Analysis of Watermarking Techniques for Graph Coloring Problem

Gang Qu and Miodrag Potkonjak
Computer Science Department University of California, Los Angeles, CA 90095-1596
{gangqu,miodrag}@cs.ucla.edu

Abstract

We lay out a theoretical framework to evaluate watermarking techniques for intellectual property protection (IPP). Based on this framework, we analyze two watermarking techniques for the graph coloring(GC) problem . Since credibility and overhead are the most important criteria for any efficient watermarking technique, we derive formulae that illustrate the trade-off between credibility and overhead. Asymptotically we prove that arbitrarily high credibility can be achieved with at most 1-color-overhead for both proposed watermarking techniques.

1 Introduction

Protecting software from piracy is one of the most crucial issues in computer science. The time-to-market pressure drives intellectual property (IP) into the center of several trends sweeping through today's electronic design automation (EDA) and application specific integrated circuits (ASIC) industries. The requirement for the exchange of IP in the design of system-on-chip is well documented. As the price of hardware drops and the price of licensed software goes up, piracy becomes more lucrative. From the IP providers' point of view, there is an urgent need for protection technique(s) to recoup huge R&D investments on their IP and to keep profits beyond the reach of pirates.

Watermarking or data hiding is designed to meet this demand. Basically, watermarking intentionally embeds digital information into the software for purposes such as identification and copyright. Such information could be the author's name, company name or other messages highly related to the owner and/or the legal users of the software. If necessary, this information can be used in court to prove the authorship of the software or the legal users entitled to distribute copies.

In recent years, the industry of IPP has grown vigorously. Numerous techniques have been proposed to watermark digital image, audio, video, text, and even hardware design process. Ideas were explained, experiments were carried out, and the experimental results have been well analyzed. Most authors also discussed the advantages and disadvantages of their proposed techniques. However, such discussion is anecdotal and lacks formal analysis and systematic comparison of existing watermarking techniques. As the market for IPP techniques expands, IP providers will face more and more choices and it will be natural for them to ask the question: *which technique is the best to watermark our IP?* Among the most common concerns are:

- how much information can we embed without degrading the quality of our IP?
- how much information do we have to embed to prove our authorship?

These are the major questions we address in this paper. As the first attempt to theoretically analyze watermarking techniques, the primary objective of this paper is to lay out analytical foundations for all kinds of watermarking techniques, not only for those discussed here.

We take two watermarking techniques for the graph coloring problem as examples. The first one forces some well-chosen pairs of vertices to be labeled with different colors by adding extra edges in between. The second technique selects one (or more) independent set(s) from the original graph and marks each set with exactly one color.

We explain each technique by a small example, then we do the asymptotical analysis which answers the above questions about credibility and overhead (number of extra colors required in the GC). Surprisingly, the result shows that arbitrarily high credibility can be achieved with at most one color overhead for both techniques we propose. This is tested by numerical simulation on random graphs. Finally we color several sets of random graphs, graphs from real-life benchmarks and the DIMACS challenge graphs. For most instances, the watermarked graphs can be colored with no overhead with the same amount of run-time.

In the next section, we briefly review the related work. We propose and discuss two watermarking techniques for GC in the following two sections. We report the experimental results in section 5 and then conclude.

2 Related Work

The most relevant related work are efforts in digital watermarking, IPP, and the theory of random graphs.

An efficient digital watermark must have high credibility, low overhead and be resilient, transparent and perceptual invisible. Recently, many techniques for watermarking digital data (text, image, audio, video and multimedia) have been developed [1, 3, 4, 10, 12]. These techniques simply add a signature to the digital data and thus change the original data. The transparency of the signature relies on human's insensitiveness to the changes of the data.

