CHALMERS | GÖTEBORG UNIVERSITY

A Random Walk in Statistical Physics

Pontus Svenson

Department of Theoretical Physics
Chalmers University of Technology and Göteborg University
Göteborg, Sweden 2001

Thesis for the degree of Doctor of Philosophy*

A Random Walk in Statistical Physics

Pontus Svenson†

Department of Theoretical Physics
Chalmers University of Technology and Göteborg University
Göteborg, Sweden 2001

* The thesis is available at http://fy.chalmers.se/~tfkps/

†email: tfkps@fy.chalmers.se

A Random Walk in Statistical Physics
Pontus Svenson
ISBN 91-7291-095-X

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 1778
ISSN 0346-718X

Department of Theoretical Physics
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Chalmersbibliotekets reproservice
Göteborg, Sweden 2001

To my mother and the memory of my father

A Random Walk in Statistical Physics

Pontus Svenson
Department of Theoretical Physics
Chalmers University of Technology and Göteborg University
SE-412 96 Göteborg, Sweden

Abstract

This thesis deals with some aspects of the physics of disordered systems. It consists of four papers and an introductory part.

An introduction, suitable for physicists, to theoretical computer science and computational complexity is contained in chapter 2. The definition of the Turing model of computation and of some important complexity classes are given, the Church-Turing hypothesis described, and the proofs of some important theorems reviewed.

Additional work by physicists on optimisation problems is described in chapter 3, while chapter 4 is an introduction to computer simulation of physical models. This chapter also contains some results for the distribution of sub-structures of lattice polymers.

Chapter 5 introduces the concept of small world graphs and reviews previous work on physical models placed on such graphs.

Paper I studies the relaxation of some optimisation problems that, unless a very plausible conjecture in computer science is false, have worst case instances that require exponential time to solve. These problems can also be interpreted as spin glass models, and have previously been found to exhibit threshold phenomena akin to those of physical models undergoing phase transitions. The graph colouring problem is revisited in paper IV, wherein the phase transition is studied using small world graphs.

In paper II the ferromagnetic Ising model on random graphs is found to display a freezing phenomenon for a range of connectivities.

Damage spreading is an important tool for studying the stability of models. Paper III finds a very good data collapse for the damage plotted as a function of temperature for small world graphs with different rewiring probabilities. Small worlds have so far almost only been studied using a $1D$ chain as starting lattice. The work presented in this thesis contains the first application of $2D$ and $3D$ small worlds to physical systems.

Keywords: Spin glasses, relaxation, computational complexity, NP-completeness, boolean satisfiability, graph colouring, disordered energy landscapes, damage spreading, Ising model, small world graphs, phase transitions, polymers.

This thesis consists of an introductory text and the following four appended research papers, henceforth referred to as Paper I-IV:

I. P. Svenson and M. G. Nordahl,
Relaxation in graph coloring and satisfiability problems,
Phys. Rev. **E59**(4) (1999) 3983-3999 (cond-mat/9810144).

II. P. Svenson,
Freezing in random graph ferromagnets,
Phys. Rev. **E64** (2001) 036122 (cond-mat/0105161).

III. P. Svenson and D. A. Johnston,
Damage spreading in small world Ising models,
Submitted to Phys. Rev. **E** (cond-mat/0107555).

IV. P. Svenson, *From Néel to NPC: Colouring small worlds*,
(cs.CC/0107015).

Acknowledgements

I thank my supervisor Stellan Östlund for support, interesting discussions, and for teaching me to simplify the problems I study until they become tractable. I also thank Mats Nordahl for discussions during my first years at the department and for introducing me to complex systems and computational complexity. Henrik Johannesson inspired me to become a PhD-student while I did my MS thesis, and our group probably would not function without Yvonne Steen, so they too must be thanked here.

I'm grateful to Des Johnston and the Maths Department at Heriot-Watt for making my stays in Edinburgh enjoyable. The staff at the comic-book shop should also be thanked, as well as the European Commission who payed for it all.

My past and present colleagues at the department have provided much entertainment and many interesting discussion (sometimes even about physics) over the years. For this I thank you all. I am especially grateful to those of you who have persisted in having the wrong opinions in our debates.

A conservative estimate[‡] places the total amount of CPU time that I have used during my years at the department at 16 years. In addition to my own mon-el and kal-el, the following computers were abused:
@fy.chalmers.se: curie, klein, gibbs, bardeen, yukawa, hamilton, tycho, albert, euler, meitner, thomson, hall, xie, vanhove, ising, unicorn, popcorn, pom, fysparc1, fysparc10, fysparc11, fysparc12, fysparc13, fysparc14
@ma.hw.ac.uk: reunion, handa, coll, wiay, mull, inchcolm, gruinard, lingay, rousay, oxna, rockall, ascrib, burray, davaar, inchcape, lewis, oldany, oronsay, pladda, riska, scarp, skye, staffa, stroma, turing, vallay, sanda, muck, jura, hoy, bellrock, bassrock
@epcc.ed.ac.uk: lomond, tufa, amber

My thanks to the owners of these machines and an apology for abusing them. Your joy at being able to do `top` without seeing `tfkps` comes at the price of losing your computer support.

Lastly, I am grateful to Neil Gaiman, Grant Morrison, Philip K. Dick, and Tori Amos for providing entertaining diversions and to the Coca-Cola Company for being there.

[‡] i.e., running `zgrep tid `find . -name ´*.m*´`` and adding the results

”I learn so as to be contented”

Contents

1

Introduction

"And so it begins."
— Kosh, *Babylon 5*

1.1 On the physics of this thesis

This thesis deals with several seemingly disparate subjects. My first years as a graduate student were spent studying optimisation problems and computational complexity. Some of the computer science background needed to understand paper I is given in chapter 2, while chapter 3 is an admittedly incomplete and subjective view of work on constraint satisfaction problems by other physicists. All of the work presented in this thesis is numerical. Computer simulation is a vital and important part of modern physics, equal in importance but different from both theoretical and experimental physics. Chapter 4 contains an introduction to computer simulation of physical models and also explains some of the numerical techniques used in the papers.

My next big interest was polymers. Unfortunately, this thesis does not contain an explanation of protein designability [1], but an amusing power-law for the number of times that small patterns occur in polymer simulations is described in section 4.5.

Frustration is an important concept for both optimisation problems and polymers. Paper II shows that it is possible to get a kind of dynamic frustration even in a simple ferromagnetic spin model. Chapter 5 introduces random and small world graphs. The latter were used in paper III, the first paper that studies spin models on small world networks based on two and three-dimensional lattices. Finally, in paper IV we return to the beginning by putting one of the optimisation problems studied in paper I on small world graphs.

The work presented in the papers is mostly self-contained. The aim of this introduction is to give a readable account of some of the background necessary to understand the papers. The list of references is not complete (this is particularly true for the chapter on small worlds), but should instead be taken as a starting point for further study.

Most of the contents of this thesis is in the borderline of what is traditionally considered physics. Lately, physicists have more and more started to study so called complex systems, such as the interaction between species in an ecology or between agents in financial markets. There are several possible definitions of the term "complex system". One viewpoint is to consider a system complex if it is difficult to solve. The $1D$ Ising model is unarguably less complex than the $2D$, which is in turn less complex than the (unsolved) $3D$ version. A more quantitative definition is to determine the complexity of the patterns that the model generates and let this be the complexity of the model. For example, the Ising model below the ordering temperature T_c is spatially homogeneous, without any complex patterns. Well above T_c there is no order and the system is completely random. But close to or at T_c, something different happens. There is no ordering, yet the patterns are not random. Coarse-graining (magnifying) the system results in a new system that has the same appearance as the old — the system possesses self-similarity. Patterns showing self-similarity are more complex than the others, and it takes longer time to simulate them on a computer.

It is important not to confuse complexity with randomness. Both completely ordered and completely random systems should have low complexities. Crutchfield and Feldman[2] have analysed the complexity of one dimensional spin systems using a complexity measure related to the amount of memory needed to predict the state of the system given knowledge of part of it; in [3] they review several different ways of measuring the complexity of patterns appearing in physical systems.

The computer science view of complexity is that a model is complex if it is difficult to simulate it on a computer or other machine. It is this aspect of complexity that motivates the emphasis on computational complexity in this thesis, but it is important to bear in mind that this is not the complete definition of complexity or of complex systems. More discussion of these definitions can be found in [4, 5, 6]. In particular, the contribution by Anderson in [4] presents some arguments for why it is wrong to rely too much on the computer science definition of complexity in studying real world complex systems, and instead advocates for spin glasses as the proper paradigm for complex systems science. Spin glass models and concepts have been used to study many different models in economics, biology, and social science. Their ubiquity stems from the belief that they capture many universal features of realistic models in most areas of complex systems, yet still are simple enough to study successfully (see e.g., [6, 7, 8]).

A vital part of complex systems is the modelling of various kinds of agents on social networks. These models often have very little resemblance

to traditional physics, but are nevertheless a legitimate part of physics, just as the study of electrons and quarks is.

In my opinion physics is more about the methods used to study a problem and the attitude (i.e., the scepticism and honesty) towards it than about the particular problem that is studied. The best definition of physics is either that physics is what physicist do or, paraphrasing Judge Stewart, "I don't know what physics is, but I recognise it when I see it".

Physics is also about having fun. The discovery of metastable states in ferromagnetic Ising models on random graphs using local dynamics might some day be useful for something, but I don't really care about that. Thinking about those poor spins, forced to live on strange lattices where they are deeply frustrated, has given me a considerable amount of pleasure over the years, and that's the true reason for doing it.

The rest of this chapter gives an introduction, suitable for non-physicists, to spin models and some associated concepts such as phase transitions and energy landscapes.

1.2 Spin models for dummies

Most of the work in this thesis deals with spin models of various kinds. The physical reason for calling the entities that make up the models spin is that they can be viewed as small magnets. Microscopically, magnetism is caused by the spin of atoms. A fridge-magnet, for instance, is really nothing but a large number of spins that interact ferromagnetically. The word ferromagnetically means that the spins all want to have their magnetic north pole in the same direction, thus causing a net magnetisation of the sample. If we somehow destroy the ferromagnetic ordering, the small spins will have their north poles in different random directions, destroying the net magnetisation. One way of doing this is by heating the sample.

The primary motivation for studying the models used in this thesis does, however, not come from a need to understand magnetism better. The word spin should instead be interpreted as meaning just an entity that can be in a certain number of different states.

We will denote a spin in the system we wish to study by S_i. Here i is an index that runs from 1 to N where N is the total number of entities in the system. The values that the spins take can be of different kinds. The simplest are so called Ising spins (named after the physicist who first used them to study magnetism, see [9, 10, 11, 12] and also the web site `http://www.bradley.edu/las/phy/ising.html`).

Ising spins are similar to binary variables in that they can take only two values. We call these "up" and "down" and denote them by either ± 1 or 0 and 1. Graphically, Ising spins are often represented by arrows that point up or down. The next simplest kind of spins are called Potts spins and were introduced independently by Potts [13] and Kihiara [14]. They can

have q different values 0, 1, ..., $q-1$ (i.e., $S_i \in Z_q$). (Ising spins are thus a special case of Potts spins with $q = 2$.) We can also talk about spins that are vectors (i.e., arrows); these are called Heisenberg spins. If the vectors are constrained to lie in a plane, we get XY-spins. XY and Heisenberg spins have many interesting properties but will not be used in this thesis.

In addition to specifying what kind of spins a model uses, we must also specify how the spins interact with each other. For Ising spins, interactions can be of two kinds. A ferromagnetic interaction means that the two spins would like to have the same value; if in contrast the spins prefer to have different values, the interaction is called antiferromagnetic. One way of thinking about this is to see the spins as people choosing between two activities that take place at the same time. If two persons are friends, they would prefer to go to the same activity, i.e., the interaction between them is ferromagnetic. Enemies, on the other hand, prefer to go to different activities, thus having an antiferromagnetic interaction. Two friends that are not at the same activity reduce the happiness of the system, and the same happens if two enemies are at the same activity. Physicists don't like to talk about happiness, though, so we call it energy instead. Another word for the same thing is cost-function; in biology the word used is instead fitness and counts the number of satisfied interactions instead of the number of unsatisfied.

Mathematically, the cost-function for this model can be written as

$$H = -\sum_{i,j} J_{ij} S_i S_j. \tag{1.1}$$

Physicists often refer to the H in equation 1.1 as the hamiltonian of the system. The symbols $\sum_{ij}$ means that we sum over all pairs of i and j. The product $S_i S_j$ is 1 if the two spins have the same value (which reduces the energy or unhappiness of the system) and -1 if they are different. J_{ij} are constants that give information on whether i and j are friends or enemies. The simplest case is when there are no enemies in the system, $J_{ij} = 1$.

We can also add a term

$$\sum_i h_i S_i \tag{1.2}$$

to the hamiltonian. This represents a magnetic field that could vary in strength from site to site.

The Ising model is one of the most important models of statistical mechanics. It and its generalisations have been used to model a variety of natural phenomena, ranging from biology to computer science and social science (e.g., [6, 15, 16, 17, 18]). For instance, many social systems can be modelled by letting spin up/down denote different opinions or preferences. In such models, a ferromagnetic interaction is interpreted as two people who prefer to agree, while an antiferromagnetic interactions means

that they want to disagree. A magnetic field adds a bias that can be interpreted as "prejudices" or "stubbornness", while the randomness induced by a finite temperature can be seen as a "free will".

The activity analogy also allows us to introduce the important concept of symmetry. Assuming that both activities are equally interesting, it doesn't matter which activity a particular person chooses. The only thing that matters is that their friends choose the same and enemies the opposite. This introduces a *symmetry* in the system — if we flip all spins, the energy stays the same. Note that the symmetries of a system depend on both the type of spins and also the interaction that is used. The symmetry can be *broken* by applying a "magnetic field" — one example of this could be if one of the activities is more interesting than the other.

There can also be interactions involving three or more spins. If we have three Ising spins, there are eight different possible combinations of up and down. We could assign different energies or happinesses to each of these eight possibilities. Mathematically, this can always be written as

$$J_{ijk}S_iS_jS_k + J_{ij}S_iS_j + J_{ik}S_iS_k + J_{jk}S_jS_k + J_iS_i + J_jS_j + J_kS_k + C \qquad (1.3)$$

where the magnetic field C and the coupling constants J's are suitably chosen. There could also be interactions involving even more spins. The most general hamiltonian for an Ising spin system can be written as

$$H = -\sum_{\alpha}\sum_{i_1,\ldots,i_\alpha} J_{i_1\ldots i_\alpha}S_{i_1}\ldots S_{i_\alpha}, \qquad (1.4)$$

where we first sum over the number of spins α involved in the interaction, and then over all α-tuples of spins in the system.

Depending on how one chooses the J_{ij} in equation 1.4, it is possible to get models of varying complexity. In order for the energy to be well-defined, the matrix J_{ij} must be symmetric, i.e., $J_{ij} = J_{ji}$ for all i, j. There are two sources of complexity in the definition of J_{ij}, one geometrical (i.e., which spins i and j are interacting) and one arising from the values of the interactions.

Note that even if all $J_{ij} \geq 0$, the model can still be disordered. For instance, paper II studies a ferromagnetic model on a random graph. Here the structural/geometrical randomness gives rise to a dynamic frustration that causes the system to become stuck in local minima of the energy landscape.

Things get more interesting if we allow negative J_{ij}'s — it is now possible to get frustration in the model. The simplest case where this happens is illustrated in figure 1.1 where there are three spins antiferromagnetically linked to each other.

Perhaps the simplest way to understand this frustration is to look at it as a spin having an effective antiferromagnetic interaction with itself: a spin model will be frustrated if there is a loop where the product of the

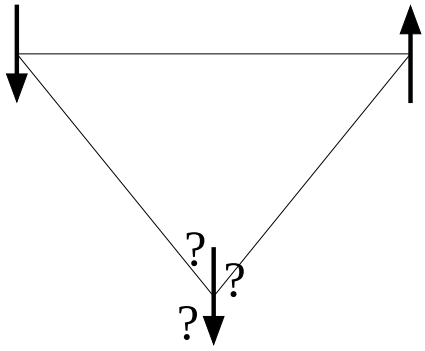

Figure 1.1: The simplest kind of frustration arises if all interactions in a triangle are antiferromagnetic. Regardless of the orientation of the lower spin, one of its interactions will be unsatisfied.

interactions along it is negative. One consequence of this is that even if all $J_{ij} = -1$, the model can be unfrustrated if the geometry of the interactions is such that the lattice has no odd loops. Such a lattice is called bipartite. Perhaps the most common graph to put spins on is the d-dimensional simple cubic lattice, consisting of all points in Z^d. Of course, in numerical simulations it is impossible to have an infinite number of lattice points, so instead Z_L^d is used and one looks at how various quantities scale with the linear size L of the system, hoping to be able extract info on the $L \to \infty$ limit.

A model which has both frustration and disorder is called a *spin glass* (e.g., [6, 7, 8, 19, 20, 21, 22, 23, 24]; see also the web-site `http://online.itp.ucsb.edu/online/lnotes/balents/bignotes.html`). Several of the systems considered in this thesis are spin glasses, and concepts related to it provide the underlying motivation for much of the work that does not refer to it. There are different kinds of spin glasses and the questions on the importance of various kinds of frustrated energy landscapes are not limited to spin glasses; indeed as is shown in paper II, it is possible to get a kind of dynamic frustration even in ferromagnetic models which should be easily solvable.

A simple example of a spin glass is the Edwards-Anderson model, which consists of spins in a d-dimensional lattice with hamiltonian

$$H = \sum_{ij} J_{ij} s_i s_j, \tag{1.5}$$

where the J_{ij} are random variables. The Sherrington-Kirkpatrick model is the case where the J_{ij} are gaußian distributed (and has a second moment that scales inversely with system size), while the $\pm J$ model is the case where J_{ij} is $\pm J$ with equal probability if i and j are nearest neighbours and 0 otherwise. Experimentally realisable spin glasses include $Cd_{0.5}Mn_{0.5}Te$

and $Fe_{0.5}Mn_{0.5}TiO_3$; spin glasses are also relevant in the study of the normal phases of high-T_c superconductors.

Physically, spin glass ordering can be caused by the so called Ruderman-Kittel-Kasuya-Yosida interaction

$$H = \sum_{ij} \frac{1}{R_{ij}^3} \cos(2k_F R_{ij}) S_i S_j, \tag{1.6}$$

where R_{ij} is the distance between spins i and j and k_F is the Fermi wavevector. For more information, see [25] and [26].

Spin models can also be used without an implied hamiltonian. An example of such models are cellular automata; these have been extensively studied by Wolfram [27] and others and are also useful in computing science. These models have explicit dynamical rules that determine how the spins should change in time. In a cellular automaton, the value of a spin S_i in the next time-step is determined by some function (generally deterministic and the same for all spins) of the spins in the neighbourhood of i. Neighbourhoods can here have different meanings — in simple cellular automata the spins are generally placed on a line and the neighbourhood means r spins to the right and to the left of S_i. Another example are the so called voter models. In these, spins are updated by aligning them with a randomly selected neighbour in each time-step.

1.3 Energy landscapes

A spin configuration can be seen as a point in an N-dimensional space: the value of S_i gives coordinate i. Since we can associate an energy to each spin configuration, this gives rise to an abstract *energy landscape* which we can move in by changing the values of the spins.

The space of these spin configurations is called a hypercube. For small N, it is possible to visualise the corresponding hypercube, see figures 1.2 and 1.3. The energy can be seen as giving the height of each site in figure 1.2; this forms the energy landscape.

For larger N, the visualisation gets more difficult, but it is nevertheless useful to think of the spins as living in a landscape where the height represents the energy.

A schematic illustration of a simple energy landscape is shown in figure 1.4. Here it is trivial to find the point with lowest energy, but for the funnel-like landscape in figure 1.5, it is slightly more complicated — the bumps of the funnel could cause us to mistake a local minimum for a global.

Figure 1.6 shows a $2D$ representation of a much more complicated energy landscape. This is an example of a spin glass, while the landscape shown in figure 1.4 can be taken to represent a ferromagnetic Ising model.

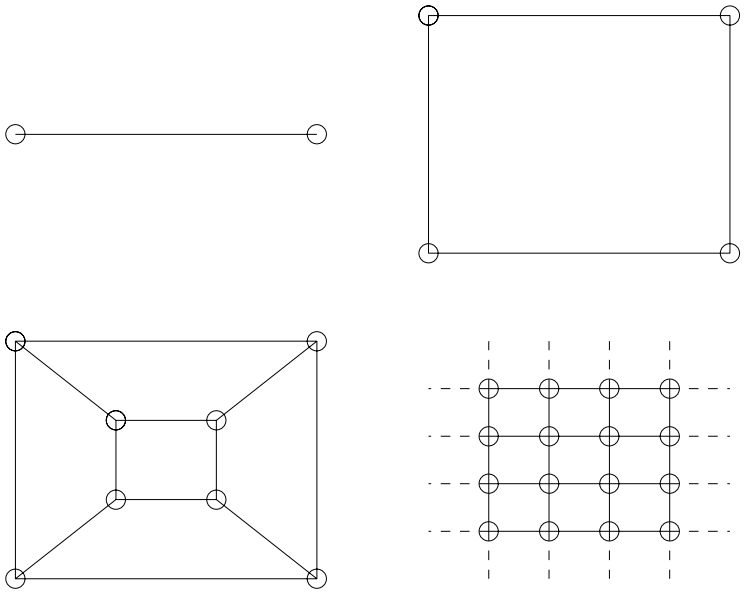

Figure 1.2: Representation of the 1,2,3, and 4-dimensional hypercubes. For the 4-dimensional case, periodic boundary conditions are indicated by dotted lines.

