The Complexity of Proving That a Graph Is Ramsey

Massimo Lauria[1], Pavel Pudlák[2], Vojtěch Rödl[3], and Neil Thapen[2]

[1] Royal Institute of Technology, Stockholm
lauria@kth.se
[2] Academy of Sciences of the Czech Republic
{pudlak,thapen}@math.cas.cz
[3] Emory University, Atlanta
rodl@mathcs.emory.edu

Abstract. We say that a graph with n vertices is c-Ramsey if it does not contain either a clique or an independent set of size $c \log n$. We define a CNF formula which expresses this property for a graph G. We show a superpolynomial lower bound on the length of resolution proofs that G is c-Ramsey, for *every* graph G. Our proof makes use of the fact that every Ramsey graph must contain a large subgraph with some of the statistical properties of the random graph.

Introduction

The proof of the existence of c-Ramsey graphs, i.e., graphs that have no clique or independent set of size $c \log n$, was one of the first applications of the probabilistic method in combinatorics [12]. The problem, posed by Erdős, of constructing such graphs explicitly is still open [9]. In this paper we are interested in the problem: how hard is it to *certify* that a graph G of size n is c-Ramsey. By definition, c-Ramsey graphs are in **coNP**; finding polynomial certificates would put this set in **NP**. We believe that this is not possible and our result is a first step in this direction. We show that there is no resolution proof that G is c-Ramsey with length shorter than $n^{\Omega(\log n)}$. Let us stress that this is not a worst-case result: for a fixed constant $c \geq 2$, the lower bound $n^{\Omega(\log n)}$ holds for *every* c-Ramsey graph G. The brute force approach to checking that G satisfies the property takes time $n^{O(\log n)}$ and can be turned into a resolution proof. Hence our result is asymptotically optimal.

We prove our result for one natural formalization of the c-Ramsey property, a formalization that we call *binary*. For an alternative formalization, the *unary* formalization, we derive the same lower bound only for tree-like resolution proofs (see Section 1.1).

Since most SAT solvers used in practice are essentially proof search algorithms for resolution [19], our lower bound on resolution proof size shows that the verification problem is hard for quite a large class of algorithms. Also note that, while it does not follow from the resolution lower bound that there is no algorithm which will construct a Ramsey graph in polynomial time, it does follow

F.V. Fomin et al. (Eds.): ICALP 2013, Part I, LNCS 7965, pp. 684–695, 2013.

that, given an algorithm, there is no polynomial-size resolution proof that the algorithm decides if a graph is c-Ramsey.

The finite Ramsey theorem states that for any k, there is some N such that every graph of size at least N contains a clique or independent set of size k. We write $r(k)$ for the least such N. Computing the actual value of $r(k)$ is challenging, and so far only a few values have been discovered. In fact, even to determine the asymptotic behavior of the function $r(k)$ is a longstanding open problem, in spite of the nontrivial results proved about it, see e.g., [12,23,10].

A c-Ramsey graph is a witness that $r(c \log n) > n$, so proving that a graph is Ramsey is in some sense proving a lower bound for $r(k)$. Previously, proof complexity has focused on upper bounds for $r(k)$. Krishnamurthy and Moll [18] proved partial results on the complexity of proving the exact upper bound, and conjectured that the propositional formalization of the exact upper bound is hard in general. Krajíček later proved an exponential lower bound on the length of bounded depth Frege proofs of the tautology proposed by Krishnamurthy and Moll [17]. The upper bound $r(k) \leq 4^k$ has short proofs in a relatively weak fragment of sequent calculus, in which every formula in a proof has small constant depth [17,21]. Recently Pudlák [22] has shown a lower bound on proofs of $r(k) \leq 4^k$ in resolution. We discuss this in more detail in Section 1. There are also results known about the off-diagonal Ramsey numbers $r(k, s)$ where cliques of size k and independent sets of size s are considered. See [13,1,14,7] for estimates and [8] for resolution lower bounds.

The paper is organized as follows. In Section 1 we formally state our main result, mention some open problems, and then outline the method we will use. In Section 2 we apply this to prove a simple version of our main theorem, restricted to the case when G is a random graph. In Section 3 we prove the full version. This will use one extra ingredient, a result from [20] that every Ramsey graph G has a large subset with some of the statistical density properties of the random graph.

