STATISTICAL PHYSICS METHODS IN COMPUTATIONAL BIOLOGY

by

Osvaldo Zagordi
SISSA-ISAS

Submitted in partial fulfillment of the requirements
for the degree of PhilosophiæDoctor
Sector of Statistical and Biological Physics
SISSA-ISAS
July 2007

External supervisors
Dr. Michele Leone
Prof. Riccardo Zecchina

Local supervisor
Prof. Cristian Micheletti

Contents

Introduction

Computational biology and bioinformatics[1] provide theoretical tools to analyze biological systems. This is done by means of new theories, that also give new ways to interpret the biological knowledge, and algorithms, that allow to discover information previously hidden. Common examples are given by the thousands and thousands of sequence alignments daily performed by biologists, often without even knowing how the algorithm behind works. Further, it is customary to compute the probability that a given sequence hosts a gene or not. Sequencing genomes wouldn't have been possible without the development of new computational tools capable of assembling DNA sequence fragments, and comparing the organisms from an evolutionary point of view would still rely on observing phenotypic attributes without evolutionary models.

Statistical physics offers an alternative way to look at questions coming from biology, as it asks how order and correlation emerge in environment where noise and disorder play a fundamental role. It suffices to think at how many concepts from equilibrium and out of equilibrium statistical physics concepts are currently used to derive coarse grained models for biological systems, from molecular to cellular and ecological level.

It should also be said physicists often believe that physics can be useful to biologists especially because of the huge amount of data that recent experiments are now producing. Actually, biologists often address themselves to physics because of the scarcity of good quality data they have. At least one of the topics reported here goes in this direction.

Statistical physics has also recently provided many hints to computer scientists about combinatorial problems, both from analytic and algorithmic point of view. Thanks to techniques originally developed for disordered systems, some light has been shed on the mechanisms underlying the ap-

[1] A NIH committee set the "official" definition for these two terms in 2000, recognizing that, though distinct, a significant overlap of activities exist at their interface.

pearance of phase transitions in the difficulty of random combinatorial problems.

Whether it is possible to exploit this new achievements in biological research is currently under investigation. After a general and obviously non exhaustive introduction to the general topics of optimization and statistical physics, we present three bioinformatics problems as possible applications of these new tools.

Chapter 1

Physics, Optimization and Statistical Inference

This introductory chapter is devoted to present at a general level the interconnection between statistical physics and optimization. This is also developed through the introduction of graphical models as factor graphs.

1.1 Physics and Constraint Satisfaction Problems

1.1.1 Systems of interacting spins

An ubiquitous goal in several fields in physics is trying to minimize an energy. Statistical physics doesn't make an exception to the rule, as we see in the next few examples.

The famous Ising problem is introduced in physics as a model for magnetic materials, where the energy is defined over a set of magnetic interacting particles. Let us consider N spin variables $\sigma \in \{-1,-1\}$, interacting via the Hamiltonian

$$\mathcal{H} = -\sum_{\langle ij \rangle} \sigma_i \sigma_j - H \sum_i \sigma_i \quad . \tag{1.1}$$

H is an external magnetic field and the $\langle\ \rangle$ symbol restrains the sum over all possible couples of

interacting nearest neighbours. If we consider a linear 1D system, then the interaction will take place only between a spin and those immediately following and preceding it. In a 2D square lattice a spin will interact with four neighbours, in a 3D cubic lattice with six and so on and so forth. Of course more complicated lattices can be introduced, but, as long as the hamiltonian remains of the form in 1.1, the lowest energy will always be found when all spins are aligned to the external magnetic field.

Of course there are cases where finding the lowest energy state (the ground state *GS*), is not as easy as in the previous example. The very next step in an ideal difficulty scale is the Ising spin glass, defined by

$$\mathcal{H} = -\sum_{\langle ij \rangle} J_{ij}\sigma_i\sigma_j - H\sum_i \sigma_i \quad , \tag{1.2}$$

where J_{ij} are random variables sampled from a distribution to be specified. Again, choosing which spins interact with each other amounts to select a particular lattice. In the extreme opposite case, if the interaction takes place among all possible couples, the system is *fully connected*. Here a caveat must be made: while the σ are random variables that undergo thermal equilibrium, while J_{ij} are assigned and do not thermalize. They are called *quenched* variables.

We want to stress here, as soon as the coupling constant J, depending on the variables pair, can take both positive and negative values we are mixing ferromagnetic and anti-ferromagnetic couplings. It is immediately seen that finding the GS is not as trivial as before, due to the presence of competitive interactions. Let's illustrate this with the example of figure 1.1. In the first figure the

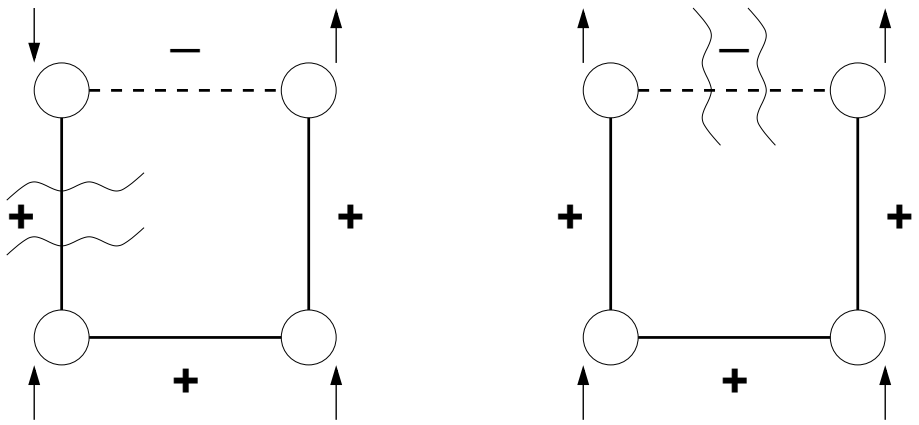

Figure 1.1: Typical example of frustration on a plaquette for an Ising sping glass on a lattice.

two left spins contribute positively to the energy, because they point in opposite directions but are coupled by a positive J. If the upper left spin is flipped, then the upper right one should be flipped too due to the anti-ferromagnetic interaction. But then another unfavoured interaction would emerge and so on. Such a system, where it is impossible to satisfy simultaneously all constraints because they are conflicting, is called *frustrated*.

1.2 Relation between Statistical Physics and Optimization

It is well known that the mean energy is computed with the Boltzmann distribution:

$$\langle E \rangle = \frac{1}{Z} \sum_{\{\vec{\sigma}\}} \mathcal{H}(\vec{\sigma}) e^{-\beta \mathcal{H}(\vec{\sigma})} \quad , \tag{1.3}$$

where the the sum runs over possible configurations $\{\vec{\sigma}\}$ of the spins σ_i, β is the inverse of the temperature T and Z is the partition function, entering here as a normalizing factor. As $\beta \to 0$ the distribution over the configuration is flat. As β increases, lower energy configurations are weighted more and more, and as $\beta \to \infty$ only the GS survives, as the distribution is concentrated on the lowest energy states, where the maximum possible number of constraints are satisfied.

In the case of the spin glass, it turns out that the mean energy, being a function of the J_{ij}, is itself a random variable. In chapter 2, several techniques to compute the average quantities on the quenched variables will be shown.

It is very easy to define a Hamiltonian which counts the number of satisfied constraints:

$$\mathcal{H} = \sum_a (1 - \theta(J_a \sigma_{a_1} \sigma_{a_2})) \tag{1.4}$$

where a is the couple of spins that interact, as $\langle ij \rangle$ in the previous case.

By performing the same limit $\beta \to \infty$ the GS is selected, and we have the maximum number of constraints that can be simultaneously satisfied.

1.2.1 Constraint satisfaction problems (CSP)

A constraint satisfaction problems (CSP) is defined by a set of variables $(X_1, \ldots, X_N)$ and a set of constraints $(C_1, \ldots, C_M)$. Every variable has a domain D_i of possible values v_i, and every constraint

involves a subset of the variables $C_m = C(X_{m_1}, \ldots, X_{m_k})$. An assignment of the variables (i.e. specifying values for all variables) is a solution when it satisfies all the constraints.

Random XOR-SAT

A well known example of a CSP is the p-XORSAT problem in computer science, equivalent in physics to the p-spin model on random graph at zero temperature.

Given N boolean variables $x_i \in \{0,1\}$, and $M = \gamma N$ parity checks constraints, the random p-XORSAT problem consists in finding an assignment such that all the constraints are satisfied. These constraints are also called *clauses* or *function nodes*, as we will see later on. A parity check is defined as follows

$$x_{i_1^m} + \ldots + x_{i_p^m} = y_m \mod 2, \qquad m = 1, \ldots, M \tag{1.5}$$

where, for each m, the p indices $i_1^m, \ldots, i_p^m \in \{1, \ldots, N\}$ denoting the variables involved in m are chosen randomly with flat distribution among the $\binom{N}{p}$ possible set of p distinct variables, and the values y_m takes value 0 or 1 with equal probability. In matrix notation, this be written as $\hat{A}\vec{x} = \vec{y} \mod 2$, where $\hat{A}$ is a $M \times N$ random sparse matrix with exactly p ones per row and y is a random vector of 0s and 1s.

Mapping this problem into a spin Hamiltonian is easily done via the set of transformations

$$\sigma = (-1)^x \tag{1.6}$$

$$J = (-1)^y \quad . \tag{1.7}$$

The Hamiltonian reading

$$\mathcal{H} = \sum_{m=1}^{M} \left(1 - J_m \sigma_{i_1^m} \cdots \sigma_{i_p^m}\right) \quad , \tag{1.8}$$

the XORSAT problem is equivalent to its zero-temperature limit. If a solutions to the CSP exists, it will correspond to a zero energy GS for the Hamiltonan.

This model has been extensively studied, and it has been shown to present two relevant phase transitions, depending on the ratio γ (number of constraints over number of variables), at least concerning the number of solutions and their spatial organiztion [1]. Let us consider the possible assignment of the N variables as the 2^N vertices of the N dimensional hypercube. At an intuitive level, if no

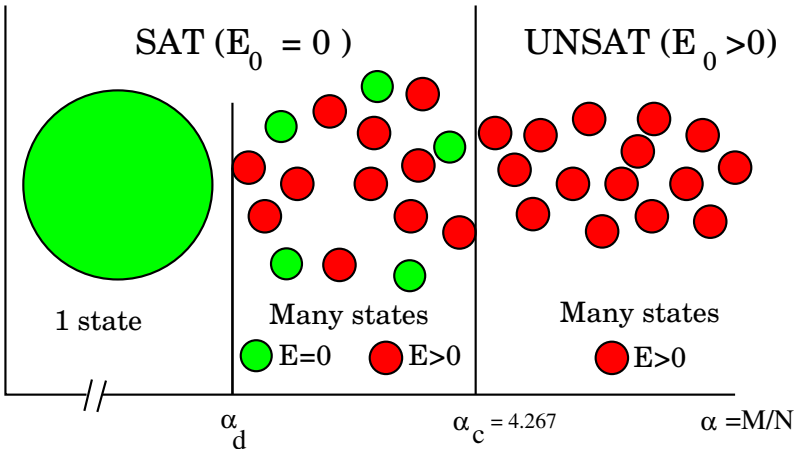

Figure 1.2: Pictorial representation of the clustering phenomenon. As the ratio α constraint over variables increases, one encounters two phase transitions. See text.

constraints exist, then all vertices will be a solution of the problem. Adding a constraint determines that 2^{N-1} vertices are not a solution anymore (the number of solutions is halved at each constraint added). So, adding new constraints, more and more vertices of the hypercube are eliminated, until all vertices are forbidden. If the constraints were all independent, then it would take $M = N$ constraints to eliminate all solutions. But as this is not the case, the SAT/UNSAT transition (the critical value γ_c such that no solutions are found for $\gamma > \gamma_c$), takes place at $\gamma_c = 0.917935$ for $p = 3$. Above this threshold, the system is UNSAT, i.e. no solutions exist, or, equivalently, its GS energy is greater than zero. Another interesting phase transition manifests itself at a lower value of γ (specifically at $\gamma_d = 0.818469$ for $p = 3$) and is related to the spatial organization of the solutions in the hypercube. Roughly speaking, at values $\gamma < \gamma_d$, all solutions are organized in a single cluster, while for $\gamma > \gamma_d$, a non-trivial structure of the solution space starts to emerge. In this region $\gamma_d < \gamma < \gamma_c$, GS exist, and are organized in an exponential number of different clusters made of an exponential number of solutions each, with a typical overlap between clusters. Moreover, as this clusters are separated by barriers (otherwise talking about clusters wouldn't make much sense) and metastable states emerge, from a dynamical point of view the solutions become almost inaccessible by any local search algorithm (this is why it is called dynamical transition). This is why it is also called *hard phase*. A pictorial representation of such phenomenon is reported in figure 1.2.

Random K-SAT

Another CSP we want to introduce here is the K satisfiability (or *K*sat). As in the previous example, it consists of N boolean variables and $M = \alpha\, N$ constraints or clauses. These clauses have the form of a Boolean OR function of K variables, which can appear negated or not. One searches for a configuration that satisfies all the constraints. In an equivalent manner we can say that a logical conjunction (AND) is present among all clauses and the entire proposition has to be satisfied. It is worth noting here that the *K*sat problem for $K \geq 3$ was the first problem shown to be NP-complete [2].

In Boolean form we have

$$\Phi(x_1,\ldots,x_N) = \bigwedge_{m=1}^{M} (\ell_{m_1} \vee \ldots \vee \ell_{m_K}) \quad , \tag{1.9}$$

where $\ell_{m_1} \ldots \ell_{m_K}$ are the K literals appearing in clause i, i.e. the variables considered with their eventual negation.

Making the transformation into a spin system, we write the energy of a clause m involving spins $\sigma_{m_1},...,\sigma_{m_K}$ as:

$$E_m = \prod_{r=1}^{K} \frac{(1 + J_m^r\, \sigma_{i_r})}{2} \quad . \tag{1.10}$$

The K coupling constants $\mathbf{J_m} = (J_m^1,\ldots,J_m^K)$ take values ± 1. The interpretation in terms of Boolean functions is the following: the energy of the clause is zero if at least one of the spins σ_{i_r} is opposite to the corresponding coupling J_m^r. If all spins are equal to their couplings, the energy is equal to 1.

Here the mapping is that x_i is TRUE when $\sigma_i = 1$. The energy E_m depends on the OR of the K variables $l\ell_{m_1},\ldots,\ell_{m_K}$, where ℓ_{m_r} is the original x_{m_r} when $J_m^r = -1$ and is its negation when $J_m^r = +1$. When $\ell_{m_1} \vee \ldots \vee \ell_{m_K}$ is true (the clause is satisfied), the enrgy is zero. If otherwise it is unsatisfied, then the energy is $E_m = 1$. Let's just briefly mention the fact that in *K*sat only one of the 2^K assignment of the variables in a clause violates it.

What we have said for the XORSAT problem above finds a similar correspondance in the *K*sat case. As before, were the constraints all independent, for the case $K = 3$, a SAT/UNSAT transition would be found at $\alpha = \frac{\ln 2}{\ln \frac{8}{7}} \simeq 5.19$, but, as it is not the case, it actually takes place at a lower value. Variation bounds are reported for this and other systems in [3]. In fact, it results $\alpha_c = 4.267$, but a

very rich behaviour of the solutions space is found before this point. Let's just mention that a very HARD phase is found in the region $4.15 < \alpha < 4.267$, where the clustering of solutions is so sharp that it is impossible for a local search algorithm to find a solution of large instances of the problem. Statistical physics methods have been extensively applied to *K*sat [4, 5, 6, 7, 8, 9, 10] giving a deep comprehension of the phenomenology of this interesting problem.

Validity of the former thresholds

In the following, several techniques that have been used to derive these results will be presented. It must be said that these techniques are not always rigorous in a strictly mathematical sense. Nevertheless one of the most interesting aspects of this field is that, sitting at the crossroad of computer science and statistical physics, it has given the opportunity to both communities to share their results with the other. So far, the rigorous results found by mathematicians are in agreement with those previously obtained by physicists [11]. In particular for random CSP, such as random k-SAT and random graph coloring, excellent rigorous estimates of critical constraints over variables ratio exist.

We conclude this subsection citing two other famous CSPs, vertex-cover (VC) [12] and colouring [13].

1.3 Statistical Inference

The problems presented above, together with many others in computer science, share an important common feature: the locality of the dependencies between variables. Such feature is also present in many problems of probabilistic inference and sets the basis for a unified treatment of the topic. In the following we will explain the main concepts of statistical inference, and show how CSP and statistical physics problems can be casted in a way such that an inference treatment is possible.

1.3.1 Bayesian networks

A probabilistic graphical model (or Bayesian network) is briefly a set of nodes representing variables, which can assume a discrete number of states, together with the dependance relations among them. We will illustrate this concept with an example taken from the literature [14], also reported in [15]. In fig.1.3 we report a very simple Bayesian network which represents the following statements:

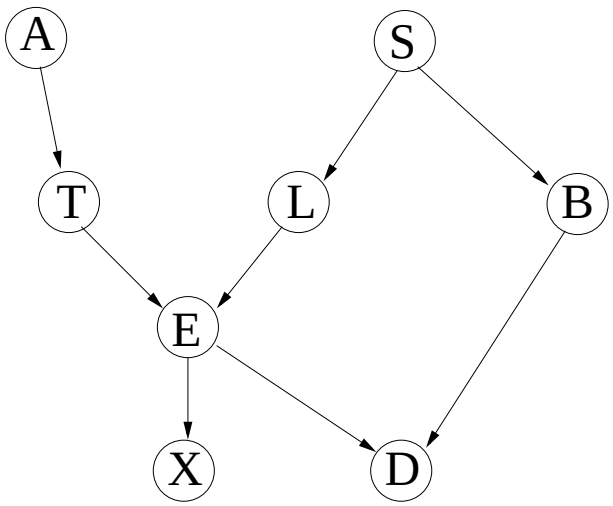

Figure 1.3: Asia example illustrating a simple Bayesian network.

- a recent trip to Asia increases the chances of tuberculosis (T),
- smoking is a risk factor for lung cancer (L) and bronchitis (B),
- either (E) tuberculosis or lung cancer can be detected by X-ray (X), but cannot be distinguished,
- dyspnoea(D) may be caused by bronchitis (B) or either tuberculosis or lung cancer.

Of course such a diagram is just an example, and can be adapted to many different contexts, see for example the pedagogical article [16], where a cell-signaling pathway is taken as a prototype for Bayesian networks.

More generally, a Bayesian network is an acyclic directed graph $G = (V,E)$, of variables V and edges E, where probability distributions for variables on the vertices of the graph are given. The graph is directed, meaning that for each edge an order exist between variables connected by it, and has no cycle (no direct paths from a variable to itself can be found). The probabilities for variables are given in terms of conditional probabilities in the form of $p(x_i|\mathrm{Par}(x_i))$; the probability of i to be in the state i, given the state of its parents $\mathrm{Par}(x_i)$. j is a parent of i if the edge (j,i) is in the graph. Note that the parents can be more than one.

So, in our *Asia* example, $p(x_\mathrm{B}|x_\mathrm{S})$ is the conditional probability of a patient having bronchitis given that he does or does not smoke, as $p(x_\mathrm{D}|x_\mathrm{E},x_\mathrm{B})$ is the conditional probability of a patient having dyspnoea given that he has either tuberculosis or lung cancer or bronchitis.

Finally, the joint probability distribution of all the variables, denoting in our example a combination of symtopms, test-results and diseases, is given by the product of all the conditional probabilities:

$$p(\{x\}) = \prod_{i=1}^{N} p(x_i|\mathrm{Par}(x_i)) \quad , \tag{1.11}$$

where $p(x_i|\mathrm{Par}(x_i)) = p(x_i)$ if no parents are present for variable i.

We are interested now in *marginal probabilities*, or *marginals*, i.e. the probability for a given variable to be in one of its possible states. This is nothing more than the sum of the joint probability over all the other variables, for example the marginals for the variable x_N is:

$$p(x_N) = \sum_{\substack{x_i \\ i \neq N}} p(x_i) \quad . \tag{1.12}$$

Another class of problem of this kind appears as follows: given a Bayesian network, both the graph and the probability distributions, and the observations on a subset of the variables $O \subseteq V$, what is the probability for the other variables to be in a given state? In other words, marginals $p(x_{V\setminus O}|x_O)$ must be computed.

Going back to our example, we may want to know the probability that a patient has tuberculosis, given that we observe dyspnoea. This amounts to calculate:

$$p(\mathrm{T}|\mathrm{D}=1) = \frac{\sum_{\mathrm{E,B,X,L}} p(\mathrm{E,B,X,L,D}=1)}{\sum_{\mathrm{E,B,X,T',L}} p(\mathrm{E,B,X,T',L,D}=1)} \tag{1.13}$$

As it can be easily understood from the formulas 1.12 and 1.13 the number of operations needed for the inference grows exponentially as the number of parent variables increases. Even with a dozen of binary variables, inference can be a painful process. We will see in the following how this hindrance can often be overcome.

1.3.2 Factor graphs

As it is clear from eq.(1.11), we are dealing with a joint probability which can be expressed as a product of several factors: a *factor graph* is a schematic way to express this factorization and will be needed in the following. In general, let's consider a set of variables $X = \{x_1, \ldots, x_N\}$, and a function

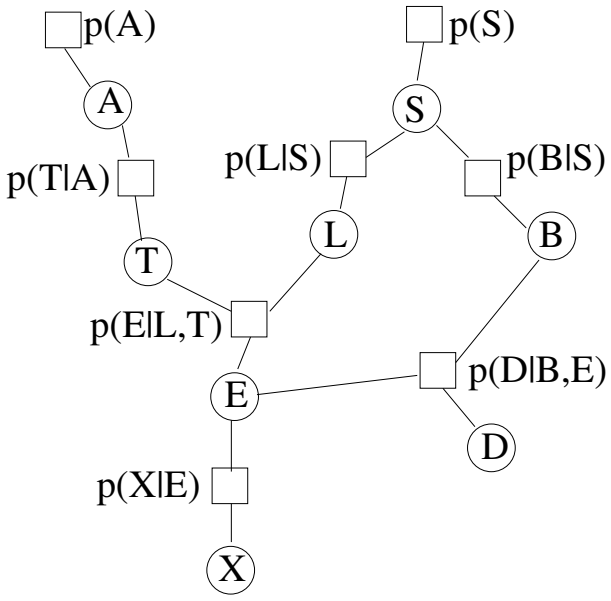

Figure 1.4: Factor graph for the Asia example

that can be expressed as a product of local functions

$$g(\{x_1,\ldots,x_N\}) = \prod f_i(x_j \in X_i) \quad , \tag{1.14}$$

with X_i different subsets of X. Then we can associate to this function a bipartite graph made of *variable nodes* for each variable x_i and *function nodes* for each function f_j such that x_i is connected to f_j if and only if x_i is argument of f_j. In fig.1.4 we depict the factor graph that represents the former example Asia. Note that every variable can be connected only to functions and functions to variables, i.e. the graph is *bipartite*.

While Bayesian networks are acyclic graphs, the factor graphs representation of the problems of interest for us can contain cycles. When this happens, treating it as a Bayesian network is an approximation that neglects the presence of cycles. We will see in the following that the algorithm that perform statistical inference in terms of local updates performs exactly this approximation.

1.3.3 Inference in statistical physics

We have seen how a practical statistical inference problem, as the individuation of the most probable disease given a set of symptoms, can be represented in a statistical framework as a factor graph. We will show here how a typical statistical physics question can be recast in a similar manner.

In statistical physics one is often interested in the computation of the probabilities, or *expected values*. For example the magnetization of a spin system is simply the probability of the spin to be in a certain state: $m(H,\beta) = \langle \sigma_i \rangle_{H,\beta}$. The possibility to access the behaviour of this quantity would give us information on the presence of phase transitions. Similarly, another central quantity is the spin-spin correlation $C_{ij}(H,\beta) = \langle \sigma_i, \sigma_j \rangle_{H,\beta}$, very similar to the quantities introduced in the previous discussion.

As stated above, the locality of the interaction among variables is a common trait of the systems we are dealing with. Let us see how this locality allows us to recast a spin system into a factor graph representation.

Factor graph representation of Ising model

As a simple example we can consider the Ising model in one dimension. The Hamiltonian 1.1 gives a probability distribution expressed by:

$$\begin{aligned} p(\vec{\sigma}) &= \frac{e^{-\beta \mathcal{H}(\vec{\sigma})}}{Z} && (1.15)\\ &= \frac{1}{Z} e^{\beta J \sum_{\langle ij \rangle} \sigma_i \sigma_j + \beta H \sum_i \sigma_i} && (1.16)\\ &= \frac{1}{Z} \prod_{\langle ij \rangle} e^{\beta J \sigma_i \sigma_j} \prod_i e^{\beta H \sigma_i} \quad . && (1.17) \end{aligned}$$

Defining $\beta J \sigma_i \sigma_j = \ln(\Psi(\sigma_i \sigma_j))$ and $\beta H \sigma_i = \ln(\Phi(\sigma_i))$ we can rewrite

$$p(\vec{\sigma}) = \frac{1}{Z} \prod_{\langle ij \rangle} \Psi(\sigma_i \sigma_j) \prod_i \Phi(\sigma_i) \quad . \tag{1.18}$$

The probability distribution is now in a form similar to 1.11.

As depicted in figure 1.5, there are two types of function nodes; one connected to a single variable node, and representing the effect of the magnetic field H on each variable. The other connected to two variables, and depicting the interaction term between two neighbours.

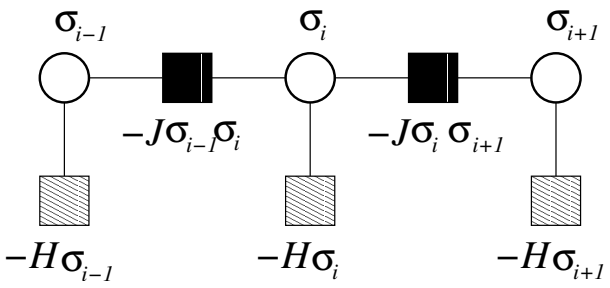

Figure 1.5: Factor graph representation of Ising model

1.3.4 Inference in combinatorial optimization

What we have shown for the Ising model has an immediate counterpart in combinatorial optimization. The locality of the interactions among variables in CSP allow us to reconsider it as a statistical inference problem as well. Let us consider a CSP whose cost function $E(\vec{x})$, where $\vec{x}$ is the congifuration of all variables $\{x_1,\dots,x_N\}$, can be expressed as a sum

$$E(\vec{x}) = \sum_{a=1}^{M} E_a(\vec{x}_a) \quad , \tag{1.19}$$

and every term E_a depends on a subset a of variables. Let us suppose that we want to sample the Boltzmann distribution $p(\vec{x}) = e^{-\beta E(\vec{x})}$. This can be rewritten as

$$\begin{aligned} p(\vec{x}) &= e^{-\beta E(\vec{x})} = e^{-\beta \sum_a E_a(\vec{x}_a)} \\ &= \prod_a e^{-\beta E_a(\vec{x}_a)} = \prod_a p_a(\vec{x}_a) \quad , \end{aligned} \tag{1.20}$$

that is again a joint probability distribution as in equation 1.14. This simple argument should make clear how one goes from a pure probabilistic problem as statistical inference (defined in terms of conditional probabilities) to a CSP where the relations among variables are defined in terms of constraints (functions to be satisfied).

