Distributed Memetic Algorithm for Outsourced Database Fragmentation

Distributed Memetic Algorithm for Outsourced Database Fragmentation

Yong-Feng Ge, *Graduate Student Member, IEEE*, Wei-Jie Yu, *Member, IEEE*, Jinli Cao,
Hua Wang, *Member, IEEE*, Zhi-Hui Zhan, *Senior Member, IEEE*,
Yanchun Zhang, *Member, IEEE*, and Jun Zhang, *Fellow, IEEE*

***Abstract*—Data privacy and utility are two essential requirements in outsourced data storage. Traditional techniques for sensitive data protection, such as data encryption, affect the efficiency of data query and evaluation. By splitting attributes of sensitive associations, database fragmentation techniques can help protect data privacy and improve data utility. In this article, a distributed memetic algorithm (DMA) is proposed for enhancing database privacy and utility. A balanced best random distributed framework is designed to achieve high optimization efficiency. In order to enhance global search, a dynamic grouping recombination operator is proposed to aggregate and utilize evolutionary elements; two mutation operators, namely, merge and split, are designed to help arrange and create evolutionary elements; a two-dimension selection approach is designed based on the priority of privacy and utility. Furthermore, a splicing-driven local search strategy is embedded to introduce rare utility elements without violating constraints. Extensive experiments are carried out to verify the performance of the proposed DMA. Furthermore, the effectiveness of the proposed distributed framework and novel operators is verified.**

***Index Terms*—Database fragmentation, database privacy and utility, distributed memetic algorithm (DMA).**

I. Introduction

With the development of cloud computing and storage, outsourced data storage has shown its commercial advantages at low cost and high stability. For now, in the area of outsourced data storage, the confidentiality of sensitive data is still a big concern [1]–[3]. According to [4], data privacy protection and security are two primary inhibitors when choosing cloud data service. To tackle this challenging issue, many researchers have contributed from various angles [5], [6]. A classical and direct solution for database privacy issues is adopting data encryption [7]. Each value in the original database is transferred into another version in the encryption data. Database encryption can protect the information in every single record from leaking. Although encryption can help solve the concern of data privacy, data decryption is time consuming. The efficiency of data query and evaluation is affected accordingly [8], [9], which is also essential in data outsourcing service. To be specific, the time complexity of database encryption is $O(nm)$, where n and m represent the number of attributes and records in the database [7]. When making queries, the time complexity of database decryption is $O(nm)$ [10].

Database fragmentation [11], [12] is another solution for data outsourcing service, which can help the provider to achieve data privacy as well as maintain query efficiency. Database fragmentation is a technique that splits the entire database into multiple fragments [9], [13]. Thus, the exposure of any single fragment does not lead to a violation of privacy. To be specific, database privacy requirements can be represented by a set of sensitive associations between attributes. In database fragmentation, attributes of sensitive associations can be divided into different fragments and the access to different fragments is controlled. Database privacy is protected by fragments that satisfy all the privacy requirements. From this angle, various database fragmentation approaches have been proposed [14], [15]. The time complexity of database fragmentation is $O(n^2)$, where n indicates the number of attributes in the database [16]. Privacy protection of database encryption is based on every single record, while the granularity of database fragmentation is each attribute. In addition, fragmentation does not need a transformation operation when making queries or evaluations [16]. Query processing models of database fragmentation can be divided into two categories. In the first category [17], collusion between fragments is forbidden, which means fragments cannot join each other. In contrast, the

Manuscript received February 21, 2020; revised July 13, 2020; accepted September 17, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFB2102102; in part by the Outstanding Youth Science Foundation under Grant 61822602; in part by the National Natural Science Foundations of China under Grant 61772207 and Grant 61873097; in part by the Key-Area Research and Development of Guangdong Province under Grant 2020B010166002; in part by the Guangdong Natural Science Foundation Research Team under Grant 2018B030312003; and in part by the Fundamental Research Funds for the Central Universities, Sun Yat-sen University, under Grant 19wkpy149. This article was recommended by Associate Editor H. Ishibuchi. *(Corresponding authors: Wei-Jie Yu; Zhi-Hui Zhan.)*

Yong-Feng Ge and Zhi-Hui Zhan are with the School of Computer Science and Engineering and the Guangdong Provincial Key Laboratory of Computational Intelligence and Cyberspace Information, South China University of Technology, Guangzhou 510006, China (e-mail: zhanapollo@163.com).

Wei-Jie Yu is with the School of Information Management, Sun Yat-sen University, Guangzhou 510275, China (e-mail: ywj21c@163.com).

Jinli Cao is with the Department of Computer Science and Information Technology, La Trobe University, Melbourne, VIC 3083, Australia.

Hua Wang, Yanchun Zhang, and Jun Zhang are with the Institute for Sustainable Industries and Liveable Cities, University of Victoria, Melbourne, VIC 3011, Australia.

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2020.3027962

second category of the query processing model [12] supports the collusion between fragments, and the access authority of each user is controlled by trusted query mediators. To maintain the advantage of database fragmentation in query and evaluation efficiency, attributes of high evaluation and query frequencies should be assigned to the same fragment [18]. Overall, the goal of database fragmentation is to identify a solution satisfying all the given confidentiality constraints while achieving the highest evaluation and query efficiency, which can also be regarded as optimizing database privacy and utility.

To tackle the database fragmentation problem, traditional approaches can be divided into two categories, that is: 1) enumeration approaches and 2) greedy approaches. To achieve database fragmentation, enumeration approaches, such as graph search [12] and fragmentation tree search [19], were proposed. These approaches cannot effectively solve database fragmentation problems with a higher numbers of attributes. To improve the efficiency in database fragmentation, greedy strategies [16] were designed. Due to the limitation of searchability, the performance of the proposed approaches on database fragmentation with complex privacy and utility requirements cannot reach an ideal level.

Ciriani *et al.* [16] proved that the database fragmentation problem is NP-hard. When facing complex optimization problems, traditional approaches are very likely to lose effectiveness. Evolutionary algorithms (EAs) have been utilized in solving NP-hard optimization problems and have shown advantages of high efficiency and robustness [20]–[23]. In the last few decades, various kinds of EAs have been proposed, including genetic algorithms (GAs) [20], [24] and memetic algorithms (MAs) [25]–[27]. For now, GAs [28], [29] have been adopted in database fragmentation. In [28], a GA was introduced to satisfy database privacy. In [29], a benefit-driven GA was designed to help balance privacy and utility issues in database fragmentation. These studies reveal that EAs are effective in database fragmentation. The performance of these approaches can be enhanced from two directions. The performance of GAs is restricted by low exploitative ability. As one of the recently growing areas in evolutionary computation, MAs emphasize the balance between exploration and exploitation. Through combining population-based global search and meme-based local refinement, MAs have achieved success in various domains, such as minimum vertex cover [30], automatic data clustering [31], and large-scale integration floor planning [32]. In addition, the efficiency of these approaches is restricted by the serial model. The distributed framework for EAs can effectively improve optimization efficiency.

To enhance algorithmic efficiency as well as balance global and local search, a distributed MA (DMA) is proposed in this article. A balanced best random distributed framework (BBRDF) is designed to achieve better optimization efficiency. In the global search procedure of DMA, novel recombination, mutation, and selection operators are proposed. A dynamic grouping recombination operator is proposed, which can help dynamically aggregate fragment information. Two mutation operators, namely: 1) merge and 2) split operators, are designed. With the help of these two operators, new fragments are formed through the merge operator, and fragments of constraint concerns are removed in the split operator. Besides, a two-dimension selection operator is proposed to evaluate the quality of each generated individual objectively. Furthermore, to improve solution precision, a splicing-driven local search (SDLS) is proposed to introduce rare utility fragments into the population without violating any constraints. With the help of extensive experiments, the proposed DMA shows significantly better performance than state-of-the-art approaches in database fragmentation. Moreover, the effectiveness of each proposed framework and operator is verified.

The remainder of this article is organized as follows. In Section II, the related work of database fragmentation is outlined. In Section III, the problem of database fragmentation is formally defined, and a relevant example is given. Subsequently, the proposed DMA, including a distributed framework, a dynamic recombination operator, two mutation operators, and an SDLS, is described in detail. Extensive experiments with discussion are given in Section V. After that, Section VI concludes.

II. Related Work

A. Approaches for Database Fragmentation

The first category of database fragmentation is based on the consideration of privacy protection. Vertical partition of a database for privacy was first introduced by [14], in which the database is vertically split into two parts. Two vertical partitioned databases cannot fulfill complex constraints. In [15], data fragmentation with encryption was utilized to break sensitive associations between information. In [11], to preserve data privacy, two models, that is: 1) hierarchical and 2) ring models, were designed for integrating information from the vertically partitioned database. The mechanism and advantages of the corresponding models were outlined. In [33], a novel model, called controlled query evaluation, was proposed, in which the inference-proofness of fragmentation was proved formally even if an attacker has prior knowledge. Gibbs *et al.* [34] mentioned that the fragmentation of customer data across multiple databases can be owned and maintained by separate functional units within an organization. Database utility is neglected in these works, which is crucial in actual database applications. To learn the association rules from vertically partitioned databases as well as preserve sensitive raw data from disclosing, the privacy-preserving mining technique was proposed [35]. In these approaches, database utility is neglected.

