

Complete Convergence of Message Passing Algorithms for some Satisfiability Problems

Uriel Feige¹, Elchanan Mossel² and Dan Vilenchik³

¹ Microsoft Research and The Weizmann Institute. urifeige@microsoft.com

² U.C. Berkeley mossel@stat.berkeley.edu

³ Tel Aviv University vilenchi@post.tau.ac.il.

Abstract. Experimental results show that certain message passing algorithms, namely, *survey propagation*, are very effective in finding satisfying assignments in random satisfiable 3CNF formulas. In this paper we make a modest step towards providing rigorous analysis that proves the effectiveness of message passing algorithms for random 3SAT. We analyze the performance of **Warning Propagation**, a popular message passing algorithm that is simpler than survey propagation. We show that for 3CNF formulas generated under the planted assignment distribution, running warning propagation in the standard way works when the clause-to-variable ratio is a sufficiently large constant. We are not aware of previous rigorous analysis of message passing algorithms for satisfiability instances, though such analysis was performed for decoding of Low Density Parity Check (LDPC) Codes. We discuss some of the differences between results for the LDPC setting and our results.

1 Introduction and Results

The effectiveness of some message passing algorithms, in particular **Survey Propagation**, was experimentally shown for "hard" formulas with clause-variable ratio below (yet rather close to) the conjectured satisfiability threshold, ~ 4.2 [3]. In this paper we analyze the performance of **Warning Propagation** (WP for brevity), a simple popular message passing algorithm, when applied to random satisfiable formulas generated under the planted distribution with a constant clause-variable ratio. We show that the standard way of running message passing algorithms – run message passing until convergence, simplify the formula according to the resulting assignment, and satisfy the remaining subformula (if nonempty), if possible, using a simple "off the shelf" heuristic – works for planted random satisfiable formulas with a sufficiently large yet constant clause-variable ratio. We are not aware of previous rigorous analysis of message passing algorithms for non-trivial SAT distributions.

1.1 Different SAT Distributions

A CNF formula over the variables x_1, x_2, \dots, x_n is a conjunction of clauses C_1, C_2, \dots, C_m where each clause is a disjunction of one or more literals. Each literal is

either a variable or its negation. A formula is said to be in k -CNF form if every clause contains exactly k literals. A CNF formula is satisfiable if there is a boolean assignment to the variables s.t. every clause contains at least one literal which evaluates to true. 3SAT is the language of all satisfiable 3CNF formulas. Although 2SAT is known to be in P, 3SAT is one of the most famous NP-complete problems. In [12] it is proved that it is NP-hard to approximate MAX-3SAT (the problem of finding an assignment that satisfies as many clauses as possible) within a ratio better than $7/8$. Given the difficulty of designing algorithms that work well in the worst case, we consider the average case performance of algorithms. One possibility for rigorously modeling average-case instances is to use random models.

Algorithmic theory of random structures has been the focus of extensive research in recent years (see [10] for a survey). As part of this trend, uniformly random 3CNFs (generated by selecting at random $m = m(n)$ clauses over the variables $\{x_1, \dots, x_n\}$) have attracted much attention. Random 3SAT is known to have a sharp satisfiability threshold in the clause-to-variable ratio [9]. Namely, a random 3CNF with clause-to-variable ratio below the threshold is satisfiable **whp** (with high probability, meaning with probability tending to 1 as n goes to infinity) and one with ratio above the threshold is unsatisfiable **whp**. This threshold is not known exactly (and not even known to be independent of n). The threshold is known to be at least 3.52 [14] and at most 4.506 [5].

In this work we mainly consider formulas with large clause-variable ratio. At such ratios almost all 3CNF formulas are not satisfiable, therefore more refined definitions are due. We consider three distributions on 3SAT instances. The first, analogous to the well known random graph model $G_{n,p}$, is the distribution in which every clause, out of $2^3 \binom{n}{3}$ possible clauses, is included with probability $p = p(n)$. We denote this distribution by $\mathcal{P}_{n,p}$. The second distribution is obtained from $\mathcal{P}_{n,p}$ by conditioning on satisfiability, namely $\mathcal{P}_{n,p}^{\text{sat}}[F] = \mathcal{P}_{n,p}[F|S]$ where S is the event that F is satisfiable. Lastly, we consider the planted distribution, $\mathcal{P}_{n,p}^{\text{plant}}$, which is obtained from $\mathcal{P}_{n,p}$ by conditioning on satisfiability by a specific "planted" assignment φ . Equivalently, in $\mathcal{P}_{n,p}^{\text{plant}}$, first an assignment φ to the variables is picked uniformly at random. Then, every clause satisfied by φ is included with probability $p = p(n)$. Throughout, we use φ to denote the planted assignment when the relevant instance is clear from context.