1.3.5 Graphical models and CSP

We will see in the following how mapping a CSP onto a graphical model (a factor graph) has led many progress in the understanding of their behaviour. In particular we will see how computing marginals of variables involved in a CSP has led to develop algorithm able to solve it. What was

known before for a class of problems (typical subject of interest in physics) has proven useful to solve single instance of the same problem (a topic that is typically relevant in computer science).

Chapter 2

Methods for disordered systems: replicas and cavity

Disordered systems play an important role in physics trying to model the main features of a class of materials, and also because their study has led to the development of new tecniques and new concepts. In this chapter the most common methods used to address this problem are presented, of course without pretending to give an exhaustive treatment of the subject.

2.1 Disordered systems

In the previous chapter a quick mention has been made to an example of a disordered system: the Ising spin glass, with the Hamiltonian 1.2. We stress here that we are interested in those systems exhibiting *quenched disorder*. Systems where the interactions are specified by random variables that do not change. This reflects physical systems where the time scale over which this variables change is long compared to the one over which dynamical variables change. This poses several problems. Let us consider again the Hamiltonian

$$\mathcal{H} = -\sum_{\langle ij \rangle} J_{ij}\sigma_i\sigma_j - H\sum_i \sigma_i \quad . \tag{2.1}$$

As we said, the interaction constants J_{ij} are drawn by a probability distribution $p(J)$

As the Hamiltonian depends on the specific realization J of the disorder, then all observables follow this dependance too. So, the partition function reading

$$Z_J = \sum_{\{\vec{\sigma}\}} \exp\{-\beta \mathcal{H}_J(\vec{\sigma})\} \quad , \tag{2.2}$$

the realization-dependent free energy density is

$$f_J = -\frac{1}{\beta N} \ln Z_J \quad , \tag{2.3}$$

where N is the system size.

But even from physical arguments the typical features of a disordered system do not vary from sample to sample, so some property must be taken into account to explain this fact: the *selfaveragenes* property. This amounts to say that, for large enough systems, physical properties as free energy do not depend on the particular realization of the Js. In formulas

$$\lim_{N\to\infty} f_J(\beta, N) = f_\infty(\beta) \quad . \tag{2.4}$$

Equivalently we can say that the average of a self averaging quantity over the disorder is well defined and coincides with its thermodynamic limit:

$$f = -\lim_{N\to\infty} \frac{1}{\beta N} \overline{\log Z_J} = f_\infty \quad . \tag{2.5}$$

In other words we are asking the free energy to have small fluctuations (of the order of 1 over the system size)

$$\overline{f^2} - \overline{f}^2 \approx \frac{1}{N} \quad . \tag{2.6}$$

One can convince himself that this is the case for short range interaction systems with the following argument. Let us suppose that our system, while going to infinity, is splitted in an infinite number of smaller syubsystems, such that their size grows to infinity too. The global free energy is the sum of the free energies of the subsystems, plus a negligeable surface contribution. As these are i.i.d. random variables, because of the central limit theorem we can expect that the 2.6 holds. It is worth noting that this applies to the free energy, logarithm of the partition function. A similar conclusion

cannot be drawn for the partition function itself.

A similar argument is still lacking for long range systems, where nonetheless selfaveraging is seen numerically.

2.2 Replica method

As we have seen before, it is meaningful to calculate the average free energy, obtained as

$$f = \overline{f_J} = \int dJ p(J) f_J = -\frac{1}{\beta N} \int dJ p(J) \ln Z_J \quad , \tag{2.7}$$

which immediately present the difficulty of integrating a logarithm. In fact, it is different by other computation statistical mechanics is used to. An easier approach might be to calculate the logarithm of the mean

$$f_a = -\frac{1}{\beta N} \ln \int p(J) Z_J \quad , \tag{2.8}$$

the *annealed average*, but in many cases this would lead us to wrong results. Section 4.7 presents a more detailed analysis of the correctness of the annealed average for a particular disordered system.

2.2.1 The replica trick

A method to perform the calculation avoiding this problem is the so called *replica trick* [17]. It exploits the simple expansion $x^n \simeq 1 + n \ln x$ for $n \to 0$, applied to the n^{th}-power of the partition function Z_J

$$(Z_J)^n = \sum_{\{\vec{\sigma}^1\}} \sum_{\{\vec{\sigma}^2\}} \cdots \sum_{\{\vec{\sigma}^n\}} \exp\{-\sum_{a=1}^{n} \beta \mathcal{H}_J(\vec{\sigma}^a)\} \quad . \tag{2.9}$$

This represents the partition factor of a system made of n **non-interacting** replicas of the same realization of disorder J. So, as

$$\overline{(Z_J)^n} = \overline{1 + n \ln Z_J} = 1 + n \, \overline{\ln Z_J} \quad , \tag{2.10}$$

we can write

$$\overline{\ln Z_J} = \frac{\overline{(Z_J)^n} - 1}{n} = \frac{Z_n - 1}{n} \quad , \tag{2.11}$$

having introduced $Z_n \equiv \overline{(Z_J)^n}$. Z_n is the average over the disorder of the partition function of the n replicas, its free energy density being

$$f_n = -\frac{1}{\beta n N} \ln Z_n \quad . \tag{2.12}$$

Finally

$$\begin{aligned} \overline{f} &= -\frac{1}{\beta N} \overline{\ln Z_J} \\ &= -\frac{1}{\beta N} \lim_{n\to 0} \frac{Z_n - 1}{n} \\ &= -\frac{1}{\beta N} \lim_{n\to 0} \frac{e^{-\beta n N f_n} - 1}{n} \\ &= \lim_{n\to 0} f_n = f_0 \quad . \end{aligned} \tag{2.13}$$

The strategy is then to consider a system made of n non-interacting replicas, compute the partition function and the free energy, and perform the analytic continuation $n \to 0$. The physical meaning of considering a continuous number of systems, and taking its limit to zero, has been the subject of many discussions, that somehow go far beyond the scope of this chapter.

2.2.2 Replica symmetry and its breaking

Even without showing any full replica calculation we will discuss some of the problems that may arise and how they have been addressed.

We have said that when considering the replicated system, the n replicas share the same realization of the disorder. Even if the replicas do not interact, the average over the **same** realizations of the disorder gives rise to an effective interaction that brings all the replicas to the same energetically favoured regions, repelling them from the unfavourable ones.

RS As long as only one energetically stable regions ("valley") exist, the typical overlap between two replicas will give information on the typical size of this region. Such situation is said to be *replica symmetric*. The distribution of the overlap q between two random replicas is given by $P(q) = \delta(q - q_0)$[1].

[1] The overlap between states a and b is defined as $q^{ab} = \frac{1}{N}\sum_{i=1}^{N} \sigma_i^a \sigma_i^b$.

1RSB Things get more complicated when many valleys exist, well separated in the phase space. In this situation, called *one step replica symmetry breaking (1RSB)*, the overlap distribution is given by $P(q) = x\delta(q-q_0) + (1-x)\delta(q-q_1)$. This means that, when picking up two random replicas, their solutions will be in the same valley (with typical overlap q_0) with probability x, and will otherwise be in different regions, with an overlap q_1 that takes into account the typical distance between them. Note that in some cases $q_0 = 0$, meaning that the valleys are orientated in random directions of the phase space.

fRSB This is the case where the function $P(q)$ shows more peaks, meaning that several level of correlation can be found between low energy regions.

In the combinatorial problems we have presented above earlier we spoke of a clustering phenomenon in the space of the solutions. A rigorous proof of clustering in Ksat has been presented in [18], and in [19] for 3 Ising spin glass. This symmetry breaking is exactly where it is originated from, and it is due to the replica theory that such peculiar feature has been revealed. Nevertheless, this theory has been developed exactly to draw some general conclusion on an entire class of systems, rather than on single instances of a problem. When dealing with disordered materials it gives, as this is what it was meant to do, the properties of a material, not of a specific piece of it. Computer science, on the other hand, is also interested into tecniques allowing us to solve particular instances of a problem. In this field between statistical physics and computer science, the former has given an important contribution thanks to the cavity method.

2.3 Cavity

Trying to solve the spin glass problem in the dilute case (each spin interacts with only a few of other spins) has always been considered important both for its similarity to systems of two or three dimensions, and for its close connection to optimization problems. In this case a huge help has come from the cavity method. Within this theory the average over the disorder is performed at the end, contrarily to the replica method. We have already mentioned at the end of the first chapter that the graphical representation in terms of factor graphs of disordered systems has led to the solution of single instances of CSP. This wouldn't have been possible without the insights given by the cavity method.

2.3.1 The Bethe lattice for frustrated systems

First, let us introduce the system known as *Bethe lattice*. This lattice is usually defined as the interior of a tree known as *Cayley tree*. A Cayley tree is a tree built starting from a central site and building a shell of k neighbours. Each neighbour is then connected to a second shell of k new sites for every spin in the first one. The fact that there is no overlap among neighbours chosen at each step makes this lattice a tree. This tree has an infinite number of external nodes, and this is why the interior is usually considered and is called Bethe lattice. Nevertheless, this is a good definition as long as one is interested into ferromagnetic systems. If one wants to deal with frustration, this would be eliminated by the tree property, so that a new definition must be introduced. The typical choice [20] is then to consider a different kind of lattice, where every vertex is connected to $k+1$ other vertices, usually randomly chosen among all N vertices. This lattice is locally tree-like, meaning that loops are indeed present, but typically of size $\log N$. Frustration is then preserved, but showing its effect only at global scale. This is an important concept that will be found again and again in this work.

2.3.2 The cavity method in a single state

Let us consider a spin σ_0 connected to k other spins. The trick of the cavity method is to consider the effect of spins "behind" these k spins in absence of σ_0. This effect is introduced via the *cavity fields* h_i, such that the magnetization of σ_i in absence of σ_0 would be $m_i = \tanh(\beta h_i)$. In the limit $N \to \infty$ these fields are uncorrelated as long as σ_0 is absent (note that if the graph is tree-like they are uncorrelated also at finite size). Then, by adding it, we introduce their correlation. We have then a *cavity spin* σ_0 coupled to k spins σ_i via a coupling constant J_i, and every σ_i "feels" a cavity field h_i. Let us now write the partition function of a spin σ among the k

$$\sum_{\sigma \pm 1} \exp\{\beta J \sigma_0 \sigma + \beta h \sigma\} = \exp\{\beta [w(J,h) + \sigma_0 u(J,h)]\} \quad , \tag{2.14}$$

this introduces the functions w and u. By evaluating the 2.14 at $\sigma_0 \pm 1$ and using sum rules for hyperbolic cosine we have that

$$u(J,h) = \frac{1}{2\beta} \ln \frac{1 + \tanh(\beta h)\tanh(\beta J)}{1 - \tanh(\beta h)\tanh(\beta J)} \quad . \tag{2.15}$$

Now, as

$$\tanh^{-1}(x) = \frac{1}{2}\ln\frac{1+x}{1-x} \quad ,$$

we have

$$u(J,h) = \frac{1}{\beta}\tanh^{-1}[\tanh(\beta h)\tanh(\beta J)] \tag{2.16}$$

The partition function for spin σ_0 can then be written as

$$\begin{aligned} Z_0 &= \sum_{\sigma_0\sigma_1\ldots\sigma_k} \exp\{\beta J\sigma_0\sigma_i + \beta h_i\sigma_i\} \\ &= \prod_{i=1}^{k} c(J_i,h_i)\sum_{\sigma_0}\exp\{\beta\sum_{i=1}^{k}u(J_i,h_i)\sigma_0\} \quad , \end{aligned} \tag{2.17}$$

and the magnetization of σ_0 (its mean value) is then

$$\begin{aligned} \langle\sigma_0\rangle &= \frac{\prod_i[c(J_i,h_i)]}{\prod_i[c(J_i,h_i)]} \times \frac{\exp\{\beta\sum_i u(J_i,h_I)\} - \exp\{\beta\sum_i u(J_i,h_I)\}}{\exp\{\beta\sum_i u(J_i,h_I)\} + \exp\{\beta\sum_i u(J_i,h_I)\}} \\ &= \tanh\left(\beta\sum_i u(J_i,h_i)\right) = \tanh(\beta h_0) \quad . \end{aligned} \tag{2.18}$$

In other words, the field acting on spin σ_0 is the sum of the u, functions of the other fields h_i.

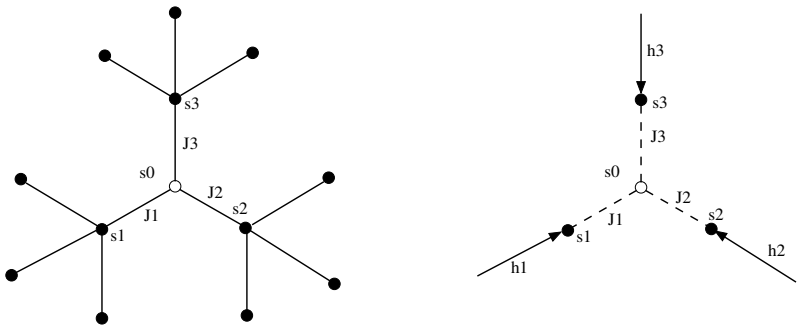

Figure 2.1: Cavity method takes a cavity spin σ_0 and considers the interaction with it of its neighbours as if it was absent.

Let us now specialize equation 2.16 at zero temperature, and when the interaction takes place among more than two spins. This will be more directly connected to the optimization problems we are interested in.

We consider again a cavity spin σ_0, now interacting with $k-1$ spins, as in figure 2.2. The

analogous of eq. 2.14 is

$$\begin{aligned} &\sum_{\sigma_1 \ldots \sigma_{k-1}} \exp\{-\beta E(\sigma_0, \sigma_1, \ldots, \sigma_{k-1}) + \beta(h_1\sigma_1 + \cdots + h_{k-1}\sigma_{k-1})\} \\ &\equiv \exp\{\beta(w_{a\to 0} + u_{a\to 0}\sigma_0)\} \quad . \end{aligned} \tag{2.19}$$

Now, as $\beta \to \infty$, we have $k-1$ spins trying to minimize the energy, so that

$$\varepsilon_a = \min_{\sigma_1 \ldots \sigma_{k-1}} [E_a(\sigma_0, \ldots, \sigma_{k-1}) - (h_1\sigma_1 + \cdots + h_{k-1}\sigma_{k-1})] = -(w_{a\to 0} + u_{a\to 0}\sigma_0) \quad . \tag{2.20}$$

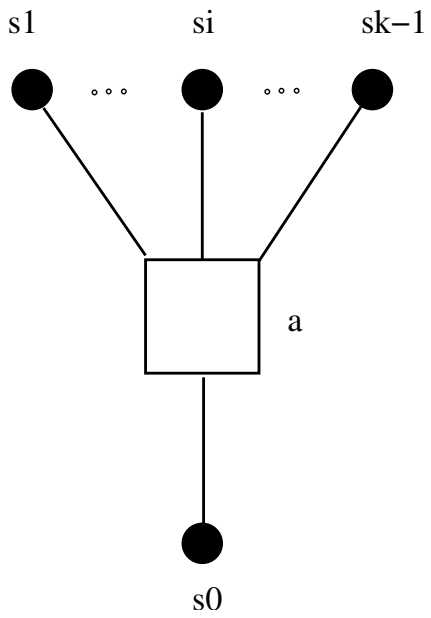

Figure 2.2: Cavity method is applicable when the interaction is represented by a function node.

Of course in the kind of problems we are dealing with, a spin participates in general to more than one interaction, i.e. it is connected to several clauses, as depicted in fig.2.3.

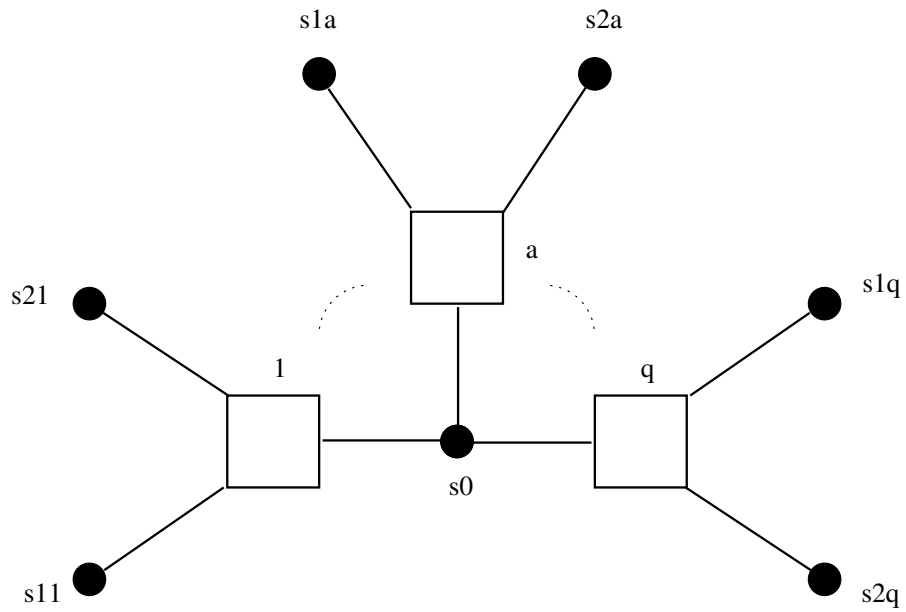

Figure 2.3: Cavity spin σ_0 participates in q functions, and is then connected to $2q$ first neighbour variables.

Difference between messages and fields

It might be worth underline here that a field on a variable (and equivalently its magnetization) is calculated by taking into account all messages arriving on it. When a message has to be sent from a variable to a function, or from a function to a variable, instead one has to consider all messages except those coming from the receiver of the message to be calculated **except** *i* **itself**. Such feature will be evident also in the following, where belief propagation (BP) equations will be introduced.

In fact the cavity method can be expressed as follows. We have a system of N spins, we add a new spin σ_0 and form a certain number q of clauses, chosing other spins to participate to this interactions. Let us suppose that this clauses are defined by $k = 3$ variables, as in the 3-SAT mentioned in chapter 1, and reported in figure 2.3[2]. The spins σ_i^a were distant in the graph previous to the addition of σ_0, and therefore in the cavity method they are considered uncorrelated. These spins minimize their

[2]Even if our notation and examples will be typical of the 3-SAT problem, it should be clear that the method can be generalized to other problems as well. We hope that this will make the treatment easier to understand, though not the most general.

energy by orientating themselves accordin to the fields felt in absence of the interaction with σ_0:

$$E(\text{cavity}) = \sum_{a=1}^{q} \min_{\sigma_1^a,\sigma_2^a} \left[-(h_a^1\sigma_1^a + h_a^2\sigma_2^a)\right] + \text{const} = -\sum_{a=1}^{q}(|h_a^1| + |h_a^2|) + \text{const} \quad . \tag{2.21}$$

When σ_0 is added, 2.21 becomes

$$\begin{aligned} E(\text{cavity} + \sigma_0) &= \sum_{a=1}^{q} \min_{\sigma_1^a,\sigma_2^a} \left[E_a(\sigma_0,\sigma_1^a,\sigma_2^a) - (h_a^1\sigma_1^a + h_a^2\sigma_2^a)\right] + \text{const} \\ &= \sum_{a=1}^{q} w_a(J,h) - \sigma_0 \sum_{a=1}^{q} u_a(J,h) + \text{const} \end{aligned} \tag{2.22}$$

Self consistent equations

In both cases we have derived an expression for the field that a cavity spin σ_0 feels due to the configurations that its neighbours would assume without interacting with itself:

$$h_0 = \sum_{a=1}^{q} u_a(J,h) \tag{2.23}$$

It should be clear that a spin selected as cavity one, will then play a different role when one of its neighbours is chosen as cavity. This gives rise to a self consistent equation for the cavity fields

$$\mathcal{P}(h) = E_J\left[\int \prod_{i=1}^{k} [dh_i \mathcal{P}(h_i)]\delta\Big(h - \sum_{i=1}^{k} u(J_i,h_i)\Big)\right] \quad , \tag{2.24}$$

where $E_J[\cdots]$ denotes an average over graphs and over the distribution of the coupling constants J. Consequently, another self consistent condition holds, namely

$$Q(u) = E_J\left[\int \mathcal{P}(h) dh \delta\Big(u - u(J,h)\Big)\right] \quad . \tag{2.25}$$

Taking this average means that the solution (if any) for this set of self consistent equations will be a distribution probability of the fields h and the biases u, not a solution of a specific instance of the problem. As we will se below, these distributions are useful to calculate other **average** quantities as the energy density.

Computation of the energy

At this stage one might start to wonder if this is enough to calculate the distribution $\mathcal{P}(h)$. For example, it happens that starting with a flat distribution of the fields and iterating the 2.24 one goes to a self consistent fixed point. Then, are we able to compute the energy of the system and other relevant quantities? Before adding σ_0 the energy of the N spins is $E_N = \text{const} - \sum_{a=1}^{q}(|h_a^1| + |h_a^2|)$, while the energy of the system with $N+1$ spins is $E_{N+1} = \text{const} + \sum_{a=1}^{q} w_a(J, h_a^1, h_a^2) - |\sum_{a=1}^{q} u_a(J, h_a^1, h_a^2)|$, then the shift is

$$\Delta E_1 \equiv \Delta E(\text{site addition}) = \sum_{a=1}^{q}(w_a(J, h_a^1, h_a^2) - |h_a^1| - |h_a^2|) - |\sum_{a=1}^{q} u_a(J, h_a^1, h_a^2)| \quad . \tag{2.26}$$

Assuming that the energy is linear in N, at least asymptotically, one might guess that the energy shift is equal to the energy density

$$\varepsilon = \frac{E}{N} = \frac{E_{N+1} - E_N}{N} = \Delta E^1 + \text{correction} \quad . \tag{2.27}$$

The correction is not of order 1/N as one might expect, but larger. In fact adding a new site from N to $N+1$ we have to make sure that the proportion between nodes and function keeps the same. To this end we have to delete (or add) some function nodes. The energy before deleting the node is given by

$$E(\text{node}) = \min_{\sigma_1,\sigma_2,\sigma_3}[E(\sigma_1,\sigma_2,\sigma_3) - h_1\sigma_1 - h_2\sigma_2 - h_3\sigma_3] + \text{const} \quad ,$$

after deleting it the three spins are free to satisfy the fields (at least from the point of view of the deleted function) and then $E(\text{no node}) = -|h_1| - |h_2| - |h_3| + \text{const}$. The energy shift due to a function deletion will be

$$-\min_{\sigma_1,\sigma_2,\sigma_3}[E(\sigma_1,\sigma_2,\sigma_3) - (h_1\sigma_1 + h_2\sigma_2 + h_3\sigma_3)] - (|h_1| + |h_2| + |h_3|) \quad . \tag{2.28}$$

On average, and for generic k, $\Delta E_2 = \Delta E(\text{function deletion})$ is given by

$$\Delta E_2 = E_J \left[\int \prod_{i=1}^{k} [dh_i \mathcal{P}(h_i)] \left(-\min_{\sigma_1,\sigma_2,\sigma_3}[E(\sigma_1,\ldots,\sigma_k) - (h_1\sigma_1 + \cdots + h_k\sigma_k)] - (|h_1| + \cdots + |h_k|) \right) \right] . \tag{2.29}$$

How many function deletions (or additions) should be performed in order to keep the function/nodes ratio constant? The system is made of N spins and $M = \alpha N$ clauses. Each clause is connected to k spins, while the connectivity of each spin (the number of function nodes it appears in) is a poissonian distributed random number with mean $k\alpha$[3]. Then, adding a new node it must be linked to $k\alpha$ functions (on average) and some random functions must be deleted.

In fact solving

$$\frac{M}{N} = \alpha \rightarrow \frac{M + k\alpha - \zeta}{N+1} = \frac{\alpha N + k\alpha - \zeta}{N+1} = \alpha \tag{2.30}$$

one finds that $\zeta = (k-1)\alpha$ functions must be randomly deleted in order to keep the ratio constant.

The final energy density is then

$$\varepsilon_0 = \Delta E_1 + (k-1)\ \alpha\ \Delta E_2 \quad . \tag{2.31}$$

We stress again that writing the above equations we have assumed that the spins connected to σ_0 are far apart in the graph before adding it. Further, we have assumed that the system is in a single pure state. As it happens with ferromagnetic ground states, correlation decays at large distances, then we have considered the variables as uncorrelated.

[3]There are $kM = k\alpha N$ edges. Then each spin is on average in $k\alpha$ function nodes

2.3.3 The cavity method with many states

Complexity

As it has been explained when discussing the replica approach, there are cases where disordered systems have multiple pure states. Roughly speaking, a state is a set of configurations whose energy cannot be decreased by a finite number of spin flips (in the $N \to \infty$ limit). In the $\beta \to \infty$ limit a state is a *cluster* of configurations all of equal energy, connected to each other by a finite number of spin flips, but separated by other clusters by a large ($O(N)$) distance. The main hypotheses done on these states is that the logarithm of the number of states at energy E, $\ln \mathcal{N}(E)$ is an extensive quantity. We define then

$$\Sigma(\alpha, \varepsilon = \frac{E}{N}) \equiv \frac{\ln \mathcal{N}(E)}{N} \quad , \tag{2.32}$$

conventionally called *complexity*. It is worth noting here that when a system shows a complexity increasing with E, it is reasonable to expect that a local search algorithm will fail to find the minimum energy configurations. This is because it will get trapped by the exponentially more numerous states at higher energy that surrounds the lowest energy clusters.

Fields and biases with many states

What we have assumed about the uncorrelation of the cavity fields in the previous case of a single pure state is not valid any more where the solutions space splits in separate clusters. Cavity variables are now assumed to be uncorrelated within states, but not in general. When a cavity spin is added, the difference in energy between before and after the addition will in general be related to different states. The equation we have written above are valid within a pure state, but not in general. The assumption is now that the distributions according to which the fields h (and consequently the biases u) are sampled, are themselves random "variables" of two distributions called h-survey and u-survey. Before averaging over the graphs we have the histograms $P^e(h)$ and $Q^e(u)$, giving the probability in a single graph of finding a message h or u flowing along the edge e. When the average over the graph is performed we end up with histograms of histograms $\mathcal{P}(P(h))$ and $\mathcal{Q}(Q(u))$, giving the probability of finding a message h or u along a random edge.

