Nieuwsbrief van de Nederlandse Vereniging voor Theoretische Informatica

Mieke Bruné, Jan Willem Klop, Jan Rutten (redactie)*

February 21, 1997

Contents

*CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands. Email: mieke@cwi.nl.

1 Van de Redactie

Na een lange onderbreking is hier dan weer een Nieuwsbrief van de NVTI, de Nederlandse vereniging voor Theoretische Informatica. Zoals bekend is de NVTI ontstaan uit de voormalige WTI, via de VTI als korte tussenfase. Sinds de WTI van enkele jaren geleden is de situatie sterk veranderd, met name door het ontstaan van de onderzoekscholen (IPA, OZL, SIKS) waarover in dit eerste nummer in het kort informatie te vinden is. De rol van de NVTI is daarmee een geheel andere dan de rol van de vroegere WTI. Ook voor de Nieuwsbrief geldt dat: veel informatie die vroeger in de WTI Nieuwsbrief werd opgenomen, wordt nu door de onderzoekscholen verspreid in o.a. hun jaarverslagen en email-aankondigingen.

De huidige Nieuwsbrief is daarom erg afgeslankt, en bevat geen lijsten met publicaties en rapporten van de diverse onderzoeksgroepen, geen lijsten met bezoeken en bezoekers en geen personalia. Ook is de verschijningsfrequentie teruggebracht van (in de oude WTI-tijd) halfjaarlijks naar jaarlijks. Dit ook omdat een deel van de vroegere WTI Nieuwsbrief nu is overgenomen door onze web site (zie de Inleiding door de NVTI-voorzitter verderop in dit nummmer). Deze www-pagina's zijn nog onder constructie, maar bieden al nuttige informatie zoals de adreslijst (ook in dit nummer opgenomen).

Als een nieuw gebruik van de mogelijkheden die dit medium Nieuwsbrief ons biedt, hebben we een aantal bijdragen opgenomen van daartoe aangezochte auteurs; dit zijn korte inleidende beschrijvingen van onderwerpen die naar het oordeel van de schrijvers van belang zijn op ons gebied. De redactie houdt zich aanbevolen voor suggesties uwerzijds voor dergelijke bijdragen, of ook voor heel andere rubrieken.

In dit nummer zijn tevens de Statuten van de NVTI opgenomen. We danken met name Jan van Leeuwen voor zijn uitvoerig commentaar op eerste versies en Marlin van der Heijden (juridisch medewerkster van het CWI) voor de intensieve begeleiding van het tot stand komen van deze Statuten. Er is veel tijd geinvesteerd in het up-to-date maken van de ledenlijst, nu met email en http-adressen voor zover ter redactie bekend. We vragen iedere lezer zijn of haar adresgegevens te controleren en eventueel aan te vullen (bij mieke@cwi.nl).

De verschijning van de Nieuwsbrief zal in de komende jaren gelijke tred houden met de jaarlijkse theoriedagen. Dit jaar, de derde in successie, zal deze op 28 februari gehouden worden; het programma is in dit nummer opgenomen. Hierbij is het het Bestuur en de redactie een genoegen om dank te zeggen voor de financiele steun die wij genieten van onze sponsors: onze 'structurele' sponsor SION, die het voortbestaan van de Nieuwsbrief mogelijk maakt, inclusief het houden van onze theoriedagen; en de uitgeverij Elsevie Science B.V. voor hun sponsoring van de Theoriedag 1997.

De redactie,
Mieke Bruné
Jan Willem Klop
Jan Rutten

2 Samenstelling Bestuur

Prof.dr. J.C.M. Baeten (TUE)
Prof.dr. J.W. Klop (VUA/CWI) secretaris
Prof.dr. J.N. Kok (RUL)
Prof.dr. G. Rozenberg (RUL) voorzitter
Dr. J.J.M.M. Rutten (CWI)
Dr. J. Torenvliet (UvA)

3 Van de voorzitter

Geacht NVTI-lid,

De Nederlandse Vereniging voor Theoretische Informatica (NVTI) is bedoeld om de belangen te behartigen van alle geinteresseerden in de theoretische informatica en in het bijzonder om te dienen als een communicatieforum voor deze gemeenschap. De NVTI-nieuwsbrief, samen met de verenigingsdagen en de web site (http://www.cwi.nl/NVTI/), speelt hierbij een belangrijke rol. De nieuwsbrief bevat veel informatie die, naar onze mening, van belang is voor onze gemeenschap.

Uw commentaar op de nieuwsbrief stellen wij zeer op prijs. Voor een goed functioneren van de NVTI is het noodzakelijk dat er creatieve ideeen komen van de zijde van de leden. Met nadruk nodigen wij iedereen uit om mee te denken en ideeen en voorstellen te sturen naar mijzelf, J.W. Klop, of een ander bestuurslid.

Zoals u waarschijnlijk al weet, vindt op 28 februari de Theoriedag plaats. Voor deze dag is een interessant programma opgesteld (zoals in deze nieuwsbrief te lezen is). Ik hoop velen van u te ontmoeten op deze dag.

Tenslotte wil ik van de gelegenheid gebruik maken u een gelukkig en succesvol 1997 toe te wensen.

G. Rozenberg, voorzitter NVTI

4 Theoriedag 1997

Vrijdag 28 februari 1997, Jaarbeurs Congrescentrum Utrecht

Het is ons een genoegen u uit te nodigen tot het bijwonen van de Theoriedag 97 van de NVTI, de Nederlandse Vereniging voor Theoretische Informatica, die zich ten doel stelt de theoretische informatica te bevorderen en haar beoefening en toepassingen aan te moedigen. De Theoriedag 97 zal gehouden worden op vrijdag 28 februari 1997 in het Jaarbeurs Congrescentrum te Utrecht, en is een voortzetting van de reeks jaarlijkse bijeenkomsten van de NVTI die twee jaar geleden met de oprichtingsbijeenkomst begon. Evenals vorige jaren hebben wij een aantal prominente sprekers uit binnen- en buitenland bereid gevonden deze dag gestalte te geven door voordrachten over recente en belangrijke stromingen in de theoretische informatica. Naast deze wetenschappelijke inhoud heeft de dag ook een informatief gedeelte, in de vorm van een algemene vergadering waarin de meest relevante informatie over de NVTI gegeven zal worden, alsmede presentaties van de onderzoekscholen OZL, IPA en SIKS.

Lidmaatschap NVTI

Alle leden van de voormalige WTI (Werkgemeenschap Theoretische Informatica) zijn automatisch lid van de NVTI geworden. Aan het lidmaatschap zijn geen kosten verbonden; u krijgt de aankondigingen van de NVTI per email of anderszins toegestuurd. Was u geen lid van de WTI en wilt u lid van de NVTI worden: u kunt zich aanmelden bij het contactadres beneden (M. Bruné, CWI), met vermelding van de relevante gegevens, naam, voorletters, affiliatie indien van toepassing, correspondentieadres, email, telefoonnummer.

Programma
09.30 Ontvangst en koffie
10.00 - 10.10 Opening door G. Rozenberg
10.10 - 11.00 Voordracht N.G. de Bruijn
11.00 - 11.30 Koffie
11.30 - 12.20 Voordracht M.H. Overmars
12.20 - 13.00 Informatie over Onderzoekscholen OZL, IPA, SIKS
13.00 - 14.20 Lunch
14.20 - 14.50 Algemene Vergadering
14.50 - 15.40 Voordracht K.G. Larsen
15.40 - 16.00 Thee
16.00 - 16.50 Voordracht T. Bäck

Lunchdeelname
Het is mogelijk aan een georganiseerde lunch in het Jaarbeurs Congrescentrum deel te nemen; hiervoor is aanmelding verplicht. Dit kan per email of telefonisch bij Mieke Bruné (mieke@cwi.nl, 020-592 4249), tot een week tevoren (21 februari). De kosten kunnen ter plaatse voldaan worden; deze bedragen f 25,95. Wij wijzen erop dat in de onmiddellijke nabijheid van de vergaderzaal ook uitstekende lunchfaciliteiten gevonden kunnen worden, voor wie niet aan de georganiseerde lunch wenst deel te nemen.

Abstracts van voordrachten
A mathematical model for biological memory and consciousness
N.G. de Bruijn
Eindhoven University of Technology

An important feature of the model is the distribution of associative memory tasks over a large number of local agents (like neurons or local neural networks) without the use of any addressing system. The idea is that at any moment t only a relatively small subset A(t) of the set of all agents is awake. These agents store the associations that are around at that moment. At a later

moment t+p the intersection of the sets A(t) and A(t+p) can give answers to questions about what was recorded at time t. If the definition of the sets A(t) is generated by random processes it can be guaranteed that such intersections are almost always non-empty. In this way the capacity of the system can be seen as roughly proportional to the square root of the size of the system. Thinking of something like 10 billion agents we find that the capacity of the system can easily be about 10000 times bigger than the capacity of a single agent.

A vital part of what we call "consciousness" is related to the fact that the central processor of the brain has a very strong interaction with what was memorized during the last second. This can be understood by realizing that A(t) changes just slowly: if p is of the order of a second than the intersection of A(t) and A(t+p) can still be very big.

For the local agents the suggested model is the one of the "thinking soup", where associative information is recorded in DNA-like molecules.

References:

"A model for information processing in human memory and consciousness." Nieuw Archief Wiskunde (4) vol 12 (1994) 35-48.

" Can people think? ". Journal of Consciousness Studies, vol 3, No 5/6, (1996).

Fixturing and Manipulating Industrial Parts

by Mark H. Overmars

Utrecht University

In industrial applications parts need to be handled in various ways. There is a need to rotate parts into a required orientation, to hold them such that certain actions can be performed on them (like drilling a hole) and to assemble them into bigger parts. Robotic manipulation deals with the problem of performing such operations in an automated way. Of course it is possible to use robotic arms for such tasks but often simpler devices suffice. For example, many sensorless devices have been designed to orient parts, ranging from bowl feeders to vibrating plates. Designing such devices though is often some sort of black art that can only be done by a person with much experience. In this talk I consider two such problems and indicate how one can automatically design the devices, given a description of the geometry of the part.

The first problem is fixturing. Here the goal is to design a device that can hold the part in such a way that it constrains all possible motions of the part. This is important for performing actions on the part. I consider different fixture models and indicate under which circumstances parts can be fixtured with these models. I also present algorithms for computing such fixtures.

The second problem deals with orienting parts. Here we use a conveyor belt with fences. Parts move along the conveyor belt in arbitrary orientations. Once a part hits a fence it will align itself with the fence and slide along it. This restricts the number of possible orientations of the part. The goal is to design a sequence of fences such that, independent of the initial orientation, the part always ends up in the same final orientation. I show that for most parts such a sequence exists and give algorithms to compute optimal designs.

Compositional Model Checking

Kim G. Larsen

BRICS, Aalborg University, Denmark

It is well-known that the major problem in applying automatic verification techniques to analyze concurrent systems is the potential combinatorial explosion of the state space arising from parallel composition.

In this talk we present a compositional model checking technique, which in several experiments has proven successfully in conquering the above mentioned state-explosion problem. The technique allows components of a concurrent system to be gradually moved from the system into the specification (consituting a parallel equivalent of weakest preconditions), thus avoiding any global state-construction or even examination. Essential to the success of the technique is that intermediate specifications are kept small using efficient minimization heuristics.

The compositional technique seems to be universally applicable. In the talk we first present the technique for finite-state systems and recall the promising experimental results by Henrik Andersen

(LICS'95). We then show how the technique may be extended to real-timed systems (modeled as networks of timed automata) and report on obtained experimental results (CONCUR'95, RTSS'95, TAPSOFT'97). Finally, we indicate how to instantiate the technique to shared-variable models.

Selected Theoretical Aspects of Evolutionary Algorithms

Thomas Bäck
Department of Computer Science Leiden University and Center for Applied Systems Analysis (CASA), Informatik Centrum Dortmund

Evolutionary algorithms are probabilistic search and optimization methods gleaned from the model of organic evolution. Using a population of search points, stochastic variation operators imitating recombination and mutation, and a selection method that favors better search points for survival and propagation into the next generation, these algorithms yield surprisingly good solutions for a variety of optimization problems in industrial as well as research applications. The theoretical analysis of evolutionary algorithms, however, is still behind their practical successes.

This talk will provide a perspective on evolutionary algorithm theory that emphasizes the aspects of convergence velocity and convergence reliability, i.e., the speed of approaching a local optimum and the property of finding the global optimum, given infinite time, with probability one (the so-called global convergence of a stochastic optimization algorithm). The most recent results on convergence velocity of population-based evolution strategies with recombination are presented, and it is demonstrated how the convergence velocity analysis can be transferred from evolution strategies to genetic algorithms.

An important parameter control mechanism in evolutionary computation, the self-adaptation of strategy parameters, facilitates the simultaneous search on the object variable and strategy parameter levels. After giving a brief introduction to the self-adaptation principle, recent theoretical results concerning the robustness of this parameter adjustment method are summarized. The mechanism is also illustrated by empirical results on simple test cases where the dynamic behavior of the strategy parameter is theoretically predictable.

The talk will be concluded by indicating some perspectives for future research in evolutionary computation theory.

5 Mededelingen van de onderzoekscholen

Hieronder volgen korte beschrijvingen van de onderzoekscholen:

- Instituut voor Programmatuurkunde en Algoritmiek;
- Landelijke Onderzoekschool Logica;
- School voor Informatie- en KennisSystemen;

5.1 Instituut voor Programmatuurkunde en Algoritmiek (IPA), door: Jos Baeten

Het Instituut voor Programmatuurkunde en Algoritmiek verzorgt de opleiding van onderzoekers op het gebied van de programmatuurkunde en algoritmiek. Dit gebied behelst de bestudering en ontwikkeling van formalismen, methoden en technieken voor het ontwerp, de analyse en de constructie van programmatuur en programmatuurcomponenten. Onder programmatuur zijn begrepen alle vormen van formele beschrijving van het gedrag van informatie-verwerkende systemen op verschillende niveaus van abstractie, dus zowel hoog-niveau, niet-constructieve specificaties van systeemgedrag als concrete beschrijvingen van algoritmen en executeerbare code.

