NVTI Nieuwsbrief 2012

Nederlandse Vereniging voor Theoretische Informatica

Inhoudsopgave

Van het bestuur en de redactie

Geachte NVTI leden,

2012 is het Alan Turingjaar. Deze geniale wetenschapper werd 100 jaar geleden geboren in Londen. Over de hele wereld is ruime aandacht voor deze grondlegger van de Informatica, door middel van vele symposia, boeken en essays. De creatieve ideeën van Alan Turing zijn ook voor de huidige generatie Nederlandse informatici nog zeer inspirerend, getuige bijvoorbeeld het IPA onderzoeksspeerpunt: "Beyond Turing". Ook de artikelen in deze nieuwsbrief zijn direct of indirect gelinkt aan het Turingjaar.

Benedikt Löwe geeft een overzicht van de diverse activiteiten die rond het Turingjaar zijn georganiseerd. Jos Baeten, Bas Luttik en Paul van Tilburg beschrijven "Reactive Turing Machines" om nu eindelijk die ene fundamentele tekortkoming in de (on)berekenbaarheid van de Turing Machine te verhelpen. Zullen deze Reactieve Turing Machines nu eindelijk net zo intelligent reageren als wij mensen?

Naast Informatica was Turing ook hevig geïnteresseerd in biologie. We vonden het daarom passend ook twee artikelen op te nemen over de link tussen theoretische informatica en natuurwetenschappen. Sanne Abeln en Daan Frenkel behandelen een theorie die eiwitstructuren kan beschrijven. Alexandru Baltag en Sonja Smets bespreken hoe mathematische logica kan worden gebruikt bij het beschrijven van quantum-mechanische effecten.

Tenslotte voor de puzzelliefhebbers: Joost Batenburg en Walter Kosters bespreken de ongekende diepten achter de geliefde Nonogram-puzzels. Ook hier ligt een diepe connectie met Alan Turing. Weten welke? Blader dan gauw door deze Nieuwsbrief!

Theoriedag 23 maart 2012, Vergadercentrum Vredenburg 19, Utrecht.
Wanneer u deze nieuwsbrief ontvangt, weet u het al: de landelijke NVTI theoriedag komt er alweer aan. Hier belichten traditiegetrouw vier bekende sprekers uit binnen- en buitenland de complex-algoritmische dan wel de logisch-semantieke zijde van de theoretische munt. We hebben hoge verwachtingen van Jan Rutten, vele jaren trouwe NVTI secretaris, Susanne Albers, Nelly Litvak en Vincent Danos. Verderop in deze nieuwsbrief vindt u de aankondigingen van hun lezingen.

Let erop dat de locatie dit jaar gewijzigd is. We zitten nog steeds op directe loopafstand van Utrecht Centraal, maar nu in Vergadercentrum Vredenburg 19, Utrecht.

Overig nieuws. Na jarenlange trouwe bestuursdienst hebben Jan Willem Klop en Wim Hesselink zich vorig jaar teruggetrokken uit het NVTI bestuur vanwege hun emeritaat. We zijn hen erg dankbaar voor hun jarenlange inspanning om de NVTI tot een blijvend interessante vereniging te maken.

Hierbij zijn we ook alle sponsors van NVTI zeer erkentelijk voor hun steun in geld en natura: NWO, Elseviers, IPA, SIKS en het CWI. Tenslotte, Susanne van Dam, Joke Lammerink, Jan Schipper en Caroline Waij, bedankt voor jullie support voor de Nieuwsbrief en de Theoriedag.

Voor het welslagen van de NVTI dag hebben we natuurlijk een flink publiek nodig. Kom niet alleen zelf, maar neem ook uw collega's mee; vergeet vooral uw studenten en promovendi niet! Op de NVTI theoriedag ontmoet je op een ontspannen manier de thema's en onderzoekers uit de volle breedte van de Theoretische Informatica.

Jaco van de Pol en Femke van Raamsdonk,
Namens Bestuur en Redactie van NVTI,
2 Maart 2012

Huidige samenstelling van het bestuur

NVTI Theory Day March 23, 2012

Programme, Abstracts, and Registration

We are happy to invite you for the Theory Day 2012 of the NVTI (Nederlandse Vereniging voor Theoretische Informatica, Dutch Asssociation for Theoretical Computer Science). The NVTI supports the study of theoretical computer science and its applications.

NVTI Theory Day 2012
Friday March 23, 2012, 9:30-16:45
Vredenburg 19, Utrecht (close to Central Station)
`http://www.vredenburg19.nl`

We have an interesting program, covering important streams in theoretical computer science, with excellent speakers from The Netherlands and abroad:

Susanne Albers (Humboldt University Berlin, Germany)
Vincent Danos (University of Edinburgh, Scotland)
Nelly Litvak (University of Twente)
Jan Rutten (CWI and Radboud University)

It is possible to participate in the organized lunch, for which registration is required. The costs of around 15 Euro can be paid (cash) at the location. We just mention that in the direct vicinity of the meeting room there are plenty of nice lunch facilities as well.
It is also possible to participate in the organized dinner, which will take place in Restaurant Luden, close to the station, `http://www.luden.nl/`, around 18.00. The dinner (meat, fish, or vegetarian) costs around 30 euro, drinks not included.
Both for the lunch and for the dinner: Please register with Ms Caroline Waij (cpwaij@few.vu.nl or 020-5983563) no later than one week before the meeting (March 16, 2012).

The NVTI theory days are sponsored (financially or in kind) by NWO (Netherlands Organisation for Scientific Research), Elseviers Science, CWI (Dutch Center of Mathematics and Computer Science), and the Dutch research schools IPA (Institute for Programming Research and Algorithmics) and SIKS (Dutch research school for Information and Knowledge Systems).

Please find the full program and abstracts of the lectures below.

Kind regards,
Femke van Raamsdonk,
NVTI secretary.

Program of the NVTI Day on Friday March 23, 2012

9.30-10.00:	Arrival with Coffee	
10.00-10.10:	Opening	
10.10-11.00:	Speaker:	Jan Rutten (CWI and Radboud University)
	Title:	Brzozowski's algorithm (co)algebraically
11.00-11.30:	Coffee/Tea	
11.30-12.20:	Speaker:	Susanne Albers (Humboldt University Berlin, Germany)
	Title:	Energy-Efficient Algorithms
12.20-12.40:	Speaker:	Joep van Wijk (NWO)
12.40-14.10:	Lunch (see above for registration)	
14.10-15.00:	Speaker:	Nelly Litvak (University of Twente)
	Title:	Analysis of algorithms for complex stochastic networks
15.00-15.20:	Coffee/Tea	
15.20-16.10:	Speaker:	Vincent Danos (University of Edinburgh, Scotland)
	Title:	Energy, Termination and Reversible Communicating Processes
16.10-16.45:	Business meeting NVTI	

Abstracts of the Lectures of NVTI Theory Day of March 23, 2012

10.10-11.00
Speaker: Jan Rutten (CWI and Radboud University)
Title: Brzozowski's algorithm (co)algebraically
Abstract:
Brzozowski's algorithm is a somewhat unusual recipe for minimizing finite state automata: take an automaton, reverse its transitions, make it deterministic, take the part that is reachable, and then repeat all of this once more. The result will be a deterministic automaton that is the minimization of the original one.

Though an elementary description and correctness proof of the algorithm is not very difficult, the algorithm comes to most as a bit of a surprise. Here we try to add to its understanding by presenting a new proof that is based on a result by Arbib and Manes (from the late seventies) on the duality between reachability and observability (the latter is another word for minimality).

We will (a) first present a reformulation of Arbib and Manes' duality result in terms of a bit of elementary algebra and coalgebra. Algebra is the natural mathematical setting to model reachability, whereas coalgebra is the natural formalism to model observability. Next, we will (b) derive (the correctness of) Brzozowski's algorithm as a corollary from this algebra-coalgebra duality.

Our reasons for giving this new formulation of Brzozowski's algorithm are threefold:

1. The duality between reachability and observability is in itself a very beautiful but not very well-known result.
2. Based on our proof, Brzozowski's algorithm can be easily generalised to Moore automata and (probably :-) probabilistic automata.
3. Our proof forms an illustration of the strength provided by the combined use of algebra and coalgebra.

This is joint work; see: F. Bonchi, M. Bonsangue, J. Rutten, A. Silva. Brzozowski's algorithm (co)algebraically. CWI Technical Report SEN-1114, 2011, pp. 1–13. Available at: `http://homepages.cwi.nl/~janr/papers/files-of-papers/2011_SEN1114.pdf`

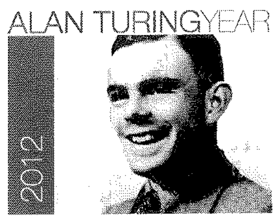
11.30-12.20
Speaker: Susanne Albers (Humboldt University Berlin, Germany)
Title: Energy-Efficient Algorithms
Abstract:
Energy conservation is a major concern today. In this lecture we will study algorithmic techniques to save energy in computing devices. More specifically, we will study power-down mechanisms that transition a system into low-power standby or sleep states when idle. Furthermore we will investigate dynamic speed scaling, a relatively recent approach to save energy in variable-speed microprocessors. For both mechanisms, we will survey important results and present some contributions that we have developed over the last two years. In particular we will address a setting that integrates power-down and speed scaling techniques.

14.10-15.00
Speaker: Nelly Litvak (University of Twente)
Title: Analysis of algorithms for complex stochastic networks
Abstract:
Complex networks receive massive attention in academia and society. Typical examples include world wide web, scientific citations, and (on-line) social networks. Algorithms and measurements on complex networks solve many important tasks. A most prominent example is the Google PageRank algorithm for ranking of web search results. In this talk I will discuss two different topics related to analysis of network algorithms. First, I will present a two-dimensional Markov process, a so-called cat-and-mouse Markov chain, that naturally arises from the analysis of an on-line method for PageRank computation. The cat performs a Markov random walk, and the mouse moves only when found by the cat. We derive asymptotic properties of this process and obtain its very unusual scaling behaviour in case when the cat component behaves as a simple reflected random walk on non-negative integers. Next, I will discuss degree-degree correlations in scale-free networks. Real-life networks are usually scale-free, that is, they contain a considerable fraction of hubs, the nodes with extremely high degrees. Degree-degree correlations describe whether nodes with high degrees usually have high degree neighbours. This is an important factor for network's robustness and performance. We analyse a so-called assortativity measure that is widely used in the literature to characterize degree-degree correlations.
This is a joint work with Philippe Robert and Remco van der Hofstad.

15.20-16.10
Speaker: Vincent Danos (University of Edinburgh, Scotland)
Title: Energy, Termination and Reversible Communicating Processes
Abstract:
We investigate a class of probabilistic reversible communicating processes where the quantitative dynamics is derived from a set of formal energy parameters. This is a kind of distributed Metropolis-Hastings algorithm. With the right energy landscape, a lower bound on the energy cost of communication guarantees that a process "terminates" in the sense that it reaches a probabilistic equilibrium state. This implies that the process hits success states in finite average time, if there are any.

The Alan Turing Year

Benedikt Löwe, Turing Centenary Advisory Committee



The year 2012 marks the centenary of the birth of Alan M. Turing, one of the fathers of the modern computer, a key figure in the decryption of the secret codes of the Nazis in the second world war, and a contributor of central ideas to many areas of modern science, in particular in the mathematical sciences. In addition to the central contributions to so many research fields, Turing became iconic not least due to the tragic events surrounding his suicide; the public apology of prime minister Gordon Brown on 10 September 2009 for the "utterly unfair treatment" created a general public interest in Turing as a person that went well beyond those who normally deal with models of computation, encryption algorithms, artificial intelligence, and the mathematical theory of pattern formation in nature. It thus does not come as a surprise that the various research communities resting on Turing's research contributions decided to celebrate his life and work during the year 2012 in a series of events and activities as the *Alan Turing Year* (ATY). Scores of academic and non-academic events will take place during 2012 in many countries of the world.

Naturally, most of the highlights of the ATY will be happening in the United Kingdom; many of the conferences with annually changing locations of the related fields have decided to hold their 2012 conference in Alan Turing's own country: for instance, *Computability in Europe* (CiE) 2012, *EuroCrypt* 2012, and *Computability and Complexity in Analysis* (CCA) 2012 will be held in Cambridge, the place where Turing studied and became a fellow of King's College in 1935; *Logic Colloquium* (LC) 2012 will be held in Manchester where Turing spent his last six years of his life. Turing's actual 100th birthday will be celebrated on Saturday, 23 June 2012. There will be separate celebrations in Cambridge (at King's College in combination with CiE 2012), in Manchester (as part of the *Turing Centenary Conference*), and at Bletchley Park, where Turing worked on the decryption of the German Enigma code in the Second World War. A list of all of the events can be found on the ATY webpage at `http://www.turingcentenary.eu/`.

Some countries such as Germany and the Netherlands have their own ATY programmes with multiple events. The programme of the Dutch *Alan Turing Jaar* can be found at `http://www.alanturing2012.nl/`: It kicks off on 7 March 2012 with a student event in Utrecht called *Turing's Legacy*. This congress is part of a series of annual symposia for computer science students organized by the *Stichting Nationaal Informatica Congres* (SNiC), this year focussing on the work of Alan Turing and its contemporary applications.

On 16 April 2012, the computer science programme of the *Radboud Universiteit* offers a Masterclass entitled *Machines maken en Breken: 100 jaar Turing* for high school students.

In 2012, the 22nd IFIP World Computer Congress will be held in Amsterdam. In conjunction with it, the CWI in Amsterdam will host the IFIP conference *Theoretical Computer Science* on 26 to 28 September 2012 with a strong Turing focus.

In October, the activities will continue in Amsterdam: on 5 October, the *Nederlandse Vereniging voor Logica & Wijsbegeerte der Exacte Wetenschappen* is organizing a special event on Alan Turing called *Turing100.NL* with talks on logic, complexity, and history of computing. Among others, the science journalist Bennie Mols who recently published a new book entitled *Turings Tango: Waarom de mens de computer de baas blijft* and Turing's biographer Andrew Hodges will be speakers at this event. In the evening, the German amateur theatre group *University Players* from Hamburg will perform the play "Breaking the Code" by Hugh Whitemore based on Hodges's Turing biography (jointly organized with the *Internationaal Homo/Lesbisch Informatiecentrum en Archief*; IHLIA). The next day, these Amsterdam Turing celebrations will continue at Science Park during the *Science Park Amsterdam Open Dag* (6 October 2012) where the research institutions of *Science Park Amsterdam* (AMOLF, NIKHEF, SARA, CWI, and the FNWI of the UvA) open their doors for everyone who is interested with an exciting programme for laypeople. In 2012, Alan Turing, whose research links numerous of these research institutions, will be an important theme of the day.

Currently the last scheduled event of the *Alan Turing Jaar* in the Netherlands is the *ATIA Alan Turing Year Conference, AI&I: truly personalised medicine* focussing on Artificial Intelligence (AI) and Agent Technology (A) with special focus on medical applications and innovative services in the biomedical field. The ATIA, or *Alan Turing Institute Almere* is a fitting choice of location to reflect on the Dutch *Alan Turing Jaar* and close the activities with the final performance of "Breaking the Code" by the *University Players.*

Most of the Dutch events are open for all researchers interested in the life and work of Alan Turing, and we encourage all members of the NVTI to consider attending one or several of these event in honour of an important figure of our field.

A Simple Lattice Model Reproduces Folding Specificity and Aggregation Behaviour of Proteins

Sanne Abeln* Daan Frenkel†

February 29, 2012

Abstract

Naturally occurring proteins have the capability of folding into a single highly specific structure, based on their amino acid sequence order alone. Reproducing this self-assembly process is far from trivial - either by physical or theoretical models. However, a simple lattice model simulated with a Markov Chain Monte Carlo method *can* reproduce highly specific folding. Here we show that is also possible to observe aggregation behaviour of proteins with these lattice models.

Introduction

Most naturally occurring proteins have evolved such that they fold into a specific structure that is able to perform the function of the protein. The design of proteins that fold quickly and uniquely into a specified native conformation is quite challenging. For this reason much of the numerical work on protein folding has focused on the folding behaviour of isolated proteins.

The protein lattice model has successfully been used to simulate protein folding and has also been used to design novel proteins that will fold into a unique, pre-selected compact structure (1–5), see Figure 1 for a 2-dimensional example of such a lattice model. Upon heating the native state of such lattice proteins, a sharp transition takes place from the folded to the unfolded state; such a sharp transition is also seen for real proteins. Note that the 3D-lattice produces a sharper transition than the 2D-lattice, e.g. comparing Figures 1 and 3

Of course, such lattice models are not sufficiently detailed to reproduce the behaviour of any specific protein. Here we will focus on the competition between protein aggregation and folding. This is a generic problem and hence lattice models can be used to gain insight into the factors that favour one process or the other.

Evolutionary pressure generally ensures that proteins do not aggregate in their natural biochemical environment, as aggregates may compromise the biological function of the proteins or may even be cyto-toxic. The immunity against aggregation of real globular proteins is not properly reproduced by most lattice-models for protein solutions. Even if the model proteins fold well in isolation, assemblies of many such proteins often exhibit aggregate-formation close to the folding temperature (Figure 2 F, 'low temperature aggregation'). Much of the earlier work on lattice proteins and similar coarse-grained models therefore focuses on small peptides that lack a well-defined hydrophobic core in the folded

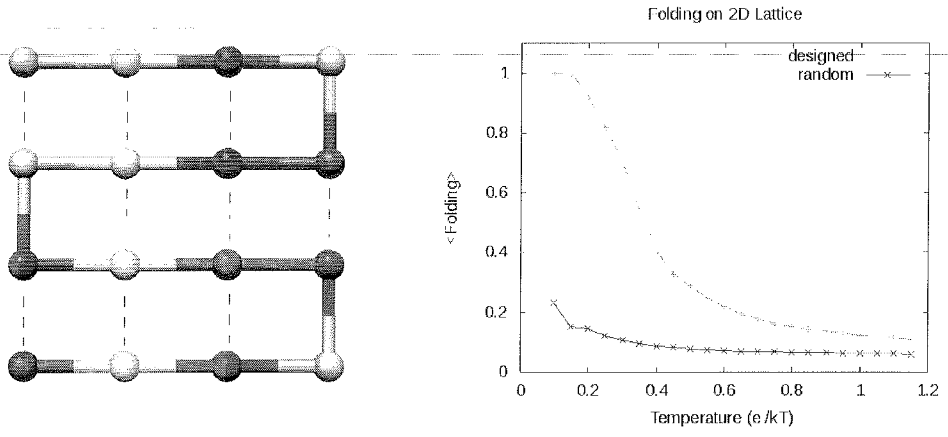
*Corresponding author: s.abeln@vu.nl, Bioinformatics section, Department of Computer Science, VU University, Amsterdam, Netherlands

†Department of Chemistry, University of Cambridge, UK

state (6–9). The unphysical aggregation behaviour of many lattice proteins means that such models are of little use for studying the behaviour of solutions that contain many folded soluble proteins. In particular, the existing models are 'ill suited' for studying how subtle changes can cause initially soluble proteins to form amyloid fibres that are implicated in neurodegenerative diseases (10). Nor can the current models be used to study the assembly of large, functional complexes that play a role in the biology of multi-component systems (e.g. Ref. 11).

Multi-protein systems are computationally very challenging and, at least at present, coarse-grained (lattice) models are therefore indispensable. Here we show a modified interaction potential that enables coarse-grained simulations of multi-protein solutions.

Square or Cubic lattice model



(a) (b)

Figure 1: (a) Example of a folded protein chain on a 2D square lattice. The dotted lines indicate contacts. Hydrophobic residues and oppositely charged residues attract each other. (b) Temperature against amount of folding for a designed sequence and a random sequence. At low temperatures the designed sequence folds with high specificity in its native fold.

One of the most widely used 3D coarse-grained protein models represents the protein as a chain on a simple square (Figure 1) or cubic lattice (Figure 3) (3–5). This model can reproduce the sharp folding transition of real proteins (Figures 1 (b) and 3). Note that the cubic-lattice model produces a sharper folding transition than the square-lattice model.

The present work also starts from such a protein model where the peptide chain is modelled with one residue per cubic lattice site (12). Non-bonded residues can only interact when they reside on neighbouring lattice sites. The internal energy of a protein configuration is given by:

$$E = \tfrac{1}{2}\sum_{i}^{N}\sum_{j}^{N}\epsilon_{a(i),a(j)}C_{ij} + \sum_{i}^{N}\epsilon_{w,a(i)}C_{wi} \tag{1}$$

where $a(i)$ denotes the amino acid at residue i and w indicates the solvent. The contact matrix $C_{ij} = 1$ when non-bonded residues i and j are located on neighbouring lattice sites, see the dotted lines Figure 1 (a). If i and j are not neighbours then $C_{ij} = 0$. The pair-potential $\epsilon_{x,y}$ gives the pairwise interactions between the amino acids x and y .

Pair-potentials

The residues of lattice proteins interact via pairwise-additive, short-ranged interactions. In what follows, we shall focus on pair interactions that are 'knowledge-based' in the sense that they are constructed to reflect the amino acid proximity in real protein structures (4). Despite the different geometry, 3D-lattice models and real protein structures have a similar coordination number: residues on the lattice have four contact partners and residues in protein structures have on average four contacts at C-beta typical interaction ranges of $6 - 7\mathring{A}$.

In knowledge-based pair-potentials, the interaction (free) energies $\epsilon_{i,j}$ between amino acid residues may, in their simplest form, be calculated as (13, 14):

$$\epsilon_{i,j} = -kT \ln\left(\frac{c_{i,j}}{\omega_{i,j}}\right) \tag{2}$$

Here $c_{i,j}$ is the number of contacts between amino acid types i and j, and $\omega_{i,j}$ is the expected number of contacts between amino acid types i and j in a set of experimentally determined protein structures.

$$\omega_{i,j} = \frac{n_i q_i n_j q_j}{\sum_k q_k n_k} \tag{3}$$

We will use a basic version of this scheme, where $\omega_{i,j}$ is based on the total number of residues of the amino acid type and the residue coordination number.

However, we will calculate the solvent term explicitly from the protein structures, making it possible to understand the effect of the solvent terms, see Ref. (15) for further details.

In this work two different pair-potentials are used: the Betancourt potential (16) (Be) that has no explicit water term ($C_{wi} = 0$) and the introduced in this work (P1) that includes a water term calculated explicitly from protein structures, using the method described above, following equations 2 and 3.

Monte Carlo simulation

To simulate the properties of the (multi) protein system, we used standard Monte Carlo simulations where trial moves are accepted according to the Metropolis rule:

$$P_{acc} = \min\left\{1, \exp\left(\frac{-\Delta E}{k_B T}\right)\right\} \tag{4}$$

where T is the simulation temperature, k_B is the Boltzmann constant and ΔE is the difference in energy between the new and old configuration of the system. Trial moves are either internal moves, changing the configuration of a chain (end move, corner flip, crank shaft and point rotation), or rigid body moves, changing the position of the chain relative to other objects (rotation, translation), see Ref. (5) for more details.

Parallel tempering, or temperature replica exchange, was used to speed up both equilibration and the sampling of uncorrelated configurations. Multiple simulations at different temperatures were run in parallel, while trying to swap temperatures every 50000 moves with 10000 trial temperature swaps in each simulation. A trial swap between the temperatures of two replicas was accepted with a probability (17):

$$P_{acc} = \min\left\{1, \exp\left(\frac{-\Delta E \cdot \Delta(1/T)}{k_B}\right)\right\} \quad (5)$$

Designing structures and sequences

Due to the coarse-grained nature of the lattice, one typically designs a sequence for a given lattice structure, rather than using a naturally occurring protein structure-sequence combination. Once the matrix $\epsilon_{x,y}$ has been specified (see below), we can design model proteins that will fold preferentially into a unique structure that is chosen beforehand. Figure 1 illustrates the result of a design process on a 2D-lattice.

The design procedures make use of a Monte Carlo scheme that minimises the energy of the amino acid sequence in the target structure whilst keeping the amino acid composition diverse – this diversity is needed to ensure the uniqueness of the native state (see Ref. (5) for details) .

Folding temperature

A peptide is defined to be in a folded state if:

$$X_f = \begin{cases} 1 & \text{if } C_n > 0.8 \cdot \max\{C_n\} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

where C_n is the number of native contacts, i.e. those contacts that are also present in the folded target structure. We define the folding temperature T_f as the temperature at which $< X_f >= 0.5$.

Aggregation behaviour

Proteins are surrounded by other proteins in the cellular environment. It is therefor not only interesting if a protein is able to fold, it is also relevant to investigate whether a protein will remain stable as a single molecule, i.e. remain soluble, or will start aggregating with other proteins.

Grand canonical simulation

A grand canonical Monte Carlo simulation was performed to investigate the aggregation behaviour of the model proteins.

In a grand canonical Monte Carlo scheme trial insertions and deletions of free chains can be performed. The acceptance rate and deletion rate, given below, ensure that the free chains remain at a constant low concentration.

Trial insertions of new chains (with an identical sequence) were accepted with:

$$P_{acc} = \min\left\{1, \frac{V}{N+1}\exp(\mu\beta)\right\} \quad (7)$$

and deleted with:

$$P_{acc} = \min\left\{1, \frac{N}{V}\exp(-\mu\beta)\right\} \quad (8)$$

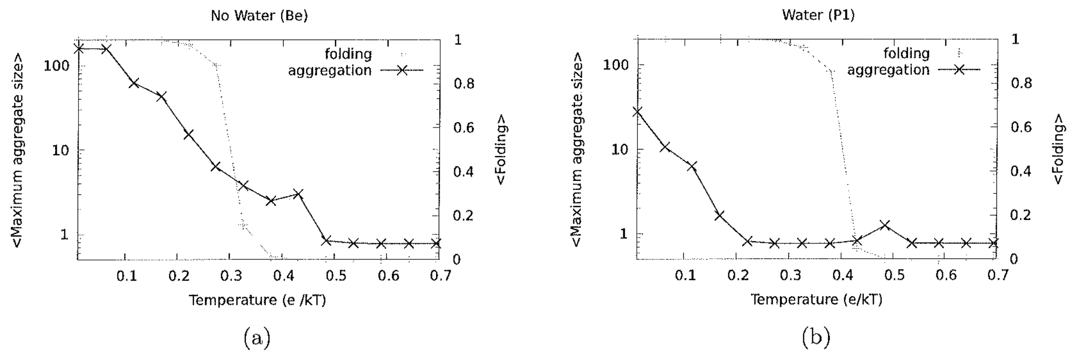


Figure 3: Folding and aggregation on a 3D Lattice. (a) At the folding temperature this protein would start to aggregate. (b) At the folding temperature this protein does not aggregate, and is stable as a monomer.

resulting in clustering of polar groups during the simulation. In real proteins polar residues also cluster together - and this explains the apparent attraction in the knowledge-based potentials. However, in real proteins this clustering occurs typically at the surface of the proteins, whilst clustering of buried polar residues is rare. The reason for the surface clustering is generally not that polar residues attract each other, but that they tend to be more strongly attracted to water than to other polar residues. The pairwise interaction terms between the solvent and polar amino acids indeed shows a stronger attraction than polar-polar terms when calculated explicitly from protein structures (see P1 potential in Supplement). In a lattice simulation the solvent terms can be incorporated cheaply by considering interactions of amino-acid residues with “empty” (i.e. solvent) lattice sites.

Simulations with water term

Figure 3 (b) shows a typical protein designed and simulated with a pair-potential that includes explicitly calculated water interactions. As before, it is easy to design proteins that fold uniquely into a pre-designed native state. But, importantly, around the folding temperature the protein remains soluble, thus mimicking the biologically relevant situation where most proteins are soluble in their folded state. If we lower the temperature well below the folding temperature, we find that even these “water-soluble” proteins eventually aggregate.

Interestingly, figure 3 (b) also shows a small peak in the aggregation curve at temperatures just above T_f. This peak is due to a phenomenon where the unfolded form of the proteins starts to aggregate due to exposed hydrophobic patches (see also Figure 2 E). Such high-temperature aggregation has also been observed for some real proteins.

Discussion

Using simple pair-potentials between amino acids to design foldable model proteins leads to aggregation around the folding temperature. We show that by including explicitly calculated solvent interactions in the pair-potentials the designed proteins remain soluble at their folding temperatures and below.

