COMBINATORICA
Bolyai Society – Springer-Verlag

Combinatorica **37** (2) (2017) 253–268
DOI: 10.1007/s00493-015-3193-9

THE COMPLEXITY OF PROVING THAT A GRAPH IS RAMSEY

MASSIMO LAURIA, PAVEL PUDLÁK, VOJTĚCH RÖDL, NEIL THAPEN

Received October 17, 2013
Online First February 1, 2016

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 c-Ramsey graph must contain a large subgraph with some properties typical for random graphs.

Introduction

The proof of the existence of c-Ramsey graphs, that is, 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 [10]. The problem, posed by Erdős, of constructing such graphs explicitly is still open [8]. The condition of being explicitly constructed can be interpreted in various ways. From the point of view of complexity theory, the most interesting question is whether such graphs can be constructed in polynomial time. In this paper we consider the following related problem: how hard is it to *certify* that a graph G of size n is c-Ramsey? Do polynomial-size certificates exist? A natural certificate would be a resolution proof that G is c-Ramsey. We show that no such proof exists with length shorter than $n^{\Omega(\log n)}$, for any fixed constant $c \geq 2$. Let us stress that our lower bound $n^{\Omega(\log n)}$ holds for *every* c-Ramsey graph G. On the other hand, a brute-force check 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.

Mathematics Subject Classification (2000): 03F20, 05C55

Since most SAT solvers used in practice are essentially proof search algorithms for resolution [15], our lower bound on resolution proof size shows that the problem of verifying that a graph is c-Ramsey 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 c-Ramsey graph in polynomial time, it does follow that, given an algorithm, there is no polynomial-size resolution proof of the fact that the algorithm produces c-Ramsey graphs.

We briefly survey some previous results on the proof complexity of the Ramsey theorem. 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; the least such N is denoted by $r(k)$. Since a c-Ramsey graph is a witness that $r(c\log n) > n$, proving that a graph is c-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 [14] 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 [13]. 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 [13,17]. More recently Pudlák [18] has shown a lower bound on proofs of $r(k) \leq 4^k$ in resolution. There are also results on the proof complexity of the off-diagonal Ramsey theorem where the size of cliques and the size of independent sets are two different parameters, see [7].

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 [16] that every c-Ramsey graph G has a large subset with some of the statistical density properties of the random graph. We conclude the paper with some open problems.

1. Definitions and results

1.1. Resolution

Resolution is one of the most studied propositional proof systems. In this system one can prove every logically valid propositional formula (that is,

every tautology) which is in disjunctive normal form (DNF). However, since a resolution proof is a proof "by contradiction", we usually view resolution rather as a system for *refuting* unsatisfiable propositional CNFs (propositional formulas in conjunctive normal form).

A resolution refutation is a sequence of disjunctions of propositional variables and negated variables; in this context we call the variables and negated variables *literals* and the disjunctions *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 general *resolution refutation* of a CNF ϕ is a derivation of the empty clause from the clauses of ϕ. That is, it is a sequence of clauses that are either clauses of ϕ or are derived from previous clauses in the sequence by applications of the rule. We will also consider *tree-like* refutations where the proof is presented in the form of a binary tree with its vertices labeled by clauses in such a way that the leaves are labeled by clauses of ϕ (the same clause can appear many times) and the root is labeled by the empty clause.

For an unsatisfiable formula ϕ, we define $L(\phi)$ to be the length, measured by the number of clauses, of the shortest resolution refutation of ϕ. If ϕ is satisfiable we consider $L(\phi)$ to be infinite.

1.2. Formalizing the Ramsey property

Let $c \geq 2$ be a constant, whose value will be fixed for the rest of the paper.[1]

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 our formalization of 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 if 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 c-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. We will also sometimes talk about the j-th *coordinate* of a vertex v, meaning the j-th bit in its representation as a string.