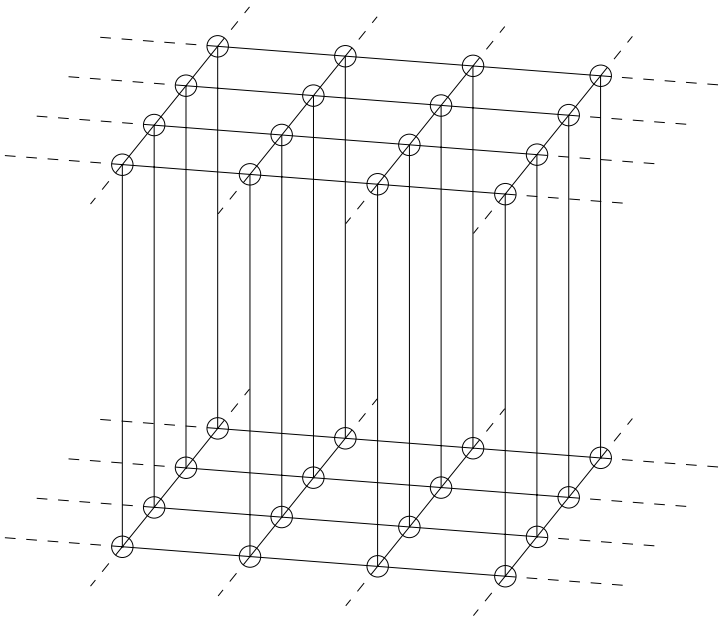

Figure 1.3: 5-dimensional hypercube. Dashed lines denote periodic boundary conditions. The 6-dimensional hypercube can also be visualised in 3D — it consists of two 5-cubes glued together and with periodic boundary conditions in all direction.

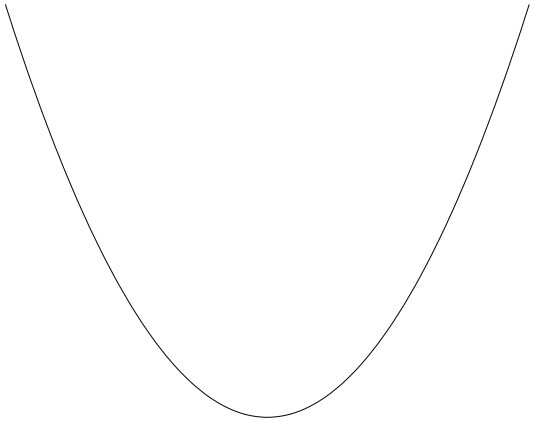

Figure 1.4: A trivial energy landscape. It is very easy to find the (unique) global minimum of this system.

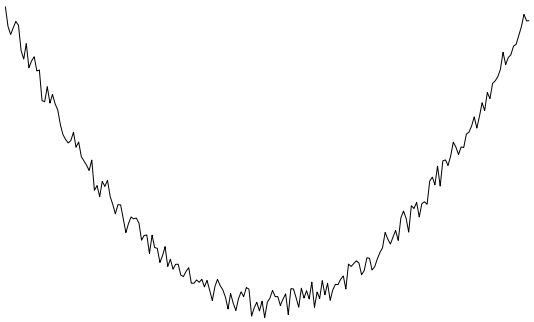

Figure 1.5: A slightly more complicated energy landscape. It is still possible to find the global minimum, but it will sometimes be necessary to climb over energy-barriers on the way to it.

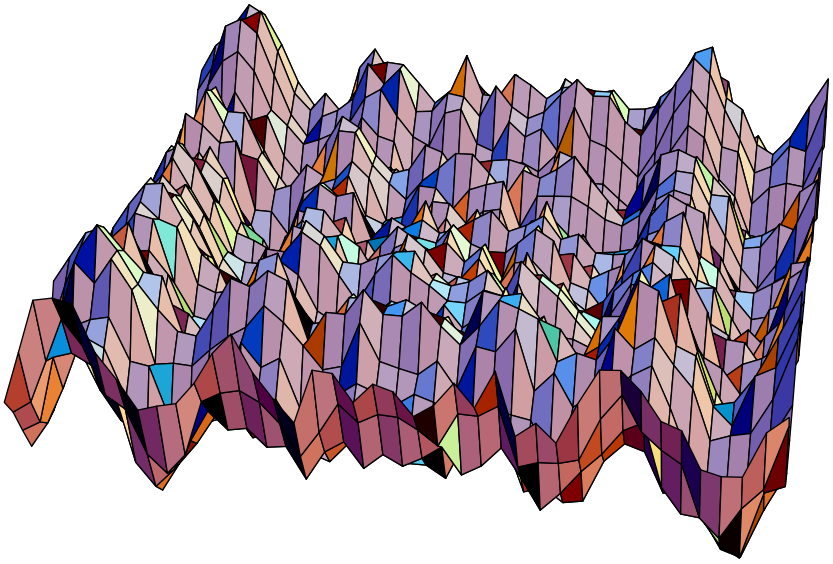

Figure 1.6: A more complex energy landscape. Finding a global minimum of this system is very difficult.

1.4 Phase transitions

One of the most interesting problems in physics is that of phase transitions. Consider a glass of water. If we heat it, it will boil and we get steam. If we instead cool it, it will freeze and turn into ice. These transitions from one state of matter into another are called phase transitions. There are also many other such transitions in physics, for instance a magnet will become nonmagnetic and a superconductor will become an ordinary conductor when the temperature is raised.

In order to describe the physics at the phase transitions, so called *critical exponents* have been introduced. These are numbers that relate physical quantities to the distance in temperature from the transition. For example, the critical exponent β relates the *order parameter* m, a description of "how much" the system is in a specific ordered phase, to the temperature as $m \sim |\frac{T-T_c}{T_c}|^\beta$, where T_c is the transition temperature.

Universality is one of the perhaps most surprising results in physics. It means that the critical exponents are independent of most of the details of the system under study. Often only the dimensionality and the symmetries of the system are important. For much more details on phase transitions and universality, see, e.g., [25, 28].

2

What every physicist should know about computer science

2.1 Introduction — not everything can be computed

"Not knowing everything is all that makes it okay, sometimes."
— Delirium, in Brief Lives

Contrary to popular belief, computer science is not about programming computers. It is about solving problems, and determining which problems are possible to solve. The machine that is used for this or the language in which the method is described is irrelevant and of no interest for computer scientists.

For each problem there are several different algorithms than can be used to solve it. Which one of these algorithms is the "best" one to use? One way of determining this could be to use the one that is most beautiful in the mathematical sense, but for practical purposes it is more interesting to look at the amount of resources the algorithm needs to solve the problem. The most interesting resources are time and space. If we can only solve a problem by using 10^{23} bytes and 10^2 years, the problem is for all practical purposes unsolvable. Since all small problems are easy to solve, computer scientists are interested in how the amount of resources needed to solve a problem scales with its size, N. The size of a problem is a somewhat ambiguous quantity; it is defined as the amount of memory needed to represent an instance of it. For matrix multiplication, N could be either the total number of elements in the matrix or the number of rows or columns (whichever is largest). If we want to factorise a number x, N is the number of digits or bits needed to represent it, i.e, $N = \log x$.

In addition to helping physicists simulate more and more complex systems, computers have also helped mathematicians in providing proofs for

some problems that have so far eluded the search for simple proofs. The most famous example of a theorem that can (so far) only be proven with the help of a computer is the Four Colour Theorem, which states that four colours suffice to colour a two dimensional map of countries [29, 30]. Another example is the proof of the Robbins conjecture by an automated theorem prover in 1996 [31]. A boolean algebra is a set with an unary negation operator $\neg$ and and a binary operator $\vee$ fulfilling associativity and commutativity as well as $\neg(\neg x \vee y) \vee \neg(\neg x \vee \neg y) = x$. The Robbins conjecture states that this third axiom can be derived from the Robbins equation $\neg(\neg(x \vee y) \vee \neg(x \vee \neg y)) = x$. In contrast to the proof of the Four Colour Theorem, the proof of the Robbins conjecture is short enough that it is possible to check it by hand.

Computers can also be used to discover new knowledge. Perhaps the best example of this is the "Assistant Mathematician" program written by Lenat (e.g., [32]) in the 70's. This program, which started with the notion of sets and some simple rules to determine what is "interesting", managed to discover not only integers, addition, and multiplication, but also prime numbers and several elementary results from number theory, such as the Goldbach conjecture. Even more surprising is that it also discovered the concept of maximally divisible number (a number that has more divisors than any number smaller than it, e.g., 12) which, although studied by Ramanujan, is a concept still unknown to many mathematicians. (Note that the program did not prove anything, it only made conjectures regarding what appeared interesting to it.)

Despite these successes there are still many thing that computers are unable to do. One of the problems in modern artificial intelligence (e.g., [33, 34]) is that even though it is possible to make systems that show a certain amount of intelligence in restricted domains it is not yet possible to make a program that has the same general level of intelligence and common sense as a human.

For these reasons it is important to know what limits there are on the power of computers. We can distinguish two different types of questions. The first regards what can be done at all, the second what can be done with limited resources.

It is quite easy to demonstrate that there must exist functions that are uncomputable by computers: the number of programs that can be run on a specific computer is countably infinite, while the number of functions, taking a number as input and giving a number as output, is uncountably infinite.

One of the simplest physical problem that is unsolvable is the tiling problem. Here we are given a set of shapes and the goal is to tile the plane with them. This is of course a very simple problem if the number of distinct shapes in the set is restricted (e.g., if the set consists of just one polygon, we know what shapes it may be in order to be able to tile the plane). But the general problem of determining if an arbitrary set of shapes can tile the

plane is uncomputable; a very readable proof of this for a simple variation of the tiling problem can be found in [35].

Another example of an unsolvable problem is the halting problem. Here the objective is to determine if a computer program will halt or if it loops forever. Assume that there is a program $H(x, y)$ that determines if the program represented by x will halt on input y. Consider the following program M, assumed to take x as input:

```
if(H(x,x)) then
    loop forever;
else
    return 1;
```

It simply determines whether or not the program input to it will halt given itself as input. Now consider computing $M(M)$. If this computation halts, the conditional in M will ensure that it loops forever. But if it does not halt, $H(x, x)$ will return false, which means that the else branch will be taken, so that M halts. The only way to resolve the contradictions is to reject the existence of the machine that computes H, which proves the uncomputability of the halting problem.

The books by Feynman [36] and Dewdney [37] give good elementary introductions to the theory of computation. The rest of this chapter deals with more quantitative differences in the amount of time needed to solve problems.

2.2 Some problems are harder than others

Computer scientist distinguish between problems that are tractable and those that are intractable. Intractable problems are those for which it is known that the fastest way to solve them requires exponential time, while a problem is tractable (or feasible) if there is a polynomial time algorithm to solve it. The border line between these two types of problems is of course fuzzy, since there are a lot of problems for which it is unknown whether they can be solved in polynomial time or not.

Perhaps the simplest example of an intractable problem is the Towers of Hanoi problem. In this puzzle, we are given three bins. One of the bins has N plates on it, sorted in decreasing size from bottom and up. The goal is to move all the plates to another bin. The rules are that we are only allowed to move one plate at a time and that all configurations where a larger plate resides on top of a smaller are forbidden. Figure 2.1 shows an instance with $N = 5$. This problem has beautiful recursive and iterative solutions, but it is easy to show that there is no way to solve it using less than exponential time. Hence, it is an intractable problem.

Some problems abruptly change character when changed just slightly. Consider the problem of dividing a set of numbers into two subsets of sizes

Figure 2.1: The Towers of Hanoi puzzle involves moving the plates one at a time from the leftmost to the rightmost bin in such a way that there never is a plate with a larger plate above it.

n_1 and n_2 so that the difference between the sums of the numbers in set one and those in set two is maximised. This is a trivial problem — just place the largest numbers in one set and the smallest in the other. This can certainly be done in polynomial time. If, on the other hand, the goal instead is to minimise the difference, the problem gets much tougher — the only currently known solution is to try all possible partitions of the set, which takes exponential time.

Another example comes from graph theory. Consider the Euler problem — is there a path through the graph that starts and ends on the same node and visits each edge exactly once? A related problem is the Hamiltonian circuit problem (or travelling salesperson (**TSP**) problem): is there a path of length less than l such that each node is visited exactly once? The former problem is trivial to solve in polynomial time, while the latter is **NP**-complete (see below) and thus almost certainly has no algorithm that runs in polynomial time for all inputs.

Computer scientist have most often only bothered with worst-case analysis of algorithms, but the phase transition found in some constraint satisfaction problems further stresses the importance of obtaining more refined tools to analyse a problem instance's difficulty. It is similar to the situation with sorting lists — here the fastest known [38] algorithm has a worst-case complexity of $N \log N$. But the algorithm that is most widely used has a worst-case complexity of N^2, but an average-case complexity of $N \log N$ and with a smaller constant, and is hence the one that is used most often.

2.3 The Turing machine

> Whether we are based on carbon or silicon makes no fundamental difference. We should each be treated with appropriate respect.
> — Chandra, 2010

In order to compare the complexities of algorithms with each other, it is important to use the same tools to analyse them. For this a universal

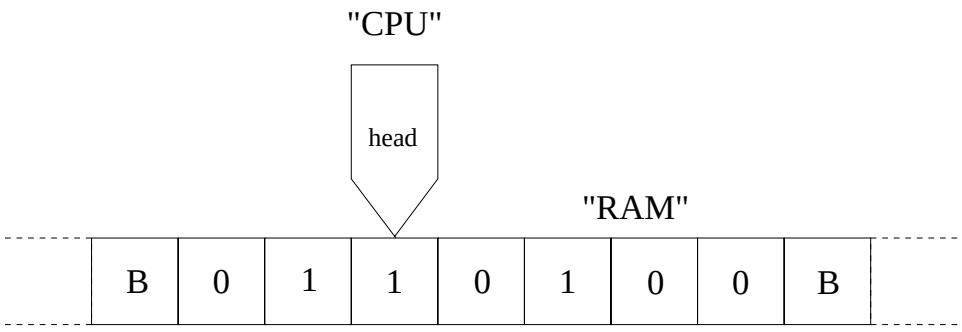

Figure 2.2: Schematic diagram of a Turing machine. It consists of a tape ("memory") and a tape control head ("CPU"). Each cell of the tape contains either a symbol from the alphabet of the machine (here 0 or 1) or is blank (denoted B here). In each time step, the head scans the current cell, changes its internal state and moves one step to the left or right, possibly changing the contents of the cell.

model of computation is needed. One of the simplest such models is the Turing Machine.

A Turing machine consists of a tape and a control mechanism that can read from, write to, and move the tape. We must also specify an alphabet of symbols that may be written on the tape and a set of internal states that the Turing machine may be in. In each time step the control mechanism determines the next state by looking at the tape and its current state and then consulting a transition table that tells it what it should do for all possible combinations of current state and read symbol. The transition table tells the control mechanism what state the machine should enter next, which way it should move the tape, and what symbol it should write in the current cell of the tape (either overwriting the old symbol or just copying it).

The definition of the Turing machine is roughly based on the way a human being computes things. If a person is given a complicated problem to solve, they will solve it step by step. Each step consists of some simple computation that can be done in the head, followed by writing down the result of this computation on paper. The state of the person's brain also changes in each step, to reflect the work that has been done and that which is left to do. For some steps, the person might need to make reference to previous results on the paper. This is completely analogous to a Turing machine — the paper is the tape, the human brain is the tape control unit. We do not yet know the full transition table of the human computer, but the subset used for some simple problems can be determined.

It is easy to see how to construct Turing machines that solve simple problems. But a modern computer can be programmed so that it can be used to solve not just one problem but any (computable) problem. In the

same way it is possible to construct Turing machines that are universal in the sense that they can simulate other Turing machines, designed for solving specific problems. The way to do this is to allow the machine to have two inputs — in addition to the data on which it is to work, it is also provided with a program that contains a complete description of the Turing machine that it should simulate.

Universal Turing machines can be constructed that are very small: Rogozhin [39] has made one version with 4 states and 6 symbols and one with only 2 states and 18 symbols, while Minsky [40] has one with 7 states and 4 symbols. Since the universal Turing machine must simulate the dedicated Turing machine it is often much slower than the Turing machine it simulates. But if the problem takes time t to solve using a dedicated Turing machine, the time needed for a universal Turing machine to solve it is at most a polynomial of t — thus it is possible to simulate other machines efficiently.

There are various modifications that can be made to the Turing machine that do not change its computing capabilities. The tape can be either infinite or semi-infinite (i.e., it has a start but no end), or the machine could even have access to several tapes. None of these additions change anything substantial about the machine, and the more advanced versions can all be simulated (with a polynomial slowdown) by a single tape Turing machine. To simulate a machine with k tapes, the universal Turing machine simply divides its tape into k different parts, adding special markers to its alphabet to see where the tapes start and end. Special care must be taken when the tapes grow, but the details are trivial. The machines used in analysing algorithms often have separate tapes for input and output, and universal Turing machines also often have a separate tape for the program they are running.

There are some examples of where Turing machines have been used in "proper" mathematics. Matiyasevich has used Turing machines to show that Hilbert's 10th Problem (to give an algorithm that solves a Diophantine equation) is unsolvable (e.g., [41]), and to produce a polynomial in 10 variables whose non-negative values give exactly the set of all prime numbers [42]. These results are not important from a practical point of view, since there are far better methods to test for primality, but they are important from a conceptual point of view of mathematics.

2.4 Complexity classes and problems

We will often talk about the time complexity (or just complexity) of a problem. It is understood that this actually means the time complexity of the best known algorithm that solves that problem.

It is convenient to group together all those problems that can be solved by a universal Turing machine with time complexity that is any polynomial

in the problem size N. This class is called **P**. Sometimes it is enough to know with some probability that a problem has a solution. The class of problems for which it is possible to find a solution with probability $2/3$ in polynomial time is called **BPP** (for bounded error probability polynomial). The choice of $2/3$ is of course arbitrary, any probability larger than $1/2$ would do just as well. The class **BPP** can be called the class of problems that can be solved efficiently. (The quantum analogue of this class is important for quantum computers. Some evidence that it is more powerful than its classical counterpart has been obtained in [43].)

Why are there no separate classes for problems that require time proportional to N, N^2, N^3, and so on? The reason is that this difference can be removed by considering more advanced version of the Turing machine. If the computer has superinstructions that are the equivalent of several smaller instructions, it is possible to show that the coefficient of all but the linear part of the polynomial can be made arbitrary small. This means that there is no point, from a theoretical computer scientist point of view, to argue about whether a problem can be solved in N or N^{30} time. (It is also a fact that most problems than can be solved in polynomial time have solutions that are linear, quadratic or at most cubic (matrix multiplication) in problem size.)

There are many important problems for which it is not known if a polynomial time algorithm exists. For some of these problems it is however possible to verify that a proposed solution actually is a solution in polynomial time. So if we guess a candidate solution, or if a friendly genie gives us a candidate, we can see in polynomial time if the guess was correct.

The class of all problems for which this it true is called **NP** (for non-deterministic polynomial). The name comes from the fact that a Turing machine that is allowed to make non-deterministic choices in its computation would be able to solve it in polynomial time.

The archetypical **NP** problem is boolean satisfiability. This is the problem of determining whether or not the variables in a given boolean formula can be assigned so that the formula evaluates to true. For example, $x \wedge y$ can be satisfied by setting both x and y to true, while $x \wedge \neg x$ obviously can not be satisfied. A special form of satisfiability is k-**SAT**, where the formula is written in conjunctive normal form, that is, it consists of the logical and of M clauses, each clause being the logical or of k possibly negated variables. For instance, $(x \vee y) \wedge (\neg x \vee z)$ is an instance of 2-**SAT** with two clauses and three variables. It is easy to see that a general boolean formula can always be written in conjunctive normal form.

It is interesting to note that **Horn-SAT**, a special instance of 3-**SAT** where each clause contains at most one unnegated variable, is in **P**. It can be solved in polynomial time using resolution (e.g., [33]). This method takes advantage of the fact that all Horn-clauses can be written as an implication, e.g., $\neg x \vee \neg y \vee z$ is equivalent to $x \wedge y \rightarrow z$. Also 2-**SAT** can be solved in polynomial time. This algorithm is based on the fact that a clause

$c_1 \vee c_2$ (where c_i can be either a variable or the negation of a variable) means that if $\neg c_1$ is true, then c_2 must also be true. We introduce a directed graph whose nodes are the literals x and $\neg x$, where x is a variable in the formula, and interpret each clause as two edges in this graph, one between $\neg c_1$ and c_2 and the other between $\neg c_2$ and c_1. Satisfiability of the 2-**SAT** formula can be checked simply by seeing if there is a loop from x to $\neg x$ and back to x for some variable x. If there is such a loop, the formula is clearly not satisfiable.

