THE RESOLUTION COMPLEXITY OF INDEPENDENT SETS AND VERTEX COVERS IN RANDOM GRAPHS

PAUL BEAME, RUSSELL IMPAGLIAZZO,
AND ASHISH SABHARWAL

Abstract. We consider the problem of providing a resolution proof of the statement that a given graph with n vertices and average degree roughly Δ does not contain an independent set of size k. For randomly chosen graphs and $k \leq n/3$, we show that such proofs asymptotically almost surely require size roughly exponential in n/Δ^6. This, in particular, implies a $2^{\Omega(n)}$ lower bound for constant degree graphs, and for $\Delta \approx n^{1/6}$, shows that there are almost always no short resolution proofs for k as large as $n/3$ even though a maximum independent set is likely to be much smaller, roughly $n^{5/6}$ in size. Our result implies that for graphs that are not too dense, almost all instances of the independent set problem are hard for resolution. Further, it provides an unconditional exponential lower bound on the running time of resolution-based search algorithms for finding a maximum independent set or approximating it within a factor of $\Delta/(6 \ln \Delta)$. We also give relatively simple upper bounds for the problem and show them to be tight for the class of exhaustive backtracking algorithms. We deduce similar complexity results for the related vertex cover problem on random graphs, proving, in particular, that no polynomial-time resolution-based method can achieve an approximation within a factor of 3/2.

Keywords. resolution, proof complexity, hardness of approximation, independent set, vertex cover

Subject classification. 03F20, 68Q17

1. Introduction

An independent set in an undirected graph is a set of vertices no two of which share an edge. The problem of determining whether or not a given graph contains an independent set of a certain size is one of the classic NP-complete problems (Karp 1972). Consequently, the complementary problem of determining non-existence of independent sets of that size in the graph is co-NP-complete. This paper studies the problem of providing a resolution-based logical proof of the non-existence of independent sets.

Any result that holds for nearly all graphs can be alternatively formalized as a result that holds with very high probability when a graph is chosen at random from a "fair" distribution. We use this approach and study the resolution complexity of the independent set problem in random graphs chosen from a standard distribution. Independent sets and many other combinatorial structures in random graphs have very interesting mathematical properties as discussed at length in the texts by Bollobás (1985) and Janson, Łuczak & Ruciński (2000). In particular, the size of the largest independent set can be described with high certainty and accuracy in terms of a simple graph parameter, namely, its edge-density.

This work proves that given almost any graph G and a number k, exponential-size resolution proofs are required to show that G does not contain an independent set of size k. In fact, when G has no independent set of size k, exponential-size resolution proofs are required to show that independent sets of even a much larger size $k' \gg k$ do not exist in G. This yields running time lower bounds for certain classes of algorithms for approximating the size of the largest independent sets in random graphs.

Closely related to the independent set problem are the problems of proving the non-existence of cliques or vertex covers of a given size. Our results for the independent set problem also lead to bounds for these problems. As the approximations for the vertex cover problem act differently from those for independent sets, we state the results in terms of vertex covers as well as independent sets. (Clique approximations are essentially identical to independent set approximations.)

Many algorithms for finding a maximum-size independent set have been proposed. Influenced by algorithms of Tarjan (1972) and Tarjan & Trojanowski (1977), Chvátal (1977) devised a specialized proof system for the independent set problem. In this system he showed that with probability asymptotically approaching 1, proofs of non-existence of large independent sets in random graphs with a linear number of edges must be exponential in size. Chvátal's system captures many backtracking algorithms for finding a maximum independent set, including those of Tarjan (1972), Tarjan & Trojanowski (1977), Jian (1986), and Shindo & Tomita (1990). In general, the transcript of any f-*driven algorithm* (Chvátal 1977) for independent sets running on a given graph can be translated into a proof in Chvátal's system.

Our results use the well-known *resolution* proof system of Robinson (1965) for generic propositional logic, rather than Chvátal's specialized proof system for independent sets. Given a graph G and an integer k, we consider encoding the existence of an independent set of size k in G as a propositional formula and examine the proof complexity of such formulas in resolution. Resolution on one of the encodings we present captures the behavior of Chvátal's proofs on the corresponding graphs. For all our encodings, we show that given a randomly chosen graph G of moderate edge density, almost surely, the size of any resolution proof of the statement that G does not have an independent set of a certain size must be exponential in the number of vertices in G. This implies an exponential lower bound on the running time of many algorithms for searching for, or even approximating, a maximum independent set or minimum vertex cover in G.

Although resolution is a relatively simple and well-studied proof system, one may find the concept of resolution proofs of graph theoretic problems somewhat unnatural. The tediousness of propositional encodings and arguments related to them contributes even more to this. Chvátal's proof system, on the other hand, is completely graph theoretic in nature and relates well to many known algorithms for the independent set problem. By proving that resolution can efficiently simulate Chvátal's proof system, we provide another justification for studying the complexity of resolution proofs of graph problems.

We show that our results extend the known lower bounds for Chvátal's system (Chvátal 1977) to resolution and also extend them to graphs with many more than a linear number of edges, yielding bounds for approximation algorithms as well as for exact computation. More precisely, we show that no resolution-based technique can achieve a polynomial-time approximation of the maximum independent set size within a factor of $\Delta/(6 \ln \Delta)$. For the vertex cover problem, we show an analogous result for approximation factors better than $3/2$. We discuss how these results complement the known hardness of approximation results by Håstad (1999), Trevisan (2001), and Dinur & Safra (2005), which are conditioned on $\mathrm{P} \neq \mathrm{NP}$.

Recently, by computing a property related to the Lovász number of a random graph, more precisely its vector chromatic number, Coja-Oghlan (2005) gave an expected polynomial-time $O(\sqrt{\Delta}/\ln \Delta)$-approximation algorithm for the size of the maximum independent set in random graphs of density Δ. Our results immediately imply that this new approach is provably stronger than resolution-based algorithms.

In the proof complexity realm, exponential bounds for specialized structured formulas and for unstructured random k-CNF formulas have previously been shown by several researchers including Beame *et al.* (1998); Chvátal & Szemerédi (1988); Haken (1985); Razborov (2004); Urquhart (1987); and Ben-Sasson & Wigderson (2001). However, much less is known for large classes of structured formulas. Our results significantly extend the families of structured random formulas for which exponential resolution lower bounds are known beyond the graph coloring example recently shown by Beame *et al.* (2005). (Note that our results neither imply nor follow from those of Beame *et al.* (2005). Although the non-existence of an independent set of size n/K in a graph of n vertices implies that the graph is not K-colorable, the argument requires an application of the pigeonhole principle which is not efficiently provable in resolution (Haken 1985).)

1.1. Main Idea. Our approach can be summarized as follows. For obtaining our lower bounds, instead of looking at the general problem of disproving the existence of *any* large independent set in a graph, we focus on a restricted class of independent sets that we call *block-respecting independent sets*. We show that even ruling out this smaller class of independent sets requires exponential-size resolution proofs. These restricted independent sets are simply the ones obtained by dividing the n vertices of the given graph into k consecutive blocks of equal size (assuming k divides n) and choosing one vertex from each block. Since it is easier to rule out a smaller class of independent sets, the lower bounds we obtain for the restricted version are stronger and imply lower bounds for the general problem.

We note that while block-respecting independent sets are a helpful tool in analyzing general resolution proofs, we are able to derive a better lower bound for DPLL proofs by applying a counting argument directly to the general problem. This stronger DPLL lower bound is obtained in the same spirit as a stronger $2^{\Omega(n \log n)}$ lower bound obtained by Dantchev & Riis (2001) for DPLL proofs of the weak pigeonhole principle, which does not follow from similar bounds techniques for general resolution.

Most of the known resolution complexity lower bounds can be proved using a general result of Ben-Sasson & Wigderson (2001) that is derived from earlier arguments by Haken (1985) and Clegg, Edmonds & Impagliazzo (1996). It provides a relationship between the size of resolution proofs and their *width*, the length of their longest clause. Fitting into this framework, our main lower bound argument uses the property that any proof of non-existence of an independent set of a certain size in a random graph is very likely to refer to a relatively large fraction of the vertices of the input graph, and that any clause capturing key properties of this large fraction of vertices must have large width.

More precisely, the proof can be broadly divided into two parts, both of which use the fact that random graphs are almost surely locally sparse. We first show that the minimum number s of input clauses that are needed for any refutation of the problem is large for most graphs. We then use combinatorial properties of independent sets in random graphs to say that any clause minimally implied by a relatively large subset of these s clauses has to be large. Here minimally implied means that implied by the size-s set of clauses but not by any proper subset of it. These two arguments together allow us to deduce that the width, and therefore size, of any such refutation must be large.

The lower bounds obtained this way turn out to apply even when the (unsatisfiable) propositional formula under consideration encodes the existence of a k-independent set for k much larger than the maximum independent set one expects to see in the underlying random graph. We exploit this fact to derive an almost-certain hardness of approximation result for resolution-based algorithms for the independent set problem.

1.2. Road Map. We begin by discussing resolution and DPLL systems in Section 2, and then outline some basic properties of independent sets, especially in the context of random graphs, in Section 3. In Section 4 we describe three natural propositional encodings of the independent set problem and compare proof sizes for the different encodings. In Sections 5 we give a simulation of Chvátal's proof system by resolution. Section 6 discusses the proof complexity of related graph theory problems, namely, minimum vertex cover and maximum clique. After describing relatively simple proof complexity upper bounds based on exhaustive backtracking algorithms in Section 7, we present our main technical contribution — resolution lower bounds — in Sections 8 to 10. Our hardness of approximation results are derived in Section 11. Finally, in Section 12 we prove a stronger lower bound that applies to exhaustive backtracking algorithms (as well as the DPLL procedure) and qualitatively matches our upper bounds for the same. Note that proofs in Sections 4 to 7 contain somewhat tedious details that the reader may want to skip initially.

2. Resolution and DPLL Proofs

A propositional formula F is said to be in conjunctive normal form (CNF) if it is a conjunction of *clauses*, where each clause is a disjunction of *literals* and each literal is either a variable or its negation. The construction of Tseitin (1968) can be used to efficiently convert, using additional variables, any propositional formula into CNF form. In this paper, we will only consider CNF formulas.

2.1. General Resolution. The resolution principle, originally proposed by Robinson (1965) for efficient automated deduction, forms the basis of many popular systems for practical theorem proving. Lower bounds on resolution proof sizes thus have a bearing on the running time of these algorithms.

To prove that a given input CNF formula F is unsatisfiable, we start with the original input clauses of F and repeatedly choose pairs of clauses to apply the *resolution rule*: given $(A \vee x)$ and $(B \vee \neg x)$, one can derive $(A \vee B)$. This rule has the property that a derived clause is satisfied by any assignment that satisfies both the parent clauses. Since we are interested in proving that F is unsatisfiable, the goal is to start with F and derive the empty (and trivially unsatisfiable) clause, FALSE.

Let ϕ be a set of clauses. A *resolution derivation* of clause C from ϕ is a sequence of clauses $\pi = C_1, C_2, \ldots, C_s = C$ where each clause C_i is either an element of ϕ or is derived by applying the resolution rule to two clauses C_j and C_k, $j, k < i$, occurring earlier in π. A resolution derivation of the empty clause, FALSE, from ϕ is called a *refutation* or *proof* of ϕ. Given any resolution refutation, we can associate with it in a natural way a directed acyclic graph where edges go from parent clauses to the clause obtained by resolving them on a certain literal. The special case where this graph is a tree is referred to as *tree-like* resolution and is discussed in Section 2.2.

Let F be a set of clauses encoding a given problem as a CNF formula and P a resolution proof of F. The *size* of P, $size(P)$, is the number of clauses appearing in P. The *resolution complexity* of F, $\mathsf{RES}(F)$, is the minimum of $size(P)$ over all proofs P of F; if no such proof exists, $\mathsf{RES}(F) = \infty$. The *width* of a clause is the number of literals occurring in it. The width of F, $w(F)$, and the width of P, $w(P)$, are the maximum of the widths of all clauses in F and P, respectively. The *refutation width* of F, $w(F \vdash \Lambda)$, is the minimum of $w(P)$ over all proofs P of F. As we shall see in Section 2.3, in order to prove a lower bound on $\mathsf{RES}(F)$, it is sufficient to prove a lower bound on $w(F \vdash \Lambda) - w(F)$.

2.2. DPLL Procedure and Tree-like Resolution. The Davis-Putnam-Logemann-Loveland or DPLL procedure (Davis *et al.* 1962; Davis & Putnam 1960) is both a proof system and a collection of algorithms for finding proofs. A simple DPLL algorithm to refute a CNF formula F is to repeatedly choose a variable x of F and recursively refute both $F|_{x \leftarrow 0}$ and $F|_{x \leftarrow 1}$. Variants of this algorithm form the most widely used family of complete algorithms for formula satisfiability. As a proof system, the DPLL procedure forms a special case of resolution where the proof graph is a tree, that is, any derived clause that is used more than once in the proof must be re-derived. The *tree-resolution* *complexity* of a given formula F is defined to be the minimum of $size(P)$, the size of P, over all *tree-like* resolution proofs P of F.

DPLL refutations can be thought of as trees where we branch at each node based on the value of a variable. DPLL refutations can be easily converted into tree-like resolution proofs of essentially the same size, and vice versa. Given this correspondence, we will use these two terms interchangeably and denote the tree-resolution complexity of a formula F by $\mathsf{DPLL}(F)$. This correspondence also shows that a lower bound on the size of tree-like resolution proofs implies a lower bound on DPLL refutation sizes and hence on the running time of DPLL-based algorithms. The transcript of the execution of any algorithm for finding a maximum independent set in a given graph is also a proof that no independent set of a larger size exists. Our results show that the running time of any such DPLL-based algorithm working on randomly chosen input graphs with not too many edges will almost certainly be exponential in the size of the graph.

Although we described DPLL algorithms here as working on propositional formulas, they capture a much more general class of algorithms that are based on branching and backtracking. For instance, basic algorithms for finding a maximum independent set, such as that of Tarjan (1972), branch on a vertex v by either including v in the current independent set and deleting it and all its neighbors from further consideration, or excluding v from the current independent set; they then proceed recursively, finding a maximum independent set in the remaining graph. More complicated algorithms, such as that of Tarjan & Trojanowski (1977), branch in a similar manner on small subsets of vertices, reusing subproblems already solved. Such algorithms also fall under the category of resolution based (not necessarily tree-like) algorithms and our lower bounds apply to them as well.

2.3. The Size-Width Relationship. Our proof uses the relationship between the size and the width of a resolution proof. Ben-Sasson & Wigderson (2001) showed that any short proof of unsatisfiability of a CNF formula can be converted to one of small width. Therefore, a lower bound on the width of all resolution proofs of a given formula implies a lower bound on the size of all such proofs.

Proposition 2.1 (Ben-Sasson & Wigderson 2001). *For any CNF formula F,* $\mathsf{DPLL}(F) \geq 2^{w(F \vdash \Lambda) - w(F)}$.

Proposition 2.2 (Ben-Sasson & Wigderson 2001). *For any CNF formula F over n variables and $c = 1/(9 \ln 2)$,* $\mathsf{RES}(F) \geq 2^{c(w(F \vdash \Lambda) - w(F))^2/n}$.

3. Independent Sets in Random Graphs

For any undirected graph $G = (V, E)$, let $n = |V|$ and $m = |E|$. A *k-independent set* in G is a set of k vertices no two of which share an edge. We will describe several natural ways of encoding in clausal form the statement that G has a k-independent set. Their refutations will be proofs that G does *not* contain any k-independent set. We will be interested in size bounds for such proofs.

Combinatorial properties of random graphs have been studied extensively (see, for instance, Bollobás 1985; Janson *et al.* 2000). We use the standard model $\mathbb{G}(n, p)$ for graphs where n vertices are fixed and each of the $\binom{n}{2}$ possible edges is included independently at random with probability $p \in [0, 1]$. $G \sim \mathbb{G}(n, p)$ denotes a graph G chosen at random from this distribution. We will state most of our results in terms of parameters n and Δ, where $\Delta \overset{\text{def}}{=} np$ is roughly the average degree of G (the exact average degree is $(n-1)p$).

We will need both worst case and almost certain bounds on the size of the largest independent set in graphs of density Δ.

PROPOSITION 3.1 (Turan's Theorem). *Every graph G with n vertices and average degree Δ has an independent set of size $\lfloor \frac{n}{\Delta+1} \rfloor$. In general, for any integer k satisfying $\Delta < \frac{n}{k-1} - 1$, G has an independent set of size k.*

For $\epsilon > 0$, let $k_{\pm\epsilon}$ be defined as follows:

$$k_{\pm\epsilon} = \lfloor \frac{2n}{\Delta}(\ln \Delta - \ln \ln \Delta + 1 - \ln 2 \pm \epsilon) \rfloor$$

PROPOSITION 3.2 (cf. Janson *et al.* 2000, Theorem 7.4). *For every $\epsilon > 0$ there is a constant C_ϵ such that the following holds. Let $\Delta = np$, $C_\epsilon \le \Delta \le n/\ln^2 n$, and $G \sim \mathbb{G}(n, p)$. With probability $1 - o(1)$ in n, the largest independent set in G is of size between $k_{-\epsilon}$ and $k_{+\epsilon}$.*

This shows that while random graphs are very likely to have an independent set of size $k_{-\epsilon}$, they are very unlikely to have one of size $k_{+\epsilon} + 1$. The *number* of independent sets of a certain size also shows a similar threshold behavior. While there are almost surely no independent sets of size $(2n/\Delta) \ln \Delta$, the following lemma, which follows by a straightforward extension of the analysis by Janson *et al.* (2000, Lemma 7.3), shows that there are exponentially many of size $(n/\Delta) \ln \Delta$. We use this bound later to put a limit on the best one can do with exhaustive backtracking algorithms that systematically consider all potential independent sets of a certain size.

Lemma 3.3. *There is a constant* $C > 0$ *such that the following holds. Let* $\Delta = np$, $\Delta \leq n/\ln^2 n$, *and* $G \sim \mathbb{G}(n,p)$. *With probability* $1 - o(1)$ *in* n, G *contains at least* $2^{C(n/\Delta)\ln^2\Delta}$ *independent sets of size* $\lfloor (n/\Delta)\ln\Delta \rfloor$.