1 Definitions and Results

Resolution [6] is a system for refuting propositional CNFs, that is, propositional formulas in conjunctive normal form. A resolution refutation is a sequence of disjunctions, which in this context we call *clauses*. Resolution has a single inference rule: from two clauses $A \vee x$ and $B \vee \neg x$ we can infer the new clause $A \vee B$ (which is a logical consequence). A *resolution refutation* of a CNF ϕ is a derivation of the empty clause from the clauses of ϕ. For an unsatisfiable formula ϕ we define $L(\phi)$ to be the length, that is, the number of clauses, of the shortest resolution refutation of ϕ. If ϕ is satisfiable we consider $L(\phi)$ to be infinite.

Let $c > 0$ be a constant, whose value will be fixed for the rest of the paper.

Definition 1 (c-Ramsey graph). *We say that a graph with n vertices is* c-Ramsey *if there is no set of $c \log n$ vertices which form either a clique or an independent set.*

We now describe how we formalize the c-Ramsey property in a way suitable for the resolution proof system. Given a graph G on $n = 2^k$ vertices, we will define a formula Ψ_G in conjunctive normal form which is satisfiable if and only there is a homogeneous set (a set inducing an empty or complete subgraph) of size ck in G, that is, if and only if G is not Ramsey. We identify the vertices of G with the binary strings of length k. In this way we can use an assignment to k propositional variables to determine a vertex.

The formula Ψ_G has variables to represent an injective mapping from a set of ck "indices" to the vertices of G, and asserts that the vertices mapped to form either a clique or an independent set. It has a single extra variable y to indicate which of these two cases holds.

In more detail, for each $i \in [ck]$ we have k variables $x_1^i, \ldots, x_k^i$ which we think of as naming, in binary, the vertex of G to which i is mapped. We have an additional variable y, so there are $ck^2 + 1$ variables in total. To simplify notation we will write propositional literals in the form "$x_b^i = 1$", "$x_b^i \neq 0$", "$x_b^i = 0$" and "$x_b^i \neq 1$". The first and the second are aliases for the literal x_b^i. The third and the fourth are aliases for literal $\neg x_b^i$.

Thus the formula Ψ_G expressing that G is c-Ramsey is a conjunction of clauses asserting the following:

1. **The map is injective.** For each vertex $v \in V(G)$, represented as $v_1 \cdots v_k$ in binary, and each pair of distinct $i, j \in [ck]$, we have the clause
$$\bigvee_{b=1}^{k} (x_b^i \neq v_b) \vee \bigvee_{b=1}^{k} (x_b^j \neq v_b).$$
These clauses guarantee that no two indices i and j map to the same vertex v.

2. **If $y = 0$, then the image of the mapping is an independent set.** For each pair of distinct vertices $u, v \in V(G)$, represented respectively as $u_1 \ldots u_k$ and $v_1 \ldots v_k$, and each pair of distinct $i, j \in [ck]$, if $\{u, v\} \in E(G)$ we have the clause
$$y \vee \bigvee_{b=1}^{k} (x_b^i \neq u_b) \vee \bigvee_{b=1}^{k} (x_b^j \neq v_b).$$
These clauses guarantee that, if $y = 0$, then no two indices are mapped to two vertices with an edge between them.

3. **If $y = 1$, then the image of the mapping is a clique.** For each pair of distinct vertices $u, v \in V(G)$, represented respectively as $u_1 \ldots u_k$ and $v_1 \ldots v_k$, and each pair of distinct $i, j \in [ck]$, if $\{u, v\} \notin E(G)$ we have the clause
$$\neg y \vee \bigvee_{b=1}^{k} (x_b^i \neq u_b) \vee \bigvee_{b=1}^{k} (x_b^j \neq v_b).$$
These clauses guarantee that, if $y = 1$, then no two indices are mapped to two vertices without an edge between them.

The formula Ψ_G has $\binom{ck}{2}\left(n + \binom{n}{2}\right)$ clauses in total, and so is unusual in that the number of clauses is exponentially larger than the number of variables. However the number of clauses is polynomial in the number n of vertices of G.

If G is Ramsey, then Ψ_G is unsatisfiable and has only $c \log^2 n + 1$ variables. So we can refute Ψ_G in quasipolynomial size by a brute-force search through all assignments:

Proposition 1. *If G is c-Ramsey, the formula Ψ_G has a (treelike) resolution refutation of size $n^{O(\log n)}$.*

At this point, we should recall the formalization of the Ramsey theorem that is more usually studied in proof complexity. This is the family RAM_n of propositional CNFs, where RAM_n has one variable for each distinct pair of points in $[n]$ and asserts that the graph represented by these variables is $\frac{1}{2}$-Ramsey. Hence RAM_n is satisfiable if and only if any $\frac{1}{2}$-Ramsey graph on n vertices exists. In contrast, our formula Ψ_G is satisfiable if and only if our particular graph G is not c-Ramsey.