In 1996 is IPA tot volle ontplooiing gekomen. In het begin van dat jaar zetten de laatste van de deelnemende universiteiten hun handtekening, zodat de school officieel opgericht was, met als deelnemers TUE, UU, KUN, UT, RUL en VU. De RUG neemt deel als geassocieerd lid en er is een samenwerkingsovereenkomst met het CWI.

Op 15 en 16 april 1996 waren in Veldhoven de IPA lentedagen, met als thema "Different faces of algorithmics". Van 16 tot en met 20 september waren de herfstdagen in Noordwijk aan Zee, met als thema "Optimization and coordination". Van 25 tot en met 29 november organiseerde IPA, in samenwerking met de Deense onderzoekschool BRICS (Basic Research In Computer Science) en de Finse onderzoekschool TUCS (TUrku centre for Computer Science), in Veldhoven de School on Embedded Systems. Dit was een hoogst succesvol evenement, met meer dan 150 deelnemers uit meer dan 14 landen, gesponsord door de EU. Een van de sprekers was de kersverse Turing Award winnaar Amir Pnueli. Tijdens de week richtten IPA, BRICS en TUCS samen het European Educational Forum op, dat wil uitgroeien tot een Europese onderzoekschool.

Zowel tijdens de herfstdagen als tijdens de School on Embedded Systems was er een speciale Feedbacks dag, gericht op het bevorderen van interactie tussen wetenschap en bedrijfsleven. Op 20 september was het thema Legacy Systems, op 27 november Embedded Systems. Beide dagen werden goed bezocht, met meer dan 30 bedrijven vertegenwoordigd.

In het onderzoek wil IPA zich met name profileren op een beperkt aantal speerpunten. De zes speerpunten zijn: Testen, Renovatie, Embedded systems, Natural computation, Software architectuur en Algoritmen voor planning en ontwerp. Voor deze speerpunten willen we ook extra middelen inzetten.

Eind 1996 heeft IPA een erkenningsaanvraag ingediend bij de KNAW. De voorbereiding van deze aanvraag was de aanzet tot enige herbezinning over afbakening met andere onderzoekscholen. Tussen de verschillende informatica(-gerelateerde) onderzoekscholen bestaan inmiddels goede samenwerkingsovereenkomsten. Deze twee ontwikkelingen samen hebben enkele onderzoekleiders ertoe gebracht, niet meer lid te willen zijn van meerdere onderzoekscholen. Zo concentreert Overmars (UU) zijn inbreng in het Helmholz Instituut, Meyer (UU) in SIKS en Koster (KUN) in NICI. Aan de andere kant groeit IPA ook: zo zal RUG volwaardig lid worden, met inbreng van onderzoeksgroepen van Hesselink, Spaanenburg en Udding, en vinden besprekingen plaats met de UvA, over de onderzoeksgroepen van Bergstra, Klint, Vitányi en Boasson.

We kunnen al wat activiteiten noemen in 1997. In 1997 vinden de lentedagen plaats op 10 en 11 maart in Mierlo. Op 12 maart is er in Mierlo weer een Feedbacks dag. Verder gaan in 1997 de drie basiscursussen van start. Dit zijn intensieve cursussen, ongeveer een week elk, die verplicht zijn voor alle nieuwe promovendi van de onderzoekschool. Er is een cursus op het gebied Algoritmiek en Complexiteit, een over Formele Methoden en een over Software Technologie. Van 11 tot 22

augustus is er in Veldhoven een eerste zomerschool in een serie over "Foundations of Computer Science", met als thema "Computational and syntactical methods", en docenten o.a. Barendregt, Baeten, Klop, Nielsen, Winskel en Tucker. Dan is er van 25 tot 29 augustus in Turku (Finland) een school over Natural Computing, georganiseerd door het European Educational Forum.

Verder zijn we van plan in 1997 in het bijzonder aandacht te besteden aan de relatie met het bedrijfsleven. We willen enkele projecten opzetten als contractresearch, en een aantal bedrijven als institutionele sponsor verkrijgen.

Namens het management team van IPA wens ik eenieder het beste in 1997.
Jos Baeten, wetenschappelijk directeur

Lijst van onderzoeksleiders:
KUN: Meijer, Plasmeijer, Vaandrager, Barendregt
RUL: Rozenberg, Kok
TUE: Baeten, Backhouse, Aarts, Hilbers, Rem, Hammer, Rooda
UT: Brinksma, Nijholt
UU: Swierstra, Meertens, Van Leeuwen
VUA: De Bakker, Klop
RUG: Hesselink, Spaanenburg, Udding
CWI: Vitányi, Groote, De Bakker, Klop, (Klint, Apt)
(UvA: Klint, Bergstra, Apt).

5.2 Landelijke Onderzoekschool Logica (OzsL), door: Jan van Eijck, Erik-Jan van der Linden

De Landelijke Onderzoekschool Logica (OzsL) is in 1992 erkend door de KNAW in de eerste groep van 19 scholen. De school startte in 1992 haar werk als nationale organisatie voor onderzoek en opleiding van onderzoekers op het gebied van de logica en haar toepassingen in wiskunde, taalkunde en informatica.

Bijna vijf jaar later, aan de vooravond van de tweede vijfjaarsperiode van de school, kan een aantal ontwikkelingen worden gesignaleerd.

Logica en informatiewetenschap

Logica was en is voor onderzoekers in OzsL vooral een *methodologisch* prespectief op onderzoek. In de afgelopen vijf jaar is daarbij gekomen dat onderzoek in de Nederlandse logica-gemeenschap *inhoudelijk* lijkt te convergeren op de studie van symbolische informatiestructuren en de computationele processen waarmee die gemanipuleerd worden. Integratie met niet-symbolische benaderingen is daarbij van belang. Het onderzoek wordt hiermee meer probleem-georienteerd.

Deze beweging is gepaard gegaan met een verbreding van de interesse in toepassingsgebieden binnen de school. Het duidelijkst wordt dit zichtbaar aan de nieuwe associaties die de school is aangegaan voor de tweede vijfjaarsperiode met onderzoekers op het gebied van beeldbewerking, geografische informatiesystemen, robotica en kennissystemen.

Dat *computationele* aspecten van de logica van groot belang zijn, mag blijken uit het initiatief om een *computational logic lab*, op te zetten aan een van de instituten van de school (ILLC, UvA). Dit lab is een van de speerpunten in het onderzoek dat wordt gestart door Johan van Benthem in het kader van de aan hem toegekende SPINOZA-premie.

Onderzoekersopleiding

De arbeidsmarkt voor promovendi in theoretische vakken ligt steeds minder op de universiteit, en steeds vaker daarbuiten. Waar promovendi ook zullen landen, er worden altijd hoge eisen gesteld aan hun concurrerend vermogen. Naast promotie-onderzoek van hoge kwaliteit is een brede kennis van het vak die verder strekt dan het onderwerp van de dissertatie van groot belang. Verder spelen communicatief vermogen en gemak om zich te bewegen in een internationale gemeenschap een steeds grotere rol.

De school heeft in de afgelopen jaren in deze behoeften voorzien met een cursus-programma, masterclasses, presentatie workshops, een conferentie voor en door promovendi, en bezoek aan een internationale zomerschool. Voorbeelden van cursussen en masterclasses die interessant zijn voor theoretische informatici worden hieronder opgesomd.

- *Dynamic logic* (coordinators Jan van Eijck, Martin Stokhof, Erik-Jan van der Linden)
- *Proofs and types* (coordinators Erik Barendsen, Herman Geuvers and Dirk Roorda)
- *Basic course Proof Theory* (A.S. Troelstra)
- *Masterclass Foundations of Computer Science* (Gurevich; with local co-ordination by van Emde Boas and van Leeuwen).
- *Basic Course Logic Programming and Constraint Satisfaction* (Apt)
- *Basic Course Non-monotonic Reasoning* (van der Hoek and Witteveen)
- *Masterclass Inductive inference* (Smith)
- *Advanced Course Term Rewriting* (Klop, de Vrijer)
- *Basic Course Kolmogorov complexity* (Vitányi)

De Onderzoekschool waakt erover dat haar onderwijs toegankelijk is voor promovendi uit andere scholen en omgekeerd. Verkokering in de Nederlandse promotie-opleiding in de vorm van lesgelden voor promovendi buiten de eigen school, of gesloten evenementen, zijn uiteindelijk in het nadeel van de kwaliteit van de opleiding van alle Nederlandse promovendi. Grensverkeer tussen scholen moet voor promovendi altijd mogelijk zijn.

Andere scholen

Toegankelijkheid van onderwijs veronderstelt samenwerking met andere scholen. Onderzoekssamenwerking strekt zich met gemak uit over grenzen tussen de inmiddels meer dan 100 scholen in Nederland. Dit is vooral voor *interdisciplair* onderzoek van groot belang. OzsL moedigt participatie van stafleden van de school in andere scholen dan ook aan, en probeert de samenwerking met andere scholen zo goed mogelijk te regelen. Zeker ook voor vakgebieden als de theoretische informatica is openheid in het nederlandse scholensysteem van groot belang.

5.3 School voor Informatie- en KennisSystemen (SIKS), door: Koen Versmissen

We schrijven begin 1993 als een groep Nederlandse onderzoekers werkzaam in de kunstmatige intelligentie, databases en software engingeering, zich afzondert in het Vlaamse. Reden voor dit uitstapje: een workshop om erachter te komen of er voldoende onderlinge samenhang is om een draagvlak te bieden voor een gezamenlijk op te richten onderzoekschool. De conclusie: een volmondig 'jazeker!'. De jaren daarna worden er voor de wetenschappers diverse themadagen georganiseerd, terwijl achter de schermen druk wordt gewerkt aan de formele oprichting. Begin 1996 is het dan zover: de *School voor Informatie- en KennisSystemen*, kortweg SIKS, wordt formeel opgericht.

De centrale onderzoeksvraag op het gebied van informatie- en kennissystemen kan als volgt geformuleerd worden:

> Hoe kunnen grote hoeveelheden informatie op een zinvolle en efficiënte manier worden opgeslagen, gestructureerd en bewerkt?

Deze vraag kan worden gesplitst in drie deelvragen:

- hoe wordt de werkelijkheid gemodelleerd?

- hoe wordt een beschrijving van de werkelijkheid gerealiseerd m.b.v. een computerprogramma?
- wat is er nodig om specifieke toepassingsterreinen adequaat te ondersteunen?

Binnen SIKS worden deze vragen onderzocht aan de hand van de volgende vier centrale onderzoekthema's:

- analyseren van applicatiegebieden
- modelleren en specificeren
- realiseren
- heuristische technieken

Eind januari komen de onderzoekleiders van SIKS twee dagen bijeen om zich te bezinnen op het hierboven samengevatte onderzoeksprogramma. De bedoeling hiervan is om recente ontwikkelingen te vertalen in het bijstellen van de onderzoeksfocus. Op deze manier hoopt SIKS zich optimaal voor te bereiden voor de aanvraag van erkenning door de KNAW, later dit jaar.

In SIKS wordt deelgenomen door de volgende groepen (met tussen haakjes de onderzoeksleiders):

- VU, Fac. Wiskunde en Informatica, Vakgroep Informatica
  - Kunstmatige Intelligentie (prof.dr. J. Treur)
  - Informatiesystemen (prof.dr. R. van de Riet)
  - Software Engineering (prof.dr. J.C. van Vliet)
- UvA, Fac. Psychologie, Vakgroep Sociaal-Wetenschappelijke Informatica (prof.dr. B.J. Wielinga)
- TUD, Fac. Technische Wiskunde en Informatica, Vakgroep Informatie Systemen (prof.dr. J.L.G. Dietz)
- TUE, Fac. Wiskunde en Informatica, Sectie Informatiesystemen (prof.dr. P.M.E. de Bra)
- RUL, Fac. Wiskunde en Natuurwetenschappen, Software Engineering and Information Systems (prof.dr. G. Engels)
- UM, Fac. Algemene Wetenschappen, Vakgroep Informatica (prof.dr. H.J. van den Herik)
- UT, Fac. Informatica, Groep Informatiesystemen
  - Databases for Object-Oriented and Logical Languages (prof.dr. P.M.G. Apers)
  - Knowledge Based Systems (prof.dr. N.J.I. Mars)
  - Design Methodology Research Group (dr. J.N. Brinkkemper)
  - Intelligent Information Systems for Science and Engineering (prof.dr. H.A. Akkermans)
- UU, Fac. Wiskunde en Informatica, Vakgroep Informatica, Groep Intelligente Systemen (prof.dr. J.-J.Ch. Meyer)

Daarnaast zijn er samenwerkingsovereenkomsten[1] met het CWI, de UvA (Vakgroep Rechtsinformatica), de EUR (Fac. Economische Wetenschappen en EURIDIS) en de KUB (CentER en de Fac. Wijsbegeerte).

De penvoerder van SIKS is de Vrije Universiteit, die in de persoon van prof. van de Riet ook de bestuursvoorzitter levert. De wetenschappelijk directeur van SIKS is prof. Meyer van de Universiteit Utrecht. In de school wordt deelgenomen door een kleine 70 senior onderzoekers en ongeveer 40 promovendi.

Het afgelopen jaar zijn door SIKS de volgende activiteiten georganiseerd:

[1] nog niet alle formeel bekrachtigd

6 Wetenschappelijke bijdragen

In deze rubriek willen we, te beginnen met dit nummer, een aantal prominente onderzoekers uit de NVTI-gemeenschap uitnodigen een korte inleidende bijdrage te schrijven over een onderwerp dat zij van belang achten in het huidige onderzoeksveld. Voor dit eerste nummer zijn H.P. Barendregt, H.J. Hoogenboom, J.F. Groote en P. Vitányi bereid gevonden een dergelijke bijdrage te verzorgen. De bijdrage van H.P. Barendregt is eerder verschenen in het W&N bulletin van de KUN.

- een driedaagse AIO-cursus *Heuristisch zoeken en datamining*
- een themadag over *object-oriëntatie*

Bovendien was SIKS mede-organisator van de VOIS[2] AIO-dagen en NAIC[3] '96.