Proof. Let X_k be a random variable whose value is the number of independent sets of size k in $G = (V, E)$. The expected value of X_k is given by:

$$\begin{aligned}
\mathbb{E}\left[X_k\right] &= \sum_{S \subseteq V, |S|=k} \Pr\left[S \text{ is an independent set in } G\right] \\
&= \binom{n}{k}(1-p)^{\binom{k}{2}} \\
&\geq \left(\frac{n}{k}\right)^k e^{-cpk^2} \qquad \text{for } c > 1/2,\ p = o(1) \text{ in } n, \text{ and large enough } n \\
&= \left(\frac{n}{k}\, e^{-c\Delta k/n}\right)^k
\end{aligned}$$

Let $c = 0.55$ and $C = 0.05/\ln 2$ so that $\Delta^{1-c}/\ln\Delta \geq 2^{C\ln\Delta}$. Setting $k = \lfloor (n/\Delta)\ln\Delta \rfloor$ and observing that $\left((n/k)e^{-c\Delta k/n}\right)^k$ decreases with k,

$$\begin{aligned}
\mathbb{E}\left[X_{\lfloor (n/\Delta)\ln\Delta \rfloor}\right] &\geq \left(\frac{\Delta}{\ln\Delta} e^{-c\ln\Delta}\right)^{(n/\Delta)\ln\Delta} \\
&\geq 2^{C(n/\Delta)\ln^2\Delta}.
\end{aligned}$$

We now use the standard second moment method to prove that X_k for $k = \lfloor (n/\Delta)\ln\Delta \rfloor$ asymptotically almost surely lies very close to its expected value. We begin by computing the expected value of X_k^2 and deduce from it that the variance of X_k is small.

$$\begin{aligned}
\mathbb{E}\left[X_k^2\right] &= \sum_{\substack{S \subseteq V \\ |S|=k}} \Pr\left[S \text{ is independent}\right] \sum_{i=0}^{k} \sum_{\substack{T \subseteq V \\ |T|=k \\ |S \cap T|=i}} \Pr\left[T \text{ is independent}\right] \\
&= \binom{n}{k}(1-p)^{\binom{k}{2}} \sum_{i=0}^{k} \binom{k}{i}\binom{n-k}{k-i}(1-p)^{\binom{k}{2}-\binom{i}{2}}
\end{aligned}$$

$$\begin{aligned}
\text{Therefore} \quad \frac{var\left[X_k\right]}{\mathbb{E}\left([X_k]\right)^2} &= \frac{\mathbb{E}\left[X_k^2\right]}{\left(\mathbb{E}\left[X_k\right]\right)^2} - 1 \\
&= \frac{\binom{n}{k}(1-p)^{\binom{k}{2}} \quad \sum_{i=0}^{k}\binom{k}{i}\binom{n-k}{k-i}(1-p)^{\binom{k}{2}-\binom{i}{2}}}{\left[\binom{n}{k}(1-p)^{\binom{k}{2}}\right]^2} - 1
\end{aligned}$$

This last expression has been studied by Janson *et al.* (2000, page 181, equation 7.8). Following their calculation (Janson *et al.* 2000, Lemma 7.3), we obtain that $var[X_k]/(\mathbb{E}[X_k^2])^2 \to 0$ for $k = \lfloor (n/\Delta) \ln \Delta \rfloor$ as $n \to \infty$, when $\Delta \geq \sqrt{n} \ln^2 n$. When $\Delta \leq \sqrt{n} \ln^2 n$, another argument along their lines (Janson *et al.* 2000, Theorem 7.4) provides the same result. Applying the second moment method, this leads to the desired bound. □

4. Encoding Independent Sets as Formulas

In order to use a propositional proof system to prove that a graph does not have an independent set of a particular size, we first need to formulate the problem as a propositional formula. This is complicated by the difficulty of counting set sizes using CNF formulas.

One natural way to encode the independent set problem is to have indicator variables that say which vertices are in the independent set and auxiliary variables that count the number of vertices in the independent set. This encoding is discussed in Section 4.1. The clauses in this encoding, although capturing the simple concept of counting, are somewhat involved. Moreover, the existence of two different types of variables makes this encoding difficult to reason about directly.

A second encoding, derived from the counting-based encoding, is described in Section 4.2. It is based on a mapping from the vertices of the graph to k additional nodes as an alternative to straightforward counting, and uses variables of only one type. This is essentially the same encoding as the one used by Bonet, Pitassi & Raz (1997) for the clique problem, except that in our case we need to add an extra set of clauses, called ordering clauses, to make the lower bounds non-trivial. (Otherwise, lower bounds trivially follow from known lower bounds for the pigeonhole principle (Haken 1985) which have nothing to do with the independent set problem; in the work of Bonet *et al.* (1997) this problem did not arise because the proof system considered was cutting planes where, as shown by Cook *et al.* (1987), the pigeonhole principle has short proofs.)

Section 4.3 finally describes a much simpler encoding which is the one we analyze directly for our lower bounds. This encoding considers only a restricted class of independent sets that we call *block-respecting independent sets*, for which the problem of counting the set size is trivial. Hence, the encoding uses only one type of variable that indicates whether or not a given vertex is in the independent set. Refutation of this third encoding rules out the existence of the smaller class of block-respecting independent sets only. Intuitively, this should be easier to do than ruling out all possible independent sets. In fact, we show that the resolution and DPLL refutations of this encoding are bounded above in size by those of the mapping encoding and are at worst a small amount larger than those of the counting encoding. As a result, we can translate our lower bounds for this third encoding to each of the other encodings. Further, we give upper bounds for the two general encodings which also apply to the simpler block-respecting independent set encoding.

For the rest of this paper, identify the vertex set of the input graph with $\{1, 2, \dots, n\}$. Each encoding will be defined over variables from one or more of the following three categories:

- ○ $x_v, 1 \leq v \leq n$, which is TRUE iff vertex v is chosen by the truth assignment to be in the independent set,
- ○ $y_{v,i}, 0 \leq i \leq v \leq n$, $0 \leq i \leq k$, which is TRUE iff precisely i of the first v vertices are chosen in the independent set, and
- ○ $z_{v,i}, 1 \leq v \leq n, 1 \leq i \leq k$, which is TRUE iff vertex v is chosen as the i^{th} node of the independent set.

A desirable property of all independent set encodings is their *monotonicity*, i.e., for $k' > k$, proving the non-existence of an independent set of size k' in that encoding must not be any harder than doing so for size k, up to a polynomial factor. This property indeed holds for each of the three encodings we consider below.

4.1. Encoding Based on Counting. The *counting encoding*, $\alpha_{count}(G, k)$, of the independent set problem is defined over variables x_v and $y_{v,i}$. As mentioned previously, this encoding is somewhat tedious in nature. It has the following three kinds of clauses:

(a) Edge Clauses: For each edge (u, v), $\alpha_{count}(G, k)$ has one clause saying that at most one of u and v is selected; $\forall (u, v) \in E, u < v : (\neg x_u \vee \neg x_v) \in \alpha_{count}(G, k)$

(b) Size-k Clause: There is a clause saying that the independent set chosen is of size k; $y_{n,k} \in \alpha_{count}(G, k)$

(c) Counting Clauses: There are clauses saying that variables $y_{v,i}$ correctly count the number of vertices chosen. For simplicity, we first write this condition not as a set of clauses but as more general propositional formulas. For the base case, $\alpha_{count}(G, k)$ contains $y_{0,0}$ and the clausal form of $(y_{v,0} \leftrightarrow (y_{v-1,0} \wedge \neg x_v))$ for $v \in \{1, \dots n\}$. Further, $\forall i, v, 1 \le i \le v \le n$, $1 \le i \le k$, $\alpha_{count}(G, k)$ contains the clausal form of $(y_{v,i} \leftrightarrow ((y_{v-1,i} \wedge \neg x_v) \vee (y_{v-1,i-1} \wedge x_v)))$, unless $i = v$, in which case $\alpha_{count}(G, k)$ contains the clausal form of the simplified formula $(y_{v,v} \leftrightarrow (y_{v-1,v-1} \wedge x_v))$.

Translated into clauses, these conditions take the following form. Formulas defining $y_{v,0}$ for $v \ge 1$ translate into $\{(\neg y_{v,0} \vee y_{v-1,0}), (\neg y_{v,0} \vee \neg x_v), (y_{v,0} \vee \neg y_{v-1,0} \vee x_v)\}$. Further, formulas defining $y_{v,i}$ for $v > i \ge 1$ translate into $\{(y_{v,i} \vee \neg y_{v-1,i} \vee x_v), (y_{v,i} \vee \neg y_{v-1,i-1} \vee \neg x_v), (\neg y_{v,i} \vee y_{v-1,i} \vee y_{v-1,i-1}), (\neg y_{v,i} \vee y_{v-1,i} \vee x_v), (\neg y_{v,i} \vee y_{v-1,i-1} \vee \neg x_v)\}$, whereas in the case $i = v$ they translate into $\{(\neg y_{v,v} \vee y_{v-1,v-1}), (\neg y_{v,v} \vee x_v), (\neg x_v \vee \neg y_{v-1,v-1} \vee y_{v,v})\}$.

Lemma 4.1. *For any graph G over n vertices and $k' > k$,*

$$\begin{aligned} \mathsf{RES}(\alpha_{count}(G, k')) &< n \cdot \mathsf{RES}(\alpha_{count}(G, k)) + 2n^2 \quad \textit{and} \\ \mathsf{DPLL}(\alpha_{count}(G, k')) &< n \cdot \mathsf{DPLL}(\alpha_{count}(G, k)) + 2n^2. \end{aligned}$$

Proof. If G contains an independent set of size k, then there are no resolution refutations of $\alpha_{count}(G, k)$. By our convention, $\mathsf{RES}(\alpha_{count}(G, k)) = \mathsf{DPLL}(\alpha_{count}(G, k)) = \infty$, and the result holds. Otherwise consider a refutation π of $\alpha_{count}(G, k)$. Using π, we construct a refutation π' of $\alpha_{count}(G, k')$ such that $size(\pi') \le (n - k + 1) \cdot size(\pi) + 2(k' - k)(n - k)$, which is less than $n \cdot size(\pi) + 2n^2$. Further, if π is a tree-like refutation, then so is π'.

$\alpha_{count}(G, k')$ contains all clauses of $\alpha_{count}(G, k)$ except the size-k clause, $y_{n,k}$. Therefore, starting with $\alpha_{count}(G, k')$ as initial clauses and using π modified not to use the clause $y_{n,k}$, we derive a subclause of $\neg y_{n,k}$. This clause, however, cannot be a strict subclause of $\neg y_{n,k}$ because $\alpha_{count}(G, k) \setminus \{y_{n,k}\}$ is satisfiable. Hence, we must obtain $\neg y_{n,k}$. Call this derivation D_n. By construction, $size(D_n) \le size(\pi)$. Making a copy of D_n, we restrict it by setting $x_n \leftarrow \text{FALSE}, y_{n,k} \leftarrow y_{n-1,k}$ to obtain a derivation D_{n-1} of $\neg y_{n-1,k}$. Continuing this process, construct derivations D_p of $\neg y_{p,k}$ for $p \in \{n-1, n-2, \dots, k\}$ by further setting $x_{p+1} \leftarrow \text{FALSE}, y_{p+1,k} \leftarrow y_{p,k}$. Again, by construction, $size(D_p) \le size(\pi)$. Combining derivations $D_n, D_{n-1}, \dots, D_k$ into π' gives a derivation of size at most $(n - k + 1) \cdot size(\pi)$ of clauses $\neg y_{p,k}, k \le p \le n$, which is tree-like if π is.

Continuing to construct π', resolve the above derived clause $\neg y_{k,k}$ with the counting clause $(\neg y_{k+1,k+1} \vee y_{k,k})$ of $\alpha_{count}(G, k')$ to obtain $\neg y_{k+1,k+1}$. Now for v going from $k+2$ to n, resolve the already derived clauses $\neg y_{v-1,k+1}$ and $\neg y_{v-1,k}$

with the counting clause $(\neg y_{v,k+1} \vee y_{v-1,k+1} \vee y_{v-1,k})$ of $\alpha_{count}(G, k')$ to obtain $\neg y_{v,k+1}$. This gives a tree-like derivation of size less than $2(n-k)$ of clauses $\neg y_{p,k+1}, k+1 \leq p \leq n$, starting from clauses $\neg y_{q,k}, k \leq q \leq n$. Repeating this process $(k'-k)$ times gives a tree-like derivation of size less than $2(k'-k)(n-k)$ of clauses $\neg y_{p,k'}, k' \leq p \leq n$, starting from clauses $\neg y_{q,k}, k \leq q \leq n$, derived previously. In particular, $\neg y_{n,k'}$ is now a derived clause. Resolving it with the size-k' clause $y_{n,k'}$ of $\alpha_{count}(G, k')$ completes refutation π'. □

4.2. Encoding Based on Mapping. This encoding, denoted $\alpha_{map}(G, k)$, uses a mapping from n vertices of G to k nodes of the independent set as an indirect way of counting the number of vertices chosen by a truth assignment to be in the independent set. It can be viewed as a set of constraints restricting the mapping (see Figure 4.1). The idea is to map the nodes of the independent set to the sequence $(1, 2, \ldots, k)$ in the increasing order of their index as vertices in the graph. This encoding is defined over variables $z_{v,i}$ and has the following five kinds of clauses:

(a) Edge Clauses: For each edge (u, v), there are clauses saying that at most one of u and v is chosen in the independent set; $\forall (u, v) \in E, i, j, 1 \leq i < j \leq k : \ (\neg z_{u,i} \vee \neg z_{v,j}) \in \alpha_{map}(G, k)$

(b) Surjective Clauses: For each node i, there is a clause saying that some vertex is chosen as the i^{th} node of the independent set; $\forall i, 1 \leq i \leq k :$ $(z_{1,i} \vee z_{2,i} \vee \ldots \vee z_{n,i}) \in \alpha_{map}(G, k)$

(c) Function Clauses: For each vertex v, there are clauses saying that v is not mapped to two nodes, i.e., it is not counted twice in the independent set; $\forall v, i, j, 1 \leq v \leq n, 1 \leq i < j \leq k : \ (\neg z_{v,i} \vee \neg z_{v,j}) \in \alpha_{map}(G, k)$

(d) 1-1 Clauses: For each node i, there are clauses saying that no two vertices map to the i^{th} node of the independent set; $\forall i, u, v, 1 \leq i \leq k, 1 \leq u < v \leq n : \ (\neg z_{u,i} \vee \neg z_{v,i}) \in \alpha_{map}(G, k)$

(e) Ordering Clauses: For every pair of consecutive nodes, there are clauses saying that vertices are not mapped to these in the reverse order. This, by transitivity, implies that there is a unique mapping to k nodes once we have chosen k vertices to be in the independent set. $\forall u, v, i, 1 \leq u < v \leq n, 1 \leq i < k : \ (\neg z_{u,i+1} \vee \neg z_{v,i}) \in \alpha_{map}(G, k)$.

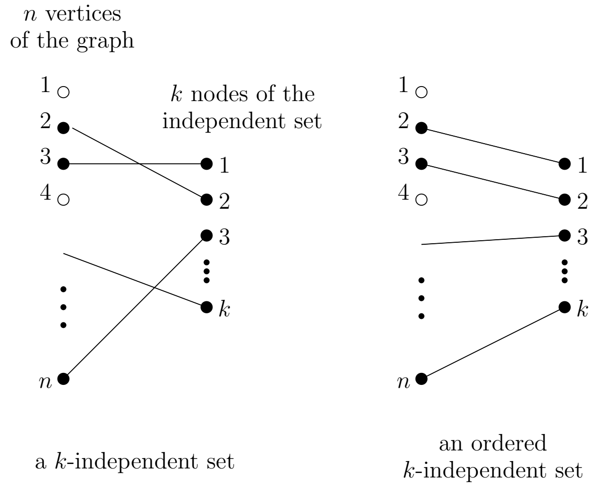
n vertices of the graph
1
2
3
4
n
k nodes of the independent set
1
2
3
k
a k-independent set
1
2
3
4
n
1
2
3
k
an ordered k-independent set

Figure 4.1: Viewing independent sets as a mapping from n vertices to k nodes

LEMMA 4.2. *For any graph G and $k' \geq k$,*

$$\begin{aligned} \mathsf{RES}(\alpha_{map}(G,k')) &\leq \mathsf{RES}(\alpha_{map}(G,k)) \quad \textit{and} \\ \mathsf{DPLL}(\alpha_{map}(G,k')) &\leq \mathsf{DPLL}(\alpha_{map}(G,k)). \end{aligned}$$

PROOF. If G contains an independent set of size k, then there are no resolution refutations of $\alpha_{map}(G,k)$. By our convention, $\mathsf{RES}(\alpha_{map}(G,k)) = \mathsf{DPLL}(\alpha_{map}(G,k)) = \infty$, and the result holds. Otherwise consider a refutation π of $\alpha_{map}(G,k)$. Observe that all clauses of $\alpha_{map}(G,k)$ are also clauses of $\alpha_{map}(G,k')$. Hence π is also a refutation of $\alpha_{map}(G,k')$, proving the desired bounds. □

4.3. Encoding Using Block-respecting Independent Sets. Fix $b = n/k$ for the rest of the paper and assume for simplicity that k divides n (denoted $k \mid n$). Partition the vertices of G into k subsets, called *blocks*, of size b each, so that block i, $1 \leq i \leq k$, consists of vertices $\{b(i-1)+1, \ldots, b(i-1)+b\}$.[1] A *block-respecting independent set* of size k in G is an independent set in G with precisely one vertex in each of these k blocks. Note that the size of a block-respecting independent set is implicit in the partitioning of the vertices into blocks. Clearly, if a graph does not contain *any* k-independent set, then it certainly does not contain any block-respecting independent set of size k either.

[1] The restriction $k \mid n$ is only to make the presentation cleaner. We can extend our arguments to all $k < n$ by letting each block have either b or $b+1$ vertices for $b = \lfloor n/k \rfloor$. The calculations are nearly identical to what we present here.

In this sense, proving the non-existence of block-respecting independent sets is, in principle, easier than proving the non-existence of (usual) independent sets. We will formalize this shortly.

We now define a CNF formula $\alpha_{block}(G, k)$ over variables x_v which asserts that G contains a block-respecting independent set of size k. The encoding $\alpha_{block}(G, k)$ contains the following three kinds of clauses:

(a) Edge Clauses: For each edge (u, v), there is one clause saying that not both u and v are selected; $\forall (u, v) \in E, u < v : \; (\neg x_u \vee \neg x_v) \in \alpha_{block}(G, k)$

(b) Block Clauses: For each block, there is one clause saying that at least one of the vertices in it is selected; $\forall\, i, 0 \le i < k : \; (x_{bi+1} \vee x_{bi+2} \vee \ldots \vee x_{bi+b}) \in \alpha_{block}(G, k)$

(c) 1-1 Clauses: For each block, there are clauses saying that at most one of the vertices in it is selected; $\forall\, i, p, q, 0 \le i < k, 1 \le p < q \le b : (\neg x_{bi+p} \vee \neg x_{bi+q}) \in \alpha_{block}(G, k)$

$\alpha_{block}(G, k)$ is satisfiable iff G has a block-respecting independent set of size k (under the fixed order of vertices and the induced partition into k blocks). Note that there is no exact analog of Lemmas 4.1 and 4.2 for the block encoding. In fact, the non-existence of a block-respecting independent set of size k doesn't even logically imply the non-existence of one of size k' for all $k' > k$. This monotonicity, however, does hold when $k \mid k'$.

