Nederlandse organisatie
voor toegepast
natuurwetenschappelijk
onderzoek

FEL

Fysisch
Laborat

89-3568 II

E

DTIC FILE COPY

AD-A217 930

DTIC
ELECTE
FEB 12 1990
S D
E

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

90 02 ^^ ^9-

89-3560 II

Nederlandse organisatie
voor toegepast
natuurwetenschappelijk
onderzoek

TNO-rapport

FEL

Fysisch en Elektronisch
Laboratorium TNO

Postbus 96864
2509 JG 's-Gravenhage
Oude Waalsdorperweg 63
's-Gravenhage

Telefoon 070 - 26 42 21

89-3568 II

0

rapport no. FEL-89-B65

exemplaar no. 7

titel

Neurale netwerken en radarsystemen

auteur(s):

Ir. H.J. Borgers

RAPP. NR.		
UITGELEEND AAN:	Datum retour	Klant-nr.

148 - 12777 B - 830191* - 10

gericering

titel : ongerubriceerd

samenvatting : ongerubriceerd

rapport : ongerubriceerd

Page : 36

Total bladzijden : 155

Total bijlagen : 1

datum : augustus 1989

EMENT A
: release;
mited

rapport no. : FEL-89-B65
titel : Neurale netwerken en radarsystemen

auteur(s) : Ir. H.J. Borgers
instituut : Fysisch en Elektronisch Laboratorium TNO

datum : augustus 1989
hdo-opdr.no. :
no. in iwp '89 : 710.3

SAMENVATTING

In een modern scenario is het gebruik van een qualitatief hoogwaardige radar niet meer weg te denken. Voor het verwerken van de data van een dergelijke radar is real time automatische signaalverwerking noodzakelijk. Door de toegenomen dreiging (stealth, stoortechnieken, zeer snel vliegende objecten) voldoen huidige algoritmen en computerarchitecturen maar in een zeer beperkte mate aan de eisen die aan huidige en toekomstige systemen noodzakelijkerwijs gesteld moeten worden.

Gezien de uitmuntende prestaties van het visuele systeem van mens en dier en de huidige snelle ontwikkelingen van de theorie en de praktijk van de neurale netwerken, is door de sectie Radar Signaalverwerking van het FEL/TNO bezien of, en zo ja hoe, neurale netwerken te gebruiken zijn voor het verwerken van radardata.

Het onderzoek bestond uit het doen van een literatuurstudie, het ontwerpen van een algemene softwaresimulatie-omgeving en het verrichten van enkele relevante simulaties.
Als deelresultaat hiervan zijn in dit verslag de fundamenten van de neurale-netwerktheorie plus diverse voor radarsignaalbewerking te gebruiken netwerken beschreven.

De eindconclusie van dit rapport is dat de snelheid van de neurale netwerken en de mogelijkheid de netwerken hogere-ordecorrelaties te (laten) leren redenen zijn voor voortzetting van het onderzoek.

report no. : FEL-89-B65
title : Neural networks and radar systems

author(s) : H.J. Borgers
institute : TNO Physics and Electronics Laboratory

date : August 1989
NDRO no. :
no. in pow '89 : 710.3

ABSTRACT

In modern scenarios an extensive use is made of high performance radar systems. To evaluate the data of this kind of radar systems, it is necessary to use automatic real-time signal processing. Because of the ever increasing threat (stealth, jamming, objects flying at high speed), algorithms and architectures that are currently in use, do not meet the requirements of future radar systems.

Man and animals show very convincingly that real-time signal processing is possible for at least visual data. This capability has urged researchers in many different areas to investigate the principles of neural networks. In the radar group at FEL-TNO research was initiated to explore the application of neural networks in radar signal processing.

The main goals of this research program were to summarise the literature on neural networks, to develop a flexible tool for implementing these networks in software and to perform several relevant simulations. As part of the results of this research, this report describes the fundamentals of neural network theory and shows a number of ways in which neural networks can be used to process radar signals.

From the analysis and the simulation results can be concluded that the processing speed and the ability to train the networks with higher order correlations, are the main reasons to proceed with the research program.

INHOUD

Veel gebruikte afkortingen:

BAM	Bidirectioneel Associatief Geheugen (memory)
BM	Boltzmann Machine
BPA	Back Propagation Algoritme
CC	Cerebrale Cortex
H	Harmonie
LA	Lineaire Associator
MIP	Multi Instruction Processor
MLP	Multi Layer Perceptron
MN	Maximalisatie Netwerk
NN	Neuraal Netwerk
SIP	Single Instruction Processor

1 INLEIDING

1.1 Radarsystemen en neurale netwerken

Enkele tientallen jaren geleden bestond een radarbeeld uit niet veel meer dan een reeks stippen en strepen die, bekeken in een verduisterde ruimte, de operator deden beslissen dat er mogelijk een object naderde. Na het plegen van overleg en het nogmaals aandachtig bestuderen van het scherm door zijn collega's concludeerde men dat er wellicht inderdaad iets aan de hand was en enkele minuten later stegen vliegtuigen op om de situatie ter plekke te gaan bezien.

Een dergelijke radarinstallatie is in een modern scenario van een beperkt nut. Het is niet verwonderlijk dat aan een tegenwoordige radarinstallatie andere eisen worden gesteld dan dat het de buitenwereld weergeeft middels stippen en vlekken die door de operator op hun betekenis dienen te worden beoordeeld. Immers, de interpretatie van radarbeelden is door de grotere aantallen vliegende objecten en het gebruik van radarmisleidende en "stealth" technieken veel moeilijker geworden, terwijl de beslissingstijd door de veel hogere snelheden van (laagvliegende) vliegtuigen en missiles enorm is afgenomen.

Het vergroten van het zendvermogen van de radar om zo de beslissingstijd te verlengen en/of de radarresolutie te verbeteren levert grote en onoverkomelijke technische en tactische problemen op (immers, een radar met meer zendvermogen is eenvoudiger te ontdekken door een eventuele tegenpartij). De weg naar een betere radar leidt daarom niet in de richting van krachtiger, maar van qualitatief betere radarinstallaties. Een qualitatief betere radar kan verkregen worden door enerzijds betere (hoge resolutie) radarantennes (of antennenetwerken) te ontwerpen, zodat de operator van betere (meer) informatie kan worden voorzien en anderzijds door verregaande geautomatiseerde signaalbewerking toe te

passen, zodat de operator voor een groot deel wordt ontlast van de interpretatie van de stortvloed aan gegevens.

Het gebruik van de computer bij het verwerken van radarbeelden is inmiddels niet meer weg te denken. Bij een moderne radar wenst men dat "slimme" algoritmen clutter filteren, corrigeren voor indirecte reflecties (bijvoorbeeld via zee, startbaan of gebouwen) en voor misleidende technieken, natuurlijke of kunstmatige ruis- en stoorsignalen onderdrukken, objecten classificeren en automatisch koers en snelheid meten. Op een modern radarscherm neemt men dan ook liever niet langer strepen en stippen waar, maar ziet men geclassificeerde objecten middels geprojecteerde labels en symbolen, worden niet interessante gegevens onderdrukt (gebouwen, regenbuien) en worden hulpgegevens op commando weergegeven (bijvoorbeeld een plattegrond van het gebied). Op deze manier kan de operator snellere en betere beslissingen nemen of deze zelfs geheel aan de (surveillance-, gevechts)computer overlaten.

Helaas is de situatie niet zo rooskleurig als zij hierboven wordt geschetst. De speurtocht naar algoritmen die bovengenoemde taken kunnen uitvoeren is lang en vol hindernissen. Het is een uiterst moeilijke taak gebleken om radarbeelden, zelfs indien deze met een zeer hoge resolutie worden verkregen, door de computer te laten bewerken. Met name als de objecten passief (niet coöperatief) zijn, is de automatische interpretatie van een radarbeeld momenteel maar in een zeer beperkte mate mogelijk.
De oorzaken hiervan zijn van tweeërlei aard:

Allereerst is het met de huidige stand van de technologie bij lange na niet duidelijk hoe goede algoritmen voor het interpreteren van beelden (met radar, video, sonar of op andere wijze verkregen) geschreven moeten worden. Dit vormt momenteel een grote, of wellicht zelfs de grootste, hindernis bij het invoeren van allerlei technieken die van deze interpretatie gebruik zouden kunnen maken. Hierbij valt naast radar

bijvoorbeeld te denken aan het gebruik van vision bij robotarmen in fabrieken en voor de beveiliging van gebouwen, voor het controleren en regelen van het verkeer en voor het automatisch besturen van transportmiddelen of zelfs voor autonome machines die ingezet zouden kunnen worden voor het doen van onderzoek of het verrichten van arbeid in de ruimte of op andere planeten.

In de tweede plaats schiet de huidige hardware tekort bij het "real-time" uitvoeren van beeldverwerkende algoritmen. De huidige generatie computers is indrukwekkend snel, maar toch vele factoren te traag voor het verwerken van de hoeveelheid gegevens die afkomstig zijn van (hoge-resolutie)radarantennes of videocamera's. Eenvoudige calculaties leren dat men per "bit" binnenkomende data over bijzonder weinig processortijd beschikt. Het ligt dan ook niet in de lijn van de verwachtingen dat de huidige computerarchitectuur in de toekomst geschikt zal blijken voor taken als vision, maar dat zij slechts op een veel hoger (symbolisch) niveau ingezet kan worden bij de verwerking van informatie.

Wat ons moet bemoedigen, ondanks bovenstaande bezwaren, is dat het overduidelijk is dat een real-time interpretatie van beeldinformatie wel degelijk mogelijk is. Mens en dier demonstreren overduidelijk dat dit zelfs schijnbaar moeiteloos mogelijk is. Hierbij valt te denken aan visuele dataverwerking (ogen), maar ook aan de zeer veel op radar gelijkende sonar van de dolfijn en de vleermuis. Het is ook juist deze schijnbare moeiteloosheid die in de vijftiger en zestiger jaren tot een enorme onderschatting van de complexiteit van deze taak leidde.

Gezien bovengenoemde feiten ligt het voor de hand op zijn minst een lijn van onderzoek op te zetten naar signaalverwerking in hersencel(achtige) netwerken. De beloning die men zich kan denken bij het werkelijk begrijpen van de hersenen zal uiteraard zeer groot zijn. Maar ook als slechts gedeeltelijk inzicht in de werking wordt verkregen, mag verwacht worden dat vele nieuwe inzichten verkregen zullen worden.

Men kan natuurlijk stellen dat onderzoek aan hersencelnetwerken uitsluitend moet worden verricht door speciaal daarvoor opgeleide neurologen en dat de rest van de wetenschappers zich maar beter met het eigen "conventioneel" onderzoek kan bezighouden. Door recente ontwikkelingen zijn echter nieuwe, veelbelovende inzichten en daaruit voortvloeiende algoritmen en architecturen naar voren gekomen, die praktische toepassingen van neurale netwerken in vele vakgebieden op korte termijn mogelijk lijken te maken, waaronder toepassingen voor radarsignaalverwerking. Het momenteel algemeen voor handen zijnde computervermogen maakt praktisch gericht onderzoek (middels simulaties) zonder meer mogelijk. In laboratoriumradarsystemen elders in de wereld worden al op dit moment "neurale netwerken" gebruikt voor bijvoorbeeld het verwerken en interpreteren van radarbeelden (zie bijvoorbeeld [11]). Het onderzoek aan neurale systemen is daarmee uit de fundamenteel theoretische onderzoekssfeer gekomen.

Bovenstaande overwegende lijkt het gerechtvaardigd een deel van het onderzoek naar de verwerking van radarsignalen toe te spitsen op neurale netwerken. Het maken van een inventarisatie van de huidige kennis op dit gebied en het doen van een vooronderzoek naar de inzetbaarheid van neurale netwerken voor radar is een uitdagende taak, die door de sectie "Signaalverwerking" van het FEL-TNO is aangegrepen. Dit verslag maakt deel uit van de resultaten van dit onderzoek.

1.2 Doelstellingen

Het voor U liggende verslag vormt de afronding van de eerste fase van onderzoek naar de mogelijkheden van het gebruik van neurale netwerken in radarsystemen. Deze eerste fase vond plaats in de periode october 1987 - maart 1988 en bestond uit het doen van een verkennende literatuurstudie, het ontwerpen van een softwaresimulatie-omgeving waarmee de te bestuderen netwerken snel ontworpen en getest kunnen worden en het uitvoeren van enkele verkennende simulaties aan diverse, mogelijk veelbelovende, neurale netwerken.

De doelstelling van dit verslag is meerledig.

Allereerst is het de bedoeling de lezer een elementaire hoeveelheid kennis over en "feeling" voor neurale netwerken bij te brengen. Daartoe is het eerste deel van dit verslag (H2 tm H5) gewijd aan de theorie, of zo U wilt filosofie, achter het functioneren van neurale netwerken. Met behulp van de literatuurverwijzingen kan de lezer zich verder verdiepen in de vele deelgebieden die dit onderwerp biedt. Dit verslag kan vanuit dit oogpunt worden gelezen als een inventarisatie van de bestaande kennis (in voorgenoemde periode).

In de tweede plaats wil dit verslag duidelijk maken waar mogelijk vanuit de radartechnologie praktische aanknopingspunten met de neurale netwerken te leggen zijn. Daarom wordt in het tweede deel van het verslag (H6) een aantal voorbeelden gegeven die aanknopingspunten opleveren voor toepassingen van de netwerken in de radartechnologie.

Het uiteindelijke doel van dit verslag is te bezien of de vorderingen rond de neurale netwerken grond zijn voor verder onderzoek voor radarsignaalverwerking. Een positief antwoord op deze vraag zou moeten leiden tot meer specifieke detailstudies.

2 HISTORIE VAN HET ONDERZOEK NAAR NEURALE NETWERKEN

2.1 Waarom een overzicht van de ontwikkelingen?

Onderzoek aan neurale netwerken is niet nieuw. In dit eerste hoofdstuk zal kort op het verleden van het onderzoek naar neurale netwerken worden ingegaan, een verleden dat wordt gekenmerkt door "ups" en "downs", waarbij technische, maar ook andere oorzaken voor deze onregelmatige gang van zaken hebben gezorgd.
De reden voor het geven van dit overzicht is enerzijds een indruk te geven van de ontwikkelingen op het gebied van neurale architecturen en om duidelijk te maken dat er momenteel wel degelijk "iets" aan de hand is, maar eveneens te laten zien dat onterecht hooggespannen verwachtingen op de lange termijn niet gunstig uitwerken en dat moet een bruikbare waarschuwing zijn bij onder meer het lezen van dit verslag.
Gezien de momenteel explosieve opleving van de belangstelling voor neurale netwerken vrezen velen voor een herhaling van voorgaande (zie bijvoorbeeld [59][60]]) golfbewegingen (zie hieronder).
Indien men geïnteresseerd is in een meer volledig overzicht dan hier wordt gegeven, wordt verwezen naar [32].

2.2 Vroege "opkomst en ondergang"

Onderzoek naar neurale netwerken vindt zijn oorsprong in de *neurologie* en de *psychologie*. Al in de negentiende eeuw trachtten neurologen de werking van de hersenen te achterhalen door experimenten en microscopisch onderzoek.
Gezien de primitieve aard van de instrumenten en de onbekendheid met het proces van *informatieverwerking*, resulteerde dit in niet veel meer dan het in kaart brengen van de globale hersenstructuur. Experimenten met directe electrische prikkeling van hersengebieden of het operatief verwijderen van delen van de hersenen dienden hierbij als basisinstrument.

Met het ontwikkelen van de *computer* vlak na de tweede wereldoorlog drong meer in het algemeen het besef door dat datgene wat de mens in zijn hoofd heeft zitten, in feite niets anders kon zijn dan een gigantische "calculator" ofwel *informatieverwerkende machine*. Vanuit de biologie en neurologie was inmiddels meer bekend geworden over de opbouw van de hersenen en met name over de werking van hersencellen. De belangstelling voor de "rekencapaciteit" van de hersenen resulteerde in de jaren vijftig en zestig in veel onderzoek aan de hersenen. De hersenencelnetwerken werden in de populaire literatuur van die tijd vaak vergeleken met op de computer gelijkende elektronische schakelingen, waarbij de cellen als "buizen" of "transistoren" werden voorgesteld. De metingen aan de cellen rechtvaardigden deze vergelijkingen overigens niet.

Door dit onderzoek kwam men tot het besef dat de "sleutel" tot de werking van de hersenen moest schuilen in de *samenwerking* van de hersencellen onderling. De cellen zelf bleken immers vrij "eenvoudige" objecten, die individueel niet verantwoordelijk konden worden geacht voor het gedrag van mens en dier. De focussering van het onderzoek verplaatste zich door dit inzicht langzaam van de werking van een enkele cel naar de werking van (grote) aantallen cellen. Helaas was over de *communicatie* tussen cellen en over de aard van het netwerk weinig (gemeten) data voorhanden. In deze situatie is overigens ook nu nog weinig verandering gekomen. Het is erg moeilijk zo niet onmogelijk een hersennetwerk intact te houden en er meetsignalen in te injecteren.

Een andere weg van onderzoek moest gevolgd worden. Bij de nieuwe aanpak ontwierp men (papieren) *modellen* van netwerken, waarna getracht werd te achterhalen welke rekenkundige capaciteiten deze netwerken bezaten. Als pioniers op dit gebied kunnen Hebb en Lashley worden genoemd, die experimenteerden met de eerste vormen van "gelaagde netwerken" en "gedistribueerd geheugen" (zie hoofdstuk 5). Neurologisch en psychologisch onderzoek dienden bij deze vorm van onderzoek slechts als achterafcontrole voor de plausibiliteit van een architectuur of cel-

(neuron)modellering. Door deze "eenvoudige" netwerken te begrijpen, hoopte men grotere en meer ingewikkelde netwerken te leren begrijpen.

Deze manier van aanpak leidde tot de eerste successen. In de vijftiger jaren ontwikkelden diverse onderzoekers zoals Rosenblatt en Selfridge al krachtiger mathematische modellen om netwerken te beschrijven.
Zo beschreef Rosenblatt een convergentieprocedure voor *perceptrons* (een eenvoudig netwerk bestaande uit een enkele laag niet onderling verbonden neuronen) waarmee bewezen kon worden dat bepaalde vormen van berekeningen door deze netwerken geleerd konden worden mits de netwerken voldoende "trainingsdata" werd voorgehouden.
Nu had men in die jaren het "voordeel" dat het werken met bestaande computersystemen nog niet zo een opgang had gemaakt als tegenwoordig, zodat er wat betreft "automatische gegevensverwerking" nog weinig concurrentie was. Dit leidde er toe dat men op basis van deze nieuwe kennis al snel tot overenthousiaste uitspraken kwam. Onder andere werd beweerd dat de netwerken tot berekeningen in staat waren die met gewone computers onmogelijk konden worden uitgevoerd.

Deze beweringen moeten als hoogst ongelukkig gezien worden. Het gevolg was namelijk dat er twee kampen ontstonden, de "neurale" en de "von Neumann" computertak.
Een sleutelrol in de strijd tussen deze twee kampen speelde Marvin Minsky, die zelf in het verleden zeer geboeid was geweest door perceptrons, maar teleurgesteld raakte door de beperkingen van deze netwerken. In de jaren zestig schreef hij samen met Seymour Papert een boek genaamd "Perceptrons" [62], waarmee hij middels enkele rigoureuze bewijzen aantoonde dat perceptrons maar een zeer beperkte klasse van functies konden uitvoeren. Interessante rekenkundige problemen, zoals een *exclusive or* functie, konden nimmer door het perceptron geleerd worden. Tevens beargumenteerde Minsky dat van complexere perceptrons met meerdere lagen niets extra's verwacht mocht worden. Minsky schreef:

"The perceptron has shown itself worthy of study despite (and even because of!) its severe limitations. It has many features that attract attention: its linearity; its clear paradigmatic simplicity as a kind of parallel computation. There is no reason to suppose that any of these virtues carry over to the many-layered version. Nevertheless, we consider it to be an important research problem to elucidate (or reject) our intuitive judgment that the extension is sterile."

Deze uitdaging aan het adres van de "connectionisten" (voorstanders van de neurale netwerken) kon niet worden beantwoord. Dit, samen met Minsky's reputatie en vooral de indrukwekkende prestaties en snelle opkomst van de "von Neumann" computer, gaf de "doodsteek" aan het op neurale architecturen gebaseerde onderzoek. Geld en mankracht werden overgeheveld naar de "symbolische" von Neumann computer, wat alleen maar tot meer succes van deze machine leidde. Op hun beurt deden de "computationists" nu al snel beweringen over de onbegrensde capaciteiten van hun machine.

Met uitzondering van de werkzaamheden van enkele personen als Grossberg, Anderson en Willshaw, lag het onderzoek naar neurale architecturen in de jaren zeventig zo goed als stil. Alle energie in het ontwikkelen van algoritmen werd gestopt in de symbolische programmering van von Neumann computers. Het perfecte geheugen van deze machine gecombineerd met de enorme rekensnelheid en het steeds goedkoper en sneller worden, was alleen maar aanleiding tot het vervolgen van deze weg.
Zelfs bij het simuleren van het menselijk kunnen met behulp van AI (*Artificial Intelligence*) systemen, wierp men zich vrijwel volledig op deze manier van programmeren. Niets leek het ontwikkelen van ware intelligente systemen met behulp van de "von Neumann" computer in de weg te staan. De computer werd gezien zijn enorme capaciteiten op het gebied van gegevensverwerking soms zelfs als superieur aan de mens afgebeeld. Geen mens immers die de machine nog kon verslaan of zelfs maar in de verste verte kon benaderen als het aankwam op rekenen of het opboesten van gegevens.

2.3 Hernieuwde belangstelling

In de jaren tachtig werd en wordt nog steeds enorm veel energie gestopt in het verder uitontwikkelen van hard- en software voor de "von Neumann" computer. Het wordt echter langzaam maar zeker duidelijk dat deze computer ook zijn grenzen heeft en dat deze zich aandienen.

Alhoewel men door toepassing van VLSI technologie electronische schakelingen al complexer en sneller kan maken zijn er limieten aan dit versnellen en verkleinen. De schakelingen worden tegenwoordig zo klein dat steeds duurdere apparatuur nodig is voor het produceren van de schakelingen, een ontwikkeling die echter momenteel (nog) ruimschoots gecompenseerd kan worden door de verkoop van steeds grotere aantallen chips. Materiaaleigenschappen en natuurwetten beginnen echter eveneens beperkingen op te leggen aan nog kleinere afmetingen. Zo betekenen dunnere draden op een chip automatisch hogere stroomdichtheden door de draden, wat weer meer (fatale) warmte-ontwikkeling oplevert. Hogere snelheden op de chip leiden automatisch tot hogere klokfrequenties, waarbij de VLSI electronica helaas minder goed functioneert zodat verdere verhoging niet zonder meer mogelijk is. Vertragingstijden bij het informatietransport, zelfs indien dit transport plaatsvindt met de snelheid van het licht, maken onbeperkt verhogen van de frequentie eveneens onmogelijk.

Behalve deze hardwarebeperkingen laat ook de huidige software-ontwikkeling te wensen over. In het vakgebied van de AI is al sinds langere tijd duidelijk dat de *klassieke logica*, met zijn formele redenatieschema's en notatie van kennis maar een zeer beperkte en bepaald onvoldoende modellering van "*menselijk redeneren*" weergeeft. De mens, die in een omgeving leeft waar veel onzeker, onnauwkeurig of zelfs tegenstrijdig is, redeneert niet volgens de modellen die gebaseerd zijn op de klassieke logica. Pogingen theorieën te ontwerpen die meer ruimte bieden om te redeneren volgens de informelere, meer "menselijke" methode

zijn maar op zeer beperkte schaal succesvol gebleken. Een goed overzicht hiervan wordt gegeven door Prade in [15].
Andere aspecten aan menselijke intelligentie zoals het *leren* en de *creativiteit* zijn nog minder grijpbaar voor de AI gebleken. Al met al is de huidige AI programmatuur nog bij lange na niet gelijkwaardig aan de mens, alhoewel indrukwekkende deelresultaten zonder meer bereikt zijn. Dit leidde tot een hernieuwd respect voor de taken die onze hersenen uitvoeren. Wellicht dat de mens (of een dier) inderdaad niet zo'n snelle rekenaar is, maar hij is wel bijzonder goed in het "overleven" in een hem vijandig gezinde wereld, een complexe taak waartoe de hersenen toch over een bijzonder grote hoeveelheid "rekencapaciteit" moeten beschikken.

Het bovengenoemde maakt in ieder geval twee dingen duidelijk, waarover momenteel dan ook algemene consensus bestaat, namelijk (1) dat het verder opvoeren van de rekencapaciteit van computers zal moeten geschieden door over te stappen op *parallelle* hardware in plaats van het alleen maar verder versnellen van de VLSI technologie en (2) dat wat betreft het ontwikkelen van (intelligente) software men vaak nog in de kinderschoenen staat en met veel respect dient op te zien naar de menselijke prestaties.

Tegelijkertijd met het verdwijnen van het ongebreidelde optimisme wat betreft de prestaties van de "von Neumann" hard- en software werden in het begin van de jaren tachtig enkele belangrijke doorbraken bereikt op het gebied van de neurale netwerken. Met name Hopfield ontwikkelde een nieuwe mathematische beschrijving van neurale netwerken waarmee het mogelijk bleek sommige netwerken te beschouwen als zeer snelle, parallelle "zoekmachines" naar minima in een "energie"landschap. Simulaties van deze algoritmen op de inmiddels goedkoop geworden computer versnelden het inzicht in de nieuwe resultaten.

Bij het bestuderen van dergelijke "Hopfield" netwerken bleek dat deze voor veel onderwerpen toepasbaar waren en een veelheid van primitieve

applicaties konden op basis van deze netwerken worden ontworpen. Het bekendste voorbeeld hiervan is het bouwen van *associatieve geheugens* met behulp van neurale cellen (zie voor een populaire inleiding en applicatie bijvoorbeeld [23]) maar ook andere applicaties als het oplossen van het "traveling salesman" probleem (zie bijvoorbeeld [17]), of een logische puzzle (zie 5.6 in dit verslag) bleek met dit soort netwerken mogelijk. Als "klap op de vuurpijl" werd bovendien ongeveer tegelijkertijd een leeralgoritme ontwikkeld, genaamd de *back propagation regel*, waarmee meerlagige netwerken wel degelijk problemen zoals het "exclusive or" probleem konden leren oplossen, daarmee volledig beantwoordend aan de twee decennia eerder gestelde "uitdaging" van Minsky.

Deze theoretische en praktische successen, gecombineerd met de roep om parallellisme als dé hardware oplossing voor de toekomst plus het teruggekeerde respect voor de door de hersenen geëxecuteerde "software" en de aantrekkingskracht en hoeveelheid "magie" die er toch al heerst rond het functioneren van onze hersenen, leidden vanaf 1986 tot een explosieve belangstelling voor neurale netwerken. Ter illustratie hiervan de volgende gegevens (verkregen uit [59]):

De internationale conferentie over neurale netwerken (INNC) in Santa Barbara in 1986 trok 50 deelnemers. De eerste IEEE conferentie hierover (IEEE NNC) in San Diego in juni 1987 trok echter reeds 1500 deelnemers en leverde 200 bijdragen op. In het voorjaar van 1987 organiseerde AAAI eveneens een conferentie over neurale netwerken in Seattle en in november werd door IEEE een conferentie over neurale netwerken gehouden in Bouldor. In Europa werden in november en december 1987 lezingen gegeven in Rome, Parijs, München en London.
In populaire bladen verscheen een groot aantal artikelen over deze onderwerpen (onder andere [6], [7], [8], [10], [12], [17], [20], [22], [23], [28], [58], [59], [60]) in de periode van dit schrijven. Nieuwe tijdschriften over dit onderwerp ontstonden of werden aangekondigd en een groot aantal kleinere maar ook grotere firma's (in totaal circa 200)

richtten zich op het vervaardigen van "neurale" hard- en software (bijvoorbeeld Hecht-Nielsen Neurocomputer Corporation (San Diego), Nestor Inc. (Rhode Island), Verac Inc. (Vancouver), NCI (New Yersey), Neuraltech Inc. (Portola valley), Neuronics Inc.(Chigago), SAIC (Tucson) en grotere bedrijven als TI en TRW).

De grote vraag is uiteraard of al deze belangstelling in neurale netwerken nu al gerechtvaardigd is. Aan de ene zijde heeft men de beloften van de prestaties van een werkend exemplaar volgens deze filosofie en een aantal veelbelovende mathematische en praktische resultaten, aan de andere zijde zijn er nog geen applicaties buiten de onderzoekssfeer ontwikkeld en worden in veel populaire artikelen beweringen gedaan die valse verwachtingen wat betreft de prestaties van de huidige netwerken wekken.
De taak van dit verslag is meer inzicht in deze situatie te verschaffen.

2.4 Neurale netwerken en radar

De ontwikkelingen op radargebied zoals geschetst in de inleiding vullen op een aantal manieren bovenstaand overzicht aan.

Radar kenmerkt zich wat betreft de signaalverwerking door de grote hoeveelheden informatie die "real-time" verwerkt moet worden. Door een qualitatief beter verwerken van de informatie te eisen, zoals werd aangegeven in de inleiding, loopt men al snel tegen de beperkte rekencapaciteit aan die beschikbaar is op een huidige computer. Parallellisme is noodzakelijk wil men betere signaalverwerking toepassen en rechtvaardigt onderzoek naar onder andere neurale netwerken, daar juist zij interessante aanknopingspunten voor parallelle hardware bieden.

Wil men radarbeelden interpreteren met behulp van automatische machines dan zullen er nieuwe algoritmen ontwikkeld moeten worden die iets weergeven van de "moeiteloze" informatieverwerking waarover de mens

beschikt. Er is een groot aantal taken tijdens het waarnemen die een mens met weinig moeite uitvoert en waarvoor momenteel geen goede algoritmen bekend zijn. Een voorbeeld hiervan is het herkennen van typen objecten aan de hand van gebrekkige, onnauwkeurige of gestoorde informatie. Een (getrainde) operator is hiertoe vaak redelijk goed in staat maar de machine levert dit grote problemen op.
Het onderzoek naar dit soort algoritmen rechtvaardigt dus eveneens onderzoek naar neurale netwerken, omdat met de bestudering hiervan wellicht meer inzicht wordt verkregen in de algoritmen die de mens gebruikt.

Er is nog een aantal andere eigenschappen die neurale netwerken interessant maken voor radarapplicaties. Hierbij valt onder meer te denken aan de *betrouwbaarheid* van neurale netwerken. Het is bekend dat sommige typen netwerken nog steeds redelijk goed functioneren indien een groot aantal willekeurig gekozen cellen is uitgevallen (in de orde van zelfs enige tientallen procenten). Deze eigenschappen lijken zeer voordelig in omgevingen waar beschadigingen optreden (militaire radar) of extreem lange levensduur (ruimtevaart) of hoge betrouwbaarheid (luchtvaart) wordt geëist. Ook de mogelijkheid tot zelfstandig *leren* door deze netwerken lijkt veelbelovend naarmate de autonomie van systemen toeneemt en/of weinig a priori bekend is en adaptie vereist is.

Van een werkelijke opleving in de belangstelling voor "neuraal" radaronderzoek is echter nog niet veel te bemerken. Onbekendheid met en het prille stadium van het neuraal onderzoek kunnen hiervoor als oorzaken worden genoemd
Wel zijn reeds de eerste voorzichtige stappen gezet (zie bijvoorbeeld [24]). Het is de taak van dit verslag te bezien of er niet meer gedaan kan worden, zelfs met de huidige stand van zaken.

2.5 Vooruitzichten

Wat de toekomst ons zal brengen hangt af van een groot aantal factoren. Allereerst is men op theoretisch gebied nog lang niet zover dat men de processen die zich in de hersenen afspelen volledig kan begrijpen. Of nieuwe inspanningen resultaten zullen opleveren laat zich slechts raden, maar dit is ongetwijfeld rechtstreeks afhankelijk van de hoeveelheid geïnvesteerde energie.
Tevens is het nog maar de vraag in hoeverre "neurale machines" inzetbaar zullen zijn. Het lijkt onwaarschijnlijk dat dit soort machines bijvoorbeeld erg goed zal blijken in zeer precieze taken als het vermenigvuldigen van getallen met tien cijfers achter de komma. En het is ook de vraag of men genoegen kan nemen met een machine die "hier en daar" wel eens een foutje maakt, zoals de mens dat immers ook doet.
Alhoewel respect voor de hersenen gepast is, is het niet reëel te verwachten dat zelfs met het copiëren van de hersenarchitectuur, (indien dat al mogelijk blijkt) alle problemen zullen zijn opgelost.
Veel zal ook afhangen van de ontwikkelingen op hardwaregebied. Indien het inderdaad waar blijkt dat de huidige ("von Neumann") architectuur vastloopt op fundamentele grenzen als de snelheid van het licht en als bovendien technologieën ontwikkeld kunnen worden die (goedkoop) parallelle machines weten te realiseren, bijvoorbeeld door gebruik te maken van biologische of optische hardware, dan kunnen de neurale netwerken mogelijk de huidige computer gaan evenaren of zelfs gaan overschaduwen. Een meer symbiotische verhouding is overigens eveneens mogelijk en momenteel nog even waarschijnlijk!

3 PARALLELLE SYSTEMEN

3.1 Inleiding

Alvorens de details van neurale netwerken te behandelen zal in dit hoofdstuk worden bezien waarom parallelle systemen, en neurale netwerken in het bijzonder, recentelijk zoveel aandacht krijgen. Welke zijn de problemen met seriële dataverwerking en waarom kunnen parallelle systemen voor verbetering zorgen?

3.2 Problemen met de seriële architectuur

In hoofdstuk 2 werd reeds aangegeven dat de huidige generatie seriële systemen de trend naar steeds krachtiger en snellere computers niet zullen kunnen volhouden. Daarvoor werd reeds een aantal redenen gegeven. In deze paragraaf zal hierop verder worden ingegaan.

De seriële "von Neumann" computer is ontworpen tijdens en vlak na de tweede wereldoorlog. De opbouw van de computer laat ook duidelijk zien dat zij in die tijd ontworpen is (of dit nu toeval is of niet). Hiermee wordt bedoeld dat de machine bestaat uit drie geheel aparte onderdelen, te weten de *processor*, het *geheugen* en het *communicatiekanaal* tussen geheugen en processor (zie ook figuur 3.1). Deze architectuur stemt overeen met de destijds voor handen zijnde technologie voor het maken van de processor (middels transistoren of buizen), het geheugen (middels magnetisch materiaal) en het kanaal (draad). Deze verschillen in technologie tussen processor, geheugen en kanaal onderschrijven als het ware een "von Neumann" architectuur.

De boven beschreven opbouw heeft een aantal zeer aantrekkelijke eigenschappen, die het mogelijk maken efficiënt gebruik te maken van de processor (de van oudsher "dure" component). De processor is namelijk constant bezig met het ophalen en wegzetten van instructies en data uit het geheugen. Doordat het per definitie onmogelijk is met deze

architectuur tegelijkertijd in het geheugen te lezen en te schrijven (immers de CPU kan maar één *instructie* tegelijkertijd uitvoeren) doen zich geen "timing" problemen voor. Moeten bijvoorbeeld de instructies x:=2*y en y:=3*x worden uitgevoerd, dan zal het eindresultaat van de waarden voor x en y afhankelijk zijn van de volgorde waarin deze instructies zijn opgenomen. Deze volgorde ligt echter bij de von Neumann machine eenduidig vast, zodat interpretatieproblemen zich nooit zullen voordoen. Dit vastliggen van de volgorde van instructies door het nooit tegelijkertijd optreden van meerdere processoracties maken dat de processor (in principe) ook niet zal hoeven te wachten voor toegang tot het geheugen of in een ander soort van timing-probleem zal geraken, zodat het vrijwel al zijn kracht op het eigenlijke data verwerken kan richten.
Aan deze opzet kleeft echter ook een aantal nadelen. In de eerste plaats is de processor wel efficiënt bezig met het executeren van instructies, maar een groot deel, zo niet het grootste deel van de tijd behelzen die instructies het ophalen en wegzetten van data/instructies. In feite is dit "verloren" tijd, er wordt niet werkelijk "bewerkt" in deze tijd.
Bij complexe programma's zullen met de toename van de complexiteit in de regel ook de afmetingen van het geheugen toenemen. Daardoor zal de afstand tussen de geheugenplaatsen en processor toenemen, wat leidt tot een verlaagde processorsnelheid, omdat deze langer moet wachten tot de data werkelijk ter plekke is. In moderne supercomputers vormt deze wachttijd het meest belangrijke knelpunt (ook wel de "von Neumann bottleneck" genoemd) voor het ontwerpen van nog snellere computers.
Ook de interne snelheid van de processor kan niet onbeperkt worden opgevoerd wegens technologische beperkingen aan de werking en fabricage van VLSI chips (tegen redelijke kosten).

Er kan echter ook op een andere manier kritiek worden geleverd op de "von Neumann" architectuur.
Alhoewel de fabricage van processoren wel iets ingewikkelder is dan die van geheugenchips, kunnen beide zonder bezwaar worden gecombineerd op een enkele chip! Het is vanuit technologisch oogpunt dan ook helemaal

niet meer noodzakelijk te denken in afzonderlijke termen als geheugen en processoren, daar deze technologisch gezien gelijk zijn.
Zo bezien is het dan zelfs erg jammer de strikte scheiding tussen geheugen en processoren aan te houden. Het dure chipmateriaal van het geheugen ligt er met deze architectuur immers maar ongebruikt bij. Zo nu en dan wordt een bepaalde geheugencel uitgelezen, dit in scherpe tegenstelling tot de processorchip, die de hele tijd druk bezig is. De data die in het geheugen ligt opgeslagen wordt op die manier maar gedurende een fractie van de tijd werkelijk gebruikt. De data ligt in feite "passief" niets te doen.

Een derde klasse van kritiek op deze architectuur kan worden geformuleerd in termen van betrouwbaarheid. Het kleinste foutje in de hard- of software zal er voor zorgen dat de computer totaal, maar dan ook totaal, van de kaart zal raken. Nu wil het gelukkige "toeval" dat men middels de VLSI technologie over extreme betrouwbaarheidseisen kan beschikken. Toch is het zelfs bij het gebruik van deze hoogwaardige technologie gebruikelijk dat 90% tot practisch 100% van de gefabriceerde chips defect is. Een miniscuul foutje maakt de gehele chip waardeloos en totaal onbruikbaar. Dit is hinderlijk tijdens de produktie, maar helemaal fataal is het indien deze fouten optreden tijdens het in bedrijf zijn van de computer.

In een moderne chip worden vele "trucs" toegepast om problemen van de eerste klasse te omzeilen, of beter, om hier de grenzen te verleggen. Zo kan men het communiceren van processor en geheugen versnellen door bijvoorbeeld een aantal lees cq schrijf acties achter elkaar te plegen. De tijd voor het "openen" en "sluiten" van het datakanaal heeft men dan slechts 1 maal voor de hele "trein" van gegevens. Zo kan de CPU bijvoorbeeld steeds 100 opeenvolgende instructies binnenhalen, deze in een wachtrij in de processor stoppen en ze vervolgens uitvoeren. Door het optreden van (verre) spronginstructies leidt dit niet altijd tot een verbetering en moeten zorgvuldige afwegingen worden gemaakt tussen blokgroottte, de grootte van de sprongen, enz.

Het gebruik van een lokaal geheugen in de chip zelf kan in deze situatie uitkomst bieden en de huidige trend in het processorontwerp is dan ook om veel van dit locaal geheugen op de chip zelf aan te brengen. Op soortgelijke manier kunnen vaak geraadpleegde data in een locaal geheugen (registers) worden opgeslagen.
Dit zijn uiteraard bruikbare "trucs" voor het versnellen van computers, maar de genoemde bezwaren blijven bestaan en zijn nu ook van toepassing op het locale geheugen. Een en ander is bovendien beperkt toepasbaar, want met de toename van de grootte van het locale geheugen ontstaan weer dezelfde problemen als voorheen! Als belangrijke bezwaren gelden bovendien dat er een enorme hoeveelheid "verkeersregeling" toegepast moet worden om alles correct te laten verlopen want er moeten ophaal- en wegzetstrategieën worden toegepast, er dient opgepast te worden voor het dubbel optreden van data (data die zowel in locaal geheugen als in globaal geheugen aanwezig is) en bovendien moet de programmatuur liefst van tevoren helemaal worden aangepast aan de snelheidseisen (of "on line" door bij te houden hoe vaak een stuk data gebruikt wordt).
Dit alles zou waarschijnlijk in het geheel niet tot snelheidsverbetering leiden, ware het niet dat voor dit soort taken vaak complete "dedicated" stukken hardware op de chip gebouwd kunnen worden, die de eigenlijke processor van dit werk ontlasten.