The second category of database fragmentation is designed for enhancing database utility. Database fragmentation for maximizing query efficiency has been studied in the area of distributed database [36]. In [37], an integrated methodology for fragmentation and allocation was proposed. Data are distributed across multiple sites in terms of utility. The authors have also verified the efficiency and effectiveness of the integrated methodology. In [28], the GA was utilized to generate fragmentation, which can satisfy the given constraints. In [38], to maximize the number of local accesses compared to accesses from remote sites, a decentralized approach was presented. Based on the observation of the access patterns, dynamic table fragmentation and allocation are both achieved. Subsequently, a bond energy algorithm with a modified similarity measure was proposed in [39] to optimize the communication cost and storage cost. In [40], the performance of the minimum spanning tree-based fragmentation approach and K-means clustering-based fragmentation approach were compared. To reduce the communication cost during query processing, frequent access patterns were utilized in [41], and multiple fragmentation strategies were proposed. In these approaches, privacy requirements are not taken into account.

The third category of database fragmentation focuses on both database privacy and database utility. To satisfy confidential constraints and minimize the cost of executing queries over fragments, the association between fragments was studied. A novel algorithm based on a fragmentation tree was proposed [19], in which pruning strategy is involved to limit the execution time. A graph search approach was proposed in [12], which can obtain near-optimal fragmentation when given a set of confidentiality constraints. The publication of data in terms of multiple loose associations between different pairs of fragments was studied in [42]. Given a specific level of protection for sensitive associations, the proposed algorithm can provide satisfying fragmentation. In [16], the problem of optimizing fragmentation regarding constraints and affinity between attributes was defined. Also, two heuristic algorithms that can optimize the number of fragments and the sum of affinity values were proposed. Moreover, the authors proved that the identified problems are NP-hard. When facing complex optimization situations, these approaches are very likely to lose their effectiveness due to the limited search ability. For enhancing the search efficiency in database fragmentation, evolutionary approaches have also been adopted. In [29], to achieve a better balance between database privacy and utility, a benefit-driven GA was proposed. A matching-based reconstruction was designed to integrate individuals of different fragmentation information. To rearrange elements in the generated individuals, a benefit-driven mutation operator was embedded.

Moreover, a comprehensive survey of fragmentation techniques was given in [13]. Overall, the third category of database fragmentation can outperform the other two categories since it considers both database privacy and database utility. In addition, due to the limitation of search efficiency, previous approaches cannot achieve an ideal balance between exploration and exploitation, which directly affects the accuracy of results. In this article, a dynamic grouping recombination operator and two mutation operators are proposed to enhance the exploration search ability. An SDLS strategy is designed to improve the exploitation search ability.

B. Memetic Algorithm

Inspired by natural evolution and the notion of meme, MA was proposed [25]. In general, MA can be regarded as a combination of population-based global search and local improvement procedures [43]. Previous studies of MA have revealed that MA is more likely to achieve the balance between exploration and exploitation during the optimization [25], [26]. Several surveys [43], [44] have been made to explore the promising research directions in MA.

Recent studies have been conducted on the application of MA to solve many complex problems. A game-based MA [30] was proposed for the minimum vector cover of the network, in which a game-based local search was implemented based on the best rule of the snowdrift game. In [45], a mutual information-based two-phase MA was proposed for the large-scale fuzzy cognitive map, in which MA was utilized to optimize the edge weights based on observed response sequences. In [46], a niching MA was designed for the multisolution traveling salesman problem. To improve search efficiency, a niche preservation technique with a selective local search strategy was proposed. An integer-coded MA [47] containing a recycling local improvement strategy was designed for enhancing the performance on wireless-sensor network problems. In [48], a greedy stochastic local search strategy was designed, and the proposed MA was utilized in course scheduling. A hybrid multiobjective MA [49] was proposed to tackle the periodic vehicle routing problem.

These studies verified the effectiveness of MA on solving complex optimization problems. These approaches all adopt serial models, which limit the optimization performance in solution accuracy and speed. In this article, a DMA containing a BBRDF is proposed.

III. Problem Definition

In database fragmentation, each privacy requirement is represented by a confidentiality constraint. These confidentiality constraints [19] are defined as follows.

Definition 1: Let $\mathcal{A}$ be the attributes of relation schema R, a confidentiality constraint is a subset $c \subseteq \mathcal{A}$.

Definition 2: For a set of confidentiality constraints $\mathcal{C} = \{c_1, \ldots, c_n\}$ $\forall c_i, c_j \in \mathcal{C}, i \neq j : c_i \not\subset c_j$, which indicates confidentiality constraints $\mathcal{C}$ cannot contain a constraint c_i that is a subset of another constraint c_j.

In the same manner, each utility requirement of database fragmentation [12] is defined as follows.

Definition 3: Let $\mathcal{A}$ be the set of attributes, a utility requirement is a subset $u \subseteq \mathcal{A}$ with a weight value $w(u)$.

Given a relation schema R, a set of constraints $\mathcal{C}$ over R, a fragmentation $\mathcal{F}$ is legal if:

1) $\forall F \in \mathcal{F}, \forall c \in \mathcal{C} : c \nsubseteq F$ (each individual fragment satisfies the constraints);
2) $\forall r \in R : \exists F \in \mathcal{F}$ such that $r \in F$ (fragments cover all attributes);
3) $\forall F_i, F_j \in \mathcal{F}, i \neq j : F_i \cap F_j = \varnothing$ (fragments do not have attributes in common).

When tackling database fragmentation, the main objective is to identify a legal fragmentation that can achieve the highest utility. Suppose a legal fragmentation $\mathcal{F}$ is identified as an optimal solution for the given relation schema R, constraints C, and a weighted list of utility requirements $\mathcal{U}$. It should meet all of the following conditions.

TABLE I
Example of Relation, Constraints, and Utility Requirements

Name	**YoB**	**Edu**	**ZIP**	**Disease**	**Race**	**Income**
Alice	1974	B.Sc	90015	Flu	Asian	5K
Bob	1965	MBA	90038	Diabetis	White	2K
Carol	1976	Ph.D	90001	Calculi	Black	4K
Greg	1975	M.Sc	90025	Flu	Black	3K

Constraints			C-1 = { Name, Disease }
			C-2 = { Edu, Income }
			C-3 = { YoB, Edu, ZIP }			

Utility requirements			U-1 = { Name, YoB }
			U-2 = { YoB, Income }
			U-3 = { Disease, Race }			

1) $\forall F \in \mathcal{F}, \forall c \in \mathcal{C} : c \nsubseteq F$.
2) $\forall r \in R : \exists F \in \mathcal{F}$ such that $r \in F$.
3) $\forall F_i, F_j \in \mathcal{F}, i \neq j : F_i \cap F_j = \varnothing$.
4) $\forall \mathcal{F}'$ satisfying the first two conditions such that $\text{utility}(\mathcal{F}') \leq \text{utility}(\mathcal{F})$

which means the optimal fragmentation $\mathcal{F}$ can satisfy all predefined constraints and provide higher utility than any other fragmentation which can also meet all constraints.

The threat model of database fragmentation [50], [51] is set as follows.

1) Service providers are considered as "honest but curious" [52]. The responses to queries are always accurate. Service providers are curious and may infer and analyze outsourced data.
2) Servers storing fragments can communicate and collude with each other to extract knowledge about outsourced data.
3) Queries of users are operated under the given protocol. Users are curious and may attempt to acquire unauthorized information related to privacy [53].
4) Clients of users are assumed to be secure. Only trustful users can access clients.
5) Architectures connecting clients and servers are assumed to be trustworthy.

Table I illustrates an example of a medical relation to be released. Confidentiality constraints and utility requirements over it are also listed. To fulfill constraint C-1 and protect the medical privacy of patients, attributes "Name" and "Disease" cannot be inserted into the same fragment. To satisfy U-3, attributes "Disease" and "Race" should be assigned into the same fragment. This way, the utility of the database is improved since data of disease and race can be investigated without leaking any medical privacy.

IV. Distributed Memetic Algorithm

In this section, we first introduce the proposed BBRDF. After that, strategies in representation and initialization are outlined. Then, operators for enhancing global search, that is, dynamic-grouping recombination (DGR), merge and split-based mutation operators are introduced in detail. To improve the precision of the solution, an SDLS strategy is designed. Finally, the overall process of the proposed approach is illustrated.

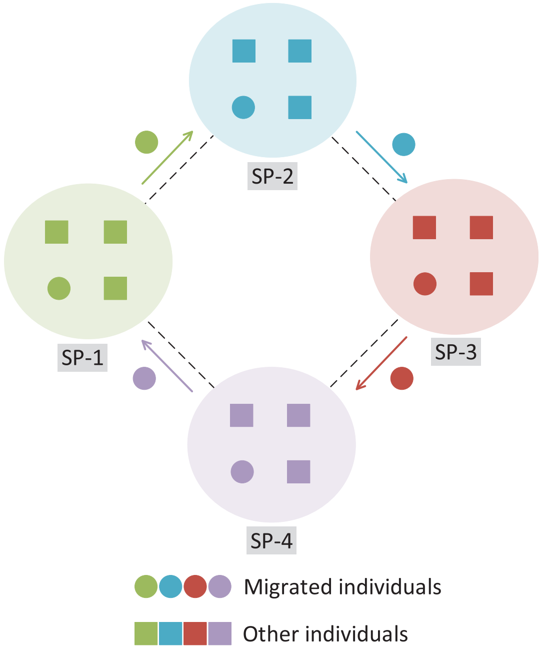

Fig. 1. Example of a general DEA framework.

A. Balanced Best-Random Distributed Framework

To achieve higher optimization efficiency in database fragmentation, BBRDF is proposed. BBRDF is implemented according to the general framework of distributed EA (DEA). The entire population of the algorithm is divided into multiple subpopulations (SPs). Each SP evolves independently. To achieve effective communication between SPs, they are connected according to a predefined topology. With a given interval, migration is performed. Before migration, elite individuals of each SP is selected. During migration, according to the communication topology, selected individuals of each SP are sent to its neighbor SP. After migration, the migrated individuals are inserted into the corresponding SPs. Individuals, as well as evolution information, are exchanged through migration.