Naast de al genoemde bezinningsdagen staan voor dit jaar nog een aantal andere activiteiten van SIKS op de rol. Van 21 tot 23 april is er een AIO-cursus *Redeneervormen voor AI*. Ook in oktober komt er een AIO-cursus, met als onderwerp waarschijnlijk *Modelleren*. Daarnaast zal geprobeerd worden om een buitenlandse coryfee te strikken voor het geven van een master class van enkele dagen. Ook komen er weer twee themadagen, de eerste in mei te Amsterdam, de andere in november te Delft. De thema's hiervan moeten nog vastgesteld worden.

Veel informatie over SIKS is te vinden op onze WWW pagina's, te vinden onder de URL http://www.cs.ruu.nl/siks/home.html. Verder kunt u zich met vragen en opmerkingen richten aan: Koen Versmissen, coördinator SIKS, Postbus 80.089, 3508 TB Utrecht. Tel: 030-2534083. E-mail: koen@cs.ruu.nl.

[2] Vereniging voor Onderzoek naar InformatieSystemen
[3] Nederlandse AI Conferentie

6.1 Computer Wiskunde

H.P. Barendregt Katholieke Universiteit, Nijmegen

Computers en wiskunde

In de jaren 50 werd er nog serieus gedacht dat er in een land als Nederland slechts behoefte zou zijn aan een paar computers. Tien jaar later kwamen er echter langzaam maar zeker steeds meer van die apparaten. Redelijk veel mensen wisten dat computers gebruikt werden voor het doen van berekeningen en het bijhouden van een administratie. Aan wiskundigen werd dan wel gevraagd: 'Jij gebruikt zeker een computer?' De meeste wiskundigen haastten zich te antwoorden dat dat niet zo was: 'Computers maken alleen numerieke berekeningen. Zelfs een getal-theoreticus hoeft nauwelijks met computers te werken.' We zullen zien dat er in de loop van de tijd veel is veranderd.

In het begin van de jaren 60 werden de eerste programma's ontwikkeld voor het uitvoeren van symbolische berekeningen. Een getal als $\sqrt{2}$ kan met behulp van dergelijke programma's exact op een computer gerepresenteerd worden. Dit getal wordt gewoon als symbool gerepresenteerd en hiermee wordt dan 'gerekend'. Deze programma's waren in de jaren 80 zo ver geperfectioneerd dat ze als commerciële pakketten 'Computer Algebra' verkocht werden. Bekende systemen zijn Mathematica, Maple en Reduce. Deze systemen kunnen de meest uiteenlopende vormen van symbolische (en numerieke) berekeningen uitvoeren. Bijvoorbeeld het werken met polynomen van meerdere veranderlijken, zoals het ontbinden in factoren. Het symbolisch oplossen van lineaire vergelijkingen behoort verder tot de mogelijkheden. Ook berekeningen komend uit de analyse kunnen gedaan worden, bijvoorbeeld het vinden van een primitieve functie en verder ook meer ingewikkelde differentiaalvergelijkingen. De mogelijkheden worden slechts beperkt door het al dan niet voorhanden zijn van een efficiënt algoritme voor het doen van de symbolische of numerieke berekeningen. Als een mens het kan uitrekenen, dan ook de computer. De hulpmiddelen zijn nu zo krachtig en gebruikersvriendelijk, dat ze gebruikt worden door veel onderzoekers werkend in de exacte wetenschappen. Ook in de economie en sociale wetenschappen worden systemen voor Computer Algebra gebruikt. Zelfs zuivere wiskundigen, die uit zijn op het ontwikkelen van nieuwe theorie, maken gebruik van deze systemen door deze op de juiste manier in te zetten (onder andere door er experimenten mee te doen).

Onafhankelijk van deze ontwikkeling is er sinds het einde van de jaren 60 gewerkt aan formele talen voor de exacte representatie van willekeurige wiskundige uitspraken en begrippen. In tegenstelling tot de systemen voor Computer Algebra kunnen er in deze talen ook niet-berekenbare begrippen weergegeven worden. In de wiskunde ontstaat niet-berekenbaarheid door de zogenaamde kwantoren $\forall x{\in}\mathbb{N}.Px$ ('voor alle natuurlijke getallen x geldt Px') en $\exists x{\in}\mathbb{N}.Px$ ('er is een x natuurlijk getal zodat Px'), waarbij $\mathbb{N} = \{0, 1, 2, 3, \ldots\}$ de verzameling natuurlijke getallen is. Omdat deze verzameling oneindig is, kan een computer niet nagaan of alle elementen een bepaalde eigenschap hebben. Een

wiskundige kan echter door middel van *bewijzen* wel met niet-berekenbare begrippen omgaan. Deze bewijzen zullen zodanig weergegeven moeten worden, dat een computer deze kan gebruiken. Hiervoor is het nodig dat het niveau van detail van deze bewijzen voldoende groot is dat het door de machine gevolgd kan worden. Indien dit niveau bereikt is, dan spreekt men wel van bewijs-objecten. Het is niet zo dat een computer deze bewijs-objecten zelf kan vinden. Daarvoor heeft die de samenwerking met een mens nodig. In een interactieve sessie kunnen bewijzen ontwikkeld worden, waarbij de mens de goede ideeën geeft en de computer helpt met het benodigde ambachtelijke vakmanschap. In de jaren 80 zijn er in de USA en Europa verschillende prototype systemen voor bewijs-ontwikkeling en -verificatie geconstrueerd. De reden dat dit nog slechts prototypen zijn is gelegen in het feit dat het genereren van een bewijs-object op dit moment nog veel moeite kost, meer dan een wiskundige bereid is te doen.

Logica en type theorie

De Griekse filosoof Aristoteles heeft een duidelijke invloed gehad op de huidige technologie van de Computer Wiskunde. In zijn werken worden de onderwerpen de axiomatische methode, het zoeken naar de logica en bewijs-verificatie versus -constructie behandeld. Op al deze fronten heeft Aristoteles belangrijk werk geïnitieerd.

De axiomatische methode

Volgens Aristoteles bestaat een wiskundige theorie uit begrippen (zoals bijvoorbeeld in de meetkunde punten, lijnen en incidentie—het liggen van een punt op een lijn) en uit stellingen over die begrippen. Begrippen vormt men door middel van definities uit andere begrippen. Omdat men ergens moet beginnen zijn er primitieve begrippen, die niet nader gedefinieerd worden. Stellingen bewijst men met behulp van redeneringen uit andere stellingen. Weer moet men ergens beginnen: bij de primitieve stellingen ook wel axioma's genoemd. Deze behoeven geen bewijs. Niet lang nadat Aristoteles deze methode beschreven had, heeft Euclides zijn beroemde boeken 'De Elementen' geheel in de stijl van de axiomatische methode geschreven.

De axiomatische methode is zeer krachtig, omdat men alleen uitspraken bewijst op grond van wat er gegeven is aan primitieve begrippen en axioma's. Een groot gedeelte van de wiskunde kan zonder meer axiomatisch opgebouwd worden. Voor enkele gebieden, waaronder de combinatoriek, is dat moeilijker.

Het zoeken naar de logica

Om uit gegeven geldige uitspraken nieuwe af te leiden gebruiken wij redeneringen. De juiste redeneringen waren volgens Plato aan ons bekend omdat wij in contact staan met de zuivere rede. Aristoteles vroeg zich echter af, of alle mogelijke juiste redeneerstappen in kaart konden worden gebracht. Met andere woorden, of het redeneren gevangen kon worden in een eindig aantal regels. Plato vond deze vraagstelling onjuist. Volgens hem was ons contact met de

zuivere rede duidelijk genoeg. Ondanks deze waarschuwing maakte Aristoteles een indrukwekkend begin met het zoeken naar de logica, door voor een fragment, de zogenaamde monadische logica, de juiste regels in kaart te brengen. Hierin worden alleen predicaten met één argument beschreven, zoals bijvoorbeeld $E(n) =_{\text{Def}} (n \text{ is even})$; binaire predicaten, zoals $G(n,m) =_{\text{Def}} (n > m)$, komen in de monadische logica niet voor. Met name gaf Aristoteles regels voor de implicatie $A \rightarrow B$ (als A, dan B) en de universele kwantificatie (voor alle x geldt Px). Het duurde meer dan 2300 jaar totdat het plan van Aristoteles voltooid was. Het logische systeem werd gevonden door Frege in 1879 en Gödel bewees in 1930 dat het voldoende sterk was voor alle benodige redeneerstappen voor het ontwikkelen van axiomatische wiskunde.

Verificatie versus constructie van bewijzen

Aristoteles heeft opgemerkt dat het verifieren van de correctheid van een bewijs in principe niet moeilijk is: men volgt het bewijs stap voor stap en kijkt of het klopt. Aan de andere kant wees hij erop dat het vinden van een bewijs wel moeilijk is: bij een gegeven stelling (die weliswaar bewezen is, maar waarbij het bewijs bijvoorbeeld verloren is gegaan) zijn wij niet altijd in staat binnen redelijke tijd een bewijs terug te vinden. (Indien wij de beschikking hebben over willekeurig veel tijd, dan is dit in principe wél mogelijk. Men kan namelijk alle mogelijke bewijzen genereren en kijken of de gevraagde stelling er toevallig uit komt. Op den duur moet dit het geval zijn, want we weten dat er is een bewijs voor de stelling is. Dit is echter onrealistisch, vergelijkbaar met het wachten op een aap aan een tekstverwerker, totdat die op een gegeven moment per ongeluk een gedicht van Paul van Ostaayen zal intikken.) Deze observeringen zijn juist en hebben in onze eeuw tot twee belangrijke corollaria geleid. N.G. de Bruijn heeft rond 1968 een formeel systeem (Automath) ontworpen, waarin bewijzen zodanig formaliseerdbaar zijn, dat deze op efficiënte wijze machinaal geverifieerd kunnen worden en Turing heeft in 1936 bewezen dat voor een willekeurig gegeven uitspraak niet automatisch bepaald kan worden of deze al dan niet volgt uit een gegeven verzameling axiomas.

Brouwer

Ondertussen kwam er uit onverwachte hoek kritiek op de logica van Aristoteles. De jonge Nederlandse wiskundige L. E. J. Brouwer durfde in zijn proefschrift van 1907 de bewering te poneren dat de principes van de logica van Aristoteles onbetrouwbaar waren. Met name voor het principe van het uitgesloten derde

$$A \vee \neg A$$

(A of niet A) was dat volgens Brouwer het geval. Vrijwel niemand wilde hem geloven. Brouwer drukte zulke feiten uit op een voor hem typerende wijze: "Medestanders hebben onderstaande stellingen nog weinig gevonden." Pas nadat Brouwer een drietal jaren hard had gewerkt op een heel ander gebied van de wiskunde en de oplossing van een aantal belangrijke problemen had gevonden, begon men hem ook op het gebied van de grondslagen *au sérieux* te nemen. De

analyse van Brouwer kwam er hierop neer, dat het principe van het uitgesloten derde tot gevolg heeft dat men een uitspraak als

$$\exists n{\in}\mathbb{N}.P(n),$$

kan bewijzen zonder dat een 'getuige' n gevonden kan worden, waarvoor $P(n)$ bewijsbaar is.

.....

Beschouw een uitspraak A, waarvan het (nog) niet bekend is of deze geldt of niet. Laat

$$P(n) \;=\; [(n = 0 \;\&\; A) \vee (n = 1 \;\&\; \neg A)],$$

dat wil zeggen (n=0 en A of n=1 en niet A). Dan kunnen we met het principe van het uitgesloten derde bewijzen dat geldt

$$\exists n{\in}\mathbb{N}.P(n).$$

Immers, als A geldt, dan kunnen we nemen $n = 0$ en als $\neg A$ geldt, dan nemen we $n = 1$. Volgens de logica van Aristoteles geldt $A \vee \neg A$, dus er is in alle gevallen een n zodat $P(n)$ geldt. Maar aangezien we echter niet weten welk van de twee mogelijkheden A of $\neg A$ geldt, kunnen we geen getal n vinden zodat we kunnen bewijzen $P(n)$; want daartoe moeten we eerst A bewijzen of weerleggen.

.....

Voorbeeld van een bewijsbare uitspraak $\exists n{\in}\mathbb{N}.P(n)$ zonder 'getuige' n, waarvoor $P(n)$ bewijsbaar is.

In feite heeft Brouwer een iets andere interpretatie aan de logische operatoren (zoals &, $\vee$, $\rightarrow$, $\forall$ en $\exists$) gegeven. De zogenaamde intuïtionistische logica van Brouwer is een verfijning van de klassieke logica van Aristoteles, omdat in Brouwer's logica een existentiele uitspraak alleen bewijsbaar is, indien er een getuige gevonden kan worden.

Heyting heeft in 1930 voor de geldige redeneringen van de intuïtionistische wiskunde een formeel systeem geïntroduceerd. In 1956 heeft Beth (met klassieke logica in de metataal) en in 1976 hebben Veldman en de Swart (met intuïtionistische logica in de metataal) aangetoond dat hiermee de regels voor het intuïtionistische redeneren volledig in kaart zijn gebracht. De formalisering van de logica van Frege en van Heyting waren volledig, maar niet erg natuurlijk. Gentzen heeft in 1935 een veel mooiere vorm gegeven aan beide versies van logica. De klassieke logica in de vorm van de 'sequenten calculus' en voor de intuïtionistische logica in de vorm van het systeem van 'natuurlijke deductie'.

Gebaseerd op de intuïtionistische operationele interpretatie van de logische voegwoorden heeft N.G. de Bruijn in 1968 als belangrijke bijdrage aan de Computer Wiskunde de reeds genoemde klasse van Automath talen ingevoerd. De weergave van bewijzen in Automath bestaat in feite uit een natuurlijke deductie bewijs. Hierbij is de zogenaamde type theorie van Russell essentiëel uitgebreid en gebruik gemaakt van de lambda notatie van Church. De correctheid van een op deze manier weergegeven bewijs kan in dit systeem op efficiënte manier

geverifieerd worden. De bewijzen in Automath zijn in feite intuïtionistisch. Wil men verder klassiek redeneren, dan kan dat, eenvoudig weg door het principe van het uitgesloten derde als axioma aan te nemen.