All problems that are in **P** are of course also in **NP**, but it is currently an open question whether or not there are problems in **NP** that are not in **P**. The problems in **NP** most likely not to be solvable in polynomial time are the so called **NP**-complete problems. This class is related to relative difficulties of problems. If a problem π is such that an effective way to solve it would mean that all problems in some class **A** could also be solved efficiently, π is said to be **A**-complete.

The type of problems that are most often used in complexity theory are the decision problems. Here we are not given a function to minimise or maximise (as in physics or economy). Instead, the task is to determine if some specific input belongs to a given set, such as the set of all graphs that are colourable using at most three different colours. If is easy to see that an optimisation problem can always be converted into a decision problem. Instead of asking for the best solution we simply ask whether there is a solution whose fitness (defined in some way) is better than a bound x. An efficient algorithm for the optimisation version of the problem can thus be used to solve the decision problem efficiently.

It is perhaps less obvious that the converse is also true. But given an efficient method to determine if a solution that is better than x exists, better and better approximations to the best value can be obtained using binary search, i.e., by determining which interval the best solution lies in and then halving this repeatedly until the desired accuracy is attained. After the best value has been determined, the problem can be manipulated (by e.g., restricting the domains of some variables or changing their interactions) to determine the variable assignments in a solution. In the satisfiability problem, for example, the value of a variable x_l can be determined by considering a new formula where x_l is set to true. If this formula is satisfiable there is a solution with x_l true, otherwise there must be a solution with x_l false. In either case we can continue with the next variable and determine its value. Note that this procedure does not make it possible to determine all solutions to the problem, it only provides one of them. Enumerating all the solutions requires exponential time.

Another important concept is that of the complement of a problem. The complement of a decision problem π is simply the same problem but with opposite answers, i.e., if π' is the complement of π, then $x \in \pi'$ if and only if $x \notin \pi$. The complement of satisfiability is the problem of determining that there are no assignments that satisfy a given formula. The complement of

a problem that is in **P** is also in **P**, i.e., **P** = **co-P**. This fails however for **NP**, since an evidence for a yes instance can not be converted to saying anything about whether or not the complemented problem has a solution. This is an important distinction between deterministic and non-deterministic computation.

To define completeness, we need to formalise the way a problem can be used to solve other problems. If a solution to π can be converted into a solution of π', we say that π' reduces to π. There is some ambiguity in the definitions of a reduction in the literature. In most textbooks, the exact definition is glossed over and it is simply said that a reduction should require at most polynomial time. A more general definition is that a reduction is a conversion between two problems that requires space that is at most logarithmic in problem size. (It can be shown that this implies that it can not use more than polynomial time). Completeness can now be defined formally. A problem π is complete for a class **A** if π is in **A** and all other problems in **A** can be reduced to π. The class **A**-complete thus consists of the most important, most difficult problems in **A**. If we can solve a **NP**-complete problem efficiently, we have shown that **P** =**NP** since all other problems in **NP** can be reduced to π and hence solved in polynomial time. A problem is said to be **NP**-hard if all **NP** problems can be reduced to it but it is not necessarily in **NP**.

In addition to the time complexity classes introduced here, it is also possible to use space as a resource. For example, **PSPACE** is the class of all problems that can be solved by a universal Turing machine with no time limits but using at most a polynomial amount of memory. There are very interesting asymmetries between time and space. For instance, while it is open whether or not **P** = **NP**, it is known that **PSPACE** is the same as the class of all problems solvable in polynomial space by a non-determinstic Turing machine, so that the addition of non-determinacy does not have any influence on the memory requirements of computations.

NPC is the class of all problems that are in **NP** and are at least as hard as all other **NP** problems, in the worst-case sense. There has also been some work on the concept of *average case complexity*, leading to the introduction of the class **AvP**. **AvP** is the class of all problems such that the average time needed to solve them is bounded by a polynomial. The average is here taken with respect to some probability distribution of the inputs to the problem. See [44] and references therein for a more complete discussion of various forms of average complexity classes.

2.5 More definitions of NP

The ide of “non-deterministic polynomial” computation is one of the most difficult to understand concepts of computer science. It helps to think of it in several different ways — it is the class of problems that can be

verified in polynomial time *and* the class of problems that have a search-tree with a polynomial depth. In order to give the reader more viewpoints on non-deterministic Turing machines, the following list gives several different definitions of **NP**.

- **NP** is the class of problems you can solve in polynomial time if you have an infinite supply of processors (and finding the solution on one processor is solving the problem)
- **NP** is the class of problems you can solve if you can always make a binary decision about which computation path to follow — and always be right
- **NP** is the class of problems for which you can verify that a solution to a given instance is correct in polynomial time
- **NP** is the class of problems for which each instance has a "small" witness (polynomial in the size of the instance) for that it is a "yes" instance
- **NP** is the class of of problems you could solve in polynomial time, if only you could recognize quickly whether or not an arbitrary boolean formula were satisfiable
- **NP** is the class of problems for which there is a short certificate (poly in size of the instance) which can be verified with high confidence by a polynomial time verifier which uses a logarithmic (in the instance size) number of random bits and checks a constant number of bits
- **NP** is the class of problems which can be verified in an interactive game where one player is polynomial time and the other is computationally unrestricted
- **NP** is the class of problems recognizable in polynomial time on a nondeterministic polynomial time machine
- **NP** is the class of problems expressible with a single second order existential quantifier over unquantified second order predicates
- **NP** is the class of problems expressible with a single existential variable with a polynomial (in the length of the free variable) bound on the quantified variable and a polynomial-time computable predicate

2.6 Algorithms for solving constraint satisfaction problems

The obvious method to solve an **NPC** problem is to search through all possible variable assignments until one is found that solves the problem. In some cases, such as **Horn-SAT** or 2**-SAT**, the problem structure is such that there are more efficient methods to construct a solution, but these are exceptions.

The most general techniques for solving constraint satisfaction problem are so called *branch-and-bound* methods. These work by considering all the variables in the problem one after another and simplifying the problem. For

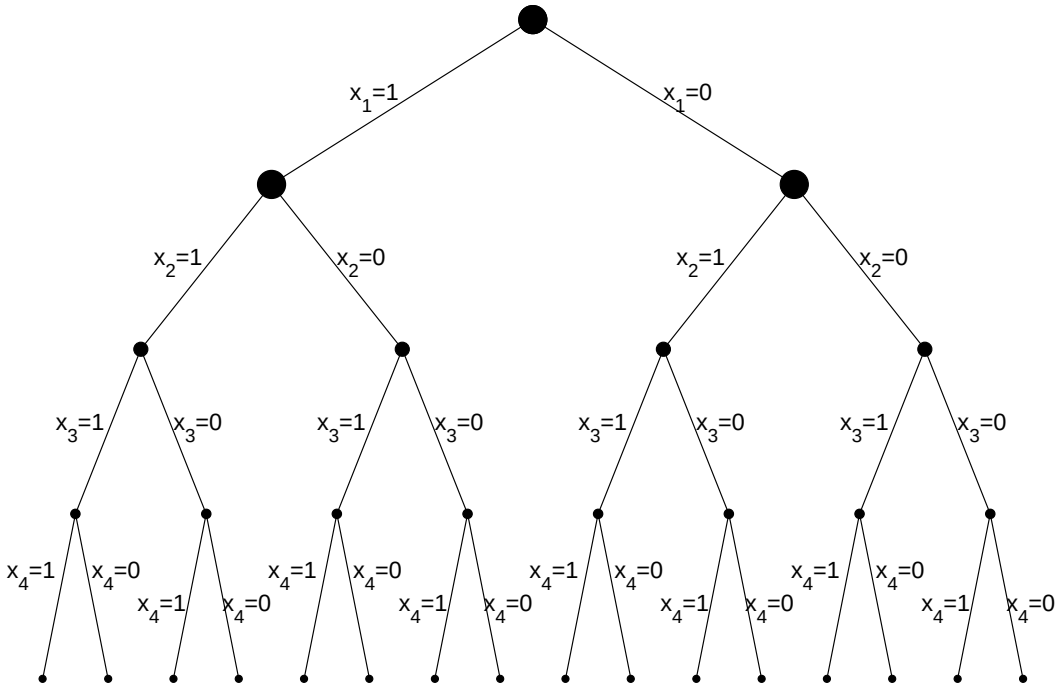

Figure 2.3: The search tree for a problem with four variables, x_1 to x_4 that can each take the value 0 or 1. The leafs at the bottom represent complete solutions, while the internal nodes represent partial solutions where some variables are unassigned. For a problem with N variables, the search tree has N levels and 2^N leafs. A branch and bound algorithm starts at the top and works its way down towards the leafs. If the heuristic used is good, the algorithm will not visit most of the leafs, instead discovering dead ends early and backtracking at internal nodes with low depth. A good heuristic would *prune* the search tree early by not exploring subtrees of nodes that contain a contradiction or of subtrees of nodes that can be shown to be non-optimal.

each variable that we consider look at the smaller problem that results if the variable is set to one of its possible values and eliminated from the problem. If this value causes a constraint to become unsatisfied (or if we by some other means can know that the variable doesn't have that value in the optimal solution of the problem), unset the variable and try another of its possible values. The algorithm then calls itself recursively on this smaller problem. If all values have been tried, one of the previous variables must be changed. The algorithm does this by backtracking until it finds a variable for which all values have not yet been tried. The way that these methods explore the space of possible solution can be very well represented in a *search-tree* such as that shown for a problem with four variables in figure 2.3.

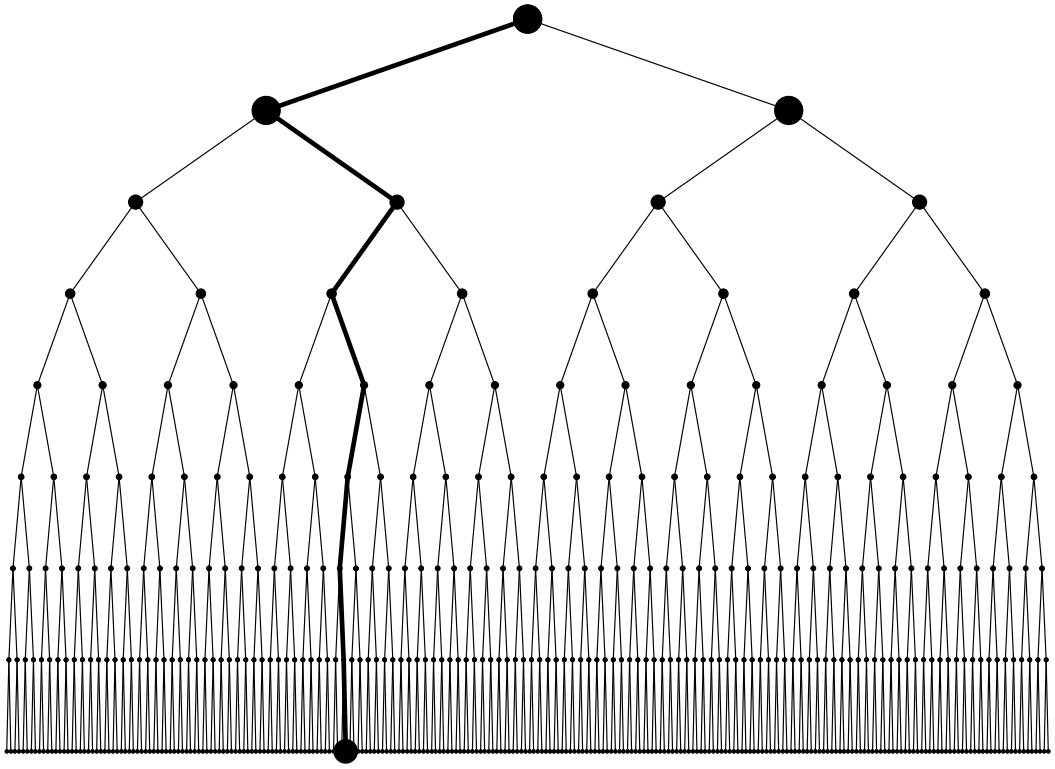

Figure 2.4: This figure shows a slightly larger search-tree. The leaf that corresponds to the solution to the problem is marked by a circle and the path to it is marked by a fat line. The search-tree represents an **NP** problem if the length of this path is a polynomial in the problem size N.

Figure 2.4 shows a larger search-tree with the path to the solution of the problem marked.

In order to construct an efficient search method, it is important to prune the search tree, i.e., discover quickly if a subtree does not possess any leafs that are solutions to the problem. It is therefore important to be able to analyse partial solutions and see if they contain contradictions or are such that the optimal value in the sub-tree below it is worse than the best found so far. In the worst case, the solution is in the last leaf that is examined; this is the reason why the worst-case complexity is exponential.

There are some variants of the general branch-and-bound algorithm that can be used for specific problems; see the literature [45, 46, 47] (and below where short descriptions of the Brelaz and Davis-Putnam heuristics are given).

The Monte Carlo method is one example of a *local search algorithm*, another important class of solution methods. In contrast to branch-and-bound, these methods always work on complete solutions, i.e., they do a walk on the leafs of the complete search-tree. The most straighforward way of ordering the states here are on the N-dimensional hypercube, but other organisations are possible too — we could for instance put the states on a line as drawn in figure 2.3.

Before applying one of these methods to a problem instance, it is often useful to do some preprocessing on it. For k-**COL**, for instance, all nodes with degree less than k can be removed from the problem; these nodes can always be coloured later. The best heuristic for graph colouring is the one introduced by Brelaz [48], which is quite easy to describe. As always, the important part of it is how to decide which variable to branch on first. Brelaz idea for this was to select the node that is the most constrained first, i.e., the one that has the largest number of colours among its already-coloured neighbours.

For satisfiability problems, the method of choice was designed by Davis and Putnam [49] and is called the DP algorithm. It branches first on the variable that occurs in the largest number of clauses. (Note that both these algorithms are more sophisticated than they appear to be in this description. The interested reader is referred to the literature for a more detailed description of them*.)

Physics has inspired several methods for solving optimisation problems. The oldest example is simulated annealing [50], but physics ideas are also useful in for example neural nets (e.g., [51]). Very recently, there have also been attempts to base search algorithms on more modern ideas from physics, such as renormalisation [52] and self-organised criticality [53].

The latter algorithm is called extremal optimisation and can be used for any optimisation problem where it is possible to define a fitness for

*A nice illustration of the DP procedure can be found at `http://www.thoralf.uwaterloo.ca/htdocs/ST_ALGORS/st_dp.html`

each variable. For a spin model, this fitness is simply the contribution of this spin to the energy of the system, or the amount by which the energy changes if the spin is flipped. In each time-step, the variables are ranked according to their local fitness. So far this is similar to greedy search algorithms, which would choose to flip the spin that would decrease the energy the most. Extremal optimisation, however, chooses to flip the spin with rank k with a probability that is proportional to $k^{-\tau}$. The use of this probability function is inspired by self-organized criticality [54, 55, 56]. The value of τ that is best to use depends on the problem at hand, see [53, 57].

2.7 $\exists$ NPC

The first problem shown to be in **NPC** was the satisfiability problem [58]. This was the first paper that asked questions about what problems were and were not solvable efficiently and also provided answers; it launched an entire new field of computer science. It is interesting to note that Gödel, in a letter to von Neumann, had previously speculated that the **SAT** problem might be solvable in linear or square time; see the column by Hartmanis for details [59].

The proof can be found in detail in [47] and [60], here the main ideas will be sketched. **SAT** is in **NP** since it is trivial to check in polynomial time whether or not a given assignment of the variables in a formula really satisfies the formula.

Since we do not even know all problems that are in **NP**, it seems a formidable task to be able to show that all problems in **NP** reduce to **SAT**. The proof relies on the insight that if a problem π is in **NP** then there is a non-deterministic Turing machine that will solve it. We will introduce a way of describing the computation that this non-deterministic Turing machine must do, and then show that this description is in fact a boolean formula, where there are variables representing the complete configuration of the Turing machine at each time step, including the non-deterministic choices that the machine can make. Solving this boolean formula (i.e., the satisifiability problem) in polynomial time would then enable us to construct explicitly a Turing machine that will solve π in polynomial time. But this is the definition of completeness, and hence boolean satisfiability is **NP**-complete.

As argued for above, it is no restriction to assume that π is a decision problem, i.e., given some input the problem is to answer yes or no. We will also assume that there is some specific location on the tape that serves as the output of the program; the contents of this cell will then be taken to give the value of the formula. How can the operation of a non-deterministic Turing machine be described? In each time step, we have to keep track of which state the control mechanism is in and the contents of the tape. If the size of the input data is n, we know that the machine will terminate

after at most n^k steps, for some k. The configuration of the Turing machine at time t will corespond to the t'th row in a table with n^k rows.

How many columns are needed in the table? The tape is infinite, so it would seem that an infinite number of columns would be needed, but this is not the case. Since the tape head can move only one step at each time step it is enough to have n^k cells of tape. (This is a general feature of models of sequential computation: they can not consume more space resources than time.)

In order to describe also the current state of the machine, the alphabet of the machine will be modified subtly. The cell where the tape head is currently positioned will be marked with a symbol that encodes both the contents of the tape at that cell and the current state of the control mechanism.

We also need to capture the non-determinacy of the machine in some way. This is done by introducing choices c_i for each time step i. These choices can represent for instance coin-tosses in a randomised algorithm or the kind of oracle-queries allowed for a non-deterministic Turing machine. It is no restriction to assume that at each step there are two choices, so that $c_i = 0$ or 1.

Now it is easy to see how to convert the description of the machine into a boolean formula. First note that the top row of the table will be fixed by the input data (for technical reasons, it is also necessary to assume that the extreme right and left columns are fixed to be the blank symbol, see [47] or [60] for details). Given a row, we now need to determine how the row beneath it will look like. The only cells that may change are the one at which the tape head is currently positioned and its nearest neighbours (if the head moves to them). The new states of these cells are uniquely determined by the states of the cells above them and the cell determining the non-deterministic choice, and they can in fact be determined by a boolean circuit with those cells as input (we ignore trivial and uninteresting details such as encoding the alphabet using only 0's and 1's, and refer the interested reader to the references). This construction can be continued for all the rows of the table, and it is clear that the boolean formula that determines the state of the designated output cell after n^k steps can be constructed in polynomial time.

By solving this boolean formula, the operation of the Turing machine solving π can be determined, including the correct values of the non-deterministic guesses c_i. (Since π was assumed to be in **NP**, the formula is guaranteed to have at least one solution.)

There are today many more problems that have been shown to be **NP**-complete. To do this for a given problem π it suffices to show that **SAT** can be reduced to π — since all problems in **NP** can be reduced to **SAT**, it follows that any problem in **NP** can be reduced to π by composing the two reductions. The book by Garey and Johnson [61] has an extensive list of **NP**-complete problems; more can be found online at

http://www.nada.kth.se/∼viggo/problemlist/compendium.html.

The big open question in computer science it whether **P** = **NP** or not. This question is so important that it is one of the 7 "Millennium Prize Problems" suggested by the Clay Mathematics Institute†. Even though it is not known whether there exists a polynomial time algorithm to solve **SAT**, some lower bounds on the time needed to solve it are known for machines that also have a restricion on the amount of space that they can use; see [62] for a review.

Another important **NP**-complete problem is graph colouring. This is the problem of colouring all the nodes in a graph so that no node is connected to another with the same colour. It is related to the Four Colouring Theorem, and has a very simple physical interpretation: it is the problem of determining a ground state with zero energy of an antiferromagnetic Potts model. The number of states q in the Potts model corresponds to the number of allowed colours in graph colouring. Graph colouring is also intimately related to all sorts of scheduling problems. Here each event is a node in a graph, and two nodes are linked if and only if the events they represent must not take place at the same time. The number of timeslots that are available corresponds to the number of colours allowed to colour the graph.

Many interesting results can be shown concerning graph colouring. For instance, it is known that there exists graphs which are not colourable with three colours yet contain no triangles [63].

These graphs are actually quite simply constructed using induction. The first example is a graph with one node and zero edges. This graph is not colourable using 0 colours and contains no triangles. Now assume that there exists graphs not colourable using 0, 1, $\ldots k$ colours; we want to construct one that is not colourable using $k+1$ colours. To do this, make copies of the previous graphs 0 to k. Then consider all the ways of selecting one node from each of the previous graphs. For each such way, add a new node that has edges to the the selected nodes in the previous graphs. This graph is not colourable using $k+1$ colours and does not contain any triangles, which proves the theorem.

Another **NP**-complete graph problem is graph partitioning. Here the nodes of a graph are to be divided into k subsets so that no edges join two nodes from different subsets. Physically, this corresponds to a model with ferromagnetic interactions and where there is also some condition on the magnetisation. Many spin glasses have also been shown to be **NP**-complete, including the Sherrington-Kirkpatrick and $3D$ $\pm J$ models.

