A Dichotomy Theorem for the Resolution Complexity of Random Constraint Satisfaction Problems

Siu On Chan and Michael Molloy
Department of Computer Science
University of Toronto
{siuon,molloy}@cs.toronto.edu

Abstract

We consider random instances of constraint satisfaction problems where each variable has domain size $O(1)$, each constraint is on $O(1)$ variables and the constraints are chosen from a specified distribution. The number of constraints is cn where c is a constant. We prove that for every possible distribution, either the resolution complexity is almost surely polylogarithmic for sufficiently large c, or it is almost surely exponential for every $c > 0$. We characterize the distributions of each type. To do so, we introduce a closure operation on a set of constraints which yields the set of all constraints that, in some sense, appear implicitly in the random CSP.

1 Introduction

Constraint satisfaction problems (CSP's) form an active area of research in many areas of computer science. They generalize SAT by allowing variables to take values from a domain more general than $\{\text{true}, \text{false}\}$, and having more general restrictions on values jointly taken by variables in each clause. The widespread interest in random k-SAT has spread to its generalisations, such as random instances of 1-in-k-SAT [3], NAE-k-SAT [3, 5], k-XOR-SAT [17, 28] and $(2 + p)$-SAT [4]. All of these can be expressed as CSP's. As a result, the interest has spread to random instances of CSP's, rigourously in e.g. [32, 18, 31, 30] and experimentally even earlier (see [23] for a survey).

Unsatisfiability of k-SAT and CSP instances can be demonstrated by resolution proof systems, and many natural algorithms for k-SAT and CSP can be simulated as resolution proof procedures. In fact, virtually every complete SAT-solver or CSP-solver used in practice is resolution based. The running time of any such algorithm is lower-bounded by the resolution complexity of the input. In a seminal paper [13], Chvátal and Szemerédi consider the resolution complexity of random k-SAT formulas, $k \geq 3$; i.e. the asymptotic order of the length of a shortest resolution refutation. As the clause-variable ratio c grows, the resolution complexity decreases monotonically, but is still almost surely (a.s.)[1] exponential for any constant c. This explained the empirical observation that SAT-solvers take a very long time on these formulas when the number of clauses is high enough that the formula is a.s. unsatisfiable [29]. Their result has been generalized and extended in many directions, to super-constant clause-variable ratio [22, 10, 9], and to general classes of CSP's [31, 34]. There are also a.s. exponential lower bounds on the resolution complexity of specific graph problems, such as k-colourability and k-independent sets [8, 7].

In contrast, many random models a.s. have at most polynomial resolution complexity when the clause-variable ratio is a sufficiently large constant; we call this property POLY. It is natural to ask which models of random CSP's have Property POLY, and for which models the resolution complexity a.s. remains exponentially high for every constant clause-variable ratio. This question has been resolved for several specific models and family of models in the past (see below). Our main contribution is to resolve this question for *every* model from a very broad family which contains, in some sense, all random models with constant clause and domain size.

Practitioners have long been using random CSP's to gain insights into difficult problems. This paper may shed some insight into what can cause CSP's to have high resolution complexity, even for a very high linear number of constraints. We show that for a high linear *random* number of constraints, polynomial resolution complexity can arise only when there are many long path-like substructures, called *petals* that constrain the joint assignments to pairs of distant variables.

Roughly speaking, the models for random CSP's we con-

[1] We say that an event A occurs *almost surely* if $\mathbb{P}(A)$ tends to 1 as n tends to infinity.

sider are as follows (formal definitions will appear in Section 3): One begins by randomly selecting k-tuples of n variables on which to place a constraint. Then, for each chosen k-tuple, one chooses a random constraint. The distribution $\mathcal{P}$ from which this random constraint is chosen is what specifies the model. For example, with one distribution, the model is random k-SAT, with another it is random k-XOR-SAT and with yet another it is d-colourability of random graphs. We denote a random instance by $\mathsf{CSP}_{n,M}(\mathcal{P})$, where M is the number of k-tuples selected.

The main result of this paper is a characterisation of exactly which models have property POLY. Denote by $\operatorname{supp}\mathcal{P}$ the *support* of $\mathcal{P}$, i.e. $\operatorname{supp}\mathcal{P} = \{C \mid \mathcal{P}(C) > 0\}$. Informally, $\operatorname{supp}\mathcal{P}$ is the set of "types" of constraints that appear in the random CSP. It turns out that whether POLY holds for $\mathsf{CSP}_{n,M}(\mathcal{P})$ depends only on the set $\operatorname{supp}\mathcal{P}$ and not on the actual distribution over $\operatorname{supp}\mathcal{P}$.

For a particular $\mathcal{P}$, let $\mathcal{C}$ denote $\operatorname{supp}\mathcal{P}$. It turns out that it is a bit deceptive to focus only on $\mathcal{C}$. The reason is that long paths of constraints can induce a constraint, C', on their endpoints. If those endpoints lie in a constraint $C \in \mathcal{C}$ then, in effect, they are constrained by the more restrictive constraint: $C' \cup C$. So we determine all constraints C that are *likely* to be induced by long paths, and for each, we form $\mathcal{C}'$ by adding $C' \cup C$ for each $C \in \mathcal{C}$. Naturally, we need to iterate until we reach what we call the *closure* of $\mathcal{C}$, $\operatorname{cl}(\mathcal{C})$. We say that $\operatorname{cl}(\mathcal{C})$ is *complete* if it contains the unsatisfiable constraint; i.e. the constraint that forbids every k-tuple of values. These definitions appear more formally in Section 3.3.

It is possible that $\operatorname{cl}(\mathcal{C})$ will be applicable to other problems regarding random CSP's, since it contains the constraints which appear *implicitly* in the CSP as opposed to the constraints of $\mathcal{C}$ that appear *explicitly*. For the purposes of this paper, focussing on $\operatorname{cl}(\mathcal{C})$ rather than $\mathcal{C}$ yields a simple characterization of those $\mathcal{P}$ that have the property POLY:

Theorem 1.1. *If* $\operatorname{cl}(\operatorname{supp}\mathcal{P})$ *is complete and* c *is sufficiently large, then a.s.* $\mathsf{CSP}_{n,M=cn}(\mathcal{P})$ *has at most polylogarithmic resolution complexity.*

Theorem 1.2. *If* $\operatorname{cl}(\operatorname{supp}\mathcal{P})$ *is incomplete and* $c > 0$*, then with uniformly positive probability (w.u.p.p.)*[2] $\mathsf{CSP}_{n,M=cn}(\mathcal{P})$ *has at least exponential resolution complexity.*

The above two theorems together form a dichotomy theorem. Given $\mathcal{P}$ it is easy to determine $\operatorname{cl}(\operatorname{supp}\mathcal{P})$ and hence to decide POLY in $O(1)$ time (see Remark 3.10).

Note that Theorem 1.1 is stronger than what we aimed for: It implies not only a.s. polynomial, but a.s. *polylogarithmic*, resolution complexity. So we have the following interesting corollary:

[2]We say that an event occurs *with uniformly positive probability* if $\liminf \mathbb{P}(A) > 0$.

Corollary 1.3. *If for* c *sufficiently large,* $\mathsf{CSP}_{n,M=cn}(\mathcal{P})$ *has subexponential resolution complexity then for* c *sufficiently large,* $\mathsf{CSP}_{n,M=cn}(\mathcal{P})$ *has polylogarithmic resolution complexity.*

For the case where POLY does not hold, i.e. the case covered by Theorem 1.2, we determine whether $\mathsf{CSP}_{n,M}(\mathcal{P})$ in fact a.s. has exponential resolution complexity. A *cyclic* CSP is a CSP whose underlying hypergraph forms a cycle (the formal definition appears in Subsection 3.3). See Definition 3.6 for the meaning of "null-constraining".