In the context of satisfiability algorithms, $\mathcal{P}_{n,p}^{\text{sat}}$ is arguably the most interesting and natural distribution to study. However, as pointed out frequently, $\mathcal{P}_{n,p}^{\text{sat}}$ seems hard to tackle rigorously and experimentally. The planted 3SAT distribution is an intermediate step towards analyzing $\mathcal{P}_{n,p}^{\text{sat}}$, and is an interesting, quite natural distribution on its own right, the analog of the planted clique, planted bisection, planted coloring, and planted bipartite hypergraphs studied e.g. in [2, 13, 6]. The planted 3SAT distribution is also discussed e.g. in [8, 7]. Our main result (Theorem 2) relates to the planted 3SAT model, but some of our other results (such as Proposition 1 and Corollary 1) are relevant to the $\mathcal{P}_{n,p}^{\text{sat}}$ setting as well.

1.2 3SAT and Factor Graphs

Let \mathcal{F} be a 3CNF formula on n variables and m clauses. The *factor graph* (e.g. [16]) of \mathcal{F} , denoted by $FG(\mathcal{F})$, is the following graph representation of \mathcal{F} . The factor graph is a bipartite multigraph, $FG(\mathcal{F}) = (V_1 \cup V_2, E)$ where $V_1 = \{x_1, x_2, \dots, x_n\}$ (the set of variables) and $V_2 = \{C_1, C_2, \dots, C_m\}$ (the set of clauses). $(x_i, C_j) \in E$ iff x_i appears in C_j . For a 3CNF \mathcal{F} with m clauses it holds that $\#E = 3m$ (as every clause contains exactly 3 variables). To make presentation clearer, we denote by $\#A$ the size of a set A and by $|a|$ the absolute value of a real number a . For simplicity we assume that every clause contains three distinct variables, therefore FG is a graph (no parallel edges).

1.3 Warning Propagation

Warning Propagation (WP) is a simple iterative message passing algorithm, and serves as an excellent intuitive introduction to more involved message passing algorithms such as Belief Propagation [19] and Survey Propagation [3]. These algorithms are based on the *cavity method* in which the messages that a clause (or a variable) receives are meant to reflect a situation in which a "cavity" is formed, namely, the receiving clause (or variable) is no longer part of the formula. Messages in the WP algorithm can be interpreted as "warnings", telling a clause the values that variables will have if the clause "keeps quiet" and does not announce its wishes, and telling a variable which clauses will not be satisfied if the variable does not commit to satisfying them. We now present the algorithm in a formal way.

Let \mathcal{F} be a CNF formula. For a variable x , let $N^+(x)$ be the set of clauses in \mathcal{F} in which x appears positively (namely, as the literal x), and $N^-(x)$ be the set of clauses in which x appears negatively. For a clause C , let $N^+(C)$ be the set of positively appearing variables and respectively $N^-(C)$ the set of negatively appearing ones. There are two types of messages involved in the WP algorithm. Messages sent from a **variable** x_i to a **clause** C_j in which it appears:

$$x_i \rightarrow C_j = \sum_{C_k \in N^+(x_i), k \neq j} C_k \rightarrow x_i - \sum_{C_k \in N^-(x_i), k \neq j} C_k \rightarrow x_i$$

If x_i appears only in C_j then we set the message to 0. The intuitive interpretation of this message should be x_i signals C_j what is currently its favorable assignment by the other clauses it appears in (a positive message means TRUE, negative one means FALSE and a 0 message means undecided). The second type are messages sent from a **clause** C_j to a **variable** x_i appearing in C_j :

$$C_j \rightarrow x_i = \prod_{x_k \in N^+(C_j), k \neq i} I_{<0}(x_k \rightarrow C_j) \prod_{x_k \in N^-(C_j), k \neq i} I_{>0}(x_k \rightarrow C_j)$$

where $I_{<0}(b)$ equals 1 if $b < 0$ and 0 otherwise (and symmetrically $I_{>0}(b)$ for the case $b > 0$). If C_j contains only x_i (which cannot be the case in 3CNF formulas) then the message is set to 1. $C_j \rightarrow x_i = 1$ can be intuitively interpreted as C_j

sending a *warning* to x_i asking it to satisfy C_j (as all other literals signaled C_j that currently they evaluate to FALSE). Lastly, we define the current assignment of a variable x_i to be

$$B_i = \sum_{C_j \in N^+(x_i)} C_j \rightarrow x_i - \sum_{C_j \in N^-(x_i)} C_j \rightarrow x_i$$