Bewijs ontwikkeling en verificatie

Zoals aangegeven door Aristoteles bestaat een belangrijk deel van het doen van wiskunde eruit om in een bepaalde axiomatische context definities te geven voor nieuwe begrippen en bewijzen te geven voor nieuwe stellingen. Wanneer deze definities en bewijzen volledig geformaliseerd zijn, dan kan automatisch geverifieerd worden of de definities welgevormd zijn en of de bewijzen kloppen. In een systeem als Automath is de verificatie zelfs efficiënt. Het is echter behoorlijk lastig om een bewijs volledig te formaliseren. Om deze taak gemakkelijker te maken zijn er zogenaamde 'bewijs ontwikkel systemen' gebouwd. In een interactieve sessie tussen gebruiker en machine worden de complexe formele bewijzen gevormd. Het komt ongeveer neer op het volgende. De gebruiker geeft aan welke stelling bewezen moet worden. Dit wordt het 'doel' genoemd. De machine vraagt dan langs welke weg de gebruiker denkt dat het bewijs mogelijkerwijs loopt. Daarbij worden nieuwe doelen gegenereerd. Wanneer dit proces voldoende vaak en op de juiste wijze herhaald wordt, dan zal het volledig geformaliseerde bewijs voltooid zijn en de computer zal kunnen verifieren of dit correct is of niet. De serie kommando's die de gebruiker moet intikken om een formeel bewijs te genereren heet ook wel een 'taktiek'. Men kan zich afvragen waarom computer ondersteunde theorievorming gewenst is. Immers, er is een goede traditie in de wiskunde om een hoge graad van betrouwbaarheid te bereiken. Als antwoord kan ten eerste opgemerkt worden dat de wiskundige graad van precisie in de loop van de geschiedenis gegroeid is. De axiomatische meetkunde van Euclides is pas door Hilbert geheel rigoureus opgezet en de analyse van Newton en Leibniz is pas door Weierstrass en Dedekind gepreciseerd. De graad van betrouwbaarheid verkregen door computer verificatie is een verdergaande stap in deze ontwikkeling. Ten tweede zijn er door de digitalisering van de informatie technologie belangrijke situaties waarbij de verificatie van bepaalde uitspraken van industrieel belang is. Bijvoorbeeld is dit het geval bij de constructie van hardware en software. De correctheid van een stuk hardware kan vertaald worden naar de correctheid van een omvangrijke uitspraak. Deze uitspraak is wiskundig niet diepgaand, maar door de omvang is computer ondersteuning bij de verificatie onmisbaar. Reeds meer dan een decennium wordt deze methode bij het ontwerpen van hardware gebruikt. De uitspraken die overeenkomen met de correctheid van software zijn moeilijker. Daarom is er voor dit gebied nog geen volledig succes. Voor een klasse van kleine maar belangrijke programma's, de zogenaamde communicatie protocollen, die bijvoorbeeld een afstandsbediening van een TV regelen, wordt er wel vruchtbaar gebruik gemaakt van computer ondersteunde verificatie. In Nijmegen gebeurt dit door collega Vaandrager en zijn groep.

Uitspraken in de wiskunde zijn niet omvangrijk, maar wel diep. Het verschil met uitspraken uit de hard- of software specificatie zit hem erin dat voor de bewijzen van deze laatste categorie een grote hoeveelheid relatief uniforme stappen

nodig zijn, terwijl voor wiskundige bewijzen er een relatief kleine verzameling stappen gedaan moet worden, maar wel met een grote diversiteit. Hoewel in de wiskunde (gewone nog niet geformaliseerde) bewijzen onderworpen worden aan een strenge kritiek van vakgenoten ('peer review'), is het ook voor deze discipline nuttig om computer geverifieerde bewijzen te hebben. Het sociologische verificatieproces kan nu gedeeltelijk overgenomen worden door machines. De 'peer reviews' blijven relevant omdat een machine niet kan bepalen wat het belang van een nieuwe stelling is. Verder ligt het in de lijn der verwachting, dat een wiskundige op den duur essentiële hulp zal krijgen van systemen voor bewijs ontwikkeling. De systemen voor Computer Algebra worden, zoals vermeld, op deze manier al gebruikt voor onderzoek in de zuivere wiskunde.

Methodologie

Men kan zich afvragen of een bewijs dat door een computer geverifieerd is, eigenlijk wel correct is. Met andere woorden, de vraag komt op of het programma dat de verificatie uitvoert zelf wel klopt. Deze vraag is op een methodologisch bevredigende manier beantwoord door N.G. de Bruijn, die aantoonde dat het mogelijk is voor bewijzen die in voldoende mate geformaliseerd zijn een zeer kort verifierend programma te gebruiken. Dit heeft ermee te maken dat de correctheid van een redenering kan worden bepaald door voor iedere stap na te gaan of deze voorkomt in de korte lijst van logische stappen die toegestaan zijn. Door nu met de hand dit korte verificatie programma heel goed na te kijken, heeft men de hoogst mogelijke graad van zekerheid. Dit geeft aan wat het belang van geformaliseerde bewijzen is: de skeptische gebruiker kan desgewenst zijn eigen verificatie programma schrijven. Een systeem dat alleen uitspraken doet zonder bewijs erbij, is *a priori* minder betrouwbaar.

Omdat er tijdens bewijzen af en toe gerekend moet worden, zal men soms gebruik willen maken van algoritmen. Om het bewijs nu geloofwaardig te houden moet de correctheid van dit algoritme ook nog bewezen worden. Omdat bij vele efficiënte algoritmen nog geen correctheidsbewijs voorhanden is, wordt er door sommige gebruikers van een verificatie systeem water in de wijn gedaan door deze correctheid axiomatisch aan te nemen. Anderen streven naar de hoogste graad van betrouwbaarheid door alleen een minimum aan niet formeel geverifieerde algoritmen toe te laten. Omdat er verschillende soorten gebruikers zijn, kan een systeem voor bewijsverificatie in principe voorzien worden van een 'joystick', die bepaalt hoeveel algoritmen men om pragmatische redenen aanneemt correct te zijn.

Op weg naar Computer Wiskunde

Onder 'Computer Wiskunde' wordt verstaan het met behulp van systemen voor bewijs ontwikkeling tot stand brengen van gevalideerde uitspraken. De menselijke intuïtie zal hierbij nog wel geruime tijd een rol bij blijven spelen: pas als de gebruiker weet hoe het bewijs loopt, kan samen met het systeem een formeel bewijs ontwikkeld worden. Hoewel er reeds spectaculaire successen geboekt zijn, met name op het gebied van het ontwerpen van hardware met een hoge graad

van betrouwbaarheid, zijn er nog vele fronten waarop de methode aanzienlijke verbetering behoeft. De belangrijkste is het gebruikersgemak. Indien het formaliseren van een stuk wiskunde vergelijkbaar wordt met het schrijven ervan op een modern systeem voor wiskundige tekstverwerking, zoals LaTeX, dan zal er een wezenlijke vooruitgang geboekt zijn. Het is mogelijk dat het formaliseren zelfs nog gemakkelijker kan worden, omdat LaTeX 'plat' is, dat wil zeggen geen wiskundige betekenis kent, terwijl de in type theorie geformaliseerde wiskundige bewijzen een rijke structuur bezitten, die bij het intikken van pas kan komen. Om dit te bereiken is onderzoek nodig waarbij fundamenteel logische, wiskundige en implementatie kennis ingebracht zal moeten worden.

Wanneer de systemen voor Computer Wiskunde voldoende ver ontwikkeld zullen zijn, zal de rol van een referee alleen nog bestaan uit de beoordeling van de relevantie van een artikel, want de correctheid is immers al gegarandeerd. Verder kan er internationaal via het internet gewerkt worden aan een grote database van gevalideerde wiskunde. De bewijzen hieruit kunnen dan door andere onderzoekers gebruikt worden voor het vormen van nieuwe theorie.

Omdat alle wiskundige objecten exact op een systeem voor Computer Wiskunde gerepresenteerd kunnen worden, kan men in principe een ingewikkelde integraal selecteren en op een knop drukken om de numerieke waarde in willekeurig veel decimalen weer te geven. Numerieke wiskunde en Computer Algebra zullen geïntegreerd worden met Computer Wiskunde. Uiteraard zullen de specifieke methoden van die twee vakken relevant zijn en blijven.

Tenslotte is Computer Wiskunde van belang voor het onderwijs. Met dit hulpmiddel zullen er interactieve boeken mogelijk worden, waarbij de lezer de opgave kan krijgen om een stuk wiskunde te ontwikkelen en meteen kan zien of het werk goed gemaakt is. Misschien dat hiermee de schade, veroorzaakt door het nagenoeg afschaffen van bewijzen op het VWO, nog ongedaan gemaakt kan worden.

Henk Barendregt

Met dank aan Frans Janssen voor hulp in de formulering.

6.2 Propositional Logic, Satisfiability and Computation

Jan Friso Groote, Hans van Maaren, Joost Warners

The satisfiability problem (SAT) of propositional logic can be described as follows. Given a propositional *formula* Φ involving m propositions p_i, which are connected by the logical binary connectors '$\wedge$' ('and'), '$\vee$' ('or'), '$\rightarrow$' ('implication') and '$\leftrightarrow$' ('equivalence') and the unary connector '$\neg$' ('negation'), does there exist an assignment of the values *true* and *false* to the proposition(–letter)s, such that the formula Φ evaluates to *true*. This problem is in general hard to solve; in fact, the satisfiability problem is the original NP–complete problem (Cook 1971 [3]). Thus it is not known whether there exists an algorithm that is garantueed to yield the correct answer to any instance of SAT, and runs in polynomial time. It is assumed to be highly unlikely that such an algorithm exists. Note that specific classes of SAT can be efficiently solved, such as 2–SAT [1] and Horn formulae [6].

In the last decades there has been extensive research on solving the SAT problem. This has various reasons:

- SAT is considered to be of fundamental importance in a number of different disciplines, such as computer science, mathematical logic, electrical engineering and operations research.
- As pointed out, the SAT problem is at the basis of the complexity theory; in a sense it is the easiest NP–complete problem. Still, developing effective algorithms for SAT might lead to finding good algorithms for more involved hard problems. On the other hand, a possible approach to such problems is to first transform them to SAT, and to subsequently solve them using a SAT solver.
- The SAT problem has a large number of practical applications. SAT–formulations and solvers have been used in the context of automated reasoning, VLSI design, safety issues in railway stations, network design etc. As a result of the enormous increase in computing power, it has become tractable to solve real–world problems.

Initially, algorithms for solving SAT came from mathematical logic (for example truth tables and tableau methods [4]) and computer science (e.g. the Davis–Putnam procedure [5] and resolution [13]). These are all *symbolic* methods, and they are *complete*, i.e. they yield a correct answer to any instance of SAT (given enough run–time).

More recently, a large number of different approaches, both complete and incomplete, have been proposed and applied. An *incomplete* method is not guaranteed to yield the correct answer to an instance of SAT. However, if the instance under consideration is satisfiable, an incomplete method might yield a correct answer in relatively very short time, by finding a satisfiable assignment. If an incomplete method fails in finding a satisfiable assignment, no definite conclusions can be drawn.

Examples of complete methods are the Davis–Putnam–Loveland procedure [12] (a modification on the DP–procedure involving *branching*; its success depends very much on the *branching rule* used, see e.g. [11]), Binary Decision Diagrams [2] and Branch and Bound or Branch and Cut algorithms [9, 10], using mathematical programming techniques. Successful incomplete techniques include local search [14, 7] and continuous optimization techniques [8][4].

The afore mentioned techniques have been applied to various types of SAT problems, both random and structured. Unfortunately, the understanding as to why certain techniques appear to be more efficient than others is limited. Moreover, there is in general hardly any understanding what causes a given instance of the SAT problem to be efficiently (i.e. in reasonable time) solvable by a certain technique. In this respect, it appears that researchers who are working on the SAT problem from their own particular perspective with their own particular algorithms are often hardly aware of the research that is carried out by others. We feel that it would be very fruitful for everyone involved

[4] It may be noted that most algorithms operate on propositional formulas in *conjunctive normal form* (CNF). An arbitrary formula can be transformed to CNF in linear time [15].

to learn more about the research that is done and has been done by others, on (essentially) the same problem. Specifically, learning about successes and failures of certain techniques can give us a deeper understanding of the techniques, and thus lead to enabling us to solve instances of SAT more effectively.

To facilitate the communication between researchers, we are organizing a seminar on *Propositional Logic, Satisfiability and Computation* (PLSAC). The aim of this seminar is to discuss the SAT problem in its broadest sense, both from a computational (practical) and a theoretical viewpoint. It will take place at the CWI, Kruislaan 413, Amsterdam, every two weeks on Thursday from 14:00 to 16:00. The first seminar was on Thursday February 13.

References

[1] B. Aspvall, M.F. Plass, and R.E. Tarjan. A linear–time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3):121–123, 1979.

[2] R.E. Bryant. Graph–based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C–35(8), 1986.

[3] S.A. Cook. The complexity of theorem proving procedures. In *Proceedings of the 3rd annual ACM symposium on the Theory of Computing*, pages 151–158, 1971.

[4] D. Van Dalen. *Logic and structure.* Springer–Verlag, Berlin, 3rd edition, 1994.

[5] M. Davis and H. Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7:210–215, 1960.

[6] W.F. Dowling and J.H. Gallier. Linear–time algorithms for testing the satisfiability of propositional Horn formulae. *Journal of Logic Programming*, 1(3):267–284, 1984.

[7] J. Gu. Local search for the satisfiability (SAT) problem. *IEEE Transactions on Systems, Man and Cybernetics*, 23(4):1108–1129, 1993.

[8] J. Gu. Global optimization for satisfiability (SAT) problem. *IEEE Transactions on Knowledge and Data Engineering*, 6(3):361–381, 1994.

[9] J.N. Hooker. Generalized resolution and cutting planes. *Annals of Operations Research*, 12:217–239, 1988.

[10] J.N. Hooker and C. Fedjki. Branch–and–cut solution of inference problems in propositional logic. *Annals of Mathematics and Artificial Intelligence*, 1:123–139, 1990.

[11] J.N. Hooker and V. Vinay. Branching rules for satisfiability. *Journal of Automated Reasoning*, 15(3):359–383, 1995.

