Belief Propagation guided decimation on random k-XORSAT

Arnab Chatterjee
TU Dortmund, Faculty of Computer Science, Otto-Hahn-Str. 12, 44227 Dortmund, Germany

Amin Coja-Oghlan
TU Dortmund, Faculty of Computer Science, Otto-Hahn-Str. 12, 44227 Dortmund, Germany

Mihyun Kang
TU Graz, Institute of Discrete Mathematics, Steyrergasse 30, 8010 Graz, Austria

Lena Krieg[1]
TU Dortmund, Faculty of Computer Science, Otto-Hahn-Str. 12, 44227 Dortmund, Germany

Maurice Rolvien
Universitty of Hamburg, Department of Informatics, Vogt-Kölln-Str. 30, 22527 Hamburg, Germany

Gregory B. Sorkin
The London School of Economics and Political Science, Department of Mathematics, Columbia House, Houghton St, London WC2A 2AE, United Kingdom

Abstract

We analyse the performance of *Belief Propagation Guided Decimation*, a physics-inspired message passing algorithm, on the random k-XORSAT problem. Specifically, we derive an explicit threshold up to which the algorithm succeeds with a strictly positive probability $\Omega(1)$ that we compute explicitly, but beyond which the algorithm with high probability fails to find a satisfying assignment. In addition, we analyse a thought experiment called the *decimation process* for which we identify a (non-) reconstruction and a condensation phase transition. The main results of the present work confirm physics predictions from [Ricci-Tersenghi and Semerjian: J. Stat. Mech. 2009] that link the phase transitions of the decimation process with the performance of the algorithm, and improve over partial results from a recent article [Yung: Proc. ICALP 2024].

2012 ACM Subject Classification Mathematics of computing → Probability and statistics; Mathematics of computing → Combinatoric problems; Mathematics of computing → Combinatorics; Mathematics of computing → Probabilistic algorithms

Keywords and phrases random k-XORSAT, belief propagation, decimation process, random matrices

Digital Object Identifier 10.4230/LIPIcs.ICALP.2025.20

Related Version *Full Version*: `arxiv.org/abs/2501.17657`

Funding *Amin Coja-Oghlan*: supported by DFG CO 646/3, DFG CO 646/5 and DFG CO 646/6.
Mihyun Kang: supported by Austrian Science Fund (FWF) 10.55776/I6502.
Lena Krieg: supported by DFG CO 646/3

1 Introduction and results

1.1 Background and motivation

The random k-XORSAT problem shares many characteristics of other intensely studied random constraint satisfaction problems ('CSPs') such as random k-SAT. For instance, as the clause/variable density increases, random k-XORSAT possesses a sharp satisfiability

[1] corresponding author

threshold preceded by a reconstruction or 'shattering' phase transition that affects the geometry of the set of solutions [2, 11, 16, 23]. As in random k-SAT, these transitions appear to significantly impact the performance of certain classes of algorithms [6, 15]. At the same time, random k-XORSAT is more amenable to mathematical analysis than, say, random k-SAT. This is because the XOR operation is equivalent to addition modulo two, which is why a k-XORSAT instance translates into a linear system over $\mathbb{F}_2$. In effect, k-XORSAT can be solved in polynomial time by means of Gaussian elimination. In addition, the algebraic nature of the problem induces strong symmetry properties that simplify its study [3].

Because of its similarities with other random CSPs combined with said relative amenability, random k-XORSAT provides an instructive benchmark. This was noticed not only in computer science, but also in the statistical physics community, which has been contributing intriguing 'predictions' on random CSPs since the early 2000s [18, 21]. Among other things, physicists have proposed a message passing algorithm called *Belief Propagation Guided Decimation* ('BPGD') that, according to computer experiments, performs impressively on various random CSPs [20]. Furthermore, Ricci-Tersenghi and Semerjian [24] put forward a heuristic analysis of BPGD on random k-SAT and k-XORSAT. Their heuristic analysis proceeds by way of a thought experiment based on an idealized version of the algorithm. We call this thought experiment the *decimation process*. Based on physics methods Ricci-Tersenghi and Semerjian surmise that the decimation process undergoes two phase transitions, specifically a reconstruction and a condensation transition. A key prediction of Ricci-Tersenghi and Semerjian is that these phase transitions are directly linked to the performance of the BPGD algorithm. Due to the linear algebra-induced symmetry properties, in the case of random k-XORSAT all of these conjectures come as elegant analytical expressions.

The aim of this paper is to verify the predictions from [24] on random k-XORSAT mathematically. Specifically, our aim is to rigorously analyse the BPGD algorithm on random k-XORSAT, and to establish the link between its performance and the phase transitions of the decimation process. A first step towards a rigorous analysis of BPGD on random k-XORSAT was undertaken in a recent contribution by Yung [25]. However, Yung's analysis turns out to be not tight. Specifically, apart from requiring spurious lower bounds on the clause length k, Yung's results do not quite establish the precise connection between the decimation process and the performance of BPGD. One reason for this is that [25] relies on 'annealed' techniques, i.e., essentially moment computations. Here we instead harness 'quenched' arguments that were partly developed in prior work on the rank of random matrices over finite fields [3, 8].

Throughout we let $k \geq 3$ and $n \geq k$ be integers and $d > 0$ a positive real. Let $\boldsymbol{m} \stackrel{\text{dist}}{=} \mathrm{Po}(dn/k)$ and let $\boldsymbol{F} = \boldsymbol{F}(n, d, k)$ be a random k-XORSAT formula [2] with variables $x_1, \ldots, x_n$ and $\boldsymbol{m}$ random clauses of length k. To be precise, every clause of $\boldsymbol{F}$ is an XOR of precisely k distinct variables, each of which may or may not come with a negation sign. The $\boldsymbol{m}$ clauses are drawn uniformly and independently out of the set of all $2^k\binom{n}{k}$ possibilities. Thus, d equals the average number of clauses that a given variable x_i appears in.

1.2 Belief Propagation Guided Decimation

The first result vindicates the predictions from [24] concerning the success probability of BPGD algorithm. BPGD sets its ambitions higher than merely finding a solution to the k-XORSAT instance $\boldsymbol{F}$: the algorithm attempts to sample a solution uniformly at random. To this

[2] Two random variables $\boldsymbol{X}, \boldsymbol{Y}$ are equal in distribution $\boldsymbol{X} \stackrel{\text{dist}}{=} \boldsymbol{Y}$ if they have the same distribution functions. Here, $\boldsymbol{m}$ follows a Poisson distribution with mean dn/k.

end BPGD assigns values to the variables $x_1, \dots, x_n$ of $\boldsymbol{F}$ one after the other. In order to assign the next variable the algorithm attempts to compute the marginal probability that the variable is set to 'true' under a random solution to the k-XORSAT instance, given all previous assignments. More precisely, suppose BPGD has assigned values to the variables $x_1, \dots, x_t$ already. Write $\boldsymbol{\sigma}_{\mathrm{BP}}(x_1), \dots, \boldsymbol{\sigma}_{\mathrm{BP}}(x_t) \in \{0,1\}$ for their values, with 1 representing 'true' and 0 'false'. Further, let $\boldsymbol{F}_{\mathrm{BP},t}$ be the simplified formula obtained by substituting $\boldsymbol{\sigma}_{\mathrm{BP}}(x_1), \dots, \boldsymbol{\sigma}_{\mathrm{BP}}(x_t)$ for $x_1, \dots, x_t$. We drop any clauses from $\boldsymbol{F}_{\mathrm{BP},t}$ that contain variables from $\{x_1, \dots, x_t\}$ only, deeming any such clauses satisfied. Thus, $\boldsymbol{F}_{\mathrm{BP},t}$ is a XORSAT formula with variables $x_{t+1}, \dots, x_n$. Its clauses contain at least one and at most k variables, as well as possibly a constant (the XOR of the values substituted in for $x_1, \dots, x_t$).

Let $\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{BP},t}}$ be a uniformly random solution of the XORSAT formula $\boldsymbol{F}_{\mathrm{BP},t}$, assuming that $\boldsymbol{F}_{\mathrm{BP},t}$ remains satisfiable. Then BPGD aims to compute the marginal probability $\mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{BP},t}}(x_{t+1}) = 1 \mid \boldsymbol{F}_{\mathrm{BP},t}\right]$ that a random satisfying assignment of $\boldsymbol{F}_{\mathrm{BP},t}$ sets x_{t+1} to true. This is where Belief Propagation ('BP') comes in. An efficient message passing heuristic for computing precisely such marginals, BP returns an 'approximation' $\mu_{\boldsymbol{F}_{\mathrm{BP},t}}$ of $\mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{BP},t}}(x_{t+1}) = 1 \mid \boldsymbol{F}_{\mathrm{BP},t}\right]$. We will recap the mechanics of BP in Section 2.2 (the value $\mu_{\boldsymbol{F}_{\mathrm{BP},t}}$ is defined precisely in (2.9)). Having computed the BP 'approximation', BPGD proceeds to assign x_{t+1} the value 'true' with probability $\mu_{\boldsymbol{F}_{\mathrm{BP},t}}$, otherwise sets x_{t+1} to 'false', then moves on to the next variable. The pseudocode is displayed as Algorithm 1.

Algorithm 1 The BPGD algorithm.

Data: a random k-XORSAT formula $\boldsymbol{F}$ with variables $x_1, \dots, x_n$ conditioned on being satisfiable

for $t = 0, \dots, n-1$ **do**

  compute the BP approximation $\mu_{\boldsymbol{F}_{\mathrm{BP},t}}$;

  set $\boldsymbol{\sigma}_{\mathrm{BP}}(x_{t+1}) = \begin{cases} 1 & \text{with probability } \mu_{\boldsymbol{F}_{\mathrm{BP},t}} \\ 0 & \text{with probability } 1 - \mu_{\boldsymbol{F}_{\mathrm{BP},t}} \end{cases}$;

return $\boldsymbol{\sigma}_{\mathrm{BP}}$;

Let us pause for a few remarks. First, if the BP approximations are exact, i.e., if $\boldsymbol{F}_{\mathrm{BP},t}$ is satisfiable and $\mu_{\boldsymbol{F}_{\mathrm{BP},t}} = \mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{BP},t}}(x_{t+1}) = 1 \mid \boldsymbol{F}_{\mathrm{BP},t}\right]$ for all t, then Bayes' formula shows that BPGD outputs a uniformly random solution of $\boldsymbol{F}$. However, there is no universal guarantee that BP returns the correct marginals. Accordingly, the crux of analysing BPGD is precisely to figure out whether this is the case. Indeed, the heuristic work of [24] ties the accuracy of BP to a phase transition of the decimation process thought experiment, to be reviewed momentarily.

Second, the strategy behind the BPGD algorithm, particularly the message passing heuristic for 'approximating' the marginals, generalizes well beyond k-XORSAT. For instance, the approach applies to k-SAT verbatim. That said, due to the algebraic nature of the XOR operation, BPGD is *far* easier to analyse on k-XORSAT. In fact, in XORSAT the marginal probabilities are guaranteed to be half-integral as seen in Fact 6, i.e.,

$$\mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{BP},t}}(x_{t+1}) = 1 \mid \boldsymbol{F}_{\mathrm{BP},t}\right] \in \{0, 1/2, 1\}. \tag{1.1}$$

As a consequence, on XORSAT the BPGD algorithm effectively reduces to a purely combinatorial algorithm called Unit Clause Propagation [18, 24] as per Proposition 14, a fact that we will exploit extensively (see Section 2.7).

