# Eidolon: A Post-Quantum Signature Scheme Based on k-Colorability in the Age of Graph Neural Networks

Asmaa Cherkaoui[1*][0009−0007−7068−5059], Ramón Flores[2][0000−0002−5537−3299], Delaram Kahrobaei[3,4,5,6,7][0000−0001−5467−7832], and Richard C. Wilson[7][0000−0001−7265−3033]

[1] Laboratory of Mathematical Analysis, Algebra and Applications (LAM2A), Faculty of Sciences Ain Chock (FSAC), University Hassan II, Casablanca, Morocco
esma1maysan@gmail.com *Corresponding author.
[2] Department of Geometry and Topology, Faculty of Mathematics, University of Seville, Seville, Spain
ramonjflores@us.es
[3] Departments of Computer Science and Mathematics, Queens College, City University of New York, USA
delaram.kahrobaei@qc.cuny.edu
[4] PhD Program in Mathematics, and Initiative for the Theoretical Sciences, Graduate Center, City University of New York, USA
[5] Department of Computer Science and Engineering, Tandon School of Engineering, New York University, USA
[6] Department of Computer Science, University of York, United Kingdom
richard.wilson@york.ac.uk

**Abstract.** We propose `Eidolon`, a post-quantum signature scheme grounded in the NP-complete $k$-colorability problem. Our construction generalizes the Goldreich Micali Wigderson zero-knowledge protocol to arbitrary $k \geq 3$, applies the Fiat–Shamir transform, and uses Merkle-tree commitments to compress signatures from $O(tn)$ to $O(t \log n)$. We generate instances by planting a coloring while aiming to preserve the statistical profile of random graphs. We present an empirical security analysis of such a scheme against both classical solvers (ILP, DSatur) and a custom graph neural network (GNN) attacker. Experiments show that for $n \geq 60$, neither approach is able to recover a valid coloring matching the planted solution, suggesting that well-engineered $k$-coloring instances can resist the considered classical and learning-based cryptanalytic approaches. These experiments indicate that the constructed instances resist the attacks considered in our evaluation.

**Keywords:** Post-quantum cryptography · $k$-colorability · NP-hard signatures · zero-knowledge proofs · fiat–shamir transform · graph neural networks.

## 1 Introduction

With the rising threat of quantum computers to traditional cryptography, recent attention has focused on post-quantum cryptographic methods. Based on the current belief that there is no quantum speed-up for NP-complete problems, these problems are potentially a rich source of potential cryptosystems. In particular, graph theory contains several NP-complete problems, including homomorphism, $k$-colorability, and Hamiltonicity. In this paper, we study the application of $k$-colorability to a post-quantum signature system.

When considering an NP-hard problem as the basis for a cryptosystem, care must be taken in the concrete realization of a particular instance. This is because many instances can be solved in acceptable time using heuristic algorithms. Indeed, a number of heuristics are available for graph coloring and can quickly solve the problem for many graphs.

In this regard, machine learning has emerged as a particular threat because of its ability to extract powerful heuristics directly from data. In this paper, we explore the issue of instance hardness for $k$-colorability, from both a theoretical and empirical perspective, to demonstrate the security of the system. In [20], the authors demonstrate how graph neural networks can be used to solve combinatorial optimization problems. Their approach is broadly applicable to canonical NP-hard problems in the form of quadratic unconstrained binary optimization problems, such as maximum cut, minimum vertex cover, maximum independent set, as well as Ising spin glasses and higher-order generalizations thereof in the form of polynomial unconstrained binary optimization problems.

The rapid progress in quantum computing has intensified the search for cryptographic primitives that remain secure in a post-quantum world. While lattice- and code-based schemes currently dominate standardization efforts, combinatorial problems offer an alternative foundation rooted in computational complexity theory. The use of $k$-colorability in cryptography is limited by the gap between worst-case hardness and typical-instance behavior: many graph instances that are hard in the worst-case sense can still be handled efficiently by heuristic algorithms such as DSatur [4], and more recently by data-driven methods leveraging machine learning.

Recent work by Schuetz, Brubaker, and Katzgraber [20] demonstrates that graph neural networks (GNNs) can effectively approximate solutions to canonical NP-hard problems, including graph coloring, by learning implicit heuristics from data. This motivates examining whether such methods affect the security of schemes based on combinatorial hardness. To date, to the best of our knowledge, no signature scheme based on $k$-colorability has been proposed that simultaneously offers a complete construction, incorporates instance generation that embeds a secret coloring while aiming to maintain statistical indistinguishability from a random graph, and undergoes empirical validation against both classical and learning-based attacks.

In this paper, we present `Eidolon`, a post-quantum digital signature scheme derived from the $k$-colorability problem by generalizing the zero-knowledge identification protocol of Goldreich, Micali, and Wigderson (GMW) [8], originally demonstrated for 3-colorability, to arbitrary $k \geq 3$, and then applying the Fiat–Shamir transform. Our design features a planted coloring mechanism, inspired by the "quiet solution" framework of Krzakala and Zdeborová [15], which embeds a secret $k$-coloring into a $k$-partite random graph with calibrated edge density, ensuring witness existence while aiming to maintain statistical indistinguishability from an Erdős–Rényi random graph. To reduce the signature size obtained from a direct Fiat–Shamir transformation of the underlying zero-knowledge protocol, we use Merkle-tree vector commitments, compressing the per-round vertex commitments into a single root and thereby reducing the overall signature size from $O(tn)$ to $O(t \log n)$, where $t$ denotes the number of Fiat–Shamir rounds and $n = |V|$.

We evaluate the scheme empirically using two attack strategies: the classical DSatur heuristic and a custom-designed graph neural network (GNN). Our GNN is inspired by the framework of Schuetz, Brubaker, and Katzgraber [20]. Both attackers are tested on the same class of $k$-partite random graphs. Our experiments show that, for graphs with $n \geq 60$ vertices and our chosen density parameters, neither DSatur nor our GNN is able to recover a coloring matching the planted solution, providing empirical evidence, within the tested regime, that the hardness of our $k$-colorability instances withstands the considered classical and learning-based cryptanalytic approaches.

In summary, our contributions are threefold: (i) we design `Eidolon`, a Fiat–Shamir signature scheme based on $k$-colorability with a statistically hidden planted coloring; (ii) we reduce its asymptotic signature size via Merkle-tree vector commitments; and (iii) we provide an empirical hardness study against both classical heuristics and GNN-based attacks. The rest of the paper is organized as follows: Section 2 reviews graph coloring and instance generation; Section 3 details our signature construction; and Section 5 describes our attack models and experimental setup.

## 2 Graph instance generation

Let $G = (V, E)$ be a graph with vertices $V$ and edges $E$. A $k$-coloring of a graph is an assignment of one of $k$ colors to each vertex so that no two adjacent vertices share the same color. It is known that determining whether a graph has a $k$-coloring is NP-complete for $k \geq 3$. The minimum number of colors needed is the *chromatic number* $\chi(G)$, and computing it is NP-hard.

