*Combinatorics, Probability and Computing* (2021), **30**, pp. 1–16
doi:10.1017/S0963548320000218

CAMBRIDGE UNIVERSITY PRESS

ARTICLE

# On the subgraph query problem

Ryan Alweiss[1,*,†], Chady Ben Hamida[1], Xiaoyu He[2,‡] and Alexander Moreira[1]

[1]Department of Mathematics, Princeton University, Princeton, NJ 08541, USA and [2]Department of Mathematics, Stanford University, Stanford, CA 94305, USA
*Corresponding author. Email: alweiss@math.princeton.edu

(Received: 14 November 2019; revised 24 March 2020; first published online 27 July 2020)

**Abstract**
Given a fixed graph $H$, a real number $p \in (0, 1)$ and an infinite Erdős–Rényi graph $G \sim G(\infty, p)$, how many adjacency queries do we have to make to find a copy of $H$ inside $G$ with probability at least 1/2? Determining this number $f(H, p)$ is a variant of the *subgraph query problem* introduced by Ferber, Krivelevich, Sudakov and Vieira. For every graph $H$, we improve the trivial upper bound of $f(H, p) = O(p^{-d})$, where $d$ is the degeneracy of $H$, by exhibiting an algorithm that finds a copy of $H$ in time $o(p^{-d})$ as $p$ goes to 0. Furthermore, we prove that there are 2-degenerate graphs which require $p^{-2+o(1)}$ queries, showing for the first time that there exist graphs $H$ for which $f(H, p)$ does not grow like a constant power of $p^{-1}$ as $p$ goes to 0. Finally, we answer a question of Feige, Gamarnik, Neeman, Rácz and Tetali by showing that for any $\delta < 2$, there exists $\alpha < 2$ such that one cannot find a clique of order $\alpha \log_2 n$ in $G(n, 1/2)$ in $n^\delta$ queries.

**2020 MSC Codes:** Primary: 05C57 games on graphs; 05C80 random graphs; 05D10 Ramsey theory

## 1. Introduction

The *subgraph query problem*, introduced by Ferber, Krivelevich, Sudakov and Vieira [7], has been the subject of recent attention in extremal combinatorics and theoretical computer science. The problem is to determine the smallest number of adaptive queries of the form 'is $(u, v) \in E(G)$?' that need to be made to an Erdős–Rényi random graph $G \sim G(n, p)$ to find a copy of a given subgraph $H$ with probability at least 1/2.

Several variants of the problem appear in the literature. Ferber, Krivelevich, Sudakov and Vieira [7, 8] first studied the subgraph query problem when $H$ is a long path or cycle of order comparable to $n$, exhibiting asymptotically optimal algorithms for finding long paths and cycles. For example, as long as

$$p \geqslant \frac{\log n + \log\log n + \omega(1)}{n}$$

is above the threshold for Hamiltonicity in $G(n, p)$, they showed that a Hamiltonian cycle can be found by the time one reveals $(1 + o(1))n$ edges. Here and henceforth we write log for the natural logarithm and lg for the base-2 logarithm.

† Research supported by an NSF Graduate Research Fellowship.
‡ Research supported by an NSF Graduate Research Fellowship.

In connection with the online Ramsey number, Conlon, Fox, Grinshpun and He [5] studied the case where $H = K_m$ is a fixed complete graph, $p \to 0$, and the number of vertices $n$ is allowed to be arbitrarily large. They defined the function $f(H, p)$ to be the number of queries needed to find a copy of $H$ in the countably infinite random graph $G(\infty, p)$ with probability 1/2, and proved that

$$p^{-(2-\sqrt{2})m+O(1)} \leqslant f(K_m, p) \leqslant p^{-(2/3)m-O(1)}. \tag{1.1}$$

In this paper, we study the behaviour of $f(H, p)$ as $p \to 0$ for an arbitrary fixed graph $H$. We will use the phrases 'build $H$ in $T$ time' and 'find $H$ in $T$ queries' interchangeably for the statement $f(H, p) \leqslant T$.

Recall that a graph $H$ is *d-degenerate* if every subgraph of $H$ contains a vertex of degree at most $d$, and the *degeneracy* of $H$ is the least $d$ for which $H$ is $d$-degenerate. Equivalently, $H$ is $d$-degenerate if and only if there is an acyclic orientation of $H$ with maximum out-degree at most $d$. Degeneracy is the natural notion of sparsity in graph Ramsey theory (see *e.g.* the recent proof of the Burr–Erdős conjecture by Lee [12]).

In the subgraph query problem, a $d$-degenerate graph can be built by adding one vertex at a time so that each new vertex has degree at most $d$ at the time it is built. Since a common neighbour of $d$ given vertices can be found in $O(p^{-d})$ queries, this shows that $f(H, p) = O_H(p^{-d})$ whenever $H$ is $d$-degenerate.

Our first main result is that this trivial bound is never tight when $d \geqslant 2$. Define the *depth* $\Delta$ of a graph $H$ with degeneracy $d$ to be the smallest $\Delta$ for which there exists an acyclic orientation of $H$ with maximum out-degree at most $d$ and longest directed path of length at most $\Delta$ (we use the convention that the length of the path with $n+1$ vertices is $n$). Let $\log_t(x)$ denote the $t$-fold iterated logarithm of $x$.

**Theorem 1.1.** *If $H$ is a graph with degeneracy $d \geqslant 2$ and depth $\Delta \geqslant 1$, then*

$$f(H, p) = O_H\left(\frac{p^{-d}\log_{\Delta+1}(p^{-1})}{\log_{\Delta}(p^{-1})}\right).$$

Roughly speaking, one of the main innovations is to exploit the observation that in a random graph $G(n, 1/n)$, the degrees of vertices are approximately Poisson with mean 1. Thus the maximum degree is $\Omega(\log n/\log\log n)$ despite the fact that the average degree is constant. Repeatedly finding these vertices of exceptionally large degree allows us to find $H$ slightly faster.

We will also show that the behaviour in Theorem 1.1 can be correct up to the polylogarithmic factor. Let the triforce graph be the graph obtained from the triangle $K_3$ by adding a common neighbour to each pair of vertices (see Figure 1 in Section 2).

**Theorem 1.2.** *If $H$ is the triforce graph and*

$$\ell = \frac{\log(1/p)}{2\log\log(1/p)},$$

*then*

$$f(H, p) = \Omega(p^{-2}/\ell^4).$$

Note that the triforce is 2-degenerate, so Theorems 1.1 and 1.2 together prove that $f(H, p) = o(p^{-2})$ and $f(H, p) = \Omega(p^{-2+\varepsilon})$ for every $\varepsilon > 0$. This is the first example of a graph for which it is known that $f(H, p)$ does not grow like a power of $p^{-1}$.

The question of querying for subgraphs in random graphs was also studied by Feige, Gamarnik, Neeman, Rácz and Tetali [6], and by Rácz and Schiffer in the related planted clique model [14]. Feige *et al.* restricted their attention to the balanced random graph $G(n, 1/2)$ and asked for the

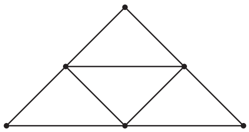

**Figure 1.** The triforce graph.

minimum number of queries needed to find a clique of order $(2-o(1))\lg n$ (which approaches the clique number) with probability at least 1/2. For $\delta < 2$, define $\alpha_\star(\delta)$ to be the supremum over $\alpha \leqslant 2$ for which a clique of order $\alpha \lg n$ can be found with probability at least 1/2 in at most $n^\delta$ queries for all $n$ sufficiently large. They showed under the additional assumption that only a bounded number of rounds of adaptiveness are used that $\alpha_\star(\delta) < 2$ for all $\delta < 2$, and asked if this could be proved unconditionally.