Reweighting

When cavity equations have to be written in this case, the different number of states at different energies must be taken into account. Now the surveys depend on the energy of the states. Adding a new cavity spin and computing the surveys at an energy density $\varepsilon = \frac{E}{N}$, we are considering all the surveys that were at an energy $E - \Delta E$ before the cavity iteration. The number of states is then

$$\begin{aligned} \mathcal{N}(E-\Delta E) &= \exp\{N\Sigma(E-\Delta E)\} \simeq \exp\{N\Sigma(E) - N\Delta E \frac{\partial \Sigma(E)}{\partial E}\} \\ &\propto \exp\{-\Delta E \frac{\partial \Sigma(E)}{\partial \varepsilon}\} = \exp\{-y\,\Delta E\} \quad . \end{aligned} \tag{2.33}$$

As we said, if Σ is increasing, y is positive, and then negative values ΔE are favoured.

Cavity equations with many states

Let us write the cavity equations relating the h-surveys before averaging over the graph. One finds

$$\begin{aligned} P_0(h) &= \int \prod_{a=1}^{q} [dh_a^1 P_a(h_a^1) dh_a^2 P_a(h_a^2)] \delta\left(h - \sum_{a=1}^{q} u(J_a, h_a^1, h_a^2)\right) \times \\ &\quad \times \exp\{-y\Delta E_1\} \\ &= \int \prod_{a=1}^{q} [dh_a^1 P_a(h_a^1) dh_a^2 P_a(h_a^2)] \delta\left(h - \sum_{a=1}^{q} u(J_a, h_a^1, h_a^2)\right) \times \\ &\quad \times \exp\{-y \sum_{a=1}^{q} (w_a(J, h_a^1, h_a^2) - |h_a^1| - |h_a^2|) + y| \sum_{a=1}^{q} u_a(J, h_a^1, h_a^2)|\} \quad . \end{aligned} \tag{2.34}$$

Writing altogether the update equations for the fields and the biases we find

$$Q_a^e(u) = \int dh_a^1 dh_a^2 P(h_a^1) P(h_a^2) \delta\Big(u - u(J, h_a^1, h_a^2)\Big) \exp\{y\,[w_a(J, h_a^1, h_a^2) - |h_a^1| - |h_a^2|]\} \quad , \tag{2.35}$$

and

$$P_0^e(h) = \int du_1 \cdots du_q Q_1^e(u_1) \cdots Q_q^e(u_q) \delta\Big(h - \sum_{i=1}^{q} u(J_i, h_i)\Big) \exp\{y \mid \sum_{a=1}^{q} (w_a(J, h_a^1, h_a^2)|\} \quad . \tag{2.36}$$

The way we have distributed the reweighting term among the two surveys is not unique, and other choices are possible.

Population dynamics

The above equations can be used to compute the messages flowing to a new added site. The assumed existence of a thermodynamic limit allows in principle to write a self consistency of the iteration in a way similar to what we have discussed for the RS case. In the present case, this must be extended to the functional $\mathcal{P}(P(h))$ giving the probability, when one picks up an edge $i \to a$ at random, to observe on this edge a h-survey $P_{i\to a}(h)$ equal to $P(h)$. Alternatively, one can use the functional $Q(Q(u))$ giving the probability, when one picks up an edge $a \to j$ at random, to observe on this edge a u-survey $Q_{a\to j}(u)$ equal to $Q(u)$. In the following we shall rather work with The u-surveys turn out to have a simpler structure in practice, but obviously a fully equivalent description can be obtained working with h-surveys.

We can then define a stochastic process aimed at sampling the survey space [9, 20, 21]. Let us then define a process that, at each iteration, performs the following operations:

- pick up at random a number of neighbours k, according to the Poisson distribution of mean 3α, denoted by $f_{3\alpha}(k)$[4];
- pick up at random k u-surveys $Q_1(u_1),...,Q_k(u_k)$ from the distribution $Q(Q(u))$;
- compute a h-survey $P_1(g)$ as the reweighted convolution

$$P_1(g) = C_1 \int du_1...du_k Q_1(u_1)...Q_k(u_k) \exp\left(y|\sum_{a=1}^{k} u_a|\right) \delta\left(g - \sum_{a=1}^{k} u_a\right) . \tag{2.37}$$

- pick up at random a number of neighbours k', with the probability $f_{3\alpha}(k')$;
- pick up at random k' u-surveys $Q_{k+1}(u_1),...,Q_{k+k'}(u_{k'})$ from the distribution $Q(Q(u))$;
- compute a h-survey $P_2(h)$ as the convolution

$$P_2(h) = C_2 \int du_1...du_k Q_{k+1}(u_1)...Q_{k+k'}(u_{k'}) \exp\left(y|\sum_{a=1}^{k'} u_a|\right) \delta\left(h - \sum_{a=1}^{k'} u_a\right) . \tag{2.38}$$

- pick up at random a set of couplings $\mathbf{J}$ characterizing a new function node, from the a priori distribution of couplings.

[4]Remember that there are $3M$ edges shared by $N = M/\alpha$ variables. This is peculiar to 3-SAT, but it can be generalized to other problems with other distributions.

- compute a new u-survey, $Q_0(u)$, as

$$Q_0(u) = C_0 \int dgdhP_1(g)P_2(h)\delta\left(u - \hat{u}_{\mathbf{J}}(g,h)\right)\exp\left(y\left[\hat{w}_{\mathbf{J}}(g,h) - |g| - |h|\right]\right) , \qquad (2.39)$$

where C_0 is a normalisation constant insuring that $Q(u)$ has an integral equal to one. Note that the above steps are not expressed in the most general form, rather, they refer again to the 3-SAT problem.

This iteration defines a stochastic process in the space of u-surveys, which in turn defines a flow for $\mathcal{Q}(Q(u))$, of which we would like to compute the fixed point. Following [20], this in done in practice by a population dynamics algorithm: one uses a representative population of $\mathcal{N}$ u-surveys from which the various $Q_\ell(u), \ell \in \{1,...,k+k'\}$ used in the iteration are extracted. After $Q_0(u)$ has been computed, one of the u-surveys in the population, chosen randomly, is erased and substituted by $Q_0(u)$. After some transient, this population dynamics algorithm generates sets of u-surveys which are sampled with a frequency proportional to the seeked $\mathcal{Q}(Q(u))$.

The point of this stochastic process approach is to avoid trying to write explicitely the complicated functional equation satisfied by $\mathcal{Q}(Q(u))$. This is one crucial place where the cavity method turns out to be superior to the replica method: with replicas one performs the average over disorder from the beginning, and one is forced to work directly with the functional $\mathcal{Q}(Q(u))$. As this is very difficult, people have thus been constrained to look for approximate solutions of $\mathcal{Q}(Q(u))$ where the functional is taken in a simple subspace, allowing for some explicit computations to be done.

2.4 Monte Carlo

2.4.1 Markov chain Monte Carlo

Without daring to cover such a vast subject here, we only want to mention the very basic principles underlying this technique. Monte Carlo is usually referred to all simulation techniques that rely on the generation of random numbers. More specifically, in our discussion we are often refer to it as a tool to sample some probability distribution. This is an ubiquitous task in statistical mechanics, inference and combinatorial optimization: to sample configurations $\vec{x}$ according to a probability distribution $P(\vec{x})$. The Monte Carlo approach consists in finding a Markov chain (a system whose state at time $t+1$ only depends on its state at time t) designed such that it converges to the desired dis-

tribution. Then, with a good pseudo-random number generator, simulating this chain on a computer leads us to sampling the desired distribution. This method in particular is called *Markov chain Monte Carlo*.

2.4.2 Simulated Annealing

As we have already stated, there is a strict relation between optimization and statistical physics. As we have introduced in section 1.2, it suffices to interpret the Hamiltonian as a cost function and look for the minima of it. A way to search for these minima is given by *simulated annealing* (SA). Given a generic cost function $E(\vec{x})$, one typically starts from a random configuration and an inverse temperature β. Then one moves the system to a neighbouring configuration $\vec{x}'$, whose energy is $E + \Delta E$, and the move is accepted with a probability depending on $\beta\Delta E$ given as

$$P(\vec{x} \to \vec{x}') = \begin{cases} 1 & \text{if } \Delta E < 0 \\ e^{-\beta \Delta E} & \text{if } \Delta E > 0 \end{cases}$$

The trick of SA is to let the temperature decrease to zero (or equivalently $\beta \to \infty$) during the simulation in a certain *annealing time*. Setting β to infinity from the beginning of the process would correspond to the so called *greedy* search, i.e. only energy decreasing moves would be accepted. Keeping a finite temperature and decreasing it allows the system to escape from local minima, while at the same time directing itself to lower energy regions. Even if a theorem states that in the limit of infinite time the system is guaranteed to stop in its global minimum, for obvious practical reasons we need to set a annealing *schedule*, or how the temperature behaves during the simulation. There is no general rule, and it is always necessary to test different schedules to obtain reliable results. Of course in this case we have no guarantee that the configuration found is a global minimum, but if the schedule has been wisely chosen, there is a good chance that it won't be too distant. However, SA is (at least in its simplest formulation) a local search algorithm, and there is no such hope for this class of methods to overcome the difficulties presented by systems as 3-SAT in its hard phase.

Chapter 3

Belief Propagation and Survey Propagation as cavity method on single instances

We have largely anticipated that a breakthrough in this field has come when it has become possible not only to derive average quantities on a class of problems, but also to solve real hard exemplars of this class. Basically, this became evident when a statistical inference algorithm as belief propagation was used to address constraint satisfaction problems. This chapter introduces the classes of algorithms it belongs to (message passing), a simple explanation of BP and its "extended" version, survey propagation.

3.1 Message Passing algorithms

A very successful idea in computer science is to exploit the possibility for some calculations to be performed in a distributed way. We will illustrate this with an example present in the literature [22].

Suppose a commander wants to count the number of soldiers in his troop. A natural way would be to ask his soldiers to pass by him one by one and count them, then adding one to count himself. Another way would be to arrange the soldiers in a line, part in front of him and part behind (fig.3.1).

Then all the soldiers follow the rules:

Rules to count soldiers in line

- if you are the front soldier say “one” to the soldier behind you,
- if you are the rearmost soldier say “one” to the soldier in front of you,
- if somebody behind or ahead says something, add one and say the result to the soldier on the other side.

Now the clever commander hears the two soldiers ahead and behind him, adds their numbers, adds one to count himself and finds the result. It should be clear that this trick is possible because every soldier, and the commander as well, separates the troop in two different groups, which can be counted separately. Of course this wouldn’t apply if the soldiers form a loop. Even a swarm of soldiers can be

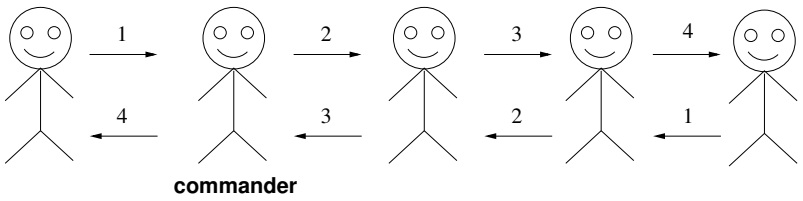

Figure 3.1: Counting the soldiers when they are in line

counted if they communicate in a way such that no cycles are present (fig.3.2), given the following set of rules:

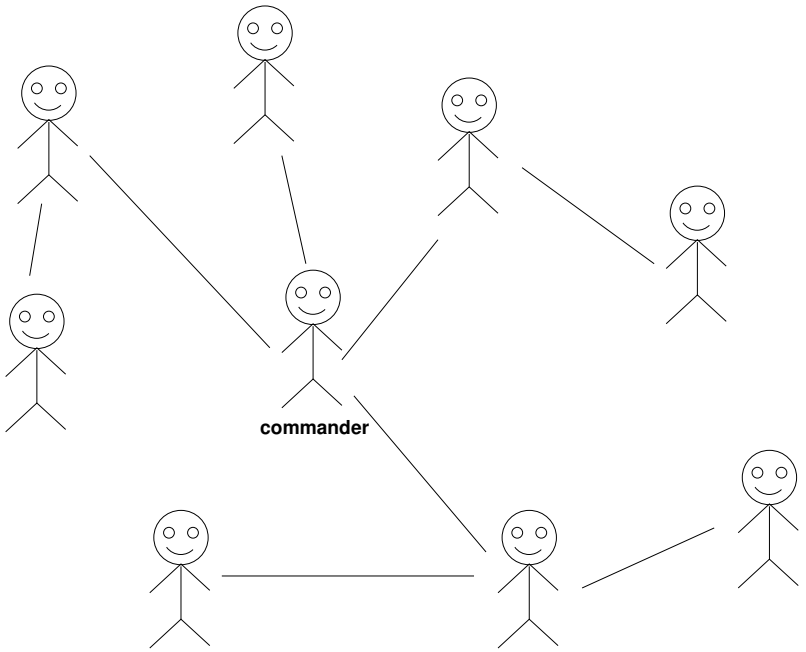

Figure 3.2: Counting the soldiers when they are in a tree

Rules to count soldiers in a tree-like swarm

- count the number n of your neighbours,
- count the number m of messages you receive from your neighbours and their values $v_1, \ldots, v_n$, and keep track of the number V of the sum of the messages received,
- if the number of received messages is $n-1$, send $V+1$ to the neighbour you have not heard from (one is yourself),
- if the number of received messages is n, then
  1) $V+1$ is the total,
  2) send to the each neighbour i the number $V+1-v_i$.

It is worth saying that in bothe cases, the rules allow **every** soldier to be aware of the total, once that all the messages have been propagated.

3.2 Belief Propagation

Now that the context should be clear, we can introduce the BP algorithm, then presenting a few examples where it has been applied.

3.2.1 Notations

Let's state some notations to be used later on. In computing marginals it's quite common to sum over all variables except a few of them, often all but one. To this end we introduce the *not-sum* or *summary*, so that, for example,

$$\sum_{\sim x_2} f(x_1,x_2,x_3) = \sum_{x_1,x_3} f(x_1,x_2,x_3)\,. \tag{3.1}$$

Another notation that will be used in the following is the symbol $\setminus$ (`\setminus`), used to subtract an element from a set.

3.2.2 The update rule

We have a factor graph for the Bayesian network associated to the joint probability function in eq.(1.14) $g(\{x_1,\dots,x_N\}) = \prod f_i(x_i \in X_i)$, and we want to calculate the marginals. As we have already said BP is a message passing algorithms, to be run on factor graphs. As there are two kind of nodes on these graph (functions and variables) there will also be two kind of messages: from variable to function and viceversa.

Update rules in BP

- a variable x sends a message to the function f along an edge e that is the product of all messages received by the functions other than f,
- a function f sends a message to the variable x along an edge e that is the not-sum on x itself of the product of the function f times the messages received from the variables other than x.

In symbols:

variable→function

$$\mu_{x\to f}(x) = \prod_{h\in n(x)\setminus f} \mu_{h\to x}(x)\,,$$

function→variable

$$\mu_{f\to x}(x) = \sum_{\sim x} \Big(f(X) \prod_{y\in n(f)\setminus x} \mu_{y\to f(y)}(y) \Big)\,,$$

where $n(z)$ is the set of neighbours of node z and X the set of variables argument of f (variable nodes connected to f). Finally, the marginal for the variable x_i is equal to the product of all messages directed toward it:

$$g_i(x_i) = \prod_{f\in n(x_i)} \mu_{f\to x_i}(x_i)\,. \tag{3.2}$$

Let's briefly note that this quantity can also be computed as $g_i(x_i) = \mu_{x_i\to f_j(x_i)}(x_i)\cdot\mu_{f_j(x_i)\to x_i}(x_i)$, i.e. the product of the two messages travelling in opposite direction along an edge connected to x_i. This is easily understood as the message transmitted by x_i along an edge is equal to the product of all the messages arriving to it except the one travelling along the same edge in opposite direction.

The update rules are nothing more than a rewriting in terms of **local updates** of the eq.(1.14), with the striking advantage of a much lower computational complexity. Computing the marginals in a standard way would require a number of operation exponential in the connectivity (all the possible states of every variable must be considered), while BP operations are expected to grow only polinomially in the system size. Of course such a simplification pays something to the correctness of the algorithm, and in fact BP is demonstrated to be exact in tree-like graphs. It gives a good approximation of the right marginals if it doesn't "feel" the cyclic structure before converging to the fixed point. In fact it is used with success also in graphs where loops exist. There is no guarantee that the procedure works in a general graph, but there are many systems now where BP is equivalent to the best known algorithm. A notable example is that of low-density-parity-check codes [23], that represent a case for which BP is the best available decoding strategy. Other cases exist where BP experiences a difficult convergence (our example of chapter 5 falls into this category) or converges

to wrong results.

3.2.3 BP applied to K-SAT

Another example of general graph in which BP has proven to be useful is K-SAT. It is believed that the loops, even if present, do not harm the correctness of the algorithm because of their length (tipically $O(\log N)$). Nevertheless there is a phase where BP ceases to be informative because it gives zero magnetization on all variables. This is the phase where many states exist, and it is usually explained as follows. In the RSB phase different parts of the graphs contribute with different states, and then, averaging on the states, the only solution is the paramagnetic one. However, this picture is still object of investigation, and we refer to [24] for further detail. The difficulty encountered by BP in calculating the right marginals has motivated the research to extending it to the more general algorithm presented below, *survey propagation* (SP).

3.3 Survey Propagation

It is worth repeating here that the u-surveys are in fact quite easy objects. One might think that a probability distribution of probability distributions has to be considered within a huge functional space. Contrarily, in the case of our interest (integer variables and $T = 0$ temperature), they are parametrized by a few real numbers. Even looking at the equations where the us are present, one can notice that they involve differences in energies that are integers, at least for the combinatorial problems we are dealing with. Further, we are interested in "frozen" variables, i.e. variables that in a given cluster are fixed (their magnetization is 1). So, in the case of K-sat, where u can only take two values the u-survey on edge $a \to i$, the survey

$$Q_{a\to i}(u) = \frac{1}{\mathcal{N}_{cl}} \sum_{\ell} \delta(u, u^{\ell}_{a\to i}) \tag{3.3}$$

can be written as

$$Q_{a\to i}(u) = (1-\eta_{a\to i})\delta(u,0) + \eta_{a\to i}\delta(u,1) \quad , \tag{3.4}$$

with ℓ denoting the cluster, and the only parameter here is the real number $\eta_{a\to i}$ denoting the probability that on the edge $a \to i$ the bias u will take value 1.

$u_{a\to i} = 1$ means that the clause a is sending a message to i because the other variables connected to a ($j \in V(a) \setminus i$) are not satisfying it. These messages depend themselves on the other biases that

clauses b connected to js are sending. And these depend on other biases as well, and so on and so forth. A simplifying assumption must be made, very similar to those presented so far. As in the cavity method, we suppose that this quantity can be calculated as the product of independent contribution from the other nodes. Again this will be exact on a tree, and approximated (and then to be verified) on a graph with loops. Let us write then

$$\begin{aligned}\eta_{a\to i} &= \prod_{j\in V(a)\setminus i} \rho_{j\to a} \\ &= \prod_{j\in V(a)\setminus i} C_{j\to a} \sum_{\{u_{b\to j}\}} \prod_{b\in V(j)\setminus a} Q_{b\to j}(u_{b\to j})\, \theta\Big(\sum_{b\in V_a^u(j)} u_{b\to j}\Big) \prod_{b\in V_a^s(j)} \delta(u_{b\to j},0) \quad . \end{aligned} \tag{3.5}$$

Let us explain the terms appearing in equation 3.5:

- $C_{j\to a}$ is a normalization constant, explained below;
- $\sum_{\{u_{b\to j}\}} \prod_{b\in V(j)\setminus a} Q_{b\to j}(u_{b\to j})$ is the probability distribution on all biases $b_{a\to j}$;
- $\theta(\sum_{b\in V_a^u(j)} u_{b\to j})$ is the constraint that clauses which make j unsatisfy a ($V_a^u(j)$) send a message;
- $\prod_{b\in V_a^s(j)} \delta(u_{b\to j},0)$ enforces that clauses that make j satisfy a do not send any message.

The last two constraints are due to the fact that we are interested in SAT configurations, then variable assignments that do not contradict any clause.

In order to compute the constants $C_{j\to a}$ let us subdivide all possible messages into three classes:

$\Pi^u_{j\to a}$ such that at least one message from $V_a^u(j)$ is equal to 1;

$\Pi^s_{j\to a}$ such that at least one message from $V_a^s(j)$ is equal to 1;

$\Pi^0_{j\to a}$ such that all $u_{b\to j}=0$ for all $b\in V(j)\setminus a$.

Then

$$C_{j\to a} = \frac{1}{\Pi^u_{j\to a}+\Pi^s_{j\to a}+\Pi^0_{j\to a}} \quad . \tag{3.6}$$

Only the messages in the first class contribute to ρ, then we have

$$\eta_{a\to i} = \prod_{j\in V(a)\setminus i} \frac{\Pi^u_{j\to a}}{\Pi^u_{j\to a}+\Pi^s_{j\to a}+\Pi^0_{j\to a}} \quad . \tag{3.7}$$

We note that equation 3.7 again makes explicit the assumption of the cavity method, to consider independently the contribution of the neighbours to a variable i as this was absent.

We can now write explicitly the expression for $\Pi^u_{j\to a}$:

$$\Pi^u_{j\to a} = \Big[1-\prod_{b\in V^u_a(j)}(1-\eta_{b\to j})\Big]\prod_{b\in V^s_a(j)}(1-\eta_{b\to j}) \tag{3.8}$$

$$\Pi^s_{j\to a} = \Big[1-\prod_{b\in V^s_a(j)}(1-\eta_{b\to j})\Big]\prod_{b\in V^u_a(j)}(1-\eta_{b\to j}) \tag{3.9}$$

$$\Pi^0_{j\to a} = \prod_{b\in V(j)\setminus a}(1-\eta_{b\to j}) \tag{3.10}$$

A term like $\prod_e(1-\eta_e)$ enforces the fact that no warning must travel along edge e, then equation 3.8 counts the probability that messages arrive from clauses making j unsatisfy a and none arrive from those making j satisfy it, as we are considering SAT contributions only: these are the only messages that contribute to ρ, among all non contradictory messages.

One can then establish an iterative algorithm, called survey propagation (SP) [25], that, starting from random values of η

- calculates Π from (3.8), (3.9) and (3.10);
- update the η using (3.7).

Iterating these steps one gets a solution for the surveys $\eta^*_{a\to i}$ and calculates the magnetizations. These are easily obtained by observing that the probability for a given variable to be frozen to 1 by choosing a cluster randomly is equal to the probability that all biases from neighbouring clauses are positive. In formulas we can write some auxiliary quantities in terms of which the magnetization can be written.

$$\begin{aligned}
\hat{\Pi}^+_i &= \Big[1-\prod_{a\in V_+(i)}(1-\eta^*_{a\to i})\Big]\prod_{a\in V_-(i)}(1-\eta^*_{a\to i}) \\
\hat{\Pi}^-_i &= \Big[1-\prod_{a\in V_-(i)}(1-\eta^*_{a\to i})\Big]\prod_{a\in V_+(i)}(1-\eta^*_{a\to i}) \\
\hat{\Pi}^0_i &= \prod_{a\in V(i)}(1-\eta^*_{a\to i}) \quad ,
\end{aligned} \tag{3.11}$$

The probabilities that, taking variable i in a cluster at random, this is frozen to values $+$ or $-$

(1 or 0) is given by

$$W_i^{(+)} = \frac{\hat{\Pi}_i^+}{\hat{\Pi}_i^+ + \hat{\Pi}_i^- + \hat{\Pi}_i^0} \tag{3.12}$$

$$W_i^{(-)} = \frac{\hat{\Pi}_i^-}{\hat{\Pi}_i^+ + \hat{\Pi}_i^- + \hat{\Pi}_i^0} \quad . \tag{3.13}$$

While

$$W_i^{(0)} = 1 - W_i^{(+)} - W_i^{(-)} \tag{3.14}$$

gives the probability that the variable is unfrozen in the cluster chosen, i.e. it appears with the same probability in one of the two states.

This algorithm presented here is able to give information on the marginals in the hard phase where BP is useless. This marginals can be used to devise a procedure to actually solve the problem, called *decimation* [25]. In this algorithm, SP is run on the instance and the magnetizations are computed. The most polarized variable (or group of variables) is fixed according to its polarization, and the graph is simplified by eliminating the constraints satisfied by this variable. Then the process is repeated until a solution is found or the formula has become easy enough to be solved by a local search algorithm. Within this framework it is possible to solve formula as large as $N = 10^5$ within minutes.

An extended approach able to find optimal solution also in the UNSAT phase (MAX-SAT problem) is presented in [26].

Chapter 4

Analysis of Random Boolean Networks

4.1 Introduction

A Boolean network (BN) is a dynamical system of binary variables, whose state is defined as a (Boolean) function of other variables.

They were introduced in the late sixties by Stuart Kauffman [27] under the name of $N-K$ model, and since then researchers have directed most of their attention toward their dynamical properties. The last few years have seen an increased attention toward this topic (figure 4.1), also because computational biology has found in these systems a powerful tool to analyze the amount of data emerging in the post-genomic era. In this field we just remind the development of a new approach as probabilistic Boolean networks [28]. The problems addressed with the use of BNs go from cell differentiation to immune response, evolution and neural networks.

Being the scientific community mainly interested, over the years, in the dynamical properties of BNs, it focused the attention on the transition that these systems present between an "ordered" regime and a "chaotic" one.

Looking at the time evolution of two different starting points one can recognize three phases:

- ordered, if their time evolution brings them close one to the other, i.e. their distance be-

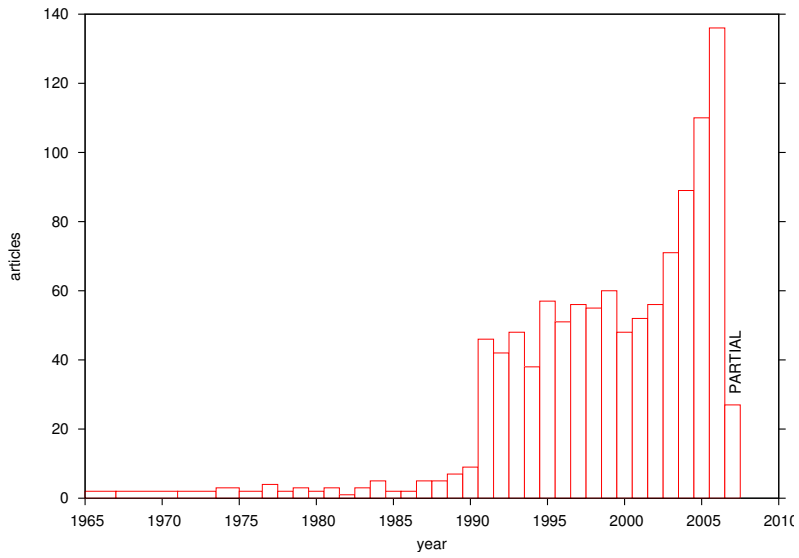

Figure 4.1: Number of articles published by year with the words "boolean network" in the title, abstract, keywords. Data from ISI Web of Knowledge.

comes smaller;

- chaotic, where the distance between them grows exponentially;

- critical, when the evolution is dominated by fluctuations, and none of the former applies.