However, worst-case hardness alone is insufficient for cryptography: many concrete instances are easy to solve. For example, if $k = |V|$, a trivial coloring exists; complete graphs have $\chi(G) = |V|$; and heuristic algorithms like DSatur often find valid colorings quickly on structured or sparse graphs. This creates a constraint for signature schemes based on $k$-colorability: the prover must know a valid $k$-coloring, which serves as the secret key, while the public graph should not reveal that coloring to the verifier or to an adversary.

Thus, we must *construct* instances where:

- a valid $k$-coloring is known by design to the prover,
- the graph appears statistically indistinguishable from a random hard instance,
- and recovering the coloring remains infeasible for classical and ML-based attackers.

This ensures the protocol is both *correct* (the prover can always respond) and *secure* (the secret remains hidden).

An (Erdös-Rényi) random graph is a graph $G_R(n, p)$ on $n$ vertices, where each vertex pair is joined uniformly at random with probability $p$. A number of results are available on the hardness of coloring random graphs. In [7] it is shown that coloring a graph with less than $2\chi(G) - \delta$ is NP-hard. For fixed $p$, almost every random graph has a chromatic number [2]

$$\chi(G_R(n, p)) = \frac{n}{r}(1 + \epsilon) \tag{1}$$

with $0 \leq \epsilon \leq 3 \log\log n / \log n$ and

$$r = 2\log_d n - \log_d \log_d n + 2\log_d(e/2) + 1. \tag{2}$$

These results hold in the limit of large $n$. Furthermore, we observe that very sparse and very dense graphs are easy to color, and so we select $p = 1/2$. In this case, it has been shown that $\chi(G_R(n, p)) \sim \log(1/(1-p))/2\log np$ [17]. However, these results are asymptotic and may not hold for the small graphs under consideration here. We reserve selection of $k$ for the planted coloring until our empirical analysis, noting only that it should be the same size as the expected chromatic number of the equivalent random graph.

### 2.1 Planted $k$-colorable graph construction

In order to operate the digital signature scheme above, it is necessary to be able to construct a graph and $k$-coloring in polynomial time, where the coloring is difficult to discover. We use the natural algorithm for planting a known coloring in a random graph [15]. In particular, we begin by selecting a vertex set $V$, $|V| = n$ (the problem size) and $k \approx \chi[G_R(n, p)]$. We then follow the following steps.

- $V$ is partitioned into $k$ sets such that $n_i = |V_i|, n_i \in \{\lfloor n/k \rfloor, \lceil n/k \rceil\}$ and $\sum_i n_i = n$
- We iterate through all pairs of vertices $(u, v)$
- Let $P_u, P_v$ be the partitions of the vertices. We join the vertices with probability $p$ if $P_u \neq P_v$ and probability zero otherwise.

The resulting graph may be colored with $k$ colors simply by assigning one color to each partition. In [15], the authors demonstrate that this is a *quiet solution* which is hidden in the random graph in the sense that the properties of the graph are not much altered by the existence of the known extra solution. In particular, they conjecture that the hardness of the problem is the same as for the original graph.

In general we work with random graphs such that the probability of an edge between two given vertices is a fixed number $p \in [0,1]$, in such a way that if the graph has $n$ vertices, the expected final number of edges is $p\binom{n}{2}$. We consider here a multipartite graph $\Gamma$ with $n$ vertices and a partition in the set of vertices given by $V = V_1 \cup \ldots V_k$. Given a number $s \in [0,1]$, we compute here the probability $p$ of existence of an edge in $\Gamma$ such that the expected number of edges in the graph is $s\binom{n}{2}$.

Recall that a complete graph in $n_j$ vertices has $\binom{n_j}{2}$ edges. Hence, the number of forbidden edges in $\Gamma$ is $S = \sum_{j=1}^{k} \binom{n_j}{2}$. Then, the maximum number of edges of the graph $\Gamma$ is $\binom{n}{2} - S$, and given a probability $p$ of existence of an edge, the expected number of edges in the graph is $p(\binom{n}{2} - S)$. We have imposed that

$$p(\binom{n}{2} - S) = s\binom{n}{2},$$

and hence

$$p = \frac{s\binom{n}{2}}{\binom{n}{2} - S}.$$

Observe that $s\binom{n}{2}$ is always bounded in the graph $\Gamma$ by $\binom{n}{2} - S$, and hence we always obtain $p \leq 1$. Edges in the graph are selected with probability $p$. We discuss the specific values of $p$ and $k$ in Section 5 on our security analysis.

## 3 Protocol description

We describe the protocol underlying `Eidolon`, based on graph $k$-colorability. Let

$$f : \{1, \ldots, k\} \times \{0,1\}^r \longrightarrow \{0,1\}^s$$

be a statistically hiding and computationally binding commitment function, as in [8]. One round of the protocol is as follows. The protocol is then repeated independently $T$ times.

- **Coloring.** The prover holds a valid $k$-coloring $\phi : V \to \{1, \ldots, k\}$.
- **Permutation.** The prover samples a fresh random permutation $\pi \in S_k$ to mask color labels.
- **Commitment.** For each vertex $i \in V$, the prover samples fresh randomness $r_i$ and commits to the permuted color

  $$c_i \;=\; f\big(\pi(\phi(i)),\, r_i\big).$$

  The prover sends all commitments $\{c_i\}_{i \in V}$ to the verifier.
- **Challenge.** The verifier chooses a uniform random edge $(u, v) \in E$ and requests openings for its endpoints.
- **Response.** The prover reveals $\big(\pi(\phi(u)), r_u\big)$ and $\big(\pi(\phi(v)), r_v\big)$.
- **Verification.** The verifier checks

  $$f\big(\pi(\phi(u)), r_u\big) \stackrel{?}{=} c_u, \qquad f\big(\pi(\phi(v)), r_v\big) \stackrel{?}{=} c_v,$$

  and verifies that

  $$(u, v) \in E, \qquad \pi(\phi(u)), \pi(\phi(v)) \in \{1, \ldots, k\}, \qquad \pi(\phi(u)) \neq \pi(\phi(v)).$$

  The round is accepted if all checks pass.

**Zero-knowledge intuition.** In each round, the prover uses a fresh permutation $\pi$ and fresh commitment randomness. The verifier therefore learns only that the challenged edge joins two differently colored vertices under the permuted coloring.

### 3.1 Soundness

To analyze the soundness of the zero-knowledge proof protocol for $k$-colorability, consider the case where the graph is *not* $k$-colorable. In such a scenario, no matter how the prover attempts to simulate a valid coloring, there will inevitably be a set of *violating edges*–edges whose endpoints receive the same color under any attempted coloring. Let $t$ denote the number of these bad edges, and let $m$ be the total number of edges in the graph. Since the verifier selects one edge uniformly at random in each round, the probability that a cheating prover is caught in a single round is at least $\frac{t}{m}$, while the probability of escaping detection is at most $1 - \frac{t}{m}$.

