Iterative single Tardos decoder with controlled probability of false positive

Peter Meerwald and Teddy Furon

INRIA Rennes Bretagne Atlantique, 35000 Rennes, France

Email: {peter.meerwald,teddy.furon}@inria.fr

Abstract

- Traitor tracing based on Tardos codes
- Iterative accusation algorithm with side information to catch as many colluders as possible

Traitor Tracing

Identify a small set of dishonest users illegally distributing their content copies. Embed the user's codeword in the content copy via watermarking. The content is split into blocks and each block carries a '0' or '1' symbol.

Setup

- m: number of bits in codeword, c: number of colluders
- n: number of users/codewords $\mathbf{x}_j = (x_j(1), \dots, x_j(m))$

Coding The Tardos code [1] is the optimum code construction. A matrix $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$ is generated:

1. Randomly draw sequence $\mathbf{p} = (p(1), \dots, p(m))$ with $p(i) \overset{i.i.d}{\sim} f(p) : (0,1) \to \mathbb{R}^+$, $p \to (\pi^2(1-p))^{-1/2}$.
2. Randomly draw $x_j(i)$ s.t. $\mathbb{P}(x_j(i) = 1) = p(i)$.

Collusion The colluders mix their copies to forge a pirated copy. The watermark decoder retrieves a pirated sequence $\mathbf{y} \in \{0,1\}^m$. Marking assumption: $y(i) \in \{x_j(i)\}$ for $1 \le j \le n$.

Decoding Identify the colluders given $\mathbf{y}$, $\mathbf{p}$ and $\mathbf{X}$.

Goal Identify as many colluders as possible while maintaining a low probability of false accusation, e.g. $P_{\mathsf{fp}} = 10^{-3}$.

Accusation Process

The optimal single decoder is given by [2, Sec. 3.1]:

$$s_j = \sum_{i=1}^{m} \log \frac{\mathbb{P}(y(i)|x_j(i), p(i))}{\mathbb{P}(y(i)|p(i))}, \quad (1)$$

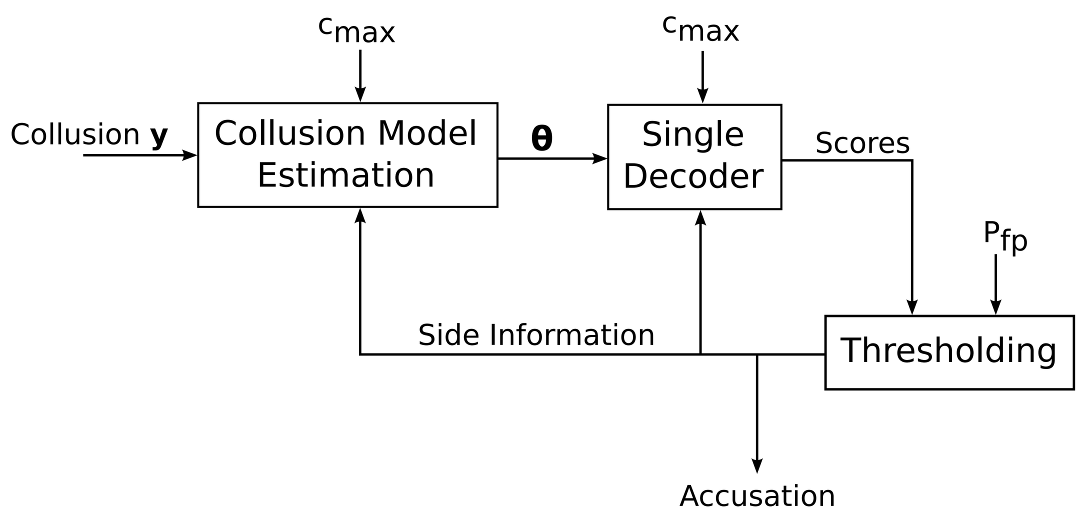
with

$$\mathbb{P}(y(i) = 1) = \textstyle\sum_{\sigma=0}^{c} \theta(\sigma)\mathbb{P}(\sigma|p(i)), \quad (2)$$