Theorem 1.4. *Suppose* $\operatorname{cl}(\operatorname{supp}\mathcal{P})$ *is incomplete.*

(a) If for some null-constraining subdomain $\mathcal{D}'$*, every cyclic CSP formed from* $\operatorname{supp}\mathcal{P}$ *is satisfiable using only values from* $\mathcal{D}'$ *then* $\mathsf{CSP}_{n,M}(\mathcal{P})$ *a.s has at least exponential resolution complexity for all* c*;*

(b) else w.u.p.p. $\mathsf{CSP}_{n,M=cn}(\mathcal{P})$ *has at most polylogarithmic resolution complexity.*

So in case (b) there is some $\epsilon > 0$ such that with probability at least ϵ, $\mathsf{CSP}_{n,M=cn}(\mathcal{P})$ has at most polylogarithmic resolution complexity and with probability at least ϵ, $\mathsf{CSP}_{n,M=cn}(\mathcal{P})$ has exponential resolution complexity. The proof implies that small resolution complexity must be caused by problematic cycles of length $O(1)$.

In the course of proving Theorem 1.1, we study a certain convergence property of random walks on general directed graphs (Theorem 5.1). This result may be of independent interest.

2 Related Work

The first result along these lines was by Chvátal and Szemerédi [13] who proved that random 3-SAT a.s. has exponentially high resolution complexity for every constant c; i.e. it does not have property POLY. This result was extended to the case where c grows with n in [9, 10] and the proof was simplified greatly by Ben-Sasson and Wigderson in [11] where they introduced their Width Lemma (which we use here). Achlioptas et al [2] began with the easy observation that random $(2+p)$-SAT, a mixture of random 2-SAT and random 3-SAT has polynomial resolution complexity if the number of clauses is so high that the 2-clauses alone are unsatisfiable; thus random $(2+p)$-SAT has POLY. They then proved that for any smaller clause-density, the resolution complexity is exponentially high, thus establishing a sharp threshold for exponential resolution complexity in this model. They also showed how this can explain the empirical observation that resolution-based SAT-solvers have a difficult time with random 3-SAT even below the generally

conjectured value of the satisfiability threshold (see also [1] for random k-SAT with $k > 3$).

Mitchell [31, 30] extended the (by then standard) techniques for proving such theorems about random k-SAT to the more general setting of random CSP's. He used these techniques to study the (d, k, t)-model – where the domain size is d and the constraints are uniformly random amongst those with k variables and t restrictions. He proved that for a wide range of triples (d, k, t), the model does not have property POLY. Molloy and Salavatipour [34] determined precisely which triples (d, k, t) have property POLY; moreover, for those that do have POLY they determined a sharp threshold for exponential resolution complexity. See also [7, 14] for other examples of specific models of random CSP's that are shown to not have property POLY.

All of these models are specific instances of the general family of models considered in this paper. Thus Theorems 1.1 and 1.2 imply all the results described above, except for those that actually determine the sharp threshold for exponential resolution complexity and those where c is superlinear.

There is also a body of work studying some random CSP's where the constraint-sizes and/or the domains grow with n (see e.g. [37, 21, 18, 20]). Such models do not fall into our general family and so this paper says nothing about them. For example, our theorems do not imply the resolution lower bounds in [37].

3 Preliminaries

Here we give formal definitions of some of the concepts discussed in the introduction, along with other concepts required for the remainder of the paper.

3.1 The Random Model

We use the family of models introduced in [32]. The same family, in a slightly less general form, was introduced independently by Creignou and Daudé [16]. The variables of our problem all have the same domain of permissible values, $\mathcal{D} = \{1, \ldots, d\}$, and all constraints will have size k, for some fixed integers d, k. Given a k-tuple of variables, $(x_1, \ldots, x_k)$, a *restriction* on $(x_1, \ldots, x_k)$ is a k-tuple of values $(\delta_1, \ldots, \delta_k)$ where each $\delta_i \in \mathcal{D}$. A set of restrictions on a k-tuple $(x_1, \ldots, x_k)$ is called a *constraint*. A *constraint satisfaction problem* (CSP) consists of a domain-size d, a constraint-size k, a collection of variables, and a set of constraints on k-tuples of those variables. We say that an assignment of values to the variables of a constraint C *satisfies* C if that assignment is not one of the restrictions on C. An assignment of values to all variables in a CSP satisfies that CSP if every constraint is simultaneously satisfied.

It will be convenient to consider a set of canonical variables $X_1, \ldots, X_k$ which are used only to describe the "pattern" of a constraint. These canonical variables are not variables of the actual CSP. For any d, k there are d^k possible restrictions and 2^{d^k} possible constraints over the k canonical variables. We denote this set of constraints as $\mathcal{C}^{d,k}$. For our random model, one begins by specifying a particular probability distribution, $\mathcal{P}$, over $\mathcal{C}^{d,k}$. Different choices of $\mathcal{P}$ give rise to different instances of the model.

The Random Model: Specify M, n and $\mathcal{P}$ (typically $M = cn$ for some constant c; note that $\mathcal{P}$ implicitly specifies d, k). First choose a random constraint hypergraph with M hyperedges, in the usual manner; i.e., where each k-uniform hypergraph with n vertices and M hyperedges is equally likely. Next, for each hyperedge e, we choose a constraint on the k variables of e as follows: we take a random permutation from the k variables onto $\{X_1, \ldots, X_k\}$ and then we select a random constraint according to $\mathcal{P}$, mapping it onto a constraint on our k variables in the obvious manner. We use $\mathsf{CSP}_{n,M}(\mathcal{P})$ to denote a random CSP drawn from this model with parameters $n, M, \mathcal{P}$.

A constraint set is *symmetric* if for any permutation σ of $\{1, \ldots, k\}$, any $C \in \mathcal{C}$, we have $\tilde{\sigma}(C) \in \mathcal{C}$, where $\tilde{\sigma}$ is the map induced by σ with the obvious definition: $\tilde{\sigma}(C) = \{(\delta_{\sigma(1)}, \ldots, \delta_{\sigma(k)}) \mid (\delta_1, \ldots, \delta_k) \in C\}$. Since the random model takes a random permutation from the k variables in a hyperedge to the k canonical variables before selecting the constraint, $\mathcal{P}$ can be assumed to be symmetric, i.e. $\mathcal{P}(C) = \mathcal{P}(\tilde{\sigma}(C))$ for all $\tilde{\sigma}$ and all C.

The *constraint hypergraph* of a CSP is the k-uniform hypergraph whose vertices correspond to the variables, and whose hyperedges correspond to the k-tuples of variables which have *constraints*. Of course, when $k = 2$, the constraint hypergraph is simply a graph, and so we often call it the *constraint graph*.

Remark 3.1. Constraints on k variables can be simulated by constraints on k' variables for any $k' > k$. This makes it straightforward to extend our results to the case where the constraint sizes can vary. We only require all constraints to have the same size for convenience.

Remark 3.2. When d and/or k grow with n, the satisfiability threshold will often occur at a superlinear number of constraints (see e.g. [37, 21, 20]). The structure of the constraint hypergraph in that case is very different than that of one with a linear number of constraints. This is why we restrict our attention to the case $d, k = O(1)$.

3.2 Resolution Complexity

The *resolution complexity* of a boolean CNF-formula ϕ, denoted $\mathbf{RES}(\phi)$, is the length of the shortest resolution proof that ϕ is unsatisfiable. (If ϕ is satisfiable, then

RES$(\phi) = \infty$.) Mitchell [30] discusses two natural ways to extend the notion of resolution complexity to the setting of CSP, **C-RES** and **NG-RES**. All commonly used resolution-type CSP algorithms correspond nicely to the **C-RES** complexity of the input, but there are some that do not correspond to the **NG-RES**. For that reason, we focus in this paper on the **C-RES** complexity, as did Mitchell in [30] (our results also translate to **NG-RES** complexity.) In short, given an instance $\mathcal{I}$ of a CSP, we construct an equivalent boolean CNF-formula CNF($\mathcal{I}$) in a specific natural manner, and define the resolution complexity **C-RES**$(\mathcal{I}) =$ **RES**(CNF($\mathcal{I}$)).

3.3 The Closure Operation

In this section, we formally define the closure of a constraint set. Then we characterize those constraint sets which have a complete closure in terms of the existence of a subdomain of values which easily satisfies long paths. Such a subdomain will be shown to cause exponential resolution complexity. We begin with some definitions.