An example of the general DEA framework is given in Fig. 1. As shown in the example, each SP is labeled by a unique color, and migrated individuals are marked as a circle. Four SPs communicate according to the ring topology. With a predefined interval, the chosen migrated individuals are shared through the network. This way, population diversity is maintained during the entire evolution process.

At the beginning of evolution, the entire population of BBRDF is divided into N SPs, and each SP evolves independently. For enhancing the population quality as well as the diversity of each SP, in each SP of BBRDF, two individuals are chosen for migration. The first individual is the best in each SP, and the second individual is randomly selected in each SP. Subsequently, the best individual of each SP is migrated to its neighbor in the clockwise direction. The migrated best individuals can exchange evolutionary information in the target SPs and lead the search of other individuals. In contrast, the random individuals migrated anticlockwise, which can help maintain the population diversity in target SP. Then,

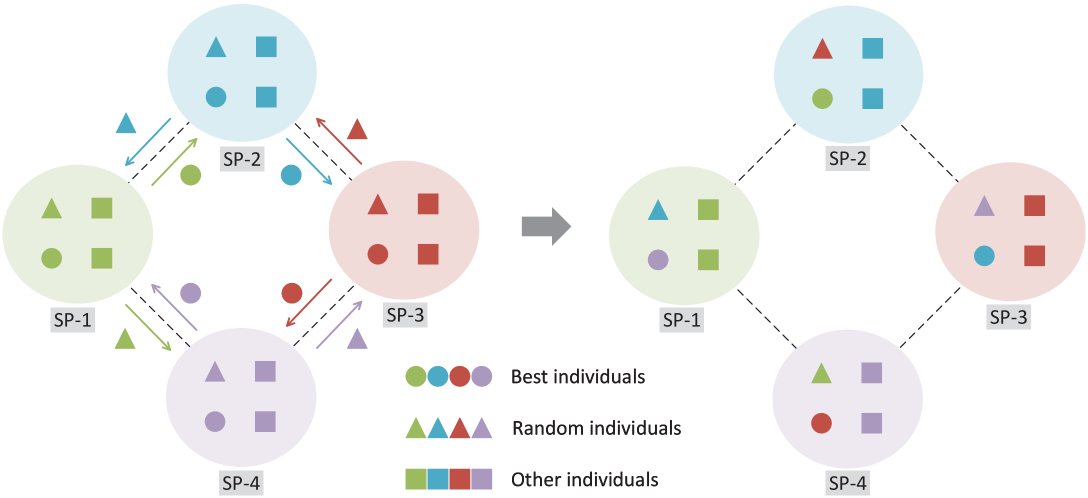

Fig. 2. Example of the migration process.

each SP receives the best individual and a random individual from neighbors in two directions. Migrated individuals from different SPs can avoid the problem of redundant information.

An example of the BBRDF is given in Fig. 2. In the example, each SP and its individuals are marked by a unique color. First, in each SP, the best individual and a randomly chosen individual are listed for migration. The best individual is represented by a circle and sent clockwise. In contrast, the random individual is represented by a triangle and sent anticlockwise. Each SP receives two kinds of individuals from corresponding two neighbors. Population diversity and quality are both enhanced.

B. Representation and Initialization

During the evolution, each fragmentation containing multiple fragments is represented by an individual during the evolution. Each fragment is represented by a vector that contains the corresponding attributes. Each attribute is regarded as an element in the further description. Constraint and utility requirements can also be indicated by vectors containing the corresponding elements.

For a given fragmentation, its privacy fitness and utility fitness are calculated as follows:

$$fp = \sum_{i=1}^{\mathrm{NC}} Vi \tag{1}$$

$$V_i = \begin{cases} 1, & \text{if constraint } i \text{ is violated} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$fu = \sum_{i=1}^{\mathrm{NU}} S_i \times W_i \tag{3}$$

$$S_i = \begin{cases} 1, & \text{if utility } \ i \text{ is satisfied} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where *fp* is the value of privacy fitness, NC represents the number of given constraints, V_i is a Boolean value indicates whether the ith constraint is violated by the corresponding fragmentation, *fu* is the value of utility fitness, NU is the number of utility requirements, and S_i indicates whether the ith utility requirement is satisfied. Considering different utility requirements are of different access frequency, a predefined weight W_i is given for each utility requirement.

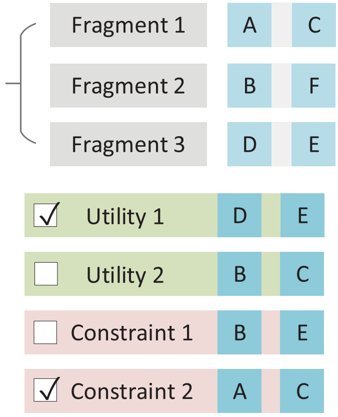

Fig. 3. Example of individual representation and fitness evaluation.

An example of fragmentation representation is given in Fig. 3. As shown in the example, the individual contains six elements and divided into three fragments. Fragment 1 involves elements A and C while fragment 2 and fragment 3 contain elements B and F and elements D and E, respectively.

Utility 1 can be satisfied by the given individual since elements D and E are both included in fragment 3. In contrast, utility 2 is not satisfied. Similarly, constraint 2 is violated by the given individual. For the given individual, its utility fitness is the weight of utility 1, and its privacy fitness is 1.

According to the proposed distributed framework, the entire population is initialized and divided into multiple groups at the beginning of the overall process. To initialize the population, a heuristic strategy is proposed. In the beginning, one utility requirement is randomly chosen for each initial individual. In each initial individual, the first fragment is constructed according to the chosen utility requirement. Subsequently, unallocated elements are chosen by random. For each unallocated element, it has the same chance to construct a new fragment or join an existing fragment without violating given constraints.

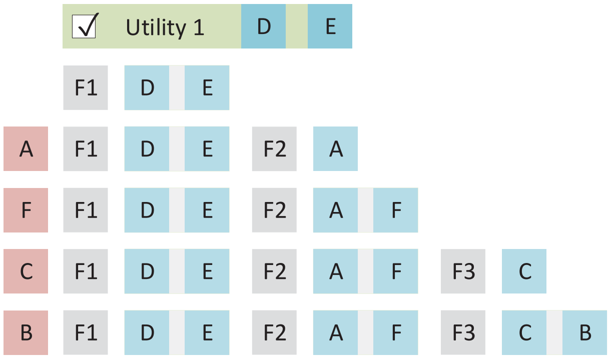

Fig. 4. Example of heuristic initialization.

An example of the proposed heuristic strategy is given in Fig. 4. In the beginning, the first utility containing two elements is randomly chosen as the base for constructing the heuristic individual. Then, the first fragment $F1$ containing these two elements, D, E, is created. Subsequently, element A is randomly selected to construct a new fragment. Element F is inserted into the existing fragment $F2$. In the same manner, element C is inserted into a new fragment, and element B is inserted into the same fragment.

First, the proposed heuristic initialization can guarantee that each individual satisfies at least one utility requirement. Second, the generated initial individuals do not violate any constraint. Third, since elements are randomly inserted into existing fragments or new fragments, initial population diversity is guaranteed.

C. Dynamic-Grouping Recombination

One primary target of the recombination operator is to extract valuable fragmentation information from the existing parent individuals and generate child individuals with higher fitness. According to the problem definition, an individual of higher fitness means it can exclude fragments containing illegal constraints and satisfy more utility requirements. Fragments in each individual can be used in recombination to generate better individuals.

To utilize more information in fragments, a DGR operator is proposed. If two elements are allocated in the same fragments in parent individuals, they are more likely to be assigned in the same fragment in the child individuals. Thus, DGR is designed as follows. First, a random order for element allotment is generated. Each element is assigned according to this random order. Second, each element creates its grouping table. Two elements are regarded as neighbors if they appear in the same fragments. For each element, all its neighbors are listed in its grouping table. It is to be noted that elements in each grouping table can duplicate, which means some elements can appear twice in a single grouping table. If elements are allocated in the same fragments in both parent individuals, they are more likely to be allotted together. For each element, it can randomly choose the same fragment as its neighbor in parent individuals or construct a new fragment. Once an element is allocated in the child individual, its fragment index is dynamically updated in the grouping table. The following elements can acquire new choices in fragment allocation.

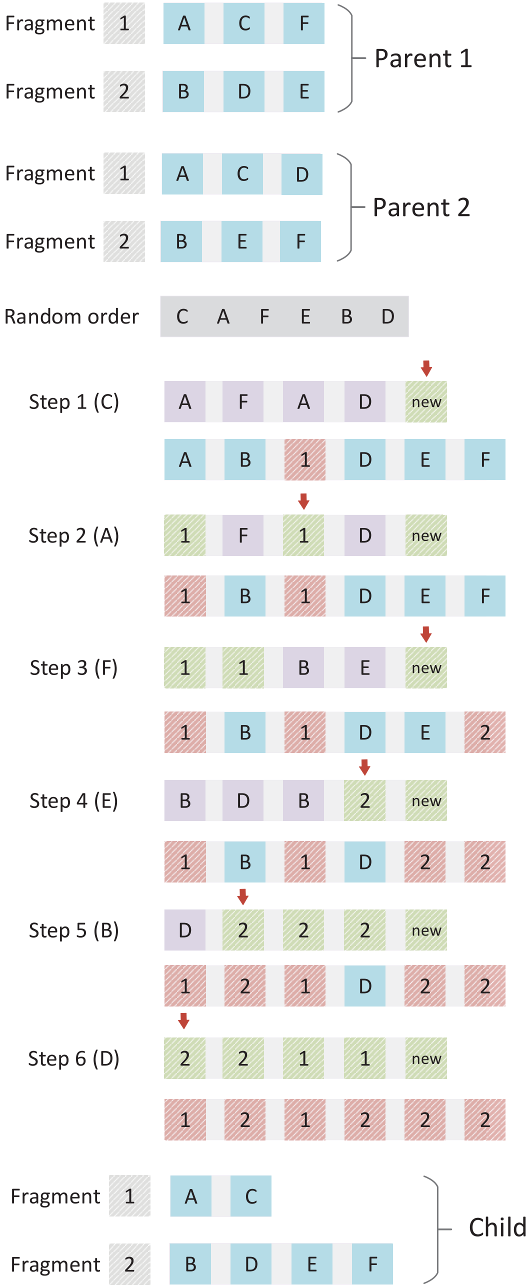

Fig. 5. Example of DGR.