De tweede en derde klasse van problemen, die van de passiviteit van het geheugen en de extreem hoge eisen aan de betrouwbaarheid van hard- en software, lijken fundamenteel voor de "von Neumann" architectuur. Wat men bijvoorbeeld kan doen om de betrouwbaarheid op te voeren is simpelweg alles dubbel uit te voeren, zodat, indien een fout in de hardware optreedt, de reservehardware kan worden ingeschakeld. Het is echter in het algemeen zo goed als onmogelijk tijdig te ontdekken dat een fout optreedt in de hardware.

Al met al lijkt de "von Neumann" architectuur wellicht aan het eind van zijn kunnen te komen en "knutselen" aan de architectuur met "trucs" als bovengenoemde zal mogelijk niet tot werkelijk significante

snelheidsverbeteringen leiden. Sommigen richten hun hoop dan maar op andere, snellere technologieën, zoals het toepassen van licht in plaats van electrische stromen, maar de bezwaren tegen de architectuur blijven ook dan gehandhaafd. Bovendien is het onduidelijk hoe componenten in dit soort exotische technologieën gemaakt moeten worden, om nog maar te zwijgen van zaken als massaproductie, energieverbruik, afmetingen en, niet in de laatste plaats, kosten.

Het bovengeschetste beeld is nogal somber wat betreft de "von Neumann" architectuur en dat geeft dan ook precies de stemming weer waarin tegenwoordig over deze computer gesproken wordt (zie bijvoorbeeld artikelen als [20][63][64][65][66]).
Er is echter wel degelijk een weg naar "fundamenteel" snellere architecturen denkbaar (alhoewel niet bewezen kan worden dat deze architecturen voor alle problemen snellere programmatuur zullen opleveren), maar deze weg vereist dat het seriële "von Neumann" concept verlaten wordt.
Deze weg, vaak de "only way out" [20] genoemd, volgt een andere aanpak voor het doen van berekeningen. Dit is de weg voor het doen van *parallelle* bewerkingen in tegenstelling tot de hierboven geschetste *seriële* methode.

3.3 Klassiek parallellisme

Om de gangbare vormen van parallellisme te verduidelijken, wordt hier (zoals vaak gebeurt) als analogie van een computer een hypothetische fabriek gedacht. Deze fabriek zal dus dienst doen als *metafoor*, dus als vergelijkmiddel om duidelijk te maken waar voor- en nadelen van een parallelle architectuur liggen. Uiteraard moet de analogie niet te ver worden doorgetrokken, een metafoor dient slechts "ter lering ende vermaak".

In de seriële "von Neumann" fabriek staat een (zeer) flexibele robot (de CPU), die geacht wordt alle handelingen in de fabriek te verrichten

(databewerking) (zie Figuur 3.2). Hiertoe gaat de robot als volgt te werk: allereerst haalt hij een metalen plaat oid. uit het magazijn (gegevens uit het geheugen). Dan gaat de robot naar een machine (co-processor?) en boort een aantal gaten (bewerkt de data). Vervolgens gaat hij wederom naar het magazijn, legt de plaat weg, haalt een tweede plaat, bewerkt deze en legt hem weer weg. Tenslotte haalt hij beide platen op plus een schroef en een moer, verbindt de platen met een speciale machine hiervoor en legt het resultaat weer weg. Dit soort bewerkingen blijft zich herhalen en op het laatst wordt het kant-en-klare-produkt door de robot in het magazijn opgeborgen. Nu begint de robot aan een volgend werkstuk.

Er is geen practische fabriek die op deze "von Neumann" manier te werk zou gaan. Het probleem met de geschetste opzet is dat er veel te veel heen en weer wordt gelopen door de robot plus dat de hele fabriek platligt als de robot een bepaald onderdeel mist, exact dezelfde bezwaren dus als werden aangevoerd bij de seriële "von Neumann" dataverwerking (om de analogie hechter te maken zouden de omschrijvingen van wat de robot met het materiaal moet doen, het programma, ook in het magazijn moeten liggen).

Hoe zou men het produktieproces in deze fabriek kunnen versnellen? Hiervoor zijn veel mogelijkheden voorhanden. Zo kan men de robot sneller maken of meerdere artikelen tegelijkertijd uit het magazijn laten halen, dan naar de werkplaats laten lopen, de zaken daar neerleggen in een locaal magazijntje en aan de gang kunnen gaan "zolang de voorraad strekt". Deze versnelling komt overeen met de bovenstaande aanpak en heeft dezelfde soort nadelen als boven beschreven voor de "von Neumann" aanpak.

Een meer parallelle aanpak is echter eveneens mogelijk. Zo zou men meerdere robots dezelfde cyclus kunnen laten doorlopen.
Elke robot rent nu heen en weer en tracht zijn product te maken. Dit levert een versnelling op, alhoewel de drukte in de gangen (of eventueel

meerdere gangen) en het magazijn beperkend zal worden. Eventueel kunnen daarom meerdere magazijnen worden opgericht, alhoewel dit wel weer extra hardwarekosten (magazijnen en gangen) met zich meebrengt (Figuur 3.3). Verder ontstaan er problemen als robots tegelijkertijd machines willen bedienen en er moet al helemaal voorkomen worden dat zij er onbedoeld met elkaars halfproducten vandoor gaan. Een goede administratie en coördinatie is dus noodzakelijk.

Deze oplossing komt overeen met een soort van *multitasking* oplossing met behulp van *Multi Instruction Processoren* (MIP). Vrij onafhankelijke processoren voeren alle hun eigen programma uit, gebruikmakend van een centraal of decentraal geheugen (magazijn).
Deze aanpak werkt vooral goed als de robots maximaal onafhankelijk zijn. Is er voor een bepaalde produktiestap hulp nodig van een tweede robot, of zijn er halfprodukten nodig van een andere robot, dan loopt deze aanpak al snel spaak, tenzij er veel wordt gedaan om er voor te zorgen dat robots elkaar kunnen vinden en op elkaar zullen wachten. Dit is dan helaas wel verloren (robot)tijd.

Al met al blijft het een rennen (datatransport) van jewelste in de fabriek, waardoor veel kostbare robot (CPU) tijd verloren gaat. Een andere vorm van parallelliteit kan hier uitkomst bieden. Indien men het centrale magazijn opdoekt maar overstapt op het gebruik van een "lopende band", kan men volstaan met kleine, locale magazijntjes. De robots staan zo goed als stil en de produkten leggen maar een klein afstandje af tussen de robots (of andersom) (Figuur 3.4).
De productiesnelheid is nu net zo snel als die van de langzaamste robot, althans als er voor elke productiestap exact 1 robot is.
Heeft men minder robots dan zal men de robots steeds naar urgente plaatsen moeten leiden en zal men moeten werken met gebufferde tussenstations die opgenomen worden in de band. Heeft men meer robots dan plaatsen aan de band, dan zullen deze geen extra nut hebben.
Eventueel kan men op knelpunten dubbele banden opstellen of zelfs de gehele band meerdere malen uitvoeren.

Deze aanpak komt overeen met *pipeline* processing. Ieder processor doet zijn eigen taak (*Single Instruction Processoren*, SIP) en de processoren staan in een strikte volgorde. Deze aanpak werkt vooral goed als alle stappen aan de band even snel gemaakt kunnen worden. "Vector" processing kan soms op sommige plaatsen worden toegepast, dwz op sommige stations aan de band kunnen meerdere robots staan. Moeten bijvoorbeeld 10 schroeven worden ingedraaid dan kan men de lengte van de band verkorten en de produktiesnelheid verhogen door 10 robots op één plaats simultaan schroeven te laten indraaien.

Een indeling zoals die hier gemaakt is naar MIP en SIP machines wordt in de literatuur veel gemaakt. Ofwel men maakt gebruik van ingewikkelde, vrij onafhankelijke MI processoren en tracht tegelijkertijd zoveel mogelijk taken te doen, ofwel van gespecialiseerde, strak gecoördineerde, SI processoren. Een mengeling van de aanpakken is uiteraard ook mogelijk.

In de "geschiedenis" van super-computing en parallellisme zijn vormen van parrallelliteit zoals bovengenoemde al vaak uitgeprobeerd. De meest toegepaste vorm van parallellisme wordt gevonden in "pipeline" SI (en aanverwante "vector" processing) oplossingen. Deze architectuur loopt echter vast op zijn langzaamste stap en blijft kwetsbaar voor beschadigingen van de band. Bovendien kan er niet "paralleller" gewerkt worden als er stappen in het berekenings-/bewerkingsproces zijn. Het aantal processoren dat efficiënt parallel kan werken blijft hierdoor laag (enkele).
Een wellicht flexibeler vorm van parallelliteit wordt geboden door MI parallelliteit, dus middels ingewikkelde processoren die desnoods per stuk het gehele probleem zelfstandig kunnen oplossen. Deze architecturen doen het goed op machines waar "onafhankelijke" processen op draaien, zoals "multi-user" computers, of bij problemen waar "zoekbomen" zijn op te splitsen in onafhankelijke takken. Transputers lenen zich dan goed voor een dergelijke architectuur. De resultaten worden echter minder als

de taken korter en meer afhankelijk worden, daar er dan veel tijd verloren gaat door onderlinge communicatie (wie doet wat en wanneer). Als vele processoren aan een taak moeten werken loopt de werkelijk nuttig bestede rekentijd hierdoor drastisch terug.

Als inleiding op het denken volgens een "neuraal-netwerk"filosofie zal hieronder een derde fabriek en vorm van parallellisme worden geschetst. Dit zal leiden tot het ontwerp van een "gedistribueerd parallelle" (en vrij bizarre!) fabriek. (De term "gedistribueerd" wordt hier gehandhaafd in overeenstemming met de literatuur maar verschaft mijns inziens weinig extra inzicht. De term "chaotisch parallellisme" geeft naar mijn idee meer de kern van het ontwerp weer.)

3.4 Gedistribueerd parallellisme

Gegeven de bovenstaande fabrieksmetafoor wordt de fabriek nu als volgt herbouwd. Uitgegaan wordt van robots die gespecialiseerd zijn in één taak (of hooguit een aantal taken) zoals het indraaien van een schroef. Deze robots worden nu niet zoals voorheen in een nette produktielijn gezet, noch gaan zij geheel individueel aan de slag, maar zij worden op een *willekeurige* plaats op de productievloer geplaatst! Tussen de robots worden lopende banden gelegd, eveneens volgens een *willekeurig* patroon. Elke robot heeft een *aantal* ingaande en uitgaande banden (Zie figuur 3.5) ter beschikking.

Als op een inkomende band een halfproduct verschijnt (of op meerdere banden een combinatie van halfprodukten) dat de robot kan behandelen, zal hij deze handeling als regel gaan uitvoeren, zo niet, dan laat hij het halfprodukt ongewijzigd. Vervolgens zet de robot het al of niet behandelde halfproduct op een *willekeurige*(!) uitgaande band. Daar elk halfproduct vroeg of laat wel een robot zal tegenkomen die het weer verder kan completeren, zullen na verloop van tijd volledig afgebouwde producten over de banden schuiven. De fabriek "werkt" ondanks het *chaotische* karakter dus wel degelijk.

Op het eerste gezicht lijkt deze opzet van de fabriek nogal vreemd en inefficiënt. Toch biedt zij een aanzienlijk aantal voordelen.

In de eerste plaats is er totaal geen "timing" tussen de robots noodzakelijk. Zij voeren hun eigen handeling uit zonder zich te bekommeren om de rest. Er is slechts een uiterst minimale *coördinatie* tussen de robots, maar toch werkt het geheel blijkbaar samen, dat wil zeggen, het product wordt wel degelijk door de robots samen gemaakt.

In de tweede plaats is het helemaal niet erg als er een robotstation stuk raakt of tijdelijk is uitgeschakeld. Er zullen andere robotstations zijn die hetzelfde kunnen, dus zolang het transport op de banden maar doorloopt gaat de productie gewoon verder, alhoewel met verminderde snelheid. Daarmee zijn de voornaamste nadelen van strenge timing en gevoeligheid voor beschadigingen van de SI opzet (een enkele productielijn) overkomen.

In de derde plaats maakt deze opzet eenvoudig parallelliteit mogelijk door meerdere robots van hetzelfde type op te nemen. Hierdoor zal een product vaker zo een soort robot tegenkomen, waardoor deze stap in de productie versneld wordt. De fabriek functioneert het beste als de robots in een mengverhouding in de fabriek geplaatst worden die is afgestemd op de frequentie (en tijdsduur) van de handelingen.

In de vierde plaats kan men op een heel eenvoudige manier de productiesnelheid opvoeren. Hiertoe voegt men domweg meer robots en banden toe (het liefst in bovengenoemde verhouding) of er worden meer halfproducten op de banden gezet, het liefst in die verhouding zodat een halfproduct vrij vaak een robot passeert (te weinig is zonde van het "niets doen" van de robot maar ook niet te vaak, want dit zou tot opstoppingen in de fabriek leiden, waardoor het erg lang zal duren voordat een product klaar is, het staat immers lange tijd in

wachtrijen). Maar ook als de fabriek wat dit betreft over- of ondervoerd wordt gaat de productie wel degelijk gewoon door.
Problemen met het niets doen van robots kunnen overigens ook worden opgelost door de band sneller te laten lopen. Op die manier komen er meer halfproducten voorbij en duurt het minder lang voordat een werkloze robot weer iets bruikbaars voorhanden krijgt. Het is in het geheel niet erg als de band zo snel gaat dat producten, waarmee de robot wel wat had kunnen doen, toch ongemoeid voorbijrazen omdat de robot nog bezig is. Zij komen immers nog wel een keer langs bij deze of een soortgelijke robot.

In de vijfde plaats kan, mits het productieplan dit toelaat, eveneens worden afgeweken van een vaste volgorde van handelingen. Als het niet uitmaakt of schroef A eerder dan B geplaatst wordt, zal dit ook in een willekeurige volgorde in de fabriek mogen gebeuren. Dit komt de snelheid weer ten goede.

In de zesde niet onbelangrijke plaats tenslotte kan de fabriek met een kleine aanpassing zijn efficiëntie vrij eenvoudig verhogen door de robots adaptief te maken. Immers, indien een robot zelf meet hoe vaak een bepaald halfprodukt langskomt kan het bijvoorbeeld besluiten een halfprodukt wat erg vaak langskomt te gaan behandelen en zich hiertoe "omschakelen". Blijkbaar immers is er behoefte aan dat type robot op die plaats. Op deze wijze kan een soort van evolutionair proces, de "struggle for product", mogelijk zorgen voor een optimale verdeling van typen robots over de fabriek.

Op deze manier worden veel van de voordelen van MI en SI opzetten behouden terwijl de nadelen (gecompliceerde MI robots, robots die lange tijd niets doen of op elkaar wachten, gevoeligheid voor fouten, strenge communicatie tussen de robots enz.) voor een groot deel verdwijnen. Opvallend is het gemak waarmee de productiekracht kan worden opgevoerd (meer robots toevoegen), en de enorme foutongevoeligheid van de fabriek (zolang er nog maar een robot van elke soort bestaat gaat de productie

door). Tevens is het opvallend hoe "dom" de robots kunnen zijn (een soort van SI robots die op een willekeurige manier producten van de band halen en op banden zetten). Het is ook in het geheel geen probleem als een robot een groot deel van de tijd faalt. Vroeg of laat komt een product de robot, of een robot van een gelijke soort, wel weer tegen.

Uiteraard komen er wel nieuwe problemen kijken bij deze vorm van productie.
Als eerste is te noemen de *explosie* van bandmateriaal (communicatie-kanalen). Als er niet voldoende banden zijn, kunnen niet goed willekeurige wegen door de fabriek gevolgd worden, wat de productiesnelheid hindert.
In de tweede plaats kunnen er slechts *stochastische garanties* gegeven worden over de tijd die nodig is om een product te voltooien. We hebben hier immers te maken met stochastische routes door de fabriek! Overigens kan de stroom van gereedgekomen producten wel redelijk constant zijn. Gemiddeld wordt er vrij constant geproduceerd (bij een voldoende grote fabriek) door de "wet van de grote aantallen".
Ten derde zijn de robots wel wat *gecompliceerder* dan bovengenoemde "pipeline" SI robots. Zij moeten immers kunnen herkennen of een halfproduct in een bepaald geschikt stadium verkeert. Dit hoeft echter niet al te ingewikkeld te zijn. In ieder geval kan de robot veel eenvoudiger zijn dan een MI robot.

Het wordt aan de lezer overgelaten de metafoor verder uit te werken. Voor meer betreffende de klassiek parallelle machines wordt verwezen naar [64].

3.5 De hersenen als een gedistribueerd parallelle machine

Zonder hier in detail op de structuur van, en de processen in, de hersenen in te gaan, kan reeds worden opgemerkt dat de opbouw van de hersenen opvallende overeenkomsten toont met bovenstaande *gedistribueerde* (of chaotische) opzet.

Alhoewel de hersenen in staat zijn ingewikkelde problemen op te lossen als spraak, beeldherkenning, coördinatie van de ledematen of het spelen van een partij schaak, zijn de processen waartoe de hersencellen in staat zijn van een opmerkelijk simpele aard (zie hoofdstuk 4) die in geen verhouding staan met de complexiteit van de op te lossen problemen. De deelsystemen, de cellen zijn dus eerder van het SI dan het MI type en gezien deze eenvoud en hun traagheid moet er in de hersenen van verregaande parallelliteit sprake zijn.
De verbindingen tussen de cellen zijn uiterst simpel en laten het versturen van complexe boodschappen niet toe. Communicatie tussen de cellen moet dus zeer eenvoudig van aard zijn. Voorts is er geen vorm van "timing" of althans nauwkeurige timing, er is geen "synchronisatie"-mechanisme, althans niet op cellulair niveau. Tenslotte zijn de cellen vrij onnauwkeurig: er is veel ruis en er is mogelijk ook een hoog uitvalpercentage. Het is ook uitgesloten dat bij een persoon de verbindingen tussen de cellen exact van tevoren gedefinieerd zijn, daarvoor is eenvoudigweg niet genoeg code voorhanden in het DNA en bovendien is deze exactheid hoogstwaarschijnlijk niet goed mogelijk met behulp van biologische groeiprocessen.
Alles wijst er dus op dat parallelliteit door de hersenen verregaand gebruikt wordt, maar dat een vorm van "klassiek parallellisme" niet plausibel is. De cellen zijn geen ingewikkelde processoren die delen van het probleem gecoördineerd kunnen oplossen (daarvoor zijn zij te simpel) en evenmin is er sprake van een strakke "pipeline" van cellen waar het probleem er "voor" in gaat en er "achter" uit komt (daarvoor zijn zij niet strak genoeg georganiseerd). Beide "klassieke" vormen van parallellisme hebben daarom waarschijnlijk weinig gemeen met de vorm van parallellisme die de hersenen op cellulair niveau gebruiken.

De gedistribueerde filosofie sluit echter qua eigenschappen wèl goed aan bij de opbouw en onderdelen van de hersenen. Het *chaotische* patroon van de verbindingen, de ongevoeligheid voor ruis en beschadiging, de eenvoud van de cellen en de communicatie tussen de cellen zijn allemaal van een

soortgelijke aard als bij de "gedistribueerde fabriek". Vandaar dat het gerechtvaardigd lijkt te veronderstellen dat de hersenen mogelijk werken volgens een gedistribueerd parallelle filosofie.

Het is moeilijk om op dit punt aan te geven welke exact de kenmerken zijn van gedistribueerd parallellisme.
Kenmerkend van een gedistribueerde opzet is steeds dat men het "geheel" opsplitst in delen, die stuk voor stuk simpel en onnauwkeurig zijn, maar gecombineerd leiden tot goede resultaten. De "robots" of "cellen" zijn erg eenvoudig en kennen slechts een rudimentaire vorm van onderlinge communicatie. Door de "wet van de grote aantallen" functioneert het geheel echter toch goed en snel. Het resultaat van de opzet is dat het geheel efficiënt is (de deelsystemen zijn "druk bezet"), dat de resultaten beter worden naarmate er meer deelsystemen worden toegevoegd (en dat deze toename liefst vrijwel lineair is!), dat de deelsystemen bij tijd en wijle best fouten mogen maken en tenslotte, dat geen enkel deelsysteem essentieel is en het systeem daarmee "extreem" fout-ongevoelig is.

Het zijn deze eigenschappen die gedistribueerd parallellisme veelbelovend lijken te maken. Dit is hopelijk voldoende toegelicht in de voorgaande beschouwingen. De vraag rest uiteraard hoe precies deze systemen ontworpen en geprogrammeerd moeten worden, zodat zij ook werkelijk functioneren.
Deze vraag zal in de rest van het verslag hoofdthema zijn bij de behandeling van een bepaalde klasse van gedistribueerd parallelle systemen, de neurale netwerken.

4 CELLEN IN NEURALE NETWERKEN

4.1 Inleiding

In het voorgaande is in algemene zin gesproken over gedistribueerd parallelle systemen, met een minimum aan technisch detail. In dit en de volgende hoofdstukken zal meer specifiek worden ingegaan op theoretische en technische aspecten van de gedistribueerde systemen die in dit verslag het onderwerp van studie zijn: de neurale netwerken.
In dit hoofdstuk wordt de "basis" van een neuraal netwerk behandeld: de neurale cel.

4.2 Celmodellering

Een parallel gedistribueerd systeem in de vorm van een Neuraal Netwerk (NN) bestaat in principe uit niets meer dan met elkaar verbonden cellen (plus eventuele input- en outputkanalen met de buitenwereld). Er is dus geen extern geheugen of rekencapaciteit of iets van dien aard. De bedrading (wat is met wat verbonden) plus de celwerking (hoe worden de inkomende signalen verwerkt tot uitgaande signalen) specificeren dus in het geheel de werking van het NN. De neurale cellen zijn als het ware geheugen en processor tegelijk.

Deze schijnbare eenvoud laat echter een enorme ruimte aan de daadwerkelijke implementatie wat betreft de bedrading en de celwerking over, zodat ondanks deze "eenvoud" het bestuderen en ontwerpen van NN bij lange na niet triviaal is.

Een centrale rol in het netwerk spelen dus de cellen. Zij bepalen hoe binnenkomende signalen worden verwerkt tot nieuwe signalen en wat de aard van deze signalen zijn. Een model van de celwerking vormt dan ook de kern van elke theorie rond een NN.

Intermezzo

Bij het opstellen van een model kan men op twee manieren te werk gaan. Men kan uitgaan van theoretische overdenkingen en daarmee modellen opstellen (uitgangspunt: wat zou een cel moeten doen) of van biologische modellen (uitgangspunt: wat doet een cel dat ik moet nabootsen).
Wellicht dat de "zuivere theoreticus" huivert bij de aanpak volgens een biologische modellering, maar beide aanpakken hebben ontegenzeglijk hun voordelen en kunnen de andere aanpak weer ondersteunen. Een theoretische aanpak heeft als voordeel dat de processen die optreden meestal goed begrepen zijn. Als nadeel geldt echter dat men maar moet hopen op een toevallige goede inval van de theoreticus om verder te komen. De vorderingen kunnen langzaam zijn of zelfs helemaal de verkeerde richting op gaan.
Een "biologisch" getinte aanpak kan leiden tot snelle resultaten en zal zeker een bron zijn voor het ontwikkelen van nieuwe ideeën, maar moet uiteraard wel worden ondersteund door theoretisch onderzoek.
Een citaat uit [46] in dit kader:

"Theorists almost always assume that they are cleverer than natural selection,
This is usually a mistake".

De mengelmoes van biologische en theoretische overwegingen kenmerkt veel van het werk in NN'en en wordt ook in dit verslag teruggevonden.

4.3 Een algemeen biologisch celmodel

Figuur 4.1 toont een biologische neurale cel. Functioneel worden onderscheiden:

- De inkomende verbindingen (de dendrieten)
- Het cellichaam (het neuron)

- De uitgaande verbinding (de axon)

Als regel veronderstelt men dat een cel één enkele uitgaande en meerdere inkomende verbindingen heeft. Door deze verbindingen lopen de (unidirectionele!) signalen waarmee de cellen met elkaar communiceren. Gezien de beperkingen van de chemische technologie is de communicatie over de lijnen simpel. De biologische kanalen kunnen alleen "aan"- of "uit"signalen vervoeren (mogelijk gelden er op deze regel uitzonderingen, bijvoorbeeld in het visuele systeem. Zie [46]).
Dit wil overigens niet zeggen dat complexere berichten dan "uit" of "aan" niet mogelijk zijn. Door frequentiecodering (te vergelijken met binaire codering) kunnen wel degelijk ingewikkelder berichten worden uitgewisseld. Dit bereikt de cel door sneller of minder snel "aan"pulsen te versturen. Verondersteld wordt echter wel dat deze processen onnauwkeurig zijn, zodat de te verzenden informatie niet meer dan enkele bits groot is.

De ingaande en uitgaande lijnen zijn met elkaar verbonden. De plaats waar het contact tot stand komt noemt men de "synaps". De synapsen koppelen de inkomende signalen als het ware door naar het cellichaam. Afhankelijk van het type synaps en de afmetingen van de synaps zal deze doorkoppeling verschillende effecten hebben.
Onderscheiden worden exalterende en inhiberende synapsen. Exalterende synapsen winden de cel als het ware op. Komen op een lijn verbonden met een exalterende synaps signalen binnen, dan zal de aangesloten cel "actiever" worden, dat wil zeggen met een hogere frequentie uitgaande signalen gaan versturen. Inhiberende cellen doven of verlagen juist de celuitgangsactiviteit. De mate van opwinden of doven is verschillend per synaps en bovendien afhankelijk van de frequentie van het inkomende signaal.
Het cellichaam verwerkt de vele inkomende signalen en bepaalt daarmee de uiteindelijke uitgangsactiviteit. De cel voert daarbij een soort van integratie van de inkomende signalen uit. Aan de hand van de gevonden

gewogen "som" zal de cel actie ondernemen door aan de uitgang zijn vuurfrequentie te veranderen.
Ruisprocessen in de cel kunnen eveneens bijdragen tot het veranderen van de vuurfrequentie. Ook de oude toestand van de cel (het verleden) kan een rol spelen in het functioneren in de toekomst. Deze "korte-termijn"-geheugenfunctie van de cel dient in een NN echter beperkt te zijn. Mogelijk is in de cel informatie aanwezig over bijvoorbeeld de vuurfrequentie in het korte verleden, maar het is niet waarschijnlijk dat bijvoorbeeld de toestand van de afgelopen tien seconden intern ligt opgeslagen.

Bovenstaand model schetst het globale "prototype" van een neurale cel. Natuurlijk zouden door de natuur andere keuzen gemaakt kunnen zijn (bijvoorbeeld wat betreft de vorm van de communicatie). Vanuit simulatietechnisch oogpunt zal het model ook vaak worden aangepast. Zo zal de vuurfrequentie bijvoorbeeld meestal worden vertaald naar meer praktische reële getallen, maar het is uiteraard niet nodig of zelfs maar wenselijk een te strikte analogie met een biologisch neuraal netwerk na te streven.

Ondanks de grote speelruimte binnen het model van een neurale cel is er met dit model toch een aantal keuzen gemaakt. De belangrijkste hiervan is de keuze voor het versplinteren en vermengen van rekencapaciteit en geheugen. Door het ontbreken van uitgebreide buffermechanismen in de cel is de enige manier om langdurig informatie in een NN op te slaan ofwel de sterkte van de synaptische overgangen te veranderen en/of de vorm van de integratiefunctie te veranderen. In de meeste artificiële NN zien we beide vormen van lange-termijngeheugen toegepast worden. Men kan dus stellen dat de te herinneren informatie in een NN decentraal ligt opgeslagen in de verbindingen tussen de cellen. Hetzelfde geldt voor de reken- of processorcapaciteit, die wordt vertegenwoordigd door de gewogen integratiefunctie van het cellichaam (zie 4.5).

4.4 Hersencellen

In deze paragraaf zal kort bezien worden hoe menselijke hersencellen bovenstaande verder "invullen". Voor meer detail en verdere verwijzingen zie [1][46][72].

Globaal voldoen de hersencellen aan het model van een enkele uitgaande en meerdere inkomende verbindingen. Deze "bedradingen" worden in het begin van het leven van een organisme gelegd en zijn verder "vast" (naar wordt aangenomen). De sterkte van de synapsen is echter wèl onderhevig aan veranderingen, zoals werd gemeten in vele experimenten waarbij bijvoorbeeld door chirurgische ingrepen stukken hersenen onbruikbaar werden gemaakt. Na verloop van tijd blijken andere stukken van de hersenen de verloren taak dan te kunnen overnemen [46]. Details over de synaptische leermechanismen zijn echter nauwelijks voorhanden.
Per cel kan het aantal verbindingen variëren van plusminus honderd tot enkele tienduizenden. Het is niet bekend hoeveel van deze verbindingen werkelijk functioneel zijn en evenmin hoeveel redundantie er aanwezig is. Gezien de onbetrouwbaarheid van biologische systemen mag worden aangenomen dat de percentages redundante verbindingen mogelijk zeer aanzienlijk kunnen zijn.
De inhiberende of exalterende werking van een synaps hangt af van het type "transmitter" (een chemische stof) en de hoeveelheid van de transmitter die in een synaps voorkomt. Hierover is slechts weinig bekend (zie [72] voor een recent overzicht). De mate van opwinden of doven is mogelijk afhankelijk van het oppervlak van het synaptische contact (wat onderling globaal tot een factor tien verschilt). De invloed van een bepaald type transmitter kan zeer kort (enkele msec) tot vrij lang (seconden, minuten of zelfs uren) zijn.

De signalen die over de verbindingen vervoerd worden hebben de vorm van "pulstreinen". De frequentie van die treinen kan oplopen tot maximaal 100 a 200 hertz. De duur van een enkele puls ligt in de orde van 1 msec.

Het cellichaam verwerkt de signalen. Hierover is niet veel meer bekend dan dat er sprake moet zijn van een vorm van integratie, immers, hoe meer exalterende synapsen actief zijn, hoe meer de cel zich "opwindt" (met een hogere frequentie "vuurt") en vice versa. Veel details over de celwerking zijn er echter niet.

Het cellichaam zelf is enkele micrometers groot. De verbindingen strekken zich uit over afstanden van typisch een millimeter tot maximaal een centimeter. Deze verbindingen zijn overigens niet totaal "random" maar vertonen globaal gezien wel degelijk "structuur". De cellen liggen enigszins in "lagen" opgeslagen. Zie hiervoor hoofdstuk 5 en [46]. In de hersenen zijn in totaal ongeveer 100 miljard hersencellen aanwezig. Mogelijk is hiervan slechts een gering percentage functioneel.

4.5 Celmodellen

Wanneer men bovenstaand celmodel meer detaillistisch wenst in te vullen ontstaan de eigenlijke "problemen". Neuronaal onderzoek levert maar zeer karige aanwijzingen over de signaalverwerking. Een veelheid van modellen over de celwerking is in het verleden beproefd.
Een aantal modellen wordt hieronder gepresenteerd (een meer volledig overzicht wordt gegeven in [33]):

4.5.1 Differentiaalmodellen

De oudste modellen zijn vooral gebaseerd op biologische gegevens. Voorbeelden hiervan zijn de op differentiaalberekening berustende modellen als die van Grossberg en Anderson. De variabelen komen overeen met hoeveelheden van chemische stoffen in de cel. Terugkoppelingen in de cel zorgen ervoor dat de variabelen geen oneindig grote waarden aannemen. Zie bijvoorbeeld [1][3][19][2]).
Een typisch voorbeeld van zo een model wordt gegeven door formule [4.1ab]

$$dS_j/dt = \sum_i S_i(t)*W_i(t) - K_1*S_j(t) - T_j \qquad [4.1a]$$

$$dW_i/dt = K_2*S_i(t)*S_j(t) - K_3*W_i(t) \qquad [4.1b]$$

In [4.1a] verandert de uitgang (S_j) van de cel j naar gelang de toestand van de andere cellen (S_i) middels een gewogen (W_i) inputsommatie. Tevens is een drempel(threshold)waarde T_j van invloed die hier tijdinvariant is gekozen, waarmee S_j verschoven kan worden. Om de activatiewaarde S_j beperkt te houden is er een automatische afname (terugkoppeling) van activiteit middels K_1. De gewichten (W_i) veranderen ook met een snelheid middels [4.1b]. De grootte van de verandering is hier afhankelijk van de grootte van de correlatie met de buurcellen. Ook de gewichtsgrootte wordt beperkt middels de terugkoppelingsterm K_3.

Tot voor enige jaren (zie bijvoorbeeld de in september/october 1983 uitgegeven IEEE TMC) bediende men zich voornamelijk van dit soort vergelijkingen voor het doen van onderzoek aan NN'en. Bij het bestuderen van eigenschappen ziet men zich dan geplaatst voor de enorme complexiteit van stelsels van (niet lineaire) DV's (discreet dan wel continu).

Alhoewel deze modellen biologisch gezien wellicht zeer plausibel zijn, moge het duidelijk zijn dat de DV's zich niet lenen voor verregaande berekeningen. Deze situatie is te vergelijken met het rekenen aan analoge electronische systemen. Worden deze systemen groter dan enkele elementen, dan levert de DV aanpak geen bruikbare resultaten meer. De werking wordt onbegrijpelijk en bewijzen zijn niet te verkrijgen.

Het is niet verwonderlijk dat berekeningen met dit soort modellen in het verleden niet (of slechts zeer moeizaam) tot resultaten hebben geleid. Hooguit kon bijvoorbeeld worden aangegeven welke waarden van K_1,K_2,K_3 tot interessante "activiteit" van een netwerk leiden. Het werken met

stelsels DV's vindt tegenwoordig weinig doorgang meer bij het bestuderen van NN'en.

4.5.2 Lineaire modellen

Het wiskundig meest eenvoudige model beroept zich op lineaire signaalverwerking. De signalen S_j zijn weer positieve of negatieve reële getallen.
Een cel i is ook hier weer verbonden met een andere cel j middels een synaps met waarde W_{ij}. De nieuwe uitgangswaarde van een cel wordt eenvoudig berekend uit

$$S_j(t+1) = \sum_i W_{ij} * S_i(t) - T_j \qquad [4.2]$$

De lineariteit van dit model maakt het mogelijk de werking te evalueren middels de krachtige lineaire algebra.

Helaas is er door de strikte lineariteit weinig interessants te verwachten van netwerken bestaande uit dit soort cellen. De eindtoestand is immers altijd eenvoudig een lineaire transformatie van de begintoestand. Ook indien men de uitgang van het netwerk terugkoppelt naar de ingang levert dit niets extra's op, daar de overdracht tussen de uiteindelijke uit- en ingang een lineaire functie blijft (zie ook [33]) [Opmerking: indien rekening wordt gehouden met tijdvertragingen in de cellen en verbindingen ontstaat wel ander (dynamisch) gedrag door terugkoppeling. Hier wordt echter geduid op de eindtoestand van het netwerk, dus nadat de toestand van het netwerk stabiel is geworden.] Het gebruik van dit soort cellen in NN is dan ook beperkt, tenzij men geïnteresseerd is in het uitvoeren van lineaire mappings. Interessante mappings zijn als regel echter niet lineair.

De eenvoud van het model blijft echter aantrekkelijk. Goed gefundeerde leerregels voor netwerken van dit soort cellen (leer een lineaire mapping met een bepaalde eigenschap) kunnen eenvoudig worden afgeleid. Lineaire systemen, gecombineerd met een leerregel, worden "lineaire associatoren" genoemd en zijn in het verleden onder andere onderzocht door Anderson en Kohonen. In hoofdstuk 5 komen deze wederom ter sprake.

4.5.3 Niet lineaire modellen

Het zwakke punt van lineaire netwerken, namelijk dat alleen eenvoudige lineaire transformaties kunnen worden uitgevoerd, kan overkomen worden door het toevoegen van niet lineariteiten. De eenvoudigste mogelijkheid hiertoe is het toevoegen van een drempel(threshold)functie aan de uitgang. De cel staat "aan" indien de totale gewogen input groter is dan een drempelwaarde, in het andere geval staat de cel "uit". Ofwel

$$S_j(t+1) = f\left(\sum_i W_{ij} * S_i - T_j\right) \qquad [4.3]$$

$f(x) \quad = 1$ als $x \geq 0$
$f(x) \quad = 0$ als $x < 0$

met f een functie die de "aan"waarde oplevert als zijn argument groter dan nul is en in het andere geval de "uit"waarde. f is dus gelijk aan de tekenfunctie, die middels T_i kan verschuiven rond een omslagpunt.

Het bereik van wiskundige functies die met dit soort cellen kunnen worden gemaakt is veel groter dan die van de lineaire associator. Zo kan bijvoorbeeld een exclusive or mapping worden uitgevoerd, een mapping die met puur lineaire cellen niet mogelijk is (zie hoofdstuk 5).
Een nadeel van het werken met niet-lineariteiten is dat de mathematica moeilijker wordt. De lineaire algebra kan niet langer worden gebruikt voor de berekeningen. Er is dan ook bijvoorbeeld geen algemeen leer-algoritme bekend voor netwerken met deze cellen. Voor een overzicht van

netwerken met dit soort niet-lineaire cellen zie bijvoorbeeld [21] en hoofdstuk 5 van dit verslag.

Er zijn behalve de tekenfunctie uiteraard ook andere keuzen voor f mogelijk. Indien men in plaats van slechts een "aan"- en een "uit"waarde de celtoestand met een reëel getal wenst weer te geven, hebben andere f-keuzen zin. Zo kan men bijvoorbeeld kiezen voor

$$f(x) = x \quad \text{voor } x \geq 0$$
$$f(x) = 0 \quad \text{voor } x < 0 \qquad [4.4]$$

Veel onderzoek wordt momenteel verricht aan nette continu stijgende maar begrensde functies als

$$f(x) = \arctan(x) \qquad [4.5a]$$

of

$$f(x) = 1 \,/\, (1 + e^{-x}) \qquad [4.5b]$$

Leeralgoritmes voor het leren van de gewichten en de drempels van cellen met deze uitgangsfunctie (of een andere continu stijgende functie) zijn recentelijk beschikbaar gekomen (de back-propagation rule, zie hoofdstuk 5 en [8][9][36][71]).

De hoofdmoot van de huidige netwerken maakt gebruik van cellen van het hier geschetste type. Helaas worden de gewonnen capaciteiten van dit soort netwerken, verkregen door de niet lineariteiten, "betaald" door een moeilijker mathematisch te bestuderen gedrag.

4.5.4 Statistische modellen

Een vrij nieuwe ontwikkeling bij het bestuderen van NN'en is gebruik te maken van de resultaten verkregen uit de *thermodynamica* en *statistische mechanica*. Op het eerste gezicht komt dit wellicht vreemd over, maar nadere beschouwing van deze vakgebieden maakt de band duidelijk. Bij deze vakgebieden bestudeert men immers deeltjes die zich in een beperkt aantal toestanden kunnen bevinden die worden beïnvloed door de toestanden van buurdeeltjes. In feite komt dit nauw overeen met de NN-filosofie.
Zo bevinden zich in een dunne plak magnetisch materiaal kleine gebiedjes die ofwel omhoog danwel naar beneden gemagnetiseerd kunnen worden, wat analoog gedacht wordt aan de twee toestanden "vuren" of "niet vuren" van een neurale cel. De gebiedjes kunnen van toestand veranderen indien het magnetisch veld dat gegenereerd wordt door omliggende cellen (equivalent met het gewogen gemiddelde van buurcellen) een bepaalde drempelwaarde (threshold) overtreedt. De berekeningen hierbij zijn van een zelfde vorm als bij de niet-lineaire cellen zoals hierboven behandeld. Als extra restrictie bij deze beschouwing geldt overigens wel dat de invloed van een deeltje A op een deeltje B gelijk dient te zijn aan die van B op A. Aldus wordt de eis van symmetrische verbindingen tussen de cellen geëist, wil de analogie bruikbaar zijn!

Thermodynamische of statistisch mechanische modellen vormen aldus een bruikbare analogie en de mathematica uit deze gebieden leent zich daarmee voor het beschrijven van celmodellen en NN'en. Bruikbare begrippen uit deze vakgebieden als Energie en Temperatuur zijn eveneens te gebruiken voor de bestudering van NN'en. Bij de "thermodynamische" cellen maakt men gebruik van het begrip temperatuur om de ruis in een netwerk te modelleren. Een van de verrassende ontdekkingen is dat deze ruis behulpzaam of zelfs noodzakelijk kan zijn voor de werking van een netwerk (zie hoofdstuk 5).