Lemma 4.3. *For any graph G, $k' \ge k$, $k \mid k'$, and $k' \mid n$,*

$$\begin{aligned} \mathsf{RES}(\alpha_{block}(G, k')) &\le \mathsf{RES}(\alpha_{block}(G, k)) \quad \textit{and} \\ \mathsf{DPLL}(\alpha_{block}(G, k')) &\le \mathsf{DPLL}(\alpha_{block}(G, k)). \end{aligned}$$

The result holds even when the 1-1 clauses are omitted from both encodings.

Proof. If G contains a block-respecting independent set of size k, then there is no resolution refutation of $\alpha_{block}(G, k)$. By our convention, $\mathsf{RES}(\alpha_{block}(G, k)) = \mathsf{DPLL}(\alpha_{block}(G, k)) = \infty$, and the result holds. Otherwise consider a refutation π of $\alpha_{block}(G, k)$. The two encodings, $\alpha_{block}(G, k)$ and $\alpha_{block}(G, k')$, are defined over the same set of variables and have identical edge clauses. We will apply a transformation σ to the variables so that the block and the 1-1 clauses of $\alpha_{block}(G, k)$ become a subset of the block and the 1-1 clauses of $\alpha_{block}(G, k')$, respectively.

σ works as follows. Each block of vertices in $\alpha_{block}(G,k)$ consists exactly of k'/k blocks of vertices in $\alpha_{block}(G,k')$ because $k \mid k'$. σ sets all but the first n/k' vertices of each block of $\alpha_{block}(G,k)$ to FALSE. This shrinks all block clauses of $\alpha_{block}(G,k)$ to block clauses of $\alpha_{block}(G,k')$. Further, it trivially satisfies all 1-1 clauses of $\alpha_{block}(G,k)$ that are not 1-1 clauses of $\alpha_{block}(G,k')$. Hence $\pi|_\sigma$ is a refutation of $\alpha_{block}(G,k')$ which in fact uses only a subset of the original block and 1-1 clauses of the formula. □

4.4. Relationships Among Encodings. For reasonable bounds on the block size, resolution refutations of the block encoding are essentially at least as efficient as those of the other two encodings. We state the precise relationship in the following lemmas.

LEMMA 4.4. *For any graph G over n vertices, $k \mid n$, and $b = n/k$,*

$$\begin{aligned} \mathsf{RES}(\alpha_{block}(G,k)) &\leq b^2 \cdot \mathsf{RES}(\alpha_{count}(G,k)) \quad \textit{and} \\ \mathsf{DPLL}(\alpha_{block}(G,k)) &\leq (2 \cdot \mathsf{DPLL}(\alpha_{count}(G,k)))^{\log_2 2b}. \end{aligned}$$

PROOF. Fix a resolution proof π of $\alpha_{count}(G,k)$. We describe a transformation ρ on the underlying variables such that for each initial clause $C \in \alpha_{count}(G,k)$, $C|_\rho$ is either TRUE or an initial clause of $\alpha_{block}(G,k)$. This lets us generate a resolution proof of $\alpha_{block}(G,k)$ from $\pi|_\rho$ of size not much larger than $size(\pi)$. ρ is defined as follows: for each $i \in \{0,1,\ldots,k\}$, set $y_{bi,i} =$ TRUE and $y_{bi,j} =$ FALSE for $j \neq i$; set all $y_{v,i} =$ FALSE if vertex v does not belong to either block $i+1$ or block i; finally, for $1 \leq j \leq b$, replace all occurrences of $y_{bi+j,i+1}$ and $\neg y_{bi+j,i}$ with $(x_{bi+1} \vee x_{bi+2} \vee \ldots \vee x_{bi+j})$, and all occurrences of $\neg y_{bi+j,i+1}$ and $y_{bi+j,i}$ with $(x_{bi+j+1} \vee x_{bi+j+2} \vee \ldots \vee x_{bi+b})$. Note that setting $y_{bi,i} =$ TRUE for each i logically implies the rest of the transformations stated above.

We first prove that ρ transforms the initial clauses of $\alpha_{count}(G,k)$ as claimed. The edge clauses are the same in both encodings. The size-k clause $y_{n,k}$ and the counting clause $y_{0,0}$ of $\alpha_{count}(G,k)$ transform to TRUE. The following can also be easily verified by plugging in the substitutions for the y variables. The counting clauses that define $y_{v,0}$ for $v \geq 1$ are either satisfied or translate into the first block clause $(x_1 \vee \ldots \vee x_b)$. Further, the counting clauses that define $y_{v,i}$ for $v \geq 1, i \geq 1$ are either satisfied or transform into the i^{th} or the $(i+1)^{st}$ block clause, i.e., into $(x_{b(i-1)+1} \vee \ldots \vee x_{b(i-1)+b})$ or $(x_{bi+1} \vee \ldots \vee x_{bi+b})$. Hence, all initial clauses of $\alpha_{count}(G,k)$ are either satisfied or transform into initial clauses of $\alpha_{block}(G,k)$.

We now describe how to generate a valid resolution proof of $\alpha_{block}(G,k)$ from this transformation. Note that the substitutions for $y_{bi+j,i+1}$ and $y_{bi+j,i}$ replace these variables by a disjunction of (at most b) positive literals. Any resolution step performed on such a y in the original proof must now be converted into a set of equivalent resolution steps, which will lengthen the transformed refutation. More specifically, a step resolving clauses $(y \vee A)$ and $(\neg y \vee B)$ on the literal y (where y is either $y_{bi+j,i+1}$ or $y_{bi+j,i}$) will now be replaced by a set of resolution steps deriving $(A' \vee B')$ from clauses $(x_{u_1} \vee \ldots \vee x_{u_p} \vee A')$, $(x_{v_1} \vee \ldots \vee x_{v_q} \vee B')$, and initial clauses of $\alpha_{block}(G,k)$, where all x variables mentioned belong to the same block of G, $\{u_1, \ldots, u_p\}$ is disjoint from $\{v_1, \ldots, v_q\}$, $p + q = b$, and A' and B' correspond to the translated versions of A and B, respectively.

The obvious way of doing this is to resolve the clause $(x_{u_1} \vee \ldots \vee x_{u_p} \vee A')$ with all 1-1 clauses $(\neg x_{u_i} \vee \neg x_{v_1})$ obtaining $(\neg x_{v_1} \vee A')$. Repeating this for all $x_{v_j}, 2 \leq j \leq q$, gives us clauses $(\neg x_{v_j} \vee A')$. Note that this reuses $(x_{u_1} \vee \ldots \vee x_{u_p} \vee A')$ q times and is therefore not tree-like. Resolving all $(\neg x_{v_j} \vee A')$ in turn with $(x_{v_1} \vee \ldots \vee x_{v_q} \vee B')$ gives us $(A' \vee B')$. This takes $pq + q < b^2$ steps. Hence the blow-up in size for general resolution is at most a factor of b^2. Note that this procedure is symmetric in A' and B'; we could also have chosen the clause $(\neg y \vee B)$ to start with, in which case we would need $qp + p < b^2$ steps.

The tree-like case is somewhat trickier because we need to replicate clauses that are reused by the above procedure. We handle this using an idea similar to the one used by Clegg *et al.* (1996) for deriving the size-width relationship for tree-like resolution proofs. Let $newSize(s)$ denote the maximum over the sizes of all transformed tree-like proofs obtained from original tree-like proofs of size s by applying the above procedure and creating enough duplicates to take care of reuse. We prove by induction that $newSize(s) \leq (2s)^{\log_2 2b}$. For the base case, $newSize(1) = 1 \leq 2b = 2^{\log_2 2b}$. For the inductive step, consider the subtree of the original proof that derives $(A \vee B)$ by resolving $(y \vee A)$ and $(\neg y \vee B)$ on the literal y as above. Let this subtree be of size $s \geq 2$ and assume without loss of generality that the subtree deriving $(y \vee A)$ is of size $s_A \leq s/2$. By induction, the transformed version of this subtree deriving $(x_{u_1} \vee \ldots \vee x_{u_p} \vee A')$ is of size at most $newSize(s_A)$ and that of the other subtree deriving $(x_{v_1} \vee \ldots \vee x_{v_q} \vee B')$ is of size at most $newSize(s - s_A - 1)$. Choose $(x_{u_1} \vee \ldots x_{u_p} \vee A')$ as the clause to start the new derivation of $(A' \vee B')$ as described in the previous paragraph. The size of this refutation is at most $b \cdot newSize(s_A) + newSize(s - s_A - 1) + b^2$. Since this can be done for any original proof of size s, $newSize(s) \leq b \cdot newSize(s_A) + newSize(s - s_A - 1) + b^2$ for $s \geq 2$ and $s_A \leq s/2$. It can be easily verified that $newSize(s) = 2bsb^{\log_2 s} = (2s)^{\log_2 2b}$ is a solution to this. This proves the bound for the DPLL case. □

Lemma 4.5. *For any graph G over n vertices and $k \mid n$,*

$$\begin{aligned} \mathsf{RES}(\alpha_{block}(G,k)) &\leq \mathsf{RES}(\alpha_{map}(G,k)) \quad \textit{and} \\ \mathsf{DPLL}(\alpha_{block}(G,k)) &\leq \mathsf{DPLL}(\alpha_{map}(G,k)). \end{aligned}$$

Proof. In the general encoding $\alpha_{map}(G,k)$, a vertex v can potentially be chosen as the i^{th} node of the k-independent set for *any* $i \in \{1,2,\ldots,k\}$. In the restricted encoding, however, vertex v belonging to block j can be thought of as either being selected as the j^{th} node of the independent set or not being selected at all. Hence, if we start with a resolution (or DPLL) refutation of $\alpha_{map}(G,k)$ and set $z_{v,i} =$ FALSE for $i \neq j$, we get a simplified refutation where all variables are of the form $z_{v,j}$, with vertex v belonging to block j. Renaming each such $z_{v,j}$ to x_v, we get a refutation in the variables of $\alpha_{block}(G,k)$ that is no larger in size than the original refutation of $\alpha_{map}(G,k)$.

All we now need to do is verify that for every initial clause of $\alpha_{map}(G,k)$, this transformation either converts it into an initial clause of $\alpha_{block}(G,k)$ or satisfies it. The transformed refutation will then be a refutation of $\alpha_{block}(G,k)$ itself. This reasoning is straightforward:

(a) Edge clauses $(\neg z_{u,i} \vee \neg z_{v,j})$ of $\alpha_{map}(G,k)$ that represented edge $(u,v) \in E$ with u in block i and v in block j transform into the corresponding edge clause $(\neg x_u \vee \neg x_v)$ of $\alpha_{block}(G,k)$. If vertex u (or v) is not in block i (or j, resp.), then the transformation sets $z_{u,i}$ (or $z_{v,j}$, resp.) to FALSE and the clause is trivially satisfied.

(b) Surjective clauses of $\alpha_{map}(G,k)$ clearly transform into the corresponding block clauses of $\alpha_{block}(G,k)$ – for the i^{th} such clause, variables corresponding to vertices that do not belong to block i are set to FALSE and simply vanish, and we are left with the i^{th} block clause of $\alpha_{block}(G,k)$.

(c) It is easy to see that all function and ordering clauses are trivially satisfied by the transformation.

(d) 1-1 clauses $(\neg z_{u,i} \vee \neg z_{v,i})$ of $\alpha_{map}(G,k)$ that involved vertices u and v both from block i transform into the corresponding 1-1 clause $(\neg x_u \vee \neg x_v)$ of $\alpha_{block}(G,k)$. If vertex u (or v) is not in block i, then the transformation sets $z_{u,i}$ (or $z_{v,i}$, resp.) to FALSE and the clause is trivially satisfied.

Thus, this transformed proof is a refutation of $\alpha_{block}(G,k)$ and the desired bounds follow. □

5. Simulating Chvátal's Proof System

In this section, we show that resolution on $\alpha_{block}(G, k)$ can efficiently simulate Chvátal's proofs (Chvátal 1977) of non-existence of k-independent sets in G. This indirectly provides bounds on the running time of various algorithms for finding a maximum independent set in a given graph. We begin with a brief description of Chvátal's proof system. Let (S, t) for $t \geq 1$ be the *statement* that the subgraph of G induced by a vertex subset S does not have an independent set of size t. $(\phi, 1)$ is given as an axiom and the goal is to derive, using a series of applications of one of two rules, the statement (V, k), where V is the vertex set of G and k is given as input. The two inference rules are –

Branching Rule: for any vertex $v \in S$, from statements $(S \setminus N(v), t-1)$ and $(S \setminus \{v\}, t)$ one can infer (S, t), where $N(v)$ is the set containing v and all its neighbors in G;

Monotone Rule: from statement (S, t) one can infer any (S', t') that (S, t) *dominates*, i.e., $S \supseteq S'$ and $t \leq t'$.

For a graph G with vertex set $V(G)$, let $Chv(G, k)$ denote the size of the smallest proof in Chvátal's system of the statement $(V(G), k)$. Following our convention, $Chv(G, k) = \infty$ if no such proof exists. As an immediate application of the monotone rule, we have:

Proposition 5.1. *For $k' > k$, $Chv(G, k') \leq Chv(G, k) + 1$.*

Proposition 5.2. *Let G and G' be graphs with $V(G) = V(G')$ and $E(G) \subseteq E(G')$. For any k, $Chv(G', k) \leq 2 \cdot Chv(G, k)$ and the number of applications of the branching rule in the two shortest proofs is the same.*

Proof. Let π be a proof of $(V(G), k)$ in G. We convert π into a proof π' of $(V(G'), k)$ in G' by translating proof statements in the order in which they appear in π. The axiom statement translates directly without any change. For the derived statements, any application of a monotone inference can be applied equally for both graphs. For an application of the branching rule in π, some (S, t) is derived from $(S \setminus N(v), t-1)$ and $(S \setminus \{v\}, t)$. To derive (S, t) for G', the only difference is the replacement of $(S \setminus N(v), t-1)$ by $(S \setminus N'(v), t-1)$, where $N'(v)$ is the set containing v and all its neighbors in G'. If these two statements are different then since $N'(v) \supseteq N(v)$, the latter follows from the former by a single application of the monotone rule. In total, at most $size(\pi)$ additional inferences are added, implying $size(\pi') \leq 2size(\pi)$. □

The following lemma shows that by traversing the proof graph beginning with the axioms one can locally replace each inference in Chvátal's system by a small number of resolution inferences.

Lemma 5.3. *For any graph G over n vertices and $k \mid n$,*

$$\mathsf{RES}(\alpha_{block}(G,k)) \;\leq\; 4n \cdot Chv(G,k).$$

Proof. Let V denote the vertex set of G. Partition V into k consecutive blocks of size n/k each. Let G_{block} be the graph obtained by adding to G all edges (u,v) such that vertices u and v belong to the same block of G. In other words, G_{block} is G modified to contain a clique on each block, so that every independent set of size k in G_{block} is block-respecting with respect to G. By Proposition 5.2, the shortest proof in Chvátal's system, say π_{Chv}, of (V,k) in G_{block} is at most twice in size as the shortest proof of (V,k) in G. We will use π_{Chv} to guide the construction of a resolution refutation π_{RES} of $\alpha_{block}(G,k)$ such that $size(\pi_{RES}) \leq 2n \cdot size(\pi_{Chv})$, proving the desired bound.

Observe that without loss of generality, for any statement (S,t) in π_{Chv}, t is at least the number of blocks of G containing vertices in S. This is so because it is true for the final statement (V,k), and if it is true for (S,t), then it is also true for both $(S \setminus \{v\}, t)$ and $(S \setminus N(v), t-1)$ from which (S,t) is derived. Call (S,t) a *trivial* statement if t is strictly bigger than the number of blocks of G containing vertices in S. The initial statement $(\phi, 1)$ of the proof is trivial, whereas the final statement (V,k) is not. Furthermore, all statements derived by applying the monotone rule are trivial.

π_{RES} will have a clause associated with each non-trivial statement (S,t) occurring in π_{Chv}. This clause will be a subclause of the clause $C_S \stackrel{\text{def}}{=} (\bigvee_{u \in N_S} x_u)$, where N_S is the set of all vertices in $V \setminus S$ that are in blocks of G containing at least one vertex of S. π_{RES} will be constructed inductively, using the non-trivial statements of π_{Chv}. Note that the clause associated in this manner with (V,k) will be the empty clause, making π_{RES} a refutation.

Suppose (S,t) is non-trivial and is derived in π_{Chv} by applying the branching rule to vertex $v \in S$. Write the target clause C_S as $(C_S^b \vee C_S^r)$, where C_S^b is the disjunction of all variables corresponding to vertices of N_S that are in the same block as v, and C_S^r is the disjunction of all variables corresponding to vertices of N_S that are in the remaining blocks. Before deriving the desired subclause of C_S, derive two clauses Cl_1 and Cl_2 as follows depending on the properties of the inference that produced (S,t):

Case 1: Both $(S \setminus \{v\}, t)$ and $(S \setminus N(v), t-1)$ are trivial. It is easy to see that since (S,t) is non-trivial, if $(S \setminus \{v\}, t)$ is trivial then v is the only vertex of S in its block. Let Cl_1 be the initial block clause for the block containing v, which is precisely $(x_v \vee C_S^b)$. The fact that $(S \setminus N(v), t-1)$ is also trivial implies that the neighbors of v include not only every vertex of S appearing in the block containing v but also all vertices in $S \cap B$, where B is some other block that does not contain v. Resolving the block clause for block B with all edge clauses $(\neg x_v \vee \neg x_u)$ for $u \in S \cap B$ gives a subclause Cl_2 of $(\neg x_v \vee C_S^r)$.

Case 2: $(S \setminus \{v\}, t)$ is trivial but $(S \setminus N(v), t-1)$ is non-trivial. Set Cl_1 exactly as in case 1. Given that $(S \setminus N(v), t-1)$ is non-trivial, by the inductive assumption the prefix of π_{RES} constructed so far contains a subclause of $C_{S\setminus N(v)}$. Since the given proof applies to G_{block}, $N(v) \cup v$ contains every vertex in the block containing v as well as all neighbors of v in G that are not in v's block. Therefore, the subclause of $C_{S\setminus N(v)}$ we have by induction is a subclause of $(C_S^r \vee x_{u_1} \vee \ldots \vee x_{u_p})$, where each u_i is a neighbor of v in S in blocks other than v's block. Derive a new clause Cl_2 by resolving this clause with all edge clauses $(\neg x_v \vee \neg x_{u_i})$. Observe that Cl_2 is a subclause of $(\neg x_v \vee C_S^r)$.