An example of DGR is given in Fig. 5. As shown in the figure, two parent individuals are chosen for recombination. Each contains six elements. Random order is then generated. In the first step, element C is chosen. Four neighbors of element C, that is, A, F, A, D, are listed in the first row. In addition, "new" represents constructing a new fragment. Since no element is allocated, no grouping information can be given. This way, element C is put into new fragment 1. Accordingly, the content of element C in the second row is changed to 1, which indicates the current result of recombination. Afterward, element A is tacked in step 2. According to the grouping information, element C acts as its neighbor in both parent individuals. In the first row of this step, positions of element C are labeled by 1, which represents the grouping information given by element C. Thus, element A has $(2/3)$ possibility to be allocated

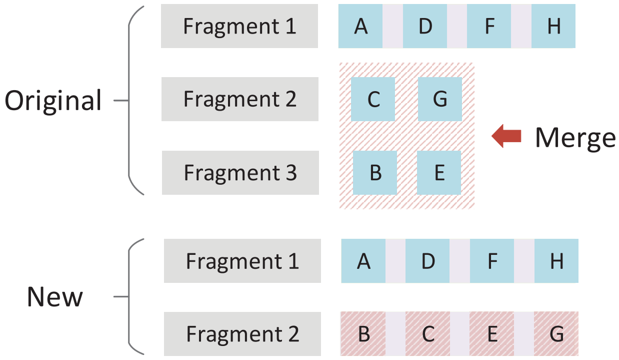

Fig. 6. Example of merge mutation where fragment 2 and fragment 3 are merged in the new individual.

in the same fragment as element C (fragment 1) and (1/3) possibility to be allocated in a new fragment. After the allocation, grouping information is updated. In step 3, since element F is the neighbor of element A and element C in Parent 1, it also has (2/3) chance to be put into fragment 1 but makes a different choice from element A. Grouping information is updated accordingly, and all the other elements are allocated according to the predefined random order and dynamically updated grouping information. In child individual, elements A and C are allocated into fragment 1 and all the other elements are in fragment 2.

With the help of the dynamically updated grouping table, elements are allocated. Elements are more likely to be assigned to the fragments as its neighbors in previous parent individuals. Furthermore, with a dynamic possibility, a new fragment is constructed for the element to achieve exploration. This way, part of grouping information in the parent individuals is kept while new grouping is formed through random exploration.

D. Merge and Split-Based Mutation

With the help of new fragmentation introduced by the mutation operator, population diversity can be maintained. In general, the mutation operator is carried out by randomly changing positions of elements. In database fragmentation, besides the general granularity of the element, the fragment can also be regarded as granularity, which means elements in the fragments can be integrally managed.

During the optimization of database fragmentation, without effective guidance, two kinds of situations are likely to appear. First, some important but complex utility requirements are difficult to satisfy. Second, some promising fragmentations containing evolutionary outcomes are removed because of violating constraints. To tackle these two situations, two novel mutation operators, that is, merge mutation and split mutation, are proposed. In addition, to improve the population diversity in a random manner, random mutation is also adopted.

1) Merge-Mutation (M-M): During the optimization of database fragmentation, some complex utility requirements combining many elements are difficult to satisfy. To help satisfy these utility requirements as well as accelerate the search process, the first mutation operator, called merge mutation, is proposed. When executing merge mutation, two fragments in the original fragmentation are randomly chosen. Then, all elements in these two fragments are combined, and two fragments are merged. An example of a merge mutation is given in Fig. 6. As shown in this example, the second fragment and the third fragment of the original individual are chosen. Four elements in these two fragments are combined. In the newly generated individual, the number of fragments decreases to two and the second fragment contains all the combined elements, namely, element B, element C, element E, and element G. With the help of the merge mutation, some unusual combinations can be constructed to fulfill some complex utility requirements.

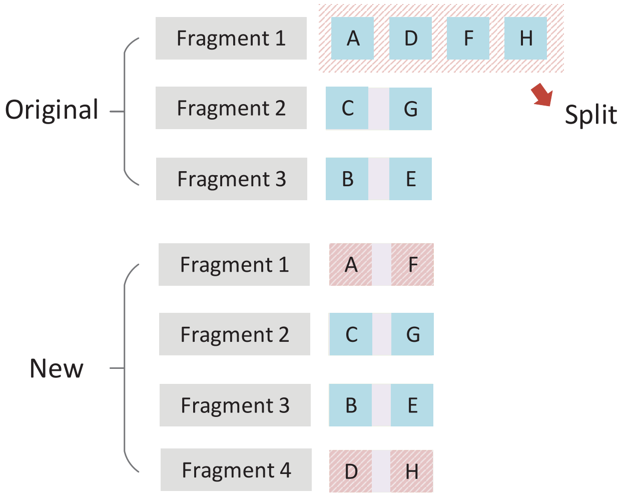

Fig. 7. Example of split mutation where fragment 1 in original individual is processed.

2) Split-Mutation (S-M): Another situation is that some promising fragmentations are removed since they violate constraints. To keep the promising fragmentations so as to promote the search process, the second mutation operator, called a split mutation, is proposed. During the proposed split mutation, one fragment in the original fragmentation is randomly selected. All elements in the chosen fragment are randomly divided into two groups. Each group contains part of the elements in the original fragment. These two groups are inserted into the new individual and act as two fragments to replace the chosen fragment. An example of the split mutation is shown in Fig. 7. In this example, fragment 1 is selected for the split. Afterward, four elements in the fragment are divided into two groups, namely, fragments A and F and fragments D and H. In the new individual, the number of fragments increases to four. Through executing the split mutation operator, some misleading element combinations can be removed, and the generated individuals are more likely to satisfy more constraints.

3) Random Mutation (R-M): Other than the proposed merge and split mutation operators, the general random mutation operator is also adopted to adjust the positions of elements by the granularity of a single element. In random mutation, with a predefined possibility P_r, each element in the original individual is randomly allocated to a new fragment.

E. Two-Dimension Selection

To compare different solutions of database fragmentation, both privacy fitness and utility fitness should be considered.

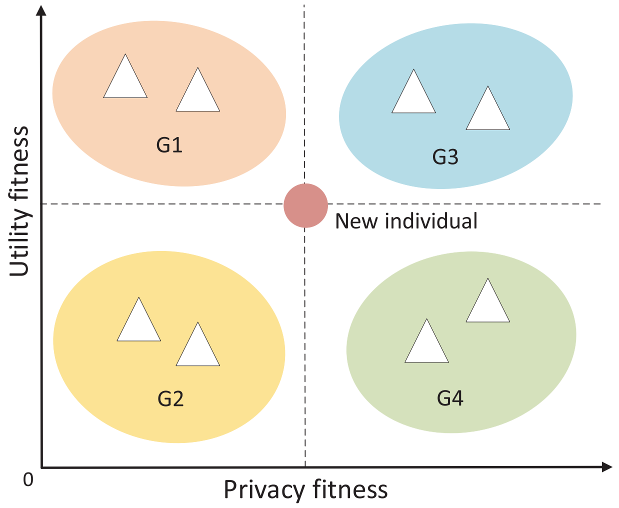

Fig. 8. Illustration of two-dimension selection.

According to the problem definition, a fragmentation is legal only if its privacy fitness equals 0. Also, the objective of the optimization is to find the optimal fragmentation that can satisfy all predefined constraints and can provide higher utility than any other fragmentation. Based on the definition, privacy fitness is of higher priority. When comparing two individuals, privacy fitness is first compared. If two individuals are of the same privacy fitness, utility fitness is compared. To be specific, fragmentation F_1 is fitter than fragmentation F_2 if:

1) $fp(F_1) < fp(F_2)$;
2) $fp(F_1) = fp(F_2)$ and $fu(F_1) > fu(F_2)$

where fp and fu are privacy fitness and utility fitness of the given solution.

As an example, in Fig. 8, compared with the red newly generated individual, existing individuals in $G1$ and $G2$ are of higher overall fitness and individuals in $G3$ and $G4$ are of lower overall fitness.

F. Splicing-Driven Local Search

During the optimization of database fragmentation, it is common that part of utility requirements is satisfied by all individuals while other utility requirements are not involved in any fragmentation. The unbalance of satisfaction can cause the optimization process to get trapped. Although some utility requirements can be achieved by merge mutation, it is also likely to be removed due to the involvement of illegal fragments. To construct legal database fragmentation including the rare utility requirement to expand the search space of the optimization, an SDLS is proposed.

The execution of the proposed SDLS is divided into three steps. First, for each selected individual, its unsatisfied utility requirements are listed. One of the utility requirements is randomly chosen for fragment construction. Subsequently, positions of all elements in the selected utility requirement are marked. Finally, all elements in the original individual are extracted from its current fragment and combined to construct a new fragment.

As shown in Fig. 9, an instance of SDLS is given. Three utility requirements are listed in this example. The first two utility requirements, which are marked as green, have been satisfied by the original individual. On the contrary, utility 3, which is marked as red, has not been satisfied. Based on this situation, all the elements in utility 3 are extracted from their original fragments and combined to construct a new fragment. Fragment 4 is formed and inserted into the new individual. As a result, utility 3 is satisfied by the newly generated individual.