Although this is the general case, we often assume a worst-case scenario where $t \geq 1$ but unknown. In this case, we conservatively lower-bound the detection probability per round by $\frac{1}{m}$. The protocol is repeated independently for $m^2$ rounds to drive the cheating probability down. Therefore, the probability that a cheating prover escapes detection across all $m^2$ rounds is:

$$\left(1 - \frac{1}{m}\right)^{m^2} = \left[\left(1 - \frac{1}{m}\right)^{m}\right]^{m}.$$

It is well known that:

$$\lim_{m\to\infty} \left(1 - \frac{1}{m}\right)^{m} = \frac{1}{e},$$

so it follows that:

$$\left(1 - \frac{1}{m}\right)^{m^2} \approx \left(\frac{1}{e}\right)^{m} = e^{-m}.$$

This quantity becomes *negligibly small* as $m$ increases, for instance, $e^{-40} < 2^{-55}$. Thus, the verifier's probability of accepting a false claim is exponentially small in the number of edges.

The number of rounds is taken to depend on the number of edges $m$, rather than on the number of vertices $n$ or the number of colors $k$, because the verifier's challenges are edge-based and violations are detected on edges.

### 3.2 Identification and signature construction

We use a GMW-type identification protocol for graph $k$-coloring, following the classical approach of Goldreich, Micali, and Wigderson [8]. The prover commits to a randomly permuted coloring of the public graph and, upon receiving a challenge edge, opens only the two commitments corresponding to its endpoints. The verifier then checks that the revealed colors are valid, distinct, and consistent with the commitments.

| **PROVER** | | **VERIFIER** |
|---|---|---|
| Hold a valid coloring $\phi : V \to \{1, \dots, k\}$ | | |
| Sample a fresh permutation $\pi \in S_k$ and fresh $r_v$ for all $v \in V$ | | |
| Compute $c_v = f\big(\pi(\phi(v)),\, r_v\big)$ for all $v \in V$ | | |
| Set $C = \{c_v\}_{v \in V}$ | | |
| | $\xrightarrow{C}$ | |
| | | Sample a uniform random edge $(u, v) \in E$ |
| | $\xleftarrow{(u,v)}$ | |
| Reveal $(\pi(\phi(u)), r_u)$ and $(\pi(\phi(v)), r_v)$ | | |
| | $\xrightarrow{(\pi(\phi(u)),r_u),\ (\pi(\phi(v)),r_v)}$ | |
| | | Accept if $(u, v) \in E$ $f(\pi(\phi(u)), r_u) = c_u$ and $f(\pi(\phi(v)), r_v) = c_v$ $\pi(\phi(u)), \pi(\phi(v)) \in \{1, \dots, k\}$ and $\pi(\phi(u)) \neq \pi(\phi(v))$ |

Table 1: Graph $k$-Coloring Identification Scheme (one round; repeat $T$ times with fresh $\pi$ and randomness)

### 3.3 Fiat–Shamir signature construction

To obtain a non-interactive signature scheme, we apply the Fiat–Shamir transform [5] to the identification protocol in the random-oracle model. The signer first computes the round commitments using fresh permutations and fresh randomness, then evaluates

$$h = \mathcal{H}(\mathrm{Encode}(G, k, C^{(0)}, \dots, C^{(t-1)}, M))$$

on the public data, the commitments, and the message $M$, where $\mathrm{Encode}(\cdot)$ is a canonical encoding procedure. A public hash-to-edges parser deterministically derives the challenge edges from $h$, and the signer includes the openings corresponding to the endpoints of those edges. Verification recomputes the same hash from $(G, k)$, the commitments, and $M$, derives the same challenge edges, and checks the openings together with color distinctness.

*Challenge derivation.* The challenge edges are derived deterministically from the hash value $h$. More precisely, we sample $t$ edge indices with replacement so as to match the independent public-coin challenges of the identification protocol. To avoid modulo bias, the mapping from hash output to edge indices is implemented using domain-separated rejection sampling.

**Algorithm 1** HashToEdges $(h, t, E)$

$m \leftarrow |E|$; $b \leftarrow \lambda$ ▷ $\lambda$-bit blocks from $\mathcal{H}$
**for** $i = 0$ to $t-1$ **do**
  $j \leftarrow 0$
  **repeat**
    $B \leftarrow \mathcal{H}(\texttt{"EdgeDerive-v1"} \parallel h \parallel \langle i \rangle_{32} \| \langle j \rangle_{32})$
    $x \leftarrow \mathrm{int}(B)$ ▷ interpret $B$ as a big-endian integer in $[0, 2^{\lambda}-1]$
    $M \leftarrow \lfloor 2^{\lambda}/m \rfloor \cdot m$
    $j \leftarrow j + 1$
  **until** $x < M$ ▷ rejection sampling removes modulo bias
  $\mathrm{idx}_i \leftarrow x \bmod m$; $e_i \leftarrow E[\mathrm{idx}_i]$
**end for**
**return** $(e_0, \ldots, e_{t-1})$

*Public inputs and secret key.* Public parameters are $(G = (V, E), k, f, \mathcal{H}, \lambda)$, where $V = \{1, \ldots, n\}$, $E \subseteq \{\{u, v\} \mid 1 \le u < v \le n\}$, $f$ is a statistically hiding, computationally binding commitment, and $\mathcal{H} : \{0,1\}^* \to \{0,1\}^{\lambda}$ is modeled as a random oracle. The *secret key* is a valid $k$-coloring $\phi : V \to \{1, \ldots, k\}$.

*Canonical encoding with domain separation.* We follow standard practice for domain separation [1,6] and canonical serialization [18,14], using a fixed domain-separation tag $\mathsf{TAG} = \texttt{"FS-GkColor-v1"}$ and a canonical serializer Encode$(\cdot)$:

$$\begin{aligned}\mathrm{Encode}(G, k, X_0, \ldots, X_{t-1}, M) = \; & \langle \mathsf{TAG} \rangle \parallel \langle n \rangle_{64} \parallel \langle k \rangle_{32} \parallel \langle m \rangle_{64} \parallel \mathrm{Edges}(E) \\ & \parallel \langle t \rangle_{32} \parallel X_0 \| \cdots \| X_{t-1} \parallel \langle |M| \rangle_{64} \parallel M.\end{aligned}$$

Here $\langle \cdot \rangle_b$ is a $b$-bit big-endian length/value encoding, and $\parallel$ is concatenation. We fix a vertex order $1 < \cdots < n$ and encode edges as

$$\mathrm{Edges}(E) = \langle m \rangle_{64} \parallel \big\|_{\{u,v\} \in E^{\uparrow}} \big( \langle u \rangle_{\lceil \log_2 n \rceil} \parallel \langle v \rangle_{\lceil \log_2 n \rceil} \big),$$

where $E^{\uparrow}$ lists edges with $u < v$ in lexicographic order. This canonicalization ensures all parties hash identical byte strings.