Case 3: $(S \setminus \{v\}, t)$ is non-trivial but $(S \setminus N(v), t-1)$ is trivial. Set Cl_2 as in case 1. Since $(S \setminus \{v\}, t)$ is non-trivial, by the inductive assumption the prefix of π_{RES} constructed so far contains a subclause Cl_1 of $C_{S\setminus\{v\}}$, i.e., a subclause of $(x_v \vee C_S)$.

Case 4: Both $(S \setminus \{v\}, t)$ and $(S \setminus N(v), t-1)$ are non-trivial. In this case, derive Cl_1 as in case 3 and Cl_2 as in case 2.

It is easy to verify that Cl_1 is a subclause of $(x_v \vee C_S)$ and Cl_2 is a subclause of $(\neg x_v \vee C_S^r)$. If either Cl_1 or Cl_2 does not mention x_v at all, then we already have the desired subclause of C_S. Otherwise resolve Cl_1 with Cl_2 to get a subclause of C_S. This completes the construction. Given any non-trivial statement in π_{Chv}, it takes at most $2n$ steps to derive the subclause associated with it in the resolution proof, given that we have already derived the corresponding subclauses for the two branches of that statement. Hence, $size(\pi_{RES}) \leq 2n \cdot size(\pi_{Chv})$. □

It follows that lower bounds on the complexity of α_{block} apply to Chvátal's system and hence also to many algorithms for finding a maximum independent set in a given graph that are captured by his proof system, such as those of Tarjan (1972), Tarjan & Trojanowski (1977), Jian (1986), and Shindo & Tomita (1990).

6. Relation to Vertex Cover and Coloring

This section discusses how the independent set problem relates to vertex covers and colorings of random graphs in terms of resolution complexity.

6.1. Vertex Cover. As for independent sets, for any undirected graph $G = (V, E)$, let $n = |V|$, $m = |E|$, and $\Delta = m/n$. A *t-vertex cover* in G is a set of t vertices that contains at least one endpoint of every edge in G. I is an independent set in G if and only if $V \setminus I$ is a vertex cover of G. Hence, the problem of determining whether or not G has a t-vertex cover is the same as that of determining whether or not it has a k-independent set for $k = n - t$. We use this correspondence to translate our bounds on the resolution complexity of independent sets to those on the resolution complexity of vertex covers.

Consider encoding in clausal form the statement that G has a t-vertex cover. The only defining difference between an independent set and a vertex cover is that the former requires at most one of the endpoints of every edge to be included, where as the latter requires at least one. Natural methods to count remain the same, that is, explicit counting variables, implicit mapping variables, or blocks. Similar to the independent set encoding variables, let $x'_v, 1 \leq v \leq n$, be a set of variables such that $x'_v = \text{TRUE}$ iff vertex v is chosen to be in the vertex cover. Let $y'_{v,i}, 1 \leq v \leq n, 1 \leq i \leq t$, denote the fact that exactly i of the first v vertices are chosen in the vertex cover. Let $z'_{v,i}, 1 \leq v \leq n, 1 \leq i \leq t$, represent that vertex v is mapped to the i^{th} node of the vertex cover.

The *counting encoding* of vertex cover, $VC_{count}(G, t)$, is defined analogous to $\alpha_{count}(G, k)$ except for the change that for an edge $(u, v) \in E$, the edge clause for vertex cover is $(x'_u \vee x'_v)$ and not $(\neg x'_u \vee \neg x'_v)$. The rest of the encoding is obtained by setting $k \leftarrow t, x_v \leftarrow x'_v, y_{v,i} \leftarrow y'_{v,i}$. The *mapping encoding* of vertex cover, $VC_{mapping}(G, t)$ is similarly defined analogous to $\alpha_{mapping}(G, k)$ by setting $k \leftarrow t, z_{v,i} \leftarrow z'_{v,i}$, except for the change in the edge clauses for edges $(u, v) \in E$ from $(\neg z_{u,i} \vee \neg z_{v,i})$ to $(z'_{u,i} \vee z'_{v,i})$. For $b = n/(n - t)$, the *block encoding* of vertex cover over $(n - t)$ blocks of size b each, $VC_{block}(G, t)$, is also defined analogous to $\alpha_{block}(G, k)$ by setting $k \leftarrow (n - t), x_v \leftarrow \neg x'_v$. It says that each edge is covered, and exactly $b - 1$ vertices from each block are selected in the vertex cover, for a total of $(n - t)(b - 1) = t$ vertices. Note that the 1-1 clauses of α_{block} translate into "all-but-one" clauses of VC_{block}.

It is not surprising that the resolution complexity of various encodings of the vertex cover problem is intimately related to that of the corresponding encodings of the independent set problem. We formalize this in the following lemmas.

Lemma 6.1. *For any graph G over n vertices,*

$$\mathsf{RES}(VC_{count}(G, t)) \quad \leq \quad \mathsf{RES}(\alpha_{count}(G, n - t)) + 6nt^2.$$

Proof. If G has an independent set of size $(n-t)$, then there is no resolution refutation of $\alpha_{count}(G, n-t)$. Consequently, $\mathsf{RES}(\alpha_{count}(G, n-t)) = \mathsf{DPLL}(\alpha_{count}(G, n-t)) = \infty$, trivially satisfying the claimed inequalities. Otherwise, consider a refutation π of $\alpha_{count}(G, n-t)$. We use π to construct a refutation π' of $VC_{count}(G, t)$ that is not too big.

Recall that the variables of π are $x_u, 1 \leq u \leq n$, and $y_{v,i}, 0 \leq i \leq v \leq n, 0 \leq i \leq n-t$. The variables of π' will be $x'_u, 1 \leq u \leq n$, and $y'_{v,i}, 0 \leq i \leq v \leq n, 0 \leq i \leq t$. Notice that the number of independent set counting variables $y_{v,i}$ is not the same as the number of vertex cover counting variables $y'_{v,i}$. We handle this by adding dummy counting variables, transforming π, and removing extra variables. To obtain π', apply transforms σ_1, σ_2, and σ_3 defined below to π.

σ_1 simply creates new counting variables $y_{v,i}, 0 \leq v \leq n, (n-t+1) \leq i \leq v$, and adds counting clauses corresponding to these variables as unused initial clauses of π. σ_2 sets $x_u \leftarrow \neg x'_u, y_{v,i} \leftarrow y'_{v,v-i}$. Intuitively, σ_2 says that i of the first v vertices being in the independent set is equivalent to exactly $v-i$ of the first v vertices being in the vertex cover. σ_3 sets $y'_{v,i} \leftarrow$ FALSE for $0 \leq v \leq n, (t+1) \leq i \leq v$. Since σ_1, σ_2, and σ_3 only add new clauses, rename literals or set variables, their application transforms π into another, potentially simpler, refutation on a different set of variables and initial clauses. Call the resulting refutation π''.

The initial edge clauses $(\neg x_u \vee \neg x_v)$ of π transform into edge clauses $(x'_u \vee x'_v)$ of $VC_{count}(G, t)$. The initial size-$(n-t)$ clause of π transforms into the initial size-t clause of $VC_{count}(G, t)$. Finally, the initial counting clauses of π, including those corresponding to the variables added by σ_1, transform into counting clauses of $VC_{count}(G, t)$ and n extra clauses. To see this, note that σ_2 transforms counting formulas $y_{0,0}$ into $y'_{0,0}$, $(y_{v,0} \leftrightarrow (y_{v-1,0} \wedge \neg x_v))$ into $(y'_{v,v} \leftrightarrow (y'_{v-1,v-1} \wedge x'_v))$, for $i \geq 1 : (y_{v,i} \leftrightarrow ((y_{v-1,i} \wedge \neg x_v) \vee (y_{v-1,i-1} \wedge x_v)))$ into $(y'_{v,v-i} \leftrightarrow ((y'_{v-1,v-i-1} \wedge x'_v) \vee (y'_{v-1,v-i} \wedge \neg x'_v)))$, and $(y_{v,v} \leftrightarrow (y_{v-1,v-1} \wedge x_v))$ into $(y'_{v,0} \leftrightarrow (y_{v-1,0} \wedge \neg x'_v))$. Applying σ_3 to set $y'_{v,i} \leftarrow$ FALSE for $(t+1) \leq i \leq v$ removes all but the initial counting clauses of $VC_{count}(G, t)$ and the counting formulas corresponding to the variables $y'_{v,t+1}, t+1 \leq v \leq n$, that simplify to $(\neg y'_{v-1,t} \vee \neg x'_v)$. Call this extra set of $(n-t)$ clauses $Bdry(G, t)$, or *boundary* clauses for (G, t).

At this stage, we have a refutation π'' of size at most $size(\pi)$ starting from clauses $VC_{count}(G, t) \cup Bdry(G, t)$. The boundary clauses together say that no more than t vertices are chosen in the vertex cover. This, however, is implied by the rest of the initial clauses. Using this fact, we first give a derivation π_{Bdry} of every boundary clause starting from the clauses of $VC_{count}(G, t)$. Appending π'' to π_{Bdry} gives a refutation π' of $VC_{count}(G, t)$.

Let $S_i = \bigvee_{i'=0}^{\min\{i,t\}} y'_{n-i,t-i'}$ for $0 \leq i \leq n-t$. Let $R_{v,i,j} = (\neg y'_{v,i} \vee \neg y'_{v,j})$ for $0 \leq i < j \leq v \leq n$ and $j \leq t$. We first give a derivation of these S and R clauses, and then say how to derive the boundary clauses from these. $S_0 = y'_{n,t}$ is an initial clause, and $S_i, i \geq 1$, is obtained by sequentially resolving S_{i-1} with the counting clauses $(\neg y'_{n-i+1,t-i'} \vee y'_{n-i,t-i'} \vee y'_{n-i,t-i'-1})$ for $0 \leq i' < \min\{i,t\}$. Similarly, when $i = 0$, $R_{v,0,v}$ is derived by resolving counting clauses $(\neg y'_{v,0} \vee \neg x'_v)$ and $(\neg y'_{v,v} \vee x'_v)$ on x'_v, clauses $R_{v,0,j}$ for $0 < j < v$ are derived by sequentially resolving $R_{v-1,0,j}$ with the counting clauses $(\neg y'_{v,j} \vee y'_{v-1,j} \vee x'_v)$ and $(\neg y'_{v,0} \vee \neg x'_v)$. Note that $R_{v,0,v}$ and $R_{v,0,j}$ are defined and derived only when $j \leq t$. When $i > 0$, $R_{v,i,v}$ is derived by sequentially resolving $R_{v-1,i-1,v-1}$ with the counting clauses $(\neg y'_{v,i} \vee y'_{v-1,i-1} \vee \neg x'_v)$ and $(\neg y'_{v,v} \vee y'_{v-1,v-1} \vee \neg x'_v)$, and resolving the result on x'_v with the counting clause $(\neg y'_{v,v} \vee x'_v)$. Finally, $R_{v,i,j}$ for $j < v$ is derived by resolving $R_{v-1,i,j}$ with the counting clauses $(\neg y'_{v,i} \vee y'_{v-1,i} \vee x'_v)$ and $(\neg y'_{v,j} \vee y'_{v-1,j} \vee x'_v)$, resolving $R_{v-1,i-1,j-1}$ with the counting clauses $(\neg y'_{v,i} \vee y'_{v-1,i-1} \vee \neg x'_v)$ and $(\neg y'_{v,j} \vee y'_{v-1,j-1} \vee \neg x'_v)$, and resolving the result of the two on x'_v.

To derive the boundary clause $(\neg y'_{v-1,t} \vee \neg x'_v)$ for any v, resolve each pair of clauses $(\neg y'_{v,t-i'} \vee y'_{v-1,t-i'-1} \vee \neg x'_v)$ and $R_{v-1,t-i'-1,t}$ for $0 \leq i' \leq \min\{n-v,t\}$, and resolve all resulting clauses with S_{n-v}. Note that when $\min\{n-v,t\} = t$, there is no $R_{v-1,t-i'-1,t}$, but the corresponding counting clause itself is of the desired form, $(\neg y'_{v,0} \vee \neg x'_v)$. This finishes the derivation π_{Bdry} of all clauses in $Bdry(G,t)$. As stated earlier, appending π'' to π_{Bdry} gives a refutation π' of $VC_{count}(G,t)$.

For general resolution, $size(\pi') = size(\pi'') + size(\pi_{Bdry}) \leq size(\pi) + size(\pi_{Bdry})$. Each S_i in π_{Bdry}, starting with $i = 0$, is derived in $\min\{i,t\}$ resolution steps from previous clauses, and each $R_{v,i,j}$, starting with $i = 0, v = j = 1$, requires at most 5 resolution steps from previous clauses. Hence, $size(\pi_{Bdry}) \leq nt + 5nt^2 \leq 6nt^2$ for large enough n, implying that $size(\pi') \leq size(\pi) + 6nt^2$. Note that this approach doesn't quite work for tree-like resolution proofs because π_{Bdry} itself becomes exponential in size due to the heavy reuse of clauses involved in the derivation of the various $R_{v,i,j}$. □

Given that the encodings $\alpha_{count}(G, n-t)$ and $VC_{count}(G,t)$ are duals of each other, the argument made for the lemma above can also be made the other way, immediately giving us the following reverse result:

Lemma 6.2. *For any graph G over n vertices,*

$$\mathsf{RES}(\alpha_{count}(G,k)) \;\; \leq \;\; \mathsf{RES}(VC_{count}(G,n-k)) + 6nk^2.$$

Lemma 6.3. *For any graph G over n vertices and $(n-t) \,|\, n$,*

$$\begin{aligned}\mathsf{RES}(VC_{block}(G,t)) &= \mathsf{RES}(\alpha_{block}(G,n-t)) \quad \textit{and} \\ \mathsf{DPLL}(VC_{block}(G,t)) &= \mathsf{DPLL}(\alpha_{block}(G,n-t)).\end{aligned}$$

This result also holds without the 1-1 clauses of α_{block} and the corresponding all-but-one clauses of VC_{block}.

Proof. If G has an independent set of size $(n-t)$, then it also has a vertex cover of size t. In this case, there are no resolution refutations of $VC_{block}(G,t)$ or $\alpha_{block}(G,n-t)$, making the resolution complexity of both infinite and trivially satisfying the claim.

Otherwise, consider a refutation π of $\alpha_{block}(G,n-t)$. We use π to construct a refutation π' of $VC_{block}(G,t)$, which is of the same size and is tree-like if π is. π' is obtained from π by simply applying the transformation $x_v \leftarrow \neg x'_v, 1 \leq v \leq n$. Since this is only a 1-1 mapping between literals, π' is a legal refutation of size exactly $size(\pi)$. All that remains to argue is that the initial clauses of π' are the clauses of $VC_{block}(G,t)$. This, however, follows immediately from the definition of $VC_{block}(G,t)$.

Given the duality of the encodings $VC_{block}(G,t)$ and $\alpha_{block}(G,n-t)$, we can repeat the argument above to translate any refutation of the former into one of the latter. Combining this with the above, the resolution complexity of the two formulas is exactly the same. □

6.2. Coloring. A *K-coloring* of a graph $G=(V,E)$ is a function $col : V \to \{1,2,\ldots,K\}$ such that for every edge $(u,v) \in E$, $col(u) \neq col(v)$. For a random graph G chosen from a distribution similar to $\mathbb{G}(n,p)$, the resolution complexity of the formula $\chi(G,K)$ saying that G is K-colorable has been addressed by Beame *et al.* (2005).

Suppose G is K-colorable. Fix a K-coloring col of G and partition the vertices into color classes $V_i, 1 \leq i \leq K$, where $V_i = \{v \in V : col(v) = i\}$. Each color class, by definition, must be an independent set, with the largest of size at least n/K. Thus, non-existence of a $k \overset{\text{def}}{=} n/K$ size independent set in G implies the non-existence of a K-coloring of G.

Let $\alpha(G,k)$ be an encoding of the k-independent set problem on graph G. The correspondence above can be used to translate properly encoded resolution proofs of $\alpha(G,k)$ into those of $\chi(G,K)$. A lower bound on $\mathsf{RES}(\chi(G,K))$ (e.g., Beame *et al.* 2005) would then imply a lower bound on $\mathsf{RES}(\alpha(G,k))$. However, such a translation between proofs must involve a resolution counting argument showing that K sets of vertices, each of size less than n/K, cannot cover all

n vertices. This argument itself is at least as hard as a resolution proof of PHP^n_{n-K}, the (weak) pigeonhole principle on n pigeons and $(n-K)$ holes, for which an exponential lower bound has been shown by Raz (2004). This makes any translation of a proof of $\alpha(G,k)$ into one of $\chi(G,K)$ necessarily large, ruling out any interesting lower bounds for independent sets as a consequence of those for coloring.

On the other hand, non-existence of a K-coloring does not imply the non-existence of a k-independent set. In fact, there are very simple graphs with no K-coloring but with an independent set as large as $(n-K)$ (e.g., a clique of size $(K+1)$ along with $(n-K-1)$ nodes of degree zero). Consequently, our lower bounds for independent sets do not give any interesting lower bounds for K-coloring.

7. Upper Bounds

Based on a very simple exhaustive backtracking strategy, we give upper bounds on the DPLL (and hence resolution) complexity of the independent set and the vertex cover encodings we have considered.

LEMMA 7.1. *There is a constant C_0 such that if G is a graph over n vertices with no independent set of size k, then*

$$\mathsf{DPLL}(\alpha_{map}(G,k)) \quad \leq \quad 2^{C_0 k \ln(ne/k)}.$$

This bound also holds when $\alpha_{map}(G,k)$ does not include the 1-1 clauses.

PROOF. A straightforward way to disprove the existence of a k-independent set is to go through all $\binom{n}{k}$ subsets of vertices of size k and use as evidence an edge from each subset. We use this strategy to construct a refutation of $\alpha_{map}(G,k)$.

To begin with, apply transitivity to derive all ordering clauses of the form $(\neg z_{u,j} \vee \neg z_{v,i})$ for $u < v$ and $i < j$. If $j = i+1$, this is simply one of the original ordering clauses. For $j = i+2$, derive the new clause $(\neg z_{u,i+2} \vee \neg z_{v,i})$ as follows. Consider any $w \in \{1, 2, \ldots, n\}$. If $u < w$, we have the ordering clause $(\neg z_{w,i+1} \vee \neg z_{u,i+2})$, and if $u \geq w$, then $v > w$ and we have the ordering clause $(\neg z_{v,i} \vee \neg z_{w,i+1})$. Resolving these n ordering clauses (one for each w) with the surjective clause $(z_{1,i+1} \vee \ldots \vee z_{n,i+1})$ gives the new ordering clause $(\neg z_{u,i+2} \vee \neg z_{v,i})$ associated with u and v. This clearly requires only n steps and can be done for all $u < v$ and $j = i+2$. Continue to apply this argument for $j = i+3, i+4, \ldots, k$ and derive all new ordering clauses in n steps each.