Our last theorem answers this question affirmatively. We are grateful to Huy Pham [13] for communicating to us the main idea of the proof.

**Theorem 1.3.** *For all* $2/3 < \delta < 2$,

$$\alpha_\star(\delta) \leqslant 1 + \sqrt{1 - \frac{(2-\delta)^2}{2}} < 2.$$

The proof is an adaptation of the lower bound proof for (1.1) by [5] to take the size of the vertex set into account. The exact value of $\alpha_\star(\delta)$ remains open for all $\delta$, and the best known lower bound is $\alpha_\star(\delta) \geqslant 1 + \delta/2$ when $1 \leqslant \delta < 2$ (see [6, Lemma 6]).

**Organization.** In Section 2 we describe a new algorithm for finding any $d$-degenerate graph and prove that it achieves the runtime described in Theorem 1.1. In Section 3 we give a new argument for proving lower bounds on $f(H, p)$, proving Theorem 1.2. In Section 4 we give a short proof of Theorem 1.3, using a variation of the methods in [5]. Finally, Section 5 highlights a few of the many open questions that remain about $f(H, p)$.

We will write $b = p^{-1}$ for the expected number of queries needed to find a single edge in $G(\infty, p)$. No attempt will be made to optimize the implicit constants in any of our proofs. We use $A \lesssim B$ to mean $A = O(B)$. For the sake of clarity of presentation, we systematically omit floor and ceiling signs whenever they are not crucial.

## 2. Upper bounds

### *2.1 An illustrative example*

As mentioned in the Introduction, there is a straightforward algorithm for finding any $d$-degenerate graph $H$ in $O_H(b^d)$ time. In this section we prove Theorem 1.1 by providing a new algorithm that beats the trivial algorithm by an iterated logarithmic factor.

We begin by illustrating how the algorithm works with a specific 2-degenerate graph.

**Definition 1.** *The* triforce *is the graph on six vertices and nine edges obtained from the triangle $K_3$ by adding a common neighbour to each pair of vertices. See Figure 1.*

The key step in building the triforce quickly is to build a large book.

**Definition 2.** *The book $B_{d,t}$ is the graph on $d+t$ vertices obtained by removing the edges of a clique $K_t$ from a complete graph $K_{d+t}$. The $t$ vertices of the removed clique are called the pages of the book and the remaining $d$ vertices are called its spine. See Figure 2.*

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

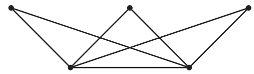

**Figure 2.** The book $B_{2,3}$.

Note that when $d$ and $t$ are fixed positive integers, $B_{d,t}$ is $d$-degenerate and thus we have $f(B_{d,t}, p) = O_t(b^d)$. The key observation is that this can be improved substantially even if we allow $t$ to grow slowly as $p$ tends to 0.

**Lemma 2.1.** *If $d \geqslant 2$ and*

$$\ell = \frac{\log b}{2 \log \log b},$$

*then $f(B_{d,\ell}, p) = O(b^d \ell^{-1/2})$.*

**Proof.** We will exhibit an algorithm which finds $B_{d,\ell}$ in $G(\infty, p)$ with constant probability (w.c.p.) in $O(b^d \ell^{-1/2})$ time. The algorithm has three steps.

First we find w.c.p. $d-1$ vertices $v_1, \ldots, v_{d-1}$ of the spine forming a clique in $O(b^{d-2})$ time, which is possible because $K_{d-1}$ is $(d-2)$-degenerate. Assume this step succeeds.

Next we build a large pool $S$ of common neighbours of $v_1, \ldots, v_{d-1}$, which will serve as candidates for the remaining vertex $v_d$ of the spine and for the pages of the book. In $d-1 = O(1)$ queries we can check a single new vertex $u$ to see if it is a common neighbour of $v_1, \ldots, v_{d-1}$, and $u$ has a probability $p^{d-1}$ of being such a common neighbour. We check a total of $4b^d \ell^{-1/2}$ possible $u$, and each common neighbour successfully found is added to $S$. Since the outcomes of all queries are independent, $|S|$ is distributed like the binomial random variable $\mathrm{Bin}(4b^d \ell^{-1/2}, p^{d-1})$ with mean $4b\ell^{-1/2}$, so w.c.p. $|S| \geqslant 2b\ell^{-1/2}$.

For the last step, assuming the previous two steps succeed, we will find a star $K_{1,\ell}$ contained in $S$. Along with vertices $v_1, \ldots, v_{d-1}$ already chosen, this forms the desired book.

To find this star, remove vertices from $S$ until it has size exactly $2b\ell^{-1/2}$, and then query all pairs of vertices in $S$ in $O(b^2 \ell^{-1})$ time. The induced subgraph on $S$ is just an Erdős–Rényi random graph $G(2b\ell^{-1/2}, p)$. It suffices to show that w.c.p. there exists a vertex of degree at least $\ell$ therein. This fact is a consequence of the observation that the degrees are approximately Poisson.

To give a quick proof of this fact, divide $S$ into two sets $S_1, S_2$ of size $r = b\ell^{-1/2}$, let $u_1, \ldots, u_r$ be the vertices of $S_1$, and let $X_i$ be the number of neighbours of $u_i$ in $S_2$. Then $\{X_i\}_{i=1}^r$ are $r$ i.i.d. random variables distributed like $\mathrm{Bin}(r, p)$, so

$$\mathbb{P}[X_i \geqslant \ell] \geqslant \binom{r}{\ell} p^\ell (1-p)^{r-\ell} \geqslant \frac{(r-\ell)^\ell p^\ell (1-p)^r}{\ell!}.$$

As $p \to 0$, we can bound $r - \ell > b\ell^{-1/2}/2$, $(1-p)^r \to 1$ and $\ell! < \ell^\ell$. Thus

$$\mathbb{P}[X_i \geqslant \ell] \geqslant \Omega\left(\frac{1}{2^\ell \ell^{3\ell/2}}\right).$$

When

$$\ell = \frac{\log b}{2 \log \log b},$$

this fraction is certainly $\Omega(b^{-4/5})$. In particular, since there are $r = b^{1-o(1)}$ independent random variables $X_i$, w.c.p. some $X_i$ is at least $\ell$, as desired.

Letting the vertex of degree $\ell$ be the last vertex $v_d$ of the book's spine and its $\ell$ neighbours in $S$ be the pages of the book, we have found a copy of $B_{d,\ell}$ w.c.p. in $O(b^d)$ total queries, as desired. $\square$

We are now ready to prove a stronger version of Theorem 1.1 when $H$ is the triforce.

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

**Theorem 2.2.** *If $H$ is the triforce graph and*

$$\ell = \frac{\log b}{2 \log \log b},$$

*then $f(H, p) = O(b^2 \ell^{-1/2})$.*

**Proof.** We exhibit an algorithm for finding $H$ w.c.p. in $O(b^2\ell^{-1/2})$ time.

Using Lemma 2.1 with $d = 2$, build a copy of $B_{2,\ell}$. Let $x$ and $y$ be the two vertices of its spine and let $Z$ be its pages. In $O(b^2\ell^{-1/2})$ time we can w.c.p. find two sets of vertices $S_x$, $S_y$, each of size $b\ell^{-1/2}$, so that everything in $S_x$ is adjacent to $x$ and everything in $S_y$ is adjacent to $y$. Now we will query all pairs between $S_x$ and $Z$ as well as all pairs between $S_y$ and $Z$. This takes only $O(b\ell^{1/2})$ time, which is negligible.