**`Eidolon` signature scheme** `Eidolon` uses as public parameters a graph $G = (V, E)$, an integer $k \in \mathbb{N}$, a statistically hiding and computationally binding commitment

$$f : \{1, \ldots, k\} \times \{0,1\}^r \to \{0,1\}^s,$$

and a hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^{\lambda}$, modeled as a random oracle. The secret key is a valid $k$-coloring $\phi : V \to \{1, \ldots, k\}$. Let $m = |E|$ and set $t = m^2$.

To sign a message $M \in \{0,1\}^*$, the signer samples, for each $i \in \{0, \ldots, t-1\}$, a fresh permutation $\pi_i \in S_k$ and, for every vertex $v \in V$, fresh randomness $r_v^{(i)}$, and computes

$$c_v^{(i)} = f\big(\pi_i(\phi(v)),\, r_v^{(i)}\big).$$

Let $C^{(i)} = (c_v^{(i)})_{v \in V}$. Then compute

$$h = \mathcal{H}\big(\mathrm{Encode}(G, k, C^{(0)}, \ldots, C^{(t-1)}, M)\big),$$

where Encode$(\cdot)$ is a fixed canonical encoding with domain separation. The value $h$ is parsed deterministically into a sequence of challenge edges $(e_0, \ldots, e_{t-1})$, where $e_i = (u_i, v_i) \in E$. For each $i$, the signer includes the openings

$$\big(\pi_i(\phi(u_i)),\, r_{u_i}^{(i)}\big) \qquad \text{and} \qquad \big(\pi_i(\phi(v_i)),\, r_{v_i}^{(i)}\big).$$

The signature is

$$\sigma = \Big( \{C^{(i)}\}_{i=0}^{t-1},\ \{\pi_i(\phi(u_i)), r_{u_i}^{(i)}, \pi_i(\phi(v_i)), r_{v_i}^{(i)}\}_{i=0}^{t-1} \Big).$$

**Algorithm 2** EIDOLON.SIGN$(G, k, \phi, M)$

**Input:** message $M$, private coloring $\phi : V \to \{1, \ldots, k\}$
**for** $i = 0, \ldots, t-1$ **do**
  choose a fresh random permutation $\pi_i \in S_k$
  **for** each $v \in V$ **do**
    choose fresh randomness $r_v^{(i)}$ and compute

$$c_v^{(i)} \leftarrow f(\pi_i(\phi(v)),\, r_v^{(i)}).$$

  **end for**
  set $C^{(i)} \leftarrow (c_v^{(i)})_{v \in V}$
**end for**
Compute

$$h \leftarrow \mathcal{H}\big(\text{Encode}(G, k, C^{(0)}, \ldots, C^{(t-1)}, M)\big).$$

Parse $h$ deterministically to obtain challenges $e_0, \ldots, e_{t-1}$, where each $e_i$ encodes an edge $(u_i, v_i) \in E$
**for** $i = 0, \ldots, t-1$ **do**
  set

$$open_i \leftarrow \big(\pi_i(\phi(u_i)),\, r_{u_i}^{(i)},\, \pi_i(\phi(v_i)),\, r_{v_i}^{(i)}\big)$$

**end for**
Output signature

$$\sigma \leftarrow \Big(\{C^{(i)}\}_{i=0}^{t-1},\ \{open_i\}_{i=0}^{t-1}\Big)$$

**Algorithm 3** EIDOLON.VERIFY$(G, k, M, \sigma)$

**Input:** public parameters $(G, k)$, message $M$, signature $\sigma$
Parse $\sigma$ as $\{C^{(i)}\}_{i=0}^{t-1}$ and openings $\{open_i\}_{i=0}^{t-1}$
Recompute

$$h' \leftarrow \mathcal{H}\big(\text{Encode}(G, k, C^{(0)}, \ldots, C^{(t-1)}, M)\big)$$

Parse $h'$ into $e'_0, \ldots, e'_{t-1}$, where $e'_i = (u'_i, v'_i)$
**for** $i = 0, \ldots, t-1$ **do**
    Let $open_i = (\alpha_u, r_u, \alpha_v, r_v)$
    Check that

$$f(\alpha_u, r_u) \stackrel{?}{=} c^{(i)}_{u'_i} \qquad \text{and} \qquad f(\alpha_v, r_v) \stackrel{?}{=} c^{(i)}_{v'_i}.$$

    **if** either equality fails **then**
        **return** Reject
    **end if**
    **if** $\alpha_u = \alpha_v$ or $\alpha_u, \alpha_v \notin \{1, \ldots, k\}$ **then**
        **return** Reject
    **end if**
**end for**
**return** Accept

### 3.4 Merkle compression of commitments

In the plain scheme, each round $i$ includes the full commitment vector

$$C^{(i)} = (c^{(i)}_v)_{v \in V}, \qquad c^{(i)}_v = f\big(\pi_i(\phi(v)),\, r^{(i)}_v\big) \in \{0,1\}^s.$$

If all $t$ commitment vectors are included in the signature, the size is

$$|\sigma|_{\text{plain}} \;\approx\; t\, n\, s + 2t\big(|\alpha| + |r|\big),$$

where $n = |V|$, $s$ is the commitment length, $|\alpha| \le \lceil \log_2 k \rceil$, and $|r|$ is the commitment randomness length. This yields asymptotic size $O(tn)$.

We compress each vector $C^{(i)}$ to a single Merkle root $R_i$ using a collision-resistant hash function $H : \{0,1\}^* \to \{0,1\}^\lambda$. Let $\text{enc}(v)$ be a fixed-length encoding of the vertex index $v$, and define

$$L^{(i)}_v = H\big(\mathsf{leaf} \,\|\, \text{enc}(v) \,\|\, c^{(i)}_v\big).$$

The hash function $\mathcal{H}$ is modeled as a random oracle. For concrete instantiations, one should use a standardized hash with output length and security strength matched to the target security level.

A binary Merkle tree is built over the leaves $(L^{(i)}_v)_{v \in V}$, and $R_i$ denotes its root. Instead of publishing the whole vector $C^{(i)}$, the signer publishes only $R_i$. For each challenged vertex, the signature includes the corresponding opening $(\alpha, r)$ together with a Merkle authentication path.

In the Fiat–Shamir transformation, the hash is applied to the roots rather than to the full commitment vectors:

$$h = \mathcal{H}\big(\text{Encode}(G, k, R_0, \ldots, R_{t-1}, M)\big).$$

The challenge edges are then derived from $h$ as before. For each round $i$, if the challenge is $e_i = (u_i, v_i)$, the signer reveals

$$(\alpha_{u_i}, r_{u_i}, \mathsf{path}^{(i)}_{u_i}) \qquad \text{and} \qquad (\alpha_{v_i}, r_{v_i}, \mathsf{path}^{(i)}_{v_i}),$$

where $\alpha_{u_i} = \pi_i(\phi(u_i))$ and $\alpha_{v_i} = \pi_i(\phi(v_i))$.

**Lemma 1.** *Suppose $H$ is collision-resistant and $f$ is binding. Fix a round $i$ and a vertex index $v$. Given the root $R_i$, it is infeasible to produce two distinct valid openings for position $v$.*

*Proof.* If two distinct openings $(\alpha, r) \neq (\alpha', r')$ produce the same commitment value, then the binding property of $f$ is violated. Otherwise the corresponding leaves differ. If both leaves authenticate to the same root $R_i$, this yields a collision in the Merkle tree, contradicting the collision resistance of $H$.