In general, we can define a constraint satisfaction problem (or **CSP**) as a problem with N variables x_i and M constraints c_i. In general the variable x_i could take on the values $0, \ldots, d_i - 1$, but often the simplification that all $d_i = d$ is made. Each of the constraints consists of a list of variables and

†http://www.claymath.org/prizeproblems/

the combinations of these variables that are not allowed. A constraint involving two variables can be seen as a matrix where the rows and columns correspond to the first and second variable, respectively. For graph colouring the constraints are diagonal matrices for each of the edges in the graph. The ratio between the number of constraints and the number of variables turns out to be an important parameter. In graph colouring, the connectivity or average degree of the graph, $\gamma = \frac{1}{2}\frac{M}{N}$ is used, while for k-**SAT** it is the ratio of the number of clauses and the number of variables $\alpha = \frac{M}{N}$ that is the relevant parameter.

2.8 The structure of NP

There are many interesting questions about **NP**. As stated above, the **P** = **NP** question is perhaps the major open question in theoretical computer science. One of the most beautiful results is that if **NP** $\neq$ **P** then there must be problems in **NP** that are in neither **P** nor **NPC**; the class of all such problems is called **NPI** (for "**NP** intermediate").

The proof of this result consists of an explicit constructions of a problem π in **NPI**. The same method can then be used to construct a problem that is more difficult than all **P** problems but not as difficult as π. This process can be continued indefinitely, so that if **P** $\neq$ **NP**(which is almost certainly the case), there is actually an infinite hierarchy of problems between the "easy" ones in **P** and the "difficult" ones in **NPC**.

π will be constructed in such a way that if it can be decided in polynomial time, then it will be the same as satisfiability, which by assumption is not in **P**. Similarly, if π were in **NPC**, the construction of it will ensure that it is in fact a trivial problem, hence solvable in polynomial time, resulting in the same contraditcion. The only way to resolve these contradictions is if π is in **NPI**.

π will be very simple: given a string x that represents a boolean formula as input, x will solve π if and only if x is a valid and satisfiable formula, and a certain function $f(n)$ where n is the length of x is even. If we can construct $f(n)$ so that it is even if π is in **P** and odd if π is in **NPC**, π will have the required properties and the proof that **NPI** exists will be complete.

The construction of f will ensure that these conditions are met for all inputs of size larger than N_0, where N_0 is some constant. This is enough for the proof, since it is the time complexity as input size goes to infinity that determines the complexity class of π. A finite number of exceptions can always be handled by special cases.

The Turing machine that computes f will work by examining in turn all polynomial time Turing machines and all reductions, giving them all strings as input. We therefore need to assume that there is an enumeration of all polynomial time Turing machines M_i and of all reductions R_i (i.e., R_i is the Turing machine that performs reduction i). We also need to equip the

machines with a stop-watch, so that we can halt them after t time steps. Both of these assumptions are trivial; they are necessary because we need to try the M_i and R_i on progressively longer strings. If a M_i is found that seems to determine π, the output of f will be even. If, on the other hand, a reduction is found that seems to reduce **SAT** to π, f will be odd. This ensures that the promised contradictions arise. f will change parity (and hence move on to trying the next machine or reduction) as soon as it has determined that the Turing machine or reduction currently under test does not correspond to a machine that computes π or reduces **SAT** to it.

The computation of $f(n)$ will consist of 2 steps. Each step lasting for exactly n time steps. During the first step, the tape-head of the Turing machine will advance to the right on the tape, and it will also calculate $f(j)$ for as many $j = 1, 2, \ldots$ as it has time. Let the last such value calculated by $f(j_{max}) = k$. If k is even, let $i = \frac{k}{2}$, otherwise $i = \frac{k-1}{2}$.

The next step of the calculation will now determine the value of $f(n)$ — it will be either k or $k+1$. If k is even, we will try to use the Turing machine M_i, patiently run through all possible strings of length up to n as input data, and try to find some string for which this machine does not give the same result as the machine determining π. If such a string is found, then the computation terminates and $f(n) = k+1$, if such a string is not found before time has run out, we let $f(n) = k$.

One of the contradictions can now be seen easily. For if π is indeed in **P**, then there is some machine M_l that determines it. For all strings that are sufficiently long, f will thus be a constant function, since we can never find a string for which M_l and the machine computing π do not agree. Furthermore, the output of f will be an even number. But this means that for all sufficiently long strings, π is the same as **SAT**, which by assumption did not have a polynomial time solution. So something must be wrong, and we conclude that π can not lie in **P**.

It is now simple to see what must happen if $f(n)$ is odd. In an exactly analogous way, we now test all possible reductions on all possible strings, until we either find one string that can not be reduced to π by the currently tested reduction or time runs out. By similar reasoning as above, if there is a reduction of **SAT** to π (which there must be if π is to be **NP**-complete), then the function f will be constant and odd for all sufficiently long strings. But this means that π is, apart from a finite number of strings, a trivial problem and hence certainly in **P**. Again we have reached a contradiction, and we can conclude that if **P** $\neq$ **NP** then there are indeed problems that are harder than any **P** problem yet easier than any **NPC** problem. One problem that has been conjectured to belong to **NPI** is the graph isomorphism problem, where the task if to determine if two graphs are isomorphic.

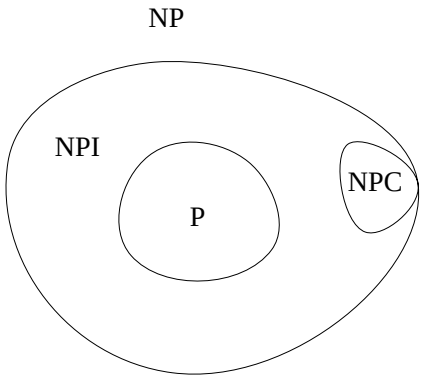

Figure 2.5: The map of **NP**, assuming that **NP**$\neq$ **P**. **NPI** can be further subdivided into an infinite hierarchy of problem classes by the technique used in showing that **NPI** exists.

2.9 Even harder problems

There are several other complexity classes that can be defined in similar way as **NP**. For instance, the subset of **NP**-problems whose optimisation versions do not ask for a bound for the fitness but ask if the fitness is exactly x is called **DP**. Formally, this is the class of decision problems l than can be written as the intersection $l_1 \cap l_2$ where $l_1 \in$ **NP** and $l_2 \in$ **co-NP**. The problems l_1 and l_2 are the ones that ask if there is any solution to the problem with better or worse fitness than x, respectively. For more details on **DP**, see [47]. These problems are much harder than those in **NP**, since they require the ability to check both if a problem has a solution and also if another one does not have a solution. Note that **DP** is not the same as the intersection of **NP** and **co-NP**.

Another extension of **NP** is the possibility of adding an oracle to the Turing machine. Here the Turing machine is equipped with a black box that can instantly solve some problem π. We call the set of problems that this extended Turing machine can solve in polynomial time $\mathbf{P}^{\pi}$, while $\mathbf{NP}^{\pi}$ is the class of all problem that can be solved in polynomial time by a non-deterministic Turing machine with access to an oracle deciding π. This notation is unfortunately somewhat misleading. It is possible to prove that there exists an oracle π_1 such that $\mathbf{NP}^{\pi_1} = \mathbf{P}^{\pi_1}$, but there is also an oracle π_2 such that $\mathbf{NP}^{\pi_2} \neq \mathbf{P}^{\pi_2}$ If π is satisfiability or another **NP**-complete problem, we arrive at the class $\mathbf{P}^{\mathbf{NP}}$ or $\Delta_2\mathbf{P}$. This is the first class in the polynomial hierarchy of classes that are harder than **NPC**. Spin game models that are in various classes of the polynomial hierarchy have been studied in [64].

A very nice survey of important computational complexity results from the last decade can be found in Fortnow's amusing paper [65].

2.10 Other models of computation — universality

> "A Fractran program is defined by a list of positive rational numbers $q_1, \ldots, q_n$. It acts on a positive integer m by replacing it by $(q_i m)$, where i is the least number such that $(q_i m)$ is an integer. If there is ever a time when there is no i such that $(q_i m)$ is an integer, then execution stops. For any computable function $f(.)$ there is a Fractran program which when started with 2^n reaches $2^{f(n)}$ without going through any intermediate powers of 2."

In addition to the Turing machine, several other models of computation have been proposed. The Church-Turing hypothesis states that it does not matter which of these we use — they are all equivalent. If a function can be computed with one of them, it can also be computed with the others. Furthermore, the thesis also claims that all reasonable models of computation give rise to the same set of computable functions. The Church-Turing hypothesis can not be proven, but there is compelling evidence for its validity in the fact that all reasonable models of computation can in fact simulate each other, albeit with large slowdowns. Some other models of computation are the recursive functions, Post's string rewriting systems, and the λ-calculus. In the Post system [66], rules are given for rewriting strings based only on their first symbol — it is quite amazing that this simple system is capable of universal computation. The λ-calculus [67] is the basis for modern functional programming languages — it consists of function definitions and applications.

There has recently been much interest in so called quantum computers. For some problems [68], it has been shown that a quantum computer is more powerful than a classical, but none of these problems are known to be **NP**-complete. It is amusing to note that non-physical quantum computers (that utilise e.g., non-linear quantum mechanics) can solve **NPC** problems in polynomial time (e.g., [69, 70]).

An interesting extension of the idea of the Turing machines has been taken by Pudlák [71], who considers populations of Turing machines that undergo genetic evolution. This is a model of parallel computation, and is more powerful than a single normal Turing machine. The set of all problems computable in polynomial time on such machines is equal to the set of all problems computable in polynomial space on a classical Turing machine, **PSPACE**. This class is strongly believed to be stricly larger than **P** for classical Turing machines, although this has not yet been proven.

DNA computers have also been used to solve some **NP** problems in what seems to be polynomial time [72]. Here the trick to solving seemingly exponential problems is massive parallelism. We fill a container with DNA strings representing all possible solutions to some problem. Using chemistry the DNA strings representing correct solutions are then separated from the others. A problem here is the difficulty of retrieving the correct

solution if there are not many of them. This is exactly the same difficulty that prevents the implementation of a quantum λ-calculus [73].

3

The physics of constraint satisfaction problems

3.1 A phase transition

A constraint satisfaction problem is also a spin glass, which makes them potentially interesting for physicists. In addition to the help that computer science can get from this, it is also possible that the artificial disordered systems studied here can provide new insights into the study of physical disordered systems. Fu and Anderson [74] where among the first to see this connection.

The most interesting feature of these search problems for physicists is that they have been shown to give rise to threshold phenomena very similar to those occurring at physical phase transitions [75, 76, 77]. Another early study of the phase transition is [78], which studies the effects of parallelizing search methods.

For some optimisation problem, it is possible to define a constrainedness parameter so that a randomly chosen problem instance that has a low degree of constrainedness is always solvable, while one that is highly constrained never has a solution. This is of course in a sense trivial, but it is surprising that the boundary between the two cases is sharp. Kirkpatrick and Selman [75] have shown that this transition sharpens as problem size is increased and that finite size scaling can be used to describe it. Friedgut and Achlioptas [79, 80] have shown rigorously that there is a sharp transition for all problem sizes, but there is not yet any proof that the transition happens for the same parameter value for different sizes. Note that some of the methods commonly used to generate more complicated random constraint satisfaction problems have been shown not to have a transition in the thermodynamic limit [81]. The transition can be seen in figure 3.1,

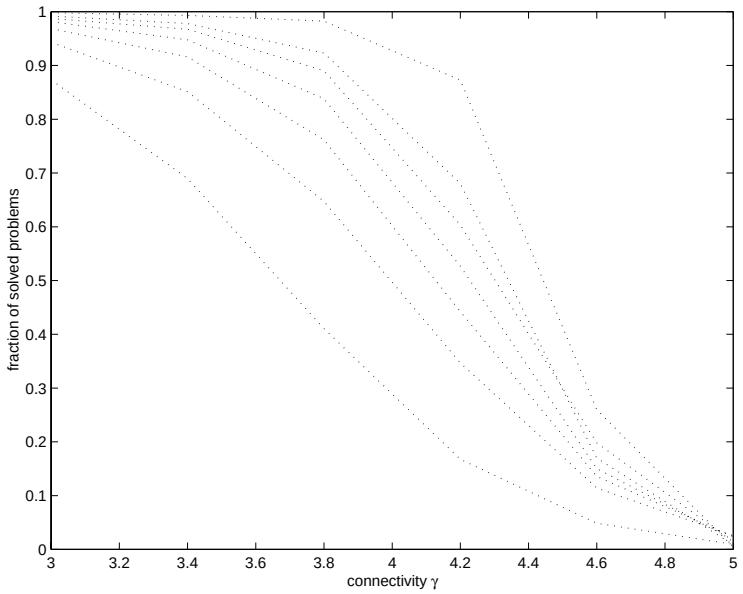

Figure 3.1: This figure illustrates the phase transition in solvability that occurs at $\alpha \approx 4.3$ for the 3-**COL** problem. The plot shows the fraction of colourable graphs for system sizes ranging from 10 (leftmost curve) to 100. The Brelaz algorithm was run on up to 50000 different graphs for each value of γ. The gradual sharpening of the transition as system size increases is indicative of finite-size scaling. Note the restricted range on the x-axis.

which shows how the fraction of solvable problems changes from 1 to 0 for several different problem sizes.

In addition, there is also an "easy-hard-easy" transition in the difficulty of finding a solution or showing that none exist. Is it very easy to find a solution for underconstrained problems — since most variable assignments do not lead to conflicts with others, not much backtracking will be needed. For overconstrained problems, on the other hand, the increased number of constraints makes the search methods quickly run into inconsistencies and not many nodes of the search tree will have to be examined. For problems in the region between over and underconstrained (termed critically constrained), the search method will have to spend a long time searching through dead ends that it can avoid in the other phases.

By treating all the constraints in the problem as independent, it is possible to make an approximation for the number of solutions of a problem with $M = \alpha N$ constraints and N variables (see, e.g., [82, 83, 84]). For simplicity, consider k-**SAT**. Each constraint here involves k variables and forbids one of the 2^k possible combinations of assignments to these variables. Approximate the probability that a constraint is violated in an assignment by $p_v = \frac{1}{2^k}$. Assuming that the constraints are independent then gives $(1-p_v)^M$

as the probability of a formula with M clauses having no clause that is violated. Multiplying with the number of possible assignments, 2^N, then gives an approximation to the number of solutions for k-**SAT**

$$N_{\text{sol}} = 2^N (1 - \frac{1}{2^k})^{\alpha N}. \tag{3.1}$$

To approximately determine the location of the phase transition, solve for α in $N_{\text{sol}} = 1$, giving

$$\alpha_c = -\frac{\ln 2}{\ln (1 - \frac{1}{2^k})}. \tag{3.2}$$

For $k = 3$, this gives $\alpha_c = 5.17$, which is above the true value of $\alpha_c \approx 4.3$.

The annealed approximation described above gives qualitative explanations for both the solvable-unsolvable transition and the easy-hard-easy pattern of the amount of resources necessary to solve the problem. Mammen and Hogg [85] have found that the size of the smallest minimal unsolvable subproblems shows a behaviour that coincides roughly with that of the search cost, and also that search cost appears to be a strictly increasing function of this size. A minimum unsolvable subproblem is a subset of the problem that is unsolvable but becomes solvable if any variable and the constraints in which it appears are deleted from the problem. Obtaining the minimal unsolvable subproblem would thus be a very good heuristic for search algorithm. However, this problem is in general as difficult as finding the optimum solution itself. In recent years, there has been quite a lot work on determining better exact bounds for the phase-transition, see, e.g., [86, 87, 88, 89, 90]. The best bound for 3-**COL** is $4.03 < \gamma_c < 5.05$, while for 3-**SAT** it is known that $3.003 < \alpha_c < 4.64$.

3.2 Nature of the phase transition

The k-**SAT** problem has been studied in detail by Monasson and Zecchina [91, 92, 93, 94, 95] and others [96, 97, 98] who have found interesting analogies between it and random field models. Among other things, they have found that the entropy stays finite at the transition.

It has been established that the occurrence of the sat-unsat-phase transition is due to a finite fraction of the variables in the problem becoming over-constrained, that is they must have the same value in all solutions of the problem [99]. The set of all fully constrained variables is called the backbone and has been compared to percolation. The backbone vanishes in the sat-phase — the presence of any finite number of fully constrained variables could otherwise be used to add an infinitesimal number of clauses that would cause the problem to become unsat. The fraction of sites in the backbone is the proper order parameter for **SAT** and has been shown to have different behaviour for 2-**SAT** and 3-**SAT**. For 2-**SAT**, the fraction

smoothly increase above the threshold $\alpha_c = 1$, while for 3-**SAT** there is a discontinuous or first-order transition. The $2 + p$-**SAT** problem (in which a fraction p of the clauses have three literals while the rest have two) has a continuous transition for $p \geq 0.4$. It has been shown that even though this problem is in **NPC** for all $p > 0$, problem instances do not become exponentially hard until $p > 0.4$ [99, 100]. This led to some early speculation that there could be a relation between the order of the phase transition in a problem and its worst-case computation properties. Recent results however indicate that this is not the case. Achlioptas et al [101] are the first to calculate the exact position of the threshold for an **NPC** problem. The problem they analyse is the 1-in-k-**SAT** problem, which is normal satisfiability but with the added constraint that each clause should have exactly one true literal. They also show that the transition here is second order, thus showing that the order of the transition is in general not related to the problem complexity. This is a very important result, since it means that the hopes of physicist to connect the **P** = **NP** question with the order of the transition have been shown to be futile.

3.3 Analysis of dynamics

Walsh [102] has made an interesting comparison between search methods for constraint satisfaction problems and renormalisation group flows from the theory of critical phenomena. Walsh has studied how the constrainedness changes during search using a variant of the Davis-Putnam algorithm. As described in chapter 2, this algorithm changes the clauses as it traverses the search tree. This means in particular that the ratio α between clauses and variables will change as the solution is approached.

By plotting the constrainedness as a function of search depth and for different initial values of α, an interesting picture is found. For problems that are critically constrained, the constrainedness does not vary much as search progresses. For overconstrained problems, the constrainedness increases rapidly, while for underconstrained problems it decreases just as rapidly. That is, the constrainedness parameter α shows much the same behaviour as the coupling constant of a critical system. Here, starting at the critical coupling temperature means that the coupling constant is constant, while starting above or below the critical temperature will cause the coupling constant to be drawn towards either the high or low temperature fixed point representing the disordered and ordered phase, respectively. The comparison is of course to be expected, but is nonetheless interesting, since it provides a qualitative comparison between search procedures and renormalisation group flows.

A similar — but much more complete — analysis of the phase space of a search method has been performed in recent papers by Cocco and Monasson [103, 104]. In these papers, the authors study the phase diagram

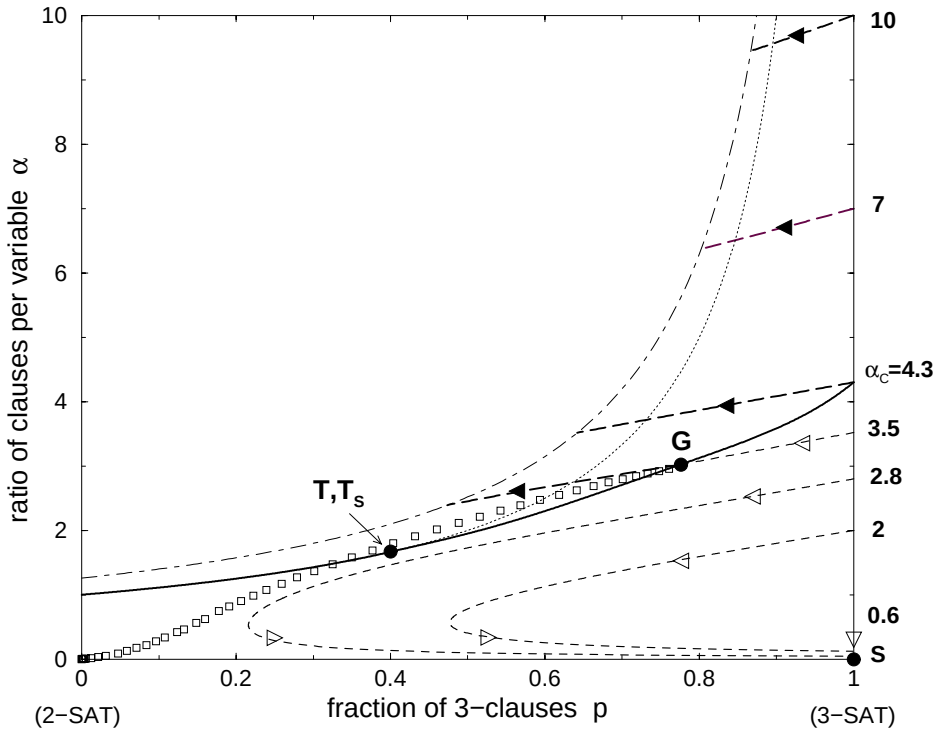

Figure 3.2: This figure, taken from [104], shows the phase diagram of the $2+p$-**SAT** model. The full line is $\alpha_c(p)$; the lines with arrows indicate the trajectories of the DP procedure. For small α, the DP method quickly finds a satisfying assignment of variables, while large α cause the program to backtrack early in the search-tree. For α around the transition, the trajectories are more complicated. For a full explanation of the plot, please consult [104].

of 3-**SAT**, also using the Davis-Putnam algorithm. Since some variables are also removed from the problem, some clauses might change character from involving three variables to just containing 2. This, too, can be captured using the terminology of the $2+p$-**SAT** problem — instead of describing a problem instance using just α, we add p and hence get a two-dimensional phase-diagram. The evolution of α and p can now be tracked as the search-algorithm progresses and we thus get a dynamical trajectory in (α, p)-space. Cocco and Monasson find that the DP-algorithm finds a solution very quickly for all $\alpha < 3.03$, which is far below the transition in the energy landscape that was found in paper I and verified by Biroli et al [105].