We claim that w.c.p. there exist $x' \in S_x$, $z \in Z$ and $y' \in S_y$ so that $x' \sim z$ and $z \sim y'$. This follows because w.c.p., $\Theta(\ell^{1/2})$ vertices of $Z$ have a neighbour in $S_x$, and among these vertices w.c.p. at least one has a neighbour in $S_y$. Now let $z'$ be any common neighbour of $x$ and $y$ in $Z$ other than $z$. It follows that the six vertices $x, y, z, x', y', z'$ form a triforce, and we have found it in $O(b^2\ell^{-1/2})$ queries w.c.p., as desired. □

### *2.2 The general upper bound*

Roughly speaking, the main trick in the proofs of Lemma 2.1 and Theorem 2.2 is that we can find vertices of much larger than average degree in a random graph with constant average degree. We will iterate this trick many times to prove the general statement in Theorem 1.1.

We will construct an arbitrary $d$-degenerate graph $H$ recursively. If the vertices of $H$ are ordered $v_1, \ldots, v_n$ in the degeneracy order, the algorithm will maintain a 'cloud' of candidates $C_i$ for the image of vertex $v_i$, which shrinks as the algorithm progresses. On step $i$, the algorithm chooses $v_i$ from $C_i$ and then shrinks the clouds corresponding to neighbours of $v_i$ to stay consistent with this choice.

**Proof of Theorem 1.1.** Let $H$ be a graph on $n$ vertices with degeneracy $d \geqslant 2$ and depth $\Delta$. Order its vertices $v_1, \ldots, v_n$ so that each $v_i$ has at most $d$ neighbours $v_j$ with $j < i$, and the longest left-to-right path $v_{i_0}, \ldots, v_{i_r}$ with $i_0 < \cdots < i_r$ has length $r = \Delta$. Let $\Delta_i$ be the length (in edges) of the longest left-to-right path ending at $v_i$, so that $\Delta_i \leqslant \Delta$ for all $i$. Finally, define

$$L(x) = \frac{\log x}{3n \log \log x}$$

and $\ell_i = L^{(\Delta_i)}(b)$ is obtained from $b$ by iterating $L$ $\Delta_i$ times.

We describe an algorithm for finding an injection $\phi$ from $H$ to $G(\infty, p)$ in a series of rounds, assuming $p$ is sufficiently small. There are many points at which the algorithm may fail. However, each round succeeds with probability $\Omega_H(1)$ conditional on the success of all previous rounds, and there are $n = O_H(1)$ rounds, so the entire algorithm succeeds with $\Omega_H(1)$ probability. The algorithm can then be repeated a number of times depending only on $H$ until its success probability reaches $1/2$; this only changes the implicit constant in $f(H, p)$.

We begin by setting aside $n$ disjoint sets ('clouds') $C_1, \ldots, C_n$ which will change throughout the algorithm. We initialize these to $C_1^{(0)}, \ldots, C_n^{(0)}$ of order $|C_i^{(0)}| = b^d/\ell_i$, so that $C_i^{(0)}$ is the set of candidates for $\phi(v_i)$. We proceed in $n$ rounds, so that $C_j^{(k)}$ will refer to the state of cloud $C_j$ after round $k$. After the $k$th round we will have non-empty disjoint sets $C_j^{(k)}$, and we always have $C_j^{(k-1)} \supset C_j^{(k)}$. In the round $k$ a number of queries are made to decide the value of $\phi(v_k) \in C_k^{(k-1)}$. Thus $C_k^{(k)}$ is the singleton $\{\phi(v_k)\}$ and $C_k$ remains a singleton until the end. For each $j$ with $j > k$

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

and $v_j \sim v_k$, the set $C_j^{(k-1)}$ is updated to a subset $C_j^{(k)}$ consisting of all elements of $C_j^{(k-1)}$ adjacent to $\phi(v_k)$. We say that a vertex $v_j$ is *living* after round $k$ if $j > k$ and *dead* otherwise.

Two properties are maintained. The first is that after round $k$, for any $i \leqslant k$ and $j \leqslant n$ and any $u_i \in C_i^{(k)}$ and $u_j \in C_j^{(k)}$, $u_i \sim u_j$ if $v_i \sim v_j$. In other words, the adjacency relations are correct within the dead vertices and between the dead vertices and the clouds $C_i^{(k)}$ for the living ones.

The second property is that the size of the set $C_j^{(k)}$ must be

$$c_j^{(k)} := \begin{cases} b^{d-m}/\ell_j & \text{if } v_j \text{ is living and has } m < d \text{ dead left-neighbours,} \\ \ell_j & \text{if } v_j \text{ is living and has exactly } d \text{ dead left-neighbours,} \\ 1 & \text{if } v_j \text{ is dead.} \end{cases}$$

The queries on round $k$ are made to guarantee two properties. First, on round $k$, vertices are thrown out of $C_k^{(k-1)}$ until it has exactly $\ell_k$ vertices (this is possible because $c_k^{(k-1)} \geqslant \ell_k$ when $b$ is sufficiently large). Then consider the $j$ so that $j > k$ and $v_j \sim v_k$. If such a $v_j$ has exactly $d-1$ dead left-neighbours, then $j$ is called *active* on round $k$ and otherwise $j$ is called *inactive*. For each active $j$, all pairs in $C_k^{(k-1)} \times C_j^{(k-1)}$ are queried.

Each round is divided into an *active portion*, which happens first, and then an *inactive portion*.

The active portion of round $k$ succeeds if a candidate $u_k \in C_k^{(k-1)}$ is found to have at least $c_j^{(k)}$ neighbours in $C_j^{(k-1)}$ for all the active $j$. One such candidate $u_k$ is picked for $\phi(v_k)$ and $C_j^{(k)}$ is chosen to be exactly $c_j^{(k)}$ neighbours of $u_k$ in $C_j^{(k-1)}$.

Then, for all inactive $j$, all pairs $\{u_k\} \times C_j^{(k-1)}$ are queried. The inactive portion of round $k$ succeeds if, after these queries, a total of $c_j^{(k)}$ neighbours are found for $u_k$ in $C_j^{(k-1)}$ for each of the inactive $j$ as well. We say that the round succeeds if both the active and inactive portions succeed. The algorithm only continues past round $k$ if round $k$ succeeds.

By induction on $k$, the algorithm maintains all the required properties and produces a valid injection $\phi\colon H \to G(\infty, p)$ if it succeeds on every round. It remains to show that the probability of success on each round is $\Omega_H(1)$.

For each $u \in C_k^{(k-1)}$ and $j > k$ for which $v_j \sim v_k$, let $d_j(u)$ be the number of neighbours $u$ has in $C_j^{(k-1)}$. Note that $d_j(u)$ is distributed like $\mathrm{Bin}(c_j^{(k-1)}, p)$, since each vertex of $C_j^{(k-1)}$ is adjacent to $u$ independently with probability $p$.