Watermarking for the purpose of IPP, on the other hand, is more difficult because it has to take into consideration of the correct functionality of the watermarked IP. One method, called *constraint-based watermarking*, translates the to-be-embedded signature into a set of additional constraints during the design and implementation of IP in order to uniquely encode the signature into the IP. This has been effectively applied at the level of behavior[6], logic synthesis and physical design[7], as well as in FPGA [9].

Random graphs play a very important role in many fields of computer science. The two most frequently occurring models of random graphs are $\mathcal{G}(n, M)$ and $\mathcal{G}(n,p)$. The first consists of all graphs with n vertices and M edges, the second consists of all graphs with n vertices and the edges are chosen independently with probability $p(0 < p < 1)$. We will focus on the second model and use these conventional notations: $G_{n,p}$ for an element of $\mathcal{G}(n,p)$, $q = 1 - p$, $b = \frac{1}{q}$. $\alpha(G_{n,p})$ is the independent number of graph $G_{n,p}$ (i.e., the maximal cardinality of independent sets.), and $\chi(G_{n,p})$ denotes the chromatic number of $G_{n,p}$ (i.e., the minimum number of colors required to color the graph.). For almost all graphs $G_{n,p}$, we have [2]:

$$(1) \qquad \alpha(G_{n,p}) = (2 + o(1)) \log_b n$$

$$(2) \qquad \chi(G_{n,p}) = (\tfrac{1}{2} + o(1)) \tfrac{n}{\log_b n}$$

The graph (vertex) coloring problem is to label the vertices of a graph with minimal number of colors such that vertices connected by an edge are not labelled with the same color. In the next two sections, we propose two techniques for watermarking the GC, and lay out the theoretical framework of technique evaluation through the analysis of these two techniques.

3 Watermarking Technique #1 — Adding Edges

Given a graph $G(V, E)$ and a message M to be embedded in G. We order the vertices set $V = (v_0, v_1, ..., v_{n-1})$ and convert the message to binary (e.g. using ASCII) $M = m_0 m_1 \ldots$. The message M is embedded into the graph G as follows:

```
Input: a graph $G(V, E)$,
       a message $M = m_0 m_1 \ldots$
Output: new graph with message $M$ embedded
Algorithm:
  copy $G(V, E)$ to $G'(V, E')$;
  foreach bit $m_i$
    { find the nearest two vertices $v_{i_1}$, $v_{i_2}$ that
      are not connected to vertex $v_i$;
      if $m_i = 0$ add edge $(v_i, v_{i_1})$ to $E'$
      else add edge $(v_i, v_{i_2})$ to $E'$
    }
  report graph $G'(V, E')$;
```

Figure 1: Pseudo code for watermarking technique # 1.

By the nearest two vertices v_{i_1} and v_{i_2} which are not connected to vertex v_i, we mean that $i_2 > i_1 > i \pmod n$, the edges $(v_i, v_{i_1}), (v_i, v_{i_2}) \notin E$ and $(v_i, v_j) \in E$ for all $i < j < i_1, i_1 < j < i_2 \pmod n$. The essence of this technique is that by adding an extra edge between two vertices chosen based on the message to be embedded, these two vertices have to be colored by different colors that may not be necessary in the original graph G.

Figure 2 shows a graph where message $1998_{10} = 11111001110_2$ has been embedded by the dotted edges.

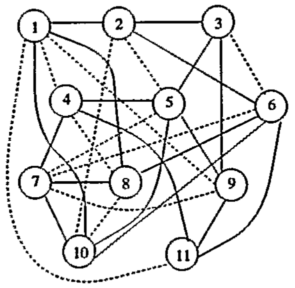

Figure 2: Example for watermarking technique # 1.

We take the following basic assumptions for the simplicity of a non-trivial analysis.

