie, passend op de bijbehorende knoop-strengen. Wanneer knoop x bijvoorbeeld gecodeerd wordt door `5'-ACGG GTGG-3'` en knoop y door `5'-ATCC TAGT-3'`, dan wordt de verbindende tak (x, y) de DNA-brug die de laatste helft van de basen van de vertrekknoop x verbindt met de eerste helft van de basen van de aankomstknoop y, complementair, dus: `3'-CACC TAGG-5'`, zie hiervoor Figuur 9. Deze enkelstrengs knopen en takken worden, in voldoende aantallen, samengevoegd in een reageerbuis waarin ook ligase aanwezig is, en vormen spontaan dubbelstrengs paden in de graaf door annealing. De aanwezige ligase zorgt ervoor dat ook de fosfaat-bindingen in de ruggengraat tussen de korte stukjes DNA gelegd wordt, zodat het molecuul niet uiteenvalt wanneer het dubbelstrengs molecuul in enkele strengen gesplitst wordt. In ons (theoretische) voorbeeld hebben we de knopen weergegeven door stukjes DNA met een lengte van acht basen, Adleman nam voor zijn experiment een graaf van zeven knopen (de linker graaf in Figuur 7) en codeerde met twintig basen per knoop. Dit is voornamelijk om de kans te verminderen dat de DNA-coderingen van de knopen op elkaar lijken; een voorwaarde om annealing tussen verschillende knopen te vermijden.

In het laboratorium wordt vervolgens de test 2. op de paden uitgevoerd. Eerst worden paden met het juiste aantal doorlopen knopen uitgefilterd door de DNA-strengen op lengte (aantal nucleotiden) te scheiden. In het voorbeeld van Adleman heeft het pad 7 knopen die elk 20 nucleotiden lang zijn. De juiste lengte voor een pad is daarmee 140 nucleotiden.

Vervolgens wordt voor elke knoop apart de aanwezigheid getest door het complement van de knoop-code te synthetiseren en daarmee selectie op het patroon van de knoop uit te voeren, een van de basisoperaties op DNA zoals eerder beschreven. De strengen die hechten aan het complement kunnen worden losgespoeld om aan de test voor de volgende knoop te worden onderworpen.

Als we weten dat elke knoop tenminste een keer voorkomt en dat het pad de juiste lengte heeft, weten we ook dat geen knoop dubbel kan voorkomen (dan moet het pad langer zijn). Bovendien moet het pad automatisch beginnen en eindigen in de speciale begin- en eindknoop.

Tenslotte moet gekeken worden of er moleculen zijn die het selectieproces hebben overleefd. Enkele moleculen DNA in een oplossing zijn onzichtbaar; het is als zoeken naar een speld in een hooiberg. Met behulp van PCR kan het aantal spelden miljoenen keren vermeerderd worden zodat het DNA van de overgebleven paden op een gel zichtbaar gemaakt kan worden. De voor PCR benodigde primers worden zo gekozen dat ze passen op de begin- en eindknoop.

2.3 Commentaar

Natuurlijk toont dit experiment alleen nog niet aan dat een DNA computer ook daadwerkelijk gebouwd kan worden. Het is pionierswerk, het begin van onderzoek in deze richting. Adleman zelf begrijpt dat natuurlijk ook wel. Hij waarschuwt dat zijn technieken verfijnd en geautomatiseerd moeten worden om te verhinderen dat we voor grotere problemen (met honderden knopen) een badkuip met DNA moeten vullen: duur en onhandelbaar. Ook is Adleman bezorgd over de tijd die zijn experiment hem kostte in het laboratorium: ongeveer zeven werkdagen voor dit kleine voorbeeld. De meeste tijd ging daarbij zitten in de selectie op patroon, om na te gaan of elke stad daadwerkelijk in het pad voorkomt. Dit komt gedeeltelijk omdat Adleman geen geschoold biochemicus is. Ook hier zullen betere algoritmen en verdere automatisering versnelling bieden.

