TSALockMark: An Asymmetric and Robust Watermarking Scheme for Relational Databases with Distortion Constraints

Ke Yang[1,2,3], Shuguang Yuan[1,2,3](✉), Jing Yu[1,2,3], Zhaochen Li[1,2,3], Yuyang Wang[1,2,3], and Chi Chen[1,2,3]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{yangke,yuanshuguang,yujing,lizhaochen,wangyuyang,chenchi}@iie.ac.cn
[2] State Key Laboratory of Cyberspace Security Defense, Beijing, China
[3] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. Relational databases are crucial for data publication and dissemination. Database watermarking techniques are applied for copyright protection. However, (i) Key exposure of existing watermarking schemes compromises security by making watermarks visible and vulnerable to removal. (ii) The watermarking scheme frameworks lack the flexibility for multi-party scenarios, restricting their ability to adapt watermark information for diverse tasks. In this paper, we present an asymmetric and robust watermarking scheme called TSALockMark. It supports a many-to-many configuration to allocate multiple keys, facilitates multi-party watermarking detection, supports ownership authentication and approval dissemination, and resists key exposure and malicious attacks. Moreover, we introduce the number control of the least significant bits and particle swarm optimization algorithm to preserve semantics and statistics. Experiments on three real-world datasets demonstrate the proposed scheme's functionality, robustness, and performance, offering effective solutions to data dissemination challenges.

Keywords: TSALockMark · Relational databases · Asymmetric Watermarking · Distortion Constraints

1 Introduction

Relational databases are ubiquitous in industries owing to their structured data organization, reliability, and support for complex queries. Watermarking techniques provide a solution for copyright protection of relational databases, embedding watermark information by selecting and modifying data. During detection, copyright is verified when the same watermarks are extracted from the data.

Agrawal and Kiernan's seminal work [1] introduced a robust watermarking scheme for relational databases, inspiring subsequent research that has aimed to make watermarks meaningful [16, 18] and balance robustness with data distortion [15, 7, 6, 9, 8, 3, 11, 13, 12, 14, 5]. These approaches are symmetric, using the same key for both embedding and detection. Security requires the secret key only

to be held by the data owner, limiting public ownership verification. Nevertheless, asymmetric watermarking, which supports public verification, has received less attention in relational databases. Notable efforts in this direction include [2] and [4], where Cui *et al.* [2] partition the watermark into private and public components by a symmetric key, and Raju and Agostino [4] apply a public key to generate a watermark for ownership declaration and a private key to embed a watermark for copyright protection. However, there still exist key challenges:

Security Threat on Key Exposure. Whether symmetric or asymmetric schemes, the key and the watermark are one-to-one. Once the key is exposed or public, the watermark becomes visible and vulnerable. Hence, the adversary can easily find and erase the watermark, compromising the scheme's security.

Limited Capabilities for Multi-party Scenarios. Existing frameworks of all watermarking schemes are fixed and lack extensibility; they may involve only a data sender and receiver claiming a single watermark. These schemes cannot dynamically add participants (e.g., regulators) or adjust watermark information for multiple tasks, such as authoritative ownership verification or identifying unapproved content dissemination via a trusted third party.

In this paper, we introduce TSALockMark, an asymmetric and robust watermarking scheme inspired by TSA locks. As illustrated in Fig. 1a, TSA locks enable customs to inspect luggage with universal 'master' keys while users retain their own keys. TSALockMarker supports a many-to-many configuration, allocating multiple keys for multi-party detection. Notably, exposed keys do not result in watermark removal due to asymmetry. The scheme framework is extensible for multiple tasks. It can be extended to protect copyright while providing multi-party/public ownership verification, and to enable regulators to authenticate ownership or approve data dissemination. These features are unique compared to other solutions. Furthermore, we introduce distortion constraints to maintain semantics and statistics. Experimental results on real-world datasets confirm the capability of our scheme. Comparative experiments with AHK [1], S^2R^2W [13], and SCPW [14] reveal that our method remains robust despite deletion of 8 out of 10 attribute columns and removal of 90% of rows. Evaluations of overhead and watermark capacity further demonstrate the strong performance of our scheme. The contributions of this paper are as follows:

- We propose a multi-key asymmetric watermarking scheme for copyright protection and multi-party detection, which shows practicality and extensibility.
- We apply semantic analysis and the particle swarm optimization algorithm to modulate the least significant bits to preserve statistics and semantics.
- We evaluate our scheme on three real-world datasets to assess its functionality, robustness, and performance.