Since most proteins should not aggregate under physiological conditions, a prerequisite to any modelling approach of pathological protein aggregation should be that the same

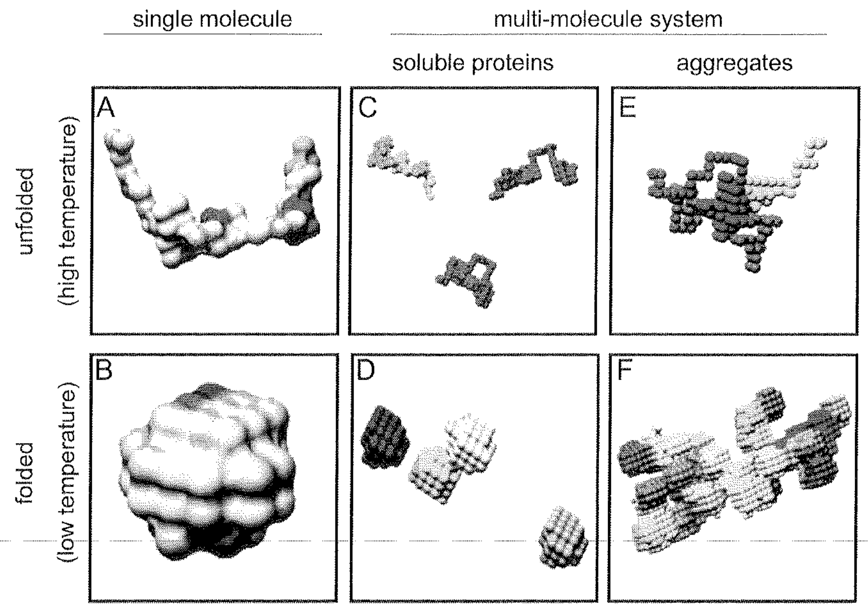


Figure 2: Schematic representation of aggregation behavior of lattice proteins.
This figure illustrates a common problem with existing lattice-protein models. Simulated as single molecules, proteins unfold at high temperatures (A), and fold into specific structures at low temperatures (B). In a multi-molecule system, several scenarios for folding, unfolding and aggregation are possible (C, D, E, F). As in nature, we would expect folded globular proteins to be soluble at low temperatures (D). However, existing models show a tendency to form large aggregates of folded proteins (F), due to (unphysically) strong attractions between hydrophilic residues.

where $\beta = \frac{1}{k_B T}$, N is the number of free chains in the simulation box before the move, V is the volume of the box, and μ the chemical potential. The volume was kept constant at $80 \times 80 \times 80$ lattice points and $exp(\mu\beta)$ was kept constant at 3×10^{-6} chains per lattice site, see Ref. (18) for more details.

Simulations without water term

Figure 3 (a) shows a typical example of a model protein designed with the Be-potential: this model protein starts to aggregate around the folding temperature. In fact, the majority ($> 95\%$) of proteins designed with the Be-potential aggregated during Monte Carlo simulations at temperatures around T_f. This may not be surprising as the design procedure does not bias against aggregation, whereas in nature there will be a strong evolutionary pressure against aggregation.

The aggregation propensity of the proteins was then tested by simulating the proteins at various temperatures around T_f at low concentrations (3.0×10^{-6} free molecules per lattice point) in the grand canonical ensemble.

To understand the cause of the aggregation, the formed aggregates were considered in more detail. Two features that are not compatible with naturally occurring proteins were found: 1) the designed proteins contained large hydrophobic patches on their surface 2) polar residues often clustered together. The Be-potential, and most other knowledge-based amino acid pair-potentials, assign negative interaction energies to polar-polar interactions, type of model would *not* predict aggregation of normal globular proteins. The knowledge-based pair-potential that we propose here has precisely this feature.

Protein aggregation, in specific amyloid formation, is associated with several neurodegenerative disease. It is therefore of considerable interest to model the onset of amyloid formation. Particularly, the early stages of amyloid formation are prohibitively expensive to simulate with an atomistic model. During these stages pre-fibrillar aggregates are formed of 10-50 protein molecules (19) that undergo significant structural changes over time.

In this work the lattice model provides a convenient and - importantly - cheap reference model to study the factors that make otherwise normal proteins aggregation prone. On the other hand, off-lattice coarse-grained protein models show promising results for studying peptide aggregation (6, 7, 9), and generally use pair-wise knowledge-based potentials developed on-lattice. Therefore such models could immediately benefit from the results obtained by this work, and become more appropriate for studying the competition between protein solubility and aggregation .

The calculation of the potential is simple to implement, and the concept of adding solvent interactions is easily adaptable to more complex interaction potentials and more detailed protein structure models. Moreover, the proteins designed with the potential that includes a solvent term tend to have a better-defined hydrophobic core. In fact, solvent exposure specific amino-acid substitution terms have long been recognised as a powerful tool in distant homology detection between proteins (20). We suggest that it may also be useful to include solvent-amino acid interactions in pair-potentials for structure prediction and design.

References

[1] Covell, D. G. and Jernigan, R. L. Conformations of folded proteins in restricted spaces. *Biochemistry*, 29(13):3287–3294, April 1990.

[2] Miyazawa, S. and Jernigan, R. L. A new substitution matrix for protein sequence searches based on contact frequencies in protein structures. *Protein Eng*, 6(3):267–278, April 1993.

[3] Sali, A., Shakhnovich, E. and Karplus, M. Kinetics of Protein Folding : A Lattice Model Study of the Requirements for Folding to the Native State. *Journal of Molecular Biology*, 235(5):1614–1638, 1994. ISSN 0022-2836. doi:DOI:10.1006/jmbi.1994.1110.

[4] Shakhnovich, E. I. Proteins with selected sequences fold into unique native conformation. *Phys. Rev. Lett.*, 72(24):3907–3910, June 1994. doi:10.1103/PhysRevLett.72.3907.

[5] Coluzza, I., Muller, H. G. and Frenkel, D. Designing refoldable model molecules. *Phys Rev E Stat Nonlin Soft Matter Phys*, 68(4 Pt 2):46703, October 2003.

[6] Harrison, P. M., Chan, H. S., Prusiner, S. B. and Cohen, F. E. Thermodynamics of model prions and its implications for the problem of prion protein folding. *J Mol Biol*, 286(2):593–606, February 1999. doi:10.1006/jmbi.1998.2497.

[7] Dima, R. I. and Thirumalai, D. Exploring protein aggregation and self-propagation using lattice models: phase diagram and kinetics. *Protein Sci*, 11(5):1036–1049, May 2002. doi:10.1110/ps.4220102.

[8] Marchut, A. J. and Hall, C. K. Side-chain interactions determine amyloid formation by model polyglutamine peptides in molecular dynamics simulations. *Biophys J*, 90(12):4574–4584, June 2006. doi:10.1529/biophysj.105.079269.

[9] Auer, S., Meersman, F., Dobson, C. M. and Vendruscolo, M. A generic mechanism of emergence of amyloid protofilaments from disordered oligomeric aggregates. *PLoS Comput Biol*, 4(11), November 2008. doi:10.1371/journal.pcbi.1000222.

[10] Dobson, C. M. Protein folding and misfolding. *Nature*, 426(6968):884–890, December 2003. doi:10.1038/nature02261.

[11] Sauer, U., Heinemann, M. and Zamboni, N. Genetics. Getting closer to the whole picture. *Science*, 316(5824):550–551, April 2007. doi:10.1126/science.1142502.

[12] Shakhnovich, E. I. and Gutin, A. M. Engineering of stable and fast-folding sequences of model proteins. *Proc Natl Acad Sci U S A*, 90(15):7195–7199, August 1993.

[13] Tanaka, S. and Scheraga, H. A. Medium- and long-range interaction parameters between amino acids for predicting three-dimensional structures of proteins. *Macromolecules*, 9(6):945–950, 1976.

[14] Skolnick, J., Jaroszewski, L., Kolinski, A. and Godzik, A. Derivation and testing of pair potentials for protein folding. When is the quasichemical approximation correct? *Protein Sci*, 6(3):676–688, March 1997. doi:10.1002/pro.5560060317.

[15] Abeln, S. and Frenkel, D. Accounting for protein-solvent contacts facilitates design of nonaggregating lattice proteins. *Biophysical journal*, 100(3):693–700, February 2011. ISSN 1542-0086. doi:10.1016/j.bpj.2010.11.088.

[16] Betancourt, M. R. and Thirumalai, D. Pair potentials for protein folding: choice of reference states and sensitivity of predicted native states to variations in the interaction schemes. *Protein Sci*, 8(2):361–369, February 1999.

[17] Marinari, E. and Parisi, G. Simulated Tempering: A New Monte Carlo Scheme. *EPL (Europhysics Letters)*, 19(6):451–458, 1992.

[18] Abeln, S. and Frenkel, D. Disordered flanks prevent peptide aggregation. *PLoS computational biology*, 4(12):e1000241, December 2008. ISSN 1553-7358. doi:10.1371/journal.pcbi.1000241.

[19] Orte, A., Birkett, N. R., Clarke, R. W., Devlin, G. L., Dobson, C. M. and Klenerman, D. Direct characterization of amyloidogenic oligomers by single-molecule fluorescence. *Proc Natl Acad Sci U S A*, 105(38):14424–14429, September 2008. doi:10.1073/pnas.0803086105.

[20] Shi, J., Blundell, T. L. and Mizuguchi, K. FUGUE: sequence-structure homology recognition using environment-specific substitution tables and structure-dependent gap penalties. *J Mol Biol*, 310(1):243–257, June 2001. doi:10.1006/jmbi.2001.4762.

Reactive Turing Machines*

Jos C. M. Baeten[†][‡] Bas Luttik[†][§] Paul van Tilburg[†]

February 28, 2012

Abstract

We propose reactive Turing machines (RTMs), extending classical Turing machines with a process-theoretical notion of interaction, and use it to define a notion of executable transition system. We show that every computable transition system with a bounded branching degree is simulated modulo divergence-preserving branching bisimilarity by an RTM, and that every effective transition system is simulated modulo the variant of branching bisimilarity that does not require divergence preservation. We conclude from these results that the parallel composition of (communicating) RTMs can be simulated by a single RTM. We prove that there exist universal RTMs modulo branching bisimilarity, but these essentially employ divergence to be able to simulate an RTM of arbitrary branching degree. We also prove that modulo divergence-preserving branching bisimilarity there are RTMs that are universal up to their own branching degree. Finally, we establish a correspondence between executability and finite definability in a simple process calculus.

1 Introduction

The Turing machine [19] is widely accepted as a computational model suitable for exploring the theoretical boundaries of computing. Motivated by the existence of universal Turing machines, many textbooks on the theory of computation present the Turing machine not just as a theoretical model to explain which functions are computable, but as an accurate conceptual model of the computer. There is, however, a well-known limitation to this view. A Turing machine operates from the assumptions that: (1) all input it needs for the computation is available on the tape from the very beginning; (2) it performs a terminating computation; and (3) it leaves the output on the tape at the very end. That is, a Turing machine computes a function, and thus it abstracts from two key ingredients of computing: *interaction* and *non-termination*. Nowadays, most computing systems are so-called *reactive systems* [12], systems that are generally not meant to terminate and consist of computing devices that interact with each other and with their environment.

Concurrency theory emerged from the early work of Petri [16] and has now developed into a mature theory of reactive systems. We mention three of its contributions particularly relevant for our work. Firstly, it installed the notion of transition system —a generalisation of the notion of finite-state automaton from classical automata theory— as the prime mathematical model to represent

*For an extended version of this article with proofs and constructions, see [4].

[†]CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

[‡]Department of Mathematics and Computer Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands

[§]Department of Computer Science, VU University Amsterdam, 1081 HV Amsterdam, The Netherlands discrete behaviour. Secondly, it offered the insight that language equivalence — the underlying equivalence in classical automata theory— is too coarse in a setting with interacting automata; instead one should consider automata up to some form of bisimilarity. Thirdly, it yielded many algebraic process calculi facilitating the formal specification and verification of reactive systems.

In Sect. 2 we propose a notion of *reactive* Turing machine (RTM), extending the classical notion of Turing machine with interaction in the style of concurrency theory. The extension consists of a facility to declare every transition to be either *observable*, by labelling it with an action symbol, or unobservable, by labelling it with τ. Typically, a transition labelled with an action symbol models an interaction of the RTM with its environment (or some other RTM), while a transition labelled with τ refers to an internal computation step. Thus, a conventional Turing machine can be regarded as a special kind of RTM in which all transitions are declared unobservable by labelling them with τ.

The semantic object associated with a conventional Turing machine is either the function that it computes, or the formal language that it accepts. The semantic object associated with an RTM is a behaviour, formally represented by a transition system. A function is said to be effectively computable if it can be computed by a Turing machine. By analogy, we say that a behaviour is effectively executable if it can be exhibited by an RTM. In concurrency theory, behaviours are usually considered modulo a suitable behavioural equivalence. In this paper we shall use *(divergence-preserving) branching bisimilarity* [10], which is the finest behavioural equivalence in Van Glabbeek's spectrum (see [8]).

In Sect. 3 we set out to investigate the expressiveness of RTMs up to divergence-preserving branching bisimilarity. We shall present an example of a behaviour that is not executable up to branching bisimilarity. Then, we establish that every computable transition system with a bounded branching degree can be simulated, up to divergence-preserving branching bisimilarity, by an RTM. If the divergence-preservation requirement is dropped, then every effective transition system can be simulated. These results will then allow us to conclude that the behaviour of a parallel composition of RTMs can be simulated on a single RTM.

In Sect. 4 we define a suitable notion of universality for RTMs and investigate the existence of universal RTMs. We shall find that, since bisimilarity is sensitive to branching, there are some subtleties pertaining to the branching degree bound associated with each RTM. Up to divergence-preserving branching bisimilarity, an RTM can at best simulate other RTMs with the same or a lower bound on their branching degree. If divergence-preservation is not required, however, then universal RTMs do exist.

In Sect. 5, we consider the correspondence between RTMs and a process calculus consisting of a few standard process-theoretic constructions. We establish that every executable behaviour is, again up to divergence-preserving branching bisimilarity, finitely definable in our calculus. Recursive specifications are the concurrency-theoretic counterparts of grammars in the theory of formal languages. Thus, the result in Sect. 5 may be considered as the process-theoretic version of the correspondence between Turing machines and unrestricted grammars.

In Sect. 6 we conclude the paper with a discussion of related work and some ideas for future work.

2 Reactive Turing Machines

We fix a finite set $\mathcal{A}$ of *action symbols* that we shall use to denote the observable events of a system. An unobservable event will be denoted with τ, assuming that $\tau \notin \mathcal{A}$; we shall henceforth denote the set $\mathcal{A} \cup \{\tau\}$ by $\mathcal{A}_\tau$. We also fix a finite set $\mathcal{D}$ of *data symbols*. We add to $\mathcal{D}$ a special symbol $\Box$ to denote a blank tape cell, assuming that $\Box \notin \mathcal{D}$; we denote the set $\mathcal{D} \cup \{\Box\}$ of *tape symbols* by $\mathcal{D}_\Box$.

Definition 2.1. A *reactive Turing machine* (RTM) $\mathcal{M}$ is a quadruple $(\mathcal{S}, \rightarrow, \uparrow, \downarrow)$ consisting of a finite set of *states* $\mathcal{S}$, a distinguished *initial state* $\uparrow \in \mathcal{S}$, a subset of *final states* $\downarrow \subseteq \mathcal{S}$, and a $(\mathcal{D}_\Box \times \mathcal{A}_\tau \times \mathcal{D}_\Box \times \{L, R\})$-labelled transition relation

$$\rightarrow \subseteq \mathcal{S} \times \mathcal{D}_\Box \times \mathcal{A}_\tau \times \mathcal{D}_\Box \times \{L, R\} \times \mathcal{S} \ .$$

An RTM is *deterministic* if $(s, d, a, e_1, M_1, t_1) \in \rightarrow$ and $(s, d, a, e_2, M_2, t_2) \in \rightarrow$ implies that $e_1 = e_2$, $t_1 = t_2$ and $M_1 = M_2$ for all $s, t_1, t_2 \in \mathcal{S}$, $d, e_1, e_2 \in \mathcal{D}_\Box$, $a \in \mathcal{A}_\tau$, and $M_1, M_2 \in \{L, R\}$, and, moreover, $(s, d, \tau, e_1, M_1, t_1) \in \rightarrow$ implies that there do not exist $a \neq \tau$, e_2, M_2, t_2 such that $(s, d, a, e_2, M_2, t_2) \in \rightarrow$

If $(s, d, a, e, M, t) \in \rightarrow$, we write $s \xrightarrow{a[d/e]M} t$. The intuitive meaning of such a transition is that whenever $\mathcal{M}$ is in state s and d is the symbol currently read by the tape head, then it may execute the action a, write symbol e on the tape (replacing d), move the read/write head one position to the left or one position to the right on the tape (depending on whether $M = L$ or $M = R$), and then end up in state t. RTMs extend conventional Turing machines by associating with every transition an element $a \in \mathcal{A}_\tau$. The symbols in $\mathcal{A}$ are thought of as denoting observable activities; a transition labelled with an action symbol in $\mathcal{A}$ will semantically be treated as observable. Observable transitions are used to model interactions of an RTM with its environment or some other RTM, as will be explained more in detail below when we introduce a notion of parallel composition for RTMs. The symbol τ is used to declare that a transition is unobservable. A classical Turing machine is an RTM in which all transitions are declared unobservable.

Example 2.2. Assume that $\mathcal{A} = \{c!d, c?d \mid c \in \{i, o\} \ \& \ d \in \mathcal{D}_\Box\}$. Intuitively, i and o are the input/output communication channels by which the RTM can interact with its environment. The action symbol $c!d$ ($c \in \{i, o\}$) then denotes the event that a datum d is sent by the RTM along channel c, and the action symbol $c?d$ ($c \in \{i, o\}$) denotes the event that a datum d is received by the RTM along channel c.

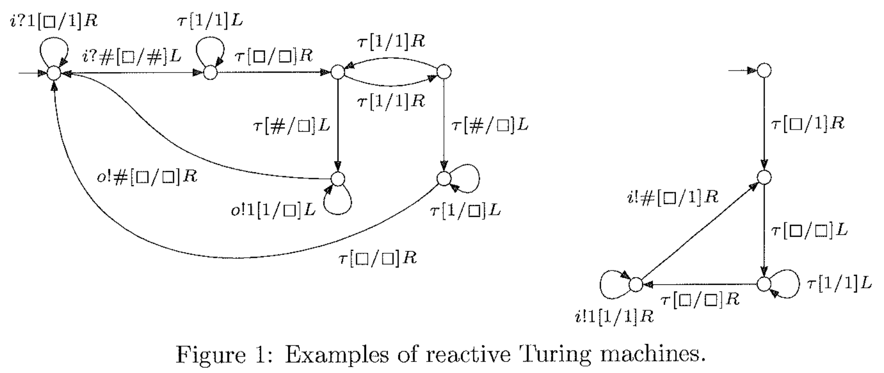


Figure 1: Examples of reactive Turing machines.

The left state-transition diagram in Fig. 1 specifies an RTM that first inputs a string, consisting of an arbitrary number of 1s followed by the symbol #, stores the string on the tape, and returns to the beginning of the string. Then, it performs a computation to determine if the number of 1s is odd or even. In the first case, it simply removes the string from the tape and returns to the initial state. In the second case, it outputs the entire string, removes it from the tape, and returns to the initial state. The right state-transition diagram in Fig. 1 outputs on channel i the infinite sequence $1\#11\#111\#\ldots\#1^n\#\ldots$ $(n \geq 1)$.

To formalise our intuitive understanding of the operational behaviour of RTMs we shall below associate with every RTM a transition system.

Definition 2.3. An $\mathcal{A}_\tau$*-labelled transition system* T is a quadruple $(\mathcal{S}, \rightarrow, \uparrow, \downarrow)$ consisting of a set of *states* $\mathcal{S}$, an *initial* state $\uparrow \in \mathcal{S}$, a subset $\downarrow \subseteq \mathcal{S}$ of *final* states, and an $\mathcal{A}_\tau$-labelled *transition relation* $\rightarrow \subseteq \mathcal{S} \times \mathcal{A}_\tau \times \mathcal{S}$. If $(s, a, t) \in \rightarrow$, we write $s \xrightarrow{a} t$. If s is a final state, i.e., $s \in \downarrow$, we write $s\downarrow$. The transition system T is *deterministic* if, for every state $s \in \mathcal{S}$ and for every $a \in \mathcal{A}_\tau$, $s \xrightarrow{a} s_1$ and $s \xrightarrow{a} s_2$ implies $s_1 = s_2$, and, moreover, $s \xrightarrow{\tau} s_1$ implies that there do not exist an action $a \neq \tau$ and a state s_2 such that $s \xrightarrow{a} s_2$.

With every RTM $\mathcal{M}$ we are going to associate a transition system $\mathcal{T}(\mathcal{M})$. The states of $\mathcal{T}(\mathcal{M})$ are the configurations of the RTM, consisting of a state of the RTM, its tape contents, and the position of the read/write head on the tape. We represent the tape contents by an element of $(\mathcal{D}_\Box)^*$, replacing precisely one occurrence of a tape symbol d by a *marked* symbol $\check{d}$, indicating that the read/write head is on this symbol. We denote by $\check{\mathcal{D}}_\Box = \{\check{d} \mid d \in \mathcal{D}_\Box\}$ the set of *marked* tape symbols; a *tape instance* is a sequence $\delta \in (\mathcal{D}_\Box \cup \check{\mathcal{D}}_\Box)^*$ such that δ contains exactly one element of $\check{\mathcal{D}}_\Box$. Note that we do not use δ exclusively for tape instances; we also use δ for sequences over $\mathcal{D}$. A tape instance thus is a finite sequence of symbols that represents the contents of a two-way infinite tape.

Henceforth, we shall not distinguish between tape instances that are equal modulo the addition or removal of extra occurrences of the symbol $\Box$ at the left or right extremes of the sequence. That is, we shall not distinguish tape instances δ_1 and δ_2 if $\Box^\omega\delta_1\Box^\omega = \Box^\omega\delta_2\Box^\omega$.

Definition 2.4. A *configuration* of an RTM $\mathcal{M} = (\mathcal{S}, \rightarrow, \uparrow, \downarrow)$ is a pair (s, δ) consisting of a state $s \in \mathcal{S}$, and a tape instance δ.

Our transition system semantics defines an $\mathcal{A}_\tau$-labelled transition relation on configurations such that an RTM-transition $s \xrightarrow{a[d/e]M} t$ corresponds with a-labelled transitions from configurations consisting of the RTM-state s and a tape instance in which some occurrence of d is marked. The transitions lead to configurations consisting of t and a tape instance in which the marked symbol d is replaced by e, and either the symbol to the left or to right of this occurrence of e is replaced by its marked version, according to whether $M = L$ or $M = R$. If e happens to be the first symbol and $M = L$, or the last symbol and $M = R$, then an additional blank symbol is appended at the left or right end of the tape instance, respectively, to model the movement of the head.

We introduce some notation to concisely denote the new placement of the tape head marker. Let δ be an element of $\mathcal{D}_\Box^*$. Then by $\delta^<$ we denote the element of $(\mathcal{D}_\Box \cup \check{\mathcal{D}}_\Box)^*$ obtained by placing the tape head marker on the right-most symbol of δ if it exists, and $\check{\Box}$ otherwise, i.e.,

$$\delta^< = \begin{cases} \zeta\check{d} & \text{if } \delta = \zeta d \quad (d \in \mathcal{D}_\Box, \zeta \in \mathcal{D}_\Box^*)\ , \text{ and} \\ \check{\Box} & \text{if } \delta = \varepsilon\ . \end{cases}$$

(We use ε to denote the empty sequence.) Similarly, by ${}^>\delta$ we denote the element of $(\mathcal{D}_\Box \cup \check{\mathcal{D}}_\Box)^*$ obtained by placing the tape head marker on the left-most symbol of δ if it exists, and $\check{\Box}$ otherwise, i.e.,

$${}^>\delta = \begin{cases} \check{d}\zeta & \text{if } \delta = d\zeta \quad (d \in \mathcal{D}_\Box, \zeta \in \mathcal{D}_\Box^*)\ , \text{ and} \\ \check{\Box} & \text{if } \delta = \varepsilon\ . \end{cases}$$

Definition 2.5. Let $\mathcal{M} = (\mathcal{S}, \rightarrow, \uparrow, \downarrow)$ be an RTM. The *transition system* $\mathcal{T}(\mathcal{M})$ *associated with* $\mathcal{M}$ is defined as follows:

1. its set of states is the set of all configurations of $\mathcal{M}$;
2. its transition relation $\rightarrow$ is the least relation satisfying, for all $a \in \mathcal{A}_\tau$, $d, e \in \mathcal{D}_\Box$ and $\delta_L, \delta_R \in \mathcal{D}_\Box^*$:

$$(s, \delta_L \check{d} \delta_R) \xrightarrow{a} (t, \delta_L{}^{<} e \delta_R) \text{ iff } s \xrightarrow{a[d/e]L} t \text{ , and}$$
$$(s, \delta_L \check{d} \delta_R) \xrightarrow{a} (t, \delta_L e {}^{>} \delta_R) \text{ iff } s \xrightarrow{a[d/e]R} t \text{ ;}$$

3. its initial state is the configuration $(\uparrow, \check{\Box})$; and
4. its set of final states is the set of terminating configurations $\{(s, \delta) \mid s\downarrow\}$.

Turing introduced his machines to define the notion of *effectively computable function*. By analogy, our notion of RTM can be used to define a notion of *effectively executable behaviour*.

Definition 2.6. A transition system is *executable* if it is associated with an RTM.

Parallel composition. To illustrate how RTMs are suitable to model a form of interaction, we shall now define on RTMs a notion of parallel composition, equipped with a simple form communication. (We are not trying to define the most general or most suitable notion of parallel composition for RTMs here; the purpose of the notion of parallel composition defined here is just to illustrate how RTMs may run in parallel and interact.) Let $\mathcal{C}$ be a finite set of *channels* for the communication of data symbols between one RTM and another. Intuitively, $c!d$ stands for the action of sending datum d along channel c, while $c?d$ stands for the action of receiving datum d along channel c.

First, we define a notion of parallel composition on transition systems. Let $T_1 = (\mathcal{S}_1, \rightarrow_1, \uparrow_1, \downarrow_1)$ and $T_2 = (\mathcal{S}_2, \rightarrow_2, \uparrow_2, \downarrow_2)$ be transition systems, and let $\mathcal{C}' \subseteq \mathcal{C}$. The *parallel composition* of T_1 and T_2 is the transition system $[T_1 \parallel T_2]_{\mathcal{C}'} = (\mathcal{S}, \rightarrow, \uparrow, \downarrow)$, with $\mathcal{S}$, $\rightarrow$, $\uparrow$ and $\downarrow$ defined by

1. $\mathcal{S} = \mathcal{S}_1 \times \mathcal{S}_2$;
2. $(s_1, s_2) \xrightarrow{a} (s_1', s_2')$ iff $a \in \mathcal{A}_\tau - \{c!d, c?d \mid c \in \mathcal{C}', d \in \mathcal{D}_\Box\}$ and either
   (a) $s_1 \xrightarrow{a} s_1'$ and $s_2 = s_2'$, or $s_2 \xrightarrow{a} s_2'$ and $s_1 = s_1'$, or
   (b) $a = \tau$ and either $s_1 \xrightarrow{c!d} s_1'$ and $s_2 \xrightarrow{c?d} s_2'$, or $s_1 \xrightarrow{c?d} s_1'$ and $s_2 \xrightarrow{c!d} s_2'$ for some $c \in \mathcal{C}'$ and $d \in \mathcal{D}_\Box$;
3. $\uparrow = (\uparrow_1, \uparrow_2)$; and
4. $\downarrow = \{(s_1, s_2) \mid s_1 \in \downarrow_1 \,\&\, s_2 \in \downarrow_2\}$.

Definition 2.7. Let $\mathcal{M}_1 = (\mathcal{S}_1, \rightarrow_1, \uparrow_1, \downarrow_1)$ and $\mathcal{M}_2 = (\mathcal{S}_2, \rightarrow_2, \uparrow_2, \downarrow_2)$ be RTMs, and let $\mathcal{C}' \subseteq \mathcal{C}$; by $[\mathcal{M}_1 \parallel \mathcal{M}_2]_{\mathcal{C}'}$ we denote the *parallel composition* of $\mathcal{M}_1$ and $\mathcal{M}_2$. The transition system $\mathcal{T}([\mathcal{M}_1 \parallel \mathcal{M}_2]_{\mathcal{C}'})$ associated with the parallel composition $[\mathcal{M}_1 \parallel_{\mathcal{C}} \mathcal{M}_2]_{\mathcal{C}'}$ of $\mathcal{M}_1$ and $\mathcal{M}_2$ is the parallel composition of the transition systems associated with $\mathcal{M}_1$ and $\mathcal{M}_2$, i.e., $\mathcal{T}([\mathcal{M}_1 \parallel \mathcal{M}_2]_{\mathcal{C}'}) = [\mathcal{T}(\mathcal{M}_1) \parallel \mathcal{T}(\mathcal{M}_2)]_{\mathcal{C}'}$.

Example 2.8. Let $\mathcal{A}$ be as in Example 2.2, let $\mathcal{M}$ denote the left-hand side RTM in Fig. 1, and let $\mathcal{E}$ denote the right-hand side RTM in Fig. 1. Then the parallel composition $[\mathcal{M} \parallel \mathcal{E}]_i$ exhibits the behaviour of outputting, along channel o, the string 11#1111#$\cdots$#1^n# ($n \geq 2$, n even).

Behavioural equivalence. In automata theory, Turing machines that compute the same function or accept the same language are generally considered equivalent. In fact, functional or language equivalence is underlying many of the standard notions and results in automata theory. Perhaps most notably, a *universal* Turing machine is a Turing machine that, when started with the code of some Turing machine on its tape, simulates this machine up to functional or language equivalence. A result from concurrency theory is that functional and language equivalence are arguably too coarse for reactive systems, because they abstract from all moments of choice (see, e.g., [1]). In concurrency theory many alternative behavioural equivalences have been proposed; we refer to [8] for a classification.

The results about RTMs that are obtained in the remainder of this paper are modulo *branching bisimilarity* [10], which is the finest behavioural equivalence in Van Glabbeek's linear time – branching time spectrum [8]. We shall consider both the divergence-insensitive and the divergence-preserving variant. (The divergence-preserving variant is called *branching bisimilarity with explicit divergence* in [10, 8], but in this paper we prefer the term *divergence-preserving* branching bisimilarity.)

We proceed to define the behavioural equivalences that we shall employ in this paper to compare transition systems. Let $\longrightarrow$ be an $\mathcal{A}_\tau$-labelled transition relation on a set $\mathcal{S}$, and let $a \in \mathcal{A}_\tau$; we write $s \xrightarrow{(a)} t$ if $s \xrightarrow{a} t$ or $a = \tau$ and $s = t$. Furthermore, we denote the transitive closure of $\xrightarrow{\tau}$ by $\longrightarrow^{+}$, and we denote the reflexive-transitive closure of $\xrightarrow{\tau}$ by $\longrightarrow^{*}$.

Definition 2.9. Let $T_1 = (\mathcal{S}_1, \longrightarrow_1, \uparrow_1, \downarrow_1)$ and $T_2 = (\mathcal{S}_2, \longrightarrow_2, \uparrow_2, \downarrow_2)$ be transition systems. A *branching bisimulation* from T_1 to T_2 is a binary relation $\mathcal{R} \subseteq \mathcal{S}_1 \times \mathcal{S}_2$ and, for all states s_1 and s_2, $s_1 \mathrel{\mathcal{R}} s_2$ implies

1. if $s_1 \xrightarrow{a}_1 s_1'$, then there exist $s_2', s_2'' \in \mathcal{S}_2$ such that $s_2 \longrightarrow_2^{*} s_2'' \xrightarrow{(a)}_2 s_2'$, $s_1 \mathrel{\mathcal{R}} s_2''$ and $s_1' \mathrel{\mathcal{R}} s_2'$;
2. if $s_2 \xrightarrow{a}_2 s_2'$, then there exist $s_1', s_1'' \in \mathcal{S}_1$ such that $s_1 \longrightarrow_1^{*} s_1'' \xrightarrow{(a)}_1 s_1'$, $s_1'' \mathrel{\mathcal{R}} s_2$ and $s_1' \mathrel{\mathcal{R}} s_2'$;
3. if $s_1\downarrow_1$, then there exists s_2' such that $s_2 \longrightarrow_2^{*} s_2'$, $s_1 \mathrel{\mathcal{R}} s_2'$ and $s_2'\downarrow_2$; and
4. if $s_2\downarrow_2$, then there exists s_1' such that $s_1 \longrightarrow_1^{*} s_1'$, $s_1' \mathrel{\mathcal{R}} s_2$ and $s_1'\downarrow_1$.

The transition systems T_1 and T_2 are *branching bisimilar* (notation: $T_1 \leftrightarrows_{\mathrm{b}} T_2$) if there exists a branching bisimulation from T_1 to T_2 such that $\uparrow_1 \mathrel{\mathcal{R}} \uparrow_2$.

A branching bisimulation $\mathcal{R}$ from T_1 to T_2 is *divergence-preserving* if, for all states s_1 and s_2, $s_1 \mathrel{\mathcal{R}} s_2$ implies

5. if there exists an infinite sequence $(s_{1,i})_{i \in \mathbb{N}}$ such that $s_1 = s_{1,0}$, $s_{1,i} \xrightarrow{\tau} s_{1,i+1}$ and $s_{1,i} \mathrel{\mathcal{R}} s_2$ for all $i \in \mathbb{N}$, then there exists a state s_2' such that $s_2 \longrightarrow^{+} s_2'$ and $s_{1,i} \mathrel{\mathcal{R}} s_2'$ for some $i \in \mathbb{N}$; and
6. if there exists an infinite sequence $(s_{2,i})_{i \in \mathbb{N}}$ such that $s_2 = s_{2,0}$, $s_{2,i} \xrightarrow{\tau} s_{2,i+1}$ and $s_1 \mathrel{\mathcal{R}} s_{2,i}$ for all $i \in \mathbb{N}$, then there exists a state s_1' such that $s_1 \longrightarrow^{+} s_1'$ and $s_1' \mathrel{\mathcal{R}} s_{2,i}$ for some $i \in \mathbb{N}$.

The transition systems T_1 and T_2 are *divergence-preserving branching bisimilar* (notation: $T_1 \leftrightarrows_{\mathrm{b}}^{\Delta} T_2$) if there exists a divergence-preserving branching bisimulation from T_1 to T_2 such that $\uparrow_1 \mathrel{\mathcal{R}} \uparrow_2$.

The notions of branching bisimilarity and divergence-preserving branching bisimilarity originate with [10]. The particular divergence conditions we use to define divergence-preserving branching bisimulations here are discussed in [9], where it is also proved that divergence-preserving branching bisimilarity is an equivalence.

An unobservable transition of an RTM, i.e., a transition labelled with τ, may be thought of as an internal computation step. Divergence-preserving branching bisimilarity allows us to abstract from internal computations as long as they do not discard the option to execute a certain behaviour. The following notion will be used as a technical tool in the remainder of the paper.

Definition 2.10. Given some transition system T, an *internal computation from state s to s'* is a sequence of states $s_1, \ldots, s_n$ in T such that $s = s_1 \xrightarrow{\tau} \ldots \xrightarrow{\tau} s_n = s'$. An internal computation is called *deterministic* iff, for every state s_i $(1 \leq i < n)$, $s_i \xrightarrow{a} s_i{}'$ implies $a = \tau$ and $s_i{}' = s_{i+1}$. If $s_1, \ldots, s_n$ is a deterministic internal computation from s to s', then we refer to the set

$$\{s_1, \ldots, s_n\}$$

as the set of *intermediate states* of the deterministic internal computation.

Proposition 2.11. Let T be a transition system and let s and t be two states in T. If there exists a deterministic internal computation from s to s', then all its intermediate states are related by the maximal divergence-preserving branching bisimulation on T.

3 Expressiveness of RTMs

Our notion of RTMs defines the class of executable transition systems. In this section, we investigate the expressiveness of this notion up to branching bisimilarity, using the notions of effective transition system and computable transition system as a tool.

In Sect. 3.1, we recall the definitions of effective transition system and computable transition system, observe that executable transition systems are necessarily computable and, moreover, have a bounded branching degree. Then, we proceed to consider executable transition systems modulo (divergence-preserving) branching bisimilarity. We present an example of a (non-effective) transition system that is not executable up to branching bisimilarity. Finally, we shall adapt a result by Phillips [17] showing that every effective transition system is branching bisimilar to a computable transition system with branching degree at most two. Phillips' proof introduces divergence, and we shall present an example illustrating that this is unavoidable.

In Sect. 3.2, we establish, for an arbitrary boundedly branching computable transition system, that an RTM exists that simulates the behaviour represented by the transition system up to divergence-preserving branching bisimilarity. Thus, we confirm the expressiveness of RTMs: modulo divergence-preserving branching bisimilarity, which is the finest behavioural equivalence in van Glabbeek's spectrum [8], the class of executable transition systems coincides with the class of boundedly branching computable transition systems. Moreover, in view of Phillips' result, we obtain as a corollary that every effective transition system can be simulated up to branching bisimilarity at the cost of introducing divergence.

We shall obtain two more interesting corollaries from the result in Sect. 3.2. Firstly, if a transition system is deterministic, then, by our assumption that the set $\mathcal{A}$ of action symbols is finite, it is clearly boundedly branching; hence, every deterministic computable transition systems can be simulated, up to divergence-preserving branching bisimilarity, by a deterministic RTM. Secondly, the parallel composition of boundedly branching computable transition systems is clearly boundedly branching and computable; hence, a parallel composition of RTMs can be simulated, up to divergence-preserving branching bisimilarity, by a single RTM.

3.1 Effective and Computable Transition Systems

Let $T = (\mathcal{S}, \rightarrow, \uparrow, \downarrow)$ be a transition system; the mapping *out* : $\mathcal{S} \rightarrow 2^{\mathcal{A}_\tau \times \mathcal{S}}$ associates with every state its set of outgoing transitions, i.e., for all $s \in \mathcal{S}$,

$$out(s) = \{(a, t) \mid s \xrightarrow{a} t\} \ ,$$

and *fin* denotes the characteristic function of $\downarrow$. We shall restrict our attention in this section to *finitely branching* transition systems, i.e., transition systems for which it holds that $out(s)$ is finite for all states s. (The restriction is convenient for our definition of computable transition system below, but it is otherwise unimportant since in all our results about computable transition systems we shall further restrict to boundedly branching transition systems. The restriction is not necessary for the definition of effective transition system, and, in fact, our results about effective transition systems do not depend on it.)

Definition 3.1. Let $T = (\mathcal{S}, \rightarrow, \uparrow, \downarrow)$ be an $\mathcal{A}_\tau$-labelled finitely branching transition system. We say that T is *effective* if *out* and *fin* are recursively enumerable sets. We say that T is *computable* if *out* and *fin* are recursive functions.

The notion of effective transition system originates with Boudol [6]. For the notion of computable transition system we have reformulated the definition in [2] to suit our needs. We temporarily step over the fact that, in order for the formal theory of recursiveness to make sense, we need suitable codings into natural numbers of the concepts involved. For now, we rely on the intuition of the reader. (The reader may already want to consult [18, §1.10] for more explanations.) A transition system is effective iff there exist an algorithm that enumerates its transitions and an algorithm that enumerates its final states. Similarly, a transition system is computable iff there exists an algorithm that lists the outgoing transitions of a state and also determines if it is final.

Proposition 3.2. The transition system associated with an RTM is computable.

Hence, unsurprisingly, if a transition system is not computable, then it is not executable either. It is easy to define transition systems that are not computable, so there exist behaviours that are not executable. The following example takes this a little further and illustrates that there exist behaviours that are not even executable up to branching bisimilarity.

Example 3.3. (In this and later examples, we denote by φ_x the partial recursive function with index $x \in \mathbb{N}$ in some exhaustive enumeration of partial recursive functions, see, e.g., [18].) Assume that $\mathcal{A} = \{a, b, c\}$ and consider the $\mathcal{A}$-labelled transition system $T_0 = (\mathcal{S}_0, \rightarrow_0, \uparrow_0, \downarrow_0)$ with $\mathcal{S}_0$, $\rightarrow_0$, $\uparrow_0$ and $\downarrow_0$ defined by

$$\begin{aligned}
\mathcal{S}_0 &= \{s, t, u, v, w\} \cup \{s_x \mid x \in \mathbb{N}\} \ , \\
\rightarrow_0 &= \{(s, a, t), (t, a, t), (t, b, v), (s, a, u), (u, a, u), (u, c, w)\} \\
&\qquad \cup \{(s, a, s_0)\} \cup \{(s_x, a, s_{x+1}) \mid x \in \mathbb{N}\} \\
&\qquad \cup \{(s_x, a, t), (s_x, a, u) \mid \varphi_x \text{ is a total function}\} \ , \\
\uparrow_0 &= s \ , \text{ and} \\
\downarrow_0 &= \{v, w\} \ .
\end{aligned}$$

The transition system is depicted in Fig. 2.

To argue that T_0 is not executable up to branching bisimilarity, we proceed by contradiction. Suppose that T_0 is executable up to branching bisimilarity. Then T_0 is branching bisimilar to a computable transition system T_0'. Then, in T_0', the set of states reachable by a path that contains exactly x a-transitions ($x \in \mathbb{N}$) and from

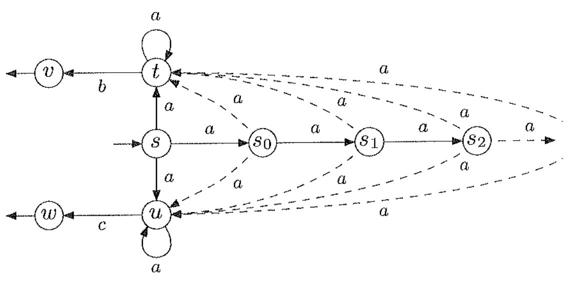


Figure 2: The transition system T_0.

which both a b- and a c-transition are still reachable, is recursively enumerable. It follows that the set of states in T_0' branching bisimilar to s_x ($x \in \mathbb{N}$) is recursively enumerable. But then, since the problem of deciding whether from some state in T_0' there is a path containing exactly one a-transition and one b-transition such that the a-transition precedes the b-transition, is also recursively enumerable, it follows that the problem of deciding whether φ_x is a total function must be recursively enumerable too, which it is not. We conclude that T_0 is not executable up to branching bisimilarity. Incidentally, note that the language associated with T_0 is $\{a^n b, a^n c \mid n \geq 1\}$, which *is* recursively enumerable (it is even context-free).

Phillips associates, in [17], with every effective transition system a *branching bisimilar* computable transition system of which, moreover, every state has a branching degree of at most 2. (Phillips actually establishes weak bisimilarity, but it is easy to see that branching bisimilarity holds.)

Definition 3.4. Let $T = (\mathcal{S}, \to, \uparrow, \downarrow)$ be a transition system, and let B be a natural number. We say that T has a branching degree *bounded by* B if $|out(s)| \leq B$, for every state $s \in \mathcal{S}$. We say that T is *boundedly branching* if there exists $B \in \mathbb{N}$ such that the branching degree of T is bounded by B.

Proposition 3.5 (Phillips). For every effective transition system T there exists a boundedly branching computable transition system T' such that $T \leftrightarrow_{\mathrm{b}} T'$.

A crucial insight in Phillips' proof is that a divergence (i.e., an infinite sequence of τ-transitions) can be exploited to simulate a state of which the set of outgoing transitions is recursively enumerable, but not recursive. The following example, inspired by [7], shows that introducing divergence is unavoidable.

Example 3.6. Assume that $\mathcal{A} = \{a, b\}$, and consider the transition system $T_1 = (\mathcal{S}_1, \to_1, \uparrow_1, \downarrow_1)$ with $\mathcal{S}_1$, $\to_1$, $\uparrow_1$ and $\downarrow_1$ defined by

$$\begin{aligned}
\mathcal{S}_1 &= \{s_{1,x},\ t_{1,x} \mid x \in \mathbb{N}\}\ , \\
\to_1 &= \{(s_{1,x}, a, s_{1,x+1}) \mid x \in \mathbb{N}\} \cup \{(s_{1,x}, b, t_{1,x}) \mid x \in \mathbb{N}\}\ , \\
\uparrow_1 &= s_{1,0}\ \text{, and} \\
\downarrow_1 &= \{t_{1,x} \mid \varphi_x(x) \text{ converges}\}\ .
\end{aligned}$$

The transition system is depicted in Fig. 3.

Now, suppose that T_2 is a transition system such that $T_1 \leftrightarrow_{\mathrm{b}}^{\Delta} T_2$, as witnessed by some divergence-preserving branching bisimulation relation $\mathcal{R}$; we argue that T_2 is not computable by deriving a contradiction from the assumption that it is.

Clearly, since T_1 does not admit infinite sequences of τ-transitions, if $\mathcal{R}$ is divergence-preserving, then T_2 does not admit infinite sequences of τ-transitions either. It follows that if $s_1\ \mathcal{R}\ s_2$, then there exists a state s_2' in T_2 such that $s_2 \longrightarrow_2^* s_2$, $s_1\ \mathcal{R}\ s_2'$, and $s_2' \xrightarrow{\tau}\!\!\!\!\not\ \ $. Moreover, since T_2 is computable and does not admit infinite sequences of consecutive τ-transitions, a state s_2' satisfying the

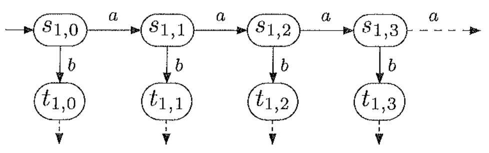


Figure 3: The transition system T_1.

aforementioned properties is produced by the algorithm that, given a state of T_2, selects an enabled τ-transition and recurses on the target of the transition until it reaches a state in which no τ-transitions are enabled.

But then we also have an algorithm that determines if $\varphi_x(x)$ converges:

1. it starts from the initial state $\uparrow_2$ of T_2;
2. it runs the algorithm to find a state without outgoing τ-transitions, and then it repeats the following steps x times:
   (a) execute the a-transition enabled in the reached state;
   (b) run the algorithm to find a state without outgoing τ-transitions again;

   since $\uparrow_1 \mathcal{R} \uparrow_2$, this yields a state $s_{2,x}$ in T_2 such that $s_{1,x} \mathcal{R}\ s_{2,x}$;
3. it executes the b-transition that must be enabled in $s_{2,x}$, followed, again, by the algorithm to find a state without outgoing τ-transitions; this yields a state $t_{2,x}$, without any outgoing transitions, such that $t_{1,x} \mathcal{R}\ t_{2,x}$.

From $t_{1,x} \mathcal{R}\ t_{2,x}$ it follows that $t_{2,x} \in \downarrow_2$ iff $\varphi_x(x)$ converges, so the problem of deciding whether $\varphi_x(x)$ converges has been reduced to the problem of deciding whether $t_{2,x} \in \downarrow_2$. Since it is undecidable if $\varphi_x(x)$ converges, it follows that $\downarrow_2$ is not recursive, which contradicts our assumption that T_2 is computable.

3.2 Simulation of Boundedly Branching Computable Transition Systems

Let $T = (\mathcal{S}_T, \rightarrow_T, \uparrow_T, \downarrow_T)$ be a boundedly branching computable transition system, say with branching degree bounded by B. It is reasonably straightforward to construct an RTM $\mathsf{Sim} = (\mathcal{S}_{\mathsf{Sim}}, \rightarrow_{\mathsf{Sim}}, \uparrow_{\mathsf{Sim}}, \downarrow_{\mathsf{Sim}})$, which we call the *simulator* for T, such that $\mathcal{T}(\mathsf{Sim}) \leftrightarrow_b^{\Delta} T$. The construction is detailed in [4].

Theorem 3.7. For every boundedly branching computable transition system T there exists an RTM Sim such that $T \leftrightarrow_b^{\Delta} \mathcal{T}(\mathsf{Sim})$.

Recall that, by Proposition 3.5, every effective transition system is branching bisimilar to a computable transition system with branching degree bounded by 2. According to Theorem 3.7, the resulting transition can be simulated with an RTM up to divergence-preserving branching bisimilarity. We can conclude that RTMs can simulate effective transition systems up to branching bisimilarity, but, in view of Example 3.6, not in a divergence-preserving manner.

Corollary 3.8. For every effective transition system T there exists a reactive Turing machine Sim such that $T \leftrightarrow_b \mathcal{T}(\mathsf{Sim})$.

Note that if T is deterministic, then $|out(s)| \leq |\mathcal{A}_\tau|$ for every state s in T, so every deterministic transition system is, in fact, boundedly branching. All computations involved in the simulation of T are deterministic; if Sim is non-deterministic, then this is due to a state of which the set of outgoing transitions includes a transition with some action a more than once. It follows that a deterministic computable transition system can be simulated up to divergence-preserving branching bisimilarity by a deterministic RTM. The following corollary to Theorem 3.7 summarises the argument.

Corollary 3.9. For every deterministic computable transition system T there exists a deterministic RTM $\mathcal{M}$ such that $\mathcal{T}(\mathcal{M}) \leftrightarroweq_{\mathrm{b}}^{\Delta} T$.

Using Theorem 3.7 we can now also establish that a parallel composition of RTMs can be simulated, up to divergence-preserving branching bisimilarity, by a single RTM. To this end, note that the transition systems associated with RTMs are boundedly branching and computable. Further note that the parallel composition of boundedly branching computable transition systems is again computable. It follows that the transition system associated with a parallel composition of RTMs is boundedly branching and computable, and hence, by Theorem 3.7, there exists an RTM that simulates it up to divergence-preserving branching bisimilarity. Thus we get the following corollary.

Corollary 3.10. For every pair of RTMs $\mathcal{M}_1$ and $\mathcal{M}_2$ and for every set of communication channels $\mathcal{C}$ there is an RTM $\mathcal{M}$ such that $\mathcal{T}(\mathcal{M}) \leftrightarroweq_{\mathrm{b}}^{\Delta} \mathcal{T}([\mathcal{M}_1 \parallel \mathcal{M}_2]_{\mathcal{C}})$.

4 Universality

Recall that a *universal Turing machine* is a Turing machine that can simulate an arbitrary Turing machine on arbitrary input. The assumptions are that a finite description of the to be simulated Turing machine (e.g., a Gödel number, see [18]) as well as its input are available on the tape of the universal Turing machine, and the simulation is up to functional or language equivalence. We adapt this scheme in two ways. Firstly, we let the simulation start by inputting the description of an arbitrary RTM $\mathcal{M}$ along some dedicated channel u, rather than assuming its presence on the tape right from the start. This is both conceptually desirable —for our aim is to give interaction a formal status—, and technically necessary —for in the semantics of RTMs we have assumed that the tape is initially empty. Secondly, we require the behaviour of $\mathcal{M}$ to be simulated up to divergence-preserving branching bisimilarity.

Thus, we arrive at the following tentative definitions. For an arbitrary RTM $\mathcal{M}$, denote by $\overline{\mathcal{M}}$ a deterministic RTM with no other behaviour than outputting a Gödel number $\ulcorner\mathcal{M}\urcorner$ of $\mathcal{M}$ in an appropriate representation along channel u after which it halts its unique final state. A *universal* RTM is then an RTM $\mathcal{U}$ such that, for every RTM $\mathcal{M}$, the parallel composition $\left[\mathcal{U} \parallel \overline{\mathcal{M}}\right]_{\{u\}}$ simulates $\mathcal{T}(\mathcal{M})$.

Although such a universal RTM $\mathcal{U}$ exists up to branching bisimilarity, as we shall see below, it does not exist up to divergence-preserving branching bisimilarity. To see this, note that the transition system associated with any particular RTM $\mathcal{U}$ has a branching degree that is bounded by some natural number B. It can then be established that, up to divergence-preserving branching bisimilarity, that $\mathcal{U}$ can only simulate RTMs with a branching degree bounded by B. The argument is formalised in the following proposition; see [4] for a proof.

Proposition 4.1. There does not exist an RTM $\mathcal{U}$ such that for all RTMs $\mathcal{M}$ it holds that $\left[\mathcal{U} \parallel \overline{\mathcal{M}}\right]_{\{u\}} \leftrightarroweq_{\mathrm{b}}^{\Delta} \mathcal{T}(\mathcal{M})$.

If we insist on simulation up to divergence-preserving branching bisimilarity, then we need to relax the notion of universality.

Definition 4.2. Let B be a natural number. An RTM $\mathcal{U}_B$ is *universal up to B* if for every RTM $\mathcal{M}$ of which the associated transition system $\mathcal{T}(\mathcal{M})$ has a branching degree bounded by B it holds that $\mathcal{T}(\mathcal{M}) \leftrightarroweq_{\mathrm{b}}^{\Delta} \left[\overline{\mathcal{M}} \parallel \mathcal{U}_B\right]_{\{u\}}$.

The construction of the simulator for a transition system of which the branching degree is bounded by B can be adapted to get the definition of an RTM $\mathcal{U}_B$ that is universal up to B. (See also [4] for this adaptation.) It suffices to slightly modify the initialisation part. Instead of writing the codes of the functions *out* and *fin* and the initial state directly on the tape, the *initialisation fragment* of $\mathcal{U}_B$ receives the code $\ulcorner\mathcal{M}\urcorner$ of an arbitrary $\mathcal{M}$ along some dedicated channel u. Then, it recursively computes the codes of the functions *out* and *fin*, and the initial state of $\mathcal{T}(\mathcal{M})$ and stores these on the tape.

Theorem 4.3. For all RTMs $\mathcal{M}$ with a branching degree bounded by B, it holds that $\mathcal{T}(\mathcal{M}) \leftrightarrow_{\mathrm{b}}^{\Delta} [\mathcal{M} \parallel \mathcal{U}_B]_{\{u\}}$.

At the expense of introducing divergence it is possible to define a universal RTM. Recall that, by Proposition 3.5, every effective transition system is branching bisimilar to a boundedly branching transition system. The proof of this result exploits a trick, first described in [2] and adapted by Phillips in [17], to use a divergence with (infinitely many) states of at most a branching degree of 2 to simulate, up to branching bisimilarity, a state with arbitrary (even countably infinite) branching degree.

Corollary 4.4. There exists an RTM $\mathcal{U}$ such that $\mathcal{T}(\mathcal{M}) \leftrightarrow_{\mathrm{b}} [\mathcal{M} \parallel \mathcal{U}]_{\{u\}}$ for every RTM $\mathcal{M}$.

5 A Process Calculus

We have presented reactive Turing machines and studied the ensued notion of executable behaviour modulo (divergence-preserving) branching bisimilarity. In process theory, behaviour is usually specified in some process calculus. In this section, we shall present a simple process calculus with only standard process-theoretic notions and establish that every executable behaviour can be defined with a finite specification in our calculus up to divergence-preserving branching bisimilarity. The process calculus we shall define below is closest to value-passing CCS [15] for a finite set of data. It deviates from value-passing CCS in that it combines parallel composition and restriction in one construct (i.e., it includes the special form of parallel composition already presented in Sect. 2), omits the relabelling construction, and distinguishes successful and unsuccessful termination. Our process calculus may also be viewed as a special instance of the fragment of TCP_τ, excluding sequential composition (see [1]).

We shall briefly introduce the syntax and informally describe its operational semantics. We refer to the textbook [1] for an elaborate treatment. Recall the finite sets $\mathcal{C}$ of channels and $\mathcal{D}_\Box$ of data on which the notion of parallel composition defined in Sect. 2 is based. For every subset $\mathcal{C}'$ of $\mathcal{C}$ we define a special set of actions $\mathcal{I}_{\mathcal{C}'}$ by:

$$\mathcal{I}_{\mathcal{C}'} = \{c?d, c!d \mid d \in \mathcal{D}_\Box, c \in \mathcal{C}'\} \ .$$

The actions $c?d$ and $c!d$ denote the events that a datum d is received or sent along *channel* c. Furthermore, let $\mathcal{N}$ be a countably infinite set of names. The set of *process expressions* $\mathcal{P}$ is generated by the following grammar ($a \in \mathcal{A}_\tau \cup \mathcal{I}_{\mathcal{C}}, N \in \mathcal{N}, \mathcal{C}' \subseteq \mathcal{C}$):

$$p \ ::= \ \mathbf{0} \mid \mathbf{1} \mid a.p \mid p+p \mid [p \parallel p]_{\mathcal{C}'} \mid N \ .$$

Let us briefly comment on the operators in this syntax. The constant $\mathbf{0}$ denotes *deadlock*, the unsuccessfully terminated process. The constant $\mathbf{1}$ denotes *skip*, the successfully terminated process. For each action $a \in \mathcal{A}_\tau \cup \mathcal{I}_C$ there is a unary operator $a.$ denoting action prefix; the process denoted by $a.p$ can do an a-transition to the process denoted by p. The binary operator $+$ denotes *alternative composition* or non-deterministic *choice*. The binary operator $[\_ \parallel \_]_{C'}$ denotes the special kind of *parallel composition* that we have also defined on RTMs. It enforces communication along the channels in C', and communication results in τ. (By including the restricted kind of parallel composition, we deviate from the definition of TCP$_\tau$ discussed in [1], but we note that our notion of parallel composition is definable with the operations $\parallel$, $\partial_\_(\_)$ and $\tau_\_(\_)$ of TCP$_\tau$ in [1].)

A *recursive specification* E is a set of equations of the form: $N \stackrel{\text{def}}{=} p$, with as left-hand side a name N and as right-hand side a TCP$_\tau$ process expression p. It is required that a recursive specification E contains, for every $N \in \mathcal{N}$, at most one equation with N as left-hand side; this equation will be referred to as the *defining equation* for N in $\mathcal{N}$. Furthermore, if some name occurs in the right-hand side of some defining equation, then the recursive specification must include a defining equation for it.

Let E be a recursive specification, and let p be a process expression. We say that p is *E-interpretable* if all occurrences of names in p have a defining equation in E. There is a standard method to associate with p a transition system $\mathcal{T}_E(p)$. The details can be found, e.g., in [1].

In [4] we present the details of a construction that associates with an arbitrary RTM a TCP$_\tau$ recursive specification that defines its behaviour up to divergence-preserving branching bisimilarity. Thus, we get the following correspondence.

Corollary 5.1. For every executable transition T there exists a finite recursive specification E and an E-interpretable process expression p such that $T \leftrightarrow_b^\Delta \mathcal{T}_E(p)$.

Note that if E is a finite recursive specification in our calculus, and p is an E-interpretable process expression, then $\mathcal{T}_E(p)$ is a boundedly branching computable transition system. Hence, up to divergence-preserving branching bisimilarity, we get a one-to-one correspondence between executability and finite definability in our process calculus.

Corollary 5.2. A transition system is executable modulo divergence-preserving branching bisimilarity if, and only if, it is finitely definable modulo divergence-preserving branching bisimilarity in the process calculus with deadlock, skip, action prefix, alternative composition and parallel composition with value-passing handshaking communication.

For the aforementioned corollary it is important that our calculus does not include sequential composition. If sequential composition is added to our calculus, then there exist recursive specifications with an associated transition system that is unboundedly branching (see, e.g., [3]).

6 Concluding remarks

Our reactive Turing machines extend conventional Turing machines with a notion of interaction. Interactive computation has been studied extensively in the past two decades (see, e.g., [5, 11, 14]). The goal in these works is mainly to investigate to what extent interaction may have a beneficial effect on the power of sequential computation. We discuss the notion of *persistent Turing machine* of [11] and the notion of *interactive Turing machine* of [14] in more detail because they are closest to our notion of reactive Turing machine.

Persistent Turing machines. In [11], the notion of *persistent Turing machine* (PTM) is proposed. A PTM is a non-deterministic 3-tape Turing machine with a read-only input tape, a write-only output tape and a work tape. Semantically, a PTM repeatedly executes the following three steps, which together constitute a *macrostep*:

1. it allows the environment to write a finite string as input to the computation on the input tape;
2. it runs the non-deterministic 3-tape Turing machine until it terminates (a computation step from the non-deterministic 3-tape Turing machine is referred to as a *microstep*);
3. it allows the environment to read the output from the output tape after which both the input and the output tape are emptied for the next macrostep.

Note that the work tape is not emptied after a macrostep; its contents persist until the next macrostep. Interaction with the environment is achieved by assuming that the environment periodically writes new input to the input tape and reads the output from the output tape.

A *universal* PTM is defined as a PTM that inputs the code of an arbitrary PTM in the first step, and simulates it on macrostep level. It is proved in [11] that there exist universal PTMs; the proof uses a conventional universal Turing machine.

The macrostep semantics is used to define for every PTM a *persistent stream language* with an associated notion of equivalence, and an *interactive transition system* with associated notions of isomorphism and bisimilarity. The notion of interactive transition system with the associated notions of behavioural equivalence are non-standard, and it should be investigated how they relate to the conventional notions of transition system and behavioural equivalence as discussed in this paper. Our transition system semantics for RTMs is on the microstep level, and is hence more refined with respect to moments of choice both when it comes to inputting a new string, and with respect to the moments of choice in internal computations.

Interestingly, in the conclusions of [11] it is conjectured that parallel composition does affect the notion of interactive computability, in the sense that the parallel interactive computation is more expressive than sequential interactive computation. Our result at the end of Sect. 3 that the parallel composition of RTMs can be faithfully simulated by a single RTM seems to refute this claim.

Interactive Turing machines. Van Leeuwen and Wiedermann proposed *interactive Turing machines* (ITMs) in [14] (the formal details are worked out by Verbaan in [20]). An ITM is a conventional Turing machine endowed with an input port and an output port. In every step the ITM may input a symbol from some finite alphabet on its input port and outputs a symbol on its output port. ITMs are not designed to halt; they compute translations of infinite input streams to infinite output streams.

Already in [14], but more prominently in subsequent work (see, e.g., [21]), van Leeuwen and Wiedermann consider a further extension of the Turing machine paradigm, adding a notion of advice [13]. An *interactive Turing machines with advice* is an ITM that can, when needed, access some advice function that allows for inserting external information into the computation. It is established that this extension allows the modelling of non-uniform evolution. It is claimed by the authors that non-uniform evolution is essential for modelling the Internet, and that the resulting computational paradigm is more powerful than that of conventional Turing machines.

Our RTMs are not capable of modelling non-uniform evolution. We leave it as future work to study an extension of RTMs with advice. In particular, it would be interesting to consider and extension with behavioural advice, rather than functional advice, modelling advice as an extra parallel component representing the non-uniform behaviour of the environment with which the system interacts.

Expressiveness of process calculi. In [2], Baeten, Bergstra and Klop prove that computable process graphs are finitely definable in ACP$_\tau$ up to weak bisimilarity; their proof involves a finite specification of a (conventional) Turing machine. Their result was extended by Phillips in [17], who proved that all recursively enumerable process graphs are finitely definable up to weak bisimilarity. We have further extended these results by adopting a more general notion of *final state* and more refined notions of behavioural equivalence.

RTMs may prove to be a useful tool in establishing the expressiveness of richer process calculi. For instance, the transition system associated with a π-calculus expression is effective, so it can be simulated by an RTM, at least up to branching bisimilarity. We conjecture that the converse —every executable transition system can be specified by a π-calculus expression— is also true, but leave it for future work to work this out in detail.

Petri showed already in his thesis [16] that concurrency and interaction may serve to bridge the gap between the theoretically convenient Turing machine model of a sequential machine with unbounded memory, and the practically more realistic notion of extendable architecture of components with bounded memory. The specification we present in the proof of Corollary 5.1 is another illustration of this idea: the unbounded tape is modelled as an unbounded parallel composition. It would be interesting to further study the inherent trade-off between unbounded parallel composition and unbounded memory in the context of RTMs, considering unbounded parallel compositions of RTMs with bounded memory.

Acknowledgement. We thank Herman Geuvers for discussions, and Clemens Grabmayer for suggesting to us the term *reactive Turing machine*, and the anonymous referees for their useful comments.

References

[1] J. C. M. Baeten, T. Basten, and M. A. Reniers. *Process Algebra (Equational Theories of Communicating Processes)*. Cambridge University Press, 2009.

[2] J. C. M. Baeten, J. A. Bergstra, and J. W. Klop. On the consistency of Koomen's fair abstraction rule. *Theoretical Computer Science*, 51:129–176, 1987.

[3] J. C. M. Baeten, P. J. L. Cuijpers, B. Luttik, and P. J. A. van Tilburg. A process-theoretic look at automata. In F. Arbab and M. Sirjani, editors, *Proceedings of FSEN 2009*, volume 5961 of *LNCS*, pages 1–33, Berlin Heidelberg, 2010. Springer.

[4] J. C. M. Baeten, B. Luttik, and P. J. A. van Tilburg. Reactive Turing machines. *CoRR*, abs/1104.1738v4, 2012.

[5] A. Blass, Y. Gurevich, D. Rosenzweig, and B. Rossman. Interactive small-step algorithms I: Axiomatization. *Logical Methods in Computer Science*, 3(4), 2007.

[6] G. Boudol. Notes on algebraic calculi of processes. In K. R. Apt, editor, *Logics and Models of Concurrent Systems*, number F13 in NATO-ASI Series, pages 261–303, Berlin, 1985. Springer.

[7] P. Darondeau. Bisimulation and effectiveness. *Inf. Process. Lett.*, 30(1):19–20, 1989.

[8] R. J. van Glabbeek. The Linear Time – Branching Time Spectrum II. In E. Best, editor, *Proceedings of CONCUR '93*, number 715 in LNCS, pages 66–81. Springer Verlag, 1993.

[9] R. J. van Glabbeek, B. Luttik, and N. Trcka. Branching bisimilarity with explicit divergence. *Fundam. Inform.*, 93(4):371–392, 2009.

[10] R. J. van Glabbeek and W. P. Weijland. Branching time and abstraction in bisimulation semantics. *Journal of the ACM*, 43(3):555–600, 1996.

[11] D. Q. Goldin, S. A. Smolka, P. C. Attie, and E. L. Sonderegger. Turing machines, transition systems, and interaction. *Inf. Comput.*, 194(2):101–128, 2004.

[12] D. Harel and A. Pnueli. On the development of reactive systems. In K. R. Apt, editor, *Logics and Models of Concurrent Systems*, volume F-13 of *NATO ASI Series*, pages 477–498, New York, 1985. Springer-Verlag.

[13] R. M. Karp and R. J. Lipton. Turing machines that take advice. *L'Enseignement Mathématique*, II^e Série(Tome XXVIII):191–209, 1982.

[14] J. van Leeuwen and J. Wiedermann. Beyond the turing limit: Evolving interactive systems. In Leszek Pacholski and Peter Ruzicka, editors, *SOFSEM*, volume 2234 of *Lecture Notes in Computer Science*, pages 90–109. Springer, 2001.

[15] R. Milner. *Communication and Concurrency.* Prentice Hall, 1989.

[16] C. A. Petri. *Kommunikation mit Automaten.* PhD thesis, Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2, 1962.

[17] I. C. C. Phillips. A note on expressiveness of process algebra. In G. L. Burn, S. Gay, and M. D. Ryan, editors, *Proceedings of the First Imperial College Department of Computing Workshop on Theory and Formal Methods*, Workshops in Computing, pages 260–264. Springer-Verlag, 1993.

[18] H. Rogers. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill Book Company, 1967. Reprinted by MIT Press, 1987.

[19] A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936.

[20] P. Verbaan. *The Computational Complexity of Evolving Systems.* PhD thesis, Utrecht University, 2005.

[21] J. Wiedermann and J. van Leeuwen. How we think of computing today. In A. Beckmann, C. Dimitracopoulos, and B. Löwe, editors, *CiE*, volume 5028 of *LNCS*, pages 579–593. Springer, 2008.

Correlated Knowledge: An Epistemic-Logic View on Quantum Entanglement

Alexandru Baltag* and Sonja Smets†

This paper is published in: *International Journal of Theoretical Physics* (2010) 49: pp. 3005 - 3021

Abstract

In this paper we give a logical analysis of both classical and quantum correlations. We propose a new logical system to reason about the information carried by a complex system composed of several parts. Our formalism is based on an extension of epistemic logic with operators for "group knowledge" (the logic **GEL**), further extended with atomic sentences describing the results of "joint observations" (the logic **LCK**). As models we introduce *correlation models*, as a generalization of the standard representation of epistemic models as vector models. We give sound and complete axiomatizations for our logics, and we use this setting to investigate the relationship between the information carried by each of the parts of a complex system and the information carried by the whole system. In particular we distinguish between the "distributed information", obtainable by simply pooling together all the information that can be separately observed in any of the parts, and "correlated information", obtainable only by doing joint observations of the parts (and pooling together the results). Our formalism throws a new light on the difference between classical and quantum information and gives rise to a informational-logical characterization of the notion of "quantum entanglement".

1 Introduction

This paper forms part of the recent research trend aimed at connecting the Logical Foundations of Quantum Physics to fundamental issues in Logic and Theoretical Computer Science, and in particular to theories of distributed computation and quantum information (see e.g. [2, 16, 3, 5, 6, 7, 8, 20, 18]). Instead of starting with traditional "quantum logic", we focus on one of the logical formalisms (namely, *Epistemic Logic*) that was successfully used in Theoretical Computer Science. Because of their spatial features, epistemic logics have traditionally been used to model the flow of classical information in distributed systems, including secure communication protocols, distributed computation and multi-agent robotic systems in Artificial Intelligence. Our aim is to extend this domain of applications to include quantum information flow. We thus adapt and generalize epistemic logic to reason about multi-partite quantum systems and quantum correlations.

Our main focus in this paper is the relationship between the information carried by a complex system and the information carried by each of the parts of the system. As examples of complex systems, we look at any physical system composed of more elementary subsystems (particles), but also at any group of "agents" that can perform observations (of their local environment), logical inferences or introspective reflection. The "parts" of a complex system do not have to be independent, on the contrary their behavior (e.g. the agents' observations) might happen to be correlated, either due to prior communication or to other causes (e.g. quantum entanglement). As a result, the pieces of information carried by the parts might not be independent either. Different types of complex systems will manifest different types of correlations or dependencies between the information carried by their parts.

We are interested in modeling these various types of complex systems from an operational point of view. That is to say, we look for the operational criteria that characterize various types of correlations between the local parts of a given system. What we show is that there are several ways in which the information carried by the individual parts of a system can be combined. Depending on how the

*ILLC, University of Amsterdam, A.Baltag@uva.nl
†ILLC, University of Amsterdam, S.J.L.Smets@uva.nl

information between the parts is pooled together, the system will exhibit different types of correlations. Our study of informational correlations will allow us to mark the difference between classical and quantum correlations.

To formalize these notions, we use a version of *epistemic logic*, in which the *epistemic "agents" are to be identified with the elementary parts* (basic "subsystems", or "locations") *of a complex physical system.* As is standard practice in Computer Science, we adopt the "external" view on knowledge advocated by Halpern et alia [21], which allows for applications of epistemic logic to situations that reach beyond the traditional study of real agents' "knowledge": in this view, any of the localized parts of a complex system can be seen as a "virtual agent", while the system itself as well as any of its subsystems can be seen as a virtual "group of agents". The "implicit knowledge" that (is not necessarily actually "possessed" by, but) can be "externally" attributed to such a virtual agent is given simply by the *information that is potentially available at the corresponding location*: something is "known" if it is a *consequence of features that can in principle be observed at that location.* Implicit knowledge thus gives us the information (about the overall system) that is carried by a part (of that system). In this sense, we can think of epistemic logic as a "spatial logic", meant to express the relationships between (the information available at) different locations in a complex system. Usually, when considering logics with spatial features, one thinks about a topology or even a metric space, but in this paper we are only interested in the "local" features indicating which information is carried by which part of the system. The relation between systems and their subsystems is what is relevant for us, and not any sense of "distance" or nearness between systems.

A natural question raised in the epistemic logic literature is the following: is there a sense in which one can externally assign knowledge to a *group* of (virtual) agents? In other words: *what is the implicit knowledge of a group*? The relevance of this question to our inquiry comes from the fact that, in our setting, implicit "group knowledge" would capture the information carried by a complex physical system. The standard answer to this question in the current literature is given by *distributed knowledge*: this is the information obtainable by pooling together (and closing under logical inference) the "knowledge" of each of the "parts" (agents). According to this view, the implicit knowledge of a group is the same as its distributed knowledge; in other words, the information carried by a complex system is nothing but the "sum" of the information carried by its parts.

Our main claim is that, while this standard answer is adequate for classical physics, it *fails for quantum systems.* An entangled system carries more information than the sum of its parts. Moreover, there are also examples of *social situations* in which this standard answer fails for real-life agents: *whenever a group of agents can cooperate to make joint observations, the implicit knowledge of the group* (defined, as for a single agent, in terms of what follows from the features that can in principle be observed by the group) *will typically go beyond distributed knowledge* (which only takes into account the results of separate, uncorrelated observations by each of the members of the group). Hence, to correctly model the information implicitly carried by a complex system or group of agents, we propose the notion of *"correlated knowledge"*, that takes into account the correlations between the pieces of information carried by the parts of the system, or the potential results of joint observations that can be performed by the group.

For our investigation of group knowledge, we propose a logical system, called *"General Epistemic Logic"*(**GEL**), based on an extension to groups of traditional epistemic logic and on a generalization of the usual notion of epistemic model. In Section 3, we give a *sound and complete axiomatization* for **GEL**. In subsequent sections, we show that correlated knowledge is useful to capture the non-classicality of a physical system (or the existence of correlations between the agents' observational capabilities in a group). We use the logic **GEL** to give an *informational-logical characterization of the properties of "separability" and "entanglement".*

In the final section of this paper we extend the logic **GEL** with operators that make explicit the agents' observational capabilities, obtaining a *"Logic of Correlated Knowledge"* **LCK**. We introduce a notion of "correlated models", that generalizes the well-known representation (due to Halpern et alia [21] of epistemic models as vector models. We give a *sound and complete axiomatization* for **LCK** with respect to correlated models.

2 Background Notions

Information. In this paper we are mainly concerned with the "qualitative" ("logical", or "semantic") aspects of information. In contrast to the *syntactic* (or *quantitative*) approach to information, which is primarily concerned with quantitative measures of information (be it in terms of Shannon entropy or von Neumann entropy), the semantic approach is standard in Logic and Computer Science and focuses on the meaning or "aboutness" of information. Semantic information can be true or false and is always "about" something: propositions have a meaning or an "information content". The main issue in this semantic approach is to find the formal laws governing information flow of a specific type, so that we can analyze it, reason about it, verify its correctness etc. There are of course several different ways to model and view "semantic information". In [13, 14], two compatible views are being discussed: the first looks at "information as range" and the second at "information as correlation". The first is tied to a possible world model, where an increase in information is related to a decrease of the range of worlds that are considered possible and vice versa (see [13, 4]). Indeed, "Information as range" takes the point of view of an "external observer", whose information-state is represented only via the relations between the worlds that he or she considers possible. The characteristic feature of the second notion, "information as correlation", is the idea that information can manifest itself as a correlation between different situations or their states of affairs. Here, an increase in information is related to a decrease, restriction, or constraint, on the number of possible correlations between different situations. The ideas behind "information as correlation" trace back to Situation theory [11, 19] and originated as a tool to capture the "aboutness" or the semantic content of information even in the absence of observers. As a third way to study semantic information, we mention that it is possible to combine the "information as range" and "information as correlation" views in one model. As an example we mention here the epistemic constraint logic in [13] and the interpreted systems in [21]. Also our general epistemic logic in this paper will address issues that are of concern when we look at "information as range" and at "information as correlation".

Implicit Knowledge. We adopt the distinction in the CS literature between implicit and explicit knowledge (see e.g [21]), and use the notion of "information carried by a system" interchangeably with the notion of "implicit knowledge" (in line also with [8, 10]). Explicit knowledge is the actual information possessed by a real agent, the information stored in its "database" or its subjective internal state. Explicit knowledge is not necessarily introspective and might not be closed under logical consequence. In contrast, implicit knowledge can be visualized as what a *virtual* agent *could* in principle *come to "know"* by performing observations on that system and deriving logical consequences. It is the *potential knowledge* of the agent (what follows implicitly from his possible observations). This is an *"external" sense of knowledge* : it is the "knowledge" that "we" externally assign to an agent (or a piece of hardware, a particle or even just a spatial location), based on what that agent could observe (or what is in principle observable at that location). Implicit knowledge thus embodies an objective concept of "locally available (semantic) information", and has nothing to do with some subjective state internal to the subsystem: it sums up all information that is potentially available at that location.

As far as implicit knowledge is concerned, there are no problems with positive and negative introspection nor with logical omniscience. Indeed, it seems reasonable to say that if a system carries the information P then it implicitly carries the information that P is true, and that *all consequences* of P are true, and that the system carries the information P. In "agent" terms, this means assuming that the capabilities of our virtual agent include, not only observations, but also logical inferences and acts of reflection (introspection). So implicit knowledge can be assumed to be always truthful and closed under logical inference and under (positive and negative) introspection. In other words, implicit knowledge satisfies the axioms of the modal system known as S5.

3 General Epistemic Logic

Consider a complex system composed of n basic *components* (or "locations"). We denote each basic component with a label from a given (finite) set $N = \{1, ..., n\}$. Note that the information carried by this complex system may be *distributed* throughout space, it can be *localized*, concentrated at specific spatial "locations" or "components" of the system. Hence, some information is potentially available only at some locations, but not at others. We also want to consider (the information) carried by *subsystems* composed of several (but not necessarily all) components. To capture these spatial features we introduce a *generalized "epistemic logic"* as a special type of *spatial logic*, allowing us to have epistemic operators ("knowledge") for both *individuals* ("agents") and *groups* (of agents). We use the notion of component and agent interchangeably: metaphorically, imagine associating to *each subsystem a virtual agent that can "observe" only the state of that subsystem.* In the case of a physical system, this sums up all the information that is obtainable by performing *local measurements* on that subsystem (and closing under logical inference). We use *sets* $I \subseteq N$ of labels to denote complex subsystems (or groups of "agents"). The largest group is the "whole world" N, while the smallest groups are singletons $\{i\}$ consisting of an individual agent. Our logic will have an "information" (or "implicit knowledge") operator K_I for each subsystem I. As mentioned above, knowledge ("episteme") is used here only in the *implicit, external* sense, as "information that is in principle available" (via local observations) at a given location. So one may think of the proposition $K_I P$ as saying that the subsystem, or group, I (potentially) carries the information that P is the case. For groups $\{i\}$ of one individual agent, we use the simplified notation K_i instead of $K_{\{i\}}$. The use of K_I-operators to capture qualitative spatial features of complex systems, extends our previous approach in [8] which in its turn is based on [7, 5] and inspired by [16, 2].

Observational Equivalence. Let s and s' be two possible states of the world (or "possible worlds"), if the implicit information carried by a system I is *the same* in these two states we write $s \overset{I}{\sim} s'$, and say that the states are *"observationally equivalent", or "indistinguishable"*, for system I. Again, we simply write $s \overset{i}{\sim} s'$ when $I = \{i\}$. Observational equivalence for I means in agent's terms that the virtual agent or group of agents (associated to) *I can make exactly the same observations* (at location I) in two states of the world. We will use the relation of observational equivalence to give an interpretation to the K_I-modalities in our logic:

General Epistemic Frame. For a given set N of basic components, a set of states (or "possible worlds") Σ, a family of binary relations $\{\overset{I}{\sim}\}_{I \subseteq N} \subseteq \Sigma \times \Sigma$ for every subsystem $I \subseteq N$, we define a *general epistemic frame* to be a Kripke frame (or multi-modal frame) $(\Sigma, \{\overset{I}{\sim}\}_{I \subseteq N})$ subject to the following four conditions:

1. all $\overset{I}{\sim}$ are *labeled equivalence relations* (for every set $I \subseteq N$);
2. *Information is Monotonic w.r.t. groups*: if $I \subseteq J$ then $\overset{J}{\sim} \subseteq \overset{I}{\sim}$;
3. *Observability Principle*: if $s \overset{N}{\sim} s'$ then $s = s'$.
4. *Vacuous Information:* $s \overset{\emptyset}{\sim} s'$ for all $s, s' \in \Sigma$.

We read $s \overset{I}{\sim} s'$ as: *subsystem I cannot distinguish state s from s'* (via any local observations, performable on I). In other words, $\overset{I}{\sim}$ gives us a notion of *''observational equivalence" relative to subsystem I.* The above conditions seem natural for this interpretation. The first condition says that the relation is reflexive, transitive and symmetric. The second condition assumes that every observation that can be performed by an agent of a group is in principle available to the whole group. In other words, the members of a group *have the capacity to share information* among themselves. In terms of systems we then say that *if a subsystem carries the information that P then the whole system carries the information that P.* In logical terms this captures the fact that *information behaves monotonically with respect to group inclusion.* And relationally, it says that the states of the world that are observationally equivalent for a system/group I are also observationally equivalent for any subgroup $J \subseteq I$. The third condition says that if two possible states of the world are indistinguishable with respect to the "whole world" N then they are the same. We call this the *"observability" principle*, as it identifies states of the world that differ in ways that are not observable even by the whole world. Finally, the last condition says that *the empty group has no non-trivial information whatsoever*: being unable to make any observations, the empty group cannot distinguish between any two states.

Local Information State. Given a general epistemic frame $(\Sigma, \{\overset{I}{\sim}\}_{I \subseteq N})$, we construct the notion of an I-local state, for each subsystem $I \subseteq N$:

$$s_I := \{s' \in \Sigma : s \overset{I}{\sim} s'\}.$$

Here, s_I captures the "agent i's local state of information", while s_I does the same for groups.

Σ-Propositions and Models. Given a general epistemic frame $(\Sigma, \{\overset{I}{\sim}\}_{I \subseteq N})$, a *$\Sigma$-proposition* is any subset $P \subseteq \Sigma$. Intuitively, we say that a state s satisfies the proposition P if $s \in P$. We define a *general epistemic model* to be a structure $\mathbf{\Sigma} = (\Sigma, \{\overset{I}{\sim}\}_{I \subseteq N}, ||\,.\,||)$, consisting of a general epistemic frame together with a valuation map $||\,.\,|| \colon \Omega \to P(\Sigma)$, mapping every element of a given set of atomic sentences into Σ-propositions. We use the standard notation for *satisfaction* of atomic sentences in a given state of model Σ denoted by $s \models p$ or $s \in || \, p \, ||$. For every model $\mathbf{\Sigma}$, we have the usual Boolean operators on Σ-propositions: $P \wedge Q := P \cap Q$, $P \vee Q := P \cup Q$, $\neg P := \Sigma \backslash P$, $P \Rightarrow Q := \neg P \vee Q$. We also have the constants $\top_\Sigma := \Sigma$ and $\bot_\Sigma := \emptyset$. Finally, the "knowledge" operator is defined on Σ-propositions by putting $K_I P := \{s \in \Sigma : t \in P \text{ for every } t \overset{I}{\sim} s\}$.

General Epistemic Logic (GEL). The language of the logic **GEL** has the following syntax, starting from atomic sentences $p \in \Omega$:

$$\varphi \; := \; p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_I \varphi$$

The semantics is given by an *interpretation map* associating to each sentence φ of **GEL** a proposition $|| \, \varphi \, ||$. Equivalently, we extend *the satisfaction relation* $s \models \varphi$ from atomic sentences to arbitrary formulas φ. The definition is by induction in terms of the obvious compositional clauses. In particular, for the K_I modality we set:

$$s \models K_I \varphi \text{ iff } t \models \varphi \text{ for all states } t \overset{I}{\sim} s.$$

So $K_I \varphi$ is true in a given state s, or a system carries the information that φ is the case in state s, if and only if φ holds in all states of the world that are observationally equivalent for I to s. We would like to stress the fact that this captures the idea that "information is based on potential observations".

Proof System. In addition to the rules and axioms of propositional logic, the *proof system* of **GEL** includes:

1. *K_I-Necessitation.* From $\vdash \varphi$, infer $\vdash K_I \varphi$
2. *Kripke's Axiom.* $\vdash K_I(\varphi \Rightarrow \psi) \Rightarrow (K_I \varphi \Rightarrow K_I \psi)$
3. *Truthfulness.* $\vdash K_I \varphi \Rightarrow \varphi$
4. *Positive Introspection.* $\vdash K_I \varphi \Rightarrow K_I K_I \varphi$
5. *Negative Introspection.* $\vdash \neg K_I \varphi \Rightarrow K_I \neg K_I \varphi$
6. *Monotonicity of Group "Knowledge".* For $I \subseteq J$, we have $\vdash K_I \varphi \Rightarrow K_J \varphi$
7. *Observability.* $\vdash \varphi \Rightarrow K_N \varphi$

Using standard results in Modal Correspondence theory (see e.g. [15]), it is easy to show the following:

Theorem 1. *The above proof system is sound and complete with respect to general epistemic frames.*

Proof. Soundness is trivial: rule 1 and axiom 2 hold in any Kripke model; axioms 3,4,5 hold in any model in which $\overset{I}{\sim}$ are equivalence relations; axiom 6 holds in any model satisfying the condition that "information is monotonic" (i.e. $\overset{J}{\sim} \subseteq \overset{I}{\sim}$); axiom 7 holds in any model satisfying the Observability Principle. For *completeness*, let φ_0 be a sentence that is *consistent* (with respect to the above proof system). We need to show that φ_0 is satisfiable in some general epistemic model. For this, we use the standard "canonical model" construction familiar from Modal Logic [15], in which states are "theories" (=sets of sentences in the language of **GEL**) that are maximally consistent with respect to the above proof system.

Let Ω be the set of all such maximally consistent theories. We define equivalence relations $\overset{I}{\sim}$ on Ω, by putting for every two theories $T, T' \in \Omega$: $T \overset{I}{\sim} T'$ iff $\forall\varphi(K_I\varphi \in T \iff K_I\varphi \in T')$. For the valuation, we put $\|p\| := \{T : p \in T\}$. Axioms 3,4 and 5 ensure that $\overset{I}{\sim}$ are equivalence relations; axiom 6 ensures that information is monotonic ($\overset{J}{\sim} \subseteq \overset{I}{\sim}$ for $I \subseteq J$); axiom 7 ensures that the Observability Principle holds. However, the "Vacuous Information" condition does *not* hold in Ω. Still, we can ensure it by *restricting our model* Ω to the set $\Omega_0 := \{T \in \Omega : T \overset{\emptyset}{\sim} T_0\}$, where T_0 is some fixed maximally consistent theory $T_0 \in \Omega$ such that $\varphi_0 \in T_0$. (The existence of T_0 is ensured by the Lindenbaum Lemma, as usually.) Since $\overset{I}{\sim} \subseteq \overset{\emptyset}{\sim}$ (by monotonicity of information), this restriction preserves all the properties of $\overset{I}{\sim}$. In addition, it obviously ensures that the "Vacuous Information" condition holds (since $T \overset{\emptyset}{\sim} T_0 \overset{\emptyset}{\sim} T'$ for all $T, T' \in \Omega_0$). So we obtain a general epistemic model $\mathbf{\Omega_0}$. Finally, using all our axioms (including rule 1 and axiom 2), we can prove in the usual way a "*Truth Lemma*", stating that a sentence φ holds at a state $T \in \Omega$ iff it belongs to T (seen as a theory): $T \models \varphi$ iff $\varphi \in T$. As a consequence, φ_0 holds at state T_0 in the model Ω_0 (since $\varphi_0 \in T_0$), and hence φ_0 is satisfiable. $\dashv$

4 Distributed Knowledge

The notion of distributed knowledge was first introduced in the work of Halpern and Moses [22]. In [21] the authors say that a group has distributed knowledge of φ if roughly speaking the agents' combined knowledge implies φ. Or "distributed knowledge can be viewed as what a wise man - one who has complete knowledge of what every member of the group knows - would know." [21, p3]. The intuition is that, in addition to making individual observations, the agents of the group can "combine" their knowledge, by sharing all they know: *they can announce to the group the knowledge obtained by each member on the basis of their* separate *observations.*

Similar as for implicit knowledge, we introduce a modal operator for *the distributed knowledge of a group I*, denoted by DK_I. This operator DK_I can be defined using a "distributed observational equivalence" relation, which is given by the intersection $\bigcap_{i\in I} \overset{i}{\sim}$ of all the individual observational equivalence relations. The idea is that two states are indistinguishable for the group if and only if they are indistinguishable for *all* the members of the group. *Distributed knowledge DK_I* is simply defined as the Kripke modality for $\bigcap_i \overset{i}{\sim}$, i.e. given by:

$$s \models DK_I\varphi \text{ iff } \text{ for every state } t \in \Sigma, \text{ if } \forall i \in I \ s \overset{i}{\sim} t \text{ then } t \models \varphi.$$

In [21], the authors identify implicit knowledge of a group I with distributed knowledge DK_I. We doubt however that this identification holds in all situations. Recall that implicit knowledge was explained as what the agent/group could come to know based on potential observations. So the question is: *what are, in general, the observational capabilities of a group*? Looking at the definition, we see that the use of the intersection of individual observational equivalencies makes sense only *if we assume that a group's observations are nothing but observations done by either of the members of the group.* But this is in general a *highly unreasonable assumption*: joint observations by a group are not in general the same as independent observations by each of the members of the group.

In a group whose implicit knowledge is the same as distributed knowledge, each agent can only share with the group the end-result of all her separate, independent observations, but agents are not allowed to correlate (the results of) their observations, in a simultaneous or sequential manner. But according to us, the natural notion of (implicit) group knowledge is something *different*, something that could be called "correlated knowledge": this is what the group could come to know by performing joint (correlated) observations and sharing the results.

Separability. For system $J \subseteq I$, we say that the system I is *J-separable* in state s if we have $s_J \cap s_{I\setminus J} = s_I$. A system I is called *fully separable* in state s if we have $\bigcap_{i\in I} s_i = s_I$. *Full separability means that group I's knowledge in state s is the same as its distributed knowledge*, i.e. $s \models K_IP$ iff $s \models DK_IP$, for all sets $P \subseteq \Sigma$. Note that if system I is fully separable then it is J-separable for all $J \subseteq I$, but the converse is false in general. We call a state *I-entangled* if it is not I-separable.

Classical Epistemic Frame. A general epistemic frame $(\Sigma, \{\overset{I}{\sim}\}_{I\subseteq N})$ is called **classical** if *all its states are fully separable*, i.e. if it satisfies $\overset{I}{\sim} = \bigcap_{i\in I} \overset{i}{\sim}$ for all systems I. So a frame is classical if and only if *any group's "knowledge" in any state (coincides with its distributed knowledge, and hence) can be obtained by pooling together the information of each of its components*: $K_I = DK_I$

Classically, a group cannot distinguish two states if and only if no member of the group can distinguish them. Note that not only classical or macroscopic systems will satisfy the conditions of a classical epistemic frame. Indeed, one may also encounter classical epistemic frames in the *quantum world*. This happens when the subsystems are *separated*.

The Vector Model Representation of Classical Epistemic Models. Classical epistemic frames (models) can be given a "normal form" representation as vector frames (models):

Fact ([12]) *Let, for each i, $\Sigma_i := \{s_i : s \in \Sigma\}$ be the set of all i-local states. Every classical epistemic frame Σ can be canonically embedded into the Cartesian product $\Sigma_1 \times \Sigma_2 \times \cdots \Sigma_n$, via some embedding e satisfying*

$$s \overset{I}{\sim} s' \text{ iff } e(s)_i = e(s')_i \text{ for all } i \in I.$$

To show this, put $e(s) := (s_i)_{i\in N}$, where $s_i = \{s' \in \Sigma : s \overset{i}{\sim} s'\}$ is the i-local state of s, as defined in the previous section. This vector representation corresponds to another well-known way to model epistemic logic: the "interpreted systems" representation, in the style of Halpern et alia [21], in which *the global states are simply taken to be tuples of local states*, with *identity of the i-th components as the indistinguishability relation $\overset{i}{\sim}$*. But note that such a representation *is not possible in the case of general (non-classical) epistemic frames*!

Surprisingly enough, the same validities hold in the class of classical epistemic models as in the class of general epistemic models. This is shown by the following result:

Proposition 1. *The proof system of **GEL** (given in the previous section) is sound and complete with respect to classical epistemic models as well.*

The *proof* is in Halpern et alia [21]: essentially, this is their proof of completeness for epistemic logic extended with *distributed knowledge* operators.

In conclusion, *the language of general epistemic logic **GEL** cannot distinguish between general epistemic models and the classical ones.* In order to be able to distinguish them, we will have to extend the language of GEL to a richer "logic of correlated knowledge" **LCK** (Section 6).

5 Quantum "Knowledge"

A single quantum system can be represented by a state space Σ consisting of *rays*[1] in a Hilbert space H. A quantum system composed of N subsystems $\Sigma_1, ..., \Sigma_n$ is represented by the state space $\Sigma_1 \otimes \cdots \otimes \Sigma_n$ corresponding to the *tensor product* $H_1 \otimes \cdots H_n$. Note that this tensor product is much richer than the Cartesian product $\Sigma_1 \times \cdots \Sigma_n$: when the state of a system s is *entangled*, then it *cannot be decomposed as a tuple of local states*. So *we cannot think of a composed quantum system as a classical epistemic frame.* Nevertheless, we will show that they *can still be thought of (non-classical) general epistemic frames*, in a natural way.

To see this, we consider the following question: what is the "state" of an entangled subsystem I? For instance, what is the state of component 1 in the binary system $\mid 00\rangle + \mid 11\rangle$? As we saw, the answer for *I-separable* states $s = s_I \otimes s_{N\setminus I}$ is simply given by the local state s_I. But we also saw that one cannot talk in any meaningful way about the I-local state of an i-entangled system. To proceed, we first give the standard QM definition in terms of density operators, then we justify its usefulness for our purposes by looking at the results of local observations.

[1]i.e. vector identified up to a multiplication with a non-zero scalar

The State of a Subsystem. If a global system is in state s (thus having an associated density operator ρ_s), then Quantum Mechanics describes the state $s_{(I)}$ of any of its subsystems I (possibly entangled with its environment $N \setminus I$ in the state s) by the density operator

$$s_{(I)} := tr_{N \setminus I}(\rho_s).$$

In other words, the "state" of subsystem I is obtained by taking the partial trace $tr_{N \setminus I}$ (with respect to the subsystem's environment $N \setminus I$) of (the density operator associated to) the global state s.

Notice that, when the subsystem I is entangled with its environment $N \setminus I$, the above description does not really give us a "state" in the sense of this paper (i.e. a *pure state*), but a "mixed state". Nevertheless, as an abstract description, it can still give us an *indistinguishability relation* $\overset{I}{\sim}$ on global states: the specific definition of the "state" of a subsystem is not relevant for us in itself, but only the resulting notion of "identity of states" of the given subsystem. This leads us to the following definition:

Observational Equivalence in Quantum Systems. Two quantum states s, s' of a global quantum system N are *observationally equivalent* (*"indistinguishable"*) for a subsystem $I \subseteq N$ if the mixed states of subsystem I are *the same* in s and s'. Formally:

$$s \overset{I}{\sim} s' \text{ iff } tr_{N \setminus I}(\rho_s) = tr_{N \setminus I}(\rho'_s).$$

This indirect definition using density operators may look ad-hoc and unnatural, but it can be justified in terms of what an observer can learn about an entangled subsystem I *by observing only that subsystem* (so by performing local measurements on I). Indeed, it is known that a mixed state corresponds to a probability measure over pure states, but what is not always well-appreciated is the *meaning* of the mixed state $s_{(I)}$ describing a (possibly entangled) subsystem:

Quantum I-equivalence via local observations. Provided that we have *an unlimited supply* of identical I-entangled systems in the *same* (entangled) global state s, imagine that the virtual agent associated to I can perform *all possible local measurements* (in various bases) on (various copies of) subsystem I. The agent can also repeat the same tests on different copies and observe the *frequency of each result.* After many tests, he can approximate the *probability of every given result, for each possible local measurement.* The list of all these probabilities (for each result of each type of measurement) gives us the "information carried by subsystem I", or the "information obtainable by local observations at location I". Two global states s, s' are I-indistinguishable if all these probabilities are the same in s and s', i.e. if the two states behave the same way under I-local measurements.

Quantum I-equivalence via remote evolutions. A third way to define observational equivalence is via *invariance* under changes that do *not* affect the information carried by subsystem I (see also [8]):

An evolution (unitary map) U is said to be *I-remote* (or "remote from I ") if it corresponds to applying only a local unitary map on the subsystem $N \setminus I$ (the "non-I " part of the system, also known as I's "environment"): i.e., if U is of the form $Id_I \otimes U_{N \setminus I}$, where Id_I is the identity map on subsystem I and $U_{N \setminus I}$ is a unitary map on the subsystem $N \setminus I$. In other words, U is I-remote if it is $N \setminus I$-local.

Intuitively, I-remote evolutions should not affect the "state" of subsystem I; hence, we could *define* the "state" of I as what is left *invariant* by all I-remote evolutions. As a consequence, two states will be I-indistinguishable if they differ only by some I-remote evolution.

The following result shows the equivalence of these three ways of defining observational equivalence:

Proposition 2. *For $I \subseteq N$, $s\, s' \in \Sigma$, the following are equivalent:*

(1). $tr_{N \setminus I}(\rho_s) = tr_{N \setminus I}(\rho_{s'})$;

(2). *for every I-local measurement, the probability of obtaining any given result is the same in state s as in state s';*

(3). *$s' = U(s)$ for some I-remote unitary map.*

So we can define the quantum equivalence relation $s \overset{I}{\sim} s'$, and hence our notion of *implicit knowledge* K_I, by any of the clauses given above. Using for example the third clause we obtain that $K_I P$ holds at s iff, for all I-remote evolutions U, P holds at $U(s)$. If P is implicitly known by I in state s, i.e. if $s \in K_I P$, then we say that *the subsystem I carries the information that P*.

A *quantum epistemic frame* is a (state space Σ associated to a) Hilbert space endowed with the quantum I-equivalence relations $\overset{I}{\sim}$ (as defined above) for every subsystem I.

Proposition 3. *Quantum epistemic frames are (i.e. satisfy all the postulates of) general epistemic frames.*

Properties. In addition to the properties of the K_I operator in GEL, we add:

- If s is I-separable, then $s \overset{I}{\sim} s'$ iff $s_I = s'_I$
- If I is fully separable then we have: $s \overset{I}{\sim} s'$ iff $s \overset{i}{\sim} s'$ for all $i \in I$. As a consequence, *the quantum "group" knowledge K_I of a fully separated system I is the same as the "distributed knowledge" DK_I of the "group" I.*
- In general (for non-fully separated systems I), the previous statement is false: the information K_I carried by a quantum (sub)system I is *not* the "sum" $DK_{i \in I}$ of the information carried by its i-component systems. In other words: *quantum epistemic frames are "non-classical".*

Example. For instance, in a Bell state when the information stored in two subsystems is correlated according to the identity rule, the agents associated to these subsystems will *never recover fully the information possessed by the global system if they cannot correlate the results of their individual observations.* Indeed, the following observation shows that in the Bell state $|00\rangle + |11\rangle$ composed of two entangled qubits 1 and 2, the two subsystems 1 and 2 *are in the same mixed state*:

Proposition 4. *The following are equivalent:*

1. $s \overset{1}{\sim} |00\rangle + |11\rangle$
2. $s \overset{2}{\sim} |00\rangle + |11\rangle$
3. $s = |z0\rangle + |z'1\rangle$, *for some orthogonal vectors* $z, z' \in H_1$.

(The equivalence of e.g. the second and the third clause can be shown by finding a 2-remote evolution $U_1 \otimes Id_2$ such that $U_1(|0\rangle) = z$ and $U_1(|1\rangle) = z'$, and using the definition of I-equivalence in terms of remote evolutions.)

This proposition tells us that, in the state $|00\rangle + |11\rangle$, the two subsystems *carry exactly the same information.* So pooling together their "knowledge" will *not* lead to any increase of information: the "distributed knowledge" of the group $\{1, 2\}$ is *the same* as the implicit knowledge of each of the qubits 1 and 2. In contrast, the "group knowledge" of $\{1, 2\}$ is much stronger: $s \overset{\{1,2\}}{\sim} |00\rangle + |11\rangle$ is equivalent to $s = |00\rangle + |11\rangle$. The group "knows" its own state, so it knows (that the qubits are in) the Bell state $|00\rangle + |11\rangle$.

Informational Characterizations of Separability and Entanglement Recall that we already gave an "epistemic" characterization of entanglement and separability in general epistemic frames:

A state s is I-separable iff I's knowledge in state s is the same as its distributed knowledge.

In the special case of quantum systems, this gives us the standard Quantum Mechanical notion of separability [2]. But this characterization cannot be expressed in the language of epistemic logic since it involves a second-order quantifier over all subsets P of the state space: it requires that, for every such subset, s satisfies $K_I P$ iff it satisfies $DK_I P$.

[2] In QM, a separable quantum state refers to a non-entangled state. For instance the global state of a bi-partite system is separable if it belongs to the Cartesian product $H_1 \times H_2$ of the two corresponding Hilbert spaces; in this case, each of the subsystems is in a well-defined (pure) local state.

However, in line with the ideas presented in [8] we *can* use epistemic logic to give "informational characterizations" of separability and entanglement in a quantum system, *provided we are given only one (logical constant denoting a) fully separable state.* Indeed, let $w = w_0 \otimes \cdots w_n$ be some (fixed) fully separable state; for example, we may take $w = \overline{0} = |0\rangle^{\otimes N} = |0\rangle \otimes |0\rangle \cdots |0\rangle$. Then we have:

> Two subsystems I and J are *entangled* in a (global) state s iff s satisfies $K_J K_I \neg w$ or (equivalently, $K_I K_J \neg w$. The state s is I-entangled iff the subsystem I and $N \setminus I$ are entangled in s, i.e. if s satisfies $K_I K_{N \setminus I} s$. The system is separable if it is not entangled: $\neg K_I K_{N \setminus I} \neg w$.

To summarize: *two physical systems are entangled if and only if they potentially carry (non-trivial) information about each other* (assuming no prior communication).

Example. For $n = 2$, we consider the set $\Sigma_{(1)}$ of all 1-separable (=2-separable=fully separable) global states, as our model. Then given that the system is in state $|00\rangle$, subsystem 1 is in state $|0\rangle$ and "implicitly knows" his own state. 1 implicitly knows also that it is not possible that the whole system is in state $|\,10\rangle$. Hence, $|00\rangle \models K_1 \neg |10\rangle$. Subsystem 1 does not implicitly know the local state of the other component when they are fully separable. So subsystem 1 does not implicitly know that the global state is not $|01\rangle$ or in other words: $|00\rangle \models \neg K_1 \neg |01\rangle$. Subsystem 2 implicitly knows that the global state cannot be $|11\rangle$, however subsystem 1 doesn't know that the local state of subsystem 2 is not $|\,1\rangle$. Hence subsystem 1 does not know that 2 excludes state $|11\rangle$, that is to say $|00\rangle \models \neg K_1 K_2 \neg |11\rangle$.

6 Correlated Knowledge

In addition to our presentation so far, our complex systems can be modeled more accurately if we add structure to our general epistemic frames and enrich the language of the logic. In this section we will only provide a brief sketch of how this can be brought about, leaving the further details to be explored in future work. The idea is to *capture explicitly the "observational capabilities" of the individual agents and of the groups.* This allows us to generalize the concrete semantics given by "interpreted systems" (i.e. the vector model representation of classical epistemic models) to a type of general epistemic frames that we call *correlation models.*

We generalize vector models in *three stages*: the first type of models we consider are *relation-based models.* In these models, the *states are relations between the agents' possible observations.* Given sets $O_1, \ldots, O_n$ of possible observations for each agent, a *joint observation* will be a tuple of observations $o = \vec{o} = (o_i)_{i \in N} \in O_1 \times \cdots \times O_n$. A state of the world can be characterized by the joint observations that can be performed on it, so a state is a set of such tuples namely a relation. A model will have as its state space any set $\Sigma \subseteq \mathcal{P}(O_1 \times \cdots \times O_n)$. The state s_I of a subsystem I of a global system in state s will be naturally given by the projection: $s_i = \{(o_i)_{i \in I} : o \in s\}$. So the observational equivalence is then given by: $s \overset{I}{\sim} t$ iff $\{(o_i)_{i \in I} : o \in s\} = \{(o_i)_{i \in I} : o \in t\}$.

The second, wider stage of generalization is given by *multi-set models.* Instead of sets of tuples of observations as in the relational models, we now consider multi-sets. Working with multisets has the advantage that we can model the case when agents record the frequencies of their observations. Now the states are multi-sets of joint observations, i.e. functions s from tuples of observations from $O_1 \times \cdots \times O_n$ into natural numbers. The state s_I of a subsystem I in a global state s will be naturally given by

$$s_I((e_i)_{i \in I}) := \sum \{s(o) : o \in O_1 \times \cdots \times O_n \text{ such that } o_i = e_i \text{ for all } i \in I\}.$$

The third type of models we consider are *correlation models.* We generalize natural numbers to an abstract set R of possible *observational results*, together with some abstract operation $\sum : \mathcal{P}(R) \to R$ of *composing results.* This operation may be *partial* (i.e. defined only for some subsets $A \subseteq R$), but it is required to satisfy the condition: $\sum\{\sum A_k : k \in K\} = \sum(\bigcup_{k \in K} A_k)$ whenever $\{A_k : k \in K\}$ are pairwise disjoint. In this case, $(R, \sum)$ will be called a *result structure.*

Correlation Models Given a result structure R and a tuple $\vec{O} = (O_i)_{i \in N}$ of sets of possible observations, a *correlation model over* $(R, \sum, \vec{O})$ is given by a set $\Sigma \subseteq \{s : s \text{ is a function} : O_1 \times \cdots O_n \to R\}$ of maps assigning results to (global) joint observations $o = (o_i)_{i \in N}$. So global states will then be functions from $O_1 \times \cdots \times O_n$ into R. We put $O_I := \times_{i \in I} O_i = \{(o_i)_{i \in I} : o_i \in O_i \text{ for every } i \in I\}$. As before, in a global state s, the state s_I of a subsystem I will be a map from $s_I : O_I \to R$, given by:

$$s_I((e_i)_{i \in I}) := \sum \{s(o) : o \in O_1 \times \cdots \times O_n \text{ such that } o_i = e_i \text{ for all } i \in I\}.$$

To put this more succinctly, for every tuple $e = (e_i)_{i \in I} \in O_I$ of I-observations, let

$$\overline{e} := \{o = (o_i)_{i \in N} \in O_1 \times \cdots O_n : o_i = e_i \text{ for all } i \in I\}.$$

Then we can define, for every $e \in O_I$:

$$s_I(e) = \sum \{s(o) : o \in \overline{e}\}.$$

Correlated Knowledge *Correlation models are general epistemic models*, in which we take our *observational equivalence to be identity of the corresponding local states*:

$$s \overset{I}{\sim} t \text{ iff } s_I = t_I.$$

The "group knowledge" K_I in a correlation model will be called *correlated knowledge.*

It is easy to see that, in general, *correlation models are not necessarily classical (as epistemic frames)*. Hence, *correlated knowledge is in general different from distributed knowledge.*

Examples:

- The *relation-based models* mentioned first can be recovered as special cases of correlation models, if we take $R = \{0, 1\}$ and logical *disjunction* as the composition operation.
- *Epistemic vector models* can be seen as special cases of relation-based models (in which every state is a singleton consisting of only one joint observation), and hence they also are correlation models.
- The *multi-sets models* are also correlation models, with R being the set of natural numbers, and *addition* as the composition operation.
- *Quantum epistemic systems* $\Sigma_1 \otimes \Sigma_2 \otimes \cdots \otimes \Sigma_n$ are correlation models, in which the sets of observations O_i are given by the (state spaces associated to) Hilbert spaces Σ_i. Joint observations $(o_i)_{i \in I}$ are interpreted as projectors onto the corresponding state in $\bigotimes_{i \in I} \Sigma_i$. The result structure is the interval $R = [0, 1]$ with renormalized addition. The "result" of a joint observation $(o_i)_{i \in I}$ made on a state s is interpreted as the *probability* that the outcome of a local measurement (in any basis that includes $o = \otimes_{i \in I} o_i$) of the I-subsystem of a (global system in) state s will be o. It is well-know that any quantum state $s \in \bigotimes_{i \in N} \Sigma_i$ is uniquely characterized (up to multiplication by a non-zero scalar) by the function mapping any fully separable state $o = o_1 \otimes \cdots o_n \in \Sigma_1 \times \cdots \times \Sigma_n$ to the probability $| < s, o > |^2$ of s collapsing to o (after a measurement in a basis that includes o).[3]

The Logic of Correlated Knowledge. We extend the general epistemic logic **GEL** with atomic sentences describing the results of possible joint observations by groups of agents, obtaining *the logic of correlated knowledge* **LCK**:

$$\varphi \quad ::= \quad p \quad | \quad o^r \quad | \quad \neg\varphi \quad | \quad \varphi \wedge \psi \quad | \quad K_I \varphi$$

where $r \in R$ and $o = (o_i)_{i \in I} \in O_I$ is a I-tuple of observations, for any subset $I \subseteq N$ of agents. (Recall that $O_I := \times_{i \in I} O_i$.) The *semantics* of o^r is naturally given by: $s \models o^r$ iff $s_I(o) = r$.

[3]A different type of relational models for a generalized version of QM is proposed in [17]. Note that our models are "relational" in the sense that *quantum "states" correspond in our settings to relations* (in relation-based models) *or functions* (in correlation models). In contrast, in the categorical approach of [17], *relations* (between finite sets) *play the role of morphisms*, i.e. they are *the analogue of linear maps* (between Hilbert spaces) in QM.

Notation For any group $I \subseteq N$, any set $E \subseteq O_I$ of I-observations and any E-tuple of results $r = (r_e)_{e \in E} \in R^E$, one for each observation $e \in E$, put

$$E^r := \bigwedge_{e \in E} e^{r_e}.$$

Proof System. Fix a finite set $N = \{1, \ldots, n\}$ of agents, a finite result structure $(R, \sum)$ and a tuple of finite observation sets $\vec{O} = (O_1, \ldots, O_n)$. The *proof system of* **LCK** *over* $(R, \sum, \vec{O})$ includes the rules and axioms 1-7 of the logic **GEL**, and in addition, for every $I \subseteq N$, the following axioms:

8. *Observations always yield results*: for every $I \subseteq N$, we have

$$\bigwedge_{o \in O_I} \bigvee_{r \in R} o^r$$

9. *Observations have unique results*: i.e. for $r \neq p$, $o \in O_I$, we have

$$o^r \Rightarrow \neg o^p$$

10. *Groups know the results of their (joint) observations*: for $o \in O_I$, $r \in R$, we have

$$o^r \Rightarrow K_I o^r$$

11. *Group knowledge is correlated knowledge (i.e. is based on joint observations)*: for every tuple $(r_o)_{o \in O_I}$ of results, one for each possible joint observation $o = (o_i)_{i \in I} \in O_I$ by group I, we have

$$(O_I^r \wedge K_I \varphi) \Rightarrow K_\emptyset (O_I^r \Rightarrow \varphi)$$

12. *Result Composition Axiom*: for every tuple $e = (e_i)_{i \in I} \in O_I$ of I-observations and every tuple $r = (r_o)_{o \in \overline{e}}$, one for each global observation $o \in \overline{e}$, put $\sum r := \sum \{r_o : o \in \overline{e}\}$; then we have

$$\overline{e}^{\,r} \Rightarrow e^{\sum r}$$

(In axioms 11 and 12 we used the notation E^r from above: first with $E = O_I$; then with $E = \overline{e}$, $I = N$.)

Theorem 2. ***(Soundness and Completeness for LCK)*** *For every finite set $N = \{1, \ldots, n\}$ of agents, every finite result structure $(R, \sum)$ and every tuple of finite observation sets $\vec{O} = (O_1, \ldots, O_n)$, the above proof system is sound and complete with respect to correlation models over $(R, \sum, \vec{O})$.*

Proof. *Soundness* is trivial: axioms 8 and 9 hold because each state s induces a function $s_I : O_I \to R$; axioms 10 and 11 hold because of the definition of indistinguishability in correlation models in terms of the s_I functions: $s \overset{I}{\sim} t$ iff $s_I = t_I$; axiom 12 holds due to the specific definition of the map s_I (in terms of the map s and of the composition operation $\sum$) in correlation models. For *completeness*, let φ_0 be a consistent sentence. First, we introduce the restricted canonical model Ω_0, constructed as in the proof of Theorem 1 (with the same definition of valuation extended to the atomic constants o^r, i.e. $\|o^r\| := \{T \in \Omega_0 : o^r \in T\}$). As in Theorem 1, we show that this is a general epistemic model and that it satisfies a "Truth Lemma" ($T \models \varphi$ iff $\varphi \in T$), and hence that φ_0 holds at T_0. Second, we can define a map $T \mapsto s_T$, associating to each theory $T \in \Omega_0$ some function $s_T : O_1 \times \cdots \times O_n \to R$, given for each $o \in O_1 \times \cdots \times O_n$ by: $s_T(o) = r$ iff $o^r \in T$. Axioms 8 and 9 (with $I = N = \{1, \ldots, n\}$) ensure that the map $T \mapsto s_T$ is *well-defined*, and axiom 11 (with $I = N$, hence with $K_I \varphi$ meaning the same as φ) ensures that this map is *injective*. Hence, we can "identify" theories T with the corresponding functions s_T, or in other words we can "lift" the epistemic model structure from Ω_0 to a subset $\{s_T : T \in \Omega_0\} \subseteq \{s : s \text{ is a function} : O_1 \times \cdots O_n \to R\}$, by putting $s_T \overset{I}{\sim} s_{T'}$ iff $T \overset{I}{\sim} T'$ and $\|p\| = \{s_T : p \in T\}$ (and similarly for the atomic constants o^r). We can easily see that *this is a correlation model*: axioms 10 and 11 ensure that the indistinguishability condition for correlation models ($s \overset{I}{\sim} t$ iff $s_I = t_I$) is satisfied, and axiom 12 ensures that our specific definition (in terms of the map s and of the composition operation $\sum$) of the map s_I in correlation models is satisfied. As general epistemic models, Ω_0 and this correlation model are in fact isomorphic, hence φ_0 is satisfied in (state s_{T_0} of) this correlation model. ⊣

7 Concluding Remarks

In this paper we modeled complex systems (from classical to quantum) using the setting of *General Epistemic Frames* and we introduced a particular type of such frames called *Correlation Models*. Our aim has been to throw new light on the difference between classical and quantum information and to gain a better understanding of quantum correlations. As such this paper should be of particular interest to quantum logicians, looking for an abstract formal logical setting that can naturally accommodate entanglement. Note that entanglement posed a problem to the lattice-theoretic approach of traditional Quantum Logic (see e.g. [1, 23]), which opened the road to the quest for possible solutions and alternative settings. Our paper might also be of interest to quantum information theorists who want to abstract away from Hilbert space models, looking for higher levels of abstraction similar to those successfully developed for classical computing. We note that a shorter version of this paper was presented during the 6th QPL workshop on Quantum Physics and Logic held at Oxford University in 2009 and will appear in [9].

Acknowledgements During the writing of this paper, S. Smets' research was supported by a post-doctoral fellowhip of the Flemish Fund for Scientific Research. S. Smets thanks the Flemish Academic Centre for Science and the Arts (VLAC) in Brussels for supporting part of this work by granting her a four-month VLAC-fellowship in 2008. During the writing of this paper, A. Baltag's research was partially supported by the Netherlands Organisation for Scientific Research (NWO), grant number B 62-635, which is herewith gratefully acknowledged. We would also like to kindly thank the anonymous referees for their very useful comments.

References

[1] D. Aerts, "Description of compound physical systems and logical interaction of physical systems". In E.G. Beltrametti and B. C. van Fraassen (Eds.), *Current Issues on Quantum Logic, Ettore Majorana, International Science Series, Physical Sciences*, vol.8., Dordrecht: Kluwer Academic, 381-405, 1981.

[2] S. Abramsky and B. Coecke, "A Categorical Semantics of Quantum Protocols", in the proceedings of the 19th IEEE conference on Logic in Computer Science (LiCS'04). Available at arXiv:quant-ph/0402130

[3] S. Abramsky and R. Duncan, "A Categorical Quantum Logic", *Mathematical Structures in Computer Science*, 469-489, 2006.

[4] A. Baltag and L. Moss and H. van Ditmarsch, "Epistemic Logic and Information Update", *Handbook on the Philosophy of Information*, Elsevier, the Netherlands, 2008.

[5] A. Baltag and S. Smets, "The Logic of Quantum Programs", in P. Selinger (ed.) *Proceedings of the 2nd International Workshop on Quantum Programming Languages (QPL2004), TUCS General Publication*, 33:39–56 Turku Center for Computer Science, 2004. PHILSCI00001799

[6] A. Baltag and S. Smets, "Complete Axiomatizations Of Quantum Actions", *International Journal of Theoretical Physics*, 44(12):2267–2282, 2005.

[7] A. Baltag and S. Smets, "LQP: The Dynamic Logic of Quantum Information", in *Mathematical Structures in Computer Science*, Special Issue on Quantum Programming Languages, 16(3):491–525, 2006.

[8] A. Baltag and S. Smets, "A Dynamic-Logical Perspective on Quantum Behavior", in L. Horsten and I. Douven, *Special Issue: Applied Logic in the Philosophy of Science*, *Studia Logica*, 89:185-209, 2008.

[9] A. Baltag and S. Smets, "Correlated Information: A Logic for Multi-Partite Quantum Systems" in B. Coecke, P. Panangaden and P. Selinger, *Electronic Notes in Theoretical Computer Science*, to appear 2010.

[10] J. Barwise, "On the model theory of common knowledge", *The Situation in Logic*, CSLI Lecture notes, Center for the Study of Language and Information, pp. 201-220, 1989.

[11] J. Barwise and J. Perry, *Situations and Attitudes*, MIT Press, 1983.

[12] J. van Benthem, *Exploring Logical Dynamics*, CSLI Publications, Stanford, 1996.

[13] J. van Benthem and M. Martinez, "The Stories of Logic and Information", in *Handbook of the Philosophy of Information*, Elsevier, the Netherlands, 2008. Available at http://dare.uva.nl/record/262024

[14] J. van Benthem, "Information as correlation vs. Information as range: a proposal for identifying and merging two basic logical traditions", in L. Moss, ed., *In Memory of Jon Barwise*, in press. Available at http://www.illc.uva.nl/Publications/ResearchReports/PP-2006-07.text.pdf.

[15] P. Blackburn, M. de Rijke and Y. Venema, *Modal Logic* Series: Cambridge Tracts in Theoretical Computer Science (No. 53), Cambridge University Press, 2001.

[16] B. Coecke, "The Logic of Entanglement", Research Report, March 2004, arXiv:quant-ph/040214.

[17] B. Coecke and B. Edwards, "Toy Quantum Categories", http://arxiv.org/abs/0808.1037. To appear in Proceedings of Quantum Physics and Logic 2008, *Electronic Notes in theoretical Computer Science.*

[18] M. L. Dalla Chiara, R. Giuntini and R. Leporini, "Quantum Computational Logics. A Survey", in V. Hendricks and J. Malinowski (eds.), *Trends in Logic. 50 Years of Studia Logica, Kluwer*, 229271, 2003.

[19] K. Devlin, *Logic and Information*, Cambridge University Press, 1991.

[20] S. Gudder, "Quantum Computational Logic", *International Journal of Theoretical Physics*, 42:1, 39-47, 2003.

[21] R. Fagin and J. Halpern, Y. Moses, M. Vardi, *Reasoning about Knowldege*, MIT Press, 1995.

[22] J.Y. Halpern and Y. Moses. "Knowledge and common knowledge in a distributed environment" *Journal of the ACM*, 37(3):549-587, 1990.

[23] F. Valckenborgh, "Compound Systems in Quantum Axiomatics", Doctoral Thesis, Vrije Universiteit Brussel, 2001.

Nonograms

K. Joost Batenburg[*] Walter A. Kosters[†]

Abstract

Nonograms, also known as *Japanese puzzles*, are in fact image reconstruction problems that can be solved by logic reasoning. Nonograms can have widely varying difficulty levels. Although the general Nonogram problem is NP-hard, the instances that occur in puzzle collections can usually be solved by hand.

This paper focuses on a subclass of Nonograms that can be solved by a sequence of local reasoning steps. A *difficulty measure* is defined for this class of so-called *simple* Nonograms, which corresponds to the number of steps required to reconstruct the image. In the first part of this paper, we investigate the difficulty distribution among this class, analyze the structure of Nonograms that have lowest difficulty, and give a construction for the asymptotically most difficult problems. We also provide some graphs to give insight into the search spaces of the problems at hand. The second part of the paper deals with the task of constructing Nonograms, based on a given gray level image. We propose an algorithm that generates a set of Nonograms of varying difficulty that all resemble the input image. Finally we mention some issues for non-simple Nonograms.

1 Introduction

A *Nonogram*, also known as a *Japanese puzzle* in some countries, is a type of logic puzzle which can be considered as an image reconstruction problem. The goal is to find an image on a rectangular pixel grid that adheres to certain row and column (briefly: line) constraints. Usually, the image is black-and-white, although Nonograms with more than two gray values exist as well. In addition to elementary logic, solving Nonograms requires some elementary integer calculations. The combination of a logic problem with integer calculations results in a combinatorial problem that can be approached using methods from combinatorial optimization, logical reasoning or both, which makes Nonograms highly suitable for educational use in Computer Science [13, 16].

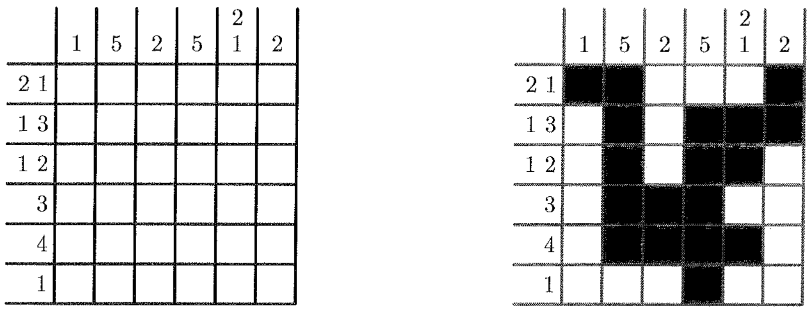
From a more general point of view, Nonograms fit into the concept of *Discrete Tomography*. This is an extensive research area, closely related to the field of *Computed Tomography*, with applications ranging from medicine to crystallography. For more information, the interested reader is referred to [6, 7, 4, 5].

Figure 1(a) shows an example of a small Nonogram. Its (unique) solution is shown in Figure 1(b). The *Nonogram description* for each row and column indicates the order and length of consecutive unconnected black segments along those lines. For example, the Nonogram description "2 1" in the first row indicates that from left to right, the row contains a black segment of length 2 followed by a single black pixel. The black segments are separated by one or more white pixels and there may be additional white pixels before the first segment, and after the last segment.

Several implementations of Nonogram solvers can be found on the Internet; see, e.g., [14, 19]. In [2, 10, 15, 16], evolutionary algorithms are described for solving Nonograms; and in [18] neural networks are used. A heuristic algorithm for solving Nonograms is proposed in [12]. The related problem of *constructing* Nonograms that are uniquely solvable is discussed

[*]CWI, Amsterdam, The Netherlands; `K.J.Batenburg@cwi.nl`

[†]LIACS, Universiteit Leiden, The Netherlands; `kosters@liacs.nl`



(a) 6 × 6 Nonogram (b) Solved Nonogram

Figure 1: A small Nonogram and its unique solution.

in [9]. In [3], a reasoning framework is proposed for solving Nonograms that uses a 2-SAT model for efficient computation of reasoning steps.

In [17], it was first proved that the general Nonogram problem is NP-hard. This also follows from the fact that Nonograms can be considered as a generalization of the reconstruction problem for hv-convex sets in discrete tomography, which is NP-hard [20]. On the other side of the difficulty spectrum are the Nonograms that can be found in puzzle collections, which can usually be solved by hand, applying a sequence of elementary reasoning steps. In this paper, we focus on this latter class of Nonograms, referred to as the *simple* type in [3]. Such Nonograms can be solved without resorting to branching, yet there can still be a large variance in the number of steps required to find solutions. In [1] a difficulty measure for this class is proposed and analyzed. In particular, a construction for a family of Nonograms that have asymptotically maximal difficulty, up to a constant factor, is provided.

This paper, based on [1, 3], is structured as follows. In Section 2, notation is introduced to describe the objects of this paper and their properties. We provide an efficient algorithm to process single lines, leading to a Nonogram solver that deals with so-called simple Nonograms. Both the simple class and the difficulty measure are defined in Section 3, where also further motivation is provided for studying this particular difficulty measure, and its distribution is analyzed for small Nonograms. Section 4 considers the question what the maximum difficulty can be, as a function of Nonogram size. A construction is given that obtains asymptotically maximal difficulty for Nonograms of arbitrarily large size. Section 5 deals with an application of this difficulty concept: constructing Nonograms of varying difficulty that resemble a gray level input image. An algorithm is proposed for this task, illustrated by some computational experiments. In Section 6 we mention some results (taken from [3]) that extend beyond those for simple Nonograms. Section 7 concludes this paper.

2 Basic notions and algorithms

We first define notation for a single line (i.e., row or column) of a Nonogram. After that, we combine these into rectangular puzzles. Let $\Sigma = \{0, 1\}$, the alphabet of pixel values (more general alphabets are also allowed). We usually refer to 1 as *black* and 0 as *white*. While solving a Nonogram, the value of a pixel can also be *unknown*. Let $\Gamma = \Sigma \cup \{?\} = \{0, 1, ?\}$, where the symbol ? refers to the unknown pixel value.

A *(general) description* d of length $k > 0$ is an ordered series $(d_1, d_2, \ldots, d_k)$ with $d_j = \sigma_j\{a_j, b_j\}$, where $\sigma_j \in \Sigma$ and $a_j, b_j \in \{0, 1, 2, \ldots\}$ with $a_j \leq b_j$ $(j = 1, 2, \ldots, k)$. The curly braces are used here in order to stick to the conventions from regular expressions; so, in $\sigma_j\{a_j, b_j\}$ they do not refer to a set, but to an ordered pair. Any such d_j will correspond with between a_j and b_j characters σ_j, as defined below. Without loss of generality we will assume that consecutive characters σ_j differ, so $\sigma_j \neq \sigma_{j+1}$ for $j = 1, 2, \ldots, k-1$. We will sometimes write σ^* as a shortcut for $\sigma\{0, \infty\}$ (for $\sigma \in \Sigma$) and σ^+ as a shortcut for $\sigma\{1, \infty\}$, where ∞ is suitably large number. We use σ^a as a shortcut for $\sigma\{a, a\}$ $(a \in \{0, 1, 2, \ldots\})$, and we sometimes omit parentheses and commas; also σ^0 is omitted. A finite string s over Σ *adheres* to a description d (as defined above) if $s = \sigma_1^{c_1}\sigma_2^{c_2}\ldots\sigma_k^{c_k}$, where $a_j \leq c_j \leq b_j$ for $j = 1,\ldots,k$. As an "example", consider the following description:

$$d = (0\{0,\infty\}, 1\{a_1,a_1\}, 0\{1,\infty\}, 1\{a_2,a_2\}, 0\{1,\infty\},\ldots,1\{a_r,a_r\}, 0\{0,\infty\})$$

with $a_i > 0$ ($i = 1,2,\ldots,r$). This is exactly what we consider to be a *Nonogram description* $a_1a_2\ldots a_r$ for a line (row or column), where we only mention the lengths of consecutive non-touching series of 1s. Note that it has length $2r+1$ and can also be written as $0^*1^{a_1}0^+1^{a_2}0^+\ldots1^{a_r}0^*$.

A string $s \in \Gamma^\ell$ ($\ell \geq 0$) can be *(fully) fixed* to a string $t \in \Sigma^\ell$ (referred to as a *fix*) if $s_j = t_j$ whenever $s_j \in \Sigma$ ($1 \leq j \leq \ell$). Loosely speaking, one should replace the ?s, or unknowns, with pixel values; we also say that we fix these string elements. If $s \in \Gamma^\ell$ can be fixed to a string in Σ^ℓ that adheres to a given description d, s is called *fixable* with respect to d; in that case the Boolean function value $\mathit{Fix}(s,d)$ is defined to be `true`, and otherwise `false`. The formal operation *Settle* (s,d) constructs a (unique) string from a fixable string s and a description d by replacing all ? symbols in s for which all strings in Σ^ℓ that adhere to the description d have the same unique value, by this value. In other words, all pixels that must have a certain value in order to adhere to the description, are set to that value. As an example, for $s =$?1?1?0????? (with $\ell = 11$) and Nonogram description $d = 3\ 2\ 1$ (so general description $0^*1^30^+1^20^+1^10^*$), we have *Settle* $(s,d) =$ 011100?1???.

In [3], an efficient, polynomial-time algorithm is described for performing the *Settle* operation on a string, by using dynamic programming. Before we elaborate on this, we introduce some more general notations, also because it makes it easier to formulate our results. For a string $s = s_1s_2\ldots s_\ell$ of length ℓ over Γ, define its prefix of length i by $s^{(i)} = s_1s_2\ldots s_i$ ($1 \leq i \leq \ell$), so $s = s^{(\ell)}$; $s^{(0)}$ is the empty string. Similarly, for a description $d = (d_1,d_2,\ldots,d_k)$, put $d^{(j)} = (d_1,d_2,\ldots,d_j)$ for $1 \leq j \leq k$, so $d = d^{(k)}$; $d^{(0)} = \epsilon$ is the empty description. Furthermore, let $A_j = \sum_{p=1}^{j} a_p$ and $B_j = \sum_{p=1}^{j} b_p$; put $A_0 = B_0 = 0$. We note that a string of length $\ell < A_k$ is certainly not fixable with respect to d, simply because it has too few elements; similarly, a string of length $\ell > B_k$ is not fixable with respect to d. Finally, for $\sigma \in \Sigma$, $s \in \Gamma^\ell$ and $1 \leq i \leq \ell$, let $L_i^\sigma(s)$ denote the largest index $h \leq i$ such that $s_h \notin \{\sigma, ?\}$, if such an index exists, and 0 otherwise. We will put $\mathit{Fix}(i,j) = \mathit{Fix}(s^{(i)}, d^{(j)})$. The value $\mathit{Fix}(\ell,k)$ then determines whether s is fixable with respect to d.

The value $\mathit{Fix}(i,j)$ can be expressed recursively using only terms $\mathit{Fix}(i',j')$ with $i' < i$ and $j' < j$. This allows for efficient evaluation of $\mathit{Fix}(i,j)$ by dynamic programming. As boundary values we note that $\mathit{Fix}(0,j) =$ `true` if and only if $A_j = 0$ ($j = 0,1,2,\ldots,k$); and $\mathit{Fix}(i,0) =$ `false` for $i = 1,2,\ldots,\ell$. We clearly have $\mathit{Fix}(i,j) =$ `false` if $i < A_j$ or $i > B_j$ ($0 \leq i \leq \ell$, $0 \leq j \leq k$), as indicated above.

The main recursion, valid for general alphabets Σ, is:

Proposition 1 *The function Fix satisfies*

$$\mathit{Fix}(i,j) = \bigvee_{p=\max(i-b_j,A_{j-1},L_i^{\sigma_j}(s))}^{\min(i-a_j,B_{j-1})} \mathit{Fix}(p,j-1) \qquad (1)$$

This holds for i and j with $1 \leq i \leq \ell$, $1 \leq j \leq k$ and $A_j \leq i \leq B_j$.

Note that an empty disjunction is `false`; this happens for example if $L_i^{\sigma_j}(s) \geq i - a_j + 1$. For $j = 1$ we have $\mathit{Fix}(i,1) =$ `true` if and only if $L_i^{\sigma_1}(s) = 0$.

Proof The validity of the recursion can be shown as follows. The last part of $s^{(i)}$ must consist of between a_j and b_j characters σ_j; say we want σ_j at positions $p+1, p+2,\ldots,i$. We then must have $a_j \leq i - p \leq b_j$. Also note that all elements $s_{p+1}, s_{p+2},\ldots,s_i$ must be either ? or σ_j; this holds exactly if $L_i^{\sigma_j}(s) \leq p$. Finally, the first part of $s^{(i)}$, i.e., $s^{(p)}$, must adhere to $d^{(j-1)}$. Clearly, p must be between A_{j-1} and B_{j-1}, otherwise this would not be possible. □

Note that the A_j and B_j terms can be considered to represent general tomographic restrictions. It is natural to implement this recursive formula by means of dynamic programming, using lazy evaluation: once a `true` $Fix(p, j-1)$ is found, the others need not be computed.

Now given a string s over Γ that is fixable with respect to a description d, it is easy to find those unknowns that have the same value from Σ in every fix: these elements are then set at that value. Indeed, during the computation of $Fix(s, d)$ (which of course yields `true`), one can keep track of all possible values that lead to a fix. In Equation (1) those $Fix(p, j-1)$ that are `true` correspond with a fix, where the string elements $s_{p+1}, s_{p+2}, \ldots, s_i$ are all equal to σ_j. Now one only has to verify, for each string element of s, whether precisely one element from Σ is allowed. In practice this can be realized by using a separate string, whose elements are filled when "specifying" s, and where those elements that are filled only once are tagged. (Here the fact that $|\Sigma| = 2$ comes in handy.) Note that for this purpose lazy evaluation is not an option, since we need to examine all fixes.

The complexity of the computation of $Fix(\ell, k)$ is bounded by $k \cdot \ell^2$: at most $k \cdot \ell$ values of $Fix(i, j)$ must be computed, and each such computation can be performed in $O(\ell)$ time, including the evaluation of $L_i^{\sigma_j}(s)$. In practice, especially when using lazy evaluation, the complexity is much lower.

An $m \times n$ *Nonogram puzzle description* D consists of $m > 0$ row Nonogram descriptions $r_1, r_2, \ldots, r_m$ and $n > 0$ column Nonogram descriptions $c_1, c_2, \ldots, c_n$. An *image* $P = (P_{ij}) \in \Sigma^{m \times n}$ *adheres* to the description if all lines adhere to their corresponding description. A *Nonogram* N consists of a pair (D, P), where D is a Nonogram puzzle description and P is an image in $\Gamma^{m \times n}$. Usually we will assume that all lines in P are still fixable with respect to their corresponding Nonogram descriptions. Solving such a puzzle means finding an image $P' \in \Sigma^{m \times n}$ that adheres to D, and where every line in P is fixed to the corresponding line in P'. The image P can be viewed as a partial solution.

3 The difficulty of simple Nonograms

A Nonogram puzzle description is called *simple* if it can be (uniquely) solved by applying a sequence of *Settle* operations, each time using only information from a single row or column, starting from an image with only ?s. In other words, it is never necessary to consider information from several rows and columns simultaneously. Nearly all Nonograms that appear in puzzle collections satisfy this property. From this point on, we focus on the class of *simple* Nonograms. Note that for Nonograms of the simple type, there is a bijective map between the set of images and their descriptions. Therefore, we sometimes use the term *Nonogram* to refer to either the image, or its description.

Even though the order of applying the *Settle* operations does not affect whether or not a solution can be found, the required number of operations depends heavily on the order in which rows and columns are selected. We define the following operations:

- The operation *h-sweep* (N) applies the *Settle* operation to all rows of the Nonogram N: a horizontal sweep.
- The operation *v-sweep* (N) applies the *Settle* operation to all columns of the Nonogram N: a vertical sweep.

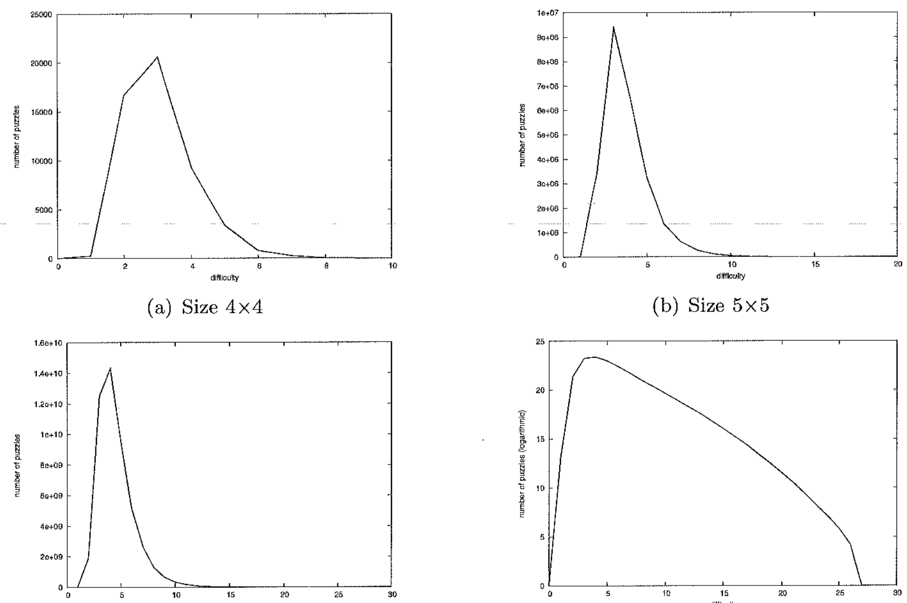
Both operations return the "updated" Nonogram, usually having fewer unknowns.

Now the difficulty of a Nonogram of the simple type is determined by starting with an image N for which all pixel values are unknown, and running the algorithm in Figure 2. The algorithm starts with a horizontal sweep, intertwines horizontal and vertical sweeps, and counts the total number of sweeps until the Nonogram is solved. It is clear that any $m \times n$ Nonogram of simple type has complexity at most equal to $mn+1$, since every sweep (except perhaps the first one) must at least fix one unknown pixel. By definition, the algorithm terminates if and only if N is of the simple type.

```
Difficulty (N) :
    diff ← 0;
    while N is not solved do
        if diff is even then N ← h-sweep (N);
        else N ← v-sweep (N); fi
        diff ← diff + 1;
    od
    return diff;
```

Figure 2: Algorithm that solves a *simple* Nonogram and determines its difficulty.

We remark that our definition of "difficulty" is rather subjective. Quantifying the amount of work required to solve a particular Nonogram is not straightforward, as it depends on the particular solution strategy employed. In Section 5, we will consider the task of constructing Nonograms of *varying difficulty* that resemble a gray level input image. As these Nonograms are intended to be solved by human puzzlers, it is important that the difficulty measure corresponds to the amount of work required by a puzzler to solve the Nonogram. We observed that while solving Nonograms, people rarely combine information from several rows and columns simultaneously, which motivates studying the simple class. A major advantage of the proposed measure is that it does not depend on the *order* in which individual rows and columns are considered. The only degree of freedom in this strategy is whether one starts with the rows or columns. This choice can make a difference of at most 1 in the resulting difficulty. An interesting property of Nonograms is that small local changes in the image can have a profound impact on the solution process for the corresponding Nonogram. The difficulty can vary wildly, by changing just a single pixel.



(a) Size 4×4

(b) Size 5×5

(c) Size 6×6

(d) Idem as (c), logarithmic

Figure 3: Number of Nonograms of a given size as a function of difficulty level. In (d), the vertical axis has logarithmic scaling.

The proposed difficulty measure can be computed efficiently, by using the *Settle* algorithm from Section 2. This allows for enumeration of a large set of Nonograms, to perform a statistical analysis of the difficulty distribution. Figure 3(a-c) shows the difficulty histogram for all simple square Nonograms of size 4×4 up to 6×6, obtained by a complete enumeration.

Note that all these Nonograms have a unique solution. It can be observed that a large fraction of all simple $n \times n$ Nonograms ($4 \le n \le 6$) has low difficulty (close to n), while high difficulty Nonograms (difficulty close to $\frac{1}{2}n^2$) occur rarely. For $n = 6$, out of $2^{36} \approx 7 \cdot 10^{10}$ images, 70.76 % yields a Nonogram of the simple type. (In Section 6 we will discuss the remaining ones.) Figure 3(d) shows the same histogram using a logarithmic scale. It can be observed that the average difficulty is 4.51, whereas the difficulty can be as large as 26.

Similar trends can be observed for larger Nonograms, but an exhaustive search is no longer easily possible in that case. An interesting question is how the maximum possible difficulty varies with Nonogram size. A Nonogram of high difficulty should satisfy two properties:

- In each consecutive *h-sweep* and *v-sweep* only a few new pixels should be determined. Ideally, this number of newly discovered pixel values should be bounded by a constant.
- In each consecutive *h-sweep* and *v-sweep* the value of at least one new pixel should be determined, as otherwise the Nonogram is not of the *simple* type.

For a Nonogram of size $n \times n$, $n^2 + 1$ is obviously an upper bound on its difficulty. However, it is not clear at all that the maximum difficulty that can be reached increases linearly with the number of pixels. In the next section, we will show how to construct arbitrarily large Nonograms for which the asymptotic difficulty is $\frac{1}{2}n^2$, which demonstrates that the upper bound can be attained up to a constant factor.

Before we examine these Nonograms of high difficulty, we will consider two boundary cases; proofs can be found in [3]. We first note that it is easy to see whether a line can be fully fixed by a single application of the *Settle* operation. Indeed, let $d = a_1 a_2 \dots a_k$ be a Nonogram description with $\sum_{i=1}^{k} a_i + k - 1 \le \ell$ (which means that $?^\ell$ is fixable with respect to d). Then we have *Settle* $(?^\ell, d) \in \Sigma^\ell$ if and only if $\sum_{i=1}^{k} a_i + k - 1 = \ell$. This property can be easily used to characterize Nonograms of difficulty 1: all rows should have it. Finally, we characterize those situations where the *Settle* operation cannot infer any information:

Lemma *Let $d = a_1 a_2 \dots a_k$ again be a Nonogram description with $\sum_{i=1}^{k} a_i + k - 1 \le \ell$. Then we have Settle $(?^\ell, d) = ?^\ell$ if and only if $\sum_{i=1}^{k} a_i + k - 1 \le \ell - \max_{1 \le i \le k} a_i$.* $\square$

4 Difficult Nonograms

In this section we will construct certain $m \times n$ Nonograms of the simple type that require approximately $\frac{1}{2}mn$ sweeps, thereby attaining a very high difficulty. More precisely, we show:

Theorem *Let m satisfy $m = 8k + 2$ for some integer $k \ge 1$, and take an even integer n with $n \ge 14$. Then there exists an $m \times n$ Nonogram that requires $A(m, n) = (m+2)(2n-15)/4+10$ sweeps (if $k > 1$). If $k = 1$, so $m = 10$, the Nonogram requires $6n-37$ sweeps. For square $n \times n$ Nonograms of this special type (so $n = 8k + 2$ with integer $k \ge 2$) we need $(n^2 - \frac{11}{2}n + 5)/2$ sweeps.*

The remaining part of this section is devoted to the construction of these special $m \times n$ Nonograms, and to the proof that the number of sweeps is equal to $A(m, n)$, as mentioned in the theorem. Slightly abusing notation, we will employ regular expressions for Nonogram descriptions, e.g., we will use $2(13)^2$ instead of the "official" 2 1 3 1 3; and we will write "description" instead of the more formal "Nonogram description".

Figure 4 shows the construction for $m = n = 18$. It is possible to give similar constructions for slightly varied values of m and n, for instance for odd width, but we will not go into detail on this. The slightly different value (2 less than the general formula predicts) if $k = 1$ is explained by a small case difference in the construction, see below.

The construction proceeds as follows. There are k rows with description n, i.e., consisting of only 1s. These rows, the so-called *split rows*, being the $(8i - 1)$th rows ($1 \le i \le k$), are fully fixed in the first *h-sweep*. Furthermore, all columns, except for the first, second and last one, have description 1^{3k+1} (where σ^r denotes a sequence of r copies of a sequence σ).

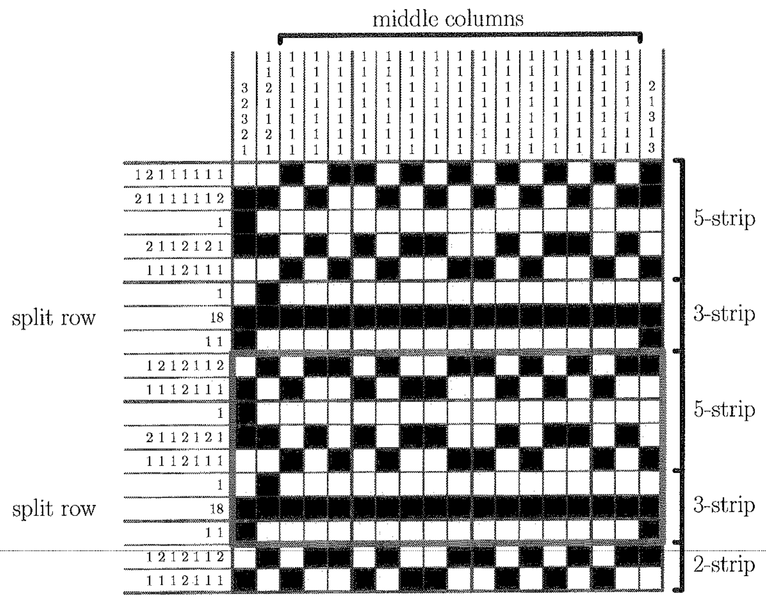


Figure 4: Overview of the construction of an 18 × 18 Nonogram with difficulty 115. The construction can be extended in the vertical direction by inserting consecutive copies of the marked block. Extension in the horizontal direction is straightforward.

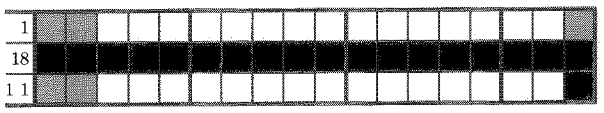


Figure 5: Contents of a 3-strip after the third sweep ($n = 18$). Gray squares denote unknown pixels.

After the first *v-sweep*, the rows immediately above and below the split rows are therefore filled with 0s in all these columns, referred to as the *middle columns*. Any three such rows together, i.e., split row and rows immediately above and below it, form a so-called *3-strip*. Each row above a split row has description 1, each row below a split row has description 1^2. The second 1 can in the third sweep also be fixed easily at the end of the row. So together any 3-strip will —after the third sweep— look like Figure 5.

Now the Nonogram is in fact separated by the 3-strips into k parts of height 5, called the *5-strips*, and a final part consisting of the bottom two rows, called the *2-strip*. All these parts must be solved in turn, as will be clear from the sequel.

Note that the 5-strips are all alike, except for the first one, which is used for bootstrapping the solver procedure. Within each 5-strip, the middle row will be filled with 10^{n-1} after the third sweep (its description is a single 1), and then the top two rows of the 5-strip will be solved, largely pixel by pixel, from right to left; after that the bottom two rows of the 5-strip will be solved in a similar fashion, from left to right, again largely pixel by pixel. The traversals from the top two rows to the bottom two rows within each 5-strip, and from each 5-strip to the next 5-strip or the final 2-strip, require special care. These traversals, combined with 5-strip solving, all invert the direction in which pixels are fixed, thereby constituting a zig-zag pattern.

The descriptions for the first, second and last columns are $(32)^k1$, $(112)^k1$ and $2(13)^k$, respectively.

Let us, to begin with, concentrate on the first (and special) 5-strip. The descriptions of its rows are $121^{n/2-3}$, $21^{n/2-3}2$, 1 (as said above), $21^{n/2-7}(21)^2$ and $1^{n/2-6}21^3$, respectively. The other 5-strips have a slightly different description for the first two rows, namely $(12)^21^{n/2-7}2$

and $1^3 21^{n/2-6}$. The row descriptions for the final 2-strip are the same: these two rows can be viewed as the top part of a regular 5-strip. In Figure 4 the resulting solved 18×18 Nonogram is shown; note the two split rows.

One can verify that after the first three sweeps, the following pixels are fixed: most pixels from the 3-strips (as mentioned above, cf. Figure 5; the five remaining unknown pixels are used for the traversals within and between the 5-strips), the bottom right pixel of the Nonogram (at 0), the two topmost pixels of the second columns from the left and right (at 01), and the entire middle row from each 5-strip (at 10^{n-1}, as said above). This last filling has the important property that in the middle columns, all 1s are now almost pinned: they must be in either first or second row, fourth or fifth row, and so on. This enforces that all 5-strips must be solved in order, and really after one another.

Concentrating on the first two rows, one can see that after the fourth sweep, only six pixels are fixed. The order, or rather the number of the sweep in which the pixels are found (again for $n = 18$; circles denote black pixels) is shown in Figure 6. Here, for each two unknown pixels immediately above one another (except for the leftmost two, where this fact is not known yet), exactly one must be 1. This is inferred pixel by pixel, coming from the right, and alternating between top and bottom row, thus contributing to the large number of sweeps needed. The 2s in the descriptions are necessary for the construction of the traversal; this also holds, in several variations, for other rows in 5-strips. Note that in the third sweep no new pixel values are found for these rows.

31	2	$\textcircled{30}$	27	$\textcircled{27}$	$\textcircled{26}$	23	$\textcircled{22}$	19	$\textcircled{18}$	15	$\textcircled{14}$	11	$\textcircled{10}$	7	$\textcircled{6}$	2	$\textcircled{4}$
$\textcircled{29}$	$\textcircled{1}$	29	$\textcircled{28}$	28	25	$\textcircled{24}$	21	$\textcircled{20}$	17	$\textcircled{16}$	13	$\textcircled{12}$	9	$\textcircled{8}$	5	$\textcircled{1}$	$\textcircled{4}$

Figure 6: Order in which pixel values are found for the first two rows.

Finally, let us examine the number of sweeps. From the construction it is clear that the addition of *two* new columns (among the middle columns) increases the number of sweeps by $m + 2$. Indeed, in every 5-strip we need an extra 8 sweeps, and the final 2-strip adds another 4; together we get $8k + 4 = m + 2$ of them. Therefore, $A(m, n)$ should satisfy $A(m, n + 2) = A(m, n) + m + 2$.

Furthermore, it is easy to see that every extra 5-strip and its accompanying 3-strip (as shown in Figure 4) adds $4n + c$ sweeps, for some integer constant c. Careful inspection shows that $c = -30$. We conclude that $A(m, n)$ should satisfy $A(m + 8, n) = A(m, n) + 4n - 30$. Using $A(18, 18) = 115$ we arrive at the closed formula.

Note that the traversal within the first 5-strip, that reverses the right-left direction into a left-right direction, slightly differs from those in the other 5-strips. This causes the small difference in the number of sweeps for $k = 1$. $\square$

5 Generating simple Nonograms

In this section we describe an algorithm that produces a series of Nonograms of the simple type of varying difficulty. The algorithm is rather flexible and offers many options that can be customized. We only sketch these options here. The website [8] offers an implementation.

The generated Nonograms should resemble a given gray value image $P \in \{0, \ldots, 255\}^{m \times n}$ We want these Nonograms not to look alike, and therefore maintain a set $\mathcal{L}$ of Nonograms from which a newly generated Nonogram should differ. The newly found Nonogram is then appended to $\mathcal{L}$.

As a subroutine, the algorithm for generating Nonograms uses a straightforward generalization of the *Difficulty* algorithm from Figure 2, referred to as *FullSettle*: instead of the difficulty, the *FullSettle* operation returns the set of unknown pixels, where we let the sweeps continue until they make no further progress. (This is equivalent to saying that there is no single *Settle* operation that reveals an unknown pixel.) Note that in Section 2 the *Difficulty* algorithm was applied to Nonograms of the simple type, where the algorithm terminates by definition, whereas in the current application the Nonograms may not be solved, and termination of *FullSettle* is effected when a sweep does not yield any new fixed pixels.

Furthermore, a function *Init* (P) is used, that returns a 0–1 Nonogram that somehow resembles P, e.g., by applying a threshold operation or a binary edge detection filter to the gray level input image.

```
Generate $(P, \mathcal{L})$ :
    $p \leftarrow$ Init $(P)$; $U \leftarrow$ FullSettle $(p)$;
    while $U \neq \varnothing$ do $p \leftarrow$ Adapt $(p, U, P, \mathcal{L})$; $U \leftarrow$ FullSettle $(p)$; od
    return $(p$, Difficulty $(p))$;
```

Figure 7: Algorithm that generates a uniquely solvable Nonogram and its difficulty.

Pseudo-code for the algorithm *Generate* is shown in Figure 7. The main ingredient is the function *Adapt* $(p, U, P, \mathcal{L})$ that returns a Nonogram p' that is equal to p, except for (at least) one pixel that is 0 in p but is 1 in p'. Note that, since the number of black pixels strictly increases, the loop in *Generate* indeed terminates: an all black Nonogram is certainly uniquely solvable. Also note that upon entering *Adapt* at least one $(i, j) \in U$ satisfies $p_{ij} \neq 0$; indeed, if *FullSettle* $(p) \neq \varnothing$, it cannot be the case that all the unknown pixels must be 1. The function *Adapt* proceeds as in Figure 8.

```
Adapt $(p, U, P, \mathcal{L})$ :
    $min \leftarrow \infty$;
    for all $(i, j) \in U$ (in random order) do
        if $p_{ij} = 0$ then
            $p_{ij} \leftarrow 1$; % try new image, that differs in one pixel
            $value \leftarrow \alpha \cdot |FullSettle\ (p)| + \beta \cdot P_{ij} + \gamma \cdot \sum_{L \in \mathcal{L}} L_{ij}$;
            if $value < min$ then $min \leftarrow value$; $(k, \ell) \leftarrow (i, j)$; fi
            $p_{ij} \leftarrow 0$; % restore original image
        fi
    od
    $p_{k\ell} \leftarrow 1$;
    return $p$;
```

Figure 8: Algorithm that slightly adapts an image p.

Here suitable non-negative parameters α, β and γ must be chosen. So we want the Nonogram to have a small amount of unknowns, we would like the changed pixel to be dark in the original image, and many Nonograms from $\mathcal{L}$ to be white in that particular pixel. If $\alpha = \gamma = 0$, the final Nonogram will resemble the original P, but will usually be quite dark. However, if $\beta = 0$, resemblance will be worse. High γ-values ensure diversity. Clearly, in particular if $\mathcal{L}$ is large, it might be hard or even impossible to guarantee that the generated Nonogram sufficiently differs from those in $\mathcal{L}$.

In this way we get Nonograms of different difficulty, but usually quite hard ones. In order to obtain Nonograms of more varying and usually lower difficulty, the algorithm from Figure 9 can be used. It returns a set of at most *depth* uniquely solvable Nonograms together with their difficulties, whose sets of black pixels strictly include that of the original p, and can therefore in general be expected to have lower difficulty. Note that each Nonogram added to $\mathcal{M}$ has at least one black pixel more than its predecessor. In fact, in practice uniquely solvable Nonograms are encountered in nearly every iteration.

Figure 10 contains some examples. All pictures are of size 30×38. The first picture is the original gray value image, from which the second is obtained by thresholding (aiming at 35 % black pixels). For the third picture, an edge detection filter is applied. For the fourth picture, empty lines were addressed (in Figure 10 this visible near the ear). The pictures in the middle row are Nonograms of the simple type that have been generated consecutively by the Generate algorithm starting from the third picture in the top row, and can therefore

```
$Vary\ (p, P, \mathcal{L}, depth)$ :
    $\mathcal{M} \leftarrow \varnothing$; $d \leftarrow 0$;
    while $d < depth$ and $p$ has white pixels do
        $min \leftarrow \infty$; $U \leftarrow$ {white pixels in $p$};
        for all $(i, j) \in U$ (in random order) do
            $p_{ij} \leftarrow 1$; $value \leftarrow \alpha \cdot |FullSettle\ (p)| + \beta \cdot P_{ij} + \gamma \cdot \sum_{L \in \mathcal{L}} L_{ij}$;
            if $value < min$ then $min \leftarrow value$; $(k, \ell) \leftarrow (i, j)$; fi
            $p_{ij} \leftarrow 0$;
        od
        $p_{k\ell} \leftarrow 1$; $d \leftarrow d + 1$;
        if $|FullSettle\ (p)| = 0$ then $\mathcal{M} \leftarrow \mathcal{M} \cup \{(p, Difficulty\ (p))\}$; fi
    od
    return $\mathcal{M}$;
```

Figure 9: Algorithm that generates Nonograms of varying difficulty.

expected not to look alike entirely; the numbers indicate the difficulties; the parameters of *Generate* were set at $\alpha = \gamma = 8$ and $\beta = 1$. The pictures in the bottom row are obtained from those immediately above them by the *Vary* algorithm with $depth = 60$; the final Nonogram generated in the main loop is depicted (other ones could also have been chosen). The set $\mathcal{L}$ contains the Nonograms from the second line created so far. Note that usually the difficulty decreases steadily during this process, but certainly not always.

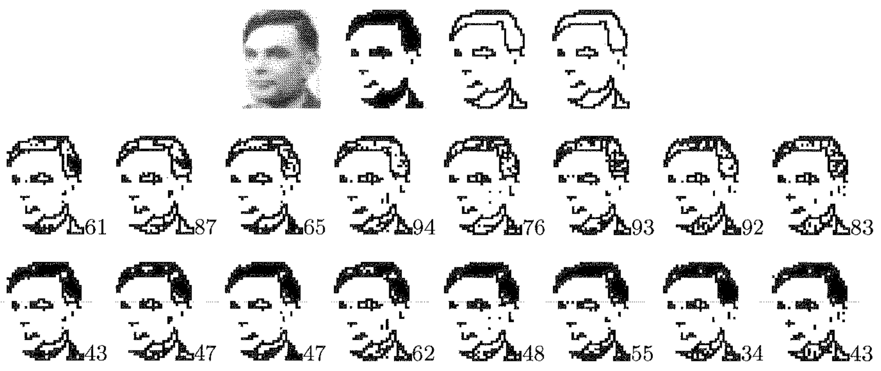


Figure 10: Nonograms of Alan Turing (1912–1954). The theme "Enigma" seems appropriate.

6 ... and beyond

In this section we will mention some issues related with non-simple Nonograms. The presentation is based on [3], but here we will restrict ourselves to examples and graphs to illustrate our points.

Let us first address other solving strategies. In the example from Figure 11a, *Settle* operations do not provide any further information (note the row with the empty Nonogram description ϵ, which is only adhered to by a string with five 0s). Therefore, the Nonogram is not of the simple type. It is, however, uniquely solvable. This can even be proved by straightforward logic. For the four pixels a, b, c and d in the bottom left corner one can deduce a dependency graph as in Figure 11b, where, e.g., an arrow from d to $\neg b$ means that if d is black, b must be white. This follows from the description of the second column. Equivalently, one can look at the corresponding Boolean formula, part of it being $(\neg d \vee \neg b)$. Indeed, we arrive at a 2-SAT problem in this way.

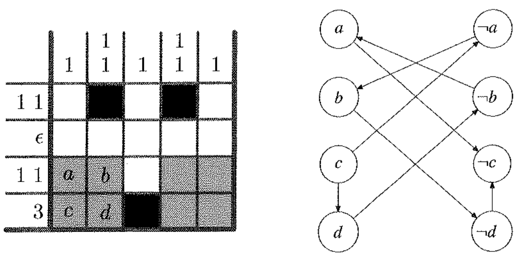


Figure 11: (a) Partially solved Nonogram; (b) (part of) its corresponding dependency graph.

We can now search for variables that must have the same truth value in *all* satisfying assignments. Assume that at least one such assignment exists. Then a variable x is `false` in all satisfying assignments if and only if there is a path from x to $\neg x$ in the dependency graph. Alternatively, x must be `true` in all satisfying assignments if and only if there is such a path from $\neg x$ to x. For the example from Figure 11a we can infer that c is `false` (or 0); now a few more *Settle* operations suffice to solve the puzzle.

Note that existence of a cycle in the dependency graph, containing both x and $\neg x$, implies that no satisfying truth assignment exists. If we can assume that a given Nonogram has at least one solution, and that we only fix the value of pixels that must have the same value in all solutions, such a cycle will never occur.

The dependency graph model provides a polynomial-time algorithm for finding all variables that must have the same value in all satisfying assignments of the 2-SAT problem. Note that it is indeed possible to apply general SAT- or CSP-solvers to the problem of solving Nonograms; see [3] for some more information.

In Section 3 all Nonograms of small sizes were enumerated, see Figure 3. Figure 11 presented a small Nonogram of non-simple type, where its solution required some more sophisticated arguments. However, as the example from Figure 12 shows, small Nonograms can still be harder. The above-mentioned 2-SAT approach does not yield any progress here.

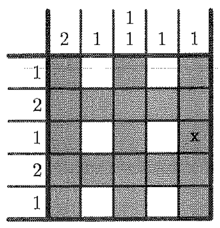
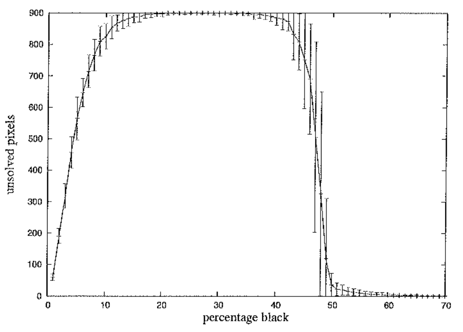


Figure 12: Partially solved 5 × 5 Nonogram, where the fact that pixel `x` must be white is hard to infer. The gray pixels are still unknown.

It is clear that different levels of solvers exist, the one that by definition solves simple Nonograms (called *FullSettle* in Section 5) being the easiest. Figure 13 and Figure 14 provide some statistical information regarding this issue, where different percentages of black pixels and puzzle sizes are addressed. Here, *Solver0* applies a combination of the *Settle* operation and the 2-SAT approach, as described above. And *Solver1* furthermore also allows for single "guesses" of the value of a pixel, which in case of a contradiction can be fixed to the other value.

We finally mention some issues concerning (non-)solvability of Nonograms, and we pay special attention to puzzles with multiple solutions. Nonograms (not those in puzzle collections, of course) can have more than one solution. Figure 15a shows a small example, referred to as an *elementary switching component*: four pixels, two black and two white, such that interchanging the black and the white pixels does not change the description. However, where the non-uniqueness problem for Discrete Tomography allows an elegant description based



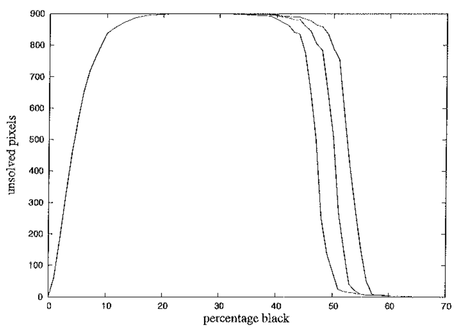
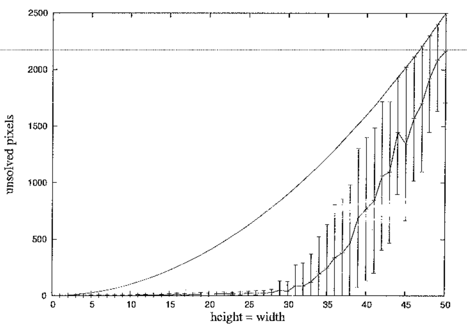


Figure 13: (a) Average number of unsolved pixels for randomly generated 30 × 30 puzzles, for a varying percentage of black pixels, when using *Solver1*; error bars indicate the standard deviation (100 runs for every percentage). (b) As (a), for *FullSettle* (top graph), *Solver0* (middle graph) and *Solver1* (bottom graph), without standard deviation.



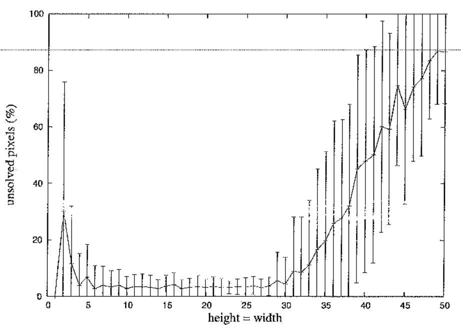
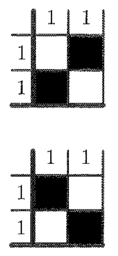
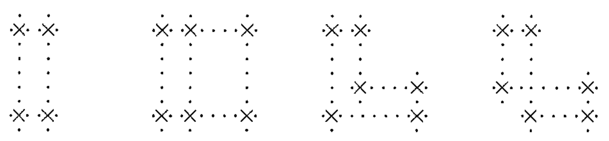


Figure 14: (a) Average number of unsolved pixels for randomly generated puzzles of different size, with a fixed percentage of 50 % black pixels, again using *Solver1*; error bars indicate the standard deviation. The smooth curve is the total number of pixels. (b) As (a), but now showing these values as percentage of the total number of pixels.



(a) Elementary switching component with its two solutions



(b) More complex types of switching components for Nonograms

Figure 15: Switching components in Nonograms, containing up to six unknowns.

on these elementary switching components [11], the non-uniqueness problem for Nonograms appears to be much more complex. The problem of deciding whether *another* solution of a Nonogram exists, given a particular solution, is NP-hard [17].

We now focus on cases, where only a small subset of the pixel values is not uniquely determined by the Nonogram description. Suppose that we have a given Nonogram, and that it is possible (using the approach of this paper, for example) to determine the value of all pixels, except for a small set of $u > 0$ unknown pixels or *unknowns*. We first note that each line with unknown pixels should contain at least 2 unknowns. This implies that $u \geq 4$ and $u \neq 5$. If $u = 4$, the unknowns form a rectangle (Figure 15b, left), similar to an elementary switching component. However, in Nonograms the existence of such switching components is not only determined by the value of the four corner pixels, but also by the values of the pixels along the four sides of the rectangle and by the pixels adjacent to the four corners. On a single line with 2 unknowns, depicted from left to right, we note that left of the leftmost unknown we must have a 0 pixel (or the image boundary), and a similar observation holds for the rightmost unknown. And we have precisely 2 solutions here. Between the two unknowns we must have only 1s, or a series of solitary 1s with variable length blocks of 0s in between: in regular expression notation $(0^+1)^*0^+$.

If we consider $u = 6$, we either have two lines with 3 unknowns each (and in the other direction three lines with 2 unknowns each), or three lines with 2 unknowns each in both directions (Figure 15b, right). In the former case we have 3 solutions, where in between two unknowns we can only have the $(0^+1)^*0^+$ situation, in the latter case there are precisely 2 solutions — as in the $u = 4$ case. In all these situations, unknowns cannot touch.

7 Conclusions and further research

Nonograms are interesting study objects, due to their links with both combinatorial optimization and logic reasoning, as well as their rich variety of combinatorial properties. In this paper, we focused on the set of *simple* Nonograms, which can be solved by a series of reasoning steps involving only a single column or row at a time. We proposed a *difficulty measure* for this class, which corresponds roughly with the solution strategy followed by human puzzlers and has favourable computational properties.

After studying the basic notions, we described a family of Nonograms that have asymptotically maximal difficulty, up to a constant factor. An interesting question remains if the difficulty can still be increased to cn^2, where $c \in (\frac{1}{2}, 1]$. Next, we briefly described an algorithm for *generating* Nonograms of varying difficulty. The basic steps of this algorithm allow for a broad spectrum of variants, each yielding different types of Nonograms. Finally we mentioned some issues related to non-simple Nonograms. We intend to explore these in future work.

References

[1] K.J. Batenburg, S. Henstra, W.A. Kosters, and W.J. Palenstijn. Constructing simple Nonograms of varying difficulty. *Pure Mathematics and Applications (Pu.M.A.)*, 20:1–15, 2009.

[2] K.J. Batenburg and W.A. Kosters. A discrete tomography approach to Japanese puzzles. In *Proceedings of the 16th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC)*, pages 243–250, 2004.

[3] K.J. Batenburg and W.A. Kosters. Solving Nonograms by combining relaxations. *Pattern Recognition*, 42:1672–1683, 2009.

[4] T.M. Buzug. *Computed Tomography: From Photon Statistics to Modern Cone-Beam CT.* Springer, 2008.

[5] G.T. Herman. *Fundamentals of Computerized Tomography: Image Reconstruction from Projections.* Springer, second edition, 2010.

[6] G.T. Herman and A. Kuba. *Discrete Tomography: Foundations, Algorithms and Applications.* Birkhäuser, 1999.

[7] G.T. Herman and A. Kuba. *Advances in Discrete Tomography and its Applications.* Birkhäuser, 2007.

[8] W.A. Kosters. Website Nonogram [accessed 9.2.2012], `www.liacs.nl/~kosters/nono/`, 2012.

[9] E.G. Ortiz-García, S. Salcedo-Sanz, J.M. Leiva-Murillo, Á.M. Pérez-Bellido, and J.A. Portilla-Figueras. Automated generation and visualization of picture-logic puzzles. *Computers and Graphics*, 31:750–760, 2007.

[10] E.G. Ortiz-García, S. Salcedo-Sanz, Á.M. Pérez-Bellido, L. Carro-Calvo, A. Portilla-Figueras, and X. Yao. Improving the performance of evolutionary algorithms in grid-based puzzles resolution. *Evolutionary Intelligence*, pages 169–181, 2009.

[11] H. J. Ryser. Combinatorial properties of matrices of zeros and ones. *Canadian Journal of Mathematics*, 9:371–377, 1957.

[12] S. Salcedo-Sanz, E.G. Ortiz-García, Á.M. Pérez-Bellido, J.A. Portilla-Figueras, and X. Yao. Solving Japanese puzzles with heuristics. In *Proceedings IEEE Symposium on Computational Intelligence and Games (CIG)*, pages 224–231, 2007.

[13] S. Salcedo-Sanz, J.A. Portilla-Figueras, E.G. Ortiz-García, Á.M. Pérez-Bellido, and X. Yao. Teaching advanced features of evolutionary algorithms using Japanese puzzles. *IEEE Transactions on Education*, 50:151–156, 2007.

[14] S. Simpson. Website Nonogram solver [accessed 9.2.2012], `www.comp.lancs.ac.uk/~ss/nonogram/links.html`, 2011.

[15] J.-T. Tsai. Solving Japanese nonograms by Taguchi-based genetic algorithm. *Applied Intelligence*, 2012 [to appear].

[16] J.-T. Tsai, P.-Y. Chou, and J.-C. Fang. Learning intelligent genetic algorithms using Japanese nonograms. *IEEE Transactions on Education*, 2012 [to appear].

[17] N. Ueda and T. Nagao. NP-completeness results for Nonogram via parsimonious reductions, preprint, 1996.

[18] J.K. Vis, W.A. Kosters, and K.J. Batenburg. Discrete tomography: A neural network approach. In *Proceedings of the 23rd Benelux Conference on Artificial Intelligence (BNAIC)*, pages 328–335, 2011.

[19] Website Griddlers [accessed 24.2.2012], `www.griddlers.net`, 2012.

[20] G.J. Woeginger. The reconstruction of polyominoes from their orthogonal projections. *Information Processing Letters*, 77:225–229, 2001.

INSTITUUT VOOR PROGRAMMATUURKUNDE EN ALGORITMIEK

ipa

www.win.tue.nl/ipa/

Institute for Programming research and Algorithmics

The research school IPA (Institute for Programming Research and Algorithmics) educates researchers in the field of programming research and algorithmics. This field encompasses the study and development of formalisms, methods and techniques to design, analyse, and construct software systems and components. IPA has three main research areas: Algorithmics & Complexity, Formal Methods, and Software Technology & Engineering. Researchers from nine universities (University of Nijmegen, Leiden University, Technische Universiteit Eindhoven, University of Twente, Utrecht University, University of Groningen, Vrije Universiteit Amsterdam, University of Amsterdam, and Delft University), the CWI and Philips Research (Eindhoven) participate in IPA.

In 1997, IPA was formally accredited by the Royal Dutch Academy of Sciences (KNAW). This accreditation was extended in 2002 and 2007. In setting its agenda for 2007 - 2012, IPA chose five focus areas, where we expect important developments in the near future and want to stimulate collaboration. In the focus area:

Beyond Turing we want to explore novel paradigms of computation that incorporate concepts that are no longer adequately modelled by the classical Turing machine such as nonuniformity of memory, adaptivity and mobility.

Algorithms & models for life sciences we wish to apply algorithmic theory and formal models to contribute to the understanding of biological processes, entities and phenomena.

Hybrid systems we want to continue to contribute to the confluence of systems and control theory and computer science in integrated methods for modelling, simulation, analysis, and design of such systems.

Model-driven software engineering we want to study various fundamental aspects of the model-driven approach to software engineering.

Software analysis we want to make progress in the extraction of facts from source code and their analysis, to obtain instruments for measuring the various quality attributes of software.

For descriptions of these areas see `www.win.tue.nl/ipa/about.html`.

Activities in 2011

IPA has two multi-day events per year which focus on current topics, the Lentedagen and the Herfstdagen. In 2011, the Lentedagen were on Bioinformatics, and the Herfstdagen were combined with the ASCI Conference, a SIKS day, NWO's SIREN event and the STW events (Progress, Sentinels, SAFE and ProRISC).

IPA organises courses on each of its major research fields, Algorithms and Complexity, Formal Methods and Software Technology & Engineering. These courses intend to give an overview of the research of IPA in these fields, and are organised at regular intervals on a cyclic schedule. In

2011, the Algorithms and Complexity course was organised. In addition, ASCI, IPA and SIKS collaborated successfully on a course on *Security*.

IPA Lentedagen on Bioinformatics *April 11 - 15, Hotel Prinsen, Vlijmen*
The IPA Spring Days are an annual multi-day event, dedicated to a specific theme of current interest to the research community of IPA. This year's Spring Days are dedicated to Bioinformatics, fitting in the focus area of Algorithms and Models for the Life Sciences of the research school for the period 2007-2012.

Traditionally Bioinformatics is concerned with the application of Computer Science and Statistics in the field of Molecular Biology. Emerging in the late 80s Bioinformatics initially focused on genomics and genetics, particularly in areas of genomics involving large-scale DNA sequencing. At present, Bioinformatics is a much wider research area addressing biologically motivated research questions including algorithms for structures as network graphs and evolutionary trees, computational modelling of molecule interaction, visualisation and analysis of biological and medical data.

In a number of talks, examples of algorithmics and formal methods that are tailored to problems stemming from the Life Sciences are presented. Also, approaches to cope with the overwhelming amount of data generated by today's Biology and Medicine, and the need for The programme aims to give an impression of the wealth and diversity of Bioinformatics research at the crossroads of Computer Science, Biology and Medicine.

The programme was composed by Wan Fokkink (VUA), Elena Marchiori (RU) and Erik de Vink (TU/e). More information about the programme is available through the archive on the IPA-website: `www.win.tue.nl/ipa/archive/springdays2011/`.

IPA Herfstdagen *November 14 - 15, Conference Centre De Koningshof, Veldhoven*
The successful SIREN // NL event of 2010 was further strengthened with STW's traditional conferences. The resulting conference was called ICT.OPEN.

More information about the programme is available through the archive on the IPA-website: `www.win.tue.nl/ipa/archive/falldays2011/`.

IPA Ph.D. Defenses in 2011

R. Bakhshi (VUA, 13 January)
Gossiping Models: Formal Analysis of Epidemic Protocols
Promotores: Prof.dr. W.J. Fokkink and Prof.dr. M.R. van Steen
IPA Dissertation Series 2011-01

B.J. Arnoldus (TU/e, 1 February)
An Illumination of the Template Enigma: Software Code Generation with Templates
Promotor: Prof.dr. M.G.J. van den Brand. Co-promotores: Dr.ir. J.J. Brunekreef and Dr. A. Serebrenik
IPA Dissertation Series 2011-02

E. Zambon (UT, 20 January)
Towards Optimal IT Availability Planning: Methods and Tools
Promotores: Prof.dr. S. Etalle and Prof.dr. R.J. Wieringa
IPA Dissertation Series 2011-03

L. Astefanoaei (UL, 19 January)
An Executable Theory of Multi-Agent Systems Refinement
Promotores: Prof.dr. F.S. de Boer and Prof.dr. J.-J.Ch. Meyer
IPA Dissertation Series 2011-04

J. Proença (UL, 11 May)
Synchronous coordination of distributed components
Promotor: Prof.dr. F. Arbab. Co-promotores: Dr. D. Clarke and Dr. E.P. de Vink
IPA Dissertation Series 2011-05

A. Moralı (UT, 21 April)
IT Architecture-Based Confidentiality Risk Assessment in Networks of Organizations
Promotores: Prof.dr. R.J. Wieringa and Prof.dr. S. Etalle
IPA Dissertation Series 2011-06

M. van der Bijl (UT, 12 May)
On changing models in Model-Based Testing
Promotores: Prof.dr. H. Brinksma and Prof.dr.ir. A. Rensink. Co-promotor: Dr.ir. G.J. Tretmans
IPA Dissertation Series 2011-07

C. Krause (UL, 21 June)
Reconfigurable Component Connectors
Promotor: Prof.dr. F. Arbab. Co-promotor: Dr. E.P. de Vink
IPA Dissertation Series 2011-08

M.E. Andrés (RU, 1 July)
Quantitative Analysis of Information Leakage in Probabilistic and Nondeterministic Systems
Promotor: Prof.dr. B.P.F. Jacobs. Co-promotores: Dr. P. van Rossum and Dr. C. Palamidessi
IPA Dissertation Series 2011-09

M. Atif (TU/e, 28 September)
Formal Modeling and Verification of Distributed Failure Detectors
Promotores: Prof.dr.ir. J.F. Groote and Prof.dr. M.G.J. van den Brand. Co-promotor: Dr. M.R. Mousavi
IPA Dissertation Series 2011-10

P.J.A. van Tilburg (TU/e, 27 October)
From Computability to Executability – A process-theoretic view on automata theory
Promotor: Prof.dr. J.C.M. Baeten. Co-promotor: Dr. S.P. Luttik
IPA Dissertation Series 2011-11

Z. Protic (TU/e, 3 October)
Configuration management for models: Generic methods for model comparison and model co-evolution
Promotor: Prof.dr. M.G.J. van den Brand. Co-promotor: Dr.ir. T. Verhoeff
IPA Dissertation Series 2011-12

S. Georgievska (TU/e, 3 October)
Probability and Hiding in Concurrent Processes
Promotores: Prof.dr. J.C.M. Baeten and Prof.dr. W.J. Fokkink. Co-promotor: Dr. S. Andova
IPA Dissertation Series 2011-13

S. Malakuti (UT, 15 September)
Event Composition Model: Achieving Naturalness in Runtime Enforcement
Promotor: Prof.dr.ir. M. Aksit. Co-promotor: Dr. C. Bockisch
IPA Dissertation Series 2011-14

M. Raffelsieper (TU/e, 2 November)
Cell Libraries and Verification
Promotores: Prof.dr. H. Zantema and Prof.dr.ir. J.F. Groote. Co-promotor: Dr. M.R. Mousavi
IPA Dissertation Series 2011-15

C.P. Tsirogiannis (TU/e, 14 November)
Analysis of Flow and Visibility on Triangulated Terrains
Promotor: Prof.dr. M.T. de Berg
IPA Dissertation Series 2011-16

Y.-J. Moon (UL, 25 October)
Stochastic Models for Quality of Service of Component Connectors
Promotor: Prof.dr. F. Arbab. Co-promotores: Dr. A. Silva and Dr. E.P. de Vink
IPA Dissertation Series 2011-17

R. Middelkoop (TU/e, 30 November)
Capturing and Exploiting Abstract Views of States in OO Verification
Prof.dr. M.G.J. van den Brand. Co-promotores: Dr. R. Kuiper and Dr. C. Huizing
IPA Dissertation Series 2011-18

A.N. Tamalet (RU, 28 November)
Towards Correct Programs in Practice
Promotor: Prof.dr. M.C.J.D. van Eekelen
IPA Dissertation Series 2011-20

H.J.S. Basten (UvA, 15 December)
Ambiguity Detection for Programming Language Grammars
Prof.dr. P. Klint. Co-promotor: Dr. J.J. Vinju
IPA Dissertation Series 2011-21

M. Izadi (UL, 6 December)
Model Checking of Component Connectors
Promotores: Prof.dr. F. Arbab and Prof.dr. A. Movaghar
IPA Dissertation Series 2011-22

L.C.L. Kats (TUD, 15 December)
Building Blocks for Language Workbenches
Promotores: Prof.dr. A. van Deursen. Co-promotor: Dr. E. Visser
IPA Dissertation Series 2011-23

S. Kemper (UL, 20 December)
Modelling and Analysis of Real-Time Coordination Patterns
Promotores: Prof.dr. F.S. de Boer and Prof.dr. F. Arbab
IPA Dissertation Series 2011-24

J. Wang (UL, 20 December)
Spiking Neural P Systems
Promotor: Prof.dr. J.N. Kok. Co-promotor: Dr. H.J. Hoogeboom
IPA Dissertation Series 2011-25

Activities in 2012

IPA is planning several activities for 2012, including the Lentedagen (April 16-20) which will be dedicated to *Model-Driven Software Engineering*, the Course on Software Technology & Engineering (TU/e, Eindhoven), and the Herfstdagen, which will again be part of ICT.OPEN (November). More information on these events will appear on the IPA-website as dates and locations for these events are confirmed.

Addresses

Visiting address
Technische Universiteit Eindhoven
Main Building HG 7.22
Den Dolech 2
5612 AZ Eindhoven
The Netherlands

Postal address
IPA, Fac. of Math. and Comp. Sci.
Technische Universiteit Eindhoven
P.O. Box 513
5600 MB Eindhoven
The Netherlands

tel. (+31)-40-2474124 (IPA Secretariat)
fax (+31)-40-2475361
e-mail `ipa@tue.nl`
url `www.win.tue.nl/ipa/`

School for Information and Knowledge Systems (SIKS) in 2011

Richard Starmans (UU)

Introduction

SIKS is the Dutch Research School for Information and Knowledge Systems. It was founded in 1996 by researchers in the field of Artificial Intelligence, Databases & Information Systems and Software Engineering. Its main concern is research and education in the field of information and computing sciences, more particular in the area of information and knowledge systems.The School currently concentrates on seven focus areas in the IKS field: Agent Technology, Computational Intelligence, Knowledge Representation and Reasoning, Web-based Information Systems, Enterprise Information Systems, Human Computer Interaction, and Data Management, Storage and Retrieval.

SIKS is an interuniversity research school that comprises 14 research groups from 11 universities and CWI. Currently, nearly 500 researchers are active, including 200 Ph.D.-students. The Vrije Universiteit in Amsterdam is SIKS' administrative university, the office of SIKS is located at Utrecht University. As off September 2011 Prof.dr. P. De Bra (TUE) was appointed scientific director. He took over the chair from Prof. dr. R. Wieringa (UT) who held the position for nearly six years. SIKS received its first accreditation by KNAW in 1998 and its first re-accreditation in 2003. In June 2009 SIKS was re-accredited for another period of 6 years.

Activities

We hereby list the main activities (co-)organized or (co-)financed by SIKS. We distinguish tutorials, advanced courses and other activities (including master classes, workshops, one-day seminars, conferences, summer schools, doctoral consortia and research colloquia)

SIKS-tutorials:

“Learning and Reasoning”, June 07-08, 2011, Landgoed Huize Bergen, Vught
Course directors: Dr. A. Ten Teije (VU), Dr. P. Groot (RUN)

“Information Retrieval”, June 09-10, 2011, Landgoed Huize Bergen, Vught
Course director: Prof. dr. ir. Th. Van der Weide (RUN)

“Research methods and methodology for IKS", November 23-25, 2011, Conference Center Zonheuvel, Doorn
Course directors: Dr. H. Weigand (UvT), Prof.dr. R.J. Wieringa (UT), Prof.dr. H. Akkermans (VUA), Prof.dr. J.-J.Ch. Meyer (UU), Dr. R.J.C.M. Starmans (UU).

Advanced courses:

“Agent Based Simulation”, February 21-22, 2011, Woudschoten, Zeist
Course director: Dr. V. Dignum (TUD), Dr. F. Dignum (UU)

“Human Technology Interaction”, April 11-15, 2011, Hotel New York, Rotterdam
Course director: Prof. dr. C.M. Jonker (TUD)

"Summer course on Datamining", August 29-Septmeber 01, 2011, Maastricht
Course director: Dr. E. Smirnov (UM)

"The Semantic Web", September 26-27, 2011, Woudschoten, Zeist
Course directors: Dr. L. Hollink (TUD), Dr. R. Hoekstra (VU), Dr. S. Wang (VU)

"Computational Intelligence", October 13-14, 2011, Mitland Hotel, Utrecht.
Course directors: Dr. A. Feelders (UU), Prof. dr. T. Heskes (RUN),Prof.dr. A. Siebes (UU),

"Security", October 10, 17, 24, 2011, Amsterdam, Nijmegen, Eindhoven (Cooperation with ASCI and IPA)
Course directors: Dr. T. Willemse (TUE), Dr. W. Pieters (UT), Dr. G. van 't Noordende (UVA)

"Big Data and Cloud Computing", November 30 - December 01, 2011, Conference Hotel Drienerburght, Enschede
Course directors: Dr. ir. D. Hiemstra (UT), Prof. dr. A. de Vries (CWI/TUR), E. Lammerts (SARA)

Other activities:

- Conference: DIR 2011, February 04, 2011, Amsterdam
- Conference: Benelearn 2011, May 20, 2011, The Hague
- Conference: "6th SIKS-Conference on Enterprise Information Systems" (EIS), October 31, 2011, Delft
- Conference: BNAIC 2011, November 03-04, 2011, Gent
- Conference: Dutch-Belgian Database Day 2011 (DBDBD), December 02, 2011, Enschede
- Conference: ICT Open, November 15-16, 2011, Veldhoven
- Conference: CSERC 2011, April 7 and 8, 2011, Heerlen
- Conference: 4th International Conference on Educational Data Ming, July 06-08, 2011, Eindhoven
- Masterclass: "Human Computer Interaction", November 24, 2011, Amsterdam
- NVTI Theory Day 2011, March 04, 2011, Utrecht
- SIKS-PhD Annual Cultural Event, May 25, 2011, Amsterdam
- SIKS-day 2011, October 02, 2011, Utrecht
- Seminar: "Something on Search and Searchers", 15 April 2011, Amsterdam
- Seminar: UU-SIKS Seminars (2 times in 2011)
- Colloquia: SIKS-Agent Colloquia (8 times in 2011), Utrecht/Delft/Amsterdam
- Colloquia: SIKS-MICC Colloquia (6 times in 2011), Maastricht
- Colloquia: SIKS-TiCC Colloquia (6 times in 2011), Tilburg
- Autumnschool on Human Robot Cooperation, 07-11 November 2011, Soesterberg
- Summerschool "Information Foraging 2011" August 22-September 02, 2011, Nijmegen
- Summerschool EASSS 2011, June 11-15, 2011, Girona (Spain)
- Symposium: Software Operation Knowledge, November 17, 2011, Utrecht
- Symposium: "Strategic Decision Making in Complex Games", 15 Juni, 2012, Maaastricht
- Symposium: SIKS-TiCC Symposium on Language modeling, December 2011, Tilburg
- Symposium: SIKS-symposium on Model Engineering, November 11, 2011, Eindhoven
- Symposium: SIKS-Symposium on Brain Computer Interfaces, October 21, 2011, Enschede
- Symposium: SIKS-ICS Symposium on Norms, Logic and Dependence, October 18, 2011, Utrecht
- Symposium: SIKS-symposium on Data Mining, October 11, 2011, Eindhoven
- Symposium: SIKS-Symposium Beyond the Semantic Web, March 04,2011, Amsterdam
- Workshop: European Workshop on Multi-agent Systems (EUMAS 2011),November 14-15, 2011, Maastricht
- Workshop on Multi Agent Organisations, December 19-23, 2011, Leiden
- Workshop on Social Signaling Processing (TiSSP),November 28, 2011, Tilburg
- Workshop on Value Modeling and Business Ontology, February, 07-08, 2011, Gent

Ph.D.-defenses in 2011

In 2011 49 researchers successfully defended their Ph.D.-thesis and published their work in the SIKS-dissertation Series.

2011-01
Botond Cseke (RUN)
Variational Algorithms for Bayesian Inference in Latent Gaussian Models
Promotor: Prof.dr. T.M. Heskes (RUN)
Promotie: 24 Januari 2011

2011-02
Nick Tinnemeier(UU)
Organizing Agent Organizations. Syntax and Operational Semantics of an Organization-Oriented Programming Language
Promotor: Prof. dr. J.-J. Ch. Meyer (UU)
Copromotor: Dr. M.M. Dastani (UU), Dr. F. M. de Boer (CWI)
Promotie: 7 February 2011

2011-03
Jan Martijn van der Werf (TUE)
Compositional Design and Verification of Component-Based Information Systems
Promotor: : Prof.dr. K.M. van Hee (TU/e), Prof. Dr. W. Reisig (Humboldt-Universität zu Berlin)
Copromotores: Copromotores : Prof.dr. W. Scheper (UU), dr. N. Sidorova (TU/e)
Promotie: 15 February 2011

2011-04
Hado Philip van Hasselt (UU)
Insights in Reinforcement Learning; Formal analysis and empirical evaluation of temporal-difference learning algorithms
Promotores: Prof.dr. J.-J.Ch. Meyer (UU), Prof.dr. L.R.B. Schomaker (RUG)
Copromotor: Dr. M.A. Wiering (RUG)
Promotie: 17 January 2011

2011-05
Bas van de Raadt (VU)
Enterprise Architecture Coming of Age - Increasing the Performance of an Emerging Discipline
Promotor: : Prof. dr. J.C. van Vliet (VU)
Promotie: 25 February 2011

2011-06
Yiwen Wang(TUE)
Semantically-Enhanced Recommendations in Cultural Heritage
Promotores: Prof.dr. P.M.E. De Bra (TUE), Prof.dr. A.Th. Schreiber (VU)
Copromotor: dr. L.M. Aroyo (VU)
Promotie: 08 Februay 2011

2011-07
Yujia Cao (UT)
Multimodal Information Presentation for High Load Human Computer Interaction
Promotor: : Prof. dr. ir. A. Nijholt (UT)
Promotie: 03 February 2011

2011-08
Nieske Vergunst (UU)
BDI-based Generation of Robust Task-Oriented Dialogues
Promotores: Prof.dr. J.-J. Ch. Meyer (UU)
Copromotor:Dr. ir. R.-J. Beun (UU), Dr. R van Eijk (UU)
Promotie: 09 March 2011

2011-09
Tim de Jong (OU)
Contextualised Mobile Media for Learning
Promotor: Prof.dr. E.J.R. Koper (OU), Prof. Dr. M. Specht (OU)
Promotie: 10 June 2011

2011-10
Bart Bogaert (UvT)
Cloud Content Contention
Promotores: Prof.dr. H.J. van den Herik (UvT), Prof.dr. E.O. Postma (UvT)
Promotie: 30 March 2011

2011-11
Dhaval Vyas (UT)
Designing for Awareness: An Experience-focused HCI Perspective
Promotores: Prof. dr. ir. A. Nijholt (UT), Prof. dr. G. van der Veer (OU)
Copromotor:Dr. D. Heylen (UT)
Promotie: 18 February 2011

2011-12
Carmen Bratosin (TUE)
Grid Architecture for Distributed Process Mining
Promotores: Prof.dr. W.M.P. van der Aalst (TUE)
Copromotor:Dr. N. Sidirova (TUE)
Promotie: 29 March 2011

2011-13
Xiaoyu Mao (UvT)
Airport under Control; Multiagent Scheduling for Airport Ground Handling
Prof. dr. H.J. van den Herik (UvT), Prof. dr. E.O. Postma (UvT)
Copromotor:Dr. ir. N. Roos (UM), Dr. A.H. Salden (Almende B.V.)
Promotie: 25 May 2011

2011-14
Milan Lovric (EUR)
Behavioral Finance and Agent-Based Artificial Markets
Promotores: Prof. Dr. J. Spronk (EUR), Prof. Dr. Ir. U. Kaymak (EUR, TU/e)
Promotie: 25 March 2011

2011-15
Marijn Koolen (UVA)
The Meaning of Structure: the Value of Link Evidence for Information Retrieval
Promotor: Prof. dr. J.S. MacKenzie-Owen (UvA)
Copromotor: Dr. ir. J. Kamps (UvA)
Promotie: 15 April 2011

2011-16
Maarten Schadd (UM)
Selective Search in Games of Different Complexity
Promotores: Prof.dr. G. Weiss (UM)
Copromotor: Dr. M.H.M. Winands (UM), Dr. ir. J.W.H.M. Uiterwijk (UM)
Promotie: 25 May 2011

2011-17
Jiyin He (UVA)
Exploring Topic Structure: Coherence, Diversity and Relatedness
Promotor: Prof. dr. M. de Rijke (UvA)
Promotie: 18 May 2011

2011-18
Mark Ponsen (UM)
Strategic Decision-Making in complex games
Promotores: Prof.dr. G. Weiss (UM)
Copromotor: Dr. K. Tuyls (UM), Dr. J. Ramon (Katholieke Universiteit Leuven)
Promotie: 15 June 2011

2011-19
Ellen Rusman (OU)
The Mind ' s Eye on Personal Profiles
Promotor: Prof. dr. E.J.R. Koper (OU) en Prof. dr. P.B. Sloep (OU)
Copromotor: Dr. J.M. van Bruggen (OU)
Promotie: 17 June 2011

2011-20
Qing Gu (VU)
Guiding service-oriented software engineering - A view-based approach
Promotor: Prof. dr. J.C. van Vliet (VU)
Copromotor: Dr.P.Lago (VU)
Promotie: 06 October 2011

2011-21
Linda Terlouw (TUD)
Modularization and Specification of Service-Oriented Systems
Promotor: Prof. dr. J.L.G. Dietz (TUD)
Promotie: 05 July 2011

2011-22
Junte Zhang (UVA)
System Evaluation of Archival Description and Access
Promotor: Prof. dr. T. Thomassen (UVA)
Copromotor: Dr.J. Kamps (UVA)
Promotie: 30 September 2011

2011-23
Wouter Weerkamp (UVA)
Finding People and their Utterances in Social Media
Promotor: Prof. dr. M. de Rijke (UVA)
Promotie: 18 October 2011

2011-24
Herwin van Welbergen (UT)
Behavior Generation for Interpersonal Coordination with Virtual Humans On Specifying, Scheduling and Realizing Multimodal Virtual Human Behavior
Promotor: Prof. dr. ir . A. Nijholt (UT)
Promotie: 09 September 2011

2011-25
Syed Waqar ul Qounain Jaffry (VU)
Analysis and Validation of Models for Trust Dynamics
Promotor: Prof. dr. J. Treur (VU)
Copromotor: Dr. M. Hoogendoorn (VU)
Promotie: 09 September 2011

2011-26
Matthijs Aart Pontier (VU)
Virtual Agents for Human Communication - Emotion Regulation and Involvement-Distance Trade-Offs in Embodied Conversational Agents and Robots
Promotor: Prof. dr. J. Treur (VU), Prof. dr. E. Konijn (VU)
Copromotores: Dr. T. Bosse (VU), Dr. J. Hoorn (VU)
Promotie: 19 September 2011

2011-27
Aniel Bhulai (VU)
Dynamic website optimization through autonomous management of design patterns
Promotor: Prof. dr. G. van der Veer (VU)
Copromotor: Dr. S. Bhulai (VU)
Promotie: 25 November 2011

2011-28
Rianne Kaptein (UVA)
Effective Focused Retrieval by Exploiting Query Context and Document Structure
Promotor: Prof. dr. J.S. Mackenzie Owen (UvA)
Copromotor: Dr.J. Kamps (UVA)
Promotie: 07 October 2011

2011-29
Faisal Kamiran (TUE)
Discrimination-aware Classification
Promotor: Prof. dr. Paul De Bra (TUE)
Copromotor: Dr. T. Calders (TUE)
Promotie: 11 October 2011

2011-30
Egon van den Broek (UT)
Affective Signal Processing (ASP): Unraveling the mystery of emotions
Promotores: Prof. dr. ir. A. Nijholt (UT), Prof. dr. T. Dijkstra (RUN)
Copromotor: Dr. J.H.D.M. Westerink (Philips Research)
Promotie: 16 September 2011

2011-31
Ludo Waltman (EUR)
Computational and Game-Theoretic Approaches for Modeling Bounded Rationality
Promotor: Prof. dr. ir. R. Dekker (EUR), Prof.dr.ir. U.Kaymak (EUR, TUE)
Promotie: 13 October 2011

2011-48
Mark Ter Maat (UT)
Response Selection and Turn-taking for a Sensitive Artificial Listening Agent
Promotor: Prof. dr. ir. A. Nijholt (UT),Prof. Dr. D.K.J. Heylen (UT)
Promotie: 30 November 2011

2011-49
Andreea Niculescu (UT)
Conversational interfaces for task-oriented spoken dialogues: design aspects influencing interaction quality
Promotores: Prof. dr. ir. A. Nijholt (UT)
Copromotor: Dr.ir. B. van Dijk (UT)
Promotie:22 November 2011

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; één 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.