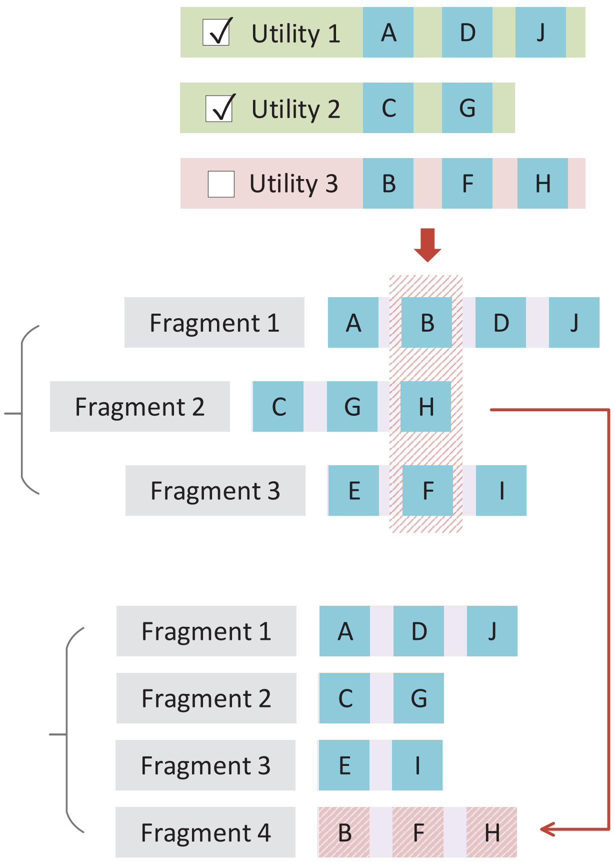

Fig. 9. Example of SDLS.

After executing the SDLS, rare utility requirements are introduced in the improved individuals. Grouping information of these utility requirements can be utilized and extended in the further optimization process. In the proposed SDLS, only elements in the given utility requirements are extracted to construct a new fragment. Considering the utility requirements do not conflict with the given constraints, the new fragment has no concern of violating any constraint.

G. Overall Process

Algorithm 1 shows the entire process of the proposed algorithm. The entire process is divided into two parts, that is, the global controller at the master node and SP at the slave node. At the beginning of the procedure of the global controller, the counter of generation g is initialized as 0, and the entire population is randomly initialized. Afterward, privacy fitness fp and utility fitness fu of each initialized individual are evaluated. The entire population is divided into N SPs and each SP is sent to one slave node to evolve independently. Migration operator is carried out every MI generations. During migration, the global controller receives migrated from each SP and

Algorithm 1 Pseudocode of the Proposed DMA

```
procedure GLOBAL CONTROLLER (AT MASTER NODE)
    Set $g = 0$ ($g$ is current generation of population)
    Initialize the population $P$
    Evaluate $fp$ and $fu$ for initial individuals
    Spawn $N$ subpopulations
    while stopping criterion is not met do
        if $g \% MI = 0$ then
            receive migrated individuals from each subpopulation
            for each received individual do
                send the individual to its corresponding subpopulation
            end for
        end if
        $g = g + 1$
    end while
    Output convergence data
    Output the best solution
end procedure

procedure SUBPOPULATION EVOLUTION (AT SLAVE NODE)
    while stopping criterion is not met do
        for each pair of parent individuals do
            Set the parent individual of less competitiveness as father individual
            Perform dynamic-grouping recombination
            Execute mutation operator on offspring
            if Offspring is better than father individual then
                Replace the father individual by offspring
            else
                Put offspring into $LSP$
            end if
        end for
        Execute local search strategy on $LSP$
        for each individual in $LSP$ do
            Replace the corresponding father individual by it
        end for
    end while
end procedure
```

sends them to their corresponding SPs. At the end of evolution, the global controller outputs convergence data and the best solution.

In each slave node, each SP evolves independently. For each pair of parent individuals, the fitter individual is set as the mother individual, and the other is set as the father individual. Subsequently, DGR is performed. One of the mutation operators is selected by random with the same probability and carried out on the generated offspring. If the mutated offspring is better than the father individual, replace the father individual by the offspring. Otherwise, put the offspring into local search population *LSP*. After finalizing the global search on each pair of parent individuals, a local strategy is executed on *LSP*. The outcome of the local search is used to replace the corresponding father individual. Moreover, the pseudocode of all the proposed operators in DMA is provided in Algorithm S1 of the supplementary file.

V. Experimental Results

A. Experimental Setup

To evaluate the performance of the proposed DMA, 18 test cases are utilized. These test cases are generated according to the problem definition and metadata published by the Australian Institute of Health and Welfare.[1] For each test case, a set of constraints and a list of utility requirements are given. Also, the weight of each utility requirement is generated by random, and its value is between 1 and 100. Table SI of the supplementary file shows the properties of each test case, including the number of elements NE, number of constraints NC, size of constraints SC, number of utility requirements NU, and the corresponding size SU.

The parameters of the proposed algorithm are set as follows. Population size NP is set as 80, number of SPs is set as 4, migration interval MI is set as 30, random mutation possibility P_r is set as 0.2, and the maximum number of fitness evaluations $MaxFEs$ is set as $NE \times 10^3$. Furthermore, the effectiveness of the parameter setting and the corresponding parameter sensitivity are studied in the following part.

The proposed distributed framework is based on the message passing interface (MPI). Each SP is assigned to a single core and evolves in parallel. The proposed and all the compared algorithms are implemented in C++ and performed on a local cluster containing 60 compute nodes (OS: Ubuntu 16.04; CPU: 3.40-GHz 4-Core Intel i5-7500; and Memory: 8 GB).

B. Comparisons With State-of-the-Art Algorithms

The DMA algorithm is designed to tackle database fragmentation problems, which is important in the area of database storage. To verify the performance of the proposed DMA, experiments are carried out to compare DMA with four state-of-the-art algorithms for database fragmentation, namely, the benefit-driven GA [29], heuristic algorithm [16], graph search [12], and fragmentation tree [19]. These four state-of-the-art algorithms are listed as follows and the parameters are set according to their original papers.

1) *GA-BD [29]:* This GA adopts a matching-based recombination operator and a benefit-driven mutation operator to help achieve the balance between database privacy and utility.
2) *HA [16]:* This heuristic algorithm is based on the utility matrix and greedy strategy. It has shown efficient performance in handling constraints and utility in database fragmentation.
3) *GS [12]:* In this approach, the fragmentation search space is modeled as a graph, and a novel levelwise graph expansion is utilized to reduce the search time.
4) *FT [19]:* In this algorithm, a fragmentation tree is built over a given fragmentation lattice. Each fragment is represented as a node in the tree. A heuristic approach is designed to search near-optimal solutions on this tree.

In Table II, the mean and standard deviation of the utility fitness values over 25 independent runs are presented and the best results are highlighted in boldface. To be noted, since all these approaches can achieve legal solutions whose privacy fitness values are 0, the privacy fitness values are not shown in this table. According to the result table, the proposed DMA can achieve the best performance on most of the test cases. Overall, the proposed DMA acquires the best results on 15 test cases. Due to the benefit-driven strategy utilized by GA-BD, it can outperform on these two test cases of less complex search space. On the test cases of higher values of NE, the

[1] https://www.aihw.gov.au/about-our-data/metadata-standards

TABLE II
COMPARISONS WITH STATE-OF-THE-ART ALGORITHMS

Approaches	**DMA**		**GA-BD**		**HA**	**GS**		**FT**
	Mean	**Std**	**Mean**	**Std**	**Result**	**Mean**	**Std**	**Result**
T_1	**2.25E+02**	0.00E+00	1.72E+02	3.85E+01 $^{-}$	1.50E+01 $^{-}$	2.20E+00	1.08E+01 $^{-}$	1.15E+02 $^{-}$
T_2	**2.40E+02**	9.54E+00	2.18E+02	1.59E+01 $^{-}$	1.20E+02 $^{-}$	6.80E+00	2.31E+01 $^{-}$	2.00E+01 $^{-}$
T_3	5.12E+02	7.94E+01	**5.23E+02**	7.26E+01 $^{+}$	3.00E+02 $^{-}$	6.60E+00	2.01E+01 $^{-}$	2.75E+02 $^{-}$
T_4	**4.79E+02**	3.29E+01	3.83E+02	5.21E+01 $^{-}$	8.00E+01 $^{-}$	0.00E+00	0.00E+00 $^{-}$	9.00E+01 $^{-}$
T_5	**8.66E+02**	2.09E+01	6.95E+02	1.47E+02 $^{-}$	5.00E+01 $^{-}$	1.04E+01	2.82E+01 $^{-}$	1.15E+02 $^{-}$
T_6	**8.70E+02**	6.80E+01	7.25E+02	1.42E+02 $^{-}$	1.10E+02 $^{-}$	3.00E+00	1.47E+01 $^{-}$	6.05E+02 $^{-}$
T_7	8.12E+02	8.01E+01	**8.60E+02**	1.04E+02 $^{+}$	1.85E+02 $^{-}$	8.20E+00	1.93E+01 $^{-}$	2.85E+02 $^{-}$
T_8	**8.97E+02**	1.55E+02	7.78E+02	1.20E+02 $^{-}$	3.10E+02 $^{-}$	8.00E-01	3.06E+00 $^{-}$	1.55E+02 $^{-}$
T_9	**7.32E+02**	1.54E+02	5.36E+02	1.66E+02 $^{-}$	3.60E+02 $^{-}$	0.00E+00	0.00E+00 $^{-}$	2.10E+02 $^{-}$
T_{10}	**8.16E+02**	1.07E+02	5.13E+02	1.23E+02 $^{-}$	4.95E+02 $^{-}$	4.00E-01	1.96E+00 $^{-}$	4.50E+02 $^{-}$
T_{11}	**7.66E+02**	1.31E+02	4.76E+02	1.76E+02 $^{-}$	5.45E+02 $^{-}$	0.00E+00	0.00E+00 $^{-}$	2.55E+02 $^{-}$
T_{12}	**7.49E+02**	1.67E+02	4.12E+02	1.49E+02 $^{-}$	3.80E+02 $^{-}$	2.60E+00	9.50E+00 $^{-}$	4.15E+02 $^{-}$
T_{13}	**5.50E+02**	1.32E+02	2.55E+02	7.95E+01 $^{-}$	3.15E+02 $^{-}$	3.60E+00	1.76E+01 $^{-}$	3.60E+02 $^{-}$
T_{14}	**6.43E+02**	1.44E+02	4.89E+02	1.10E+02 $^{-}$	4.75E+02 $^{-}$	6.00E-01	2.94E+00 $^{-}$	2.45E+02 $^{-}$
T_{15}	5.93E+02	1.27E+02	3.09E+02	6.85E+01 $^{-}$	2.15E+02 $^{-}$	3.00E+00	1.47E+01 $^{-}$	**6.05E+02** $^{+}$
T_{16}	**8.07E+02**	1.91E+02	5.88E+02	8.47E+01 $^{-}$	5.45E+02 $^{-}$	0.00E+00	0.00E+00 $^{-}$	4.95E+02 $^{-}$
T_{17}	**6.81E+02**	1.27E+02	3.23E+02	1.24E+02 $^{-}$	3.05E+02 $^{-}$	1.00E+00	4.90E+00 $^{-}$	3.10E+02 $^{-}$
T_{18}	**8.22E+02**	1.93E+02	3.17E+02	6.64E+01 $^{-}$	6.05E+02 $^{-}$	0.00E+00	0.00E+00 $^{-}$	4.60E+02 $^{-}$
$-/\approx/+$			16/0/2		18/0/0	18/0/0		17/0/1