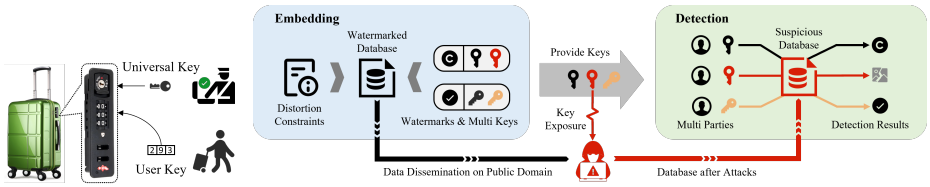
The rest of this paper is organized as follows: The scheme framework is outlined in Section 2, and the algorithms are detailed in Section 3. Section 4 presents the experiments. Finally, section 5 concludes our work.

2 Scheme Framework

The watermarking framework consists of embedding and detection, and a simplified example is shown in Fig. 1b. Given a relational database D, an *embedding party* employs the embedding algorithm to insert a watermark W using each key from a key set $K = \{k_0, \ldots, k_{\epsilon-1}\}$. Multiple watermarks can be embedded with different key sets. To preserve data usability, distortion constraints are enforced during embedding. This process produces the watermarked database D_w.

The framework is asymmetric to support both private and public detection. Accordingly, K is divided into distributable keys K_p for multi-party/public detection and non-distributable keys K_s for private detection. During dissemination, D_w may be updated or attacked (e.g., subset and attribute attacks), and keys in K_p may be exposed, leading to watermark removal. Moreover, malicious parties may collaborate by sharing their keys to remove the watermark.

The detection algorithm extracts W' from a suspicious database D' by K_p or K_s. Our scheme can tolerate leakage of keys in K_p because part of the watermark remains concealed via K_s, ensuring security. Notably, the watermark of any exposed key is independent of the watermark embedded by other keys.



(a) (b)

Fig. 1: (a) TSA Lock. (b) A simplified watermarking framework. It begins with embedding, watermarks © and ✓ are embedded into a database under constraints using their corresponding keys. 🔑 remains private, while other keys are provided to multiple parties. 🔑 is exposed, and attackers remove the corresponding ©. During detection, 🔑 holder cannot detect ©, while 🔑 holder can, meaning that 🔑 exposure does not affect 🔑 detection. 🔑 holders can detect ✓.

3 TSALockMark

3.1 Watermark Embedding

Consider a relational database $D = \{PK, A_1, A_2, \ldots, A_n\}$ with η rows and n numerical attributes, where PK is the primary key. The pseudo-code for watermark embedding is provided in Algorithm 1. Each key k in K is independently used to select *target values* from attributes for embedding the watermark W. In lines 2-3, rows and target values are selected by the embedding density γ and key

k, indicating that a key can theoretically embed $\frac{\eta}{\gamma}$ watermark bits into D. $H(\cdot)$ is a hash function, $\circ$ is a concatenation operator. In lines 6-8, the watermark bit is selected by the watermark length ζ and embedded into the least significant bits (LSBs) of the target values. The number of LSBs ξ is under semantic control in line 5, and the watermarked target values comply with the statistical constraint in line 9. The details are in Section 3.3. Additionally, we employ the attribute classifier from [17] to maintain attribute indices, preventing attribute name attacks, which distort the link between attribute names and values. Finally, the algorithm outputs the watermarked database D_w.

