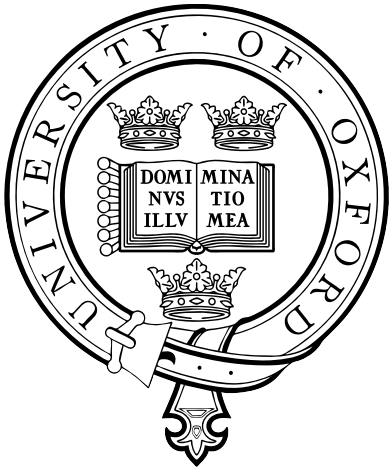
Cryptographic Applications of Graph Theoretic Constructions



Douglas Stebila

Pembroke College

University of Oxford

A thesis submitted for the degree of

Master of Science in Mathematics and the Foundations of Computer Science

Trinity Term 2004

Acknowledgements

I am grateful for the helpful discussions with my supervisor Prof D. J. A. Welsh and for his work with Prof C. McDiarmid in organizing the Combinatorial Theory seminar in the Maths Institute at the University of Oxford, from which the idea for this dissertation was first made known to me.

Abstract

While the difficulty of solving certain graph theoretical problems has been at the heart of important questions in complexity theory, the hard problems used for cryptographic purposes are usually number theoretic in nature. We explore the use of graph structures for cryptographic purposes.

Finding a hidden clique in an otherwise random graph has potential as a hard problem. We extend a previous result of Juels and Peinado (A. Juels and M. Peinado. Hiding cliques for cryptographic security. In *Proc. 9th Ann. ACM-SIAM Symp. on Discrete Algorithms*, pages 678–684, 1998.) to demonstrate that, assuming it is hard to find a clique of size $(1+\epsilon)\log_{1/p}(n)$ in a random graph from $\mathcal{G}_{n,p}$, it is also hard to find a hidden clique of the same size embedded in a graph from $\mathcal{G}_{n,p}$; this assumption has been an open problem for 30 years. Although our result holds asymptotically, the problem can be solved with non-trivial probability for graphs with, say, 1000 vertices. Hidden cliques can be used in a zero-knowledge protocol for authentication or in a generalized encryption scheme. We discuss the practicality of graph-based cryptographic systems and conclude that protocols based on hidden cliques exchange data structures of too large a size to be feasible.

We also report on the infeasibility of using knowledge of hidden Hamiltonian cycles or knowledge of graph colourings as the basis of a cryptosystem.

Contents

List of Figures

Chapter 1

Introduction

Cryptography is one of the oldest applied uses of mathematics, and is today an exciting confluence of mathematics, computer science, and even quantum computing. With the safety of military communications and financial transactions depending on the security of cryptosystems, it is important that the problems used be computationally intractable.

The goal of twentieth century cryptographers has been to mathematically characterize the difficulty of problems used for cryptographic purposes using the language of computational complexity from theoretical computer science [12]. Ironically, though cryptography has seen great commercial success and can now be found in devices of all sizes - from desktop computers to mobile phones to miniscule sensors - cryptographers have almost universally failed in the goal in characterizing the difficulty of the problems in question. The problem of factoring large numbers, for example, which is at the heart of the RSA [42] and Diffie-Hellman [13] asymmetric cryptosystems, has had its complexity class delineated, but the feasibility of problems in this complexity class has not been determined. The schemes used for symmetric cryptosystems, such as DES [39] and AES [40], are not even mathematically characterized.

While modern day cryptography has focused on number theoretic problems, computational complexity has often employed graph theoretic problems to characterize complexity classes. A natural question, then, is whether difficult graph theoretic problems can be used as the basis for cryptosystems. Surprisingly, this question has been addressed only sporadically and only recently. In this dissertation we collect together and extend existing discussions on the tractability of graph theoretic problems for cryptographic purposes and the practicality thereof.

Chapter 2 provides the necessary background in graph theory, complexity, and probability theory for the later discussions on hard problems and random graphs. Chapter 3 addresses the difficulty of problems involving cliques in graphs; the chapter includes the main theoretical result of this dissertation, an extension of a theorem of Juels and Peinado [26]. In Chapter 4, we discuss the use of other graph theoretic structures, such as Hamiltonian cycles and k-colourings, for hard problems. Chapter 5 describes two cryptographic schemes based on hidden cliques, and includes an original discussion on the practicality of graph-based security protocols. We conclude in Chapter 6 with some open questions related to hard graph theoretic problems for cryptography.

Chapter 2

Background

2.1 Graph theory

A *graph* $G = (V, E)$ is an ordered pair of sets, consisting of a set V of *vertices* and a set $E \subseteq V \times V$ of *edges*. For simplicity, we assume $V \cap E = \emptyset$. We often abbreviate an edge (u, v) as uv. If E is a multiset, then G is said to be a *multigraph*. If v is a vertex in G, the edge vv is called a *loop*. If the edge set of G contains no loops and is not a multiset (that is, the graph has no parallel edges), then G is a *simple graph*. In the remainder of this paper, we use the term "graph" to denote a simple graph. Graphs are often depicted with dots representing a vertex and lines between the dots representing edges. The manner in which the dots and lines are drawn is not material and does not convey any additional information.

The vertex set of a graph G is denoted $V(G)$ and the corresponding edge set is $E(G)$. We often write $v \in G$ as a shorthand for $v \in V(G)$ and $e \in G$ as a shorthand for $e \in E(G)$ for vertices and edges respectively. Two vertices u and v in G are *adjacent* (or are *neighbours*) if $uv \in E(G)$, and this is denoted $u \sim v$; two vertices which are not adjacent are said to be *independent*.

A graph $H = (W, F)$ is a *subgraph* of a graph $G = (V, E)$, denoted $H \leq G$, if $W \subseteq V$,

$F \subseteq E$, and F only contains edges between vertices in W. The subgraph *induced* by a vertex set $X \subseteq V(G)$ in a graph G is $G[X] = (X, E')$, where $E' = \{uv \in E(G) : u, v \in X\}$. A *spanning subgraph* G' of a graph G is one in which $V(G') = V(G)$.

A *complete* graph K_n on n vertices is a graph in which every pair of vertices is adjacent. A *clique* K in a graph G is a subset of vertices for which the induced subgraph $G[K]$ is complete. A *maximal clique* is a clique for which there is no vertex $v \in V(G) \setminus K$ such that $K \cup \{v\}$ is still a clique of G. The *clique number* cl(G) is the maximum size of a clique in G. Let $C_k(G)$ denote the number of cliques of size k in G. Figure 2.1 shows a graph on 7 vertices for which the vertices $\{1, 2, 5, 6, 7\}$ form a clique of size 5.

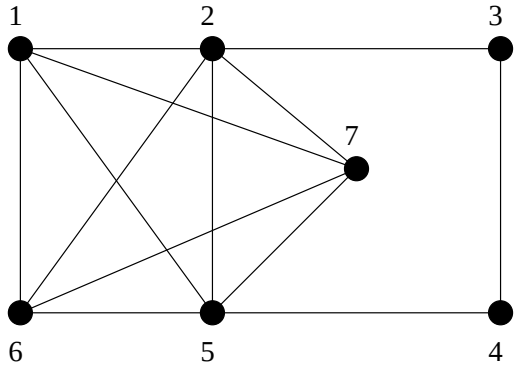


Figure 2.1: A graph on 7 vertices containing a 5-clique.

The *complement* of a graph $G = (V, E)$ is the graph $\overline{G} = (V, \overline{E})$, where $\overline{E} = \{uv : u, v \in V, uv \notin E, u \neq v\}$. An *independent set* is a set of vertices all of which are pairwise independent; that is, an independent set in a graph is a clique in the complement of that graph.

The *degree* of a vertex $v \in G$, denoted $\deg_G(v)$, or $\deg(v)$ where there is no chance of ambiguity, is the number of neighbours of v in G. The *minimum degree* of a graph G is $\delta(G) = \min\{\deg_G(v) : v \in V(G)\}$ and the *maximum degree* of a graph G is $\Delta(G) = \max\{\deg_G(v) : v \in V(G)\}$. A graph is *k-regular* if every vertex has degree k.

2.2 Random graphs

The theory of random graphs was first developed by Erdős and Rényi [15, 16, 17, 18] as a tool to study extremal problems in graph theory. In most models of random graphs, we define a vertex set $V = \{1, \ldots, n\}$, and construct an edge set in which each edge is present with some probability. The two most frequently occurring models of random graphs are $\mathcal{G}_{n,M}$ and $\mathcal{G}_{n,p}$.

$\mathcal{G}_{n,M}$ is the probability distriubtion of all graphs with n vertices and M edges; each such graph occurs with equal probability, namely $\binom{n}{M}^{-1}$.

$\mathcal{G}_{n,p}$ is the probability distribution of graphs with n vertices in which each edge occurs independently with probability p, $0 < p < 1$. Let G be the random variable for a graph drawn from $\mathcal{G}_{n,p}$. If $|E(G)| = m$, then

$$\mathbb{P}(G) = p^m(1-p)^{\binom{n}{2}-m}.$$

The expected size of the edge set of a graph G chosen from $\mathcal{G}_{n,p}$ is

$$\mathbb{E}(|E(G)|) = \frac{(n-1)np}{2}.$$

For a graph G, we use the notation $\mathbb{P}_{\mathcal{G}_{n,p}}(G)$ and $\mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(G)$ to denote the probability of the graph G be chosen from the probability distributions $\mathcal{G}_{n,p}$ and $\mathcal{G}_{n,p}\langle k\rangle$ respectively.