$$\mathbb{P}(\sigma|p(i)) = \tbinom{c}{\sigma} p(i)^{\sigma}(1-p(i))^{c-\sigma}, \quad (3)$$

$$\mathbb{P}(y(i) = 1|x_j(i)) = \textstyle\sum_{\sigma=x_j(i)}^{c-1+x_j(i)} \theta(\sigma)\mathbb{P}(\sigma|x_j(i), p(i)), \quad (4)$$

$$\mathbb{P}(\sigma|x, p(i)) = \tbinom{c-1}{\sigma-x} p(i)^{\sigma-x}(1-p(i))^{c-1-\sigma+x}. \quad (5)$$



Overview of the iterative, side-informed decoder.

Colluders are expected to have higher scores than innocent users. We accuse users whose scores are higher than threshold τ.

Three cases are possible:

- (i) m is big enough and the c colluders' scores are ranked first,
- (ii) some but not all the colluders are ranked first,
- (iii) m is too short and one innocent has the biggest score.

Iterative decoding In case (ii), at least one colluder is caught and added as side information to the set $\mathcal{X}_{\mathsf{SI}}$. This allows

- More discriminative scores
- More accurate collusion model estimation

Let $\rho_i = \sum_{j \in \mathcal{X}_{\mathsf{SI}}} x_j(i)$. This changes equations (2) - (5) to:

$$(2) \leftarrow \sum_{\sigma=\rho_i}^{c-k+\rho_i} \theta(\sigma)\mathbb{P}(\sigma|p(i)),$$

$$(3) \leftarrow \tbinom{c-k}{\sigma-\rho_i} p(i)^{\sigma-\rho_i}(1-p(i))^{c-k-\sigma+\rho_i},$$

$$(4) \leftarrow \sum_{\sigma=x_j(i)+\rho_i}^{c-k-1+x_j(i)+\rho_i} \theta(\sigma)\mathbb{P}(\sigma|x_j(i), p(i)),$$

$$(5) \leftarrow \tbinom{c-k-1}{\sigma-\rho_i-x} p(i)^{\sigma-\rho_i-x}(1-p(i))^{c-k-1-\sigma-\rho_i+1}.$$

Thresholding

- Generate new codewords of innocents based on $\mathbf{p}$ and compute their scores.
- Estimate the threshold τ such that the probability of being an innocent is below ϵ using Monte-Carlo simulation.
- Large n implies a too small probability $\epsilon = n^{-1}P_{\mathsf{fp}}$. For this reason we implement an estimator based on rare event analysis [3].

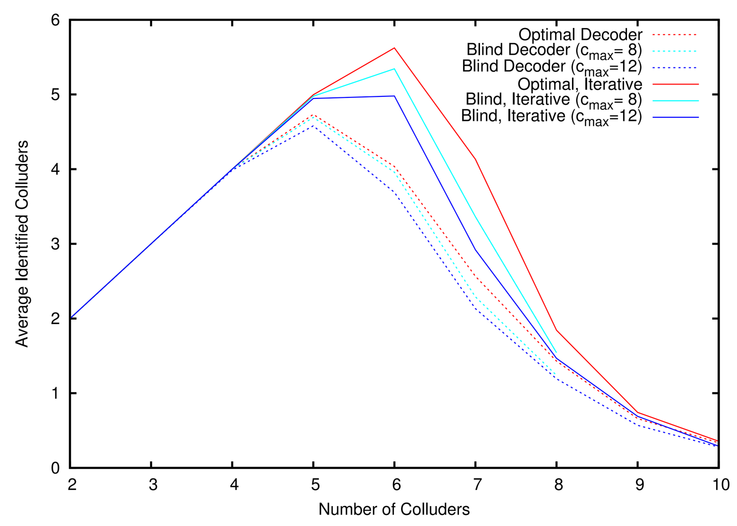
Collusion Model Estimation

For an estimated collusion size $\hat{c}$, the collusion process $\boldsymbol{\theta}$ can be estimated from the observation of $\mathbf{y}$:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in [0,1]^{\hat{c}+1} \ s.t. \ \theta(0)=0, \theta(\hat{c})=1}{\arg\max} \log \mathbb{P}(\mathbf{y}|\mathbf{p}, \boldsymbol{\theta}) \quad (6)$$

with $\mathbb{P}(\mathbf{y}|\mathbf{p}, \boldsymbol{\theta}) = \prod_{i=1}^{m} \mathbb{P}(y(i)|p(i))$.

Due to lack of identifiability, one cannot estimate c, but only $\hat{\boldsymbol{\theta}}$ for a given $\hat{c}$. We impose $\hat{c} = c_{\mathsf{max}}$ (performance degradation is illustrated below).



Identified traitors (*interleaving* collusion, $n = 10^5$, $m = 2048$, $P_{\mathsf{fp}} = 10^{-4}$); optimal and blind decoders with different c_{max}.

Fast Score Computation

For a large number of users, score computation is limited by memory bandwidth. Two speedup techniques can be used:

Weight precomputation Computation of an individual's score s_j can be written as $s_j = \sum_{i=1}^{m} W[x_j(i)](i)$ where $\mathbf{W}$ is a $2 \times m$ matrix containing the precomputed log-likelihood ratios:

$$W[0](i) = \log \frac{\mathbb{P}(y(i)|p(i))}{\mathbb{P}(y(i)|0, p(i))},$$

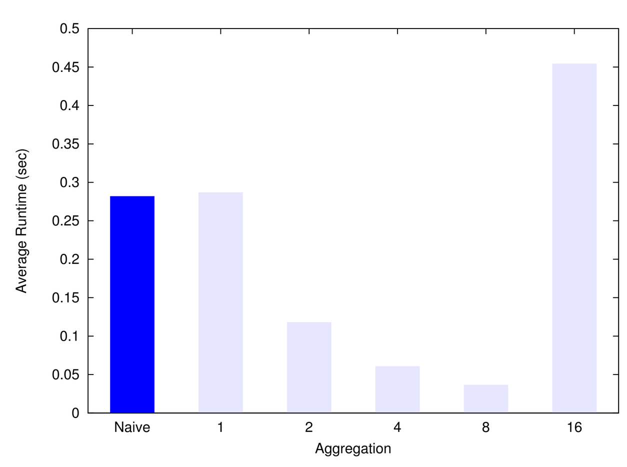
$$W[1](i) = \log \frac{\mathbb{P}(y(i)|p(i))}{\mathbb{P}(y(i)|1, p(i))}.$$

Aggregation b bits are grouped together into an unsigned integer data type native to the processor, e.g. $b = 32$. Chunks of $a \le b$ bits, e.g. $a = 8$, can be processed in parallel using a table lookup. The weight matrix $\mathbf{W}$ is turned into an aggregated weight matrix $\mathbf{W}'$ of size $2^a \times \lceil m/a \rceil$ with elements

$$W'[q](i') := \sum_{l=1}^{a} W[\mathrm{bit}(q,l)](a(i'-1)+l) \quad (7)$$

where $1 \le i' \le \lceil m/a \rceil$, $q \in \{0,1\}^a$, and $\mathrm{bit}(q,l)$ denotes the l-th bit of value q.

Best performance is obtained for $a = 8$ according to experiments.



Score computation ($n = 10^5$, $m = 2048$) with aggregation a on Intel Core2 (2.6 GHz). *Naive* stores a codeword bit in a byte.

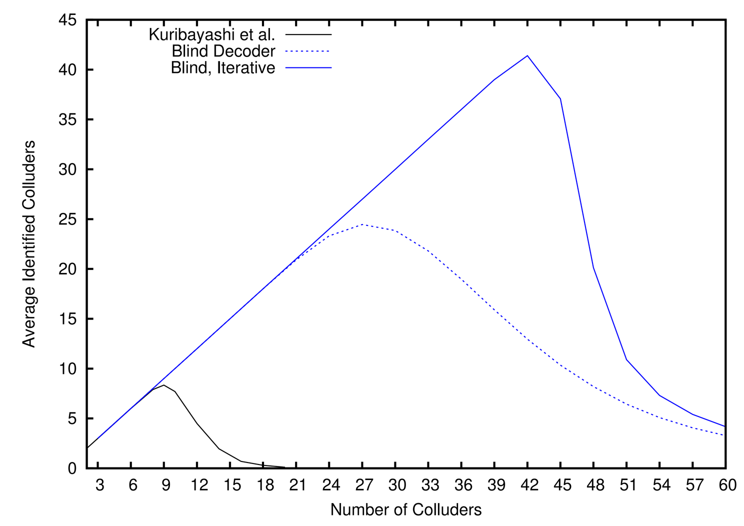
Source Code (C++)

Available at `http://www.irisa.fr/texmex/people/furon/src.html`.

Funded by French national project MEDIEVALS ANR-07-AM-005.

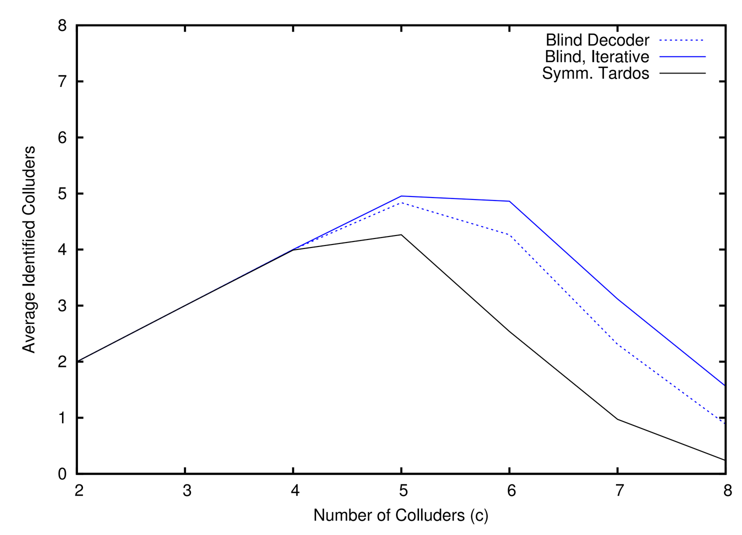
Decoding Results and Comparison

Kuribayashi setup [4] $n = 10\,000$ users, code length $m = 10\,000$, $P_{\mathsf{fp}} = 10^{-4}$, *majority voting* collusion



Identified colluders ($c \in \{2, 3, 6, \dots, 60\}$, $c_{\mathsf{max}} = 80$).

Jourdas & Moulin setup [5] $n = 33\,554\,432$ users, code length $m = 7\,440$, $P_{\mathsf{fp}} = 10^{-3}$, *interleaving* collusion and AWGN ($\sigma^2 = 1$)



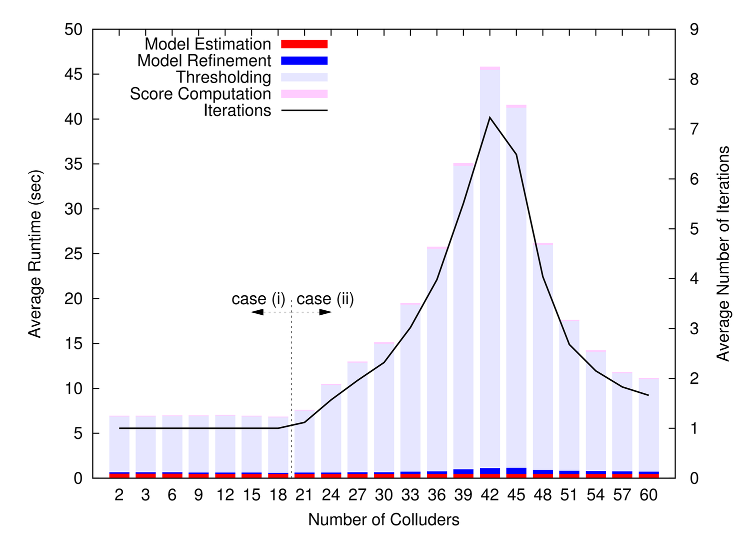
Identified colluders ($c \in \{2, \dots, 8\}$, $c_{\mathsf{max}} = 8$).

For $c = 5$, Jourdas & Moulin only accuse one colluders with $P_{\mathsf{fn}} = 0.004$. The proposed iterative decoder accuses 4.95 colluders, $P_{\mathsf{fn}} = 0.0016$. We also compare against a symmetric Tardos decoder.

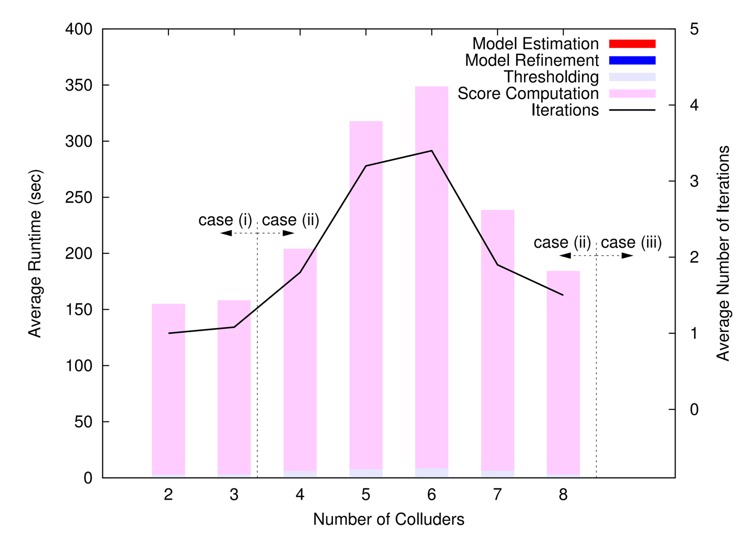
Runtime Results

We analzye the runtime of the decoder's components (model estimation, thresholding, score computation) on a single core of an Intel Core2 CPU (2.6 GHz) and plot the average number of iterations.

Kuribayashi setup



Jourdas & Moulin setup



References

[1] G. Tardos, "Optimal probabilistic fingerprint codes," in *Proc. 35th ACM Symp. on Theory of Computing*, San Diego, CA, USA, 2003, pp. 116–125.

[2] L. Pérez-Freire and T. Furon, "Blind decoder for binary probabilistic traitor tracing codes," in *Proc. First IEEE Int. Workshop on Information Forensics and Security*, London, UK, Dec. 2009, pp. 56–60.

[3] F. Cérou, T. Furon, and A. Guyader, "Experimental assessment of the reliability for watermarking and fingerprinting schemes," *EURASIP Jounal on Information Security*, 2008, ID 414962, 12 pages.

[4] M. Kuribayashi, N. Akashi, and M. Morii, "On the systematic generation of Tardos fingerprinting codes," in *Proc. IEEE Workshop on Multimedia Signal Processing, MMSP '08*, Cairns, Australia, Oct. 2008, pp. 748–753.

[5] J.-F. Jourdas and P. Moulin, "High-rate random-like spherical fingerprinting codes with linear decoding complexity," *IEEE Trans. on Information Forensics and Security*, vol. 4, no. 4, pp. 768–780, Dec. 2009.