Bij een "thermodynamische" cel kiest men als uitgangsfunctie meestal

$$p(f(x)=1) = 1/ (1+ e^{-x/T}) \qquad [4.6]$$

met $p(f(x)=1)$ de kans dat de cel hoog wordt gegeven x. Figuur 4.2 geeft een schets van deze kansfunctie bij verschillende "temperaturen". Door de temperatuur te verhogen of te verlagen kan meer of juist minder kanskarakter worden ingebracht in de werking van de cel (overeenkomend met de interne ruis in de cel). Bij een temperatuur van nul graden is het gedrag gelijk aan die van een cel met een tekenfunctie (zie [4.3]). Bij een oneindig hoge temperatuur is de celtoestand volledig willekeurig. Immers, volgens [4.6] is de kans op een bepaalde celtoestand dan gelijk aan vijftig procent, ongeacht de grootte van de inputs. Merk op dat de uitgang van de cel hier discreet, maar de kansfunctie continu is!

Het interessante van dit soort cellen is dat de verdelingen van de aan- en uitstaande cellen zich net zo gedragen als de verdelingen van bijvoorbeeld aangeslagen en niet aangeslagen electronen. Deze verdelingen lenen zich erg goed voor berekeningen met de hulpmiddelen die reeds ontwikkeld zijn voor berekeningen aan statistisch mechanische systemen. Bewijzen over het gedrag van het NN laten zich relatief eenvoudig formuleren. Men kan de "toestand" van het netwerk vertalen naar een "energie"maat. Deze manier van beschrijven maakt het gedrag van het netwerk begrijpelijk (net zoals energie het gedrag van bijvoorbeeld een gas "begrijpelijk" maakt) en berekenbaar. Het rekenen met behulp van statistische gegevens en het geven van statistische garanties is immers veel eenvoudiger dan het voorspellen van het gedrag van de afzonderlijke cellen in een netwerk.

Niet alleen levert deze manier van rekenen voordelen op, ook de netwerken met dit soort elementen hebben extra's te bieden. Zo is het bijvoorbeeld gelukt middels deze netwerken logische problemen te laten oplossen (zie hoofdstuk 5). Evenzo is men in staat gebleken met dit soort netwerken goede oplossingen te genereren voor complexe problemen als het "handelsreiziger"probleem (traveling salesman problem).

Leeralgoritmes voor dit soort netwerken zijn eveneens bekend [69][38] maar zeer traag en praktisch minder goed bruikbaar. Goede introducties tot typen netwerken met dit soort cellen, de Harmony en de Boltzmann netwerken, worden gegeven in respectievelijk [37] en [38]. Meer detail wordt gevonden in [69] [16] [48] [53] [68] [42] [45] en [67].

In de populaire literatuur krijgt dit soort netwerken helaas nog nauwelijks aandacht, zij vormen mijns inziens echter de potentieel meest veelbelovende netwerken voor complexe toepassingen van NN'en.

Een nadeel van dit soort netwerken is de eis van de (biologisch niet plausibele) symmetrische verbindingen tussen de cellen en de onduidelijkheid over de "temperatuurregulatie" of ruismaat in het netwerk. Het is echter wellicht mogelijk de resultaten te generaliseren naar niet symmetrische verbindingen. Hoe dit te doen en wat voor extra netwerkeigenschappen dit zal opleveren is nog niet bekend.

4.5.5 Cellen met tijdvertraging

In voorgaande modellen was altijd sprake van een "onmiddellijke" integratie. In de cel of in de kanalen trad geen "delay" op. Uiteraard zijn dit soort delays wel in te brengen en zijn deze mogelijk zelfs zinvol. Op het moment van dit schrijven staan dit soort modellen in de belangstelling maar hebben nog geen theoretische onderbouwing. Er zal in dit verslag niet verder worden ingegaan op dit soort modellen.

4.6 AI en het celmodel

Dit hoofdstuk wordt besloten met enige op de AI gebaseerde overdenkingen bij het verschijnsel "cel". De AI richt zich al gedurende langere tijd op het ontwikkelen van "intelligente", aan de mens gelijkwaardige software. Het mag daarom niet verwonderlijk heten (alhoewel het wellicht niet noodzakelijk is) als de AI in de loop der tijd structuren heeft ontwikkeld die gelijkenis vertonen met de "celnetwerken".

Een "AI" visie op een cel zou kunnen zijn dat de cel een soort van "micro-expert" is. Hiermee wordt bedoeld dat de cel een oordeel velt (aan of uit, waar of niet waar) op basis van kennis (de gewichten en de drempel) en binnenkomende informatie (de uitgangen van andere cellen oftewel "micro-experts").
Een micro-expert heeft nu de taak een oordeel te vellen over een bepaalde toestand die in zijn microwereldje optreedt. De toestand van de cel (of micro-expert) wordt hier geïnterpreteerd als "ik oordeel dat de toestand optreedt" als de cel "aan"staat en "ik oordeel niet dat de toestand optreedt" als de cel uitstaat. Als de cel aanstaat betekent dit dus dat een bepaald patroon is ontdekt. In het geval de cel uitstaat is deze informatie er niet (gebrek aan bewijs) of er is informatie binnengekomen dat het patroon niet optreedt. In het geval de cel uitstaat is het dus niet duidelijk of dit komt door gebrek aan bewijs of negatief bewijs!

Er zijn binnen de AI een aantal methodes bekend om te redeneren met "onzekerheden". De meest bekende is gebaseerd op de formule van Bayes [49]. Met deze formule kan men afwegingen maken op basis van onvolledig en onzeker bewijs. Andere algoritmen voor het werken met onzekerheid en onnauwkeurigheid worden gegeven in [15], het is dus zeker niet zo dat de Bayesiaanse aanpak de enig mogelijke is!

Uitgangspunt voor het gebruik van Bayes is dat indien men over een bepaalde hoeveelheid bewijsmateriaal E beschikt, het oordeel over een (gecorreleerd) gegeven H mag worden afgeschat volgens

$S = f(P(H|E))$

met f een oordeelfunctie die 1 of 0 oplevert (waar of niet waar) voor de hypothese H en $P(H|E)$ de kans dat H geldt (de hypothese waar is) gegeven bewijsmateriaal E (evidence). Een keuze voor f is bijvoorbeeld $f(P)=1$ te kiezen als P() groter is dan 50% en anders $f(P)=0$ te kiezen.

Indien E een enkel gegeven E_i is volgt volgens Bayes dat

$$P(H|E_i) = P(E_i|H)/P(E_i) * P(H) \qquad [4.7]$$

indien E_i waar is en
$P(H|E_i) = P(H)$
indien onbekend is of E_i waar of niet waar is. P(H) en $P(E_i)$ zijn de a priori kansen op het waar zijn van H en E_i. In het geval meerdere bewijsstukken E_i worden aangedragen ter ondersteuning van de hypothese H, leert Bayes dat P(H|E) geschreven kan worden als

$$P(H|E) = P(E_1|H)/P(E_1) * P(E_2|H)/P(E_2) * \ldots * P(E_n|H)/P(E_n) * P(H) \qquad [4.8]$$

mits E_1, E_2, enz. onafhankelijke gebeurtenissen zijn (zie [49][75]). Daar de a priori termen $P(E_1|H)/P(E_1)$, $P(E_2|H)/P(E_2)$ enz. en de a priori term P(H) van tevoren (middels eenvoudige correlatiemetingen) gemeten kunnen zijn kan P(H|E) berekend worden. Aldus kan op een basis van bewijs E een oordeel over P(H|E), dus de kans op H gegeven E, gegeven worden.

Formule 4.8 kan nu herschreven worden tot

$$\begin{aligned} P(H|E) &= \exp\{\ln\{ P(E_1|H)/P(E_1) * P(E_2|H)/P(E_2) * \ldots \\ &\qquad P(E_n|H)/P(E_n) * P(H) \}\} \\ &= \exp\{ \ln(P(E_1|H)/P(E_1)) + \ln(P(E_2|H)/P(E_2)) + \\ &\qquad \ln(P(E_n|H)/P(E_n)) - \ln(1/P(H)) \} \\ &= \exp\{ W_1 + W_2 + \ldots W_n - T \} \end{aligned} \qquad [4.9]$$

Het oordeel S_j uitgaande van bewijsstukken S_i kan met [4.9] dus geschreven worden als

$$S_j = f(\exp(\sum_i W_{ij} * S_i - T_j)) \qquad [4.10]$$

Deze vorm komt opvallend overeen met die van de basisformule [4.3] van de niet lineaire cel.

Met andere woorden, wellicht dat de "klassieke" AI en NN'en niet zo ver uit elkaar liggen als vaak wordt beweerd. Een samenvloeiing van de neurale netwerkvisie en de klassieke AI visie kan mogelijk tot goede resultaten leiden.

Het fraaie van een analogie als deze is bovendien dat zij ons direct leert hoe de gewichten gekozen dienen te woren (althans bij deze interpretatie). De gewichten zijn hier de samengesteld uit (nulde en eerste orde) correlatietermen. Merk overigens op dat de gewichten bij deze keuze eveneens symmetrisch zijn daar (per definitie)

$$P(E|H)/P(E) = P(H|E)/P(H) \qquad [4.11]$$

5 NEURALE NETWERKEN

5.1 Inleiding

Neurale cellen zijn relatief eenvoudige eenheden. Deze eenvoud maakt dat een neurale cel op zich weinig indrukwekkende taken kan vervullen. Vergeleken met bijvoorbeeld een computer zijn de capaciteiten zeer beperkt (zowel wat de reken- als wat de geheugencapaciteit betreft). De eenvoud van een neurale cel is echter geen nadeel maar juist de kracht achter het concept van neurale netwerken. Is de cel zelf een eenvoudig onderdeel, een net opgebouwd uit zeer veel van deze eenvoudige cellen kan taken vervullen die vele malen *complexer* zijn. Juist de combinatie van de eenvoud van de cellen (eenvoudige implementatie in electronische of chemisch/biologische technologie) en de complexiteit van het geheel is oorzaak van veel van de mogelijk aantrekkelijke eigenschappen van dit soort systemen en wellicht ook de oorzaak van het aantreffen van neurale netwerken bij mens en dier.

Het is verre van vanzelfsprekend hoe een net te ontwerpen. Het is geenszins duidelijk welke verbindingen tussen de cellen gelegd moeten worden, welk type cel in een net (voor een bepaalde taak) het best gebruikt kan worden en hoe de synapswaarden gekozen moeten worden.

Kiest men voor verschillende invullingen van bovenstaande vrijheden, dan ontstaan noodzakelijkerwijs verschillende netwerken met verschillende eigenschappen.
De vraag is uiteraard niet welke keuze "juist" is, want de natuur gebruikt neurale netwerken (de hersenen) voor zeer uiteenlopende taken als *zien*, *herinneren*, *redeneren*, *bewegen* enz. Voor elke soort taak zal een bepaald *type* netwerk optimaal zijn en het mag verwacht worden dat deze optimaliteit na de vele miljoenen jaren van evolutie door de natuur goeddeels bereikt is.

Onderzoek aan neurale netwerken is dus geen "speurtocht" naar het uiteindelijke net dat alle taken kan volbrengen, maar het zoeken naar een geschikt net voor een bepaalde taak.
Dit hoofdstuk is een afspiegeling van deze visie. Naar functionaliteit zal bekeken worden welke netwerken in de literatuur bekend zijn, welke hun eigenschappen zijn en welke modificaties eventueel gewenst zijn.

Overigens is dit niet de enige manier van aanpak voor onderzoek betreffende de werking van de hersenen. Men kan ook een visie nastreven waarbij men tracht te achterhalen wat het *geheel* van de hersenen doet. Hierbij tracht men dan niet uit cellen netwerken samen te stellen, maar door middel van deelnetwerken (of beter, deelsystemen) de functionaliteit van de gehele hersenen te beschrijven. Deze manier van aanpak bevindt zich dan echter op een ander niveau als het onderzoek naar neurale netwerken. Een inspirerend voorbeeld van een dergelijke opzet wordt gegeven in [14]. In dit verslag zal hierop niet verder worden ingegaan.

Ondanks de verschillen tussen de netwerken mag het toch niet als een verrassing komen dat door de toch "gelijksoortige" opbouw van de netwerken, verschillende netwerken vaak vergelijkbare eigenschappen hebben. Veel van de verkregen resultaten bij een bepaald type netwerk zijn dan ook (ten dele) toepasbaar op andersoortige netwerken. Ongevoeligheid voor beschadiging bijvoorbeeld is een eigenschap die met name bij *geheugennetwerken* naar voren wordt gebracht, maar ook min of meer blijft gelden voor andere netwerken. Leerregels zijn interessant bij *classificatoren* en geheugens en worden juist daar naar voren gebracht maar soortgelijke leerregels zijn ook van toepassing op netwerken die iets doen wat veel lijkt op *logisch redeneren*.
Het zou te ver gaan om bij elk type netwerk al deze aspecten telkens weer boven tafel te halen. Slechts die eigenschappen die het meest in het oog springen bij een netwerk, dat wil zeggen die het meest relevant geacht worden voor de taak, zullen worden uitgelicht. Het wordt aan de

lezer overgelaten, eventueel met behulp van de literatuurverwijzingen, een diepere analyse van een netwerk uit te voeren.

Opvallend in dit hoofdstuk is ook dat steeds verschillende rekenwijzen bij verschillende typen netwerken gebruikt worden in plaats van één enkele rekenwijze. Dit heeft meerdere oorzaken. Allereerst vormt dit verslag een afspiegeling van de gelezen literatuur en zijn er geen uitgebreide pogingen gedaan rekenwijzen om te zetten in een gemeenschappelijke vorm. Maar belangrijker is dat dit ook geen zin heeft, daar verschillende typen netwerken met verschillende typen cellen (bijvoorbeeld *continue* of *discrete* cellen) verschillende vormen van beschrijvingen aantrekkelijk maken. Daarom moet men niet verward worden door de verschillende benaderingen, maar dit zien als een noodzakelijk kwaad.

5.2 De opbouw van een neuraal net

De principes achter de opbouw van een neuraal net zijn eenvoudig. De verschillende keuzen die gemaakt kunnen worden bepalen de eigenschappen van het net.

De opbouw van een net behelst vier aspecten die hieronder behandeld worden.

Het eerste aspect betreft de *aard van de te gebruiken cellen*. Als regel zijn de cellen in één net alle van hetzelfde type, dat wil zeggen zij verrichten gelijksoortige wiskundige berekeningen. Slechts de inhoud van de synapsen en de drempels verschillen van cel tot cel. Ook het aantal ingangen zal als regel per cel verschillen.
Deze uniformiteit in een netwerk is onrealistisch als we dit vergelijken met biologische data. Het is in de hersenen aangetoond dat cellen onderling verschillen van aard. Met de hier gevolgde benadering wordt echter niet gedoeld op een neuraal net als een "geheel" maar als een onderdeel van een groter net. De uniformiteit van cellen op een meer

lokale grond is wel aangetoond. Aangezien hier slechts de aandacht uitgaat naar lokale netwerken, is het uitgangsprincipe van uniformiteit in de cellen binnen een netwerk daarom aanvaardbaar. De vraag welke de celfunctie moet zijn blijft echter open.

Het tweede aspect betreft de *verbindingen* tussen de cellen. In alle te behandelen netwerken wordt verondersteld dat een cel één enkele uitgang heeft die verbonden is met meerdere ingangen van andere of dezelfde cel. Een eenvoudige manier om dit te beschrijven is gebruik te maken van een twee-dimensionale matrix waar zowel horizontaal als verticaal alle cellen zijn uitgezet. Doorgaans is deze matrix bijzonder dun bevolkt (sparse). Tussen de 10^{11} neuronen in onze hersenen lopen naar schatting "slechts" 10^{14} verbindingen in plaats van de 10^{22} mogelijke verbindingen. Slechts 1 op de 100 miljoen plaatsen in bovengenoemde matrix is daarmee gevuld! Op een meer lokale schaal (waarin er hier interesse is) geldt echter een minder schaarse verdeling.
De aard van de verbinding zelf staat in de hier te behandelen netwerken overigens niet ter discussie. Gekozen wordt voor een verbinding zonder vertraging en zonder verzwakking. Ook deze aanname is "gevaarlijk" en niet biologisch gerechtvaardigd. Uiteraard treden in biologische systemen wel degelijk vertragingstijden op en mogelijk kunnen deze vertragingen een functie hebben. Verzwakking lijkt gezien de aard van de biologische verbindingen minder aannemelijk, alhoewel een "storing" in een langer kanaal waarschijnlijker gedacht mag worden dan een in een kort kanaal.
In sommige netwerken is het wellicht mogelijk dat dit soort "imperfecties" functioneel zijn. Met name in het low-level bewerkingsgebied (dicht bij de *sensoren* en *actuatoren*) zoals in het visuele gebied van de hersenen, kunnen vertragingstijden mogelijk kritisch zijn in bepaalde filter- of detectienetwerken (zie [25]). Hier zal echter verder worden aangenomen dat vertragingen en ruis in de kanalen geen rol spelen in een netwerk.
Ook *modulatie* van kanalen onderling komt in de te behandelen netwerken niet voor. Biologisch is echter wel degelijk aangetoond dat zulke

mechanismen voorkomen dwz. het "blokkeren" van een kanaal door een andere kanaal(!) is aangetoond. Dit verschijnsel komt echter voor zover bekend slechts sporadisch voor en zal hier niet van belang worden geacht (blokkeren kan overigens ook gemodelleerd worden door extra cellen toe te voegen aan het netwerk). Voor meer detail over de biologische aard van de verbindingen zie [46].
Behalve één enkele link tussen twee cellen, te beschrijven door een reële twee-dimensionale matrix (met daarin de synapswaarden), zijn uiteraard complexere verbindingen denkbaar. Zo kan een cel meerdere verbindingen met dezelfde cel hebben. In dit geval zijn hoger-dimensionale matrices nodig voor de beschrijving van de connecties. Evenzo is het mogelijk een overdracht door meer dan een enkel reëel getal te karakteriseren. In biologische synapsen immers is het gebruik van meerdere soorten *neurotransmitter* bij een enkele synaptische overgang aangetoond. Bij de te behandelen netwerken zal dit overigens niet ter sprake komen.
Een cruciale rol speelt de grootte van de synapsen, hun gewicht. Deze kan vast gekozen of geleerd worden. In dit verslag zullen meerdere leermechanismen ter sprake komen. Biologisch gezien is het aannemelijk dat synaptische waarden zowel voor een deel "voorgeprogrammeerd" (aangeboren) zijn als flexibel (geleerd).

Het derde aspect betreft het *activatiepatroon*, dat wil zeggen, de regel die bepaalt welke cel wanneer *verfrist* (*geupdated*, *gevuurd*) wordt. Dit activatiepatroon wordt in de literatuur slechts zelden expliciet vermeld, maar vormt een wezenlijk onderdeel van de werking van het netwerk en de bewijsvoeringen. Als verschillende voorbeelden van activatiepatronen worden hier die van het te behandelen *Hopfield geheugen* en het *Harmonie netwerk* genoemd. In het eerste netwerk wordt steeds één enkele willekeurig (*random*) gekozen cel bijgewerkt of gevuurd. Zowel het random kiezen als het na elkaar vuren is hier essentieel voor de werking en bewijsvoering! Bij het harmonienetwerk, dat bestaat uit twee "lagen" cellen, wordt afwisselend de ene en de andere laag parallel gevuurd.

Het tegelijkertijd vuren van alle neuronen wordt *synchroon vuren* en het na elkaar in een willekeurige volgorde vuren *asynchroon vuren* genoemd. Opgemerkt moet worden dat andere vormen van vuren meestal leiden tot slechts een "gedegradeerde" werking van het netwerk. De manier van vuren is in de praktijk dus wellicht minder kritisch als hier wordt aangegeven.
Biologisch gezien is elke vorm van *timing* in feite onwaarschijnlijk. Bij onderzoek is nimmer gebleken dat de cellen op een of andere wijze synchronisatiesignalen verkrijgen. Wel wordt hersenactiviteit vaak voorgesteld als golven van activiteit door het brein, maar deze golfbeweging wordt waarschijnlijk (?) eerder veroorzaakt door het doorgeven van ingangssignalen naar uitgangssignalen dan dat er sprake is van een "synchronisatie"lijn. Complexere modellen van vuren zijn uiteraard denkbaar. Hierbij valt bijvoorbeeld te denken aan "vermoeidheidsverschijnselen" in een cel die kunnen voorkomen dat een cel te snel achter elkaar vuurt. Voor meer detail hierover zie [46].

Het vierde en laatste aspect betreft de *communicatie met de buitenwereld*, dwz. de connecties met de sensoren en actuatoren. In dit verslag zal slechts worden aangenomen dat dit soort verbindingen, indien noodzakelijk, aanwezig zijn. De verbindingen met sensoren en actuatoren en de daarvan afkomstige signalen worden gelijksoortig gedacht aan die met de onderlinge cellen.

5.3 Biologische netwerken

Alvorens in de diepte van de veelheid van neurale netwerken te duiken, wordt op deze plaats kort stilgestaan bij enige biologische informatie over neurale netwerken (dat wil zeggen, betreffende de hersenen van zoogdieren). Net als in hoofdstuk 4 geldt dat de informatie hierover schaars is, wat veroorzaakt wordt door de moeilijkheid van de metingen, de complexiteit van de te meten processen, net parallelle karakter van de te meten processen en de bijna absolute onbekendheid met wat men denkt te meten (wat "doen" de hersenen?). Sommige (anatomische) details

zijn echter interessant, daar deze grote overeenkomsten vertonen met theoretische netwerken en een aanzet geven voor het ontwerpen van andere.

Figuur 5.1 toont een weergave van de menselijke hersenen. Opvallend is dat het overgrote deel van de massa van de hersenen gevormd wordt door de grote hersenen, de zogenaamde *cerebrale cortex* (CC). Aangenomen wordt dat de CC verantwoordelijk is voor al die processen waarin in dit verslag interesse is, zoals *vision, coordinatie, denken* enz. Andere taken, zoals de regeling van de organen vinden (waarschijnlijk) in andere delen dan de CC plaats en worden hier niet verder behandeld.

Het CC is zeker geen "soep" van willekeurig verbonden cellen.
De CC is opgebouwd als een "vel" van circa 1000cm^2 (ter vergelijking, in een macaque aap 100cm^2). Dit "vel" is circa 1.5 tot 5 mm dik en hevig "gefrommeld", waarschijnlijk eenvoudigweg om in de schedel te passen. Nergens in het vel zitten scheuren of lopen "ongevoelige" of "isolerende" zones. Wel is er een sterke deling in twee helften die verbonden zijn via een breed zenuwkanaal. Meer detail zal hier niet worden gegeven. Dit kan worden gevonden in [46] en de daarin gegeven verwijzingen.
De CC bevat naar schatting 10^{11} hersencellen die alle 100 tot 10000 verbindingen hebben. Overigens is het niet bekend welk deel van deze cellen en verbindingen functioneel is.

De opbouw van het "vel" is belangrijk voor dit verslag. Onderzoek heeft aangetoond dat het vel is opgebouwd uit meerdere lagen cellen, waarbij het soort vertakkingen en cellen per laag verschilt. Hierop zal later in deze paragraaf meer in detail worden teruggekomen.

Zoals gezegd vervult het CC een veelheid van taken. De best onderzochte (en meest gemakkelijk te onderzoeken) van deze is die van het zien. Door gezichtsprikkels toe te dienen kan "eenvoudig" getest worden welke delen van het CC verhoogde activiteit vertonen. Op soortgelijke wijze konden

in het verleden delen van de hersenen naar functionaliteit in kaart worden gebracht. Van diverse soorten apen zijn dit soort "lay outs" of "kaarten" (maps) gemaakt. Wat opvalt bij de kaarten is dat de grootte van het gebied, dat voor een bepaalde functie verantwoordelijk gedacht mag worden, afhankelijk lijkt te zijn van het belang van deze functie voor het organisme. Blijkbaar is het zo dat hier een "more is better" filosofie geldt: hoe belangrijker de functie, hoe meer cellen zich met deze fuctie bezighouden!
Zo zijn bij de aap grote gebieden gereserveerd voor de beweging van de tenen en armen (slingeren in bomen). Bij de mens geldt dit alleen voor de handen. Aan taken die belangrijker voor het organisme zijn wordt blijkbaar meer rekenkracht (cellichamen) en geheugen (synapsen) besteed.

Het is niet juist te veronderstellen dat deze kaarten geheel voorgeprogrammeerd zijn. De sterkste aanwijzing daarvoor is dat beschadigingen van hersengebieden er toe leiden dat andere gebieden deze functies kunnen overnemen. Het plaatsgebonden zijn van bepaalde functies is eerder het resultaat van het gevoelig zijn van bepaalde gebieden voor prikkels die min of meer "toevallig" in dat gebied uitkomen dan andersom. Met andere woorden, het is onwaarschijnlijk dat er sprake is van in detail "voorbedraden" door de evolutie, maar het is eerder zo dat sommige gebieden de neiging hebben bepaalde functies te gaan uitvoeren indien zij daartoe gestimuleerd worden.
Een mooi voorbeeld van dit "zelforganiserend" vermogen bij een aap, dat optreedt na chirurgische beschadiging aan de hersenen, wordt gevonden in [83].
Overigens geven ook verschillen tussen mensen onderling en de verandering van de lichaamseigenschappen tijdens bijvoorbeeld de groei aanleiding tot deze conclusie. De hersenen moeten hiertoe *adaptief* van aard zijn en niet star voorgeprogrammeerd, want na veel trainen verbetert de coördinatie zich. Dit geldt uiteraard niet alleen voor de coördinatie, want evenzo kan (men) de mens en het dier trainen op zeer veel andere gebieden.

Een belangrijke vraag aangaande de werking van de hersenen is of er een fundamenteel verschil is tussen het uitvoeren van *bewuste* en *onbewuste* processen. Het lijkt voor de mens zo te zijn dat er een verschil is in zaken als "ruiken", "zien", "voelen" enz. enerzijds en de interpretatie van deze processen als "ik zie dit", "ik ruik dat" anderzijds. De eerste "onbewuste", "low level" bewerkingen als contourverbetering of de detectie van beweging of geur geschiedt schijnbaar onwillekeurig: de mens heeft er geen controle over en deze processen spelen zich de gehele tijd af (zelfs tijdens slaap of narcose). Het *interpreteren* van de verkregen gegevens vereist dat de mens omschakelt naar een "bewuste" mode, die "aandacht" en "concentratie" vergt (althans, dit is zoals hij het ervaart).
Het hoe en waarom van het verschil tussen deze twee (of mogelijk vele) modes is compleet onbekend. Blijkbaar kunnen beide vormen in de mens naast elkaar bestaan. Het gegeven dat de onbewuste taken zonder problemen altijd en tegelijkertijd worden uitgevoerd en de bewuste mode *slechts één* (denk)proces toelaat zou er op kunnen duiden dat de onbewuste processen door hiervoor gereserveerde onafhankelijke stukken van de hersenen worden uitgevoerd, terwijl de bewuste processen een "multi-purpose" stuk van de hersenen in beslag nemen. Dit zijn echter slechts (interessante) hypothesen.
Wel is bekend dat "bewuste" processen altijd mimimaal tijdstappen vergen van circa 0.5 tot 1 seconde. Indien men dus besluit een bepaalde, niet willekeurige beweging te verrichten, vergt dit besluit dus minimaal 0.5 seconde. Hetzelfde geldt voor de tijd tussen het "zien" en "herkennen" van een afbeelding. Zie [4] voor meer detail.

Op de globale hersenprocessen zal hier niet verder worden ingegaan, met name ook gezien de hoogst speculatieve aard van deze processen. Het is hoogst verleidelijk uitspraken te doen over de globale werking van de hersenen en dit te vergelijken met eigenschappen van neurale cellen of netwerken zoals die hieronder gepresenteerd zullen worden. *Het is echter uiterst gevaarlijk een net met circa* 10^{11} *neuronen te vergelijken met een net van enkele tientallen tot duizenden neuronen.*

Meer detail over hersenprocessen wordt gegeven in [46][83].

De opbouw van het "vel" hersenmateriaal is interessant. Zoals gezegd is deze opbouw namelijk sterk "gelaagd". Figuur 5.2 toont een schets van de doorsnede van dit vel.
Onderscheiden worden:

Een opperlaag (superficial layer)
Deze opperlaag bevat weinig cellichamen en bestaat vooral uit axonen en dendrieten. De opperlaag is overal in de CC aanwezig.

De bovenlagen (upper layers)
De bovenlagen bevatten (kleine pyramide)cellen die hun axonen naar andere gebieden op het CC sturen (in dezelfde of de andere hersenhelft).De dikte van deze laag verschilt over het CC.

Een middenlaag (middle layer)
De middenlaag is niet overal aanwezig in het CC. Het bestaat uit dicht gepakte (stellaire) cellen met axonen die vooral doordringen in verticale richting en in de bovenlaag. De middenlaag is in grote delen van de hersenen weer zodanig dik en gelaagd dat een verdere onderverdeling in lagen gemaakt kan worden.

De diepe lagen (deep layer)
De diepe lagen tenslotte bestaan uit (grote pyramide)cellen met axonen die lopen naar actuatoren en sensoren (bijvoorbeeld via het ruggemerg) en andere delen van de hersenen dan de CC.

Een (gewaagde) conclusie uit bovenstaande is dat de onderste diepe laag dient voor communicatie met de rest van het lichaam, de middelste lagen het "denkwerk" verrichten en dat de bovenste laag zorgt voor communicatie tussen verschillende delen van het CC. Mogelijk is de middenlaag weer sterk "gelaagd". Bij de behandeling van de opbouw van

verschillende typen netwerken zal deze "gelaagdheid" eveneens naar voren komen.

5.4 Classificatienetwerken

In de volgende paragrafen zal getoond worden hoe neurale netwerken zich lenen voor *classificatie*. Het uitgangsprincipe in deze paragraaf is dat het te evalueren net deel uitmaakt van een groter geheel dat input naar en output van het netwerk verzorgt. De taak van het netwerk is steeds, gegeven de input, een aantal outputcellen aan dan wel uit te zetten, oftewel het netwerk dient de ingangsgegevens te "classificeren", waarbij de outputcellen staan voor de verschillende klassen.

Kenmerkend voor de hier te behandelen classificatienetwerken is het ontbreken van terugkoppeling in het netwerk. Terugkoppeling is "blijkbaar" niet nodig voor deze taak(?). Terugkoppeling zal pas geïntroduceerd worden in de zogenaamde "associatieve" (geheugen)netwerken in paragraaf 5.5.

In figuur 5.3 is een schets van een niet teruggekoppeld netwerk, hier "classificatie"netwerk genoemd (alhoewel het mogelijk ook andere taken kan vervullen) gegeven. De figuur toont hoe inputs en outputs verbonden zijn met de cellen. De cellen verbonden met inputs worden de *input units* genoemd, die met outputs de *output units*. Een cel kan overigens tegelijkertijd zowel input als output unit zijn. Een cel die niet verbonden is met een output of een input wordt een verborgen of *hidden unit* genoemd. De aard van de signalen (bijvoorbeeld {0,1} of het interval [0,1]) en de aard van de cellen kan per implementatie van een classificatienetwerk verschillen. In onderstaande zullen de meest gebruikelijke behandeld worden.

5.4.1 Het maximalisatienetwerk

Het *maximalisatienetwerk* wordt hier als eerste gepresenteerd als een "rudimentair" classificatienetwerk. Het maximalisatienetwerk (MN) heeft

als taak die cel "aan" te zetten die hoort bij de ingang met de grootste waarde.

Het eenvoudigste MN is getoond in figuur 5.4. De ingangen hebben continue waarden (aangenomen op het interval [0,1] alhoewel dit niet noodzakelijk is). De cellen zijn van het *hard limiter* type (zie formule [4.3]).De gewichten worden gekozen zoals getoond in 5.4 en de drempels van de cellen zijn gelijk aan nul.

De werking van het netwerk van figuur 5.4 is eenvoudig. De cel met een positief netto ingangssignaal zal "aan" gaan staan ten teken dat deze bijbehorende ingang de grootste waarde heeft. Een probleem ontstaat als beide ingangen exact even groot zijn. Afhankelijk van de gekozen celdefinitie zullen beide cellen dan aan- of uitstaan. Deze situatie dient dan geïnterpreteerd te worden als dat er geen "grootste" ingangswaarde is.

Een meer realistisch MN zal als taak hebben het grootste ingangssignaal van N ingangen te classificeren. De uitbreiding naar deze situatie is echter niet triviaal.

Een eerste stap naar een dergelijk netwerk is getoond in figuur 5.5. De toegevoegde cel is een "lineaire drempel" cel. Voor deze cel geldt dat de uitgang gelijk nul is indien de totale gewogen ingangssom kleiner is dan nul en anders gelijk is aan deze som.

De uitgang van deze cel is nu gelijk aan die van de grootste van de twee ingangen. Dit is eenvoudig te controleren in het netwerk. Indien bijvoorbeeld $x_0 > x_1$ geldt voor de uitgang dat $z = 1/2*(x_0-x_1) + 1/2*x_0 + 1/2*x_1 = x_1$. Het geheel wordt een *comparator* genoemd.

Met behulp van de extra toegevoegde cel kunnen nu meerdere van dit soort "micro" MN'en worden samengevoegd tot een die de boven beschreven taak kan vervullen. Figuur 5.6 toont een MN netwerk voor 8 ingangen. De

drempels in de uitgangscellen zijn 2.5 (alle waarden tussen 2.0 en 3.0 voldoen) en in alle andere cellen 0. De gewichten zijn gelijk aan die in figuur 5.5 voor de comparatoren en 1 voor de overige gewichten.

Opvallend is dat aldus een sterk gelaagd netwerk ontstaat. Een en ander kan uitgebreid worden tot een MN dat een willekeurig aantal ingangen heeft. Zo een MN met N ingangen bestaat dan uit grofweg $^{2}\log(N)$ lagen cellen.

Het hier getoonde netwerk kan nauwelijks een "classificator" genoemd worden maar vormt wel een eerste aanzet naar een meer algemenere vorm van classificatoren. Het netwerk kan overigens altijd gebruikt worden in combinatie met de hierna te behandelen classificatoren. Indien meerdere klassen "hoog" zijn kan dit netwerk die klasse kiezen die het "hoogst" is. Figuur 5.7 verduidelijkt deze situatie, waarbij het MN optreedt als "eindtrap" van een classifier.
Zie ook [21] voor toepassing van dit netwerk in deze context en verdere literatuurverwijzingen.

5.4.2 Het perceptron

Het *perceptron* is een van de eerst onderzochte neurale netwerken en heeft aanleiding gegeven tot ups en downs in de historie van het neurale onderzoek (zie hoofdstuk 2). De reden hiervoor is dat het perceptron enerzijds verrassend complexe "berekeningen" kan uitvoeren, maar anderzijds niet alle berekeningen. Deze kritiek op het perceptron (het kan niet alles) mag achteraf niet relevant lijken, daar het ook niet verwacht mag worden dat het perceptron "alles" kan, het gaf toch aanleiding tot heftige discussies (zie hoofdstuk 2).

Het perceptron bestaat in zijn eenvoudigste vorm uit slechts een enkele cel die tegelijk input- en outputcel is. De cel is van het type "hard limiter" en kan uit- of aanstaan. De ingang van de cel is verbonden met het te bestuderen "onderwerp", de continue [illegible]allen (of discrete) op de

ingangsverbindingen zijn dus een maat van de eigenschappen van het object onder studie. Het "aan"staan van de cel moet nu geïnterpreteerd worden als "het object is van klasse A", het uitstaan als "het object is niet van klasse A". Figuur 5.8 verduidelijkt deze situatie (geschetst voor 2 ingangen).

Een analyse van de gedane classificatie is eenvoudig. De "besluitgrens", dit is de grens waarop de cel "twijfelt" tussen "aan" en "uit", kan mathematisch eenvoudig berekend worden. Voor de twee dimensionale ingang als in figuur 5.8 geldt:

$$x_1 = -W_0/W_1 . x_0 + T/W_1 \qquad [5.1]$$

ofwel een rechte. Voor het driedimensionale geval is de besluitgrens een vlak, voor het vierdimensionale geval een hypervlak enz.

Hiermee is de werking van het perceptron (met dit soort cellen) geheel verklaard. Het perceptron is in staat te classificeren ofwel te clusteren en gebruikt steeds een N-1 dimensionaal vlak in de N dimensionale inputruimte om twee klassen te scheiden. Indien meerdere cellen gebruikt worden kunnen meerdere klassen gevormd worden (één klasse per cel). Een observeerder van het perceptron kan dan eventueel samengestelde klassen specificeren (bv. cel klasse A en cel klasse B aan is een nieuwe klasse "A&B").
Indien de cellen meer uitgangswaarden dan 0 en 1 kunnen hebben (bijvoorbeeld continu) kan een maximum netwerk dienst doen om die klasse te kiezen met de hoogste uitgang.

Het opzien dat het perceptron bij zijn introductie baarde was echter niet zozeer het gevolg van de classificatiecapaciteiten van het perceptron, maar vooral van de mogelijkheid het perceptron te laten *leren* classificeren door het geven van voorbeelden. Dit betekent dat het niet nodig is de gewichten en drempels expliciet te berekenen voor een

klasse X, maar dat aan de hand van voldoende voorbeelden van objecten uit X, getoond aan het perceptron, het perceptron deze gewichten en de drempel zelf kan kiezen.

Het volgende *leeralgoritme voor een cel met een N dimensionale input*, de zogenaamde *perceptron convergentieprocedure*, realiseert het kiezen van de juiste gewichten en drempels:

Stap 1: [5.2]

Initialiseer: zet alle gewichten en de drempels op kleine randomwaarden ongelijk 0. De waarden van de drempels en de gewichten na t stappen worden T(t) en $W_i(t)$ genoemd. T(t) blijft constant gedurende de iteratie.

Stap 2:

Presenteer een input $x_0(t), x_1(t) .. x_{N-1}(t)$ en de gewenste output d(t).

Stap 3:

Bereken de werkelijke output van het perceptron

$$y(t) = f\left[\sum_i W_i(t)x_i(t) - T(t)\right]$$

met f de hard-limiter functie.

Stap 4:

Pas de gewichten aan volgens

$$W_i(t+1) = W_i(t) + n[d(t)-y(t)]x_i(t) \qquad 0 \le i < N$$
$$T(t+1) = T(t)$$

n is een voldoende klein (bijvoorbeeld n<1) positief getal.

Stap 5:

Herhaal stap 2.

Merk op dat indien het perceptron correct is tijdens het leren, in stap 4 de gewichten niet worden aangepast (de term d(t)-y(t) is dan gelijk aan nul). Het perceptron leert dus alleen indien het fouten maakt!

Rosenblatt [84] heeft bewezen dat bovenstaande convergentieprocedure tot een correct resultaat leidt, mits de gegeven punten door een hypervlak gescheiden kunnen worden. Figuur 5.9 geeft een voorbeeld van een perceptron dat convergeert (n=0.01). De figuur toont dat het perceptron reeds na enkele stappen (4) "uitgeleerd" is.

Classificeren met behulp van een hypervlak wordt uiteraard ook toegepast in meer klassieke classificatoren. Een perceptron kan met kleine aanpassingen zo worden ontworpen dat de gewichten en de drempel zo overeenkomen met de gewichten van een klassieke Maximum Likelihood Gauss classificator. Het is dus zeker niet zo dat de prestaties van het perceptron "uniek" zijn. Het is slechts de (parallelle) implementatie van een "Gauss-achtige" classificator. Voor meer over deze analogie zie [21].

In het geval dat de te leren punten niet te scheiden zijn door een hypervlak blijft het perceptron oscilleren, het blijft immers fouten maken. Een kleine aanpassing kan dit "euvel" voorkomen. Indien namelijk de cellen vervangen worden door cellen met een continue uitgangsfunctie worden de gewichten aangepast afhankelijk van de grootte van de afwijking tussen gewenste en werkelijke uitgangswaarde. Op deze manier wordt een soort van kleinste-kwadratenschatter gerealiseerd. Zie ook [21].

Zoals gezegd kent het perceptron zijn beperkingen. Het kan alleen correct classificeren indien de samples door een hypervlak te scheiden zijn. De meest fundamentele kritiek op het perceptron is weergegeven in figuur 5.10. Figuur 5.10 toont het zogenaamde "exclusieve OR" (XOR) probleem. De taak van het perceptron is slechts "hoog" te staan indien exact één van beide ingangen hoog is. Het is in de figuur eenvoudig te

zien dat een geschikt hypervlak niet te kiezen valt. Het gevraagde is daarmee fundamenteel niet realiseerbaar.