In figure 3.2 (figure 4 from [104]), this phase diagram is shown. Several trajectories are shown. Those that start at large α quickly find their path to the unsat fix-point, while those that start at small α find the sat fix-point. For intermediate α, the DP-algorithm need to spend time backtracking be-

fore it can decide whether a problem instance is satisfiable or not; thus the trajectories for $\alpha > 3$ go back-and-forth a lot before finally reaching either a solution or exhausting the search-tree. The hardness of the problem instance is in part determined by the number of times that the DP method must cross the sat-unsat-phase border. Cocco and Monasson also manage to get quantitative results for the time needed to solve the problem based on where the trajectory first crosses the border.

In [106] a similar analysis is made for analog computation for a linear programming problem. Their model of analog computation in this case consists of the solution of a differential equation. Majumdar and Krapivsky [107] have performed an extensive analysis of the binary search problem. While this problem is not **NP**-hard, the analysis is nevertheless interesting since it allows calculation of the height of the tree for an arbitrary distribution of elements to sort.

3.4 Other work

The ground-state energy of some spin glass and optimisation models on random regular hypergraphs have recently been calculated exactly using a replica symmetry breaking ansatz [108]. The same techniques also allows one to get approximate solutions for quantities such as the entropy and ground state energy for problems on hypergraphs with fluctuating connectivities.

Good introductions to the satisfiability problem, some algorithms used to solve it, and the phase transition phenomenon found in it can be found in the introductory article by Hayes [109] and the lecture notes by Pitassi [110]. The survey paper by Cook and Mitchell [44] contains a nice overview of the satisfiability problem, the algorithms used to solve it, and the threshold phenomena found for k-**SAT** from the point of view of a computer scientist. Martin et al [111] is very good and recent overview of what physicists have done in this field.

One of the optimisation problems that has been studied most by physicists is the travelling salesperson problem. It too has been shown to exhibit a transition in solvability [112]. For a nice introduction to a physicist's view of this problem as well as a list of references, see [113].

4

Computer simulation

4.1 Monte Carlo simulation

Statistical physics relies on the computation of measurable quantities with help of the *partition function*

$$Z = \sum_{S_i=\pm 1} \exp\left(-\beta H(\{S_i\})\right), \tag{4.1}$$

where $\beta = \frac{1}{k_B T}$ is the inverse temperature, k_B the Boltzmann constant, and H the hamiltonian of the system. The notation $\sum_{S_i=\pm 1}$ means that we should sum over all possible spin configurations; if the system has N Ising spins, there are 2^N terms in the sum. The *Boltzmann weight* $\exp\left(-\beta H(\{S_i\})\right)$ is interpreted as the probability that the system is in that particular spin configuration; Z can also be seen as the generating function of this probability distribution. (Note that the Boltzmann weights can be derived from information theory using a maximum entropy argument (see, e.g., [25]).)

All measurable quantities can be expressed as an expectation value using this probability distribution. The average energy, for instance, is

$$\langle \epsilon \rangle = \frac{1}{Z} \sum_{S_i=\pm 1} H(\{S_i\}) \exp\left(-\beta H(\{S_i\})\right) = -\frac{\partial \log Z}{\partial \beta}. \tag{4.2}$$

How can we use computers to calculate this? In principle, this is trivial. Just calculate the energy by explicitly summing over all configurations in equation 4.2. The problem with this is that since there are 2^N different configurations, only small systems can be studied.

There is a much simpler method: Monte Carlo simulation [114]. This method determines the energy by a simple average over just a few of the possible configurations, without taking the Boltzmann factors into account.

Monte Carlo (MC) simulation works by starting in a random spin configuration S (i.e., one that is disordered, representative of the high temperature phase). The spin configuration is then changed into a new configuration S', most often by flipping a single spin (Glauber dynamics [115]) or exchanging two spins (Kawasaki dynamics [116, 117]). The difference in energy ΔE of these two configurations is then calculated, and the new configuration accepted with probability $f(\Delta E)$, else rejected. There are various possibilities for f. In the work reported in this thesis we have used $f(\Delta E) = 1$ for $\Delta E \leq 0$, and $f(\Delta E) = e^{-\beta \Delta E}$ for $\Delta E > 0$. The Boltzmann factors are thus considered implicitly in the algorithm. Instead of multiplying the measurements with them and summing over all configurations, they determine which configurations the sum runs over. Normally, time is not increased until N new configurations have been tried; this is called one MC step per spin, or MC sweep, or epoch.

There are several other possible choices for f. The main limitation is that detailed balance must hold, i.e., the ratio between the probability to make a transition from configuration S to configuration S' and the probability of the inverse transition must be equal to the ratio of the configurations' Boltzmann factors (this is the reason for the choice of f above).

For $T = 0$, the Monte Carlo method reduces to a hill-climbing search procedure. It starts with a random configuration and tries to change this locally into a configuration with lower energy. This means that the MC method can get stuck in local minima. Finite temperature simulations also allow transitions that increase the energy, but with exponentially decreasing probabilities.

A variant of the MC method is the *simulated annealing* method [118]. Here the temperature is initially high but is reduced during the simulation. This minimises the risk of getting stuck in a local minima early on and improves the chance of reaching the global minimum searched for. Simulated annealing has been used to solve several optimisation problems with good results [50].

A general problem with the MC method is that care must be taken to ensure that the generated configurations are independent of each other. Data is not collected at each time step but instead with some granularity t_g. Normally, t_g is determined by requiring that some correlation function of the spin configurations is small. Another problem is that the initial random configuration is often not characteristic of the temperature at which the simulation runs. The system must be equilibrated for a long time before measurements can start. The time needed for this equilibration diverges as T approaches the critical temperature as $\tau \sim \xi^z \sim |T - T_c|^{-\nu z}$, where ξ is the correlation length, ν the standard critical exponent relating the correlation length to the temperature, and z the dynamical critical exponent (see, e.g., [28]). This confirms that patterns get more complex (since it takes longer for simulations to determine them) as the temperature approaches T_c.

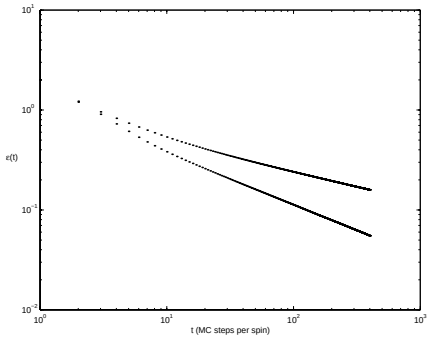

Figure 4.1: Energy relaxation from $T = 0$ Monte Carlo simulations of a square lattice Ising model with $N = 10^4$ spins averaged over 1000 different runs. Results are shown for both spin exchange dynamics, that conserve the order parameter, (upper curve) and single spin flip dynamics, which allow the magnetisation to change (lower curve). There is a clear power law relaxation $\epsilon \sim t^{-r}$ with $r = 1/3$ and $1/2$ for the different dynamics.

4.2 Relaxation

> — Um. What's the name of the word for things not being the same always. You know. I'm sure there is one. Isn't there? There must be a word for it ... the thing that lets you know time is happening. Is there a word?
> — Change.
> Delirium and Dream, in Brief Lives

Figure 4.1 shows how the energy

$$\epsilon = -\frac{1}{N} \sum_{<i,j>} s_i s_j \tag{4.3}$$

changes with time for the standard Ising model with N spins $s_i = \pm 1$ placed on a $2D$ square lattice using both Glauber and Kawasaki dynamics. The sum runs over nearest neighbours $< i, j >$ only. A power law relaxation is clearly seen. Note the anomalous behaviour for early times — this is an example of a cross-over behaviour, where there is one exponent for very early times and another for later times.

In addition to being interesting in itself to study, this relaxation (or coarsening) behaviour also has applications in social science, were the goal is to study how an opinion or a rumour spreads through a population, and epidemiology, where it is important to know how a disease will spread.

Experimentally, it is found that all models with scalar order parameter and discrete up down symmetry have the same behaviour (i.e, are in the same universality class) as the Ising model. In Halperin and Hohenberg's [119] classification, this is model A, while the universality class of

the model with conserved order parameter is model B. Extensive simulation results for some models in these classes can be found in [120]. Note that there is also a hidden assumption that the interactions are local — random graph or small world models do show different behaviour.

In order to explain the relaxation, we first need a continuum description of model A. The appropriate continuum description is the Ginzburg-Landau model, consisting of a order parameter field $m(\vec{r})$ and with free energy

$$F = \int d\vec{r} \frac{1}{2}(\nabla m)^2 + \frac{\mu}{2}m^2 + \frac{\lambda}{4}m^4 - hm \tag{4.4}$$

where h is the magnetic field. Arguments for the validity of this expression as the proper continuum version of the Ising model can be found in [28].

We are interested in how the energy behaves after a quench from a high temperature, disordered state into a temperature below the ordering temperature T_c. The system can not at once go into the composition that is favoured at this temperature. Instead small domains (droplets) that have the same value of the order parameter will form. If these droplets are sufficiently large, they will grow, otherwise shrink. When all droplets have disappeared and the system is homogeneous, the coarsening has stopped. Figure 4.2 shows these droplets in the $2D$ Ising model at $T = 0$ following a quench from a disordered state (see also `http://fy.chalmers.se/~tfkps/java/ising.html`).

How can this growth of the droplets be described quantitatively? Recall equation 4.4; it gives rise to the following equation of motion for the order parameter

$$\frac{dm}{dt} = -\Gamma \frac{\delta F}{\delta m(\vec{r})} = -\Gamma[-\nabla^2 m + \mu m + \lambda m^3 - h] \tag{4.5}$$

where Γ defines the time scale of the system.

Assuming spherical symmetry and making an ansatz $m = m(r - R(t))$, this is rewritten as

$$-\frac{1}{\Gamma}\frac{dm}{dr}\frac{dR}{dt} = \frac{d^2m}{dr^2} + \frac{(d-1)}{r}\frac{dm}{dr} - \mu m - \lambda m^3 + h. \tag{4.6}$$

We want to describe the domain wall, which is in the region $r \approx R$ and has a small width. It is thus possible to replace r by R in the equation. Introducing $v = \frac{dR}{dt}$ gives

$$\frac{d^2m}{dr^2} + [\frac{(d-1)}{R} + \frac{v}{\Gamma}]\frac{dm}{dr} - \mu m - \lambda m^3 + h = 0. \tag{4.7}$$

Equation (4.7) contains a dissipative term $[\frac{(d-1)}{R} + \frac{v}{\Gamma}]\frac{dm}{dr}$. In the absence of a magnetic field h, it can be argued [121] that this term must vanish, giving the velocity v of the interface as

$$v = \frac{dR}{dt} = -\frac{\Gamma(d-1)}{R(t)} \tag{4.8}$$

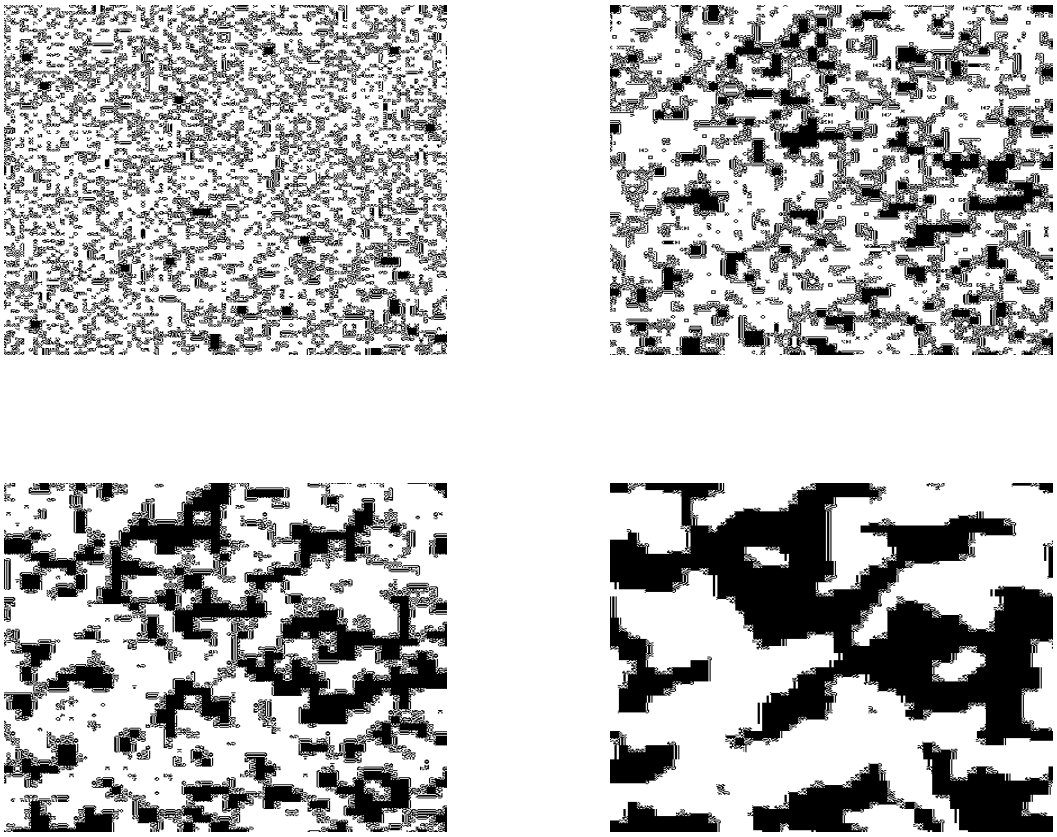

Figure 4.2: This figure shows the time evolution of the $2D$ square lattice Ising model at $T = 0$. Starting from a highly disordered state (top left), the system coarsens and droplets are formed.

with solution

$$R(t) = \sqrt{R_0^2 - 2\Gamma(d-1)t}. \tag{4.9}$$

The interpretation of this result is that after a time t, only those domains whose length scale is larger than $R(t)$ will remain — the system will be ordered on all smaller length scales.

The length scale at which the system has ordered grows as a power law of time. What does this imply for the energy? A domain of radius R will have a surface energy contribution proportional to $\epsilon_d = R^{d-1}$ in d dimensions. If the total volume of the system is N, there will be on average $n_d = N/R^d$ such domains, leading to the total energy of the system being $\epsilon \sim \frac{1}{N}\epsilon_d n_d = R^{-1}$. From the result (4.9) for the growth of the length scale, we then get that the energy of the regular Ising model should grow as $\epsilon \sim t^{-1/2}$, which is consistent with the results from simulations shown in figure 4.1.

For the model with conserved order parameter things get more complicated, but it is possible to argue for the $\epsilon \sim t^{-1/3}$ law that is realised experimentally [121, 122]. This is a model where the ground state is not trivial to find. It can serve as a model for e.g., a system of two types of atoms that can diffuse but can not be converted into each other.

Other universality classes for coarsening phenomena have been identified by Lai, Mazenko and Valls [123]. In addition to model A and B, they introduce two types of system that have logarithmic coarsening, $R(t) \sim R_0 + (AT\ln\frac{t}{\tau})^m$. Such logarithmic coarsening has been found in a simple tiling model by Shore et al [124].

For strongly disordered systems, additional complications arise, since there is currently no consensus on how the energy of an excitation relates to any length scale in the system, or even if there are length scales at all in the systems. For instance, figure 20 in paper I seems to indicate that for the graph colouring problem there are no domain structures in the normal sense.

Experimentally, coarsening can be measured using the equal-time structure factor

$$S(\vec{r} - \vec{r'}, t) = \langle m(\vec{r}, t) m(\vec{r'}, t)\rangle \tag{4.10}$$

which is independent of time in equilibrium, but shows a scaling behaviour $S(\vec{d}, t) = f(|\vec{d}|/R(t))$ during equilibration.

For a list of experimental systems that obey the scaling relation $R(t) \sim t^{-1/2}$, see the paper by Brown and Rikvold [125] which also contains an extensive numerical study of a $3D$ system obeying equation 4.4 of Model A. Note that for large-dimensional systems, it is necessary to have a large linear size and run the simulation for a very long time in order to see the scaling; this is the reason why it was necessary to use extremely large system sizes in paper I.

4.3 Persistence and damage spreading

Recently, a new measure has been introduced in the context of coarsening phenomena. Instead of just measuring the energy or the magnetisation of the spins in the model, it might also be interesting to look at how often a certain spin changes from up to down or vice versa. The fraction of spins that have not yet changed values is referred to as the fraction of persistent spins. This has direct applications in e.g., voter models, where it measures how often a given voter changes their opinion. A voter model is similar to a kinetic Ising model but has no Hamiltonian. Instead, each time a spin is considered a random neighbour is selected and the two spins are aligned with each other. This can be seen as a toy model for how opinions and rumours spread through a population. The most interesting property of voter models is that in dimensions 1 and 2 all the "voters" will eventually reach the same opinion, while in higher dimensions it is possible for minority opinions to survive.

Another interesting quantity to study is damage spreading [126, 127]. Here two or more system are simulated at the same time, using the same random numbers to determine configuration changes and whether or not they are accepted. The question asked is whether or not a small difference in the initial spin configurations spreads throughout the system or if it heals. Interestingly, the qualitative behaviour in damage spreading has been shown to depend on the updating order as well as the choice of algorithm [128, 129, 130].

A natural use of damage spreading is for playing "what if"-type scenarios in models of complex systems. For a voter model, for instance, damage spreading studies how much influence a (small) set of voters can have over the final outcome of the election.

Damage spreading works by duplicating an equilibrium spin configuration of a system and changing a fraction d_0 of the spins. Both systems are then subjected to the same thermal noise and the distance between them is calculated. In Monte Carlo simulations, both systems are simulated simultaneously: the same spin is selected for spin-flip in both systems, and the same random number ("thermal noise") is used to determine whether an energy-raising flip should be performed.

After equilibrating both systems, the Hamming distance (the number of different spins) between the spin configurations S^α and S^β

$$h(S^\alpha, S^\beta) = \frac{1}{N}\sum_i (1 - \delta_{S_i^\alpha}^{S_i^\beta}) \tag{4.11}$$

(where δ is the Kronecker delta function) is measured. The Hamming distance can also be expressed in terms of the Parisi overlap [20]

$$q = \frac{1}{N}\sum_i S_i^\alpha S_i^\beta = 1 - 2h. \tag{4.12}$$

In analogy with the ordering temperature T_c, the spreading temperature T_d is defined as the temperature below which the difference between the systems disappears after again equilibrating the systems [131, 132].

Coarsening as well as persistence and damage spreading are examples of things that are dependent also on the update procedure. This is an important feature of complex systems that is often not emphasised — the dynamics can be part of the *definition* of the model, not, as in Monte Carlo simulations of traditional physical systems, just a tool to study it.

4.4 Difficulties in numerical simulations

It is very important to be careful when drawing conclusions based on numerical simulations.

The first problem with computer simulations is of course to make sure that there are no bugs in the computer code. In addition to thoroughly checking the source code and testing it on systems where the results are known, all algorithms used should be implemented at least two times and the programs run against each other to make sure that they give the same results*.

It is also important to check several different system sizes and make sure that production runs use a large enough system so that finite size effects are minimised. If the results do not converge, it is sometimes possible to make an ansatz for how the results scale with the size and extrapolate to infinity.

If there are several different parameters in the model, their full ranges must be explored. If the definition of the model relies on random numbers, it is vitally important to make an average over several different runs. If the Monte Carlo method is used, there should also be an average over several restarts of the algorithm using different initial conditions. It might also be necessary to use several different kinds of initial conditions (e.g., in damage spreading simulations both equilibrated and random configurations should be damaged).

It is important to estimate error-margins for all measured quantities. Second moments should be computed for everything that is measured, and the results from different runs should be compared against each other. It is far better to store too much information from each run than too little. Measure and store everything. For spin models, the fraction of persistent spins as well as the number of times that individual spins have been updated are example of quantities that are helpful to store when debugging code. If the Monte Carlo algorithm allows several kinds of moves (such as in the polymer simulations described below), the acceptance rate for each move should be stored.

*It goes without saying that the programs should be written at different times so that they are independent. This should be as natural as running the programs through `gprof`.

All random number generators use a seed to generate a sequence of pseudo-random numbers. This seed, along with all parameters describing the system, should be saved in the output of each run, in order to be able to reproduce exactly the same results. It is also nice to store the time used in each data run in the output file — this allows for the collection of interesting statistics from old data files when one should be writing one's thesis.