We now construct a tree-like refutation starting with the initial clauses and the new ordering clauses we derived above. We claim that for any $i \in \{1, 2, \ldots, k\}$ and for any $1 \leq v_i < v_{i+1} < \ldots < v_k \leq n$, a subclause of $(\neg z_{v_i,i} \vee \neg z_{v_{i+1},i+1} \vee \ldots \vee \neg z_{v_k,k})$ can be derived. We first argue why this claim is sufficient to obtain a refutation. For $i = k$, the claim says that a subclause of $\neg z_{v_k,k}$ can be derived for all $1 \leq v_k \leq n$. If any one of these n subclauses is a strict subclause of $\neg z_{v_k,k}$, it has to be the empty clause, resulting in a refutation. Otherwise, we have $\neg z_{v_k,k}$ for every v_k. Resolving all these with the surjective clause $(z_{1,k} \vee \ldots \vee z_{n,k})$ results in the empty clause.

We now prove the claim by induction on i. For the base case, fix $i = 1$. For any given k vertices $v_1 < v_2 < \ldots < v_k$, choose an edge (v_p, v_q) that witnesses the fact that these k vertices do not form an independent set. The corresponding edge clause $(\neg z_{v_p,p} \vee \neg z_{v_q,q})$ works as the required subclause.

For the inductive step, fix $v_{i+1} < v_{i+2} < \ldots < v_k$. We will derive a subclause of $(\neg z_{v_{i+1},i+1} \vee \neg z_{v_{i+2},i+2} \vee \ldots \neg z_{v_k,k})$. By induction, derive a subclause of $(\neg z_{v_i,i} \vee \neg z_{v_{i+1},i+1} \vee \ldots \vee z_{v_k,k})$ for any choice of $v_i < v_{i+1}$. If for some such v_i, $\neg z_{v_i,i}$ does not appear in the corresponding subclause, then the same subclause works here for the inductive step and we are done. Otherwise, for every $v_i < v_{i+1}$, we have a subclause of $(\neg z_{v_i,i} \vee \neg z_{v_{i+1},i+1} \vee \ldots \vee \neg z_{v_k,k})$ that contains $\neg z_{v_i,i}$. Resolving all these subclauses with the surjective clause $(z_{1,i} \vee z_{2,i} \vee \ldots \vee z_{n,i})$ results in the clause $(z_{v_{i+1},i} \vee \ldots \vee z_{v_k,i} \vee \neg z_{u_1,j_1} \vee \ldots \vee \neg z_{u_p,j_p})$, where each z_{u_c,j_c} lies in $\{z_{v_{i+1},i+1}, \ldots, z_{v_k,k}\}$. Observe that for each positive literal $z_{v_q,i}, i+1 \leq q \leq k$, in this clause, $(\neg z_{v_q,i} \vee \neg z_{v_{i+1},i+1})$ is either a 1-1 clause or an ordering clause. Resolving with all these clauses finally gives $(\neg z_{v_{i+1},i+1} \vee \neg z_{u_1,j_1} \vee \ldots \vee \neg z_{u_p,j_p})$, which is the kind of subclause we wanted to derive. This proves the claim.

Associate each subclause obtained using the iterative procedure above with the tuple $(v_i, v_{i+1}, \ldots, v_k)$ for which it was derived, giving a total of $\sum_{i=1}^{k} \binom{n}{i} \leq (ne/k)^k$ subclauses. Each of these subclauses is used at most once in the proof. Further, the derivation of each such subclause uses at most n new ordering clauses, each of which can be derived in at most n^2 steps. Thus, with enough copies to make the refutation tree-like, the size of the proof is $O(n^3(ne/k)^k)$, which is at most $2^{C_0 k \ln(ne/k)}$ for a large enough constant C_0. □

Lemma 7.2. *There is a constant C'_0 such that if G is a graph over n vertices with no independent set of size k, then*

$$\mathsf{DPLL}(\alpha_{count}(G, k)) \quad \leq \quad 2^{C'_0 k \ln(ne/k)}.$$

Proof. As in the proof of Lemma 7.1, we construct a refutation by looking at each size-k subset of vertices and using as evidence an edge from that subset.

For every i, v such that $0 \leq i \leq v < n$, first derive a new counting clause $(\neg y_{v+1,i+1} \vee y_{v,i} \vee y_{v-1,i} \vee \ldots \vee y_{i,i})$ by resolving original counting clauses $(\neg y_{u+1,i+1} \vee y_{u,i+1} \vee y_{u,i})$ for $u = v, v-1, \ldots, i+1$ together, and resolving the result with the counting clause $(\neg y_{i+1,i+1} \vee y_{i,i})$. Next, for any edge (i,j), $i > j$, resolve the edge clause $(\neg x_i \vee \neg x_j)$ with the counting clauses $(\neg y_{i,i} \vee x_i)$ and $(\neg y_{j,j} \vee x_j)$ to get the clause $(\neg y_{i,i} \vee \neg y_{j,j})$. Call this new clause $E_{i,j}$. We now construct a tree-like refutation using the initial clauses, these new counting clauses, and the new $E_{i,j}$ clauses.

We claim that for any $i \in \{1, 2, \ldots, k\}$ and for any $1 \leq v_i < v_{i+1} < \ldots < v_k \leq n$ with $v_j \geq j$ for $i \leq j \leq k$, we can derive a subclause of $(\neg y_{v_i,i} \vee y_{v_i-1,i} \vee \neg y_{v_{i+1},i+1} \vee y_{v_{i+1}-1,i+1} \vee \ldots \vee \neg y_{v_k,k} \vee y_{v_k-1,k})$ such that if $y_{v_{j-1},j}$ occurs in the subclause for some j, then so does $\neg y_{v_j,j}$. Note that for $v_j = j$, the variable $y_{v_j-1,j}$ does not even exist and will certainly not appear in the subclause. Given this claim, we can derive for $i = k$ a subclause B_j of $(\neg y_{j,k} \vee y_{j-1,k})$ for each $j \in \{k+1, \ldots, n\}$ and a subclause B_k of $\neg y_{k,k}$. If any such B_j is the empty clause, the refutation is complete. Otherwise every B_j contains $\neg y_{j,k}$. Let j' be the largest index such that $B_{j'}$ does not contain $y_{j'-1,k}$. Since B_k has to be the clause $\neg y_{k,k}$, such a j' must exist. Resolving all $B_j, j \in \{j', \ldots, k\}$, with each other gives the clause $y_{n,k}$. Resolving this with the size-k clause $y_{n,k}$ gives the empty clause.

We now prove the claim by induction on i. For the base case $i = 1$, fix $1 \leq v_1 < v_2 < \ldots < v_k \leq n$. Choose an edge (v_p, v_q) that witnesses the fact that $v_1, \ldots, v_k$ do not form an independent set. Resolve the corresponding edge clause $(\neg x_{v_p} \vee \neg x_{v_q})$ with the counting clauses $(\neg y_{v_p,p} \vee y_{v_p-1,p} \vee x_p)$ and $(\neg y_{v_q,q} \vee y_{v_q-1,q} \vee x_q)$ to get $(\neg y_{v_p,p} \vee y_{v_p-1,p} \vee \neg y_{v_q,q} \vee y_{v_q-1,q})$, which is a subclause of the desired form.

For the inductive step, fix $v_{i+1} < v_{i+2} < \ldots < v_k$. By induction, derive a subclause C_j of $(\neg y_{j,i} \vee y_{j-1,i} \vee \neg y_{v_{i+1},i+1} \vee y_{v_{i+1}-1,i+1} \vee \ldots \vee \neg y_{v_k,k} \vee y_{v_k-1,k})$ for any j in $\{i, i+1, \ldots, v_{i+1} - 1\}$. If for some such j, neither $\neg y_{j,i}$ nor $y_{j-1,i}$ appears in C_j, then this subclause also works here for the inductive step and we are done. Otherwise for every j, C_j definitely contains $\neg y_{j,i}$, possibly $y_{j-1,i}$, and other positive or negative occurrences of variables of the form $y_{v',i'}$ where $i' > i$. Now use all these C_j to derive clauses C'_j such that each C'_j contains $\neg y_{j,i}$ but not $y_{j-1,i}$. The other variables appearing in C'_j will all be of the form $y_{v',i'}$ for $i' > i$.

If $\{v_{i+1}, \ldots, v_k\}$ is not an independent set, then there is an edge (v_p, v_q) witnessing this. In this case, simply use $E_{p,q}$ as the desired subclause and the inductive step is over. Otherwise there must be an edge (i, v_q) from vertex i touching this set. Let C'_i be the clause E_{i,v_q}. For j going from $i+1$ to k, do the following iteratively. If $y_{j-1,i}$ does not appear in C_j, then set $C'_j = C_j$. Otherwise set C'_j to be the clause obtained by resolving C_j with C'_{j-1}. If C'_{j-1} does not contain $\neg y_{j,i}$, then it can be used as the desired subclause for this inductive step and the iteration is stopped here, otherwise it continues onto the next value of j. If the desired subclause is not derived somewhere along this iterative process, then we end up with all C'_j containing $\neg y_{j,i}$ but not $y_{j-1,i}$. Resolving all these with the new counting clause $(\neg y_{v_{i+1},i+1} \vee y_{v_{i+1}-1,i} \vee y_{v_{i+1}-2,i} \vee \ldots \vee y_{i,i})$ finally gives a subclause of the desired form. This proves the claim.

Associate each subclause obtained using the iterative procedure above with the tuple $(v_i, v_{i+1}, \ldots, v_k)$ for which it was derived, giving a total of $\sum_{i=1}^{k} \binom{n}{i} \leq (ne/k)^k$ subclauses. Each of these subclauses is used at most once in the proof. Further, the derivation of each such subclause uses one new counting clause and one new clause $E_{i,j}$, each of which can be derived in at most n steps. Thus, with enough copies to make the refutation tree-like, the size of the proof is $O(n(ne/k)^k)$, which is at most $2^{C'_0 k \ln(ne/k)}$ for a large enough constant C'_0. □

Theorem 7.3 (Independent Set Upper Bounds). *There are constants c_0, c'_0 such that the following holds. Let $\Delta = np$, $\Delta \leq n/\ln^2 n$, and $G \sim \mathbb{G}(n,p)$. Let k be such that G has no independent set of size k. With probability $1 - o(1)$ in n,*

$$\begin{aligned} \mathsf{DPLL}(\alpha_{map}(G,k)) &\leq 2^{c_0 (n/\Delta) \ln^2 \Delta}, \\ \mathsf{DPLL}(\alpha_{count}(G,k)) &\leq 2^{c'_0 (n/\Delta) \ln^2 \Delta}, \quad \textit{and} \\ \mathsf{DPLL}(\alpha_{block}(G,k)) &\leq 2^{c_0 (n/\Delta) \ln^2 \Delta}. \end{aligned}$$

The bounds also hold when the 1-1 clauses are removed from $\alpha_{map}(G,k)$ or $\alpha_{block}(G,k)$. The block encoding bound holds when $k \mid n$.

Proof. By Proposition 3.1, $n/(\Delta+1) < k \leq n$. Hence $k \ln(ne/k) \leq n \ln(e(\Delta+1))$. We will use this fact when Δ is a relatively small constant.

Fix any $\epsilon > 0$ and let C_ϵ be the corresponding constant from Proposition 3.2. When $\Delta < C_\epsilon$, the desired upper bounds in this theorem are of the form $2^{O(n)}$. Moreover, the upper bounds provided by Lemmas 7.1 and 7.2 for the mapping and the counting encoding, respectively, are exponential in $k \ln(ne/k) \leq n \ln(e(\Delta+1))$, and thus also of the form $2^{O(n)}$ when $\Delta < C_\epsilon$. Hence, for large enough constants c_0 and c'_0, the claimed bounds hold with probability 1 for the mapping and the counting encodings when $\Delta < C_\epsilon$. Lemma 4.5 extends this to the block encoding as well.

Assume for the rest of this proof that $C_\epsilon \leq \Delta \leq n/\ln^2 n$. Let $k_{min} \leq k$ be the smallest integer such that G does not have an independent set of size k_{min}. By Proposition 3.2, with probability $1 - o(1)$ in n, $k_{min} \leq k_{+\epsilon} + 1$.

For the mapping-based encoding,

$$\begin{aligned}
\mathsf{DPLL}(\alpha_{map}(G,k)) &\leq \mathsf{DPLL}(\alpha_{map}(G,k_{min})) && \text{by Lemma 4.2}\\
&\leq 2^{C_0 k_{min} \ln(n/k_{min})} && \text{by Lemma 7.1}\\
&\leq 2^{C_0 (k_{+\epsilon}+1) \ln(n/(k_{+\epsilon}+1))} && \text{almost surely}\\
&\leq 2^{(c_0 n/\Delta) \ln^2 \Delta} && \text{for large enough } c_0.
\end{aligned}$$

The bound for $\alpha_{block}(G,k)$ follows immediately from this bound for $\alpha_{map}(G,k)$ and Lemma 4.5. Further, Lemma 7.1 implies that these bounds hold even when the corresponding 1-1 clauses are removed from the mapping and the block encodings. For the counting-based encoding,

$$\begin{aligned}
\mathsf{DPLL}(\alpha_{count}(G,k)) &\leq n \cdot \mathsf{DPLL}(\alpha_{count}(G,k_{min})) + 2n^2 && \text{by Lemma 4.1}\\
&\leq n 2^{C_0' k_{min} \ln(n/k_{min})} + 2n^2 && \text{by Lemma 7.2}\\
&\leq n 2^{C_0' k_{min} \ln(n/k_{min})} + 2n^2 && \\
&\leq n 2^{C_0' (k_{+\epsilon}+1) \ln(n/(k_{+\epsilon}+1))} + 2n^2 && \text{almost surely}\\
&\leq 2^{(c_0' n/\Delta) \ln^2 \Delta}
\end{aligned}$$

for a large enough constant c_0'. This finishes the proof. □

Corollary 7.4 (Vertex Cover Upper Bounds). *There are constants* c_0, c_0'' *such that the following holds. Let* $\Delta = np$, $\Delta \leq n/\ln^2 n$, *and* $G \sim \mathbb{G}(n,p)$. *Let* t *be such that* G *has no vertex cover of size* t. *With probability* $1 - o(1)$ *in* n,

$$\begin{aligned}
\mathsf{RES}(VC_{count}(G,t)) &\leq 2^{c_0''(n/\Delta)\ln^2\Delta}, \quad \textit{and}\\
\mathsf{DPLL}(VC_{block}(G,t)) &\leq 2^{c_0(n/\Delta)\ln^2\Delta}.
\end{aligned}$$

The bounds also hold when the all-but-one clauses are removed from $VC_{block}(G,t)$. *The block encoding bound holds when* $(n-t) \mid n$.

Proof. Apply Theorem 7.3 with k set to $(n-t)$ and use Lemmas 6.1 and 6.3 to translate the result of the Theorem to the encodings of vertex cover. Note that $\mathsf{RES}(\alpha_{count}(G,n-t)) \leq \mathsf{DPLL}(\alpha_{count}(G,n-t))$. □

8. Key Concepts for Lower Bounds

This section defines the key concepts that will be used in the lower bound argument given in the next section. Fix a graph G and partition its n vertices into k consecutive subsets of size b each. For any edge (u, v) in G, call it an *inter-block edge* if u and v belong to different blocks of G, and an *intra-block edge* otherwise.

Definition 8.1. *A truth assignment to variables of* $\alpha_{block}(G, k)$ *is* critical *if it sets exactly one variable in each block to* TRUE.

Critical truth assignments satisfy all block, 1-1, and intra-block edge clauses, but may leave some inter-block edge clauses unsatisfied.

Definition 8.2. *The block multi-graph of* G*, denoted* $B(G)$*, is the multi-graph obtained from* G *by identifying all vertices that belong to the same block and removing any self-loops that are thus generated.*

$B(G)$ contains exactly k nodes and possibly multiple edges between pairs of nodes. The degree of a node in $B(G)$ is the number of inter-block edges touching the corresponding block of G. Given the natural correspondence between G and $B(G)$, we will write *nodes of* $B(G)$ and *blocks of* G interchangeably. For a subgraph H of G, $B(H)$ is obtained in an analogous way by identifying all vertices of H that are in the same block of G and removing self-loops.

Definition 8.3. *Let* S *be a set of blocks of* G*.* H *is* block-induced by S *if it is the subgraph of* G *induced by all vertices present in the blocks* S*.* H *is a* block-induced subgraph *of* G *if there exists a subset* S *of blocks such that* H *is block-induced by* S*.*

If H is block-induced by S, then $B(H)$ is induced by S in $B(G)$. The reverse, however, may not hold. If H is a block-induced subgraph, then there is a *unique* minimal block set S such that H is block-induced by S. This S contains exactly those blocks that have non-zero degree in $B(H)$. With each block-induced subgraph, associate such a *minimal* S and say that the subgraph is induced by $|S|$ blocks. Note that every block in any such minimal S must have non-zero degree.

Definition 8.4. *The* block width *of a clause* C *with respect to* G*, denoted* $w^G_{block}(C)$*, is the number of different blocks of* G *the variables appearing in* C *come from.*

Clearly, $w(C) \geq w^G_{block}(C)$. For a block-induced subgraph H of G, let $E(H)$ denote the conjunction of the edge clauses of $\alpha_{block}(G,k)$ that correspond to the edges of H. Let H be induced by the block set S.

Definition 8.5. *H* critically implies *a clause C, denoted $H \overset{c}{\rightarrow} C$, if $E(H) \rightarrow C$ evaluates to true for all critical truth assignments to the variables of $\alpha_{block}(G,k)$.*

Definition 8.6. *H* minimally implies *C, denoted $H \overset{m}{\rightarrow} C$, if $H \overset{c}{\rightarrow} C$ and for every subgraph H' of G induced by a proper subset of S, $H' \overset{c}{\not\rightarrow} C$.*

Note that "minimally implies" should really be called "minimally critically implies," but we use the former phrase for brevity. Note further that if $H \overset{m}{\rightarrow} C$, then every block of H has non-zero degree.

Definition 8.7. *The* complexity *of a clause C, denoted $\mu_G(C)$, is the minimum over the sizes of subsets S of blocks of G such that the subgraph of G induced by S critically implies C.*

Proposition 8.8. *Let G be a graph and Λ denote the empty clause.*

(a) For $C \in \alpha_{block}(G,k)$, $\mu_G(C) \leq 2$.