[12] D.W. Loveland. *Automated theorem proving: a logical basis.* North–Holland, Amsterdam, 1978.

[13] J.A. Robinson. A machine–oriented logic based on the resolution principle. *Journal of the ACM*, 12:23–41, 1965.

[14] B. Selman, H. Kautz, and B. Cohen. Local search strategies for satisfiability testing. In *Dimacs Series in Discrete Mathematics and Theoretical Computer Science*, volume 26, pages 521–532. 1996.

[15] G.S. Tseitin. On the complexity of derivation in propositional calculus. *Studies in Constructive Mathematics and Mathematical Logic*, part 2:115–125, 1968. Reprinted in J. Siekmann and G. Wrightson (editors), *Automation of reasoning* vol. 2, Springer–Verlag Berlin, 1983.

6.3 DNA Computing

H.J. Hoogenboom Leiden University
Presentation for the IPA Introductiedagen, session 'trends in programming', September 1996, Noordwijk – molecular computation

Introduction. Within Computer Science the field of *Natural Computation* studies computational techniques inspired by natural phenomena. A well-known example of such a technique is evolutionary computation (genetic algorithms) where properties of objects are coded using a sequence of bits (their 'genetic information'), and the most suitable object emerges from a pool of candidates by a process similar to natural selection. Note that evolutionary computation is usually performed on classical 'silicon' computers. With *DNA computing* one intends to turn the table by proposing strands of DNA as hardware, and not only as a programming paradigm. These strands are to be manipulated by biological and biotechnological methods. On the one hand nature has provided us with a large box of enzymes designed to perform specific operations on DNA, on the other hand biotechnological research leads to new and powerful lab techniques useful for genetic manipulations.
The potential of molecular computation is quite impressive. A simple test tube of DNA may easily contain 10^{15} strands of DNA, all of which can be operated in a massively parallel fashion, leading to "the ultimate computer: a trillion calculations in parallel" [8]. The scale of DNA is so small that it provides a very dense storage medium which, under favourable conditions, can be kept over a long time. Finally, biotechnological research is constantly improving on the tools that are available for synthesising, manipulating and analysing DNA.
In 1994 Leonard Adleman reported his initial and exciting experiments in Science [1]. He solved a very simple instance of the Hamiltonian path problem using DNA, providing the first laboratory sample of molecular computation.

The biochemical tool box. A single strand of DNA (deoxyribonucleic acid) consists of a sequence of nucleotides linked together by a strong backbone. These nucleotides may differ by their attached 'bases' only, four of which occur in ordinary DNA: Adenine, Thymine, Guanine, and Cytosine. Two single strands of DNA can zip together into a stable double stranded molecule, provided the base pairs in the two strands are complementary: Adenine only matches to Thymine, Guanine to Cytosine. This phenomenon is called Watson-Crick complementarity, after its discoverers. For our purposes DNA can be conveniently represented by (double) strings over the base alphabet {A,T,C,G}.
There are several basic techniques that are used to compute with DNA:

denaturing double stranded DNA; splitting it into single strands, for instance by increasing the temperature.

annealing 'sticky ends'; pieces of double stranded DNA with single stranded overhang can be recombined into a double strand if the extending pieces match through complementarity; an enzyme called *ligase* repairs the phosphor-ribose backbones.

$$\begin{array}{r} \texttt{TATA} \\ \texttt{ATATCGT} \end{array} + \begin{array}{r} \texttt{GCACT} \\ \texttt{GA} \end{array} \rightarrow \begin{array}{r} \texttt{TATAGCACT} \\ \texttt{ATATCGTGA} \end{array}$$

separation of strands of DNA by their length; usually exploiting the fact that the travelling speed through a gel is dependent on the size of the molecule.

selecting single stranded sequences that match specific patterns; by binding them to a surface that carries the complement of the pattern.

multiplying a specific (double) strand of DNA, 'amplification'; perhaps most efficient amplification technique is by a process called PCR, *polymerase chain reaction*. In this reaction double strands are repeatedly split into single strands which are grown into full double strands by an enzyme called polymerase (with a sufficient supply of nucleotides).

Adleman's experiment. Adleman proposed the following 'abstract' nondeterministic algorithm for solving the directed Hamiltonian path problem (find a path with given initial and final nodes that passes through each vertex in the graph exactly once) on a graph with n nodes. He then implemented this algorithm on the new hardware: DNA.

1. generate random paths through the graph
2. keep only paths from the initial to the final node

3. keep only paths that enter exactly n vertices
4. keep only paths that enter all of the vertices (at least once)
5. if any paths remain, the graph contains a Hamiltonian path

The input graph is coded into DNA as follows. Start by synthesising different base sequences for each of the vertices in the graph. The edges in the graph are represented by base sequences that will anneal to each of their adjacent vertices. When these short sequences are mixed together in large quantities arbitrary large double stranded molecules will form, representing paths in the graph.
Consider for example the vertex sequences `GTAGACCT` and `GCGTTCAC`, that can be joined by the edge sequence `TGGACGCA`, to form

```
GTAGACCTGCGTTCAC
  TGGACGCA
```

Paths from the initial to the final vertex are selected by special tuned forms of PCR, amplifying only segments that start and end with specific patterns. As we know the number of bases used to code each vertex, the paths that enter precisely n vertices can be recovered by selection on molecular length.
Selecting on the base sequences for each of the vertices successively, we keep only those paths that enter each of the vertices (exactly) once. As very small amounts of DNA can't be detected, the remaining residue is amplified by PCR to see whether any path has survived the selection.

Comments. Adleman is well aware that his experiment was only an initial step towards real applications. We summarise his own comments on the feasibility of his technique.

He states that his methods could be scaled up to accommodate larger graphs. Critics have computed that this inevitably will mean bath-tubs full of DNA to code problems of a more realistic size. Adleman argues that his algorithm may be a poor implementation, and that suitable codings must be searched for. Not only space, but also time requirements are problematic. The original experiment took approximately seven days of lab work. Automation and alternative molecular algorithms may help to overcome this slow performance.

There are several details of the implementation that may result in possible errors. The main biochemical techniques used, PCR and separation procedures, are extremely error prone. Moreover, DNA may form non existing pseudo-paths by accidental ligation between similarly coded nodes. Additionally, DNA may form into hairpin loops.
A large part of present research is dealing with these sources of errors. New, refined implementations have been proposed to minimise faulty results [2].

A comment on the power of the method is in order here. Adleman has computed that there are about 10^{14} strands of DNA in his test tubes, and he believes that 10^{20} is a plausible number to deal with. Even if we take into account the large reaction times, this number of parallel operations outperforms the present supercomputers by several thousand-folds.

The potential of the method lies in massively parallel searches. It is however not clear at this moment whether the techniques can be used to solve real computational problems, like the multiplication of 100-digit numbers.

Further Models. Adleman's seminal experiment initiated a lot of new research, answering Adleman's question for alternative models and better tuned DNA implementations of algorithms. We will have now a short look at some of these (which were selected according to personal taste).

Lipton [5] attacked another problem known to be NP-complete, the satisfiability problem. He proposed a method to code truth assignments as strands of DNA, and to select one strand by manipulating test tubes of DNA. The type of problem is, like Adleman's, again a combinatorial search. The novelty of Lipton's approach is that the problem defines the order in which several tubes are manipulated, *not* the initial coding (as in Adleman's approach).
Lipton also stressed that the operations on DNA should be classified, in order to facilitate a more 'abstract' way of modelling the implementations.

In his original paper, Adleman posed the question whether DNA computations could be 'programmed', as a further step towards DNA computers. Recall that his own experiment was special purpose. Several implementations of a Turing machine were proposed, but perhaps the most detailed description is given by Rothemund [9], who lists all the enzymes needed to perform the specific operations on his DNA Turing tapes. His intricate model does not indicate the power of molecular computation. Making no use of the inherent parallellism of DNA, a single step of his Turing machine is estimated to take four hours, slower than Turing's own pencil and paper implementation I guess.

Note that the question whether molecular computation is Turing complete is also answered by work on the *splicing system* model, proposed by Head [3], as early as 1987 (!) see for instance [4], or the work of Păun [7].

The model of Roweis et al. [10] views a single strand of DNA as addressable memory, initially set to zero. A particular memory address can be set to one, by synthesising the strand of DNA complementary to the specific address, and annealing this to the original strand. Strands with specific bits set can be selected by the usual selection on DNA patterns. A robotic machinery to handle tubes of memory strands is proposed. They conclude that some challenging problems (breaking the Data Encryption Standard) need only modest volumes of DNA.

The initial contents of Adleman's test tubes are self-assembling: after synthesising only a small number of relatively short segments of DNA (in large amounts) the paths form themselves. Winfree et al. [11] investigate how this principle can be used to self-assemble computations of a programmable computer. (The models for simulating Turing machines that have been proposed all involve several laboratory steps for one cycle of the Turing machine computation.) One can argue that in Adleman's case the self-assembly is of a regular (finite automaton) type. Winfree shows that, using two-dimensional properties of DNA, this can be improved to self-assembly of context-free type: DNA fragments literally grow into derivation trees of context-free grammars. A more speculative proposal is to assemble intricately twisted pieces of DNA into computations of cellular automata (having Turing power).

Concerns. During the second Workshop on DNA Based Computers [2] people active in the 'conceptual' field of DNA computing met with people familiar with biotechnological manipulation with DNA. They expressed great optimism towards the final goal, the construction of a desk-top DNA computer, and pictures of robots manipulating large racks of tubes were sketched. Of course some realistic concerns were expressed related to the development of such hardware.
We summarize the main points that came up in the discussions.

- Almost no lab work has been done on the computational models, so far most of them have been conceptual models only. It is a common feeling that these concepts need some initial test for feasibility.

- Real-life *DNA properties* must be taken into account. Although DNA is a very versatile and stable molecular structure, it is not the linear one-dimensional string we like to think it is. It may engage into non-complementary binding and form into hairpin loops.

- We need a more robust *complexity analysis*, involving reaction rates. Just claiming that "Adleman efficiently solved an NP-complete problem" by attacking a specific seven city instance does not increase a possible acceptance of DNA computing by other computer scientists.

- Ordinary parallel computers allow at least basic *communication* between processors. In the DNA model no realistic ways of achieving this have been proposed yet.

- A common current intuition says that molecular computation will not outperform silicon computers on every type of problem. So far, DNA models have proved to be strong in massively parallel searches, but even a two bit adder turns out to be a major technological problem. The field definitely needs a '*killer application*' to prove its strength.

Conclusion. The consensus of the second Workshop on DNA Based Computers was that one should try to build a DNA computer now, by attempting to use the tremendous resources available for biochemical research (as someone said: "if you need something, just ask the chemists"). The final question is of course how far we are from the desktop DNA computer. On the one hand, we have to be patient: one was using the vacuum tube first, before moving to the transistor. On the other hand, progress in biotechnology (driven by commercial motivations other than DNA computing) has been tremendous, surely contributing to great expectations for the feasibility of DNA computing.

Thank you. Papers in the field were studied in an informal Leiden seminar, together with Grzegorz Rozenberg, Nikè van Vugt, and Ray Dassen. Ray maintains an extensive annotated bibliography on molecular computing, available at `http://www.wi.leidenuniv.nl/home/jdassen/dna.html`.

References

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

[2] DNA Based Computers, Second Annual Meeting, preliminary proceedings, June 10–12, 1996. DIMACS Workshop, Princeton University, Princeton, NJ, USA.

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

[4] T. Head, G. Păun, D. Pixton. Language Theory and Molecular Genetics – generative mechanisms suggested by DNA recombination. In: G. Rozenberg, A. Salomaa (eds.) Handbook of Formal Language Theory, vol. 2, Springer-verlag, 1997.

[5] R.J. Lipton. DNA Solution of Hard Computational Problems. Science 268 (1995) 542–545.

[6] R.J. Lipton, E.B. Baum (eds.) DNA Based Computers: Proceedings of a DIMACS Workshop. American Mathematical Society, DIMACS Series in Discr. Math. and Theor. Comp. Sci., Vol. 27, 1996.

[7] G. Păun. On the Power of the Splicing Operation. Int. Journal of Computer Mathematics 59 (1995) 27–35.

[8] R. Pool. The Ultimate Computer: A trillion calculations in parallel. New Scientist, 13 July 1996, pp. 26–31.

[9] P.W.K. Rothemund. A DNA and Restriction Enzyme Implementation of Turing Machines. In: DNA Based Computers 1996, [6], pp. 75–119.

[10] S. Roweis, E. Winfree, R. Burgoyne, N.V. Chelyapov, M.F. Goodman, P.W.K. Rothemund, and L.M. Adleman. A Sticker Based Model for DNA Computation. In: DNA Based Computers, Second Annual Meeting 1996, [2], pp. 1–27.

[11] E. Winfree, X. Yang, N.C. Seeman. Universal Computation via Self-assembly of DNA: some theory and experiments. In: DNA Based Computers, Second Annual Meeting 1996, [2], pp. 172–190.

6.4 Physics of Computation and the Quantum Computing Challenge

P. Vitányi

February 18, 1997

Abstract

New computation devices increasingly depend on particular physical properties rather than on logical organization alone as used to be the case in conventional technologies. The laws of physics imposes limits on increases in computing power. Two of these limits are interconnect wires in multicomputers and thermodynamic limits to energy dissipation in all computers. Quantum computing is a new computational technology which promises to eliminate problems of latency and wiring associated with parallel computers and the rapidly approaching ultimate limits to computing power imposed by the fundamental thermodynamics.

1 Introduction

In a sequential computation such as performed by a Turing machine or a von Neumann architecture computer, one can safely ignore many physical aspects of the underlying computer system and analyse the computational complexity of a program in a purely logical fashion. In the realm of nonsequential computation one cannot ignore the reality of the physical world we live in to such an extent. The appropriateness of the analysis may stand or fall with the account taken of physical reality. Moreover, nonclassical or nonstandard physical realizations of computers may have totally unexpected properties.

A popular model to analyse parallel algorithms is the parallel random access machine (PRAM) were many processors can read and write a single shared memory in unit time per operation. Typically, for n inputs we have $p(n)$ processors and clever algorithms are designed which, say, add n numbers of n^ϵ bits in $O(\log n)$ parallel time (the longest chain of operations executed by any

*Partially supported by the European Union through NeuroCOLT ESPRIT Working Group Nr. 8556, and by NWO through NFI Project ALADDIN under Contract number NF 62-376 and NSERC under International Scientific Exchange Award ISE0125663. Address: CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands. Email: paulv@cwi.nl

single processor in the lot). However, something is wrong here. Since $p(n)$ processors are necessary and sufficient for the algorithm, we cannot dispense with any one of them, and hence the results of the calculations of each pair of processors must interact somewhere. This means that we have to signal between each pair of processors, and, taking the outermost ones, the distance between them is $\Omega(p(n)^{1/3})$ because of the geometry of space. Hence the time required for interaction is $\Omega(p(n)^{1/3})$ by the bounded speed of light. That is, what we called 'parallel time' is in fact a series of 'consecutive steps' were the length of each step depends on physical considerations. A similar problem of relation between the theoretical model and physical realization occurs with NC networks (polynomial number of processors and polylogarithmic depth).

In fact, optimality of PRAM algorithms may be misleading, because in any physically realizable machine architecture it may be the case that a much simpler and unsophisticated algorithm outperforms the optimal PRAM algorithm. Do networks help with this problem? We can simulate PRAMs fast by networks of processors communicating by message passing at the cost of a multiplicative slowdown square logarithmic in the number of processors n for simulation on a $\log n$-dimensional hypercube, [Upfal and Wigderson, 1987]. This doesn't solve the problem mentioned above, since the hypercube nodes need to be order $n^{1/3}$ apart for the majority of pairs (see below). Together it turns out that rather than saving time, the simulation costs at least a logarithmic in n factor more time than the original. Rolf Landauer, [Landauer, 1991] has emphasized "information is physical". So is communication.

At the outset of high density electronic chip technology (VLSI = Very Large Scale Integration), a flurry of activity in analyzing computational complexity focussed on the AT^2 measure, where A is the total (two dimensional) chip area and T is the time (maximal number of transitions or steps of any component on the chip). Typically, up to a polylogarithmic factor the results say $AT^2 = \Omega(n^2)$ for many problems (for example input n bits and determine whether or not they sum to $n/2$). It seems difficult to reach significantly higher lower bounds. Superficially, it seems that this measure is nice since it gives a lower bound tradeoff for time versus area. However, it does not say much about *physical* chips. For n input bits, assume that at the start of the computation we have them on chip. Since each bit physically takes $\Omega(1)$ area we have that $A = \Omega(n)$ outright (for example for the Kolmogorov random inputs which are the overwhelming majority). That means that most input bits are $\Omega(n^{1/2})$ distant from most other input bits by the geometry of space argument. In any computation where none of the input bits can be ignored, each pair of bits needs to interact somewhere, and hence information must be exchanged across $\Omega(n^{1/2})$ distance. This means, by the bounded speed of light, that $T = \Omega(n^{1/2})$. Together this trivially shows $AT^2 = \Omega(n^2)$.

Even if we assume that $A = n/f(n)$ ($f(n)$ unbounded) then the chip can contain at most $n/f(n)$ input bits and the computation needs to proceed through entering about n input bits. Since the circumference of the chip is $\Omega(\sqrt{n/f(n)})$

this takes at least $\Omega(\sqrt{n \cdot f(n)})$ time, resulting in $AT^2 = \Omega(n^2)$ again. If we account for the bounded 'pinability', bounded number of pins through which the input can be entered, we find $AT^2 = \Omega(n^3/f(n))$. All these estimates are gross underestimates because they ignore actual computing time on chip.

The AT^2 measure was widely studied [Thompson, 1979, Ullman, 1984] perhaps due to the fact that the argument used is to bisect the postulated but unknown embedded communication network (divide them into two parts whith approximately equal number of nodes by a cut of the layout), and express both A and T in terms of the unknown *minimum bisection width* of the network (minimum number of edges and nodes on the cut). Fortuitiously, using AT^2, the unknown minimum bisection width gets divided out. According to [Mead & Conway, 1980], a measure like AT has physical significance because it is related to the maximal energy consumption and energy disipation of a chip. If the gates constitute a constant fraction of A, and if all gates switch at each clock cycle, conventional technologies dissipate $\Omega(AT)$ energy in the form of heat. Because of overheating and meltdown this is a main factor which determines viability. Related measures were defined and first investigated in [Kissin, 1982–1991].

Physics has a treasure trove of nonconventional technologies which may yield computation opportunities. We cite three novel items. The first one is quantum cryptography, [Bennett, *et al.*, 1992]. Viewed first as science fiction, after a working prototype had been demonstrated this idea has now been taken on by commercial developers. British Telecom recently announced a working setup using optical fiber communication in excess of 10 kilometer. A second new development is quantum coherent computation. Because new developments in quantum coherent computation (if physically realizable) allow breaking most commonly used cryptosystems, see Section 4, quantum cryptography may be the only safe principle for public cryptography currently known, [Brassard, 1994]. In contrast with other systems, whose safety rests (or rested) on unproven cryptographic assumptions, the safety of quantum cryptography rests on the validity of quantum mechanics.

A third new principle is computation using DNA. Recently a small instance of the 'Hamiltonian path problem' was encoded in molecules of DNA and solved inside of a test tube using standard methods of molecular biology, [Adleman, 1994]. This has raised excitement about the following questions: Can practical molecular computers actually be built? Might they be as much as a billion times faster that current super computers? According to [Adleman, 1994], "To some, a computer is a physical device in the real world. But being a computer is something that we externally impose on an object. There might be a lot of computers out there, and I suspect there are".

2 Geometry of Space

Models of parallel computation that allow processors to randomly access a large shared memory, such as PRAMs, or rapidly access a member of a large number of other processors, will necessarily have large latency. If we use n processing elements of, say, unit size each, then the tightest they can be packed is in a 3-dimensional sphere of volume n. Assuming that the units have no "funny" shapes, e.g., are spherical themselves, no unit in the enveloping sphere can be closer to all other units than a distance of radius R,

$$R = \left(\frac{3 \cdot n}{4\pi}\right)^{1/3} \quad (1)$$

Because of the bounded speed of light, it is impossible to transport signals over n^α ($\alpha > 0$) distance in $o(n)$ time. In fact, the assumption of the bounded speed of light says that the lower time bound on *any* computation using n processing elements is $\Omega(n^{1/3})$ outright.

We study the following problem. Let $G = (V, E)$ be a finite undirected graph, without loops or multiple edges, *embedded* in 3-dimensional Euclidean space. Let each embedded node have unit *volume*. For convenience of the argument, each node is embedded as a sphere, and is *represented* by the single point in the center. The *distance* between a pair of nodes is the Euclidean distance between the points representing them. The *length* of the embedding of an edge between two nodes is the distance between the nodes. How large does the *average* edge length need to be?

We illustrate the approach with a popular architecture, say the *binary d-cube*. Recall, that this is the network with $n = 2^d$ nodes, each of which is identified by a d-bit name. There is a two-way communication link between two nodes if their identifiers differ by a single bit. The network is represented by an undirected graph $C = (V, E)$, with V the set of nodes and $E \subseteq V \times V$ the set of edges, each edge corresponding with a communication link. There are $d2^{d-1}$ edges in C. Let C be embedded in 3-dimensional Euclidean space, each node as a sphere with unit volume. The distance between two nodes is the Euclidean distance between their centers.

Lemma 1 *The average Euclidean length of the edges in the 3-space embedding of C is at least $7R/(16d)$.*

One can derive a general theorem that gives similar lower bounds that are optimal in the sense of being within a constant multiplicative factor of an upper bound for several example graphs of various diameters, [Vitányi, 1988, Vitányi, 1994, Koppelman, 1995]. Here we have not yet taken into account that longer wires need larger drivers and have a larger diameter, that the larger volume will again cause the average interconnect length to increase, and so on, which explosion may make embedding altogether impossible with finite length interconnects as exhibited in related contexts in [Vitányi, 1985].

3 Adiabatic Computation and Thermodynamics

All computations can be performed logically reversibly, [Bennett, 1973], at the cost of eventually filling up the memory with unwanted garbage information. This means that reversible computers with bounded memories require in the long run irreversible bit operations, for example, to erase records irreversibly to create free memory space. The minimal possible number of irreversibly erased bits to do so is believed to determine the ultimate limit of heat dissipation of the computation by Landauer's principle, [Landauer, 1961, Bennett, 1973, Bennett, 1982, Proc. PhysComp, 1981, 1992, 1994]. In reference [Bennett *et al.*, 1993] we and others developed a mathematical theory for the unavoidable number of irreversible bit operations in an otherwise reversible computation.

Methods to implement (almost) reversible and dissipationless computation using conventional technologies appear in [Proc. PhysComp, 1981, 1992, 1994], often designated by the catch phrase 'adiabatic switching'. Many currently proposed physical schemes implementing adiabatic computation reduce irreversibility by using longer switching times. This is done typically by switching over equal voltage gates after voltage has been equalized slowly. This type of switching does not dissipate energy, [Proc. PhysComp, 1981, 1992, 1994], the only energy dissipation is incurred by pulling voltage up and down: the slower it goes the less energy is dissipated. If the computation goes infinitely slow, zero energy is dissipated. Clearly, this counteracts the purpose of low energy dissipation which is faster computation.

In [Li & Vitányi, 1996] it is demonstrated that even if adiabatic computation technology advances to switching with no time loss, a similar phenomenon arises when we try to approach the ultimate limits of minimal irreversibility of an otherwise reversible computation, and hence minimal energy dissipation. This time the effect is due to the logical method of reducing the number of irreversible bit erasures in the computation irrespective of individual switching times. By computing longer and longer (in the sense of using more computation steps), the amount of dissipated energy gets closer to ultimate limits. Moreover, one can trade-off time (number of steps) for energy: there is a new time-irreversibility (time-energy) trade-off hierarchy. The bounds we derive are also relevant for quantum computations which are reversible except for the irreversible observation steps, [Deutsch, 1985–1992, Benioff, 1980–1986, Benioff, 1995].

Extrapolations of current trends show that the energy dissipation per binary logic operation needs to be reduced below kT (thermal noise) within 20 years. Here k is Boltzmann's constant and T the absolute temperature in degrees Kelvin, so that $kT \approx 3 \times 10^{-21}$ Joule at room temperature. Even at kT level, a future laptop containing 10^{18} gates in a cubic centimeter operating at a gigahertz dissipates 3 million watts/second. For thermodynamic reasons, cooling the operating temperature of such a computing device to almost absolute

zero (to get kT down) must dissipate at least as much energy in the cooling as it saves for the computing.

4 Quantum Coherent Parallel Computation

Classical methods of parallel computation are plagued by wiring problems (Section 2) and heat dissipation problems (Section 3). To counteract such problems attending further miniaturization of parallel computing devices current research considers quantum mechanics based technologies. Classical use of such technologies deals with reducing feature width on chip to below the nanometer level, [Kiehl, 1994], or interacting quantum dots subnanotechnology layouts for cellular automata, [Lent *et al.*, 1994].

This section deals with the prospect of a very *nonclassical* emergent possible computer technology (quantum coherent computing or QCC) which has recently acquired great anticipated economic value. This happened by one of the most fortuitious demonstrations in computing that QCC can break the universally used public key cryptosystems by being able to factor and do the discrete logarithm in polynomial time, [Shor, 1994] with preliminary work in [Deutsch, 1985–1992, Bernstein and Vazirani, 1993, Simon, 1994]. This result opened the vista of a veritable breakthrough in computing. There are apparently formidable obstacles to surmount before a workable technology can be obtained, [Unruh, 1995].

The QCC approach as first advocated in [Benioff, 1980–1986] is currently aimed to exploit the accepted theory that quantum evolution of an appropriate system consists in a superposition of many (potentially infinitely many) simultaneous computation paths. It is theoretically possible that through the specific quantum mechanical rules of interference of the different paths one can boost the probability associated with desirable evolutions and suppress undesirable ones for certain algorithms. Upon observation of the system state one of the states in superposition is realized. By quantum specific algorithmic techniques the desired outcome can theoretically by observed with arbitrary high probability, or the desired outcome can be computed from the observed data with arbitrary high probability.

The QCC approach will partially alleviate the wiring problem (Section 2) because an exploding number of different computation paths will be simultaneously followed (with appropriate probability amplitudes, to be sure) by the same single physical apparatus requiring but a tiny amount of physical space. This is the substance of R. Feynman's dictum "there is room at the bottom" in the context of his proposal of QCC, [Feynman, 1982—1987]. Of course, since the different computation paths of a quantum computation cannot communicate as is often a main feature in a parallel distributed computation, it is only a very special type of room which is available at the bottom. Moreover, since the quantum evolution in a computation if unobstructed by observation and decoherence is re-

versible, the pure form of QCC, apart from the irreversible observation phase, is energy dissipation free. QCC seems to a very large extent to achieve the optimal adiabatic computation aimed for in Section 3. Although there seems to be agreement that energy gets dissipated in the irreversible observation phase, to the author's knowledge it is not yet clear how much. This seems to require a quantum Kolmogorov complexity based on 'qubits' (quantum bits) as defined in context of quantum information theory by [Schuhmacher, 1994], analogous to the classical bits of information theory of [Shannon, 1948]. Through a sequence of proposals [Benioff, 1980–1986], [Feynman, 1982—1987], [Deutsch, 1985–1992], there has emerged a Turing machine model of quantum coherent computing.

4.1 Background: Probabilistic Turing Machines

The simplest way to describe it seems by way of probabilistic machines. Suppose we consider the well known probabilistic Turing machine which is just like an ordinary Turing machine, except that at each step the machine can make a probabilistic move which consists in flipping a (say fair) coin and depending on the outcome changing its state to either one of two alternatives. This means that at each such probabilistic move the computation of the machine splits into two distinct further computations each with probability $1/2$. Ignoring the deterministic computation steps, a computation involving m coinflips can be viewed as a binary computation tree of depth m with 2^m leaves, where each node at level $t \leq m$ corresponds to a state of the system which after t coinflips occurs with probability $1/2^t$. For convenience, we can label the edges connecting a state x directly with a state y with the probability that a state x changes into state y in a single coin flip (in this example all edges are labeled '$1/2$').

As an example, given an arbitrary Boolean formula containing n variables, a probabilistic machine can flip its coin n times to check each of the 2^n possible truth assignments to determine wether there exists an assignment to the variables which makes the formula true. If there are m distinct such assignments then the probabilistic machine finds that the formula is satisfyable with probability at least $m/2^n$—since there are m distinct computation paths leading to a satisfyable assignment.

Now suppose the probabilistic machine is hidden in a black box and the computation proceeds without us knowing the outcomes of the coin flips. Suppose that after t coin flips in the computation we open part of the black box and observe the bit at the position of the Turing machine tape which denotes the truth assignment for variable x_5 ($5 \leq t$) which already received its truth assignment. Before we opened the black box all 2^t initial truth assignments to variables $x_1, \ldots, x_t$ were equally possible, each had probability $1/2^t$. After we observed the state of variable x_5, say 0, the probability space of possibilities has collapsed to the truth assignments which consist of all binary vectors with a 0 in the 5th position each of which has probability renormalized at $1/2^{t-1}$.

4.2 Quantum Turing Machines

A quantum Turing machine is related to the probabilistic Turing machine. Consider the same computation tree. However, instead of a probability $p_i \geq 0$ associated with each node i, such that $\sum p_i = 1$, the summation taking place over the states a computation can possible be in at a particular time instant, there is an amplitude α_i associated with each state $|i\rangle$ of an observable of the system (the notation $|\cdot\rangle$ has good reasons in quantum mechanics notation related to the particular matrix mathematics involved). The amplitudes are complex numbers satisfying $\sum ||\alpha_i||^2 = 1$, where if $\alpha_i = a + b\sqrt{-1}$ then $||\alpha_i|| = \sqrt{a^2 + b^2}$ and the summation is taken over all distinct states of the observable at a particular instant. The transitions are governed by a matrix U which represents the program executed. This program has to satisfy the following constraints. If the set of possible ID's (complete instantaneous description) of the Turing machine is X, where X is say $\{0,1\}^n$ to simplify the discussion, then U maps the column vector $\underline{\alpha} = (\alpha_x)_{x \in X}$ to $U\underline{\alpha}$. Here $\underline{\alpha}$ is a vector of amplitudes of the quantum superposition of the distinct possible states in X before a step, and $U\underline{\alpha}$ the same after the step concerned. The special property which U needs to satisfy in quantum mechanics is that it is *unitary*, that is, $U \times U^\dagger = I$ where I is the identity matrix and $U^\dagger$ is the conjugate transpose of U ('conjugate' means that all $\sqrt{-1}$'s are replaced by $-\sqrt{-1}$'s and 'transpose' means that the rows and columns are interchanged). In other words, U is unitary iff $U^\dagger = U^{-1}$.

The unitary constraint on the evolution of the computation enforces two facts.

1. If $U^0\underline{\alpha} = \underline{\alpha}$ and $U^t = U \cdot U^{t-1}$ then $\sum_{x \in X} ||(U^t\underline{\alpha})_x||^2 = 1$ for all t (discretizing time for convenience).

2. A quantum computation is reversible (replacing U by $U^\dagger = U^{-1}$).

The common example here is a simple computation on a one-bit computer. The quantum superposition of states of the computer is denoted by

$$|\Psi\rangle = \alpha\ |0\rangle + \beta\ |1\rangle,$$

where $||\alpha||^2 + ||\beta||^2 = 1$. The different possible states are $|0\rangle = \binom{1}{0}$ and $|1\rangle = \binom{0}{1}$. Our unitary operator will be

$$U = \frac{\sqrt{2}}{2} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}.$$

It is easy to verify using common matrix calculation that

$$\begin{aligned} U\ |0\rangle &= \sqrt{2}/2\ |0\rangle - \sqrt{2}/2\ |1\rangle \\ U\ |1\rangle &= \sqrt{2}/2\ |0\rangle + \sqrt{2}/2\ |1\rangle \\ U^2\ |0\rangle &= 0\ |0\rangle - 1\ |1\rangle = -\ |1\rangle \end{aligned}$$

$$U^2 \, |1\rangle \;\; = \;\; 1 \, |0\rangle + 0 \, |1\rangle = \, |0\rangle$$

If we observe the computer in state $U \, |0\rangle$, then the probability of observing state $|0\rangle$ is $(\sqrt{2}/2)^2 = 1/2$ and the probability to observe $|1\rangle$ is $(-\sqrt{2}/2)^2 = 1/2$. However, if we observe the computer in state $U^2 \, |0\rangle$, then the probability of observing state $|0\rangle$ is 0 and the probability to observe $|1\rangle$ is 1. Similarly, if we observe the computer in state $U \, |1\rangle$, then the probability of observing state $|0\rangle$ is $(\sqrt{2}/2)^2 = 1/2$ and the probability to observe $|1\rangle$ is $(\sqrt{2}/2)^2 = 1/2$. But now, if we observe the computer in state $U^2 \, |1\rangle$, then the probability of observing state $|0\rangle$ is 1 and the probability to observe $|1\rangle$ is 0. Therefore, the operator U inverts a bit when it is applied twice in a row, and hence has acquired the charming name *square root of 'not'*. It is a simple exercise to write U in terms of an if–then–else program:

$$\begin{aligned} \textbf{if } |\Psi\rangle = \, |0\rangle \textbf{ then } \; & |\Psi\rangle := \sqrt{2}/2 \; |0\rangle - \sqrt{2}/2 \; |1\rangle \\ \textbf{else } \; & |\Psi\rangle := \sqrt{2}/2 \; |0\rangle + \sqrt{2}/2 \; |1\rangle \end{aligned}$$

Without mentioning it, and perhaps without the reader even noticing, we have applied as a matter of course an absolutely crucial difference between quantum computation and probabilistic computation.

4.3 Observables

According to quantum mechanics a physical system gives rise to a complex linear vector space $\mathcal{H}$, such that each vector of unit length represents a state of the system $|\Psi\rangle \in \mathcal{H}$.

A *quantum measurement* gives rise to a Hermitian operator $\hat{A}$ (the *observable*) and a decomposition of $\mathcal{H}$ into orthogonal subspaces (different *states* of the observable)

$$\mathcal{H} = A_1 \oplus A_2 \oplus \cdots \oplus A_n,$$

with $\hat{A} = \sum_{i=1}^{n} \alpha_i P_i$ where P_i is the projector of state $|\Psi\rangle$ on A_i (say, $|a_i\rangle$). If we measure observable $\hat{A}$ in system state $|\Psi\rangle$, with $|\Psi\rangle = \sum_{i=1}^{n} c_i \, |a_i\rangle$, then the following happens with probability $||c_k||^2$:

1. The outcome of the measurement α_k is registered.
2. The superposition $|\Psi\rangle$ collapses to superposition $|a_k\rangle \in A_k$.
3. The probability of observing $|a_k\rangle$ is renormalized to 1.

4.4 Interference.

In computing the above amplitudes, subsequent to two applications of U, according to matrix calculus we found that

$$\begin{aligned} U^2 |1\rangle &= \sqrt{2}/2\,(\sqrt{2}/2(\,|0\rangle - |1\rangle) + \sqrt{2}/2(\,|0\rangle + |1\rangle)) \\ &= \tfrac{1}{2}(\,|0\rangle - |1\rangle + |0\rangle + |1\rangle) = |0\rangle. \end{aligned}$$

In a probabilistic calculation, flipping a coin two times in a row, we would have found that the probability of each computation path in the complete binary computation tree of depth 2 was $1/4$, and the states at the four leaves of the tree were $|0\rangle$, $|1\rangle$, $|0\rangle$, $|1\rangle$, resulting in a total probability of observing $|0\rangle$ being $1/2$ and the total probability of observing $|1\rangle$ being $1/2$ as well.

The principle involved is called *interference*, like with light. If we put a screen with a single small enough hole in between a light source and a target, then we observe a gradually dimming illumination of the target screen, the brightest spot being colinear with the light source and the hole. If we put a screen with two small holes in between, then we observe a diffraction pattern of bright and dark stripes due to interference. Namely, the light hits all of the screen via two different routes (through the two different holes). If the two routes differ by an even number of half wave lengths, then the wave amplitudes at the target are added, resulting in twice the amplitude and a bright spot, and if they differ by an odd number of half wave lengths then the wave amplitudes are in opposite phase and are subtracted resulting in zero and a dark spot. Similarly, with quantum computation, if the quantum state is

$$|\Psi\rangle = \alpha\,|x\rangle + \beta\,|y\rangle,$$

then for $x = y$ we have a probability of observing $|x\rangle$ of $||\alpha + \beta||^2$, rather than $||\alpha||^2 + ||\beta||^2$ which it would have been in a probabilistic fashion. For example, if $\alpha = \sqrt{2}/2$ and $\beta = -\sqrt{2}/2$ then the probability of observing $|x\rangle$ is 0 rather than $1/2$, and with the sign of β inverted we observe $|x\rangle$ with probability 1.

4.5 Quantum Parallelism and Realizations

The currently successful trick used in [Shor, 1994, Simon, 1994] is to use a sequence S_n of n unitary operations S (similar to U above) on a register of n bits originally in the all-0 state $|\Psi\rangle = |00\ldots0\rangle$. The result is a superposition of

$$S_n\,|\Psi\rangle = \sum_{x\in\{0,1\}^n} 1/\sqrt{2^n}\;\;|x\rangle$$

of all the 2^n possible states of the register, each with amplitude $1/\sqrt{2^n}$ (and hence probability of being observed of $1/2^n$.) Now the computation proceeds in parallel along the exponentially many computation paths in quantum coherent

superposition. A sequence of tricky further unitary operations and observations serves to exploit interference (and so-called entanglement) phenomena to effect a high probability of eventually observing a desired outcome.

Physical realizations of QCC will have to struggle with the fact that the coherent states of the superposition will tend to deteriorate by interaction with each other and the universe, a phenomenon called *decoherence*. In [Unruh, 1995] it is calculated that that QCC calculations using physical realizations based on spin lattices will have to be finished in an extremely short time. For example, factoring a 1000 bit number in square quantum factoring time we have to perform 10^6 steps in less than the thermal time scale $\hbar/kT$ which at 1 K is of order 10^{-9} seconds. Such a QCC computation would need to proceed at optical frequencies. See also [Chuang, *et al.*, 1995].

Another problem is *error correction*: measurements to detect errors will destroy the computation. A novel partial method for error correction has been suggested in [Berthiaume *et al.*, 1994]. A comprehensive discussion on these problems in practically applying QCC is contained in [Landauer, 1995]. New methods using quantum information theory and quantum error-correcting codes seem most promising.

References

[Adleman, 1994] L. Adleman, Molecular computation of solutions to combinatorial problems, *Science*, Vol 266, Nov 1994, 1021-1024; A Vat of DNA May Become Fast Computer Of the Future, Gina Kolata in: *The New York Times*, April 11, 1995, Science Times, pp. C1, C10.

[Barenco *et. al*] A. Barenco, C.H. Bennett, R. Cleve, D.P. DiVicenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, Elementary gates for quantum computation, submitted to *Physical Review A*, March 1995.

[Benioff, 1980–1986] P. Benioff, J. Stat. Phys., 22(1980), 563-591, also *J. Math. Phys.*, 22(1981), 495-507, *Int. J. Theoret. Phys.*, 21(1982), 177-201, *Phys. Rev. Letters*, 48(1982), 1581-1585, *J. Stat. Phys.*, 29(1982), 515-546, *Phys. Rev. Letters*, 53(1984), 1203, *Ann. New York Acad. Sci.*, 480(1986), 475-486.

[Benioff, 1995] P. Benioff, Review of quantum computation, In: *Trends in Statistical Physics*, Council of Scientific Information, Trivandrum, India, To be published.

[Bennett, 1973] C.H. Bennett. Logical reversibility of computation. *IBM J. Res. Develop.*, 17:525–532, 1973.

[Bennett, 1982] C.H. Bennett. The thermodynamics of computation—a review. *Int. J. Theoret. Phys.*, 21(1982), 905-940.

[Bennett, *et al.*, 1992] C.H. Bennett, F. Bessette, G. Brassard, L. Salvail and J. Smolin, Experimental quantum cryptography, *J. Cryptology*, 5:1(1992), 3-28; C.H. Bennett, G. Brassard and A. Ekert, Quantum cryptography, *Scientific American*, Oct. 1992, 50-57.

[Bennett *et al.*, 1993] C.H. Bennett, P. Gács, M. Li, P.M.B. Vitányi and W.H Zurek, Thermodynamics of computation and information distance *Proc. 25th ACM Symp. Theory of Computation.* ACM Press, 1993, 21-30.

[Bernstein and Vazirani, 1993] Bernstein, E. and U. Vazirani, "Quantum complexity theory", *Proc. 25th ACM Symposium on Theory of Computing*, 1993, pp. 11 – 20.

[Berthiaume *et al.*, 1994] A. Berthiaume, D. Deutsch and R. Jozsa, The stabilisation of quantum computations, *Proc. 3rd IEEE Workshop on Physics and Computation (PhysComp '94)*, IEEE Computer Society Press, 1994.

[Brassard, 1994] G. Brassard, Cryptology Column—Quantum computing: The end of classical cryptography? *SIGACT News*, 25:4(Dec 1994), 15-21.

[Chuang, *et al.*, 1995] I.L. Chuang, R. Laflamme, P. Shor, and W.H. Zurek, Quantum computers, factoring and decoherence, Report LA-UR-95-241, Los Alamos National Labs, 1995 (quant-ph/9503007).

[Deutsch, 1985–1992] D. Deutsch, Quantum theory, the Church-Turing principle and the universal quantum computer. *Proc. Royal Society London.* Vol. A400(1985), 97-117; see also *Proc. Royal Society London*, A425(1989), 73-90; with R. Josza, *Proc. Royal Society London*, A439(1992), 553-558.

[Feynman, 1982—1987] R.P. Feynman, Simulating physics with computers, *Int. J. Theoret. Physics*, 21(1982), 467-488; Quantum mechanical computers. *Foundations of Physics*, 16(1986), 507-531. (Originally published in *Optics News*, February 1985); Tiny Computers Obeying Quantum Mechanical Laws. In: *New Directions in Physics: The Los Alamos 40th Anniversary Volume*,, N. Metropolis and D. M. Kerr and G. Rota, Eds.,Academic Press,, Boston, 1987, 7-25.

[Kiehl, 1994] R.A. Kiehl, Research toward Nanoelectronic computing technologies in Japan, In: Proc. 3rd Workshop on Physics and Computation (PhysComp'94), IEEE Computer Society Press, 1994, 1-4.

[Kissin, 1982–1991] G. Kissin, Measuring Energy Consumption in VLSI Circuits: a Foundation, Proc. 14th ACM Symp. Theor. Comp., 1982, 99-104; Lower and Upper Bounds on the Switching Energy Consumed by VLSI Circuits, *J. Assoc. Comp. Mach.*,, 38(1991), pp. 222-254.

[Koppelman, 1995] D.M. Koppelman, A lower bound on the average physical length of edges in the physical realization of graphs, Manuscript Dept ECE, Lousiana State Univ. Baton Rouge, 1995.

[Landauer, 1961] R. Landauer. Irreversibility and heat generation in the computing process. *IBM J. Res. Develop.*, 5:183–191, 1961.

[Landauer, 1991] R. Landauer, Information is physical, *Physics Today,* 44:May(1991), 23-29.

[Landauer, 1995] R. Landauer, Is quantum mechanics useful? *Proc. Roy. Soc. Lond.*, to be published.

[Lent *et al.*, 1994] C.S. Lent, P.D. Tougaw, W. Porod, Quantum cellular automata: The physics of computing with arrays of quantum dot molecules. In: Proc. 3nd Workshop on Physics and Computation (PhysComp'94), IEEE Computer Society Press, 1994, 5-13; also J. Appl. Phys., 74(1993), 3558, 4077, 6227, 75(1994), 1818.

[Li & Vitányi, 1993] M. Li and P.M.B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications.* Springer-Verlag, New York, 1993.

[Li & Vitányi, 1996] M. Li and P.M.B. Vitányi. Reversibility and adiabatic computation: trading time and space for energy, Proc. Royal Society of London, Series A, 452(1996), 769-789.

[Mead & Conway, 1980] C. Mead and L. Conway. *Introduction to VLSI Systems.* Addison-Wesley, 1980.

[Proc. PhysComp, 1981, 1992, 1994] Proc. 1981 Physics and Computation Workshop. *Int. J. Theoret. Phys.*, 21(1982). *Proc. 1992 Physics and Computation Workshop.* IEEE Computer Society Press, 1992. *Proc. 1994 Physics and Computation Workshop.* IEEE Computer Society Press, 1994.

[Schuhmacher, 1994] , B.W. Schumacher, On Quantum coding, *Phys. Rev. A*, in press to appear in 1995; (with R. Josza), A new proof of the quantum noiseless coding theorem, *J. Modern Optics*, 41(1994), 2343-2349.

[Shannon, 1948] C.E. Shannon, A mathematical theory of communication, *Bell System Tech. J.*, **27**(1948), 379-423, 623-656.

[Shor, 1994] Shor, P., Algorithms for quantum computation: Discrete log and factoring, *Proc. 35th IEEE Symposium on Foundations of Computer Science*, 1994, 124-134.

[Simon, 1994] Simon, D., On the power of quantum computation, *Proc. 35th IEEE Symposium on Foundations of Computer Science*, 1994.

[Thompson, 1979] C. Thompson, Area-time complexity for VLSI, Proc. 11th ACM Symp. Theor. Comp., 1979, 81-88.

[Ullman, 1984] J. Ullman, *Computational Aspects of VLSI*, Computer Science Press, Rockville, MD, 1984.

[Unruh, 1995] Unruh, W. G., Maintaining coherence in quantum computers, *Physical Review A*, 51(1995), 992-.

[Upfal and Wigderson, 1987] E. Upfal and A. Wigderson, How to share memory on a distributed system, *J. Assoc. Comp. Mach.*, 34(1987), 116-127.

[Vitányi, 1985] Area penalty for sublinear signal propagation delay on chip, *Proceedings 26th IEEE Symposium on Foundations of Computer Science*, 1985, 197-207.

[Vitányi, 1986] P.M.B. Vitányi, Non-sequential computation and Laws of Nature, In: VLSI Algorithms and Architectures (Proceedings Aegean Workshop on Computing, 2nd International Workshop on Parallel Processing and VLSI), *Lecture Notes In Computer Science 227,* Springer Verlag, 1986, 108-120.

[Vitányi, 1988] P.M.B. Vitányi, Locality, communication and interconnect length in multicomputers, *SIAM J. Computing,* 17 (1988), 659-672.

[Vitányi, 1994] P.M.B. Vitányi, Multiprocessor architectures and physical law. In: Proc. 3rd Workshop on Physics and Computation (PhysComp'94), IEEE Computer Society Press, 1994, 24-29.

Pages Deleted

The following pages contain(ed) the memberlist of the NVTI. These pages have been deleted to protect the privacy of our members.

8 Statuten

Artikel 1.
1. De vereniging draagt de naam: "Nederlandse Vereniging voor Theoretische Informatica".
2. Zij heeft haar zetel te Amsterdam.
3. De vereniging is aangegaan voor onbepaalde tijd.
4. De vereniging stelt zich ten doel de theoretische informatica te bevorderen haar beoefening en haar toepassingen aan te moedigen.

Artikel 2.
De vereniging kent gewone leden en ereleden. Ereleden worden benoemd door het bestuur.

Artikel 3.
De vereniging kan niet worden ontbonden dan met toestemming van tenminste drievierde van het aantal gewone leden.

Artikel 4.
Het verenigingsjaar is het kalenderjaar.

Artikel 5.
De vereniging tracht het doel omschreven in artikel 1 te bereiken door
a. het houden van wetenschappelijke vergaderingen en het organiseren van symposia en congressen;
b. het uitgeven van een of meer tijdschriften, waaronder een nieuwsbrief of vergelijkbaar informatiemedium;
c. en verder door alle zodanige wettige middelen als in enige algemene vergadering goedgevonden zal worden.

Artikel 6.
1. Het bestuur schrijft de in artikel 5.a bedoelde bijeenkomsten uit en stelt het programma van elk van deze bijeenkomsten samen.
2. De redacties der tijdschriften als bedoeld in artikel 5.b worden door het bestuur benoemd.

Artikel 7.
Iedere natuurlijke persoon kan lid van de vereniging worden. Instellingen hebben geen stemrecht.

Artikel 8.
Indien enig lid niet langer als zodanig wenst te worden beschouwd, dient hij de ledenadministratie van de vereniging daarvan kennis te geven.

Artikel 9.
Ieder lid ontvangt een exemplaar der statuten, opgenomen in de nieuwsbrief van de vereniging. Een exemplaar van de statuten kan ook opgevraagd worden bij de secretaris. Ieder lid ontvangt de tijdschriften als bedoeld in artikel 5.b.

Artikel 10.
Het bestuur bestaat uit tenminste zes personen die direct door de jaarvergadering worden gekozen, voor een periode van drie jaar. Het bestuur heeft het recht het precieze aantal bestuursleden te bepalen. Bij de samenstelling van het bestuur dient rekening gehouden te worden met de wenselijkheid dat vertegenwoordigers van de verschillende werkgebieden van de theoretische informatica in Nederland in het bestuur worden opgenomen. Het bestuur kiest uit zijn midden de voorzitter, secretaris en penningmeester.

Artikel 11.
Eens per drie jaar vindt een verkiezing plaats van het bestuur door de jaarvergadering. De door de jaarvergadering gekozen bestuursleden hebben een zittingsduur van maximaal twee maal drie jaar. Na deze periode zijn zij niet terstond herkiesbaar, met uitzondering van secretaris en penningmeester. De voorzitter wordt gekozen voor de tijd van drie jaar en is na afloop van zijn ambtstermijn niet onmiddellijk als zodanig herkiesbaar. In zijn functie als bestuurslid blijft het in de vorige alinea bepaalde van kracht.

Artikel 12.
Het bestuur stelt de kandidaten voor voor eventuele vacatures. Kandidaten kunnen ook voorgesteld worden door gewone leden, minstens een maand voor de jaarvergadering via de secretaris. Dit dient schriftelijk te gebeuren op voordracht van tenminste vijftien leden. In het geval dat het aantal kandidaten gelijk is aan het aantal vacatures worden de gestelde kandidaten door de jaarvergadering in het bestuur gekozen geacht. Indien het aantal kandidaten groter is dan het aantal vacatures wordt op de jaarvergadering door schriftelijke stemming beslist. Ieder aanwezig lid brengt een stem uit op evenveel kandidaten als er vacatures zijn. Van de zo ontstane rangschikking worden de kandidaten met de meeste punten verkozen, tot het aantal vacatures. Hierbij geldt voor de jaarvergadering een quorum van dertig. In het geval dat het aantal aanwezige leden op de jaarvergadering onder het quorum ligt, kiest het zittende bestuur de nieuwe leden. Bij gelijk aantal stemmen geeft de stem van de voorzitter (of indien niet aanwezig, van de secretaris) de doorslag.

Artikel 13.
Het bestuur bepaalt elk jaar het precieze aantal bestuursleden, mits in overeenstemming met artikel 10. In het geval van aftreden of uitbreiding wordt de zo ontstane vacature aangekondigd via mailing of nieuwsbrief, minstens twee maanden voor de eerstvolgende jaarvergadering. Kandidaten voor de ontstane vacatures worden voorgesteld door bestuur en gewone leden zoals bepaald in artikel 12. Bij aftreden van bestuursleden in eerste of tweede jaar van de driejarige cyclus worden de vacatures vervuld op de eerstvolgende jaarvergadering. Bij aftreden in het derde jaar vindt vervulling van de vacatures plaats tegelijk met de algemene driejaarlijkse bestuursverkiezing. Voorts kan het bestuur beslissen om vervanging van een aftredend bestuurslid te laten vervullen tot de eerstvolgende jaarvergadering. Bij uitbreiding van het bestuur in het eerste of tweede jaar van de cyclus worden de vacatures vervuld op de eerstvolgende jaarvergadering. Bij uitbreiding in het derde jaar vindt vervulling van de vacatures plaats tegelijk met de driejaarlijkse bestuursverkiezing. Bij inkrimping stelt het bestuur vast welke leden van het bestuur zullen aftreden.

Artikel 14.
De voorzitter, de secretaris en de penningmeester vormen samen het dagelijks bestuur. De voorzitter leidt alle vergaderingen. Bij afwezigheid wordt hij vervangen door de secretaris en indien ook deze afwezig is door het in jaren oudste aanwezig lid van het bestuur. De secretaris is belast met het houden der notulen van alle huishoudelijke vergaderingen en met het voeren der correspondentie.

Artikel 15.
Het bestuur vergadert zo vaak als de voorzitter dit nodig acht of dit door drie zijner leden wordt gewenst.

Artikel 16.
Minstens eenmaal per jaar wordt door het bestuur een algemene vergadering bijeengeroepen; ffiffin van deze vergaderingen wordt expliciet aangeduid met de naam van jaarvergadering; deze vindt plaats op een door het bestuur te bepalen dag en plaats.

Artikel 17.
De jaarvergadering zal steeds gekoppeld zijn aan een wetenschappelijk symposium. De op het algemene gedeelte vaan de jaarvergadering te behandelen onderwerpen zijn
a. Verslag door de secretaris;
b. Rekening en verantwoording van de penningmeester;
c. Verslagen van de redacties der door de vereniging uitgegeven tijdschriften;
d. Eventuele verkiezing van bestuursleden;
e. Wat verder ter tafel komt. Het bestuur is verplicht een bepaald punt op de agenda van een algemene vergadering te plaatsen indien uiterlijk vier weken van te voren tenminste vijftien gewone leden schriftelijk de wens daartoe aan het bestuur te kennen geven.

Artikel 18.
Deze statuten kunnen slechts worden gewijzigd, nadat op een algemene vergadering een commissie voor statutenwijziging is benoemd. Deze commissie doet binnen zes maanden haar voorstellen via het bestuur aan de leden toekomen. Gedurende drie maanden daarna kunnen amendementen schriftelijk worden ingediend bij het bestuur, dat deze ter kennis van de gewone leden brengt, waarna een algemene vergadering de voorstellen en de ingediende amendementen behandelt. Ter vergadering kunnen nieuwe amendementen in behandeling worden genomen, die betrekking hebben op de voorstellen van de commissie of de schriftelijk ingediende amendementen. Eerst wordt over elk der amendementen afzonderlijk gestemd; een amendement kan worden aangenomen met gewone meerderheid van stemmen. Het al dan niet geamendeerde voorstel wordt daarna in zijn geheel in stemming gebracht, tenzij de vergadering met gewone meerderheid van stemmen besluit tot afzonderlijke stemming over bepaalde artikelen, waarna de resterende artikelen in hun geheel in stemming gebracht worden. In beide gevallen kunnen de voorgestelde wijzigingen slechts worden aangenomen met een meerderheid van tweederde van het aantal uitgebrachte stemmen. Aangenomen statutenwijzigingen treden onmiddellijk in werking.

Artikel 19.
Op een vergadering worden besluiten genomen bij gewone meerderheid van stemmen, tenzij deze statuten anders bepalen. Elk aanwezig gewoon lid heeft daarbij het recht een stem uit te brengen. Stemming over zaken geschiedt mondeling of schriftelijk, die over personen met gesloten briefjes. Uitsluitend bij schriftelijke stemmingen worden blanco stemmen gerekend geldig te zijn uitgebracht.

Artikel 20.
a. De jaarvergadering geeft bij huishoudelijk reglement nadere regels omtrent alle onderwerpen, waarvan de regeling door de statuten wordt vereist, of de jaarvergadering gewenst voorkomt.
b. Het huishoudelijk reglement zal geen bepalingen mogen bevatten die afwijken van of die in strijd zijn met de bepalingen van de wet of van de statuten, tenzij de afwijking door de wet of de statuten wordt toegestaan.

Artikel 21.
In gevallen waarin deze statuten niet voorzien, beslist het bestuur.