- the graph is a random graph $G_{n,p}$,
- the message to be embedded is random, and
- to color $G_{n,p}$, we need exactly χ colors, where χ is given by:

$$(3) \qquad \chi(G_{n,p}) = \lceil \tfrac{n}{2 \log_b n} \rceil$$

It follows immediately that after embedding k bits into the graph $G_{n,p}$ by adding k extra edges according to the watermark, the resulting graph remains random with the same number of vertices and a new edge probability:

$$(4) \qquad p' = p + \tfrac{2k}{n(n-1)}$$

So formula (3) for the chromatic number still holds, we denote this number by χ'. The *overhead* is defined to be $\chi' - \chi$, i.e., the number of extra colors used to color the watermarked graph.

Intuitively, the more information we embed, the more color we may require to mark the graph. One natural question is: how much information can we embed into the graph without introducing a large amount of overhead?

Theorem 3.1
Adding $k(n)$ edges to a random graph $G_{n,p}$, $\overline{\lim}_{n\to\infty} \chi' - \chi = \infty$ if and only if $k(n) \in \omega(n \log n)$.

Corollary 3.2
Adding $k(n)$ edges to graph $G_{n,p}$, if $\lim_{n\to\infty} \frac{k(n)}{n \ln n} = l$, then $\overline{\lim}_{n\to\infty} \chi' - \chi \leq 1 + \lceil \frac{l}{1-p} \rceil$. In particular, if $k(n) \in o(n \log n)$, the overhead is at most 1.

$\overline{\lim}_{n\to\infty} \chi' - \chi$ measures the absolute value of overhead, another measure for the overhead could be $\overline{\lim}_{n\to\infty} \frac{\chi' - \chi}{\chi}$. It is easy to prove that $\overline{\lim}_{n\to\infty} \frac{\chi' - \chi}{\chi} = 0$ iff $k(n) \in o(n^2)$.

By enforcing a pair of vertices to be labelled with different color, we embed one bit of our signature. The next question is: how many bits do we need to embed to provide a strong proof of our signature?

Theorem 3.3
Adding $k(n)$ edges to a random graph $G_{n,p}$, let $\mathcal{E}$ be the event that these edges are added randomly, then $\lim_{n\to\infty}$ **Prob**$[\mathcal{E}] = 0$ if $k(n) \in \omega(\frac{n}{\log n})$.

To summarize the "*adding edges*" watermarking technique, we see if we add $k(n) \in \omega(\frac{n}{\log n}) \cap o(n \log n)$ extra edges into graph $G_{n,p}$, as n goes large, arbitrarily high credibility can be achieved with at most 1-color-overhead. More precisely, we define the *watermark potential*(by adding edges) for graph $G_{n,p}$:

$$(5) \qquad WP(G_{n,p}) = \chi(G_{n,p}) - \tfrac{n}{2 \log_b n}$$

This function describes the power of the "*adding edges*" watermarking technique on random graphs.

4 Watermarking Technique #2 — Selecting MIS

A maximal independent set (MIS) of a graph is a subset,S, of vertices such that vertices in S are not connected and those not in S are connected to at least one vertex of S. This second technique takes advantage of the fact that vertices in one MIS can be labelled with only one color.

Given a graph $G(V, E)$ and a message M to be embedded in G. We order the vertices set $V = (v_0, v_1, ..., v_{n-1})$ and convert the message to binary (e.g. using ASCII) $M = m_0 m_1 \ldots$. The message M is embedded into the graph G as shown in Figure 3.

The MIS containing M is constructed in the following way: the vertex v_i, where i is equal to the value of the first $\lfloor \log_2 n \rfloor$ bits of M, is selected as the first vertex of the MIS. Once v_i has been selected, all its neighbors cannot be in the same MIS, so we cut them as well as v_i itself. The remaining vertices are reordered and the process continues. When we get a MIS, we color it with one

```
Input: a graph $G(V, E)$,
       a message $M = m_0 m_1 \dots$
Output: new graph with message $M$ embedded
Algorithm:
 current_graph = original_graph $G(V, E)$;
 previous_graph = current_graph;
 MIS = $\phi$;
 do
 { if (current_graph is empty)
    {  current_graph = previous_graph - MIS;
       previous_graph = current_graph;
       report MIS;
       MIS = $\phi$;
    }
    if (current_graph has more than 2 vertices or is connected)
    {  find the vertex $v$ corresponding to the next $\lfloor \log_2 n \rfloor$
          bits of $M$, where $n$ is the size of the current graph.
       cut $v$ and all its neighbors from the current graph.
       current_graph = current_graph - {$v$ and its neighbors};
       MIS = MIS + $v$;
       reorder the vertices in current_graph;
       advance message $M$.
    } else
       MIS = MIS + current_graph;
 }while($M$ is not empty)
 report the current_graph;
```