proposed DMA can achieve better performance and show its advantages in search ability. This is due to the proposed distributed framework, that is, BBRDF, which can help exchange information between SPs efficiently. Besides, the global search and local search are balanced and they both contribute to solution precision. The average ranks of mean values achieved by DMA and the compared state-of-the-art approaches on 18 test cases over 25 runs are calculated and plotted in Fig. S1 of the supplementary file. The average rank of DMA is around 1.2, which is much lower than the average ranks of other compared approaches, which can also verify the advantage of its performance.

To show the advantage of DMA in a statistical sense, the Wilcoxon rank-sum test with 0.05 level is adopted, and results are also listed in Table II, in which the comparison results are labeled as $-/\approx/+$, where "$-$," "$\approx$," and "$+$" indicate that the compared approach is significantly worse than, equivalent to, and better than the complete version of the DMA algorithm, respectively. It is clear that DMA can significantly outperform in most of the test cases. To sum up, DMA can provide significantly better results on 16, 18, 18, and 17 test cases than the compared state-of-the-art algorithms.

Besides, Fig. S2 of the supplementary file shows the convergence curves of utility fitness values achieved by five approaches on all the test cases. Each point on the plot is located by calculating the average values of the corresponding approach in 25 independent runs. Take the convergence curve of HA as an example. With the help of its greedy strategy, HA can achieve quick convergence at the beginning stage of the search process. In both simple test cases such as T_3 and complex test cases such as T_{17}, its converge curves are highest at the very beginning. However, with the development of evolution, its search is very likely to be trapped in the local optima, and the greedy strategy is not helpful in jumping out. Thus, its convergence curve is not variable afterward. The same situation also happens in other approaches, such as GS and FT. Due to the limitation of the search strategy, they are very likely to be trapped during the evolution process. For GA-BD, although it can sustainably identify better solutions during the entire evolution process, its search speed is limited by its population model and cannot achieve an ideal degree. Also, fragment information is not adequately utilized. The proposed DMA can achieve the highest convergence speed both on simple test cases and complex test cases. Optimization efficiency is enhanced by the proposed BBRDF. Global search procedure can improve population diversity and explorative ability, which is crucial in complex test cases. The proposed local search strategy is helpful in improving solution precision.

C. *Effect of the Proposed Distributed Framework*

To investigate the effectiveness of the entire BBRDF and migrated individuals, three variants of DMA are implemented and compared with the original version. These four variants are listed as follows.

1) *DMA-No-BBRDF:* DMA algorithm without the proposed distributed framework BBRDF.
2) *DMA-No-Best:* DMA algorithm without the best individual migrated in BBRDF.
3) *DMA-No-Random:* DMA algorithm without a random individual migrated in BBRDF.
4) *DMA-Multiobjective:* Since two kinds of fitness, that is, privacy fitness and utility fitness, are involved in database fragmentation, it can also be solved by a multiobjective approach. In this variant, one single population is employed and the NSGA-II framework [54] is utilized to achieve the multiobjective optimization of privacy fitness and utility fitness. Multiobjective individuals can enhance the exploration search. Except for BBRDF, all the other operators proposed in DMA are utilized in DMA-multiobjective.

Table SII of the supplementary file shows the comparisons of experimental results where the best results are highlighted in boldface. The Wilcoxon rank-sum test at a significant level 0.05 is executed and labeled in the table. According to the given table, the proposed DMA can obtain the best optimization results on 14 out of all test cases. For the test cases of lower numbers of elements, the variant DMA-no-BBRDF can achieve results without significant difference. This is mainly because these test cases are of lower complexity. This way, the advantage of population diversity in BBRDF cannot show off. For the test cases of higher numbers of elements, BBRDF can achieve population diversity during the optimization and obtain significantly better results. Compared with DMA-multiobjective, DMA can achieve significantly better performance on 18 test cases. Although the individuals in DMA-multiobjective can continuously provide evolutionary information for enhancing the exploration search, its convergence speed is limited by the multiobjective optimization, which makes DMA-multiobjective of lower solution accuracy. DMA can achieve better convergence speed due to the enhancement of exploitation search.

In addition, Fig. S3 of the supplementary file presents the average ranks of DMA and four variants on all test cases. We can see a clear gap of the average rank between DMA and other variants. The proposed BBRDF-distributed framework is effective in enhancing the performance of DMA.

D. Effect of the Proposed Operators

The main operators of the proposed algorithm contain DGR, M-M, M-S, and SDLS. To verify the effectiveness of these operators, four DMA variants are implemented and compared with the original DMA algorithm. These variants are listed as follows.

1) *DMA-no-HI:* The heuristic initialization in DMA is replaced by random initialization.
2) *DMA-no-DGR:* The DGR operator is removed from the DMA algorithm.
3) *DMA-no-M-M:* Merge-mutation is removed from the DMA algorithm.
4) *DMA-no-M-S:* Split-mutation is removed from the DMA algorithm.
5) *DMA-no-SDLS:* The SDLS operator is removed from the DMA algorithm.

Table SIII of the supplementary file shows the comparisons of experimental results where the best results are highlighted in boldface. According to the given table, the complete version of the proposed DMA approach can achieve the best performance on 12 out of all 18 test cases. To be specific, compared with DMA-no-HI, the advantage of performance obtained by the complete version verifies the heuristic information given in the initial population is effective in the subsequent evolutionary process. Compared with DMA-no-DGR, the complete version can outperform on 17 test cases, which means the proposed DGR operator can help improve the performance of DMA in various optimization situations. Similarly, the proposed SDLS can also enhance the performance of DMA by improving the precision of solutions. Furthermore, the average ranks achieved by DMA and the other variants are plotted in Fig. S4 of the supplementary file. Overall, DMA can achieve the lowest average rank, which indicates DMA can outperform with the help of all the proposed operators. The effectiveness of the proposed operators is also verified.

As shown in the table, the results of the statistical analyses are represented in a "$-/\approx/+$" manner. Through analyzing the comparison results, it is clear that the full version of the DMA algorithm can obtain significantly better results than the other compared versions on the majority of the test cases. In other words, all the proposed approaches are effective in enhancing the performance of DMA.

E. Sensitivity Analysis

Migration interval *MI*, population size *NP*, and SP size *NSP* are three manually defined parameters in DMA. Considering their importance, the performance of DMA may be sensitive to their values. To investigate the sensitivity of DMA on these three parameters, we compare the performance of DMA adopting different values of *MI*, *NP*, and *NSP*.

In general, if the value of *MI* is too low, which means evolution information is exchanged frequently, the evolution process tends to be relatively exploitative. In the test cases of a simple environment, DMA with low *MI* can help directly indicate the optimal solution. On the contrary, DMA with high *MI* can help maintain population diversity. Since evolution information is exchanged with a lower frequency, different SPs are guided independently. This kind of DMA can outperform on the complex test cases. The average ranks achieved by DMA utilizing seven different migration intervals are plotted in Fig. S5 of the supplementary file. As shown in the plot, when the values of *MI* is set in the defined range, different versions of DMA do not make big differences, and the average ranks are all located between 3.5 and 4.2.

Also, the performance of DMA may be sensitive to the values of *NP* and *NSP*. If the value of *NP* is set to low and the value of *NSP* is set too high, the search of DMA tends to be exploitative. Elite individuals are selected from large SPs and evolutionary information in elite individuals is quickly exchanged between big SPs. In contrast, the search for DMA can be relatively explorative. Since different SPs are guided by different elite individuals, population diversity is maintained. To test whether DMA is so sensitive to *NP* and *NSP*, DMA with different values of *NP* and *NSP* is carried out and the average ranks achieved are plotted in Fig. S6 of the supplementary file. As shown in the figure, the average ranks achieved by each variant are all between 4 and 5.5. There is no big difference between average ranks achieved by DMA with values of *NP* and *NSP* in the predefined range.