Put differently, a refutation of RAM_n is a proof that $r(k) \leq 2^{2k}$. This was recently shown to require exponential size (in n) resolution refutations [22]. On the other hand a refutation of Ψ_G is a proof that G is c-Ramsey, and hence that G witnesses that $r(k) > 2^{\frac{k}{c}}$.

We now state our main result. We postpone the proof to Section 3.

Theorem 1. *Let G be any graph with n vertices. Then $L(\Psi_G) \geq n^{\Omega(\log n)}$.*

If G is not c-Ramsey then this is trivial, since Ψ_G is satisfiable and therefore $L(\Psi_G)$ is infinite by convention. If G is c-Ramsey, then by Proposition 1 this bound is tight and we know that $L(\Psi_G) = n^{\Theta(\log n)}$.

1.1 Open Problems

Our formalization Ψ_G, apart from being natural, is motivated by the τ-tautologies introduced by Krajíček [15]. Although it has similar properties we are not able to show that it is a τ-tautology.

We call Ψ_G the *binary encoding* because the vertices of the graph are represented by strings of propositional variables. One can also consider the *unary encoding* Ψ_G^u in which a vertex of the graph is determined by a single propositional variable. More precisely, the mapping from an index i to the vertices of G is represented by n variables $\{p_v^i : v \in V(G)\}$ and we have clauses asserting that for each i, exactly one of the variables p_v^i is true. Otherwise the structure of Ψ_G^u is similar to that of Ψ_G. As before, if G is a c-Ramsey graph we have the brute-force upper bound $L(\Psi_G^u) = n^{O(\log n)}$. But we are not able to prove a superpolynomial lower bound on resolution size. However we are able to prove such a lower bound if we restrict to treelike resolution, as a corollary of our main theorem (we are grateful to Leszek Kołodziejczyk for pointing out this simple proof).

Theorem 2. *Let G be any c-Ramsey graph with n vertices. Then Ψ_G^u requires treelike resolution refutations of size $n^{\Omega(\log n)}$.*

Proof. (Sketch) Suppose we have a small treelike resolution refutation of the unary formula Ψ_G^u. We can produce from it an at most polynomially larger treelike Res(k) refutation of the binary formula Ψ_G as follows. Replace each variable p_v^i asserting that index i is mapped to vertex v with the conjunction $\bigwedge_{b=1}^k x_b^i = v_b$. The substitution instance of Ψ_G^u is then almost identical to the Ψ_G, except for the additional clauses asserting that every index maps to exactly one vertex; but these are easy to derive in treelike Res(k).

It is well known that every treelike depth $d+1$ Frege proof can be made into a DAG-like depth d Frege proof with at most polynomial increase in size [16]. In particular, we can turn our treelike Res(k) refutation of Ψ_G into a resolution refutation. The lower bound then follows from Theorem 1. □

Lower bounds for DAG-like resolution would have interesting consequences for various areas of proof complexity [3,11]. The problem of proving a superpolynomial lower bound on Ψ_G^u is related to the following open problem (rephrased from [5]): consider a random graph G distributed according to $\mathcal{G}(n, n^{-(1+\epsilon)\frac{2}{k-1}})$ for some $\epsilon > 0$. Does every resolution proof that there is no k-clique in G require size $n^{\Omega(k)}$?

Another natural problem is to extend our lower bound to proof systems stronger than resolution. A superpolynomial lower bound on the proofs of Ψ_G in Res(log) (resolution with logarithmic size conjunctions in clauses) would imply a superpolynomial lower bound on general resolution proofs of Ψ_G^u.

1.2 Resolution Width and Combinatorial Games

The *width* of a clause is the number of literals it contains. The width of a CNF ϕ is the width of its widest clause. Similarly the width of a resolution refutation Π is the width of its widest clause. The width of refuting an unsatisfiable CNF ϕ is the minimum width of Π over all refutations Π of ϕ. We will denote it by $W(\phi)$.

A remarkable result about resolution is that it is possible to prove a lower bound on the proof length by proving a lower bound on the proof width.