(b) $\mu_G(\Lambda)$ is the number of blocks in the smallest block-induced subgraph of G that has no block-respecting independent set.

(c) Subadditive property*: If clause C is a resolvent of clauses C_1 and C_2, then $\mu_G(C) \leq \mu_G(C_1) + \mu_G(C_2)$.*

Proof. Each initial clause is either an edge clause, a block clause or a 1-1 clause. Any critical truth assignment, by definition, satisfies all block, 1-1, and intra-block edge clauses. Further, an edge clause corresponding to an inter-block edge (u,v) is implied by the subgraph induced by the two blocks to which u and v belong. Hence, complexity of an initial clause is at most 2, proving part (a).

Part (b) follows from the definition of μ_G. Part (c) follows from the simple observation that if G_1 critically implies C_1, G_2 critically implies C_2, and both G_1 and G_2 are block-induced subgraphs, then $G_{1\cup 2}$, defined as the block graph induced by the union of the blocks G_1 and G_2 are induced by, critically implies both C_1 and C_2, and hence critically implies C. □

9. Proof Sizes and Graph Expansion

This section contains the main ingredients of our lower bound results and is technically the most interesting and challenging part at the core of this paper. We use combinatorial properties of block graphs and independent sets to obtain a lower bound on the size of resolution refutations for a given graph in terms of its expansion properties. Next, we argue that random graphs almost surely have good expansion properties. Section 10 combines these two to obtain an almost certain lower bound for random graphs.

The overall argument in a little more detail is as follows. We define the notion of "boundary" for block-induced subgraphs as a measure of the number of blocks in it that have an isolated vertex and thus contribute trivially to any block-respecting independent set. Lemmas 9.2 and 9.4 relate this graph-theoretic concept to resolution refutations. The main lower bound follows in three steps from here. First, Lemma 9.8 argues that one must almost surely consider a large fraction of the blocks of a graph to prove the non-existence of a block-respecting independent set in it. Second, Lemma 9.9 shows that almost all subgraphs induced by a large fraction of blocks must have a large boundary. Finally, Lemma 9.10 combines these two to obtain an almost certain lower bound on the width of any refutation.

We begin by defining the notion of boundary.

Definition 9.1. *The* boundary *of a block-induced subgraph H, denoted $\beta(H)$, is the set of blocks of H that have at least one isolated vertex.*

9.1. Relating Proof Size to Graph Expansion. We first derive a relationship between the width of clauses and the boundary size of block-induced subgraphs that minimally imply them.

Lemma 9.2. *Let C be a clause in the variables of $\alpha_{block}(G,k)$ and H be a block-induced subgraph of G. If $H \overset{m}{\rightarrow} C$, then $w^{G}_{block}(C) \geq |\beta(H)|$.*

Proof. We use a *toggling property* of block-respecting independent sets (Figure 9.1) to show that each boundary block of H contributes at least one literal to C.

Let H be induced by a set S of blocks. Fix a boundary block $B \in S$. Let H_B be the subgraph induced by $S \setminus \{B\}$. By minimality of H, $H_B \overset{c}{\not\rightarrow} C$. Therefore, there exists a critical truth assignment γ such that $\gamma(E(H_B)) =$ true but $\gamma(C) =$ false. Since $\gamma(C) =$ false and $H \overset{c}{\rightarrow} C$, it follows that $\gamma(E(H)) =$ false. Further, since $\gamma(E(H_B)) =$ true, $\gamma(E(H) \setminus E(H_B))$ must

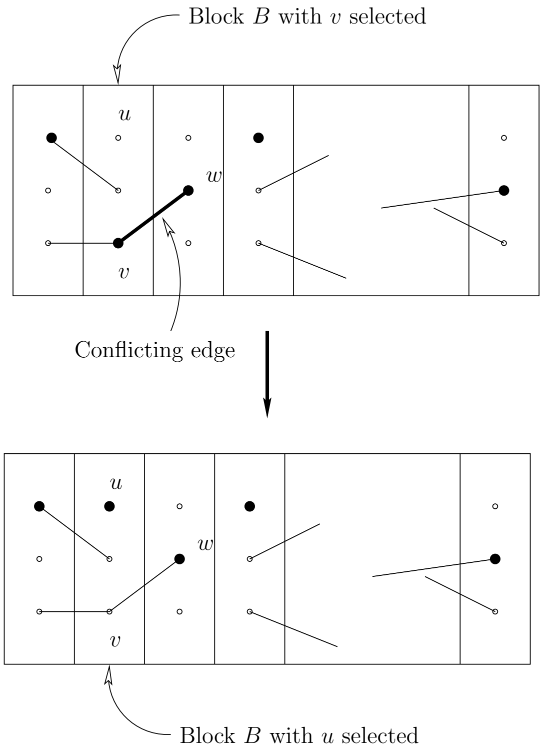

Figure 9.1: Toggling property of block-respecting independent sets; selected vertices are shown in bold

be FALSE, implying that γ violates the edge clause corresponding to an inter-block edge $(v, w), v \in B, w \notin B$. In particular, $\gamma(v) = \text{TRUE}$.

Fix an isolated vertex $u \in B$. Create a new critical truth assignment $\bar{\gamma}$ as follows: $\bar{\gamma}(v) = \text{FALSE}$, $\bar{\gamma}(u) = \text{TRUE}$, and $\bar{\gamma}(x) = \gamma(x)$ for every other vertex x in H. By construction, $\bar{\gamma}(E(H_B)) = \gamma(E(H_B)) = \text{TRUE}$. Further, since u does not have any inter-block edges and γ is critical, even $\bar{\gamma}(E(H))$ is TRUE. It follows from $H \xrightarrow{c} C$ that $\bar{\gamma}(C) = \text{TRUE}$. Recall that $\gamma(C) = \text{FALSE}$. This is what we earlier referred to as the toggling property. Since γ and $\bar{\gamma}$ differ only in their assignment to variables in block B, clause C must contain at least one literal from B. □

The subgraph of G induced by the empty set of blocks clearly has a block-respecting independent set while the subgraph induced by all blocks does not. This motivates the following definition. Let $s + 1$ denote the minimum number of blocks such that some subgraph of G induced by $s + 1$ blocks does not have a block respecting independent set.

Definition 9.3. *The* sub-critical expansion *of G, denoted $e(G)$, is the maximum over all $t, 2 \leq t \leq s$, of the minimum boundary size of any subgraph H of G induced by t' blocks, where $t/2 < t' \leq t$.*

Lemma 9.4. *Any resolution refutation of $\alpha_{block}(G,k)$ must contain a clause of width at least $e(G)$.*

Proof. Let π be a resolution refutation of $\alpha_{block}(G,k)$. By Proposition 8.8, $\mu_G(\Lambda) = s+1$ and all initial clauses of $\alpha_{block}(G,k)$ have complexity at most 2. Therefore, for all $t, 2 \leq t \leq s$, there exists a clause C^t in π such that $\mu_G(C^t) > t$ and no ancestor of C^t has complexity greater than t.

Since $\mu_G(C^t) > 2$, C^t cannot be an initial clause. It must then be a resolvent of two parent clauses C_1^t and C_2^t. By Proposition 8.8 (c) and the fact that no ancestor of C^t has complexity greater than t, one of these clauses, say C_1^t, must have $\mu_G(C_1^t)$ between $(t+1)/2$ and t. If H^t is a block-induced subgraph that witnesses the value of $\mu_G(C_1^t)$, then by Lemma 9.2, $w_{width}^G(C_1^t) \geq |\beta(H^t)|$. Hence, $w(C_1^t) \geq |\beta(H^t)|$ so that π contains a clause of width at least $|\beta(H^t)|$.

When t is chosen such that $|\beta(H^t)|$ is maximized, we have by the definition of $e(G)$ that $|\beta(H^t)| \geq e(G)$. This finishes the proof. □

Corollary 9.5. *Let $c = 1/(9 \ln 2)$ and $k \,|\, n$. For any graph G with its n vertices partitioned into k blocks of size $b = n/k$ each,*

$$\begin{aligned} \mathsf{RES}(\alpha_{block}(G,k)) &\geq 2^{c(e(G)-b)^2/n} \quad \textit{and} \\ \mathsf{DPLL}(\alpha_{block}(G,k)) &\geq 2^{e(G)-b}. \end{aligned}$$

Proof. This follows immediately from Lemma 9.4 and Propositions 2.2 and 2.1 by observing that the initial width of $\alpha_{block}(G,k)$ is b. □

9.2. Lower Bounding Sub-critical Expansion. Throughout this section, the probabilities are with respect to the random choice of a graph G from the distribution $\mathbb{G}(n,p)$ for some fixed parameters n and p. Let $B(G)$ be a block graph corresponding to G with block size b.

For the rest of this paper, we will fix b to be 3, which corresponds to the largest independent set size ($k = n/3$) for which the results in this section hold. Although the results can be generalized to any $b \geq 3$, our best bounds are obtained for the simpler case of $b = 3$ that we present. Note that for $b = 2$, all clauses in $\alpha_{block}(G, n/2)$ are of length two. Consequently, when unsatisfiable, $\alpha_{block}(G, n/2)$ has a resolution refutation of size $O(n^2)$, ruling out any super-polynomial lower bounds.

Definition 9.6. *$B(G)$ is (r, q)-*dense *if some subgraph of G induced by r blocks (i.e., some subgraph of $B(G)$ with r nodes) contains at least q edges.*

The following lemma shows that for almost all random graphs G, the corresponding block graph $B(G)$ is locally sparse.

Lemma 9.7. *Let $G \sim \mathbb{G}(n, p)$ and $B(G)$ be a corresponding block graph with block size 3. For $r, q \geq 1$,*

$$\Pr[B(G) \textit{ is } (r, q)\textit{-dense}] < \left(\frac{ne}{3r}\right)^r \left(\frac{9er^2p}{2q}\right)^q .$$

Proof. Let H be a subgraph of G induced by r blocks. H contains $3r$ vertices. For $G \sim \mathbb{G}(n, p)$, the number of edges contained in H has the binomial distribution with parameters $\binom{3r}{2}$ and p. Therefore,

$$\Pr\left[H \text{ has at least } q \text{ edges}\right] \leq \binom{\binom{3r}{2}}{q} p^q < \binom{\frac{9r^2}{2}}{q} p^q \leq \left(\frac{9er^2p}{2q}\right)^q .$$

Summing this over all $\binom{n/3}{r} \leq (ne/3r)^r$ subgraphs H induced by r blocks gives the desired bound. □

We use this local sparseness property of the block graphs of almost all random graphs to prove that the smallest block-induced subgraph one needs to consider for proving that G does not have a block-respecting independent set is almost surely large.

Lemma 9.8. *There is a constant C such that the following holds. Let $\Delta = np$ and $s < Cn/\Delta^3$. The probability that $G \sim \mathbb{G}(n, p)$ contains a subgraph induced by at most s blocks that has no block-respecting independent set is $o(1)$ in s.*

Proof. The probability that G contains a subgraph induced by at most s blocks that has no block-respecting independent set is the same as the probability that there is some *minimal* subgraph H of G induced by $r \leq s$ blocks that has no block-respecting independent set. By minimality, H has no isolated vertices and hence no boundary blocks. Consequently, each of the r blocks that induce H must have at least 3 inter-block edges. Hence, the subgraph of $B(G)$ with the r nodes corresponding to the r blocks that induce H must have at least $3r/2$ edges.

Thus, the probability that G contains such a block-induced subgraph H is at most

$$\sum_{r=1}^{s} \Pr[B(G) \text{ is } (r, 3r/2)\text{-dense}].$$

By Lemma 9.7, we have $\Pr[B(G)$ is $(r, 3r/2)$-dense$] < D(r)$ where

$$\begin{aligned} D(r) &= \left(\frac{ne}{3r}\right)^r (3erp)^{3r/2} \\ &= \left(\frac{ne}{3}(3ep)^{3/2}\, r^{1/2}\right)^r \\ &= \left(Q(n,p)\; r^{1/2}\right)^r \end{aligned}$$

for $Q(n,p) = (ne/3)(3ep)^{3/2}$. Now

$$\begin{aligned} \frac{D(r+1)}{D(r)} &= \frac{\left(Q(n,p)\;(r+1)^{1/2}\right)^{r+1}}{\left(Q(n,p)\; r^{1/2}\right)^r} \\ &= Q(n,p)\;(r+1)^{1/2}\;\left(\frac{r+1}{r}\right)^{r/2} \\ &\leq Q(n,p)\;(r+1)^{1/2}\; e^{1/2} \\ &\leq \frac{ne}{3}\left(\frac{3e\Delta}{n}\right)^{3/2} e^{1/2}(r+1)^{1/2} \\ &= \left(\frac{3e^6\Delta^3(r+1)}{n}\right)^{1/2} \end{aligned}$$

This quantity, and hence $D(r+1)/D(r)$, is at most $1/2$ for $1 \leq r < Cn/\Delta^3$, where $C \stackrel{\text{def}}{=} 1/(12e^6)$ is a constant. Let $s+1 = Cn/\Delta^3$. It follows that the probability that G contains such a block-induced subgraph H is bounded above by a geometric series in r with common ratio $1/2$. It is therefore at most twice the largest term of the series which is less than $D(1)$. Now

$$D(1) \;=\; Q(n,p) \;=\; \frac{ne}{3}\left(\frac{3e\Delta}{n}\right)^{3/2} \;=\; \left(\frac{3e^5\Delta^3}{n}\right)^{1/2} \;=\; \left(\frac{3e^5C}{s+1}\right)^{1/2}.$$

Therefore, $D(1)$ is $o(1)$ in s as claimed. □

We again use the local sparseness property to prove that for almost all random graphs, any subgraph induced by not too many blocks must have a large boundary. The intuition is that for sparse subgraphs, most blocks have degree less than 3 and thus belong to the boundary.

Lemma 9.9. *There is a constant c such that the following holds. Let $\Delta = np$, $0 < \epsilon \leq 1/6$, $b' = 3(1-\epsilon)$, $t \leq cn/\Delta^{\frac{b'}{b'-2}}$, and $G \sim \mathbb{G}(n,p)$. The probability that there exists $r \in (t/2, t]$ such that G has a subgraph H induced by r blocks with $\beta(H) \leq \epsilon r$ is $o(1)$ in t.*

Proof. Fix b', ϵ, and t satisfying the conditions of the lemma. Let H be a subgraph of G induced by r blocks. By definition, all r blocks inducing H must have non-zero degree in $B(H)$. Moreover, if H has at most ϵr boundary blocks, the other $(1-\epsilon)r$ blocks of non-zero degree inducing it must have degree at least 3. Hence, the r nodes of $B(G)$ that induce H form a subgraph with at least $(1-\epsilon)r3/2 = b'r/2$ edges. Therefore, H has at most ϵr boundary blocks only if $B(G)$ is $(r, b'r/2)$-dense. Thus, by Lemma 9.7, the probability that such an H exists is at most

$$\begin{aligned} \Pr[B(G) \text{ is } (r, b'r/2)\text{-dense}] &< \left(\frac{ne}{3r}\right)^r \left(\frac{9erp}{b'}\right)^{b'r/2} \\ &= \left(\frac{e}{3}\left(\frac{3e\Delta}{1-\epsilon}\right)^{b'/2} \left(\frac{r}{n}\right)^{(b'-2)/2}\right)^r \end{aligned}$$

For $r > t/2$, it suffices to obtain an upper bound on this probability that is exponentially small in r. Rearranging the terms in the expression above, $\Pr[B(G) \text{ is } (r, b'r/2)\text{-dense}] \leq 2^{-r}$ when

$$\begin{aligned} \frac{r}{n} &\leq \left(\frac{3}{2e}\right)^{2/(b'-2)} \left(\frac{1-\epsilon}{3e\Delta}\right)^{b'/(b'-2)} \\ &= \left(\frac{1-\epsilon}{2e^2}\right)^{2/(b'-2)} \frac{1-\epsilon}{3e\Delta^{b'/(b'-2)}}. \end{aligned}$$

Note that $\epsilon \leq 1/6$ and $b' = 3(1-\epsilon) \geq 5/2$. Hence $(1-\epsilon)/(2e^2)^{2/(b'-2)}$ is at least $(5/(12e^2))^4$ and it suffices to have

$$\frac{r}{n} \leq \left(\frac{5}{12e^2}\right)^4 \frac{5}{18e\Delta^{b'/(b'-2)}} = \frac{c}{\Delta^{b'/(b'-2)}}$$

for a constant $c \overset{\text{def}}{=} 5^5/(12^4 18e^9)$. Therefore, the probability that $B(G)$ is $(r, b'r)$-dense is at most 2^{-r} for $r \leq cn/\Delta^{b'/(b'-2)}$. It follows that the probability that there exists such an H with $r \in (t/2, t]$ is at most $\sum_{r=\lceil (t+1)/2 \rceil}^{t} 2^{-r}$. This sum is $o(1)$ in t as required. □

Lemmas 9.8 and 9.9 combine to give the following lower bound on subcritical expansion:

Lemma 9.10. *For each $\epsilon \in (0, 1/6]$ there is a constant c_ϵ such that the following holds. Let $\Delta = np$, $b' = 3(1-\epsilon)$, $W = n/\Delta^{b'/(b'-2)}$, and $G \sim \mathbb{G}(n,p)$. The probability that $e(G) < c_\epsilon W$ is $o(1)$ in W.*

Proof. Let C be the constant from Lemma 9.8 and c be the one from Lemma 9.9. Let $s+1$ be the minimum number of blocks such that some subgraph of G induced by $s+1$ blocks does not have a block-respecting independent set. By Lemma 9.8, $s \leq Cn/\Delta^3$ with probability $o(1)$ in n. Now let $t = \min(C,c)W$. Conditioned on $s > Cn/\Delta^3$ and because $b' < 3$, we have that $t \leq s$ as in the definition of $e(G)$. By Lemma 9.9, the probability that some subgraph of G induced by r blocks with $t/2 < r \leq t \leq s$ has less than $\epsilon r > \epsilon t/2 = c_\epsilon W$ boundary blocks is $o(1)$ in n, where $c_\epsilon = (\epsilon/2)\min(C,c)$. It follows from a union bound on the two bad events (s is small or some subgraph has small boundary) that $e(G) < c_\epsilon W$ with probability $o(1)$ in n. □

10. Lower Bounds for Resolution and Associated Algorithms

We now use the ideas developed in Sections 8 and 9, and bring the pieces of the argument together in a general technical result from which our resolution complexity lower bounds follow.

