1-1-2020

D-Matrix: A Novel Ranking Procedure for Prioritizing Data Items

Lingping Kong
VSB – Technical University of Ostrava

Vaclav Snasel
VSB – Technical University of Ostrava

Swagatam Das
Indian Statistical Institute, Kolkata

Received July 5, 2020, accepted July 22, 2020, date of publication August 6, 2020, date of current version August 19, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3014871

D-Matrix: A Novel Ranking Procedure for Prioritizing Data Items

LINGPING KONG[1], VÁCLAV SNÁŠEL[1], (Senior Member, IEEE), AND SWAGATAM DAS[2], (Senior Member, IEEE)
[1]Faculty of Electrical Engineering and Computer Science, VŠB-Technical University of Ostrava, 708 00 Ostrava, Czech Republic
[2]Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata 700108, India

Corresponding author: Václav Snášel (vaclav.snasel@vsb.cz)

This work was supported in part by the ESF in Science without Borders Project, within the Operational Programme Research, Development and Education, under Grant CZ.02.2.69/0.0/0.0/16_027/0008463, and in part by the Ministry of Education, Youth and Sports of the Czech Republic through the Project Metaheuristics Framework for Multi-Objective Combinatorial Optimization Problems (META MO-COP) under Grant LTAIN19176.

ABSTRACT In this article, we propose a ranking method based on a matrix, called D-matrix, with the special identical diagonal values. This ranking system has five properties: (1) it can provide both biased and bias-free ranking, and except for that, the working matrix can be built in two ways: results merging and results separating for both biased and bias-free matrices. (2) it can perform the webpage ranking with a sparse matrix to generate ratings for pages instead of constructing complicated, irreducible, and stochastic matrices as the Google PageRank matrix does, thereby accelerating the computation speed. (3) this D-matrix has a solution no matter how much data is selected. If there are no comparisons among items, then all the items end up with the same equal ratings. (4) the ranking system has the least effects on data variation. If one item changes, only those connecting to it get different ratings, those without connection retain the same ratings. (5) this D-matrix has a $\ddot{R}$ support matrix with a delicate diagonal value which may or may not appear crucial. These five features are illustrated with five different and comprehensive examples. Besides that, a 2017 game of the National Football League data is tested where the D-matrix generates a reasonable result. Furthermore, we introduce a new approximate non-dominated sorting method based on D-matrix and thereby put forth a new algorithm for solving the multi-objective optimization problems. Experimental results indicate that our algorithm can maintain a better spread of solutions on many standard test functions.

INDEX TERMS Ranking, matrices, possibility theory, Markov processes, Google.

I. INTRODUCTION

To rank an object is to order objects based on their importance in a finite set of size n. A ranking system determines a way of assigning the ranks [1]. Usually, such a system uses relative information to produce a rating for each object [2]. A ranking list can be created by sorting the ratings [3]. Most of the ranking methods can be modified, extended, and adapted to rate other fields and competitive situations [4].

There are many ranking methods [5] using the paired comparison idea [6], [7] to design the fundamental model of object preference because most of the time it is easy to get the relative advantage considered two objects at a time, pairwise relationships are best portrayed with a matrix [8].

The associate editor coordinating the review of this manuscript and approving it for publication was Usama Mir.

That information is summarized in a corresponding matrix, and the objects' ratings can be produced by this matrix. For example, Massey [9] and Franceschet and Bozzo [10] uses the least-squares theory to develop his matrix ranking method, and his fundamental philosophy can be summarized as an idealized equation $r_i - r_j = y_k$, where r_i, r_j are items' ratings, and y_k is the margin of victory for game k. There is an equation of this form $Xr = y$, a system of linear equations, and players' numbers as unknowns. The final Massey matrix pattern is $Mr = p$, where $M = X^TX$, and $p = X^Ty$, while p is called cumulative point differentials, the $r_{n\times 1}$ is the vector of unknown rating. But a noteworthy property of Massey is that the columns of M are linearly dependent. This causes a non-unique solution. The workaround solution is to change the values of any row in M and p. Colley [11] and Lutzer [12] starts his ranking method with an idea from probability and specifically, the Laplace's law of succession, which possesses several advantages. Firstly, an avoidance of the pre-play ratings of $\frac{0}{0}$, secondly, if one loses at first play, the ratings produce a $\frac{1}{3}$ rating instead of 0. Further, all items start with equal ratings, one's gaining rating means another suffering a loss. The linear system of Colley can be written as $Cr = b$, where C is the Colley coefficient matrix, is invertible, thus, this system always has a unique solution. Keener's [13] method takes advantage of the special nature of the eigenvalues and eigenvectors in a matrix, and this method also suggests applying a nonlinear skewing function $h(x)$. Importantly, the skewing functions can be constructed by a user-specified design. The keystone of Keener's method is an equation $Ar = \lambda r$, where the rating vector r and λ are the eigenvector and associated eigenvalue for matrix A. PageRank [14] is developed for ranking web pages. Although Google's PageRank [15] ranking method belongs to the Markov Chain family, the ratings could be obtained using an iterative method and it still uses transition information to build the matrix. The web pages are identified as the objects, and the hyperlinks could be used to determine the order of preference [3]. To fully utilize the Markov chains theory [16], the original linking matrix should do some adjustments to handle with dangling node. The final PageRank's system can be written as $\pi^T = \pi^T G$, G matrix is irreducible, and stochastic, π is rating vector and is also an eigenvector of G corresponding to eigenvalue 1.