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 map onto

[1] All results hold true for any $c > 0$, but it is not known whether for $1/2 < c < 2$ there are infinitely many c-Ramsey graphs.

either a clique or an independent set in a one-to-one manner. Furthermore, Ψ_G has one additional 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 the coordinates of the vertex of G to which i is mapped. With the additional variable y, 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 $\neg x_b^i$.

Formally, the formula Ψ_G expressing that G is c-Ramsey is a conjunction of the clauses listed in 1, 2 and 3 below. The conjunction of the clauses in each group expresses the property stated in the heading.

1. **The mapping from indices to G 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 $u_1 \cdots u_k$ and $v_1 \cdots v_k$, 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 $u_1 \cdots u_k$ and $v_1 \cdots v_k$, 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 $m = c\log^2 n + 1$ variables. It is a conjunction of $\binom{ck}{2}\left(1+\binom{n}{2}\right) = 2^{O(\sqrt{m})}$ 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 n, the number of vertices of G.

If G is c-Ramsey, that is, if Ψ_G is unsatisfiable, we can refute Ψ_G in quasipolynomial size by systematically eliminating variables, as is done in the proof of the completeness of resolution, because it has only $c\log^2 n + 1$ variables. We state this fact as:

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

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 $L(\Psi_G) = n^{\Theta(\log n)}$.

For comparison, we recall the usual formalization of the Ramsey theorem as previously studied in proof complexity. (The reader who is only interested in our main result may skip to the next subsection.) 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 [18]. On the other hand a refutation of Ψ_G is a proof that G is c-Ramsey, and hence that G witnesses that $r(k) > 2^{\frac{k}{c}}$.

1.3. 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 $W(\phi)$. We assume that the clauses of ϕ are always present in the proof, so the width of a proof is at least the width of the formula.

Due to a remarkable result of Ben-Sasson and Wigderson, in many cases it is possible to prove a lower bound on the proof length by proving a lower bound on the proof width. We will use this approach for our formula.

Theorem 2 ([4]). *For any CNF ϕ with m variables and width k,*

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

To prove a lower bound on the width, it often helps to view resolution refutations as strategies for a combinatorial game associated with the formula. We will also present our proof in this way.

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

Formally, Prover is a deterministic procedure (function) that, given the time and the contents of memory, decides what to erase from memory and which variable to ask Adversary the value of. Similarly, Adversary is a procedure that, given the content of Prover's memory and the variable Prover asks, produces a bit as a reply.

If ϕ is in fact unsatisfiable, then Prover can always eventually win by asking variable-by-variable for the total assignment and forgetting nothing. 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 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 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, Prover needs only $W(\phi)+1$ memory locations in order to win the game against any Adversary.*

The idea of the proof is to view a refutation of ϕ as a strategy for Prover. The players start at the empty clause and move upwards to the initial clauses. Prover's queries are the variables with respect to which clauses are resolved. The content of Prover's memory corresponds to the current clause.

Thus the task of proving a lower bound on the size of resolution refutations reduces to proving a lower bound on the size of the memory needed by the Prover. Specifically, we need to prove a lower bound of the form $\Omega(k^2)$.

1.4. The clique formula

We now specify the game for the formula that we are interested in. First, however, we observe that by fixing the bit y we can reduce the proof of the lower bound to a proof of a lower bound for a simpler 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). Thus Clique(G) is the conjunction of the clauses from group 1 and group 3 where we delete the literal $\neg y$. 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

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

Hence, it suffices to prove a lower bound on $L(\text{Clique}(G))$. We observe that we do not lose much by this reduction because we can easily construct a refutation of Ψ_G from refutations of $\Psi_G\restriction_{y=1}$ and $\Psi_G\restriction_{y=0}$ whose size is at most $L(\text{Clique}(\bar{G}))+L(\text{Clique}(G))+1$.