Deze kritiek op het perceptron was destijds daarom zo vernietigend omdat er geen andere netwerken bekend waren die wel een XOR konden realiseren, dat wil zeggen niet compleet met leerregel (convergentieprocedure). Deze kritiek kan tegenwoordig weerlegd worden. In de volgende paragraaf zal worden getoond hoe meerlaagse perceptrons het XOR probleem weten op te lossen.

5.4.3 Multi-layer perceptrons

Multi layer perceptrons (MLPs) zijn netwerken die bestaan uit meerdere perceptrons die met elkaar op een "feedforward" manier verbonden zijn. Figuur 5.11 en 5.12 tonen respectievelijk een twee- en een drielaags MLP. Deze MLP's kunnen veel "krachtiger" berekeningen realiseren dan het enkellaagse perceptron uit de vorige paragraaf, maar een probleem is de juiste keuze van de gewichten en de drempels. Hierop zal later in dit hoofdstuk worden teruggekomen.

Indien de cellen in een MLP een lineaire uitgangsfunctie hebben heeft een MLP niets meer aan rekenkracht te bieden dan een enkellaags perceptron. Dit is eenvoudig in te zien. Hiertoe wordt de werking van een enkellaags perceptron geschreven als:

$$\underline{y} = A.\underline{x} + \underline{b} \qquad [5.3]$$

met $\underline{y}$ en $\underline{x}$ de uitgangs- en ingangsvectoren, A de gewichtsmatrix en $\underline{b}$ de drempel(threshold)vector. Merk op dat f hier lineair ($f(t)=t$) gekozen is!
Indien nu $\underline{y}$ als ingang dient voor een tweede laag geldt voor de uitgang van de tweede laag

$$\underline{z} = C.\underline{y} + \underline{d} = C.A.\underline{x} + C.\underline{b} + \underline{d} \qquad [5.4]$$

echter, indien gekozen wordt A'=C.A en $\underline{b}'$=C.$\underline{b}$ + $\underline{d}$ wordt [5.4] herschreven tot

$$\underline{z} = A'.\underline{x} + \underline{b}' \qquad [5.5]$$

met andere woorden, het tweelaags perceptron realiseert exact eenzelfde vorm van classificatie als het enkellaagse perceptron.

De "truc" zit hem echter in het toepassen van *niet lineaire uitgangsfuncties*. Bij een tweelaags perceptron geldt in het algemeen dat

$$\underline{y} = f(A.\underline{x} + \underline{b}) \qquad [5.6a]$$
$$\underline{z} = f(C.\underline{y} + \underline{d}) \qquad [5.6b]$$

dus

$$\underline{z} = f(C.f(A.\underline{x} + \underline{b}) + \underline{d}) \qquad [5.7]$$

Deze vorm is niet noodzakelijk gelijk aan die van een enkellaags perceptron. In het geval bijvoorbeeld $f(t)=t^2$ wordt gekozen is [5.6a] in het ééndimensionale geval van de vorm

$$y = A.x^2 + B.x + c \qquad [5.8]$$

maar [5.7] van de vorm

$$z = A'.x^3 + B'.x^2 + C'.x + d \qquad [5.9]$$

Een N-laags perceptron realiseert zo in het algemeen een (multi-dimensionaal) polynoom van de graad N+1. In dit geval heeft een MLP dus

wel degelijk meer te bieden aan berekeningscapaciteit (een polynoom van een hogere orde)!

In het geval voor de cellen weer hard-limiter cellen gekozen worden, zoals in het eerder behandelde perceptron, laat figuur 5.13 zien hoe het *exor probleem* met dit netwerk correct opgelost kan worden. De klasse die mogelijk gevormd kan worden bestaat nu uit de overlap van K hypervlakken. Indien er twee inputs zijn kunnen 2 input units (plus 1 output unit) een "exor" klasse selecteren, 3 units een driehoek, 4 units een vierhoek enz. Elke *convexe vorm* kan op deze manier benaderd worden. Figuur 5.14 toont een voorbeeld van een convexe besluitgrens.

In het driedimensionale geval kunnen op dezelfde manier kubusklassen, bolvormige klassen enzovoort gerealiseerd worden, dus mits wederom convex. Voor het hoger (N-)dimensionale geval generaliseert dit naar N-dimensionale convexe klassen.

Bij een perceptron bestaande uit drie lagen niet-lineaire cellen is wederom meer mogelijk. In dit geval kunnen de convexe klassen van het tweelaags perceptron in de derde laag gecombineerd worden tot *niet-convexe klassen*. Figuur 5.15 geeft een voorbeeld van zo een niet convexe besluitgrens, gerealiseerd met hardlimiter cellen.

Het grote probleem bij het ontwerpen van MLP's is de keuze van de gewichten en het aantal cellen in de lagen voor een gewenste classificatie. Er bestaat een (hieronder te behandelen) algoritme voor deze taak, maar dit algoritme is niet te gebruiken voor elke type niet lineaire cel. Bovendien bestaat het gevaar dat dit algoritme faalt en vastloopt in *locaal maximale classificaties*. Inzicht hierin wordt verschaft in het onderstaande, maar is op dit moment ook zeker nog onderwerp van (wereldwijde) studie.

Enkele vuistregels voor de keuze van het aantal cellen in een MLP kunnen wel gegeven worden. Zo is het aan te bevelen een zo klein mogelijk

aantal cellen te gebruiken voor de beoogde classificatie. Bij te veel cellen is er "te veel" vrijheid in het systeem en zijn (oneindig) veel gewichtsverdelingen correct. Een leeralgoritme zal dan waarschijnlijk door deze ruimte van mogelijke oplossingen blijven lopen, met eventueel negatieve gevolgen. Een indicatie voor het minimale aantal cellen wordt gegeven door de vorm van de klassegrens (als deze tenminste op voorhand bekend is!). Zie [21] voor meer detail en verdere verwijzingen.

5.4.4 Leren in MLP's

Zoals gezegd bestaat er geen algemeen algoritme voor het vinden van een willekeurige classificatie in een MLP. In het geval van de hard limiter cellen in bovenstaande paragraaf is een dergelijk algoritme niet bekend (alhoewel het uiteraard denkbaar is dat het bestaat!). Indien de cellen echter een continue uitgangsfunctie hebben en in het bijzonder indien deze functie monotoon stijgend of dalend is, bestaat er wel een convergentieprocedure. In deze paragraaf zal dit leeralgoritme, bekend onder de naam *Back Propagation Algorithm*, behandeld worden. Details over dit algoritme worden gegeven in onder andere [8][71] en [9].

Het gebruik van continue uitgangsfuncties f maakt op het eerste gezicht het ontwerpen van een classificator meer complex. Immers, de rechtlijnige besluitgrenzen worden nu vervangen door gekromde curven. Maar aangezien deze uitgangsfuncties differentieerbaar zijn, zijn zij vanuit mathematisch oogpunt "eenvoudiger".

Het Back Propagation Algorithm (BPA) is een generalisatie van de al eerder getoonde perceptron convergentie procedure formule [5.2]. De essentie van het BPA is dat steeds voorbeelden van juiste classificaties aan de ingang en uitgang worden aangeboden. De *foutvector* aan de uitgang wordt nu vertaald naar een foutvector voor de ingang van deze bovenste laag. De gewichten worden aangepast en vervolgens dient dezelfde foutvector van de ingang van de bovenste laag als foutvector voor de uitgang van de op een na onderste laag. De fout wordt *teruggepropageerd*

door het netwerk. Dit kan voor een willekeurig aantal lagen herhaald worden. De details hiervan zullen hieronder besproken worden.

Al met al is het BPA niet anders dan een algoritme voor het minimaliseren van een foutterm (het verschil tussen de werkelijke en gewenste uitgangsvector) en het gevaar bestaat dan ook dat het algoritme convergeert naar een *locaal optimale classificator* en niet in staat is de *globaal optimale classificator* te vinden. Om deze situatie te verhelpen kunnen meerdere pogingen worden gewaagd met verschillende aantallen cellen, verschillende *gain* termen (de eerdere waarde n in formule [5.2]) of met andere startwaarden.

Helaas is het BPA in zijn basisvorm vrij traag. Enkele honderden tot duizenden convergentiestappen zijn minimaal nodig voor het verkrijgen van complexe classificaties. Er zijn overigens aanpassingen van het algoritme bekend die in sommige situaties tot verbeterde resultaten leiden. Zie [39] als voorbeeld voor een aanpassing met behulp van adaptieve gain termen.

[8] geeft een interessant voorbeeld van de toepassing van het BPA in een 3-laags perceptron. Het netwerk wordt in [8] getraind met data die ontleend is aan het sprookje "roodkapje". Door het aanbieden van situaties en gewenste reacties van roodkapje zoals "Grote ogen en oren gaan samen met het gillen van roodkapje (de wolf wordt waargenomen)" enz., ontwikkelt het netwerk zelf cellen in de middelste laag die bij nader onderzoek blijken te staan voor concepten in de buitenwereld als "wolf", "oma" en "houthakker".

5.4.5 Mathematische basis van het BPA

Voor een beter begrip en verdere uitontwikkeling van het BPA is inzicht vereist in de mathematische grondslag van dit algoritme. [9] en [39] voorzien hierin. In onderstaande is een en ander sterk verkort samengevat.

Veronderstel dat een MLP is opgebouwd uit cellen met differentieerbare uitgangsfunctie f (mogelijk verschillend per cel en of per cellaag!). Met gegeven inputs en gewichten W_{ij} (synaps van cel i met invoer van cel j) en drempel T_i, kan de uitgangswaarde berekend worden. Met "$targ_i$" de gewenste uitgangswaarde van cel i en "y_i" de werkelijke waarde wordt nu als foutterm gedefinieerd:

$$Err = \sum_i [targ_i - y_i]^2 \qquad [5.10]$$

met i variërend over de uitgangscellen. De opgave is deze waarde E te minimaliseren. Voor de uitgang y_i mag geschreven worden:

$$\begin{aligned} y_i &= f_i(net_i) \\ net_i &= \sum_j (W_{ij}.x_i + T_i) \end{aligned} \qquad [5.11]$$

met net_i de totale input van een cel en f_i de (continu stijgende) uitgangsfunctie van cel i.

Om nu Err te minimaliseren wordt zoals gebruikelijk met een "steepest descents" methode gedefinieerd:

$$\Delta W_{ij} = - n \, dErr/dW_{ij} \qquad [5.12]$$

met een geschikte (positieve) gain term "n".

Berekening van de term $dErr/dW_{ij}$ levert

$$dErr/dW_{ij} = - \delta_i \cdot y_j \qquad [5.13]$$

Met

$$\delta_i = (targ_i - y_i).\ f_i'(net_i) \qquad [5.14]$$

voor de uitgangslaag en

$$\delta_i = f_i'(net_i) \sum_j (W_{ji}.\delta_j) \qquad [5.15]$$

voor diepere lagen. δ_i (delta i) kan dus berekend worden aan de hand van de uitgangsgegevens voor de toplaag, of aan de hand van de delta's (die reeds berekend zijn voor die laag) van alle cellen in de bovengelegen laag die verbonden zijn met de aan te passen cel. δ_i is dus recursief gedefinieerd. De aanpassing voor een gewicht W_{ij} is nu

$$\Delta Wij = -n\ dErr/dW_{ij} = n.\delta_i.y_j \qquad [5.16]$$

waarbij y_j niets anders is dan de bij W_{ij} behorende ingangswaarde van verbonden cel j uit een diepere laag (of de ingangswaarde).

Een soortgelijke afleiding geldt voor de aanpassing van de drempelterm. Voor meer detail zie [39][9].

Een analyse van een meerlaags perceptron met één uitgang en één ingang maakt duidelijk hoe het mogelijk is dat een bepaalde willekeurige classificering gemaakt kan worden. In dit geval wordt in feite een functie Y = F(X) geleerd. Stel nu bijvoorbeeld dat voor f een functie gekozen wordt van de vorm

$$f_i(a) = \arctan(a)$$

Een enkellaags perceptron kan nu als meest complexe vorm realiseren (met een enkele ingangs- en uitgangscel):

$$y = f_i(W.x+b) = \arctan(W.x+b)$$

Dit is een niet bijzonder ruime klasse van functies. In het geval twee lagen worden toegepast geldt (met N cellen in de eerste laag):

$$y = \arctan\left(\sum_i (W_{2i}.\arctan(W_{1i}.x - T_{1i})) - T_2\right) \qquad [5.17]$$

met W_{2i},T_2 en W_{1i},T_{1i} de gewichten en drempels in respectievelijk de bovenste en onderste laag van het MLP.

Met een dergelijk relatief eenvoudig MLP zijn toch ingewikkelde mappings mogelijk. Door het optellen van een willekeurig aantal van deze arctangensfuncties kunnen immers zeer complexe functies gerealiseerd worden (vergelijkbaar met het samenstellen van functies door middel van Fourier golfcomponenten).

Een interessante mogelijkheid geopperd in [9] is het kiezen van cellen met een sinusvormige functie *f*. Op deze manier ontstaat een soort van *Fourier analyse* van een golfvorm. De cellen kiezen aan de hand van het BPA geschikte frequenties, amplitudes (gewichten), fasen en offsets (drempels) om een golfvorm te decoderen. Het bijzondere hierbij is dat de gekozen frequenties adaptief zijn, dit in tegenstelling tot de klassieke Fourier analyse die werkt met vaste frequenties!

Indien de uitgangslaag van een tweelaags perceptron lineair is en de ingangslaag een sinusvormige uitgangsfunctie heeft is de algemene vorm nu eenvoudig

$$y = \sum_i (W_{2i}.\sin(W_{1i}.x - T_{1i})) - T_2 \qquad [5.18]$$

Het aantal cellen met ongelijke frequenties (en amplitudes ongelijk nul) is hier dus een maat voor het aantal gedetecteerde frequenties in het te bestuderen signaal x.
Een en ander laat zich eenvoudig generaliseren voor meerdimensionale inputs.

Een bezwaar tegen deze aanpak is wel dat de sinusfunctie niet monotoon stijgend is. Dit kan aanleiding geven tot fatale oscillaties, zodat de claim gelegd in [9] dat dit zonder meer mogelijk moet zijn, onjuist is. Niettemin is onderzoek in deze richting zeer zeker de moeite waard.

Ter afsluiting van deze paragraaf dient opgemerkt te worden dat het de vraag is of er een neurologische rechtvaardiging voor het BPA te geven is. Het is onduidelijk hoe de delta's in een neurologisch systeem gerepresenteerd kunnen worden (teruggekoppelde signalen of chemicaliën?) en hoe de fouttermen aan de uitgang berekend kunnen worden. Een discussie over het wel of niet moeten toegeven aan deze kritiek wordt in [39] gevoerd.

5.5 Neurale geheugens

In de volgende paragrafen zal worden getoond hoe met neurale netwerken geheugens gemaakt kunnen worden. Met een geheugen wordt hier bedoeld een medium M dat in staat is een geleerd item "x" op te slaan zodat het later weer aan de "uitgang" van het medium te voorschijn kan komen indien aan de "ingang" een sleutel (of adres) wordt aangeboden waarvan tijdens de leerfase bekend was dat het hoort bij x. De fase waarin het medium wordt bekendgemaakt dat er een sleutel bij x hoort wordt de "leerfase", de fase waarin x gereconstrueerd wordt aan de hand van de sleutel, de "herinnerfase" genoemd. Voor een meer exactere definitie van deze begrippen en algemene overwegingen bij het begrip geheugen zie [26].

5.5.1 Het computergeheugen

Als voorbeeld van een "klassiek" geheugen wordt hier allereerst het computergeheugen aangehaald (dat wil zeggen, de hardware-organisatie van een computergeheugen).
Het computergeheugen kan men zich voorstellen als een laatjesgeheugen waarbij elk laatje een stuk data (een item) kan bevatten en een uniek adres heeft. De opslag en reconstructie van adres-/datavectoren geschiedt perfect (zonder fouten) mits het adres van een item uniek is. Het computergeheugen is erg efficiënt in zijn verhouding van aantallen laatjes of cellen en te onthouden items. Elke te onthouden x neemt exact één "geheugencel" (een meerbitswoord) in beslag.

Ondanks bovenstaande goede eigenschappen kent het computergeheugen ook slechte. Bij beschadiging van het medium M zal in ieder geval een deel van de geheugens totaal verloren gaan en een ander deel mogelijk totaal onbeschadigd blijven. Snijden we het medium symbolisch doormidden dan blijven er twee *perfecte*, maar *halve* geheugens over. Deze situatie laat zich vergelijken met het doorscheuren van een foto. Er blijven dan twee "perfecte" foto's over, waar dus een deel van het getoonde op staat met de oude "scherpte".

Het computergeheugen is erg bruikbaar mits de adresvectoren bekend en uniek zijn en men ook alleen middels adressen data wenst te reconstrueren. Wil men echter op een andere manier reconstrueren (bv, wat is het adres van de cel waarin bepaalde data staat) dan is de enige oplossing (zonder verdere gebruikmaking van extra hard-/software in de vorm van bv. hashtabellen) alle cellen te doorlopen en de data te vergelijken met de gezochte data. De zoektijd (reconstructietijd) is dan lineair met het aantal eerder te onthouden x, een vaak onaanvaardbare toename in zoektijd. In feite is een computergeheugen dus ongeschikt voor het associatief (met de geheugeninhoud gecorreleerd) ophalen van gegevens, daar men altijd eerst een adres nodig heeft om te zien wat er

in het geheugen staat. Andere geheugenaccesseermethoden zijn eenvoudigweg niet mogelijk.

Hashtabellen of andere (softwarematig) data-ordenende technieken leveren maar een gedeeltelijke oplossing voor bovenstaand probleem. Zij werken slechts indien men van tevoren weet op welke manier men wenst te reconstrueren (dus welk deel van x onbekend/bekend is), zodat men op voorhand geschikte hashtabellen kan aanmaken. In het algemene geval is dit echter niet het geval Zo kan men efficiënt een woord van 4 letters beginnend met een p opzoeken (alfabetisch en naar lengte ordenen van de gegevens) maar er zijn veel meer voorzieningen nodig indien een woord van vier letters opgezocht moet worden waarvan drie willekeurige letters gegeven zijn. Zijn meerdere afhankelijke vormen van opzoeken nodig, bijvoorbeeld op beginletter of eindletter van een woord, dan moet men de geheugens zelfs dubbel gaan uitvoeren! Immers, men heeft dan te maken met twee (afhankelijke) ordeningen.

Onderstaande paragrafen bestuderen geheugens (bestaande uit neurale cellen) die betere (of zo U wilt "andere") eigenschappen hebben op de gebieden van snelle reconstructie met behulp van "willekeurige" zoekcriteria en gevoeligheid voor beschadiging.

5.5.2 Gedistribueerde codering

Om te begrijpen waar de tekortkomingen van het hiervoor geschetste "computergeheugen" vandaan komen en hoe verbeteringen in deze situatie aan te brengen zijn, zal het verschijnsel *gedistribueerd geheugen* in deze paragraaf aan de hand van een plaatscoderingsprobleem worden geïntroduceerd. Het plaatscoderingsgeheugen heeft als taak een punt met een x- en een y-coördinaat te onthouden.

In een computergeheugen kunnen x- en y-coördinaten eenvoudig bij elkaar worden opgeborgen op een adres (of twee volgende adressen). Indien men echter wil weten of op een bepaalde plaats een punt zit, kan men niet

anders dan alle adressen aflopen en de daarin opgeslagen coördinaten bezien. Uiteraard kan men allerlei softwarevoorzieningen aanmaken die efficiënter zoeken mogelijk maken maar dit kost aanzienlijk meer geheugen en dit is hier ook niet ter sprake. In onderstaande zal worden bekeken hoe de coderingstaak in een gedistribueerd geheugen vervuld kan worden.

Figuur 5.18 geeft een rudimentaire vorm van gedistribueerd geheugen. In de figuur wordt gebruik gemaakt van units die "aan" gaan staan (de uitgang wordt hoog) indien in een bijbehorende kolom x cq rij y een te onthouden punt aanwezig is.
De units uit de horizontale en verticale rij zijn met elkaar verbonden op een verder niet ter zake doende manier. Als een unit in de horizontale of verticale rij wordt aangezet zal tevens de bijbehorende verticale respectievelijk horizontale unit aan gaan staan. Op deze manier kan zowel de x-plaats dienen als "sleutel" voor het herinneren van het y-deel als omgekeerd, zonder lange zoektijden.

Het in figuur 5.18 geschetste geheugen heeft de prettige eigenschappen van "perfecte" codering of het perfect herinneren van een y-positie gegeven een x-positie (en vice versa). De "punt" die op de plaats x,y gezet is zorgt voor de link tussen de x- en y-cel (bijvoorbeeld middels symmetrische verbindingen met positieve synapsen). Maar er kleven een aantal nadelen aan deze opzet.

Het is nog steeds zo dat indien een klein stukje van het medium (units) wordt beschadigd de data voor die xy-posities totaal verloren gaat. Als het geheugen doormidden gescheurd wordt blijven ook nu twee perfecte, maar halve geheugens over.
Een ander bezwaar is dat het geheugen slechts geschikt is voor het onthouden en ophalen van één item tegelijk. Immers, als twee x- en twee y-units tegelijkertijd actief zijn is het niet uit te maken welke de te coderen punten zijn (en omgekeerd bij het reconstrueren). Figuur 5.18b laat zien hoe dit soort "ghost" geheugens ontstaan.

Voor parallel gebruik is dit geheugen dus ongeschikt.

Figuur 5.18c is een modificatie van 5.18. Om het verschijnsel van "ghost" items te voorkomen is voor elke combinatie van x en y een aparte cel gereserveerd. Een groot bezwaar bij deze opzet van het geheugen is dat indien weinig punten met grote nauwkeurigheid onthouden moeten worden de hardwarekosten erg hoog zijn. Stel dat het vlak in figuur 5.18 de afmetingen 1 bij 1 heeft. Als nu met een resolutie van 0.1 X en Y gecodeerd moeten worden zijn hiervoor in het medium 100 "cellen" nodig. Met andere woorden, met een tweedimensionaal geheugen is de resolutie evenredig met de wortel uit het aantal cellen, een onaantrekkelijk gegeven indien maar weinig punten met hoge resolutie onthouden moeten worden (vergelijk: in een computergeheugen zijn net zoveel laatjes nodig als te onthouden punten en is de resolutie slechts afhankelijk van de te kiezen grootte van een laatje, niet van het aantal te onthouden punten). Voor het (horizontale of vertikale) oplossend vermogen a van het in [5.18c] getoonde netwerk geldt dus:

$$a \sim \sqrt{N} \qquad [5.19]$$

De reden van dit slechte resultaat wat betreft de verhouding resolutie en aantallen geheugencellen (hardware) kan begrepen worden met behulp van de informatietheorie. Indien een geheugenplaats (een unit in figuur 5.18c) een kans "p" heeft om aan te staan, dan is de hoeveelheid informatie die een plaats bevat (zie ook [34]) gelijk aan:

$$-p \cdot {}^{2}\log(p) - (1-p) \cdot {}^{2}\log(1-p) \qquad [5.20]$$

Indien p=1/2 is deze informatie hoeveelheid gelijk aan 1 [bit], maar indien er maar weinig punten onthouden moeten worden is deze hoeveelheid vele malen kleiner, daar p dan veel kleiner is dan 1. In dit geval mag gerust gesproken worden van "hardwareverspilling".

De manier om dit euvel te verhelpen is een codering toe te passen die de units ongeveer de helft van de tijd actief maakt.

5.5.3 Een efficiënt gedistribueerd geheugen

Het nu volgende gedistribueerde geheugen heeft als kenmerk dat een enkele geheugenunit niet direct staat voor een koppeling tussen x en y, zoals in figuur 5.18c, maar dat een hele verzameling units tezamen de plaats coderen. Op deze manier kan met weinig units toch een hoge resolutie worden gehaald, daar de kans p dat een cel aan gaat staan dichter bij 1/2 komt te liggen (zie [5.20]).

Figuur 5.19 geeft op een zelfde manier als figuur 5.18 een plaatscoördinatengeheugen weer. Een geheugenunit wordt hier echter niet langer actief als een te onthouden item in de desbetreffende kolom of rij aanwezig is, zoals in 5.18, maar als het punt binnen een radius r van de willekeurig in het vlak liggende unit ligt. Voor de eenvoud van de beschrijving wordt hier gekozen voor units met gelijke radius.

Een unit zal nu actief worden indien een "item" binnen het bereik van de unit aanwezig is. Afhankelijk van de grootte van de radius r en het aantal items zal het aantal cellen dat bij een te onthouden punt aanstaat verschillen. Zoals gezegd is de situatie "ideaal" als ongeveer de helft van de units aanstaan.

Het aldus geconstrueerde *gedistribueerde geheugen* heeft een aantal opmerkelijke eigenschappen. Allereerst is er een veel gunstiger verband tussen de relatie van het aantal units N en de resolutie van het geheugen. Uit berekening blijkt [zie 34] dat bij benadering geldt dat het oplossend vermogen (a) evenredig is met r en N volgens

$$a \sim r.N \qquad [5.22]$$

In het algemene geval, waar geen tweedimensionale maar een k-dimensionale ruimte gebruikt wordt voor de codering geldt

$$a \sim r^{k-1}.N \qquad [5.23]$$

Dit staat in scherp contrast met de resolutie bij het voorgaande geheugen waarvoor gold

$$a \sim N^{1/k} \qquad [5.24]$$

De resolutie is bij dit geheugen evenredig met het aantal units N, in plaats van evenredigheid met de k-de wortel uit N. Voor hoger dimensionale (maar ook tweedimensionale) geheugens is de gedistribueerde representatie dus zeer aantrekkelijk.

Opmerkelijk is dat de resolutie van het geheugen toeneemt als r toeneemt. Intuïtief wordt wellicht verondersteld dat juist met kleinere r verbetering wordt bereikt, maar dit is niet juist. Met een grotere r worden gemiddeld meer units actief en dit geeft een verbetering van de resolutie, niet een verslechtering. Uiteraard is dit slechts zo totdat een unit gemiddeld vaker dan in 50% van de gevallen actief is. Hierboven vermindert de resolutie weer. Het is dus zaak r ook niet te groot te kiezen.

Maar dit geheugen heeft ook nog andere aantrekkelijke eigenschappen. Stel bijvoorbeeld dat men het geheugen fysisch deelt zoals weergegeven in de figuur 5.21. Er resten dan twee geheugens waar de oorspronkelijke xy-punten nog in beide geheel aanwezig zijn, zij het met een verminderde resolutie! Deze situatie is geheel anders dan die bij het eerder behandelde geheugens en te vergelijken met een "hologram" dat men in twee stukken breekt. Ook dan ontstaan twee complete afbeeldingen van het origineel, maar wel met een verminderde scherpte.

Voorts is duidelijk dat het geheugen goed bestand is tegen beschadigingen en onnauwkeurigheden. Als bijvoorbeeld een flink aantal units vernietigd zijn, daalt wel geleidelijk de resolutie van het geheugen, maar items zijn niet "in een maal" verdwenen. Ook het door een defect onterecht aanstaan van units hoeft niet tot problemen te leiden. Als er voldoende cellen zijn zullen andere "overlappende" cellen deze fout met eenvoudige mechanismen kunnen corrigeren (bijvoorbeeld, laat de meerderheid van overlappende cellen in een gebiedje "gelijk" krijgen).

Het aldus ontstane geheugen is dus ongevoelig voor beschadigingen, kan met een relatief klein aantal cellen een hoge resolutie halen (zeker als hoger dimensionale representaties gebruikt worden), is ongevoelig voor interne ruis en "splitsing" zal leiden tot twee volledige, maar in resolutie gedegradeerde, geheugens. Tot zo ver het goede nieuws.

Uiteraard is er ook slecht nieuws. In de eerste plaats zijn de bovenstaande argumenten sterk afhankelijk van het "schaars" zijn van geheugenitems in de ruimte. Zitten de punten te dicht op elkaar (typisch op een kleinere afstand van elkaar dan de radius van een cel) dan ontstaan wederom "ghost items", items die spontaan ontstaan door een vorm van *interferentie* in het geheugen. Figuur 5.22 laat zien hoe dit mogelijk is.

Indien het geheugen niet *sparse* is en de items dicht bij elkaar komen te liggen, faalt de opzet dus. In het beste geval vloeien deze plaatsen dan samen tot een grote "vlek" (Overigens kan men hier ook positief tegenaan kijken en zeggen dat er sprake is van een soort van automatische "clustering", maar deze eigenschap stond niet op het verlanglijstje bij het ontwerpen van het geheugen, dus dit zou hier niet erg objectief zijn).

Belangrijk (zie ook [26]) om te onthouden bij de verdere beoordeling van de te behandelen neurale geheugens, die zijn opgebouwd volgens de gedistribueerde filosofie, is dan ook dat qualificatie van het geheugen

niet kan geschieden op grond van enkel technische prestaties, maar dat gekeken moet worden naar het toepassingsgebied: een "vliegtuig" is niet beter of slechter dan een "auto", alhoewel beide middelen van vervoer zijn.

Tenslotte het volgende. Het is niet vreemd dat er in een computer, waar exactheid van de data-opslag cruciaal is en waar de technologie er voor zorgt dat er "nooit" fouten in het geheugen optreden (zo wel dan zijn de gevolgen buitengewoon fataal) een "laatjes"geheugen wordt aangetroffen. Maar het zou evenmin verwonderlijk moeten zijn dat in een biologisch geheugen, waar beschadigingen (mogelijk) zeer regelmatig optreden en waar (mogelijk) veel ruis is, een associatief geheugen wordt aangetroffen.
Deze argumenten staan dan nog los van de "psychische" voordelen van het "associatief" herinneren. Gezien het kader van dit verslag zal hierop niet verder worden ingegaan.

5.5.4 Associatieve Neurale Geheugens

Gedistribueerde geheugens volgens de "neurale" filosofie kunnen in vele soorten en maten worden geproduceerd. In dit verslag zullen slechts de *lineaire associator*, opgebouwd uit cellen met een lineaire uitgangsfunctie, en het *Hopfield geheugen*, opgebouwd uit cellen met een hard limiter uitgangsfunctie, behandeld worden. Andere typen van associatief geheugen volgens het neurale concept zoals het *Hamming net* en het *Carpenter/Grossberg* net (zie [21]) worden hier niet beschreven. Overigens kunnen ook de classifiers uit de eerste paragrafen worden gezien als een soort van gedistribueerde geheugens. Ook hier is sprake van een relatie tussen ingangs- en uitgangsvectoren. De ingangsvectoren kunnen gezien worden als de "adressen", de uitgangsvectoren als de te onthouden "data".

Alvorens over te gaan tot de behandeling van de associatoren of geheugens zullen allereerst enkele afspraken gemaakt worden over de te gebruiken notaties.

In het algemeen kunnen wij ons een geheugen voorstellen als een "medium" M dat zich een aantal *data-items* x *herinnert*. De data x zijn het medium ooit voorgehouden en moeten middels het medium en een *sleutel* of *adres* weer op te roepen zijn. De toestand van M na het "leren" van m data x zal worden genoteerd als M(m). Het is overigens niet zo dat M(m) zich per definitie alle m-items volledig zal herinneren. Is dit wel het geval dan zal het geheugen *perfect* genoemd worden. Alle fysische geheugens zullen bij een bepaalde grootte van m verzadigingsverschijnselen vertonen en hun perfectie verliezen. Ten gunste van de wiskundige beschrijving zal er van worden uitgegaan dat de te onthouden data n-dimensionale vectoren $\underline{x}$ zijn.

Om de concepten "herinneren" en "leren" concreter te maken zijn zij in respectievelijk [5.25] en [5.26] omschreven als de opslag of leerformule [5.25] respectievelijk de uitlees- of reconstructieformule [5.26].

$$M(m) = f\ (\ M(m-1), \underline{x}, \text{adres}\ ) \qquad [5.25]$$

$$\underline{x}' = g\ (\ M(m), \text{adres}\ ) \qquad [5.26]$$

[5.25] stelt dat het medium in toestand m-1 via een leerfunctie f overgaat in een nieuwe toestand m, waarbij een te leren data vector $\underline{x}$ en een daarbij behorend adres (dat later dienst doet bij de herinnerfase) het geheugen moduleren.
[5.26] stelt dat het medium dat middels [5.26] is bewerkt, in staat moet zijn $\underline{x}'$ te reconstrueren als het adres aan M wordt aangeboden.
Voor een perfect geheugen zal gelden dat $\underline{x}' = \underline{x}$. Voor een minder perfect geheugen zal dit niet voor alle $\underline{x}$ het geval zijn en/of zal de afwijking

van $\underline{x}'$ van $\underline{x}$ (te meten als een afstand in de n dimensionale ruimte) slechts klein zijn.
$\underline{x}'$ noemen we een stabiel punt van M.

Een belangrijke rol in [1] en [2] speelt het adres. Het verhoogt de helderheid en de eenvoud van de wiskundige beschrijving aanzienlijk indien dit adres samen wordt genomen met de data. Het geheel van data en adres vector wordt een "item" genoemd. Indien het adresdeel een *willekeurig* stuk van het item is, wordt gesproken van een *associatief* geheugen, immers, het item is dan op een willekeurige manier op te roepen indien een deel van het item aan het medium wordt aangeboden. In de rest van dit artikel zal deze aanname steeds gemaakt worden. Het onderscheid tussen te onthouden data en het adres vervalt daarbij: het adres en data zijn samengenomen tot één item.

Bij een computergeheugen bestaat een te onthouden item $\underline{x}$ uit een in de computer strikt gescheiden data en een adresdeel en moet bij het aanbieden van slechts het adresdeel van $\underline{x}$ de gehele vector $\underline{x}$ weer te reconstrueren zijn. Zoals in de vorige paragraaf werd getoond is dit echter ook de enige manier om data op te roepen, wat resulteert in lange zoektijden indien gezocht wordt naar bepaalde data (dus op "inhoud"). [5.25] en [5.26] kunnen herschreven worden middels de "item" definitie tot

$$M(m+1) = f\,(M(m), \underline{x}) \qquad [5.27]$$

$$\underline{x}' = g\,(M(m), \underline{x}'') \qquad [5.28]$$

met $\underline{x}''$ een gedeeltelijk gestoorde versie van $\underline{x}$. Indien $\underline{x}'=\underline{x}$ is het geheugen "perfect".

In plaats van "herinneren" kan hier beter worden gesproken van "reconstructie" van $\underline{x}$ middels aanbieden van een onvolledige versie van $\underline{x}$.

Zonder enig rekenwerk is al duidelijk dat indien de te leren vectoren $\underline{x}$ veel op elkaar lijken, dubbelzinnigheid zal ontstaan en foutherstellen onmogelijk wordt. Immers, de vectoren $\underline{x}''$ kunnen dan op meerdere items "passen" (op ongeveer gelijke afstand van meerdere $\underline{x}$ zitten). Welke $\underline{x}$ gereconstrueerd moet worden is dan niet duidelijk. Als de $\underline{x}''$ veel op elkaar lijken zal de aard van g() en de storing uitmaken welke $\underline{x}$ gereconstrueerd wordt. Dit zal bijvoorbeeld die $\underline{x}$ zwiijn die het dichtstbij in de vector ruimte is (kleinste kwadraat schatting) als deze tenminste bestaat. Belangrijk is echter op te merken dat indien de $\underline{x}''$ veel van $\underline{x}$ afwijken, onmogelijk langer aan [5.28] kan worden voldaan. Het betreft hier een fundamentele tekortkoming (of beter, misbruik) van alle geheugens **M** gespecificeerd volgens [5.27] en [5.28], en niet de tekortkoming van een speciale realisatie!

5.5.5 De lineaire associator

Een eerste poging om een **M** te construeren met goede eigenschappen voor reconstructie van een (random) gestoorde $\underline{x}$ leidt tot de zogenaamde lineaire associator (LA). Bij deze LA neemt [5.28] de vorm aan van

$$\underline{x}' = A(m).\underline{x}'' \qquad [5.30]$$

met A een matrix met reële getallen.

Uit [5.30] volgt dat voor alle te onthouden $\underline{x}$ moet gelden dat $\underline{x} = A.\underline{x}$. Met andere woorden, de matrix A wordt zodanig gekozen dat de items $\underline{x}$ eigenvectoren van A zijn met eigenwaarden exact 1. We zullen deze LA kort bezien, zonder ons overigens af te vragen of A uit mathematisch oogpunt wel bestaat en zo ja, hoeveel moeite het kost A te construeren.

Stel nu dat we het geheugen een gestoorde vector $\Delta\underline{x} = \underline{x} + \underline{x}_{error}$ aanbieden, met $\underline{x}_{error}$ een random foutvector, dan volgt uit [5.30] dat

$$\underline{x} = A(m).\Delta\underline{x} = \underline{x} + A.\underline{x}_{error} \qquad [5.31]$$

Er vindt dus (gestoorde) reconstructie van $\underline{x}$ plaats. Gehoopt wordt dat $|A.\underline{x}_{error}|<|\underline{x}_{error}|$.
In dit geval is $\underline{x}$ namelijk een betere benadering van $\underline{x}$ dan $\Delta\underline{x}$ en is M nuttig als geheugen.

Deze hoop is echter lang niet altijd gerechtvaardigd. Immers, alleen als $\underline{x}_{error}$ in de zgn null-space van A ligt of $\underline{x}_{error}$ voornamelijk een eigenvectorcomponent bevat met een eigenwaarde kleiner dan 1 is de hoop gegrond. De uitgangsvector kan dan teruggekoppeld worden naar de ingang en herhaald [5.30] toepassen zal de reconstructie alsmaar verbeteren, totdat $\underline{x}_{error}$ samenvalt met een item $\underline{x}$ (een eigenvector van A) of een lineaire combinatie van items $\underline{x}$. Hierna zal de reconstructie geen verbetering opleveren, omdat ook alle lineaire combinaties van eigenvectoren van A stabiele $\underline{x}$ zijn! Fataal is de reconstructie indien A eigenvectoren met eigenwaarden groter dan 1 bevat. Dan is de reconstructie $\underline{x}$ van $\Delta\underline{x}$ mogelijk veel slechter.

Deze eigenschappen beperken A en $\underline{x}$ zozeer dat er van een praktische bruikbaarheid geen sprake meer is. Indien twee vectoren $\underline{x}_1$ en $\underline{x}_2$ "geleerd" zijn, zijn daarmee automatisch alle vectoren die lineaire combinaties van $\underline{x}_1$ en $\underline{x}_2$ zijn eveneens stabiel. Er zijn als het ware oneindig veel "ghost memories" ontstaan, dit zijn stabiele punten van M die helemaal niet gewenst zijn.
Dit ontstaan van "ghost memories" en de eis dat de ruis *orthogonaal* op de geleerde vectoren staan voor een goede "ontstoring", maakt dat de LA niet praktisch bruikbaar is.

Overigens is het niet duidelijk hoe A berekend moet worden. Lang niet altijd zal aan de eisen voor de eigenvectoren met eigenwaarden exact 1 voldaan kunnen worden.

Middels lineaire neurale cellen kan de matrixvermenigvuldiging die de lineaire associator vereist eenvoudig gerealiseerd worden. Het meerdere malen aanbieden van de uitgang aan de ingang teneinde de "ruis" te doen verdwijnen wordt verkregen door de cellen niet eenmaal parallel te vuren (een enkele matrix-vermenigvuldiging) maar dit een herhaald aantal malen achter elkaar te doen (bedenk dat alle eigenvectoren van A exact 1 moeten zijn!).
In de begintoestand staan de celuitgangen dus volgens $\underline{x}''$. Zie figuur 5.23.
Merk op dat er een essentieel verschil bestaat tussen de layout van dit netwerk en die van de eerder behandelde associatornetwerken. In dit geval zijn er namelijk verbindingen tussen de cellen in de laag. Er is *terugkoppeling* aanwezig.

Gelukkig valt er aan de slechte prestaties van de LA wel degelijk wat te verbeteren en wel indien we de cellen vervangen door een versie waarin *niet-lineariteiten* in de uitgangsfunctie worden toegepast (vergelijk de situaties bij de correlatornetwerken!). De eenvoudigste van deze niet-lineariteiten is wellicht de "hard-limiter"- ofwel "teken"functie, maar zelfs deze eenvoudige niet-lineariteit levert grote voordelen op.
Gebruik makend van de "teken"functie ontstaat de zogenaamde Hopfield associator ofwel het *Hopfield geheugen*.

5.5.6 De Hopfield associator

Het Hopfield geheugen is momenteel een veel in de belangstelling staand netwerk. De bezwaren van de autoassociator worden voor een groot deel verholpen in dit netwerk.