Figure 3: Pseudo code for watermarking technique # 2.

color, remove it from the original graph and start constructing a second MIS if M has not been completely embedded.

A small example of an 11-node graph with the embedded message $1998_{10} = 11111001110_2$ is shown in Figure 4, where we color the graph with three colors $\{v_1, v_4, v_7, v_{10}\}$, $\{v_0, v_2, v_5, v_6\}$, and $\{v_3, v_8, v_9\}$.

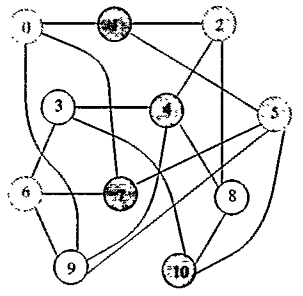

Figure 4: Example for watermarking technique # 2.

If we apply the algorithm in Figure 3 on a random graph $G_{n,p}$, the first vertex can be selected arbitrarily, and its pn neighbors will be eliminated. So there will be at most $(1-p)n = qn$ vertices qualified for the second vertex of the same MIS. In general, we have:

Lemma 4.1
For almost all random graph $G_{n,p}$, the first MIS constructed by this technique is of size $\log_b n$, where $b = \frac{1}{1-p}$.

To get a credibility watermark, we have to add $\omega(\frac{n}{\log n})$ edges by the first technique, here we only need to select one MIS intentionally as:

Theorem 4.2
Given a random graph $G_{n,p}$, select an MIS as in Figure 3, let $\mathcal{E}$ be the event that events that this MIS is chosen randomly, then $\lim_{n\to\infty}$ **Prob** $[\mathcal{E}] = 0$. Furthermore, this introduces at most 1-color-overhead.

How much information have we embedded in this MIS? By selecting one vertex from an n-vertex graph, we can embed $\lfloor \log_2 n \rfloor$ bits. From Lemma 4.1, at most $\log_2 n \log_b n$ bits of information could be embedded into the MIS. To embed long messages, we have to construct more MISes, which may result in huge overhead.

Theorem 4.3
Given a random graph $G_{n,p}$, select $k(n)$ MISes as in Figure 3, then the overhead is at most $k(n)$ and on average at least $\frac{k(n)}{2}$.

Corollary 4.4
Given a random graph $G_{n,p}$, select one MIS as in Figure 3, let $\mathcal{E}$ be the event that this MIS is chosen randomly. Also for the same original graph $G_{n,p}$, add $k(n)$ edges as in Figure 1, let $\mathcal{E}'$ be the event that these edges are added randomly. We have

(6) $\lim_{n\to\infty}$ **Prob** $[\mathcal{E}] \le \lim_{n\to\infty}$ **Prob** $[\mathcal{E}']$

for all $k(n) \in o(n \log n)$.

5 Experimental Results

The main goal of our experiment is to compare the difficulty of coloring the original graph vs. the watermarked graph, as well as the quality of the solution. For this purpose, we choose three types of graphs: random graphs $G_{n,p}$, graphs generated from real-life benchmarks, and the DIMACS challenge graphs.