It is also important to make sure that the random number generator used is random enough. Different generators are good for different things, and one should never just use the system default generator without making sure that it is good enough. Results should always be verified using several different random number generators. A generator that is adequate for all the simulations reported in this thesis is the Mitchell-Moore generator described in [133]. This simple generator has the added advantage of being very fast. For verifying the results, I have used some different linear congruential generators, most often the standard C library's `drand48()`.

There are several important papers on random number generators that should be read by everyone who does numerical simulations [133, 134, 135, 136]. Other important papers to read for anyone considering doing numerical simulations are [137, 138, 139]

4.5 Monte Carlo simulation of polymers

As an example of computer simulations of a system that is not a spin model, this section will present a study of the number of occurrences of small sub-structures in polymers.

Lattice models of polymers have attracted much attention from physicists. Part of the reason for this is the similarities between these models and other interesting physical problems, such as spin glasses [140]. The advance of high-speed computers has also contributed to this trend, since they make it possible to simulate larger and larger systems, and in some cases even to make exact enumerational studies. The problem of protein folding, how a protein attains its complete $3D$ structure is one example where both these factors contribute. Folding generally takes place in a frustrated energy landscape and has been shown to be equivalent to a spin glass (see, e.g., [141]) and to be an **NP**-complete optimisation problem [142]. This problem is also very important from a practical point of view. Experimentally, it is found that many proteins share the same $3D$ structure [143] and that the structures seem to be built out of a small set of recurring motifs. Much work has been devoted to the concept of designability, the fact that many different protein sequences fold into the same three-dimensional structure. Using exact enumeration of all sequences and folds on small lattices, Wingreen and co-workers (e.g., [1, 144, 145]) have shown that some structures are native states of many more sequences than others and that these folds share some of the qualitative features of real protein structures.

Here we look at a simple lattice model for polymers and determine the distribution of small substructures appearing in the collapsed state. The motivation for this is that if there are a few patterns that are repeated often in the ground state, it would be worthwhile to try to build a collapsed state by concatenating these in some sort of Monte Carlo procedure.

The model we use is very simple. We place $n = 50$ beads on a $3D$ lattice of linear size $L = 100$. The system is then equilibrated and we start measuring the distribution of substructures. This process is repeated for n_r restarts.

Chain-crossings are not allowed and the number of beads on each site is restricted to be at most 1. The energy is simply the number of bead-bead contacts,

$$H = -\sum_{i=1}^{N} \sum_{j=i+2}^{N} \Delta(\mathbf{r}_i, \mathbf{r}_j) \tag{4.13}$$

where $\mathbf{r}_i$ denotes the position of the i'th bead and $\Delta(\mathbf{a}, \mathbf{b})$ is one if $\mathbf{a}$ and $\mathbf{b}$ are nearest neighbour sites. The second sum starts at $i + 2$ because we choose not to include the $N - 1$ contacts along the backbone of the chain in the energy.

At each Monte Carlo step in our simulations we try to move each of the N beads in the chain. For the head and tail beads, we only consider a rotation around its nearest neighbour (we could possibly also allow rotations

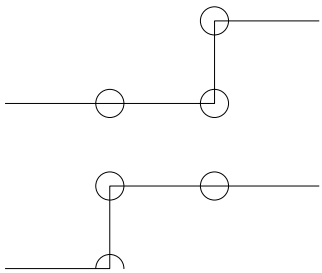

Figure 4.3: The two local configurations that are exchanged in the three beads move.

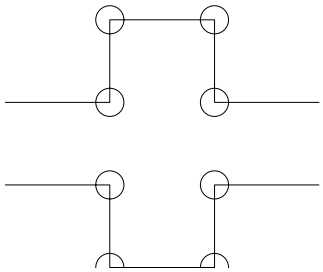

Figure 4.4: The two local configurations that are exchanged in the crankshaft move.

around more distant neighbours). For the beads in the bulk of the chain, three different moves are considered. The simplest is the so called "3 bead flip" move, which is illustrated in figure 4.3 and can be described as moving a kink in the chain. A more complicated move is the crankshaft move (figure 4.4), which flips a "pocket" of the chain inside out. The final move was included in order to make it possible to untie knots of the chain — as shown in figure 4.5 it is similar to the crankshaft move but involves moving 3 beads instead of 2. The crankshaft move is attempted with a probability p_1, while the "3 bead crankshaft" move is attempted with probability p_2. For a more detailed description of and motivation for using these moves, see the reviews by Sokal [146, 147].

In the simulations, we look at the number of times that specific sub-configurations of $l+1$ beads occur. The substrings are represented as strings of length l where each letter in the strings denotes a direction: forward, right, left, up or down. We choose arbitrarily to look for a specific start sub-configuration and then storing the sub-configuration of the next $l+1$ beads. In the results presented here we chose to look for appearances of the pattern "two steps in the same direction, but not three". The motivation for this is that it is a simple pattern that should appear many, but not too many, times for each run. We also need a pattern that specifies local coordinate axes, so that we can compare appearances of the same pattern

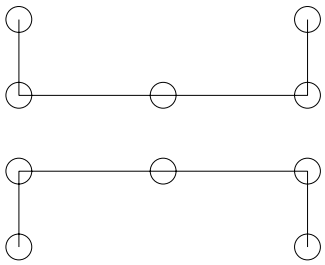

Figure 4.5: The two local configurations that are exchanged in the knot move.

but rotated 90 degrees around one of the axes.

For the results presented here we looked for substrings of length $l = 9$; we also did runs with smaller l's. In order to check the results, we calculated the entropy of the distribution of strings in the different runs. This was stable against both different runs and different sizes of the substrings.

Not very surprisingly, the results turn out to be a power-law. All plots show n on the x-axis and the number of strings that occur n times on the y-axis. A power-law thus means that there is no characteristic number of occurrences. If instead the plots were entered around some value it would mean that most strings occur that many times (this would happen if strings were selected randomly).

We used simulated annealing to equilibrate the systems, running each system for on the order of 10^4 moves before measuring. For each temperature, an average over at least 15000 different runs was performed; each run lasted for at least 20000 time steps.

As can be seen in figure 4.6, there is a power-law for small temperatures. As temperature is increased, this power-law is destroyed and the curves become almost exponential, eventually approaching a gaußian distribution. This can be seen in figure 4.7.

The conclusion to make from figure 4.6 is that it is not possible to build up polymer configurations from a small number of small substrings. There are as many substrings that occur from 1 to 10 times as there are that occurs from 1000 to 10000 times, so the former cannot be ignored.

This section has presented an amusing result from simulations of lattice polymers. It would be interesting to check other models where strings occur naturally (e.g., as interfaces between domains in spin models) and see if they display similar power-laws.

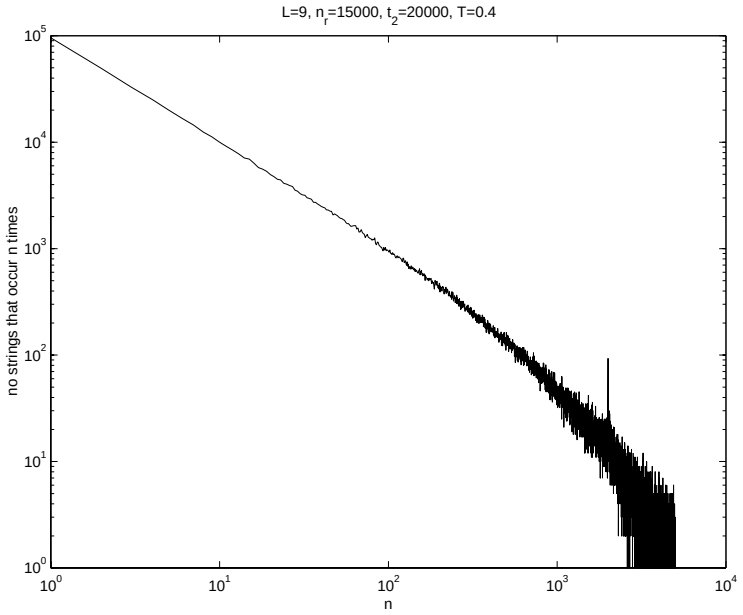

Figure 4.6: This figure shows the number of sub-strings that occur n times for a run with temperature $T = 0.4$. Here we get a reasonable power-law with exponent close to -1.

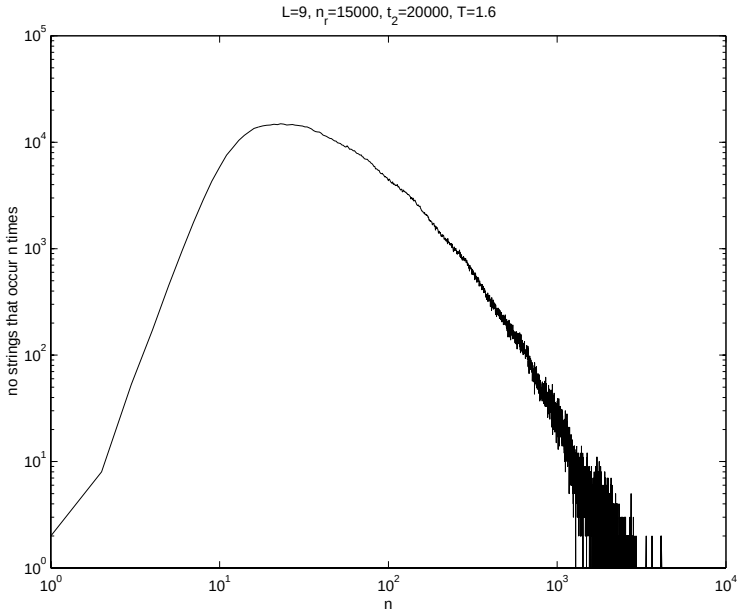

Figure 4.7: This figure shows a distribution centered around a mean value; i.e., random selection of strings. The data shown here was collected at temperature $T = 1.6$.

5

Regular, random and small world graphs

5.1 Regular lattices

In order to properly describe a general network or graph, two things are needed. First, we need a list of the *nodes* of the graph. The nodes can be named and have various properties associated to them, but for describing the graph it is enough that they can be enumerated from 0 to $N-1$. Second, we must know which nodes are connected to which. This is most easily thought of as a list of *edges* (i,j) that are connected. Each edge can have various properties associated to it (e.g., a bond-strength J_{ij}). Mathematically it is sometimes convenient to implement the graph as a function $\phi(i)$ that gives a list of the neighbours of node i. In physics, on the other hand, it is most useful to think of a symmetric matrix J_{ij} where J_{ij} is nonzero if and only if node i interacts with node j. The value of J_{ij} then gives the strength of the interaction (and a negative J_{ij} means an antiferromagnetic interaction, potentially causing frustration in the model).

Many of the graphs used in physics are considerably simpler. Some system are translationally symmetric, meaning that they look the same when moved. This means that it is enough to specify completely the edges among a small subset of the nodes. The rest of the graph can be constructed by placing several such cells next to each other. Some examples of this kind of regular lattices are shown in figures 5.1 and 5.2.

Regular lattices have proven very useful in traditional condensed matter physics. It is of course an oversimplification to assume that only atoms that are close to each other interact. However, since most physical interactions decrease rapidly with distance, it is an approximation that works surprisingly well. All regular lattices have some features in common. By looking

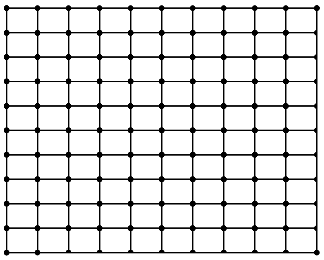

Figure 5.1: A square lattice.

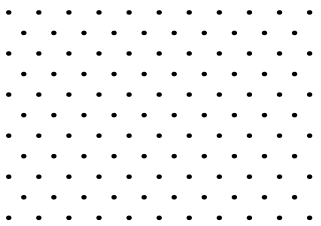

Figure 5.2: A triangular lattice

at the graphs in figure 5.1 it is for instance apparent that these graphs are clustered, in the sense that if we remove one node, its neighbours will still have a short path between them. Another interesting characteristic of regular lattices is that the average distance between nodes is quite large. For a lattice with N sites in D dimensions, it grows as $N^{1/D}$.

A natural extension of the regular lattice is to consider other graphs where all nodes are equivalent (i.e., have the same neighbourhood). The simplest example of such a graph is the complete graph with N nodes, K_N. This consists of N nodes where each node is connected to each of the other (so the graph has

$$\left(\begin{array}{c} N \\ 2 \end{array} \right)$$

edges). This graph is particularly useful when one wants to use approximate methods to study a physics problem. In what is called mean field theory, one makes the approximation that all neighbourhoods are equivalent. This obviously ignores all fine-structure of the problem and it is not a priori clear that it gives correct results. It turns out, however, that if the fluctuations in the system are small enough, mean field theory is exact. In particular, for phase transitions there is an upper critical dimension above which mean field theory is exact.

5.2 Random graphs

Traditionally, two different models of random graph processes have been used [148]. In the first model, $\mathcal{G}(N,p)$, each possible edge (i,j) is considered and included in the graph with a probability p. The other model, $\mathcal{G}(N,M)$ instead selects without replacement M of the

$$\left(\begin{array}{c} N \\ 2 \end{array} \right)$$

possible edges. Note that these models are *not* completely equivalent. For the latter model, the graph is guaranteed to have exactly M edges, while the number of edges is a stochastic variable for the former. In the thermodynamic limit, choosing

$$M = p \left(\begin{array}{c} N \\ 2 \end{array} \right)$$

gives graphs that should share all physical properties. An important quantity characterising different random graphs is their connectivity or average degree. This is given by $\gamma = 2\frac{M}{N} = p(N-1)$ for the two ensembles.

Graph theory is a fascinating mathematical subject with many deep results; see for instance [148, 149]. One of the most interesting results (which will also be useful in understanding some of the results presented in this thesis) is that there is a phase transition as the connectivity of the graph grows. For small connectivities, the graph consists of many isolated trees* of nodes. At $\gamma = 1$ this suddenly changes and a giant component emerges, containing on the order or N nodes. This percolating transition is somewhat surprising — note that the graph can not be connected until it has a connectivity of at least $2(N-1)/N$. Another important result is that the average path-length between two nodes scales as $\log N$ for large N.

Another example of a random graph that is useful in physics is the ϕ^γ Feynman diagram or regular random graph with connectivity γ. This consists of all the graphs where each node is connected to exactly γ other nodes. Notice the difference between this and $\mathcal{G}(N,M)$ — in this ensemble, the average connectivity is fixed, but it can fluctuate between different nodes. The Feynman graphs are particularly useful for theoretical studies since they can be generated by the appropriate action (see, e.g., [150, 151, 152, 153, 154, 155]). A very interesting result for this kind of graphs is that for ferromagnetic models, the loops in them don't matter — the same results are obtained for random regular graphs as for Bethe lattices [156]

A natural generalisation of graphs is to replace the edges by triples i,j,k or even n-tuples. Such structures are called hypergraphs and have natural applications in for instance the k-**SAT** problem. The physical properties of spin models on 3-graphs have been studied by Barrat and Zecchina [157].

*A tree is a connected graph without loops.

5.3 Small world graphs

There are many different kinds of networks in Nature. Perhaps the first that comes to mind is the social network of a society. Here each node represents a person, while there is an edge between two persons if they know each other. What does this graph look like? It is very unlikely that it would be a regular lattice — our acquaintances are not ordered in such a simple way. The social network however shares an important feature with regular lattices: they are clustered. Clustered means that there is a high probability that two neighbours of a given node also are direct neighbours themselves. An alternative way to think about it is to consider the average path length between two neighbours of a node i. Since both nodes are neighbours of i, this is obviously smaller than 2. If node i is now removed from the graph, we have to find a new shortest path between the nodes. If this new path length is still small, the graph is clustered. All regular lattices are obviously clustered, and social networks are clustered too: if person A knows persons B and C, there is a high probability that B and C will also know each other.

Another important feature of social networks is the so called small world effect: When two strangers meet, it sometimes happens that the two people turn out to have mutual acquaintances.

The idea behind small world networks was first introduced by Milgram [158] in 1967. Milgram's experiment consisted of studying the path of letters addressed to a stockbroker in Pittsburgh. The letters were given to people in rural Nebraska with the rule that the current holder of the letter must hand it over to somebody with whom they were on a first-name basis. The average number of links in the chain of people between Nebraska and Pittsburgh was six, hence the term "Six degrees of separation". The number is of course not exact (a severe shortcoming of the experiment was that only one third of the letters were actually delivered!), but the phenomenon that people are linked via a small number of nodes has been verified by later, more careful experiments (e.g, [159]).

The small world effect has later been popularised by occurring in media, such as the movie "Six Degrees of Separation". There are also various amusing games using the same concept, such as the web site `http://www.cs.virginia.edu/oracle/` where a user can find the distance between an arbitrary actor and Kevin Bacon. Actors here represent the nodes of the graph, and two actors are linked if they have participated in the same movie. It should be noted that the actors represented in the database are American and European ones. The network of actors in Indian movies, for instance, probably has few connections to this.

Another example are the Erdös numbers. Named after the famous mathematician Paul Erdös [160], these are defined recursively: Erdös has Erdös number 0; a person has Erdös number $n+1$ if they have co-authored a paper with somebody who has Erdös number n (there are at least 507

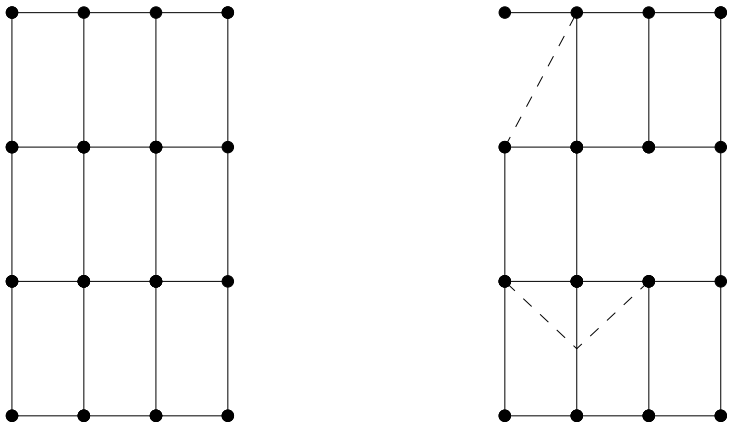

Figure 5.3: This figure shows the construction of a small world starting from a $2D$ square lattice (left). In the right figure, two edges have been rewired and are shown as dashed lines.

persons with Erdös number 1; see the web site
`http://www.oakland.edu/~grossman/erdoshp.html`).

Regular lattices do get shorter and shorter distances between nodes as the dimensionality increase (the diameter scales as $N^{1/d}$ for a d-dimensional lattice with N nodes), but this is still to large. Instead, new graph models are needed.

A small world graph is intermediate between a regular lattice and a random graph — it has both clustering (like a regular lattice) and short maximum distances (like the random graph). It is constructed by considering in turn all the bonds of a lattice and with some probability p replacing them with a random bond. So by changing the rewiring probability p we can interpolate between the regular lattice and a random graph. An example of a small world obtained by rewiring the $2D$ square lattice is shown in figure 5.3.

Note that the small world for $p = 1$ differs slightly from a random graph, since all nodes are guaranteed to have a local connectivity of at least $\gamma/2$ where γ is the connectivity of the regular lattice. The distribution of connectivities is more broad for the small world with $p = 1$ than for the corresponding random graph.

The advance of the Internet and other communications networks has highlighted the need to be able to not only describe but also design networks that communicate efficiently. Efficiently here has two distinct meaning — the obvious one that a message from A to B should be transmitted along the shortest possible path, and also an equally important one that

the network should be fail-safe. If a node suddenly disappears, it should be possible to quickly find alternate paths between the rest of the nodes that don't involve the dead node. A very clear definition of small world behaviour in terms of *efficiency* has been given by Latora and Marchiori [161]. Clustering is here a local efficiency, measured as the time needed to communicate in the network, assuming unit velocity of signal propagation. The efficiency between two nodes is thus

$$\epsilon_{ij} = \frac{1}{d_{ij}} \tag{5.1}$$

where d_{ij} is the shortest distance between nodes i and j and $d_{ij} = \infty$ if there is no path between the nodes. The global efficiency is the average of this over all pairs of nodes in the graph. A high global efficiency corresponds to a small diameter of the graph. The local efficiency for a node i is calculated as an average of ϵ_{ik} over all neighbours k of i, and the total local efficiency of the graph is then the average of this over all nodes. The local efficiency is a measure of the fault tolerance of the network, or of the clustering.

A small world graph still has the same poissonian distribution of node-connectivities as a random graphs, but many graphs in Nature instead have a power-law distribution. Such graphs are called scale free and have recently attracted a lot of interest, see, e.g., [162, 163, 164, 165].

A surprising connection between scale-free networks and traditional physics has been found by Bianconi and Barabás [166], who map a growing network model to a Bose gas and find that there is a possibility of Bose-Einstein condensation occurring for the system — in this phase all nodes are connected to one central node.