Verder is de koppeling tussen strengen DNA niet zo vanzelfsprekend foutloos als gesuggereerd. Er kunnen fouten optreden: DNA vormt dan een dubbele streng ondanks het feit dat er een aantal niet op elkaar passende baseparen zijn. Ook kunnen er niet-lineaire structuren gevormd worden, bijvoor-

beeld een klein lusvormig uitsteeksel aan het lineaire dubbelstrengs molecuul. Dit vraagt om een nauwkeurig ontwerp van de DNA-codes die toegekend worden aan elk van de knopen uit de graaf. Ze moeten zo ontworpen worden dat het DNA als het ware uitgenodigd wordt om zonder fouten samen te voegen.

Fouten kunnen ook optreden bij de biochemische operaties. Vooral PCR en de scheiding op patroon zijn beruchte bronnen van onnauwkeurigheid. In het biochemisch laboratorium is dat in het algemeen geen probleem. Een enkel misplaatst molecuul verdwijnt in de massa. Hier, bij het oplossen van het HPP kan het echter betekenen dat we ten onrechte een Hamilton pad denken te vinden. De moraal is dat we in de algoritmiek niet gewend zijn om te denken over zulke foutmarges. In onze DNA computer zullen we echter extra waarborgen moeten inbouwen om zeker te zijn dat geen foutieve oplossingen van het probleem gegenereerd worden.

Adleman legt er de nadruk op dat zijn experiment gezien moet worden als didactisch voorbeeld, als bewijs dat rekenen met moleculen inderdaad mogelijk is, slechts een eerste stap op weg naar praktische toepassingen. Hij suggereert te zoeken naar nieuwe scheidingstechnieken, naar andere polymeren dan DNA, in het bijzonder naar niet-biologische moleculen die als hardware kunnen dienen. Deze zouden compacter zijn en stabieler. Adleman verwacht daarbij veel van de zgn. combinatorische scheikunde. In deze tak van onderzoek worden grote hoeveelheden chemische moleculen met kleine variaties in structuur gegenereerd en daarna op wenselijke eigenschappen geselecteerd.

2.4 Andere modellen

Kortom, dit experiment is slechts een eerste (maar belangrijke) stap in de goede richting; een zogenaamd '*proof of principle*'. Er wordt daarom gewerkt aan het voorstellen van nieuwe DNA-rekenmodellen, zowel theoretisch (op papier) als daadwerkelijk in het laboratorium. Een van de eerste aandachtspunten was het om aan te tonen dat het in theorie inderdaad mogelijk is een programmeerbare DNA-computer te ontwerpen. Adlemans techniek lost immers een vast probleem op, en kan niet anders geprogrammeerd worden. Om die programmeerbaarheid aan te tonen is het standaard 'universele' basis computermodel, de Turing machine, gemodelleerd in DNA hardware. Nieuw hierbij is dat daarbij handig gebruik gemaakt van enzymen om de strengen te knippen om zo stukjes DNA (met informatie) te modificeren.

De theoreticus is tevreden met het programmeren van een Turing machine. Practisch georiënteerde informatici zien meer in het bouwen van schakelingen met eenvoudige logische (Boolese) operaties. Die schakelingen zijn inmiddels ook geïmplementeerd in DNA.

3 Complexiteit

3.1 Efficiëntie

De efficiëntie van een algoritme wordt uitgedrukt als functie $\epsilon(n)$ van de grootte n van de probleeminstantie. Dat wil zeggen $\epsilon(n)$ is de maximale tijd die het algoritme nodig heeft om een willekeurige instantie van het probleem ter grootte n op te lossen, de zogenaamde *worst-case* efficiëntie. Voor algoritmen op grafen is n bijvoorbeeld het aantal knopen van de graaf.