Het enige verschil tussen het Hopfield geheugen en de lineaire associator is de uitgangsfunctie van de gebruikte cellen. Het Hopfield geheugen maakt gebruik van *hard-limiter* cellen, die hier voor de eenvoud van de behandeling de waarde -1 of +1 kunnen opleveren. De te leren vectoren zijn dus van het binaire type (Een transformatie van het domein (-1,+1) naar het meer vertrouwde (0,+1) kan zonder problemen desgewenst worden uitgevoerd).

Het Hopfield geheugen kent een leerregel voor het aanmaken van de matrix A zoals geïntroduceerd in de voorgaande paragraaf.
Voor [5.27] wordt nu gekozen

$$A[m] = \sum_{t=1}^{m} R(\underline{x}^t) \qquad [5.32]$$

met x^t het t-de te onthouden item en R een integer array gedefinieerd volgens

$$R(\underline{x}) = \underline{x}.\underline{x}^T - I \qquad [5.33]$$

R is dus een matrix met elementen A_{ij} met waarde +1 of -1 als i ongelijk j en 0 indien i=j. Merk op dat $A_{ij} = A_{ji}$, dus A is symmetrisch. I is de eenheidsmatrix.
A[m] volgens [5.32] zal eveneens symmetrisch zijn en tevens een nuldiagonaal bevatten. De elementen A_{ij} van A[m] zullen integer waarden aannemen met een absoluut maximum |m|.

Het leren van een nieuw item $\underline{x}$ komt dus overeen met het optellen van een matrix R($\underline{x}$) bij A[m].
A[m+1] wordt in $\underline{x}$ en A[m] uitgedrukt door [5.32] te herschrijven als

$$A[m+1] = \sum_{t=1}^{m+1} \underline{x}^t . \underline{x}^{tT} - I = A[m] + \underline{x}^{(m+1)} . \underline{x}^{(m+1)T} - I \qquad [5.34]$$

A[m+1] ontstaat dus uit de optelling van een geheel van de nieuw te leren vector $\underline{x}^{m+1}$ af te leiden matrix R en de oude toestand van het medium A[m]. Deze eenvoudige "leerfunctie" verhoogt de praktische bruikbaarheid van het geheugen.

Hiermee is ad hoc aangegeven hoe geheugens "geleerd" worden. De volgende vraag is hoe reconstructie plaatsvindt bij gegeven A[m].

Reconstructie volgens [5.28] voor alle elementen x_i van $\underline{x}$ geschiedt volgens [5.35]:

$$x_i = \text{sgn} \left(\sum_j A_{ij} . x_j \right) \qquad [5.35]$$

[5.35] moet herhaald op x_i worden toegepast totdat $\underline{x}$ niet meer verandert. i dient daarbij willekeurig gekozen te worden (de niet willekeurige variant is ook denkbaar maar levert slechtere resultaten die hier niet verder behandeld zullen worden. Zie [1]).

Figuur 5.24 schetst het Hopfield netwerk. Merk op dat alle drempels in de cellen gelijk 0 zijn.

Voor stabiele $\underline{x}$, dus voor items die het Hopfield netwerk aan de uitgang na een aantal slagen als stabiele patronen kan aanbieden, moet voor alle i gelden dat

$$x_i = \text{sgn} \left(\sum_j A_{ij} . x_j \right) \qquad [5.36]$$

Middels een *energiebeschouwing* van het netwerk valt te bewijzen dat dit soort stabiele $\underline{x}$ bestaan en dat het netwerk altijd in een eindig aantal

slagen een stabiele $\underline{x}$ bereikt (mits de cellen na elkaar vuren). Dit nu te leveren bewijs is gebaseerd op twee stellingen namelijk (1) dat deze energie in een stabiel punt *locaal minimaal* is, dat wil zeggen, geen enkele verandering van een enkele cel i (en meer dan één celverandering per tijdseenheid is niet toegestaan!) kan de energie nog verder verlagen en (2) dat indien een cel van toestand verandert, dit altijd een toestand van het netwerk met een *lagere energie* zal opleveren.
Het bewijs hiervan wordt hieronder gegeven.

Als de *energie* E van een netwerktoestand wordt gedefinieerd (threshold termen worden nul verondersteld):

$$E = - \sum_i \sum_j A_{ij} . x_i . x_j \qquad [5.37]$$

Op de analogie tussen de hier geïntroduceerde "energie"term en het verschijnsel energie in de fysica zal teruggekomen worden in paragraaf 5.4.
Stel nu dat een bepaalde cel i wordt verfrist. Voor de nieuwe toestand van deze cel geldt:

$$x_i' = \mathrm{sgn} \sum_k A_{ik} . x_k \qquad [5.38]$$

De energie E van het netwerk verandert hierdoor met de hoeveelheid ΔE gelijk aan:

$$\Delta E = E' - E = - \sum_{j=1}^{n} A_{ij} . \delta x_i . x_j - \sum_{k=1}^{n} A_{ki} . x_i . \delta x_i - A_{ii} . (\delta x_i)^2 \qquad [5.39]$$

met $\delta x_i = x_i' - x_i$.
Aangenomen dat de diagonaalelementen niet negatief zijn, en dat de elementen symmetrisch zijn, kan bovenstaande herschreven worden tot

$$\Delta E \leq -2.\delta x_i . \sum_j A_{ij} . x_j \qquad [5.40]$$

Als de cel nu niet van teken verandert, dus $x_i' = x_i$ dan is er niets om te bewijzen. Immers, deze x is dan blijkbaar stabiel. Als $x_i' < x_i$, dus $x_i' = -1$ en $x_i = +1$ dan volgt uit [5.40] dat

$$\Delta E \leq 2.-2.-2.k \quad \text{met } k = \sum_j A_{ij} . x_j < 0$$

dus $\Delta E < 0$ (dat $k<0$ volgt uit het feit dat de nieuwe celtoestand -1 is). In dit geval is de stelling dus bewezen. Evenzo volgt uit een verandering van x_i van -1 naar +1 dat $\Delta E \leq 0$ dus eveneens negatief (of nul) moet zijn. Daar de A_{ij} niet oneindig groot kunnen zijn volgt uit bovenstaande dat een verandering van een cel x_i altijd een lager of gelijkblijvend energieniveau kan opleveren. Het aantal malen dat de energie gelijk blijft is echter beperkt daar dit alleen overgangen van x_i van -1 naar +1 betreft.

Samenvattend : Voor het netwerk wordt een "energiefunctie" gedefinieerd, die bij elke netwerktoestand een bepaalde mate van "energie" aan het netwerk toekent. Voor deze energiefunctie laat zich bewijzen dat een celverandering middels [5.35] altijd leidt tot een verlaging of gelijkblijven van de energie, maar dat het gelijkblijven maar een eindig aantal maal achter elkaar kan plaatsvinden. Tevens laat zich bewijzen dat de energie voor elke toestand van het netwerk een eindige grootte heeft. Uit bovenstaande volgt dan dat het netwerk na een eindig aantal "updates" altijd in een stabiele toestand zal geraken.

De vraag is echter of deze stabiele $\underline{x}$ wel de geleerde $\underline{x}$ zijn. Dit blijkt in het algemeen niet zo te zijn!
Uit [5.34] en [5.35] volgt immers

$$A[m].x^t_i = \sum_j A_{ij}[m].x^t_j = \sum_{j=1}^{n} \sum_{m=1}^{k} x^m_i.x^m_j.x^t_j =$$

$$(n-1).x^t_i + \sum_{j=/i} \sum_{t=/m} x^m_i.x^m_j.x^t_j \qquad [5.41]$$

$\underline{x}$ is dus bij benadering een eigenvector van A met eigenwaarde n-1 (met n het aantal elementen van het netwerk, ofwel de dimensionaliteit van de vector $\underline{x}$). $\underline{x}$ wordt echter "misvormd" door de $\Sigma\Sigma$ term. Deze term is het onderwerp van veel studie bij het Hopfield netwerk.
Bewezen kan worden dat de andere eigenwaarden van A klein zijn en dat de eigenvectoren van niet geleerde x orthogonaal staan op de geleerde $\underline{x}$ (dit valt af te leiden uit de bijzondere vorm van A). Dit zijn prettige eigenschappen die onderstaande uitspraken rechtvaardigen.

De $\Sigma\Sigma$ term blijkt onder bepaalde voorwaarden het karakter te hebben van een random verstoring. Mits de onrealistische aanname dat de te onthouden $\underline{x}$ random gegenereerde (dus ongecorreleerde) items zijn, volgt dat de $\Sigma\Sigma$ term een asymptotisch normale stochastische variabele is met een gemiddelde van 0 en een variantie σ = (n-1)(m-1). [5.41] bestaat dus uit n-1 maal de oorspronkelijke x_i plus een kansterm met variantie (n-1)(m-1).
Bij een redelijke random verdeling van +1 en -1 over de x_i en een geringe correlatie tussen de veschillende $\underline{x}$ zal de $\Sigma\Sigma$ term klein zijn, mits de term (n-1) / $\sqrt{\sigma}$ groot is. Hierin schuilt het "geheim" van de goede reconstructie eigenschappen van een Hopfield netwerk.
Merk op dat de eerdere orthogonaliteits eis voor de LA omgezet is in de *geringe correlatie* eis voor het Hopfield netwerk.

De tekenfunctie voert een soort van "onderdrukking" uit van de $\Sigma\Sigma$ term. Immers, de tekenfunctie is ongevoelig voor de kleine $\Sigma\Sigma$ termen (en

meestal zal deze term klein zijn mits aan bovenstaande voorwaarden is voldaan) daar |n-l| in het algemeen groter zal zijn dan de $\Sigma\Sigma$ term. De reconstructie leidt er dan dus toe dat $\underline{x}' = \underline{x}$. Het blijft echter mogelijk dat in enige gevallen de term $\Sigma\Sigma$ wel degelijk een bijdrage levert en een afwijkende stabiele $\underline{x}$ wordt bereikt. Dan geldt dus dat $\underline{x}' \neq \underline{x}$.

Ook uit een nauwkeurigere wiskundige behandeling [27] blijkt dat indien $m \ll n$ (met m het te onthouden aantal $\underline{x}$), of meer specifiek indien geldt dat

$$\sqrt{(n-1)} / \sqrt{(m-1)} \gg 1 \qquad [5.42]$$

geldt dat de stabiele $\underline{x}$ meestal (bijna altijd) samenvallen met de geleerde $\underline{x}$ of hier niet te veel van afwijken. Voorts geldt dat er een "attractieregio" is die foute $\underline{x}$ middels een soort van kleinste kwadraten schatting naar de stabiele $\underline{x}$ "trekt". M kan men zich dan als het ware voorstellen als een flexibele mat met "putten" op plaatsen van de stabiele geheugens. $\underline{x}''$ die voldoende dicht bij een put komen zullen in deze put gevangen worden. Zie figuur 5.25.

Met andere woorden, onder voorwaarde [5.42] is het Hopfield geheugen bij bepaalde toepassingen goed bruikbaar. Met ruis gestoorde vectoren worden volledig hersteld.

Een eigenschap van het netwerk is dat het vrij ongevoelig is voor beschadiging. Immers, de kennis is gedistribueerd over alle A_{ij} en verstoring van een enkele A_{ij} zal niet fataal zijn. Men zal alleen meer stochastische onzekerheid krijgen over de reconstructie, het samenvallen van $\underline{x}'$ met $\underline{x}$ en de grootte van de regio van attractie.
In tegenstelling tot het eerder behandelde computergeheugen is het bij dit geheugen dus zo dat beschadiging van M leidt tot verslechtering van het geheugen, maar dit neemt de vorm aan van het minder betrouwbaar

worden van alle onthouden items en niet het totaal vergeten van enkelen. Een symbolisch halveren van M leidt dus tot twee vergelijkbare geheugens, waar alle gegevens nog in staan, alhoewel met een verminderde "scherpte".

Het netwerk verdient een completere behandeling dan het hier gekregen heeft. De verschuivingen van de stabiele x en de verandering van de attractieregio's onder invloed van het toenemen van m (het meer verzadigd raken van M) dient onderzocht te worden. Tevens dient bekeken te worden of generalisatie van de x_i naar [-1,1] in plaats van (-1,1), modificaties van de leerregel en het gebruik van andere niet lineaire bewerkingen dan de tekenfunctie waardevol zijn.
Dit soort onderzoek wordt momenteel op diverse plaatsen in de wereld verricht. Zie [27] als voorbeeld van een recent artikel waarin gepoogd wordt het Hopfield netwerk mathematisch te "kraken".

5.5.7 Praktische beschouwing van het Hopfield netwerk

Gebruik makend van de resultaten uit de vorige paragraaf kunnen een aantal praktische eigenschappen van het Hopfield geheugen worden afgeleid.
Pluspunten van het behandelde Hopfield netwerk zijn:

- het netwerk maakt associatief ophalen van data mogelijk (dus met een willekeurig maar voldoende deel van een item als "adres").
- er is een eenvoudig algoritme voor het leren van x.
- er is een goede reconstructie van een verstoorde x mogelijk mits [5.42], dus mits er voldoende cellen zijn.
- een snelle reconstructie is mogelijk (doordat alle stappen in de richting van het eindantwoord worden gemaakt).
- er zijn geen orthogonaliteits eisen aan de ruis, de ruis moet alleen voldoende klein zijn.
- er is een hoge mate van onafhankelijkheid tussen de processoren daar zij in een willekeurige volgorde na elkaar "gevuurd" worden (geen onderlinge synchronisatie).

- het netwerk is vrij ongevoelig voor beschadigingen.

Nadelen zijn:

- het stochastische karakter van de reconstructie geeft geen garanties over bepaalde items maar alleen over gemiddelden.
- er zijn slechts stochastische garanties over het samenvallen van geleerde en stabiele items en de grootte van de attractieregio te geven.
- er is een explosieve hoeveelheid synapsen (n kwadraat!) nodig bij toenemende n.
- het algoritme vereist dat slechts 1 processor (cel) tegelijk bijgewerkt mag worden. Een versie waarbij alle processoren parallel werken is wenselijk.

Zoals gezegd valt het uiteindelijke oordeel over het al of niet voldoen van het Hopfield netwerk of enig ander geheugen pas te geven als dit wordt afgewogen tegen de toepassing waarin dit geheugen gebruikt moet worden.

Goed leesbare introducties tot het Hopfield geheugen en voorbeelden van de werking ervan worden gegeven in [23]. [27] en [18] bevatten meer wiskundig georiënteerde beschrijvingen. Voor applicaties op diverse gebieden van het Hopfield geheugen zie [23][24][17]. In [23] wordt een aanpassing van het Hopfield geheugen gegeven dat parallel vuren van de cellen mogelijk maakt, het zogenaamde bidirectionele associatieve geheugen (BAM).

5.6 Netwerken voor probleem oplossen

Als laatste catagorie van netwerken worden in deze paragrafen de *Boltzmann* en *Harmonie* netwerken behandeld. Deze netwerken worden gebruikt voor het oplossen van problemen, meer gebruikelijk *problem solving* genoemd.

Wellicht dat gezien de brede toepasbaarheid van dit terrein en het gegeven dat "problem solving" intuïtief veel lijkt op "denken", maken dat deze netwerken het meest veelbelovend lijken voor meer complexe problemen.

In deze paragraaf zal allereerst worden uitgelegd wat onder "problem solving" wordt verstaan. Daarna zal worden uitgelegd hoe neurale netwerken kunnen worden gebruikt als *soft constraint problem solvers*. De Boltzmann en Harmonie netwerken zullen dan als speciale realisaties hiervan behandeld worden.

5.6.1 Problem solving

De begrippen *probleem* en *probleem oplossen* zullen in deze paragraaf informeel geïntroduceerd worden.

Een probleem kan als volgt worden omschreven. Allereerst is er een *wereld* waarin het probleem beschreven kan worden met behulp van *concepten* of *entiteiten*. In de "wereld" van het schaakspel zijn dergelijke concepten bijvoorbeeld "pion", "D-4", "speler 1" enz. Er zal hier niet getracht worden het begrip "concept" meer te formaliseren. Op een bepaald moment bevindt de wereld zich in een *toestand*, wat beschreven kan worden met behulp van *relaties* tussen de concepten zoals "speler 1 heeft een koning op D-4", "de toren op A-2 bedreigt de loper op A-8", "speler 2 staat mat" enz.

De probleemoplosser (problem solver) heeft als taak vanuit een gegeven toestand een gewenste omschreven toestand te bereiken. Afhankelijk van de gewenste begin- en eindtoestand en de "aard" van de wereld kan deze opgave variëren van triviaal tot onmogelijk complex. Er is namelijk nog een derde aspect aan het probleem oplossen dat het probleem oplossen tot zo een ingewikkelde bezigheid kan maken.

Behalve de concepten in de wereld en de toestand van de wereld zijn er namelijk nog de *randvoorwaarden* of *constraints* die heel specifiek voor het probleem en de wereld zijn. Deze randvoorwaarden geven aan welke toestanden zijn toegestaan en hoe een toestand mag of kan overgaan in een andere toestand. Zo mag bij het schaakspel een koning niet langer dan één zet achter elkaar schaak staan, een loper alleen schuin bewegen, geen enkel stuk zomaar van het bord gezet worden, een speler maar één stuk per "toestandsverandering" verplaatsen (met uitzondering van de rocade) enz. Deze constraints moeten in acht worden genomen tijdens het probleem oplossen.

De aard van de constraints kan zeer verschillend zijn. In het geval de constraints *hard* zijn, is het absoluut verboden een toestand tijdens het probleem oplossen toe te staan die niet geheel in overeenstemming is met deze constraints. Een voorbeeld is het schaakspel, dat uitsluitend bestaat uit dit soort harde constraints. Nergens tijdens het spel mag men eventjes "een beetje" van de regels afwijken om het doel te bereiken. Dergelijke harde constraints zijn vaak kenmerkend voor "kunstmatige" problemen, dat wil zeggen, problemen in door de mens "geschapen" probleemwerelden als schaken (of kaartspelen, monopoly, logische problemen enz.).
Er zijn echter ook problemen waar de constraints minder hard van aard zijn, de zogenaamde *zachte constraints (soft constraints)*. Deze zachte constraints mogen zo nu en dan wel geschonden worden, maar gemiddeld en/of per geval niet te veel. Problemen die zich in de "werkelijke wereld" voordoen zijn meestal van deze aard. Bij het plannen van een lesrooster voor een school bijvoorbeeld, zijn er een groot aantal constraints die bepalen welke klassen in welke leslokalen op welke uren kunnen zitten en welke leraren welke vakken mogen geven. Er zijn daar harde constraints als dat in een lokaal maar een enkele groep mag zitten en dat geen enkele groep in twee lokalen tegelijk kan zitten, maar er zijn ook talloze zachte constraints als: "die leraar moet bij voorkeur dat vak geven", "op aaneensluitende uren moeten groepen bij voorkeur in hetzelfde lokaal of nabij gelegen lokalen les krijgen", "er moeten

weinig gaten in het rooster van klas of leraar zijn", "vrijdagmiddag moet de school voor de meesten om half drie eindigen" enz.

Het voldoen aan harde constraints is een criterium voor het *geldig zijn* van een oplossing, het voldoen aan zoveel mogelijk zachte constraints staat garant voor het vinden van een *goede* oplossing.

Het interessante is dat zelfs problemen waar op het oog alleen maar harde constraints zijn, zoals bijvoorbeeld bij schaken, in de praktijk alleen maar kunnen worden opgelost door extra zachte constraints te introduceren. Alhoewel niet noodzakelijk voor het oplossen van het probleem indien men maar over oneindig veel rekentijd zou beschikken, worden er nu toch extra zachte constraints toegevoegd. Bij schaken bijvoorbeeld gebruikt men constraints als "in een toestand moet een speler bij voorkeur meer materieel hebben dan zijn tegenstander" of "in een toestand moeten stukken bij voorkeur elkaar dekken".
De zachte constraints zijn zeer bruikbaar bij het vinden van korte-termijn-oplossingen (de volgende zet) voor het bereiken van de eindoplossing (winnen) zonder het tot in het oneindige doorrekenen van zet en tegenzetten. De schaker zal tijdens het spel (de opsteller van het rooster zal tijdens het bedenken van het rooster) in ieder geval aan de harde constraints voldoen (geen illegale zetten) maar ook zoveel mogelijk aan de zachte.

De tactiek voor probleem oplossen kan zeer verschillend zijn. Gegeven een begin-toestand en een (vaag omschreven) te bereiken eindtoestand (mat, alle groepen krijgen de juiste lessen in een lokaal) kan men op vele manieren te werk gaan. Zo kan men in een bibliotheek opzoeken of er "zetten" bekend zijn waarvan men weet dat zij tot mogelijk goede resultaten leiden (bijvoorbeeld de opening van een schaakspel). Men kan ook proberen alle mogelijke combinaties door te rekenen en dan die kiezen die tot de goede oplossing leidt (bij sommige eindproblemen bij het schaken bijvoorbeeld als "zet met de koningin en de koning de tegenstander mat" of in spelen als "boter, kaas en eieren"). Tenslotte

kan men een volgende toestand kiezen die in ieder geval aan zeer veel van de "soft constraints" lijkt te voldoen (middenspel: sla een waardevol stuk van de tegenstander, dek een stuk, tracht de centrumvelden te beheersen enz.). Natuurlijk zijn combinaties van deze *tactieken* denkbaar.

Zachte "constraint" problemen vertegenwoordigen een zeer ruime categorie aan problemen. Andere voorbeelden van problemen waar zachte constraints behulpzaam zijn, zijn bijvoorbeeld het bepalen van manoeuvres van legereenheden in een oorlogsgebied, het laten rijden van treinen voor goederenvervoer, het besturen van een bedrijf of een land of het bakken van een taart.
In het oplossen van dit soort problemen met behulp van computers is in het verleden reeds veel energie gestoken. Zeer veel bruikbare programmatuur voor probleem oplossen is reeds ontwikkeld (bijvoorbeeld de schaakcomputer).
Een "vervelende" ondervinding uit de praktijk is echter dat de tactieken voor het oplossen van een probleem heel probleemafhankelijk zijn. Kort samengevat komt het er op neer dat bij een specifiek probleem *specifieke kennis* hoort die gebruikt moet worden om het probleem op te lossen.
Dit betekent dat men als men een probleemoplossende machine wil bouwen, zo ongeveer weer "opnieuw" kan beginnen bij elk prcbleem, tenzij men tracht een "kennistaal" en "probleemoploswetten" te ontwikkelen als krachtige hulpmiddelen. Expertsystemen en meer algemeen de logica kan men zien als pogingen hiertoe.

De hoop dat probleemoplossende machines te bouwen zijn wordt ingegeven door het feit dat mensen goed in staat zijn tactieken die bij een bepaald probleem behulpzaam zijn bij andere te gebruiken. Niet bij elk probleem moet een mens weer geheel opnieuw getraind worden. Hoe een mens hiertoe in staat is, is niet bekend, maar men zou dit vermogen *intelligentie* kunnen noemen (alhoewel er geen exacte definitie van dit begrip bekend is). Bij dit vermogen komen zaken als *redeneren* naar voren. Dit redeneren of combineren van gegevens kan mogelijk ook door

een machine worden beheerst, zodat men ook deze problemen op een "intelligente" manier weet op te lossen.

Met name de tak van onderzoek naar *kunstmatige intelligentie*, (artificial intelligence of AI) heeft als doelstelling "intelligente" processen in kaart te brengen en er algemene "wetmatigheden" in te ontdekken. Vragen als "wat is een probleem", "hoe is een probleem te omschrijven" en "hoe is een probleem op te lossen" zijn kernvragen in de A.I.
Een voorbeeld van een bescheiden succes van de A.I. is de ontwikkeling van de propositielogica en de predicaten logica waarmee een soort van redeneren kan worden weergegeven. Voor een gedegen inleiding in deze zaken en een uitgebreide literatuurlijst op dit gebied zie [15].
Een minder formele inleiding op deze materie wordt gegeven in [54].

De conclusie van deze zeer korte inleiding is dat een machine voor het oplossen van problemen zeer ruim inzetbaar zal zijn. In deze machines moeten minimaal de probleemtoestand en de harde en/of zachte constraints gerepresenteerd kunnen worden.
In het onderstaande zal kort bezien worden hoe een constraint probleem opgelost kan worden met behulp van een klassieke problem solver (met gebruikmaking van de logica), vervolgens zal worden getoond hoe de neurale Boltzmann en Harmony netwerken gebruikt kunnen worden voor het oplossen van constraint problemen.

5.6.2 Problem solvers voor hard constraint problemen

In het geval een probleem alleen wordt gekenmerkt door harde constraints kan een logische problem solver in principe gebruikt worden om het probleem op te lossen. Het logische probleem dat hier als voorbeeld zal dienen wordt als volgt omschreven.

Het Omroepprobleem

Vier omroepen (AVRO, TROS, VARA en VERONICA) trachten hoge kijkcijfers te behalen. Daartoe organiseren zij alle een TV-programma, te weten de programma's "altijd prijs", "reken maar", "welles nietes" en het gewaagde "niets om het lijf".

Elk programma heeft zijn presentator (Fred Oster, André van Duin, Jos Brink of Martine Bijl) en een panellid (Vanessa, Albert Mol, Loes Haasdijk of Gerard Cox).

De vraag aan U is om (zonder vooroordelen) vast te stellen welke omroep welk programma organiseert en welke presentator en panellid in het programma actief zijn, gegeven de volgende harde constraints:

Allereerst geldt het "exclusive or" principe, dwz, een ieder komt exact 1 maal voor in elk programma.
Voorts zijn er gegevens. Deze zijn:

1 Fred Oster presenteert het programma "reken maar" en Fred heeft Albert Mol als panellid.
2 De VARA heeft Martine Bijl aangetrokken.
3 Vanessa zit bij Jos Brink en treedt (uiteraard) op in "niets om het lijf". Vanessa is aangetrokken door VERONICA.
4 Van Duin zit niet in "welles nietes".
5 De AVRO organiseert niet "reken maar" en heeft Gerard Cox niet gecontracteerd.

De oplossing van dit harde constraint probleem is zeker niet triviaal. Zonder papier is deze opgave waarschijnlijk voor bijna niemand te doen. Met papier is met enig "puzzelwerk" wel een oplossing te vinden. Figuur 5.26 kan hierbij behulpzaam zijn. Een kruisje in dit figuur wil zeggen dat twee concepten samengaan, een rondje juist het tegendeel.

Bij dit probleem zijn er verschillende harde constraints die er voor zorgen dat maar één oplossing van het probleem goed is. Was er geen enkele constraint dan zouden alle concepten op een willekeurige manier

kunnen worden gecombineerd. Er zouden dan 2^{96} "oplossingen" zijn. Er zijn echter wel constraints. In de eerste plaats is gegeven dat in elk programma exact één panellid optreedt, van exact één omroep is enz. Deze harde constraints beperken het aantal mogelijke oplossingen met een factor 2^{48}.
Verder is gegeven dat als bijvoorbeeld panellid A in programma B zit, ditzelfde panellid niet ook nog in een ander programma kan zitten. Dit soort harde constraints beperken het mogelijke aantal oplossingen verder tot $(4!)^6$ oplossingen. Dan is er nog de constraint die zegt dat bij dit probleem geldt dat "als A in B en B in C dan ook A in C". Als dit gegeven wordt meegenomen blijven er nog $(4!)^3$ (ofwel 13824) oplossingen mogelijk.
Gegeven echter de "inputs" (verdere constraints) is er maar één enkele oplossing mogelijk, maar welke?

Een manier om bovenstaand probleem op te lossen is alle 2^{96} *combinaties* te proberen en steeds te testen of voldaan is aan alle constraints. Een logische problem solver gaat echter (gelukkig) op een andere manier te werk.
Een logische problem solver heeft inwendig "regels" opgeslagen om te kunnen afleiden wat toestanden noodzakelijkerwijs moeten zijn, gegeven de constraints. Dit vermogen is iets extra's wat niet in de puzzel verstopt zit maar wat puur een eigenschap van de machine is! Desgewenst kan dit vermogen "redeneren", "logisch combineren" of "intelligentie" genoemd worden, maar bepalend is dat dit vermogen er voor zorgt dat de machine sneller dan door "domweg" proberen aan de juiste oplossing kan komen.
Zonder op de details van het logisch redeneren in te gaan is aan te voelen dat dit regels zijn van de vorm "als A in B en B in C dan ook A in C", en "als A in B of C of D of E en gegeven A in B dan niet A in C, niet A in D en niet A in E". Deze "redenatieregels" maken dat de machine het probleem *gericht* kan oplossen.

Er zijn een aantal problemen met het ontwerp van dit soort machines, waarvan de belangrijkste in dit kader wel is dat de machine bij zachte constraints niet te gebruiken is. Het is in de klassieke logica niet mogelijk regels en constraints als bovenstaande aan te passen naar een vorm als "als A in B en B in C dan meestal A in C" of toegepast op bovenstaande: "in bijna elk programma zit exact één panellid". Een klassiek logische problem solver kent geen mechanismen om "bijna elk" of "meestal" te representeren. Tegenspraken tussen constraints en toestanden zijn absoluut niet toegestaan in deze machines. Wordt iets dergelijks toch geforceerd door de gebruiker, bijvoorbeeld door tegenstrijdige ingangsgegevens aan te leveren, dan zal in het beste geval de machine niets genereren of stoppen en de tegenstrijdigheden aangeven en wachten tot zij verholpen zijn maar in het slechtste geval zal de machine onzin gaan produceren of fataal vastlopen.

Uiteraard heeft onderzoek naar zachte constraints wel degelijk bruikbare programmatuur (en aanpassingen van de klassieke logica) opgeleverd. Het is absoluut onjuist te denken dat een "computer" om wat voor reden dan ook fundamenteel niet in staat zou zijn met soft constraints te werken. Denk bijvoorbeeld aan een schaakcomputer die een stuk offert om later een betere positie te bereiken. Zie [15] voor een groot aantal mogelijke uitbreidingen van de logica voor het toestaan van zachte constraints.

Er is in dit verslag geen ruimte om verder in te gaan op de aard van probleem oplossen en constraints. Hopelijk geeft de korte inleiding die hier gegeven is voldoende basis voor de evaluatie van *neurale problem solvers*.

5.6.3 De Boltzmann machine

De achterliggende gedachte bij problem solvers die werken volgens het *Boltzmann principe*, zoals de Boltzmann machine en het Harmonie netwerk, is dat er voor het netwerk een "overall" functie gedefinieerd is die aangeeft hoe goed de door het netwerk gegenereerde oplossing op een

bepaald moment voldoet. Dit geschiedt middels een functie die in sommige artikelen de *energiefunctie* of de *harmonie-functie* wordt genoemd.

Deze functie, die hier verder genoteerd zal worden als H, komt overeen met wat hierboven de "soft constraint functie" is genoemd. H levert bij elke toestand van het netwerk een getal op dat gaat weergeven hoe goed aan de constraints is voldaan.

Gegeven dat een dergelijke functie gevonden kan worden is duidelijk welke de taak van het netwerk moet zijn. Het netwerk moet trachten die "toestand" te bereiken die een maximale waarde voor H oplevert (of een minimale waarde van de energiefunctie, die gedefinieerd is als -H). Hoe dit moet gebeuren is onderwerp van deze paragraaf.

Er is een zeer bruikbare analogie bekend vanuit de fysica waarvoor goede mathematische modellen beschikbaar zijn. Deze analogie betreft verdelingen in *spin-glass* modellen (zie [6]). Dit zijn modellen waar "zachte constraints" bepalen hoe het rooster van een materiaal er moet gaan uitzien. In deze modellen treedt zogeheten *frustratie* (frustration) op, daar niet aan alle zachte constraints kan worden voldaan.
Een bekend voorbeeld van zo een model zijn modellen van een dunne plak magnetisch materiaal. Als alle magneetjes willekeurig of om en om georiënteerd liggen is dit ongunstig omdat er dan zeer veel overgangsgebieden zijn tussen gebiedjes met twee magnetisatiegraden. Als echter alle magneetjes gelijk gericht staan is dit ook ongunstig omdat de gelijke "polen" van de magneetjes elkaar dan maximaal afstoten.
Aan de twee zachte constraints "een minimale lengte van het overgangsgebied" en "een maximale chaos in oriëntatie van de magneetjes" kan maar gedeeltelijk worden voldaan. Het gevolg is dat het materiaal in een toestand belandt waar een compromis gesloten wordt tussen deze twee constraints. Er ontstaan sliertige gebieden met de beide oriëntaties in het materiaal (zie figuur 5.27). Door manipulaties met een extern magneetveld kan een van de twee gebieden de overhand krijgen. Het andere gebied krimpt dan in tot kleine circulaire gebiedjes, de zogenaamde

"bubbles". In een bellengeheugen wordt van deze eigenschap van magnetisch materiaal gebruik gemaakt.

In het algemeen treden dit soort verschijnselen op in fysische modellen waar de deeltjes in twee (of meer) toestanden kunnen staan (op-neer-oriëntatie van magneten, spin-oriëntaties, het wel of niet aangeslagen zijn van een electron, enz.). In dit soort modellen is er steeds interactie tussen de deeltjes en is er steeds een balans tussen complete gelijkgerichtheid en complete chaos. Voor een meer fundamentele beschrijving en verdere verwijzingen zie de artikelen in [85].

Er zijn vele manieren om H te definiëren voor een neuraal netwerk en hieruit zullen verschillende algoritmen voortvloeien voor het bepalen van het maximum van H. De fysica legt echter een aantal strenge beperkingen op aan de keuze voor H, wil er gebruik gemaakt kunnen worden van de analogie.
In de eerste plaats worden de interacties tussen de deeltjes, in dit geval de interacties tussen de neurale cellen, in fysische modellen altijd gekenmerkt door symmetrie. Dit wil zeggen dat de beïnvloeding van deeltje 1 op 2 gelijk is aan die van deeltje 2 op 1. Vertaald naar neurale netwerken komt dit neer op de nogal onrealistische eis dat de beïnvloeding (lees de waarde van de synaps) van de toestand van cel 1 op die van cel 2 gelijk moet zijn aan die van cel 2 op cel 1. Dit is een serieuze beperking, die echter als voordeel heeft dat gebruik kan worden gemaakt van de kennis over spinn-glass systemen bij de bestudering van neurale netwerken.
Een tweede beperking wordt geleverd door het maar in twee toestanden mogen verkeren van de cellen (aan, uit) in plaats van een meer gegradeerde overgang tussen deze toestanden.

Beide beperkende eisen zijn overigens voor de lezer van dit verslag "oude bekenden". In het Hopfield netwerk werden deze beperkingen immers eveneens gelegd. De in deze paragraaf getoonde netwerken komen dan ook sterk overeen met het Hopfield netwerk. Merk op dat bij de bespreking

van het Hopfield netwerk ook reeds gebruik werd gemaakt van het begrip "energie", alhoewel daar opzettelijk weinig aandacht gegeven werd aan de analogie.

Overigens is het niet vreemd dat fysica en neuraal onderzoek op deze manier "verwantschap" vertonen. In beide gevallen betreft het immers onderzoek naar grote aantallen "deeltjes" die met elkaar interacteren. Het probleem is alleen dat de vormen van interactie veel minder streng gebonden zijn in neurale netwerken, daar deze nauwelijks "structuur" vertonen, dit in tegenstelling tot de bestudering van bijvoorbeeld roosters. Ernstiger nog is dat men in neurale netwerken graag wenst te werken naar een eindresultaat waar alle deeltjes samen op de een of andere manier "intelligent gedrag" (als "redeneren, herinneren, classificeren") blijken te vertonen. Zulk een streven is er in de fysica niet, alhoewel men ook hier tracht te komen tot het beschrijven van materiaal-eigenschappen, dus van grote aantallen deeltjes, maar er zijn uiteraard geen pogingen ondernomen met omschrijvingen in de richting van begrippen als "intelligentie". Een intrigerende vraag is of het loslaten van maar een klein aantal van de karakteristieken van "domme materialen" (bijvoorbeeld symmetrie van de beïnvloeding) inderdaad tot "slimme materialen" kan leiden als de hersenen. De hersenen zijn op die manier niets meer dan een wel zeer bijzonder kristal, dat wil zeggen, materiaal met heel bijzondere deeltjes (de cellen) en heel bijzondere interacties.

Het idee dat een maat H weergeeft in hoeverre aan de zachte constraints wordt voldaan zal nu verder worden uitgewerkt.

In figuur 5.28 is de basisconfiguratie weergegeven van interactie tussen twee cellen met symmetrische gewichten (drempels worden zolang 0 verondersteld).
De bedoeling is dat indien het *symmetrische* gewicht tussen de cel positief is de cellen positief *gecorreleerd* zijn, dat wil zeggen, de kans dat de ene cel aan gaat staan, gegeven dat de andere cel aanstaat is groter dan 50%. Een negatieve correlatie geeft aan dat het

tegengestelde het geval is: de kans dat een cel aanstaat, gegeven dat de andere cel aanstaat, is kleiner dan 50%. Merk op dat niets gezegd wordt over de kansen als de andere cel uitstaat.

Dit "correlatie"gegeven is nauw verwant aan de Bayes correlatiecel zoals die werd behandeld in 4.6. De gewichten tussen de cellen gaven daar eveneens correlaties weer tussen het aan- en uitstaan van cellen.

Dit correlatiegedrag is nauw verwant met de eerder gehouden "constraint" discussie. De gewichten implementeren als het ware soft constraints die locaal aangeven welke cellen gecorreleerd zijn met welke andere cellen. Elk gewicht implementeert een constraint. Aan een positieve constraint is voldaan als aan beide kanten van de (symmetrische) link de cellen hoog staan, aan een negatieve constraint is voldaan als aan een kant van de link de cel aanstaat en aan de andere zijde uit.

Figuur 5.29 toont toestanden van een Boltzmann netwerk. In het linker netwerk zijn aan alle constraints voldaan. In het rechter netwerk echter is het onmogelijk aan alle constraints tegelijk te voldoen. Dit netwerk vertoont altijd "frustratie".

Het is eenvoudig een H te vinden die een maat geeft voor het aldus beschreven gedrag. In het algemeen zal H bestaan uit een sommatie over alle verbindingen (één maal per verbinding), dus H zal van de vorm

$$H(S) = \sum_{i<j} f(W_{ij}, x_i, x_j) \qquad [5.43]$$

zijn.
De meest eenvoudige keuze, maar slechts één van de mogelijke die conform is met de beschrijving van H, is

$$H(S) = \sum_{i<j} W_{ij} . x_i . x_j \qquad [5.44]$$

met $x_i, x_j = \{0,1\}$.
Merk op dat deze definitie op een teken na gelijk is aan de "energie"definitie bij het Hopfield netwerk. De hier getoonde H heeft inderdaad de eigenschap dat H toeneemt indien x_i en x_j beide aanstaan en W_{ij} positief is (of afneemt in het geval W_{ij} negatief is). Toestanden met één cel aan en één cel uit of beide cellen uit, leveren geen bijdrage aan de waarde van H.

Voor het weergeven van drempels gaat men als volgt te werk. Er wordt verondersteld dat er in het netwerk één enkele cel is die constant aanstaat. Met deze cel x_k kan H herschreven worden als

$$H(S) = \sum_{i<j} \sum_{i,j \neq k} W_{ij} . x_i . x_j + \sum_i W_{ik} . x_i . 1$$

Dit wordt herschreven als

$$H(S) = \sum_{i<j} \sum_{i,j \neq k} W_{ij} . x_i . x_j - \sum_i T_i . x_{ij} \qquad [5.45]$$

De termen T_i worden nu de drempels (thresholds) genoemd.

5.6.4 Maximaliseren van de H functie

Zonder ons af te vragen hoe de gewichten W_{ij} en de drempels T_i gekozen moeten worden om gewenste constraints te implementeren is duidelijk dat uiteindelijk gestreefd moet worden naar die netwerktoestand waarin H maximaal is.
De *beslissingsprocedure* waarmee bepaald wordt of een cel k wel of niet aangezet moet worden teneinde deze toestand te bereiken is niet triviaal. Een eerste aanzet hiertoe is te berekenen wat de bijdrage van een cel k is aan H. Het verschil tussen cel k aan of cel k uit in H (met

dus alle andere cellen in dezelfde toestand) laat zich eenvoudig berekenen. Er volgt

$$\Delta H(S) = \sum_i W_{ki} \cdot x_i - T_k \qquad [5.46]$$

De ΔH die het aan- of uitzetten van cel k oplevert is dus eenvoudig in cel k te berekenen door de bekende sommatie berekening voor neurale cellen uit te voeren!

De vraag is wat met deze ΔH te doen teneinde H te maximaliseren. De cel zou eenvoudig in die toestand kunnen gaan staan die een positieve ΔH oplevert. Dus

$$x_k = 1 \quad \text{indien } \Delta H > 0,\ x_k = 0 \text{ elders} \qquad [5.47]$$

Dit is de bekende "hard limiter" cel, zoals deze onder andere in het Hopfield geheugen werd aangetroffen.