Examples of naturally occurring graphs that can not be modeled by random graphs include the Internet, telephone networks, airline timetables, electric power grids, neural networks, interactions between monomers in a folded protein, metabolic pathways of biological organisms, electrical circuits, networks of sexual contacts and collaboration graphs of film actors, business people, and scientists (see [167, 163, 168, 169, 170, 171, 172, 173, 174, 175, 176] and references therein).

Much of the analytical work on small world networks has used a variation of the standard rewiring model [177]. In this model edges are never deleted from the graph, only added. This means that the mean connectivity of the graph will not be preserved (this is the reason why this model was not used for the work presented in papers III and IV.

The average distance between two nodes has been proved [178, 179] to have a scaling form

$$l = \frac{L}{2d\gamma} F(p\gamma L^d) \tag{5.2}$$

where F is an universal scaling function. $x = p\gamma L^d$ is two times the average number of shortcuts of the network. F is the fraction by which the average

distance between nodes is reduced by the introduction of shortcuts. F approaches 1 for small values of its argument, while it grows as $\frac{\log x}{x}$ for large values; this agrees with our intuitive notion that l should be L for small p and $\log L$ for $p \to 1$. Additional properties of this small world model have been calculated by Almaas et al [180].

The eigenvalues of the adjacency matrix J_{ij} of both scale-free and small world graphs have been calculated by Farkas et al [181]. For random matrices, the spectral density of eigenvalues is known to converge to a semi-circle function, but they find completely different behaviour for realistic graphs. In particular, small-world networks turn out to have spectra that can not easily be described by any function. This is yet another indication that it is not enough to put models of interacting agents on random graphs. A similar study has been made by Monasson [182]. It would be interesting future work to do the same for small worlds starting from d-dimensional regular lattices.

Exact calculations of many properties of these models are still missing; the analogue between the occurrence of the small world phenomenon at $p = 0$ and a first order phase transition [183, 177] can perhaps lead to better result in the future.

Another alternative model has recently been introduced by Holme and Kim [184]. Here instead of adding single nodes triangles are added. This leads to a model that is both scale-free (i.e., has power-law distribution of connectivities) and shows clustering. The most important advantage of this model is that it provides for an easy way to control the amount of clustering in the graph.

Dynamical properties are often more difficult to analyse than static. One of the first dynamical models that was studied for small world graphs is the density classification problem. Here a cellular automaton is given the problem of determining whether the majority of its inputs is 1 or 0. This problem was shown to be easier for CA placed on small worlds than on regular lattices [168, 185]. The same authors also found that for the game "Prisoner's Dilemma", cooperation arose *less* frequently on a small world than on a lattice. Neural networks [186] have been found to both have fast responses and possess coherent oscillation on small world graphs. No other network structure has been found that combines these two properties.

Rumour propagation is another example where the network plays a vital role, as is voter models of various kinds. It can be argued that the results in paper III imply that such models placed on square lattices are inaccurate and that they instead need to be put on small world or random graph networks. Disease spreading has been investigated in among others [179] and the position of the percolation point has been found analytically for $1D$ small wold networks by Moore and Newman [187, 188]. It has been shown [189] that for a scale-free graph an epidemic will spread for an arbitrary small transmission probability. Dezsö and Albert have how-

ever shown that immunising a sufficiently large number of nodes with high connectivities can lead to a finite effective epidemiological threshold [190].

Some work has been done on evolutionary models on small world graphs. Kulkarni and co-workers [191] have found that the nodes with the highest connectivities have the most evolutionary activity in the Bak-Sneppen model. In most models of evolution, only point-mutations that change one gene are allowed (this corresponds to flipping one spin in a Monte Carlo simulation). Bagnoli and Bezzi [192] have studied the effect of allowing a small fraction of larger mutations, which has the effect of turning the hypercube of genotypes into a small world network. They find that by allowing a small number of such short-cuts the same results are obtained as when arbitrary mutations are allowed. This is of course good news for the speed of exploration of the landscape, but it is bad news for the stability!

Most of the work on small world networks has started by rewiring a one-dimensional ring lattice, but in papers III and IV we instead use the $2D$ square and $3D$ simple cubic lattices. It should be noted that these small world networks differ from those obtained by rewiring a ring lattice in one respect: their clustering coefficient does not display the same threshold behaviour as a function of p: it starts at 0 for $p = 0$ (since the regular lattices used are bipartite) and then grows to the random graph value. The graphs used here are however still clustered in the sense that if j and k are neighbours of i, then there is a short path between them that does not pass through i (i.e., they are locally efficient).

The use of small world graphs to study physical models has so far been limited. Barrat and Weigt [193] and Gitterman [194] have used them to study the crossover from $1D$ to mean field behaviour for the ferromagnetic Ising model, finding a disorder-order transition at a finite temperature $T_c(p)$ for any $p > 0$, provided that the system size is large enough. A very recent Monte Carlo study [195] of this model was unable to determine whether the transition happened only for finite p larger than some critical p_c. This is probably due to finite-size effects.

The phase transition in the XY model on a $1D$ small world lattice has been investigated by Kim et al [196]. The find the expected appearance of an order-disorder phase transition and calculate the critical exponents which are found to take mean field values. It would be useful to study the crossover from $2D$ to mean field behaviour for the XY model by putting it on the small world lattices used in paper III.

6

The papers

In paper I we study the relaxation of the energy, defined as the number of unsatisfied constraints, of k-**COL** and k-**SAT**. We find that there is a change from fast, exponential decay to power law relaxation as the constrainedness (measured by the connectivity γ in k-**COL** and by the ratio of the number of clauses to the number of variables α in k-**SAT**) is increased. We also find evidence for a freezing transition: above a certain γ_c the $T = 0$ Monte Carlo method can no longer find a state with vanishing energy. Evidence is presented that the former transition exhibits finite-size scaling behaviour. Both the changes occur for smaller values of γ and α than the solvability transition. The change in the energy landscape can be seen as the landscape transforming from a funnel-like (like that shown in figure 1.5) landscape into a more rugged one. This change in the appearance of the energy landscape for k-**COL** and k-**SAT** has been studied further by Biroli et al [105] who by using a variational approach find that for $\alpha > 3.96$ the ground-states (as well as the low-lying excitations) of 3-**SAT** organise themselves into many clusters, separated from each other and unrelated by symmetry. Paper I also presents some data for the fraction of persistent spins $r(t)$ in k-**SAT**. These data seem to indicate that there is a transition also in the behavior of $r(t)$.

Paper II puts ferromagnetic Ising models on random graphs with connectivity γ. For this model the $T = 0$ ground state is trivial for all γ, but a dynamical freezing transition for the Monte Carlo algorithm is nevertheless found: The model freezes for a range of γ, while there is exponentially fast decay for very small and large connectivities. There is a smaller range of connectivities for which the freezing persists even using simulated annealing. The related model with conserved magnetism (which is a model for graph partitioning, another **NPC**-problem) displays exponential relaxation and freezing, and here there is no improvement using simulated annealing [197]. Similar results have been found by Spirin et al [198] for the fer-

romagnetic Ising model on regular lattices. The freezing could turn out to be important for models of choice-making agents, such as the voter model. Note that Berg and Sellitto [199] have recently studied the same problem using a replica symmetric ansatz; their figure 4 supports the freezing results from paper II.

Papers III and IV deal with spin models on small world graphs. Paper III is about the ferromagnetic Ising model on small world graphs. By damaging an equilibrium spin state and determining the temperature T_d below which this damage heals, we find that there is a qualitative difference between the possible behaviours. For the $2D$ model with $p > 0$ and the $3D$ with $p \geq 0$, very good data collapse can be obtained, while the $2D$ $p = 0$ instance stands out with its own behaviour. This difference might be related to the differences between random walks on the corresponding lattices.

Paper IV continues the study of constraint satisfaction problems by looking at graph colouring on small worlds. The use of this kind of graphs provides a novel way of looking at the phase transition since the average connectivity (and hence the properties of the problem) depends on what starting lattice is used. The paper also reports on a new ensemble of graphs that are hard to colour using the Brelaz heuristic.

7

Suggestions for future work

The common theme of this thesis is complex energy landscapes, and in particular the influence of the interaction graphs on local search in them. There are still many interesting unanswered questions in this area.

The increasing speed of computers make it possible to do exact studies on larger and larger systems. This opens the door for more thorough investigations of the statistical properties of disordered energy landscapes than have previously been done [200]. In particular, looking at the geometries induced on the energy landscape by different dynamics could perhaps lead to interesting results for the constraint satisfaction problems studied here.

Another interesting issue worth addressing is the relationship between the backbone of over-constrained variables and the fraction of unflipped spins shown in figure 21 of paper I.

A possible combination of the work in papers I and II is to look at a model containing both ferromagnetic and antiferromagnetic bonds.

The most important future work that should be done on these systems is however to put them on small world and scale-free graphs. Any attempt to study models of social agents *must* use an interaction graph that is similar to those occurring in Nature. For some models, it might of course turn out that the choice of graph does not influence the results, but this must be checked numerically or argued for analytically.

It is surprising that so much work has been done using the original $1D$ version of the small world, considering that using a higher-dimensional lattice as starting point could lead to different behaviour (as indeed it does for the models studied in papers III and IV).

The crossover from a finite-dimensional universality class to mean field behaviour should also be investigated by placing Ising models on larger-dimensional small world graphs. Paper IV studies antiferromagnetic models, it would also be useful to look at how other geometrically frustrated models (see, e.g., [201]) change as the rewiring probability is increased.

Small worlds can also be used to study the behaviour of spin glasses in different dimensions. One reason why it is particularly interesting to look at spin glass models on small world graphs is the difference between $2D$ and $3D$ spin glass models. For short-range models on a $2D$ lattice, the so called droplet model is applicable, whereas current evidence [202, 203, 204, 205] seems to indicate that the situation is more complicated in $3D$, where a mix of both droplet and Parisi's replica symmetry breaking model (see, e.g., [20, 24, 21]) seems to be valid. An additional interesting difference is that the ground state of a $2D$ spin glass model can be found in polynomial time (e.g., [206]), while it is an **NP**-complete problem to do this for the $3D$ version. Very preliminary results on the stability of ground states of the $3D$ $\pm J$ model seem to indicate that there is no qualitative change between $p = 0$ and $p > 0$ [197], but this needs to be studied much further.

> "There are worlds out there where the sky is burning, the seas sleep, and the rivers dream; people made of smoke, and cities made of song. Somewhere there's danger, somewhere there's injustice, and somewhere else the tea is getting cold. Come on Ace, we've got work to do."
> — Doctor Who, the last episode

Bibliography

[1] H. Li, R. Helling, C. Tang and N. Wingreen, *Emergence of Preferred Structures in a Simple Model of Protein Folding*, *Science* **273** (1996) 666–669.

[2] J. P. Crutchfield and D. P. Feldman, *Statistical Complexity of Simple 1D Spin Systems*, *Phys. Rev. E* **55** (1997), no. 2, 1239R–1243R.

[3] D. P. Feldman and J. P. Crutchfield, *Measures of Statistical Complexity: Why?*, *Phys. Lett. A* **238** (1998) 244–252.

[4] George A. Cowan and David Pines and David Meltzer, *Complexity: Metaphors, Models, and Reality*, vol. XIX of *Santa Fe Institute Studies in the Sciences of Complexity*. Addison-Wesley, Reading, Ma, 1994.

[5] L. P. Kadanoff, *Turbulent Heat Flow: Structures and Scaling*, *Science* **54** (August, 2001) 34.

[6] P. W. Anderson, *Spin Glass Hamiltonians: A Bridge Between Biology, Statistical Mechanics and Computer Science*, in *Emerging Syntheses in Science*, D. Pines, pp. 17–20. Addison-Wesley, Reading, Ma, 1984.

[7] M. Mezard, *On the ubiquity of spin glass concepts and methods*, *Physica A* **200** (1993) 111–117.

[8] P. W. Anderson. Reference Frame articles in *Physics Today* January, March, June, and September 1988, July and September 1989, and March 1990.

[9] E. Ising *Z. Phys* **21** (1925) 613.

[10] S. Kobe, *Ernst Ising — physicist and teacher*. eprint cond-mat/9605174.

[11] B. Hayes, *The World in a Spin*, *American Scientist* (September-October, 2000).

[12] P. M. C. de Oliveira, *Computing Boolean Statistical Models*. World Scientific, 1991.

[13] R. B. Potts *Proc. Camb. Phil. Soc* **48** (1952) 106.

[14] Y. M. T. Kihiara and J. Shizume *J. Phys. Soc. Jpn.* **9** (1954) 681.

[15] W. Li, *Nonlocal Cellular Automata*, in *Pattern Formation in the Physical and Biological Sciences*, H. F. Nijhout and Lynn Nadel and Daniel L. Stein, no. V in Santa Fe Instutite Lecture Notes, pp. 189–200. Addison-Wesley, Reading, MA, 1997.

[16] R. Dickman, *Ising Meets Ornstein and Zernike, Debye and Huckel, Widom and Rowlinson, and Others*. eprint cond-mat/0012079.

[17] S. Galam, *Rational group decision making: A random field Ising model at* $T = 0$, *Physica A* **238** (1997) 66.

[18] K. K. Yee, *Introduction to Spin and Lattice Models in the Social Sciences*. eprint nlin.AO/010602.

[19] K. Binder and A. P. Young, *Spin glasses: Experimental facts, theoretical concepts, and open questions*, *Rev. Mod. Phys.* **58** (1986), no. 4, 801–976.

[20] M. Mézard, G. Parisi and M. A. Virasoro, *Spin Glass Theory and Beyond*. World Scientific, Singapore, 1987.

[21] A. P. Young, *Spin Glasses and Random Fields*. World Scientific, Singapore, 1998.

[22] D. Sherrington, *Spin Glasses*. eprint cond-mat/9806289.

[23] J. A. Mydosh, *Spin Glasses: An Experimental Introduction*. Taylor and Francis, 1993.

[24] K. H. Fischer and J. A. Hertz, *Spin Glasses*. Cambridge University Press, 1991.

[25] M. Plischke and B. Bergersen, *Equilibrium Statistical Physics*. World Scientific Press, 1994.

[26] N. W. Ashcroft and N. D. Mermin, *Solid State Physics*. Saunders College Publishing, 1976.

[27] S. Wolfram, *Statistical mechanics of cellular automata*, *Rev. Mod. Phys.* **55** (1983), no. 3, 601.

[28] J. Binney, N. Dowrick, A. Fisher and M. Newman, *The Theory of Critical Phenomena*. Oxford University Press, 1992.

[29] K. Appel and W. Haken, *Every planar map is four colorable. Part I. Discharging*, *Illinois J. Math.* **21** (1977) 429–490.

[30] K. Appel, W. Haken and J. Koch, *Every planar map is four colorable. Part II. Reducibility*, *Illinois J. Math.* **21** (1977) 491–567.

[31] W. McCune, *Solution of the Robbins Problem*, *Journal of Automated Reasoning* **19** (1997), no. 3, 263–276. See also http://www-unix.mcs.anl.gov/~mccune/papers/robbins/.

[32] D. B. Lenat, *The nature of heuristics*, *Artificial Intelligence* **19** (1982) 189–249.

[33] P. H. Winston, *Artificial Intelligence*. Addison-Wesley, Reading, Ma, 2nd ed., 1984.

[34] M. Ginsberg, *Essentials of Artificial Intelligence*. Morgan Kauffman, 1993.

[35] H. Wang, *Popular Lectures on Mathematical Logic*. Dover, New York, 1981.

[36] R. P. Feynman, *Feynman Lectures on Computation*. Addison-Wesley, Reading, Ma, 1996.

[37] A. Dewdney, *The Turing Omnibus*. Computer Science Press, Rockville, 1989.

[38] D. E. Knuth, *Sorting and Searching*, vol. 3 of *The Art of Computer Programming*. Addison-Wesley, Reading, Ma, 2 ed., 1998.

[39] Y. Rogozhin, *Small universal Turing machines*, *Theoretical Computer Science* **168** (1996) 215–240.

[40] M. L. Minsky, *Computation: Finite and Infinite Machines*. Prentice Hall, Englewood Cliffs, N.J., 1967.

[41] Y. Matiyasevich, *Hilbert's Tenth Problem*. The MIT Press, Cambridge, 1993. Excerpts available at www.informatik.uni-stuttgart.de/ifi/ti/personen/Matiyasevich/H10Pbook/index.html.

[42] Y. Matiyasevich, *Primes are nonnegative values of a polynomial in ten variables*, *Zapiski Sem. Leningrad Mat. Inst. Steklov* **68** (1977) 62–82. Translated in Journal of Soviet Mathematics **15** 33-44 (1981).

[43] E. Bernstein and U. Vazirani, *Quantum Complexity theory*, *SIAM J. on Computing* **26** (1997), no. 5, 1411–1473.

[44] S. A. Cook and D. G. Mitchell, *Finding Hard Instances of the Satisfiablility Problem*. Manuscript 1998.

[45] Robert Sedgwick, *Algorithms in C*. Addison-Wesley, 1989.

[46] A. Aho and J. E. H. J. D. Ullman, *Data structures and algorithms*. Addison-Wesley, 1983.

[47] C. H. Papadimitriou, *Computational Complexity*. Addison-Wesley, Reading, MA, 1994.

[48] D. Brelaz, *New Methods to Color Vertices of a Graph*, *Comm. ACM* **22** (1979) 251–256.

[49] M. David and H. Putnam, *A computing procedure for quantification theory*, *J. ACM* **7** (1960) 201.

[50] D. S. Johnson, C. R. Aragon, L. A. McGeoch and C. Schevon, *Optimization by Simulated Annealing: An Experimental Evaluation; part II, Graph Coloring and Number Partitioning*, *Operations Research* **39** (1991), no. 3, 378–406.

[51] J. Hertz, A. Krogh and R. G. Palmer, *Introduction to the theory of neural computation*. Addison-Wesley, 1991.

[52] J. Houdayer and O. C. Martin, *Renormalisation for Discrete Optimization*, *Phys. Rev. Lett.* **83** (1999) 1030.

[53] S. Boettcher and A. Percus, *Nature's Way of Optimizing*, *Artif. Intel.* **119** (2000) 275.

[54] P. Bak, C. Tang and K. Wiesenfeld, *Self-organized Criticality: An explanation of* $1/f$ *noise*, *Phys. Rev. Lett.* **59** (1987) 381.

[55] P. Bak, *How Nature Works*. Oxford University Press, 1997.

[56] H. J. Jensen, *Self-Organized Criticality*, vol. 10 of *Lecture Notes in Physics*. Cambridge University Press, 1998.

[57] S. Boettcher and A. G. Percus, *Optimization with Extremal Dynamics*, *Phys. Rev. Lett.* **86** (2001) 5211.

[58] S. A. Cook, *The complexity of theorem-proving procedures*, in *Proc. 3rd Ann. ACM Symp. on Theory of Computing*, pp. 151–158. ACM, New York, 1971.

[59] J. Hartmanis, *Gödel, von Neumann and the P = NP Problem*, *Bull. Europ. Assoc. Theoretical Computer Science* **38** (June, 1989) 101–107.

[60] J. C. Martin, *Introduction to languages and the theory of computation*. McGraw-Hill, New York, 2nd ed., 1997.

[61] M. R. Garey and D. S. Johnson, *Computers and Intractability. A guide to the Theory of NP-Completeness*. W H Freeman, New York, 1979.

[62] D. van Melkebeek, *Time-Space Lower Bounds for Satisfiability*, *Bull. Europ. Assoc. Theoretical Computer Science* (2001), no. 73, 57.

[63] B. Bollobás, *Extremal Graph Theory*. Academic Press, New York, 1978.

[64] P. Varga, *Minimax games, spin glasses and the polynomial-time hierarchy of complexity classes*, *Phys. Rev. E* **57** (1998), no. 6, 6487–92.

[65] L. Fortnow, *My Favorite Ten Complexity Theorems of the Past Decade*. ECCC Report 94-021.

[66] E. Post, *Finite combinatory processes-formulation, I, J. of Symbolical Logic* **1** (1936) 103–105.

[67] A. Church, *The Calculi of Lambda-Conversion*, vol. 6 of *Annals of Mathematics Studies*. Princeton University Press, 1941.

[68] P. W. Shor, *Algorithms for quantum computation: discrete logarithms and factoring*, in *Proc. 35th Symp. on Foundations of Computer Science*, pp. 124–134. 1994.

[69] D. S. Abrams and S. Lloyd, *Nonlinear quantum mechanics implies polynomial-time solution for NP-complete and #P problems*. eprint quant-ph/9801041.

[70] P. Gossett, *NP in BQP with Nonlinearity*. eprint quant-ph/9804025.

[71] P. Pudlák, *Complexity Theory and Genetics*. ECCC Report 94-013.

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

[73] P. Maymin, *Programming Complex Systems*. eprint quant-ph/9710035.

[74] Y. Fu and P. W. Anderson, *Applications of statistical mechanics to NP-complete problems in combinatorial optimisation*, *J. Phys. A* **19** (1986), no. 9, 1605–1620.

[75] S. Kirkpatrick and B. Selman, *Critical Behavior in the Satisfiability of Random Boolean Expression*, *Science* **264** (May, 1994) 1297–1301.