1.3 A tight analysis of BPGD

In order to state the main results we need to introduce a few threshold values. To this end, given d, k and an additional real parameter $\lambda \geq 0$ that depends on the time t, consider the functions [3]

$$\phi_{d,k,\lambda} : [0,1] \to [0,1], \qquad z \mapsto 1 - \exp\left(-\lambda - dz^{k-1}\right), \tag{1.2}$$

$$\Phi_{d,k,\lambda} : [0,1] \to \mathbb{R}, \qquad z \mapsto \exp\left(-\lambda - dz^{k-1}\right) - \frac{d(k-1)}{k} z^k + dz^{k-1} - \frac{d}{k}. \tag{1.3}$$

Let $\alpha_*(\lambda) = \alpha_*(d,k,\lambda) \in [0,1]$ be the smallest and $\alpha^*(\lambda) = \alpha^*(d,k,\lambda) \geq \alpha_*(d,k,\lambda) \in [0,1]$ the largest fixed point of $\phi_{d,k,\lambda}$. Figure 1 visualizes $\Phi(z)$ for different values of $\theta \sim t/n$. Further, define

$$d_{\min}(k) = \left(\frac{k-1}{k-2}\right)^{k-2}, \qquad d_{\mathrm{core}}(k) = \sup\left\{d > 0 : \alpha^*(0) = 0\right\}, \tag{1.4}$$

$$d_{\mathrm{sat}}(k) = \sup\left\{d > 0 : \Phi_{d,k,0}(\alpha^*(0)) \leq \Phi_{d,k,0}(0)\right\}. \tag{1.5}$$

The value $d_{\mathrm{sat}}(k)$ is the random k-XORSAT satisfiability threshold [3, 11, 23]. Thus, for $d < d_{\mathrm{sat}}(k)$ the random k-XORSAT formula $\boldsymbol{F}$ possesses satisfying assignments w.h.p., while $\boldsymbol{F}$ is unsatisfiable for $d > d_{\mathrm{sat}}(k)$ w.h.p. Furthermore, $d_{\mathrm{core}}(k)$ equals the threshold for the emergence of a giant 2-core within the k-uniform hypergraph induced by $\boldsymbol{F}$ [3, 22]. This implies that for $d < d_{\mathrm{core}}(k)$ the set of solutions of $\boldsymbol{F}$ is connected in a certain well-defined way, while for $d_{\mathrm{core}}(k) < d < d_{\mathrm{sat}}(k)$ the set of solutions shatters into an exponential number of well-separated clusters [15, 18]. Moreover, a simple linear time algorithm is known to find a solution w.h.p. for $d < d_{\mathrm{core}}(k)$ [15]. The relevance of $d_{\min}(k)$ will emerge in Theorem 1. A bit of calculus reveals that

$$0 < d_{\min}(k) < d_{\mathrm{core}}(k) < d_{\mathrm{sat}}(k) < k. \tag{1.6}$$

The following theorem determines the precise clause-to-variable densities where BPGD succeeds/fails. To be precise, in the 'successful' regime BPGD does not actually succeed with *high* probability, but with an explicit probability strictly between zero and one, which is displayed in Figure 2 for $k = 3, 4, 5$.

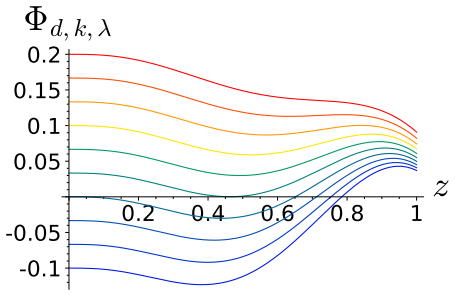


Figure 1 $\Phi_{d,k,\lambda}$ for $k = 3$ and $d = 2.4$, for λ from 0 to 0.3 (maximum at $z = 0$) and from 0.4 to 0.9

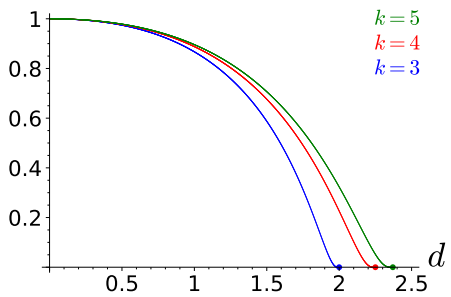


Figure 2 Success probability of BPGD for $0 < d < d_{\min}(k)$ and various k.

[3] The function $\Phi_{d,k,\lambda}$ is known in physics parlance as the "Bethe free entropy" [8, 18]. The stationary points of $\Phi_{d,k,\lambda}$ coincide with the fixed points of $\phi_{d,k,\lambda}$, as we will verify in Section 2.1.

▶ **Theorem 1.** *Let $k \geq 3$.*

(i) *If $d < d_{\min}(k)$, then*

$$\lim_{n\to\infty} \mathbb{P}\left[\texttt{BPGD}(\boldsymbol{F}) \text{ succeeds}\right] = \exp\left(-\frac{d^2(k-1)^2}{4}\int_0^1 \frac{z^{2k-4}(1-z)}{1-d(k-1)z^{k-2}(1-z)}\,\mathrm{d}z\right). \tag{1.7}$$

(ii) *If $d_{\min}(k) < d < d_{\mathrm{sat}}(k)$, then $\mathbb{P}\left[\texttt{BPGD}(\boldsymbol{F}) \text{ succeeds}\right] = o(1)$.*

Theorem 1 vindicates the predictions from Ricci-Tersenghi and Semerjian [24, Section 4] as to the performance of BPGD, and improves over the results from Yung [25]. Specifically, Theorem 1 (i) verifies the formula for the success probability from [24, Eq. (38)]. Combinatorially, the formula (1.7) results from the possible presence of bounded length cycles (so called toxic cycles) that may cause the algorithm to run into contradictions. This complements Yung's prior work, that has no positive result on the performance of BPGD. Moreover, Yung's negative results [25, Theorems 2–3] only apply to $k \geq 9$ and to $d > d_{\mathrm{core}}(k)$, while Theorem 1 (ii) covers all $k \geq 3$ and kicks in at the correct threshold $d_{\min}(k) < d_{\mathrm{core}}(k)$ predicted in [24].

1.4 The decimation process

In addition to the BPGD algorithm itself, the heuristic work [24] considers an idealised version of the algorithm, the *decimation process*. This thought experiment highlights the conceptual reasons behind the success/failure of BPGD. Just like BPGD, the decimation process assigns values to variables one after the other for good. But instead of the BP 'approximations' the decimation process uses the *actual* marginals given its previous decisions. To be precise, suppose that the input formula $\boldsymbol{F}$ is satisfiable and that variables $x_1, \ldots, x_t$ have already been assigned values $\boldsymbol{\sigma}_{\mathrm{DC}}(x_1), \ldots, \boldsymbol{\sigma}_{\mathrm{DC}}(x_t)$ in the previous iterations. Obtain $\boldsymbol{F}_{\mathrm{DC},t}$ by substituting the values $\boldsymbol{\sigma}_{\mathrm{DC}}(x_1), \ldots, \boldsymbol{\sigma}_{\mathrm{DC}}(x_t)$ for $x_1, \ldots, x_t$ and dropping any clauses that do not contain any of $x_{t+1}, \ldots, x_n$. Thus, $\boldsymbol{F}_{\mathrm{DC},t}$ is a XORSAT formula with variables $x_{t+1}, \ldots, x_n$. Let $\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}$ be a random satisfying assignment of $\boldsymbol{F}_{\mathrm{DC},t}$. Then the decimation process sets x_{t+1} according to the true marginal $\mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(x_{t+1}) = 1 \mid \boldsymbol{F}_{\mathrm{DC},t}\right]$, thus ultimately returning a uniformly random satisfying assignment of $\boldsymbol{F}$.

Algorithm 2 The decimation process.

Data: a random k-XORSAT formula $\boldsymbol{F}$, conditioned on being satisfiable

for $t = 0, \ldots, n-1$ **do**

  compute $\pi_{\boldsymbol{F}_{\mathrm{DC},t}} = \mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(x_{t+1}) = 1 \mid \boldsymbol{F}_{\mathrm{DC},t}\right]$;

  set $\boldsymbol{\sigma}_{\mathrm{DC}}(x_t) = \begin{cases} 1 & \text{with probability } \pi_{\boldsymbol{F}_{\mathrm{DC},t}} \\ 0 & \text{with probability } 1-\pi_{\boldsymbol{F}_{\mathrm{DC},t}} \end{cases}$;

return $\boldsymbol{\sigma}_{\mathrm{DC}}$;

Clearly, if indeed the BP 'approximations' are correct, then the decimation process and BPGD are identical. Thus, a key question is for what parameter regimes the two process coincide or diverge, respectively. As it turns out, this question is best answered by parametrise not only in terms of the average variable degree d, but also in terms of the 'time' parameter t of the decimation process.

1.5 Phase transitions of the decimation process

Ricci-Tersenghi and Semerjian heuristically identify several phase transitions in terms of d and t that the decimation process undergoes. We will confirm these predictions mathematically

and investigate how they relate to the performance of BPGD.

The first set of relevant phase transitions concerns the so-called non-reconstruction property. Roughly speaking, non-reconstruction means that the marginal $\pi_{\boldsymbol{F}_{\mathrm{DC},t}} = \mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(x_{t+1}) = 1 \mid \boldsymbol{F}_{\mathrm{DC},t}\right]$ is determined by short-range rather than long-range effects. Since Belief Propagation is essentially a local algorithm, one might expect that the (non-)reconstruction phase transition coincides with the threshold up to which BPGD succeeds; cf. the discussions in [5, 16].

To define (non-)reconstruction precisely, we associate a bipartite graph $G(\boldsymbol{F}_{\mathrm{DC},t})$ with the formula $\boldsymbol{F}_{\mathrm{DC},t}$. The vertices of this graph are the variables and clauses of $\boldsymbol{F}_{\mathrm{DC},t}$. Each variable is adjacent to the clauses in which it appears. For a (variable or clause) vertex v of $G(\boldsymbol{F}_{\mathrm{DC},t})$ let ∂v be the set of neighbours of v in $G(\boldsymbol{F}_{\mathrm{DC},t})$.More generally, for an integer $\ell \geq 1$ let $\partial^\ell v$ be the set of vertices of $G(\boldsymbol{F}_{\mathrm{DC},t})$ at shortest path distance precisely ℓ from v. Following [16], we say that $\boldsymbol{F}_{\mathrm{DC},t}$ has the *non-reconstruction property* if

$$\lim_{\ell\to\infty}\limsup_{n\to\infty}\mathbb{E}\Big[\Big|\mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(x_{t+1}) = 1 \,\Big|\, \boldsymbol{F}_{\mathrm{DC},t},\, \left\{\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(y)\right\}_{y\in\partial^{2\ell}x_{t+1}}\right] \tag{1.8}$$
$$-\mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(x_{t+1}) = 1 \mid \boldsymbol{F}_{\mathrm{DC},t}\right]\Big| \,\Big|\, \boldsymbol{F} \text{ satisfiable}\Big] = 0.$$

Conversely, $\boldsymbol{F}_{\mathrm{DC},t}$ has the *reconstruction property* if

$$\liminf_{\ell\to\infty}\liminf_{n\to\infty}\mathbb{E}\Big[\Big|\mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(x_{t+1}) = 1 \,\Big|\, \boldsymbol{F}_{\mathrm{DC},t},\, \left\{\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(y)\right\}_{y\in\partial^{2\ell}x_{t+1}}\right] \tag{1.9}$$
$$-\mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(x_{t+1}) = 1 \mid \boldsymbol{F}_{\mathrm{DC},t}\right]\Big| \,\Big|\, \boldsymbol{F} \text{ sat.}\Big] > 0.$$

To parse (1.8), notice that in the left probability term we condition on both the outcome $\boldsymbol{F}_{\mathrm{DC},t}$ of the first t steps of the decimation process and on the values $\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(y)$ that the random solution $\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}$ assigns to the variables y at distance exactly 2ℓ from x_{t+1}. By contrast, in the right probability term we only condition on $\boldsymbol{F}_{\mathrm{DC},t}$. Thus, the second probability term matches the probability $\pi_{\boldsymbol{F}_{\mathrm{DC},t}}$ from the decimation process. Hence, (1.8) compares the probability that a random solution sets x_{t+1} to one given the values $\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(y)$ of *all* variables y at distance 2ℓ from x_{t+1} with plain marginal probability that x_{t+1} is set to one. What (1.8) asks is that these two probabilities be asymptotically equal in the limit of large ℓ, with high probability over the choice of $\boldsymbol{F}$ and the prior steps of the decimation process.