A constraint C on variables $x_1, \ldots, x_k$ *permits* $(x_i : \delta, x_j : \gamma)$ if at least one of the d^{k-2} possible tuples $(\delta_1, \ldots, \delta_k)$ with $\delta_i = \delta$ and $\delta_j = \gamma$ is not a restriction of C. Otherwise C *forbids* $(x_i : \delta, x_j : \gamma)$. The constraint $\{(\delta_1, \ldots, \delta_k) \in \mathcal{D}^k \mid \delta_i = \delta \wedge \delta_j = \gamma\}$ forbidding precisely $(x_i : \delta, x_j : \gamma)$ is called the $(x_i : \delta, x_j : \gamma)$-*forbidder* and is denoted $F(x_i : \delta, x_j : \gamma)$.

A *path* of length r in a k-uniform hypergraph H is a sequence $\langle x_0, \ldots, x_r \rangle$ of distinct vertices together with a sequence $\langle e_1, \ldots, e_r \rangle$ of edges such that (1) the edges e_i are mutually vertex disjoint except at $\{x_0, \ldots, x_r\}$; (2) among $\{x_0, \ldots, x_r\}$, the only vertices in e_i are x_{i-1} and x_i, for $1 \leq i \leq r$. $x_0, \ldots, x_r$ are the *connecting variables* and x_0, x_r are the *endpoints* of P. A *cycle* is defined the same way as a path with the exception that $x_0 = x_r$.

A *pendant path* of length r is a path in which no vertices other than the endpoints lie in any edges of H off the path. In other words, there is no restriction on the degrees on the endpoints, each connecting variable has degree 2 in H, and every other vertex in the path has degree 1 in H. (The degree of a vertex is the number of hyperedges to which it is incident.)

A (pendant) path P of length r in a CSP is a sequence of r constraints whose underlying edges form a (pendant) path of length r in the underlying hypergraph. If some assignment α to the variables of P satisfies all the constraints of P with $\alpha(x_0) = \delta$ and $\alpha(x_r) = \gamma$, we say that P *permits* $(x_0 : \delta, x_r : \gamma)$; otherwise P *forbids* $(x_0 : \delta, x_r : \gamma)$. Sometimes we say P permits/forbids (δ, γ), omitting the endpoints when they are not important. Furthermore, for any $\mathcal{D}' \subseteq \mathcal{D}$, we say P permits (δ, γ) *using only values from* $\mathcal{D}'$ if such an α exists with Range$(\alpha) \subseteq \mathcal{D}'$.

A *cyclic CSP* is a CSP whose constraint hypergraph is a cycle.

We now come to the key definitions. The following definitions ($\sim_{\mathcal{C}}$, closure and completeness) will be motivated in the discussion preceding Example 4.1 in Section 4.

For a constraint set $\mathcal{C}$ and values $\delta, \gamma \in \mathcal{D}$, we write $\delta \sim_{\mathcal{C}} \gamma$ if there is some t such that every constraint path over $\mathcal{C}$ of length at least t permits (δ, γ).

Proposition 3.3. *$\sim_{\mathcal{C}}$ is transitive. If $\mathcal{C}$ is symmetric, $\sim_{\mathcal{C}}$ is symmetric as well.*

Definition 3.4. A symmetric constraint set $\mathcal{C}$ is *closed* if for any $\delta, \gamma \in \mathcal{D}$ such that $\delta \not\sim_{\mathcal{C}} \gamma$, any canonical variables X_i, X_j and any $C \in \mathcal{C}$, we have that $\mathcal{C}$ also contains the constraint obtained from C by forbidding $(X_i : \delta, X_j : \gamma)$. Formally, $C \cup F(X_i : \delta, X_j : \gamma) \in \mathcal{C}$. The *closure* cl($\mathcal{C}$) of a constraint set $\mathcal{C}$ is the smallest closed constraint set containing $\mathcal{C}$.

Definition 3.5. A closed, symmetric constraint set $\mathcal{C}$ is *complete* if $\delta \not\sim_{\mathcal{C}} \gamma$ for all $\delta, \gamma \in \mathcal{D}$. Equivalently, it is complete if it contains the constraint that forbids all d^k of the k-tuples. A constraint set which is not complete is *incomplete*.

The key lemma of this section is the following characterisation of incomplete constraint sets. It says that a constraint set is incomplete precisely when, in some sense, long paths can impose no constraint on a particular subdomain of values. This subdomain is called *null-constraining*.

Definition 3.6. Given $\mathcal{C}$, a subdomain $\mathcal{D}' \subseteq \mathcal{D}$ is *null-constraining* if there is some t such that for every constraint path P on $\mathcal{C}$ of length at least t and every pair of values $\delta, \gamma \in \mathcal{D}'$, P permits (δ, γ) using only values from $\mathcal{D}'$.

Lemma 3.7. *Let $\mathcal{C}$ be closed and symmetric. $\mathcal{C}$ is incomplete iff some nonempty subdomain $\mathcal{D}' \subseteq \mathcal{D}$ is null-constraining.*

Proof. If some nonempty subdomain $\mathcal{D}'$ is null-constraining, then in particular $\delta \sim_{\mathcal{C}} \delta$ for some $\delta \in \mathcal{D}'$, so $\mathcal{C}$ is incomplete.

Suppose $\mathcal{C}$ is incomplete. There are $\delta_1, \delta_2 \in \mathcal{D}$ such that $\delta_1 \sim_{\mathcal{C}} \delta_2$. Define $\mathcal{D}' = \{\delta \in \mathcal{D} \mid \delta_1 \sim_{\mathcal{C}} \delta\}$, which is nonempty. By Proposition 3.3, $\delta \sim_{\mathcal{C}} \gamma$ for all $\delta, \gamma \in \mathcal{D}'$; that is, there is a t such that every path P of length at least t permits (δ, γ), possibly using some values from $\mathcal{D} \setminus \mathcal{D}'$ for non-endpoint variables. It remains to show that P still permits (δ, γ) if values can only be chosen from $\mathcal{D}'$.

Claim 3.8. *Any constraint C on P can be replaced by a stronger constraint $C' \in \mathcal{C}$, such that if some variable x_i in C' takes a value from $\mathcal{D}'$, all other variables must take values from $\mathcal{D}'$ as well.*

Proof. Let C' be a superset of C that is maximal in $\mathcal{C}$, i.e. C' is not properly contained in any constraint in $\mathcal{C}$. Assume C' permits $(x_i : \delta, x_j : \gamma)$ for some $\delta \in \mathcal{D}'$, $\gamma \notin \mathcal{D}'$. We must have $\delta \not\sim_{\mathcal{C}} \gamma$, for otherwise $\delta \sim_{\mathcal{C}} \gamma$ and $\delta_1 \sim_{\mathcal{C}} \delta$ implies $\delta_1 \sim_{\mathcal{C}} \gamma$ and hence $\gamma \in \mathcal{D}'$, contradicting the assumption that $\gamma \notin \mathcal{D}'$. Since $\mathcal{C}$ is closed, $C' \cup F(x_i : \delta, x_j : \gamma) \in \mathcal{C}$. This contradicts the maximality of C'. □

By the above claim, we can replace every constraint in P by a stronger constraint from $\mathcal{C}$, none of which permits any $(\delta, \gamma) \in \mathcal{D}' \times (\mathcal{D} \setminus \mathcal{D}')$. The end result is a path P' of $\mathcal{C}$ of length at least t. Recall that all paths of length at least t permit (δ, γ) for any $\delta, \gamma \in \mathcal{D}'$. Therefore P' permits (δ, γ) using only values from $\mathcal{D}'$, hence so does P. □

Proposition 3.9. *If a path P of length $r \geq 2^{|\mathcal{D}|}$ forbids (δ, γ), then infinitely many paths do so, one of which is shorter than $2^{|\mathcal{D}|}$.*

Proof. Let $P = \langle x_0, \ldots, x_r \rangle$. Suppose x_0 takes the value δ, and let $\mathcal{D}_0 = \{\delta\}$. Define $\mathcal{D}_i$ to be the set of values that x_i can take without violating constraints in P, for $1 \leq i \leq r$. Then $\gamma \notin \mathcal{D}_r$ because P is (δ, γ)-forbidding. Since $r \geq 2^{|\mathcal{D}|}$, two $\mathcal{D}_i$'s coincide, i.e. $\mathcal{D}_i = \mathcal{D}_j$ for some $0 \leq i < j \leq r$. Removing the vertices $x_{i+1}, \ldots, x_{j-1}$ and identifying x_i and x_j, we get a shorter (δ, γ)-forbidding path. Vertex removal and identification can be repeated until the resulting path is shorter than $2^{|\mathcal{D}|}$. On the other hand, if we repeat the subpath between x_i and x_j, we can get arbitrarily long (δ, γ)-forbidding paths. □

Remark 3.10. Given a distribution $\mathcal{P}$, there is a simple finite-time procedure to test whether its support $\mathcal{C} = \operatorname{supp} \mathcal{P}$ satisfies the preconditions of Theorem 1.4.