More recently, beside their dynamical properties, some studies have been performed on their statics, i.e. on the organization of their fixed points in the thermodynamic limit, partly thanks to an innovative statistical mechanics approach [29, 30, 31].

4.2 Gene Regulatory Networks (GRN)

One of the things that most deeply characterizes this era of the biomedical research is probably the simultaneous increase of data about genomes and the possibility to conduct gene expression experiments. As technology procedes, we become more capable of following the behaviour of many genes interacting in their environment (the cell). It is then possible to address some questions about how genes interact with each other.

Let us focus on the process by which a gene is expressed, i.e. the protein encoded by it is

produced. It is well known that this process is mediated by other proteins, called transcription factors (TF). Transcription factors are themselves encoded by other genes. For example gene C is activated only when gene A and B are both expressed, and then the corresponding proteins are present. Or when A is absent and B is present. And this applies to A and B too. This inter-dependance sets up a network of interactions, which we will refer to as gene regulatory network (GRN).

4.3 Definition of the model

In our model for BN we consider N interacting variables $x_j \in \{0,1\}$ with $j \in \{1,\dots,N\}$ and M Boolean functions, represented by F_a with $a \in \{1,\dots,M\}$, depending on K inputs and having a single output. In general, each variable can be regulated by K other *parent variables*, and can enter in the regulation of an arbitrary number of *child variables*.

We restrict ourselves to the case $K = 2$, and then consider the truth value of a given output variable x_a as fixed by the truth values of the regulating variables x_{a_1}, x_{a_2} via the relation:

$$x_a = F_a(x_{a_1}, x_{a_2}) \tag{4.1}$$

with $a \in A \subset \{1,\dots,N\}$ running over all regulated genes. As shown in figure 4.3, not all variables need to be controlled by a Boolean function, i.e. in general we have $|A| = M$ with $0 \leq M \leq N$. On the other hand, each regulated variable is the output of one and only one function. The whole set of $M = \alpha N$ Boolean constraints completely specifies the network topology. In order to compute $\mathcal{N}_{\text{sol}}$, i.e. the number of stationary points of the network and characterize how thery are organized, we introduce a Hamiltonian that counts the number of unsatisfied Boolean constraints:

$$\mathcal{H} = \sum_{a \in A} x_a \oplus F_a(x_{a_1}, x_{a_2}) \equiv \sum_{a \in A} E_a(x_a, x_{a_1}, x_{a_2}) \quad . \tag{4.2}$$

The symbol $\oplus$ stands for the logical operation XOR, summarized together with its negation XNOR in the last two columns of table 4.1.

In order to clarify the structure of the graph, we introduce a classification of the variables that will be useful in the following. As depicted in figure 4.3 one can identify a set of $N-M$ variables that regulates and are not regulated by any function, called *external input variables*. A set of *regulatory*

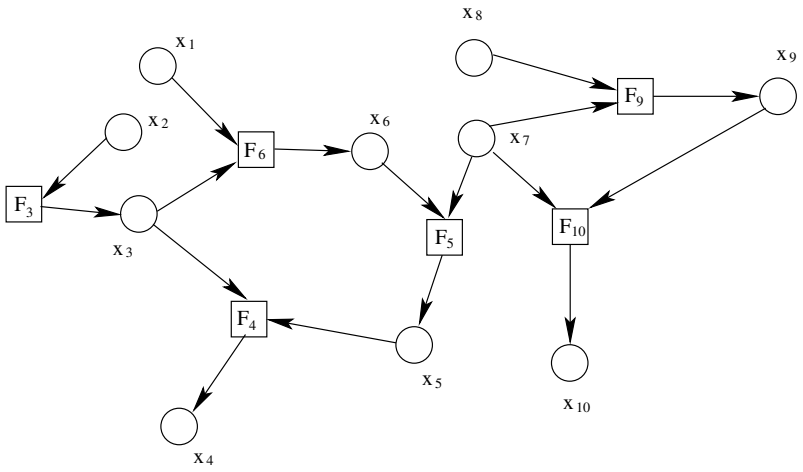

Figure 4.2: Factor graph representation of a small Boolean network: circles symbolize the variables, squares the Boolean functions. Arrows stress the directed nature of the graph. Variables x_1, x_2, x_7, x_8 are external inputs (non-regulated variables).

variables that regulate and are regulated themselves, and finally a set of *functional* variables, that are regulated without being input of any function.

x_1	x_2	0	1	x_1	$\overline{x_1}$	x_2	$\overline{x_2}$	$\wedge$							$\vee$	$\oplus$	$\overline{\oplus}$
0	0	0	1	0	1	0	1	0	0	0	1	1	1	1	0	0	1
0	1	0	1	0	1	1	0	0	0	1	0	1	1	0	1	1	0
1	0	0	1	1	0	0	1	0	1	0	0	1	0	1	1	1	0
1	1	0	1	1	0	1	0	1	0	0	0	0	1	1	1	0	1

Table 4.1: Truth table for all 16 boolean functions of $K = 2$ inputs.

We will consider *random factor graphs* characterized (among all possible random graphs) by:

(*a*) Function nodes F_a have fixed in-degree K and out-degree one.

(*b*) Variables x_a have in-degree at most one. This means that all regulating variables are collected in one single constraint F_a (see Eq. (4.1)).

Setting $\alpha \equiv M/N$, the degree distribution of variable nodes approaches asymptotically

$$\begin{aligned} \rho^{\text{out}}(d_{\text{out}}) &= e^{-2\alpha}\frac{(2\alpha)^{d_{\text{out}}}}{d_{\text{out}}!} \\ \rho^{\text{in}}(d_{\text{in}}) &= \alpha\delta_{d_{\text{in}},1} + (1-\alpha)\delta_{d_{\text{in}},0} \end{aligned} \tag{4.3}$$

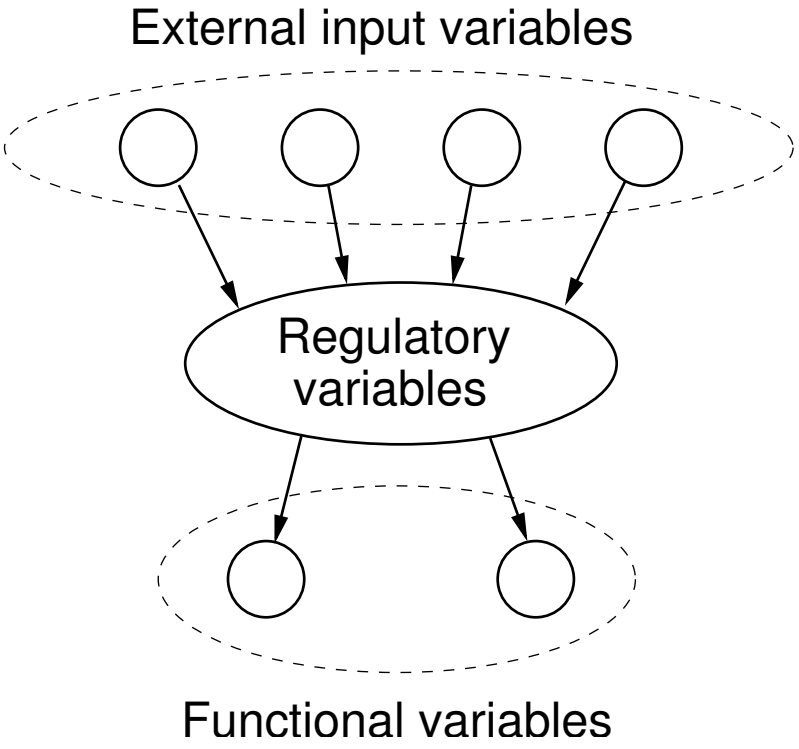

Figure 4.3: Variables can be divided in three sets depending on whether they participate as input, output, or both to the functions.

We now have to specify the functions in the factor nodes present on the random factor graph defined so far, i.e we have to specify not only the topology of the graph, but also the content. There are $2^{2^2} = 16$ possible Boolean functions with 2 inputs and 1 output. Following [32, 33], we group them into 4 classes (see again table 4.1):

- The two constant functions equal to 0 and 1.
- Four functions depending only on one of the two inputs, i.e. $x_1, \bar{x}_1, x_2, \bar{x}_2$.
- AND-OR class: There are eight functions, which are given by the logical AND or OR of the two input variables, or of their negations. These functions are *canalizing*. If, e.g., in the case $F(x_1,x_2) = x_1 \wedge x_2 = x_1$ AND x_2 the value of x_1 is set to zero, the output is fixed to zero independently of the value of x_2. It is said that x_1 is a *canalizing variable* of F with the *canalizing value* zero.

- XOR class: The last two functions are the XOR of the two inputs, and its negation. These two functions are not canalizing, whatever input is changed, the output changes too.

For the sake of clarity we concentrate only on classes depending effectively on two inputs, i.e. on those in the AND-OR class and the XOR class.

We therefore require XM functions to be in the XOR class, and the remaining $(1-X)M$ functions to be of the AND-OR type, with $0 \leq X \leq 1$ being a free model parameter. In this simple case the network ensemble is then completely defined by α and X.

4.4 The computational core phase diagram

We present here the main results obtained on the organization of fixed points in BN.

4.4.1 Propagation of external regulation (PER)

As part of the variables in the network are not regulated by any function (external input in figure 4.3), they can be considered fixed. If they are able to fix a function then also the output is fixed and the network can be pruned of the whole function. Let us see in which cases this happen.

- First of all, if two input variables of a given function are external, or they are fixed because they belong to a function that has already been fixed, than the output is fixed too.
- If the function is canalizing, it suffices that only one of the variables is external and fixed to the canalizing value, that the function is fixed too.

The PER core is the set of variables that remain in the graph once this procedure has been completed. The size of the PER core depends, in general, on the values given to the external variables, but one might expect some self-averaging property to hold true. It should also be noted that its existence crucially depends on the existence of **feedback** loops in the graph, otherwise the propagation would trivially extend to the whole network.

Now we can calculate the fraction π_{PER} of nodes in the PER core. The equation

$$1-\pi_{\text{PER}} = (1-\alpha\pi_{\text{PER}})^2 + \frac{1}{2}(1-X)2(1-\alpha\pi_{\text{PER}})\alpha\pi_{\text{PER}} \tag{4.4}$$

is explained as follows:

- the term $1-\pi_{\text{PER}}$ represents the probability that a function is deleted in the PER process;

- $(1-\alpha\pi_{\text{PER}})^2$ takes into account the probability that both function inputs do not belong to the core and then are fixed;

- $\frac{1}{2}(1-X)(1-\alpha\pi_{\text{PER}})\alpha\pi_{\text{PER}}$ considers the possibility that the function is canalized and that one of its inputs is external and fixed to the canalizing value, the term 2 being present because either of the input must be considered.

A solution is immediately found at $\pi_{\text{PER}}=0$, correct for small α. At higher values another solution different than zero sets in when α as a function of X is:

$$\alpha_{\text{PER}}(X)=\frac{1}{1+X} \quad . \tag{4.5}$$

This means that for values $\alpha>\alpha_{\text{PER}}(X)$ a PER core exist. At $X=0$ (only AND-OR-class functions) $\alpha=1$, meaning that no core is found unless $M=N$. At $X=1$ (only XOR-functions) $\alpha=0.5$, meaning that a core is found as soon as α becomes greater than 1/2.

4.4.2 Leaf Removal (LR)

Leaves in a graph are variables with degree one. In this system we distinguish in-leaves from out-leaves, variables that are inputs or outputs of the functions. This leaves are removed by the algorithm as, being underconstrained, they can trivially satisfy the function they are involved in. A function can be pruned in this algorithm in three cases (see figure 4.4). If the output is a leaf (out-leaf), this can always be set to the function value. If the function depends on two leaves (in-leaves) then they can be set to all couples of values, depending on the value needed on the output. In case of a XOR function, one in-leaf is enough to delete the function: no matter what the other input is, one can always set the leaf to a value that gives the desired output.

By iterating the LR until all possible leaves are removed, one ends up with the LR core. We won't show here the calculations that show when such a core emerges (they are reported in [29]). Again, at low values of α no LR core is present (as expected). Going to denser graphs, a discontinuous

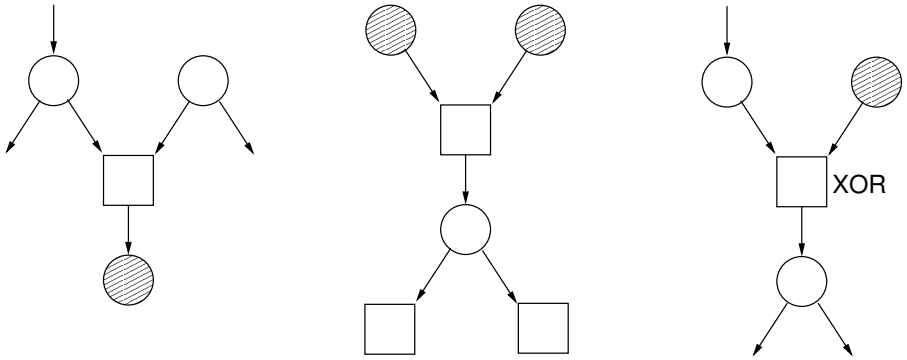

Figure 4.4: Leaf removal examples. See text for details.

transition occurs at values depending on X. For $X = 0$ this threshold is found at $\alpha = 0.5$. Increasing X, also this threshold grows, and is found at $\alpha = 0.8839$ for $X = 1$.

4.4.3 LR + PER

When both algorithms are combined, there is the hope that they will prune even further the graphs, as it actually happens. Attention must be paid to the fact that LR has to be run first, and PER after. In the opposite case the external variables would be fixed, and thus LR could not delete the corresponding functions.

Without going deeper into details, the combination of the algorithms is actually able to reduce even denser graphs, especially at intermediate values of X. In figure 4.5 the three lines separates the region where the algorithm find a core from those where they prune the whole graph.

4.5 The cavity approach

In order to apply the cavity method introduced in chapter 2, we must rewrite 4.2 formula as a spin Hamiltonian. First of all Boolean variables $x_i \in \{0,1\}$ has to be transformed into Ising spin $\sigma_i = 2x_i - 1 \in \{\pm 1\}$. Now the Boolean functions can be rewritten as

$$\begin{aligned} E_{J_1,J_2,J_3}(\sigma_1,\sigma_2,\sigma_3) &= 1 - J_1\sigma_1\left[1 - \tfrac{1}{2}(1+J_2\sigma_2)(1+J_3\sigma_3)\right] && \text{AND-OR class} \\ E_J(\sigma_1,\sigma_2,\sigma_3) &= 1 - J\sigma_1\sigma_2\sigma_3 && \text{XOR class} \quad . \end{aligned}$$

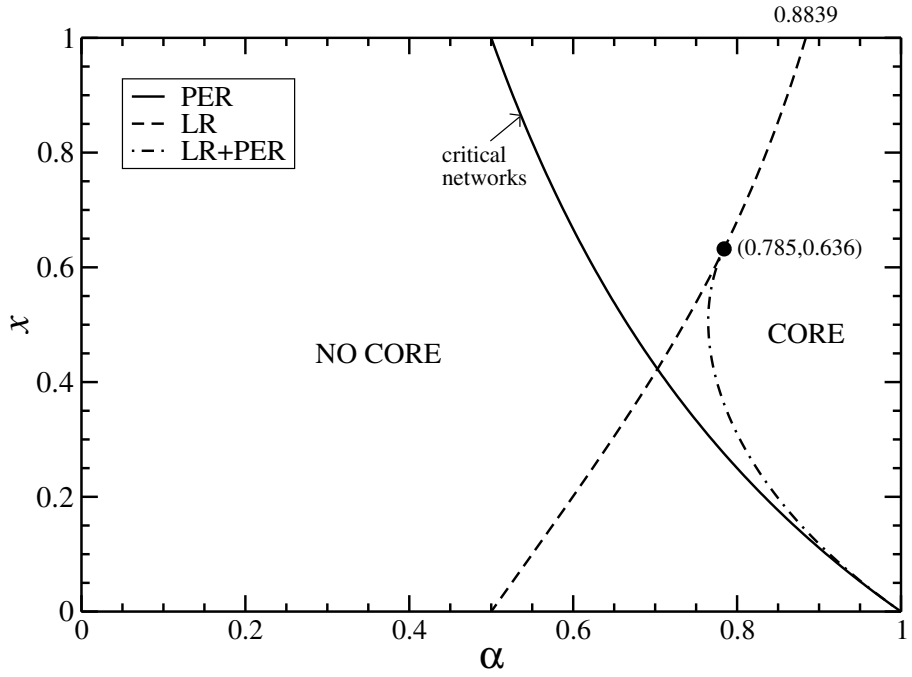

Figure 4.5: Phase diagram for the PER, LR, and LR+PER algorithms

4.5.1 Belief Propagation

The BP analysis (as delineated in chapter 3) can be performed on BNs as well.

The update equations for the messages are

$$m_{a\to i}(x_i) = \sum_{\substack{\{x_j\}\\ j\in a\setminus i}} \left[1-\frac{1}{2}E_a(\mathbf{x})\right] \prod_{j\in a\setminus i} \mu_{j\to a}(x_j) \tag{4.6}$$

$$\mu_{i\to a}(x_i) = C_{i\to a} \prod_{b\in i\setminus a} m_{b\to i}(x_i) \quad . \tag{4.7}$$

For our system, as it defines a directed graph, the following simplification hold

$$\begin{aligned} P(x_a,x_{a_1},\ldots,x_{a_K}) &= P(x_a|x_{a_1},\ldots,x_{a_K})P(x_{a_1},...,x_{a_K}) \\ &= \left[1-\frac{1}{2}E_a(x_a,x_{a_1},\ldots,x_{a_K})\right]\prod_{l=1}^{K} P_{a_l}(x_{a_l}) \end{aligned} \tag{4.8}$$

on zero energy configurations. On a tree the factorization 4.8 can be propagated all up to the external

input variables, leading to

$$\mathcal{P}(\vec{x}) = \prod_{a\in A}\left[1-\frac{1}{2}E_a(x_a,x_{a_1},...,x_{a_K})\right]\prod_{i\in \mathrm{EIV}} P_i(x_i) \quad , \tag{4.9}$$

where EIV is the set of external input variables as defined in section 4.3 and figure 4.3. The entropy can then be calculated, using the definition

$$S = -\sum_{\vec{x}} \mathcal{P}(\vec{x}) \ln \mathcal{P}(\vec{x})$$

and it turns out to be

$$\begin{aligned} S =& -\sum_{\vec{x}}\left\{\prod_{a\in A}\left[1-\frac{1}{2}E_a(x_a,x_{a_1},\ldots,x_{a_K})\right]\prod_{i\in \mathrm{EIV}} P_i(x_i)\left[\sum_{a\in A}\ln\left[1-\frac{1}{2}E_a(x_a,x_{a_1},\ldots,x_{a_K})\right]+\sum_{i\in \mathrm{EIV}}\ln P_i(x_i)\right]\right\} \\ =&(N-M)\ln(2) \quad . \end{aligned} \tag{4.10}$$

This equation is explained as follows:

- $\sum_{\vec{x}}$ counts for 2^N configurations;
- $\prod_{a\in A}\left[1-\frac{1}{2}E_a(x_a,x_{a_1},\ldots,x_{a_K})\right]$ is equal to 1 with probability 1/2, and, as M terms like this exist, they contribute with $(\frac{1}{2})^M$;
- as the external inputs are unbiased $\prod_{i\in \mathrm{EIV}} P_i(x_i) = (\frac{1}{2})^{N-M}$;
- on the configuration such that $E_a = 0$, $\ln\left[1-\frac{1}{2}E_a\right] = 0$;
- $\sum_{i\in \mathrm{EIV}} \ln P_i(x_i) = -(N-M)\ln 2$.

The result is that, on average, each configuration of the external variables leads to one solution. These argument will be covered more deeply in the following, by considering this behaviour for small instances of the network.

4.5.2 Survey Propagation

Once the results regarding the entropy are established, one might wonder how the solutions are organized in the configurations space. To this end the cavity approach can be used, leading to a non-

trivial result. It is found analytically, using the techniques of [1], that the case $X = 1$ (pure XOR) shows a jump in the complexity Σ from zero to ≈ 0.08 at $\alpha_d = 0.883867$. For $\alpha \geq \alpha_d$, Σ goes from $\Sigma(\alpha_d) \approx 0.08$ to zero for $\alpha = 1$ (UNSAT transition). Beside this, the entropy calculated by BP does not show any transtion, and the fact that the entropy is larger than the complexity qualifies for an exponential size of each cluster.

It is worht noting here that the case $X = 1$ is not completely equivalent to the XORSAT problem presented in [1], as the ensemble chosen here for the graph is different. Consequently, also the thresholds are located at different values of α.

When $X \neq 1$ the analytical prediction is not possible, but the cavity equations can be solved numerically and give an estimate for the complexity. The phase diagram is then obtained and shows the presence of a clustered 1-RSB phase below a threshold line that goes from $\alpha = \alpha_d, X = 1$ to $\alpha = 1, X = 0$, as reported in figure 4.6.

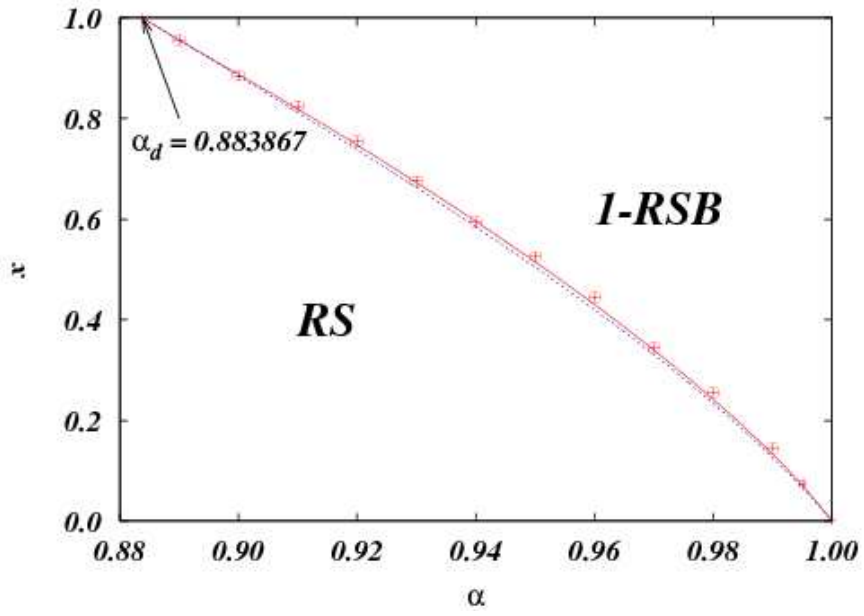

Figure 4.6: RS-1RSB transition for Boolean networks.

4.5.3 Summary of the phase diagram

The results presented so far about the organization of the fixed points in large random Boolean networks are summarized in figure 4.7.

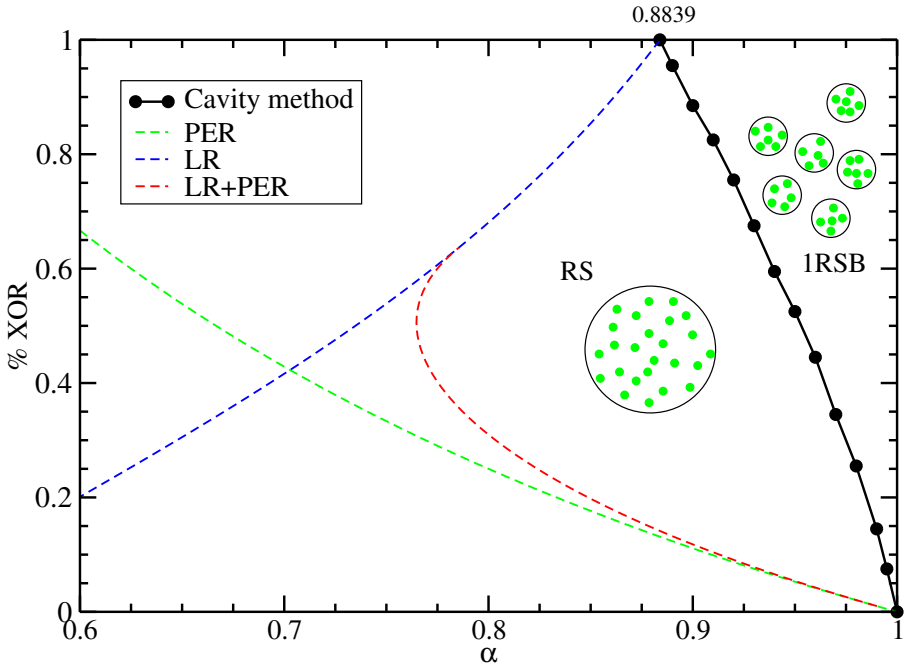

Figure 4.7: Phase diagram.

4.6 Finite size analysis: numerical results

In this and the following section we will report the results of the analysis of RBN on small instances [31].

4.6.1 Exhaustive search

No known polynomial algorithm is generally able to exhaustively find all the solutions of a Boolean constraint satisfaction problem like this one. There is however a number of efficient implementations of exhaustive search strategies - still exponential in the running times - that allow to explore problems of reasonable size.

In our case we have mapped the random BN instances onto *conjunctive normal form* (CNF) formulas. Such instances are made of several clauses put in AND disjunction, each clause being made of literals in OR conjunction. This is the natural form of the well-known satisfiability problem. The mapping is done writing in the CNF formula all the configurations which violate a clause in the RBN instance, and negating the literals for the true variables. As an example, it is simple to verify that a AND node involving x_1, x_2 e x_3 as:

$$x_1 \oplus (x_2 \wedge x_3) \ ,$$

can be written in CNF in the following way:

$$(x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_3) \ .$$

Once the problem is cast in this form, we can exploit the vast number of very well performing algorithms existing for solving satisfiability instances [34]. We are interested now in exhaustive search programs, among which we choose `relsat 2.00` [35]. This award-winning program can perform a complete search of the solution space for instances up to ≈ 500 variables (in our model) in accessible time using a common pc.