Algorithm 1 Watermark Embedding

Input: D, K, W, γ, ζ
Output: D_w
1: **for** each row r in D and each key k in K **do**
2: **if** $H(r.PK \circ k) \mod \gamma == 0$ **then**
3: The index of target value t: $t_{idx} = H(r.PK \circ k) \mod n$
4: If t is an floating-point, use the decimal part of t for embedding.
5: Gain the number of least significant bits ξ according to Section 3.3.
6: The index of watermark bit: $w_{idx} = H(r.PK \circ k) \mod \zeta$
7: Determine the bit b of t to be embedded: $b_{idx} = H(r.PK \circ k) \mod \xi$
8: Replace b with the watermark bit which index is w_{idx}.
9: Preserve statistics by PSO in Section 3.3.
10: Apply an Attribute Classifier to maintain attribute indices.
11: **return** D_w

Our algorithm theoretically permits multi-watermark embedding using different key sets. It assumes that any watermark overlap caused by multiple embeddings is negligible, ensuring the embedded watermarks do not interfere with each other. Thus, the total number of embedded bits of D approaches $\frac{\eta \times \epsilon}{\gamma}$, where ϵ is the number of keys. To verify this assumption, we used the datasets from Section 4 for watermark embedding and calculated the overlapping rate, as shown in Table 1. It can be observed that the overlap rate from multiple embeddings is minimal. During detection, errors in recovered bits can be effectively corrected using a majority voting mechanism. Moreover, the watermark overlap rate depends solely on the total number of embedding bits.

Table 1: Overlapping Rate of Watermark Embedding.

ϵ	γ	Total Number of Embedded Bits			Overlap Rate		
		CRD	FCT	UMD	CRD	FCT	UMD
10	997	267	5809	9993	0.00%	0.03%	0.02%
20	1999	304	5870	10037	0.00%	0.00%	0.01%
20	997	564	11706	19989	0.10%	0.08%	0.09%

3.2 Watermark Detection

Since each key in K independently embeds watermark bits, providing parts of keys from K suffices for detection. The scheme is asymmetrical to guard against key exposure or collusion attacks by dividing keys into non-distributable K_s and distributable K_p. K_s is for private detection and must remain confidential, while K_p is shared for multiple parties or public detection. For multi-watermark detection, when a party wishes to verify a watermark in a suspicious database D', they request detection keys K_d. The embedding party provides exclusive K_d, chosen randomly from K_p based on the party's identity and detection needs. The embedding party can carry out private detection by using K_s or K.

Watermark detection is the reverse of embedding with a majority voting correction mechanism. Each detection key extracts the watermark bit independently. Major voting allocates two arrays, ZERO and ONE, matching the watermark's length, to tally 0 and 1 detections at each position. After detection, it compares these tallies and selects the most frequent bit at each position, forming the final watermark by consensus.

3.3 Distortion Constraints

Semantic and statistical distortion constraints are applied to maintain data usability. For brevity and clarity, we denote an attribute column as A for illustration, which contains η values. The target values of D are represented by $DT = \{T_0, T_1, \ldots, T_n\}$, with $T = \{t_0, t_1, \ldots, t_{k-1}\}$ for attribute A. We denote the marked attribute as A^m and marked target values of A^m as T^m. The semantic tolerance scope $Se(\cdot)$ and statistics $St(\cdot)$ of D and D_w are adopted as $Se(D)$, $St(D)$, $Se(D_w)$, and $St(D_w)$, respectively. The distortion constraints of our scheme can be represented by $Se(D) = Se(D_w)$ and $St(D) = St(D_w)$.

Number Control of Least Significant Bits Based on Semantic. We introduce a semantic-based method to control the number of LSBs ξ in embedding so that data semantics remain unchanged. Initially, a database D semantics are analyzed by the work in [7, 4], and tools like knowledge graphs or large language models. These methods aid in creating a semantic tolerance scope $Se(D) = \{Se(T_0), \ldots, Se(T_n)\}$ for the target values of attributes. Fig. 2 depicts the semantic tolerance scope for T, assuming that T represents target values in the 'Age' attribute, where 'Age' is directly correlated with 'Educational Level'.