Each Merkle root contributes $\lambda$ bits, and each authentication path has length $\lceil \log_2 n \rceil$. The resulting signature size is therefore

$$|\sigma|_{\text{Merkle}} \;\approx\; t\lambda + 2t\big(|\alpha| + |r| + \lambda\lceil \log_2 n \rceil\big),$$

which gives asymptotic size $O(t \log n)$.

A Python prototype of the Merkle-compressed construction was implemented to validate the encoding, challenge derivation, and signature-size formulas. On a small prototype instance, the measured signature sizes matched the analytic expressions derived above.

## 4 Security model and analysis

We use the standard notion of existential unforgeability under adaptive chosen-message attack (EUF–CMA) for signature schemes, and we analyze the construction in the random-oracle model.

**Proposition 1.** *Assume that $f$ is computationally binding, that $H$ is collision-resistant, and that $\mathcal{H}$ is modeled as a random oracle. Let*

$$\sigma = \big(\{R_i\}_{i=0}^{t-1}, \{open_i\}_{i=0}^{t-1}\big)$$

*be a signature accepted by the verifier for a message $M$. Then, except with negligible probability, for every round $i$, the opening $open_i$ is consistent with the challenged edge derived from*

$$h = \mathcal{H}(\text{Encode}(G, k, R_0, \ldots, R_{t-1}, M)),$$

*and with the commitments authenticated under the Merkle root $R_i$. In particular, any successful forgery must either:*

1. *produce valid openings for all challenged positions under the corresponding Merkle roots, or*
2. *violate the binding of $f$ or the collision resistance of $H$.*

*Proof.* Let

$$h = \mathcal{H}(\text{Encode}(G, k, R_0, \ldots, R_{t-1}, M))$$

be the hash value recomputed by the verifier, and let

$$(e_0, \ldots, e_{t-1})$$

be the sequence of challenged edges derived from $h$. Since the verifier is deterministic once $(G, k, M, \sigma)$ is fixed, acceptance means that, for each round $i$, the verifier has parsed $open_i$, extracted the challenged edge $e_i = (u_i, v_i)$, and accepted all checks attached to that round.

Fix a round $i$. By construction of the verification algorithm, acceptance implies that the verifier has checked the Merkle authentication paths for the two challenged endpoints $u_i$ and $v_i$, starting from the corresponding leaf values and ending at the published root $R_i$. Therefore, if one of these two openings were not consistent with $R_i$, then either the verifier would reject, or two

distinct leaf values would authenticate to the same root. In the latter case, this would contradict the collision resistance of $H$.

Next, the verifier also checks that the revealed pairs $(\alpha_u, r_u)$ and $(\alpha_v, r_v)$ open the corresponding commitments. Hence, if an accepted opening at one of the challenged positions admitted two distinct valid decommitments, this would violate the binding property of $f$.

Therefore, except with negligible probability, every accepted round is simultaneously consistent with the challenged edge derived from the hash value, with the openings checked against the commitment function $f$, and with the Merkle root $R_i$. Applying the same argument to all rounds yields the claim.

The proposition above only establishes consistency of accepted transcripts with the published commitment structure. A complete EUF–CMA reduction would additionally require a witness-extraction argument for the Fiat–Shamir transform in this setting.

## 5 Security Analysis and Attack Models

### 5.1 Exact attacks

To set appropriate parameters and evaluate the security of **Eidolon**, we analyze the practical difficulty of recovering the secret $k$-coloring from the public graph using a well-known algorithmic approach. In the context of this paper, we are interested in any coloring of the random graph with $k$ or less colors, as this allows us to impersonate the prover. For a comprehensive analysis of the problem, see Mann [16]. We use the **exactcolors** algorithm [11], which is based on branch-and-bound with a linear programming solver (Gurobi). This is currently one of the most efficient solvers for the coloring problem on ER-graphs [3].

We begin by confirming the properties of the random graph and the difficulty of finding the chromatic number of a random graph. Figure 1 illustrates the chromatic numbers found using the **exactcolors** solver for graphs drawn from $G_R(n, \frac{1}{2})$ with between 10 and 52 vertices and the minimal coloring found using the DSatur heuristic. We choose $p = 1/2$ as this maximizes the complexity of the graphs and the associated coloring problem. The Bollabas upper and lower bounds are also plotted. It is clear that the actual chromatic numbers are much larger than the asymptotic formula, which does not apply in this range. Instead, the chromatic number seems to follow a power law with best-fit equation $\chi(n, 1/2) = 0.88n^{0.611}$.

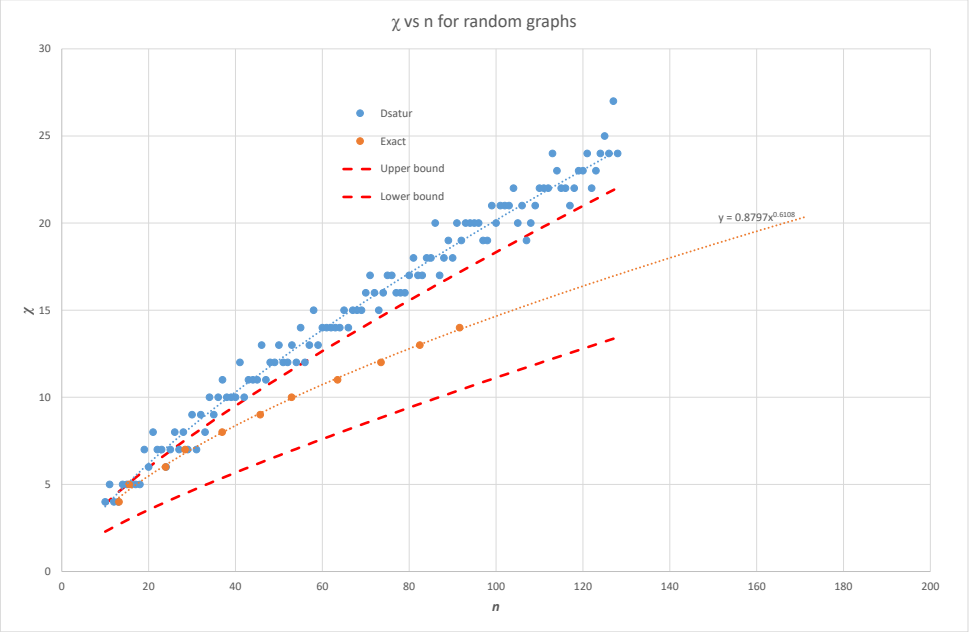


**Fig. 1.** The observed chromatic number of random graphs using an exact algorithm and the minimal coloring discovered by the DSatur algorithm. The upper and lower bounds of $\chi$ due to Bollobás are also plotted.