4.6.2 Entropy

As reported above (and [29]) it has been pointed out using an heuristic argument that, for this model of random BNs, the average number of solutions is always equal to 2^{N-M} i.e. to the number of all possible external input configurations (note that $N-M$ is exactly the number of non-regulated sites). A direct numerical check of BP equations on single samples shows that the BP predictions are always in agreement with the above-mentioned heuristic prediction, so that for any sample, the entropy density $s = S/N = (N-M)/N \log(2) = (1-\alpha)\log(2)$, independently from the sample realization and the percentage X of XOR functions, apart from terms that vanish when $N \to \infty$. In figure (4.8) we display the frequency distribution of $\ln \mathcal{N}_{sol}$ for 10000 samples at different values of α and X. Comparing these histograms one can observe that

- The most probable value of $P(s = \ln(\mathcal{N}_{sol})/N)$ depends strongly on α and only mildly on X.

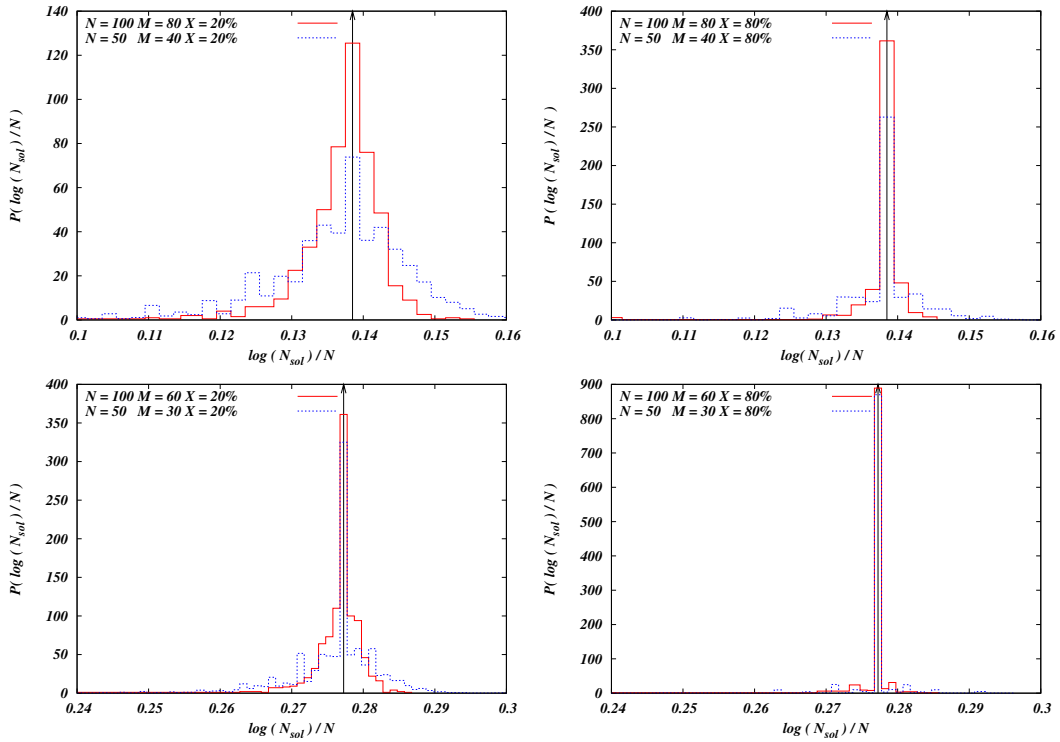

Figure 4.8: Histograms of the exhaustive algorithm estimates measured on 10000 samples of $N = 50$ (red boxes) and $N = 100$ (blue boxes) for four different choices of $\alpha \in \{0.6, 0.8\}$ and $X \in \{20\%, 80\%\}$ expressed as a percentage. The black arrow is the numerical estimate using BP, that agrees perfectly with the theoretical value $\sigma = (1-\alpha)\ln 2$.

- The distributions at increasing N seem to peak around the value $s = (1-\alpha)\ln 2$.

4.6.3 Magnetization

So far we have analyzed the behavior of the entropy alone. Although the entropy is very well predicted by BP, this is not always the case of the single variables marginal probabilities $P_i(x_i)$ These quantities are of extreme importance in any decimation procedure that is expected to find a specific fixed point via BP. Necessary condition for any decimation procedure to be effective is, for each non pathological sample, a strong positive correlation between magnetization vectors $\vec{m}^{BP}$ and $\vec{m}^{EX}$, N-dimensional vectors whose i-th component represents the magnetization of each variable x_i defined as

$$m_i^{BP/EX} = P_i^{BP/EX}(1) - P_i^{BP/EX}(0) \ , \tag{4.11}$$

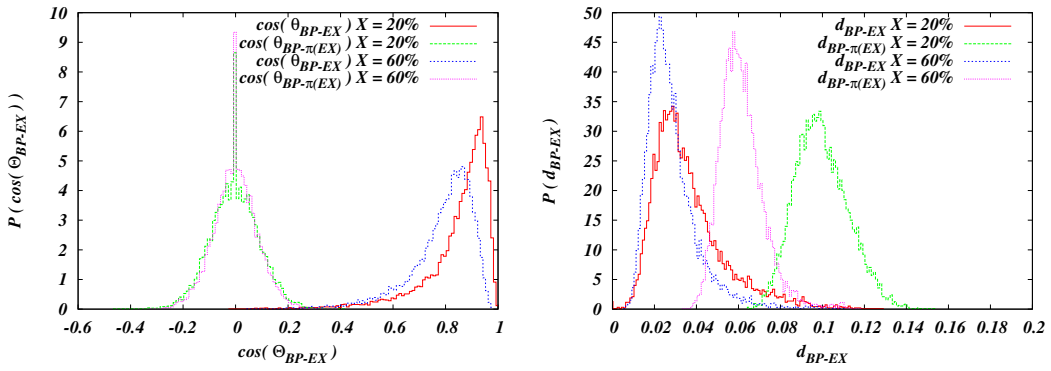

Figure 4.9: Histogram of $\cos(\theta_{BP-EX})$ compared with the reference histogram of $\cos(\theta_{EX-\pi(EX)})$ (left panel). In the right panel we display the histogram of both d_{BP-EX} and $d_{EX-\pi(EX)}$. Measurement are done at $N = 100$ $M = 90$ and a percentage of XOR functions of $X = 20\%, 60\%$ over an ensemble of 10^4 samples.

where $P_i^{EX}(x) = P_i^{true}(x) = \mathcal{N}_{sol}(x_i = x)/\mathcal{N}_{sol}$. In order to assess this point, we define two global parameters testing respectively:

- the probability distribution of the relative angular overlap of the two magnetization vectors in the N dimensional space, and
- the probability distribution of relative magnetization euclidean distances.

For each sample, the first overlap parameter is defined $\cos(\theta_{BP})$, *i.e* as the cosine of the angle between the exhaustive and BP magnetization vectors:

$$\cos(\theta_{BP-EX}) \equiv \frac{\vec{m}^{BP} \cdot \vec{m}^{EX}}{|m^{BP}|\,|\vec{m}^{EX}|} \,. \tag{4.12}$$

The second is defined as the euclidean distance between non normalized magnetizations:

$$d_{BP-EX} = \frac{1}{N}\sqrt{\sum_{i=1}^{N}\left(m_i^{BP} - m_i^{EX}\right)^2} \tag{4.13}$$

In both cases the predictions have been tested against a null hypothesis. In the null hypothesis, random magnetization vectors are extracted in the following way: for each sample $\vec{m}^{\pi(EX)}$ is a random permutation of the components of $\vec{m}^{EX}$, so that $m_i^{\pi(EX)} \equiv m_{\pi(i)}^{EX}$ where $\pi(i)$ is a random permutation of the ordered set $\{1,\ldots,N\}$. The quantities $\cos(\theta_{EX-\pi(EX)})$ and $d_{EX-\pi(EX)}$ are then calculated substituting $\vec{m}^{\pi(EX)}$ to $\vec{m}^{BP}$ in eqs.(4.12) and (4.13). Distributions of the overlaps over the sample

populations are plotted in figure 4.9. The results show a strong correlation between the true and the predicted magnetization vector distributions.

4.6.4 Solutions overlap

Let us stress again here that in this model the entropy is analytic in all the phase diagram, while the organization of the fixed points undergoes a sudden reorganization at some $\alpha_d(X)$:

- At $\alpha < \alpha_d(X)$ all solutions are in a single cluster, I.E any pair of solution is connected by a path via other solutions, where in each step only a finite number of variables can be changed.

- At $\alpha > \alpha_d(X)$ The space of solutions spontaneously breaks into an exponential number of macroscopically separated clusters of fixed points. Their number, or more precisely its normalized logarithm, is called complexity. It is a first-order phase transition.

This behavior is characterized by the appearance of a non-trivial structure of the space of solutions. In other words the fixed points, rather than being uniformly scattered over the N-dimensional hypercube, start to organize themselves in clusters, with a well defined *intra-cluster* and *inter-cluster* overlap. Nevertheless, by means of the exhaustive enumeration technique introduced above, one can easily write down all the solutions for a given sample and hence calculate all the $\mathcal{N}_{\rm sol}(\mathcal{N}_{\rm sol}-1)/2$ overlaps, defined in a standard way as $q^{ab} = \frac{1}{N}\sum_{i=1}^{N}\sigma_i^a\sigma_i^b$. Note again that we are using spin variables here $\sigma_i = -1+2x_i$, and a,b indicates two distinct solutions. The distribution of the overlaps for two sizes and three different choices of X are shown in figure 4.10. We choose a value far apart from the clustering transition line (no XOR functions), a value in the vicinity of the transition line (50% of XOR functions) and a value deep inside the clustered phase (100% of XOR functions).

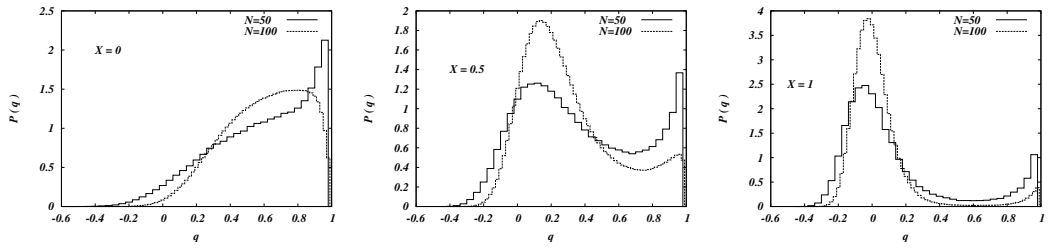

Figure 4.10: Distribution of the overlaps for 10000 samples at $\alpha = 0.93$. $X = 0$ (left panel), $X = 0.5$ (central panel), $X = 1$ (right panel), for $N = 50$ (thin line) and $N = 100$ (thick line).

It is clear that as one moves from the unclustered to the clustered phase (which at fixed α is equivalent to increasing X), the distribution changes from a broad shape to a two-peaked one. However, this two-peaked pattern is just a finite size effect, as it seems to be indicated by the reduction of the weight of the right peak of the $P(q)$ from $N = 50$ to $N = 100$ shown in the left panel of figure 4.10. As the number of clusters is exponential in the system size, the probability of extracting two random solutions from the same cluster is negligibly small even for moderate sizes.

This phenomenon can be understood by the following qualitative argument: let us suppose that the number of clusters as well as the size of each single cluster are exponential in the system size. Let us further suppose that the distribution of cluster sizes is strongly peaked around a single value, $\mathcal{N}_{\text{sol}-\text{in}-\text{cl}}$ (less stringent conditions can be found, together with pathological cases where those conditions do not hold, but a complete treatment would go beyond the scope of present discussion). In a completely clustered phase the weight contribution of the intra-cluster overlap will then be proportional to $\mathcal{N}_{\text{cl}}\mathcal{N}^2_{\text{sol}-\text{in}-\text{cl}}$ where $\mathcal{N}_{\text{cl}}$ is the number of clusters. On the other hand, the contribution to the inter-cluster overlap is proportional to $\mathcal{N}^2_{\text{cl}}\mathcal{N}^2_{\text{sol}-\text{in}-\text{cl}}$ (i.e. all couple of solutions except those in the same clusters). If $\mathcal{N}_{\text{cl}}$ is exponential in N, the intra-cluster contribution will be negligible with respect to the inter-cluster one in the thermodynamic limit.

4.7 Some considerations on the validity of the annealed approximation

In this section we will show that the logarithm of number of solution (i.e. the entropy) of our model of BNs computed within the annealed approximation approximation scheme agrees with the numerical estimate for the entropy we performed by exhaustive enumeration and by the numerical solution of the BP equations on single sample.

This feature allows us to make some conjectures, and check whether the annealed approximation is exact in the $N \to \infty$ limit. We will show that the variance of $P(\mathcal{N}_{sol})$ as well as all higher moments are proportional to $2^{(1-\alpha)N}$ in the large N limit, with a proportionality constant depending only on α and X. This implies that a non zero contribution to the entropy is given at the leading order only by the external regulating variables, in accordance with results obtained in [29, 30]. However, we will also show how, in the general case of $X < 1$, the proportionality constants are different from one

and greater than it, not ensuring the exactness of the annealed result, albeit replica results given in [29, 30] are a strong hint in that direction.

Moreover, we will see that in the thermodynamic limit:

$$\frac{\langle \mathcal{N}_{sol}^2 \rangle}{\langle \mathcal{N}_{sol} \rangle^2} \equiv C(\alpha, X) \tag{4.14}$$

where $C(\alpha, X)$ is a constant independent of N. Only in the case of pure XOR classes ($X = 1$), it is possible to show that the proportionality constant is exactly 1 in the large N limit for any order moment and any value of α, implying that the annealed approximation is exact.

Let's consider a distribution of positive random variables Z_i. As we are interested in the relation between the quenched entropy and the annealed approximation, we want to investigate the relation between $\langle \ln Z \rangle$ and $\ln \langle Z \rangle$, where $\langle \ \ \rangle$ identifies here the average over the distribution of Z_i. We can write:

$$\langle \ln Z \rangle = \ln \langle Z \rangle + \langle \ln \frac{Z}{\langle Z \rangle} \rangle = \ln \langle Z \rangle + \sum_{i=2}^{\infty} \frac{(-1)^{i+1}}{i} \left\langle \left(\frac{Z - \langle Z \rangle}{\langle Z \rangle} \right)^i \right\rangle \tag{4.15}$$

where we have expressed the quenched entropy in term of the annealed one plus a series of the moments of the distribution. In our case we take $Z = \mathcal{N}_{sol}$. In cases where the sum of the series of moments is finite or for any X and α diverges as a function $f(N)$ of the size of the graph such that $f(N)/N \to 0$, then the quenched and the annealed entropies coincide. We will show that this is the case at least for the second order moment. It is important to state, however, that the previous series expansion is valid only if $0 < Z/\langle Z \rangle \leq 2$, and that averaging the resulting series term by term is possible only if the sum can be taken out of the integral which defines the average.

These conditions are not always met in our model, and in particular we expect the existence of a certain range of X and α values beyond which the conditions are not necessarily satisfied. In particular we will also show that in the thermodynamic limit

$$\lim_{X \to 0} \lim_{\alpha \to 1} C(\alpha, X) = \infty \tag{4.16}$$

However this will not necessarily undermine the validity of the annealed calculation.

Under the hypothesis of equation (4.14), and thanks to a straightforward implementation of the

Tschebichev inequality, one can easily show that, (see A.2)

$$\Pr(\mathcal{N}_{sol} > \langle \mathcal{N}_{sol} \rangle) \leq 2^{-N\gamma} C(\alpha, X) \tag{4.17}$$

where γ is a positive constant. This implies that, in the thermodynamic limit, the probability distribution of the number of solutions has support in the interval $(0, \langle \mathcal{N}_{sol} \rangle)$. Unfortunately in order to take under control also the left tail of the distribution one should compute moments of the type $\langle \mathcal{N}_{sol}^n \rangle$ with $0 < n < 1$, which is, as indicated in A.3, a rather complicated task.

4.7.1 General calculation of $\langle \mathcal{N}_{sol} \rangle$

Given a probability distribution of classes of Boolean functions $\pi(f)$, drawn independently, where f is a generic Boolean function of $K = 2$ inputs, and given a value of $\alpha \in [0,1]$ one can write

$$\langle \mathcal{N}_{sol} \rangle = \sum_{\vec{x}} \Big\langle \prod_{m=1}^{M \sim \alpha N} \langle \delta(1; x_{0,m} f(x_{1,m}, x_{2,m})) \rangle_{\pi(f)} \Big\rangle_{\mathcal{G}} \tag{4.18}$$

where the external average is over the graph ensemble, while the internal one is over $\pi(f)$. Note that for the XOR class $f(x_1, x_2) = \varepsilon x_1 x_2$, while for the AND-OR class $f(x_1, x_2) = \varepsilon_0 / (2(\varepsilon_1 x_1 \varepsilon_2 x_2 + \varepsilon_1 x_1 + \varepsilon_2 x_2 - 1))$, with $\varepsilon_i \in \{\pm 1\}$ with a chosen probability and $X \in [0,1]$. The variables $\{x_i\} \in \{\pm 1\}^N$. Averaging over the values of $\{\varepsilon_i\}_{i=0,1,2}$ and X is therefore equivalent of averaging over the $\pi(f)$. In particular, in the case of flat classes distribution, which is the object of the present work, one has $prob(\varepsilon) = (\delta(\varepsilon - 1) + \delta(\varepsilon + 1))/2$. Therefore:

$$\langle \mathcal{N}_{sol} \rangle = \sum_{\vec{x}} \Big\langle \prod_{m=1}^{M \sim \alpha N} \langle \delta(1; x_{0,m} f(x_{1,m}, x_{2,m})) \rangle_{\{\varepsilon_i\}, X} \Big\rangle_{\mathcal{G}} \tag{4.19}$$

Let us now define $N_o = \alpha N$ regulated variables and $N_i = (1 - \alpha) N$ external inputs (including the $(1 - \alpha) e^{-2\alpha}$ isolated nodes). Assuming the input variables are extracted randomly and independently

on each of the N_o clauses; one can write

$$\begin{aligned}\langle \mathcal{N}_{sol} \rangle &= \sum_{N_o^+,N_i^+=0}^{N_o,N_i} \binom{N_o}{N_o^+}\binom{N_i}{N_i^+} \\ &\quad [\mathcal{P}(++|+)g(+++)+\mathcal{P}(+-|+)g(+-+)+ \\ &\quad \mathcal{P}(-+|+)g(-++)+\mathcal{P}(--|+)g(--+)]^{N_o^+} \\ &\quad [\mathcal{P}(++|-)g(++-)+\mathcal{P}(+-|-)g(+--)+ \\ &\quad \mathcal{P}(-+|-)g(-+-)+\mathcal{P}(--|-)g(---)]^{N_o-N_o^+} \end{aligned} \tag{4.20}$$

where $\mathcal{P}(\rho\sigma|\tau)$ is the probability of drawing two inputs of sign ρ and σ given the fact that one is looking at a clause with output variable sign τ, and $g(\rho\sigma\tau)$ is the value of the function node $\delta(1;\tau f(\rho,\sigma))$ times the probability of extracting a certain Boolean function type f. In the case of uniform $\pi(f)$, $\mathcal{P}(\rho\sigma|\tau)g(\rho\sigma\tau)$ is trivially $1/8$ $\forall$ signs triplet, leading to $\langle \mathcal{N}_{sol} \rangle = 2^{(1-\alpha)N}$ identically.

For a different distribution this is in general **not** the case. As a title of example, in the case of a mixture of pure XOR and AND functions, without any literal negation, $\pi(f) = X\delta(f;f_\oplus) + (1-X)\delta(f;f_\wedge)$ and eq.(4.20) reads

$$\begin{aligned}\langle \mathcal{N}_{sol} \rangle &= \sum_{N_o^+,N_i^+=0}^{N_o,N_i} \binom{N_o}{N_o^+}\binom{N_i}{N_i^+} \cdot \\ &\quad \left[\frac{2X(N-N^+)(N^+-1)}{(N-1)(N-2)} + (1-X)\frac{(N^+-1)(N^+-2)}{(N-1)(N-2)}\right]^{N_o^+} \cdot \\ &\quad \left[X\frac{N^+(N^+-1)}{(N-1)(N-2)} + \frac{(N-N^+-1)(N-N^+-2)}{(N-1)(N-2)}\right. \\ &\quad \left. + 2(1-X)\frac{N^+(N-N^+-1)}{(N-1)(N-2)}\right]^{N_o-N_o^+} \end{aligned} \tag{4.21}$$

Leading order terms can be computed following a calculation identical to the one shown below for the second moment, and will be omitted here.

4.7.2 General calculation of $\langle \mathcal{N}_{sol}^2 \rangle$

For the second moment:

$$\langle \mathcal{N}_{sol}^2 \rangle = \sum_{\vec{x},\vec{y}} \left\langle \prod_{m=1}^{M\sim\alpha N} \langle \delta(1;x_{0,m}f(x_{1,m},x_{2,m}))\delta(1;y_{0,m}f(y_{1,m},y_{2,m}))\rangle_{\{\varepsilon_i\},X} \right\rangle_{\mathcal{G}} \tag{4.22}$$

Averaging uniformly over the function types one obtains

$$\langle \mathcal{N}_{sol}^2 \rangle = \sum_{\vec{x}} \langle \prod_{m=1}^{M \sim \alpha N} \mathcal{G}^{(2)}(X; x_{0,m}, x_{1,m}, x_{2,m}) \rangle_{\mathcal{G}} \tag{4.23}$$

with

$$\mathcal{G}^{(2)}(X; x_0, x_1, x_2) = \frac{X}{2}(1 + x_0 x_1 x_2) + \frac{1-X}{2}((1 + x_1 + x_2 + x_1 x_2)\frac{x_0}{4} + 1) \tag{4.24}$$

Averages over different function types distributions are also possible, but beyond the scope of this chapter. Along the same line of arguments of the previous section, averaging over the Poissonian graph structure, one obtains

$$\begin{aligned} \langle \mathcal{N}_{sol}^2 \rangle \quad = \quad & \sum_{N_o^+, N_i^+ = 0}^{N_o, N_i} \binom{N_o}{N_o^+} \binom{N_i}{N_i^+} \\ & [\mathcal{P}(++|+) g(X;+++) + \mathcal{P}(+-|+) g(X;+-+) + \\ & \mathcal{P}(-+|+) g(X;-++) + \mathcal{P}(--|+) g(X;--+)]^{N_o^+} \\ & [\mathcal{P}(++|-) g(X;++-) + \mathcal{P}(+-|-) g(X;+--) + \\ & \mathcal{P}(-+|-) g(X;-+-) + \mathcal{P}(--|-) g(X;---)]^{N_o - N_o^+} \end{aligned} \tag{4.25}$$

where $g(X; \rho\sigma\tau)$ is now the value of $\mathcal{G}^{(2)}$ given X and the sign of the inputs. Equations (4.20) and (4.25) have an identical structure. This observation can be generalized and holds for any higher order momentum, changing the structure of the functions g. The details of the computation of the second moment are reported in appendix A. At the leading order it turns out that:

$$\langle \mathcal{N}_{sol}^2 \rangle \equiv I^{(2)}(\alpha, X) = 2^{2(1-\alpha)N} C(\alpha, X) \tag{4.26}$$

The value of the multiplicative constant $C(\alpha, X)$ as a function of X for increasing values of α is plotted in figure (4.11). Note that, unless in the pure XOR case, $C(\alpha, X) \neq 1$. This means that the analysis of the second order momentum is not enough to assess the theoretical validity of the annealed calculation of the entropy in the sense that the convergence of the logarithmic correction series is not ensured a priori. Moreover, whenever $C(\alpha, X) > 1$, $\sigma^2_{\mathcal{N}_{sol}} = \langle \mathcal{N}_{sol}^2 \rangle - \langle \mathcal{N}_{sol} \rangle^2 \propto 2^{N(1-\alpha)}$.

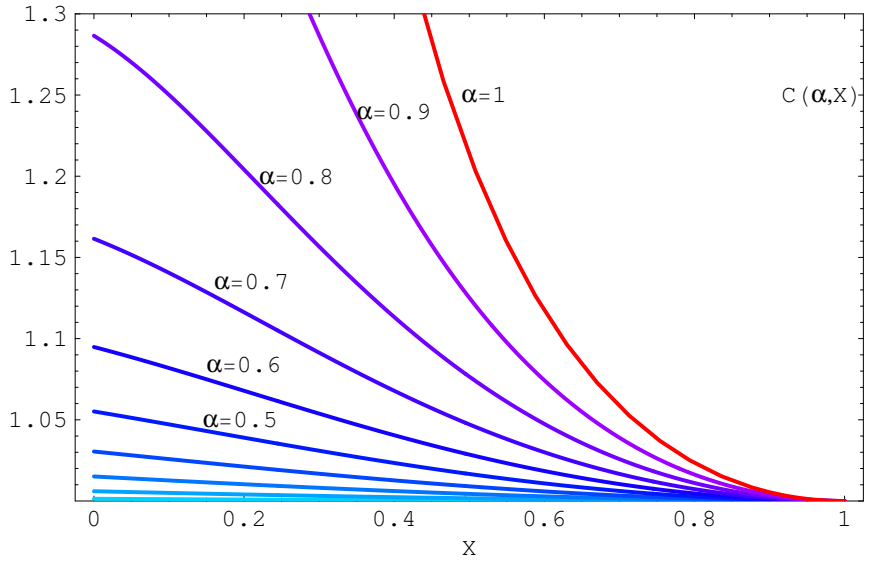

Figure 4.11: Plot of $C(\alpha, X)$ for the second moment of the distribution of the number of solutions in the large N limit. The red uppermost line is the function $e^{(X-1)}/X$, diverging for $X \to 0$

The special case of $\alpha = 1$

At $\alpha = 1$ several simplifications in the computation of the second moment hold. Both the entropic contribution due to the external regulators, and the Kullback-Leibler terms are identically zero at the saddle point, as one can check from the saddle point equation for $\langle \mathcal{N}_{sol}^2 \rangle$ presented in A. It turns out that the contribution to the saddle point includes also the term corresponding to the non-zero boundary term of integration. For the second moment in particular, one has to take into account the contribution of all terms around $N_o^+ = N_o$.

Going back to eq.(4.25) and taking explicitly into account those terms, one obtains in the large N limit

$$\langle \mathcal{N}_{sol}^2 \rangle \to I^{(2)}(1,X) + \sum_{t=0}^{\infty} \frac{e^{-t(X+1)}(t-1)^t (X+1)^t}{t!} \tag{4.27}$$

where $I^{(2)}(1,X) = e^{(x-1)}/x$. The values of both contributions diverge for $X \to 0$. Analytic estimates suggest that the divergence goes as $\sqrt{N}$. In fig. 4.12 we display the numerical estimate of $\langle \mathcal{N}_{sol}^2 \rangle$ obtained via exhaustive enumeration and the analytic estimate presented in eq. (4.25). The agreement is good, as it should be, since the computation of the second moment is exact, also for finite N.