2.2.1 Cliques in random graphs

We now turn to the study of cliques in random graphs from the $\mathcal{G}_{n,p}$ model. Let $K_r(G)$ be a random variable on $\mathcal{G}_{n,p}$ counting the number of r-cliques in a graph G.

Then

$$\mathbb{E}(K_r(G)) = \binom{n}{r} p^{\binom{r}{2}}.$$

In most random graphs, the sizes of the cliques are distributed in a small range, as demonstrated by the following theorem [4, XI.1.4]:

Theorem 1. *Let $0 < p < 1$, $0 < \epsilon < \frac{1}{2}$. Almost every random graph in $\mathcal{G}_{n,p}$ has a clique of size r for*

$$(1+\epsilon)\log_{1/p} n < r < (2-\epsilon)\log_{1/p} n$$

and no clique of size r for

$$r < (1-\epsilon)\log_{1/p} n \text{ or } r > (2+\epsilon)\log_{1/p} n.$$

2.3 Complexity

We use the standard terminology from computational complexity theory. An algorithm runs in *polynomial time* if it halts within $n^{O(1)}$ steps on any input of size n, where $f(n) \in O(g(n))$ means that there exist constants c, n_0 such that, for all $n > n_0$, $cf(n) < g(n)$. If $f(n) \in O(g(n))$, then $g(n) \in \Omega(f(n))$. If $f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n))$, then $f(n) \in \Theta(g(n))$. Finally, $f(n) \in o(g(n))$ if $\lim_{n\to\infty} f(n)/g(n) = 0$.

The class P consists of all languages L that have a polynomial time algorithm A accepting the language: for any input x, $x \in L$ if and only if $A(x)$ accepts.

NP consists of all languages L that have a polynomial time algorithm A such that for all $x \in L$, there exists a string y bounded in length by a polynomial in $|x|$ such that $A(x, y)$ accepts, and for all $x \notin L$, $A(x, y)$ rejects for all strings y. A language L is NP-*hard* if, for all languages $L' \in \mathsf{NP}$, there exists a Turing reduction from L' to L,

that is, if there exists a Turing-computable function f such that $x \in L'$ if and only if $f(x) \in L$. A language that is both NP-hard and is a member of NP is said to be NP-*complete*.

RP is the class of all languages L for which there is a randomized algorithm A running in worst-case polynomial time such that, for any input x,

$$x \in L \implies \mathbb{P}(A(x) \text{ accepts}) \geq \frac{1}{2},$$

$$x \notin L \implies \mathbb{P}(A(x) \text{ accepts}) = 0.$$

A randomized algorithm that solves an RP problem is said to have *one-sided error*. A language belongs to the complementary class coRP if $\Sigma^* \setminus L \in \mathsf{RP}$. Such a language also has one-sided error, having non-zero probability of accepting a string not in the language.

A randomized algorithm A with *zero-sided error* for a language L is one such that, for any string x,

$$x \in L \implies \mathbb{P}(A(x) \text{ accepts}) = 1,$$

$$x \notin L \implies \mathbb{P}(A(x) \text{ accepts}) = 0.$$

The class of languages accepted by a randomized algorithm with zero-sided error running in expected polynomial time is called ZPP.

The potential use of NP-complete problems as trapdoor one-way functions in the construction of public-key cryptosystems was mentioned in the earliest papers on public-key cryptography [13]. Although NP-complete problems do not have polynomial time solutions assuming $\mathsf{P} \neq \mathsf{NP}$, this is only true in the worst-case. There are many NP-complete problems for which there exist polynomial time algorithms that recognize "most" strings in the language, and thus these problems are generally unsuitable for cryptographic purposes. Levin [33] introduced the notion of average case

NP-complete problems, problems for which the uniform distribution of instances has no on average polynomial time algorithm unless every NP problem with every simple probability distribution also has an on average polynomial time algorithm. Levin provides one example of an average case complete problem, namely tiling, but there are no others known. Furthermore, there are no examples of using the difficulty of solving random instances of average case complete problems for cryptographic purposes.

For many NP-complete problems, it is often possible to give a good approximate solution. The *performance ratio* of an approximation algorithm A for a maximization problem π is the ratio $\frac{\text{opt}(\pi)}{A(\pi)}$, where $\text{opt}(\pi)$ is the optimal solution of π, and $A(\pi)$ is the solution found by A, computed over all possible inputs; for a minimization problem the performance ratio is $\frac{A(\pi)}{\text{opt}(\pi)}$. The best possible approximation algorithm is a *fully polynomial-time approximation scheme* (FPAS) which, for any $\epsilon > 0$, produces a performance ratio of $1 + \epsilon$ in time bounded by a polynomial in n and ϵ^{-1}, where n is the size of the input.

There is a rich history and literature covering complexity theory. A thorough treatment of randomized algorithms is given by Motwani and Raghavan [37]. Hemaspaandra and Ogihara [24] provide a comprehensive and up-to-date collection of results concerning complexity classes.

2.4 Probability theory

The analysis of randomized algorithms and random graphs often uses the language of probability theory. Some standard results that are especially useful in this analysis are listed below, but first we provide our notation. The probability of an event A is denoted $\mathbb{P}(A)$, and the expectation of the event is $\mathbb{E}(A)$. If $\mathcal{P}$ is a probability distribution, then $A \in_R \mathcal{P}$ denotes A being chosen according to the probability distribution

$\mathcal{P}$. If S is a set, then $A \in_R S$ denotes A being chosen from S according to the uniform distribution, and $A \subseteq_R S$ denotes A being chosen uniformly at random as a subset of S for a given size of A.

Theorem 2 (Chebyshev's Inequality). *Let X be a random variable. Then for $a > 0$,*

$$\mathbb{P}(|X| \geq a) \leq \frac{\mathbb{E}(X^2)}{a^2}.$$

Theorem 3 (The Chernoff Bound). *Let $X_1, \ldots, X_n$ be independent random variables taking the values 0 and 1 with probabilities $1 - p$ and p, respectively. Let $X = \sum_{i=1}^{n} X_i$. For all θ such that $0 \leq \theta \leq 1$,*

$$\mathbb{P}(X \geq (1 + \theta)pn) \leq e^{-\theta^2 pn/3}.$$

We say that an event A occurs with *high probability* if $\mathbb{P}(A) = 1 - o(1)$. If P is a property and S is a set, we say that *almost every* element of S has property P if the proportion of elements of S having property P is $1 - o(1)$.

A random variable X has a *binomial distribution* with parameters n and p if its probability mass function satisfies

$$f(k) = \mathbb{P}(X = k) = \begin{cases} \binom{n}{k} p^k (1-p)^{n-k}, & k = 0, 1, \ldots, n, \\ 0, & \text{otherwise.} \end{cases}$$

The expected value of X is $\mathbb{E}(X) = np$ and the variance is $\mathrm{Var}(X) = np(1 - p)$.

2.5 Cryptography

A zero-knowledge proof "is a proof that yields nothing but its validity" [21]. It can be defined formally as follows.

Introduced by Goldwasser et al. [22], an *interactive proof system* for a property $P(x)$ of an input x is a two-party protocol between a prover $\mathcal{P}$ and a probabilistic polynomial-time verifier $\mathcal{V}$ such that the following holds: for every constant $c > 0$ and all sufficiently long x,

$$P(x) = \text{true} \implies \mathbb{P}(\mathcal{V} \text{ accepts when interacting with } \mathcal{P}) \geq 1 - |x|^{-c}$$

and for every possible prover $\mathcal{P}^*$,

$$P(x) = \text{false} \implies \mathbb{P}(\mathcal{V} \text{ accepts when interacting with } \mathcal{P}^*) \leq |x|^{-c}$$

where the probabilities are taken only over $\mathcal{V}$'s coin tosses.

A *zero-knowledge proof system* for a property $P(x)$ of an input x is an interactive proof between $\mathcal{P}$ and $\mathcal{V}$ such that, for every possible probability polynomial-time verifier $\mathcal{V}^*$, the set of transcripts of the interaction of $\mathcal{P}$ and $\mathcal{V}^*$ are indistinguishable in polynomial time from any set of "forged" transcripts from a probabilistic expected polynomial-time Turing machine $M_{\mathcal{V}^*}$. A *transcript* of an interaction consists of all information transmitted during the interaction plus the results of all of the $\mathcal{V}$'s random coin tosses.

A common tool in zero-knowledge proofs is bit commitment. In a *bit commitment* scheme, $\mathcal{A}$ commits to a value x using a related evidence value y that she gives to $\mathcal{B}$; $\mathcal{B}$ cannot learn anything about x from y, but $\mathcal{A}$ can later prove to $\mathcal{B}$ that she originally committed to x. Formally, a commitment scheme is *binding* if $\mathcal{A}$ cannot convince $\mathcal{B}$ that she committed to any value other than x given evidence y, and it is *concealing* if $\mathcal{B}$ can learn nothing about x only given y. A commitment scheme is *secure* if it is both binding and concealing.

There are a number of ways to implement commitment schemes. Naor [38] demonstrates a bit commitment scheme using pseudorandomness, while Crépeau [11] gives

a protocol for bit commitment using a noisy binary symmetric channel; there are many other schemes for bit commitment.[1] Brassard et al. [7] provide a protocol for bit commitment in the setting of quantum cryptography, although this protocol was later proved not to be unconditionally secure by Mayers [35]. In fact, Brassard et al. [8] demonstrate that, in the face of a quantum computer, any classical (i.e., non-quantum) bit commitment scheme cannot be unconditionally secure.