Confirming the predictions from [24], the following theorem identifies the precise regimes of d, t where (non-)reconstruction holds. To state the theorem, we need to know that for $d_{\min}(k) < d < d_{\mathrm{sat}}(k)$ the polynomial $d(k-1)z^{k-2}(1-z) - 1$ has precisely two roots $0 < z_* = z_*(d,k) < z^* = z^*(d,k) < 1$; we are going to prove this as part of Proposition 5 below. Let

$$\lambda_* = \lambda_*(d,k) = -\log(1-z_*) - \frac{z_*}{(k-1)(1-z_*)} \tag{1.10}$$
$$> \lambda^* = \lambda^*(d,k) = \max\left\{0, -\log(1-z^*) - \frac{z^*}{(k-1)(1-z^*)}\right\} \geq 0, \tag{1.11}$$
$$\theta_* = \theta_*(d,k) = 1-\exp(-\lambda_*) > \theta^* = \theta^*(d,k) = 1-\exp(-\lambda^*). \tag{1.12}$$

Additionally, let $\lambda_{\mathrm{cond}}(d,k)$ be the solution to the ODE

$$\frac{\partial\lambda_{\mathrm{cond}}(d,k)}{\partial d} = -\frac{\alpha^*(\lambda_{\mathrm{cond}}(d,k))^k - \alpha_*(\lambda_{\mathrm{cond}}(d,k))^k}{k(\alpha^*(\lambda_{\mathrm{cond}}(d,k)) - \alpha_*(\lambda_{\mathrm{cond}}(d,k)))}, \qquad \lambda_{\mathrm{cond}}(d_{\mathrm{sat}}(k),k) = 0 \tag{1.13}$$

on $(d_{\min}, d_{\mathrm{sat}}]$ and set $\theta_{\mathrm{cond}} = \theta_{\mathrm{cond}}(d,k) = 1-\exp(-\lambda_{\mathrm{cond}}(d,k))$. Note that $\theta^* < \theta_{\mathrm{cond}} < \theta_*$.

▶ **Theorem 2.** *Let $k \geq 3$ and let $0 \leq t = t(n) \leq n$ be a sequence such that $\lim_{n\to\infty} t/n = \theta \in (0,1)$.*

(i) *If $d < d_{\min}(k)$, then $\boldsymbol{F}_{\mathrm{DC},t}$ has the non-reconstruction property w.h.p.*

(ii) *If $d_{\min}(k) < d < d_{\mathrm{sat}}(k)$ and $\theta < \theta^*$ or $\theta > \theta_{\mathrm{cond}}$, then $\boldsymbol{F}_{\mathrm{DC},t}$ has the non-reconstruction property w.h.p.*

(iii) *If $d_{\min}(k) < d < d_{\mathrm{sat}}(k)$ and $\theta^* < \theta < \theta_{\mathrm{cond}}$, then $\boldsymbol{F}_{\mathrm{DC},t}$ has the reconstruction property w.h.p.*

Theorem 2 shows that $d_{\min}(k)$ marks the precise threshold of d up to which the decimation process $\boldsymbol{F}_{\mathrm{DC},t}$ exhibits non-reconstruction for *all* $0 \leq t \leq n$ w.h.p. By contrast, for $d_{\min}(k) < d < d_{\mathrm{sat}}(k)$ there is a regime of t where reconstruction occurs. In fact, as Proposition 5 shows, for $d > d_{\mathrm{core}}(k)$ we have $\theta^* = 0$ and thus reconstruction holds even at $t = 0$, i.e., for the original, undecimated random formula $\boldsymbol{F}$. Prior to the contribution [24], it had been suggested that this precise scenario (reconstruction on the original problem instance) is the stone on which BPGD stumbles [5]. In fact, Yung's negative result kicks in at this precise threshold $d_{\mathrm{core}}(k)$. However, Theorems 1 and 2 show that matters are more subtle. Specifically, for $d_{\min}(k) < d < d_{\mathrm{core}}(k)$ reconstruction, even though absent in the initial formula $\boldsymbol{F}$, occurs at a later 'time' $t > 0$ as decimation proceeds, which suffices to trip BPGD up. Also, remarkably, Theorem 2 shows that non-reconstruction is not 'monotone'. The property holds for $\theta < \theta^*$ and then again for $\theta > \theta_{\mathrm{cond}}$, but not on the interval $(\theta^*, \theta_{\mathrm{cond}})$ as visualised in Figure 3.

But there is one more surprise. Namely, Theorem 2 (ii) might suggest that for $d_{\min}(k) < d < d_{\mathrm{sat}}(k)$ Belief Propagation manages to compute the correct marginals for $t/n \sim \theta > \theta_{\mathrm{cond}}$, as non-reconstruction kicks back in. But remarkably, this is not quite true. Despite the fact that non-reconstruction holds, BPGD goes astray because the algorithm starts its message passing process from a mistaken, oblivious initialisation. As a consequence, for $t/n \sim \theta \in (\theta_{\mathrm{cond}}, \theta_*)$ the BP 'approximations' remain prone to error. To be precise, the following result identifies the precise 'times' where BP succeeds/fails. To state the result let $\mu_{\boldsymbol{F}_{\mathrm{DC},t}}$ denote the BP 'approximation' of the true marginal $\pi_{\boldsymbol{F}_{\mathrm{DC},t}}$ of variable x_{t+1} in the formula $\boldsymbol{F}_{\mathrm{DC},t}$ created by the decimation process (see Section 2.2 for a reminder of the definition). Also recall that $\pi_{\boldsymbol{F}_{\mathrm{DC},t}}$ denotes the correct marginal as used by the decimation process.

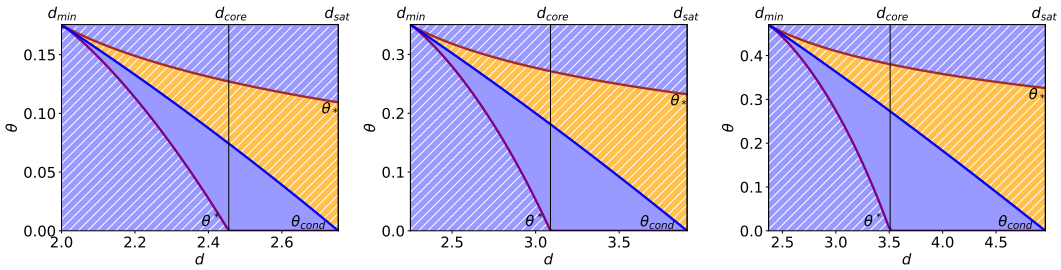
▶ **Theorem 3.** *Let $k \geq 3$ and let $0 \leq t = t(n) \leq n$ be a sequence such that $\lim_{n\to\infty} t/n = \theta \in (0,1)$.*

(i) *If $0 < d < d_{\min}(k)$ then $\mu_{\boldsymbol{F}_{\mathrm{DC},t}} = \pi_{\boldsymbol{F}_{\mathrm{DC},t}}$ w.h.p.*

(ii) *If $d_{\min}(k) < d < d_{\mathrm{sat}}(k)$ and $\theta < \theta_{\mathrm{cond}}$ or $\theta > \theta_*$, then $\mu_{\boldsymbol{F}_{\mathrm{DC},t}} = \pi_{\boldsymbol{F}_{\mathrm{DC},t}}$ w.h.p.*

(iii) *If $d_{\min}(k) < d < d_{\mathrm{sat}}(k)$ and $\theta_{\mathrm{cond}} < \theta < \theta_*$, then $\mathbb{E}\left|\mu_{\boldsymbol{F}_{\mathrm{DC},t}} - \pi_{\boldsymbol{F}_{\mathrm{DC},t}}\right| = \Omega(1)$.*

The upshot of Theorems 2–3 is that the relation between the accuracy of BP and reconstruction is subtle. Everything goes well so long as $d < d_{\min}$ as non-reconstruction holds throughout and the BP approximations are correct. But if $d_{\min} < d < d_{\mathrm{sat}}$ and $\theta^* < \theta < \theta_{\mathrm{cond}}$, then Theorem 2 (iii) shows that reconstruction occurs. Nonetheless, Theorem 3 (ii) demonstrates that the BP approximations remain valid in this regime. By contrast, for $\theta_{\mathrm{cond}} < \theta < \theta_*$ we have non-reconstruction by Theorem 2 (iii), but Theorem 3 (iii) shows that BP misses its mark with a non-vanishing probability. Finally, for $\theta > \theta_*$ everything is in order once again as BP regains its footing and non-reconstruction holds. Unfortunately BPGD is unlikely to reach this happy state because the algorithm is bound to make numerous mistakes at times $t/n \in (\theta_{\mathrm{cond}}, \theta_*)$.



(a) $k = 3$ **(b)** $k = 4$ **(c)** $k = 5$

Figure 3 The phase diagrams for $k = 3, 4, 5$ with $d \in (d_{\min}, d_{\mathrm{sat}})$ on the horizontal and θ on the vertical axis. The hatched area displays the regime $\theta < \theta^*$ and $\theta_{\mathrm{cond}} < \theta$ where non reconstruction holds. In the non hatched area, where $\theta^* < \theta < \theta_{\mathrm{cond}}$, we have reconstruction. Similarly, the blue area displays $\theta < \theta_{\mathrm{cond}}$ and $\theta > \theta_*$ where BP is correct whereas in the orange area, BP is inaccurate.

Theorems 2 and 3 confirm the predictions from [24, Section 4]. To be precise, while θ_{cond} matches the predictions of Ricci-Tersenghi and Semerjian, the ODE formula (1.13) for the threshold, which is easy to evaluate numerically, does not appear in [24]. Instead of the ODE formulation, Ricci-Tersenghi and Semerjian define λ_{cond} as the (unique) $\lambda \geq 0$ such that $\Phi_{d,k,\lambda}(\alpha_*) = \Phi_{d,k,\lambda}(\alpha^*)$; Proposition 5 below shows that both are equivalent. Illustrating Theorems 2–3, Figure 3 displays the phase diagram in terms of d and $\theta \sim t/n$ for $k = 3, 4, 5$.