If $B_i > 0$ then x_i is assigned TRUE, if $B_i < 0$ then x_i is assigned FALSE, otherwise x_i is UNASSIGNED. Assume some order on the clause-variable messages (e.g. the lexicographical order on pairs of the form (j, i) representing the message $C_j \rightarrow x_i$). Given a vector $\alpha \in \{0, 1\}^{3m}$ in which every entry is the value of the corresponding $C_j \rightarrow x_i$ message, a partial assignment $\psi \in \{TRUE, FALSE, UNASSIGNED\}^n$ can be generated according to the corresponding B_i values (as previously explained).

It would be convenient to think of the messages in terms of the corresponding factor graph. Every undirected edge (x_i, C_j) of the factor graph is replaced with two anti-parallel directed edges, $(x_i \rightarrow C_j)$ associated with the message $x_i \rightarrow C_j$ and respectively the edge $(C_j \rightarrow x_i)$.

Warning Propagation(CNF formula \mathcal{F}):

1. construct the corresponding factor graph $FG(\mathcal{F})$.
2. randomly initialize the clause-variable messages to 0 or 1.
3. repeat until no clause-variable message changed from the previous iteration:
 - 3.a randomly order the edges of $FG(\mathcal{F})$.
 - 3.b update all clause-variable messages $C_j \rightarrow x_i$ according to the random edge order.
4. compute a partial assignment ψ according to the B_i messages.
5. return ψ .

Note that in line 3.b. above when evaluating the clause-variable message along the edge $C \rightarrow x$, $C = (x \vee y \vee z)$, the variable-clause messages concerning this calculation $(z, y \rightarrow C)$ are evaluated on-the-fly using the last updated values $C_i \rightarrow y$, $C_j \rightarrow z$ (allowing feedback from the same iteration). We allow the algorithm not to terminate (the clause-variable messages may keep changing every iteration). If the algorithm does return an assignment ψ then we say that it converged. In practice it is common to limit in advance the number of iterations, and if the algorithm didn't converge by then, return a failure.

1.4 Related Work and Techniques

The Survey Propagation algorithm [3] experimentally outperforms all known algorithms in finding satisfying assignments to $\mathcal{P}_{n,p}$ formulas with clause-variable ratio ρ close to the satisfiability threshold ($4 \leq \rho \leq 4.25$). However, theoretical understanding of Survey Propagation and other message passing algorithm for random SAT problems is still lacking. This should be compared with the success of message passing algorithms for decoding low-density-parity-check (LDPC)

codes [11]. Here, the experimental success of message passing algorithms [11] was recently complemented rigorously by a large body of theoretical work, see e.g. [17, 20, 18]. Some important insights emerge from this theoretical work. In particular, it is shown that the quality of decoding improves exponentially with the number of iterations – thus all but a small constant fraction of the received codeword can be decoded correctly using a constant number of iterations. Our analysis of WP shows that much of the coding picture is valid also for $\mathcal{P}_{n,p}^{\text{plant}}$ thus providing important insights as to the success of message passing algorithms for random satisfiability problems. The planted 3SAT model is similar to LDPC in many ways. Both constructions are based on random factor graphs. In codes, the received corrupted codeword provides noisy information on a single bit or on the parity of a small number of bits of the original codeword. In $\mathcal{P}_{n,p}^{\text{plant}}$, φ being the planted assignment, the clauses containing a variable x_i contain noisy information on the polarity of $\varphi(x_i)$ in the following sense – each clause contains x_i in a polarity coinciding with $\varphi(x_i)$ with probability $4/7$. Our results are similar in flavor to the coding results. However, the combinatorial analysis provided here allows to recover an assignment satisfying *all* clauses, whereas in the random LDPC codes setting, message passing allows to recover only $1 - o(1)$ fraction of the codeword correctly. In [18] it is shown that for the erasure channel, all bits may be recovered correctly using a message passing algorithm, however in this case the LDPC code is designed so that message passing works for it. We on the other hand take a well known SAT distribution and analyze the performance of a message passing algorithm on it. Moreover, the SAT setting is more involved, as there are many assignments satisfying the formula, while for the erasure channel there is a unique codeword satisfying the combinatorial constraints given by the message.