Suppose $j$ is active in round $k$, so that $c_j^{(k-1)} = b/\ell_j$. This time we get

$$\mathbb{P}[d_j(u) \geqslant c_j^{(k)}] = \mathbb{P}[\mathrm{Bin}(b/\ell_j, p) \geqslant \ell_j] \geqslant \binom{b/\ell_j}{\ell_j} p^{\ell_j}(1-p)^{b/\ell_j - \ell_j}.$$

Using the facts that $1 - x \geqslant e^{-2x}$ for all $x \in [0, 1/2]$ and that $\ell_j \to \infty$ as $p \to 0^+$, we see that $(1-p)^{b/\ell_j - \ell_j} \geqslant e^{-2/\ell_j} \to 1$. Also, $\binom{a}{b} \geqslant (a/b)^b$ for all $a \geqslant b \geqslant 1$, and thus

$$\mathbb{P}[d_j(u) \geqslant c_j^{(k)}] \geqslant \binom{b/\ell_j}{\ell_j} p^{\ell_j}(1-p)^{b/\ell_j - \ell_j} = \Omega(\ell_j^{-2\ell_j}).$$

There are at most $n$ total $j$, so taking a product over all active $j$, we arrive at a lower bound

$$\mathbb{P}[d_j(u) \geqslant c_j^{(k)} \text{for all active } j] \geqslant \Omega_H(\ell_j^{-2n\ell_j}).$$

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

Since each $u \in C_k^{(k-1)}$ is individually a successful candidate for $\phi(v_k)$ with this probability, and these are $\ell_k$ independent events, it follows that

$$\mathbb{P}[\text{active portion round } k \text{ succeeds}] \geqslant \Omega_H(\min(1, \ell_k \ell_j^{-2n\ell_j})).$$

Finally, we observe that for every $j$ active in round $k$, $\Delta_j \geqslant \Delta_k + 1$ since every left-to-right path ending at $k$ extends to a longer one ending at $j$. Thus $\ell_j \leqslant L(\ell_k)$, and the function $L$ was chosen so that $L(x)^{2nL(x)} \leqslant x$ for $x$ sufficiently large. It follows that the active portion of round $k$ succeeds with probability $\Omega_H(1)$, as desired.

Now we look at the inactive $j$ in round $k$. Then $c_j^{(k)} = pc_j^{(k-1)}$, so

$$\mathbb{P}[d_j(u_k) \geqslant c_j^{(k)}] = \mathbb{P}[\text{Bin}(c_j^{(k-1)}, p) \geqslant pc_j^{(k-1)}] = \Omega(1).$$

Thus, conditional on the success of the active portion of round $k$, the inactive portion succeeds with probability $\Omega_H(1)$ as well.

We have now shown that the algorithm, iterated $O_H(1)$ times, succeeds with probability 1/2. It remains to bound the total number of queries made. In the active portion of each round, queries are only made between sets $C_k^{(k-1)}$ and $C_j^{(k-1)}$ if $j$ is relevant, which implies that $c_j^{(k)} = b/\ell_j$. Also, elements of $C_k^{(k-1)}$ were thrown out until it had size exactly $\ell_k = O(b)$, so the number of queries made in the active portion of any round is at most $O(b^2/L^{(\Delta)}(b)) = O(b^d/L^{(\Delta)}(b))$.

In the inactive portion of each round, queries are made between a single vertex $u_k$ and sets $C_j^{(k-1)}$ of size at most $b^d/\ell_j$ each. Thus the number of queries made in the inactive portion of any round is also $O(b^d/L^{(\Delta)}(b))$.

Since there are at most $n = O_H(1)$ rounds and at most $n$ choices of $j$ involved in each round, we find that

$$f(H,p) = O_H\left(\frac{b^d}{L^{(\Delta)}(b)}\right) = O_H\left(\frac{p^{-d} \log_{\Delta+1}(p^{-1})}{\log_{\Delta}(p^{-1})}\right),$$

as desired. □

## 3. Lower bounds

### 3.1 Preliminaries

In this section we will prove lower bounds for $f(H,p)$. Because $N$ queries necessarily involve at most $2N$ vertices, it suffices to prove lower bounds for finding a copy of $H$ in $G(2N,p)$ rather than in $G(\infty,p)$. Following [5], we will lower-bound the number of queries it takes to build a copy of $H$ by showing that the expected number of copies of $H$ we can build in some given amount of time is not too large.

**Definition 3.** *If $H$ is a graph without isolated vertices, define $t(H,p,N)$ to be the maximum (over all querying strategies) expected number of copies of $H$ that can be found in $G(\infty,p)$ in $N$ queries. Since we are working on $G(2N,p)$, if $H = H' \cup \{v_1, \ldots, v_t\}$ has $t$ isolated vertices, define $t(H,p,N) := (2N)^t \cdot t(H',p,N)$.*

If we show that we cannot build so many copies of $H$ (in expectation) in some given time, this gives us a lower bound on how long it takes to build a single copy of $H$.

**Lemma 3.1.** *If $N \geqslant f(H,p)$, then*

$$f(H,p) \cdot t(H,p,N) \geqslant N/4.$$

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

Thus upper bounds on $t(H,p,N)$ will yield lower bounds on $f(H,p)$. The proof of Lemma 3.1 is straightforward, but we include it for completeness.

*Proof.* By definition, there exists a strategy which finds $H$ with probability $1/2$ in $f(H,p)$ queries. Given $N$ queries, we can iterate this strategy $\lfloor N/f(H,p)\rfloor$ independent times on disjoint vertex sets. By linearity of expectation, this implies

$$t(H,p,N) \geqslant \left\lfloor \frac{N}{f(H,p)} \right\rfloor \cdot \frac{1}{2} \geqslant \frac{N}{4f(H,p)}. \qquad \square$$

Thus it suffices to produce upper bounds on $t(H,p,N)$. Fortunately, we can recursively bound $t(H,p,N)$ in terms of $t(H',p,N)$ for some subgraphs $H'$. The following bounds are proved in [5].

**Lemma 3.2.** [5]. *If $H$ is any graph, $p \in (0,1)$, and $N \geqslant p^{-1-\varepsilon}$ for some $\varepsilon > 0$, then the following inequalities hold:*

$$t(H,p,N) \leqslant \min_{e\in E(H)} t(H\backslash e, p, N), \tag{3.1}$$

$$t(H,p,N) \lesssim p \max_{e\in E(H)} t(H\backslash e, p, N), \tag{3.2}$$

$$t(H,p,N) \lesssim pN \min_{e\in E(H)} t(H\backslash \{u,v\}, p, N), \tag{3.3}$$

*where $u, v$ are the vertices of $e$ in (3.3). In the latter two inequalities, the implicit constants are allowed to depend only on $H$.*

In general, the bounds of Lemma 3.2 are not tight. In certain cases we will improve these bounds using the crucial observation that large enough sets of vertices in a random graph have few common neighbours. For any vertex subset $U \subseteq V(G)$ of a graph $G$, write $d(U)$ for the number of common neighbours of every vertex in $U$.