There are also some other kinds of ranking systems [17], [18]. Elo's system [19] was created for chess ranking and was approved by the United States Chess Federation. Yet, this method has became popular outside of the chess world. Elo's idea was to use a normally distributed random variable X to express a chess player's performance, and mean of X, denoted as μ, is essentially constant in the short-run. It can change only slowly with time. In other words, if a player performs one time better or worse should not change μ significantly. The equation for Elo's system is $r(new) = r(old) + K(S - \mu)$, where K is a constant and S is a value in $(1, 0, \frac{1}{2})$ for a win, lose, or a tie respectively. μ is the number of points that one expected to score against its opponent. Markov chains [20] are meant to describe stochastic processes or transition processes and were applied to analyze the sequence of vowels and consonants. The idea of Markov chains can be summarized as voting from a weaker object to a stronger object of every match-up. The system computes the stationary vector of this stochastic matrix S ($Sr = r$) to generate ratings, this rating vector shows the long-run proportion of time that an object taking a random walk on this chains spends with each object. But an undefeated object will cause this Markov matrix to be sub-stochastic, as this object makes a row of all zeros. Thus, there are many ways proposed to handle such problems, for example, assuming that the irreducible teleportation matrix $\bar{S}$, the stationary vector of $\bar{S}$ exists and is unique, $\bar{S} = \beta S + (1 - \beta)/nE$, where E is the matrix of all ones and n is the number of teams. Another ranking method is called Offense-Defense rating [21], which first rates individual attributes of each object, and then combines these strengths to produce a single number which reflects overall rating. The offense-Defense rating method separates the rating into offensive strength and defensive strength attribute values. The problem is, these two strength values have a circular relationship. Particularly, it must take the other into account while trying to separately rate offensive or defensive power. The definition of each strength rating is shown as: an object j's offensive rating is $o_j = \frac{a_{1j}}{d_1} + \frac{a_{2j}}{d_2} + \cdots + \frac{a_{mj}}{d_m}$, where $(d_1, d_2, \ldots, d_m)$ are defensive ratings. Similarly, for a given set of offensive ratings $(o_1, o_2, \ldots, o_m)$, define the defensive rating for team i to be $d_i = \frac{a_{i1}}{o_1} + \frac{a_{i2}}{o_2} + \cdots + \frac{a_{im}}{o_m}$, while a_{ij} is number of points that j scores against i for an offensive indicator and i held to j for a defensive indicator. Usually, initialize the defensive ratings with a column positive vector $(1, 1, \ldots, 1)$ values. A new idea for ranking by reordering methods is proposed [4], which moves the focus generating the results to the form from left to right, transforming input data into a ranking vector. One most apparent feature of this vector is that it is a permutation of the integers 1 through n that assigns a rank position to each object. Thus, each n length ranking vector r creates an $n \times n$ rank-differential matrix R, and each n length ascending order vector $\widehat{r}$ has a fundamental rank-differential matrix $\widehat{R}$. If we do a series of the row and column permutations to convert the matrix $\widehat{R}$ into the rank-differential matrix R, then we can get our ranking vector r after doing the same symmetric reordering to vector $\widehat{r}$.