To sum up, the performance of DMA is not so sensitive to *MI*, *NP*, and *NSP*. According to the experimental results, the parameter combination adopted in this article, namely, $MI = 30$, $NP = 80$, and $NSP = 20$ could achieve the best performance.

F. Visulization of Optimization Process

To verify the optimization process of DMA, an example of the database fragmentation process in T_1 is given in Fig. S7 of the supplementary file, in which 12 attributes are represented by *A*–*L* and *g* indicates the counter of generation. As shown in the example, different attributes are first initialized and allocated in different fragments. Then, driven by the search procedure, attributes in different fragments are gathered step by step. The number of fragments decreases from 8 to 4 in the first 80 generations. After that, attributes are migrated between fragments and the precision of the solution is improved.

G. Speedup Ratio

The direct parallel implementation is one of the advantages of EAs. To achieve a better performance in database fragmentation, DMA is implemented in the parallel island model, and each SP in the proposed DMA approach is assigned to an independent computing node. Information exchange between SPs is realized by sending and receiving messages between computing nodes. Thus, the parallel granularity of each DMA approach equals its number of SPs. The number of SPs can directly affect the running speed of DMA.

Since the speedup ratio is an important metric to evaluate the efficiency of parallel algorithms, it is also utilized to investigate the performance of DMA. To calculate the speedup ratio of DMA as well as examine its parallel efficiency, both serial running time and parallel running time are needed. The serial running time is obtained by running DMA with one SP. The parallel running time of DMA with different numbers of SPs (2, 4, 8, and 16) are also recorded. Note that the overall size of the population in each variant is uniformly set as 80. Running time and corresponding speedup ratios of DMA with different parallel granularity on various test cases are listed in Table SIV of the supplementary file. As we expect, the DMA with a larger number of SPs can achieve higher speedup ratios in all the test cases.

Fig. S8 of the supplementary file shows the variation curves of the DMA approach with different numbers of SPs on all of the test cases. It is clear that the speedup ratio of DMA increases with the parallel granularity of approach rises. In most of the test cases, the speedup ratio can close to the corresponding parallel granularity, which means the parallel efficiency of DMA is very high. Also, speedup ratios achieved on different test cases are of differences. This is because the ratio of computational time in the total running time varies on different test cases of different computation requirements.

High parallel efficiency of DMA is contributed by the island model of the island model utilized in DMA, which can help the SPs evolve independently and contributes to the entire population. Also, the implementation of DMA, which is based on the MPI parallel framework is effective in database fragmentation. The MPI framework is proved to be helpful in this kind of computation-driven optimization problem.

VI. Conclusion

Overall, to enhance database privacy and utility, a DMA has been proposed in this article. A BBRDF is designed to improve optimization efficiency. To enhance the global search, we propose a dynamic grouping recombination operator, two mutation operators, and a two-dimension selection approach. Moreover, an SDLS strategy is embedded in the approach to introduce rare utility elements without constraint concern. With the help of experiments, the proposed DMA showed significantly better performance than the existing approaches in database fragmentation and the effectiveness of each proposed framework and operators has been verified.

In the future, considering the effectiveness of the proposed operators in DMA, we will apply them to other discrete engineering optimization problems. Also, since the proposed distributed framework can enhance the information exchange between SPs as well as improve the algorithmic speed, it will be utilized in other optimization problems of the complex or large-scale property.

References

[1] Y. Yu *et al.*, "Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 4, pp. 767–778, Mar. 2017.

[2] J. Yu, G. Wang, Y. Mu, and W. Gao, "An efficient generic framework for three-factor authentication with provably secure instantiation," *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 12, pp. 2302–2313, Oct. 2014.

[3] V. Muntés-Mulero and J. Nin, "Privacy and anonymization for very large datasets," in *Proc. Conf. Inf. Knowl. Manag.*, 2009, pp. 2117–2118.

[4] D. Chen and H. Zhao, "Data security and privacy protection issues in cloud computing," in *Proc. IEEE Int. Conf. Comput. Sci. Electron. Eng.*, vol. 1, 2012, pp. 647–651.

[5] A. Khedr, G. Gulak, and V. Vaikuntanathan, "SHIELD: Scalable homomorphic implementation of encrypted data-classifiers," *IEEE Trans. Comput.*, vol. 65, no. 9, pp. 2848–2858, Nov. 2016.

[6] R. Jhawar, V. Piuri, and P. Samarati, "Supporting security requirements for resource management in cloud computing," in *Proc. IEEE Int. Conf. Comput. Sci. Eng.*, 2012, pp. 170–177.

[7] J. Köhler, K. Jünemann, and H. Hartenstein, "Confidential database-as-a-service approaches: Taxonomy and survey," *J. Cloud Comput.*, vol. 4, no. 1, p. 1, 2015.

[8] N. H. UbaidurRahman, C. Balamurugan, and R. Mariappan, "A novel DNA computing based encryption and decryption algorithm," *Procedia Comput. Sci.*, vol. 46, pp. 463–475, Jul. 2015.

[9] V. Ciriani, S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Fragmentation and encryption to enforce privacy in data storage," in *Computer Security (ESORICS)*. Singapore: Springer, 2007, pp. 171–186.

[10] C. Guo, R. Zhuang, Y. Jie, Y. Ren, T. Wu, and K.-K. R. Choo, "Fine-grained database field search using attribute-based encryption for e-healthcare clouds," *J. Med. Syst.*, vol. 40, no. 11, p. 235, 2016.

[11] R. Srinivas, K. Sireesha, and S. Vahida, "Preserving privacy in vertically partitioned distributed data using hierarchical and ring models," in *Artificial Intelligence and Evolutionary Computations in Engineering Systems*. Singapore: Springer, 2017, pp. 585–596.

[12] X. Xu, L. Xiong, and J. Liu, "Database fragmentation with confidentiality constraints: A graph search approach," in *Proc. ACM Conf. Data Appl. Security Privacy*, 2015, pp. 263–270.

[13] D. Nashat and A. A. Amer, "A comprehensive taxonomy of fragmentation and allocation techniques in distributed database design," *ACM Comput. Surveys*, vol. 51, no. 1, pp. 1–25, 2018.

[14] G. Aggarwal *et al.*, "Two can keep a secret: A distributed architecture for secure database services," in *Proc. 2nd Biennial Conf. Innov. Data Syst. Res.*, 2005, p. 2.

[15] V. Ciriani, S. D. C. Di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Fragmentation and encryption to enforce privacy in data storage," in *Proc. Eur. Symp. Res. Comput. Security*, 2007, pp. 171–186.

[16] V. Ciriani, S. D. C. D. Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Combining fragmentation and encryption to protect privacy in data storage," *ACM Trans. Inf. Syst. Security*, vol. 13, no. 3, p. 22, 2010.

[17] T. Rekatsinas, A. Deshpande, and A. Machanavajjhala, "SPARSI: Partitioning sensitive data amongst multiple adversaries," *Proc. VLDB Endow.*, vol. 6, no. 13, pp. 1594–1605, 2013.

[18] P. Berman and S. Raskhodnikova, "Approximation algorithms for min–max generalization problems," *ACM Trans. Algorithms*, vol. 11, no. 11, p. 5, 2014.

[19] V. Ciriani, S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Fragmentation design for efficient query execution over sensitive distributed databases," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2009, pp. 32–39.

[20] D. Gong, J. Sun, and Z. Miao, "A set-based genetic algorithm for interval many-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 47–60, Dec. 2018.

[21] M. Qiu, Z. Ming, J. Li, K. Gai, and Z. Zong, "Phase-change memory optimization for green cloud with genetic algorithm," *IEEE Trans. Comput.*, vol. 64, no. 12, pp. 3528–3540, Mar. 2015.

[22] S. Jiang and S. Yang, "An improved multiobjective optimization evolutionary algorithm based on decomposition for complex pareto fronts," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 421–437, Mar. 2016.

[23] C.-F. Juang, T.-L. Jeng, and Y.-C. Chang, "An interpretable fuzzy system learned through online rule generation and multiobjective ACO with a mobile robot control application," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 2706–2718, Oct. 2015.

[24] T. Back, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford, U.K.: Oxford Univ. Press, 1996.

[25] N. J. Radcliffe and P. D. Surry, "Formal memetic algorithms," in *Proc. AISB Workshop Evol. Comput.*, 1994, pp. 1–16.

[26] G. Zhang and Y. Li, "A memetic algorithm for global optimization of multimodal nonseparable problems," *IEEE Trans. Cybern.*, vol. 46, no. 6, pp. 1375–1387, Aug. 2015.

[27] J. Sun, Z. Miao, D. Gong, X.-J. Zeng, J. Li, and G. Wang, "Interval multiobjective optimization with memetic algorithms," *IEEE Trans. Cybern.*, vol. 50, no. 8, pp. 3444–3457, Aug. 2020, doi: 10.1109/TCYB.2019.2908485.

[28] S.-K. Song and N. Gorla, "A genetic algorithm for vertical fragmentation and access path selection," *Comput. J.*, vol. 43, no. 1, pp. 81–93, 2000.

[29] Y.-F. Ge *et al.*, "A benefit-driven genetic algorithm for balancing privacy and utility in database fragmentation," in *Proc. ACM Genet. Evol. Comput. Conf.*, 2019, pp. 771–776.

[30] J. Wu, X. Shen, and K. Jiao, "Game-based memetic algorithm to the vertex cover of networks," *IEEE Trans. Cybern.*, vol. 49, no. 3, pp. 974–988, Jan. 2018.