**Lemma 3.3.** *Let*

$$\ell = \frac{\log b}{2\log\log b},$$

*let $k, n \geqslant 2$ be absolute constants, and let $G = G(2N,p)$.*

(1) *If $p^kN \lesssim 1$, then there exists $C > 0$ so that*

$$\mathbb{P}\Big[\max_{|U|=k} d(U) > C\ell\Big] < p^n,$$

*where the maximum is taken over all $k$-subsets $U$ of $V(G)$.*

(2) *If $p^kN = (1/N)^{\Omega(1)}$, then there exists $C > 0$ so that*

$$\mathbb{P}\Big[\max_{|U|=k} d(U) > C\Big] < p^n.$$

**Proof.** For an arbitrary set of $k$ vertices $U$, note that $d(U) \sim \mathrm{Bin}(2N-k, p^k)$ as there are $2N-k$ other vertices of $G(2N,p)$ and each vertex has a $p^k$ chance of being adjacent to all members of $U$. Hence we find that

$$\mathbb{P}[d(U) \geqslant t] = \mathbb{P}[\mathrm{Bin}(2N-k,p^k) \geqslant t] \leqslant \binom{2N-k}{t} p^{tk} \leqslant \left(\frac{2Ne}{t}p^k\right)^t. \tag{3.4}$$

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

If $p^k N \lesssim 1$, then (3.4) implies $\mathbb{P}[d(U) \geqslant t] \leqslant (O(1/t))^t$. Next we take a union bound over at most $(2N)^k$ choices of $U$, which shows that, if we take $t = C\ell$ for a sufficiently large $C$ depending on $n$,

$$\mathbb{P}\Big[\max_{|U|=k} d(U) \geqslant t\Big] \leqslant (O(1/t))^t \cdot N^k < p^n.$$

This proves the first part of the lemma.

If $p^k N = (1/N)^{\Omega(1)}$, then (3.4) implies the stronger bound $\mathbb{P}[d(U) \geqslant t] = N^{-\Omega(t)}$. For any fixed $n$, if $C$ is a large enough constant, the probability that $\max_{|U|=k} d(U) > C$ will be below $N^{-\Omega(C)} N^k \leqslant p^n$ by the union bound. □

The power of Lemma 3.3 is that the final graph that we find after $N$ queries is a subgraph of $G(2N, p)$, so we can bound the number of common numbers of any vertex set $U$ of constant size without even seeing the graph. It allows us to prove new upper bounds on $t(H, p, b^d)$.

**Lemma 3.4.** *Let*

$$\ell = \frac{\log b}{2 \log\log b},$$

*let $H$ be a graph, let $v \in V(H)$, and let $H' = H \setminus \{v\}$.*

(1) *If $d(v) = d$, then*

$$t(H, p, b^d) \lesssim \ell t(H', p, b^d).$$

(2) *If $d(v) > d$, then*

$$t(H, p, b^d) \lesssim t(H', p, b^d).$$

**Proof.** Let $k = d(v)$, and let the neighbours of $v$ in $H$ be $v_1, \ldots, v_k$. Fix a query strategy that maximizes $t(H, p, b^d)$.

For any subset $U = \{u_1, \ldots, u_k\}$ of $k$ vertices of the final graph $G \subset G(2b^d, p)$ that is found, let $H'(U)$ be the number of maps $H' \to G$ so that, for all $1 \leqslant i \leqslant k$, $v_i$ maps to $u_i$. Then we have that

$$t(H, p, b^d) = \mathbb{E}\Big[\sum_{|U|=k} d(U) H'(U)\Big] \leqslant \mathbb{E}\Big[\Big(\max_{|U|=k} d(U)\Big)\Big(\sum_{|U|=k} H'(U)\Big)\Big], \tag{3.5}$$

where the sum is taken over all $k$-sets of vertices $U$.

Assume $d(v) = d$. By the first part of Lemma 3.3, there is a large constant $C = C(H)$ so that

$$\mathbb{P}\Big[\max_{|U|=k} d(U) > C\ell\Big] < p^{d|V(H')|+1}.$$

We will break up the expectation in (3.5) depending on the size of $\max_{|U|=k} d(U)$. If $\max_{|U|=k} d(U) \leqslant C\ell$, the contribution to the right side of (3.5) is $O(\ell t(H', p, b^d))$. Now it holds that $\max_{|U|=k} d(U) > C\ell$ with probability at most $p^{d|V(H')|+1}$, so the contribution from these terms is bounded by $p^{d|V(H')|+1}(2b^d)^{|V(H')|} = o(1)$.

Likewise, when $d(v) > d$, by the second part of Lemma 3.3 there is a $C$ so that the case of $\max_U d(U) \leqslant C$ contributes $O(t(H', p, b^d))$ to the right side of (3.5), and the case of $\max_{|U|=k} d(U) > C$ contributes $o(1)$. □

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

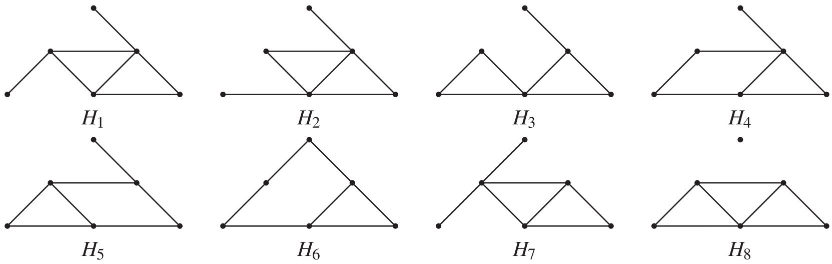


**Figure 3.** The eight subgraphs (up to isomorphism) with seven edges of the triforce.

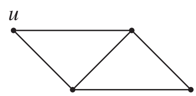


**Figure 4.** The "diamond graph", a subgraph of $H_i$ for $i = 1, 2, 7, 8$.

### 3.2 Proof of Theorem 1.2

We now begin the proof of Theorem 1.2. The main idea is to use Lemma 3.4 to obtain new upper bounds on $t(H, p, N)$ for various subgraphs $H$ of the triforce, and then combine these with Lemma 3.2 to bound $t(H, p, N)$ for the triforce itself.

We describe the subgraphs of the triforce to which we will apply Lemma 3.4. Any copy of the triforce must arise from a copy of one of the graphs formed by deleting two edges from the triforce. There are eight such graphs up to isomorphism, which we denote by $H_i$ for $1 \leqslant i \leqslant 8$ (see Figure 3).

We will prove that the first six of these graphs are hard to construct quickly, although it turns out that there is no need to analyse $H_1$, $H_2$, or $H_7$. The last subgraph $H_8$ is more difficult to handle, and we will bound copies of it using a different analysis.

**Proposition 3.5.** *For all $H_i$ such that $1 \leqslant i \leqslant 6$, $t(H_i, p, b^2) \lesssim b^2\ell^2$.*

**Proof.** For each graph $H_i$ with $1 \leqslant i \leqslant 6$, it is possible to remove two vertices of degree at least two to arrive at the path $P_3$ on four vertices. Thus we may apply the first part of Lemma 3.4 twice to show that

$$t(H_i, p, b^2) \lesssim \ell^2 t(P_3, p, b^2),$$

for all $1 \leqslant i \leqslant 6$. Lastly, note that $t(P_3, p, b^2) \lesssim bt(K_2, p, b^2) \lesssim b^2$ by applying (3.3) from Lemma 3.2.

Hence, for $H_i$ with $1 \leqslant i \leqslant 6$,

$$t(H_i, p, b^2) \lesssim \ell^2 t(P_3, p, b^2) \lesssim \ell^2 b^2$$

as desired. □

We must now deal with $H_8$, on which the reductions of Lemma 3.4 and Lemma 3.2 are not sufficient to provide the bounds that we want.

We need one last definition. Given a graph $H$ with a distinguished vertex $u$, let $t^u(H, p, N)$ be the maximum expected number of copies of $H$ we can build in time $N$ so that $u$ maps to the same vertex in each copy. It is important to emphasize that the image of $u$ is not determined ahead of time, and we may pick it adaptively based on the queries made so far.