	n	n^2	n^5	2^n	3^n
$n=10$	1×10^{-5} sec	1×10^{-4} sec	1×10^{-1} sec	1×10^{-3} sec	6×10^{-2} sec
30	3×10^{-5} sec	9×10^{-4} sec	24 sec	18 min	6.5 jaar
50	5×10^{-5} sec	2×10^{-3} sec	1.7 min	13 dagen	3855 eeuwen
60	6×10^{-5} sec	4×10^{-3} sec	13 min	366 eeuwen	10^{13} eeuwen

Hierboven staan een aantal hypothetische efficiëntiefuncties gegeven, met de rekentijd die ze vergen voor de probleemgroottes $n = 10, 30, 50, 60$. We geven voorbeelden van lineaire efficiëntie, $\epsilon(n) = n$, polynomiale efficiëntie, $\epsilon(n) = n^2$ en $\epsilon(n) = n^5$, en exponentiële efficiëntie, $\epsilon(n) = 2^n$ en $\epsilon(n) = 3^n$. Alle functies zijn geijkt op $\epsilon(1)$ gelijk aan 1×10^{-6} sec, door ze met een geschikte constante te vermenigvuldigen. Merk op dat de polynomiale efficiënties steeds redelijke tijden geven, terwijl de exponentiële efficiënties onwaarschijnlijk lange rekentijden geven bij probleemgrootte $n = 60$, en al eerder voor de machten van drie.

Soms wordt gesuggereerd dat het niet nodig is om snellere, efficiënte, algoritmen te ontwerpen, omdat al onze intellectuele inspanningen snel door de techniek worden ingehaald. Immers, zo wordt geredeneerd, elke oplossingsmethode wordt vanzelf efficiënter door een jaar te wachten en de dan gangbare computers te gebruiken. We moeten constateren dat deze redenatie alleen opgaat als het probleem al een efficiënt algoritme heeft. Dit zien we door anders naar de efficiëntiefuncties te kijken, als volgt.

	n	n^2	n^5	2^n	3^n
nu	N	N	N	N	N
$100\times$	$100N$	$10N$	$2.5N$	$N+6.6$	$N+4.2$
$1000\times$	$1000N$	$32N$	$4N$	$N+10$	$N+6.3$

Neem aan dat de probleemgrootte die we nu, met de huidige techniek, in een bepaalde tijd kunnen oplossen gelijk is aan N (bijvoorbeeld uitgedrukt in het aantal knopen van de graaf). Wanneer we een lineaire efficiëntie hebben dan kunnen we met een honderd maal snellere computer inderdaad een

honderdvoudige grootte van het probleem aan. Echter, bij een exponentiële efficiëntie wordt de handelbare probleemgrootte slechts een kleine constante groter; niet echt de doorbraak waar we op hopen.

De conclusie is duidelijk: bij problemen waarvoor slechts oplossingen van exponentiële efficiëntie bekend zijn, kunnen we slechts hopen op marginale winsten in de te behandelen probleemgrootte, tenzij we een radicaal nieuwe aanpak kiezen. Zo zijn er *heuristieken* die een goede benadering van de optimale oplossing leveren, en zijn er *Monte Carlo methoden* die met grote waarschijnlijkheid de oplossing leveren (maar niet altijd), en mogelijk nieuwe technologieën ... zoals DNA-computing. In de volgende paragraaf bespreken we waarom juist HPP een uitgelezen probleem is om een nieuwe aanpak op uit te proberen.

3.2 P vs. NP

Vanwege het opvallende verschil tussen polynomiale en exponentiële efficientie, hebben de problemen die door een algoritme in polynomiale tijd opgelost kunnen worden een naam gekregen, nl. type P. Evenzo vormen problemen waarvoor een mogelijke oplossing door een polynomiaal algoritme in polynomiale tijd geverifieerd kan worden type NP, voor *Niet-deterministisch Polynomiaal*: er kan een oplossing ('niet-deterministisch') gegokt worden en vervolgens in polynomiale tijd gecontroleerd. Eén van deze problemen is het HPP: van een gegeven pad is eenvoudig na te gaan of deze een Hamilton pad door de graaf is. We hebben al opgemerkt dat de implementatie van Adleman verwant is aan zo'n niet-deterministisch algoritme. Er wordt alleen niet een enkel mogelijk Hamilton pad door de graaf gegokt, maar er worden zoveel mogelijk paden tegelijk gegenereerd en gecontroleerd.