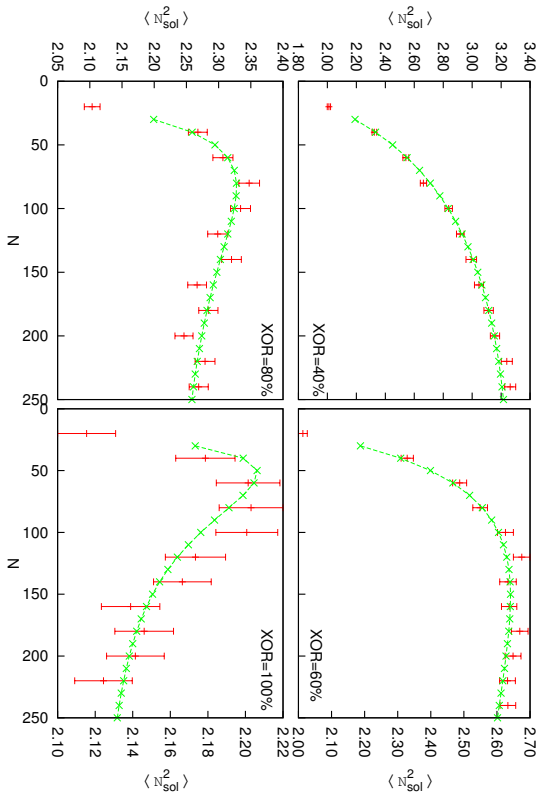

Figure 4.12: Second moment of the number of solutions distribution. Bars are computed by exhaustively counting the solutions for 100000 samples at $\alpha = 1$, crosses by evaluating the analytical formula with Mathematica.

4.8 Conclusions

In this chapter we have summarized the investigation that have been conducted in the last few years on several aspects of the fixed points solutions of a particular class of *finite size* diluted random Boolean networks with $K = 2$ inputs per function. First of all we have shown how statistical-mechanics techniques developed for disordered systems and graph theoretical approaches, already successfully used in combinatorial problems can give a deep characterization of the fixed points, well beyond what was previously known of these systems. In particular the identification of a transition from a single cluster of solutions to a clustered phase could represent the origin of many further investigations.

Then we have shown how the use of a search algorithm based on a state-of-the-art satisfiability solver can exhaustively enumerate fixed points up to moderate system sizes. For fixed system

size ensembles the average number of solutions were plotted together with average fluctuations and with two additionally ad hoc defined order parameters indicating average distance between solutions within the fixed points set. A throughout comparison of the exact results with those given by the heuristic Belief Propagation algorithm was done in order to assess the performance of BP for small size samples whose network structure significantly deviates from tree-like. BP was shown to perform significantly well in the prediction of the correct number of solutions even in small sizes cases. BP seems to loose its predicting power in the calculation of single Boolean variables marginals at the fixed points; still it was shown to retain a significant correlation with exact results in the global spatial arrangements of the solutions. Furthermore, the analytical results closing the discussion, together with their agreement with exact enumeration results, give a strong hint on the exactness of the annealed calculation of the entropy as well as the high order moments of the probability distribution of the number of solutions.

Chapter 5

Boolean-like models for Haplotype Inference

5.1 Introduction

5.1.1 Genetic variation

Individuals of the same species (like humans) obviously share much of their genomes. Nevertheless, it is known also at a popular level that no two different persons (with the exception of monozygotic twins) have exactly the same genome. Two DNAs from two individuals differ mainly because of the presence of mutations in single nucleotide positions. For example the sequence `TAACGTTA` is changed into `TCACGTTA` in a single nucleotide in second position. These variations are called SNP (pronounced *snips*) and account for most of the genetic variation between individuals (the other causes being the presence or absence of relatively short sequences). A variation is classified as a SNP if it occurs in at least 1% of the population. In the human genome a SNP is found every ≈ 300 base pairs (bp). It is believed that some of these alterations can be informative of the predisposition to certain diseases, and in the last few years a lot of effort has been put to start "big-science" projects that will collect data from different populations on SNPs.

Following Mendel, any of the forms of a SNP is named *allele*. Most of the SNPs are biallelic, meaning that it can appear in two different forms in the population. The proportion of triallelic SNPs

is much smaller, $1/570$ of the biallelic ones [36], and this explains to a large extent why most of the times the analysis is concentrated on biallelic forms. Considering again the previous example, and suppose 80% of the population presents the base A, while the rest of the population presents C. The former is called *wild-type* variant, the latter is the *mutant*.

Diploid organisms (like humans) inherit two copies of every chromosome, one from each parent, except for sex chromosomes. These two copies are almost everywhere identical, but they might present different bases at the polymorphic sites. If they both have the wild-type or mutant allele the site is *homozygous wild type* or *homozygous mutant*. If different alleles are present on the two copies then the site is *heterozygous*.

5.1.2 Recombination

The pattern of SNPs is due not only to mutation events, as their outcome is also due to another kind of phenomenon that is *recombination*. Suppose that an individual inherits from his father the alleles A_pB_p, and from his mother the alleles A_mB_m. During the meiosis the individual might produce a gamete (reproductive cell) presenting alleles A_pB_m; in this case we say that a recombinant event has taken place between the two alleles. This new genetic signature will be passed to the offspring, so a genetic variation that was not present before will be given to the population. Such an event takes place during the meiosis when DNA from the two copies of the chromosome are close and they can physically exchange their pieces. As it happens more or less at random along the DNA, if two alleles are very far apart, it is more likely that a recombination event takes place between them and they will be mixed up during meiosis. It follows that nearby segments tend to be inherited together. This is very important in *linkage analysis*, where one tests the correlation between markers on the DNA, i.e. traits that are inherited together more frequently than at random. Then, by associating markers to diseases, one is able to identify entire regions correlated to the disease and possibly even the genes contained in those regions. It must also be said here that some scientists are skeptical with respect to this approach, but discussing their point of view, though very interesting, would lead us very far from the main object of this thesis.

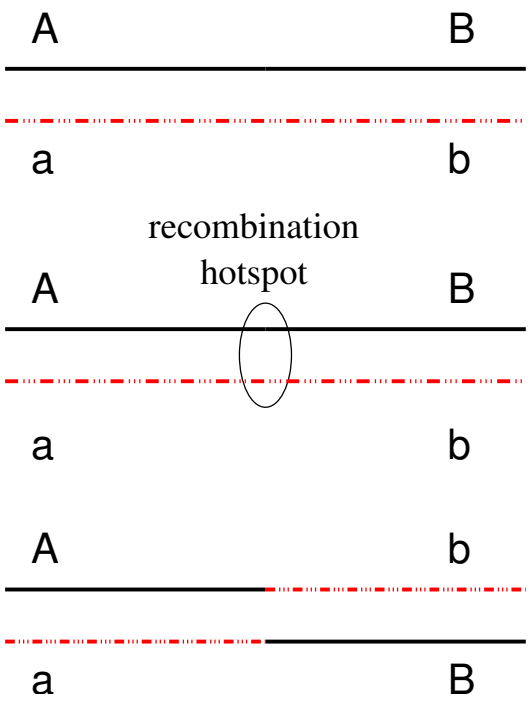

Figure 5.1: Recombination in DNA. After a recombination event two new SNP pairs show up in the population.

5.1.3 Haplotypes

Another consequence of recombination is the presence of blocks of sequences that are inherited together and are only seldom separated by a recombination event. These sequences, combinations of alleles at different *loci*, are called *haplotypes*.

The word haplotype is the contraction of *haploid genotype* and refers to the fact that it represents one half of the whole genotype taken by an individual (at least for that region). The collation of two haplotypes gives the genotype uniquely, meaning that we can immediately tell that a given site is wild-type or mutant homozygous or heterozygous. The inverse operation, as we will see in the following, poses some problem. In other words, due to the degeneracy that leads from two haplotype configurations to one genotype (wild-type-mutant and mutant-wild-type are both just heterozygous), an ambiguity is present at each heterozygous locus. This is the *unphased* information, when one does not assign each allele to a haplotype, but only considers the outcomes homozygous wild-type, homozygous mutant, heterozygous.

In disease-association studies it is sometimes preferable to deal with haplotypes rather than with the unphased genotype data. This is partly due to the fact that the statistics can benefit from a reduced dimensionality [37] when one goes from single SNPs (or just a few) to a sequence of them. It is observed in fact that the effective variability in terms of haplotypes is much smaller than all possible combinations of SNPs found, at least in those regions with low recombination rates [38]. Moreover, a cell produces at most two polypeptide chains for each protein-coding gene, even if there are many heterozygous sites. Then all possible causes of variation are encoded in these two chains, one encoded in paternal and the other in maternal haplotypes [39, 40].

Inferring the pair of haplotypes from a genotype is feasible experimentally, though expensive. The unphased information on the genotypes (just distinguishing heterozygosity from mutant and wild-type homozygosity), instead, is quickly and cheaply obtained.

Molecular genotyping Suppose that in a specific position along the DNA the base `A` is expected as wild-type and `C` as mutant. An experiment to test the genotype of an individual in that position (i.e. discern wild-type homozygous from mutant homozygous from heterozygous) is done via *PCR* (polymerase chain reaction). One amplifies the sequence in that particular region and observes the products. If only `A` (`C`) is found, this means that the SNP in both chromosomes were wild-type (mutant) and one has wild-type (mutant) homozygosity. If one founds both products, then the individual is heterozygous on that site. By repeating this at the other loci, one reconstructs the genotype for that individual, **but not disentangling one chromosome from the other**. In order to have the molecular haplotype, one has to address more complicated and expensive techniques, requiring the sequencing of DNA.

This makes computational haplotype inference attractive.

5.2 Haplotype Inference

As a result of considering only biallelic SNPs, a haplotype can be represented by a vector of entries $\{0,1\}$, while the genotype is a vector of entries $\{0,1,2\}$, as explained below. The convention used in the majority of the literature (and followed by us) is explained below:

- **Haplotypes**
  - 0 wild-type,
  - 1 mutant.
- **Genotypes**
  - 0 homozygous wild-type ($0 \boxplus 0 = 0$),
  - 1 homozygous mutant ($1 \boxplus 1 = 1$),
  - 2 heterozygous ($0 \boxplus 1 = 1 \boxplus 0 = 2$).

We have denoted this operation by the symbol $\boxplus$ (`\boxplus`). It should be now clear that a genotype is the collation of two haplotypes, or, as it is said, it is *explained* by them according to the rules above.

The haplotype inference (HI) is as follows: given a population of G distinct genotypes of length N, find the set of haplotypes such that for every genotype in the set, there exist two haplotypes explaining it. In symbols, given the $G \times N$ binary matrix $g_{j,i}$, find the $R \times N$ matrix $h_{k,i}$ such that for every j there exists a pair (j_a, j_b) such that

$$g_{ji} = h_{j_a,i} \boxplus h_{j_b,i}, \quad \forall i.$$

The first algorithm proposed to infer haplotypes from unphased data [39] requires to have at least one unambygous genotype to infer univoquely a haplotype. Then, it uses this known haplotype to see if it can help explain another genotype in a unique manner and so on. Since then other approaches have been proposed: statistical methods, aiming at maximizing the probability of observed genotypes given the haplotype frequencies or based on Bayesian estimators, and other optimization methods as *perfect philogeny*. Here we focus on the *Haplotype Inference by Pure Parsimony* (HIPP) [41].

5.2.1 Pure Parsimony Approach

While the homozygous sites uniquely define the corresponding allele in the parents, it is easily seen that a genotype presenting heterozygous sites can be explained by several different pairs of haplotypes. In particular, a genotype with k heterozygous sites can be explained by 2^{k-1} pairs of haplotypes, as in the following example:

$$\begin{aligned} 212 &= 111 \boxplus 010 \\ 212 &= 011 \boxplus 110 \quad . \end{aligned}$$

Of course a trivial solution to the HI problem is to consider twice the number of genotypes, then assigning one of the possible choices for the parent haplotypes. This is a totally unsatisfactory approach, as it is seen that few haplotypes can be responsible for the variability of many genotypes present in a population. The parsimony approach looks for the minimal set of haplotypes explaining all given genotypes.

Support for Parsimony Principle

The parsimony principle, used with success in several fields of science, finds support for this problem in comparing the observed haplotypes to the genotypes variability. *Drysdale et.al.* [38] report their finding of 10 haplotypes in a population of 121 individuals showing 13 polymorphic sites[1]. Consider also that R different haplotypes can generate up to $R(R-1)/2$ different genotypes, and there are 2^N possible haplotypes. It is then quite striking to find only 10 haplotypes out of the $8192 = 2^{13}$ possibilities and exactly those are recovered by the parsimony approach.

[1]The study presents the finding of 12 haplotypes, but if one refers only to the 121 Caucasian patients with asthma one has 18 genotypes and 10 haplotypes as reported in Table 2 of [38].

5.2.2 Traditional Formulation: Integer programming

The most common approach to deal with HIPP is by Integer Linear Programming (ILP). By ILP we mean the optimization of a linear combination of variables $x_1 \dots x_n$:

- maximize $f(x_1 \dots x_n) = \mathbf{c}^{\mathrm{T}}\mathbf{x}$,
- subject to $\mathbf{Ax} > \mathbf{b}$,
- $l_1 \leq x_1 \leq u_1$,
- $l_2 \leq x_2 \leq u_2$,
- $\dots$

The most common strategy to solve ILP problems starts by *relaxing* the problem removing the integrality constraint, then using some tricks to recover the desired integral solution.

Though already existing methods for HI can be explained in terms of parsimony criterion, the first formal definition of the problem came in 2003 by Dan Gusfield [42], who also cites a proof of its NP-completeness by Earl Hubbell. The TIP formulation presented in [42] consider a variable for all haplotypes that are possibly a genotype parent, then minimizes the number of distinct haplotypes used. As we have seen there are two possible haplotypes for every heterozygous site, and this leads to an exponential size formulation. Thus it is very limited in the size of resolvable instances. An improvement has come later (called RTIP) that catch up with this limitation expanding out a smaller set of haplotypes for each genotype, then allowing larger problems to be solved. But in both these formulations the haplotype variables are not explicitly considered.

Differently, another formulation, rather than expanding a large set of haplotypes, considers two haplotypes for each genotype by explicitly taking into account their variables. Then, it tries to fix these variables in order to minimize the number of distinct haplotypes. This formulation, together with its improvement (called respectively PolyIP and HybridIP) need some extra care when relaxed onto the LP (non-integer) problem [43].

The PolyIP formulation

As an example of a simple ILP formulation we present here the PolyIP [43], originally introduced in [44].

The PolyIP formulation

- minimize $\sum_{i=1}^{2G} x_i$ PARSIMONY

  subject to

- $h_{2i-1,j} + h_{2i,j} = g_{i,j} \quad \forall i \in \{1,\ldots,G\} \quad \forall j \in \{1,\ldots,N\}$ BIOLOG. CONSTR.
- $d_{i,k} \geq h_{i,j} - \geq h_{k,j} \qquad \forall\, 1 \leq i < k \leq 2G \quad \forall j \qquad \star$
- $d_{i,k} \geq h_{k,j} - \geq h_{i,j} \qquad \forall\, 1 \leq i < k \leq 2G \quad \forall j \qquad \star$
- $x_i \geq 2 - i - \sum_{k=1}^{i-1} d_{k,i} \quad \forall\, 1 \leq i < k \leq 2G$
- $x_i, h_{j,i}, d_{i,k} \in \{0,1\} \qquad \forall i,j,k$

Note that in this approach we have used another convention, where the genotype entry for heterozygous sites is 1 and that for mutant homozygous is 2, such that a genotype site is the algebric sum of corresponding haplotypes entries. Haplotype variables are hs and are explicitly present in the formulation for all $2G$ haplotypes. Genotype i is explained by the pair of haplotypes h_{2i-1} and h_{2i} at each position j (we denote with h_i the haplotype i, with $h_{i,j}$ its variable in position j). For every pair (i,k) of haplotypes a variable $d_{i,k}$ is introduced, equal to 1 if $h_i \neq h_k$ and zero if $h_i = h_k$. This is enforced through the equations marked with $\star$. The aim of this formulation is to minimize the number of different haplotypes, given by x_i, equal to one if h_i is unique in the haploytpe set.

5.2.3 HI as a Constraint Satisfaction Network

A very innovative approach has been introduced by Ines Lynce and João Marques-Silva [45, 46]. Their idea is to map the HI problem onto a CNF (conjunctive normal form) instance, and pass the resulting formula to a very performing SAT-solver. The formula takes the genotype as input, and finds if there is a solution of R_{min} distinct haplotypes. If not, $R_{min}+1$ is tried, and so on and so forth

until a solution is found. Exploiting the very fast SAT-solver algorithms existing has proven effective in many cases, also compared to the already established algorithms like HAPAR [47].

5.3 Message passing approach

Our mapping shares some features with the one introduced by Lynce and Marques-Silva. We set up a factor graph where four classes of variables are present:

$$\begin{array}{rcll} h_{k,i} & \in & \{0,1\} & 0 \le k < R \\ g_{j,i} & \in & \{0,1,2\} & 0 \le j < G \quad 0 \le i < N \\ t_{j,i} & \in & \{0,1,2,3\} & 0 \le j < G \quad 0 \le i < N \\ S_j & \in & \{0,\ldots,R(R+1)/2-1\} & 0 \le j < G, \end{array} \tag{5.1}$$

and two kinds of factor nodes: selector nodes and biological constraints.

Associated, haplotype and selector variables t, h, S participate to the selector node. The biological constraints connect genotypes to associated variables.

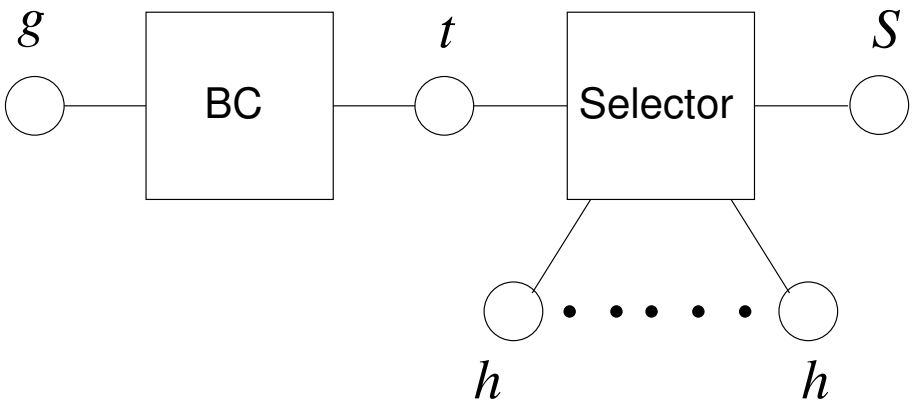

Figure 5.2: A portion of the factor graph corresponding to only one genotype site.

Rather than considering two selector variables for each genotype, we condense two selectors in a single variable S equivalent to the pair $\{S^a, S^b\}$ where $S^{a/b} \in \{0,\ldots,R-1\}$ and $S^a \ge S^b$.

Similarly, an associated variable t is equivalent to the pair $\{t^a, t^b\}$ via the simple relation $t = 2t^a + t^b$. Every selector node is connected to one selector S, one associated t and N haplotype

variables h. In formulas the selector nodes implements the constraint

$$S \equiv \{S^a, S^b\}$$
$$t \equiv \{t^a, t^b\}$$
$$h_{S^a,i} = t^a_{j,i} \quad \forall i$$
$$h_{S^b,i} = t^b_{j,i} \quad \forall i,$$

while the biological nodes implements the constraints

$$t^a_{j,i} \boxplus t^b_{j,i} = g_{j,i} \quad .$$

In some sense the genotypes polarize the associated variables at the very first steps of convergence, as they consist of BC nodes (though they can be put at finite temperature). Then the selector nodes take charge of polarizing the haplotype variables receiving the information from the associated variables.

Preliminary results

In order to have a first hint of what kind of information BP is giving, we set the problem, run BP, and look at the inferred marginals.

1. **Generate instances**
   (a) generate haplotypes of length N with the program *ms* [48],
   (b) randomly mate the haplotypes to form G genotypes[2].
2. **Compute marginals**
   (a) set R as the number of haplotypes generated in point 1a[3],
   (b) generate the factor graph with biological constraints adding a temperature to help convergence (a good choice seem to be $T = 0.5$),

[2] When one tries to generate a given number of genotypes of desired length, the actual number of generated haplotypes fluctuates.

[3] In some rare cases a haplotype is generated and not used.

(c) run BP and store the obtained marginals $p(h_{k,i})$.

3. **Inference**

   (a) if the haplotype variables show no polarization ($p(h_{k,i} = 0) = p(h_{k,i} = 1) = 0.5$, then consider it not set,

   (b) if the haplotype variables show polarization, set it to the most probable value.

4. **Assess the inference quality: haplotypes**

   (a) for all genotypes, consider the two haplotypes explaining it,

   (b) calculate the Hamming distance of the inferred haplotypes from the "true" ones,

   (c) compare them with the inferred ones, choosing the best combination among the two possible,

   (d) store the results.

5. **Assess the inference quality: genotypes**

   (a) for all inferred haplotypes pairs, generate the corresponding genotype,

   (b) calculate the Hamming distance of the genotype from the corresponding one in the population,

   (c) store the result.

We remind here that the *Hamming* distance is defined as the number of editing actions needed to change a vector into the other.

Example

- we have the genotype 212 obtained by $011 \boxplus 110$,
- we run BP with $R = 2$ and obtain the following marginals for the two haplotypes:

$$(0.52;0.48)(0.25;0.75)(0.0;1.0) \qquad (0.45;0.55)(0.65;0.35)(0.3;0.7),$$

- translated with a step function these read
  011 and 101,
- the corresponding genotype is
  $101 \boxplus 011 = 221$.

It is immediately seen that the 011 has been correctly identified, while 110 has been guessed as 101. The resulting genotype is 221 while the original one was 212 From these one obtains that for the genotypes

$$d_g(212,221) = \frac{2}{3} = 0.667\,,$$

while for the haplotypes

$$d_h(\{011,110\},\{011,101\}) = \frac{2}{6} = 0.333\,.$$

End of the Example

As shown in figure (5.3), at low enough values of the temperature, errors are quite rare, so that the genotype distance is about twice as high as the haplotype distance. This is explained as follows: one error in the haplotypes produces one error in the genotypes, but the former must be divided by $2N$, while the latter by N. As expected, when errors become more frequent, such that both haplotypes are wrong at a given site, this is not true anymore, and both distances go toward saturation.

Symmetries

This problem presents a set of symmetries that can reduce by far the efficiency of an algorithm, if not designed cautiosly. One of the symmetries is the evident permutation of two haplotypes in the same pair, solved in some formulations by ordering the haplotypes lexicographically [43, 49]. Another one seems more specific of our model. It refers peculiarly to selector variables: these can point to a pair of haplotypes to explain the first genotype, and another pair to explain the second one. Of course, as long as selectors are not completely polarized, they can point at both pairs and leaving the haplotype variables in an intermediate state. In other words selectors, and consequently haplotypes, can be "rotated" and still explain the genotypes. This problem might be solved for example ordering

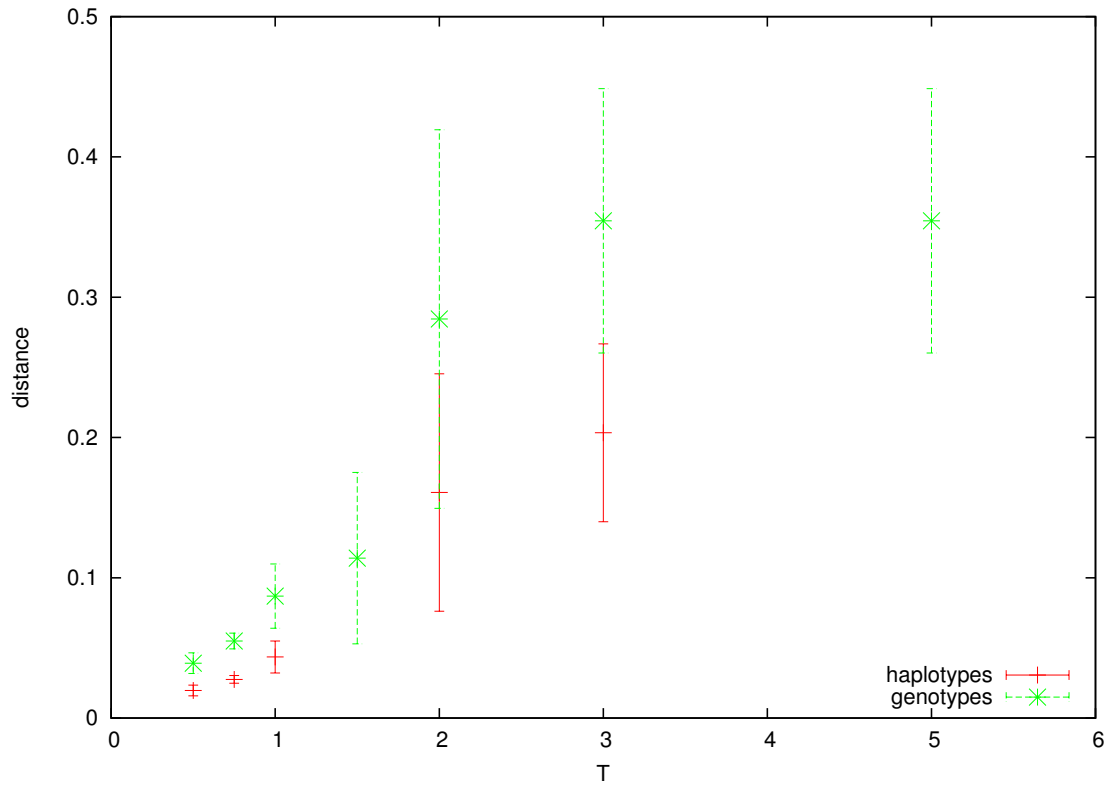

Figure 5.3: Hamming distance calculated on haplotypes and genotypes at different values of the temperature T on the BC nodes. 5 samples, 30 genotypes of 30 sites each.

the selector variables.

5.4 Perspectives

BP convergence shows an erratic behaviour. In particular, obtaining a complete solution of the instances seems too difficult to be obtained in a regular way. While some instances are easily solved, others obtained under the same conditions are too difficult to be treated. This strong dependance on the sample would suggest to tackle the problem in a slightly different mannner. An idea might be to infer the marginals for the haplotypes variables, and use them as a basis to construct a suitable starting point for the ILP problem associated. The Mixed Integer Linear Programming (MILP) solver `lp_solve 5.5` [50] provides several functions to set the initial point of the optimization procedure. The PolyIP formulation might be the starting point to test such an approach. Moreover, one could

go beyond the simple pure parsimony approach, for example focusing on the Bayesian approaches [51]. Latest advances in combinatorial approaches to HI are reported in [49].

Chapter 6

Clustering: a Data Mining Technique

Discovering hidden patterns, correlations, common features in data is the subject of that set of techniques that one refers to when speaks of data mining, statistical learning, knowledge discovery[1]. This involves the use of automated methods on large data sets, where "manual" inspection is impossible because of the amount of data. By using these methods the hypotheses done on them can be proved (or disproved) and then an information previously hidden is discovered[2].