[76] P. Cheeseman, B. Kanefsky and W. M. Taylor, *Where the Really Hard Problems Are*, in *Proceedings of IJCAI-91*, pp. 331–337. Morgan Kauffman, San Mateo, CA, 1991.

[77] T. Hogg, B. A. Huberman and C. P. Williams, *Editorial: Phase transitions and the search problem*, *Artificial Intelligence* **81** (1996) 1–15.

[78] W. G. Macready, A. G. Siapas and S. A. Kauffman, *Criticality and Parallelism in Combinatorial Optimization*. SFI working paper 95-06-054.

[79] E. Friedgut, *Sharp Thresholds of Graph Properties, and the k-SAT Problem*. To appear in J. Am. Math. Soc.

[80] D. Achlioptas and E. Friedgut, *A Sharp Threshold for k-Colorability*, *Random Structures and Algorithms* **14** (1999) 63.

[81] D. Achlioptas, L. M. Kirousis, E. Kranakis, D. Krizanc, M. S. O. Molloy and Y. C. Stamatiou, *Random Constraint Satisfaction: A More Accurate Picture*, in *Proceedings of Third International Conference on Principles and Practice of Constraint Programming*, vol. 1330, pp. 107–120. Springer, Berlin, 1997.

[82] T. Hogg, *Applications of Statistical Mechanics to Combinatorial Search Problems*, in *Annual Reviews of Computational Physics*, vol. 2, pp. 357–406. World Scientific, Singapore, 1995.

[83] C. P. Williams and T. Hogg, *Exploiting the Deep Structure of Constraint Problems*, *Artificial Intelligence* **70** (1994) 73–117.

[84] C. P. Williams and T. Hogg, *Extending Deep Structure*, in *Proc. of AAAI93*, pp. 152–157. AAAI Press, Menlo Park, CA, 1993.

[85] D. L. Mammen and T. Hogg, *A New Look at the Easy-Hard-Easy Pattern of Combinatorial Search Difficulty*, *J. of Artificial Intelligence Research* **7** (1997) 47–66.

[86] D. Achlioptas and C. Moore. Manuscript, 2001.

[87] L. M. Kirousis, E. Kranakis and D. Krizanc, *A better upper bound for the unsatisfiability threshold*.

[88] D. Achlioptas, *Threshold Phenomena in Random Graph Colouring and Satisfiability*. PhD thesis, University of Toronto, 1999. Available at http://www.research.microsoft.com/∼optas/.

[89] D. Achlioptas, P. Beame and M. Molloy, *A Sharp Threshold in Proof Complexity*. Manuscript 2001.

[90] D. Achlioptas, L. M. Kirousis, E. Kranakis and D. Krizanc, *Rigorous Results for Random* $(2+p)$*-SAT*, *Theoretical Computer Science* **265** (2001), no. 1-2, 109.

[91] R. Monasson, *Some remarks on hierarchical replica symmetry breaking in finite-connectivity systems*, *Phil. Mag. B* **77** (1998), no. 5, 1515–1521.

[92] R. Monasson, *Optimisation problems and replica symmetry breaking in finite connectivity spin-glasses*, *J. Phys. A* **31** (1998), no. 2, 513–29.

[93] R. Monasson and R. Zecchina, *Tricritical Points in Random Combinatorics: the (2+p)-SAT case*. eprint cond-mat/9810008. To appear in J Phys A.

[94] R. Monasson and R. Zecchina, *Statistical Mechanics of the Random K-satisfiability Model*, *Phys. Rev. E* **56** (1997), no. 2, 1357–1370.

[95] R. Monasson and R. Zecchina, *Entropy of the K-Satisfiability Problem*, *Phys. Rev. Lett.* **76** (1996), no. 21, 3881–3885.

[96] A. Crisanti, L. Leuzzi and G. Parisi, *The 3-SAT problem with large number of clauses in the ∞-replica symmetry breaking scheme*. eprint cond-mat/0108433.

[97] M. Leone, F. Ricci-Tersenghi and R. Zecchina, *Phase coexistence and finite-size scaling in random combinatorial problems*. eprint cond-mat/0103200.

[98] W. Barthel, A. Hartmann, M. Leone, F. Ricci-Tersenghi, M. Weigt and R. Zecchina, *Generating hard and solvable satisfiability problems: A statistical mechanics approach*. eprint cond-mat/0111153.

[99] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman and L. Troyansky, *2+p-SAT: Relation of Typical-Case Complexity to the Nature of the Phase Transition*. eprint cond-mat/9910080. To appear in Random Structures and Algorithms.

[100] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman and L. Troyansky, *Computational complexity from 'characteristic' phase transition*, *Nature* **400** (1999) 133.

[101] D. Achlioptas, A. Chtcherba, G. Istrate and C. Moore, *The Phase Transition in NAESAT and 1-in-k SAT*, in *Symposium on Discrete Algorithms*, p. 721. 2001.

[102] T. Walsh, *The Constrainedness Knife-Edge*. Available at http://www.cs.strath.ac.uk/~apes/apepapers.html.

[103] S. Cocco and R. Monasson, *Trajectories in phase diagrams, growth processes and computational complexity: how search algorithms solve the 3-Satisfiability problem*. eprint cond-mat/0009410.

[104] S. Cocco and R. Monasson, *Analysis of the computational complexity of solving random satisfiability problems using branch and bound search algorithms*. eprint cond-mat/0012191.

[105] G. Biroli and R. M. M. Weigt, *A variational description of the ground state structure in random satisfiability problems*, *Europ. J. Phys. B* **14** (2000) 551.

[106] A. Ben-Hur, J. Feinberg, S. Fishman and H. T. Siegelmann, *Probabilistic analysis of the phase space flow for linear programming*. eprint cond-mat/0110655.

[107] S. N. Majumdar and P. L. Krapivsky, *Extreme Value Statistics and Traveling Fronts: An Application to Computer Science*. eprint cond-mat/0109313.

[108] S. Franz, M. Leone, F. Ricci-Tersenghi and R. Zecchina, *Exact solutions for diluted spin glasses and optimization problems*. eprint cond-mat/0103328.

[109] B. Hayes, *Can't Get No Satisfaction*, *American Scientist* **85** (1998), no. 2, 108–112.

[110] T. Pitassi, *The Satisfiability Problem*. 1998 SFI Complex Systems Summer School Lecture Notes.

[111] O. C. Martin, R. Monasson and R. Zecchina, *Statistical mechanics methods and phase transitions in optimization problems*. eprint cond-mat/0104428.

[112] I. P. Gent and T. Walsh, *The TSP phase transition*, *Artificial Intelligence* **88** (1996) 349–358.

[113] A. Percus, *The Traveling Salesman and Related Stochastic Problems*. PhD thesis, Université Paris 6, 1997.

[114] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller *J. Chem. Phys* **21** (1953) 1087.

[115] R. J. Glauber, *Time-Dependent Statistics of the Ising Model, J. Math. Phys* **4** (1963) 294.

[116] K. Kawasaki in *Phase Transitions and Critical Phenomena*, C. Domb and M. S. Green, vol. 2. Academic Press, London, 1972.

[117] K. Kawasaki, *Diffusion Constants near the Critical Point for Time-Dependent Ising Models. I*, *Phys. Rev.* **145** (1965) 224.

[118] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, *Optimization by Simulated Annealing*, *Science* **220** (1983) 671–680.

[119] P. C. Hohenberg and B. I. Halperin, *Theory of dynamic critical phenomena*, *Rev. Mod. Phys.* **49** (1977), no. 3, 435.

[120] A. Sadiq and K. Binder, *Dynamics of Formation of Two-Dimensional Ordered Structures*, *J. Stat. Phys* **35** (1984), no. 5-6, 517–585.

[121] J. S. Langer, *An introduction to the kinetics of first-order phase transition*, in *Solids far from equilibrium*, C. Godrèche, pp. 297–364. Cambridge University Press, 1992.

[122] P. Chaikin and T. C. Lubensky, *Principles of condensed matter physics*. Cambridge University Press, 1995.

[123] Z. W. Lai, G. F. Mazenko and O. T. Valls, *Classes for growth kinetics problems at low temperatures*, *Phys. Rev. B* **37** (1988), no. 16, 9481–9494.

[124] J. D. Shore, M. Holzer and J. P. Sethna, *Logarithmically slow domain growth in nonrandomly frustrated systems: Ising models with competing interactions*, *Phys. Rev. B* **46** (1992), no. 18, 11376–11404.

[125] G. Brown and P. A. Rikvold, *Numerical Confirmation of Late-time $t^{1/2}$ growth in Three-dimensional Phase Ordering*. eprint cond-mat/0107233.

[126] S. A. Kauffman, *Metabolic stability and epigenisis in randomly constructed genetic nets*, *J. Theor Biol* **22** (1969) 437.

[127] H. E. Stanley, D. Stauffer, J. Kertész and H. J. Herrmann, *Dynamics of Spreading Phenomena in Two-Dimensional Ising Models*, *Phys. Rev. Lett.* **59** (1987) 2326.

[128] T. Vojta, *In an Ising model with spin-exchange dynamics damage always spreads*, *J. Phys. A* **31** (1998) 6595.

[129] T. Vojta and M. Schreiber, *An unexpected difference between regular and random order of updates in damage spreading simulations*, *Phys. Rev. E* **58** (1998) 7998.

[130] T. Vojta and M. Schreiber, *Differences between regular and random order of updates in damage-spreading simulations*, *Phys. Rev. E* **58** (1998) 7998.

[131] P. Grassberger, *Damage spreading in the Ising model with Glauber dynamics*, *J. Phys. A* **28** (1995) L67.

[132] P. Grassberger, *Damage spreading and critical exponents for "model A" Ising dynamics*, *Physica A* **214** (1995) 547.

[133] D. E. Knuth, *Seminumerical Algorithms*, vol. 2 of *The Art of Computer Programming*. Addison-Wesley, Reading, Ma, 3 ed., 1997.

[134] S. K. Park and K. W. Miller, *Random number generators: Good ones are hard to find*, *Comm. ACM* **31** (1988), no. 10, 1192.

[135] D. Wood, *Random, or just confused?*, *Transactor for the Amiga* **1** (1988), no. 2, 38.

[136] B. Hayes, *Randomness as a Resource*, *American Scientist* (July-August, 2001).

[137] I. P. Gent, S. A. Grant, E. MacIntyre, P. Prosser, P. Shaw, B. M. Smith and T. Walsh, *How Not To Do It*. APES eprint.

[138] T. C. Belding, *Numerical Replicationof Computer Simulations: Some Pitfalls and How to Avoid Them*. eprint nlin.AO/0001057.

[139] D. G. Mitchell and H. J. Levesque, *Some pitfalls for experimenters with random SAT*, *Artificial Intelligence* **81** (1996) 111–125.

[140] T. Garel, H. Orland and E. Pitard, *Protein folding and heteropolymers*, in *Spin Glasses and Random Fields*, A. P. Young, p. 387. World Scientific, 1998.

[141] S. S. Rao and S. M. Bhattacharjee, *Protein folding and spin glass*. eprint cond-mat/9503173.

[142] P. Crescenzi, D. Goldman, C. H. Papadimitriou, A. Piccolboni and M. Yannakakis, *On the complexity of protein folding*, *Journal of Computational Biology* **5** (1998), no. 3, 423–466.

[143] C. Chothia, *One thousand families for the molecular biologist*, *Nature* **357** (1992) 543.

[144] R. Mélin, H. Li, N. S. Wingreen and C. Tang, *Designability, thermodynamic stability, and dynamics in protein folding: a lattice model study*, *J. Chem. Phys.* **110** (1999), no. 2, 1252–1262.

[145] H. Li, C. Tang and N. S. Wingreen, *Are Protein Folds Atypical*, *Proc. Natl. Acad. Scie. USA* **95** (1998) 4987–4990.

[146] A. D. Sokal, *Monte Carlo Methods for the Self-Avoiding Walk*. eprint hep-lat/9509032.

[147] A. D. Sokal, *Monte Carlo Methods for the Self-Avoiding Walk*. eprint hep-lat/9405016.

[148] B. Bollobás, *Random Graphs*. Academic Press, New York, 1985.

[149] B. Bollobás, *Graph Theory: An Introductory Course*. Springer-Verlag, New York, 1979.

[150] C. F. Baillie, D. A. Johnston and J. P. Kownacki, *Ising spins on thin graphs*, *Nucl. Phys. B* **432** (1994) 551–570.

[151] C. F. Baillie and D. A. Johnston, *Spin Models on Thin Graphs*, *Nucl. Phys. Proc. Suppl.* **47** (1996) 649.

[152] C. F. Baillie, W. Janke, D. A. Johnston and P. Plecháč, *Spin glasses on thin graphs*, *Nucl. Phys. B* **450 (FS)** (1995) 730–752.

[153] D. A. Johnston and P. Plecháč, *Vertex Models on Feynman Diagrams*, *Phys. Lett. A* **248** (1998) 37.

[154] D. A. Johnston and P. Plecháč, *Percolation on a Feynman Diagram*. eprint cond-mat/9705101.

[155] D. A. Johnston and P. Plecháč, *Potts Models on Feynman Diagrams*, *J. Phys. A* **30** (1997) 7349–7363.

[156] D. A. Johnston and P. Plecháč, *Why Loops Don't Matter*, *J. Phys. A* **31** (1998) 475.

[157] A. Barrat and R. Zecchina, *Time scale separation and heterogeneous off-equilibrium dynamics in spin models over random graphs*, *Phys. Rev. E* **59** (1999) R1299.

[158] S. Milgram, *The small world problem*, *Psychology Today* **2** (1967) 60.

[159] C. Korte and S. Milgram, *Acquaintance linking between white and negro populations: Application of the small world problem*, *J. Personality and Social Psychology* **15** (1970) 101.

[160] P. Hoffman, *The Man Who Loved Only Numbers*. Hyperion, New York, 1998.

[161] V. Latora and M. Marchiori, *Efficient Behaviour of Small-World Networks*, *Phys. Rev. Lett.* **87** (2001) 198701.

[162] R. Albert and A.-L. Barabási, *Statistical Mechanics of Complex Networks*. eprint cond-mat/0106096.

[163] M. E. J. Newman, S. H. Strogatz and D. J. Watts, *Random graphs with arbitrary degree distribution and their applications*, *Phys. Rev. E* **64** (2001) 026118.

[164] A.-L. Barabási and E. Ravasz, *Deterministic Scale-Free Networks*. eprint cond-mat/0107419.

[165] M. E. J. Newman, *Ego-centered networks and the ripple effect*. eprint cond-mat/0111070.

[166] G. Bianconi and A.-L. Barabási, *Bose-Einstein Condensation in Complex Networks, Phys. Rev. Lett.* **86** (2001), no. 24, 5632.

[167] M. E. J. Newman, *Models of the Small World: A Review, J. Stat. Phys* **101** (2000) 819. e-print cond-mat/0001118.

[168] D. J. Watts, *Small Worlds*. Princeton University Press, 1999.

[169] S.Jespersen, I. M. Sokolov and A. Blumen, *Small-world Rouse networks as models of cross-linked polymers, J, Chem. Phys* **113** (2000), no. 17, 7652.

[170] M. Steyvers and J. B. Tenenbaum, *The Large-Scale Structure of Semantic Networks: Statistical Analyses and a Model of Semantic Growth*. Submitted to Cognitive Science.

[171] B. Hayes, *The Web of Words, American Scientist* (March-April, 1999).

[172] R. Ferrer i Cancho, C. Janssen and R. V. Solé, *The Topology of Technology Graphs: Small World Patterns in Electronic Circuits*. Manuscript 2001.

[173] S. Redner, *How Popular is Your Paper? An Empirical Study of the Citation Distribution, Europ. J. Phys. B* **4** (1998) 131.

[174] B. Hayes, *Graph Theory in Practice, I and II, American Scientist* (January-February and March-April, 2000).

[175] R. A. H. Jeong and A.-L. Barabasi, *Internet - Diameter of the World-Wide Web, Nature* **401** (1999) 130.

[176] F. Liljeros, C. R. Edling, L. A. N. Amaral, H. E. Stanley and Y. Åberg, *The web of human sexual contacts, Nature* **411** (2001) 907.

[177] M. E. J. Newman and D. J. Watts, *Renormalization group analysis of the small-world network model, Phys. Lett. A* **263** (1999) 341.

[178] M. Barthélémy and L. A. N. Amaral, *Small-World Networks: Evidence for a Crossover Picture, Phys. Rev. Lett.* **82** (1999), no. 15, 3180–3183.

[179] M. E. J. Newman and D. J. Watts, *Scaling and percolation in the small-world network model, Phys. Rev. E* **60** (1999) 7332.

[180] E. Almass, R. V. Kulkarni and D. Stroud, *Characterizing the structure of small-world networks*. eprint cond-mat/0109227.

[181] E. J. Farkas, I. Derényi, A.-L. Barabási and T. Vicsek, *Spectra of "Real-World Graphs: Beyond the Semi-Circle Law*, *Phys. Rev. E* **64** (2001) 026704.

[182] R. Monasson, *Diffusion, localization and dispersion relations on "small-world" lattices*. eprint cond-mat/9903347.

[183] M. A. de Menezes, C. F. Moukarzel and T. . P. Penna, *First-order transition in small-world networks*, *Europhys. Lett.* **50** (2000) 5.

[184] P. Holme and B. J. Kim, *Growing Scale-Free Networks with Tunable Clustering*. eprint cond-mat/0110452.

[185] D. Watts and S. Strogatz, *Collective dynamics of 'small world' networks*, *Nature* **393** (1998) 440–2.

[186] L. F. Lago-Fernández, R. Huerta, F. Corbacho and J. A. Sigüenza, *Fast response and temporal coherent oscillations in small-world networks*, *Phys. Rev. Lett.* **84** (2000) 2758.

[187] M. E. J. Newman and C. Moore, *Epidemics and percolation in small-world networks*, *Phys. Rev. E* **61** (2000) 5678.

[188] M. E. J. Newman and C. Moore, *Exact solution of site and bond percolation on small-world networks*, *Phys. Rev. E* **62** (2000) 7059.

[189] R. Pastor-Satorras and A. Vespignani, *Epidemic Spreading in Scale-Free Networks*, *Phys. Rev. Lett.* **86** (2001) 3200.

[190] Z. Denzsö and A.-L. Barabási, *Can we stop the AIDS epidemic?*. eprint cond-mat/0107420.

[191] R. V. Kulkarni, E. Almaas and D. Stroud, *Evolutionary dynamics in the Bak-Sneppen model on small-world networks*. eprint cond-mat/9905066.

[192] F. Bagnoli and M. Bezzi, *Small world effect in evolution*. eprint cond-mat/0007458.

[193] A. Barrat and M. Weigt, *On the properties of small-world network models*, *Europ. J. Phys. B* **13** (2000) 527.

[194] M. Gitterman, *Small-world phenomena in physics: the Ising model*, *J. Phys. A* **33** (2000), no. 47, 8373–8381.

[195] A. Pekalski, *Ising model on a small world network*, *Phys. Rev. E* **64** (2001) 057104.

[196] B. J. Kim, P. Holme, G. S. Jeon, P. Minnhagen and M. Y. Choi, *XY model in small-world networks*. eprint cond-mat/0108392.

[197] P. Svenson. unpublished results.

[198] V. Spirin, P. L. Krapivsky and S. Redner, *Freezing in Ising Ferromagnets*. eprint cond-mat/0105037. Phys. Rev. E, in press.

[199] J. Berg and M. Sellitto, *Metastable configurations of spin models on random graphs*. eprint cond-mat/0108427.

[200] J. Frank, P. Cheeseman and J. Stutz, *When Gravity Fails: Local Search Topology, J. of Artificial Intelligence Research* **7** (1997) 249–281.

[201] R. Moessner and J. T. Chalker, *Low-temperature properties of classical, geometrically frustrated antiferromagnets*. eprint cond-mat/9807384.

[202] E. Marinari, O. C. Martin and F. Zuliani, *Equilibrium valleys in spin glasses at low temperature*. eprint cond-mat/0103534.

[203] J. Lamarcq, J.-P. Bouchaud, O. C. Martin and M. Mézard, *Non-compact local excitation in spin glasses*. eprint cond-mat/0107544.

[204] J. Houdayer and O. C. Martin, *A geometrical picture for finite dimensional spin glasses, Europhys. Lett.* **49** (2000) 794.

[205] J. Houdayer and O. C. Martin, *Droplet Phenomenology and Mean Field in a Frustrated Disordered System, Phys. Rev. Lett.* **81** (1998) 2554.

[206] I. Bieche, R. Maynard, R. Rammal and J. P. Uhry *J. Phys. A* **15** (1982) 673.

Paper I

Paper II

Paper III

Paper IV

“October knew, of course, that the action of turning a page, of ending a chapter or of shutting a book, did not end a tale. Having admitted that, he would also avow that happy endings were never difficult to find: ”It is simply a matter,” he explained to April, ”of finding a sunny place in a garden, where the light is golden and the grass is soft, somewhere to rest, to stop reading, and to be content.”

— G.K.Chesterton, “The Man Who Was October” (Library of Dream edition)