We often describe cryptographic protocols occurring between multiple parties. The characters participating in the protocol are often Alice, denoted by $\mathcal{A}$, Bob, denoted by $\mathcal{B}$, and Eve, who is often an adversary, denoted by $\mathcal{E}$.

[1] For a comprehensive list of papers related to bit commitment schemes, see Lipmaa [34].

Chapter 3

Cliques

In this chapter we explore the difficulty of problems related to finding cliques in graphs and suitability of such problems for a cryptographic system.

3.1 The MAXCLIQUE problem

The problem of finding the size of the maximum clique in a graph is NP-hard, and is one of Karp's original NP-hard problems [27]. This problem is often called the MAXCLIQUE problem.

The problem of approximating the size of the maximum clique in a graph was first considered by Karp [28] in 1976. The best known polynomial-time approximation algorithm for finding the size of the maximum clique is that of Boppana and Halldórsson [6], which provides a performance ratio of $O(n/\log^2 n)$, using an algorithm belonging to a class known as subgraph-excluding algorithms. The strongest current negative results on approximating clique are due to Håstad [23], who shows that unless $\mathsf{P} = \mathsf{NP}$, MAXCLIQUE cannot be approximated in polynomial time within a performance ratio of $n^{1/3-\delta}$, $\delta > 0$, and furthermore that unless $\mathsf{NP} = \mathsf{coRP}$, MAXCLIQUE cannot be approximated in polynomial time within a performance ratio of $n^{1/2-\delta}$, $\delta > 0$.

Karp [28] posited that there is no polynomial time algorithm capable of finding with significant probability a clique of size $(1+\epsilon)\log_2 n$ in a random graph from $\mathcal{G}_{n,\frac{1}{2}}$, for any $\epsilon > 0$. This open problem was further extended by the conjecture of Jerrum [25] that there is no randomized polynomial time algorithm that finds a clique of size $1.01\log_2 n$ in a random graph chosen from $\mathcal{G}_{n,\frac{1}{2}}$ that is known to contain a clique of size $n^{0.49}$. Jerrum showed that in certain cases the Metropolis algorithm, a common tool in combinatorial search problems, cannot find a clique of size $(1+\epsilon)\log_2 n$ for any constant $\epsilon > 0$ in polynomial time, even if a clique of size $n^{1/2-\delta}$, $\delta > 0$, is randomly hidden in the graph.

When the graph is known to contain a large clique of size greater than $n/k + m$, k a fixed integer and $m > 0$, then a clique of size $\Omega(m^{3/(k+1)}/\log^c n)$ times a polynomial in $\log n$, for a constant c, can be found in polynomial time [1].

3.2 Clique hiding models

If, as in conjectured in the previous section, finding a reasonably small clique in a random graph from $\mathcal{G}_{n,p}$ is hard, then a party who wishes to use knowledge of a hidden clique in a cryptographic protocol has no better chance of finding a clique to use than an adversary who wishes to attack the protocol. That is, we need a method in which a graph can be constructed which contains a clique of which one party has knowledge but that other parties cannot easily discover. In this section, we discuss various techniques for randomly constructing graphs that contain cliques of a required size.

3.2.1 Naïve clique creation

The simplest method of hiding a clique in a random graph is to choose the clique vertices and then build a random graph around them, a technique we will call *naïve clique creation*.[1] A clique of size k is hidden inside a random graph as follows. Let $V = \{1, \ldots, n\}$. Let $K \subseteq_R V$ be a randomly chosen subset of k vertices. Construct the (loopless) edge set E such that

$$\mathbb{P}(uv \in E) = \begin{cases} 1, & \text{if } u, v \in K, \\ p, & \text{otherwise.} \end{cases}$$

Let $\mathcal{G}_{n,p}\langle k \rangle$ denote the probability distribution of graphs formed in this manner.

Let $G \in_R \mathcal{G}_{n,p}\langle k \rangle$ and let K be a hidden clique of size k. The expected value of the degree of a vertex is

$$\mathbb{E}(\deg(v)) = \begin{cases} (n-k)p + k - 1, & v \in K, \\ (n-1)p, & v \in V \setminus K. \end{cases} \tag{3.1}$$

The variance of the degree of a non-clique vertex v is

$$\mathrm{Var}(\deg(v)) = np(1-p). \tag{3.2}$$

3.2.2 Split naïve clique creation

As seen in equation 3.1, the expected degree of vertices in the hidden clique is different from the expected degree of vertices outside the hidden clique. The difference in expected degree can sometimes be used as a heuristic to detect the vertices of the hidden clique, as shown in section 3.3.2.

We now introduce another clique hiding technique called *split naïve clique creation*. A clique of size k is hidden in a random graph as follows. Let $V = \{1, \ldots, n\}$. Let

[1] This name is based on the terminology used in Brockington and Culberson [9].

$K \subseteq_R V$ be of size k. Construct the edge set E such that

$$\mathbb{P}(uv \in E) = \begin{cases} 1, & \text{if } u, v \in K, \\ r, & \text{if } u \in K, v \notin K, \\ p, & \text{if } u, v \notin K. \end{cases}$$

Let $\mathcal{G}_{n,(p,r)}\langle k \rangle$ denote the probability distribution of graphs formed in this manner.

Let $G \in_R \mathcal{G}_{n,(p,r)}\langle k \rangle$ and let K be a hidden clique of size k. The expected value of the degree of a vertex is

$$\mathbb{E}(\deg(v)) = \begin{cases} (n-k)r + k - 1, & v \in K, \\ (n-k-1)p + kr, & v \notin K. \end{cases} \tag{3.3}$$

Ideally, we would like to construct random graphs with hidden cliques such that the probability distribution of the degree of a vertex is independent of whether the vertex is in the hidden clique or not. Using equation 3.3 above, we can easily construct random graphs where the expected degree of a vertex is independent of whether the vertex is in a hidden clique or not: this is just a special case of split naïve clique creation. Given a value of r, an appropriate value for p can be determined by equating the two cases of equation 3.3 and solving for p to get

$$p = \frac{(n-2k)r + k - 1}{n - k - 1}. \tag{3.4}$$

Example. Let $n = 100$, $k = \lfloor 3/2 \log_2 n \rfloor = 9$, $r = 1/2$. Then for the vertices of a graph from $\mathcal{G}_{n,(p,r)}\langle k \rangle$ to have uniform expected degree, choose

$$p = \frac{(100-18)\frac{1}{2} + 8}{90} = \frac{49}{90}.$$

We can make an alternate formulation of split naïve clique creation to give a simpler expression for p and to provide a more intuitive construction process. In our original formulation of split naïve clique creation, the graph was constructed by selecting the clique vertices, inserting all clique edges, then inserting edges incident with one clique

vertex with probability r and inserting edges incident with no clique vertices with another probability p. We could instead view the construction process as selecting a graph from $\mathcal{G}_{n,r}\langle k\rangle$ and then inserting absent edges that are incident with no clique vertices with additional probability p.

In this alternative formulation, the edge set is constructed such that

$$\mathbb{P}\left(uv \in E\right) = \begin{cases} 1, & \text{if } u, v \in K, \\ r, & \text{if } u \in K, v \notin K, \\ r + (1-r)p, & \text{if } u, v \notin K. \end{cases}$$

Let $\mathcal{G}_{n,(p,r)}\langle k\rangle'$ denote the probability distribution of graphs formed in this manner. The expected value of the degree of a vertex is

$$\mathbb{E}(\deg(v)) = \begin{cases} (n-k)r + k - 1, & v \in K, \\ (n-k-1)(r + p(1-r)) + kr, & v \notin K. \end{cases} \tag{3.5}$$

To achieve uniform expected vertex degree for graphs from $\mathcal{G}_{n,(p,r)}\langle k\rangle'$, we need

$$p = \frac{k-1}{n-k-1}. \tag{3.6}$$

We note that, for a fixed value of r and p chosen appropriately for each model, $\mathcal{G}_{n,(p,r)}\langle k\rangle$ and $\mathcal{G}_{n,(p,r)}\langle k\rangle'$ yield the same probability distribution.

We use the term *quasi-random graph* to refer to a graph formed in a manner such as naïve clique creation or split naïve clique creation. That is, a quasi-random graph is one that is formed by a random process, but not necessarily the process corresponding to graphs in $\mathcal{G}_{n,p}$.

3.3 Finding hidden cliques

Many techniques for finding hidden cliques in a quasi-random graph rely on the expected value and variance of the degrees of the vertices of the graph, as we computed in the previous section.

3.3.1 The naïve method

The naïve method for searching for a clique of size k in a graph from $\mathcal{G}_{n,p}\langle k\rangle$ would be to consider every subset of vertices of size k and check if it is a clique. In fact, we only need to consider vertices that have degree at least k. The degrees of the non-hidden clique vertices are distributed according to a binomial distribution with mean $(n-1)p$ and variance $np(1-p)$, and vertices of the hidden clique have higher expected degree, so, by Chebyshev's inequality, the number of vertices with degree at least $k = (1+\epsilon)\log_p n$ is $O(n)$. Hence, the number of subsets to try is

$$O\left(\binom{n}{(1+\epsilon)\log_{1/p} n}\right).$$

Section 5.3 contains numerical information concerning the parameter size needed to ensure security against clique finding using the naïve method.

3.3.2 Asymptotic results

There are various results concerning the ability to find large hidden cliques in random graphs.

A consideration of the expected values and variance of the degrees of vertices in a graph from $\mathcal{G}_{n,p}\langle k\rangle$ shows that if $k \geq c\sqrt{n\log n}$, for a sufficiently large constant $c > 0$, then, for every graph G chosen from $\mathcal{G}_{n,p}\langle k\rangle$, with high probability any vertex of the hidden clique K has larger degree than a vertex in $V \setminus K$. Kučera [31] explores this analysis further.