Het probleem met deze aanpak is dat het zoeken naar een maximale H, waarbij steeds een cel k gekozen en gevuurd wordt volgens het hard-limiter model in het algemeen niet de globaal maximale H maar een *locaal maximum* oplevert. Deze methode voldoet dus niet goed. Er is echter een andere procedure bekend die wel een maximale H op zal leveren. Dit model is (niet verwonderlijk) wederom gebaseerd op onderzoek naar fysische systemen en het proces van het zoeken naar een maximale H wordt in dit gebied *annealing* genoemd. Dit proces wordt gebruikt als methode om in de fysica nette kristalroosters te vormen. Voor een meer gedetailleerde beschrijving zie [6][16].

Als vuurregel voor de cel wordt nu gebruikt (ipv [4.47])

$$p(x_k=1) = 1/\ (1 + \exp(-\Delta H/T)) \qquad [5.48]$$

Deze regel zegt dat de celuitgang 1 wordt met een kans p die afhankelijk is van de grootte van ΔH. Is ΔH zeer positief (tov T) dan zal p~1 zijn, is ΔH zeer negatief dan zal p~0 zijn. Indien ΔH nul is, is er geen voorkeur voor een toestand van cel k. In dit geval geldt dat $p(x_k=1)=1/2$.

Merk op dat de formule voor het bepalen van p geparameteriseerd is middels T. T wordt de *computationele temperatuur* genoemd, wederom analoog aan het begrip "temperatuur" in de fysica. T is in feite een maat voor de *ruis* in het systeem, in die zin dat bij een hogere T de kans $p(x_k=1)$ voor alle ΔH dichter bij een 1/2 komt te liggen. Met andere woorden: bij een hogere temperatuur gedragen de cellen zich meer "chaotisch" en kiezen vaker de toestand die "niet overeenkomt" met het teken van ΔH. In figuur 5.30 is $p(x_k)$ uitgezet voor diverse waarden van T. Merk op dat bij zeer lage temperaturen de curve voor p die van de hard-limiter cel benadert. In dit geval geldt dat de cel aan gaat staan bij positieve ΔH en uit bij negatieve.

Volgens Boltzmann geldt nu voor dit systeem dat voor de kans $P(S_a)$, dit is de kans dat de toestand van het netwerk gelijk is aan S_a, geldt dat

$$P(S_a) = A.\exp(H(S_a)/T) \qquad [5.49]$$

met A onbekend. Aangezien de gezochte toestand S de hoogste H heeft geldt dus dat deze toestand het meest voorkomt.

Hiermee lijkt het probleem "opgelost". Immers, de gezochte toestand komt het meeste voor dus het is een kwestie van "tellen" van het voorkomen van toestanden om te bepalen welke de gezochte S is (de meest veelvoorkomende). Er zijn echter een aantal bezwaren aan deze methode, te weten

1 Er geldt voor twee toestanden S_a en S_b dat

$$P(S_a)/P(S_b) = \exp((H(S_a)-H(S_b))/T) \qquad [5.50]$$

Als T hoog is is het verschil tussen $P(S_a)$ en $P(S_b)$ maar erg klein. Dit betekent dat bij hoge T alle toestanden ongeveer even waarschijnlijk zijn, of dat op zijn minst een groot aantal toestanden S heel vaak voorkomen. Dit maakt het tellen erg langdurig. Er moet zeer lang "gesimuleerd" worden om te bepalen welk de gezochte S is.
Het netwerk loopt als het ware zeer snel door een zeer groot aantal toestanden (en met N cellen zijn er nota bene 2^N toestanden!).

2 Analoog geldt dat als T heel laag is, weliswaar $P(S)/P(S_a)$ willekeurig groot gemaakt kan worden, maar in dit geval gaat zeer veel tijd voorbij voordat het systeem overgaat van toestand op toestand. Een overgang naar de gezochte toestand is erg onwaarschijnlijk. Het systeem is als het ware "bevroren".

Bezwaren 1 en 2 kunnen goed begrijpelijk gemaakt worden door dit proces te vergelijken met het schudden van een knikker in een geribbeld oppervlak. Zie figuur 5.31. De plaats van de knikker geeft weer in welke toestand het netwerk zich bevindt. De kuiltjes in het landschap geven locale maxima in H (hier minima in het landschap) weer. De opgave is de knikker blindelings(!) in het diepste kuiltje te schudden. Bij heftig schudden is de knikker wel vaak in de gezochte kuil, maar ongeveer net zo vaak in andere minder diepe kuiltjes (bezwaar 1). Bij heel rustig schudden kan het heel lang duren voordat de knikker over een heuveltje springt naar een volgend dal (bezwaar 2).

3 Een externe "teller" van toestanden is iets wat niet past in de filosofie van het neurale netwerken.

Het proces van annealing omzeilt bovenstaande problemen. In dit geval begint men wild te schudden (een hoge T), maar hierna schudt men al en al rustiger (afname T), totdat het schudden praktisch nul geworden is (een hele lage temperatuur).
Als dit proces van afname van T maar rustig genoeg gebeurt kan de kans dat de knikker aan het eind van dit proces in het diepste kuiltje ligt willekeurig dicht bij 1 gebracht worden. Dit nu is precies wat gewenst is!

Samenvattend bestaat een Boltzmann netwerk dus uit cellen die onderling verbonden zijn met symmetrische gewichten. Deze gewichten symboliseren constraints waaraan zoveel mogelijk voldaan moet worden. De cellen hebben een uitgangsfunctie die geparameteriseerd is met T. Voor het vinden van de toestand van het netwerk die aan de meeste constraints voldoet wordt het proces van annealing gebruikt, wat wil zeggen dat cellen allereerst met een hoge T bijgewerkt worden en dat deze T geleidelijk naar 0 gebracht wordt.
Het proces van annealing doet het netwerk belanden in een toestand van "maximale correlatie" of "maximaal voldoen aan de zachte constraints".

Hiermee is de theorie achter de Boltzmann machine enigszins uitgelegd. Meer over de theorie en met name over "leren" of keuze van gewichten in Boltzmann machines wordt gegeven in onder andere [53][38][6][16][69]. Meer over "annealing" en de algemene toepassing hiervan wordt gegeven in [68][16].
Boltzmann netwerken zijn onder andere gebruikt voor het vinden van de kortste route in het "traveling salesman" probleem (zie [68]), het maken van chip layouts (zie [68]), het "verstaan" van gesproken tekst (zie [43][44][45]) en het oplossen van logische puzzels (zie dit verslag). In principe is het concept van een machine die middels "soft constraints" optimale oplossingen geeft enorm breed toepasbaar. Denk bijvoorbeeld aan de eerder behandelde zachte constraint problemen als het maken van planningen (lesrooster).

Te bepalen blijft hoe de gewichten en drempels gekozen danwel geleerd moeten worden om de constraints werkelijk te implementeren. Deze meer praktische behandeling van een Boltzmann netwerk wordt in de volgende paragraaf gegeven, die handelt over het Harmonie netwerk. In principe is dit Harmonie netwerk niets anders dan een speciale realisatie van een Boltzmann machine.

5.6.5 Het harmonienetwerk

Zoals gezegd is het harmonienetwerk niet anders dan een speciale realisatie van een Boltzmann machine. In het harmonienetwerk wordt een zelfde definitie voor H gebruikt en er wordt eveneens gebruik gemaakt van het proces van "annealing" voor het vinden van het globale optimum van H. Het enige bijzondere aan het harmonienetwerk is de vorm van het netwerk zelf. Figuur 5.32 geeft dit netwerk weer.

Zoals getoond in de figuur bestaat het netwerk uit 3 lagen cellen. De bovenste en onderste lagen zijn verbonden met de middelste laag. In geen enkele laag zijn cellen onderling verbonden!

De drie lagen worden als volgt gekarakteriseerd. De onderste laag is de *ingangs-* of *inputlaag*. De cellen in deze laag kunnen worden opgevat als *sensoren* die de buitenwereld waarnemen. De cellen worden aan- of uitgezet middels een hiervoor geschikt mechanisme. De inputcellen geven de "ingangsgegevens" bij het probleem, zoals de posities van stukken op een schaakbord, de winstcijfers van een bedrijf, pixels van een video camera enz.

De tweede laag is de *feature* laag. Deze laag cellen vertegenwoordigt de "features" of de concepten die bij het probleem van belang zijn. Bijvoorbeeld mogelijke bordposities bij schaken, artiesten die mogelijk kunnen optreden in programma's enz. De tweede laag cellen geeft na afloop van het annealing proces de "oplossing" van het probleem weer.

Als een bepaalde cel in deze tweede laag aanstaat zal dit bijvoorbeeld geïnterpreteerd worden als "de loper in de volgende zet moet op D4 worden gezet", "de letter A is gelezen","Vanessa zit in een programma van Veronica" enz.
Of de cellen in de tweede laag aan of uit zullen gaan staan (of de concepten "actief" zijn) hangt af van de toestand van de inputcellen en de connecties met en de toestanden van de derde laag cellen.

De derde laag cellen wordt de *knowledge-laag* genoemd. Deze cellen geven de verbanden weer tussen de concepten die met behulp van de tweede laag gerealiseerd werden. Men zou ze "soft implicatie cellen" of iets dergelijks kunnen noemen. Als zo een cel "aan"staat geeft dit aan dat deze knowledge-cel actief is en cellen in de tweede laag in een aan- danwel uitpositie aan het "duwen" is. Deze cellen geven als het ware weer of een "constraint" actief is in het aan- of uitzetten van feature cellen.

Met deze interpretatie in het achterhoofd kunnen harmonienetwerken ontwikkeld worden. Uiteraard is dit een ingewikkeld gebeuren. Het is aan de ontwerper om uit te zoeken welke concepten er van belang zijn (feature cellen), welke constraints er bij een probleem gelden (welke knowledge-cellen gekozen moeten worden), hoe sterk de constraints zijn (gewichten tussen de bovenste en middelste laag) en welke de ingangsgegevens zijn. Voor sommige opgaven is dit een redelijk vanzelfsprekend gebeuren (zoals bij het traveling salesman of het logische puzzel probleem) maar bij meer ingewikkelde problemen is deze opgave wellicht hopeloos (zoals bij het schaken).
Het probleem het netwerk te "programmeren" is echter niet wezenlijk anders dan dergelijke taken te implementeren op een conventionele (von Neumann) computer. Ook hier moet veel niet triviale programmeerinspanning worden verricht. Alleen heeft deze laatste manier van programmeren het voordeel dat er al enige tientallen jaren van onderzoek in geïnvesteerd zijn: Er zijn al veel hulpmiddelen ontworpen (programmeertalen!) die het ontwerpen ondersteunen. Dit moet men niet

vergeten bij een vergelijk tussen de vormen van programmeren van neurale netwerken en von Neumann computers!

Door de speciale opbouw van het netwerk zijn er wel richtlijnen te geven voor de keuze van de gewichten. Allereerst kan men ongestraft alle gewichten en drempels naar boven of beneden schalen, mits dit maar voor elke waarde met een gelijke maat gedaan wordt. Dit schalen komt immers op niets anders neer dan het veranderen van T. Een gebruikelijke keuze voor de gewichten zijn waarden met een absolute waarde van gemiddeld rond de 1. Evengoed kan men hier echter een andere waarde voor kiezen. Bij een waarde van 1 ligt het "zinnige" bereik van T tussen ongeveer 0 en 1 "graad". Het annealingsproces speelt zich dus rond dit interval af.

Voor de keuze van de gewichten tussen laag 1 en 2 (de input- en featurelaag) kan in principe plus of min "oneindig" gekozen worden. Kiest men kleinere getallen dan "verzwakt" dit de correlatie tussen ingangscellen en feature-cellen. Dit geeft dan in feite weer dat mogelijk enige inputgegevens incorrect zijn. Het zijn hiermee "soft inputs" geworden.

Voor de gewichten en drempels in laag 2 en 3 wordt hier de aanpak van Smolensky in [37] gevolgd (met een aanpassing voor cellen met bereik [0,1] in plaats van [-1,1]).
De drempels in laag 2 geven een voorkeur of *a priori* kans weer voor het aan of uit gaan staan van de feature-cellen. Smolensky kiest er voor om hiervoor een waarde 0 te gebruiken. Er is dus geen a priori voorkeur voor het aan- of uit-staan van feature-cellen.
Voor de gewichten tussen laag 2 en 3 gaat Smolensky als volgt te werk. Voor elk knowledge-atoom wordt een "sterkte" σ (een getal groter dan 0) gekozen. Hoe groter dit getal, des te meer invloed heeft een knowledge-cel op de in laag twee liggende feature-cellen. σ is dus een maat voor de "hardheid" van de constraint. Voor de gewichten tussen de knowledge-cel k en de feature-cel i wordt nu gekozen:

$$W_{ik} = W_{ki} = t.2.\sigma/N \qquad [5.51]$$

N is het aantal inputs tussen de cel k en de verschillende feature-cellen en t is eenvoudig een tekenbit (+1 of -1).

Voor de keuze van de drempel van de knowledge-cellen wordt vervolgens een parameter μ gedefinieerd die gekozen wordt in het bereik $\mu = [0,1]$. Indien μ=1 gekozen wordt komt dit overeenkomt met een netto input 0 naar de knowledge-cel (dus de cel zal net aan gaan staan bij extreem lage temperatuur) indien alle feature-cellen een toestand hebben overeenkomstig de gewichten. In het geval μ=0 is de netto input juist gelijk aan 0 als er precies één feature-cel "fout" staat. Voor de drempel T_k wordt nu berekend:

$$T_k = 2.\sigma/N \; . \; (\; s(+) \; -1 \; + \; \mu) \qquad [5.52]$$

met s(+) het *aantal* verbindingen met positief gewicht tussen de knowledge-cel en de feature-cellen.
Overigens is het denkbaar meer negatieve waarden voor μ te kiezen, maar Smolensky kiest hier niet voor.
De drempel (of beter, de waarde van μ) implementeert als het ware hoe "perfect" de matching moet zijn tussen de toestand van de feature-cellen en de knowledge-cel voor het actief worden van de knowledge-cel.
Een waarde van μ kleiner dan 1 maakt dat een constraint ook gebruikt wordt indien deze matching niet perfect is.

Als eerste voorbeeld van hoe een probleem op te lossen met deze machine zie figuur 5.34. Het betreft hier het "koninginnenplaatsprobleem". De opgave is een aantal van N koninginnen te plaatsen op een bord van N bij N zonder dat er meer dan één koningin op dezelfde horizontale, verticale of diagonale lijn staat.
Voor bepaalde N (bijvoorbeeld N gelijk 4,6,8) bestaan er oplossingen van dit probleem.

Bij het probleem zijn er geen inputcellen, daar er geen voorkeur vooraf is voor een bepaalde plaats voor een koningin. Eventueel zou dit wel geëist kunnen worden (alhoewel er dan mogelijk geen correcte oplossing is).

De feature-cellen hebben nu de volgende betekenis: een aanstaande cel betekent dat er op een bepaalde positie wel een koningin moet staan, een uitstaande cel betekent dat dit niet het geval is. Voor elke positie op dit bord is er een feature-cel, dus in totaal zijn er N^2 feature-cellen.

De bovenste laag cellen vertegenwoordigt de "constraints". Er zijn diverse manieren mogelijk de constraints door middel van de knowledge-cellen weer te geven. Een mogelijkheid is te kiezen voor twee "soorten" knowledge-cellen, te noemen "diagonaal"cellen en "kruis"cellen, zo genoemd omdat zij "gevoelig" zijn voor respectievelijk diagonaal- en kruispatronen. Indien zo een cel actief is betekent dit dat de positief met deze cel verbonden feature-cel aan is en dat nergens anders op de bij deze positie horende kruis cq diagonaal een koningin staat. Dit is geïmplementeerd door de feature-cel te verbinden volgens figuur 5.35. Er zijn in totaal $2.N^2$ knowledge-cellen.
Bij bepaalde keuzen van σ en μ van de cellen zal een gezochte oplossing een maximale H (harmonie) hebben. Indien maar langzaam genoeg afgekoeld wordt zal deze positie gevonden worden.

Praktische simulaties gedaan in het kader van dit verslag hebben aangetoond dat een correcte antwoorden voor N=4,6,8 met de harmonie machine gevonden kunnen worden.

Een tweede voorbeeld is het oplossen van de eerder genoemde logische puzzel. Figuur 5.36 toont het harmonienetwerk voor het oplossen van deze puzzel. De feature-cellen staan voor alle mogelijke combinaties van de elementen van dit probleem (bijvoorbeeld: Vanessa bij de VARA). Voorts zijn er twee typen knowledge-cellen. Het eerste type zou omschreven kunnen worden als van het "exor" type en heeft als functie er voor te

"zorgen" dat slechts één feature cel in een bepaalde rij of kolom actief is (Vanessa kan maar bij één omroep zijn). Het tweede type is van het type "driehoek" en zorgt voor de constraint dat "als AB en BC dan AC" (Vanessa bij de VARA en de Vara heeft het programma "altijd prijs", dan ook Vanessa in het programma "altijd prijs"). De figuur verduidelijkt hoe een en ander is opgebouwd.
Bij gedane simulaties bleek het mogelijk de gezochte correcte oplossing te vinden met het hier getoonde harmonienetwerk. Daarvoor werd een exponentieel dalende temperatuur gebruikt en elke cel werd in totaal circa duizend maal gevuurd.
Merk op dat in dit geval inputcellen aanwezig zijn voor de gegeven "ingangsgegevens" bij de puzzel. Dit werd in de simulatie echter gerealiseerd door bepaalde "bekende" rijen en kolommen feature-cellen "vast" te zetten op hun op voorhand bekende, correcte waarde (die direct volgen uit de puzzelgegevens).

Er zijn twee voordelen bij het gebruik van de harmoniemachine ten opzichte van een algemeen Boltzmann netwerk. In de eerste plaats is er een "theorie" over wat de cellen dienen te vertegenwoordigen en er zijn formules die een indicatie geven voor de keuzen van gewichten. De gewichtskeuze voor elk gewicht afzonderlijk, is teruggebracht tot het eenvoudigere probleem σ's en μ's te bepalen. In de tweede plaats kan de machine steeds parallel een gehele laag cellen tegelijk vuren. Immers bij de berekening van dH werd uitgegaan van het constant blijven van alle andere cellen bij het updaten van een cel k. Aangezien geen enkele cel in een laag verbonden is met een cel in dezelfde laag mag de gehele laag parallel gevuurd worden.

Het is niet de bedoeling van deze paragraaf om in de details van de harmonie machine te duiken. Voor een wiskundige onderbouwing van de theorie zie [37]. Dit artikel is een absolute "must" indien men is geïnteresseerd in de harmonie-theorie.

Andere bruikbare artikelen zijn [67][76][77][78]. In [37][76][77] wordt getoond hoe een harmoniemachine gebruikt kan worden als netwerk voor het doorrekenen van elektrische modellen (alhoewel zeer primitief).

Samenvattend gezegd ligt de kracht van de harmoniemachine in twee zaken. Allereerst de enorme snelheid waarmee een oplossing gegenereerd kan worden, wat veroorzaakt wordt door de eenvoud van de celberekeningen en het parallel kunnen vuren van een gehele laag cellen. In praktische simulaties bleek het mogelijk oplossingen te genereren in 100 tot 1000 updates van de lagen (waarbij T dus langzaam tot 0 afnam). Een vuurslag in een implementatie van het netwerk hoeft maar enkele microseconden in beslag te nemen indien uitgevoerd in parallelle VLSI technologie.
Als tweede kracht geldt de mogelijkheid van de machine om oplossingen te generen met maximale H. Dit betekent dat correcte oplossingen worden gevonden mits althans een correcte oplossing overeenkomt met een maximale H. Bovendien is het zo dat de constraints zoals geïmplementeerd met de "knowledge"-cellen zacht van karakter zijn. Als er geen "perfecte" oplossing mogelijk is (bijvoorbeeld op een 3 bij 3 bord met 3 koningen) genereert de machine toch oplossingen die niet perfect zijn maar een minimaal aantal constraints overtreden. Dit lijkt een zeer bruikbare eigenschap indien de Harmoniemachine wordt ontworpen om te opereren in een omgeving waar bijvoorbeeld enkele ingangsgegevens onnauwkeurig, strijdig of zelfs fout kunnen zijn, zoals in veel "real world" toepassingen het geval zal zijn!

Het grote probleem bij een harmoniemachine is de keuze van de aantallen feature- en knowledge-cellen en de grootte van de σ's en μ's van de knowledge-cellen. Er is geen theorie die bepaalt hoe deze keuze gemaakt moet worden. Het is echter niet eerlijk dit als kritiek op het netwerk te hebben. Immers, er is eveneens geen methode voor het genereren van programma's gegeven een probleem! Hooguit is er het gebruik van een hogere programmeertaal als hulpmiddel. In dat kader bezien kan men de keuze van σ's en μ's ook zien als een "hogere programmeertaal", in ieder geval een hogere dan die van drempels en gewichten. Het is de vraag of

er nog "hogere" en krachtiger talen ontwikkeld kunnen worden. Indien dit zo is, zal dit de bruikbaarheid van de harmoniemachine uiteraard verhogen.

5.7 Nawoord neurale netwerken

Deze lange paragraaf liet drie "soorten" netwerken de revue passeren, te weten "classificatoren", "associatieve geheugens" en "problem solvers". Zoals gezegd is de indeling in dit soort catagorieën nogal kunstmatig. De grenzen tussen de verschillende typen netwerken zijn vaag, zowel op implementatieniveau als qua werking.

Opvallend is dat geringe verschillen in celwerking en layout geheel andere eigenschappen aan de netwerken lijken te geven. Dit is wellicht in overeenstemming met de constatering dat de mens neurale netwerken voor een veelheid van taken gebruikt. Ook een klassieke computer kan overigens geheel verschillende taken uitvoeren door een iets andere volgorde van dezelfde instructies.

Alhoewel in deze paragraaf de belangrijkste van de in de literatuur voorkomende netwerken zijn behandeld, behelst dit toch maar een klein aantal van de mogelijke netwerken. Men kan bijvoorbeeld generaliseren naar netwerken met meerdere lagen, andere typen cellen, diverse typen cellen in een net of per laag en naar andere verbindingspatronen als de symmetrische. Het is de opdracht van theoretici te bezien of dit soort generalisaties bruikbaar zijn, dat wil zeggen of zij werkelijk krachtige eigenschappen bezitten, en theorieën te ontwikkelen waarmee de werking "bewezen" kan worden, analoog aan de convergentietheorieën en de "energie-" of "harmonie"theorieën rond teruggekoppelde netwerken met symmetrische gewichten (Hopfield, Boltzmann en harmonienetwerken).

6 RADAR TOEPASSINGEN VAN NEURALE NETWERKEN.

6.1 Inleiding

Voorgaande paragrafen behandelden de neurale cellen en neurale netwerken. Zoals gezegd in de inleiding is het uiteindelijke doel van dit verslag de neurale filosofie te benutten voor radar, een doel dat welhaast vruchtbare resultaten moet afleveren gezien de wensen aan radarzijde en datgene wat neurale structuren bieden. In dit hoofdstuk zal worden bezien wat met de huidige kennis van netwerken, zoals gepresenteerd in de voorgaande hoofdstukken, aan voor de hand liggende radar toepassingen te vinden zijn.

Uiteraard is het ondoenlijk hier naar een compleet overzicht van toepassingen te streven. Radar is een verzamelnaam voor een keur van technologieën die loopt van het ontwerp van radarantennes tot het interpreteren van radarbeelden. Tussen die twee punten zit een veelheid van technieken die weer kunnen verschillen naar gelang de radartechnologie (pencil beam, phased array, dopplerradar enz.), de aard van het inzetgebied (clutter onderdrukking, jamming enz.) en de taak van het systeem (bewaking luchtruim, volgen van doelen, sturen van missiles enz.).
Dit hoofdstuk moet dan ook worden gezien als een "eerste aanzet" tot het gebruik van neurale structuren in radar en hoofddoel is ook niet exact aan te geven hoe een netwerk moet worden toegepast maar veeleer aan te geven dat de netwerken op vele plaatsen in radarsystemen bruikbaar zijn.

Een "probleem" bij het schrijven van dit hoofdstuk was toch enigszins de samenhang te behouden bij het beschrijven van de toepassingen. Dit is gedaan door een radarsysteem opgedeeld te denken in drie delen. Deze zijn:

1 Het antennedeel. Dit deel heeft als taak de buitenwereld af te zoeken op voorwerpen en signalen op te wekken als er voorwerpen in de buitenwereld zijn. De signalen moeten later bewerkt kunnen worden, zodat een reconstructie van dat deel van de buitenwereld waarin men geïnteresseerd is, mogelijk is.
2 De "low-level" signaalbewerking. De signalen die van (1) afkomen moeten bewerkt worden tot "informatie". Electrische signalen moeten worden omgezet in informatie van de vorm "op dat moment in de tijd is er op die plaats dat en dat in de buitenwereld waargenomen". Men zou dit kunnen beschrijven als "filtering".
3 De "high-level" signaalbewerking. De "informatie" verkregen met (2) zal nog verder verwerkt worden teneinde de informatie te kunnen "interpreteren". In het verleden gebeurde dit vooral door de mens, de "operator", maar in moderne systemen wordt deze taak zoveel mogelijk overgenomen door de machine. Dit "high-level" niveau maakt gebruik van kennis van de buitenwereld om tot interpretatie en verdere bewerking van de informatie te komen. Zo kan aan een "stip" een label "vliegtuig" gegeven worden, bijvoorbeeld op grond van de aard van de reflectie, de gemeten hoogte van het object en het gedrag van het object in de tijd (snelheid). Bij de high-level verwerking wordt typisch gebruik gemaakt van "kennis" teneinde informatie te kunnen verwerken en in een geschikte vorm aan de mens aan te kunnen bieden.

Figuur 6.1 geeft deze opdeling schetsmatig weer. De radar wordt in dit hoofdstuk dus opgedeeld gedacht uit drie achter elkaar te plaatsen stukken waarvan de taken in bovenstaande vaag omschreven zijn. Uiteraard is er nergens in een radarsysteem een scherpe grens te trekken waar de deelsystemen exact in elkaar overgaan. De indeling is echter bruikbaar om lijn in dit hoofdstuk te brengen.

6.2 De voordelen van neurale netwerken voor radar

Alvorens in detail te treden over de toepassing van neurale structuren in radar kan worden vastgesteld welke in het algemeen de voordelen bij radar zijn van een aanpak middels de neurale filosofie.

De belangrijkste "winst" die neurale structuren bieden is snelheidswinst. Door het massaal parallel "vuren" van cellen kunnen in een beperkt aantal "slagen" van het netwerk problemen worden opgelost. Deze aantallen slagen variëren van één (classifier netwerken) tot hooguit circa 1000 (harmoniemachine). Daar een slag niet meer behelst dan een klein aantal zeer eenvoudige mathematische bewerkingen mag verwacht worden dat een slag niet meer dan enkele microseconden in beslag hoeft te nemen.
Deze snelheidswinst is zeer belangrijk bij het toepassen in radar. Bijna altijd is men bij radar geïnteresseerd in het "real time" volgen van objecten en er is dus geen tijd voor berekeningen achteraf. Toepassing van bijvoorbeeld zoekprocedures kan daarom maar zeer beperkt plaatsvinden. Juist daarom is een sneller alternatief, dat geboden wordt door neurale netwerken, zeer aantrekkelijk.

Naast dit snelheidsvoordeel is er het voordeel van "nieuwe" algoritmen. De neurale filosofie tracht door middel van vele eenvoudige elementjes een krachtig geheel te ontwerpen. Dit biedt een nieuwe kijk op bijvoorbeeld het ontwerp van radarantennes (zie volgende paragraaf), maar ook levert het nieuwe algoritmen op voor het clusteren en "soft constraint problem solving".

Als derde voordeel van neurale structuren geldt de ongevoeligheid voor beschadigingen en ruis in de hardware. Bij de huidige stand van de technologie lijkt deze eigenschap niet bijzonder bruikbaar. De technologie is immers extreem betrouwbaar. Mogelijk dat in toepassingen waar extreme betrouwbaarheid geëist wordt (zoals in de ruimtevaart) of

waar robuustheid erg belangrijk is (wellicht in een militaire omgeving) deze eigenschappen bruikbaar zijn.

In de volgende drie paragrafen zal worden bezien hoe deze voordelen in de praktijk bewaarheid kunnen worden.

6.3 De neurale filosofie toegepast op de radarantenne

In radar kent men vele soorten en maten antennes. Eén ding hebben zij echter alle gemeen, zij zijn ontworpen om objecten in de buitenwereld te kunnen waarnemen. Een "klassieke" antenne zendt straling uit in een bepaalde richting en vangt de reflectie van die straling weer op. Aangezien bekend is in welke stand van de radar de straling werd uitgezonden en hoe lang het duurde voordat het gereflecteerde signaal werd terugontvangen is af te leiden in welke richting en op welke afstand een object aanwezig is.

Een methode om de prestaties van een radarantenne te verbeteren lijkt voor de hand te liggen. Om de plaats en positie van een object beter te bepalen moet worden getracht het oplossend vermogen van de antennebundel te vergroten. Dit kan gebeuren door het vermogen van de bundel op te voeren (minder last van ruis) of de bundel verder te concentreren.
Het opvoeren van het vermogen van de bundel onderdrukt slechts ruis en levert bovendien tactische (en technisch/economische) problemen op. Immers, een radar met een groter vermogen is eenvoudiger door een eventuele tegenstander te ontdekken, met alle fatale gevolgen vandien.
Het nadeel van het versmallen van de bundel is dat dit gepaard gaat met het verkleinen van het gebied dat wordt waargenomen. Bij het ontwerpen van een radarsysteem moet dan ook gekozen worden voor een compromis tussen "bundelbreedte" ofwel waarneemscherpte en waarneemtijd. De situatie laat zich vergelijken met het staren door een verrekijker: Bij een sterke vergroting ziet men een klein stukje land met grote scherpte, bij een zwakke vergroting ziet men een groot stuk land met een geringe scherpte.

Het ligt aan de toepassing van de radar welk type antenne men kiest. Soms kiest men zelfs voor een combinatie van twee antennes. In het geval men gebruik maakt van een "phased array antenne" kan de bundelbreedte op electronische wijze worden gewijzigd. Helaas zijn dit soort antennes erg duur en moeilijk te construeren.

De vraag is of het nu niet mogelijk is op een andere manier een hoog oplossend vermogen te verkrijgen dan door de bundelbreedte te verkleinen. Zou er volgens de gedistribueerde filosofie niet gebruik gemaakt kunnen worden van de combinatie van zeer veel gelijksoortige, eenvoudige antennes?
Deze vraag roept herinneringen op aan de discussie rond geheugennetwerken. Met opzet werd in deze paragraaf reeds gesproken over het opslaan van posities in een xy-vlak. Het is maar een kleine stap om dit principe te benutten voor radar.

Volgens de neurale filosofie kan een "gedistribueerde" radarantenne ontworpen worden. Dit kan men zich als volgt voorstellen (zie figuur 6.2): Over het oppervlak waarin men geïnteresseerd is verspreidt men een groot aantal zeer eenvoudige radarantennes. Deze antennes zijn rondom gevoelig over een gebied met straal R (bolvormig in de praktijk, voor deze discussie wordt gekozen voor een platte cirkel als gevoeligheid). Indien er zich in een gebied waarvoor een antenne-eenheid gevoelig is een object bevindt, zal dit station "aan" gaan staan. Er is dan niet meer bekend dan dat ergens in de cirkel met straal R een object aanwezig is (er is dus geen afstandsinformatie).
De werking van de "gedistribueerde radar" berust echter op een groot aantal van dit soort eenvoudige antennestations. Deze antennestations liggen dus verspreid over het gehele waar te nemen gebied. De stralen van de antennes zullen alle ongeveer gelijk zijn. Alhoewel de resolutie van een enkele cel dus slecht is, wat betekent dat de antenne zeer "low cost" kan zijn, zal het totaal een vele malen betere resolutie hebben. Volgens [5.23] is deze resolutie rechtstreeks evenredig met het aantal antennecellen dat gebruikt wordt.

Er kunnen natuurlijk verfijningen van dit concept bedacht worden. Hier gaat het echter om het basisidee zeer vele antennecellen te gebruiken die stuk voor stuk slecht zijn, maar tezamen een hoogwaardig resultaat kunnen opleveren. De praktische waarde van dit idee is moeilijk in te schatten, daar dan gekeken zou moeten worden naar de kosten en uitvoerbaarheid van de gedistribueerde radar. Oordelen hierover zijn weer afhankelijk van het toepassingsgebied (militair, civiel, ruimtevaart?).

Behalve dat op deze manier een radar geconstrueerd kan worden die het gehele (lucht)ruim tegelijkertijd in de gaten kan houden (er is immers geen "rondzoekende" bundel!) en toch een goed oplossend vermogen kan hebben (veel cellen),terwijl nota bene van zeer eenvoudige antennecellen gebruik gemaakt wordt, is er nog een ander voordeel aan deze opzet. Zoals al bleek uit de verhandeling over gedistribueerd geheugen is deze gedistribueerde radar ongevoelig voor beschadigingen van enkele antennecellen. Zo zou men bij een militaire toepassing in een vijandig gebied een groot aantal van de cellen kunnen neerlaten (aan bijvoorbeeld parachutes). Zou in de loop der tijd een aantal van deze cellen verloren gaan, bijvoorbeeld door mankementen ontstaan tijdens het neerlaten van de cellen of door het uitschakelen van de elementen door de tegenstander, dan zou dit slechts leiden tot een langzaam achteruitgaan van de resolutie van het systeem, maar niet tot het plotsklaps uitvallen van de radar. Ook in een niet militaire omgeving kan deze eigenschap bruikbaar zijn indien men hoge eisen aan de betrouwbaarheid van het systeem stelt.

Uiteraard kleven er een aantal nadelen aan deze opzet die de bruikbaarheid beperken. Genoemd zijn al het probleem van praktische uitvoerbaarheid en kosten. Elke antenne-element dient op één of andere wijze aan een (gedistribueerde?) centrale door te zenden of het element aan- of uitstaat. Dit vereist communicatie (alhoewel eenvoudig en slechts in een richting) over mogelijk grote afstanden. Een tweede

probleem is dat de positie van elk element (ongeveer) bekend dient te zijn. Bij sommige toepassingen, zoals bovengenoemde militaire, hoeft deze informatie niet op voorhand bekend te zijn.

Een meer fundamentele beperking van de radarantenne komt naar voren indien meerdere objecten tegelijkertijd waargenomen moeten worden. Zoals uitgelegd in paragraaf 5.5.3 werkt de "radar" optimaal als ongeveer de helft van de elementen actief zijn. Mochten er door te veel objecten in de ruimte te veel cellen actief dreigen te worden, dan kan dit worden tegengegaan door R te verkleinen. Een gelijksoortig probleem dat ook reeds in 5.5.3 werd geduid is het ontstaan van "ghost" objecten. Mogelijk zijn deze te verwijderen door aanpassingen van R en of slimme softwarevoorzieningen.

6.4 De neurale filosofie toegepast op de low-level signaal bewerking

Onder low-level bewerking in radar wordt hier verstaan het bewerken van radarsignalen zonder te denken in termen van "objecten" of iets dergelijks. Men zou kunnen spreken van filteren van "ruwe" radardata.

6.4.1 Ruisfiltering

Een eerste vorm van bewerking van radar data kan bestaan uit het wegfilteren van ruis uit de radardata. Dit kan gebeuren door in het tijddomein te filteren (middelen) maar ook door zeer kleine "spots" op het scherm weg te filteren. Beide manieren van filteren worden hier bedoeld.

Ten grondslag aan het idee van spatieel filteren ligt het artikel van Geman en Geman [16]. In [16] wordt het proces van annealing gebruikt voor het herstellen van met ruis verstoorde foto's.
Om spatiële filtering toe te passen wordt voor elk "pixel" op het (ruwe) radarscherm (of data array) een cel gedefinieerd. Zie figuur 6.3.

Volgens een op de harmonie filosofie georiënteerde aanpak zou men hier kunnen spreken van de feature-cellen.
De aanname bij de filtering is nu dat de resolutie zo groot is dat een waargenomen object altijd meerdere pixels doet oplichten. Voorts wordt gebruik gemaakt van knowledge-cellen met positieve symmetrische verbindingen naar pixels die bij elkaar in de buurt liggen. Figuur 6.4 geeft dit weer.
Cellen op het scherm zullen nu alleen hoog worden indien er in een omgeving meerdere pixels aanstaan. Op deze manier wordt een spatiële filtering geïntroduceerd.
De gevoeligheid van de filtering (hoeveel naburige pixels zijn nodig om weergegeven te worden op het scherm) kan worden ingesteld door de drempel van de cellen te veranderen. Een lage drempel zal er toe leiden dat relatief weinig data nodig is voor het actief worden van pixels op het scherm. Een hoge drempel laat alleen "grote" objecten door.

Een tijdsafhankelijke filtering kan op een gelijke manier worden bereikt. Hiertoe worden enige plaatjes na elkaar bewaard in een soort van "pipeline" architectuur. Deze "pipeline" van bewaarde beelden vormen wederom de feature-cellen. Tussen de beelden liggen de knowledge-cellen die hetzelfde karakter hebben als bij de spatiële filtering. Zie figuur 6.5. Als gedurende een aantal van de tijdsintervallen een pixel op een bepaalde cel actief is, zal dit pixel stabiel worden en op het scherm worden weergegeven. Het aantal hiervoor benodigde actieve cellen is wederom afhankelijk van de drempel.

Uiteraard kan een combinatie van spatiële en temporele filtering worden doorgevoerd. Er zijn dan zowel knowledge-cellen die verbonden zijn met feature-cellen in een "tijdsplaatje" als cellen die verbonden zijn met feature cellen uit diverse plaatjes (zie figuur 6.6). Op deze manier kunnen doelsafmeting en verleden interfereren.

Uiteraard is (softwarematig) filteren in het tijdsdomein of spatiële domein ook met de klassieke computer mogelijk (en zelfs met een grotere

flexibiliteit dan met een neuraal net). Belangrijk is echter dat bij toepassing van de neurale filosofie "real-time" filtering van radarbeelden te implementeren is, alhoewel helaas wel tegen hoge hardwarekosten.
Mogelijk dat het gebruik van VLSI schakelingen bovenstaande manier van filteren ook praktisch haalbaar kan maken.

6.4.2 Clutterfiltering en snelheidsdetectie

Nauw verwant aan bovenstaand idee is een vorm van clutter filtering toe te passen. Dit kan met vrijwel dezelfde hardwaregebeuren als beschreven in het bovenstaande.

In dit geval maakt men wederom gebruik van "tijdframes" (data op diverse achtereenvolgende tijdstippen) zoals boven beschreven. Voor het stabiel worden van een pixel is nu echter vereist dat dit pixel beweegt. Indien een pixel op twee tijdframes op dezelfde plaats is zal het niet stabiel worden, indien het pixel op twee opvolgende tijdstippen over een plaats verschoven is wordt dit patroon wel stabiel. Stilstaande objecten als huizen, bomen en trage regenbuien worden zo uitgefilterd. Snel bewegende objecten als vliegtuigen worden doorgelaten. Figuur 6.7 toont hoe knowledge-cellen mogelijk verbonden kunnen worden om dit filter te realiseren. Hier wordt gebruik gemaakt van slechts twee frames. Indien meerdere tijdframes beschikbaar zijn kan nog meer "snelheidsgevoeligheid" worden ingebracht (namelijk over langere perioden).

Tenslotte kan men het filter eenvoudig zo ontwerpen dat het alleen gevoelig is voor snelheden in bepaalde richtingen. Dit zou bijvoorbeeld bruikbaar kunnen zijn om die objecten uit te filteren die in een "verboden" richting bewegen, zoals schepen die op de wal afvaren of vliegtuigen die tegen de richting van luchtwegen in vliegen.

6.4.3 Contour extractie

Alhoewel bij radar normalitair geen gebruik gemaakt wordt van contouren om objecten te herkennen, daar het oplossend vermogen van de radar tov de objecten te klein is, is zoiets dergelijks in de toekomst wellicht wel degelijk mogelijk (Zie [24]).

De filtering zoals boven beschreven heeft veel weg van spatiële Fourier filtering zoals bekend uit de literatuur. Met celnetwerken kan dan ook eenvoudig een soort van contourfiltering plaatsvinden. Hiertoe moeten cellen slechts actief worden indien niet te veel en niet te weinig buurcellen actief zijn. Deze vorm van spatiële filtering wordt reeds lang toegepast bij het bepalen van contouren.