As for relevant results in random graph theory, the seminal work of [2] paved the road towards dealing with large-constant-degree planted distributions. [2] present an algorithm that **whp** k -colors planted k -colorable graphs with a sufficiently large constant expected degree. Building upon the techniques introduced by [2], [13] present an algorithm that 2-colors sparse planted 3-uniform bipartite hypergraphs and [8], solving an open question posed in [15], presents an algorithm for satisfying large constant degree planted 3SAT instances. Though in our analysis we use similar techniques to the aforementioned works, our result is conceptually different in the following sense. In [2, 13, 8] the starting point is the planted distribution, and then one designs an algorithm that works well under this distribution. The algorithm may be designed in such a way that makes its analysis easier. In contrast, our starting point is a given message passing algorithm (WP), and then we ask for which input distributions it works well. We cannot change the algorithm in ways that would simplify the analysis.

Another difference between our work and that of [2, 13, 8] is that unlike the algorithms analyzed in those other papers, WP is a randomized algorithm which makes its analysis more difficult. We could have simplified our analysis had we changed WP to be deterministic (for example, by initializing all clause-variable messages to 1 in step 2 of the algorithm), but there are good reasons why WP

is randomized. For example, it can be shown that (the randomized version) WP converges with probability 1 on 2CNF formulas that form one cycle of implications, but might not converge if step 4 does not introduce fresh randomness in every iteration of the algorithm (details omitted).

1.5 Notation

Given a 3CNF \mathcal{F} , **simplify** \mathcal{F} according to ψ , when ψ is a partial assignment, means: in every clause substitute every assigned variable with the value given to it by ψ . Then remove all clauses containing literals which evaluate to true. In all remaining clauses, remove all literals which evaluate to false (the resulting instance is not necessarily in 3CNF form). Denote by $\mathcal{F}|_\psi$ the 3CNF \mathcal{F} simplified according to ψ . For a set of variables $A \subseteq V$, denote by $\mathcal{F}[A]$ the set of clauses in which all variables belong to A .

Given a 3CNF formula \mathcal{F} , we say that a variable x is **pure** in \mathcal{F} if it appears only in one polarity (namely, always appears as the literal x or always as the literal \bar{x}). Let P_0 be the set of pure variables in \mathcal{F} , and C_0 be the set of clauses containing a pure variable. Let $L_0 = \mathcal{F}$, and $L_1 = L_0 \setminus C_0$. Let P_1 be the pure variables in L_1 , namely the variables that become pure after setting the pure variables in a satisfying manner and simplifying \mathcal{F} . Similarly, define C_1 to be the set of clauses in L_1 containing a variable from P_1 . Generally, define $L_i = L_{i-1} \setminus C_{i-1}$, P_i to be the pure variables in L_i , and C_i to be the clauses in L_i containing a variable from P_i . We say that a 3CNF \mathcal{F} is *r-pure* if $L_r = \emptyset$. The following theorem is implicitly proved in [4].

Theorem 1. *Let \mathcal{F} be randomly sampled according to $\mathcal{P}_{n,p}$, $p = d/n^2$, $d < 1.225$, then **whp** \mathcal{F} is $O(n)$ -pure.*

Note that if there exists an r s.t. \mathcal{F} is r -pure then in particular \mathcal{F} is satisfiable. To better understand our results it would be convenient to have the somewhat informal notion of a *simple formula* in mind. We call a CNF formula simple, if it can be satisfied using simple well-known heuristics (examples include formulas whose factor graph is tree-like and r -pure formulas – both solvable using the pure-literal heuristic [4], formulas with small weight terminators – to use the terminology of [1] – efficiently solvable **whp** using a RWalkSat, etc).

1.6 Our Results

Theorem 2. *Let \mathcal{F} be a 3CNF formula randomly sampled according to $\mathcal{P}_{n,p}^{\text{plant}}$, where $p \geq d/n^2$, d a sufficiently large constant, and let φ be its planted assignment. Then the following holds with probability $1 - e^{-\Theta(d)}$ (the probability taken over the choice of \mathcal{F} , the random choices in line 2 of the WP algorithm, and the random order in the first time line 4 executes):*

1. $WP(\mathcal{F})$ converges after at most $O(\log n)$ iterations.

2. Let ψ be the partial assignment returned by $WP(\mathcal{F})$, let V_A denote the variables assigned to either *TRUE* or *FALSE* in ψ , and V_U the variables left *UNASSIGNED*. Then for every variable $x \in V_A$, $\psi(x) = \varphi(x)$. Moreover, $\#V_A \geq (1 - e^{-\Theta(d)})n$.
3. $\mathcal{F}|_{\psi[V_U]}$ is a simple formula which can be satisfied in time $O(n)$.