For each type of graphs, we do the simulation in three steps: (1) color the original graph, (2) apply the watermarking techniques to embed a random message, (3) color the watermarked graph. Each graph is colored 10 times and the average result is reported. All experiments are conducted on 200MHz UltraSparcII and 40 MHz SPARC 4 processors using the algorithm in [8]. The same parameters are used for the original and watermarked graph.

Original $G_{n,0.5}$		Adding n Edges		Adding $2n$ Edges		Selecting 1 MIS		Selecting 2 MISes	
n	color	color	mesg	color	mesg	color	mesg	color	mesg
125	19	19	125	19	250	19	42	19	84
250	30	30.2	250	30.2	500	30	80	30.1	144
500	50.1	50.4	500	50.6	1000	50.2	81	50.6	162
1000	85.8	86	1000	86.8	2000	86	110	86	210

Table 1: Coloring the watermarked $G_{n,0.5}$.

Table 1 shows the results on random graphs $G_{n,0.5}$, and the corresponding watermarked graphs by adding n and $2n$ random edges or by selecting the first two MISes. The columns labeled *color* are the average numbers of colors on 10 trials for each instance, while the columns *mesg* measure the amount of information (in bits) being embedded in the graph. We do not list the optimal solutions from the 10 trials for each instance due to the space constraint, however, it is worth mentioning here that only in one case, didn't we find an optimal solution with no overhead, that is when we add one 2000-bit-message into $G_{1000,0.5}$.

Original $G_{n,p}$			Adding	Adding	Selecting	Selecting
n	p	color	n Edges	$2n$ Edges	1 MIS	2 MISes
125	0.9	46	49.1	52	47	47.3
250	0.9	77.9	79	82	77.2	77
250	0.1	9	9.1	9.8	9	9.8
500	0.1	13.9	14	14.2	14	14.2

Table 2: Coloring other watermarked $G_{n,p}$.

Table 2 is the result on dense/sparse random graphs. For dense graphs $G_{n,0.9}$, there is not much space left to add extra edges, so it is expensive to watermark dense graphs by adding edges. On the other hand, the size of MIS for dense graph is relatively small,

Original Instance			Optimal	Add Edge		Select One MIS		Select Two MISes	
Instance	Vertices	Edges	Coloring	Edges	Overhead	Vertices	Overhead	Vertices	Overhead
fpsol2.i.1.col	496	11654	65	496	0	229	0	231	0
fpsol2.i.2.col	451	8691	30	451	0	90	0	92	0
fpsol2.i.3.col	425	8688	30	425	0	64	0	66	0
inithx.i.1.col	864	18707	54	864	0	347	0	349	0
inithx.i.2.col	645	13979	31	645	0	89	0	91	0
inithx.i.3.col	621	13969	31	621	0	64	0	66	0
mulsol.i.1.col	197	3925	49	197	0	61	0	63	0
mulsol.i.2.col	188	3885	31	188	1	17	0	19	0
mulsol.i.3.col	184	3916	31	184	1	12	0	14	0
mulsol.i.4.col	185	3946	31	185	0	12	0	14	0
mulsol.i.5.col	186	3973	31	186	0	12	0	14	0
zeroin.i.1.col	211	4100	49	211	0	87	0	89	0
zeroin.i.2.col	211	3541	30	211	0	56	0	58	0
zeroin.i.3.col	206	3540	30	206	0	51	0	53	0

Table 3: Coloring watermarked real-life benchmarks.

therefore very limited information can be embedded by selecting MISes. For sparse graphs $G_{n,0.1}$, both techniques perform well.

When applying to the on-line challenge graphs at the DIMACS site [5], for the graph with 1000 vertices and 499652 edges which implies an edge probability slightly larger than 0.5, we restrict the run-time to 1 hour and get the results from 10 trials shown in Table 4. In the 10 trials for the original graph, we do find two 85-color solutions and the average number of colors is 86.1. The second column is the amount of information (in bits) being added into the graph. The last column shows the probability of coincidence, where low coincidence means high credibility. One can see both methods provide high credibility with little degradation of the solution's quality.

Edges Added	Information	Colors	Overhead	Coincidence
0	0	86.1	-	-
500	500	85.8	-0.3	2.89e-03
1000	1000	87	0.9	9.55e-06
3000	3000	87	0.9	8.71e-16
MIS Selected	Information	Colors	Overhead	Coincidence
1	110	86.2	0.1	3.52e-17
2	210	86.4	0.3	1.24e-33
3	300	87	0.9	8.51e-50
4	390	87.4	1.3	5.85e-66
5	480	87.4	1.3	2.06e-82
6	590	87.9	1.8	7.24e-99

Table 4: Coloring the watermarked DIMACS benchmark.

GC has a lot of applications in real life, for example, the register allocation problem, the cache-line coloring problem, wavelength assignment in optical networks, and channel assignment in cellular systems.

The instances of GC based on register allocation of variables in real codes and the optimal solutions are available at [11]. We watermark these graphs and then color them. The *fpsol2* and *inithx* instances are colored in 1 $\sim$ 3 minutes, while the others are all colored in less than 0.5 minute.

Table 3 reports the details. The first four columns shows the characteristic of the original graph and the known optimal solution; the next two are for technique #1, showing the number of edges (information in bits) being embedded and the overhead; the rest are for technique #2, where the *Vertices* columns are the number of vertices in the selected MIS(es). Again, in almost all examples, there is no overhead.

6 Conclusion

In this paper, we build the first theoretical framework for analyzing watermarking techniques. We propose two techniques for watermarking the graph coloring problem, which are provably capable to provide high credibility with at most 1-color overhead for large graphs. Asymptotic formulae are given on the amount of information that can be embedded into the graph without too much overhead as well as the amount of information that should be embedded to provide high credibility. Also, we watermark and then color a large range of graphs from random graphs, DIMACS challenge graphs to graphs generated from real life problems. With the same amount of run-time as that for the original graphs, for almost all instances, we obtain the optimal solution with no overhead.

References

[1] H.Berghel and L.O'Gorman. *Protecting ownership rights through digital watermarking.* IEEE computer, 29(7): 101-103, 1996.

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

[3] L.Boney, A.H.Tewfik, and K.N.Hamdy. *Digital watermark for audio signals.* International Conference on Multimedia Computing and Systems, pp. 473-480, 1996.

[4] I.J.Cox, J.Kilian, T.Leighton, and T.Shamoon. *A secure, imperceptible yet perceptually salient, spread spectrum watermark for multimedia.* Southcon, pp. 192-197, 1996.

[5] http://dimacs.rutgers.edu/

[6] I.Hong and M.Potkonjak. *Personal communication*

[7] A.B.Kahng, S.Mantik, I.L.Markov, M.Potkonjak, P.Tucker, H.Wang and G.Wolfe. *Robust IP Watermarking Methodologies for Physical Design.* 35th Design Automation Conference Proceedings, pp. 782-787, 1998

[8] D.Kirovski and M.Potkonjak. *Efficient Coloring of a Large Spectrum of Graphs.* 35th Design Automation Conference Proceedings, pp. 427-432, 1998.

[9] J.Lach, W.H.Mangione-Smith, and M.Potkonjak. *FPGA Fingerprinting Techniques for Protecting Intellectual Property.* Proceedings of CICC, 1998.

[10] M.D.Swanson, B.Zhu, B.Chau, and A.H.Tewfik. *Object-based transparent video watermarking.* IEEE Workshop in Multimedia Signal Processing, pp. 369-374, 1997.

[11] http://mat.gsia.cmu.edu/COLOR/instances.html

[12] M.M.Yeung, F.C.Mintzer, G.W.Braudaway, and A.R.Rao. *Digital watermarking for high-quality imaging.* IEEE Workshop on Multimedia Signal Processing, pp. 357-362, 1997.