PK	Educational Level	Age	…
102	High School	15	
237	High School	17	
…	…	…	
869	Undergraduate	26	
992	Undergraduate	18	

PK	*T*(Age)	Semantic Tolerance Scope
102	15	[15,18)
237	17	
…	…	…
869	26	[18,30)
992	18	

Fig. 2: Example of semantic analysis on 'Age' attribute.

To keep watermarking within semantic bounds and reduce distortion, ξ_s and ξ_f are introduced. We first adjust ξ_s of a target value $t \in T$ with a semantic tolerance $[\alpha, \beta)$. We define the bound $[t_{min}, t_{max}]$ of t after watermarking:

$$t_{min}(\xi_s) = t \wedge (\neg(2^{\xi_s} - 1)), \quad t_{max}(\xi_s) = t \vee (2^{\xi_s} - 1),$$

ensuring the bounds falls within α and β. Hence, ξ_s maintains semantic integrity. We then control ξ_f by a Fraction of the Least Significant Bits (LSBF), where $\xi_f = \text{ROUND}(\text{LSBF} \times BL_t)$. Finally, the ultimate control $\xi = \min(\xi_s, \xi_f)$.

Least Significant Bits Modulation for Statistics Preservation. Given a database D, we consider statistics $St(\cdot)$ includes μ and σ^2, and $St(D) = \{St(A_1), \ldots, St(A_n)\}$. Particle Swarm Optimization (PSO) Algorithm [10] is applied to modulate LSBs (except the embedded bits) of marked values, preserving $St(D)$. PSO is population-based, where a swarm of "particles" optimizes a fitness function $f(\cdot)$ (*fitness value*) through their movements (iterative *positions* and *velocities*) in solution space to find the optimal solution. Position and velocity are influenced by the particle's *personal* and *global bests* (*pbest* and *gbest*).

We formulate LSBs modulation as a constrained PSO problem. For any attribute A, to preserve $St(A) = St(A^m)$, T^m are iteratively modulated to T' by adjusting LSBs until $St(A) = St(A')$, while keeping embedded bits intact (denoted as $T^m_w = T'_w$). Thus, the fitness function $f(T')$ can be constructed as:

$$f(T') = W_\mu \times MD + W_\sigma \times VD, where\ MD = |\mu_A - \mu_{A'}|, VD = |\sigma^2_A - \sigma^2_{A'}|,$$

where W_μ and W_σ are weights assigned to control the optimization focus in the optimization process, MD and VD indicate mean and variance differences.

Let $DT' = \{T'_0, T'_1, \ldots, T'_n\}$ represent modulated target values, and let τ be a threshold near zero. The objective can be formulated as:

$$\forall T' \in DT', f(T') \leq \tau, \quad subjected\ to\ T^m_w = T'_w.$$

We define LSBs of T^m as $T_{LSB} = \{t_{L1}, t_{L2}, \ldots, t_{Lk}\}$ as a particle, with each element represented in binary form and modulated under objective constraints. Since t_{Li} must be an integer, its velocity component in PSO must match its binary length. In position updates, the addition should be binary and skip embedded bits. The algorithm stops upon finding the optimal T_{LSB} for T^m. Final marked target values replace their LSBs with those in T_{LSB}, optimizing post-embedding statistics. The pseudo-code is shown in Algorithm 2.

4 Experiments

Datasets. Experiments were performed on real-world datasets: CRD[4], FCT[5], and UMD[6]. We selected seven numerical attributes from CRD, the first ten from FCT, and ten from UMD for watermarking.