Theorem 3 ([4]). *For any CNF ϕ with m variables and width k,*

$$L(\phi) \geq 2^{\Omega\left(\frac{(W(\phi)-k)^2}{m}\right)}.$$

Now consider a game played between two players, called the Prover and the Adversary. The Prover claims that a CNF ϕ is unsatisfiable and the Adversary claims to know a satisfying assignment. At each round of the game the Prover asks for the value of some variable and the Adversary has to answer. The Prover saves the answer in memory, where each variable value occupies one memory location. The Prover can also delete any saved value, in order to save memory. If the deleted variable is asked again, the Adversary is allowed to answer differently. The Prover wins when the partial assignment in memory falsifies a clause of ϕ. The Adversary wins if he has a strategy to play forever.

If ϕ is in fact unsatisfiable, then the Prover can always eventually win, by asking for the total assignment. If ϕ is satisfiable, then there is an obvious winning strategy for the Adversary (answering according to a fixed satisfying assignment). However, even if ϕ is unsatisfiable, it may be that the Prover cannot win the game unless he uses a large amount of memory. Indeed, it turns out that the smallest number of memory locations that the Prover needs to win the game for an unsatisfiable ϕ is related to the width of resolution refutations. (We only need one direction of this relationship – for a converse see [2].)

Lemma 1. *Given an unsatisfiable CNF ϕ, it holds that $W(\phi) + 1$ memory locations are sufficient for the Prover in order to win the game against any Adversary.*

1.3 The Clique Formula

For any graph G, the formula $\Psi_G\restriction_{y=1}$ is satisfiable if and only if G has a clique of size ck. We will call this restricted formula Clique(G). Dually, $\Psi_G \restriction_{y=0}$ is equivalent to Clique($\bar{G}$). Since fixing a variable in a resolution refutation results in a refutation for the corresponding restricted formula, we have

$$\max\left\{L(\text{Clique}(G)), L(\text{Clique}(\bar{G}))\right\} \leq L(\Psi_G).$$

Furthermore we can easily construct a refutation of Ψ_G from refutations of $\Psi_G\restriction_{y=1}$ and $\Psi_G\restriction_{y=0}$. In this way we get

$$L(\Psi_G) \leq L(\text{Clique}(\bar{G})) + L(\text{Clique}(G)) + 1.$$

We can now describe our high-level approach. To lower-bound $L(\Psi_G)$ it is enough to lower-bound $L(\text{Clique}(G))$, which we will do indirectly by exhibiting a good strategy for the Adversary in the game on Clique(G). This game works as follows: the Adversary claims to know ck strings in $\{0,1\}^k$ which name ck vertices in G which form a clique. The Prover starts with no knowledge of these strings but can query them, one bit at a time, and can also forget bits to save memory. The Prover wins if at any point there are two fully-specified strings for which the corresponding vertices are not connected by an edge in G.

We will give a strategy for the Adversary which will beat any Prover limited to ϵk^2 memory for a constant $\epsilon > 0$. It follows by Lemma 1 that Clique(G) is not refutable in width ϵk^2. The formula Clique(G) has ck^2 variables and has width $2k$. Hence applying Theorem 3 we get

$$L(\Psi_G) \geq L(\text{Clique}(G)) \geq 2^{\Omega\left(\frac{(\epsilon k^2 - 2k)^2}{ck^2}\right)} \geq 2^{\Omega(k^2)} \geq n^{\Omega(\log n)}.$$

1.4 Other Notation

We will consider simple graphs with $n = 2^k$ vertices. We identify the vertices with the binary strings of length k. For any vertex $v \in G$ we denote its binary representation by $v_1 \cdots v_k$.

A *pattern* is a partial assignment to k variables. Formally, it is a string $p = p_1 \cdots p_k \in \{*, 0, 1\}^k$, and we say that *$p$ is consistent with v* if for all $i \in [k]$ either $p_i = v_i$ or $p_i = *$. The *size* $|p|$ of p is the number of bits set to 0 or 1. The *empty pattern* is a string of k stars.

For any vertex $v \in V(G)$ we let $N(v)$ be the set $\{u \big| \{v, u\} \in E(G)\}$ of neighbors of v. Notice that $v \notin N(v)$. For any $U \subseteq V(G)$ we let $N(U)$ be the set of vertices of G which neighbor every point in U, that is, $\bigcap_{v \in U} N(U)$. Notice that $U \cap N(U) = \emptyset$.

2 Lower Bounds for the Random Graph

We consider random graphs on n vertices given by the usual distribution $\mathcal{G}(n, \frac{1}{2})$ in the Erdős-Rényi model.

Theorem 4. *If G is a random graph, then with high probability $L(\Psi_G) = n^{\Omega(\log n)}$.*

We will use the method outlined in Section 1.3 above, so to prove the theorem it is enough to give a strategy for the Adversary in the game on Clique(G) which forces the Prover to use a large amount of memory. This is Lemma 3 below. We first prove a lemma which captures the property of the random graph which we need.

Lemma 2. *For a random graph G, with high probability, the following property P holds. Let $U \subseteq V(G)$ with $|U| \leq \frac{1}{3}k$ and let p be any pattern with $|p| \leq \frac{1}{3}k$. Then p is consistent with at least one vertex in $N(U)$.*

Proof. Fix such a set U and such a pattern p. The probability that an arbitrary vertex $v \notin U$ is in $N(U)$ is at least $2^{-\frac{1}{3}k} = n^{-\frac{1}{3}}$. The pattern p is consistent with at least $n^{\frac{2}{3}} - |U|$ vertices outside U. The probability that no vertex consistent with p is in $N(U)$ is hence at most

$$\left(1 - n^{-\frac{1}{3}}\right)^{n^{\frac{2}{3}} - |U|} \leq e^{-(1-o(1))n^{\frac{1}{3}}}.$$

We can bound the number of such sets U by $n^{\frac{1}{3}k} \leq n^{\log n}$ and the number of patterns p by $3^k \leq n^2$, so by the union bound property P fails to hold with probability at most $2^{-\Omega(n^{\frac{1}{3}})}$. □

Lemma 3. *Let G be any graph with property P. Then there is an Adversary strategy in the game on* Clique(G) *which wins against any Prover who uses at most $\frac{1}{9}k^2$ memory locations.*

Proof. For each index $i \in [ck]$, we will write p^i for the pattern representing the current information in the Prover's memory about the ith vertex. The Adversary's strategy is to answer queries arbitrarily (say with 0) as long as the index i being queried has $|p^i| < \frac{1}{3}k - 1$. If $|p^i| = \frac{1}{3}k - 1$, the Adversary privately *fixes* the ith vertex to be some particular vertex v^i of G consistent with p^i, and then answers queries to i according to v^i until, through the Prover forgetting bits, $|p^i|$ falls below $\frac{1}{3}k$ again, at which point the Adversary considers the ith vertex no longer to be fixed.

If the Adversary is able to guarantee that the set of currently fixed vertices always forms a clique, then the Prover can never win. So suppose we are at a point in the game where the Adversary has to fix a vertex for index i, that is, where the Prover is querying a bit for i and $|p^i| = \frac{1}{3}k - 1$. Let $U \subseteq V(G)$ be the set of vertices that the Adversary currently has fixed. It is enough to show that there is some vertex consistent with p^i which is connected by an edge in G to every vertex in U. But by the limitation on the size of the Prover's memory, no more than $\frac{1}{3}k$ vertices can be fixed at any one time. Hence $|U| \leq \frac{1}{3}k$ and the existence of such a vertex follows from property P. □

3 Lower Bounds for Ramsey Graphs

We prove Theorem 1, that for any c-Ramsey graph G on n vertices, $L(\Psi_G) \geq n^{\Omega(\log n)}$. As in the previous section we will do this by showing, in Lemma 5 below, that the Adversary has a strategy for the game on Clique(G) which forces the Prover to use a lot of memory.

Definition 2. *Given sets $A, B \subseteq V(G)$ we define their mutual* density *by*

$$d(A, B) = \frac{e(A, B)}{|A||B|}$$

where we write $e(A, B)$ for the number of edges in G with one end in A and the other in B. For a single vertex v we will write $d(v, B)$ instead of $d(\{v\}, B)$.

Our main tool in our analysis of Ramsey graphs is the statistical property shown in Corollary 1 below, which plays a role analogous to that played by Lemma 2 for random graphs. We use the following result proved in [20, Case II of Theorem 1]:

Lemma 4 ([20]). *There exists constants $\beta > 0$, $\delta > 0$ such that if G is a c-Ramsey graph, then there is a set $S \subseteq V(G)$ with $|S| \geq n^{\frac{3}{4}}$ such that, for all $A, B \subseteq S$, if $|A|, |B| \geq |S|^{1-\beta}$ then $\delta \leq d(A, B) \leq 1 - \delta$.*

Now fix a c-Ramsey graph G. Let S, β and δ be as in the above lemma, and let $m = |S|$. Notice that since our goal is to give an Adversary strategy for the formula Clique(G), we will only use the lower bound $\delta \leq d(A, B)$ from the lemma.

Corollary 1. *Let $X, Y_1, Y_2, \ldots, Y_r \subseteq S$ be such that $|X| \geq rm^{1-\beta}$ and $|Y_1|, \ldots, |Y_r| \geq m^{1-\beta}$. Then there exists $v \in X$ such that $d(v, Y_i) \geq \delta$ for each $i = 1, \ldots, r$.*

Proof. For $i = 1, \ldots, r$ let

$$X_i = \{u \in X \mid d(u, Y_i) < \delta\}.$$

By Lemma 4, each $|X_i| < m^{1-\beta}$. Hence $X \setminus \bigcup_i X_i$ is non-empty and we can take v to be any vertex in $X \setminus \bigcup_i X_i$. □

The next lemma implies our main result, Theorem 1.

Lemma 5. *There is a constant $\epsilon > 0$, independent of n and G, such that there exists a strategy for the Adversary in the game on* Clique(G) *which wins against any Prover who is limited to $\epsilon^2 k^2$ memory locations.*

Proof. Let $\epsilon > 0$ be a constant, whose precise value we will fix later. As in the proof of Lemma 3, the Adversary's replies when queried about the ith vertex will depend on the size of p^i, the pattern representing the current information known to the Prover about the ith vertex. If $|p^i| < \epsilon k - 1$ the Adversary can reply in a somewhat arbitrary way (see below), but if $|p^i| = \epsilon k - 1$ then the Adversary will fix a value v^i for the ith vertex, consistent with p^i, and will reply according to v^i until $|p^i|$ falls back below ϵk, at which point the vertex is no longer fixed. By the limitation on the Prover's memory, no more than ϵk vertices can be fixed simultaneously, which will allow the Adversary to ensure that the set of currently fixed vertices always forms a clique.

Let S, β and δ be as in Lemma 4 and let $m = |S|$. We will need to use Corollary 1 above to make sure that the Adversary can find a v^i with suitable density properties when fixing the ith vertex. But here there is a difficulty which does not arise with the random graph. Corollary 1 only works for subsets of the set S, and S may be distributed very non-uniformly over the vertices of G. In particular, through some sequence of querying and forgetting bits for i, the Prover may be able to force the Adversary into a position where the set of vertices consistent with a small p^i has only a very small intersection with S, so that it is impossible to apply Corollary 1.

Let α be a constant with $0 < \alpha < \beta$, whose precise value we will fix later. We write C_p for the set of vertices of G consistent with a pattern p. We write $P_{\epsilon k}$ for the set of patterns p with $|p| \leq \epsilon k$. To avoid the problem in the previous paragraph, we will construct a non-empty set $S^* \subseteq S$ with the property that, for every $p \in P_{\epsilon k}$, either

$$C_p \cap S^* = \emptyset \quad \text{or} \quad |C_p \cap S^*| > m^{1-\alpha}.$$

In the second case we will call the pattern p *active.* The Adversary can then focus on the set S^*, in the sense that he will pretend that his clique is in S^* and will ignore the vertices outside S^*.

We construct S^* in a brute-force way. We start with $S_0 = S$ and define a sequence of subsets $S_0, S_1, \ldots$ where each $S_{t+1} = S_t \setminus C_p$ for the lexicographically first $p \in P_{\epsilon k}$ for which $0 < |S_t \cap C_p| \leq m^{1-\alpha}$, if any such p exists. We stop as soon as there is no such p, and let S^* be the final subset in the sequence. To show that S^* is non-empty, notice that at each step at most $m^{1-\alpha}$ elements are removed. Furthermore there are at most $|P_{\epsilon k}|$ steps, since a set of vertices C_p may be removed at most once. Recall that $n = 2^k$ and $m \geq n^{\frac{3}{4}}$. We have

$$|P_{\epsilon k}| = \sum_{i=0}^{\epsilon k} 2^i \binom{k}{i} \leq 2^{\epsilon k} \sum_{i=0}^{\epsilon k} \binom{k}{i} \leq n^{\epsilon} n^{H(\epsilon)},$$

where $H(x)$ is the binary entropy function $-x \log x - (1-x)\log(1-x)$, and we are using the estimate $\sum_{i=0}^{\epsilon k} \binom{k}{i} \leq 2^{kH(\epsilon)}$ which holds for $0 < \epsilon < 1$. Then

$$|S^*| \geq |S| - |P_{\epsilon k}| \cdot m^{1-\alpha} \geq n^{\frac{3}{4}} - \epsilon k \cdot n^{\epsilon + H(\epsilon)} n^{\frac{3}{4}(1-\alpha)},$$

so, for large n, S^* is non-empty as long as we choose α and ϵ satisfying

$$\tfrac{3}{4}\alpha > \epsilon + H(\epsilon). \qquad (\star)$$

Notice that if S^* is non-empty then in fact $|S^*| > m^{1-\alpha}$, since S^* must intersect at least the set C_p where p is the empty pattern.

We can now give the details of the Adversary's strategy. The Adversary maintains the following three conditions, which in particular guarantee that the Prover will never win.

1. For each index i, if $|p^i| < \epsilon k$ then p^i is active, that is, $C_{p^i} \cap S^* \neq \emptyset$.
2. For each index i, if $|p^i| \geq \epsilon k$ then the ith vertex is fixed to some $v^i \in C_{p^i} \cap S^*$; furthermore the set U of currently fixed vertices v^j forms a clique.
3. For every active $p \in P_{\epsilon k}$ and every $U' \subseteq U$, we have
$$|C_p \cap S^* \cap N(U')| \geq |C_p \cap S^*| \cdot \delta^{|U'|}.$$

(Recall that $0 < \delta < 1$ is the constant from Lemma 4.) These are true at the start of the game, because no vertices are fixed and each p^i is the empty pattern.

Suppose that, at a turn in the game, the Prover queries a bit for an index i for which he currently has information p^i. If $|p^i| < \epsilon k - 1$, then by condition 1 there is at least one vertex v in $C_{p^i} \cap S^*$. The Adversary chooses an arbitrary such v and replies according to the bit of v. If $|p^i| \geq \epsilon k$, then a vertex $v^i \in C_{p^i}$ is already fixed, and the Adversary replies according to the bit of v^i.

If $|p^i| = \epsilon k - 1$, then the Adversary must fix a vertex v^i for i in a way that satisfies conditions 2 and 3. To preserve condition 2, v^i must be connected to every vertex in the set U of currently fixed vertices. To preserve condition 3, it is enough to choose v^i such that

$$d(v^i, C_p \cap S^* \cap N(U')) \geq |C_p \cap S^* \cap N(U')| \cdot \delta$$

for every active p in $P_{\epsilon k}$ and every $U' \subseteq U$. To find such a v^i we will apply Corollary 1, with one set Y for each pair of a suitable p and U'. We put

$$\begin{aligned} X &= C_{p^i} \cap N(U) \cap S^* \\ Y_{(p,U')} &= C_p \cap N(U') \cap S^* \text{ for each active } p \in P_{\epsilon k} \text{ and each } U' \subseteq U \\ r &= |\{\text{pairs } (p, U')\}| \leq |P_{\epsilon k}| \cdot 2^{|U|}. \end{aligned}$$

We know $|U| \leq \epsilon k$. By condition 1 we know p^i is active, hence $|C_{p^i} \cap S^*| > m^{1-\alpha}$. So by condition 3 we have

$$|X| \geq m^{1-\alpha}\delta^{\epsilon k} = m^{1-\alpha+\frac{4}{3}\epsilon \log \delta}.$$

For similar reasons we have the same lower bound on the size of each $Y_{(p,U')}$. Furthermore

$$r \leq 2^{\epsilon k} \cdot \epsilon k \cdot n^{\epsilon + H(\epsilon)} = \epsilon k \cdot n^{2\epsilon + H(\epsilon)} = \epsilon k \cdot m^{\frac{8}{3}\epsilon + \frac{4}{3}H(\epsilon)}.$$

To apply Corollary 1 we need to satisfy $|X| \geq rm^{1-\beta}$ and $|Y_{(p,U')}| \geq m^{1-\beta}$. Both conditions are implied by the inequality

$$\beta - \alpha > \tfrac{8}{3}\epsilon + \tfrac{4}{3}H(\epsilon) - \tfrac{4}{3}\epsilon \log \delta. \qquad (\dagger)$$

We can now fix values for the constants α and ϵ to satisfy the inequalities $(\star)$ and $(\dagger)$. Since $H(\epsilon)$ goes to zero as ϵ goes to zero, we can make the right hand sides of $(\star)$ and $(\dagger)$ arbitrary small by setting ϵ to be a small constant. We then set α appropriately.

Finally, it is straightforward to check that if the Prover forgets a bit for an index i, then the three conditions are preserved. □

Acknowledgements. Part of this work was done while Lauria was at the Institute of Mathematics of the Academy of Sciences of the Czech Republic, supported by the Eduard Čech Center. Lauria, Pudlák and Thapen did part of this research at the Isaac Newton Institute for the Mathematical Sciences, where Pudlák and Thapen were visiting fellows in the programme *Semantics and Syntax*. Pudlák and Thapen were also supported by grant IAA100190902 of GA AV ČR, and by Center of Excellence CE-ITI under grant P202/12/G061 of GA ČR and RVO: 67985840.

We would also like to thank the anonymous referees for their comments and corrections.

References

1. Ajtai, M., Komlós, J., Szemerédi, E.: A note on Ramsey numbers. Journal of Combinatorial Theory, Series A 29(3), 354–360 (1980)
2. Atserias, A., Dalmau, V.: A combinatorial characterization of resolution width. J. Comput. Syst. Sci. 74(3), 323–334 (2008)
3. Atserias, A., Fichte, J.K., Thurley, M.: Clause-learning algorithms with many restarts and bounded-width resolution. J. Artif. Intell. Res. (JAIR) 40, 353–373 (2011)
4. Ben-Sasson, E., Wigderson, A.: Short proofs are narrow - resolution made simple. In: Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, pp. 517–526 (1999)
5. Beyersdorff, O., Galesi, N., Lauria, M., Razborov, A.A.: Parameterized bounded-depth frege is not optimal. ACM Trans. Comput. Theory 4(3), 7:1–7:16 (2012)

6. Blake, A.: Canonical Expressions in Boolean Algebra. PhD thesis, University of Chicago (1938)
7. Bohman, T., Keevash, P.: The early evolution of the h-free process. Inventiones Mathematicae 181(2), 291–336 (2010)
8. Carlucci, L., Galesi, N., Lauria, M.: Paris-harrington tautologies. In: Proc. of IEEE 26th Conference on Computational Complexity, pp. 93–103 (2011)
9. Chung, F.R.K., Erdős, P., Graham, R.L.: Erdős on Graphs: His Legacy of Unsolved Problems, 1st edn. AK Peters, Ltd. (January 1998)
10. Conlon, D.: A new upper bound for diagonal ramsey numbers. Annals of Mathematics 170(2), 941–960 (2009)
11. Dantchev, S., Martin, B., Szeider, S.: Parameterized proof complexity. Computational Complexity 20, 51–85 (2011), doi:10.1007/s00037-010-0001-1
12. Erdös, P.: Some remarks on the theory of graphs. Bull. Amer. Math. Soc. 53, 292–294 (1947)
13. Erdős, P., Szekeres, G.: A combinatorial problem in geometry. In: Gessel, I., Rota, G.-C. (eds.) Classic Papers in Combinatorics. Modern Birkhäuser Classics, pp. 49–56. Birkhäuser, Boston (1987)
14. Kim, J.H.: The Ramsey number $r(3, t)$ has order of magnitude $t^2/\log(t)$. Random Structures and Algorithms 7(3), 173–208 (1995)
15. Krajicek, J.: Tautologies from pseudo-random generators. Bulletin of Symbolic Logic, 197–212 (2001)
16. Krajíček, J.: Lower bounds to the size of constant-depth propositional proofs. Journal of Symbolic Logic 59(1), 73–86 (1994)
17. Krajíček, J.: A note on propositional proof complexity of some Ramsey-type statements. Archive for Mathematical Logic 50, 245–255 (2011), doi:10.1007/s00153-010-0212-9
18. Krishnamurthy, B., Moll, R.N.: Examples of hard tautologies in the propositional calculus. In: STOC 1981, 13th ACM Symposium on Th. of Computing, pp. 28–37 (1981)
19. Pipatsrisawat, K., Darwiche, A.: On the power of clause-learning sat solvers as resolution engines. Artificial Intelligence 175(2), 512–525 (2011)
20. Prömel, H., Rödl, V.: Non-ramsey graphs are c log n-universal. Journal of Combinatorial Theory, Series A 88(2), 379–384 (1999)
21. Pudlák, P.: Ramsey's theorem in Bounded Arithmetic. In: Schönfeld, W., Börger, E., Kleine Büning, H., Richter, M.M. (eds.) CSL 1990. LNCS, vol. 533, pp. 308–317. Springer, Heidelberg (1991)
22. Pudlák, P.: A lower bound on the size of resolution proofs of the Ramsey theorem. Inf. Process. Lett. 112(14-15), 610–611 (2012)
23. Spencer, J.: Asymptotic lower bounds for Ramsey functions. Discrete Mathematics 20, 69–76 (1977)