We close this section by sketching how Lemma 3.7 implies our main theorems. If a constraint set has complete closure, then for high clause-variable ratio, the CSP will a.s. contain a small unsatisfiable subproblem, causing polylogarithmic resolution complexity (see Section 4). This proves Theorem 1.1. If a constraint set has incomplete closure, then by Lemma 3.7, there is a nonempty null-constraining subdomain. This will cause exponential resolution complexity, provided there are no short unsatisfiable cycles in $\mathsf{CSP}_{n,M}(\mathcal{P})$ (see Section 6). This will imply Theorem 1.2.

4 Complete Closures

In this section, we consider distributions $\mathcal{P}$ for which $\operatorname{cl}(\operatorname{supp} \mathcal{P})$ is complete. We will show that $\mathsf{CSP}_{n,M}(\mathcal{P})$ a.s. contains a small, structured, unsatisfiable subproblem, called a *forbidding flower*. This structured subproblem generalizes the flower in [34], which is in turn inspired by the snakes of [12].

A forbidding flower is a union of petals. Petals are recursive structures: Each petal functions like a (δ, γ)-forbidding path for appropriate (δ, γ), and subpetals may be attached to adjacent connecting variables along the main path of a petal.

Intuitively, subpetals are needed to simulate constraints in $\operatorname{cl}(\mathcal{C}) \setminus \mathcal{C}$. Indeed, if $C \in \mathcal{C}$ is a constraint, x_i, x_j two of its variables, and P a (δ, γ)-forbidding path from x_i to x_j, then P essentially strengthens C to forbid $(x_i : \delta, x_j : \gamma)$ as well. More precisely, the constraint plus the path functions like $C \cup F(x_i : \delta, x_j : \gamma)$. Repeating this, we can simulate any $C \in \operatorname{cl}(\mathcal{C})$. These simulated constraints can then be used to simulate (δ, γ)-forbidding paths for any $\delta \not\sim_{\operatorname{cl}(\mathcal{C})} \gamma$. If $\operatorname{cl}(\mathcal{C})$ is complete, then all (δ, γ)-forbidding paths can be simulated. If these simulated paths share two common endpoint variables x_1 and x_2, all assignments of x_1, x_2 are forbidden from being satisfying, giving a small unsatisfiable CSP. It is straightforward to show that this CSP has a short resolution proof of unsatisfiability.

Example 4.1. Consider $\mathcal{D} = \{1, 2, 3\}, k = 2$ and $\mathcal{C} = \{C_1, C_2\}$, where $C_1 = \{(1,1), (2,2), (3,3)\}$ and $C_2 = (\{1,2\} \times \{3\}) \cup (\{3\} \times \{1,2\})$. It is easy to see that all constraint paths of length at least 2 permit $(1,1)$. On the other hand, one can check that $1 \not\sim_{\operatorname{cl}(\mathcal{C})} 1$. Indeed, every path made of the constraint C_2 forbids $(1,3)$, hence $1 \not\sim_{\mathcal{C}} 3$, implying $1 \not\sim_{\operatorname{cl}(\mathcal{C})} 3$. Every such path forbids $(2,3)$ as well, so similarly $2 \not\sim_{\operatorname{cl}(\mathcal{C})} 3$. By definition of $\operatorname{cl}(\mathcal{C})$, $C' = C_1 \cup \{(1,3), (2,3)\} \in \operatorname{cl}(\mathcal{C})$. Any path of odd length made of the constraint C' forbids $(1,1)$. However, such a path doesn't exist in our CSP since $C' \notin \mathcal{C}$.

We wish to forbid $(1,1)$, using some structure other than paths. Consider the path $P = \langle x_0, \ldots, x_r \rangle$ which is r copies of C_1, for some odd integer r. To every adjacent pair of connecting variables (x_i, x_{i+1}) in P, attach two paths from x_i to x_{i+1}, each consisting of ℓ copies of C_2, for some integer $\ell \geq 1$. The two paths along with C_1 effectively forbids $(x_i : 1, x_{i+1} : 3)$ and $(x_i : 2, x_{i+1} : 3)$.[3] The resulting graph P' forbids $(x_0 : 1, x_r : 1)$.

To describe these structures effectively, we shall introduce configuration trees and forests. These trees and forests serve only to describe the structure of the forbidding flowers; they are not subproblems in $\mathsf{CSP}_{n,M}(\mathcal{P})$.

Definition 4.2. A *configuration tree* $\mathcal{T}$ is a nonempty rooted tree, each of whose nodes v gets a label from $\mathcal{D}^2$ (i.e. an ordered pair of values in $\mathcal{D}$), call it $(\delta(v), \gamma(v))$.

Labels in a configuration tree $\mathcal{T}$ specify which pairs of values the petals and subpetals forbid. A petal forbids the label at height 0, i.e. the label of the root. Subpetals are attached to every adjacent pair of connecting variables along

[3] For the purpose of forbidding $(x_i : 1, x_{i+1} : 3)$ and $(x_i : 2, x_{i+1} : 3)$, one actually only needs to attach a single "C_2 path" between x_i and x_{i+1}. We attach two such paths in order to be consistent with Definition 4.5.

the main path of the petal, forbidding labels at height 1. These subpetals could in turn have subsubpetals along their main paths, forbidding values at height 2, and so on. As an example, P' in Example 4.1 can yield a configuration tree with three vertices; the root is labelled $(1,1)$ and its children are labelled $(1,3)$ and $(2,3)$ respectively. In fact P' also forbids $(2,2)$, and corresponds to another configuration tree with root $(2,2)$. The relation between configuration trees and petals is defined more formally in Definition 4.5.

Definition 4.3. For a constraint path P and a subset $\Delta \subseteq \mathcal{D}^2$ of pairs of values, the path P *augmented with* Δ, denoted $P \cup \Delta$, is the constraint path with the same underlying hyperedges as P but with each constraint C between (x_i, x_{i+1}) strengthened to forbid all $(\delta,\gamma) \in \Delta$, i.e. C is replaced with $C \cup \bigcup_{(\delta,\gamma)\in\Delta} F(x_i : \delta, x_{i+1} : \gamma)$, for all i.

Notation 4.4. If a node v in a configuration tree has children $v_1, \ldots, v_d$, we denote by $\Delta(v) = \{(\delta(v_i), \gamma(v_i)) \mid 1 \le i \le d\}$ the set of labels of v's children.

Definition 4.5. A $(\mathcal{T},\mathcal{C})$-*forbidding petal* is defined recursively as follows:

1. If $\mathcal{T}$ has only one vertex v labelled (δ,γ), a $(\mathcal{T},\mathcal{C})$-forbidding petal is defined as an (δ,γ)-forbidding path P_v over $\mathcal{C}$.

2. If $\mathcal{T}$ has more than one vertex, let v be its root, let $v_1, \ldots, v_d$ be v's children, and let $\mathcal{T}_1, \ldots, \mathcal{T}_d$ be the subtrees rooted at $v_1, \ldots, v_d$. Then a $(\mathcal{T},\mathcal{C})$-forbidding petal is a path $P_v = \langle y_0, \ldots, y_r\rangle$ over $\mathcal{C}$, together with a $(\mathcal{T}_j,\mathcal{C})$-forbidding petal between every adjacent pair of connecting variables (y_i, y_{i+1}), for $1 \le j \le d$ and $0 \le i < r$. The $(\mathcal{T}_j,\mathcal{C})$-forbidding petals and P_v are mutually vertex-disjoint except at the endpoints of the petals. Also, P_v augmented with the children's labels (i.e. $P_v \cup \Delta(v)$) is $(\delta(v),\gamma(v))$-forbidding. The endpoints of the $(\mathcal{T},\mathcal{C})$-forbidding petal are y_0, y_r.