Een van de grote onopgeloste vragen van de Algoritmiek is het verband tussen de verzamelingen P en NP. In ieder geval is P bevat in NP; problemen in P kunnen opgelost worden zonder gokken. Maar zijn P en NP gelijk aan elkaar, of is NP echt groter dan P ? We weten het niet. Het probleem wordt zo fundamenteel geacht, dat het tot een van zes *millennium problemen* is verklaard, met de oplossing waarvan een miljoen dollar te verdienen is[4].

Vaak kunnen problemen in elkaar vertaald worden. Zo kan een bepaald probleem met logische formules vertaald worden in het oplossen van een HPP-probleem in een speciaal geconstrueerde graaf. Dit betekent dat een (efficiënte) oplossing voor het HPP ook een (efficiënte) oplossing van het logische probleem oplevert, aangenomen dat ook de vertaling efficiënt is.

[4]Clay Mathematics Institute Cambridge, Massachusetts, USA
`http://www.claymath.org/prizeproblems/pvsnp.htm`

Het was echter een grote ontdekking dat er problemen in NP bestaan die in zekere zin tot de moeilijkste uit NP behoren: voor èlk probleem uit NP bestaat een vertaling naar zo'n moeilijkst probleem. We noemen deze problemen NP-compleet. Het allereerste NP-complete probleem werd gegeven door Cook. Hij bewees dat elk Turing Machine[5] programma (efficiënt) vertaald kan worden in een bepaald logisch probleem, SAT genaamd[6]. Deze ontdekking heeft een belangrijk gevolg. Wanneer er een efficiënt algoritme voor SAT of een van de andere NP-complete problemen gevonden wordt, is daarmee een efficiënte oplossing voor alle NP-problemen bekend, en volgt dat $\mathsf{P} = \mathsf{NP}$.

Eén van de vele bekende NP-complete problemen is HPP. Het is dus een goede keuze om met dit probleem te werken daar het een representant is voor de hele verzameling NP. Er zijn letterlijk honderden problemen waarvan aangetoond is dat ze NP-compleet zijn. Daaronder zijn praktische problemen, zoals het probleem dat bekend staat als BIN-PACKING:

Gegeven: m floppy's met elk capaciteit voor x bytes, en k files ter grootte $y_1, y_2, \ldots, y_k$ bytes. Gevraagd: is het mogelijk de k files op de m floppy's te schrijven (zó dat geen der files over twee floppy's verdeeld hoeft te worden)?

Maar er zijn ook problemen die alleen door verstokte hobbyisten van echt belang gevonden worden, zoals MINESWEEPER, gemodelleerd naar een spelletje geleverd bij een bekend operating systeem:

Gegeven: een rechthoekig veld van m bij n vakjes groot, waarvan elk veld ofwel blanko is, ofwel gemarkeerd met een geheel getal tussen 0 en 8. Gevraagd: is het mogelijk 'bommen' over de velden te verdelen, consistent met de gegeven getallen?

Het maakt niet uit welk NP-compleet probleem aangepakt wordt, omdat een efficiënte oplossing voor elk NP-probleem omgezet kan worden in een efficiënte oplossing voor een ander probleem van dat type.

4 Ten Slotte

4.1 Natural Computing

DNA computing zoals hier beschreven is slechts een van de onderwerpen die bestudeerd worden op het grensvlak tussen Biologie en Informatica. Dit gebied genaamd *Natural Computing* onderzoekt de theorie en praktijk van re-

[5]Het algemeen aanvaarde mathematische model van een programmeerbare computer, zie ook Paragraaf 2.4.