**Lemma 3.6.** *Let $D$ be the* diamond *graph depicted in Figure 4. Then $t^u(D, p, b^2) \lesssim \ell^3$.*

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

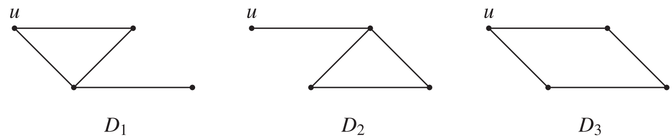


**Figure 5.** The three subgraphs formed by removing an edge from the diamond graph, up to isomorphisms fixing $u$.

**Proof.** As usual, we fix a query strategy maximizing $t^u(D, p, b^2)$ and let $G \subset G(2b^2, p)$ be the final graph built. Consider three subgraphs of $D$, which we call $D_1$, $D_2$ and $D_3$ respectively, shown in Figure 5.

Every copy of $D$ must arise from adding an edge to a graph isomorphic to one of the $D_i$. For each $u' \in V(G)$, let $X_i(u')$ be the random variable counting the number of copies of $D$ so that $u$ maps to $u'$, and the last edge built in $D$ is the edge missing from $D_i$. For each $1 \leqslant i \leqslant 3$, the number of copies of $D$ we can build so that $u$ maps to the same vertex in each copy, and so that $D$ arises from some copy of $D_i$, is bounded by $\max_{u'} X_i(u')$. Thus

$$t^u(D, p, b^2) \leqslant \sum_{i=1}^{3} \mathbb{E}\Big[\max_{u'} X_i(u')\Big]. \tag{3.6}$$

Also, define the random variable $X_i(u', j)$ to be the number of copies of $D_i$ with $u$ mapping to $u'$ that turn into a copy of $D$ after query $j$. In particular, this number is 0 if the query $j$ finds a non-edge. We have that $X_i(u') = \sum_j X_i(u', j)$.

By the first part of Lemma 3.3, in the random graph $G(2b^2, p)$ any two vertices have $O(\ell)$ common neighbours with overwhelmingly high probability. We can assume this is the case here as the contribution to the expectation $t^u(D, p, b^2)$ from other cases is $o(1)$. In particular, this means that each new edge built can turn at most $O(\ell^2)$ copies of $D_1$, $D_2$ or $D_3$ into $D$. For example, if an edge $(u', v')$ is built in $G$, then the number of copies of $D_2$ that can be completed into $D$ is exactly the number of ways to choose a common neighbour $w'$ of $u'$ and $v'$, and then a common neighbour of $v'$ and $w'$. As we assumed that codegrees are all $O(\ell)$, there are only $O(\ell^2)$ total choices for this copy of $D_2$.

This means we may assume that $X_i(u', j)$ is stochastically dominated (up to a constant) by $\ell^2\mathrm{Bin}(1, p)$. As the results of all queries are independent, it follows that $X_i(u')$ is stochastically dominated by a constant times $\ell^2\mathrm{Bin}(b, p)$. Now it is a short computation that

$$\mathbb{P}[\mathrm{Bin}(b, p) > 100\ell] < p^5.$$

In particular, there exists a $C > 0$ such that $\mathbb{P}[X_i(u') > C\ell^3] < p^5$ for all $1 \leqslant i \leqslant 3$ and all $u' \in V(G)$. Also, the maximum possible number of diamonds with a given vertex $u'$ is $(2b)^3$, so

$$t^u(D, p, b^2) \leqslant 3C\ell^3 + \mathbb{P}[X_i(u') > C\ell^3 \text{ for some } i, u'] \cdot (2b)^3 = O(\ell^3),$$

by the union bound over all $O(b^2)$ choices of $1 \leqslant i \leqslant 3$ and $u' \in V(G)$, as desired. □

Finally, we deal with the graph $H_8$. This graph behaves differently from the other ones, in that it is not the case that $t(H_8, p, b^2) = O(b^{2+o(1)})$ (in fact one can build $\Omega(b^3)$ copies of $H_8$ due to the isolated vertex). We will need to add one edge to $H_8$ and analyse the resulting graph instead.

**Proposition 3.7.** *Let $H^*$ be the graph shown in Figure 6. Then $t(H^*, p, b) = O(b\ell^3)$.*

**Proof.** Let $u$ be the vertex adjacent to the leaf of $H^*$. For any vertex $v$ of our final graph $G \subset G(2b^2, p)$, let $f(v)$ be the number of copies of the diamond $D$ so that $u$ maps to $v$. As before, we may assume that all degrees are $O(b)$ as the contribution to $t(H^*, p, b^2)$ is trivial otherwise.

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

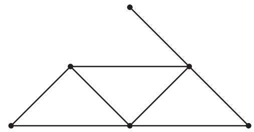

**Figure 6.** The graph $H^*$ obtained by removing an outer edge from the triforce.

Furthermore, we may assume that any two vertices have $O(\ell)$ common neighbours, as again the contribution is trivial from other cases by the first part of Lemma 3.3. Given any $v$, there are $f(v)$ choices for a copy of $D$ including it, $d(v)$ choices for the leaf off of $v$, and $O(\ell)$ choices for the remaining vertex of $H^*$, as it is the common neighbour of $v$ and of its degree 4 neighbour in $H^*$. Thus we obtain

$$\begin{aligned} t(H^*, p, b^2) &\lesssim \ell\mathbb{E}\left[\sum_v f(v)d(v)\right] \\ &\lesssim \ell\mathbb{E}\left[\left(\max_v f(v)\right)\left(\sum_v d(v)\right)\right] \\ &\lesssim b\ell\mathbb{E}\left[\max_v f(v)\right] \\ &\lesssim b\ell t^u(D, p, b^2) \\ &\lesssim b\ell^4, \end{aligned}$$

where the last inequality follows from Lemma 3.6. □

Putting all of the bounds together completes the proof of Theorem 1.2.

**Proof of Theorem 1.2.** We will apply (3.2) twice to the triforce $H$. Applying it once, we find

$$t(H, p, b^2) \lesssim p \max_{e \in E(H)} t(H \backslash e, p, b^2), \tag{3.7}$$

and there are only two non-isomorphic subgraphs of $H$ of the form $H \setminus e$. One of them is $H^*$, for which we have $t(H^*, p, b^2) = O(b\ell^4)$ by Proposition 3.7.

If $H'$ is the other subgraph of the form $H \setminus e$, where an inner edge is deleted from the triforce, then we apply (3.2) again to find

$$t(H', p, b^2) \lesssim p \max_{3 \leqslant i \leqslant 6} t(H_i, p, b^2), \tag{3.8}$$

since all the graphs $H' \setminus e$ are isomorphic to one of $H_3, H_4, H_5, H_6$. We have by Proposition 3.5 that $t(H_i, p, b^2) = O(b^2\ell^3)$.

It follows from (3.8) that $t(H', p, b^2) = O(b\ell^3)$. Together with the fact that $t(H^*, p, b^2) = O(b\ell^4)$ and (3.7), this proves that $t(H, p, b^2) = O(\ell^4)$. The theorem follows by one application of Lemma 3.1 with $N = b^2$. □

## 4. Cliques in $G(n, 1/2)$

In this section we prove Theorem 1.3 using an idea of Huy Pham [13]. The argument is a modification of the proof of Theorem 1 in [5] when the number of vertices in $G(n, p)$ is bounded beforehand.