Note that Case 2 of the above definition actually covers Case 1. Case 1 is included for clarity.

If $\mathrm{cl}(\mathcal{C})$ is complete, we shall argue later (Proposition 4.12) that forbidding petals exist for any $(\delta,\gamma) \in \mathcal{D}^2$. Their union is a forbidding flower.

Definition 4.6. A *configuration forest* $\mathcal{F} = \{\mathcal{T}_{(\delta,\gamma)}\}$ is a collection of $|\mathcal{D}|^2$ configuration trees, one for each $(\delta,\gamma) \in \mathcal{D}^2$, such that $\mathcal{T}_{(\delta,\gamma)}$'s root is labelled (δ,γ).

Definition 4.7. An $(\mathcal{F},\mathcal{C})$-*forbidding flower* between (x_1, x_2) is a collection of $(\mathcal{T}_{(\delta,\gamma)},\mathcal{C})$-forbidding petals between (x_1, x_2), one for each $(\delta,\gamma) \in \mathcal{D}^2$. These petals are mutually vertex-disjoint except at x_1, x_2.

Clearly, an $(\mathcal{F},\mathcal{C})$-forbidding flower between (x_1, x_2) is unsatisfiable, because all possible assignments to (x_1, x_2) are forbidden.

Following [12, 34], we will show that an $(\mathcal{F},\mathcal{C})$-forbidding flower a.s. exists, using first and second moment calculations. In the first moment calculation, we need to know the probability that a particular flower exists. To this end, we need to study the probability that a path of length r is (δ,γ)-forbidding (Definition 4.8). We also study a similar probability for petals (Definition 4.11).

Definition 4.8. For a distribution $\mathcal{P}$ and values $\delta,\gamma \in \mathcal{D}$, let $\pi_r^{\mathcal{P}}(\delta,\gamma)$ be the probability over $\mathcal{P}$ that a random constraint path of length r forbids (δ,γ). Define $\beta^{\mathcal{P}}(\delta,\gamma) = \limsup_r \pi_r^{\mathcal{P}}(\delta,\gamma)^{1/r}$.

The definition of $\beta^{\mathcal{P}}(\delta,\gamma)$ takes a $\limsup$ over a sequence, rather than simply a limit, because the sequence may fail to converge. An example is $\operatorname{supp}\mathcal{P} = \{C\}$, where $C = \{(1,1),(2,2)\}$ and $\mathcal{D} = \{1,2\}$. In this case $\pi_r(1,1)$ alternates between 0 and 1 as r increases.

Proposition 4.9. *If* $\delta \sim_{\operatorname{supp}\mathcal{P}} \gamma$*, then* $\beta^{\mathcal{P}}(\delta,\gamma) = 0$*, otherwise* $\beta^{\mathcal{P}}(\delta,\gamma) > 0$.

The proof is deferred to Section 5.

Definition 4.10. For a distribution $\mathcal{P}$ and a subset $\Delta \subseteq \mathcal{D}^2$, the *distribution of* $\mathcal{P}$ *augmented with* Δ, denoted $\mathcal{P} \cup \Delta$, is the distribution obtained by first picking a constraint C' from $\mathcal{P}$ and then "strengthening C' with Δ." Formally, for any $C \in \mathcal{C}$, define $(\mathcal{P}\cup\Delta)(C) = \sum \mathcal{P}(C')$, where the sum runs over all C' such that $C' \cup \bigcup_{(\delta,\gamma)\in\Delta} F(x_1 : \delta, x_2 : \gamma) = C$. (Since $\mathcal{P}$ is symmetric by assumption, the sum may as well run over all C' such that $C' \cup \bigcup_{(\delta,\gamma)\in\Delta} F(x_i : \delta, x_j : \gamma) = C$ with arbitrary $i \ne j$.)

Definition 4.11. Let $\mathcal{P}$ be a distribution. For a node v in a configuration tree $\mathcal{T}$, define $\pi_r^{\mathcal{P}}(v) = \pi_r^{\mathcal{P}\cup\Delta(v)}(\delta(v),\gamma(v))$ and $\beta^{\mathcal{P}}(v) = \limsup_r \pi_r^{\mathcal{P}}(v)^{1/r}$. For a configuration tree $\mathcal{T}$, define its weight $w^{\mathcal{P}}(\mathcal{T})$ as $\max\{1/(\beta^{\mathcal{P}}(v)k(k-1)) \mid v \in V(\mathcal{T})\}$. For a configuration forest $\mathcal{F}$, define its weight $w^{\mathcal{P}}(\mathcal{F})$ as the maximum of the weights of its trees.

Proposition 4.12. *For any* $\delta \not\sim_{\mathrm{cl}(\mathcal{C})} \gamma$*,* $(\mathcal{T}_{(\delta,\gamma)},\mathcal{C})$*-forbidding petals exist, where* $\mathcal{T}_{(\delta,\gamma)}$ *is a configuration tree whose root is labelled* (δ,γ)*. Furthermore,* $w^{\mathcal{P}}(\mathcal{T}_{(\delta,\gamma)}) < \infty$.

If $\mathrm{cl}(\mathcal{C})$ is complete, the proposition implies that an $(\mathcal{F},\mathcal{C})$-forbidding flower exists for some configuration flower $\mathcal{F}$ of finite weight.

Proof. For $h \ge 0$, define $\mathcal{C}_h$ iteratively as follows. Let $\mathcal{C}_0 = \mathcal{C}$ and $h = 0$. We initially set $\mathcal{C}_{h+1} = \mathcal{C}_h$. For any $\delta \not\sim_{\mathcal{C}_h} \gamma$, any $C \in \mathcal{C}_h$ and any canonical variables X_i, X_j, add $C \cup F(X_i : \delta, X_j : \gamma)$ to $\mathcal{C}_{h+1}$. Then increase h and repeat.

Intuitively, $\mathcal{C}_h$ represents the set of constraints that can be simulated by a constraint from $\mathcal{C}$ and a level h petal.

For any $h \geq 0$, define $\Delta_h = \{(\delta,\gamma) \mid \delta \not\sim_{\mathcal{C}_h} \gamma\}$. For any $(\delta,\gamma) \in \Delta_h \setminus \Delta_{h-1}$ (where we set $\Delta_{-1} = \emptyset$), define $\mathcal{T}_{(\delta,\gamma)}$ be the tree whose root is labelled (δ,γ) and has subtrees $\mathcal{T}_{(\delta',\gamma')}$, one for each $(\delta',\gamma') \in \Delta_{h-1}$. Then $(\mathcal{T}_{(\delta,\gamma)},\mathcal{C})$-forbidding petals exist by induction on h. Moreover, each $\mathcal{T}_{(\delta,\gamma)}$ has finite weight (apply Proposition 4.9 to the distribution $\mathcal{P} \cup \Delta(v)$, where v is the root of $\mathcal{T}_{(\delta,\gamma)}$). □

We close this section by outlining the proof of Theorem 1.1. We will prove that if $\mathrm{cl}(\mathrm{supp}\,\mathcal{P})$ is complete then for sufficiently large c, $\mathsf{CSP}_{n,M=cn}(\mathcal{P})$ will a.s. contain a small forbidding flower. It is straightforward to show that such a flower will have a short resolution refutation. The full details of this proof will appear in the full paper.

Given a configuration forest $\mathcal{F}$ with $w^{\mathcal{P}}(\mathcal{F}) < \infty$, we will focus on $(\mathcal{F},\mathcal{C})$-forbidding flowers for which each main path P_v has length $r(v) = r_n(v)$ for a specific function $r(v) = \Theta(\log n)$. We have to be careful when choosing the lengths $r(v)$. Fix $v \in \mathcal{F}$ and consider the sequence $a_r = \pi_r^{\mathcal{P}}(v)^{1/r}$. Recall that $\limsup a_r = \beta^{\mathcal{P}}(v)$. We wish to choose a length $r(v)$ for which $a_{r(v)}$ is close to $\beta^{\mathcal{P}}(v)$. To do so, we find a nice subsequence of the integers along which a_r converges to $\beta^{\mathcal{P}}(v)$, and we choose $r(v)$ from that subsequence. The following lemma, which will be proved in the next section, shows that we can use an arithmetic progression:

Lemma 4.13. *For any $\delta,\gamma \in \mathcal{D}$, $\pi_r^{\mathcal{P}}(\delta,\gamma)^{1/r}$ converges to $\beta^{\mathcal{P}}(\delta,\gamma)$ along some arithmetic progression.*

We let X denote the number of $(\mathcal{F},\mathcal{C})$-forbidding flowers in $\mathsf{CSP}_{n,cn}(\mathcal{P})$ where each petal path P_v has length $r(v)$. We then compute the first and second moments of X to show that it is a.s. positive. This is inspired by the similar calculations in [34] which were in turn inspired by [12]. Those two papers proved the a.s. existence of much simpler structures. In particular, the fact that we are dealing with petals rather than paths causes our second moment calculations to be much more complicated.

5 Random Walks on Directed Graphs

This section is mainly devoted to proving Lemma 4.13, which says that $\pi_r^{\mathcal{P}}(\delta,\gamma)^{1/r}$ converges to $\beta^{\mathcal{P}}(\delta,\gamma)$ along some arithmetic progression. The behaviour of $\pi_r^{\mathcal{P}}(\delta,\gamma)^{1/r}$ is best studied as a random walk on a related digraph, thus we are led to the analysis of convergence of such a random walk (Theorem 5.1). We remark that the directed graph is, in general, neither strongly connected nor periodic.

Let $G = (V,E)$ be a fixed digraph with positive edge weights. so that at any vertex, the sum of the outgoing edge weights is 1. A random walk on G from u is one which starts at u, and at any stage of the walk at a vertex v, we go to a neighbour w of v with probability the weight of the arc vw.

Theorem 5.1. *For any $u \in V(G)$, $V' \subseteq V(G)$, define $\pi_r(u,V')$ to be the probability that a random walk from u of length r lands on a vertex in V', and $R(u,V') = \limsup_r \pi_r(u,V')^{1/r}$. Then $\pi_r(u,V')^{1/r}$ converges to $R(u,V')$ along some arithmetic progression.*

The theorem may be of independent interest. Quite possibly it has appeared elsewhere, but we could not find it. It will be proved using the following sequence of propositions.

Proposition 5.2. *Let T be a finite set. Assume every $t \in T$ is associated with a sequence $\{a_r(t)\}$, such that $a_r(t) \geq 0$ and $\limsup_r (a_r(t))^{1/r} = a(t)$. Let $b_r = \sum_{t\in T} a_r(t)$ and $b = \limsup_r b_r^{1/r}$. Then $b = \max_{t\in T} a(t)$.*

Proof. Assume $a(s)$ maximizes $a(t)$. Clearly $b_r \geq a_r(s)$. Taking r-th root and then $\limsup_r$ on both sides, we get $b \geq a(s)$. On the other hand, $a_r(t) \leq (a(t)+o(1))^r$ for any t, yielding $b_r \leq |T|(a(s)+o(1))^r$. Taking r-th root and then $\limsup_r$ on both sides, we get $b \leq a(s)$. □

Proposition 5.3. *For any $u,v \in V(G)$, define $\pi_r(u,v)$ to be the probability that a random walk from u of length r lands on v, and $R(u,v) = \limsup_r \pi_r(u,v)^{1/r}$. Then $R(u,V') = \max\{R(u,v) \mid v \in V'\}$.*

Proof. Apply Proposition 5.2 with $T = V'$ and $a_r(v) = \pi_r(u,v)$. Observe that b_r becomes $\pi_r(u,V')$. □

It turns out $R(u,v)$ depends only on the strongly connected components of u and v.

Proposition 5.4. *Assume u,u' belong to the same strongly connected component, and so do v,v'. Then $R(u,v) = R(u',v')$.*

Proof. Let $\mathbf{p}$ be a path from u to u', and $\mathbf{q}$ a path from v' to v. Let $a > 0$ be the probability of traversing $\mathbf{p}$, and $b > 0$ that of traversing $\mathbf{q}$. Let ℓ be the sum of lengths of $\mathbf{p}$ and $\mathbf{q}$. One way to go from u to v in $r+\ell$ steps is to go along $\mathbf{p}$, then go from u' to v' in r steps and finally go along $\mathbf{q}$. Hence $\pi_{r+\ell}(u,v) \geq ab\pi_r(u',v')$. Taking $(r+\ell)$-th root and then $\limsup_r$ on both sides, we get $R(u,v) \geq R(u',v')$. Reversing the roles of (u,v) and (u',v'), we get $R(u',v') \geq R(u,v)$. □

For a vertex u in G, we denote by $[u]$ the strongly connected component of u. If we let $R([u],[v]) = R(u,v)$, the previous proposition shows that $R([u],[v])$ is well-defined. For convenience, we let[4] $R([u]) = R([u],[u])$. For the

[4]Readers familiar with quasi-stationary distribution of absorbing Markov processes (see e.g. [19, 25]) may have realized that $R([u])$ is the spectral radius of the probability transition matrix on $[u]$.

given digraph G, let us denote by G^{C} the component digraph of G. It is obtained from G by contracting every strongly connected component. For a strongly connected component $[u]$ in G, we denote by u^{C} its corresponding vertex in G^{C}. For a walk $\mathbf{w}$ in G, its induced (simple) path $\mathbf{w}^{\mathsf{C}}$ is the path in G^{C} obtained by contracting every strongly connected component of G along $\mathbf{w}$.

Proposition 5.5. *Let* p *be a (simple) path starting from* u^{C} *in* G^{C}*. Define* $\pi_r(\mathsf{p})$ *to be the probability that a random walk in* G *from* u *of length* r *has* p *as its induced path, and* $R(\mathsf{p}) = \limsup_r \pi_r(\mathsf{p})^{1/r}$*. Then* $R([u],[v]) = \max\{R(\mathsf{p}) \mid \mathsf{p}$ *is a* $u^{\mathsf{C}}, v^{\mathsf{C}}$*-path in* $G^{\mathsf{C}}\}$*.*

Proof. Apply Proposition 5.2 with $a_r(\mathsf{p}) = \pi_r(\mathsf{p})$ for any $u^{\mathsf{C}}, v^{\mathsf{C}}$-path p. Observe that $b_r = \pi_r([u],[v])$. □

Lemma 5.6. *Let* p *be a path from* u^{C} *to* v^{C} *in* G^{C}*. Then* $R(\mathsf{p}) = \max\{R([w]) \mid w^{\mathsf{C}} \in V(\mathsf{p})\}$.

Proof. Suppose p is a $u^{\mathsf{C}}, v^{\mathsf{C}}$-path in G^{C}. Assume w_0 in G maximizes $R([w])$ among $w^{\mathsf{C}} \in V(\mathsf{p})$. Take a u, w_0-path $\mathbf{p}$ and a w_0, v-path $\mathbf{q}$ in G, and let ℓ be their sum of lengths. A possible walk of length $r+\ell$ with its induced path being p is like this: It begins with $\mathbf{p}$, then goes from w_0 to w_0 in r steps, finally ends with $\mathbf{q}$. Let $a > 0$ be the probability of traversing $\mathbf{p}$ and $b > 0$ be that of $\mathbf{q}$. Then $\pi_{r+\ell}(\mathsf{p}) \geq ab\pi_r(w_0, w_0)$. Taking $(r+\ell)$-th root and then $\limsup_r$ on both sides, we get $R(\mathsf{p}) \geq R(w_0, w_0) = R([w_0])$.

For the reverse inequality, assume $v(1)^{\mathsf{C}}, \ldots, v(t)^{\mathsf{C}}$ are the vertices of p. Let $\mathbf{w}$ be a walk such that $\mathbf{w}^{\mathsf{C}} = \mathsf{p}$. Renaming if necessary, we may assume $\mathbf{w}$ enters $[v(i)]$ at the vertex $v(i)$. For $1 \leq i \leq t$, let $s(i)$ be the number of steps that $\mathbf{w}$ makes within $[v(i)]$. Now for any $1 \leq i \leq t$, the probability of $\mathbf{w}$ staying in $[v(i)]$ for $s(i)$ steps is $\pi_{s(i)}(v(i),[v(i)]) \leq (R([v(i)]) + f(s(i)))^{s(i)}$, where $f(s(i)) = o(1)$. Note that $t - 1 + \sum_i s(i) = r$, and let $\mathcal{S}_r = \{(s(1), \ldots, s(t)) \mid t - 1 + \sum_i s(i) = r\}$ be the set of all such t-tuples. We have

$$\pi_r(\mathsf{p}) \leq \sum_{\mathbf{s} \in \mathcal{S}_r} \prod_{1 \leq i \leq t} (R([v(i)]) + f(s(i)))^{s(i)}.$$

Let $R_0 = R([w_0])$ (hence $R_0 \geq R([v(i)])$ for any i). There are $|\mathcal{S}_r| \leq r^t$ terms in the sum in (5) (a loose upper bound suffices), each upper bounded by $(R_0 + f(s(i)))^{r-t+1}$.

We can use this to show that $\pi_r(\mathsf{p}) \leq r^t(R_0 + o(1))^{r-t+1}$ (details will appear in the full paper). Taking r-th root and then $\limsup_r$ on both sides, we get $R(\mathsf{p}) \leq R_0 = R([w_0])$. □

Lemma 5.7. *For any vertex* v *in* G*,* $\pi_r(v,v)^{1/r}$ *converges to* $\limsup_r \pi_r(v,v)^{1/r}$ *along multiples of an integer.*

Proof. Let $a_r = \pi_r(v,v)^{1/r}$. If $a_r = 0$ for all $r > 0$, the conclusion is trivial, so assume some $a_r > 0$. Let $S = \{r \mid a_r > 0\}$. For any $m, n \in \mathbb{N}$, $\pi_{m+n}(v,v) \geq \pi_m(v,v)\pi_n(v,v)$, hence

$$a_{r+s} \geq a_r^{r/(r+s)} a_s^{s/(r+s)}. \tag{1}$$

This implies S is closed under addition. It follows that S contains all sufficiently large multiples of d, where $d = \gcd(S)$. Equation (1) also implies a_r are supermultiplicative, and by Fekete's Lemma (e.g. [36]), a_r converges to $\limsup_r a_r$ along multiples of d. □

Proof of Theorem 5.1. The theorem is trivial if V' is not reachable from u, so let us assume this is not the case. Propositions 5.3, 5.5 and Lemma 5.6 together imply $R(u, V') = R(w, w)$ for some w lying on some path from u to V'. Lemma 5.7 asserts that $\pi_r(w,w)^{1/r}$ converges to $R(w,w)$ along some arithmetic progression $S \subseteq \mathbb{N}$. We consider paths $\mathbf{p}$ from u to w and $\mathbf{q}$ from w to V', and use the same analysis as in the first part of Proposition 5.6 to show that $\pi_r(u, V')^{1/r}$ converges to $R(u, V')$ along $S + \ell$. □

We are now ready to prove Lemma 4.13.

Proof of Lemma 4.13. Consider a constraint path $P = \langle x_0, \ldots, x_r \rangle$ with constraints chosen randomly according to $\mathcal{P}$. Suppose x_0 is allowed to take values from $\mathcal{D}_0$. Let $\mathcal{D}_i$ be the set of values that x_i can take without violating constraints in P, for $1 \leq i \leq r$. Then $\langle D_0, \ldots, D_r \rangle$ corresponds naturally to a random walk on a digraph G defined as follows. The vertex set $V(G)$ is the power set of $\mathcal{D}$. For any subdomain $\mathcal{D}' \subseteq \mathcal{D}$ and any $C \in \operatorname{supp} \mathcal{P}$, consider two canonical variables X_1, X_2 in a constraint C. Define $\theta(C, \mathcal{D}')$ to be the set of values δ_2 such that there is $\delta_1 \in \mathcal{D}'$ such that C permits $(X_1 : \delta_1, X_2 : \delta_2)$. Then we put an arc $(\mathcal{D}', \theta(C, \mathcal{D}'))$ of weight $\mathcal{P}(C)$ in G.

Now let $\mathcal{D}_0 = \{\delta\}$ and $V' = \{\mathcal{D}' \subseteq \mathcal{D} \mid \gamma \notin \mathcal{D}'\}$. It is easy to see that $\pi_r^{\mathcal{P}}(\mathcal{D}_0, V')$ is the probability over $\mathcal{P}$ that a constraint path of length r on $\operatorname{supp} \mathcal{P}$ is (δ, γ)-forbidding. The result follows by Theorem 5.1. □

We close this section by proving Proposition 4.9.

Proof of Proposition 4.9. If $\delta \sim_{\mathcal{C}} \gamma$, $\pi_r^{\mathcal{P}}(\delta, \gamma) = 0$ for all sufficiently large r, hence $\beta^{\mathcal{P}}(\delta, \gamma) = 0$. If $\delta \not\sim_{\mathcal{C}} \gamma$, let P be a (δ, γ)-forbidding path over $\mathcal{C}$ of length at least $2^{|\mathcal{D}|}$. Consider the digraph G defined in the proof of Lemma 4.13. P corresponds to a walk $\mathbf{w}$ from $\{\delta\}$ in the digraph. Since G has only $2^{|\mathcal{D}|}$ vertices, $\mathbf{w}$ visits some vertex (at least) twice, say w. Then the portion of $\mathbf{w}$ between the two visits is a circuit from w to itself, say of length $s \geq 1$. Hence $\pi_s(w,w) > 0$. The numbers $a_r = \pi_r(w,w)^{1/r}$ are supermultiplicative (see the proof of Lemma 5.7), hence

$R(w,w) \geq \pi_s(w,w)^{1/s} > 0$. If $u = \{\delta\}$ and $V' = \{\mathcal{D}' \subseteq \mathcal{D} \mid \gamma \notin \mathcal{D}'\}$, then $R(u,V') \geq R(w,w)$ by Propositions 5.3, 5.5 and Lemma 5.6. Hence $\beta^{\mathcal{P}}(\delta,\gamma) = R(u,V') > 0$. □

6 Incomplete Closures

In this section, we turn to constraint sets whose closures are incomplete (Theorems 1.2 and 1.4).

Let $\mathcal{C} = \operatorname{supp}\mathcal{P}$, and assume $\operatorname{cl}(\mathcal{C})$ is incomplete. By Lemma 3.7, some nonempty subdomain $\mathcal{D}' \subseteq \mathcal{D}$ is null-constraining. In other words, there is some integer t such that all constraint paths of length at least t permit all $(\delta,\gamma) \in \mathcal{D}' \times \mathcal{D}'$ using only values from $\mathcal{D}'$ (such $\mathcal{D}'$ and t will be fixed throughout this section). This implies, in particular, that all cycles of length at least t are satisfiable.

We prove that a.s. there is no subexponential resolution proof that the random CSP cannot be satisfied using only values from $\mathcal{D}'$ unless it contains a cycle that cannot be satisfied by those values. The underlying hypergraph will w.u.p.p. have girth at least t and hence have no such a cycle. This implies Theorems 1.2 and 1.4(a).

The proof uses what are by now standard arguments. There is some $\alpha > 0$ such that a.s. every non-cyclic sub-CSP of size at most αn has either (i) a constraint where all but one variable has degree one or (ii) a path of length at least t in which all internal variables do not lie in any other constraint. If that sub-CSP cannot be satisfied using the values of $\mathcal{D}'$ then neither can the sub-CSP formed by deleting the constraint of (i) or the constraints of the path of (ii). It follows recursively that it can indeed be satisfied using $\mathcal{D}'$. Furthermore, if it has size at least $\frac{1}{2}\alpha n$ then there must be $\Theta(n)$ such constraints and/or paths and they can serve as the *boundaries*. The details will appear in the full paper.

For Theorem 1.4(b) the rough argument is as follows: We show that $\mathcal{D}$ can be partitioned into null-constraining subdomains $\mathcal{D}'_1, ...\mathcal{D}'_t$. The random CSP will w.u.p.p. contain cycles $C_1, ..., C_t$ such that each C_i cannot be satisfied using the values from $\mathcal{D}'_i$. It will also contain a vertex x and for each C_i and each $v \in C_i$ a path of length $O(\log n)$ from x to v that forbids every pair (δ,γ) with $\delta \in \mathcal{D}'_i$ and $\gamma \notin \mathcal{D}'_i$. It is easy to see that there is a short resolution proof that this sub-CSP is unsatisfiable. The details will appear in a full version of this paper.

7 Future Work

Corollary 1.3 might, in fact, extend to the stronger statement that for every $\mathcal{P}$ and every c, with the possible exception of some "threshold values" of c, a.s. the shortest resolution refutation of $\mathsf{CSP}_{n,M=cn}(\mathcal{P})$ is either exponential or polylogarithmic. This is true for random 2-SAT and for all models studied in [34].

For those models that have property POLY, it is natural to ask for their thresholds of polynomial resolution complexity. [2] and [34] actually determine, for each of their random models for which POLY holds, a precise value c^*, above which the random CSP has a.s. polynomial resolution complexity and below which it has a.s. exponential resolution complexity. We would like to determine such a value for every $\mathsf{CSP}_{n,M=cn}(\mathcal{P})$ for which POLY holds; i.e. for which $\operatorname{cl}(\operatorname{supp}\mathcal{P})$ is complete. Upon reading Section 4, some readers may guess that c^* is the threshold for the appearance of the first *forbidding flower*. This is the case for random 2-SAT and for all the models in [2] and [34]. However, we have examples of other models for which it is not the case.

8 Acknowledgement

We would like to thank anonymous referees and Toniann Pitassi for helpful comments on an earlier draft of this paper.

References

[1] Dimitris Achlioptas, Paul Beame, and Michael Molloy. Exponential bounds for dpll below the satisfiability threshold. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 132–133, 2004.

[2] Dimitris Achlioptas, Paul Beame, and Michael Molloy. A sharp threshold in proof complexity yields lower bounds for satisfiability search. *Journal of Computer and System Sciences*, 2004. An earlier version appeared in the 33rd Annual ACM Symposium on the Theory of Computing (STOC) 2001.

[3] Dimitris Achlioptas, Arthur Chtcherba, Gabriel Istrate, and Cristopher Moore. The phase transition in 1-in-k SAT and NAE 3-SAT. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 721–722, 2001.

[4] Dimitris Achlioptas, Lefteris M. Kirousis, Evangelos Kranakis, and Danny Krizanc. Rigorous results for random $(2+p)$-SAT. *Theoretical Computer Science*, 265(1-2):109–129, 2001.

[5] Dimitris Achlioptas and Cristopher Moore. Random k-sat: Two moments suffice to cross a sharp threshold. *SIAM Journal on Computing*, 36(3):740–762, 2006.

[6] Noga Alon and Joel H. Spencer. *The Probabilistic Method*. Wiley-Interscience, 2nd edition, 2000.

[7] Paul Beame, Joseph Culberson, David Mitchell, and Cristopher Moore. The resolution complexity of random graph k-colorability. *Discrete Applied Mathematics*, 153(1):25–47, 2005.

[8] Paul Beame, Russell Impagiazzo, and Ashish Sabharwal. Resolution complexity of independent sets in random graphs. In *16th Annual IEEE Conference on Computational Complexity (CCC)*, pages 52–68, 2001.

[9] Paul Beame, Richard M. Karp, Toniann Pitassi, and Michael E. Saks. The efficiency of resolution and Davis-Putnam procedures. *SIAM Journal on Computing*, 31(4):1048–1075, 2002. An earlier version appeared in the 30th Annual ACM Symposium on the Theory of Computing (STOC) 1998.

[10] Paul Beame and Toniann Pitassi. Simplified and improved resolution lower bounds. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, page 274, Washington, DC, USA, 1996. IEEE Computer Society.

[11] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow-resolution made simple. *Journal of the ACM*, 48(2):149–169, 2001.

[12] Vašek Chvátal and Bruce Reed. Mick gets some (the odds are on his side). In *Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 1992.

[13] Vašek Chvátal and Endre Szemerédi. Many hard examples for resolution. *Journal of the ACM*, 35(4):759–768, 1988.

[14] Harold Connamacher and Michael Molloy. The exact satisfiability threshold for a potentially intractable random constraint satisfaction problem. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2004.

[15] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on the Theory of Computing (STOC)*. ACM, New York, 1971.

[16] Nadia Creignou and Hervé Daudé. Generalized satisfiability problems: minimal elements and phase transitions. *Theoretical Computer Science*, 302(1-3):417–430, 2003.

[17] O. Dubois and J. Mandler. The 3-XORSAT threshold. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 769–778, 2002.

[18] Martin Dyer, Alan Frieze, and Michael Molloy. A probabilistic analysis of randomly generated binary constraint satisfaction problems. *Theoretical Computer Science*, 290(3):1815–1828, January 2003.

[19] Seneta Eugene. *Non-negative Matrices and Markov Chains*. Springer-Verlag, New York, 1981.

[20] Alan M. Frieze and Michael Molloy. The satisfiability threshold for randomly generated binary constraint satisfaction problems. *Random Structures and Algorithms*, 28(3):323–339, 2006. An earlier version appeared in the International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM) 2003.

[21] Alan M. Frieze and Nicholas C. Wormald. Random k-SAT: A tight threshold for moderately growing k. *Combinatorica*, 25:297–305, 2005.

[22] Xudong Fu. *On the complexity of proof systems*. PhD thesis, University of Toronto, Toronto, Ont., Canada, Canada, 1996.

[23] Ian P. Gent, Ewan Macintyre, Patrick Prosser, Barbara M. Smith, and Toby Walsh. Random constraint satisfaction: Flaws and structure. *Constraints*, 6(4):345–372, 2001.

[24] Svante Janson, Tomasz Łuczak, and Andrzej Ruciński. *Random Graphs*. Wiley-Interscience, 1st edition, 2000.

[25] James Ledoux, Gerardo Rubino, and Bruno Sericola. Exact aggregation of absorbing markov processes using the quasi-stationary dsitribution. *Journal of Applied Probability*, 31:626–634, 1994.

[26] Colin McDiarmid. On the span of a random channel assignment problem. *Combinatorica*, 27(2):183–203, 2007.

[27] M. Mézard, G. Parisi, and R. Zecchina. Analytic and algorithmic solution of random satisfiability problems. *Science*, 297:812–815, August 2002.

[28] M. Mézard, F. Ricci-Tersenghi, and R. Zecchina. Two solutions to diluted p-spin models and XORSAT problems. *Journal of Statistical Physics*, 111(3–4):505–533, May 2003.

[29] David Mitchell, Bart Selman, and Hector Levesque. Hard and easy distributions of SAT problems. In *Proceedings of the 10th National Conference on Artificial Intelligence*, 1992.

[30] David G. Mitchell. *Resolution Complexity of Constraint Satisfaction*. PhD thesis, University of Toronto, 2002.

[31] David G. Mitchell. Resolution complexity of random constraints. In *Proceedings of Principles and Practices of Constraint Programming*, 2002.

[32] Michael Molloy. Models and thresholds for random constraint satisfaction problems. *SIAM Journal of Computing*, pages 935–949, 2003.

[33] Michael Molloy. When does the giant component bring unsatisfiability? *Combinatorica*, to appear.

[34] Michael Molloy and Mohammad R. Salavatipour. The resolution complexity of random constraint satisfaction problems. *SIAM Journal of Computing*, 37(3):895–922, 2007. An earlier version appeared in the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS) 2003.

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

[36] Jacobus H. van Lint and Richard M. Wilson. *A Course in Combinatorics*. Cambridge University Press, 1993.

[37] Ke Xu and Wei Li. Many hard examples in exact phase transitions with application to generating hard satisfiable instances. *Theoretical Computer Science*, 355:291–302, 2006.