[6]SAT kan als volgt geformuleerd worden. Gegeven een logische formule met Boolese variabelen. Bestaat er een toekenning van true en false aan deze variabelen zodat de formule true oplevert?

kenmethoden geïnspireerd op de natuur. Twee voorbeelden van wat oudere (en bekendere) deelgebieden zijn neurale netwerken en genetische algoritmen.

Neurale Netwerken zijn een rekenmethode afgekeken van de complexe structuur van onze hersenen. Met behulp van software wordt een (niet al te groot) netwerk van neuronen gesimuleerd. Elk neuron is een geïdealiseerde zenuwcel die signalen ontvangt van een aantal andere cellen en, als deze signalen bij elkaar een gekozen drempel overschrijden, zelf een signaal doorgeeft. Het netwerk heeft gewichtsfactoren op de verbindingen tussen de neuronen om zo sommige signalen meer te kunnen laten meetellen dan andere. Het is de bedoeling dat het netwerk data leert te classificeren aan de hand van reeds bekende voorbeelden. Deze techniek heet *trainen*: de gewichten worden dan zo bijgesteld dat het netwerk zo veel mogelijk de juiste classificatie geeft op bekende gevallen. De hoop is dan dat het netwerk dan in staat is om te generaliseren, en ook voor nieuwe gevallen een betrouwbare classificatie levert.

De toepasbaarheid van neurale netwerken wordt bijvoorbeeld onderzocht voor het aan de hand van seismografische gegevens voorspellen van geschikte boorlocaties voor olie. Normaal gebeurt dit door (dure en druk bezette) experts. Het zou tijd- en geldwinst opleveren als althans een voorselectie door de computer gedaan kan worden. Conventionele algoritmische methoden lijken minder succesvol dan neurale netwerken omdat de kennis van de experts zich niet makkelijk laat vastleggen in een aantal logische beslissingsregels. De hoop is dat het neurale netwerk zèlf een expert wordt door het te trainen aan de hand van reeds door experts beoordeelde metingen.

Genetische Algoritmen worden ingezet wanneer de opbrengst van een lastig te analyseren functie gemaximaliseerd moet worden. Mogelijke kandiaatwaarden krijgen een representatie in de vorm van een reeks enen en nullen. Zo'n reeks wordt dan opgevat als de chromosomen (het DNA) van een individu, de informatie die de eigenschappen van dat individu vastlegt. Vervolgens wordt de verzameling individuen onderworpen aan een proces van *natuurlijke selectie*. We evalueren hoe geschikt elk van de individuen is, dat wil zeggen welke waarde zij toegewezen krijgen door de functie, in deze context *fitness functie* genaamd. De beste individuen worden dan geselecteerd voor de volgende ronde. Hun chromosomen worden gecombineerd, bijvoorbeeld door telkens twee individuen te paren en nieuwe chromosomen te maken door gedeeltes van de chromosomen van de ouders samen te voegen. Op deze manier krijgen we weer een (grote) verzameling individuen, gerepresenteerd door hun chromosomen. We onderwerpen deze aan een evaluatie om te kijken of het gestelde doel, een grote waarde van onze fitness functie, gehaald wordt. Zo niet dan kan het selectieproces herhaald worden, door opnieuw de beste individuen te paren tot een verzameling nieuwe kandidaten.

Er wordt veel studie gedaan naar diverse vormen van genetische algoritmen. Belangrijk is het vinden van een geschikte vertaling van de parameters van het probleem naar chromosomen, in deze context een reeks enen en nullen. Ook het proces van natuurlijke selectie kent veel varianten die in verschillende context soms wel en soms niet succesvol zijn.