[4] https://www.kaggle.com/datasets/laotse/credit-risk-dataset
[5] https://www.kaggle.com/datasets/uciml/forest-cover-type-dataset
[6] https://www.kaggle.com/datasets/arnavsmayan/urban-mobility-dataset

Algorithm 2 Least Significant Bits Modulation

Input: T, T_w^m, $f(\cdot)$
Output: Modulated least significant bits T_{LSB}
1: for each attribute that needs to preserve statistics performs the following steps.
2: **for** $iter = 1$ to $iter_{max}$ **do**
3: **for** each particle i in p **do**
4: Update velocities and positions, compute the fitness value of i by $f(\cdot)$.
5: Compare the fitness value with *pbest* and *gbest* and update *pbest* and *gbest*.
6: **return** T_{LSB} until stopping condition is met
7: **return** T_{LSB}

Benchmarks. In comparative experiments, We evaluated the proposed scheme against AHK [1], S^2R^2W [13], and SCPW [14].
Settings. Default parameters were set as LSBF = 0.5, $\epsilon = 10$, and $\gamma = 61/1163/1999$ for CRD, FCT, UMD. This reflected approximately 5,000 watermarked rows per dataset. Benchmarks were aligned by adjusting γ. Tests were executed on a 1.9GHz CPU with 16GB RAM, under Windows 11.
Metrics. We evaluated availability, robustness, and performance. Availability measures changes in semantic tolerance scope, mean (MD: $|\mu_A - \mu_{A^m}|$), variance (VD: $|\sigma_A^2 - \sigma_{A^m}^2|$), and MAE ($MAE := \frac{1}{\eta}\sum_{i=1}^{\eta}|v_i - v_i^m|$). Robustness evaluates Correct Factor (CF) against attacks: CF $= \frac{1}{\zeta}\sum_{i=0}^{\zeta-1} w[i] \oplus w_r[i] \times 100\%$. Performance was assessed through watermarking overhead and capacity (WC).

4.1 Functionality Verification

Embedding and Detection We simulated multi-party detection by using 16 $\times$ 16 pixels "copyright" and "passed" binary image as W_0 and W_1, respectively. K_0 and K_1 were generated, each with 10 secret keys, for embedding W_0 and W_1 into experimental datasets. Detection was performed using keys K_{d0} and K_{d1}, each with 5 keys. Results in Fig. 6 show that K_0 and K_1 successfully retrieve the corresponding watermark without cross-interference, confirming the independence of multi-key systems.

We also evaluated scenarios of detection key exposure, where attackers leverage exposed keys to remove watermarks. For keys K_0 and K_1, the exposed key numbers ke were 7, 8, and 9, while non-distributable keys K_{s0} and K_{s1} for detection remained at 3, 2, and 1 keys, respectively. Fig. 7 shows the results. Any undetected watermark pixels are highlighted in red. Remarkably, the algorithm still detects over half the watermarks even when 9 out of 10 keys are exposed. Less exposure results in better recovery, achieving full recovery when $ke = 7$. Thus, key exposure does not fully nullify watermark detection.

Distortion Constraints To verify semantic constraints' effectiveness, we analyzed semantics and determined the semantic tolerance scope on CRD as an illustration. Watermarks were embedded with and without semantic control on

ξ to assess whether the semantic tolerance scope changes after embedding. Fig. 3 shows an example of embedding on *person_ age* attribute of CRD.

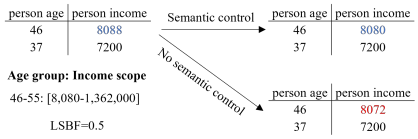


Fig. 3: An example of watermarking with and without semantic control.

Fig. 4: Statistics Evaluation.

Method	S^2R^2W	SCPW	Proposal
MD	0.000106	0.000002	0.000216
VD	0.085223	0.009356	0.000003
MAE	0.002409	0.000112	0.001495

From Fig. 3, *person_ age* relates to *person_ income*, displaying a semantic tolerance scope for income at ages 46-55 as [8080,1362000]. For the entity row (46,8088), $\xi = 6$ without semantic control leads to an embedded value of 8072, violating semantic scope. With semantic control, $\xi = 5$, and the embedded value is 8080, confirming that our scheme preserves semantic tolerance scope.

To verify statistics constraints' effectiveness, we used the S^2R^2W and SCPW, known for retaining statistics, as benchmarks. We embedded watermarks on UMD and calculated the MD, VD, and MAE of the *traffic_ flow* attribute. Fig. 4 illustrates our scheme's comparable performance in minimizing statistical distortion, with MD and VD variations in 10^{-4} and 10^{-6} orders, respectively.

4.2 Robustness Verification

To assess the robustness of our algorithm, we conducted experiments under harsh conditions and evaluated against benchmarks using the correlation factor (CF) across various attack levels. These experiments on UMD utilized 8 out of 10 detection keys from K. Fig. 5 presents results for attacks involving subset deletion, alteration, insertion, and attribute deletion.

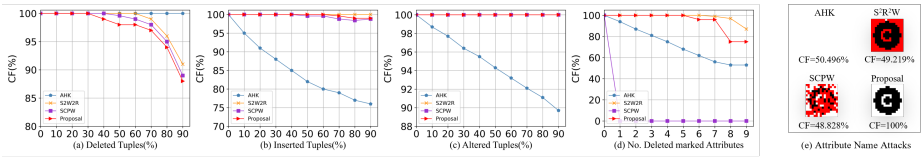


Fig. 5: Results of Subset and Attribute Attacks.

As evidenced by the results, the proposed scheme exhibits resilience to attacks comparable to S^2R^2W and SCPW, outperforming AHK. In Fig. 5 (a) , the CF of our algorithm remains above 85% even with 90% of tuple deletions. AHK maintains 100% CF due to its meaningless watermark. In subset alteration and

insertion scenarios (Fig. 5 (b) and (c)), our proposal is only minimally affected. During attribute deletion attacks (Fig. 5(d)), our scheme performs slightly below S^2R^2W but surpasses AHK. SCPW's CF falls to 0 as its detection depends on deletable attributes X, and our experiments set $|X| = 1$ for SCPW.

Fig. 5 (e) shows that attribute name attacks [17] impede AHK, S^2R^2W, and SCPW, as they depend on the correct attribute order. In contrast, our algorithm can withstand such attacks facilitated by the attribute classifier.

4.3 Performance Evaluation

Table 8 details schemes' overhead on UMD and their watermark capacity (WC) with fixed γ, η, and n. kb denotes the k-bits watermark embedded into data in SCPW. Our scheme possesses moderate embedding overhead (EO) due to PSO, yet it outperforms SCPW, which applies genetic algorithm. Our detection overhead (DO) is low, as PSO is not required for detection. Notably, multi-key embedding is independent, facilitating the implementation of parallelism to reduce the overhead. From the table, the proposed scheme provides sufficient watermark capacity to carry the watermark information.

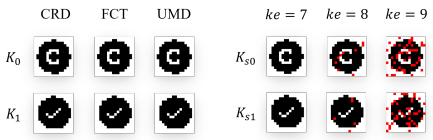


Fig. 6: Watermark Detection Results.

Fig. 7: Results of Key Exposure.

Fig. 8: Overhead and Capacity Comparison with the Benchmarks.

Method	AHK	S^2R^2W	SCPW	Proposal
EO (s)	24.535	101.666	676.753	362.254
DO (s)	10.017	50.934	2.275	11.747
WC	$\frac{\eta}{\gamma}$	$\frac{\eta \times n}{\gamma}$	$\frac{\eta \times kb}{\gamma}$	$\frac{\eta \times \epsilon}{\gamma}$

5 Conclusion

This study presents a multi-key, multi-watermark asymmetric watermarking scheme addressing the security threat caused by key exposure. The scheme's extensible framework enhances its practicality in complex scenarios. Experiments on real-world datasets and comparisons with previous work demonstrate the effectiveness of our scheme, showcasing excellent robustness against subset and attribute attacks and achieving strong performance metrics.

Acknowledgments. This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences, Grant No. XDB0690303.

References

1. Agrawal, R., Kiernan, J.: Watermarking relational databases. In: VLDB'02: Proceedings of the 28th International Conference on Very Large Databases. pp. 155–166 (2002)

2. Cui, H., Cui, X.: A public key cryptography based algorithm for watermarking relational databases. In: 2008 International Conference on Intelligent Information Hiding and Multimedia Signal Processing. pp. 1344–1347. IEEE (2008)
3. Franco-Contreras, J., Coatrieux, G.: Robust watermarking of relational databases with ontology-guided distortion control. IEEE transactions on information forensics and security **10**(9), 1939–1952 (2015)
4. Halder, R., Cortesi, A.: A persistent public watermarking of relational databases. In: Information Systems Security. pp. 216–230 (2010)
5. Hu, D., Zhao, D., Zheng, S.: A new robust approach for reversible database watermarking with distortion control. IEEE Transactions on Knowledge and Data Engineering **31**(6), 1024–1037 (2018)
6. Jawad, K., Khan, A.: Genetic algorithm and difference expansion based reversible watermarking for relational databases. Journal of Systems and Software **86**(11), 2742–2753 (2013)
7. Kamran, M., Farooq, M.: An information-preserving watermarking scheme for right protection of emr systems. IEEE Transactions on Knowledge and Data Engineering **24**(11), 1950–1962 (2011)
8. Kamran, M., Farooq, M.: A formal usability constraints model for watermarking of outsourced datasets. IEEE transactions on information forensics and security **8**(6), 1061–1072 (2013)
9. Kamran, M., Suhail, S., Farooq, M.: A robust, distortion minimizing technique for watermarking relational databases using once-for-all usability constraints. IEEE Transactions on Knowledge and Data Engineering **25**(12), 2694–2707 (2013)
10. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN'95 - International Conference on Neural Networks. vol. 4, pp. 1942–1948 vol.4 (1995)
11. Khanduja, V., Verma, O.P., Chakraverty, S.: Watermarking relational databases using bacterial foraging algorithm. Multimedia Tools and Applications **74**, 813–839 (2015)
12. Li, Q., Wang, X., Pei, Q., Lam, K.Y., Zhang, N., Dong, M., Leung, V.C.: Database watermarking algorithm based on decision tree shift correction. IEEE Internet of Things Journal **9**(23), 24373–24387 (2022)
13. Li, W., Li, N., Yan, J., Zhang, Z., Yu, P., Long, G.: Secure and high-quality watermarking algorithms for relational database based on semantic. IEEE Transactions on Knowledge and Data Engineering (2023)
14. Ren, Z., Fang, H., Zhang, J., Ma, Z., Lin, R., Zhang, W., Yu, N.: A robust database watermarking scheme that preserves statistical characteristics. IEEE Transactions on Knowledge and Data Engineering **36**(6), 2329–2342 (2024)
15. Shehab, M., Bertino, E., Ghafoor, A.: Watermarking relational databases using optimization-based techniques. IEEE transactions on Knowledge and Data Engineering **20**(1), 116–129 (2007)
16. Wang, H., Cui, X., Cao, Z.: A speech based algorithm for watermarking relational databases. In: 2008 International Symposiums on Information Processing. pp. 603–606. IEEE (2008)
17. Yang, K., Yuan, S., Yu, J., Wang, Y., Yang, T., Chen, C.: Don't abandon the primary key: A high-synchronization and robust virtual primary key scheme for watermarking relational databases. In: International Conference on Information and Communications Security. pp. 289–309. Springer (2024)
18. Zhou, X., Huang, M., Peng, Z.: An additive-attack-proof watermarking mechanism for databases' copyrights protection using image. In: Proceedings of the 2007 ACM symposium on Applied computing. pp. 254–258 (2007)