Let $G = G(n, 1/2)$. For each vertex subset $U \in V(G)$, let $e_t(U)$ be the number of queries made between pairs of vertices in $U$ after query $t$. We will study the weight function

$$w(U, t) = \begin{cases} 2^{-\binom{|U|}{2} + e_t(U)} & \text{if all queries so far in } U \text{ succeeded,} \\ 0 & \text{otherwise.} \end{cases}$$

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

In other words, $w(U, t)$ is exactly the probability that $G[U]$ is a clique conditional on the information revealed after query $t$. The standard method of conditional expectation proceeds by studying the evolution of the function

$$w_k(t) := \sum_{|U|=k} w(U, t),$$

which is a martingale, and has the property that a $k$-clique is found after query $t$ only if $w_k(t) \geqslant 1$. Our modification instead studies a restricted version of this sum. Namely, define $m(U, t)$ to be the size of the maximum matching in the known edges of $U$ after query $t$. Then

$$w_{k,m}(t) := \sum_{|U|=k, m(U,t)\geqslant m} w(U, t).$$

Restricting to only sets with large maximum matchings has the function of radically reducing the number of terms in the sum $w_{k,m}(t)$. We pay for it in that $w_{k,m}(t)$ is no longer a martingale and its expectation is harder to study. Nevertheless, it remains true that if a $k$-clique is found after query $t$, then $w_{k,m}(t) \geqslant 1$ for every $m \leqslant k/2$.

**Lemma 4.1.** *For any $0 \leqslant m \leqslant k/2$ and any fixed querying strategy that uses $t \leqslant \binom{n}{2}$ queries,*

$$\mathbb{E}[w_{k,m}(t)] \leqslant t2^{-(2k-2m-1)} \cdot \mathbb{E}[w_{k-2,m-1}(t)].$$

**Proof.** For every set $U \in \binom{[n]}{k}$, we say that $U$ is $m$-critical at query $s$ if $s$ is the smallest number for which $m(U, s) \geqslant m$. In particular, $U$ does not contribute to $w_{k,m}(t)$ until $t = s$, after which it contributes $w(U, t)$, which is a martingale. This means that if

$$w^*_{k,m}(s) := \sum_{|U|=k}{}' w(U, s),$$

where the sum is restricted to only sets $U$ which are $m$-critical at query $s$, then

$$\mathbb{E}[w_{k,m}(t) - w_{k,m}(t-1)] = \mathbb{E}[w^*_{k,m}(t)],$$

and so

$$\mathbb{E}[w_{k,m}(t)] = \sum_{s\leqslant t} \mathbb{E}[w^*_{k,m}(s)]. \tag{4.1}$$

Next we will show

$$w^*_{k,m}(s) \leqslant 2^{-(2k-2m-2)} w_{k-2,m-1}(s). \tag{4.2}$$

To see this, note that every $U$ that appears on the left side must contain the edge $(u, v)$ built after query $s$, since $m(U, s) > m(U, s-1)$. Furthermore, $U' = U\backslash\{u, v\}$ is a set with $k-2$ vertices and an $m-1$ matching. Finally, every edge in $U$ but not $U'$ is incident to $(u, v)$. It is easy to check that if $(u, v)$ is an edge that lies in every $m$-matching of $U$, then at most $2m-2$ other edges are incident to $(u, v)$. Thus there are at least $2(k-2) - (2m-2) = 2k-2m-2$ unqueried pairs in $U$ but not in $U'$, and

$$w(U, t) \leqslant 2^{-(2k-2m-2)} w(U', t).$$

Summing over all $m$-critical sets $U$, we get the desired inequality (4.2). Taking expectations of both sides,

$$\mathbb{E}[w^*_{k,m}(s)] \leqslant 2^{-(2k-2m-1)} \mathbb{E}[w_{k-2,m-1}(s)].$$

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

Note that we gained another factor of $1/2$ here because there is a $1/2$ chance that the query $(u, v)$ fails and $w^*_{k,m}(s) = 0$. Plugging this into (4.1), we get

$$\mathbb{E}[w_{k,m}(t)] \leqslant 2^{-(2k-2m-1)} \sum_{s \leqslant t} \mathbb{E}[w_{k-2,m-1}(s)].$$

The expectations on the right side are non-decreasing as a function of $s$, so we can bound this by

$$\mathbb{E}[w_{k,m}(t)] \leqslant 2^{-(2k-2m-1)} \sum_{s \leqslant t} \mathbb{E}[w_{k-2,m-1}(s)] \leqslant t 2^{-(2k-2m-1)} \cdot \mathbb{E}[w_{k-2,m-1}(t)]$$

as desired. □

Now we may iterate Lemma 4.1 until $m = 0$ to prove the following general bound.

**Lemma 4.2.** *For any $0 \leqslant m \leqslant k/2$ and any fixed querying strategy that uses $t \leqslant \binom{n}{2}$ queries,*

$$\mathbb{E}[w_{k,m}(t)] \leqslant t^m n^{k-2m} 2^{-\binom{k}{2}+m(m-1)}.$$

**Proof.** We induct on $m$. The base case $m = 0$ is just the unrestricted weight function

$$\mathbb{E}[w_{k,0}(t)] = \mathbb{E}[w_k(t)] = w_k(0) = \binom{n}{k} 2^{-\binom{k}{2}} \leqslant n^k 2^{-\binom{k}{2}},$$

for all $k$, as desired. Assuming the statement is true for some $m \geqslant 0$ and all $k \geqslant 2m$, Lemma 4.1 provides the inductive step for $m + 1$ and all $k \geqslant 2m + 2$. □

It remains to prove Theorem 1.3 using Lemma 4.2.

**Proof of Theorem 1.3.** Recall that $\mathbb{E}[w_{k,m}(t)]$ is an upper bound on the probability that one can find a $k$-clique in $t$ queries. By Lemma 4.2, we see that whenever $n, k, t$ are such that there exists $m \leqslant k/2$ for which

$$\mathbb{E}[w_{k,m}(t)] \leqslant t^m n^{k-2m} 2^{-\binom{k}{2}+m(m-1)} < \frac{1}{2},$$

then it is impossible to find a $k$-clique in $t$ queries in $G(n, 1/2)$ with probability at least $1/2$. It is cleaner to compute the base-2 logarithm of this quantity. Taking $t = n^\delta$ and $k = \alpha \lg n$ and writing $\ell = \lg n$ as a shorthand, we get

$$\begin{aligned}\lg\big(t^m n^{k-2m} 2^{-\binom{k}{2}+m(m-1)}\big) &= (\alpha\ell - m(2-\delta))\ell - \binom{\alpha\ell}{2} + m(m-1) \\ &\leqslant \left(\alpha - \frac{\alpha^2}{2}\right)\ell^2 - (2-\delta)m\ell + m^2 + O(\ell).\end{aligned}$$

If $m = c\ell$ where $c \leqslant \alpha/2$, then

$$\left(\alpha - \frac{\alpha^2}{2}\right)\ell^2 - (2-\delta)m\ell + m^2 = \left(\alpha - \frac{\alpha^2}{2} - (2-\delta)c + c^2\right)\ell^2$$

is minimized at $c = 2 - \delta/2$. Assuming that $\alpha \geqslant 2 - \delta$, we find that for this choice of $c$,

$$\lg\left(\mathbb{E}[w_{k,m}(t)]\right) \leqslant \left(\alpha - \frac{\alpha^2}{2} - \frac{(2-\delta)^2}{4}\right)\ell^2 + O(\ell).$$

In particular, this shows that whenever $\alpha \geqslant 2 - \delta$ satisfies

$$\alpha - \frac{\alpha^2}{2} - \frac{(2-\delta)^2}{4} < 0,$$

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

then for sufficiently large $n$ it is impossible to find a clique with $\alpha \lg n$ vertices in $n^{\delta}$ queries. Thus $\alpha_{\star}(\delta)$ is bounded above by the (larger) solution to the above quadratic, which is

$$\alpha_{+} = 1 + \sqrt{1 - \frac{(2-\delta)^2}{2}} > 2 - \delta,$$

as desired. □

## 5. Concluding remarks

The immediate question that arises from our work is to classify the graphs $H$ for which $f(H, p) = b^{d-o(1)}$. The natural first step is the case $d = 2$. To this end, we first establish a large family of 2-degenerate graphs $H$ for which $f(H, p) = O(b^{2-\varepsilon})$ for some $\varepsilon > 0$.