6.1 Data Clustering

Data clustering is a technique used to group objects identified by a set of *features* (or *attributes*) according to their similarity. Given a similarity measure, one looks for a partition of the original set such that the objects in the subsets are similar to each other, i.e. share at least some of the features. These subsets are called *clusters*. The uses of this technique are multiple. Before introducing our main interest (clustering as a tool in bioinformatics), we briefly mention other different fields of application.

[1] Other expressions meaning substantially the same thing exist: knowledge management, sense making, statistical modeling.

[2] Recently this topic has also gained a popular attention because of the potential impact it may have on people's privacy. It seems that governmental agencies, and private corporations are more and more interested in collecting huge amount of data on citizens or users, being now able to perform scans of their activities and try to individuate potential threats or business opportunities. Of course this also raises privacy concerns, but expanding this discussion is not the object of this chapter.

Marketing A company might want to divide its customers based on what kind of purchase they perform (imagine the fidelity-cards that most of the supermarkets deliver, they do nothing more than recording all that a customer buys). Once recognized groups of buyers, next step is to develop marketing campaign, for example promoting some goods or developing new products.

Anomaly detection Imagine that you keep observing a certain activity, recording the basic features of the events. After a while some statistics has been accumulated and one can try to cluster this data, identifying group of similar events. If a new event does not fall into any of the clusters, than one might register it as anomalous and proceed into further analysis.

Law enforcement Some case studies have been conducted to asses the efficacy of data mining techniques in detecting and characterizing high volume crimes as burglary. An experimental study has been conducted to test the efficacy of a clustering algorithm in discovering the crimes committed by the same offender, based on the description given by the victim. Of course in such a case many subjective statements enter, and the reliability can critically depend on their quality.

6.2 Clustering in biology

Clustering application in biology is one of the first steps in gene expression studies, though it is not limited to this field. In fact it can also be applied directly to sequences in order to group genes sharing regulatory regions (an example of this kind is reported below). Proteomics and metabolomics data can be analysed via clustering too, as well as data in protein structure prediction. In the following, we will mainly refer to gene expression data clustering, keeping in mind that a clustering algorithm is usually general enough to be used with many different data.

The amount of data made available by microarray experiment is by far larger than what can be dealt with by manual inspection. Typically one obtains expression values for thousands of genes (depending on the technology of the chip used) and repeat the experiment on different conditions (as many as one can afford, the limitation being essentially the money). Then genes or samples can be clustered, depending on the kind of information one wants to extract. In the first case, the output is (in case that a valid cluster solution is found) a set of *co-expressed* genes. Often the clusters also contain sets of *co-regulated* genes or genes coding for proteins involved in a specific pathway. One

of the difficulties of clustering genes is that one typically has to deal with many more objects (genes) than features (samples, or experiments). In the other case, samples are taken from ill patients, and a subclassification may indicate the presence of subtypes of the disease. This is the case of leukemia A famous example of cluster analysis applied to samples is the identification of two subclasses of acute leukemia; lymphoblastic (ALL) from myeloid (AML) [52]. In the other class, Spellman and coworkers compiled a list of genes implicated in the yeast cell cycle, also identifying clusters of coregulated genes [53]. Clustering is an example of *unsupervised learning algorithm*, meaning that one aims at developing a procedure that behaves in a completely automatic way. Of course this is true to a certain extent: in the previous examples the use of a certain clustering algorithm is part of a more complex analysis. Whether only subset of the data is to be analyzed, what preprocessing should they undergo, how to interpret the results is of course something that has to be discussed keeping in mind what kind of questions we are asking. But then, when have to be grouped considering their similarity, an unsupervised algorithm should not ask for user intervention.

6.3 Clustering methods

We introduce here some popular and less popular clustering algorithms, of course without covering them all.

6.3.1 K-means

K-means algorithm is a popular method that starts with a given number (K) of randomly chosen centroids and assign the objects to the cluster of the closest centroid. Then it recalculates the centroid of the cluster as the average of the objects and reassigns the objects to the clusters. The process is iterated until convergence (the objects do not change clusters anymore). The algorithm is easy to understand and implement, but, as different choice of the initial centroids might yield to different partitions, it is recommended to rerun the procedure with different initial choices. As the dissimilarity criterion used is based on Euclidean distance, it tries to explain the data in terms of spherical clusters. Moreover, the number K is fixed by the user. Most of the times the "right" number of clusters is unknown, then different possibilities should be considered and an external criterion used to determine the best K. K-means is an example of combinatorial algorithm for clustering [54].

6.3.2 SOM

Self organizing maps (SOM) can be viewed as a constrained version of K-means [55]. They were invented by Teuvo Kohonen [56] (they are also known as Kohonen network) and are a technique used to reproduce the dataset on a grid (then lowering the dimensionality) while preserving the spatial arrangement of the objects.

6.3.3 Hierarchical clustering

Hierarchical clustering algorithms are divided in two classes: divisive and agglomerative. In the first case it starts from a single cluster containing all the objects and starts to divide it into smaller clusters at each iteration. At the end every object is in a separate *singleton* cluster.

In the agglomerative approach one starts from each oject in a different cluster and starts joining those with the greatest similarity. The process ends up with one single cluster containing all the objects.

It is necessary to define a measure of similarity (or dissimilarity) between groups: in the divisive approach one separates the groups withg the greatest dissimilarity, while in the agglomerative one joins those with the greatest similarity.

Such an approach does not impose a predetermined number of clusters to explain the dataset. It rather covers, among all possible structures, the most probable path from all separate clusters to a single one or the opposite. How to decide what is the best structure among those obtained is a vast subject, and we will see in the following a possible approach. Another outcome of the hierarchical approach is the *dendrogram*, a pictorial view of the process of joining or separating the clusters. It is a binary tree whose terminal nodes (leaves) are the objects, and the vertices represent the joining (or separation) of two clusters. Following the tree from the leaves to the root corresponds to the agglomerative approach, the opposite to the divisive one.

The dissimilarity between two groups that are candidate to be joined can be computed in several ways. Common choices are:

- single linkage (SL), when one considers the dissimilarity of the closest pair of objects, one in each group (nearest neighbours);

- complete linkage (CL), when the dissimilarity of two groups is that of the most distant objects

(furthest neighbours);

- group average (GA), when the average of the dissimilarities (calculate on all object pairs) is considered.

Very roughly, the first measures leads to clusters that are less compact, while CL produces relatively small clusters, sometimes assigning objects violating a closeness criterion. The third one represents a compromise of the two.

6.3.4 Affinity propagation

An innovative approach to data clustering has been recently introduced by Frey and Dueck [57]. Their algorithm is an implementation of a message passing technique that considers all objects as potential *exemplars* (i.e. representatives of the cluster) and starts exchanging messages among them. Without going much into detail, we only report here the basic quantities involved. The input is a matrix of similarity $s(i,k)$ giving the similarity between object i and object k. The diagonal elements $s(i,i)$ are setted to a value that determines the number of clusters obtained at the end. There are two kind of messages:

- responsibility $r(i \to k)$, indicating how good an exemplar k would be for i,

- availability $a(k \to i)$, giving how correct would be if k chose i as its exemplar.

By iterating two update equations (not reported here) until convergence one ends up with a set of messages. For every point i the value k that maximizes $a(k \to i) + r(i \to k)$ identifies the k to which i belongs or (if $i = k$) states that i as an exemplar. A peculiarity of this algorithm is that it explicitly shows exemplars: objects representing their cluster in the most "informative" way.

6.3.5 Flame: fuzzy clustering

A novel algorithm presented by Fu and Medico [58] has a couple of remarkable feature that make it different from those previously introduced. The algorithm starts identifying clustering supporting objects (CSO), points whose attributes make them "archetypal". Then it assigns to every object in the set a membership vector, obtained by the membership of its neighbours. The membership of a CSO is completely polarized to itself, while the others are fuzzy in the sense that they belong

with a certain probability to more than a cluster. This membership is propagated through the points, combining the influence of the neighbours at each step. At the end of the process one can chose a single membership (assigning an object to the cluster with the highest value), or retain a fuzzy information, assigning an object to all clusters exceeding a certain threshold.

The advantage of the algorithm is that, contrarily to most of the competitirs, it does not assume spherical clusters. The snag might be in the fact that it choses the CSOs as the points with the highest density of neighbours. of the CSO as the most dense parts of the data set.

6.4 Likelihood based clustering

In a statistical mechanics approach introduced by Giada and Marsili [59, 60], the clustering is seen as a maximum likelihood instance. Rather than writing down a Hamiltonian, an *Ansatz* on the underlying structure is given, then, through the maximum likelihood principle, the energy function is derived. We will refer to it as LBC (likelihood based clustering).

6.4.1 The model

We consider the data set $\Xi = \{\vec{\xi}_i\}_{i=1}^N$, composed of N vectors $\vec{\xi}_i = \{\xi_i(d)\}_{d=1}^D$, with D attributes. It is quite standard in clustering techniques that these vectors are transformed such that they have zero mean $\sum_d \xi_i(d)/D = 0$ and unit variance $\sum_d \xi_i^2(d)/D = 1$. $\xi_i(d)$ are assumed to be Gaussian variables. The correlation matrix for this vectors is customarily defined as

$$C_{i,j}(D) \equiv \frac{1}{D}\sum_{d=1}^{D} \xi_i(d)\xi_j(d) \quad . \tag{6.1}$$

The pairwise correlations $C_{i,j}$ will be the main quantities in this treatment.

The model assumes that the vectors $\xi_i(d)$ are generated by

$$\xi_i(d) = \frac{\sqrt{g_{s_i}}\eta_{s_i}(d) + \varepsilon_i(d)}{\sqrt{1+g_{s_i}}} \quad . \tag{6.2}$$

where $g_s > 0$ and s_i are integer variables, $\eta_s(d)$ and $\varepsilon_i(d)$ are *iid* Gaussian variables with zero average and unit variance. The equation 6.2 is explained as follows. Objects i and j belonging to the same cluster ($s_i = s_j$) are correlated ($C_{i,j} \approx g_s/(1+g_s)$), while objects in different clusters are uncorrelated.

Moreover, cluster s is populated by n_s objects and has an *internal correlation* c_s, defined as

$$n_s = \sum_{i=1}^{N} \delta_{s_i,s}, \qquad c_s = \sum_{i,j=1}^{N} C_{i,j}\delta_{s_i,s}\delta_{s_j,s} \quad . \tag{6.3}$$

Note the internal correlation is such that $0 \leq c_s \leq n_s^2$. $\mathcal{S} = \{s_i\}_{i=1}^{N}$ describes the structure of clusters (which cluster the object i belongs to), whereas the parameters $\mathcal{G} \equiv \{g_s\}_{s=1}^{N}$ set the strength of the correlations in the cluster. When g_s is small, then equation 6.2 is dominated by the noise ε and the internal correlation is low. As a consequence the cluster is less compact. When g_s is high the objects are close to the cluster center η_s and the c_s is high. Then the cluster is more compact.

The parameters g_s can be fitted using a maximum likelihood approach. It is found that, for a given structure $\mathcal{S}$, the probability of observing the data Ξ,

$$P(\Xi|\mathcal{S},\mathcal{G}) = \prod_{d=1}^{D}\left\langle \prod_{i=1}^{N}\delta\left(\xi_i(d) - \frac{\sqrt{g_{s_i}}\eta_{s_i}(d)+\varepsilon_i(d)}{\sqrt{1+g_{s_i}}}\right)\right\rangle \tag{6.4}$$

is maximal when $g_s = \hat{g}_s$, where

$$\hat{g}_s = \sqrt{\frac{c_s - n_s}{n_s^2 - n_s}} \tag{6.5}$$

Note that this equation holds for $n_s > 1$, one has to consider $\hat{g}_s = 0$ if $n_s \leq 1$.

The maximum likelihood of structure $\mathcal{S}$ can be written as $P(\hat{\mathcal{G}},\ \mathcal{S}|\Xi) \propto e^{D\mathcal{L}_c(\mathcal{S})}$, where the log-likelihood per feature $\mathcal{L}_c$ is given by

$$\mathcal{L}_c(\mathcal{S}) = \frac{1}{2}\sum_{s:\ n_s>1}\left[\log\frac{n_s}{c_s} + (n_s-1)\log\frac{n_s^2-n_s}{n_s^2-c_s}\right] \quad . \tag{6.6}$$

The original data enter the above equation through the internal correlations c_s. The maximum likelihood cluster structure, provided by the function $\mathcal{L}_c$, is that corresponding to its maximum.

Some of the features of $\mathcal{L}_c$ are summarized here:

- $\mathcal{L}_c = 0$ if the objects are unrelated ($\hat{g}_s = 0$ or $c_s = n_s$) or if they are assigned to singleton clusters ($n_s = 1$ for all s), in other words, $\max_{\mathcal{S}} \mathcal{L}_c(\mathcal{S})$ measures the amount of structure present in the data-set;

- the maxima of $\mathcal{L}_c$ do not necessarily coincide with a single cluster containing all objects, nor with the configuration with all objects in different clusters (as for K-means);

- $\mathcal{L}_c$ does not depend on any parameter, nor on any choice of the number of clusters.

Equation 6.2 may not be the most appropriate to describe a particular data set, as there is no "perfect" clustering algorithm. But, given the model, maximum likelihood principle allows us to compute the coefficients and then to have a statistical measure of the goodness of fit.

It is expected that the statistical hypothesis equation 6.2 is particularly helpful for high dimensional data sets ($D \gg 1$)[3].

6.4.2 Algorithms

Given the likelihood function above, it is a separate task to develop an algorithm capable of finding a structure maximizing it, given a certain data set. In [59] three options are proposed, a deterministic maximization, a merging algorithm, and simulated annealing with cost function $-\mathcal{L}_c$.

Merging consists in an agglomerative hierarchical clustering, where one repeatedly joins clusters giving rise to the highest increase in the likelihood. As explained above, one starts wih N objects and, after $N-1$ steps, ends up with a single cluster. Beside this one obtains a dendrogram.

Simulated annealing provides us with a maximum that, if the annealing schedule is slow enough, has a good chance to be the optimal maximum. Nevertheless, one has to explore configurations differing in the number of clusters and then, for each number, different assignment of point to clusters.

6.5 Applications to artificial and biological data sets.

6.5.1 Toward message passing

The affinity propagation algorithm described above is the first example of a message passing algorithm for clustering, notable also because of its great computing speed. Other fast implementations of clustering algorithms exist. Just to give an example of a publicly available tool, one might think of the K-means package available for the programming language R. But a message passing technique, if properly designed, is more likely to a give a global minimum. We want to explore the possibility of writing the LBC in terms of a spin Hamiltonian and then design a message passing procedure able to give the ground state. Before doing so, we proceed in a deeper characterization of the method,

[3]The dimensionality D of the data set enters only in the calculation of the matrix $C_{i,j}$, then influencing the computation only in the first steps of the algorithm.

proposing a new method to identify the number of cluster that best describes the data, and test it on real examples. Here below we report some preliminary results [61].

6.5.2 The elbow criterion revisited: curvature

Estimating the number of clusters is typically very difficult, and no universal rule exist for the best choice. Contrarily to what one may think, choosing the size corresponding to the lowest error (as in the K-means), or the maximum likelihood (as in LBC), is typically a poor choice. It suffices to remember that the lowest error solution for K-means is the one made of all singleton clusters. A commonly used criterion to decide the size of the partition that better describes a data set is the so called *elbow criterion*. It consists in plotting a "quality measure" of the clustering versus the number of clusters, and then looking for an elbow (or a kink) in this curve. For example, in the K-means one plots the sum of distances objects-centroids and then (possibly) pinpoints the kink. It is not always easy to identify a well distinct elbow in the graph, making this search very tricky.

In our approach we have decided to explore the possibility to use the curvature of the likelihood curve as an aim for the right number of clusters.

Artificial datasets

Some results are illustrated in figure 6.1.

We infer the set of parameters $\mathcal{G}$ extracted by a real biological data set made of 2668 objects with 173 features at a number of clusters N_c equal to 10, 20, 30, and 40. Data sets with the corresponding number of clusters are then generated. Each of this data set is generated with a number of attributes L ranging from N_c to a maximum of $4N_c$.

We plot the curvature of the likelihood curve obtained on these artificial data sets. Two features appear:

- the signal become clearer (higher and less noisy values of the curvature) as one increases the attributes;

- the curvature concentrates more on the right value of clusters when the dataset is made of a larger number of clusters.

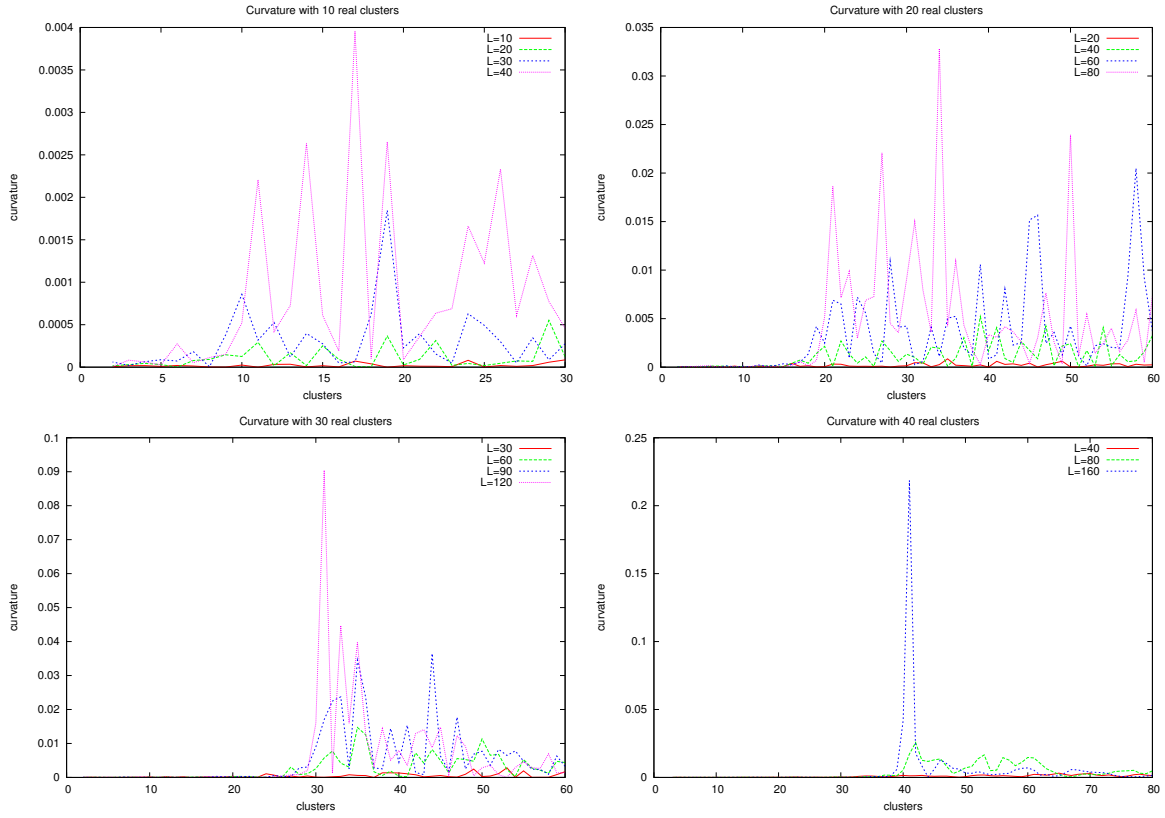

Figure 6.1: Curvature on artificial datasets

A real example

As a real example the reduced peripheral blood monocyte (RPBM) data set was chosen. This is a reduced version [62] of a data set originally presented in [63]. It consists of 235 cDNAs (objects) identified by 139 oligos (attributes). They correspond to 18 different genes, so a well designed clustering algorithm should identify 18 subsets of cDNAs co-expressed. Though the result is not perfect, some structure starts to emerge at ≈ 20 clusters.

6.5.3 Simulated annealing: an improvement

The merging algorithm (agglomerative hierarchical) is not guaranteed at all to find the minimum. But once identified the region of interest, one might invoke another optimization technique as simulated annealing (SA) and explore better the space of solutions. We report here some preliminary result on the RPBM data set.

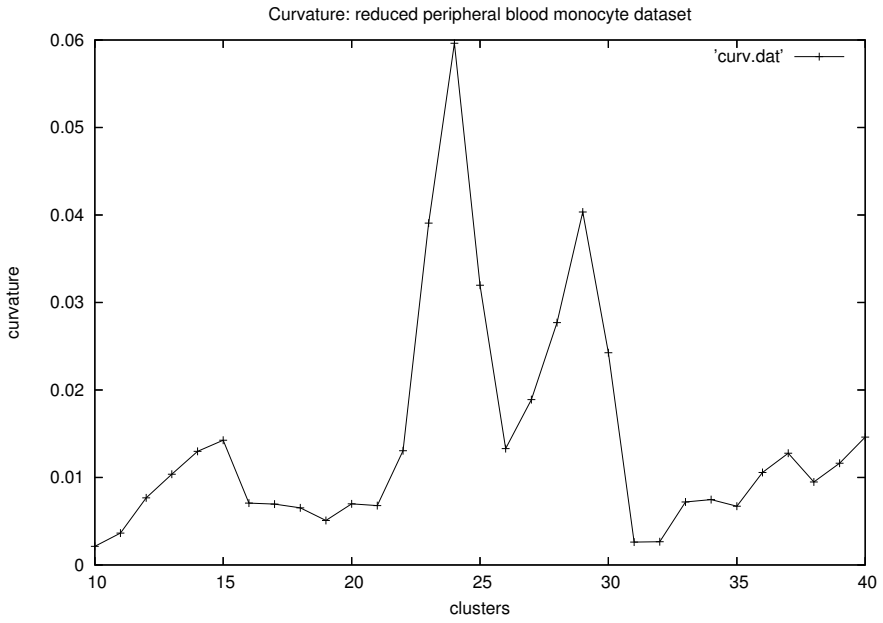

Figure 6.2: Curvature for RPBM data set

In figure 6.3 we report the energy (i.e. $-\mathcal{L}_c$) obtained by the merging alone and with the subsequent annealing. It is interesting to note that the improvement is obtained especially close to the region identified by the curvature criterion. Below and above such region no significant improvement is observed.

On such a data set we also have access to the real solution of the problem, i.e. the partitioning of the cDNAs to the 18 genes. Then we can refer to an external validation, directly comparing our solution with the biological knowledge. The overlap of our solution with the biological one is measured in terms of the *adjusted Rand index* (RI). Given two partitionings of a set of objects (even in different number of subsets) it measures their agreement. It is 1 for perfectly identical solution, zero if their completely uncorrelated, -1 if they are anticorrelated. In figure 6.4 the results for the Rand index are shown. Though not impressively high, the RI of our solution is comparable to what is found with other methods. In [62] results from different implementations of K-means are reported. There it is shown that, when used alone, it finds solution that range from 0.38 to 0.51.

Moreover, we report the curvature before and after the annealing in figure 6.5. The data before and after the annealing are quite coherent in identifying a region not far from the correct solution

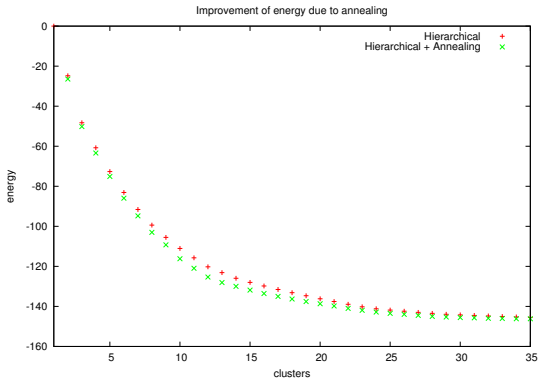

Figure 6.3: Improvement of the energy for RPBM data set due to annealing on hierarchical results.

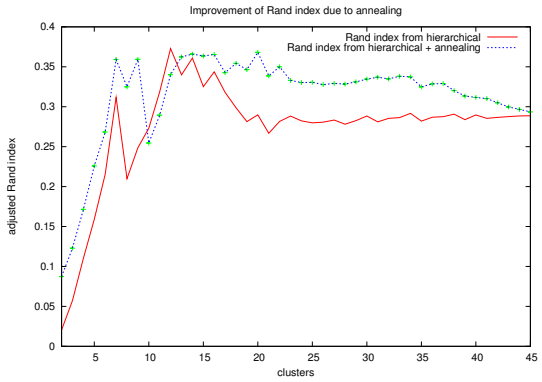

Figure 6.4: Improvement of the Rand index for RPBM data set due to annealing on hierarchical results.

(18 clusters).

In figure 6.6 we report the energy before and after the annealing for the whole peripheral blood monocyte (PBM) data set. It consist of 2329 objects (obviously specified by 139 attributes as the RPBM). The improvement of the energy is even more evident, but it must also be said that the computational effort for such a large data set is much larger.

6.6 Conclusions and future work

The results presented above show the performance of LBC on real data sets and how different algorithms implementing the same method lead to results of different quality. As there is no perfect clustering algorithm we expect to find cases in which it will be less performing than competitors, as

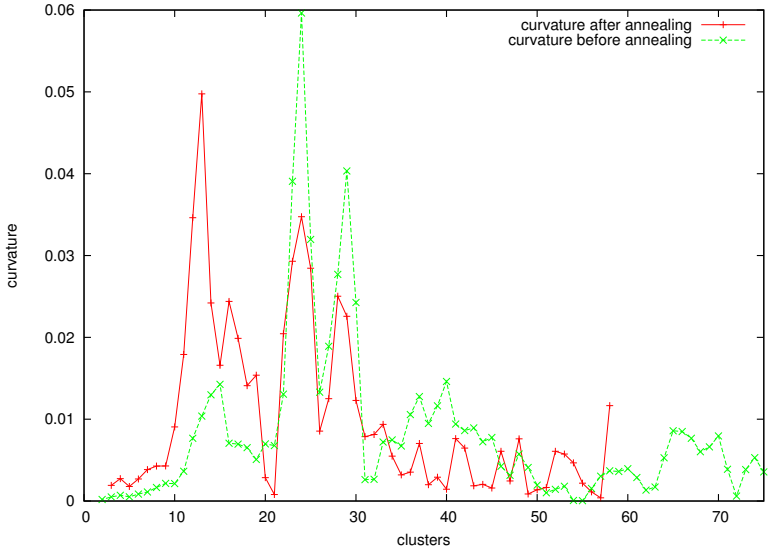

Figure 6.5: Curvature before and after the annealing. Though the data are quite noisy, some structure emerges between 20 and 30 before the annealing and between 10 and 25 after. The correct solution is at $N_c = 18$ clusters.

well as cases where the opposite holds. The fact that simulated annealing generally outperforms the hierarchical approach motivates us to go toward the design of an efficient message passing procedure able to find quickly the maximum likelihood configurations.