2 Overview

This section provides an overview of the proofs of Theorems 1–3. In the final paragraph we conclude with a discussion of further related work. *We assume throughout that $k \geq 3$ is an integer and that $0 < d < d_{\mathrm{sat}}(k)$. Moreover, $t = t(n)$ denotes an integer sequence $0 \leq t(n) \leq n$ such that $\lim_{n\to\infty} t(n)/n = \theta \in (0, 1)$.*

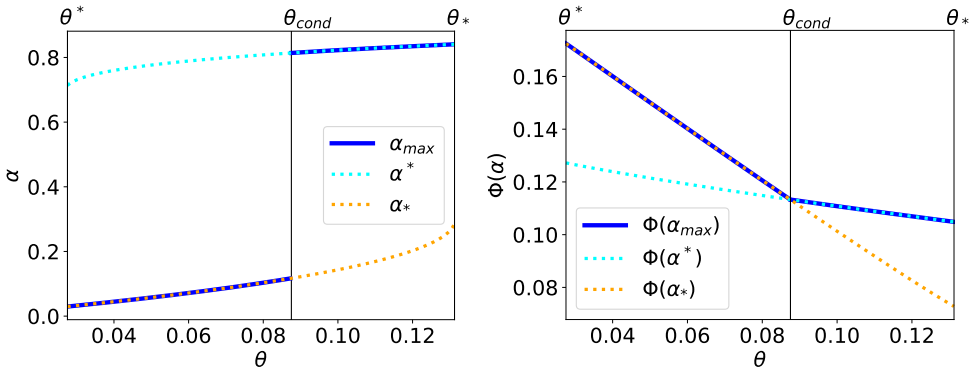
2.1 Fixed points and thresholds

The first item on our agenda is to study the functions $\phi_{d,k,\lambda}, \Phi_{d,k,\lambda}$ from (1.2)–(1.3). Specifically, we are concerned with the maxima of $\Phi_{d,k,\lambda}$ and the fixed points of $\phi_{d,k,\lambda}$, the combinatorial relevance of which will emerge as we analyse BPGD and the decimation process. We begin by observing that the fixed points of $\phi_{d,k,\lambda}$ are precisely the stationary points of $\Phi_{d,k,\lambda}$.

▶ **Fact 4.** *For any $d > 0, \lambda \geq 0$ the stationary points $z \in (0, 1)$ of $\Phi_{d,k,\lambda}$ coincide with the fixed points of $\phi_{d,k,\lambda}$ in $(0, 1)$. Furthermore, for a fixed point $z \in (0, 1)$ of $\phi_{d,k,\lambda}$ we have*

$$\Phi''_{d,k,\lambda}(z) \begin{cases} < 0 & \text{if } \phi'_{d,k,\lambda}(z) < 1, \\ = 0 & \text{if } \phi'_{d,k,\lambda}(z) = 1, \\ > 0 & \text{if } \phi'_{d,k,\lambda}(z) > 1. \end{cases} \tag{2.1}$$

We recall that $0 \leq \alpha_* = \alpha_*(d, k, \lambda) \leq \alpha^* = \alpha^*(d, k, \lambda) \leq 1$ are the smallest and the largest fixed point of $\phi_{d,k,\lambda}$ in $[0, 1]$, respectively. Fact 4 shows that $\Phi_{d,k,\lambda}$ attains its global maximum in $[0, 1]$ at α_* or α^*. Let $\alpha_{\max} = \alpha_{\max}(d, k, \lambda) \in \{\alpha_*, \alpha^*\}$ be the maximiser of $\Phi_{d,k,\lambda}$; if $\Phi_{d,k,\lambda}(\alpha_*) = \Phi_{d,k,\lambda}(\alpha^*)$, set $\alpha_{\max} = \alpha_*$. An example for $\alpha_*, \alpha^*, \alpha_{\max}$ and $\Phi(\alpha_*), \Phi(\alpha^*), \Phi(\alpha_{\max})$ is visualised in Figure 4. The following proposition characterises the fixed points of $\phi_{d,k,\lambda}$ and the maximiser $\alpha_{\max}$.



(a) $\alpha_{\max}$ **(b)** $\Phi(\alpha_{\max})$

Figure 4 $\alpha_{\max}$ and $\Phi(\alpha_{\max})$ for $d = 2.4$ and $k = 3$ from θ^* to θ_*.

▶ **Proposition 5.**

(i) *If $d < d_{\min}(k)$, then for all $\lambda > 0$ we have $\alpha_* = \alpha^*$, the function $\lambda \in (0,\infty) \mapsto \alpha_* \in (0,1)$ is analytic, and α_* is the unique stable fixed point of $\phi_{d,k,\lambda}$.*

(ii) *If $d_{\min}(k) < d < d_{\mathrm{sat}}(k)$, then the polynomial $d(k-1)z^{k-2}(1-z)-1$ has precisely two roots $0 < z_* < z^* < 1$, the numbers λ_*, λ^* from (1.10) satisfy $0 \le \lambda^* < \lambda_*$ and the following is true.*

(a) *If $\lambda < \lambda^*$ or $\lambda > \lambda_*$, then $\alpha_* = \alpha^* \in (0,1)$ is the unique stable fixed point of $\phi_{d,k,\lambda}$.*

(b) *If $\lambda^* < \lambda < \lambda_*$, then $0 < \alpha_* < \alpha^* < 1$ are the only stable fixed points of $\phi_{d,k,\lambda}$.*

(c) *The functions $\lambda \in (0,\lambda_*) \mapsto \alpha_*$ and $\lambda \in (\lambda^*,\infty) \mapsto \alpha^*$ are analytic.*

(d) *If $d_{\min}(k) < d < d_{\mathrm{sat}}(k)$, then the solution λ_{cond} of (1.13) satisfies $\lambda^* < \lambda_{\mathrm{cond}} = \lambda_{\mathrm{cond}}(d) < \lambda_*$ and $\alpha_{\max} = \alpha_*$ if $\lambda < \lambda_{\mathrm{cond}}$ while $\alpha_{\max} = \alpha^*$ if $\lambda > \lambda_{\mathrm{cond}}$.*

2.2 Belief Propagation

Having done our analytic homework, we proceed to recall how Belief Propagation computes the 'approximations' $\mu_{\boldsymbol{F}_{\mathrm{BP},t}}$ that the BPGD algorithm relies upon. We will see that due to the inherent symmetries of XORSAT the Belief Propagation computations simplify and boil down to a simpler message passing process called Warning Propagation. Subsequently we will explain the connection between Warning Propagation and the fixed points α_*, α^* of $\phi_{d,k,\lambda}$.

It is probably easiest to explain BP on a general XORSAT instance F with a set $V(F)$ of variables and a set $C(F)$ of clauses of lengths between one and k. As in Section 1.5 we consider the graph $G(F)$ induced by F, with vertex set $V(F) \cup C(F)$ and an edge xa between $x \in V(F)$ and $a \in C(F)$ iff a contains x. Let $\partial v = \partial_F v$ be the set of neighbours of $v \in V(F) \cup C(F)$. Additionally, given an assignment $\tau \in \{0,1\}^{\partial a}$ of the variables that appear in a, we write $\tau \models a$ iff τ satisfies a.

With each clause/variable pair x, a such that $x \in \partial a$ Belief Propagation associates two sequences of 'messages' $(\mu_{F,x\to a,\ell})_{\ell\ge 0}$, $(\mu_{F,a\to x,\ell})_{\ell\ge 0}$ directed from x to a and from a to x, respectively. These messages are probability distributions on $\{0,1\}$, i.e.,

$$\mu_{F,x\to a,\ell} = (\mu_{F,x\to a,\ell}(0), \mu_{F,x\to a,\ell}(1)),\ \mu_{F,a\to x,\ell} = (\mu_{F,a\to x,\ell}(0), \mu_{F,a\to x,\ell}(1)), \tag{2.2}$$

$$\mu_{F,x\to a,\ell}(0) + \mu_{F,x\to a,\ell}(1) = \mu_{F,a\to x,\ell}(0) + \mu_{F,a\to x,\ell}(1) = 1. \tag{2.3}$$

The initial messages are uniform, i.e.,

$$\mu_{F,x\to a,0}(s) = \mu_{F,a\to x,0}(s) = 1/2 \qquad (s \in \{0,1\}). \tag{2.4}$$

Further, the messages at step $\ell+1$ are obtained from the messages at step ℓ via the *Belief Propagation equations*

$$\mu_{F,a\to x,\ell+1}(s) \propto \sum_{\tau\in\{0,1\}^{\partial a}} 1\{\tau_x = s,\, \tau \models a\} \prod_{y\in\partial a\setminus\{x\}} \mu_{F,y\to a,\ell}(\tau_y), \tag{2.5}$$

$$\mu_{F,x\to a,\ell+1}(s) \propto \prod_{b\in\partial x\setminus\{a\}} \mu_{F,b\to x,\ell}(s). \tag{2.6}$$

In (2.5)–(2.6) the $\propto$-symbol represents the normalisation required to ensure that the updated messages satisfy (2.3). In the case of (2.6) such a normalization may be impossible because the expressions on the r.h.s. could vanish for both $s=0$ and $s=1$. In this event we agree that

$$\mu_{F,x\to a,\ell+1}(s) = \begin{cases} \mu_{F,x\to a,\ell}(s) & \text{if } \mu_{F,x\to a,\ell}(s) \neq 1/2 \\ 1\{s=0\} & \text{otherwise} \end{cases} \qquad (s \in \{0,1\});$$

in other words, we retain the messages from the previous iteration unless its value was $1/2$, in which case we set $\mu_{F,x\to a,\ell+1}(0) = 1$. The same convention applies to $\mu_{F,a\to x,\ell+1}(s)$. Further, at any time t the BP messages render a heuristic 'approximation' of the marginal probability that a random solution to the formula F sets a variable x to $s \in \{0,1\}$:

$$\mu_{F,x,\ell}(s) \propto \prod_{b\in\partial x} \mu_{F,b\to x,\ell}(s). \tag{2.7}$$

We set $\mu_{F,x,\ell}(0) = 1 - \mu_{F,x,\ell}(1) = 1$ if $\sum_{s\in\{0,1\}} \prod_{b\in\partial x} \mu_{F,b\to x,\ell}(s) = 0$.

▶ **Fact 6.** *The BP messages and marginals are half-integral for all t, i.e., for all $t \geq 0$ and $s \in \{0,1\}$ we have*

$$\mu_{F,x\to a,\ell}(s), \mu_{F,a\to x,\ell}(s), \mu_{F,x,\ell}(s) \in \{0, 1/2, 1\}. \tag{2.8}$$

Furthermore, for all $\ell > 2\sum_{a\in C(F)} |\partial a|$ we have $\mu_{F,x,\ell}(s) = \mu_{F,x,\ell+1}(s)$.