The secret key is embedded as a planted coloring in the graph. It seems natural to choose $\chi$ as the number of colors for the key, but embedding an additional solution may change the difficulty of the problem. To confirm the correct number of colors, we analyzed the solution time to find a coloring of size $k$ embedded in graphs with expected $\chi = 13(n = 72)$, $\chi = 14(n = 82)$ and $\chi = 15(n = 93)$ (Figure 2. The peak difficulty occurs either at $k = \chi$ or $k = \chi + 1$, and so we choose $k = \chi$.

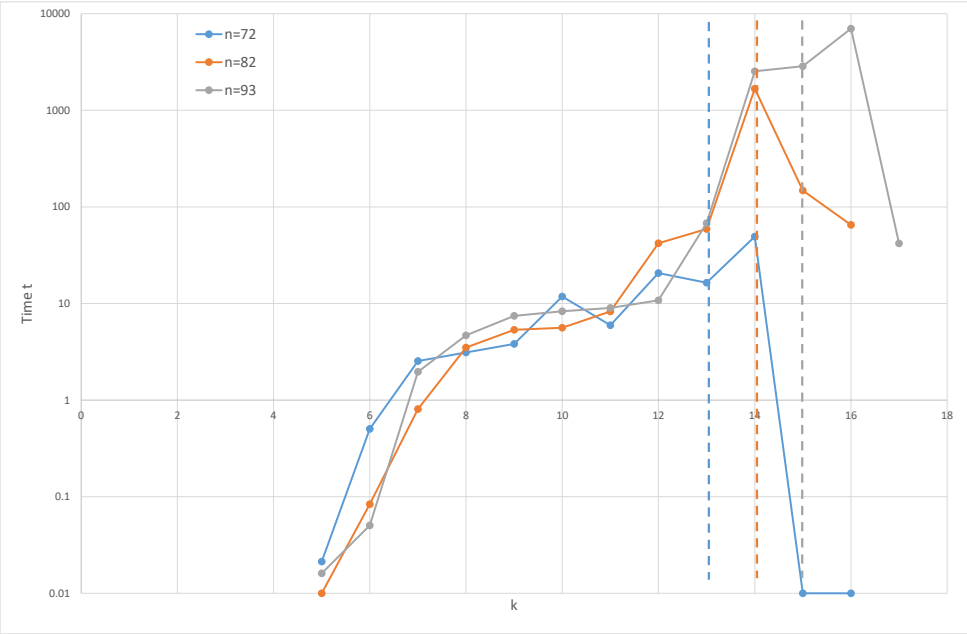


**Fig. 2.** The time taken to find the minimal coloring of graphs with different planted solution size $k$ for graphs of size 72, 82 and 93. The vertical dashed lines represent the expected chromatic number of the equivalent ER graphs.

Finally, we demonstrate the relationship between $n$ and solution time in Figure 3. We select $k$ for the planted solution to match the empirical prediction of $\chi$ for the equivalent random graph. The solution times appears, empirically, to be super-exponential. These results were produced on a AMD EPYC 7501 at 2.6GHz with 512GB of RAM.

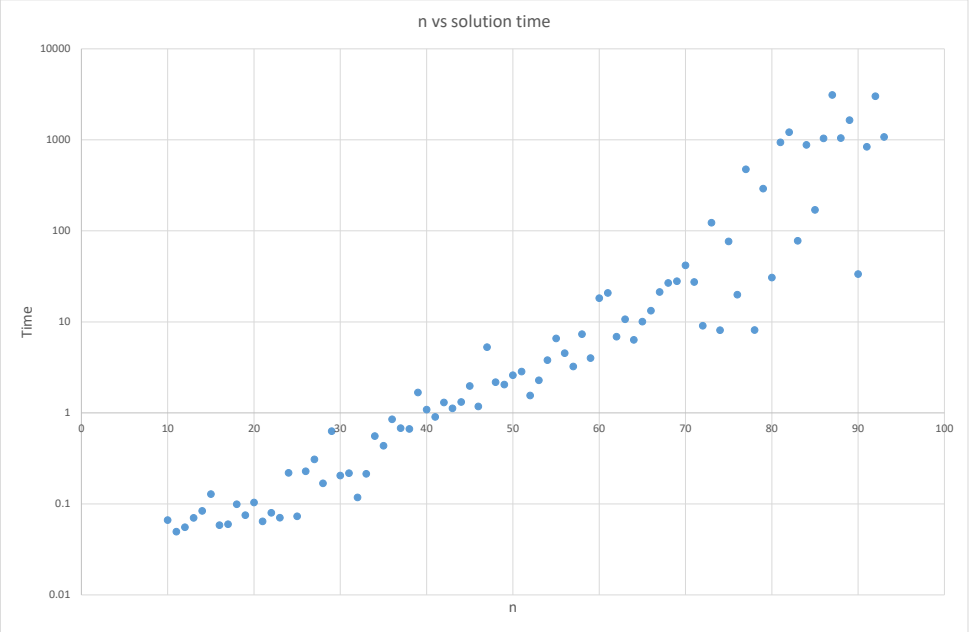


**Fig. 3.** The time-to-solution for the 'exactcolors' solver on planted graphs with planted $k$ chosen according to the empirical power law.

### 5.2 Heuristic attacks

The Dsatur algorithm [4] is a well-known greedy algorithm for coloring graphs. The algorithm sequentially selects vertices for coloring on the basis of the maximum current degree of saturation. It can recover the minimal coloring for small graphs but is known to be limited for larger graphs. We can see this in in Figure 1, where the size of the solution found by Dsatur is shown, along with the chromatic number discovered by the exact algorithm. While the method performs well for small graphs, a gap is evident for larger ones and DSatur cannot recover the minimal coloring for graphs larger than $n = 40$. This result is confirmed in Figure 4 for planted graphs, where we can see identical behavior to the ER-graphs.

In [10], Gryak, Haralick, Kahrobaei have used machine learning algorithms to solve one of the algorithmic problems, known as conjugacy decision problem for certain classes of groups. This technique has been used for the cryptanalysis of proposed schemes using the conjugacy problem. Machine learning and pattern recognition techniques have been successfully applied to algorithmic problems in free groups. In [10], the authors seek to extend these techniques to finitely presented non-free groups, with a particular emphasis on polycyclic and metabelian groups that are of interest to non-commutative cryptography. As a prototypical example, they utilize supervised learning methods to construct classifiers that can solve the conjugacy decision problem, i.e., determine whether or not a pair of elements from a specified group are conjugate. The accuracies

of classifiers created using decision trees, random forests, and N-tuple neural network models are evaluated for several non-free groups. The very high accuracy of these classifiers suggests an underlying mathematical relationship with respect to conjugacy in the tested groups.

Based on these results, we propose a modified version of the Graph Neural Network framework to tackle the GCP. The method is based on that of Schuetz et al [21]. The method uses a set of random features initially assigned to the vertices of the graph. The GNN is then used to decode these features into color labels for the vertices. The network learns through a cost function to minimize the number of color conflicts in the assignment. We use the GraphSAGE variant described in the paper with one hidden layer. We optimized the parameter settings to find $d_0 = 45, d_1 = 40, \beta = 0.0091$, dropout=0.6, epochs=20000.