Remark 1. We also have a proof of Theorem 2 with ‘ $1 - o(1)$ ’ instead of ‘ $1 - e^{-\Theta(d)}$ ’. This however involves a somewhat more complicated analysis exceeding the scope of this abstract. For the full details the reader is referred to the journal version.

Proposition 1. *Let \mathcal{F} be a 3CNF formula randomly sampled according to $\mathcal{P}_{n,p}^{\text{plant}}$, where $p \geq c \log n/n^2$, with c a sufficiently large constant, and let φ be its planted assignment. Then **whp** after at most 2 iterations $WP(\mathcal{F})$ converges, and the returned ψ equals φ . (This result can be extended to $\mathcal{P}_{n,p}^{\text{sat}}$, see below.)*

Proposition 2. *Let \mathcal{F} be an r -pure CNF formula. Then after at most $O(r)$ iterations of $WP(\mathcal{F})$, regardless of the initial messages and the order of execution, the following holds:*

1. $WP(\mathcal{F})$ converges.
2. Let ψ be the assignment returned by $WP(\mathcal{F})$. If $\psi(x) \neq \text{UNASSIGNED}$, then in every satisfying assignment x is assigned according to $\psi(x)$.
3. If \mathcal{F} contains no unit clauses then ψ is the all-*UNASSIGNED* vector.

Corollary 1. *In the setting of Theorem 1, **whp** $WP(\mathcal{F})$ converges after at most $O(n)$ iterations and the returned ψ is the all-*UNASSIGNED* vector.*

The corollary follows immediately from Theorem 1 and Proposition 2.

Proposition 3. *Let \mathcal{F} be a satisfiable CNF formula whose corresponding factor graph contains no cycles. Then \mathcal{F} is $O(n)$ -pure.*

The main idea behind the proof of Theorem 2 is to show that the formula is dense enough so that **whp** there exists a large subformula forcing WP to point in the correct direction. The rest of the formula induces a factor graph containing only trees, which are also “easy” for WP . We note that formulas in $\mathcal{P}_{n,p}^{\text{plant}}$, with n^2p some large constant, are not known to be simple (in the sense that we defined above). On the contrary, “hardness” evidence can be found in works such as [1], showing that $RWalkSat$ is very unlikely to hit a satisfying assignment in polynomial time when running on a random $\mathcal{P}_{n,p}^{\text{plant}}$ instance in the setting of Theorem 2. In the setting of Proposition 1, the formula is already dense enough so that **whp** it forces entirely WP to point to the planted assignment.

Proposition 2 combined with Proposition 3 provide a proof to the convergence of WP on trees. Our proof of this known result gives an explicit characterization of the fixed point to which WP converges (which is implicit for trees in [3]).

The remainder of the paper is structured as follows. In Section 2 we discuss some properties that a typical instance in $\mathcal{P}_{n,p}^{\text{plant}}$ possesses, and outline the proof of Theorem 2 and Proposition 1. In Section 3 we summarize our results and discuss potentially interesting lines for further research. Most details of the proofs are omitted and can be found in the journal version.

2 Properties of a Random $\mathcal{P}_{n,p}^{\text{plant}}$ Instance

In this section we discuss relevant properties of a random $\mathcal{P}_{n,p}^{\text{plant}}$ instance. To simplify presentation, we assume w.l.o.g. (due to symmetry) that the planted assignment φ is the all-one vector.

2.1 Stable Variables

Definition 1. A variable x **supports** a clause C with respect to a partial assignment ψ , if it is the only variable to satisfy C under ψ , and the other two variables are assigned by ψ .

Proposition 4. Let \mathcal{F} be as in the setting of Theorem 2 and let F_{SUPP} be a random variable counting the number of variables in \mathcal{F} whose support w.r.t. φ is less than $d/3$. Then, $E[F_{SUPP}] \leq e^{-\Theta(d)}n$.

This follows from concentration arguments as every variable is expected to support $\frac{d}{n^2} \cdot \binom{n}{2} = \frac{d}{2} + O(\frac{1}{n})$ clauses.

Following the definitions in Section 1.3, given a CNF \mathcal{F} and a variable x , we let $N^{++}(x)$ be the set of clauses in \mathcal{F} in which x appears positively but doesn't support w.r.t. φ . Let $N^s(x)$ be the set of clause in \mathcal{F} which x supports w.r.t. φ . Let $\pi = \pi(\mathcal{F})$ be some ordering of the clause-variable message edges in the factor graph of \mathcal{F} . For an index i and a literal ℓ_x (by ℓ_x we denote a literal over the variable x) let $\pi^{-i}(\ell_x)$ be the set of clause-variable edges ($C \rightarrow x$) that appear before index i in the order π and in which x appears in C as ℓ_x . For a set of clause-variable edges \mathcal{E} and a set of clauses \mathcal{C} we denote by $\mathcal{E} \cap \mathcal{C}$ the subset of edges containing a clause from \mathcal{C} as one endpoint.

Definition 2. A variable x is **stable** in \mathcal{F} w.r.t. an edge order π if the following holds for every clause-variable edge $C \rightarrow x$ (w.l.o.g. assume $C = (\ell_x \vee \ell_y \vee \ell_z)$, $C \rightarrow x$ is the i 'th message in π):

1. $|\#\pi^{-i}(y) \cap N^{++}(y) - \#\pi^{-i}(\bar{y}) \cap N^-(y)| \leq d/30$.
2. $|\#N^{++}(y) - \#N^-(y)| \leq d/30$.
3. $\#N^s(y) \geq d/3$

and the same holds for z .

Proposition 5. Let \mathcal{F} be as in the setting of Theorem 2, and let π be a random ordering of the clause-variable messages. Let F_{UNSTAB} be a random variable counting the number of variables in \mathcal{F} which are not stable. Then, $E[F_{UNSTAB}] \leq e^{-\Theta(d)}n$.

This follows from concentration arguments since $E[\#\pi^{-i}(y) \cap N^{++}(y) - \#\pi^{-i}(\bar{y}) \cap N^-(y)] = 0$, $E[\#N^{++}(y) - \#N^-(y)] = 0$, and since every variable is expected to appear in at most $O(d)$ clauses.

Let $\alpha \in \{0, 1\}^{3\#\mathcal{F}}$ be a clause-variable message vector. For a set of clause-variable message edges \mathcal{E} let $\mathbf{1}_\alpha(\mathcal{E})$ be the set of edges along which the value is

1 according to α . For a set of clauses \mathcal{C} , $\mathbf{1}_\alpha(\mathcal{C})$ denotes the set of clause-variable message edges in the factor graph of \mathcal{F} containing a clause from \mathcal{C} as one endpoint and along which the value is 1 in α .

Definition 3. A variable x is **violated** by α in π if there exists a message $C \rightarrow x$, $C = (\ell_x \vee \ell_y \vee \ell_z)$, in place i in π s.t. one of the following holds:

1. $|\#\mathbf{1}_\alpha(\pi^{-i}(y) \cap N^{++}(y)) - \#\mathbf{1}_\alpha(\pi^{-i}(\bar{y}) \cap N^-(y))| > d/30$
2. $|\#\mathbf{1}_\alpha(N^{++}(y)) - \#\mathbf{1}_\alpha(N^-(y))| > d/30$
3. $\#\mathbf{1}_\alpha(N^s(y)) < d/7$.

Or one of the above holds for z .

Proposition 6. Let \mathcal{F} be as in the setting of Theorem 2, and let X be a set of stable variables w.r.t. an arbitrary ordering π . Let α be a random clause-variable message vector. Let F_{VIO} be a random variable counting the number of violated variables in X . Then, $E[F_{VIO}] \leq e^{-\Theta(d)}\#X$.

The proof again uses concentration arguments.

2.2 Dense Subformulas

The next property we discuss is analogous to a property proved in [2] for random graphs. Loosely speaking, [2] prove that **whp** a random graph doesn't contain a small induced subgraph with a large average degree. Using first moment calculations we show:

Proposition 7. Let $c > 1$ be an arbitrary constant. Let $p \geq d/n^2$, where d is a large constant. Then **whp** over $\mathcal{F} \in \mathcal{P}_{n,p}^{\text{plant}}$ as $n \rightarrow \infty$ there exists no subset of variables U , s.t. $\#U \leq e^{-\Theta(d)}n$ and there are at least $c\#U$ clauses in \mathcal{F} containing two variables from U .

2.3 The Core Variables

We describe a subset of the variables, denoted throughout by \mathcal{H} and referred to as the *core variables*, which plays a crucial role in the analysis. Loosely speaking, a variable is considered "safe" if it is stable w.r.t. the initial random order π , and it is not violated by the initial clause-variable message assignments α . If in addition, a safe variable x_i supports many clauses w.r.t. φ (whose corresponding message is '1' in α), then its corresponding B_i value will agree with $\varphi(x_i)$ after the first iteration. This invariant needs to be preserved however in later iterations. The set \mathcal{H} captures the notion of such variables with a self-preserving quality. There are several ways to obtain these desired properties. Formally, $\mathcal{H} = \mathcal{H}(\mathcal{F}, \varphi, \alpha, \pi)$ is constructed using the following iterative procedure:

Let A_1 be the set of variables whose support w.r.t. φ is at most $d/3$.