Alon et al. [2] give a polynomial time algorithm that with high probability finds for all $k > c\sqrt{n}$, $c > 0$ a constant, the unique largest clique of size k in a random graph from $\mathcal{G}_{n,\frac{1}{2}}\langle k\rangle$. Their result can be extended to $\mathcal{G}_{n,p}\langle k\rangle$.

Feige and Krauthgamer [19] also give a polynomial time algorithm that with high probability finds a hidden clique of size $k = \Omega(\sqrt{n})$ in $\mathcal{G}_{n,p}\langle k\rangle$. Whereas the proof of Alon et al. used spectral analysis, Feige and Krauthgamer give a different algorithm based on the so-called Lovász theta function. Their algorithm additionally certifies the optimality of the found hidden clique. Furthermore, their technique applies to semirandom graphs. A *semirandom graph* G^* is constructed as follows:

1. Choose a random graph G from $\mathcal{G}_{n,p}\langle k\rangle$.

2. Choose $Q \subseteq_R V(G)$, $|Q| = k$. Let $G_{\min} = (V, E(Q))$ be the graph with just the n vertices $V(G)$ and the edges of the clique Q. Let $G_{\max} = G\langle Q\rangle$ be the graph formed by embedding the clique Q in the graph G.

3. Choose G^* "sandwiched" between $G_{\min}$ and $G_{\max}$; that is, choose G^* with vertex set $V(G)$ and edge set such that $E(G_{\min}) \subseteq E(G^*) \subseteq E(G_{\max})$. The edge set of G^* may be chosen uniformly at random or by some other process to model various adversary to attack methods.

Intuitively, a semirandom graph is a graph from which an adversary has been able to remove some edges.

There are currently no results characterising the difficulty of finding hidden cliques of size $k = o(\sqrt{n})$ in random graphs from $\mathcal{G}_{n,p}\langle k\rangle$. Jerrum [25] showed that the Metropolis algorithm, one tool for combinatorial search problems, cannot find a clique of size $k = o(\sqrt{n})$ in graphs from $\mathcal{G}_{n,p}\langle k\rangle$.

In a paper concerning his graph generalized encryption scheme, Kučera [30] proves that certain common heuristic algorithms will, asymptotically, fail with high probability to find a clique of size $k = o(\sqrt{n})$, in a graph from $\mathcal{G}_{n,p}\langle k\rangle$ (although Kučera's work is phrased in terms of independent sets instead of cliques).

3.4 A Theorem of Juels and Peinado

Although the results of the previous section demonstrate that large hidden cliques in $\mathcal{G}_{n,p}\langle k\rangle$ can often be found, there are no results about the difficulty of finding smaller hidden cliques, say cliques with sizes in the range $[(1+\epsilon)\log_b n, (2-\epsilon)\log_b n]$, $\epsilon > 0$. As reported in section 3.1, there is a long standing conjecture by Karp [28] about the hardness of finding a clique with size in the aforementioned range in a graph from $\mathcal{G}_{n,p}$, but not in a graph from $\mathcal{G}_{n,p}\langle k\rangle$.

In this section we prove a result relating the difficulty of finding a clique of the aforementioned size in a graph from $\mathcal{G}_{n,p}\langle k\rangle$ to that of finding a clique of the same size in a graph from $\mathcal{G}_{n,p}$. In particular, we provide an extension of a result of Juels and Peinado [26]. Informally, this result demonstrates that a polynomial-time algorithm that finds a clique of the aforementioned size with probability inversely proportional to a polynomial in n in a graph from $\mathcal{G}_{n,p}\langle k\rangle$ will also find a clique of the same size with probability inversely proportional to a (possibly different) polynomial in n in a graph from $\mathcal{G}_{n,p}$. More concisely, it is no easier to find a clique in a graph from $\mathcal{G}_{n,p}\langle k\rangle$ than it is to find a clique in a graph from $\mathcal{G}_{n,p}$. Thus, Karp's original conjecture can be extended to include the problem of finding cliques in graphs from $\mathcal{G}_{n,p}\langle k\rangle$.

The original proof of Juels and Peinado [26] demonstrates the result for the case $p = 1/2$. We provide an extension of the original proof for the case where p is an arbitrary constant.

3.4.1 Sketch of proof

We begin by showing that when the number of cliques in G is close to the expected number of cliques in the random graph G', the probability of choosing G from $\mathcal{G}_{n,p}\langle k\rangle$

is close to the probability of choosing G from $\mathcal{G}_{n,p}$. We also show that the variance of the number of cliques in a graph from $\mathcal{G}_{n,p}\langle k\rangle$ is small. Combining these two results, we show that most graphs in $\mathcal{G}_{n,p}\langle k\rangle$ are "good" in the sense that the number of cliques is close to the expected number of cliques. We can then reduce the proportion of "bad" graphs to an arbitrarily small amount, and so any algorithm that finds a good graph will do so with non-negligible probability.

3.4.2 Proof of main theorem

Lemma 1. Let G be chosen uniformly at random from $\mathcal{G}_{n,p}\langle k\rangle$. Then

$$\mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(G) = \mathbb{P}_{\mathcal{G}_{n,p}}(G)\frac{C_k(G)}{\mathbb{E}(C_k(\mathcal{G}_{n,p}))}.$$

Proof. The process of selecting a graph G from $\mathcal{G}_{n,p}\langle k\rangle$ can be viewed as the process of selecting a graph G' from $\mathcal{G}_{n,p}$, selecting a random set K of size k of vertices on which to plant a clique, and then embedding the clique. We must determine the probability that the graph constructed in this manner from $\mathcal{G}_{n,p}$ corresponds exactly to the graph G.

The probability that the clique K embedded into G' corresponds to a clique in G is $C_k(G)/\binom{n}{k}$.

The edges of G' not in K must also correspond exactly to the edges in G not in K. That is, for each pair of distinct vertices u, v, both not in K, $uv \in E(G)$ if and only if $uv \in E(G')$. Let $|E(G)| = m = \binom{k}{2} + \ell$; that is, there are $\binom{k}{2}$ clique edges and ℓ non-clique edges. The probability that the edges not in K match exactly is

$$p^{\ell}(1-p)^{\binom{n}{2}-\binom{k}{2}-\ell}.$$

Hence,

$$\mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(G) = \frac{C_k(G)}{\binom{n}{k}}p^{\ell}(1-p)^{\binom{n}{2}-\binom{k}{2}-\ell}.$$

By Bollobás [4, XI.I.(1)], $\mathbb{E}(C_k(\mathcal{G}_{n,p})) = \binom{n}{k}p^{\binom{k}{2}}$. Additionally, $\mathbb{P}_{\mathcal{G}_{n,p}}(G) = p^m(1-p)^{\binom{n}{2}-m}$. Thus,

$$\mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(G) = \mathbb{P}_{\mathcal{G}_{n,p}}(G)\frac{C_k(G)}{\mathbb{E}(C_k(\mathcal{G}_{n,p}))}.$$

■

Lemma 2. Let $G \in \mathcal{G}_{n,p}$. Then

$$\mathrm{Var}(C_k(G)) = O(n^4 \log n)\mathbb{E}(C_k(G))^2.$$

Proof. Consider pairs of cliques with ℓ vertices in common, as in Bollobás [4, XI.I.2]. The probability that G contains a given pair of k-cliques having ℓ vertices in common is $p^{2\binom{k}{2}-\binom{\ell}{2}}$. Then

$$\mathbb{E}(C_k(G)^2) = \sum_{\ell=0}^{k}\binom{n}{k}\binom{k}{l}\binom{n-k}{k-l}p^{2\binom{k}{2}-\binom{\ell}{2}}.$$

Consider the quotient

$$\frac{\mathbb{E}(C_k(G)^2)}{\mathbb{E}(C_k(G))^2} = \sum_{\ell=0}^{k}\frac{\binom{k}{\ell}\binom{n-k}{k-\ell}}{\binom{n}{k}p^{\binom{\ell}{2}}}.$$

Let the ℓth term of the sum be denoted by s_ℓ. Now, $s_0 = \binom{n-k}{k}/\binom{n}{k} < 1$. Recalling the basic identity that

$$\left(\frac{a}{b}\right)^b \le \binom{a}{b} \le \left(\frac{ae}{b}\right)^b$$

we find that, for $\ell = 1, \ldots, k$,

$$\begin{aligned}
s_\ell &\le \left(\frac{ke}{\ell}\right)^\ell \left(\frac{(n-k)e}{k-l}\right)^{k-\ell}\left(\frac{k}{n}\right)^k p^{-\binom{\ell}{2}} \\
&= \left(\frac{k^2}{\ell}\right)^\ell \left(\frac{(n-k)k}{k-\ell}\right)^{k-\ell}\left(\frac{e}{n}\right)^k p^{-\binom{\ell}{2}} \\
&= \left(\frac{k}{k-\ell}\right)^{k-\ell} e^k p^{-\binom{\ell}{2}}\left(\frac{k^2}{n}\right)^\ell \left(\frac{n-k}{n}\right)^{k-\ell}\frac{1}{\ell^\ell} \\
&< \left(\frac{k}{k-\ell}\right)^{k-\ell} e^k p^{-\binom{\ell}{2}}\left(\frac{k^2}{n}\right)^\ell .
\end{aligned}$$

Now,

$$\left(\frac{k}{k-\ell}\right)^{k-\ell} = \left(1+\frac{\ell}{k-\ell}\right)^{k-\ell} \le e^\ell < e^k < e^{(2-\delta)\log_{1/p} n} \in o(n^2).$$