We use a modified cost function suggested by Porumbel et al [19] which weights color violations by the inverse of the degree, favoring violations with a smaller number of neighbors to resolve. If the softmax output of the network is $\mathbf{p}_k$ for each color $k$, then the standard cost function is given by $\sum_k \mathbf{p}_k^T \mathbf{A} \mathbf{p}_k$. The modified cost function is

$$\sum_k \mathbf{p}_k^T (\mathbf{I} - \mathbf{D}^{-1}) \mathbf{A} \mathbf{p}_k \tag{3}$$

This down-weights the cost of violations on low degree vertices, as these are potentially easier to resolve.

[7]

Since the network is fixed, the expected number of colors, $k$, must be chosen in advance. This is not straightforward, as we found that values of $k$ higher than the known chromatic number of the graph produced better solutions in some cases. However, since the goal is to recover a coloring equivalent to the planted coloring (in order to break the security of the key), we use the same $k$ as the planted coloring.

We found considerable improvement from the network by initializing it with the labels given by the DSatur algorithm. This is achieved by pre-training the network with the DSatur labelling as the target output, using 70 epochs, $\beta = 0.08$ and a cross-entropy loss. Since the DSatur algorithm will typically produce more than $k$ colors, the coloring is reduced by replacing the excess colors with the least-conflicted of the initial $k$ colors before training.

Finally, the network does not usually converge to a non-conflicted configuration (i.e. a graph coloring). We therefore post-process the labelling in a similar fashion to [21], by relabelling conflicted nodes with the smallest number of additional colors.

[7] Our GNN-based attack code is available at `https://github.com/EsmaMaysan/GNN-CSP/blob/main/GNN+CSP.py`.

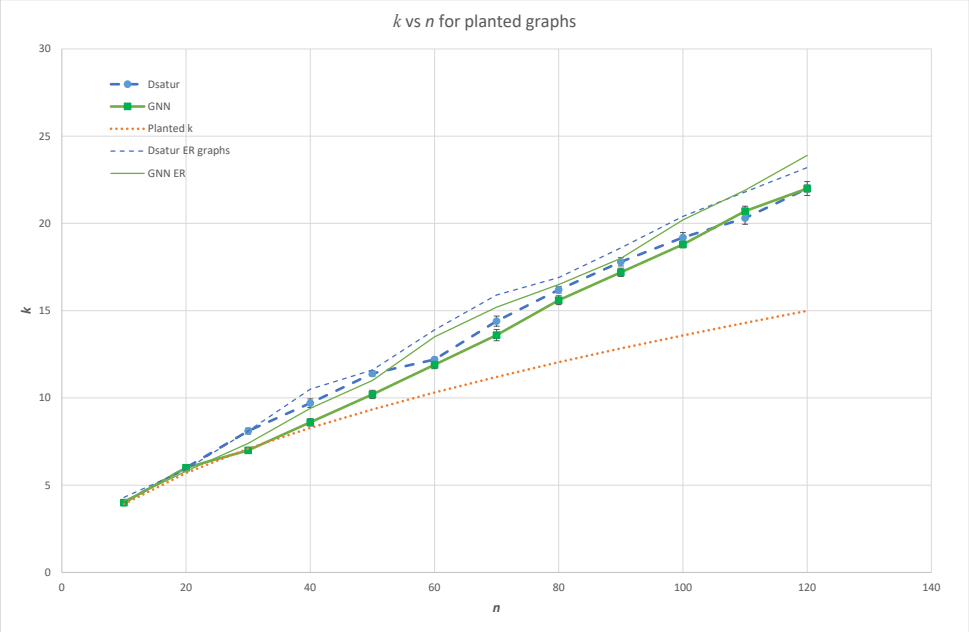


**Fig. 4.** The size of the recovered colorings for planted graphs using approximate algorithms. The dotted line shows the number of colors $k$ used in the planted graphs. The dashed line gives the number of colors used by DSatur, and the solid line those used by the GNN algorithm. The lighter lines show the results for these algorithms on ER graphs, i.e. random graphs with no planted solution.

The results are shown in Figure 4. The GNN algorithm marginally outperforms the Dsatur algorithm for intermediate sized graphs ($n \in [20, 60]$), but cannot recover the planted coloring for $n \geq 20$. Above $n = 60$, the performance is very similar to Dsatur. This suggests for $n > 60$ it is very challenging to recover the key with known algorithms. The plot also contains results for ER random graphs (lighter lines) showing very similar performance to the planted graphs.

## 6 Conclusion

We have presented `Eidolon`, a practical post-quantum digital signature scheme based on the $k$-colorability problem, combining a generalized zero-knowledge protocol, Merkle-tree compression, and carefully constructed hard instances via planted colorings.

Our empirical security analysis provides a detailed evaluation of the resistance of the scheme against both exact and heuristic attacks. In particular, the results obtained using the **exactcolors** solver indicate that the time required to recover a valid $k$-coloring grows super-exponentially with the graph size when parameters are chosen according to the empirical chromatic number. This confirms that recovering the planted coloring becomes computationally infeasible even for moderate values of $n$.

On the heuristic side, both the DSatur algorithm and the proposed GNN-based approach fail to recover the planted coloring beyond small graph sizes. While the GNN model shows slight improvements over DSatur for intermediate sizes ($n \in [20, 60]$), it does not scale effectively and exhibits similar limitations for larger graphs. In particular, for $n \geq 60$, neither method is able to produce a valid coloring matching the planted solution, and their performance remains close to that observed on random graphs without planted structure.

Overall, these results demonstrate that the hardness of the underlying combinatorial problem is preserved even in the presence of modern heuristic and machine learning techniques. This supports the viability of graph coloring as a foundation for post-quantum cryptographic constructions when parameters are carefully selected.

Future work includes refining parameter selection to align with standardized post-quantum security levels, improving the efficiency of the construction, and further investigating the resilience of the scheme against more advanced attacks, including hybrid and adaptive strategies.

**Acknowledgments.** The authors acknowledge the support from the Institut Henri Poincaré (UAR 839 CNRS-Sorbonne Université) and LabEx CARMIN (ANR-10-LABX-59-01). This project started from discussions with Farinaz Koushanfar (UCSD) by Delaram Kahrobaei. Consequently restarted during the Trimester on Post-quantum Algebraic Cryptography at IHP. DK conducted this work partially with the support of ONR Grant 62909-24-1-2002. DK thank Institut des Hautes Études Scientifiques - IHES for providing stimulating environment while this project was partially done. DK was supported by the CARMIN fellowship during the completion of this project. RF thanks IMUS-Maria de Maeztu grant CEX2024-001517-M - Apoyo a Unidades de Excelencia María de Maeztu for supporting this research, funded by MICIU/AEI/ 10.13039/501100011033.

**Disclosure of Interests.** The authors declare that they have no competing interests relevant to the content of this article.