Finally, in light of Fact 6 it makes sense to define the approximations for `BPGD` by letting

$$\mu_{\boldsymbol{F}_{\mathrm{BP},t}} = \lim_{\ell\to\infty} \mu_{\boldsymbol{F}_{\mathrm{BP},t},x_{t+1},\ell}(1), \qquad \mu_{\boldsymbol{F}_{\mathrm{DC},t}} = \lim_{\ell\to\infty} \mu_{\boldsymbol{F}_{\mathrm{DC},t},x_{t+1},\ell}(1). \tag{2.9}$$

2.3 Warning Propagation

Thanks to the half-integrality (2.8) of the messages, Belief Propagation is equivalent to a purely combinatorial message passing procedure called *Warning Propagation* ('WP') [18]. Similar as BP, WP also associates two message sequences $(\omega_{F,x\to a,\ell}, \omega_{F,a\to x,\ell})_{\ell\geq 0}$ with every adjacent clause/variable pair. The messages take one of three possible discrete values $\{\mathtt{f},\mathtt{u},\mathtt{n}\}$ ('frozen', 'uniform', 'null'). Essentially, `n` indicates that the value of a variable is determined by unit clause propagation. Moreover, `f` indicates that a variable is forced to take the value 0 once all variables in the 2-core of the hypergraph representation of the formula are set to 0. The remaining label `u` indicates that neither of the above applies. To trace the BP messages from Section 2.2 actually only the two values $\{\mathtt{n},\mathtt{u}\}$ would be necessary. However, the third value `f` will prove useful in order to compare the BP approximations with the

actual marginals. Perhaps unexpectedly given the all-uniform initialisation (2.4), we launch WP from all-frozen start values:

$$\omega_{F,x\to a,0} = \omega_{F,a\to x,0} = \mathtt{f} \qquad \text{for all } a, x. \tag{2.10}$$

Subsequently the messages get updated according to the rules

$$\omega_{F,a\to x,\ell+1} = \begin{cases} \mathtt{n} & \text{if } \omega_{F,y\to a,\ell} = \mathtt{n} \text{ for all } y \in \partial a \setminus \{x\}, \\ \mathtt{f} & \text{if } \omega_{F,y\to a,\ell} \neq \mathtt{u} \text{ for all } y \in \partial a \setminus \{x\} \text{ and } \omega_{F,y\to a,\ell} \neq \mathtt{n} \\ & \text{for at least one } y \in \partial a \setminus \{x\}, \\ \mathtt{u} & \text{otherwise,} \end{cases} \tag{2.11}$$

$$\omega_{F,x\to a,\ell+1} = \begin{cases} \mathtt{n} & \text{if } \omega_{F,b\to x,\ell} = \mathtt{n} \text{ for at least one } b \in \partial x \setminus \{a\}, \\ \mathtt{f} & \text{if } \omega_{F,b\to x,\ell} \neq \mathtt{n} \text{ for all } b \in \partial x \setminus \{a\} \text{ and } \omega_{F,b\to x,\ell} = \mathtt{f} \\ & \text{for at least one } b \in \partial x \setminus \{a\} \\ \mathtt{u} & \text{otherwise.} \end{cases}, \tag{2.12}$$

In addition to the messages we also define the *mark* $\omega_{F,x,\ell}$ of variable node x as in (2.11), or be it without omitting clause a. The following statement summarises the relationship between BP and WP.

▶ **Fact 7.** *For all $t \geq 0$ and all x, a we have*

$$\mu_{x\to a,\ell}(1) = 1/2 \quad \Leftrightarrow \quad \omega_{F,x\to a,\ell} \neq \mathtt{n}, \tag{2.13}$$

$$\mu_{a\to x,\ell}(1) = 1/2 \quad \Leftrightarrow \quad \omega_{F,a\to x,\ell} \neq \mathtt{n}, \tag{2.14}$$

$$\mu_{x,\ell}(1) = 1/2 \quad \Leftrightarrow \quad \omega_{F,x,\ell} \neq \mathtt{n}. \tag{2.15}$$

Moreover, for all $\ell > 2|C(F)|$ we have $\omega_{F,x\to a,\ell} = \omega_{F,x\to a,\ell+1}$ and $\omega_{F,a\to x,\ell} = \omega_{F,a\to x,\ell+1}$.

Fact 7 implies that the WP messages and marks 'converge' in the limit of large ℓ, in the sense that eventually they do not change any more. Let $\omega_{F,x\to a}, \omega_{F,a\to x}, \omega_{F,x} \in \{\mathtt{f}, \mathtt{u}, \mathtt{n}\}$ be these limits. Furthermore, let $V_{\mathtt{f},\ell}(F)$, $V_{\mathtt{u},\ell}(F)$, $V_{\mathtt{n},\ell}(F)$ be the sets of variables with the respective mark after $\ell \geq 0$ iterations. Also let $V_{\mathtt{f}}(F), V_{\mathtt{u}}(F), V_{\mathtt{n}}(F)$ be the sets of variables where the limit $\omega_{F,x}$ takes the respective value. The following statement traces WP on the random formula $\boldsymbol{F}_{\mathrm{DC},t}$ produced by the decimation process.

▶ **Proposition 8.** *Let $\varepsilon > 0$ and assume that $d > 0$, $t = t(n) \sim \theta n$ satisfy one of the following conditions:*

(i) *$d < d_{\min}$, or*

(ii) *$d > d_{\min}$ and $\theta \notin \{\theta_*, \theta^*\}$.*

Then there exists $\ell_0 = \ell_0(d, \theta, \varepsilon) > 0$ such that for any fixed $\ell \geq \ell_0$ with $\lambda = -\log(1-\theta)$ w.h.p. we have

$$|t + |V_{\mathtt{n},\ell}(\boldsymbol{F}_{\mathrm{DC},t})| - \alpha_* n| < \varepsilon n, \qquad |t + |V_{\mathtt{f},\ell}(\boldsymbol{F}_{\mathrm{DC},t})| - (\alpha^* - \alpha_*)n| < \varepsilon n, \tag{2.16}$$

$$|V_{\mathtt{n}}(\boldsymbol{F}_{\mathrm{DC},t}) \triangle V_{\mathtt{n},\ell}(\boldsymbol{F}_{\mathrm{DC},t})| < \varepsilon n. \tag{2.17}$$

2.4 The check matrix

Since the XOR operation is equivalent to addition modulo two, a XORSAT formula F with variables $x_1, \ldots, x_n$ and clauses $a_1, \ldots, a_m$ translates into a linear system over $\mathbb{F}_2$, as follows. Let A_F be the $m \times n$-matrix over $\mathbb{F}_2$ whose (i,j)-entry equals one iff variable x_j appears in clause a_i. Adopting coding parlance, we refer to A_F as the *check matrix* of F. Furthermore,

let $y_F \in \mathbb{F}_2^m$ be the vector whose ith entry is one plus the sum of any constant term and the number of negation signs of clause a_i mod two. Then the solutions $\sigma \in \mathbb{F}_n^n$ of the linear system $A_F\sigma = y_F$ are precisely the satisfying assignments of F.

The algebraic properties of A_F therefore have a direct impact on the satisfiability of F. For example, if A_F has rank m, we may conclude immediately that F is satisfiable. Furthermore, the set of solutions of F is an affine subspace of $\mathbb{F}_2^n$ (if non-empty). In effect, if F is satisfiable, then the number of satisfying assignments equals the size of the kernel of A_F. Hence the nullity $\operatorname{nul} A_F = \dim \ker A_F$ of the check matrix is a key quantity.

Indeed, the single most significant ingredient towards turning the heuristic arguments from [24] into rigorous proofs is a formula for the nullity of the check matrix of the XORSAT instance $\boldsymbol{F}_{\mathrm{DC},t}$ from the decimation process. To unclutter the notation set $\boldsymbol{A}_t = A_{\boldsymbol{F}_{\mathrm{DC},t}}$. We derive the following proposition from a recent general result about the nullity of random matrices over finite fields [8, Theorem 1.1]. The proposition clarifies the semantics of the function $\Phi_{d,k,\lambda}$ and its maximiser $\alpha_{\max}$. In physics jargon $\Phi_{d,k,\lambda}$ is known as the Bethe free entropy.

▶ **Proposition 9.** *Let $d > 0$ and $\lambda = -\log(1-\theta)$. Then*

$$\lim_{n\to\infty} \operatorname{nul} \boldsymbol{A}_t = \Phi_{d,k,\lambda}(\alpha_{\max}) \qquad \textit{in probability.}$$

2.5 Null variables

Proposition 9 enables us to derive crucial information about the set of satisfying assignments of $\boldsymbol{F}_{\mathrm{DC},t}$. Specifically, for any XORSAT instance F with variables $x_1, \ldots, x_n$ let $V_0(F)$ be the set of variables x_i such that $\sigma_i = 0$ for all $\sigma \in \ker A_F$. We call the variables $x_i \in V_0(F)$ *null variables*. Since the set of solutions of F, if non-empty, is a translation of $\ker A_F$, any two solutions σ, σ' of F set the variables in $V_0(F)$ to exactly the same values. The following proposition shows that WP identifies certain variables as null.

▶ **Proposition 10.** *W.h.p. the following two statements are true for any fixed integer $\ell > 0$.*

(i) *We have $V_{\mathtt{n},\ell}(\boldsymbol{F}_{\mathrm{DC},t}) \subseteq V_0(\boldsymbol{F}_{\mathrm{DC},t})$.*

(ii) *We have $|V_{\mathtt{u},\ell}(\boldsymbol{F}_{\mathrm{DC},t}) \cap V_0(\boldsymbol{F}_{\mathrm{DC},t})| = o(n)$.*

Propositions 9 and 10 enable us to calculate the number of null variables of $\boldsymbol{F}_{\mathrm{DC},t}$, so long as we remain clear of the point θ_{cond} where $\alpha_{\max}$ is discontinuous.

▶ **Proposition 11.** *If $\theta \neq \theta_{\mathrm{cond}}$ then $|V_0(\boldsymbol{F}_{\mathrm{DC},t})| = \alpha_{\max} n + o(n)$ w.h.p.*

Let us briefly summarise what we have learned thus far. First, because all Belief Propagation messages are half-integral, BP reduces to WP. Second, Proposition 8 shows that the fixed points α_*, α^* of $\phi_{d,k,\lambda}$ determine the number of variables marked `n` or `f` by WP. Third, the function $\Phi_{d,k,\lambda}$ and its maximiser $\alpha_{\max}$ govern the nullity of the check matrix and thereby the number of null variables of $\boldsymbol{F}_{\mathrm{DC},t}$. Clearly, the null variables x_i are precisely the ones whose actual marginals $\mathbb{P}\left[\boldsymbol{\sigma}_{\boldsymbol{F}_{\mathrm{DC},t}}(x_i) = s \mid \boldsymbol{F}_{\mathrm{DC},t}\right]$ are *not* uniform. As a next step, we investigate whether BP/WP identify these variables correctly.

In light of Proposition 8, in order to investigate the accuracy of BP it suffices to compare the *numbers* of variables marked `n` by WP with the true marginals. The following corollary summarises the result.

▶ **Corollary 12.** *For any d, θ the following statements are true.*

(i) *If $d < d_{\min}$, or $d > d_{\min}$ and $\theta < \theta_{\mathrm{cond}}$, or $d > d_{\min}$ and $\theta > \theta_*$, then*

$$|V_0(\boldsymbol{F}_{\mathrm{DC},t}) \triangle V_{\mathtt{n}}(\boldsymbol{F}_{\mathrm{DC},t})| = o(n) \qquad \textit{w.h.p.}$$

(ii) *If* $d > d_{\min}$ *and* $\theta_{\mathrm{cond}} < \theta < \theta_*$*, then* $|V_0(\boldsymbol{F}_{\mathrm{DC},t}) \triangle V_{\mathtt{n}}(\boldsymbol{F}_{\mathrm{DC},t})| = \Omega(n)$ *w.h.p.*

Thus, so long as $d < d_{\min}$ or $d > d_{\min}$ and $\theta < \theta_{\mathrm{cond}}$ or $\theta > \theta_*$, the BP/WP approximations are mostly correct. By contrast, if $d > d_{\min}$ and $\theta_{\mathrm{cond}} < \theta < \theta_*$, the BP/WP approximations are significantly at variance with the true marginals w.h.p. Specifically, w.h.p. BP deems $\Omega(n)$ frozen variables unfrozen, thereby setting itself up for failure. Indeed, Corollary 12 easily implies Theorem 3, which in turn implies Theorem 1 (ii) without much ado.