[31] W. Sheng, S. Chen, M. Sheng, G. Xiao, J. Mao, and Y. Zheng, "Adaptive multisubpopulation competition and multiniche crowding-based memetic algorithm for automatic data clustering," *IEEE Trans. Evol. Comput.*, vol. 20, no. 6, pp. 838–858, Feb. 2016.

[32] M. Tang and X. Yao, "A memetic algorithm for vlsi floorplanning," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 37, no. 1, pp. 62–69, Mar. 2007.

[33] J. Biskup, M. Preuß, and L. Wiese, "On the inference-proofness of database fragmentation satisfying confidentiality constraints," in *Proc. Int. Conf. Inf. Security*, 2011, pp. 246–261.

[34] M. R. Gibbs, G. Shanks, and R. Lederman, "Data quality, database fragmentation and information privacy," *Surveillance Soc.*, vol. 3, no. 1, pp. 45–58, 2005.

[35] L. Li, R. Lu, K.-K. R. Choo, A. Datta, and J. Shao, "Privacy-preserving-outsourced association rule mining on vertically partitioned databases," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 8, pp. 1847–1861, Mar. 2016.

[36] S. B. Navathe and M. Ra, "Vertical partitioning for database design: A graphical algorithm," *ACM SIGMOD Rec.*, vol. 18, no. 2, pp. 440–450, 1989.

[37] A. M. Tamhankar and S. Ram, "Database fragmentation and allocation: An integrated methodology and case study," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 28, no. 3, pp. 288–305, Dec. 1998.

[38] J. O. Hauglid, N. H. Ryeng, and K. Nørvåg, "DyFRAM: dynamic fragmentation and replica management in distributed database systems," *Distrib. Parallel Databases*, vol. 28, nos. 2–3, pp. 157–185, 2010.

[39] H. Rahimi, F.-A. Parand, and D. Riahi, "Hierarchical simultaneous vertical fragmentation and allocation using modified bond energy algorithm in distributed databases," *Appl. Comput. Informat.*, vol. 14, no. 2, pp. 127–133, 2018.

[40] A. Dahal and S. R. Joshi, "A comparative analysis on performance of minimum spanning tree and *k*-means clustering based vertical fragmentation algorithm," in *Proc. Artif. Intell. Transf. Bus. Soc. (AITB)*, 2019, pp. 1–7.

[41] P. Peng, L. Zou, L. Chen, and D. Zhao, "Adaptive distributed RDF graph fragmentation and allocation based on query workload," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 4, pp. 670–685, Jun. 2019.

[42] S. D. C. D. Vimercati, S. Foresti, S. Jajodia, G. Livraga, S. Paraboschi, and P. Samarati, "Loose associations to increase utility in data publishing," *J. Comput. Security*, vol. 23, no. 1, pp. 59–88, 2015.

[43] Y.-S. Ong, M.-H. Lim, N. Zhu, and K.-W. Wong, "Classification of adaptive memetic algorithms: A comparative study," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 36, no. 1, pp. 141–152, Apr. 2006.

[44] X. Chen, Y.-S. Ong, M.-H. Lim, and K. C. Tan, "A multi-facet survey on memetic computation," *IEEE Trans. Evol. Comput.*, vol. 15, no. 5, pp. 591–607, Oct. 2011.

[45] X. Zou and J. Liu, "A mutual information-based two-phase memetic algorithm for large-scale fuzzy cognitive map learning," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 4, pp. 2120–2134, Oct. 2018.

[46] T. Huang, Y.-J. Gong, S. Kwong, H. Wang, and J. Zhang, "A Niching memetic algorithm for multi-solution traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 24, no. 3, pp. 508–522, Jun. 2020.

[47] C.-C. Liao and C.-K. Ting, "A novel integer-coded memetic algorithm for the set *k*-cover problem in wireless sensor networks," *IEEE Trans. Cybern.*, vol. 48, no. 8, pp. 2245–2258, Apr. 2018.

[48] S. Susan and A. Bhutani, "A novel memetic algorithm incorporating greedy stochastic local search mutation for course scheduling," in *Proc. IEEE Int. Conf. Comput. Sci. Eng. (CSE) IEEE Int. Conf. Embedded Ubiquitous Comput. (EUC)*, 2019, pp. 254–259.

[49] J. Wang, W. Ren, Z. Zhang, H. Huang, and Y. Zhou, "A hybrid multiobjective memetic algorithm for multiobjective periodic vehicle routing problem with time windows," *IEEE Trans. Syst., Man, Cybern., Syst.*, early access, Aug. 17, 2018, doi: 10.1109/TSMC.2018.2861879.

[50] J. Wang, X. Chen, X. Huang, I. You, and Y. Xiang, "Verifiable auditing for outsourced database in cloud computing," *IEEE Trans. Comput.*, vol. 64, no. 11, pp. 3293–3303, Feb. 2015.

[51] M. A. Hadavi, E. Damiani, R. Jalili, S. Cimato, and Z. Ganjei, "AS5: A secure searchable secret sharing scheme for privacy preserving database outsourcing," in *Data Privacy Management and Autonomous Spontaneous Security*. Berlin, Germany: Springer, 2013, pp. 201–216.

[52] A. Bkakria, F. Cuppens, N. Cuppens-Boulahia, and J. M. Fernandez, *Confidentiality-Preserving Query Execution of Fragmented Outsourced Data* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2013, pp. 426–440.

[53] F. Liu, W. K. Ng, W. Zhang, D. H. Giang, and S. Han, "Encrypted set intersection protocol for outsourced datasets," in *Proc. IEEE Int. Conf. Cloud Eng.*, 2014, pp. 135–140.

[54] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

Yong-Feng Ge (Graduate Student Member, IEEE) received the B.S. degree in software engineering and the M.S. degree in computer science from Sun Yat-sen University, Guangzhou, China, in 2015 and 2018, respectively. He is currently pursuing the Ph.D. degree in computer science with the School of Computer Science and Engineering, South China University of Technology, Guangzhou.

His current research interests include evolutionary computation algorithms and their applications on real-world problems, such as database fragmentation.

Wei-Jie Yu (Member, IEEE) received the bachelor's and Ph.D. degrees in computer science from Sun Yat-sen University, Guangzhou, China, in 2009 and 2014, respectively.

He is currently an Associate Professor with the School of Information Management, Sun Yat-sen University. His current research interests include computational intelligence and its applications on intelligent information processing, big data, and cloud computing.

Jinli Cao received the Ph.D. degree in computer science from the University of Southern Queensland, Toowoomba, QLD, Australia, in 1997.

She is a full-time Senior Lecturer with the Department of Computer Science and Information Technology, La Trobe University, Melbourne, VIC, Australia. She has published over 115 research papers in international conferences and journals. She has been a Ph.D. supervisor with nine successful Ph.D. graduates. Her research is in the areas of data engineering and information systems, including big data analytics, recommendation systems, and data privacy protection.

Dr. Cao is an Associate Editor of *Health Information Science* and *Systems Journal* (Springer).

Hua Wang (Member, IEEE) received the Ph.D. degree from the University of Southern Queensland, Toowoomba, QLD, Australia.

He was a Professor with the University of Southern Queensland. He is currently a full-time Professor with Victoria University, Melbourne, VIC, Australia. He has more than ten years teaching and working experience in Applied Informatics at both enterprise and university. He has expertise in electronic commerce, business process modeling, and enterprise architecture. As an Chief Investigator, three Australian Research Council Discovery grants have been awarded since 2006, and 280 peer reviewed scholar papers have been published. Ten Ph.D. students have already graduated under his principal supervision.

Zhi-Hui Zhan (Senior Member, IEEE) received the bachelor's and Ph.D. degrees in computer science from Sun Yat-sen University, Guangzhou, China, in 2007 and 2013, respectively.

He is currently the Changjiang Scholar Young Professor with the School of Computer Science and Engineering, South China University of Technology, Guangzhou. His current research interests include evolutionary computation algorithms, swarm intelligence algorithms, and their applications in real-world problems, and in environments of cloud computing and big data.

Dr. Zhan's doctoral dissertation was awarded the IEEE Computational Intelligence Society Outstanding Ph.D. Dissertation and the China Computer Federation Outstanding Ph.D. Dissertation. He was a recipient of the Outstanding Youth Science Foundation from National Natural Science Foundations of China in 2018, and the Wu Wen-Jun Artificial Intelligence Excellent Youth from the Chinese Association for Artificial Intelligence in 2017. He is listed as one of the Most Cited Chinese Researchers in Computer Science. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, *Neurocomputing*, and the *International Journal of Swarm Intelligence Research*.

Yanchun Zhang (Member, IEEE) received the Ph.D. degree in computer science from the University of Queensland, Gatton, QLD, Australia, in 1991.

He is currently a Professor and the Director of the Applied Informatics Program (Centre), Victoria University, Melbourne, VIC, Australia. He has published over 350 research papers in international journals and conference proceedings. His research interests include big data, data mining, AI and health informatics (e-health).

Prof. Zhang is an Founding Editor and Editor-in-Chief of *Health Information Science and Systems Journal (Springer)* and *World Wide Web Journal* (Springer).

Jun Zhang (Fellow, IEEE) received the Ph.D. degree from the City University of Hong Kong, Hong Kong, in 2002.

He is currently a visiting scholar with Victoria University, Melbourne, VIC, Australia. His current research interests include computational intelligence, cloud computing, high-performance computing, operations research, and power electronic circuits.

Dr. Zhang was a recipient of the Changjiang Chair Professor from the Ministry of Education, China, in 2013, the China National Funds for Distinguished Young Scientists from the National Natural Science Foundation of China in 2011, and the First-Grade Award in Natural Science Research from the Ministry of Education, China, in 2009. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON CYBERNETICS, and the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS.