Combinatorics, Probability and Computing (2010) 19, 775–790. © Cambridge University Press 2010
doi:10.1017/S0963548310000313

On Random Betweenness Constraints

ANDREAS GOERDT
Technische Universität Chemnitz, Fakultät für Informatik,
Straße der Nationen 62, 09107 Chemnitz, Germany
(e-mail: goerdt@informatik.tu-chemnitz.de
http://www.tu-chemnitz.de/informatik/TI/)

Received 4 July 2009; revised 13 August 2010

Ordering constraints are formally analogous to instances of the satisfiability problem in conjunctive normal form, but instead of a boolean assignment we consider a linear ordering of the variables in question. A clause becomes true given a linear ordering if and only if the relative ordering of its variables obeys the constraint considered.

The naturally arising satisfiability problems are NP-complete for many types of constraints. We look at random ordering constraints. Previous work of the author shows that there is a sharp unsatisfiability threshold for certain types of constraints. The value of the threshold, however, is essentially undetermined. We pursue the problem of approximating the precise value of the threshold. We show that random instances of the betweenness constraint are satisfiable with high probability if the number of randomly picked clauses is $\leqslant 0.92n$, where n is the number of variables considered. This improves the previous bound, which is $< 0.82n$ random clauses. The proof is based on a binary relaxation of the betweenness constraint and involves some ideas not used before in the area of random ordering constraints.

1. Introduction

1.1. Result

Let V always be a set of n variables. A 3-clause over V is an *ordered* 3-tuple (x, y, z) consisting of three different variables. Thus we have $n(n-1)(n-2) = (n)_3$ clauses altogether. A formula, also called an ordering constraint, is a set of clauses. Given a linear ordering of all n variables, a clause evaluates to true if its variables satisfy a given constraint with respect to the ordering. A formula becomes true when all its clauses are true. This is the satisfiability problem the present paper considers.

The clause (x, y, z), interpreted as a *betweenness* constraint, is true if and only if y is between x and z, that is, we have $x < y < z$ or $z < y < x$ with respect to the ordering considered. The corresponding satisfiability problem is NP-complete [20].

We consider random ordering constraints interpreted as betweenness constraints. The random instance $F(V, m)$, or the corresponding probability space, is obtained by picking a sequence (or

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

set) of exactly m distinct clauses with uniform probability. Thus $F(V,m)$ is analogous to the well-known random graph $G(n,m)$. More closely related to $F(V,m)$ are random 3-SAT formulas: see for example [6], [3], [9], [18]. The present paper is a successor to [12].

As usual in the theory of random structures, this paper deals with properties holding with high probability, that is, $1-o(1)$ when n becomes large and $m=m(n)$ is a given function. An additional piece of notation: a sequence of events E_n in some probability spaces holds with *uniformly positive probability* (abbreviated as w.u.p.p.) if there is a constant $\varepsilon>0$ such that $\mathrm{Prob}[E_n]>\varepsilon$ for all sufficiently large n.

The probability space $F(V,p)$ is obtained by picking each clause independently with probability p. We call it the binomial space. For $p=r/n^2$ the expected number of clauses is $rn(1+O(1/n))$. Moreover, the number of clauses is $rn(1+o(1))$ with high probability. Techniques as detailed on pp. 34–35 of [5] show that the spaces $F(n,m)$ and $F(n,p)$ with $pn^3=m=O(n)$ are equivalent for most questions of interest. This applies in particular to the satisfiability problems treated here, as they are monotone problems. Following common usage we omit the technical details to show this each time.

The initial inspiration for the paper [12] came from some experiments (performed only for $n\leqslant 300$ for running time reasons). These experiments show that the random betweenness constraint becomes unsatisfiable for rn random clauses when r is between 1.5 and 1.6. Results obtained in the cited paper are collected here.

Fact 1.1. *For the random betweenness instance $F(V,m)$ with $m=rn$, the following events have high probability.*

(a) *For $r\leqslant C$ the instance is satisfiable where $C<0.82$.*
(b) *For $r>4\ln 2\approx 2.77$ the instance is unsatisfiable.*
(c) *There exist numbers $C=C(n)$, $0.8\leqslant C\leqslant 2.77$, such that for each constant $\varepsilon>0$ we have unsatisfiability for $r\geqslant C+\varepsilon$ and satisfiability for $r\leqslant C-\varepsilon$.*

Fact 1.1(c) means that we have a sharp threshold for unsatisfiability, but we do not know the threshold value precisely. This is typical when the techniques from [11] are used. Fact 1.1(a) relies on the fact that formulas (or 3-uniform hypergraphs) with the given number of edges do not have a non-empty 2-core with high probability. This is known, by [18]. The 2-core is the maximum subformula in which each variable occurs at least twice, and formulas with empty 2-core can be trivially satisfied. It appears to be non-trivial to show that $F(V,m)$ is satisfiable with high probability for any m substantially larger than $0.82n$. We make some progress and prove the following.

Theorem 1.2. *For $m=rn$ with constant $r\leqslant 0.921$, the random betweenness instance $F(V,m)$ is satisfiable w.u.p.p.*

Theorem 1.2 together with Fact 1.1(c) implies a high probability result.

Corollary 1.3. *The random betweenness constraint with $m=rn$ and $r<0.921$ is satisfiable with high probability.*

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

There are two different techniques to show that random structures are solvable (for example, colourable in the case of graphs or satisfiable in the case of k-SAT instances). On the one hand, it has been successful analysing heuristic algorithms and showing that they find a solution to a random instance. On the other hand, and more recently, non-constructive methods have been shown to be successful. In [1] it is shown that random k-SAT instances are satisfiable based on the second moment of the number of solutions and general probability estimates. Our proof consists of a first non-constructive part, based on the second moment, and a second constructive part.

1.2. More remarks on the literature

Ordering constraints differ from traditional constraints such as k-SAT or more general kinds of constraints in that the underlying assignment must be an *ordering* of all variables. This means on the one hand that each variable can receive one out of n values, its position in the ordering. On the other hand each of the n values can only be used once. Altogether we have $n! \gg 2^n$ many assignments as opposed to only 2^n in the case of satisfiability.

Beyond random k-SAT there is a considerable body of work on random constraints with finite domain from which the values for each variable are taken. A small selection of the literature, in part due to Michael Molloy, is [17], [19], [14]. The paper [17] points out that the investigation of thresholds is not only of structural interest, but also has algorithmic relevance: random instances at thresholds often have some algorithmic hardness which makes them attractive as test cases for algorithms.

As far as we know, systematic experimental studies of random ordering constraints have not been made. Our preliminary experiments indicate that instances closer to the threshold become harder. This shows that our study is relevant from the algorithmic point of view.

Ordering constraints tend to occur in knowledge representation formalisms. For example, in [15] the cyclic ordering constraint occurs. In [8] a weighted version of an extended betweenness constraint is used to describe some biological situations. Ordering constraints are investigated in [13] from the point of view of worst-case complexity. A recent breakthrough in the classification of the complexity of more general kinds of ordering constraints can be found in [4]. The papers [2] and [7] consider optimization versions of ordering constraints.

2. Outline of the proof of Theorem 1.2

We recall the following theorem.

Theorem 1.2. *For $m = rn$ with constant $r \leqslant 0.921$, the random betweenness instance $F(V, m)$ is satisfiable w.u.p.p.*

A 2-clause is simply a pair of distinct variables $x < y$, and we have $n(n-1) = (n)_2$ 2-clauses altogether. Given an ordering of the variables, the 2-clause is satisfied if and only if x is smaller than y. A boolean assignment of the set of variables V is an assignment $a : V \to \{0, 1\}$ such that $n/2$ variables receive the value 1 and $n/2$ the value 0. Thus, in our case boolean assignments are *balanced*. A clause (x, y, z) is satisfied in the *boolean sense* by a if and only if it does *not* evaluate to $(0, 1, 0)$ or to $(1, 0, 1)$. Thus we have six out of eight different possibilities for satisfying a clause in the boolean sense. A formula is satisfied in the boolean sense by a if and only if each clause is

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

satisfied by a. A boolean assignment is equivalent to a partition of V into two sets V_0 and V_1, each with $n/2$ variables: V_0 the set of variables set to 0 and V_1 the set of variables set to 1. Let A be the ordering $y_1 < y_2 < y_3 < \cdots < y_n$. The ordering A induces the partition $V_l = \{y_1, \ldots, y_{n/2}\}$ (the lower half of the ordering) and V_u, the upper half of the ordering. These notations directly imply the following result.

Proposition 2.1. *If the betweenness constraint F is satisfied by the ordering A, then F is satisfied by the boolean assignment equivalent to the partition V_l and V_u.*

The following reduction allows us to shrink a given betweenness constraint.

Definition 2.2. Let F be a betweenness constraint and let V_0, V_1 be a partition of V into two disjoint sets of $n/2$ variables each. If F is satisfied by the boolean assignment equivalent to V_0 and V_1, the constraints F_0 over V_0 and F_1 over V_1 are defined.

Let (x, y, z) be a clause from F with at least two variables from V_0. It induces clauses as follows in F_0:

- $(x, y, z) \in (V_0, V_0, V_0)$ $(= V_0 \times V_0 \times V_0)$ implies $(x, y, z) \in F_0$,
- $(x, y, z) \in (V_0, V_0, V_1)$ implies $x < y \in F_0$,
- $(x, y, z) \in (V_1, V_0, V_0)$ implies $y > z \in F_0$.

Let (x, y, z) be a clause from F with at least two variables from V_1. It induces clauses as follows in F_1:

- $(x, y, z) \in (V_1, V_1, V_1)$ implies $(x, y, z) \in F_1$,
- $(x, y, z) \in (V_1, V_1, V_0)$ implies $x > y \in F_1$,
- $(x, y, z) \in (V_0, V_1, V_1)$ implies $y < z \in F_1$.

F in the preceding definition has no clauses from (V_0, V_1, V_0) and (V_1, V_0, V_1) as it is satisfied by the boolean assignment associated to V_0 and V_1. The simple relationship between F and F_0 and F_1 is made clear as follows.

Proposition 2.3. *Let F, V_0, and V_1 be such that F_0 and F_1 are defined. F is satisfied by a linear ordering with $V_0 < V_1$ if and only if F_0 and F_1 are both satisfiable.*

We consider the random instance $F = F(V, m)$. Given a boolean assignment a we define indicator random variables X_a and Y_a: $X_a(F) = 1$ if F is satisfied in the boolean sense by a. $X = \sum_a X_a$ is the number of satisfying boolean assignments. $Y_a(F) = 1$ if and only if F is satisfied by an ordering A which induces the *same partition* as a. (That is, $V_l = V_0$ and $V_u = V_1$.) We let $Y = \sum_a Y_a$. The next remark follows from Proposition 2.1.

Remark 2.4.

(a) $Y_a \leqslant X_a$.

(b) Prob[F is satisfiable as betweenness constraint] = Prob$[Y \geqslant 1] \leqslant$ Prob$[X \geqslant 1]$.

We have $n!$ orderings as candidate solutions to a given betweenness constraint F. It is natural to consider the random variable which gives the number of satisfying orderings. However, in part

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

due to the large number $n! \gg 2^n$ of solution candidates, this random variable does not seem easy to deal with. The random variable Y is useful because it counts orderings associated to the same boolean assignment only once. It thus has to deal with only 2^n candidates. The proof of the next proposition uses analytical techniques introduced in [1] and is given in Section 4.2.

Proposition 2.5. *For the random instance $F(V,m)$ with $m = rn$, $r \leqslant 1$ we have:*

(a) $\mathrm{E}[X] \geqslant (3/2)^{n(1-\varepsilon)}$ *for any constant $\varepsilon > 0$,*
(b) $\mathrm{E}[X^2] \leqslant C \cdot (\mathrm{E}[X])^2$ *for an appropriate constant C.*

As X is a random variable which is $\geqslant 0$ and has finite variance we can use the (Paley–Zygmund) inequality: for any $0 \leqslant \Theta \leqslant 1$,

$$\mathrm{Prob}[X \geqslant \Theta \mathrm{E}[X]] \geqslant (1-\Theta)^2 (\mathrm{E}[X])^2 / \mathrm{E}[X^2]. \tag{2.1}$$

With Proposition 2.5 as $\mathrm{E}[X] \geqslant 1$ we directly get (but do not really need) the following.

Corollary 2.6. *The event $X \geqslant 1$ holds w.u.p.p.*

Given a boolean assignment a, we consider the random instance $F_a(V,m)$, which is $F(V,m)$ conditioned on the event $X_a = 1$. Thus $F_a(V,m)$ consists of m clauses, each satisfying the boolean assignment a. We have $b = (3/4)(n)_3(1 + O(1/n))$ clauses satisfying a. The probability of a given instance of m such clauses is $1/(b)_m$ (if we consider sequences of distinct clauses). In the next section we prove the main lemma.

Lemma 2.7. *Let a be an arbitrary boolean assignment. We consider the random instance $F_a(V,m)$ with $m = rn$, $r \leqslant 0.921$. Then the event $Y_a = 1$ holds w.u.p.p.*

While Proposition 2.5 holds for $r > 0.921$, at present we cannot prove Lemma 2.7 for $r \geqslant 0.93$. Lemma 3.7(b) gives the reason.

At this point the reader may wonder why we cannot derive Theorem 1.2 directly with Corollary 2.6 and the preceding lemma. This, however, is not clear. The underlying probability spaces are not as closely related as they may seem. In particular, an instance from $F(V,m)$ with $X \geqslant 1$ may no longer be very random. It thus may not have much to do with a random instance $F_a(V,m)$ to which the lemma refers. Instead we only use the second moment of X.

Proof of Theorem 1.2. For a suitable constant $\varepsilon > 0$ and any boolean assignment a, we have with Lemma 2.7

$$\mathrm{E}[Y_a] = \mathrm{Prob}[X_a = 1] \cdot \mathrm{Prob}[Y_a = 1 \mid X_a = 1] \geqslant \mathrm{Prob}[X_a = 1] \cdot \varepsilon.$$

The second estimate above is Lemma 2.7. The first equation follows from the formula of total probability as $\mathrm{Prob}[Y_a = 1 \mid X_a = 0] = 0$ (Remark 2.4(a)). Then we get $\mathrm{E}Y \geqslant \varepsilon \cdot \mathrm{E}X \to \infty$ (with Proposition 2.5(a)).

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

Furthermore, we have

$$\begin{aligned}\mathrm{E}[Y^2] &\leqslant \sum_{(a,b)} \mathrm{Prob}[X_a = 1 \text{ and } X_b = 1] \\ &= \mathrm{E}[X^2] \leqslant C(\mathrm{E}X)^2 \leqslant (C/\varepsilon^2)(EY)^2,\end{aligned}$$

using Remark 2.4(a) for the first estimate and Proposition 2.5(b) to bound $\mathrm{E}[X^2]$. Now, Theorem 1.2 follows from equation (2.1). □

3. Proof of Lemma 2.7

We switch to the binomial space because the subsequent probability calculations appear to be slightly easier. The random instance $F(n, p, q)$ is obtained by throwing in each 3-clause randomly with probability p and each 2-clause with probability q, independently. We let

$$c = (1/3)r \quad \text{and} \quad d = (2/3)r, \quad \text{or} \quad r = 3c = (3/2)d$$

for the rest of this section. The main work is to prove the following.

Lemma 3.1. *For $r = 0.921$, $F(n, c/n^2, d/n)$ is satisfiable w.u.p.p.*

Proof of Lemma 2.7 from Lemma 3.1. For $F = F_a(V, m)$, we have $Y_a(F) = 1$ if and only if both formulas F_0 and F_1 as in Definition 2.2 are satisfiable (Proposition 2.3, $V_0 = a^{-1}(0)$ and $V_1 = a^{-1}(1)$).

In F the number of clauses from each of the 6 admissible possibilities among (V_i, V_k, V_j) with $i, j, k = 0, 1$ is concentrated at its expectation, that is, asymptotically $(1/6)m$ with high probability. With high probability, two clauses which overlap in two variables do not occur, as m is linear in n. Therefore F_0 and F_1 have $(1/6)m$ many 3-clauses and $(2/6)m$ many 2-clauses, each over $n/2$ variables. Moreover, F_0 and F_1 are stochastically independent (given their respective number of clauses, which is concentrated). For $m = rn$, F_0 and F_1 are two independent random formulas with asymptotically cn 3-clauses and dn 2-clauses over n variables (scaling to n variables instead of $n/2$).

Following the remark in the Introduction on the binomial space, the fact that $F(n, c/n^2, d/n)$ is satisfiable w.u.p.p. implies that F_0 and F_1 are both satisfiable w.u.p.p., by independence of F_0 and F_1. Lemma 2.7 follows. □

Definition 3.2. The directed (multi-)graph of F has as its vertices the variables of F. Its edges are given by the following: the clause $C = (x, y, z) \in F$ induces the edges (x, y), (y, z) and (x, z) each marked with C; the clause $x < y \in F$ induces the edge (x, y).

Clearly, if the graph of F is cycle-free, then F is satisfiable (by any topological ordering of the graph). To reduce a formula F we apply the following algorithm.

Algorithm 3.3. Input: A formula F.

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

V_1 := the set of those variables which occur exactly once in a 3-clause of F and nowhere else.
V_2 := those variables x which occur *only* at the position $x < -$. This means that all 2-clauses with x are of the form $x < y$ and we have no 3-clauses with x. Here we include the case when x does not occur at all.
V_3 := the variables x which occur *only* (and at least once) as $- < x$.

The result H of the algorithm is obtained by deleting all variables from $V_1 \cup V_2 \cup V_3$ and clauses containing them from F.

The algorithm is correct in the following sense.

Lemma 3.4. *If H is satisfiable then F is satisfiable.*

We iterate Algorithm 3.3 and therefore need the following.

Definition 3.5. The $w_{2,k}$ and $w_{3,k}$ for $k \geqslant 0$ are defined inductively by

$$\begin{aligned} w_{2,0} &= w_{3,0} = 0, \\ w_{2,k+1} &= \exp\bigl(-d(1-w_{2,k})\bigr) \cdot \exp\bigl(-3c(1-w_{3,k})^2\bigr), \\ w_{3,k+1} &= \exp\bigl(-d(1-w_{2,k})\bigr) \cdot w_{2,k+1}. \end{aligned}$$

Following [18] we consider the following type of random Poisson hypertree (which is approximately the random neighbourhood of a given variable in $F(n, c/n^2, d/n)$). The vertices of the tree are variables and edges are clauses. For each variable in the tree we have $<$-clauses resp. $>$-clauses, and 3-clauses as candidate child clauses. The number of child $<$- resp. $>$-clauses is distributed according to Po(d), the Poisson distribution with parameter d. The number of child 3-clauses follows Po($3c$). All distributions are independent. The random hypertree of depth l is obtained by generating l generations of children starting from a given root variable.

For the sake of analysability we apply Algorithm 3.3 to the random hypertree of depth l in the following way. In the first iteration we apply the algorithm to the variables in depth $l-1$, in the second iteration to the variables in depth $l-2$, and so on. Note that the next lemma does not refer to the root of the hypertree (which has depth 0).

Lemma 3.6. *Let $1 \leqslant k \leqslant l-1$. Conditional on the event that x is a variable in depth $l-k$ of the random hypertree of depth l, the probability that x gets deleted in the kth iteration is*
(a) *$w_{2,k}$ if x is connected to its father by a 2-clause,*
(b) *$w_{3,k}$ if x is connected to its father by a 3-clause.*

Proof. We use induction on k. For the induction step we proceed as follows. Condition on the event that x is a variable in depth $l-k$. After the $(k-1)$st iteration of the algorithm we have the following: the probability that x has no child $<$-clauses is

$$\begin{aligned} \sum_{m \geqslant 0} \mathrm{Prob}[\mathrm{Po}(d) = m] \cdot w_{2,k-1}^m &= \sum_{m \geqslant 0} \frac{(dw_{2,k-1})^m}{m!} \exp(-d) \\ &= \exp(-d(1-w_{2,k-1})). \end{aligned}$$

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

Observe that $\exp(-d(1-w_{2,k-1}))$ is the probability of 0 under the Poisson distribution with parameter $d(1-w_{2,k-1})$. The factor $1-w_{2,k-1}$ comes from the probability that a child variable connected x by a $<$-clause is not deleted in the previous round.

A similar calculation shows that the probability that x has no child 3-clauses is $\exp\bigl(-3c(1-w_{3,k-1})^2\bigr)$; multiplying we get $w_{2,k}$. We proceed for $w_{3,k}$ in the same way. □

Lemma 3.7.

(a) *For $r=0.921$ and k sufficiently large we have $w_{3,k}>0.95$.*
(b) *For $r=0.93$ we have $w_{2,k}<0.605$ and $w_{3,k}<0.475$ for all k.*

We postpone the proof of Lemma 3.7 to Section 4.1.

Lemma 3.8. *Let $r=0.921$ and let k be such that $w_{3,k}>0.95$. We denote $S:=3\ln n$. Let H be the formula obtained after iterating Algorithm 3.3 k times starting with $F=F(n,c/n^2,d/n)$.*

(a) *The expected number of cycles of length $\geqslant S+1$ in H is $o(1)$.*
(b) *The expected number of cycles of length $2\leqslant s\leqslant S$ in H is bounded above by*

$$\frac{(3c(1-w_{3,k})+d)^s}{s}+O(1/n^{0.7}).$$

With Lemma 3.8 we can finish the main argument.

Proof of Lemma 3.1. Let $F=F(n,c/n^2,d/n)$ and let H be the formula obtained after k iterations of Algorithm 3.3 from F. We use the k from Lemma 3.8. The probability of having a cycle of length $\geqslant S+1$ in H is $o(1)$ by part (a) of Lemma 3.8.

The probability of having a cycle of any length between 2 and S is bounded above by the expectation. By part (b) of Lemma 3.8 we have a bound of

$$\begin{aligned}\sum_{s=2}^{S}\frac{(3c(1-w_{3,k})+d)^s}{s}&+S\cdot O(1/n^{0.7})\\ &<(0.7)^2/2\cdot\sum_{s\geqslant 0}(0.7)^s+o(1)\\ &=(0.49\cdot 10)/(2\cdot 3)+o(1)<1,\end{aligned}$$

where we use $3c(1-w_{3,k})+d<0.7$ and the geometric series.

Thus H has no cycle w.u.p.p. By the remark after Definition 3.2, H is satisfiable. Using Lemma 3.4 inductively we get that F itself is satisfiable w.u.p.p. □

Proof of Lemma 3.8. First the values:

$$r=3c=0.921,\quad d=0.614,\quad 3c+d=1.535,$$
$$3c/d=1.5,\quad w:=1-w_{3,k}<0.05.$$

Occasionally we assume that constants such as c and d are slightly larger than the original ones: this allows us to treat expressions such as $(c+o(1))^n$ simply as being bounded above by c^n.

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

We consider only cycles $x_0 \to x_1 \to x_2 \cdots \to x_0$ such that the x_i are distinct (simple cycles). Moreover, we assume without loss of generality that different edges $x_i \to x_{i+1}$ are induced by different clauses. Otherwise we have a 3-clause (x, y, z) and the segment $\cdots x \to y \to z \cdots$ on the cycle. This segment can be replaced with $x \to z$.

A given edge is induced by one 2-clause and $3(n-2)$ 3-clauses. Therefore the expected number of ways in which an edge occurs in the graph of F is $3(n-2)c/n^2 + d/n \leqslant 1.535/n$. The probability that a given edge is present is

$$1 - (1 - c/n^2)^{3(n-2)}(1 - d/n) = (3c + d)/n + O(1/n^2).$$

Observe that $(3c + d)/n = 1.535/n$. Disregarding dependencies between edges, F induces a directed random graph well above the strongly connected component threshold, which occurs at edge probability $1/n$; see [21], [16]. As such, F itself should contain cycles. Consider an edge induced by a 3-clause, such as $x \to z$ induced by (x, y, z), and condition on the event that it is present in F. It turns out that the probability that y survives k rounds of Algorithm 3.3 is bounded above by $w < 0.05$, and $3c \cdot 0.05 + d < 1$. This allows us to bound the expectation of the number of cycles present after k rounds of Algorithm 3.3 strictly below 1 (in principle analogously to the classical case [21]). Some stochastic dependencies between edges do, however, require technical care.

Proof of Lemma 3.8(a). Each cycle of length $\geqslant S + 1$ contains a simple path of length S. We show that the expected number of such paths in H is $o(1)$. Let $x_0 \to x_1 \to x_2 \cdots \to x_S$ be a candidate path, with t edges in t fixed slots induced by 3-clauses. The additional variables necessary to fill the 3-clauses are denoted by $(y_1, \ldots, y_t)$. We assume that y_i includes the information about its slot (one out of three) in its respective 3-clause. Altogether we have $< n^{S+1}\binom{S}{t}(3n)^t$ such candidate paths. The probability that a candidate path occurs in F is $(c/n^2)^t(d/n)^{S-t}$ as each edge is induced by a different clause. To bound the expectation of the number of these paths by $o(1)$ we proceed with a case distinction.

Case 1. Let C be a constant; we consider paths with $t \leqslant C$. This case is necessary because below we consider expressions such as $t - D$. In order to avoid $t - D$ being negative we need to assume that $t > C$. For the expected number of such paths in F we get an upper bound of

$$n^{S+1} \sum_{t \leqslant C} S^t (3n)^t (c/n^2)^t (d/n)^{S-t} < O(\mathrm{poly}(\log n)) \cdot n(3c/d)^C d^S = o(1).$$

Note that $3c/d = 1.5$ (> 1), $3 \ln d \approx -1.429$, and $S = 3 \ln n$.

Case 2. We consider paths with $t > C$ such that the number of *new* variables among $(y_1, \ldots, y_t)$ is $\leqslant t - 3$. A y_i is *not* new if it occurs among the x_j or there is a $j < i$ with $y_j = y_i$. For the expected number of paths of this type in F we get an upper bound of

$$n^{S+1} \sum_{t > C} \binom{S}{t} S^3 (3n)^{t-3} (3 \cdot \mathrm{poly}(\log n))^3 (c/n^2)^t (d/n)^{S-t}$$
$$< n \cdot \mathrm{poly}(\log n) \cdot (3c + d)^S \cdot (1/n)^3 = o(1),$$

as $3 \ln(3c + d) \approx 1.28$ and $S = 3 \ln n$.

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

For candidate paths to which the preceding two cases do not apply we consider the neighbourhood of those y_i which occur only once in the candidate. There are at least $t-4$ of them, as there are at least $t-2$ new variables among $(y_1,\ldots,y_t)$.

The following principle is our guide. Let $y=y_j$ be a variable occurring only once and let, for example, (x_i,y,x_{i+1}) be the 3-clause with y. If k iterations of Algorithm 3.3 have the effect that y is deleted, the edge $x_i\to x_{i+1}$ disappears too. A lower bound on the probability of this event is $w_{3,k}$. This can be derived by applying Lemma 3.6 to that part of the k-neighbourhood of y which is still random. This means the standard k-neighbourhood of y is modified, in that the clause (x_i,y,x_{i+1}) is no longer included. Thus, the 1-neighbourhood of y in our sense comprises the clauses containing y, except for (x_i,y,x_{i+1}). Note that the k-neighbourhood is with high probability a hypertree to which Lemma 3.6 is applicable. When we have l distinct variables such as y in the candidate path which can be treated independently, the survival probability of all of these is at most $(1-w_{3,k})^l=w^l$. However, full independence cannot be assumed, and overlap between k-neighbourhoods must be considered.

Let P be a given candidate path; let $y_1,\ldots,y_{t-4}$ be those variables among the $y_1,\ldots,y_t$ which occur only once. (Thus two variables among the $y_1,\ldots,y_t$ may occur twice.) The k-neighbourhood of $y_1,\ldots,y_{t-4}$ altogether is the union of the k-neighbourhoods of $y_1,\ldots,y_{t-4}$. We call the k-neighbourhood of $y_1,\ldots,y_{t-4}$ *exceptional* if the number of y_i whose k-neighbourhoods are hypertrees disjoint from the rest of the k-neighbourhood is $\leqslant t-4-6=t-10$. (This happens, for example, when the neighbourhoods of three of the y_i have an overlap with those of three more of the y_i.)

Let P_{EX} be the event that the path P occurs in F *and* that the k-neighbourhood of $y_1\cdots y_{t-4}$ is exceptional. $P_{\neg EX}$ is the event that the k-neighbourhood is not exceptional.

Claim. *The probability of P_{EX} in F is bounded above by*

$$(c/n^2)^t(n/d)^{S-t}\cdot O(\mathrm{poly}(\log n)/n^3).$$

Case 3. The expectation of the number of paths P for which the event P_{EX} in F holds is bounded above by

$$\begin{aligned} n^{S+1}\sum_{t>C}\binom{S}{t}(3n)^t(c/n^2)^t(d/n)^{S-t}\cdot O(\mathrm{poly}(\log n)/n^3)\\ < n(3c+d)^S\cdot O(\mathrm{poly}(\log n)/n^3)=o(1),\end{aligned}$$

as in Case 2.

Case 4. This is the most interesting case, as we use the formula H obtained after k rounds of Algorithm 3.3 from F. Let P be a candidate with t edges induced by 3-clauses and at least $t-4$ new variables. The probability in F that P is still present in H and $P_{\neg EX}$ is

$$\begin{aligned}&\mathrm{Prob}[P \text{ in } H\wedge P_{\neg EX}]\\ &\quad=\mathrm{Prob}[P \text{ in } H\mid P_{\neg EX}]\cdot\mathrm{Prob}[P_{\neg EX}]\\ &\quad< w^{t-9}(c/n^2)^t(n/d)^{S-t}.\end{aligned}\tag{3.1}$$

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

Estimate (3.1) holds because of the following observations. $\mathrm{Prob}[P_{\neg EX}] < (c/n^2)^t (n/d)^{S-t}$ because $P_{\neg EX}$ requires the path P to be present. Conditional on $P_{\neg EX}$ we have $t-9$ k-neighbourhoods which are hypertrees disjoint from the rest of the k-neighbourhood. We assume without loss of generality that the k-neighbourhood of $y_1, \ldots, y_{t-9}$ consists of such hypertrees. Thus the k-neighbourhood of $y_1, \ldots, y_{t-9}$ can be viewed as $t-9$ independent Poisson hypertrees as considered in Lemma 3.6. (We skip the detailed argument on this point and refer to [18].) By Lemma 3.6(b) the probability that y_i survives in H is $< w$, independently. Note that y_i belongs to a 3-clause to begin with.

For the expected number of paths P in H with $P_{\neg EX}$ in F, we therefore get a bound of

$$\begin{aligned} & n^{S+1} \sum_{t>C} \binom{S}{t} (3n)^t w^{t-9} (c/n^2)^t (d/n)^{S-t} \\ & < n \sum_{t>C} \binom{S}{t-9} (3cw)^{t-9} d^{S-(t-9)} (S \cdot 3c/d)^9 \\ & < n \cdot \mathrm{poly}(\log n) \cdot (3cw + d)^S = o(1), \end{aligned}$$

as $3\ln(3cw + d) \approx -1.234$ and $S = 3 \ln n$.

Summing the preceding four $o(1)$-terms, we get that the expected number of paths of length S in H is $o(1)$.

Proof of the claim. The claim is easily proved by generating the k-neighbourhood step by step. Recall that $y_1, \ldots, y_{t-4}$ are the additional variables of the 3-clauses of P which occur only once. We first generate the neighbours of $y = y_1$. The 3-clauses with y follow the binomial distribution $\mathrm{Bin}(3(n-1)(n-2)-1, c/n^2)$; for the 2-clauses we get two $\mathrm{Bin}(n-1, d/n)$ distributions. These distributions are independent as they refer to disjoint sets of clauses. The probability that we get a clause which produces an overlap (that is, it contains another variable of the path P or collides with another neighbour of y) is $O(\mathrm{poly}(\log n)/n)$. Observe that the probability of getting at least $a \log n$ new 3-clauses with y (a a small constant) is very small. It is bounded above by

$$\sum_{h \geqslant a \log n} \binom{3n^2}{h} \left(\frac{c}{n^2}\right)^h \leqslant \frac{(3c)^{a \log n}}{(a \log n)!} \sum_{h \geqslant 0} \frac{(3c)^h}{h!} < 1/n^{\Omega(\log \log n)}.$$

The same argument applies when we consider the 2-clause distribution. When we have $> a \log n$ new clauses we stop the process. We condition the following consideration on the event that this has not happened.

Once we have an overlap we proceed with the next new y_i which still occurs only once (there are at least $t-4-2$ of them). Otherwise we generate the next generation of neighbours of y. Due to our conditioning, the distributions stay essentially unchanged and we can continue analogously. When we have generated k generations of neighbours of y without overlap we go to y_2. We proceed in this way and stop only when we have three overlaps or a variable with degree $> a \log n$.

The probability of hitting upon three overlaps is $O(\mathrm{poly}(\log n)/n^3)$ as long as the number of neighbours of each variable is $< a \log n$. This is ensured by our conditioning. Moreover, the probability of $> a \log n$ neighbours at some point is much smaller than $O(\mathrm{poly}(\log n)/n^3)$, and

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

can thus be added without weakening the upper bound. This proves the claim because when the number of independent hypertrees rooted at the y_i is reduced from $t-4$ to $t-10$, we must have at least three overlaps in the neighbourhood.

Proof of Lemma 3.8(b). We refer to (a) for unexplained notions and employ the same principle. Let $x_0 \to x_1 \to \cdots \to x_{s-1} \to x_0$ be a candidate cycle of length s with $s \geqslant t \geqslant 0$ edges induced by 3-clauses. Let $(y_1, \ldots, y_t)$ be the additional variables of the 3-clauses (each variable comes with its slot in the 3-clause, one out of three possibilities). The number of such candidates is bounded above by $\frac{n^s}{s}\binom{s}{t}(3n)^t$ (the $1/s$ occurring because of cyclic permutations). The probability that a candidate occurs in F is $(c/n^2)^t(d/n)^{S-t}$. The expectation of the number of cycles of length s in F is thus bounded above by $(3c+d)^s/s$. Recall that $3c+d = 1.535$ and the expectation is quite large. In H, however, we get the claimed $(3cw+d)^s/s$ instead (up to an additive $o(1)$-term). Again the overlap structure in the k-neighbourhood needs to be considered. This is analogous to (a).

Case 1. We consider cycles of length s such that all y_i are new. Conditioning on the event that the k-neighbourhood in F consists of disjoint hypertrees, each y_i survives k rounds of Algorithm 3.3 with probability w independently. For the expected number of cycles of length s whose k-neighbourhood in F consists of disjoint hypertrees and which survive in H, we get a bound of $(3cw+d)^s/s$ (based on the binomial theorem).

The event that we have exactly one overlap in the k-neighbourhood has probability $O(\log n/n)$. Conditioning on one fixed overlap, at least $t-2$ of the y_i survive with probability $\leqslant w$ independently. For the expected number of cycles in H with such a k-neighbourhood in F, we get a bound of

$$n^s \sum_{t \geqslant 0} \binom{s}{t} (3n)^t (c/n^2)^t (d/n)^{s-t} w^{t-2} \cdot O(\mathrm{poly}(\log n)/n)$$
$$< (3cw+d)^s/s \cdot O(\mathrm{poly}(\log n)/n) < O(\mathrm{poly}(\log n)/n),$$

recalling that $3cw + d < 1$.

For two or more overlaps in the k-neighbourhood we need only consider F, and we bound the expectation by

$$(3c+d)^s/s \cdot O(\mathrm{poly}(\log n)/n^2) = O(1/n^{0.7})$$

as $s \leqslant S = 3\ln n$ and $3\ln(3c+d) \approx 1.28$.

Case 2. If one y_i is not new, the number of candidates is bounded above by

$$n^s(3n)^{t-1} \cdot O(\mathrm{poly}(\log n)) = n^s(3n)^t \cdot O(\mathrm{poly}(\log n)/n).$$

If we have no additional overlap in the k-neighbourhood then we have at least $t-2$ independent hypertrees rooted at the y_i. We get the same result as in Case 1 with one overlap.

If we have one overlap in the k-neighbourhood, we have another factor of $O(\mathrm{poly}(\log n)/n)$. We get the same result as in Case 1 with at least two overlaps.

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

Case 3. If at least two y_i are not new, we have

$$n^s(3n)^{t-2} \cdot O(\text{poly}(\log n)) = n^s(3n)^t \cdot O(\text{poly}(\log n)/n^2)$$

candidates and we get the same result as Case 1 with at least two overlaps. □

4. The remaining proofs

4.1. Proof of Lemma 3.7

Recall the following lemma.

Lemma 3.7.

(a) *For* $r = 0.921$ *and* k *sufficiently large we have* $w_{3,k} > 0.95$.
(b) *For* $r = 0.93$ *we have* $w_{2,k} < 0.605$ *and* $w_{3,k} < 0.475$ *for all* k.

Proof. We define

$$w_0 = 0, \quad w_1 = \exp(-d) \cdot \exp(-3c) \quad \text{and for } k \geqslant 1$$

$$w_{k+1} = \exp\big(-d(1-w_k)\big) \cdot \exp\left(-3c\left(1 - \underbrace{\exp\big(-d(1-w_{k-1})\big)w_k}_{\text{compare } w_{3,k} \text{ (Definition 3.5)}}\right)^2\right).$$

By induction $w_k = w_{2,k}$. The definition of w_{k+1} is monotonically increasing in both w_k and w_{k-1}. Moreover, $0 \leqslant w_k \leqslant 1$, $w_{k+1}/w_k > 1$ and the w_k are strictly monotonically increasing.

Using a Maple function we implemented the preceding definition for w_k. For numerical security reasons we subtracted $1/10^{10}$ from the right-hand side. Running this program for $r = 0.921$ and $k = 100$ yields $w_{100} = w_{2,100} \geqslant 0.98$. This ensures by Definition 3.5 that $w_{3,100} \geqslant 0.95$ as required. This finishes the proof of (a). When we substitute $r = 0.93$ and $w_k = w_{k-1} = 0.605$ in the definition of w_{k+1} above we get a value of $0.6049459 < 0.605$. This implies by induction and the monotonicity of the definition that all $w_k < 0.605$. This shows (b). □

4.2. Proof of Proposition 2.5

Recall the following proposition.

Proposition 2.5. *For the random instance* $F(V, m)$ *with* $m = rn$, $r \leqslant 1$ *we have:*

(a) $\mathrm{E}[X] \geqslant (3/2)^{n(1-\varepsilon)}$ *for any constant* $\varepsilon > 0$,
(b) $\mathrm{E}[X^2] \leqslant C \cdot (\mathrm{E}[X])^2$ *for an appropriate constant* C.

Proof. **(a)** Stirling's formula in the form

$$n! = (n/e)^n \sqrt{2\pi n}(1 + o(1))$$

yields

$$\binom{n}{n/2} = \sqrt{2/(\pi n)}\, 2^n (1 + o(1)).$$

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

Then

$$\mathrm{E}X = \sqrt{2/(\pi n)}\left(2(3/4)^r\right)^n(1+o(1)) \geqslant (3/2)^{n(1-\varepsilon)}$$

since $(3/4)^r \geqslant 3/4$ as $r \leqslant 1$.

(b) Given two assignments a, b with overlap $2l = \alpha n$, that is, we have $2l$ variables which have the same truth value under both a and b, the probability that a random clause is satisfied by both a and b is $(3/4)(1-\alpha(1-\alpha))$. This can be seen by elementary consideration and implies that

$$\begin{aligned}\mathrm{E}[X^2] &= \sum_{(a,b)} \mathrm{Prob}[X_a = 1 \text{ and } X_b = 1] \\ &= \binom{n}{n/2}\sum_{l=0}^{n/2}\binom{n/2}{l}^2\left(\frac{3}{4}\left(1-\frac{l}{n/2}\left(1-\frac{l}{n/2}\right)\right)\right)^m. \end{aligned} \tag{4.1}$$

With $\phi(\alpha) := \left(3/4\cdot\left(1-\alpha(1-\alpha)\right)\right)^r$ and using $m = rn$, the sum in (4.1) becomes

$$S_n := \sum_{l=0}^{n/2}\binom{n/2}{l}^2\left(\phi\left(\frac{l}{n/2}\right)\right)^n. \tag{4.2}$$

We apply the next lemma with $q := 2$, $t := n/2$, $z := l$ to S_n.

Lemma (Laplace lemma [1]). *Let $\phi(\alpha)$ be a positive, twice-differentiable function on $[0,1]$ and let $q \geqslant 1$ be a fixed integer. Let $t = n/q$ and let*

$$S_n := \sum_{z=0}^{t}\binom{t}{z}^q\phi(z/t)^n \quad \textit{and} \quad g(\alpha) := \frac{\phi(\alpha)}{\alpha^\alpha(1-\alpha)^{1-\alpha}},$$

where $g(\alpha)$ is defined on $[0,1]$ and $0^0 := 1$.

If there exists $\alpha_{\max} \in (0,1)$ such that $g(\alpha_{\max}) =: g_{\max} > g(\alpha)$ for all $\alpha \neq \alpha_{\max}$ and $g''(\alpha_{\max}) < 0$, then there is a constant $C = C(q, g_{\max}, g''(\alpha_{\max}), \alpha_{\max}) > 0$ such that, for all sufficiently large n, we have $S_n < Cn^{-(q-1)/2}(g_{\max})^n$.

We get from (4.2) and the Laplace lemma that $S_n \leqslant C(1/\sqrt{n})(g_{\max})^n$. From Stirling's formula and (4.1) we get (up to a multiplicative term of $1+o(1)$)

$$\mathrm{E}[X^2] \leqslant 2^n\sqrt{\frac{2}{\pi n}}\,C\frac{1}{\sqrt{n}}(g_{\max})^n = D\frac{1}{n}(2g_{\max})^n.$$

Below we show that $g_{\max} = 2(3/4)^{2r}$ (see equation (4.3)) and the claim holds because $\mathrm{E}X = \sqrt{2/(\pi n)}\,2^n(3/4)^{rn}$ (*cf.* the proof of part (a)).

We check that the Laplace lemma is applicable. For the function $\phi(\alpha)$ (the definition before equation (4.2)) we have, for $\alpha \in [0,1]$, that $\phi(\alpha) \geqslant 0$ and $\phi(\alpha)$ is twice differentiable and symmetric around $\alpha = 1/2$. For $\alpha = 1/2$ we have its minimum on $[0,1]$, which is $\phi(1/2) = (3/4)^{2r}$. (The proof is elementary calculus.)

We come to

$$g(\alpha) = \frac{\phi(\alpha)}{\alpha^\alpha(1-\alpha)^{1-\alpha}} = \frac{(3/4\cdot(1-\alpha(1-\alpha)))^r}{\alpha^\alpha(1-\alpha)^{1-\alpha}}.$$

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

It turns out that $g(\alpha)$ is maximized at

$$g_{\max} = g(1/2) = 2(3/4)^{2r}. \tag{4.3}$$

First,

$$g'(\alpha) = \underbrace{\left(\frac{3}{4}\right)^r \underbrace{(1-\alpha+\alpha^2)^{r-1}}_{=((\alpha-1)^2+\alpha)^{r-1}>0} \underbrace{\frac{1}{\alpha^\alpha(1-\alpha)^{1-\alpha}}}_{>0} \cdot h(\alpha)}_{>0},$$

where

$$h(\alpha) := \left[r(2\alpha-1) - (1-\alpha+\alpha^2)\ln\left(\frac{\alpha}{1-\alpha}\right)\right]$$

and $g'(1/2) = 0$, as $h(1/2) = 0$. Moreover,

$$g''(1/2) = 1/3 \cdot 9^r\left(2^{-4r+4}r - 24 \cdot 16^{-r}\right),$$

which is easily seen to be < 0 even for $r < 3/2$.

We consider $\alpha \in (0, 1/2), r \in (0, 3/2)$. We have

$$h'(\alpha) = \underbrace{2r - \underbrace{(2\alpha-1)}_{<0} \cdot \underbrace{\ln\left(\frac{\alpha}{1-\alpha}\right)}_{<0}}_{<2r<3} - \underbrace{(1-\alpha+\alpha^2)\cdot\left(\frac{1}{\alpha}+\frac{1}{1-\alpha}\right)}_{=:k(\alpha)}.$$

We rewrite $k(\alpha) = 1/\alpha + 1/(1-\alpha) - 1$ and

$$\begin{aligned}
k'(\alpha) &= -\frac{1}{\alpha^2} + \frac{1}{(1-\alpha)^2} = \frac{2\alpha-1}{\alpha^2(1-\alpha)^2} < 0 \\
&\Rightarrow k(\alpha) \text{ strictly monotonically decreasing in } (0, 1/2), \\
k(1/2) = 3 &\Rightarrow k(\alpha) > 3, \forall\, \alpha \in (0, 1/2) \\
&\Rightarrow h'(\alpha) < 0, \forall\, \alpha \in (0, 1/2), 0 < r < 3/2 \\
&\Rightarrow h(\alpha) \text{ is strictly monotonically decreasing in } (0, 1/2), \\
h(1/2) = 0 &\Rightarrow h(\alpha) > 0, \forall\, \alpha \in (0, 1/2) \\
&\Rightarrow g'(\alpha) > 0, \forall\, \alpha \in (0, 1/2) \\
&\Rightarrow g(\alpha) \text{ strictly monotonically increasing in } (0, 1/2).
\end{aligned}$$

□

5. Conclusion

Concerning our constant r, the contribution is that the bound of $r < 0.82$ from [12] can be improved to $r = 0.921$ by more advanced techniques. Our proof cannot easily be extended beyond $r = 0.93$. In this case we have $w_{3,k} < w_{2,k} < 0.605$ by Lemma 3.7(b). Our proof for $r = 0.921$, however, relies clearly on the fact that $w_{3,k} > 0.9$.

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press

Acknowledgement

Special thanks to Anja Lau for help with the calculus and the referee for sharing some simplifying insight.

References

[1] Achlioptas, D. and Moore, C. (2006) Random k-SAT: Two moments suffice to cross a sharp threshold. *SIAM J. Comput.* **36** 740–762.
[2] Ailon, N. and Alon, N. (2007) Hardness of fully dense problems. *Inform. Comput.* **205** 1117–1129.
[3] Beame, P., Karp, R., Pitassi, T. and Saks, M. (2002) The efficiency of resolution and Davis–Putnam procedures. *SIAM J. Comput.* **31** 1048–1075.
[4] Bodirsky, M. and Kára, J. (2008) The complexity of temporal constraint satisfaction problems. In *Proc. 40th ACM Symposium on Theory of Computing* (SToC), pp. 29–38.
[5] Bollobás, B. (1985) *Random Graphs*, Academic Press.
[6] Broder, A., Frieze, A. and Upfal, A. (1993) On the satisfiability and maximum satisfiability of random 3-CNF formulas. In *Proc. 4th ACM–SIAM Symposium on Discrete Algorithms* (SoDA), pp. 322–330.
[7] Chor, B. and Sudan, M. (1998) A geometric approach to betweenness. *SIAM J. Discrete Math.* **11** 511–523.
[8] Christof, T., Jünger, M., Kececioglu, J., Mutzel P. and Reinelt, G. (1997) A branch-and-cut approach to physical mapping with end-probes. In *Proc. 1st Annual International Conference on Computational Molecular Biology* (RECOMB), pp. 84–92.
[9] Dubois, O., Boufkhad, Y. and Mandler, J. (2000) Typical random 3-SAT formulae and the satisfiability threshold. In *Proc. 11th ACM–SIAM Symposium on Discrete Algorithms* (SoDA), pp. 126–127.
[10] Erdős, P. and Rényi, A. (1960) On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci.* **5** 17–61.
[11] Friedgut, E. (2005) Hunting for sharp thresholds. *Random Struct. Alg.* **26** 37–51.
[12] Goerdt, A. (2009) On random ordering constraints. In *Proc. Computer Science in Russia* (CSR), Vol. 5675 of *Lecture Notes in Computer Science*, Springer, pp. 105–116.
[13] Guttmann, W. and Maucher, M. (2006) Variations on an ordering theme with constraints. In *Proc. 4th IFIP International Conference on Theoretical Computer Science* (TCS), Vol. 209 of *International Federation for Information Processing*, Springer, pp. 77–90.
[14] Hatami, H. and Molloy, M. (2008) Sharp thresholds for constraint satisfaction problems and homomorphisms. *Random Struct. Alg.* **33** 310–332.
[15] Isli, A. and Cohn, A. G. (1998) An algebra for cyclic ordering of 2D orientation. In *Proc. 15th American Conference on Artificial Intelligence* (AAAI), AAAI/MIT Press, pp. 643–649.
[16] Karp, R. M. (1990) The transitive closure of a random digraph. *Random Struct. Alg.* **1** 73–93.
[17] Molloy, M. (2002) Models and thresholds for random constraint satisfaction problems. In *Proc. 34th ACM Symposium on Theory of Computing* (SToC), pp. 209–217.
[18] Molloy, M. (2005) Cores in random hypergraphs and Boolean formulas. *Random Struct. Alg.* **27** 124–135.
[19] Molloy, M. (2008) When does the giant component bring unsatisfiability? *Combinatorica* **28** 693–734.
[20] Opatrny, J. (1979) Total ordering problem. *SIAM J. Comput.* **8** 111–114.
[21] Palasti, I. (1971) On the threshold distribution function of cycles in a directed random graph. *Studia Scientiarum Math. Hungar.* **6** 67–73.

https://doi.org/10.1017/S0963548310000313 Published online by Cambridge University Press