**Definition 4.** *We call a graph $H$ $(1, 1)$-degenerate if $H$ can be vertex-partitioned into induced subgraphs $T_1, \ldots, T_n$ which are trees, such that for all $k \in \{1, \ldots, n\}$ and all $v \in T_k$,*

$$\left| N(v) \cap \bigcup_{i=1}^{k-1} T_i \right| \leqslant 1.$$

It is easy to see that if $H$ is $(1, 1)$-degenerate, then $H$ is 2-degenerate. One can show by induction on the number of trees $n$ that if $H$ is $(1, 1)$-degenerate, then $f(H, p) = O(b^{2-\varepsilon})$ for some $\varepsilon > 0$. We prove this, and conjecture that the converse is true.

**Theorem 5.1.** *If $H$ is $(1, 1)$-degenerate, then $f(H, p) = O(b^{2-\varepsilon})$ for some $\varepsilon = \varepsilon(H) > 0$.*

**Proof.** We induct on the number of trees $n$ in the $(1, 1)$-degenerate partition $T_1, \ldots, T_n$ of $H$. The explicit value of $\varepsilon(H)$ we pick is $\varepsilon(H) := \max\{|V(T_1)|, \ldots, |V(T_n)|\}^{-1}$.

If $n = 1$, then $H$ is a tree so that $f(H, p) = O(b)$, as desired. Now say $n \geqslant 2$ and let $V(H) = V(H') \cup V(T_n)$. For $\varepsilon' = \varepsilon(H')$, we can build a copy of $H'$ with probability at least $3/4$ in some time $O(b^{2-\varepsilon'})$ by the inductive hypothesis. Let $\varepsilon = \min\{\varepsilon', |V(T_n)|^{-1}\}$. Now, for each $v \in V(T_n)$, there exists at most one $u \in V(H')$ so that $v$ and $u$ are adjacent. If there does not exist such a $u$, we let $C_v$ be a set of $b^{1-\varepsilon}$ previously unexplored vertices, and if there is, let $C_v$ be a set of $b^{1-\varepsilon}$ neighbours of $u$. We can find such a set $C_v$ in at most $O(b^{2-\varepsilon})$ queries (with constant probability).

Now query all edges between $C_v$ and $C_{v'}$ for every pair of vertices $v, v' \in V(T_n)$. This takes time $O(b^{2-2\varepsilon})$. We claim that there will exist a copy of $T_n$ with $v \in C_v$ for all $v \in V(T_n)$, because the expected number of copies of $T_n$ is of the order of $b^{(1-\varepsilon)|V(T_n)|} b^{|V(T_n)|-1} = \Theta(1)$, and it is a standard result that the threshold for containment of a tree is the same as the expectation threshold. This gives that we have at least one such copy of $T_n$ with probability at least $3/4$. This copy of $T_n$ extends the original copy of $H'$ to our desired copy of $H$, all with probability at least $1/2$ in time $O(b^{2-\varepsilon})$. □

**Conjecture 5.2.** *If $H$ is a 2-degenerate graph that is not $(1, 1)$-degenerate, then $f(H, p) = b^{2-o(1)}$.*

In the case $d = 2$, we were able to construct a particular 2-degenerate graph $H$ for which $f(H, p) \geqslant b^{d-o(1)}$. The existence of such graphs when $d \geqslant 3$ remains open.

**Conjecture 5.3.** *For all integers $d \geqslant 2$, there exists a $d$-degenerate graph $H$ for which $f(H, p) = b^{d-o(1)}$.*

There is a natural random process for constructing $d$-degenerate graphs on $n$ vertices. Namely, starting with a $K_d$, $n - d$ vertices are added one at a time, and each new vertex is given $d$ neighbours uniformly at random among the previous ones. If $n$ is sufficiently large, it is plausible

https://doi.org/10.1017/S0963548320000218 Published online by Cambridge University Press

that the random $d$-degenerate graph constructed in this manner should satisfy $f(H, p) = b^{d-o(1)}$ asymptotically almost surely.

## Acknowledgements

We would like to thank Jacob Fox, Huy Pham and Yuval Wigderson for helpful discussions. We would also like to thank an anonymous reviewer for helpful comments.

## References

[1] Ajtai, M., Komlós, J. and Szemerédi, E. (1981) The longest path in a random graph. *Combinatorica* **1** 1–12.
[2] Alon, N. and Spencer, J. H. (2008) *The Probabilistic Method*, third edition, Wiley.
[3] Bollobás, B. and Erdős, P. (1976) Cliques in random graphs. *Math. Proc. Cambridge Philos. Soc.* **80** 419–427.
[4] Burr, S. A. and Erdős, P. (1975) On the magnitude of generalized Ramsey numbers for graphs. In *Infinite and Finite Sets I*, Vol. 10 of Colloquia Mathematica Societatis János Bolyai, pp. 214–240, North-Holland.
[5] Conlon, D., Fox, J., Grinshpun, A. and He, X. (2019) Online Ramsey numbers and the subgraph query problem. In *Building Bridges II*, Vol. 28 of Bolyai Society Mathematical Studies, Springer.
[6] Feige, U., Gamarnik, D., Neeman, J., Rácz, M. Z. and Tetali, P. (2020) Finding cliques using few probes. *Random Struct. Algorithms* **56** 142–153.
[7] Ferber, A., Krivelevich, M., Sudakov, B. and Vieira, P. (2016) Finding Hamilton cycles in random graphs with few queries. *Random Struct. Algorithms* **49** 635–668.
[8] Ferber, A., Krivelevich, M., Sudakov, B. and Vieira, P. (2017) Finding paths in sparse random graphs requires many queries. *Random Struct. Algorithms* **50** 71–85.
[9] Frieze, A. and Kannan, R. (2008) A new approach to the planted clique problem. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, Vol. 2, pp. 187–198, Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
[10] Hefetz, D., Krivelevich, M., Stojakovic, M. and Szabó, T. (2014) *Positional Games*, Birkhäuser.
[11] Krivelevich, M. (2014) Positional games. *Proceedings of the International Congress of Mathematicians* **3** 355–379.
[12] Lee, C. (2017) Ramsey numbers of degenerate graphs. *Ann. of Math.* **185** 791–829.
[13] Pham, H. Personal communication.
[14] Rácz, M. Z. and Schiffer, B. (2020) Finding a planted clique by adaptive probing. *Random Struct. Algorithms* **56** 142–153.

**Cite this article:** Alweiss R, Ben Hamida C, He X and Moreira A (2021). On the subgraph query problem. *Combinatorics, Probability and Computing* **30**, 1–16. https://doi.org/10.1017/S0963548320000218