Similarly, $e^k \in o(n^2)$, and

$$\begin{aligned}\log_{1/p}\left(p^{-\binom{\ell}{2}}\left(\frac{k^2}{n}\right)^{\ell}\right) &= -\binom{\ell}{2}\log_{1/p} p + 2\ell \log_{1/p} k - \ell \log_{1/p} n \\ &= \binom{\ell}{2} + 2\ell \log_{1/p} k - \ell \log_{1/p} n.\end{aligned}$$

But $\ell < k < 2\log_{1/p} n$ and $\binom{\ell}{2} < \frac{\ell^2}{2} < \ell \log_{1/p} n$, so

$$\log_{1/p}\left(p^{-\binom{\ell}{2}}\left(\frac{k^2}{n}\right)^{\ell}\right) < 0.$$

Thus,

$$p^{-\binom{\ell}{2}}\left(\frac{k^2}{n}\right)^{\ell} < 1.$$

Hence, $s_\ell \in O(n^4)$ for $1 \leq \ell \leq k$. Finally,

$$\begin{aligned}\mathrm{Var}(C_k(G)) &= \mathbb{E}(C_k(G)^2) - \mathbb{E}(C_k(G))^2 \\ &= \sum_{\ell=0}^{k} \frac{\binom{k}{\ell}\binom{n-k}{k-\ell}}{\binom{n}{k} p^{\binom{\ell}{2}}} \mathbb{E}(C_k(G))^2 - \mathbb{E}(C_k(G))^2 \\ &\in O(n^4 \log_b n - 1)\mathbb{E}(C_k(G))^2,\end{aligned}$$

which yields the desired result. ∎

Lemma 3 demonstrates that the set of so-called "bad" graphs, for which the number of cliques is much larger than the expected number of cliques, is a small fraction of $\mathcal{G}_{n,p}$. The next lemma demonstrates that the bad graphs also constitute just a small fraction of $\mathcal{G}_{n,p}\langle k\rangle$.

Lemma 3. Let $G \in \mathcal{G}_{n,p}$. Let $Z = \{G' : C_k(G') > n^{2h}\mathbb{E}(C_k(G))\}$, for some constant $h > 0$. Then, for any constant $\epsilon > 0$, $\mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(Z) \in O(n^{-h+4+\epsilon})$.

Proof. Let $E_k = \mathbb{E}(C_k(G))$. Partition Z into sets $Z_j = \{G' : n^{jh}E_k < C_k(G') \leq n^{(j+1)h}E_k\}$. Then $Z = \cup_{j=2}^{\infty} Z_j$. From Lemma 2, we have that $\mathrm{Var}(C_k(G)) =$

$O(n^4 \log n)E_k^2$. Using Chebyshev's inequality,

$$\begin{aligned}
\mathbb{P}_{\mathcal{G}_{n,p}}(Z_j) &< \mathbb{P}\left(\left|n^{jh}E_k - C_k(G')\right| \geq \left(n^{jh} - n^{(j+1)h}\right)E_k\right) \\
&\leq \frac{O(n^4 \log n)E_k^2}{E_k^2 n^{2jh}(1-n^h)^2} \\
&= \frac{O(n^{4-2jh}\log n)}{(1-n^h)^2} \\
&\in O\left(\frac{n^{4-2jh+\epsilon}}{(1-n^h)^2}\right) \\
&\in O(n^{4-2jh+\epsilon}).
\end{aligned}$$

By Lemma 1,

$$\mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(Z_j) = \mathbb{P}_{\mathcal{G}_{n,p}}(Z_j)\frac{C_k(Z_j)}{E_k} \leq n^{(j+1)h}\mathbb{P}_{\mathcal{G}_{n,p}}(Z_j) < O(n^{4-j(h-1)+\epsilon}).$$

Hence,

$$\begin{aligned}
\mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(Z) &\leq \sum_{j=2}^{\infty} \mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(Z_j) \\
&= O(n^{4+h+\epsilon})\sum_{j=2}^{\infty} O(n^{-jh}) \\
&= O(n^{4-h+\epsilon})\sum_{j=0}^{\infty} O(n^{-jh}) \\
&= O(n^{4-h+\epsilon})O(1) = O(n^{4-h+\epsilon}).
\end{aligned}$$

■

We now choose the h defining the set Z in the previous lemma to be sufficiently large so that Z, the set of "bad" graphs, is sufficiently small.

Theorem 4. *If there exists an algorithm A that finds a clique of size $k = (1+\epsilon)\log_b n$ in $\mathcal{G}_{n,p}\langle k\rangle$ with probability $1/q(n)$, for a polynomial $q(n)$, then A can find a clique of the same size in $\mathcal{G}_{n,p}$ with probability $1/q'(n)$ for some polynomial $q'(n)$.*

Proof. Suppose $q(n) = O(n^j)$. Let $Z = \{G \in \mathcal{G}_{n,p}\langle k\rangle : C_k(G) > n^{2(j+5)}E_k\}$. By Lemma 3, $\mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(Z) = O(n^{4+\epsilon-j-5}) = O(n^{-j-1+\epsilon})$, for some constant $\epsilon > 0$. Let Q be the set of all graphs in $\mathcal{G}_{n,p}\langle k\rangle \setminus Z$ for which A finds a clique of size k. Then

$$\mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(Q) = \Omega(n^{-j}) - \mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(Z) = \Omega(n^{-j}) - O(n^{-j-1}) = \Omega(n^{-j}).$$

By Lemma 1,

$$\mathbb{P}_{\mathcal{G}_{n,p}}(Q) = \mathbb{P}_{\mathcal{G}_{n,p}\langle k\rangle}(Q)\frac{E_k}{C_k(Q)}.$$

But $C_k(Q) \leq n^{2(j+5)}E_k$, so

$$\frac{E_k}{C_k(Q)} \geq \frac{1}{n^{2(j+5)}}$$

and hence,

$$\mathbb{P}_{\mathcal{G}_{n,p}}(Q) \geq \Omega(n^{-j})n^{-2(j+5)} = \Omega(n^{-3j-10}).$$

■

Theorem 4 demonstrates that finding a clique of an appropriate size in a graph from $\mathcal{G}_{n,p}\langle k\rangle$ is no easier than finding a clique of the same size in a graph from $\mathcal{G}_{n,p}$. Although we do not know whether it is indeed hard to find a clique of the appropriate size in graphs from $\mathcal{G}_{n,p}$, there has been no positive progress made in the 30 years since the problem was first stated. In Chapter 5, we explore the use of finding hidden cliques as the basis for cryptosystems and discuss the feasibility of such systems.

Chapter 4

Other graph structures

As we saw in Chapter 3, under a certain conjecture, a clique can be hidden in a random graph so that it is hard to find. In this chapter, we discuss the difficulty of finding other graph theoretic structures hidden in random graphs.

4.1 Hamiltonian cycles

The decision problem of determining whether a graph has a Hamiltonian cycle is NP-complete, and is referred to as the HAMCYCLE problem. A corresponding problem is that of finding a Hamiltonian cycle in a graph.

A Hamiltonian cycle can be hidden in a random graph as follows. Choose a random graph $G \in_R \mathcal{G}_{n,p}$, and insert a Hamiltonian cycle H into G by selecting a random permutation of the vertices and adding edges between subsequent vertices in the permutation as necessary. We say that H is the *embedded Hamiltonian cycle*. Let $\mathcal{H}_{n,p}$ represent the probability distribution of graphs formed in this way. In the literature, it is common to consider the case when $p = d/n$ and we let $\mathcal{H}_{n,d/n}$ represent the corresponding probability distribution.

The expected degree of a vertex v in a graph G from $\mathcal{H}_{n,d/n}$ is:

$$\mathbb{E}(\deg(v)) = (n-1)\frac{d}{n} + 2 \cdot \left(1 - \frac{d}{n}\right) = (n-3)\frac{d}{n} + 2$$

and the expected size of the edge set is

$$\mathbb{E}(|E(G)|) = \frac{n}{2}\mathbb{E}(\deg(v)) = \frac{(n-3)d}{2} + n.$$

When compared to the expected size of the edge set of a graph from $\mathcal{G}_{n,p}$ as demonstrated by equation 2.2, the number of edges added to complete the Hamiltonian cycle is only $n(1-p)$.

The threshold function of a Hamiltonian cycle in a random graph is demonstrated in the following theorem, which is a combination of theorems VIII.2.9 and VIII.2.11 from [4]:

Theorem 5. *Let $p = (1/n)(\log n + \log\log n + \omega(n))$. If $\omega(n) \to +\infty$, then almost every graph from $\mathcal{G}_{n,p}$ is Hamiltonian, and if $\omega(n) \to -\infty$, then almost every graph from $\mathcal{G}_{n,p}$ is non-Hamiltonian.*

In the $\mathcal{G}_{n,p}$ model where $p = d/n$ and d is a constant, it must be that $\omega(n) \to -\infty$ as $n \to \infty$, and so almost every graph from $\mathcal{G}_{n,p}$ is non-Hamiltonian.

There are many algorithms that find, with high probability, Hamiltonian cycles in dense random graphs or regular sparse random graphs. For example, Bollobás et al. [5] give an algorithm HAM with expected running time $O(n^{4+\epsilon})$, $\epsilon > 0$ such that, for $M(n) = (n/2)(\log n + \log\log n + c_n)$,

$$\lim_{n\to\infty} \mathbb{P}(HAM \text{ finds a Hamiltonian cycle in } \mathcal{G}_{n,M}) = \begin{cases} 0, & \text{if } c_n \to -\infty, \\ e^{-e^{-c}}, & \text{if } c_n \to c, \\ 1, & \text{if } c_n \to \infty. \end{cases}$$

As a result, if $d \geq (\log n + \log\log n)/2$, then HAM will find a Hamiltonian cycle in a graph from $\mathcal{G}_{n,p}$, $p = d/n$, with high probability. As well, Frieze [20] describes an

$O(n^3 \log n)$ algorithm which has high probability of finding a Hamiltonian cycle in a regular random graph.

The algorithm of Bollobás et al. [5] above finds a Hamiltonian cycle in a dense random graph. While all graphs from $\mathcal{H}_{n,d/n}$ are Hamiltonian, HAM will find with high probability some Hamiltonian cycle in a graph from $\mathcal{H}_{n,d/n}$, $d \geq (\log n + \log\log n)/2$, it is not guaranteed that the cycle found will be the embedded Hamiltonian cycle. In fact, it is likely that the cycle found will consist entirely of random edges.

The above result only applies for dense random graphs. However, Broder et al. [10] give an $O(dn^3)$ algorithm which finds a Hamiltonian cycle in a graph from $\mathcal{H}_{n,d/n}$ for any constant $d > 0$. Thus, it is not possible to build a cryptosystem on the hardness of finding a hidden Hamiltonian cycle in a graph from $\mathcal{H}_{n,d/n}$. A system in which a message is hidden as a Hamiltonian cycle or in which authentication is proved using knowledge of a hidden Hamiltonian cycle is not viable since any party can use a polynomial time algorithm to find a Hamiltonian cycle and thus impersonate the party in question.

4.2 Colourings

The decision problem of determining whether a graph can be coloured with k colours, $k \geq 3$, is NP-hard and is denoted k-COLOUR. (Determining whether a graph is 2-colourable is easy.) There is a corresponding problem of finding a k-colouring for a given graph, should one exist.

There are many possible models for constructing k-colourable random graphs, and these are listed and their relationships are explored in Dyer and Frieze [14]. One standard model which is easy to analyze and is frequently found in the literature is

as follows. Let $V = \{1, \ldots, n\}$. Each vertex is randomly assigned to one of k colour classes with probability $1/k$. For each pair (u, v) of vertices belonging to different colour classes, the edge uv is inserted with probability p, while no edges are inserted between vertices in the same colour class. Let $\mathcal{C}_{n,p,k}$ denote the probability distribution of graphs formed in this way.

The Erdős-Rényi random graph model $\mathcal{G}_{n,\frac{1}{2}}$ is the uniform distribution of graphs. However, it is not the case that $\mathcal{C}_{n,\frac{1}{2},k}$ is the uniform distribution of k-colourable graphs. This is because such a graph can normally be coloured with k colours in many distinct ways.

While `k-COLOUR` is NP-hard in the worst case, there are many classes of k-colourable random graphs for which it is possible to give a k-colouring in polynomial time.

Kučera [29] and Turner [43] give polynomial time graph colouring algorithms that k-colour a graph from $\mathcal{C}_{n,p,k}$ with high probability. However, for the graphs for which the given algorithms fail, the use of an algorithm with exponential running time is required. The probabilities of failure for their respective algorithms are high enough that the expected run time over $\mathcal{C}_{n,p,k}$ is not polynomial.

Dyer and Frieze [14] provide a polynomial time algorithm that colours a random k-colourable graph with k colours with high probability, and, when amortized over all graphs in $\mathcal{C}_{n,p,k}$ for fixed n, k, and p, has expected run time $O(n^2)$. They also prove the following theorem:

Theorem 6 (Dyer and Frieze, 1989). *Fix k and p, and let $G \in_R \mathcal{C}_{n,p,k}$. Then with high probability G has a unique k-colouring, and moreover the expected number of different k-colourings of G is $1 + o(1)$.*

Dyer and Frieze also provide similar results for other models of k-colourable random

graphs.

There is a simple algorithm for k-colouring graphs from $\mathcal{C}_{n,p,k}$ for $p \geq n^{-1/2+\epsilon}$, $\epsilon > 0$, which is described in Blum and Spencer [3]. The idea is as follows.

Theorem 7 (Blum and Spencer, 1995). *Let $p \geq n^{-1/2+\epsilon}$, $\epsilon > 0$. There exists a polynomial time algorithm that with high probability gives a k-colouring for a graph chosen randomly from $\mathcal{C}_{n,p,k}$.*

Proof. Two vertices from the same colour class are expected to have more common neighbours than two vertices from different colour classes. Suppose u and v are two distinct vertices of the same colour. Each of the other $n-2$ vertices has probability $1-1/k$ of being in a different colour class from u and v, and thus has probability $(1-1/k)p^2$ of being a neighbour to both; hence u and v have on average $(n-2)(1-1/k)p^2$ neighbours in common. If u and v are from different colour classes, then on average they have $(n-2)(1-2/k)p^2$ common neighbours.

For $p \geq n^{-1/2+\epsilon}$, these probabilities are within a $(1+o(1))$ factor of $n^{2\epsilon}(1-1/k)$ and $n^{2\epsilon}(1-2/k)$ respectively. Let X_w be the indicator random variable for the event that w is adjacent to both u and v; the X_w random variables are independent for different values of w. Thus we can apply Chernoff bounds to $X = \sum_{w \in V \setminus \{u,v\}} X_w$: for any $\delta > 0$,

$$\mathbb{P}(X < (1-\delta)\mathbb{E}(X) \text{ or } X > (1+\delta)\mathbb{E}(X)) < 2e^{-\delta^2 \mathbb{E}(X)/3}.$$

But $\mathbb{E}(X) = \Theta(n^{2\epsilon})$, so the probability on the right-hand side is so small that

$$\sum_{u,v \in V, u \not\sim v} \mathbb{P}(X < (1-\delta)\mathbb{E}(X) \text{ or } X > (1+\delta)\mathbb{E}(X)) = o(1)$$

where each term in the sum has an X corresponding to different starting vertices u and v.

Thus, with high probability all pairs of vertices in the same colour class have $n^{2\epsilon}(1 - 1/k)(1 + o(1))$ common neighbours, and all pairs of vertices in different colour classes have $n^{2\epsilon}(1 - 2/k)(1 + o(1))$ common neighbours. There exists a polynomial time algorithm that can separate these two cases, and thus the colour classes can be determined. ■

Blum and Spencer [3] provide an extension of this basic technique that finds a k-colouring for a graph from $\mathcal{C}_{n,p,k}$, $p \geq n^{-1+\epsilon}$, $\epsilon > 0$, with high probability, and this new algorithm runs in time $O(kn^2)$.

Hence the only possibility for basing a cryptosystem on colourings in random graphs would seem to be for the case in which $p \leq n^{-1}$. There are no theoretical results at present either demonstrating or prohibiting the existence of a polynomial-time algorithm for this case. However, Petford and Welsh [41] report experimental results of a randomized 3-colouring algorithm on tripartite random graphs. The algorithm performs well on almost all edge probabilities, including very small probabilities $p \leq n^{-1}$, but has high running time on a range of edge probabilities which give average vertex degree of approximately 5 or 6. Petford and Welsh [41] offer no theoretical interpretation of this degenerate case.

Because there exist polynomial-time algorithms that find with high probability k-colourings for random graphs chosen from $\mathcal{C}_{n,p,k}$, for $p \geq n^{-1+\epsilon}$, $k \geq 3$, it is generally infeasible to build a cryptosystem based on knowledge of a k-colouring in a random graph. Although there is a small range of edge probabilities for which common algorithms for k-colouring have been demonstrated experimentally to yield poor results, there are no theoretical results confirming the hardness of k-colouring in this range and are thus unsuitable for cryptographic purposes. For example, if one party $\mathcal{A}$ was to authenticate itself to another party $\mathcal{B}$ by demonstrating a k-colouring of a graph

$G_{\mathcal{A}}$ which had been chosen from $\mathcal{C}_{n,p,k}$, another party $\mathcal{E}$ could with high probability impersonate $\mathcal{A}$ by finding a k-colouring using one of the algorithms described above.

Chapter 5

Protocols

In Chapters 3 and 4 we explored the computational difficulty of finding hidden structures in quasi-random graphs. While it is easy to find hidden Hamiltonian cycles and to provide k-colourings for k-colourable random graphs, it is conjectured to be hard to find a hidden clique of an appropriate size. In this chapter, we explore various protocols built around hidden cliques in quasi-random graphs and in section 5.3 discuss the practicality of graph-based protocols.

5.1 Zero-knowledge authentication

Knowledge of a hidden structure in a graph can be used as a proof of identity in an authentication protocol. One party, $\mathcal{A}$, can prove her identity to another party, $\mathcal{B}$, by demonstrating knowledge of a property of a particular graph known to correspond to $\mathcal{A}$. If it is hard to forge knowledge of that property, then $\mathcal{B}$ can be assured that no one other than $\mathcal{A}$ can pass the test.

Consider the following basic protocol for authentication using hidden cliques. Suppose $\mathcal{A}$ creates a graph $G_{\mathcal{A}} \in_R \mathcal{G}_{n,p}\langle k\rangle$ with hidden clique $K_{\mathcal{A}}$ of size $k = 3/2 \log_b n$. If $\mathcal{B}$ has an authentic copy of $G_{\mathcal{A}}$, then $\mathcal{A}$ can prove her identity to $\mathcal{B}$ simply by providing

him with $K_{\mathcal{A}}$. Assuming finding a clique of size k in a graph from $\mathcal{G}_{n,p}\langle k\rangle$ is hard, based on the results and conjectures of Chapter 3, only $\mathcal{A}$ can demonstrate a clique of size k in $G_{\mathcal{A}}$.

This basic protocol has a major drawback. At the end of the protocol, $\mathcal{B}$ has complete knowledge of $K_{\mathcal{A}}$ and can now impersonate $\mathcal{A}$ in any test using the same graph $G_{\mathcal{A}}$. If the only time $\mathcal{A}$'s graph $G_{\mathcal{A}}$ is used is in an interaction with $\mathcal{B}$, for example to login to $\mathcal{B}$'s server, then it is acceptable for $\mathcal{B}$ to gain knowledge of $K_{\mathcal{A}}$. If, however, $\mathcal{A}$'s graph $G_{\mathcal{A}}$ is widely distributed and used by a number of independent parties to authenticate $\mathcal{A}$, then it is unacceptable for $\mathcal{B}$ to gain any knowledge of $K_{\mathcal{A}}$. Thus, we desire a system in which $\mathcal{A}$ can prove that she knows a clique of size k in $G_{\mathcal{A}}$ without revealing the actual clique $K_{\mathcal{A}}$. We now formalize this notion.

A zero-knowledge proof system, as described in section 2.5, is a solution to this problem. Juels and Peinado [26] provide the following protocol in which $\mathcal{A}$ uses a zero-knowledge proof to demonstrate knowledge of an appropriately-sized clique in her public graph $G_{\mathcal{A}}$.

Algorithm 1 Zero-knowledge authentication using hidden cliques (naïve clique creation)

1: $\mathcal{B}$ retrieves an authentic copy of $\mathcal{A}$'s key $G_{\mathcal{A}} \in_R \mathcal{G}_{n,p}\langle k\rangle$, which has a hidden clique $K_{\mathcal{A}}$ of size $k = 3/2 \log_{1/p} n$.
2: $\mathcal{A}$ chooses a random permutation π and sends commitments of the edges of $G' = \pi(G_{\mathcal{A}})$ to $\mathcal{B}$.
3: $\mathcal{B}$ flips a coin c.
4: **if** c is heads **then**
5: $\quad\mathcal{A}$ sends $\mathcal{B}$ decommitments of all of the edges of $\pi(G_{\mathcal{A}})$ along with the permutation π.
6: $\quad\mathcal{B}$ accepts if the decommitment of G' matches the authentic copy of $G_{\mathcal{A}}$. Otherwise, $\mathcal{B}$ rejects.
7: **else if** c is tails **then**
8: $\quad\mathcal{A}$ sends $\mathcal{B}$ decommitments of the edges $\pi(K_{\mathcal{A}})$.
9: $\quad\mathcal{B}$ accepts if the decommitments of $\pi(K_{\mathcal{A}})$ form a clique of size k. Otherwise, $\mathcal{B}$ rejects.
10: **end if**

The algorithm is repeated suitably many times with different random permutations and coin flips so that $\mathcal{B}$ can be convinced that $\mathcal{A}$ has knowledge of a clique of size k in $G_{\mathcal{A}}$.

In the algorithm above, $\mathcal{A}$ uses naïve clique creation for hiding the clique K, but the protocol could be modified to use split naïve clique creation.

Assuming it is hard to find cliques of size k in random graphs from $\mathcal{G}_{n,p}$, Theorem 4 implies that it is also hard to find a clique of size k in a graph from $\mathcal{G}_{n,p}\langle k\rangle$. Hence, no one except $\mathcal{A}$ will have knowledge of a clique of size k in $G_{\mathcal{A}}$. Thus the protocol is a computational zero-knowledge proof of knowledge under the conjecture. It should be noted that this protocol relies on $\mathcal{A}$ sending commitments of the edges of a graph to $\mathcal{B}$.

5.2 Kučera's generalized encryption scheme

Kučera [30] describes an encryption scheme based on hidden cliques in random graphs.[1] More precisely, Kučera describes a *generalized encryption scheme* which is a probabilistic encryption system in which some object may be an encryption of both 0 and 1 but the probability of an ambiguous encryption is low.

The idea of Kučera's *graph generalized encryption scheme* is as follows. Let b be the bit to be encoded. Let $0 < \kappa < \frac{1}{2}$ be a fixed parameter and let $k = \lfloor n^{\kappa} \rfloor$. A random graph is constructed with an embedded clique of size $k+b$. Because it is possible that a graph corresponding to an encryption of 0 could contain a clique of size $k+1$, this is a generalized encryption scheme. However, Kučera demonstrates that with high probability the scheme is unambiguous.

[1] Kučera's system is phrased in terms of independent sets, but for consistency with the rest of this work we cite it in terms of cliques.

Kučera also demonstrates that, with high probability, two common heuristic techniques will not defeat the graph generalized encryption scheme. In the basic techniques examined by Kučera, a clique is built up, vertex by vertex, as follows. In the two heuristic techniques in question, the next vertex to be added is chosen either completely at random or chosen at random from among the vertices of highest degree; in both cases, only vertices which are adjacent to all vertices already in the constructed clique are considered for addition.

Because the security of Kučera's graph generalized encryption scheme rests on the difficulty of finding large hidden cliques in quasi-random graphs and, as described in section 3.3.2, there are no results on the hardness of this problem, there are no results on the infeasibility of breaking Kučera's scheme.

Since this scheme uses cliques of size $k = o(\sqrt{n})$, the results of section 3.4 do not apply. However, it should be noted that using very small hidden cliques of size $k \in [(1+\epsilon)\log n, (2-\epsilon)\log n]$, $\epsilon > 0$, is not feasible for an encryption scheme because there are many cliques with sizes in that range; it is likely that a graph that is an encryption of 0 will be an ambiguous encryption and contain a clique of size $k+1$.

5.3 Security analysis

As cited in section 3.3.2, Kučera [30] demonstrates that certain heuristic algorithms will asymptotically fail with high probability to find a clique of size $k = o(\sqrt{n})$ in a graph from $\mathcal{G}_{n,p}\langle k\rangle$. This is just an asymptotic result, however. Brockington and Culberson [9] report the results of some experiments to determine the effectiveness of some algorithms for finding cliques embedded in random graphs with $n = 1000$ vertices.

Brockington and Culberson [9] use three different clique hiding models. We have described the first two, naïve clique creation and split naïve clique creation above. When using the latter, Brockington and Culberson take advantage of the degrees of freedom provided by tuning both parameters p and r in a $\mathcal{G}_{n,(p,r)}\langle k\rangle$ so as to minimize the amount of information leaked by the difference in degrees between clique and non-clique vertices at different stages of their algorithms. Even when maximizing the use of tuning parameters to hide degree information, Brockington and Culberson's two basic algorithms are able to find hidden cliques of size as small as $k = 16$ at least two-thirds of the time in graphs chosen from $\mathcal{G}_{n,(p,r)}\langle k\rangle$ with $n = 1000$ and average vertex degree $1/2$.

The best asymptotic algorithm for finding cliques is the naïve method, as described in section 3.3.1, in which all subsets of vertices of size k are chosen; there are $O(\binom{n}{k})$ such subsets. The following table summarizes values of n and the corresponding value of $\binom{n}{k}$ for the parameters $p = 1/2$, $k = 3/2 \log_2 n$:

n	k	$\binom{n}{k}$ (approx.)	$\lvert E\rvert$
500	14	2^{84}	1.2×10^5
1000	15	2^{103}	5.0×10^5
2000	17	2^{131}	2.0×10^6
4000	18	2^{156}	8.0×10^6
30000	23	2^{256}	4.5×10^8

Figure 5.1: Worst-case running times for finding hidden cliques.

As seen in the table above, if $p = 1/2$ and $n = 1000$, the number of subsets to be considered is approximately 2^{103}. However, this is a very loose upper bound on the amount of computation required to find a clique of size 15 in a graph from $\mathcal{G}_{1000,1/2}\langle 15\rangle$, and in particular it is not the case in practice that finding cliques in graphs from the family requires on the order of 2^{103} operations. According to Lenstra and Verheul [32], searching through a space of 2^{103} possible solutions requires approximately 10^{17} years on a Pentium II 450MHz computer, whereas Brockington and Culberson [9]

demonstrate that such cliques can be found with non-trivial probability very quickly. Moreover, since the hidden clique is of the same size as the naturally occurring background cliques, there are often many cliques of the desired size. The experiments of Brockington and Culberson [9] demonstrate that in graphs from $\mathcal{G}_{1500,1/2}\langle k\rangle$, 11% of the time, a particular algorithm finds a clique other than the explicitly hidden clique.

Although we can not give more precise estimates on the size of a graph needed to render the problem of searching for hidden cliques infeasible, we can reasonably suggest that an infeasible search would require a graph to have several thousand vertices. With even the small number of 4000 vertices, the size of the adjacency matrix of the graph is 1MB. Since the adjacency matrix is effectively a random string with each bit present with probability 1/2, we cannot expect to compress the encoding of the matrix to any significant degree. Thus, any protocol basing its security on graphs with hidden cliques would require that at least 1MB of data be exchanged each time a graph is exchanged. If, as in the zero knowledge protocol of algorithm 5.1, an exchange needs to happen t times to reduce the probability of a party successfully cheating to 2^{-t}, then at least t MB of data need to be exchanged. If t is to be chosen to match the current level of security in most of today's secured Internet transactions, which use RSA public key cryptography [42] with 1024-bit keys, t would be chosen to be approximately 77 [32]. When compared with the 1024 bits needed to represent a comparable RSA key or the 163 bits needed for a comparable elliptic curve cryptography key, exchanging 77MB of graph data is impractical.