We can now give a high-level description of our approach. To prove a lower bound on $L(\Psi_G)$ it is enough to prove a lower bound on $L(\text{Clique}(G))$, which we will do indirectly by exhibiting a good strategy for Adversary in the game on Clique(G). This game for Clique(G) works as follows. Adversary claims to know ck strings in $\{0,1\}^k$ that are coordinates of ck vertices in G that form a clique. 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. 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. (In particular, Prover wins if two strings are equal.) Note that in the nontrivial case when G does not have a clique of size ck, Adversary only pretends to have strings that determine such a clique. So he may be forced to change his answers in order to be consistent with his claim.

We will give a strategy for 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 2 we get

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

1.5. 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 a vertex $v \in V(G)$, we denote by $N(v)$ 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, $N(U) = \bigcap_{v \in U} N(v)$. Notice that $U \cap N(U) = \emptyset$.

2. Lower bounds for the random graph

As a warm-up, we will prove our result for the special case of random graphs. The reader not interested in it may safely skip this section.

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

Theorem 3. *If G is a random graph, then with high probability $L(\Psi_G) = n^{\Omega(\log n)}$. More precisely, there exists a constant $\epsilon > 0$ such that the probability that $L(\Psi_G) \leq n^{\epsilon \log n}$ tends exponentially to zero as n goes to infinity.*

We will use the approach outlined above and define a strategy for the Adversary in the game on Clique(G) which forces Prover to use a large amount of memory. This strategy, presented in Lemma 3 below, is simpler than the one used in the general case. We first prove a lemma which captures the property of the random graph which we need. Recall that $k = \log_2 n$.

Lemma 2. *For a random graph G, the following property P holds with high probability. 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 $2^{-k/3} = n^{-1/3}$. The pattern p is consistent with at least $n^{2/3} - |U|$ vertices outside U. The probability that no vertex consistent with p is in $N(U)$ is hence at most

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

We can bound the number of such sets U by $n^{k/3} \leq n^{\log n}$ 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 $2^{-\Omega(n^{1/3})}$. ∎

Lemma 3. *Let G be any graph with the property P defined in Lemma 2. 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 Prover's memory about the ith vertex. 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 $|p^i| = \frac{1}{3}k - 1$, 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 Prover forgetting bits, $|p^i|$ falls below $\frac{1}{3}k$ again, at which point Adversary considers the ith vertex no longer to be fixed.

If Adversary is able to guarantee that the set of currently fixed vertices always forms a clique, then Prover can never win. So suppose we are at a point in the game where Adversary has to fix a vertex for index i, that is, where 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 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 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 c-Ramsey graphs

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

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 c-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 which is established in Case II of the proof Theorem 1 of [16]:

Lemma 4 ([16]). *There exist 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 $\delta \leq d(A,B) \leq 1-\delta$.*

(Recall that c is fixed throughout the paper; in fact, β and δ depend on c.)

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 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, 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 Prover about the ith vertex. If $|p^i| < \epsilon k - 1$, Adversary can reply in a somewhat arbitrary way (see below), but if $|p^i| \geq \epsilon k - 1$, then 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 Prover's memory, no more than ϵk vertices can be fixed simultaneously, which will allow 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 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 we cannot assume that S is distributed uniformly with respect to the sets defined by patterns. In particular, through some sequence of querying and forgetting bits for i, Prover may be able to force Adversary into a position where the set of vertices consistent with a small p^i has only a very small intersection with S, and then it will not be possible 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 mentioned 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*. 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 straight-forward 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 $0<\epsilon<1/2$. Then

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

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

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

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 Adversary's strategy. Adversary's strategy is to maintain the following three conditions, which clearly guarantee that 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 three conditions are true at the start of the game, because no vertices are fixed and each p^i is the empty pattern. It remains to prove that they can be preserved as long as Prover does not exceed the limitation on his memory.

Suppose that, at a turn in the game, Prover queries a bit for an index i for which he currently has information p^i. We consider three cases.