Let A_2 be the set of non-stable variables w.r.t. π .

Let A_3 be the set of stable variables w.r.t. π violated by α .

1. Set $H_0 = V \setminus (A_1 \cup A_2 \cup A_3)$.
2. While $\exists a_i \in H_i$
 supporting less than $d/4$ clauses in $\mathcal{F}[H_i]$ OR
 appearing in more than $d/30$ clauses not in $\mathcal{F}[H_i]$: let $H_{i+1} = H_i \setminus \{a_i\}$.
3. Define $\mathcal{H} = H_{m+1}$ where $a_m :=$ last variable removed in step 2.

Proposition 8. *If both α and π are chosen uniformly at random then **whp** $\#\mathcal{H} \geq (1 - e^{-\Theta(d)})n$.*

The main idea of the proof is to observe that to begin with we eliminate very few variables (using the discussion in Section 2.1 to bound $\#A_1 \cup A_2 \cup A_3$). If too many variables were removed in the iterative step then a small but dense subformula exists. Proposition 7 bounds the probability of the latter occurring.

2.4 The Factor Graph of the Non-Core Variables

Proposition 8 implies that for $p = c \log n/n^2$, c a sufficiently large constant, **whp** \mathcal{H} contains already all variables. The following analysis is needed for the setting of Theorem 2. The *non-core* factor graph is the factor graph of the formula \mathcal{F} simplified according to the partial assignment that assigns all core variables to their value in the plant.

Proposition 9. ***Whp** every connected component in the non-core factor graph contains $O(\log n)$ variables.*

Proposition 9 will not suffice to prove Theorem 2, and we need a further characterization of the non-core factor graph.

Proposition 10. *With probability $1 - e^{-\Theta(d)}$, there exists no cycle in the non-core factor graph.*

2.5 Outline of Proof of Theorem 2 and Proposition 1

We start with Theorem 2 and derive Proposition 1 as an easy corollary of the analysis. The outline of the proof is as follows. We assume that the formula \mathcal{F} and the run of WP are *typical* in the sense that Propositions 8, 9 and 10 hold. First we prove that after one iteration WP sets the core variables \mathcal{H} correctly (B_i agrees with φ in sign) and this assignment does not change in later iterations. Therefore from iteration 2 and onwards WP is basically running on \mathcal{F} in which variables belonging to \mathcal{H} are substituted with their planted assignment. This subformula is satisfiable and its factor graph is a forest (namely, composed of disjoint trees). Therefore, convergence is guaranteed. The set V_A of Theorem 2 is composed of all variables from \mathcal{H} and those variables from the forest that get assigned. The set V_U is composed of the UNASSIGNED variables from the forest.

We say that a message $C \rightarrow x$, $C = (\ell_x \vee \ell_y \vee \ell_z)$, is *correct* if its value is the same as it is when $y \rightarrow C$ and $z \rightarrow C$ agree in sign with their planted assignment (in other words, $C \rightarrow x$ is 1 iff x supports C w.r.t. φ).

Proposition 11. *If $x_i \in \mathcal{H}$ and all messages $C \rightarrow x_i$, $C \in \mathcal{F}[\mathcal{H}]$ are correct at the beginning of an iteration (line 3 in the WP algorithm), then this invariant is kept by the end of that iteration.*

Proposition 12. *If $x_i \in \mathcal{H}$ and all messages $C \rightarrow x_i$, $C \in \mathcal{F}[\mathcal{H}]$ are correct by the end of a WP iteration, then B_i agrees in sign with $\varphi(x_i)$ by the end of that iteration.*

Proposition 12 follows immediately from the definition of \mathcal{H} and the message B_i . It remains to show then that after the first iteration all messages $C \rightarrow x_i$, $C \in \mathcal{F}[\mathcal{H}]$ are correct.

Proposition 13. *If \mathcal{F} is a typical instance in the setting of Theorem 2, then after one iteration of $WP(\mathcal{F})$, for every variable $x_i \in \mathcal{H}$, every message $C \rightarrow x_i$, $C \in \mathcal{F}[\mathcal{H}]$ is correct.*

Proposition 14. *Let \mathcal{F} be a typical instance in the setting of Theorem 2, then for every variable $x_j \in V \setminus \mathcal{H}$, after $O(\log n)$ iterations either $B_j = 0$ or B_j agrees in sign with $\varphi(x_j)$.*

As for satisfying the set of unassigned variables in time $O(n)$, Propositions 3 and 10 imply that the pure-literal procedure [4] solves the subformula induced by the unassigned variables in linear time. Theorem 2 then follows.