In addition, to settle the (non-)reconstruction thresholds set out in Theorem 2 we need to investigate the *conditional* marginals given the values of variables at a certain distances from x_{t+1} as in (1.8). This is where the extra value `f` from the construction of WP enters. Indeed, for a XORSAT instance F with variables $x_1, \ldots, x_n$ and an integer ℓ let $V_{0,\ell}(F)$ be the set of variables x_i such that $\sigma_i = 0$ for all $\sigma \in \ker A_F$ and $\sigma_h = 0$ for all variables $x_h \in \partial^\ell x_i$. Hence, $V_{0,\ell}(F) \subseteq V_0(F)$ is the set of variables whose ℓ-neighbourhood is contained in $V_0(F)$.

▶ **Corollary 13.** *Assume that* $d > d_{\min}$ *and let* $\varepsilon > 0$.

(i) *If* $\theta < \theta_{\mathrm{cond}}$*, then for any fixed* ℓ *we have* $|V_{\mathtt{f},\ell}(\boldsymbol{F}_{\mathrm{DC},t}) \cap V_{0,\ell}(\boldsymbol{F}_{\mathrm{DC},t})| < \varepsilon n$ *w.h.p.*

(ii) *If* $\theta > \theta_{\mathrm{cond}}$*, then there exists* $\ell_0 = \ell_0(d, \theta, \varepsilon)$ *such that for any fixed* $\ell > \ell_0$ *we have*

$$|(V_{\mathtt{n},\ell}(\boldsymbol{F}_{\mathrm{DC},t}) \cup V_{\mathtt{f},\ell}(\boldsymbol{F}_{\mathrm{DC},t})) \triangle V_{0,\ell}(\boldsymbol{F}_{\mathrm{DC},t})| < \varepsilon n \qquad \text{w.h.p.}$$

Comparing the number of actually frozen variables with the ones marked `f` by WP, we obtain Theorem 2.

2.6 Proving BPGD successful

We are left to prove Theorem 1. First, we need to compute the (strictly positive) success probability of `BPGD` for $d < d_{\min}$. At this point, the fact that `BPGD` has a fair chance of succeeding for $d < d_{\min}$ should not come as a surprise. Indeed, Corollary 12 implies that the BP approximations of the marginals are mostly correct for $d < d_{\min}$, at least on the formula $\boldsymbol{F}_{\mathrm{DC},t}$ created by the decimation process. Furthermore, so long as the marginals are correct, the decimation process $\boldsymbol{F}_{\mathrm{DC},t}$ and the execution of the `BPGD` algorithm $\boldsymbol{F}_{\mathrm{BP},t}$ move in lockstep. The sole difficulty in analysing `BPGD` lies in proving that the estimates of the algorithm are not just mostly correct, but correct up to only a *bounded* expected number of discrepancies over the entire execution of the algorithm. To prove this fact we combine the method of differential equations with a subtle analysis of the sources of the remaining bounded number of discrepancies. These discrepancies result from the presence of short (i.e., bounded-length) cycles in the graph $G(\boldsymbol{F})$. Finally, the proof of the second (negative) part of Theorem 1 follows by coupling the execution of `BPGD` with the decimation process, and invoking Theorem 3. In the next subsection we introduce a simple combinatorial Unit Clause Propagation algorithm to give a glimpse of the proof of the 'positive' part for the success probability of Theorem 1 for $d < d_{\min}$. The proof of the second part of the theorem concerning $d_{\min} < d < d_{\mathrm{sat}}$ as well as the details of both arguments can be found in the full version.

2.7 Unit Clause Propagation

The simple-minded Unit Clause Propagation algorithm attempts to assign random values to as yet unassigned variables one after the other. After each such random assignment the algorithm pursues the 'obvious' implications of its decisions. Specifically, the algorithm substitutes its chosen truth values for all occurrences of the already assigned variables. If this leaves a clause with only a single unassigned variable, a so-called 'unit clause', the algorithm

assigns that variable so as to satisfy the unit clause. If a conflict occurs because two unit clauses impose opposing values on a variable, the algorithm declares that a conflict has occurred, sets the variable to false and continues; of course, in the event of a conflict the algorithm will ultimately fail to produce a satisfying assignment. The pseudocode for the algorithm is displayed in Algorithm 3.

Algorithm 3 The UCP algorithm.

```
1  Let U = ∅ and let σ_UC : U → {0,1} be the empty assignment;
2  for t = 0, ..., n − 1 do
3      if x_{t+1} ∉ U then
4          add x_{t+1} to U;
5          choose σ_UC(x_{t+1}) ∈ {0,1} uniformly at random;
6          while F[σ_UC] contains a unit clause a do
7              let x be the variable in a;
8              let s ∈ {0,1} be the truth value that x needs to take to satisfy a;
9              if another unit clause a′ exists that requires x be set to 1 − s then
10                 output 'conflict' and let σ_UC(x) = 0;
11             else
12                 add x to U and let σ_UC(x) = s;
13 return σ_UC;
```

Let $\boldsymbol{F}_{\mathrm{UC},t}$ denote the simplified formula obtained after the first t iterations (in which the truth values chosen for $x_1, \dots, x_t$ and any values implied by Unit Clauses have been substituted). We notice that the values assigned during Steps 6–12 are deterministic consequences of the choices in Step 5. In particular, the order in which unit clauses are processed Steps 6–12 does not affect the output of the algorithm.

▶ **Proposition 14.** *We have* $\mathbb{P}[\texttt{BPGD}\ \textit{succeeds}] = \mathbb{P}[\texttt{UCP}\ \textit{succeeds}]$.

Proposition 14 allows us to analyse UCP to prove Theorem 1.

2.8 The success probability of UCP for $d < d_{\min}$

We continue to denote by $\boldsymbol{F}_{\mathrm{UC},t}$ the sub-formula obtained after the first t iterations of UCP. Let $V_n = \{x_1, \dots, x_n\}$ be the set of variables of the *XORSAT* instance F. Also, let $\boldsymbol{V}(t) \subseteq \{x_{t+1}, \dots, x_n\}$ be the set of variables of $\boldsymbol{F}_{\mathrm{UC},t}$. Thus, $\boldsymbol{V}(t)$ contains those variables among $x_{t+1}, \dots, x_n$ whose values are not implied by the assignment of $x_1, \dots, x_t$ via unit clauses. Also let $\boldsymbol{C}(t)$ be the set of clauses of $\boldsymbol{F}_{\mathrm{UC},t}$; these clauses contain variables from $\boldsymbol{V}(t)$ only, and each clause contains at least two variables. Let $\bar{\boldsymbol{V}}(t) = V_n \setminus \boldsymbol{V}(t)$ be the set of assigned variables. Thus, after its first t iterations UCP has constructed an assignment $\boldsymbol{\sigma}_{\mathrm{UC}} : \bar{\boldsymbol{V}}(t) \to \{0,1\}$. Moreover, let $\boldsymbol{V}'(t+1) = \boldsymbol{V}(t) \setminus \boldsymbol{V}(t+1)$ be the set of variables that receive values in the course of the iteration $t+1$ for $0 \le t < n$. Additionally, let $\boldsymbol{C}'(t+1)$ be the set of clauses of $\boldsymbol{F}_{\mathrm{UC},t}$ that consists of variables from $\boldsymbol{V}'(t+1)$ only. Finally, let $\boldsymbol{F}'_{\mathrm{UC},t+1}$ be the formula comprising the variables $\boldsymbol{V}'(t+1)$ and the clauses $\boldsymbol{C}'(t+1)$.

To characterise the distribution of $\boldsymbol{F}_{\mathrm{UC},t}$ let $\boldsymbol{n}(t) = |\boldsymbol{V}(t)|$ and let $\boldsymbol{m}_\ell(t)$ be the number of clauses of length ℓ, i.e., clauses that contain precisely ℓ variables from $\boldsymbol{V}(t)$. Observe that $\boldsymbol{m}_1(t) = 0$ because unit clauses get eliminated. Let $\mathfrak{F}_t$ be the σ-algebra generated by $\boldsymbol{n}(t)$ and $(\boldsymbol{m}_\ell(t))_{2 \le \ell \le k}$.

▶ **Fact 15.** *The XORSAT formula $\boldsymbol{F}_{\mathrm{UC},t}$ is uniformly random given $\mathfrak{F}_t$. In other words, the variables that appear in each clause are uniformly random and independent, as are their signs.*

Proof. This follows from the principle of deferred decisions. ◀

We proceed to estimate the random variables $\boldsymbol{n}(t), \boldsymbol{m}_\ell(t)$. Let $\boldsymbol{\alpha}(t) = |\bar{\boldsymbol{V}}(t)|/n$ so that $\boldsymbol{n}(t) = n(1-\boldsymbol{\alpha}(t))$. Recall, that $\bar{\boldsymbol{V}}(t) = V_n \setminus \boldsymbol{V}(t)$. Let $\lambda = \lambda(\theta) = -\log(1-\theta)$ with $\theta \sim t/n$ and recall that $\alpha_* = \alpha_*(d,k,\lambda)$ denotes the smallest fixed point of $\phi_{d,k,\lambda}$. The proof of the following proposition proof can be found in the full version.

▶ **Proposition 16.** *Suppose that $d < d_{\min}(k)$. There exists a function $\delta = \delta(n) = o(1)$ such that for all $0 \le t < n$ and all $2 \le \ell \le k$ we have*

$$\mathbb{P}\left[|\boldsymbol{\alpha}(t) - \alpha_*| > \delta\right] = O(n^{-2}), \quad \mathbb{P}\left[\left|\boldsymbol{m}_\ell(t) - \frac{dn}{k}\binom{k}{\ell}(1-\alpha_*)^\ell \alpha_*^{k-\ell}\right| > \delta n\right] = O(n^{-2}). \tag{2.18}$$

Proposition 16 paves the way for the actual computation of the success probability of `UCP`. Let $\mathcal{R}_t$ be the event that a conflict occurs in iteration t. The following proposition gives us the correct value of $\mathbb{P}\left[\mathcal{R}_t \mid \mathfrak{F}_t\right]$ w.h.p. Since $\mathfrak{F}_t$ is a random variable the value for the probability $\mathbb{P}\left[\mathcal{R}_t \mid \mathfrak{F}_t\right]$ is random as well.

▶ **Proposition 17.** *Fix $\varepsilon > 0$, let $0 \le t < (1-\varepsilon)n$ and define*

$$f_n(t) = d(k-1)(1-\alpha_*)\alpha_*^{k-2}. \tag{2.19}$$

Then with probability $1 - o(1/n)$ we have

$$\mathbb{P}\left[\mathcal{R}_t \mid \mathfrak{F}_t\right] = \frac{f_n(t)^2}{4(n-t)(1-f_n(t))^2} + o(1/n).$$

The proof of Proposition 17 can be found in Section 2.8.1. Moreover, in the full version we prove the following.

▶ **Proposition 18.** *Fix $\varepsilon > 0$ and $\ell \ge 1$. For any $0 \le t_1 < \cdots < t_\ell < (1-\varepsilon)n$ we have*

$$\mathbb{P}\left[\bigwedge_{i=1}^{\ell} \mathcal{R}_{t_i}\right] \sim \prod_{i=1}^{\ell} \frac{f_n(t_i)^2}{4(n-t_i)(1-f_n(t_i))^2}. \tag{2.20}$$

Finally, the following statement, proven in the full version, deals with the εn final steps of the algorithm.