- If $|p^i| < \epsilon k - 1$, then by condition 1 there is at least one vertex v in $C_{p^i} \cap S^*$. 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 Adversary replies according to the bit of v^i.
- If $\epsilon k - 1 \leq |p^i| < \epsilon k$, then 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
$$d(v^i, C_p \cap S^* \cap N(U')) \geq \delta$$
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 that $|U| \leq \epsilon k$. By condition 1, we know that p^i is active, hence $|C_{p^i} \cap S^*| > m^{1-\alpha}$. So, by condition 3, we have
$$|X| \geq m^{1-\alpha} \delta^{\epsilon k} = m^{1-\alpha} n^{\epsilon \log \delta} \geq m^{1-\alpha+\frac{4}{3}\epsilon \log \delta},$$

because $2^k = n$, $m \leq n^{3/4}$ and $\log \delta < 0$. For similar reasons, we have the same lower bound on the size of each $Y_{(p,U')}$. Furthermore,

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

To apply Corollary 1, we need to satisfy $|X| \geq rm^{1-\beta}$ and $|Y_{(p,U')}| \geq m^{1-\beta}$. Both conditions hold as long as α and ϵ satisfy

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

We now fix values for the constants α and ϵ to satisfy the inequalities $(\star)$ and $(\dagger)$. We put $\alpha = \beta/2$. Then we can satisfy $(\star)$ and $(\dagger)$ by setting ϵ to be a small constant, because $H(\epsilon)$ goes to zero as ϵ goes to zero.

Finally, it is straightforward to check that if the Prover's move is to forget a bit for an index i, then the three conditions are preserved. ∎

4. Open problems

Our formula Ψ_G, apart from being natural, is motivated by a conjecture proposed independently by Krajíček in [11] and Alekhnovich, Ben-Sasson, Razborov and Wigderson in [1]. The conjecture states that for some class of pseudorandom generators, if $\Gamma\colon \{0,1\}^n \to \{0,1\}^m$, $n < m$, is in the class, then the propositional formula expressing the fact that $\Gamma(x) \neq b$ is a hard tautology for any fixed $b \in \{0,1\}^m$ that is not in the range of Γ. Note that for $c \geq 2$, the graphs that are not c-Ramsey can be efficiently encoded by less than $\binom{n}{2}$ bits. So one can express the fact that a graph G is c-Ramsey in the same way as it is required in the conjecture about pseudorandom generators. Unfortunately, our formula Ψ_G is not of such a form, nor we were able to show that it is equivalent to such a formula. Thus it would be interesting to prove a lower bound for other formalizations of the fact that G is c-Ramsey.

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^u_G 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 $\{p^i_v\colon v \in V(G)\}$ and we have clauses asserting that for each i, exactly one of the variables p^i_v is true. Otherwise the structure of Ψ^u_G is similar to that of Ψ_G. As before, if G is a c-Ramsey graph we have the brute-force upper bound $L(\Psi^u_G) = n^{O(\log n)}$, 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 tree-like resolution, as a corollary of our main theorem. (We are grateful to Leszek Kołodziejczyk for pointing out this simple proof.)

Theorem 4. *Let G be any c-Ramsey graph with n vertices. Then Ψ_G^u requires tree-like resolution refutations of size $n^{\Omega(\log n)}$.*

Proof. (Sketch) Suppose we have a small tree-like resolution refutation of the unary formula Ψ_G^u. We can produce from it an at most polynomially larger tree-like 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 Ψ_G, except for the additional clauses asserting that every index maps to exactly one vertex; but these are easy to derive in tree-like Res(k).

It is well known that every tree-like depth $d+1$ Frege proof can be made into a DAG-like depth d Frege proof with at most polynomial increase in size [12]. In particular, we can turn our tree-like 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 [3,9]. The problem of proving a superpolynomial lower bound on Ψ_G^u is related to the following open problem (rephrased from [6]): 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 $n^{\Omega(k)}$? For tree-like resolution this was proved already in [5].