Lemma 10.1. *For each $\delta > 0$ there are constants $C_\delta, C'_\delta > 0$ such that the following holds. Let $\Delta = np$ and $G \sim \mathbb{G}(n,p)$. With probability $1-o(1)$ in n,*

$$\begin{aligned} \mathsf{RES}(\alpha_{block}(G,n/3)) &\geq 2^{C_\delta n/\Delta^{6+2\delta}} \quad \textit{and} \\ \mathsf{DPLL}(\alpha_{block}(G,n/3)) &\geq 2^{C'_\delta n/\Delta^{3+\delta}}. \end{aligned}$$

Proof. Observe that the expressions $n/\Delta^{6+2\delta}$ and $n/\Delta^{3+\delta}$ in the desired bounds decrease as δ increases. Hence, it suffices to prove the bounds for $\delta \in (0,2]$, and for $\delta > 2$, simply let $C_\delta = C_2$ and $C'_\delta = C'_2$.

Let $\epsilon = \delta/(6+3\delta)$, $b' = 3(1-\epsilon)$, and $W = n/\Delta^{b'/(b'-2)}$. For $\delta \in (0,2]$, we have that $\epsilon \in (0,1/6]$. From Lemma 9.10, there is a constant c_ϵ such that with probability $1-o(1)$ in n, $e(G) \geq c_\epsilon W$. It follows from Corollary 9.5 that for $c = 1/(9\ln 2)$ and with probability $1-o(1)$ in n,

$$\begin{aligned} \mathsf{RES}(\alpha_{block}(G,n/3)) &\geq 2^{c(c_\epsilon W-3)^2/n} \quad \text{and} \\ \mathsf{DPLL}(\alpha_{block}(G,n/3)) &\geq 2^{c_\epsilon W-3}. \end{aligned}$$

Given the relationship between ϵ and δ, there are constants $C_\delta, C'_\delta > 0$ depending only on δ such that $c(c_\epsilon W-3)^2 \geq C_\delta W^2$ and $c_\epsilon W - 3 \geq C'_\delta W$. Note also that $b'/(b'-2) = (3-3\epsilon)/(1-3\epsilon) = 3+\delta$. Hence,

$$\log_2(\mathsf{RES}(\alpha_{block}(G,n/3))) \;\geq\; C_\delta W^2/n \;=\; C_\delta n/\Delta^{\frac{2b'}{b'-2}} \;=\; C_\delta n/\Delta^{6+2\delta} \quad \text{and}$$

$$\log_2(\mathsf{DPLL}(\alpha_{block}(G, n/3))) \;\geq\; C'_\delta W \;=\; C'_\delta n/\Delta^{\frac{b'}{b'-2}} \;=\; C'_\delta n/\Delta^{3+\delta}.$$

This finishes the proof. □

Theorem 10.2 (Independent Set Lower Bounds). *For each $\delta > 0$ there are constants C_δ, C'_δ, C''_δ, C'''_δ, $C''''_\delta > 0$ such that the following holds. Let $\Delta = np$, $k \leq n/3$, $k \,|\, n$, and $G \sim \mathbb{G}(n, p)$. With probability $1 - o(1)$ in n,*

$$\begin{aligned}
\mathsf{RES}(\alpha_{map}(G,k)) &\;\geq\; 2^{C_\delta n/\Delta^{6+2\delta}}, \\
\mathsf{DPLL}(\alpha_{map}(G,k)) &\;\geq\; 2^{C'_\delta n/\Delta^{3+\delta}}, \\
\mathsf{RES}(\alpha_{count}(G,k)) &\;\geq\; 2^{C''_\delta n/\Delta^{6+2\delta}}, \\
\mathsf{DPLL}(\alpha_{count}(G,k)) &\;\geq\; 2^{C'''_\delta n/\Delta^{3+\delta}}, \\
\mathsf{RES}(\alpha_{block}(G,k)) &\;\geq\; 2^{C_\delta n/\Delta^{6+2\delta}}, \\
\mathsf{DPLL}(\alpha_{block}(G,k)) &\;\geq\; 2^{C'_\delta n/\Delta^{3+\delta}}, \quad \textit{and} \\
Chv(G,k) &\;\geq\; 2^{C''''_\delta n/\Delta^{6+2\delta}}.
\end{aligned}$$

The bounds for the block encoding require $k \,|\, (n/3)$.

Proof. All of the claimed bounds follow by applying monotonicity of the encoding at hand, using its relationship with the block encoding, and applying Lemma 10.1. Let C_δ and C'_δ be the constants from Lemma 10.1. For the mapping-based encoding,

$$\begin{aligned}
\mathsf{RES}(\alpha_{map}(G,k)) &\geq \mathsf{RES}(\alpha_{map}(G,n/3)) && \text{by Lemma 4.2} \\
&\geq \mathsf{RES}(\alpha_{block}(G,n/3)) && \text{by Lemma 4.5} \\
&\geq 2^{C_\delta n/\Delta^{6+2\delta}} && \text{by Lemma 10.1,}
\end{aligned}$$

$$\begin{aligned}
\mathsf{DPLL}(\alpha_{map}(G,k)) &\geq \mathsf{DPLL}(\alpha_{map}(G,n/3)) && \text{by Lemma 4.2} \\
&\geq \mathsf{DPLL}(\alpha_{block}(G,n/3)) && \text{by Lemma 4.5} \\
&\geq 2^{C'_\delta n/\Delta^{3+\delta}} && \text{by Lemma 10.1.}
\end{aligned}$$

For the counting-based encoding,

$$\begin{aligned}
\mathsf{RES}(\alpha_{count}(G,k)) &\geq \frac{1}{n}\left(\mathsf{RES}(\alpha_{count}(G,n/3)) - 2n^2\right) && \text{by Lemma 4.1} \\
&\geq \frac{1}{n}\left(\frac{1}{2}\mathsf{RES}(\alpha_{block}(G,n/3)) - 2n^2\right) && \text{by Lemma 4.4} \\
&\geq 2^{C''_\delta n/\Delta^{6+2\delta}} && \text{by Lemma 10.1}
\end{aligned}$$

for a large enough constant C''_δ. Similarly,

$$\begin{aligned}\mathsf{DPLL}(\alpha_{count}(G,k)) &\geq \frac{1}{n}\left(\mathsf{DPLL}(\alpha_{count}(G,n/3)) - 2n^2\right) && \text{by Lemma 4.1}\\ &\geq \frac{1}{n}\left(\frac{1}{2}\mathsf{DPLL}(\alpha_{block}(G,n/3))^{1/\log_2 6} - 2n^2\right) && \text{by Lemma 4.4}\\ &\geq 2^{C'''_\delta n/\Delta^{3+\delta}}\end{aligned}$$

by Lemma 10.1 for a large enough constant C'''_δ.

The bounds for the block encoding follow immediately from Lemmas 4.3 and 10.1. Finally, for the bound on the proof size in Chvátal's system,

$$\begin{aligned}Chv(G,k) &\geq Chv(G,n/3) - 1 && \text{by Proposition 5.1}\\ &\geq \frac{1}{4n}\mathsf{RES}(\alpha_{block}(G,n/3)) - 1 && \text{by Lemma 5.3}\\ &\geq 2^{C''''_\delta n/\Delta^{6+2\delta}} && \text{by Lemma 10.1}\end{aligned}$$

for a large enough constant c''''_δ. □

Corollary 10.3 (Vertex Cover Lower Bounds). *For each $\delta > 0$ there are constants $\widetilde{C}_\delta, C_\delta, C'_\delta > 0$ such that the following holds. Let $\Delta = np$, $t \geq 2n/3$, $(n-t) \mid n$, and $G \sim \mathbb{G}(n,p)$. With probability $1 - o(1)$ in n,*

$$\begin{aligned}\mathsf{RES}(VC_{count}(G,t)) &\geq 2^{\widetilde{C}_\delta n/\Delta^{6+2\delta}},\\ \mathsf{RES}(VC_{block}(G,t)) &\geq 2^{C_\delta n/\Delta^{6+2\delta}}, \quad \textit{and}\\ \mathsf{DPLL}(VC_{block}(G,t)) &\geq 2^{C'_\delta n/\Delta^{3+\delta}}.\end{aligned}$$

The bounds for the block encoding require $(n-t) \mid (n/3)$.

Proof. Let C_δ, C'_δ, and C''_δ be the constants from Theorem 10.2 and let $\widetilde{C}_\delta$ be any constant less than C''_δ. For the counting encoding bound, apply Theorem 10.2 with k set to $(n-t)$ and use Lemma 6.2 to translate the results to the encoding of vertex cover. For the block encoding bounds, apply Theorem 10.2 in conjunction with Lemma 6.3. □

11. Hardness of Approximation

Instead of considering the decision problem of whether a given graph G has an independent set of a given size k, one may consider the related *optimization* *problem*: given G, find an independent set in it of the largest possible size. We call this optimization problem the *maximum independent set* problem. One may similarly define the *minimum vertex cover* optimization problem.

Since the decision versions of these problems are NP-complete, the optimization versions are NP-hard and do not have any known polynomial-time solutions. From the perspective of algorithm design, it is then natural to ask whether there is an efficient algorithm that finds an independent set of size "close" to the largest possible or a vertex cover of size close to the smallest possible. That is, is there an efficient algorithm that finds an "approximate" solution to the optimization problem? In this section, we rule out the existence of any such efficient "resolution-based" approximation algorithms, for appropriately defined measures of closeness.

We note that the notion of approximation we are using in this paper is the natural one for the optimization versions of the two problems under consideration. Specifically, on input G, an approximation algorithm for the independent set problem will find an independent set of a size often *smaller* than the maximum independent set size $M(G)$. An alternative view of approximation is to consider procedures that, given G, provide short proofs of non-existence of independent sets of a size often *larger* than $M(G)$. While this second notion of approximation fits proof systems, it does not relate well to the standard notion of approximation algorithms for optimization problems and makes our bounds incomparable with known hardness-of-approximation results for the two problems. We, therefore, choose the former notion, i.e., approximation with respect to finding an optimal independent set or vertex cover.

REMARK 11.1. *The results we prove in this section contrast well with the known approximation hardness results for the two problems which are all based on the PCP (probabilistically checkable proofs) characterization of NP (Arora* et al. *1998; Arora & Safra 1998). Håstad (1999) showed that unless P = NP, there is no polynomial-time $n^{1-\epsilon}$-approximation algorithm for the clique (and hence the independent set) problem for any $\epsilon > 0$. For graphs with maximum degree Δ_{max}, Trevisan (2001) improved this to a factor of $\Delta_{max}/2^{O(\sqrt{\ln \Delta_{max}})}$. More recently, Dinur & Safra (2005) proved that unless P = NP, there is no polynomial-time $10\sqrt{5} - 21 \approx 1.36$ factor approximation algorithm for the vertex cover problem. Our results, on the other hand, hold irrespective of the relationship between P and NP, but apply only to the class of resolution-based algorithms to be defined shortly. Further, they provide an exponential lower bound not only for the worst-case input to a resolution-based algorithm but for nearly all input graphs of small enough density.*

11.1. Maximum Independent Set Approximation. We begin by making several of the above notions precise. Let A be an algorithm for finding a maximum independent set in a given graph.

Definition 11.2. *Let $\gamma \geq 1$. A is a* γ-approximation algorithm *for the maximum independent set problem if on input G with maximum independent set size $\hat{k}$, A produces an independent set of size at least $\hat{k}/\gamma$.*

In other words, if A produces an independent set of size $\bar{k}$ on input G, it proves that G does not have one of size $\bar{k}\gamma + 1$. This reasoning allows us to use our lower bounds from the previous section to prove that even approximating a maximum independent set is exponentially hard for certain resolution-based algorithms.

Definition 11.3. *A γ-approximation algorithm A for the maximum independent set problem is* resolution-based *if it has the following property: if A outputs an independent set of size $\bar{k}$ on input G, then its computation history along with a proof of correctness within a factor of γ yields a resolution proof of $\alpha_{map}(G, k)$ or $\alpha_{count}(G, k)$ for $k \leq \bar{k}\gamma + 1, k \,|\, n$.*

The manner in which the computation history and the proof of correctness are translated into a resolution refutation of an appropriate encoding depends on specific details of the approximation algorithm and varies with context. We will shortly see a nearly trivial example of this when proving Proposition 11.4.

Let $\mathcal{A}_{\gamma}^{RES-ind}$ denote the class of all resolution-based γ-approximation algorithms for the maximum independent set problem. We show that while there is a trivial algorithm in this class for $\gamma \geq \Delta + 1$, there isn't an efficient one for $\gamma \leq \Delta/(6 \ln \Delta)$.

Proposition 11.4. *For $\gamma \geq \Delta + 1$, there is a polynomial-time algorithm in $\mathcal{A}_{\gamma}^{RES-ind}$.*

Proof. Let A be the polynomial-time algorithm that underlies the bound in Turan's theorem (Proposition 3.1), that is, on a graph G with n nodes and average degree Δ as input, A produces an independent set of size $\bar{k} \geq n/(\Delta+1)$. Since the size of a maximum independent set in G is at most n, A is a $(\Delta+1)$-approximation algorithm. We will argue that A is also resolution-based.

To be resolution-based, the computation history of A on G along with a proof of correctness within a factor of $(\Delta + 1)$ must yield a resolution proof of a suitable encoding $\alpha(G, k)$ for some $k \leq k^* = \bar{k}(\Delta + 1) + 1, k \,|\, n$. When G has no edges, $\Delta = 0$ and A produces an independent set of size $\bar{k} = n$. In this case, there is nothing to prove. When G has at least one edge (u, v), $k^* \geq n+1$ and we can choose $k = n$. In this case, A indeed yields a straightforward resolution proof of $\alpha(G, k)$ for both the mapping and the counting encodings by utilizing the edge clause(s) corresponding to (u, v). Therefore, A is resolution-based as a $(\Delta + 1)$-approximation algorithm. □

While Proposition 3.2 guarantees that there is almost never an independent set of size larger than $(2n/\Delta)\ln\Delta$, Theorem 10.2 shows that there is no efficient way to prove this fact using resolution. Indeed, there exist efficient resolution proofs only for the non-existence of independent sets of size larger than $n/3$. We use this reasoning to prove the following hardness of approximation result.

Theorem 11.5 (Independent Set Approximation). *There is a constant c such that the following holds. Let $\delta > 0$, $\Delta = np$, $\Delta \geq c$, $\gamma \leq \Delta/(6\ln\Delta)$, and $G \sim \mathbb{G}(n, p)$. With probability $1 - o(1)$ in n, every algorithm $A \in \mathcal{A}_\gamma^{RES-ind}$ takes time exponential in $n/\Delta^{6+2\delta}$.*

Proof. Recall the definitions of $k_{+\epsilon}$ and C_ϵ from Proposition 3.2. Fix $\epsilon > 0$ such that $k_{+\epsilon} < (2n/\Delta)\ln\Delta$ and let $c \geq C_\epsilon$. The claimed bound holds trivially for $\Delta \geq n^{1/6}$. We will assume for the rest of the proof that $C_\epsilon \leq \Delta \leq n/\ln^2 n$.

From Proposition 3.2, with probability $1 - o(1)$ in n, a maximum independent set in G is of size $k_{max} \leq k_{+\epsilon} < (2n/\Delta)\ln\Delta$. If A approximates this within a factor of γ, then, in particular, it proves that G does not have an independent set of size $k = k_{max}\gamma + 1 \leq n/3$. Convert the transcript of the computation of A on G along with an argument of its correctness within a factor of γ into a resolution proof π of an appropriate encoding $\alpha(G, k)$. From Theorem 10.2, $size(\pi)$ must be exponential in $n/\Delta^{6+2\delta}$. □

11.2. Minimum Vertex Cover Approximation.

A similar reasoning can be applied to approximation algorithms for finding a minimum vertex cover.

Definition 11.6. *Let $\gamma \geq 1$. A is a γ-*approximation algorithm *for the minimum vertex cover problem if on input G with minimum vertex cover size $\hat{t}$, A produces a vertex cover of size at most $\hat{t}\gamma$.*

Definition 11.7. *A γ-approximation algorithm A for the minimum vertex cover problem is* resolution-based *if it has the following property: if A outputs a vertex cover of size $\bar{t}$ on input G, then its computation history along with a proof of correctness within a factor of γ yields a resolution proof of $VC_{count}(G, t)$ for $t \geq \bar{t}/\gamma - 1, (n - t) \mid n$.*

Let $\mathcal{A}_\gamma^{RES-VC}$ denote the class of all resolution-based γ-approximation algorithms for the minimum vertex cover problem. We prove that there is no polynomial-time resolution-based algorithm for the vertex cover problem that achieves an approximation factor better than 3/2.

Theorem 11.8 (Vertex Cover Approximation). *There is a constant c such that the following holds. Let $\delta > 0$, $\Delta = np$, $\Delta \geq c$, $\gamma < 3/2$, and $G \sim \mathbb{G}(n, p)$. With probability $1 - o(1)$ in n, every algorithm $A \in \mathcal{A}_\gamma^{RES-VC}$ takes time exponential in $n/\Delta^{6+2\delta}$.*

Proof. This proof is very similar to that of Theorem 11.5. Recall the definitions of $k_{+\epsilon}$ and C_ϵ from Proposition 3.2. Fix $\epsilon > 0$ such that $k_{+\epsilon} < (2n/\Delta) \ln \Delta$, and let $c \geq C_\epsilon$. The claimed bound holds trivially for $\Delta \geq n^{1/6}$. We will assume for the rest of the proof that $C_\epsilon \leq \Delta \leq n/\ln^2 n$.

From Proposition 3.2 and the relation between independent sets and vertex covers, with probability $1 - o(1)$ in n, a minimum vertex cover in G is of size $t_{min} \geq n - k_{+\epsilon} > n - (2n/\Delta) \ln \Delta$. If A approximates this within a factor of γ, then, in particular, it proves that G does not have a vertex cover of size $t = t_{min}/\gamma - 1 \geq 2n/3$. Convert the transcript of A's computation on G along with an argument of its correctness within a factor of γ into a resolution proof π of $VC_{count}(G, t)$. By Corollary 10.3, $size(\pi)$ must be exponential in $n/\Delta^{6+2\delta}$. □

We end this section with the remark that if $t = n/2$ and the block encoding $VC_{block}(G, n/2)$ were constructed with blocks of size two rather than three, one could obtain (in polynomial time) a polynomial size refutation of the resulting formula when unsatisfiable. This is because, as pointed out in the beginning of Section 9.2, all clauses of this formula would be of length two. This suggests that one might be able to obtain a resolution-based 2-appximation algorithm for the vertex cover problem. However, a short proof of $VC_{block}(G, n/2)$ may not necessarily translate into a short proof of $VC_{count}(G, n/2)$, which is what we really need for an upper bound.