Een hierop gelijkende vorm van contourfiltering is gebaseerd op [16]. In dit geval wordt gebruik gemaakt van *oriëntatie*gevoelige cellen. De cellen zullen dus actief worden indien een streep in een bepaalde richting wordt waargenomen. Een annealing schema kan wederom worden toegepast om de contour optimaal te bepalen.
Richtinggevoelige "knowledge"-cellen kunnen eenvoudig geïmplementeerd worden door de cellen die op een bepaalde lijn liggen positief met elkaar te verbinden.

Men kan echter nog verder gaan in de contourfiltering. Het netwerk kan eenvoudig worden uitgebreid met knowledge-cellen die alleen verbonden zijn met de richtinggevoelige contourcellen. Immers, als twee naburige contourcellen een lijnstuk in verticale richting aangeven, is het waarschijnlijk dat aansluitende vertikale contourcellen dit eveneens zullen doen, althans als het waar te nemen object relatief weinig "hoekig" is. In figuur 6.10 is weergegeven hoe met de contourcellen verbonden knowledge-cellen er toe kunnen leiden dat het filter in feite extra gevoelig wordt voor bepaalde richtingen. Op deze manier wordt het mogelijk een vloeiende contourlijn te verkrijgen, zelfs indien de data incompleet en gestoord is.

6.5 De neurale filosofie toegepast op high-level signaal bewerking

Tenslotte zullen de high-level bewerkingen van radar informatie worden bezien. In dit geval dient men de informatie te "interpreteren" in termen van waargenomen objecten.

6.5.1 Herkenning met het Hopfield geheugen

Een neuraal geheugen kan getraind worden voor herkenning van bepaalde vormen op een bepaalde plaats. In [24] wordt beschreven hoe contouren van vliegtuigen verkregen met hoge resolutie radarbeelden herkend kunnen worden met behulp van een Hopfield net. Opgemerkt moet wel worden dat dit artikel met een flinke korrel zout genomen moet worden. Immers, de geleerde data in een Hopfield geheugen is niet translatie of rotatie invariant (verschuiving of draaiing van het beeld) noch in staat te schalen naar kleinere of grotere objecten met een gelijkvormige contour. Een nadeel van het Hopfield netwerk is eveneens dat zeer grote aantallen cellen nodig zijn om meerdere verschillende vormen te onthouden. Als een eerste aanzet op dit gebied is [24] echter zeker een interessant artikel, mits kritisch gelezen.

Meer praktische waarde krijgt het Hopfield netwerk indien niet van de rechtstreekse pixeldata geleerd wordt maar meer invariante data als de frequentiecomponenten van de twee (of drie) dimensionale spatiële Fourier decompositie van een plaatje. Deze toepassing lijkt meer reëel dan de in [24] geschetste en zelfs tot betere resultaten te leiden, daar minder data onthouden hoeft te worden. Voorwaarde is uiteraard wel dat de Fourier componenten verkregen kunnen worden en werkelijk onafhankelijk zijn van de object(af)stand.
Soortgelijke resultaten kunnen overigens verkregen worden met het back propagation algoritme (zie onderstaande).

Het bijzondere aan het gebruik van een Hopfield netwerk voor deze doeleinden is dat zij data kan herstellen, gebruik makend van op een

willekeurige manier incomplete of gestoorde data, plus dat alle "herstel"stappen in de goede richting zijn (mits voldoende cellen in verhouding met het aantal te onthouden typen). Dit maakt een zeer snelle classificatie mogelijk. Helaas staat het Hopfield geheugen parallel vuren van de cellen niet toe maar moeten deze random na elkaar gevuurd worden. Zoals eerder aangegeven in 5.5.6 zijn hiervoor wellicht oplossing denkbaar zodat verdere versnelling van het algoritme mogelijk is.

6.5.2 Classificatie met behulp van een multi-layer perceptron

Multi-layer perceptrons zijn met behulp van het back propagation algoritme in staat willekeurige correlaties te leren tussen in- en uitgangsvectoren (zie paragaraaf 5.4.4). Indien men voor de ingangsvectoren doelspecifieke informatie gebruikt en deze in een trainingsfase correleert met typen doelen, kan men een multi-layer perceptron als doelsclassificator gebruiken.
Als trainingsdata zou men spatiële contourinformatie (verkregen met behulp van een Fourier decompositie) kunnen gebruiken, zoals in bovenstaande, maar evengoed bijvoorbeeld de frequentiecomponenten van een "echo" of de amplituden van een echo in het tijddomein.

Het bijzondere van het back propagation algoritme is dat het algoritme het mogelijk maakt naast lagere ook hogere orde correlaties te ontdekken. Voor belangrijke correlaties zullen synapsen "gereserveerd" worden. Zoals aangegeven in 5.4.3 is het algoritme helaas traag en kan het stranden op locaal optimale classificaties in plaats van de globaal optimale. In 5.4.3 e.v. is ook aangegeven hoe een en ander mogelijk te verhelpen is. Onderzoek naar de effecten van verschillende aantallen cellen in cellagen, verschillende aantallen lagen en verschillende typen uitgangsfuncties (sinus, arctangens, sign functie enz) kunnen meer inzicht geven in het praktisch ontwerpen van multi-layer perceptrons (MLP's).

Is een MLP echter eenmaal correct getraind dan kan het zeer snel classificaties doorvoeren. Het MLP classificeert immers in slechts N slagen, met N het aantal lagen layers. Een layer mag immers geheel parallel gevuurd worden.

6.5.3 Multi sensor integratie met de Boltzmann machine

Zoals beschreven in hoofdstuk 5 kan een Boltzmann machine gebruikt worden voor het oplossen van "soft constraint" problemen. In hoofdstuk 5 werd dit concept gedemonstreerd aan de hand van een "logicapuzzel", waar de constraints overigens hard waren.
De "logicapuzzel" is eenvoudig om te zetten naar een meer serieuze toepassing voor radar. Indien men het probleem weet te herschrijven in een vorm zoals die van de puzzel, doet het probleemgebied er uiteraard niet toe. Zaak is alleen dat men een aantal "exor" verbanden weet te geven als "het object is een vliegtuig, clutter of een missile" (slechts een van de gekozen mogelijkheden) , "het object beweegt snel, langzaam of staat stil" enz. plus verbanden van de vorm als A en B plus B en C dan ook A en C, bijvoorbeeld, "als het object een vliegtuig is en snel beweegt, plus dat de reflectie sterk is bij een vliegtuig, dan moet ook gelden dat de sterke reflectie snel beweegt". De informatie over wat het waargenomen object is kan nu geschieden door een zelfde annealing procedure toe te passen als bij de puzzel. Sensorinformatie van radar, IFF, contouren detectoren, frequentiepatronen van de reflecties enz. dienen nu als invoer voor feature cellen. Na een annealing procedure zullen de cellen in de "meest waarschijnlijke" configuratie bij de sensorinput gaan staan.

Op deze manier kan sensorinformatie gecombineerd worden om tot een oordeel over de aard van het voorwerp te beslissen. Natuurlijk zijn andere dan het "puzzel"netwerk mogelijk en zelfs wenselijk. Het puzzelnetwerk is in feite geschapen om N objecten die geen features gemeenschappelijk hebben(!) te classificeren. In de werkelijkheid zal men te maken hebben met een variabel aantal objecten die wel degelijk

overlap kunnen vertonen (bijvoorbeeld, het zijn beide snelvliegende voorwerpen). Het netwerk zoals getoond is slechts een eerste aanzet tot een classificator en verder onderzoek is nodig om te zien of meer flexibel inzetbare netwerken ontworpen kunnen worden.

Door de grootte van de gewichten tussen inputcellen en sensoren te veranderen kan de ene sensor "belangrijker" gemaakt worden dan de andere (meer absoluut gewicht). Een prettige eigenschap van de Boltzmann aanpak is dat sensoren elkaar mogen tegenspreken of zelfs kapot kunnen zijn (uitgangswaarde 0) maar dat het netwerk dan toch in de meest waarschijnlijke toestand zal geraken. Eis voor een goede classificatie is slechts dat er in totaal "voldoende" sensorinformatie aanwezig is.

Het op deze manier combineren van sensorinformatie heeft de voordelen van een veelzijdige combinatie van alle informatie op een bepaald moment in een zeer korte tijd (daar het algoritme parallel kan werken). Deze manier van sensorintegratie lijkt veel op een meer "klassieke" combinatie van informatie met behulp van een expertsysteem zoals getoond in [55]. Bij de Boltzmann methode worden echter geen "zoekprocedures" gebruikt (regel vuren o.i.d.) zoals in een expertsysteem maar wordt steeds alle sensor- en afgeleide informatie parallel gecombineerd, een belangrijk voordeel als men op real-time basis wenst te werken.

Een nadeel is dat er bij de Boltzmann machine mogelijk veel ontwerptijd uitgetrokken moet worden voor de keuze van de gewichten en de cellen (hetzelfde geldt overigens voor het kiezen van de gewichten in een expertsysteem). Wellicht dat trainingsproceduren voor Boltzmann netwerken (die echter op dit moment extreem traag zijn) deze situatie kunnen verbeteren.

Verder onderzoek in deze richting kan mogelijk bruikbare ontwerpen opleveren van netwerken die real-time sensorinformatie van verschillende bronnen en met een verschillende "kwaliteit" en "belangrijkheid" op een

rijke manier kunnen combineren. Een leerregel zou het praktische ontwerp zeer vereenvoudigen.

6.5.4 Bepaling van de doelsrichting en het doelsaantal bij phased array antennes

Bij *phased array antennes* (zie figuur 5.11) is het theoretisch mogelijk uitgaande van de "vorm" van de over het antenne array staande golf af te leiden hoeveel doelen en in welke richting aanwezig zijn. In de praktijk is het echter zeer lastig hiervoor bruikbare algoritmen te vinden. Voor een verhandeling over phased array antennes en algoritmen zie [86].

Een antenne array kan beschreven worden als een rij elementen die alle getroffen worden door een door een of meerdere objecten gereflecteerde golf.
Voor een wiskundige beschrijving van dit signaal zie [86].
Het aantal doelen kan geschat worden door de aantallen (verschillende) spatiële frequenties over het array te schatten (mits uiteraard geldt dat alle objecten onder verschillende hoeken worden waargenomen!), de richting waar het doel zich bevindt kan dan aan de hand van de bij het doel behorende frequentie worden herleid. Voor de spatiële frequentie geldt:

$$f_k = \sin(\alpha)/l \qquad [6.1]$$

met α de gezochte hoek en l de golflengte van de radargolven. De doelsfase ϕ_n is evenredig met de afstand van x tot het midden van het array (zie figuur 6.11).
Mogelijk kan de afstand tot het object geschat worden uit de object amplitude (indien al de objecten een even groot signaal afgeven).
Voor de amplitude van een antenne element op positie x kan berekend worden dat geldt

$$S(x) = \sum_{n}^{N} A_n.(\sin (2\pi f_n - \phi_n)) \qquad [6.2]$$

met A_n, ϕ_n en f_n de doelsamplitude, (spatiële) fase en frequentieterm. N is gelijk aan het aantal doelen in de buitenwereld.

Zoals aangegeven in 5.4.3 en verder kan een neuraal layer netwerk mogelijk gebruikt worden voor het ontbinden van een signaal in sinusvormige componenten. Indien de arraydata gebruikt worden om het netwerk te trainen zullen de celparameters (synapsen) de gezochte waarden aannemen. Immers, een layer netwerk zoals gegeven in figuur 6.12, met een lineaire uitgangcel en "sinus"cellen in de onderste laag zullen het signaal met behulp van het back propagation algoritme na training ontbinden volgens:

$$S_j = \sum_{i} W_{ij}.(\sin (W_{ki}.S_k - T_i)) - T_j \qquad [6.3]$$

De componenten W_{ki} komen overeen met de gezochte frequenties (met faseverschil T_i) van de waargenomen objecten. W_{ij} geeft weer hoe sterk de bijdrage van een component is (de amplitude van de echo).
Op deze manier zijn doelsamplitude en doelshoek bekend (bij een van tevoren onafhankelijk aantal doelen!).

Er kleven helaas een aantal bezwaren aan deze aanpak. Uit verrichtte simulaties is gebleken dat een netwerk met dit soort "sinuscellen" niet onder alle omstandigheden numeriek stabiel is (zie ook 5.4.3). Het bleek dat het mogelijk is dat sommige cellen naar als maar hogere frequenties divergeren. Theoretisch onderzoek is dus vereist voor verdere uitwerking van het idee.

In de tweede plaats is er geen garantie dat niet meerdere cellen dezelfde frequentie kunnen gaan coderen. Immers, de som van twee sinustermen van een bepaalde frequentie is weer een sinus van dezelfde

frequentie. Voor het tellen van het aantal doelen in verschillende richtingen is het dus niet voldoende het aantal cellen met amplitude ongelijk nul te tellen, maar moet het aantal cellen met verschillende frequenties geteld worden.
Bij frequenties die dicht bij elkaar liggen is dan niet langer te zeggen of het hier een of twee doelen betreft. Hierdoor wordt de resolutie van het algoritme beperkt.

Als laatste bezwaar geldt dat het trainen van het netwerk traag gaat en dat men bovendien maar over een beperkte reeks "trainingspunten" beschikt. Het aantal punten dat men van de golfvorm bezit is immers gelijk aan het aantal elementen. Deze beperking leidt ongetwijfeld tot een beperkte nauwkeurigheid bij het bepalen van de frequenties (vergelijkbaar met de slechte prestaties van de Fourier decompositie zoals behandeld in [86]).
Een groter aantal elementen zal tot betere resultaten leiden.

Bijzonder is wel dat het netwerk adaptief gebruikt kan worden. Het kan op een gegeven moment een aantal cellen naar bepaalde frequenties "dwingen" en is dan gevoelig voor deze frequenties bij dat aantal doelen. Tijdens de vlucht van de objecten veranderen deze amplitudes en frequenties langzaam. Het is uiteraard niet nodig (en het zou zelfs jammer zijn) als alle geleerde data steeds gewist zou worden. Daarmee biedt het algoritme een alternatief voor algoritmen als de Fourier analyse, waar geen a priori data beschikbaar is en algoritmen als het maximum likelihood algoritme (zie [86]) waar de a priori kennis (hier het aantal doelen maar niet de hoeken) star is. Bij het neurale netwerk wordt a priori informatie (de synapswaarden op dat moment) gecombineerd met nieuwe informatie maar tijdens dit proces kunnen de a priori waarden zich een beetje aanpassen. Daarmee combineert deze methode de twee voordelen van de Fourier decompositie en de maximum likelihood methode.

Al met al is een algoritme dat berust op ontbinding van een golfvorm in een beperkt aantal adaptief te kiezen frequenties wellicht zeer

bruikbaar voor gebruik in een phased array radar, maar is er nog veel (theoretisch) onderzoek nodig naar de praktische haalbaarheid van dit idee.

6.6 Nabespreking

Dit hoofdstuk toonde een wilde verzameling van mogelijke toepassingen van de neurale netwerken in radar. Uiteraard zijn er vele andere vormen van toepassingen mogelijk, met name omdat in de komende jaren zeker nog krachtiger netwerken en leerregels ontdekt zullen worden. Hopelijk zijn de hier gegeven voorbeelden inspiratie voor andere toepassingen.

De bovenstaande voorbeelden zijn allemaal met de in dit verslag beschreven theorie te realiseren, alhoewel verder onderzoek vaak geboden is. Geen van de netwerken doet iets wat voordien met een klassiek algoritme absoluut onmogelijk was, maar bijzonder is de wijze waarop de netwerken de taak volbrengen. Dit betekent in de praktijk meestal snelheidswinst (door het parallelle karakter van de algoritmen) en robuustheid, overigens tegen hogere hardwarekosten. Belangrijk is in te zien dat de netwerken stuk voor stuk tekortkomingen hebben, zoals aangegeven bij de behandeling in hoofdstuk 5 en 6 en vaak nog voor "verbetering" vatbaar zijn. Het zijn echt geen "alleskunners".

Van neurale netwerken moet men dan ook niet het idee hebben dat zij alle problemen zullen oplossen, maar men moet goed weten wat voor fouten kunnen optreden en hoe deze te voorkomen zijn. Ook al is een netwerk in staat te leren, dit wil niet zeggen dat er daarom geen gedegen kennis van het netwerk vereist is. Bij het ontwerp van het netwerk moet men zelf kiezen voor een geschikt aantal cellen, een geschikt verbindingspatroon, geschikte synapswaarden of leerregels, een geschikte uitgangsfunctie van de cellen en de juiste update regel. Maakt men hiervoor goede keuzen dan kan een netwerk mogelijk goed presteren maar dan nog dient men altijd goed op de hoogte van de beperkingen van het netwerk te zijn. Overenthousiaste verwachtingen van neurale netwerken

zullen alleen tot teleurstelling lijden, deskundig ontworpen netwerken tot mogelijk goede applicaties.

7 CONCLUSIES

7.1 Neurale Netwerken

De ontwikkelingen op theoretisch en praktisch gebied aangaande neurale netwerken ondergaan momenteel (1987-1988) een explosieve groei. Nieuwe netwerken en mathematische inzichten hebben geleid tot een veelheid van demonstratieve toepassingen.
Neurale netwerken onderscheiden zich van andere bestaande systemen door hun hoge graad van parallelliteit, de eenvoud van de processoren en de communicatie plus de ongevoeligheid van de netwerken voor ruis en beschadigingen.
De netwerken kunnen worden ingedeeld in drie categorieën, te weten de classificatoren, de geheugennetwerken en de "soft constraint problem solvers", alhoewel al deze categorieën enigszins in elkaar overvloeien en veel op elkaar lijken.

Het ontwerpen van een neuraal netwerk is bij lange na niet triviaal en gedegen theorie is helaas slechts voor een klein aantal netwerken beschikbaar. Een netwerk is weinig flexibel, wat wil zeggen dat met de keuze van het celtype en de aard van de verbindingen de functioneringswijze vastligt, maw. het netwerk is "dedicated" en geen "alleskunner". Het is dus op dit moment zeker niet zo dat een enkel netwerk tegelijkertijd een groot aantal verschillende functies kan verrichten. Een (eenvoudig) neuraal netwerk is wat dat betreft niet te vergelijken met de veel flexibelere computer, maar kan slechts dienen als "onderdeel" van een groter systeem.
Een neuraal netwerk biedt mogelijk een betrouwbaar en zeer snel alternatief voor het uitvoeren van veel bestaande algoritmes met een "von Neumann" computer. Leerregels maken dat een eenmaal ontworpen netwerk zich ook adaptief kan gedragen.

Wellicht dat verdere ontwikkeling van de theorie netwerken zal opleveren die meer ingewikkeldere taken dan de in dit verslag geschetste kunnen uitvoeren.

Een interessante eigenschap van de netwerken (naast de elegante architectuur en de snelheid van de netwerken) is de mogelijkheid de netwerken te trainen met behulp van niet meer dan voorbeelden. In een gebied waarin de correlaties tussen gegevens onbekend zijn, zijn veel typen netwerken in staat deze door middel van "leerregels" toch te achterhalen. Met name interessant is dat diverse typen netwerken in staat zijn hogere orde correlaties te ontdekken en te benutten, wat hun prestaties goed doet uitkomen vergeleken met bestaande algoritmen. De mogelijkheid deze netwerken te implementeren in parallelle hardware verhoogt de praktische bruikbaarheid verder.

7.2 Neurale Netwerken voor Radar

Diverse eigenschappen van de neurale netwerken maken hen bij uitstek geschikt voor toepassing in radarsystemen. Met name de snelheid van de netwerken is zeer bruikbaar in een radaromgeving waar in korte tijd zeer veel data verwerkt moet worden.

Als toepassingen kan gedacht worden aan de radarantenne, low-level beeldbewerking als contourextractie en ruisonderdrukking en high-level bewerkingen als multi sensor integratie en classificatie. Mogelijk bieden de netwerken aanknopingspunten met, en nieuwe inzichten in, bestaande technieken. Veel typen netwerken zijn in principe in staat met behulp van eenvoudige leerregels uitgaande van frequentie (of soortgelijke) karakteristieken zeer snelle classificaties uit te voeren en deze te leren aan de hand van voorbeelden. De eigenschap dat zij met leerregels zelfs hogere orde correlaties kunnen ontdekken (alhoewel het leren traag gaat) lijkt hen goed geschikt voor deze taak te maken.

succesvolle software simulatie kan hierna eventueel leiden tot het uitvoeren van het netwerk in parallelle hardware. Bij het ontwerp zal met name gekeken moeten worden naar de aantallen cellen en cellagen waarbij het perceptron goed functioneert, de celuitgangsfunctie die geschikt is voor de toepassing en mogelijke aanpassingen van de leerregel (snelheid en stabiliteit).
Uiteraard is het zaak de snelle ontwikkelingen op het neurale netwerk gebied te volgen.

Ir. H.J. Borgers

d.a.

Ir. G.H. Heebels

LITERATUURVERWIJZINGEN EN RELEVANTE LITERATUUR

1 Edwin R. Lewis, "The elements of single neurons: A review", IEEE transactions on Systems, Man and Cybernetics, Vol SMC-13, Number 5, sept/oct 1983, pp 702-710

2 Shun-Ichi Amari, "Field theory of self-organizing neural nets", IEEE transactions on Systems, Man and Cybernetics, Vol SMC-13, Number 5,sept/oct 1983, pp 741-748

3 Vincent Torre, W. Geoffrey Owen and Giulio Sandini, "The dynamics of electrically interacting cells", IEEE transactions on Systems, Man and Cybernetics, Vol SMC-13, Number 5, sept/oct 1983, pp 757-765

4 Erich Harth, "Order and chaos in neural systems: an approach to the dynamics of higher brain functions", IEEE transactions on Systems, Man and Cybernetics, Vol SMC-13, Number 5, sept/oct 1983, pp 782-789

5 Andrew G. Barto, Richard S. Sutton and Charles W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems", IEEE transactions on Systems, Man and Cybernetics, Vol SMC-13, Number 5, sept/oct 1983, pp 835-846

6 David G. Bounds, "New optimization methods from physics and biology", Nature vol. 329 17 sept 1987, pp 215-219

7 Richard Durbin and David Willshaw, "An analogue approach to the travelling salesman problem using an elastic net method", nature vol. 326 16 april 1987, pp 689-691

8 William P. Jones and Josiah Hoskins, "Back propagation, a generalised delta learning rule", BYTE october 1987, 155-162

9 Alan Lapedes and Robert Farber, "Nonlinear signal processing using neural networks: prediction and system modelling", proceedings of IEEE, preprint, Los Alamos National Laboratory

10 Vuiks, "Dreigend stigma in de computerwetenschap, vanPavlov tot Asimov", Personal Computer Magazine, september 1987, pp 160-164

11 A. Huizing, "VHF/UHF-Classificatie van doelen met behulp van neuronale netwerken", interne publicatie div 3-3, FEL/TNO

12 B.M. Forrest, D. Roweth, N. Stroud, D.J. Wallace and G.V. Wilson,"Implementing neural networks on parallel computers", the computer journal, vol. 30, no. 5 1987 pp 413-419

13 David W. Tank and John J. Hopfield, "Simple 'neural' optimization networks: an A/D converter, signal desicion circuit, and a linear programming circuit", IEEE transactions on circuits and systems, vol. CAS-33, No. 5, may 1986, pp 533-541

14 Paul J. Werbos, "Building and understanding adaptive systems: a statistical/numerical approach to factory automation and brain research", IEEE transactions on systems, man and cybernetics, vol. SMC-17, No.1, jan/feb 1987, pp 7-20

15 Henri Prade, "A computational approach to approximate and plausible reasoning with applications to expert systems", IEEE transactions on pattern analysis and machine intelligence, vol. PAMI-7, No. 3, may 1985

16 Stuart Geman and Donald Geman, "Stochastic relaxation, gibbs distributions and the bayesian restoration of images", IEEE transactions on pattern analysis and machine intelligence, vol. PAMI-6, No. 6, nov 1984

17 David W. Tank and John Hopfield, "Collective computation in neuronlike circuits", Sci. American dec 1987 pp 62-71

18 Yaser S. Abu-Mostafa and Jeannine-Marie St. Jacques, "Information capacity of the Hopfield model", IEEE transactions on information theory, vol. IT-31,No. 4, july 1985 pp 461-464

19 Michael A. Cohen and Stephen Grossberg, "Absolute stability of global formation and parallel memory storage by competitive neural networks", IEEE transactions on systems, man and cybernetics, vol SMC-13, No 5, sept/oct 1983

20 Robert van der Zwan, "Parallelle computers: rekenen op topsnelheid", intermediair, 23e jaargang 50 - 11 dec 1987 pp 9-57

21 Richard P. Lippmann, "An introduction to computing with neural nets", IEEE ASSP magazine, april 1987 pp 4-22

22 Gary Josin, "Neural-network heuristics, three heuristic algorithms that learn from experience", BYTE october 1987 pp 183-192

23 Bart Kosko, "constructing an associative memory", BYTE sept 1987,pp 137-144

24 Nabil H. Farhat, "optoelectronics builds viable neural-net memory", electronics, june 16, 1986 pp 41-44

25 Tomaso Poggio and Christof Koch, "Synapses that compute motion", Sci American, may 1987 pp 46-52

26 H.J. Borgers, "Een introductie tot het Hopfield geheugen", interne publicatie divIII-3, FEL/TNO, 1987.

27 Robert J. McEliece, Edward C. Posnar, Eugene R. Rodemich and Santosh S. Venkatesh, "The capacity of the Hopfield associative memory", IEEE transactions on information theory, vol. IT-33, No. 4, july 1987

28 W. Daniel Hillis, "The connection machine", Sci. american 1987

29 Robert Hecht-Nielsen, "Neural analog information processing", SPIE Vol. 298 Real-Time signal Processing IV 1981

30 Mati Wax and Thomas Kailath, "Detection of signals by information theoretic criteria", IEEE transactions on acoustics, speech and signal processing, Vol. Assp-33 No-2 april 1985, pp 387-392

31 A. Moopenn, John Lambe and A. P. Thakoor, "Electronic implementation of associative memory based on neural network models", IEEE transactions on systems, man and cybernetics, vol SMC-17, No. 2, march/april 1987 pp 325-331

32 J. L. McClelland, D.E. Rumelhart and G.E. Hinton, "The appeal of parallel distributed processing", Parallel distributed processing, Voll: Foundations, Rumelhart et al. 1986 pp 3-44

33 D.E. Rumelhart,G.E. Hinton and J.L. McClelland, "A general framework for parallel distributed processing", Parallel distributed processing, Vol 1: Foundations, Rumelhart et al. 1986 pp 45-76

34 G.E. Hinton, J.L. McClelland and D.E. Rumelhart, "Distributed representations", Parallel distributed processing, Vol 1: Foundations, Rumelhart et al. 1986 pp 77-109

35 D.E. Rumelhart and J.L. McClelland, "PDP models and general issues in cognitive science", Parallel distributed processing, Vol 1: Foundations, Rumelhart et al. 1986 pp 110-146

36 D.E. Rumelhart and D. Zipser, "Feature discovery by competitive learning", Parallel distributed processing, Vol 1: Foundations, Rumelhart et al. 1986 pp 151-193

37 P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory", Parallel distributed processing, Vol 1: Foundations, Rumelhart et al. 1986 pp 194-281

38 G.E. Hinton and T.J. Sejnowski, "Learning and relearning in Boltzmann machines", Parallel distributed processing, Vol 1: Foundations, Rumelhart et al. 1986 pp 282-317

39 D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning internal representations by error propagations", Parallel distributed processing, Vol 1: Foundations, Rumelhart et al. 1986 pp 318-362

40 J.L. McClelland, "Resource requirements of standard and programmable nets", Parallel distributed processing, Vol 1: Foundations, Rumelhart et al. 1986 pp 460-487

41 D. Zipser and D.E. Rabin, "P3: a parallel network simulating system", Parallel distributed processing, Vol 1: Foundations, Rumelhart et al. 1986 pp 488-506

42 D.E. Rumelhart, P. Smolensky, J.L. McClelland and G.E. Hinton, "Schemata and sequential thought processes in PDP models", Parallel distributed processing, Vol 2: Psychological and Biological Models, Rumelhart et al. 1986 pp 7-57

43 J.L. McClelland and J.L. Elman, "Interactive Processes in speech perception", Parallel distributed processing, Vol 2: Psychological and Biological Models, Rumelhart et al. 1986 pp 58-121

44 J.L. McClelland, "The programmable blackboard model of reading", Parallel distributed processing, Vol 2: Psychological and Biological Models, Rumelhart et al. 1986 pp 122-169

45 D.E. Rumelhart and J.L. McClelland, "On learning the past tenses of english verbs", Parallel distributed processing, Vol 2: Psychological and Biological Models, Rumelhart et al. 1986 pp 216-271

46 F.H.C Crick and C. Asanuma, "Certain aspects of the anatomy and physiology of the cerebral cortex", Parallel distributed processing, Vol 2: Psychological and Biological Models, Rumelhart et al. 1986 pp 333-371

47 Judea Pearl, "Evidential reasoning using stochastic simulation of causal models", Artificial Intelligence (international journal), Vol 32 Number 2 May 1987, pp 245-257

48 Geoffrey E. Hinton and Terrence J. Sejnowski, "Optimal perceptual inference", Proceedings on computer vision and pattern recognition june 19-23 1983, pp 448-453

49 Richard O. Duda, Peter A. Hart and Nils J. Nilsson, "Subjective Bayessian methods for rule-based inference systems", AFIPS conference proceedings vol 45, 1976 national computer conference, pp 1075-1082

50 Pierre Peretto and Jean-Jacques Niez, "Stochastic dynamics of neural networks", IEEE transactions on systems, man and cybernetics, vol SMC-16, no 1, jan/feb 1986, pp 73-83

51 Jiro Ihara, "Extension of conditional probability and measures of belief and disbelief in a hypothesis based on uncertain evidence", IEEE transactions on pattern analysis and machine intelligence, vol MAMI-9, no 4, july 1987, pp 561-568

52 Scott E. Fahlman, Geoffrey E. Hinton and Terrence J. Sejnowski, "Massively parallel architectures for AI: netl, thistle and Boltzmann machines", AAAI 1983, pp 109-113

53 Geoffrey E. Hinton, Terrence J. Sejnowski and David H. Ackley, "Boltzmann machines: constraint satisfaction networks that learn", Technical report CMU-CS-84-119, Carnegie- Mellon University, may 1984

54 Harry E. Stephanou and Andrew P. Sage, "Perspectives on imperfect information processing", IEEE transactions on systems, man and cybernetics, vol SMC-17, no 5, sept/oct 1987, pp 780-798

55 Philip L. Bogler, "Shafer-Dempster reasoning with applications to multisensor target identification systems", IEEE transactions on systems, man and cybernetics, vol SMC-17, no 6, nov/dec 1987, pp 968-977

56 Athanasios Papoulis, "Entropy" (chapter 15-1 to 15-4), Probability, random variables and stochastic processes, 2e edition pp 500-544

57 H.J. Borgers, "PDPS, een simulatie kernel voor PDP systemen", interne publicatie div 3-3 FEL/TNO, 1987.

58 Joachim Buhmann, Robert Divko, Helge Ritter and Klaus Schulten, "Physik und gehirn", MC-grundlagen september 1987 pp 108-120

59 Philip D. Wasserman, Tom Schwartz and Lance B. Eliot, "Neural networks (part 1)", Expert Focus, IEEE Expert, Winter 1987.

60 Philip D. Wasserman and Tom Schwartz, "Neural networks (part 2)",Expert Focus, IEEE Expert, Spring 1988.

61 J.A. Anderson, "Cognitive and Psychological Computations with Neural Models", IEEE transactions on Systems, Man and Cybernetics, Vol SMC-13, Number 5, sept/oct 1983, pp 799

62 M. Minsky and S. Papert, "Perceptrons", MIT press, Cambridge, Mass., 1968.

63 Abraham Peled, "The Next Computer Revolution", scientific american, october 1987, volume 257, Number 4

64 Geoffrey C. Fox and Paul C. Messina, "Advanced Computer Architectures", scientific american, october 1987, volume 257, Number 4

65 James D. Meindl, "Chips for Advanced Computing", scientific american, october 1987, volume 257, Number 4

66 David Gelernter, "Programming for Advaced Computing", scientific american, october 1987, volume 257, Number 4

67 Paul Smolensky, "Schema Selection and Stochastic Inference in Modular Environments", (proceedings of) AAAI 1983, pp 378-382

68 S. Kirkpatrick, C.D. Gelatt, Jr., M.P. Vecchi, "Optimization by Simulated Annealing", Science, 13 May 1983, Vol 220, Number 4598.

69 David H. Ackley, Geoffrey E. Hinton and Terrence J. Sejnowski, "A Learning Algorithm for Boltzmann machines", cognitive science 9, 147-169, 1985.

70 Geoffrey North, "A celebration of connectionism", Nature Vol. 328, 9 july 1987, pp 107.

71 David Zipser and Richard A. Anderson, "A Back-Propagation programmed Network that Simulates Response Properties of a Subset of Posterior Parietal Neurons", Nature Vol. 331, 25 february 1988, pp679-684.

72 Collin Collingridge, "The Role of NMDA Receptors in Learning and Memory", Nature Vol. 330, 17 december 1987, pp 604-605.

73 Geoffrey North, "Neural Networks: Implementation and Analyses",Nature Vol. 330, 17 december 1987, pp 522-523.

74 Edward W. Packel and J.F. Traub, "Information-Based Complexity",Nature Vol. 328, 2 july 1987, pp 29-33.

76 Paul Smolensky, "The Mathematical Role of Self-Consistency in Parallel Computation", Proceeding of the sixth annual meeting of the cognitive science society, 1984, AD-A140 877 (micro fiche).

77 Paul Smolensky and Mary S. Riley, "A parallel Model of (Sequential) Problem Solving", Proceeding of the sixth annual meeting of the cognitive science society, 1984, AD-A140 877 (micro fiche).

78 Paul Smolensky, "Thermal Parallel Models in a Computational Context", publication of the university of california, San Diego, institute for cognitive science, 1984, AD-A140 877 (micro fiche).

79 N.F. Ezquerra and L.L. Harkness, "Recognition of Targets by Radar", Publication of the Georgia Institute of Technology, USA.

80 Derek Partridge, "What's wrong with Neural Architectures", Computer Conference Spring 1987, pp 35-38..

81 Alun Anderson, "Learning from a computer cat", Nature Vol 331, 25 february 1988, pp 657-658.

82 Deborah T. Franks, Joyce A. Musselman and John W. Sapp, "Application of AI to Radar Image Understanding", Publication of Software Architecture & Engineering, Inc, February 1985, AD-A152 519 (micro fiche).

83 T.J. Sejnowski, "Open Questions About Computation in Cerebral Cortex", Parallel distributed processing, Vol 2: Psychological and Biological Models, Rumelhart et al. 1986 pp 373-389

84 R. Rosenblatt, "Principles of Neurodynamics", New York, Spartan Books 1959.

85 K. Binder et al., "Monte Carlo Methods in Statistical Physics", Springer-Verlag, Berlin Heidelberg New York, 1979.

86 A.G. Huizing, G.C. de Valk and A Theil,"High resolution angle estimation methods; an overview". FEL/TNO paper AC/243 (Panel X/RSG 3) NL-43-1.