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.

Acknowledgments. 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 in Cambridge, where Pudlák and Thapen were visiting fellows in the programme *Semantics and Syntax*. Lauria was also funded by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement no. 279611. 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. V. Rödl was supported by the NSF grants DMS 0800070 and DMS 1301698.

References

[1] M. Alekhnovich, E. Ben-Sasson, A. A. Razborov and A. Wigderson: Pseudo-random generators in propositional proof complexity, *SIAM Journal on Computing*, **34** (2004), 67–88, a preliminary version appeared in *FOCS '00*.

[2] A. Atserias and V. Dalmau: A combinatorial characterization of resolution width, *J. Comput. Syst. Sci.*, **74** (2008), 323–334.

[3] A. Atserias, J. K. Fichte and M. Thurley: Clause-learning algorithms with many restarts and bounded-width resolution, *J. Artif. Intell. Res. (JAIR)*, **40** (2011), 353–373.

[4] E. Ben-Sasson and A. Wigderson: Short proofs are narrow - resolution made simple, in: *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, 517–526, 1999.

[5] O. Beyersdorff, N. Galesi and M. Lauria: Parameterized complexity of DPLL search procedures, *ACM Transactions on Computational Logic*, **14** (2013), 1–21.

[6] O. Beyersdorff, N. Galesi, M. Lauria and A. A. Razborov: Parameterized bounded-depth Frege is not optimal, *ACM Trans. Comput. Theory*, **4** (2012), 1–16.

[7] L. Carlucci, N. Galesi and M. Lauria: Paris-Harrington tautologies, in: *Proc. of IEEE 26th Conference on Computational Complexity*, 93–103, 2011.

[8] F. R. K. Chung, P. Erdős and R. L. Graham: *Erdős on Graphs: His Legacy of Unsolved Problems*, AK Peters, Ltd., 1 edition, 1998.

[9] S. Dantchev, B. Martin and S. Szeider: Parameterized proof complexity, *Computational Complexity*, **20** (2011), 51–85.

[10] P. Erdős: Some remarks on the theory of graphs, *Bull. Amer. Math. Soc*, **53** (1947), 292–294.

[11] J. Krajíček: Tautologies from pseudo-random generators, *Bulletin of Symbolic Logic*, 197–212, 2001.

[12] J. Krajíček: Lower bounds to the size of constant-depth propositional proofs, *Journal of Symbolic Logic*, **59** (1994), 73–86.

[13] J. Krajíček: A note on propositional proof complexity of some Ramsey-type statements, *Archive for Mathematical Logic*, **50** (2011), 245–255.

[14] B. Krishnamurthy and R. N. Moll: Examples of hard tautologies in the propositional calculus, in: *STOC 1981, 13th ACM Symposium on Th. of Computing*, 28–37, 1981.

[15] K. Pipatsrisawat and A. Darwiche: On the power of clause-learning SAT solvers as resolution engines, *Artificial Intelligence*, **175** (2011), 512–525.

[16] H. Prömel and V. Rödl: Non-Ramsey graphs are c log n-universal, *Journal of Combinatorial Theory, Series A*, **88** (1999), 379–384.

[17] P. Pudlák: Ramsey's theorem in Bounded Arithmetic, in: *Proceedings of Computer Science Logic 1990*, 308–317, 1991.

[18] P. Pudlák: A lower bound on the size of resolution proofs of the Ramsey theorem, *Inf. Process. Lett.*, **112** (2012), 610–611.

Massimo Lauria

KTH Royal Institute of Technology
Stockholm
lauria@kth.se

Pavel Pudlák

Czech Academy of Sciences
Prague
pudlak@math.cas.cz

Vojtěch Rödl

Emory University
Atlanta
rodl@mathcs.emory.edu

Neil Thapen

Czech Academy of Sciences
Prague
thapen@math.cas.cz