De toepassing van genetische algoritmen is bijvoorbeeld onderzocht bij het ontwerp van vliegtuigen, en bij het verminderen van snijafval in de kledingindustrie (om maar eens twee willekeurige, door ons op internet gevonden, onderwerpen te noemen). Ook theoretische problemen zoals het Handelsreizigersprobleem, een variant van het Hamilton pad probleem waarbij afstanden meewegen, worden onderzocht als toepassingsgebied van genetische algoritmen. Het gaat daarbij niet zozeer om het bepalen van het unieke kortste pad, maar om een aanvaardbare benadering van dat pad.

Zowel bij neurale netwerken als bij genetische algoritmen wordt een paradigma uit de natuur gebruikt, namelijk de werking van hersencellen in het eerste geval en de erfelijkheidsleer in het tweede. De resulterende algoritmen worden echter geïmplementeerd in klassieke computers werkend met siliciumchips.

De aanpak van DNA computing is gewaagder. Ook bij DNA computing wordt er naar de natuur gekeken om algoritmen te implementeren, maar de aanpak is gewaagder. Immers, uiteindelijk wordt voorgesteld om de silicium hardware te vervangen en algoritmen te implementeren in bioware!

4.2 Slotwoord

Hebben we over geruime tijd een aquarium met een toetsenbord op ons buro staan? Eerlijk gezegd geloven we dat niet, die ontwikkeling zal meer tijd vergen. Maar het is duidelijk dat iedereen die geïnteresseerd is in de informatica het gebied van DNA computing nader moet bekijken. Al is het maar om beter, dieper, en vernieuwd inzicht te krijgen in de natuur van het rekenen.

4.3 Leesvoer

Het oorspronkelijke artikel van Leonard Adleman staat in Science [Adl94]. Een meer persoonlijke blik op dit eerste experiment schreef hij voor Scientific American [Adl98]:

> '*This realization caused me to sit up in bed and remark to my wife, Lori, "Jeez, these things could compute." I did not sleep the rest of the night, trying to figure out a way to get* DNA *to solve problems*'.

DNA Computing is één van de onderwerpen uit een deel van de interessante reeks *What's Happening in the Mathematical Sciences* van de American Mathematical Society [Cip96]. Aangekondigd is een boek over theoretische en experimentele kanten van DNA computing [Amo02]. Up-to-date op het vakgebied blijft u door het volgen van de jaarlijkse conferentie *international meeting on DNA Based Computers*, waarvan de eerste werd gehouden in 1995; de eerste keer buiten de Verenigde Staten kwam de conferentie naar Nederland [DNA6]. De eerste theoretische, mathematisch geöriënteerde, modellen over het rekenen met DNA en restrictie-enzymen stammen reeds uit 1987 [Hea87]; een breed overzicht van de theoretische stand van zaken is te vinden in het boek [PRS98].

Referenties

[Adl94] L.M. Adleman. Molecular Computation of Solutions to Combinatorial Problems. *Science* 266 (1994) 1021–1024.

[Adl98] L.M. Adleman. Computing with DNA. *Scientific American* (August 1998) 34-41.

[Amo02] M. Amos. *Theoretical and Experimental DNA Computation*. Natural Computing Series, Springer Verlag, 2002.

[Cip96] Barry Cipra. *What's Happening in the Mathematical Sciences*, Volume 3, American Mathematical Society, 1996

[DNA6] A. Condon, G. Rozenberg (eds.) *DNA Computing, 6th International Workshop on DNA Based Computers*, June 2000, Leiden, The Netherlands, Lecture Notes in Computer Science Vol. 2054, 2001.

[Hea87] T. Head. Formal Language Theory and DNA: an analysis of the generative capacity of recombinant behaviors. *Bulletin of Mathematical Biology* 49 (1987) 737–759.

[PRS98] G. Păun, G. Rozenberg, A. Salomaa. *DNA Computing, New Computing Paradigms*, Springer-Verlag, 1998.

[Poo96] R. Pool. The Ultimate Computer: A trillion calculations in parallel. *New Scientist* (13 July 1996) 26–31.