To prove Proposition 1, observe that when $p = c \log n/n^2$, with c a sufficiently large constant, Proposition 8 implies $\mathcal{H} = V$. Combing this with Proposition 13, Proposition 1 readily follows.

3 Discussion

We conclude with an open problem. Can our analysis be extended to show that Belief Propagation (BP) finds a satisfying assignment to $\mathcal{P}_{n,p}^{\text{plant}}$ in the setting of Theorem 2? Experimental results predict the answer to be positive. However, our analysis of WP does not extend as is to BP. In WP, all warnings received by a variable (or by a clause) have equal weight, but in BP this need not be the case (there is a probability level associated with each warning). In particular, this may lead to the case that messages received from non-core portions of the formula can effect the core, a possibility that our analysis managed to exclude for the WP algorithm.

Acknowledgements:

We thank Eran Ofek for many useful discussions. This work was done while the authors were visiting Microsoft Research, Redmond, Washington. E.M is supported by a Sloan fellowship in Mathematics, by NSF Career award DMS-0548249 and NSF grants DMS-0528488 and DMS-0504245.

References

1. M. Alekhnovich and E. Ben-Sasson. Linear upper bounds for random walk on small density random 3-cnf. In *Proc. 44th IEEE Symp. on Found. of Comp. Science*, page 352, 2003.
2. N. Alon and N. Kahale. A spectral technique for coloring random 3-colorable graphs. *SIAM J. on Comput.*, 26(6):1733–1748, 1997.
3. A. Braunstein, M. Mezard, and R. Zecchina. Survey propagation: an algorithm for satisfiability. *Random Structures and Algorithms*, 27:201–226, 2005.
4. A. Z. Broder, A. M. Frieze, and E. Upfal. On the satisfiability and maximum satisfiability of random 3-cnf formulas. In *Proc. 4th ACM-SIAM Symp. on Discrete Algorithms*, pages 322–330, 1993.
5. O. Dubois, Y. Boufkhad, and J. Mandler. Typical random 3-sat formulae and the satisfiability threshold. In *Proc. 11th ACM-SIAM Symp. on Discrete Algorithms*, pages 126–127, 2000.
6. U. Feige and R. Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Structures and Algorithms*, 16(2):195–208, 2000.
7. U. Feige and D. Vilenchik. A local search algorithm for 3SAT. Technical report, The Weizmann Institute of Science, 2004.
8. A. Flaxman. A spectral technique for random satisfiable 3CNF formulas. In *Proc. 14th ACM-SIAM Symp. on Discrete Algorithms*, pages 357–363, 2003.
9. E. Friedgut. Sharp thresholds of graph properties, and the k -sat problem. *J. Amer. Math. Soc.*, 12(4):1017–1054, 1999.
10. A. M. Frieze and C. McDiarmid. Algorithmic theory of random graphs. *Random Structures and Algorithms*, 10(1-2):5–42, 1997.
11. T. G. Gallager. Low-density parity-check codes. *IRE. Trans. Info. Theory*, IT-8:21–28, January 1962.
12. J. Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.
13. C. Hui and A. M. Frieze. Coloring bipartite hypergraphs. In *Proceedings of the 5th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 345–358, London, UK, 1996. Springer-Verlag.
14. A. C. Kaporis, L. M. Kirousis, and E. G. Lalas. The probabilistic analysis of a greedy satisfiability algorithm. In *Proc. 10th Annual European Symposium on Algorithms*, volume 2461 of *Lecture Notes in Comput. Sci.*, pages 574–585. Springer, Berlin, 2002.
15. E. Koutsoupias and C. H. Papadimitriou. On the greedy algorithm for satisfiability. *Info. Process. Letters*, 43(1):53–55, 1992.
16. F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.
17. M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. Spielman. Analysis of low density parity check codes and improved designs using irregular graphs. In *Proceedings of the 30th ACM Symposium on Theory of Computing*, pages 249–258, 1998.
18. M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. Spielman. Efficient erasure correcting codes. *IEEE Trans. Info. Theory*, 47:569–584, February 2001.
19. J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.
20. T. Richardson, A. Shokrollahi, and R. Urbanke. Design of capacity-approaching irregular low-density parity check codes. *IEEE Trans. Info. Theory*, 47:619–637, February 2001.