▶ **Proposition 19.** *For any $\delta > 0$ there exists $\varepsilon > 0$ such that $\mathbb{P}\left[\bigcup_{(1-\varepsilon)n<t<n} \mathcal{R}_t\right] < \delta$.*

Before we proceed we notice that Propositions 17–19 imply the first part of Theorem 1.

Proof of Theorem 1 (i). Pick $\delta > 0$, fix a small enough $\varepsilon = \varepsilon(\delta) > 0$ and let $\boldsymbol{R} = \sum_{t=0}^{n-1} 1\{\mathcal{R}_t\}$ be the total number of times at which conflicts occur. Proposition 14 shows that the probability that `BPGD` succeeds equals $\mathbb{P}\left[\boldsymbol{R} = 0\right]$. In order to calculate $\mathbb{P}\left[\boldsymbol{R} = 0\right]$, let

$\boldsymbol{R}_\varepsilon = \sum_{0\le t\le(1-\varepsilon)n} 1\{\mathcal{R}_t\}$ be the number of failures before time $(1-\varepsilon)n$. Proposition 18 shows that for any fixed $\ell \ge 1$ we have

$$\mathbb{E}\left[\prod_{i=1}^{\ell}(\boldsymbol{R}_\varepsilon - i + 1)\right] \sim \ell! \sum_{0\le t_1<\cdots<t_\ell\le(1-\varepsilon)n} \prod_{i=1}^{\ell} \frac{f_n(t_i)^2}{4(n-t_i)(1-f_n(t_i))^2}$$
$$= (1+o(1)) \sum_{0\le t_1,\ldots,t_\ell\le(1-\varepsilon)n} \prod_{i=1}^{\ell} \frac{f_n(t_i)^2}{4(n-t_i)(1-f_n(t_i))^2} \sim \mathbb{E}[\boldsymbol{R}_\varepsilon]^\ell. \tag{2.21}$$

Hence, the inclusion/exclusion principle (e.g., [4, Theorem 1.21]) implies that

$$\mathbb{P}\left[\boldsymbol{R}_\varepsilon = 0\right] \sim \exp(-\mathbb{E}[\boldsymbol{R}_\varepsilon]). \tag{2.22}$$

Further, using Proposition 17 and the linearity of expectation, we obtain with $\lambda(\theta) = -\log(1-\theta)$

$$\mathbb{E}[\boldsymbol{R}_\varepsilon] \sim \sum_{0\le t\le(1-\varepsilon)n} \frac{f_n(t)^2}{4(n-t)(1-f_n(t))^2} \sim \frac{1}{4n}\int_0^{1-\varepsilon} \frac{f_n(\theta n)^2}{(1-\theta)(1-f_n(\theta n))^2}\mathrm{d}\theta$$
$$= \frac{1}{4n}\int_0^{1-\varepsilon} \frac{f_n(\theta n)^2}{(1-\alpha_*)(1-f_n(\theta n))}\frac{\partial\alpha_*}{\partial\lambda}\frac{\partial\lambda(\theta)}{\partial\theta}\mathrm{d}\theta$$
$$= \frac{d^2(k-1)^2}{4}\int_0^{1-\varepsilon} \frac{z^{2k-4}(1-z)}{1-d(k-1)z^{k-2}(1-z)}\,\mathrm{d}z \qquad \text{[by (2.19)]}. \tag{2.23}$$

Finally, Proposition 19 implies that

$$\mathbb{P}\left[\boldsymbol{R} > \boldsymbol{R}_\varepsilon\right] < \delta. \tag{2.24}$$

Thus, the assertion follows from (2.22)–(2.24) upon taking the limit $\delta \to 0$. ◀

2.8.1 Proof of Proposition 17

$\boldsymbol{F}'_{\mathrm{UC},t+1}$ is the XORSAT formula that contains the variables $\boldsymbol{V}'(t+1)$ that get assigned during iteration $t+1$ and the clauses $\boldsymbol{C}'(t+1)$ of $\boldsymbol{F}_{\mathrm{UC},t}$ that contain variables from $\boldsymbol{V}'(t+1)$ only. Also recall that $G(\boldsymbol{F}'_{\mathrm{UC},t+1})$ signifies the graph representation of this XORSAT formula. Unless $\boldsymbol{V}'(t+1) = \emptyset$, the graph $G(\boldsymbol{F}'_{\mathrm{UC},t+1})$ is connected.

▶ **Lemma 20.** *Fix $\varepsilon > 0$ and let $0 \le t \le (1-\varepsilon)n$. With probability $1 - o(1/n)$ the graph $G(\boldsymbol{F}'_{\mathrm{UC},t+1})$ satisfies*

$$|E(G(\boldsymbol{F}'_{\mathrm{UC},t+1}))| \le |V(G(\boldsymbol{F}'_{\mathrm{UC},t+1}))|.$$

The proof of Lemma 20 can be found in the full version. Thus, with probability $1-o(1/n)$ the graph $G(\boldsymbol{F}'_{\mathrm{UC},t+1})$ contains at most one cycle. While it is easy to check that no conflict occurs in iteration $t+1$ if $G(\boldsymbol{F}'_{\mathrm{UC},t+1})$ is acyclic, in the case that $G(\boldsymbol{F}'_{\mathrm{UC},t+1})$ contains a single cycle there is a chance of a conflict. The following definition describes the type of cycle that poses an obstacle.

▶ **Definition 21.** *For a XORSAT formula F we call a sequence of variables and clauses $\mathcal{C} = (v_1, c_1, \ldots, v_\ell, c_\ell, v_\ell + 1 = v_1)$ a* toxic cycle *of length ℓ if*

TOX1 *c_i contains the variables x_i, x_{i+1} only, and*

TOX2 *the total number of negations in $c_1, \dots c_\ell$ is odd iff ℓ is even.*

▶ **Lemma 22.** **(i)** *If $\boldsymbol{F}'_{\mathrm{UC},t+1}$ contains a toxic cycle, then a conflict occurs in iteration $t+1$.*

(ii) *If $\boldsymbol{F}'_{\mathrm{UC},t+1}$ contains no toxic cycle and $|E(G(\boldsymbol{F}'_{\mathrm{UC},t+1}))| \le |V(G(\boldsymbol{F}'_{\mathrm{UC},t+1}))|$, then no conflict occurs in iteration $t+1$.*

Proof. Towards (i) we show that $\boldsymbol{F}'_{\mathrm{UC},t+1}$ is not satisfiable if there is a toxic cycle $\mathcal{C} = (v_1, c_1, \dots, c_\ell, v_{\ell+1} = v_1)$; then UCP will, of course, run into a contradiction. To see that $\boldsymbol{F}'_{\mathrm{UC},t+1}$ is unsatisfiable, we transform each of the clauses $c_1, \dots, c_\ell$ into a linear equation $c_i \equiv (v_i + v_{i+1} = y_i)$ over $\mathbb{F}_2$. Here $y_i \in \mathbb{F}_2$ equals 1 iff c_i contains an even number of negations. Adding these equations up yields $\sum_{i=1}^{\ell} y_i = 0$ in $\mathbb{F}_2$. This condition is violated if $\mathcal{C}$ is toxic.

Let us move on to (ii). Assume for contradiction that there exists a formula F without a toxic cycle such that $|V(G(F))| \le |E(G(F))|$ and such that given $\boldsymbol{F}'_{\mathrm{UC},t+1} = F$, UCP may run into a conflict. Consider such a formula F that minimises $|V(F)| + |C(F)|$. Since UCP succeeds on acyclic F, we have $|V(G(F))| = |E(G(F))|$. Thus, $G(F)$ contains a single cycle $\mathcal{C} = (v_1, c_1, \dots, v_\ell, c_\ell, v_{\ell+1} = v_1)$. Apart from the cycle, F contains (possibly empty) acyclic formulas $F'_1, \dots, F'_\ell$ attached to $v_1, \dots, v_\ell$ and $F''_1, \dots, F''_\ell$ attached to $c_1, \dots, c_\ell$. The formulas $F'_1, F''_1, \dots, F'_\ell, F''_\ell$ are mutually disjoint and do not contain unit clauses.

We claim that $F'_1, \dots, F'_\ell$ are empty because $|V(F)| + |C(F)|$ is minimum. This is because given any truth assignment of $v_1, \dots, v_\ell$, UCP will find a satisfying assignment of the acyclic formulas $F'_1, \dots, F'_\ell$.

Further, assume that one of the formulas $F''_1, \dots, F''_\ell$ is non-empty; say, F''_1 is non-empty. If the start variable that UCP assigns were to belong to F''_1, then c_1, containing x_1 and x_2, would not shrink to a unit clause, and thus UCP would not assign values to these variables. Hence, UCP starts by assigning a truth value to one of the variables $v_1, \dots, v_\ell$; say, UCP starts with v_1. We claim that then UCP does not run into a conflict. Indeed, the clauses $c_2, \dots, c_\ell$ may force UCP to assign truth values to $x_2, \dots, x_\ell$, but no conflict can ensue because UCP will ultimately satisfy c_1 by assigning appropriate truth values to the variables of F''_1.

Thus, we may finally assume that all of $F'_1, F''_1, \dots, F'_\ell, F''_\ell$ are empty. In other words, F consists of the cycle $\mathcal{C}$ only. Since $\mathcal{C}$ is not toxic, **TOX2** does not occur. Consequently, UCP will construct an assignment that satisfies all clauses $c_1, \dots, c_\ell$. This final contradiction implies (ii). ◀

▶ **Corollary 23.** *Fix $\varepsilon > 0$ and let $0 \le t \le (1-\varepsilon)n$. Then*

$$\mathbb{P}\left[\mathcal{R}_{t+1}\right] = \mathbb{P}\left[\boldsymbol{F}'_{\mathrm{UC},t+1} \text{ contains a toxic cycle}\right] + o(1/n).$$

Proof. This is an immediate consequence of Lemma 20 and Lemma 22. ◀

Thus, we are left to calculate the probability that $\boldsymbol{F}'_{\mathrm{UC},t+1}$ contains a toxic cycle. To this end, we estimate the number of toxic cycles in the 'big' formula $\boldsymbol{F}_{\mathrm{UC},t}$. Let $\boldsymbol{T}_{t,\ell}$ be the number of toxic cycles of length ℓ in $\boldsymbol{F}_{\mathrm{UC},t}$.

▶ **Lemma 24.** *Fix $\varepsilon > 0$ and let $1 \le t \le (1-\varepsilon)n$.*

(i) *For any fixed ℓ, with probability $1 - O(n^{-2})$ we have*

$$\mathbb{E}\left[\boldsymbol{T}_t(\ell) \mid \mathfrak{F}_t\right] = \beta_\ell + o(1), \quad \text{where } \beta_\ell = \frac{1}{4\ell}\left(d(k-1)(1-\alpha_*)\alpha_*^{k-2}\right)^\ell = \frac{1}{4\ell}\left(f_n(t)\right)^\ell.$$

(ii) *For any $1 \le \ell \le n$, with probability $1 - O(n^{-2})$ we have $\mathbb{E}\left[\boldsymbol{T}_t(\ell) \mid \mathfrak{F}_t\right] \le \beta_\ell \exp(\varepsilon\ell)$.*

The proof of Lemma 24 is provided in the full version.

Proof of Proposition 17. In light of Corollary 23 we just need to calculate the probability that $\boldsymbol{F}'_{\mathrm{UC},t+1}$ contains a toxic cycle. Clearly, if during iteration $t+1$ `UCP` encounters a variable of $\boldsymbol{F}_{\mathrm{UC},t}$ that lies on a toxic cycle, `UCP` will proceed to add the entire toxic cycle to $\boldsymbol{F}'_{\mathrm{UC},t+1}$ (and run into a contradiction). Furthermore, Lemma 24 shows that with probability $1-O(n^{-2})$ given $\mathfrak{F}_t$ the probability that a random variable of $\boldsymbol{F}_{\mathrm{UC},t}$ belongs to a toxic cycle comes to

$$\bar{\beta} = \sum_{\ell\geq 2}\ell\beta_\ell + o(1) = \sum_{\ell\geq 2}\frac{1}{4}\left(f_n(t)\right)^\ell = \frac{f_n(t)^2}{4(1-f_n(t))} + o(1) = O(1). \tag{2.25}$$

We now use (2.25) to calculate the desired probability of encountering a toxic cycle. To this end we notice that the $(t+1)$-st iteration of `UCP` corresponds to a branching process with expected offspring $f_n(t)$, unless the root variable x_{t+1} has already been assigned. With probability $1-O(n^{-2})$ the conditional probability of this latter event equals $(n\alpha_* - t)/(n-t)+o(1)$. Further, given that the root variable has not been assigned previously, the expected progeny of the branching process, i.e., the expected number of variables in $\boldsymbol{F}'_{\mathrm{UC},t+1}$, equals $1/(1-f_n(t))+o(1)$. Since with probability $1-O(n^{-2})$ given $\mathfrak{F}_t$ there remain $\boldsymbol{n}(t) = (1-\alpha_*+o(1))n$ unassigned variables in total, (2.25) implies that with probability $1-o(1/n)$,

$$\mathbb{P}\left[\mathcal{R}_{t+1}\mid\mathfrak{F}_t\right] \sim \frac{\bar{\beta}}{(1-\alpha_*)n}\cdot\frac{1-\alpha_*}{1-t/n}\cdot\frac{1}{1-f_n(t)} = \frac{f_n(t)^2}{4(1-f_n(t))^2(n-t)} + o(1/n),$$

as claimed. ◀

3 Discussion

The thrust of the present work is to verify the predictions from [24] on the `BPGD` algorithm and the decimation process rigorously. Concerning the decimation process, the main gap in the deliberations of Ricci-Tersenghi and Semerjian [24] that we needed to plug is the proof of Proposition 11 on the actual number of null variables in the decimation process. The proof of Proposition 11, in turn, hinges on the formula for the nullity from Proposition 9, whereas Ricci-Tersenghi and Semerjian state the (as it turns out, correct) formulas for the nullity and the number of null variables based on purely heuristic arguments.

Regarding the analysis of the `BPGD` algorithm, Ricci-Tersenghi and Semerjian state that they rely on the heuristic techniques from the insightful article [10] to predict the formula (1.7), but do not provide any further details; the article [10] principally employs heuristic arguments involving generating functions. By contrast, the method that we use to prove (1.7) is a bit more similar to that of Frieze and Suen [12] for the analysis of a variant of the unit clause algorithm on random k-SAT instances, for which they also obtain the asymptotic success probability. Yet by comparison to the argument of Frieze and Suen, we pursue a more combinatorially explicit approach that demonstrates that certain small sub-formulas that we call 'toxic cycles' are responsible for the failure of `BPGD`. Specifically, the proof of (1.7) combines the method of differential equations with Poissonisation. Finally, the proof of Theorem 1 (ii) is an easy afterthought of the analysis of the decimation process.

Yung's work [25] on random k-XORSAT is motivated by the 'overlap gap paradigm' [13], the basic idea behind which is to show that a peculiar clustered geometry of the set of solutions is an obstacle to certain types of algorithms. Specifically, Yung only considers the

Unit Clause Propagation algorithm and (a truncated version of) BPGD. Following the path beaten in [19], Yung performs moment computations to establish the overlap gap property. However, moment computations (also called 'annealed computations' in physics jargon) only provide one-sided bounds. Yung's results require spurious lower bounds on the clause length k ($k \geq 9$ for Unit Clause and $k \geq 13$ for BPGD). By contrast, the present proof strategy pivots on the number of null variables rather than overlaps, and Proposition 11 provides the precise 'quenched' count of null variables. A further improvement over [25] is that the present analysis pinpoints the *precise* threshold up to which BPGD (as well as Unit Clause) succeeds for any $k \geq 3$. Specifically, Yung proves that these algorithms fail for $d > d_{\mathrm{core}}$, while Theorem 1 shows that failure occurs already for $d > d_{\mathrm{min}}$ with $d_{\mathrm{min}} < d_{\mathrm{core}}$. Conversely, Theorem 1 shows that the algorithms succeed with a non-vanishing probability for $d < d_{\mathrm{min}}$. Thus, Theorem 1 identifies the correct threshold for the success of BPGD, as well as the correct combinatorial phenomenon that determines this threshold, namely the onset of reconstruction in the decimation process (Theorems 2 and 3).

The BPGD algorithm as detailed in Section 2.2 applies to a wide variety of problems beyond random k-XORSAT. Of course, the single most prominent example is random k-SAT. Lacking the symmetries of XORSAT, random k-SAT does not allow for the simplification to discrete messages; in particular, the BP messages are not generally half-integral. In effect, BP and WP are no longer equivalent. In addition to random k-XORSAT, the article [24] also provides a heuristic study of BPGD on random k-SAT. But once again due to the lack of half-integrality, the formulas for the phase transitions no longer come as elegant finite-dimensional expressions. Instead, they now come as infinite-dimensional variational problems. Furthermore, the absence of half-integrality also entails that the present proof strategy does not extend to k-SAT.

The lack of inherent symmetry in random k-SAT can partly be compensated by assuming that the clause length k is sufficiently large (viz. larger than some usually unspecified constant k_0). Under this assumption the random k-SAT version of both the decimation process and the BPGD algorithm have been analysed rigorously [7, 9]. The results are in qualitative agreement with the predictions from [24]. In particular, the BPGD algorithm provably fails to find satisfying assignments on random k-SAT instances even below the threshold where the set of satisfying assignments shatters into well-separated clusters [1, 16]. Furthermore, on random k-SAT a more sophisticated message passing algorithm called Survey Propagation Guided Decimation has been suggested [20, 24]. While on random XORSAT Survey Propagation and Belief Propagation are equivalent, the two algorithms are substantially different on random k-SAT. One might therefore hope that Survey Propagation Guided Decimation outperforms BPGD on random k-SAT and finds satisfying assignments up to the aforementioned shattering transition. A negative result to the effect that Survey Propagation Guided Decimation fails asymptotically beyond the shattering transition point for large enough k exists [14]. Yet a complete analysis of Belief/Survey Propagation Guided Decimation on random k-SAT for any $k \geq 3$ in analogy to the results obtained here for random k-XORSAT remains an outstanding challenge.

Finally, returning to random k-XORSAT, a question for future work may be to investigate the performance of various types of algorithms such as greedy, message passing or local search that aim to find an assignment that violates the least possible number of clauses. Of course, this question is relevant even for $d > d_{\mathrm{sat}}(k)$. A first step based on the heuristic 'dynamical cavity method' was recently undertaken by Maier, Behrens and Zdeborová [17].

References

1. Dimitris Achlioptas and Amin Coja-Oghlan. Algorithmic barriers from phase transitions. In *Proc. 49th FOCS*, pages 793–802, 2008.
2. Dimitris Achlioptas and Micheal Molloy. The solution space geometry of random linear equations. *Random Structures & Algorithms*, 46:197–231, 2015.
3. Peter Ayre, Amin Coja-Oghlan, Pu Gao, and Noëla Müller. The satisfiability threshold for random linear equations. *Combinatorica*, 40:179–235, 2020.
4. Béla Bollobás. *Random Graphs*. Cambridge University Press, 2001.
5. Alfredo Braunstein, Marc Mézard, and Riccardo Zecchina. Survey propagation: An algorithm for satisfiability. *Random Structures & Algorithms*, 27:201–226, 2005.
6. Amin Coja-Oghlan. A better algorithm for random k-sat. *SIAM Journal on Computing*, 39:2823–2864, 2010.
7. Amin Coja-Oghlan. Belief propagation guided decimation fails on random formulas. *Journal of the ACM*, 63(49), 2017.
8. Amin Coja-Oghlan, Alperen A. Ergür, Pu Gao, Samuel Hetterich, and Maurice Rolvien. The rank of sparse random matrices. *Random Structures & Algorithms*, 62:68–130, 2023.
9. Amin Coja-Oghlan and Angelica Pachon-Pinzon. The decimation process in random k-sat. *SIAM Journal on Discrete Mathematics*, 26:1471–1509, 2012.
10. Christophe Deroulers and Rémi Monasson. Criticality and universality in the unit-propagation search rule. *European Physical Journal B.*, 49:339–369, 2006.
11. Olivier Dubois and Jacques Mandler. The 3-xorsat threshold. In *Proc. 43rd FOCS*, pages 769–778, 2002.
12. Alan Frieze and Stephen Suen. Analysis of two simple heuristics on a random instance of k-sat. *Journal of Algorithms*, 20:312–355, 1996.
13. David Gamarnik. The overlap gap property: a topological barrier to optimizing over random structures. *Proceeding of the National Academy of Sciences*, 118, 2021.
14. Samuel Hetterich. Analysing survey propagation guided decimation on random formulas. In *Proc. 43rd ICALP*, number 65, 2016.
15. Morteza Ibrahimi, Yash Kanoria, Matt Kraning, and Andrea Montanari. The set of solutions of random xorsat formulae. *Annals of Applied Probability*, 25:2743–2808, 2015.
16. Florent Krzakala, Andrea Montanari, Federico Ricci-Tersenghi, Guilhem Semerjian, and Lenka Zdeborová. Gibbs states and the set of solutions of random constraint satisfaction problems. *Proceeding of the National Academy of Sciences*, 104:10318–10323, 2007.
17. Aude Maier, Freya Behrens, and Lenka Zdeborová. Dynamical cavity method for hypergraphs and its application to quenches in the k-xorsat problem. 2024. arXiv:2412.14794.
18. Marc Mézard and Andrea Montanari. *Information, Physics and Computation*. Oxford University Press, 2009.
19. Marc Mézard, Thierry Mora, and Riccardo Zecchina. Clustering of solutions in the random satisfiability problem. *Physical Review Letters*, 94, 2005.
20. Marc Mézard, Giorgio Parisi, and Riccardo Zecchina. Analytic and algorithmic solution of random satisfiability problems. *Science*, 297:812–815, 2002.
21. Marc Mézard, Federico Ricci-Tersenghi, and Riccardo Zecchina. Two solutions to diluted p-spin models and xorsat problems. *Journal of Statistical Physics*, 111:505–533, 2003.
22. Michael Molloy. Cores in random hypergraphs and boolean formulas. *Random Structures & Algorithms*, 27:124–135, 2005.
23. Boris Pittel and Gregory B. Sorkin. The satisfiability threshold for k-xorsat. *Combinatorics, Probability and Computing*, 25:236–268, 2016.
24. Federico Ricci-Tersenghi and Guilhem Semerjian. On the cavity method for decimated random constraint satisfaction problems and the analysis of belief propagation guided decimation algorithms. *Journal of Statistical Mechanics*, 2009.
25. Kingsley Yung. Limits of sequential local algorithms on the random k-xorsat problem. In *Proc. 51st ICALP*, number 123, 2024.