## References

1. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Sha-3 derived functions. Tech. Rep. SP 800-185, NIST (2016)
2. Bollobás, B.: The chromatic number of random graphs. Comb. **8**(1), 49–55 (1988). https://doi.org/10.1007/BF02122551, https://doi.org/10.1007/BF02122551
3. Brand, T., Faber, D., Held, S., Mutzel, P.: A customized sat-based solver for graph coloring. In: Chowdhury, R., Puglisi, S.J., Ren, B., Veldt, N. (eds.) Proceedings of the 28th Symposium on Algorithm Engineering and Experiments, ALENEX 2026, Vancouver, BC, Canada, January 11-12, 2026. pp. 142–155. SIAM (2026). https://doi.org/10.1137/1.9781611978957.11, https://doi.org/10.1137/1.9781611978957.11
4. Br'elaz, D.: New methods to color the vertices of a graph. Communications of the ACM **22**(4), 251–256
5. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings. pp. 186–194. Lecture Notes in Computer Science, Springer (1986). https://doi.org/10.1007/3-540-47721-7_12, https://doi.org/10.1007/3-540-47721-7_12
6. Fluhrer, S.e.a.: Hashing to elliptic curves. Tech. Rep. RFC 9380, IETF (2023)
7. Garey, M.R., Johnson, D.S.: The complexity of near-optimal graph coloring. J. ACM **23**(1), 43–49 (1976). https://doi.org/10.1145/321921.321926, https://doi.org/10.1145/321921.321926
8. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. J. ACM **38**(3), 691–729 (1991). https://doi.org/10.1145/116825.116852, https://doi.org/10.1145/116825.116852

9. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Comput. **17**(2), 281–308 (1988). https://doi.org/10.1137/0217017, https://doi.org/10.1137/0217017
10. Gryak, J., Haralick, R., Kahrobaei, D.: Solving the conjugacy decision problem via machine learning. Experimental Mathematics **29**(1), 66–78 (2020)
11. Held, S., Cook, W.J., Sewell, E.C.: Maximum-weight stable sets and safe lower bounds for graph coloring. Math. Program. Comput. **4**(4), 363–381 (2012). https://doi.org/10.1007/S12532-012-0042-3, https://doi.org/10.1007/s12532-012-0042-3
12. Katz, J.: Comparative book review: Cryptography: An introduction by v. v. yaschenko (american mathematical society, 2002); cryptanalysis of number theoretic ciphers by S.S. wagstaff, jr. (chapman & hall/crc press, 2003); RSA and public-key cryptography by r. a. mollin (chapman & hall/crc press, 2003); foundations of cryptography, vol. 1: Basic tools by o. goldreich, (cambridge university press, 2001). SIGACT News **36**(2), 14–19 (2005). https://doi.org/10.1145/1067309.1067316, https://doi.org/10.1145/1067309.1067316
13. Katz, J., Lindell, Y.: Introduction to Modern Cryptography, Second Edition. CRC Press (2014), https://www.crcpress.com/Introduction-to-Modern-Cryptography-Second-Edition/Katz-Lindell/p/book/9781466570269
14. Krawczyk, H., Eronen, P.: Hmac-based extract-and-expand key derivation function (HKDF). RFC **5869**, 1–14 (2010). https://doi.org/10.17487/RFC5869, https://doi.org/10.17487/RFC5869
15. Krzakala, F., Zdeborová, L.: Hiding quiet solutions in random constraint satisfaction problems. CoRR **abs/0901.2130** (2009), http://arxiv.org/abs/0901.2130
16. Mann, Z.Á.: Complexity of coloring random graphs: An experimental study of the hardest region. ACM J. Exp. Algorithmics **23** (2018). https://doi.org/10.1145/3183350, https://doi.org/10.1145/3183350
17. McDiarmid, C.: Colouring random graphs. Ann. Oper. Res. **1**(3), 183–200 (1984). https://doi.org/10.1007/BF01874388, https://doi.org/10.1007/BF01874388
18. Pornin, T.: Deterministic usage of the digital signature algorithm (DSA) and elliptic curve digital signature algorithm (ECDSA). RFC **6979**, 1–79 (2013). https://doi.org/10.17487/RFC6979, https://doi.org/10.17487/RFC6979
19. Porumbel, D.C., Hao, J., Kuntz, P.: A study of evaluation functions for the graph k-coloring problem. In: Monmarché, N., Talbi, E., Collet, P., Schoenauer, M., Lutton, E. (eds.) Artificial Evolution, 8th International Conference, Evolution Artificielle, EA 2007, Tours, France, October 29-31, 2007, Revised Selected Papers. pp. 124–135. Lecture Notes in Computer Science, Springer (2007). https://doi.org/10.1007/978-3-540-79305-2_11, https://doi.org/10.1007/978-3-540-79305-2_11
20. Schuetz, M.J.A., Brubaker, J.K., Katzgraber, H.G.: Combinatorial optimization with physics-inspired graph neural networks. Nature Machine Intelligence **4**, 367–377 (2022)
21. Schuetz, M.J.A., Brubaker, J.K., Zhu, Z., Katzgraber, H.G.: Graph coloring with physics-inspired graph neural networks. CoRR **abs/2202.01606** (2022), https://arxiv.org/abs/2202.01606

## A Standard security definitions

**Definition 1 (Probabilistic polynomial-time algorithms and negligible functions [12, Sec. 1.3],[13, Sec. 3.1]).** *An algorithm $\mathcal{A}$ is* probabilistic polynomial time *(PPT) if there exists a polynomial $p(\cdot)$ such that, for every input $x \in \{0,1\}^n$ and every random coin string $r$, the execution $\mathcal{A}(x;r)$ halts within at most $p(n)$ steps. A function $\mu : \mathbb{N} \to [0,1]$ is* negligible *if, for every polynomial $q(\cdot)$, there exists $N$ such that $\mu(n) < 1/q(n)$ for all $n \geq N$.*

**Definition 2 (EUF–CMA security [9, p. 21],[13, Ch. 12]).** *Let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vfy})$ be a signature scheme with security parameter $\lambda$. The EUF–CMA experiment is defined as follows.*

1. Setup: *The challenger samples* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ *and gives* $\mathsf{pk}$ *to the adversary $\mathcal{A}$.*
2. Signing queries: *The adversary $\mathcal{A}$ is given adaptive oracle access to* $\mathsf{Sign}_{\mathsf{sk}}(\cdot)$. *For each queried message $m$, it receives a signature $\sigma \leftarrow \mathsf{Sign}_{\mathsf{sk}}(m)$. Let $Q$ denote the set of queried messages.*

*3.* Forgery: *Eventually,* $\mathcal{A}$ *outputs a pair* $(m^\star, \sigma^\star)$.

*The adversary wins if*

$$\mathsf{Vfy}_{\mathsf{pk}}(m^\star, \sigma^\star) = 1 \qquad \textit{and} \qquad m^\star \notin Q.$$

*The scheme* $\Pi$ *is* EUF–CMA secure *if every PPT adversary wins with only negligible probability in* $\lambda$.