12. Stronger Lower Bounds for Exhaustive Backtracking Algorithms and DPLL

We conclude the technical sections of this paper with a stronger lower bound for a natural class of backtracking algorithms for the independent set and the vertex cover problems, namely the class of exhaustive backtracking search algorithms. The key difference between the algorithms captured by resolution that we have considered so far and the ones in this class is that the latter do *not* reuse computation performed for previous branches; instead, they systematically rule out all potential independent sets or vertex covers of the desired size by a possibly smart but nonetheless exhaustive search. As an illustration, we will give an example of a non-trivial exhaustive backtracking algorithm for the independent set problem shortly.

The argument for our lower bound is based on the density of independent sets and vertex covers in random graphs and is quite straightforward in the light of Lemma 3.3. We derive as a consequence a tighter lower bound for the DPLL complexity of the mapping and the counting encodings of the two problems, which allows the edge density in the underlying graph to be much higher than in Theorem 10.2 and Corollary 10.3.

Returning to the class of exhaustive backtracking algorithms, recall that the approach we used for our upper bounds (cf. Section 7) was to systematically rule out all potential independent sets of a certain size $k' = k_{min}$. This is the simplest algorithm in the class. Of course, instead of simply considering all $\binom{n}{k'}$ subsets of vertices of size k' as we did, one can imagine more complex techniques for exhaustive search. For instance, an idea similar to the one used by Beame *et al.* (2005) for the graph coloring problem would be to consider all subsets of size $u < k'$ in the first stage. For a random graph, most of these subsets are very likely to already contain an edge and need not be processed further. For any remaining subset S, one can recursively refute the existence of an independent set of size $k' - u$ in the residual graph with $|n - k - N(S)|$ vertices, where $N(S)$ denotes all neighbors of S outside S. This is also an exhaustive backtracking algorithm.

Such algorithms may require a more complex analysis than we gave in our upper bound proofs and could potentially be more efficient. However, as the following result shows, any technique that systematically rules out all possible k'-independent sets by an exhaustive backtracking search cannot improve the relatively simple upper bounds in Theorem 7.3 and Corollary 7.4 by more than a constant factor in the exponent.

Let $\mathcal{A}^{ind}_{exhaustive}$ (or $\mathcal{A}^{VC}_{exhaustive}$) denote the class of backtracking algorithms for proving non-existence of independent sets (vertex covers, resp.) of a given size in a given graph, that work by recursively subdividing the problem based on whether or not a set of vertices is included in the independent set (vertex cover, resp.) and that do not reuse computation performed in previous branches. For example, our approach in Section 7 as well as the more complex two-phase algorithm sketched above, both belong to $\mathcal{A}^{ind}_{exhaustive}$. We show that such algorithms must almost always branch exponentially often.

Theorem 12.1 (Exhaustive Backtracking Algorithms). *There are constants C and c such that the following holds. Let $\Delta = np$, $c \leq \Delta \leq n/\ln^2 n$, and $G \sim \mathbb{G}(n,p)$. With probability $1 - o(1)$ in n, every algorithm $A \in \mathcal{A}^{ind}_{exhaustive}$ (or $\mathcal{A}^{VC}_{exhaustive}$) running on input (G,k) must branch at least $2^{C(n/\Delta)\ln^2 \Delta}$ times when G does not have an independent set (vertex cover, resp.) of size k.*

Proof. Let C be the constant from Lemma 3.3. Recall the definitions of $k_{+\epsilon}$ and C_ϵ from Proposition 3.2. Fix $\epsilon > 0$ such that $k_{+\epsilon}+1 > (2n/\Delta)\ln\Delta$, and let $c \geq C_\epsilon$. With probability $1 - o(1)$ in n, any algorithm $A \in \mathcal{A}^{ind}_{exhaustive}$ succeeds in proving the non-existence of a k-independent set in G only when $k \geq k_{+\epsilon}+1$. However, Lemma 3.3 says that G almost surely contains at least $2^{C(n/\Delta)\ln^2 \Delta}$ independent sets of size $k^* = \lfloor (n/\Delta)\ln\Delta \rfloor$, which is less than $(k_{+\epsilon}+1)/2$, and hence less than k. As a result, while recursively subdividing the problem based on whether or not to include a vertex in the k-independent set, A must explore at least $2^{C(n/\Delta)\ln^2 \Delta}$ distinct k^*-independent sets before finding a contradictory edge for each and backtracking. This proves the desired lower bound on the number of times algorithm A must branch.

For the vertex cover case, note that the algorithms in $\mathcal{A}^{VC}_{exhaustive}$ are the duals of the algorithms in $\mathcal{A}^{ind}_{exhaustive}$; including a vertex in a vertex cover to create a smaller subproblem is equivalent to not including it in an independent set. Further, the number of vertex covers of size $(n-k)$ in G is exactly the same as the number of independent sets of size k in G. Hence, the above lower bound applies to the algorithms in $\mathcal{A}^{VC}_{exhaustive}$ as well. □

Theorem 12.2 (Stronger DPLL Lower Bounds). *There are constants C and c such that the following holds. Let $\Delta = np$, $c \leq \Delta \leq n/(2\ln^2 n)$, and $G \sim \mathbb{G}(n,p)$. With probability $1 - o(1)$ in n,*

$$\begin{aligned} \mathsf{DPLL}(\alpha_{map}(G,k)) &\geq 2^{C(n/\Delta)\ln^2\Delta}, \\ \mathsf{DPLL}(\alpha_{count}(G,k)) &\geq 2^{C(n/\Delta)\ln^2\Delta}, \\ \mathsf{DPLL}(VC_{map}(G,t)) &\geq 2^{C(n/\Delta)\ln^2\Delta}, \quad \textit{and} \\ \mathsf{DPLL}(VC_{count}(G,t)) &\geq 2^{C(n/\Delta)\ln^2\Delta}. \end{aligned}$$

Proof. The DPLL complexity of the encodings, by our convention, is ∞ if G does have an independent set of size k. If it does not, the tree T associated with any DPLL refutation of $\alpha_{map}(G,k)$ or $\alpha_{count}(G,k)$ can be viewed as the trace of an exhaustive backtracking algorithm $A \in \mathcal{A}^{ind}_{exhaustive}$ on input (G,k) as follows. A non-leaf node in T representing a resolution step (equivalently, a

DPLL branch) on variable x_v corresponds to the decision of A to branch based on whether or not to include vertex v in the independent set it is creating. Nodes in T representing a resolution step on the counting variables represent the counting process of A.

Given this correspondence, Theorem 12.1 immediately implies the desired lower bounds for the independent set problem. The results for the vertex cover problem can be derived in an analogous manner. Note that refuting the block encoding may be easier than ruling out all independent sets (vertex covers, resp.) of size k. Hence, Theorem 12.1 does not translate into a bound for this encoding. □

We see that these lower bounds match the upper bounds in Theorem 7.3 and Corollary 7.4 up to a constant factor in the exponent.

13. Discussion

In this paper, we used a combination of combinatorial and probabilistic arguments to obtain lower and upper bounds on the resolution complexity of several natural CNF encodings of the independent set, vertex cover, and clique problems. Our results hold almost surely when the underlying graph is chosen at random from the $\mathbb{G}(n,p)$ model. Consequently, they hold (deterministically) for nearly all graphs. A key step in the main lower bound arguments was to simplify the task by considering the induced block graph in place of the original graph. The expansion properties of the block graph then allowed us to relate refutation width with structural properties of the graph.

Our results imply exponential lower bounds on the running time of resolution-based backtracking algorithms for finding a maximum independent set (or, equivalently, a maximum clique or a minimum vertex cover) in a given graph. Such algorithms include some of the best known ones for these combinatorial problems (Jian 1986; Shindo & Tomita 1990; Tarjan 1972; Tarjan & Trojanowski 1977). For graphs of low enough density, our lower bounds for these algorithms hold for nearly all input graphs, not only in the worst case.

A noteworthy contribution of this work is the hardness of approximation result. We showed unconditionally that there is no polynomial-time resolution-based approximation algorithm that guarantees a solution within a factor less than $\Delta/(6 \ln \Delta)$ for the maximum independent set problem or within a factor less than $3/2$ for the minimum vertex cover problem. This complements the hardness results conditioned on $\mathrm{P} \neq \mathrm{NP}$ that rule out efficient approximations within factors of $\Delta_{max}/2^{O(\sqrt{\ln \Delta_{max}})}$ (Trevisan 2001) and $10\sqrt{5} - 21 \approx 1.36$

(Dinur & Safra 2005) for the two problems, respectively. (Here Δ_{max} denotes the maximum degree of the underlying graph rather than the average degree.)

On the flip side, some algorithms, such as those of Robson (1986), Beigel (1999), Chen *et al.* (2001), and Tomita & Seki (2003), employ techniques that do not seem to be captured by resolution. The techniques they use, such as unrestricted without loss of generality arguments (Robson 1986), vertex folding (Chen *et al.* 2001), creation of new vertices (Beigel 1999), and pruning of search space using approximate coloring (Tomita & Seki 2003), represent global properties of graphs or global changes therein that appear hard to argue locally using a bounded number of resolution inferences. For instance, the algorithm of Robson (1986) involves the reasoning that if an independent set contains only one element of $N(v)$, then without loss of generality, that element can be taken to be the vertex v itself. It is unclear how to model this behavior efficiently in resolution.

Restricted versions of these general properties, however, can indeed be simulated by resolution. This applies when one restricts, for instance, to vertices of small, bounded degree, as is done in many case-by-case algorithms cited earlier (Jian 1986; Shindo & Tomita 1990; Tarjan 1972; Tarjan & Trojanowski 1977).

It is also interesting and not too surprising that the spectral algorithm of Coja-Oghlan (2005) for the independent set problem achieves an $O(\sqrt{\Delta}/\ln\Delta)$-approximation in expected polynomial time and, in the light of our almost certain lower bounds, cannot be simulated by resolution.

We conclude with a few directions for extending the results in this paper. An analysis of the complexity of the independent set and other related problems under stronger proof systems, such as Cutting Planes (Bonet *et al.* 1997; Impagliazzo *et al.* 1994), bounded-depth Frege systems (Ajtai 1994), or an extension of resolution that allows "without loss of generality" reasoning as mentioned above, will further broaden our understanding of the inherent complexity of these graph problems.

An intriguing question is, how far can the hardness of approximation for resolution-based algorithms for the vertex cover problem be pushed? We showed that anything better than a 3/2-approximation is impossible using resolution-based techniques. It is unclear how this relates to the well-known greedy 2-approximation algorithm for the problem.

Finally, the DPLL upper bounds that we derived are based on a rather simple enumeration, with natural search space pruning, of all potential independent sets. As Theorem 12.1 points out, this is the best one can do using any exhaustive backtracking algorithm. Considering more complex techniques may let us close the gap of a factor of nearly $O(\Delta^5)$ in the exponent that currently exists between our lower and upper bounds for general resolution. It appears that we have not taken advantage of the full power of general resolution, specifically the reuse of derived clauses.

Acknowledgements

The authors would like to thank Mikhail Alekhnovitch for his comments, which led to simpler proofs and stronger results.

The work of the first and the third authors was supported by NSF Awards CCR-9800124 and CCR-0098066, and that of the second author was supported by NSF Award CCR-9734911. A preliminary version of this paper appeared in the 16th Annual IEEE Conference on Computational Complexity (Beame *et al.* 2001) and also in the third author's Ph.D. thesis (Sabharwal 2005).

References

Miklós Ajtai (1994). The Complexity of the Pigeonhole Principle. *Combinatorica* **14**(4), 417–433.

Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan & Márió Szegedy (1998). Proof Verification and Hardness of Approximation Problems. *Journal of the ACM* **45**(3), 501–555.

Sanjeev Arora & Shmuel Safra (1998). Probabilistic Checking of Proofs. *Journal of the ACM* **45**(1), 70–122.

Paul Beame, Joseph C. Culberson, David G. Mitchell & Christopher Moore (2005). The Resolution Complexity of Random Graph k-Colorability. *Discrete Applied Mathematics* **153**(1-3), 25–47.

Paul Beame, Russell Impagliazzo & Ashish Sabharwal (2001). Resolution Complexity of Independent Sets in Random Graphs. In *Proceedings Sixteenth Annual IEEE Conference on Computational Complexity*, 52–68. Chicago, IL.

Paul Beame, Richard M. Karp, Toniann Pitassi & Michael E. Saks (1998). On the Complexity of Unsatisfiability Proofs for Random k-CNF Formulas. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, 561–571. Dallas, TX.

Richard Beigel (1999). Finding Maximum Independent Sets in Sparse and General Graphs. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 856–857. Baltimore, MD.

ELI BEN-SASSON & AVI WIGDERSON (2001). Short Proofs are Narrow - Resolution Made Simple. *Journal of the ACM* **48**(2), 149–169.

BELA BOLLOBÁS (1985). *Random Graphs*. Academic Press, London.

MARIA LUISA BONET, TONIANN PITASSI & RAN RAZ (1997). Lower Bounds for Cutting Planes Proofs with Small Coefficients. *Journal of Symbolic Logic* **62**(3), 708–728.

JIANER CHEN, IYAD KANJ & WEIJIA JIA (2001). Vertex Cover: Further Observations and Further Improvements. *Journal of Algorithms* **41**(2), 280–301.

VASEK CHVÁTAL (1977). Determining the Stability Number of a Graph. *SIAM Journal on Computing* **6**(4), 643–662.

VASEK CHVÁTAL & ENDRE SZEMERÉDI (1988). Many Hard Examples for Resolution. *Journal of the ACM* **35**(4), 759–768.

MATTHEW CLEGG, JEFF EDMONDS & RUSSELL IMPAGLIAZZO (1996). Using the Gröbner Basis Algorithm to Find Proofs of Unsatisfiability. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, 174–183. Philadelphia, PA.

AMIN COJA-OGHLAN (2005). The Lovász Number of Random Graphs. *Combinatorics, Probability & Computing* **14**(4), 439–465.

WILLIAM COOK, COLLETTE R. COULLARD & GYÖRGY TURÁN (1987). On the Complexity of Cutting-Plane Proofs. *Discrete Applied Mathematics* **18**, 25–38.

STEFAN S. DANTCHEV & SØREN RIIS (2001). Tree Resolution Proofs of the Weak Pigeon-Hole Principle. In *Proceedings Sixteenth Annual IEEE Conference on Computational Complexity*, 69–75. Chicago, IL.

MARTIN DAVIS, GEORGE LOGEMANN & DONALD W. LOVELAND (1962). A Machine Program for Theorem Proving. *Communications of the ACM* **5**, 394–397.

MARTIN DAVIS & HILARY PUTNAM (1960). A Computing Procedure for Quantification Theory. *Communications of the ACM* **7**, 201–215.

IRIT DINUR & SHMUEL SAFRA (2005). On the Hardness of Approximating Minimum Vertex Cover. *Annals of Mathematics* **162**(1), 439–486.

ARMIN HAKEN (1985). The Intractability of Resolution. *Theoretical Computer Science* **39**, 297–305.

JOHAN HÅSTAD (1999). Clique is Hard to Approximate within $n^{1-\epsilon}$. *Acta Mathematica* **182**, 105–142.

RUSSELL IMPAGLIAZZO, TONIANN PITASSI & ALASDAIR URQUHART (1994). Upper and lower bounds for tree-like cutting planes proofs. In *9th Annual IEEE Symposium on Logic in Computer Science*, 220–228. Los Alamitos, CA.

SVANTE JANSON, TOMASZ ŁUCZAK & ANDRZEJ RUCIŃSKI (2000). *Random Graphs.* John Wiley & Sons.

TANG JIAN (1986). Algorithms for Solving Maximum Independent Set Problem. *IEEE Transactions on Computers* **35**(9), 847–851.

RICHARD M. KARP (1972). Reducibility Among Combinatorial Problems. In *Complexity of Computer Computations*, RAYMOND E. MILLER & JAMES W. THATCHER, editors, 85–104. Plenum Press, New York.

RAN RAZ (2004). Resolution Lower Bounds for the Weak Pigeonhole Principle. *Journal of the ACM* **51**(2), 115–138.

ALEXANDER A. RAZBOROV (2004). Resolution Lower Bounds for Perfect Matching Principles. *Journal of Computer and System Sciences* **69**(1), 3–27.

JOHN ALAN ROBINSON (1965). A Machine-Oriented Logic Based on the Resolution Principle. *Journal of the ACM* **12**(1), 23–41.

JOHN MICHAEL ROBSON (1986). Algorithms for Maximum Independent Sets. *Journal of Algorithms* **7**(3), 425–440.

ASHISH SABHARWAL (2005). *Algorithmic Applications of Propositional Proof Complexity.* Ph.D. thesis, University of Washington, Seattle.

MIKIO SHINDO & ETSUJI TOMITA (1990). A Simple Algorithm for Finding a Maximum Clique and Its Worst-Case Time Complexity. *Systems and Computers in Japan* **21**(3), 1–13.

ROBERT ENDRE TARJAN (1972). Finding a Maximum Clique. Technical Report 72-123, Computer Science Department, Cornell University, Ithaca, NY.

ROBERT ENDRE TARJAN & ANTHONY E. TROJANOWSKI (1977). Finding a Maximum Independent Set. *SIAM Journal on Computing* **6**(3), 537–546.

ETSUJI TOMITA & TOMOKAZU SEKI (2003). An Efficient Branch-and-Bound Algorithm for Finding a Maximum Clique. In *4th International Conference on Discrete Mathematics and Theoretical Computer Science*, volume 2731 of *Lecture Notes in Computer Science*, 278–289. Springer-Verlag, Dijon, France.

LUCA TREVISAN (2001). Non-approximability Results for Optimization Problems on Bounded Degree Instances. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, 453–461. Crete, Greece.

G. S. TSEITIN (1968). On the Complexity of Derivation in the Propositional Calculus. In *Studies in Constructive Mathematics and Mathematical Logic, Part II*, A. O. SLISENKO, editor, 115–125. Consultants Bureau, New York. Translated from Russian, 1970.

ALASDAIR URQUHART (1987). Hard Examples for Resolution. *Journal of the ACM* **34**(1), 209–219.

Manuscript received September 13, 2004

PAUL BEAME
Computer Science and Engineering
University of Washington, Box 352350
Seattle, WA 98195-2350, USA
beame@cs.washington.edu

RUSSELL IMPAGLIAZZO
Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093-0114, USA
russell@cs.ucsd.edu

ASHISH SABHARWAL
Computer Science and Engineering
University of Washington, Box 352350
Seattle, WA 98195-2350, USA
ashish@cs.washington.edu

Current address of ASHISH SABHARWAL:
Department of Computer Science
Cornell University
Ithaca, NY 14853-7501, USA
sabhar@cs.cornell.edu