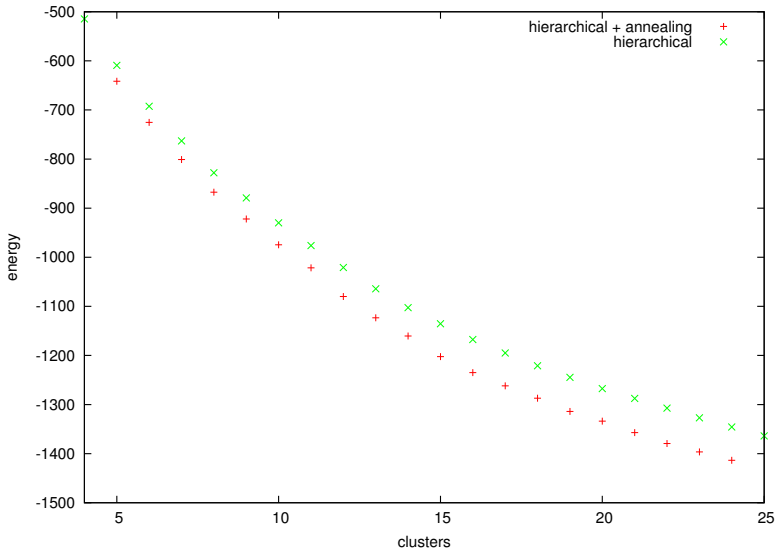

Figure 6.6: Improvement of the energy for PBM data set due to annealing on hierarchical results.

Conclusions

We have presented here a few examples of how the recent algorithmic techniques derived from statistical physics may be applied to computational biology.

The first three chapters should provide the reader with some basic understanding of the path behind this research. In particular we hope that briefly discussing the connection between combinatorial optimization and statistical inference is useful in understanding the tools described in the rest of the work.

These tools have given a profound insight to many combinatorial problems, among which we have cited random Boolean networks. These systems, after many years during which they have received attention only because of their dynamical properties, have now revealed an interesting and rich statics. This is mainly due to the application of the SP algorithm and other random graphs techniques. The analysis of small instances has found belief propagation once again able to give quite a reliable information on the marginals in small dense graphs with loops, a very demanding test for an algorithm designed to be exact on tree-like graphs. Moreover, this analysis has provided an example (one of the few) of the study of finite size corrections for a dilute disordered system. Going back to the dynamics, one might expect that characterizing the organization of fixed points might now push the study of the dynamics of such systems into new directions.

The haplotype inference problem represents a case where the experimental data are typically incomplete, and one tries to reconstruct the biologically relevant information resorting to computational methods. Unfortunately, this is also a case where we have found strong difficulties in making the algorithm converge. A different approach, for example trying to consider Bayesian approaches to the inference, might lead to a different formulation where inference methods could prove useful.

The last issue addressed, data clustering, is a combinatorial problem of great interest in many

tasks of computational biology. By superimposing an annealing procedure on the pure hierarchical approach we have shown that there is still space to improve the energy (at cost of losing the dendrogram). Even more important, we have found (at least in one real data set) an improvement of an external validation criterion as Rand index, measuring the overlap with the real biological solution. It is not surprising that hierarchical algorithms perform worse than a simulated annealing in minimizing a function, but it is encouraging that an external validation measure imporves as the energy decreases. This strongly prompts us to develop a message passing procedure capabel of finding the global minima of the energy more quickly and reliably. Last, the popular elbow criterion has been revisited in terms of an easily accessible quantity as curvature.

After the more theoretical framework that has been established in the last few years at the interface of statistical physics and computation, it is now to be expected that many applications will come. Probably, a great part of these will lie at the interface with biology.

Appendix A

The computation of the moments of $P(\mathcal{N}_{sol})$

We first present the details of the computations of the second moment and we will then give some hints on the structure of the generic n^{th} moment.

A.1 The second order momentum

With respect to $\langle \mathcal{N}_{sol}^2 \rangle$, if we keep only the leading order terms in eq.(4.25), the sums can be approximated by the following integral, where the exponent is given by the entropy contribution of the

external variables plus a Kullback-Leibler distance term vanishing at the saddle point:

$$
\begin{aligned}
\langle \mathcal{N}_{sol}^2 \rangle = I^{(2)}(\alpha, X) &= 2^{(1-\alpha)N} \int_0^1 \int_0^1 db_o^+ db_i^+ K(\alpha, X, b_o^+, b_i^+) e^{N\mathcal{F}[\alpha, X, b_o^+, b_i^+]} \\
K(\alpha, X, b_o^+, b_i^+) &= \frac{\sqrt{\alpha(1-\alpha)}}{\sqrt{b_o^+ b_i^+ (1-b_o^+)(1-b_i^+)}} e^{\alpha B(\alpha, X, b_o^+, b_i^+)} \\
\mathcal{F}[\alpha, X, b_o^+, b_i^+] &= (1-\alpha) H(b_i^+) + \alpha D_{KL}(b_o^+ | G(\alpha, X, b_o^+, b_i^+)) \\
B(\alpha, X, b_o^+, b_i^+) &= 3 - \frac{3b^+ b_o^+ + (1-X)(1-b^+) b_o^+ + (1+X)(1-b^+) b_o^+/2}{(b^+)^2 + (1-X)(1-b^+) b_o^+ + (1+X)(1-b^+)^2/2} \\
&- \frac{(1+X)(b^+(1-b_o^+) + 3(1-X)(1-b^+)(1-b_o^+)/2}{1-(b^+)^2 + (1-X)(1-b^+) b_o^+ - (1+X)(1-b^+)^2/2} \\
H(x) &= -x\log(x) - (1-x)\log(1-x) \\
D_{KL}(x|y) &= x\log\left(\frac{y}{x}\right) + (1-x)\log\left(\frac{1-y}{1-x}\right) \\
G(\alpha, X, b_o^+, b_i^+) &= (b^+)^2 + (1-X) b^+ (1-b^+) + \frac{1+X}{2}(1-b^+)^2 \\
b^+(\alpha, b_o^+, b_i^+) &= \alpha b_o^+ + (1-\alpha) b_i^+
\end{aligned}
$$

where $b_o^+ = N_o^+/N_o$ and $b_i^+ = N_i^+/N_o$. The value of the Integral $I^{(2)}(\alpha, X)$ can be calculated at the saddle point

$$
\begin{aligned}
\tilde{b}_i^+ &= \frac{1}{2} && \text{(A.1)} \\
\tilde{b}_o^+ &= G(\alpha, X, \tilde{b}_o^+, \tilde{b}_i^+) && \text{(A.2)} \\
&= \frac{2 - \alpha(1-\alpha) + \alpha X(1+3\alpha) - 2\sqrt{1 - \alpha(1-X) + \alpha^2 X(X-1)}}{2\alpha^2(1+3X)}
\end{aligned}
$$

Finite N corrections can be in principle computed extending the calculation of K to higher orders in $O(1/N)$, and performing an asymptotic expansion around the saddle point. Whenever $\alpha < 1$ and $X > 0$ it can be seen that the condition (A.3) finds the only maximum $\mathcal{F}$, which lies within the integration interval.

A.2 An upper bound on the number of solutions

In this subsection we will show that in the thermodynamic limit the support of $P(\mathcal{N}_{sol})$ is contained in the interval $(0, \langle \mathcal{N}_{sol} \rangle)$, using, as we have already shown in the previous section, that:

$$\begin{aligned} \langle \mathcal{N}_{sol} \rangle &= 2^{(1-\alpha)N} \\ \langle \mathcal{N}_{sol}^2 \rangle &= 2^{2(1-\alpha)N} C(\alpha, X) \end{aligned} \tag{A.3}$$

where $C(\alpha, X)$ is a constant independent from N. The one-tailed Chebyshev inequality states that:

$$\Pr\left(\mathcal{N}_{sol} > \langle \mathcal{N}_{sol} \rangle\right) \leq \frac{\langle \mathcal{N}_{sol}^2 \rangle}{(\mathcal{N}_{sol} - \langle \mathcal{N}_{sol} \rangle)^2} \ . \tag{A.4}$$

Under the condition $\mathcal{N}_{sol} > \langle \mathcal{N}_{sol} \rangle$ we can express $\mathcal{N}_{sol} = 2^{N\Sigma'}$ where $\Sigma' > (1-\alpha)$. Inserting eqs. (A.3) into eq. (A.4) we get:

$$\begin{aligned} \Pr\left(\mathcal{N}_{sol} > \langle \mathcal{N}_{sol} \rangle\right) &\leq \frac{2^{2(1-\alpha)N} C(\alpha, X)}{2^{2N\Sigma'}\left(1 - 2^{N(1-\alpha-\Sigma')}\right)} \leq 2^{2N(1-\alpha-\Sigma')} C(\alpha, X) \\ &= 2^{-N\gamma} C(\alpha, X) \ , \end{aligned} \tag{A.5}$$

where γ is a positive constant making the right tail of the $P(\mathcal{N}_{sol})$ distribution (i.e. for values of $\mathcal{N}_{sol} > \langle \mathcal{N}_{sol} \rangle$) exponentially small in N, as we wanted to show. Indeed, this simple result is enough to imply that no contribution to the entropy is given by instances whose number of solutions is larger than the annealed value. The support of the probability distribution of entropy values $P(S)$ must be therefore $[0, S_{annealed}]$. In order to prove that also smaller values do not take part of the support, one would need to calculate fractional order moments, as explained in the end of next section.

A.3 Higher order moments

For the general n^{th} order momentum one can write, along the same line of reasoning:

$$\begin{aligned}\langle \mathcal{N}_{sol}^n \rangle &= \sum_{\{N_o^{\sigma_1\dots\sigma_{n-1}}\}=0}^{N_o} \sum_{\{N_i^{\sigma_1\dots\sigma_{n-1}}\}=0}^{N_i} \frac{N_o!N_i!}{\prod_{\vec{\sigma}} N_o^{\sigma_1\dots\sigma_{n-1}}!N_i^{\sigma_1\dots\sigma_{n-1}}!}\cdot \\ &\delta(N_o;\alpha N)\delta(N_i;(1-\alpha)N)\prod_{\vec{\sigma}} T(\vec{\sigma})\cdot \\ &\delta(N_o;\sum_{\vec{\sigma}} N_o^{\sigma_1\dots\sigma_{n-1}})\delta(N_i;\sum_{\vec{\sigma}} N_i^{\sigma_1\dots\sigma_{n-1}}) \end{aligned} \tag{A.6}$$

with

$$\begin{aligned} T(\vec{\sigma}) &= \Bigg[\sum_{\vec{\sigma^{(1)}}}\sum_{\vec{\sigma^{(2)}}} \mathcal{P}(\sigma_1^{(1)}\sigma_1^{(2)},\dots,\sigma_{n-1}^{(1)}\sigma_{n-1}^{(2)}|\sigma_1,\dots,\sigma_{n-1})\cdot \\ &\quad g(\sigma_1^{(1)}\sigma_1^{(2)},\dots,\sigma_{n-1}^{(1)}\sigma_{n-1}^{(2)}|\sigma_1,\dots,\sigma_{n-1})\Bigg]^{N_o^{\sigma_1\dots\sigma_{n-1}}} \end{aligned} \tag{A.7}$$

where we are summing over all configurations overlaps with $N_o^{\sigma_1\dots\sigma_{n-1}}/N_i^{\sigma_1\dots\sigma_{n-1}}$ output/input variables of signs $\sigma_1\dots\sigma_{n-1}$, the $\mathcal{P}$ represent the probability of finding $n-1$ real replicas of an input variables couple in a given function, and g the value of the $(n-1)^{th}$ averaged product of the replicated Boolean function.

As before, from the leading terms of eq.(A.6), one gets

$$\begin{aligned}\langle \mathcal{N}_{sol}^n \rangle &\rightarrow I^{(n)}(\alpha,X) \\ I^{(n)}(\alpha,X) &= 2^{(1-\alpha)N}\int_0^1 \prod_{\vec{\sigma}} d\{b_o\}d\{b_i\}K^{(n)}(\alpha,X,\{b_o,b_i\})e^{N\mathcal{F}^{(n)}[\alpha,X,\{b_o,b_i\}]}\end{aligned}$$

with

$$\begin{aligned}\mathcal{F}^{(n)}[\alpha,X,\{b_o,b_i\}] &= (1-\alpha)H^{(n)}+\alpha D_{KL}^{(n)}(\{b_o\}|G^{(n)}) \\ H^{(n)}(\alpha,X,\{b_o,b_i\}) &= \sum_{\vec{\sigma}} b_i^{\sigma_1\dots\sigma_{n-1}}\log(b_i^{\sigma_1\dots\sigma_{n-1}}) \\ D_{KL}^{(n)}(\{b_o\}|G^{(n)}) &= \sum_{\vec{\sigma}} b_o^{\sigma_1\dots\sigma n-1}\log\left(\frac{G^{(n)}}{b_0^{\sigma_1\dots\sigma n-1}}\right)\end{aligned} \tag{A.8}$$

with $b_{o,i}^{\sigma_1\dots\sigma_{n-1}} = N_{o,i}^{\sigma_1\dots\sigma_{n-1}}/N_{o,i}$ and $\{b_{o,i}\} = \{b_{o,i}^{\sigma_1,\dots,\sigma_{n-1}}\}_{\vec{\sigma}\in\{\pm1\}^{n-1}}$. The explicit form of functions

$G^{(n)}$ and $K^{(n)}$ is not given here for brevity. Due to the general form of the exponent, the unique saddle point can always be computed as the one fulfilling conditions:

$$\begin{aligned} \tilde{b}_i^{\sigma_1 \ldots \sigma_{n-1}} &= \frac{1}{2^{n-1}} \\ \tilde{b}_o^{\sigma_1 \ldots \sigma_{n-1}} &= G^{(n)}(\alpha, X, \{\tilde{b}_o^{\sigma_1 \ldots \sigma_{n-1}}\}, \{\tilde{b}_i^{\sigma_1 \ldots \sigma_{n-1}}\}) \end{aligned} \tag{A.9}$$

such that

$$I^{(n)}(\alpha, X) = 2^{n(1-\alpha)N} C^{(n)}(\alpha, X) = \langle \mathcal{N}_{sol} \rangle^n C^{(n)}(\alpha, X) \tag{A.10}$$

Unfortunately, the explicit calculation of the function $C^{(n)}(\alpha, X)$ for all n seems not to be easily accessible. Let us just finally note that one could prove that the distribution of the logarithm of the number of solutions $P(S)$ tends to $\delta(S - S_{\text{annealed}})$ in the large N one by showing that:

$$\lim_{n \to 0} \lim_{N \to \infty} C^{(n)}(\alpha, X) = 1 \tag{A.11}$$

which seems reasonable given the structure of eqs. (A.8, A.9, A.10), and the numerical result displayed in section 4.6.2, but, again, too difficult to be shown explicitly.

Bibliography

[1] Mézard M, Ricci-Tersenghi F, Zecchina R (2003) Alternative solutions to diluted p-spin models and xorsat problems. J STAT PHYS 111:505. URL http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0207140.

[2] Garey M, Johnson D (1979) Computers and Intractability. Freeman and Co., New York.

[3] Franz S, Leone M (2003) Replica bounds for optimization problems and diluted spin systems. Journal of Statistical Physics 111:535–564.

[4] Kirkpatrick S, Selman B (1994) Critical behavior in the satisfiability of random boolean expressions. Science 264:1297–1301. URL http://links.jstor.org/sici?sici=0036-8075%252819940527%25293%253A264%253A5163%253C1297%253ACBITSO%253E2.0.CO%253B2-V.

[5] Monasson R, Zecchina R (1997) Statistical mechanics of the random k-satisfiability model. Physical Review E 56:1357–1370.

[6] Monasson R, Zecchina R (1996) Entropy of the k-satisfiability problem. Physical Review Letters 76:3881–3885.

[7] Monasson R, Zecchina R, Kirkpatrick S, Selman B, Troyansky L (1999) Determining computational complexity from characteristic 'phase transitions'. Nature 400:133–137.

[8] Mézard M, Parisi G, Zecchina R (2002) Analytic and algorithmic solution of random satisfiability problems. Science 297:812–5. doi:10.1126/science.1073287. URL http://www.sciencemag.org/cgi/content/abstract/297/5582/812.

[9] Mézard M, Zecchina R (2002) Random k-satisfiability problem: from an analytic solution to an efficient algorithm. Physical review E, Statistical, nonlinear, and soft matter physics 66:056126. URL http://prola.aps.org/abstract/PRE/v66/i5/e056126.

[10] Montanari A, Parisi G, Ricci-Tersenghi F (2004) Instability of one-step replica-symmetry-broken phase in satisfiability problems. Journal of Physics A Mathematical and General 37:2073–2091.

[11] Achlioptas D, Naor A, Peres Y (2005) Rigorous location of phase transitions in hard optimization problems. Nature 435:759–764. URL http://dx.doi.org/10.1038/nature03602.

[12] Weigt M, Hartmann AK (2000) Number of guards needed by a museum: a phase transition in vertex covering of random graphs. Phys Rev Lett 84:6118–21.

[13] Mulet R, Pagnani A, Weigt M, Zecchina R (2002) Coloring random graphs. Phys Rev Lett 89:268701.

[14] Lauritzen S, Spiegelhalter D (1988) Local computations with probabilities on graphical structures and their applications to expert systems. J Royal Stat Soc B 50:157–224.

[15] Yedidia J, Freeman W, Weiss Y (2001) Understanding belief propagation and its generalizations. Technical Report TR-2001-22, Mitsubishi Electric Research Laboratories. URL http://www.merl.com/publications/TR2001-022/. Http://www.merl.com/publications/TR2001-022/.

[16] Needham CJ, Bradford JR, Bulpitt AJ, Westhead DR (2006) Inference in bayesian networks. Nat Biotech 24:51–53. URL http://dx.doi.org/10.1038/nbt0106-51.

[17] Mezard M, Parisi G, Virasoro M (1987) Spin glass theory and beyond. World Scientific Teaneck, NJ, USA.

[18] Mézard M, Mora T, Zecchina R (2005) Clustering of solutions in the random satisfiability problem. Phys Rev Lett 94:197205.

[19] Cocco S, Dubois O, Mandler J, Monasson R (2003) Rigorous decimation-based construction of ground pure states for spin-glass models on random lattices. Phys Rev Lett 90:047205.

[20] Mézard M, Parisi G (2000) The bethe lattice spin glass revisited. The European Physical Journal B 20:217. URL http://it.arxiv.org/abs/cond-mat/0009418.

[21] Mézard M, Parisi G (2003) The cavity method at zero temperature. Journal of Statistical Physics 111:1–34.

[22] MacKay DJC (2003) Information Theory, Inference, and Learning Algorithms, Cambridge University Press, chapter 16. pp. 240–247. URL http://www.inference.phy.cam.ac.uk/mackay/itila/.

[23] MacKay D (2003) Information Theory, Inference and Learning Algorithms., Cambridge University Pr., chapter 47. pp. 555–573.

[24] Krzakal F, Montanari A, Ricci-Tersenghi F, Semerjian G, Zdeborová L (2007) Gibbs states and the set of solutions of random constraint satisfaction problems. Proc Natl Acad Sci USA 104:10318–23. doi:10.1073/pnas.0703685104. URL http://www.pnas.org/cgi/content/abstract/104/25/10318.

[25] Braunstein A, Mezard M, Zecchina R (2005) Survey propagation: an algorithm for satisfiability. Random Structures and Algorithms 27:201–226.

[26] Battaglia D, Kolár M, Zecchina R (2004) Minimizing energy below the glass thresholds. Physical review E, Statistical, nonlinear, and soft matter physics 70:036107.

[27] Kauffman SA (1969) Metabolic stability and epigenesis in randomly constructed genetic nets. J Theor Biol 22:437–467.

[28] Dougherty ER, Shmulevich I, Chen J, Wang ZJ, editors (2005) GENOMIC SIGNAL PROCESSING AND STATISTICS. Hindawi Publishing Corporation.

[29] Correale L, Leone M, Pagnani A, Weigt M, Zecchina R (2006) The computational core and fixed point organization in boolean networks. Journal of Statistical Mechanics: Theory and Experiment 2006:P03002. URL http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0512089.

[30] Correale L, Leone M, Pagnani A, Weigt M, Zecchina R (2006) Core percolation and onset of complexity in boolean networks. Physical Review Letters 96:018101. URL http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0412443.

[31] Leone M, Pagnani A, Parisi G, Zagordi O (2006) Finite size corrections to random boolean networks. Journal of Statistical Mechanics: Theory and Experiment 2006:P12012. URL http://stacks.iop.org/1742-5468/2006/P12012.

[32] Kauffman S, Peterson C, Samuelsson B, Troein C (2003) Random boolean network models and the yeast transcriptional network. PNAS 100:14796.

[33] Kauffman S, Peterson C, Samuelsson B, Troein C (2004) Genetic networks with canalyzing boolean rules are always stable. PNAS 101:17102.

[34] URL http://www.satlib.org/solvers.html.

[35] Bayardo RJJ, Schrag RC (1997) Using csp look-back techniques to solve real world sat instances. In: Proc. of the 14th National Conf. on Artificial Intelligence. pp. 203–208. URL http://www.bayardo.org/resources.html.

[36] Morita A, Nakayama T, Doba N, Hinohara S, Mizutani T, et al. (2007) Genotyping of triallelic snps using taqman pcr. Mol Cell Probes 21:171–6. doi:10.1016/j.mcp.2006.10.005.

[37] Balding DJ (2006) A tutorial on statistical methods for population association studies. Nat Rev Genet 7:781–791. URL http://dx.doi.org/10.1038/nrg1916.

[38] Drysdale CM, McGraw DW, Stack CB, Stephens JC, Judson RS, et al. (2000) Complex promoter and coding region beta 2-adrenergic receptor haplotypes alter receptor expression and predict in vivo responsiveness. PNAS 97:10483–10488. doi:10.1073/pnas.97.19.10483. URL http://www.pnas.org/cgi/content/abstract/97/19/10483. http://www.pnas.org/cgi/reprint/97/19/10483.pdf.

[39] Clark AG (1990) Inference of haplotypes from pcr-amplified samples of diploid populations. Mol Biol Evol 7:111–122.

[40] Clark AG (2004) The role of haplotypes in candidate gene studies. Genet Epidemiol 27:321–333. doi:10.1002/gepi.20025. URL http://dx.doi.org/10.1002/gepi.20025.

[41] Gusfield D (2001) Inference of haplotypes from samples of diploid populations: Complexity and algorithms. Journal of Computational Biology 8:305–323. doi:10.1089/10665270152530863. URL http://www.liebertonline.com/doi/abs/10.1089/10665270152530863.

[42] Gusfield D (2003) Haplotype inference by pure parsimony. In: Proceedings 14th Annual Symposium Combinatorial Pattern Matching. Springer, pp. 144–155.

[43] Brown DG, Harrower IM (2006) Integer programming approaches to haplotype inference by pure parsimony. IEEE/ACM Transactions on Computational Biology and Bioinformatics 3:141–154.

[44] Halldórsson B, Bafna V, Edwards N, Lippert R, Yooseph S, et al. (2003) A survey of computational methods for determining haplotypes. In: Istrail S, Waterman MS, Clark AG, editors, Computational Methods for SNPs and Haplotype Inference: Proc. DIMACS/RECOMB Satellite Workshop. Springer, pp. 26–47.

[45] Lynce I, Marques-Silva J (2006) Efficient haplotype inference with boolean satisfiability. In: National Conference on Artificial Intelligence (AAAI), July.

[46] Lynce I, Marques-Silva J (2006) Sat in bioinformatics: Making the case with haplotype inference. In: Proceedings of International Conference on Theory and Applications of Satisfiability Testing (in press), Seattle, USA.

[47] Wang L, Xu Y (2003) Haplotype inference by maximum parsimony. Bioinformatics 19:1773–1780. doi:10.1093/bioinformatics/btg239. URL http://bioinformatics.oxfordjournals.org/cgi/content/abstract/19/14/1773. http://bioinformatics.oxfordjournals.org/cgi/reprint/19/14/1773.pdf.

[48] Hudson R (2002) Generating samples under a wright–fisher neutral model of genetic variation. Bioinformatics 18:337–338.

[49] Graca AS, Marques-Silva J, Lynce I, Oliveira A (2007) Efficient haplotype inference with pseudo-boolean optimization. In: Proceedings of Algebraic Biology (in press), Hagenberg, Austria.

[50] URL http://lpsolve.sourceforge.net/5.5/.

[51] Stephens M, Smith NJ, Donnelly P (2001) A new statistical method for haplotype reconstruction from population data. Am J Hum Genet 68:978–989.

[52] Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, et al. (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286:531–7.

[53] Spellman PT, Sherlock G, Zhang MQ, Iyer VR, Anders K, et al. (1998) Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. Mol Biol Cell 9:3273–97.

[54] Hastie T, Tibshirani R, Friedman J, et al. (2001) The elements of statistical learning: data mining, inference, and prediction. Springer.

[55] Hastie T, Tibshirani R, Friedman J, et al. (2001) The elements of statistical learning: data mining, inference, and prediction, Springer, chapter 14. pp. 480–485.

[56] Kohonen T (1990) The self-organizing map. Proceedings of the IEEE 78:1464–1480.

[57] Frey BJ, Dueck D (2007) Clustering by passing messages between data points. Science 315:972–976. URL http://www.sciencemag.org/cgi/content/abstract/315/5814/972.

[58] Fu L, Medico E (2007) Flame, a novel fuzzy clustering method for the analysis of dna microarray data. BMC Bioinformatics 8:3. doi:10.1186/1471-2105-8-3. URL http://dx.doi.org/10.1186/1471-2105-8-3.

[59] Giada L, Marsili M (2001) Data clustering and noise undressing of correlation matrices. Physical Review E 63:61101. URL http://it.arxiv.org/abs/cond-mat/0101237.

[60] Giada L, Marsili M (2002) Algorithms of maximum likelihood data clustering with applications. Physica A: Statistical Mechanics and its Applications 315:650–664. URL http://it.arxiv.org/abs/cond-mat/0204202.

[61] Bianconi G, Marsili M, Zagordi O, Zecchina R Improvement of a hierarchical clustering algorithm. In preparation.

[62] Di Gesù V, Giancarlo R, Bosco GL, Raimondi A, Scaturro D (2005) Genclust: a genetic algorithm for clustering gene expression data. BMC Bioinformatics 6:289. doi:10.1186/1471-2105-6-289. URL http://dx.doi.org/10.1186/1471-2105-6-289.

[63] Hartuv E, Schmitt AO, Lange J, Meier-Ewert S, Lehrach H, et al. (2000) An algorithm for clustering cdna fingerprints. Genomics 66:249–256. doi:10.1006/geno.2000.6187. URL http://dx.doi.org/10.1006/geno.2000.6187.

List of Figures

List of Tables