APPENDIX A

Tekeningen

Fig. 3.1 Von Neumann architectuur
Mem = geheugen: CPU = processor

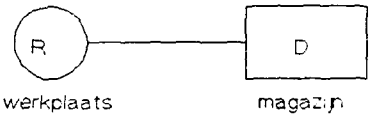

Fig. 3.2 De "von Neumann" fabriek

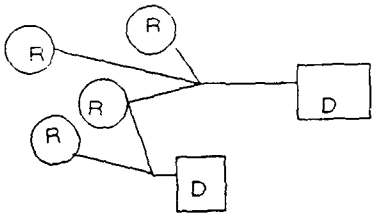

Fig 3.3 Een multi tasking fabriek

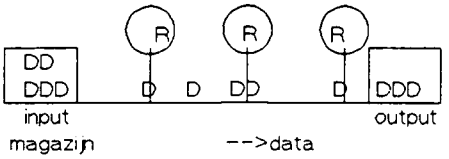

Fig. 3.4 Een pipeline fabriek

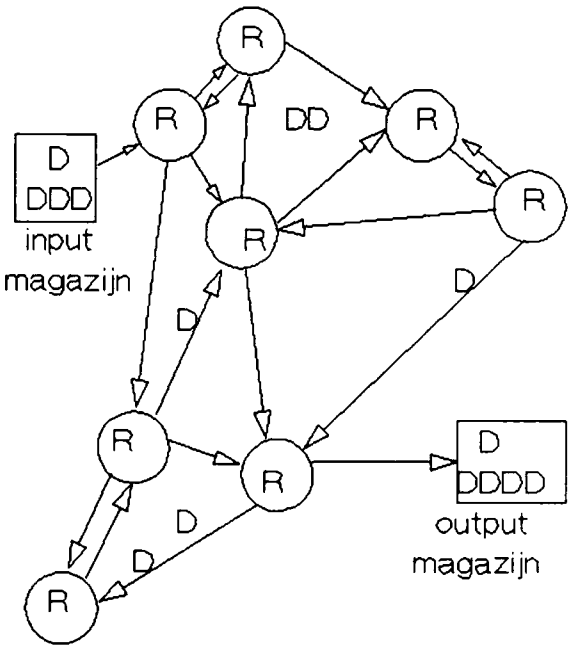

Fig. 3.5 Een gedistribueerde fabriek

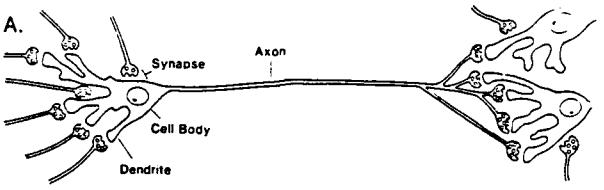

Fig. 4.1 Een neurale cel

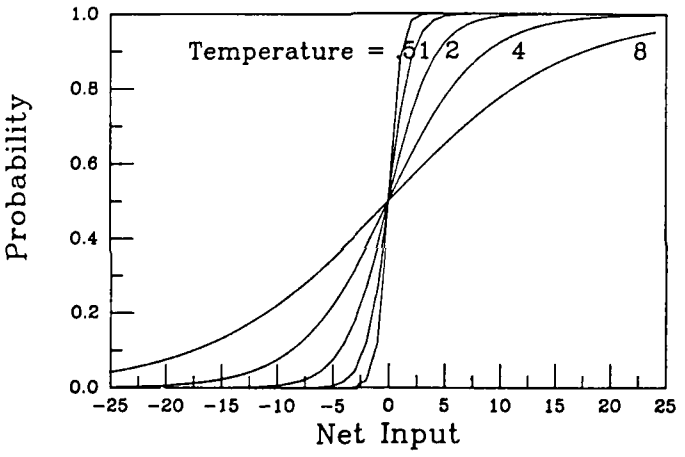

Fig. 4.2 Kansfunctie bij veschillende T

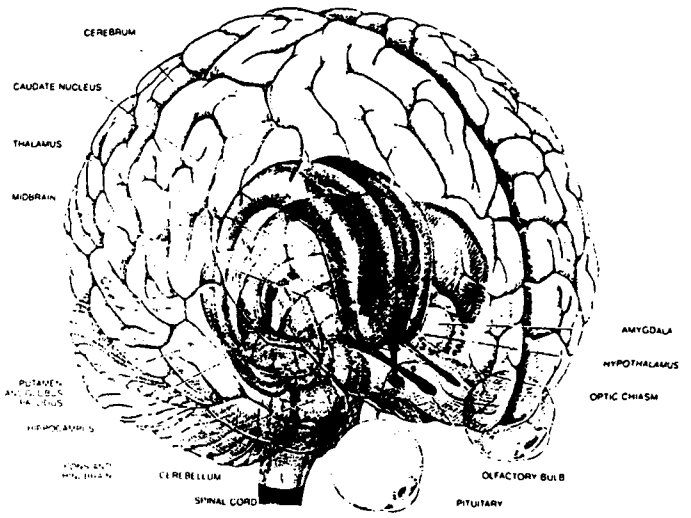

Fig. 5.1 De menselijke hersenen

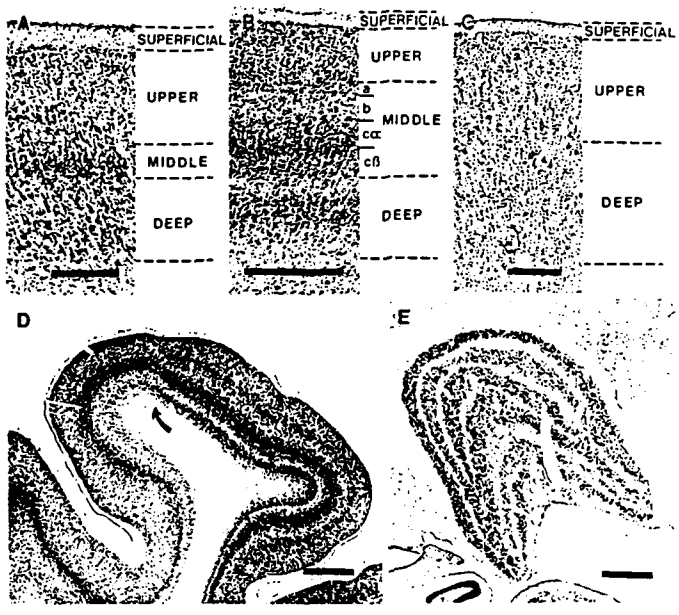

Fig. 5.2 Doorsnede van het hersenoppervlak

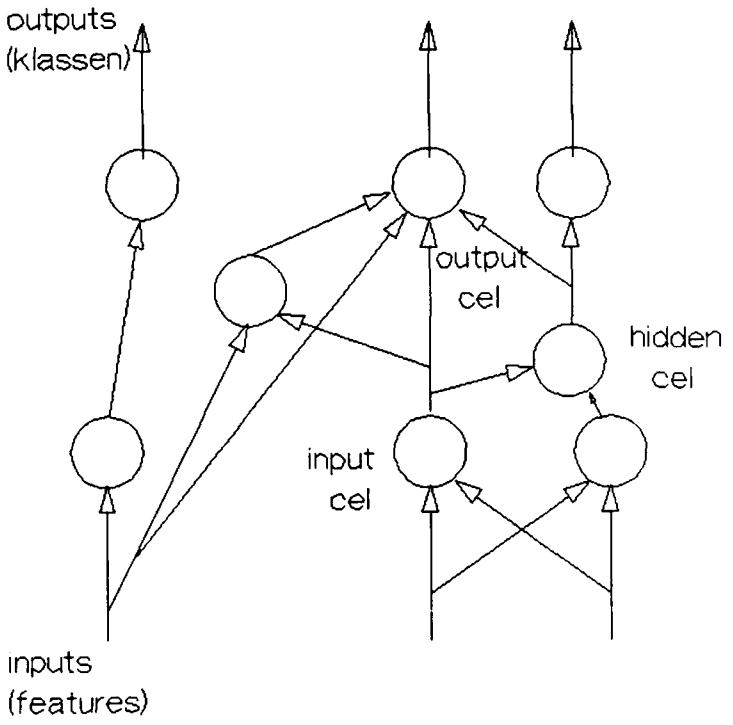

Fig. 5.3 Algemeen classificatienetwerk

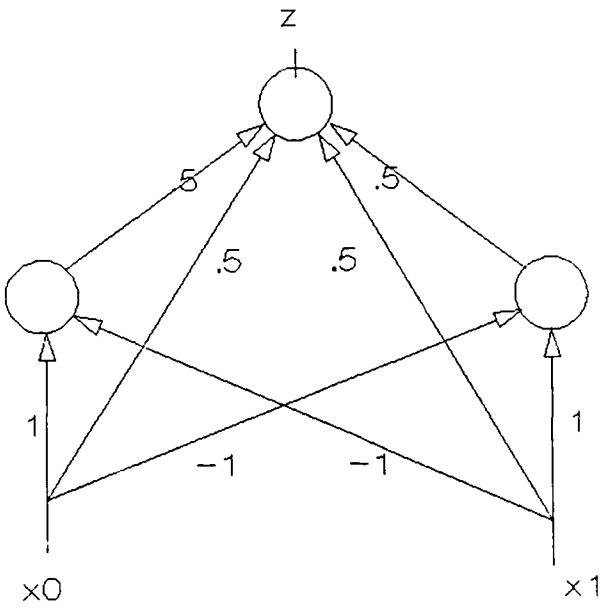

Fig. 5.4 Comparatornetwerk

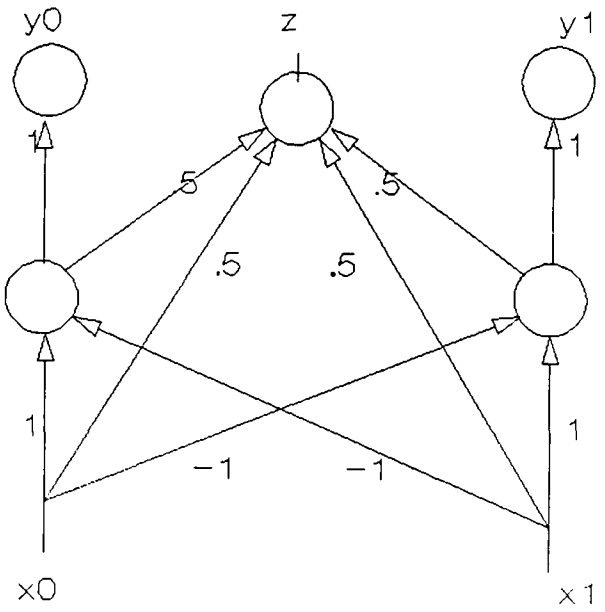

Fig. 5.5 Comparatornetwerk met uitgangscellen

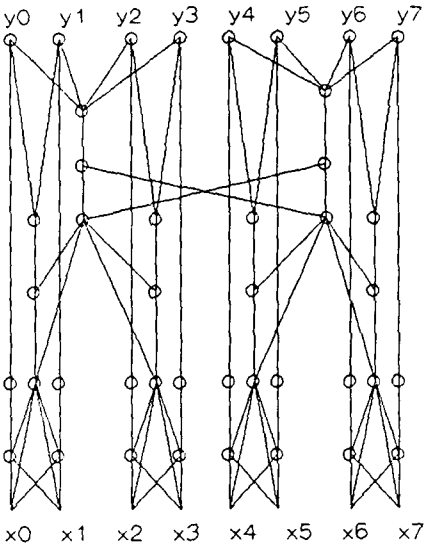

Fig. 5.6 Maximalisatienetwerk voor 8

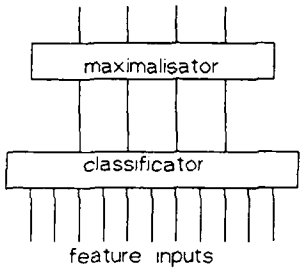

Fig. 5.7 Maximalisatienetwerk als eindtrap van een classifier

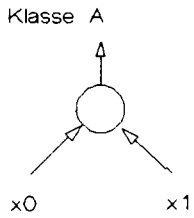

Fig. 5.8 Perceptron voor een klasse met twee feature inputs

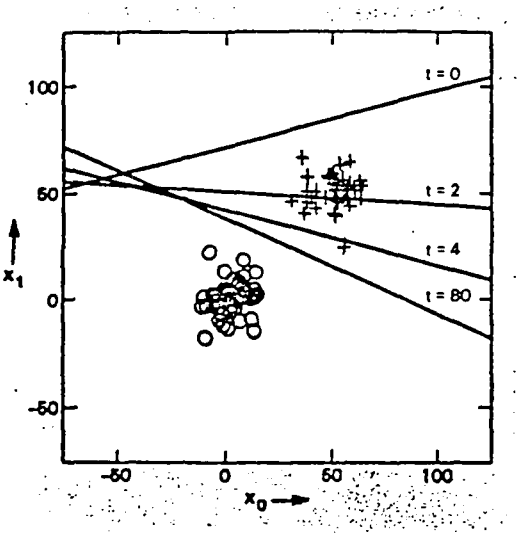

Fig. 5.9 Leersimulatie van het perceptron uit fig. 4.8 De cirkels en kruisjes geven leerdata van de te leren klasse (cel aan is klasse A, cel uit is klasse B).
De rechte geeft de besluitgrens weer op een tijdstip t tijdens het leren.
Na t = 4 worden de twee klassen correct gescheiden door de besluitgrens.

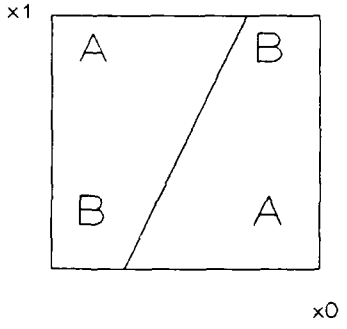

Fig. 5.10 Een enkellaags perceptron kan onmogelijk een "exclusive OR" probleem oplossen. De twee klassen zijn niet door een besluitgrens te scheiden.

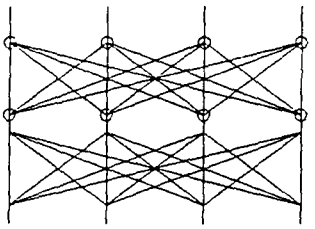

Fig. 5.11 Tweelaags perceptron

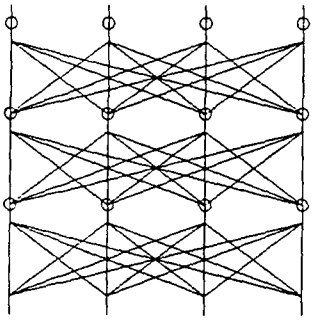

Fig. 5.12 Drielaags perceptron

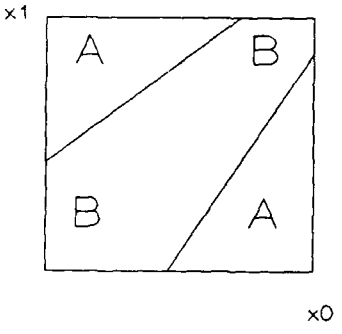

Fig. 5 13 Een tweelaags perceptron kan het "exclusive OR" probleem wel oplossen. Twee rechte besluitgrenzen vormen tezamen een hypervlakbesluitgrens.

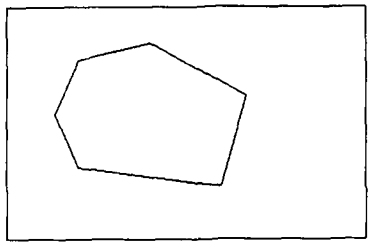

Fig. 5 .14 Convexe besluitgrens voor tweelaags perceptron

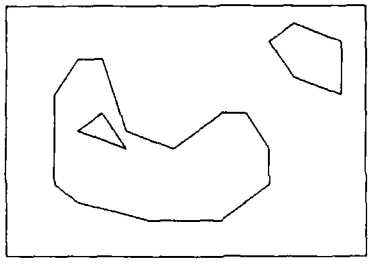

Fig. 5 .15 Niet convexe besluitgrens voor drielaags perceptron

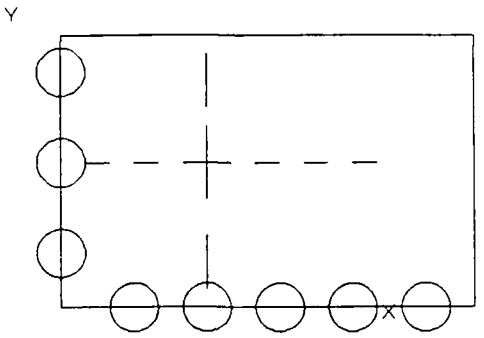

Fig. 5 .18 Gedistribueerd geheugen voor positiecodering. Het aanstaan van een rij en kolomcel betekent dat de X,Y-positie wordt onthouden.
De celverbindingen zijn niet getekend.

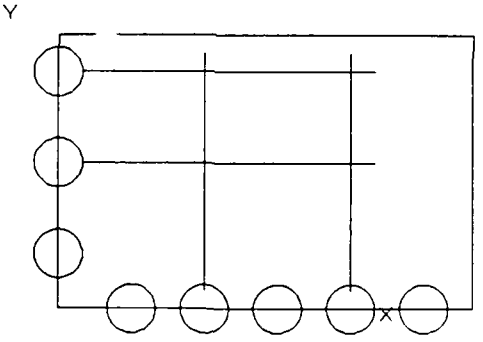

Fig.5.18b Het optreden van "ghost items". Bij het coderen van twee punten ontstaan eveneens twee "ghost" punten.

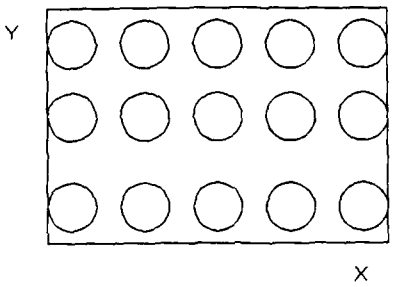

Fig.5.18c Het voorkomen van "ghost items" door voor elke positie een cel te reserveren.

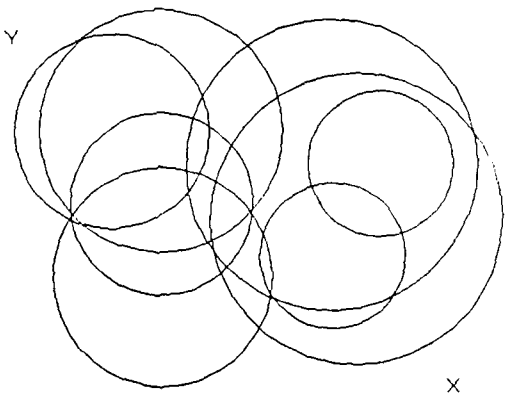

Fig. 5.19 Efficiënt gedistribueerd geheugen met behulp van overlappende celgebieden. Het aanstaan van een cel betekent dat binnen de "celstraat" een item aanwezig is.

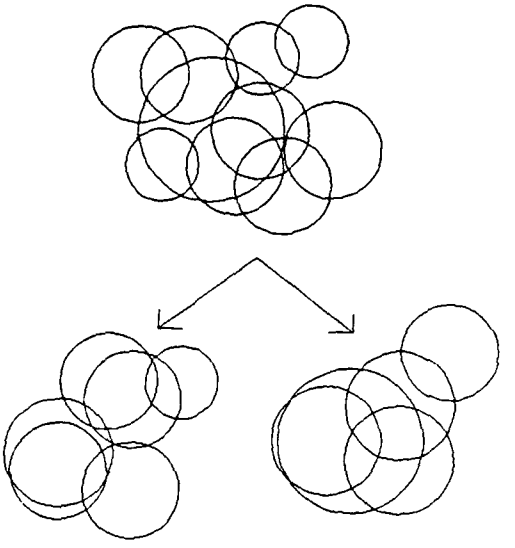

Fig. 5.21 Door het delen van het geheugen ontstaan twee "halve" geheugens met een verminderde resolutie over het gehele oppervlak.

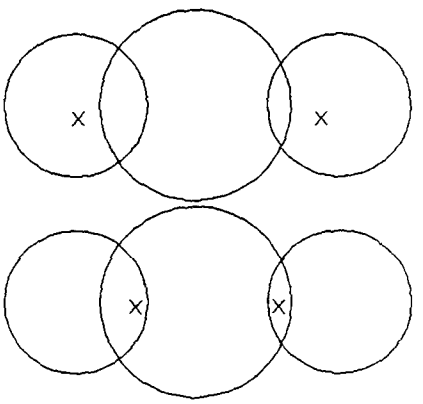

Fig. 5.22 Bij te dicht op elkaar zitten van items vloeien de gebieden in elkaar over.

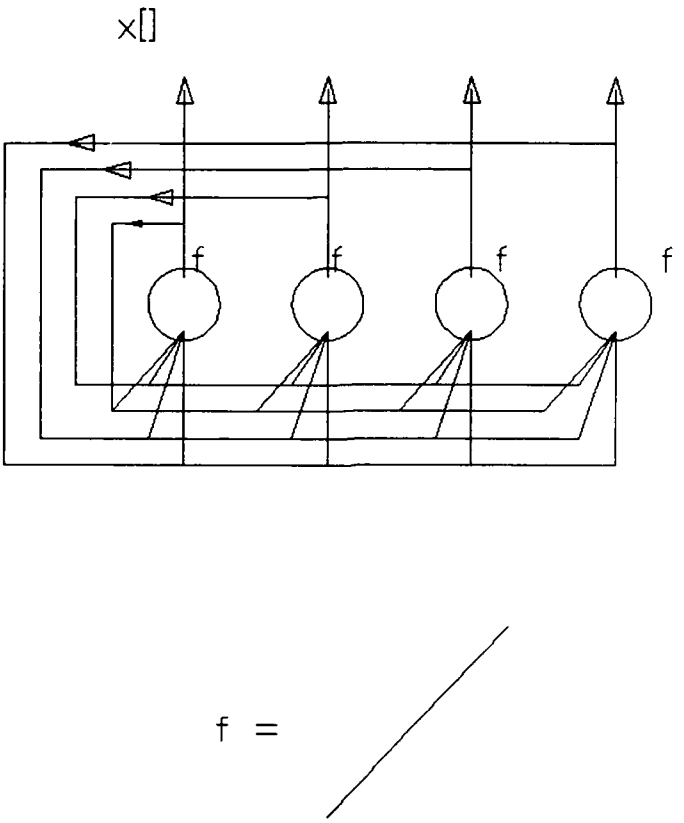

Fig. 5 .23 De lineaire associator

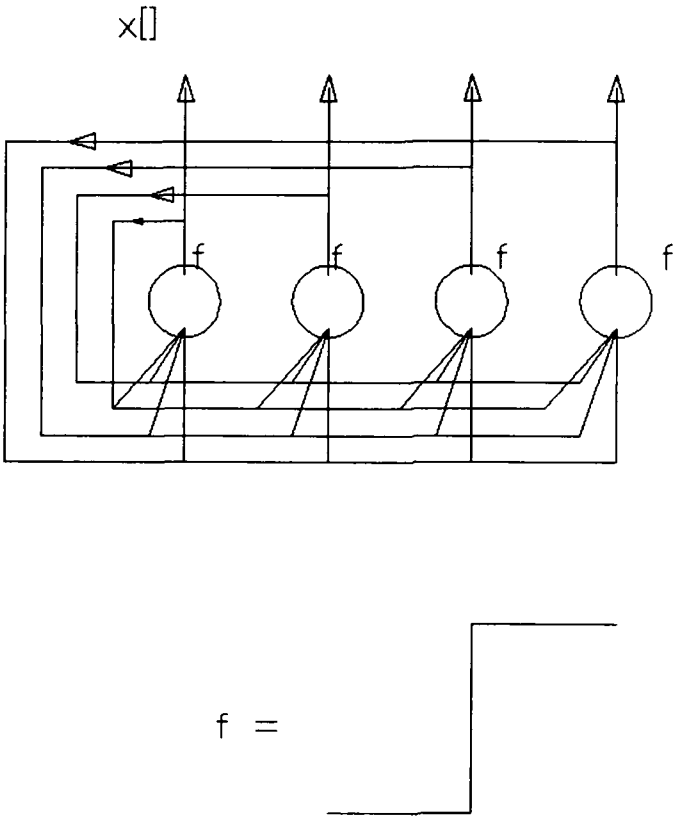

Fig. 5.24 Het Hopfield netwerk

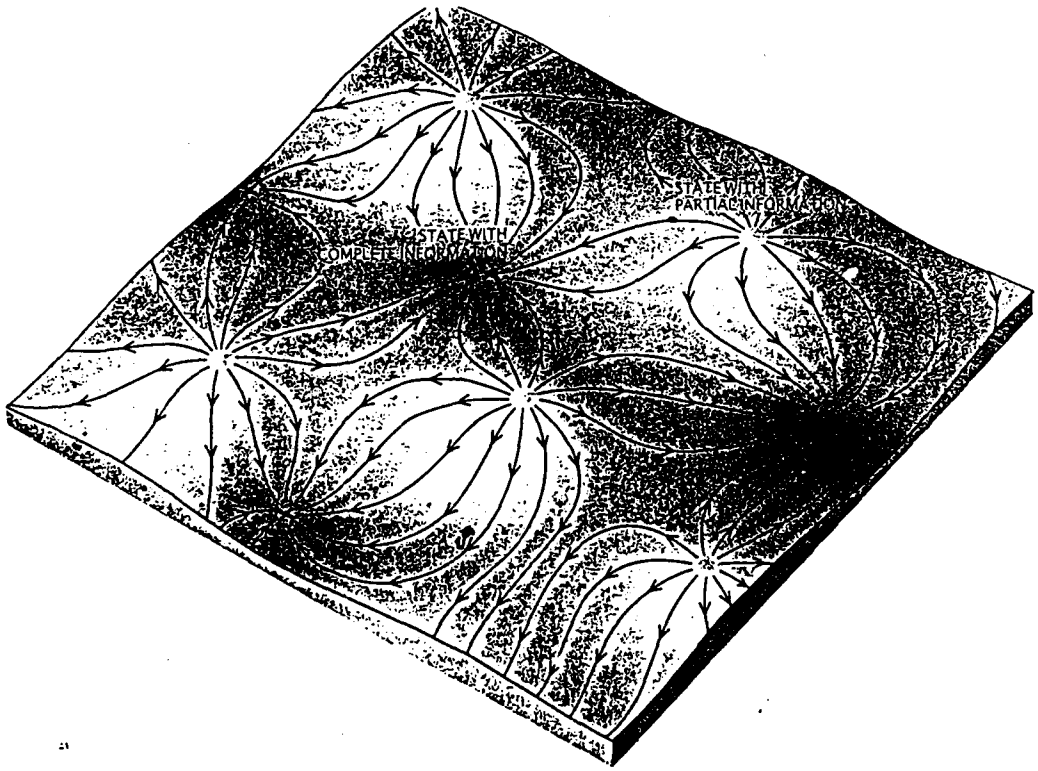

Fig. 5.25 Het zoeken naar een globaal minimum kan worden verduidelijkt door het energielandschap te zien als een "mat" met pieken en dalen. De lijnen geven mogelijke routes weer bij het stabiliseren. Daar het updaten een random karakter heeft is niet te zeggen welke route gevolgd zal worden.

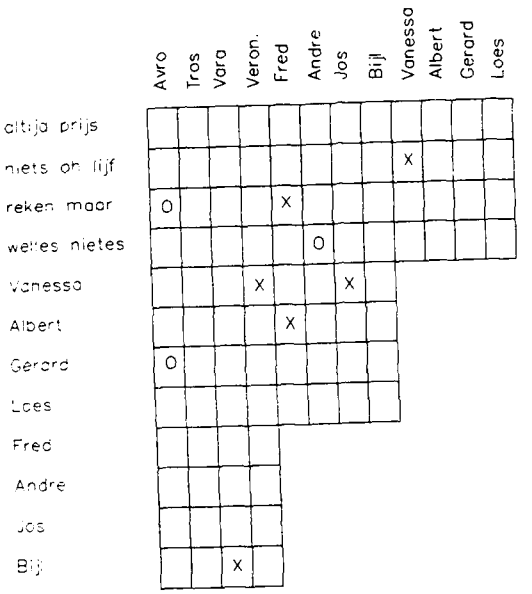

Fig. 5.26 De logicapuzzel

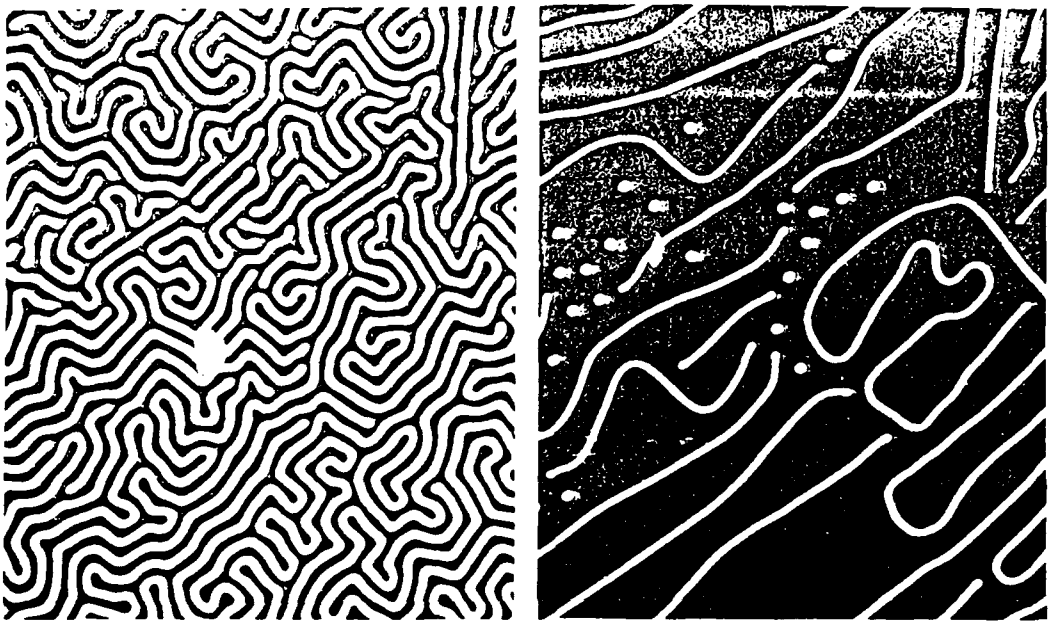

Fig. 5.27 "Frustratie" in magnetisch materiaal

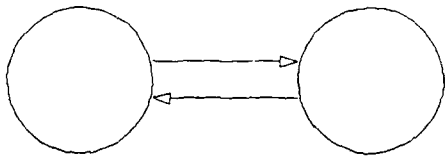

Fig. 5.28 Interactie tussen twee cellen in een Boltzmann machine. De getekende verbindingen zijn symmetrisch.

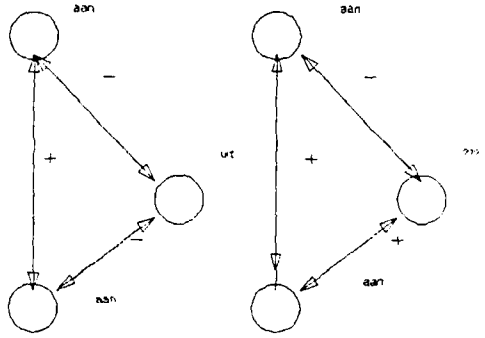

Fig. 5 .29 Twee netwerken zonder en met frustratie. De tekens geven de tekens van (symmetrische) gewichten weer.

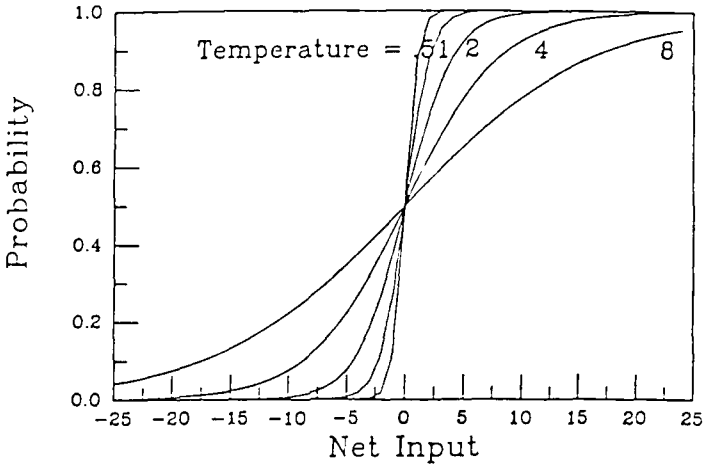

Fig. 5.30 Kansfunctie als functie van T. Bij hoge T neemt het random karakter toe, bij lage T benadert de functie het threshold model.

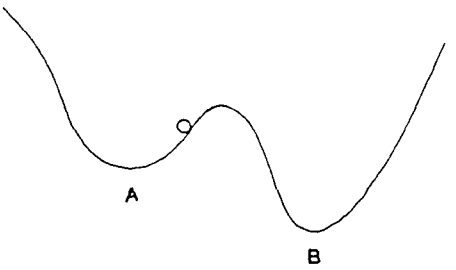

Fig. 5.31 Het annealings proces kan gevisualiseerd worden als het rollen van een knikker door een energie(harmonie)landschap. Door een schudproces uit te voeren (eerst hard, daarna al zachter) is er een grote kans dat de knikker in de diepste put blijft liggen (globaal minimum).

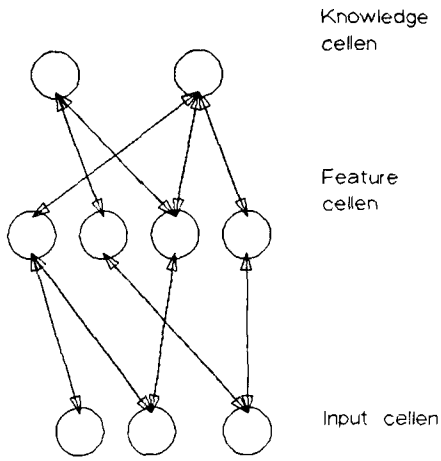

Fig. 5.32 Een harmonienetwerk bestaat uit drie lagen cellen: de input-, feature- en knowledge-cellen.

Fig. 5.34 Plaats 8 koninginnen op een schaakbord zonder dat zij elkaar "zien".

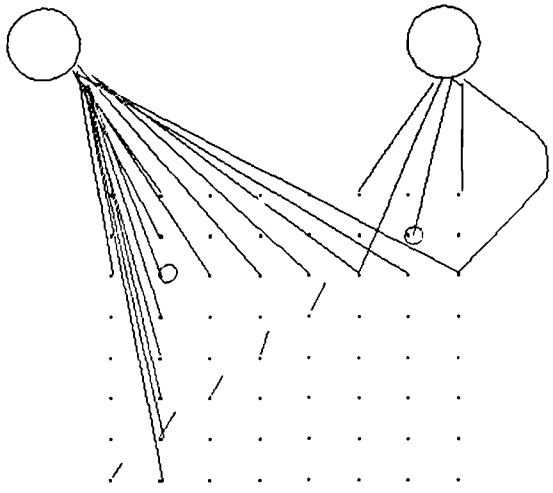

Fig. 5.35 De getekende cellen zijn verbonden met kruis- en diagonaal-feature-cellen. De cel in het midden (omcirkeld) is positief gelinkt met de knowledge-cel, de rest negatief. Een knowledge-cel gaat daardoor alleen aanstaan (T-0) indien slechts de centrale feature-cel aanstaat en geen van de cellen op de rest van een kruis of diagonaal.

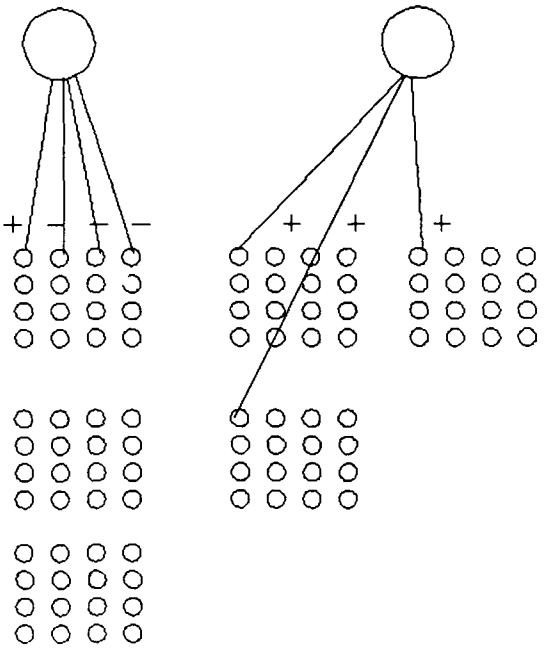

Fig. 5.36 Getekend zijn een exor-cel en een "driehoek-knowledge-cel. De exor-cel staat alleen aan als exact 1 cel in een rij (of kolom) actief is. De driehoekcel staat aan bij detectie van drie cellen AB, BC, AC actief.

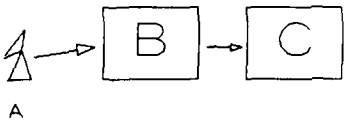

Fig. 6.1 Opdeling radar in functionele blokken
A = antenne
B = low level signaalbewerking
C = high level signaalbewerking

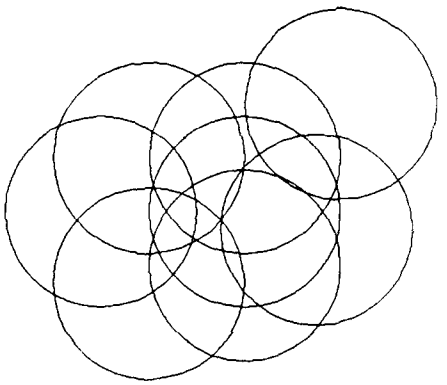

Fig. 6.2 Radarantenne volgens de gedistribueerde filosofie. Getekend zijn de gebieden waarbinnen een daarin geplaatst radarantennetje een signaal afgeeft.

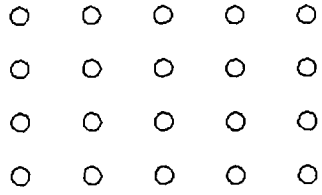

Fig. 6.3 Pixels op het radarscherm hebben elk een feature-cel.

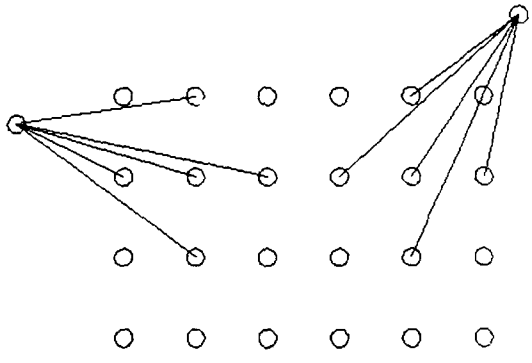

Fig. 6.4 Door het toevoegen van knowledge-cellen wordt een pixel alleen stabiel indien er buurcellen aanstaan.

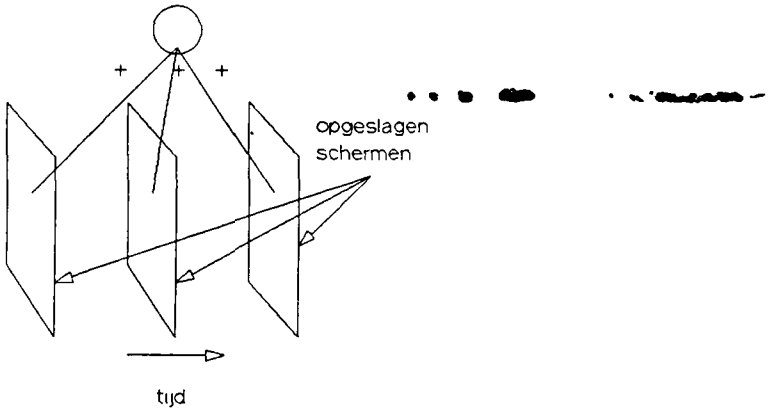

Fig. 6.5 Een pixel wordt alleen stabiel indien het in meerdere tijdframes aanwezig is.

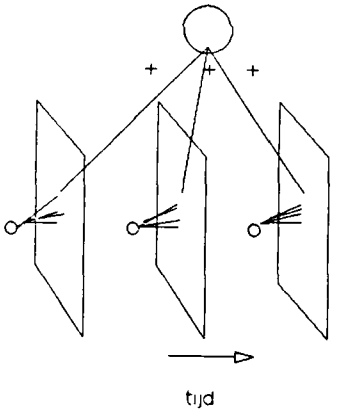

Fig. 6.6 Een combinatie van filtering in tijd en plaats

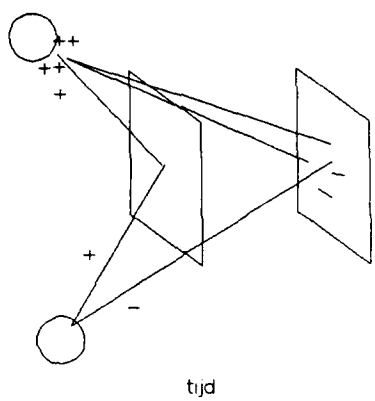

Fig. 6.7 Verschoven pixels zijn positief verbonden met de knowledge-cellen. Pixels op dezelfde plaats zijn positief en negatief verbonden. Hierdoor worden stilstaande objecten uitgefilterd.

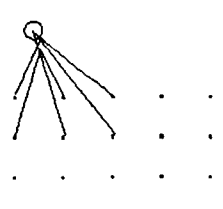

Fig. 6.9 Contourextractie door richtinggevoelige cellen. De bovenste cellen zijn - verbonden, de onderste +. Een horizontale contour wordt geëxtraheerd.

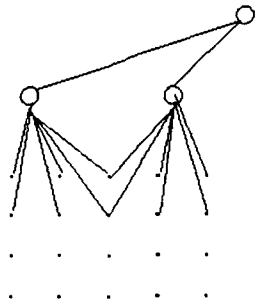

Fig. 6.10 Verbeterde contourextractie door toepassing van extra cellen die (horizontale) contouren doorverbinden. De extra cellen zijn positief verbonden met de contourcellen.

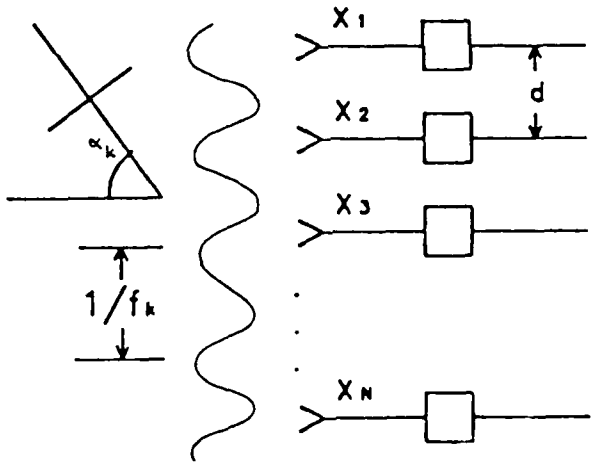

Fig. 6.11 Phased array antenne

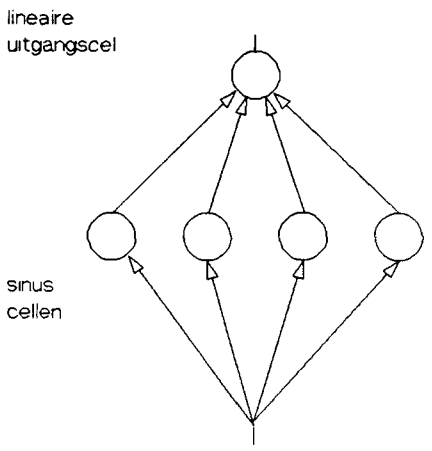

Fig. 6.12 De elementpositie x dient als invoer van het MLP.
Uitgangswaarde is de overeenkomstige elementamplitude.
De synapsen geven de objectfrequenties weer.

REPORT DOCUMENTATION PAGE (MOD-NL)

1. DEFENSE REPORT NUMBER (MOD-NL)	2. RECIPIENT'S ACCESSION NUMBER	3. PERFORMING ORGANIZATION REPORT NUMBER
TD89-3568		FEL-89-B65
4. PROJECT/TASK/WORK UNIT NO. 20154	5. CONTRACT NUMBER	6. REPORT DATE AUGUST 1989
7. NUMBER OF PAGES 155	8. NUMBER OF REFERENCES 86	9. TYPE OF REPORT AND DATES COVERED FINAL REPORT

10. TITLE AND SUBTITLE
NEURAL NETWORKS AND RADAR SYSTEMS

11. AUTHOR(S)
H.J. BORGERS

12. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
PHYSICS AND ELECTRONICS LABORATORY, PO BOX 96864, THE HAGUE, THE NETHERLANDS

13. SPONSORING/MONITORING AGENCY NAME(S)

14. SUPPLEMENTARY NOTES
IN DUTCH

15. ABSTRACT (MAXIMUM 200 WORDS, 1044 POSITIONS)
RECENTLY, THERE HAS BEEN AN EXPLOSIVE GROWTH IN INVESTIGATING NEURAL NETWORKS. THIS REPORT DESCRIBES SOME 'STATE OF THE ART' NETWORKS AND EVALUATES THE ADVANTAGES OF DISTRIBUTED SYSTEMS. ESPECIALLY, AN INVESTIGATION OF THE APPLICATIONS OF NEURAL NETWORK IN RADAR SYSTEMS IS MADE.

16. DESCRIPTORS	IDENTIFIERS
NEURAL NETWORKS	DISTRIBUTED SYSTEMS
PARALLEL PROCESSING	
RADAR	
SIMULATION	
SIGNAL PROCESSING	
COMPUTER ARCHITECTURE	

17a. SECURITY CLASSIFICATION (OF REPORT)	17b. SECURITY CLASSIFICATION (OF PAGE)	17c. SECURITY CLASSIFICATION (OF ABSTRACT)
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED

18. DISTRIBUTION/AVAILABILITY STATEMENT	17d. SECURITY CLASSIFICATION (OF TITLES)
UNLIMITED	UNCLASSIFIED