Similarly, the graph generalized encryption scheme of Kučera [30] is impractical. Recall from section 5.2 that Kučera's scheme involves hiding a clique of size $k+b$ in a random graph to encrypt a bit b. Using the same analysis as above, a graph of size at least 4000 vertices, encoded by 1MB of data, is needed to encrypt a single bit. Most practical cipher algorithms, including for example the Data Encryption Standard

(DES), produce output that is the same size as the input (cf. Menezes et al. [36, Chapter 7]). Thus, the graph generalized encryption scheme, which generates output that is 8 million times larger than the input, is quite impractical.

Chapter 6

Conclusions

We have explored the difficulty of finding hidden structures in random graphs, in particular, of finding hidden cliques, hidden Hamiltonian cycles, and valid colourings, and examined their suitability as the basis for a cryptosystem. As polynomial time algorithms are known for solving the latter two problems, they are not suitable for use as hard problems in cryptography.

The problem of finding a hidden clique in a random graph has some potential in terms of the difficulty of the problem. We have shown that it is no easier to find a embedded clique of an appropriate size hidden in a random graph than it is to find a clique in the same size in a completely random graph, where edges in the random graph are added with any constant non-trivial probability. Whether this latter problem can be solved in polynomial time has been an open problem since 1976.

While it may be difficult to find hidden cliques of an appropriate size in random graphs, it does not seem as this technique will be practical for real-world cryptography. The amount of data required to encode a graph of size sufficiently large to prevent brute-force attacks is prohibitive, especially when compared to the amount of data exchanged in existing protocols.

Nevertheless, we believe it is interesting to explore the relative difficulty of graph-based problems, as many fundamental problems in computational complexity theory are phrased in terms of graph structures. Whereas current cryptosystems base their security on the conjectured difficulty of number theoretic problems, exploring the difficulty of graph problems may eventually allow cryptosystems to be based on problems known to have infeasible computational complexity.

6.1 Open questions

1. The original open problem to give a polynomial time algorithm that finds a clique of size $(1 + \epsilon) \log n$ in a graph from $\mathcal{G}_{n,p}$ still stands. The results of Chapter 3 extend this open problem to graphs from $\mathcal{G}_{n,p}\langle k \rangle$.

2. Can Theorem 4 be extended to larger cliques or to split naïve clique creation?

3. Petford and Welsh [41] describe a small range of edge probabilities for which experimental evidence suggests that k-colouring random k-colourable graphs is difficult. Can a theoretical explanation be given for this observation, and can the observed difficulty of colouring such graphs be used with any reliability in a cryptosystem?

4. Blum and Spencer [3] give a result indicating that k-colouring a certain class of semirandom k-colourable graphs is difficult. However, the semirandom k-colourable graphs in question can be generated by a party with unbounded computational resources. Is there an efficient method for generating semirandom k-colourable graphs that are still hard to k-colour?

5. Although finding hidden cliques may be difficult, other natural problems concerning hidden graph structures, such as finding hidden Hamiltonian cycles and finding k-colourings of k-colourable random graphs, are easy. Additionally,

approximation algorithms for finding the size large cliques in a graph do not provide nearly as good approximations as those for these other problems. Is there something inherently more difficult about problems concerning cliques?

6. Are there any other graph structures which could be used as the hard problem as the basis of a cryptosystem? Can any graph-based cryptosystem use data structures of practical size?

Bibliography

[1] N. Alon and N. Kahale. Approximating the independence number via the θ-function. *Math. Programming*, 80:253–264, 1998.

[2] N. Alon, M. Krivelevich, and B. Sudakov. Finding a large hidden clique in a random graph. In *Proc. 9th Ann. ACM-SIAM Symp. on Discrete Algorithms*, pages 594–598, 1998.

[3] A. Blum and J. Spencer. Coloring random and semi-random k-colorable graphs. *J. Algorithms*, 19:204–234, 1995.

[4] B. Bollobás. *Random Graphs*. Academic Press, 1985.

[5] B. Bollobás, T. I. Fenner, and A. M. Frieze. An algorithm for finding Hamilton paths and cycles in random graphs. *Combinatorica*, 7(4):327–341, 1987.

[6] R. Boppana and M. M. Halldórsson. Approximating maximum independent sets by excluding subgraphs. *BIT*, 32:180–196, June 1992.

[7] G. Brassard, C. Crépeau, R. Jozsa, and D. Langlois. A quantum bit commitment scheme provably unbreakable by both parties. In *Proc. 34th Ann. IEEE Symp. Foundations of Comp. Sci.*, pages 42–52. IEEE, 1993.

[8] G. Brassard, C. Crépeau, D. Mayers, and L. Salvail. Defeating classical bit commitment schemes with a quantum computer, June 1998. URL `http://xxx.lanl.gov/abs/quant-ph/9806031`.

[9] M. Brockington and J. C. Culberson. Camouflaging independent sets in quasi-random graphs. In D. S. Johnson and M. Trick, editors, *Cliques, Coloring, and Satisfiability: Seconds DIMACS Implementation Challenge*, volume 26 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. American Mathematical Society, 1996.

[10] A. Z. Broder, A. M. Frieze, and E. Shamir. Finding hidden Hamiltonian cycles (extended abstract). In *Proc. 23rd Ann. ACM Symp. on Theory of Comp.*, pages 182–189, 1991.

[11] C. Crépeau. Efficient cryptographic protocols based on noisy channels. In *Advances in Cryptology – Proc. EUROCRYPT '97*, pages 306–317. Springer-Verlag, 1997.

[12] W. Diffie. Private conversation, August 2003.

[13] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6):644–654, November 1976.

[14] M. E. Dyer and A. M. Frieze. The solution of some random NP-hard problems in polynomial expected time. *J. Algorithms*, 10(4):451–489, December 1989.

[15] P. Erdős and A. Rényi. On random graphs I. *Publ. Math. Debrecen*, 6:290–297, 1959.

[16] P. Erdős and A. Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungarian Academy of Science*, 5:17–61, 1960.

[17] P. Erdős and A. Rényi. On the evolution of random graphs. *Bull. Inst. Int. Statist. Tokyo*, 38:343–347, 1961.

[18] P. Erdős and A. Rényi. On the strength of connectedness of a random graph. *Acta Math. Acad. Sci. Hungar.*, 12:261–267, 1961.

[19] U. Feige and R. Krauthgamer. Finding and certifying a large hidden clique in a semi-random graph. *Random Structures and Algorithms*, 16(2):195–208, 2000.

[20] A. M. Frieze. Finding Hamiltonian cycles in sparse random graphs. *J. Combinatorial Theory B*, 44:230–250, April 1988.

[21] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *J. ACM*, 38 (1):691–729, July 1991.

[22] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In *Proc. 17th Ann. ACM Symp. on Theory of Comp.*, pages 291–304. ACM Press, 1985.

[23] J. Håstad. Testing of the long code and hardness of clique. In *Proc. 28th Ann. ACM Symp. on the Theory of Computing*, pages 11–19. ACM Press, 1996.

[24] L. A. Hemaspaandra and M. Ogihara. *The Complexity Theory Companion.* Springer, 2002.

[25] M. Jerrum. Large cliques elude the metropolis process. *Random Structures and Algorithms*, 3(4):347–360, 1992.

[26] A. Juels and M. Peinado. Hiding cliques for cryptographic security. In *Proc. 9th Ann. ACM-SIAM Symp. on Discrete Algorithms*, pages 678–684, 1998.

[27] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.

[28] R. M. Karp. The probabilistic analysis of some combinatorial search problems. In J. F. Traub, editor, *Algorithms and Complexity: New directions and recent results*, pages 1–19. Academic Press, 1976.

[29] L. Kučera. Expected behaviour of graph colouring algorithms. In *Proc. 1977 Int. Fundamentals of Computation Theory Conf.*, volume 56 of *LNCS*, pages 447–451, 1977.

[30] L. Kučera. A generalized encryption scheme based on random graphs. In *Graph-Theoretic Concepts in Computer Science, WG '91*, number 570 in LNCS, pages 180–186. Springer-Verlag, 1991.

[31] L. Kučera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57:193–212, 1995.

[32] A. K. Lenstra and E. R. Verheul. Selecting cryptographic key sizes. In *Proc. 3rd Int. Workshop on Practice and Theory in Public Key Cryptography*, LNCS, pages 446–465. Springer-Verlag, 2000.

[33] L. Levin. Average case complete problems. *SIAM J. Computing*, 15(1):285–286, February 1986.

[34] H. Lipmaa. (Bit)-Commitment schemes, July 2004. URL `http://www.tcs.hut.fi/{~~}helger/crypto/link/commitment/`.

[35] D. Mayers. Unconditionally secure quantum bit commitment is impossible. *Phys. Rev. Lett.*, 78(17):3414–3417, April 1997.

[36] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 5th edition, 2001.

[37] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[38] M. Naor. Bit commitment using pseudorandomness. *J. Cryptology*, 4:151–158, 1991.

[39] National Institute of Standards and Technology. Data Encryption Standard (DES), October 1999.

[40] National Institute of Standards and Technology. Specification for the Advanced Encryption Standard (AES), November 2001.

[41] A. D. Petford and D. J. A. Welsh. A randomised 3-colouring algorithm. *Discrete Mathematics*, 74(1–2):253–261, 1989.

[42] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, February 1978.

[43] J. S. Turner. Almost all k-colorable graphs are easy to color. *J. Algorithms*, 917: 63–82, 1988.

Index