All aforementioned methods have their merits. Some are easy to implement and incur low computation, some perform well in specific application fields. However, a general ranking method that can be compatible with both bias and bias-free ranking situations is less studied. Hence in this article, we propose our ranking method, which gives more appropriate and reasonable ranking results due to the complex features in data. This is also our goal for this article. The main contribution of this article is as follows:

- We design a D-matrix ranking method, which ranking idea extends to any set of items that need to be ranked.
- We give a detailed illustration of using D-matrix for ranking on different example data. In which examples, how to create a result merging matrix and a result separating matrix for both biased and bias-free patterns are introduced.
- A common web-page ranking example is tested on D-matrix, which shows a low computation of page ranking compared to PageRank, and it also shows the good stability of our ranking method.
- We show another perspective on verifying the correctness of the ranking method, in which we construct a variant version of *fundamental-rank-differential matrix*.
- National Football League game of the year 2017 data is also tested on D-matrix.
- We propose an approximate non-dominated sorting method based on D-matrix. The complexity of this

TABLE 1. **The transformation of matrix A to D^T.**

$$
A=\begin{bmatrix}
a_{1,1} & \cdots & a_{1,i} & \cdots & a_{1,j} & \cdots & a_{1,n}\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
a_{i,1} & \cdots & a_{i,i} & & a_{i,j} & \cdots & a_{i,n}\\
a_{j,1} & \cdots & a_{j,i} & & a_{j,j} & \cdots & a_{j,n}\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
a_{n,1} & & a_{n,i} & & a_{n,j} & & a_{n,n}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\frac{a_{1,1}}{a_1} & \cdots & \frac{a_{1,i}}{a_1} & \cdots & \frac{a_{1,j}}{a_1} & \cdots & \frac{a_{1,n}}{a_1}\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
\frac{a_{i,1}}{a_i} & \cdots & \frac{a_{i,i}}{a_i} & \cdots & \frac{a_{i,j}}{a_i} & \cdots & \frac{a_{i,n}}{a_i}\\
\frac{a_{j,1}}{a_j} & \cdots & \frac{a_{j,i}}{a_j} & \cdots & \frac{a_{j,j}}{a_j} & \cdots & \frac{a_{j,n}}{a_j}\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
\frac{a_{n,1}}{a_n} & & \frac{a_{n,i}}{a_n} & & \frac{a_{n,j}}{a_n} & & \frac{a_{n,n}}{a_n}
\end{bmatrix}
$$

$$
\rightarrow D^T
\begin{bmatrix}
\frac{a_{1,1}}{a_1} & \cdots & \frac{a_{i,1}}{a_i} & \cdots & \frac{a_{j,1}}{a_j} & \cdots & \frac{a_{n,1}}{a_n}\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
\frac{a_{i,1}}{a_1} & \cdots & \frac{a_{i,i}}{a_i} & \cdots & \frac{a_{j,i}}{a_j} & \cdots & \frac{a_{n,i}}{a_n}\\
\frac{a_{1,j}}{a_1} & \cdots & \frac{a_{i,j}}{a_i} & \cdots & \frac{a_{j,j}}{a_j} & \cdots & \frac{a_{n,j}}{a_n}\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
\frac{a_{1,n}}{a_1} & & \frac{a_{i,n}}{a_i} & & \frac{a_{j,n}}{a_j} & & \frac{a_{n,n}}{a_n}
\end{bmatrix}
$$

sorting method is evaluated, and the sorting accuracy is analyzed.

- We propose a multi-objective optimization algorithm, which adopts a novel candidate selection operator. A quite wide set of benchmark problems, including artificial datasets and real-world datasets, are tested on our proposed algorithm.

The rest of the paper is organized as follows: Section 2 is aimed at presenting the main idea and pattern of this new ranking method, D-matrix ranking system. Section 3 describes how the D-matrix works and its relative attributes. In Section 4, the advantages of D-matrix will be presented through five different useful examples, and how to create biased and bias-free with result-separating and result-merging D-matrices, how D-matrix solve the page ranking problem and accelerating the Google ranking in this section. Section 5 is devoted to showing the D-matrix system working at the 2017 NFL games ranking. A new elite based algorithm called ISDGA is proposed for solving a multi-objective problem based on an approximate non-dominated sorting process, which will be explained in detail in section 6. Finally, Section 7 presents the conclusions.

II. D-MATRIX RANKING

The main idea behind the **D**-matrix Method is all the ratings sum to 1, as (1).

$$r_1 + r_2 + \cdots + r_n = 1. \tag{1}$$

An $A_{n\times n}$ matrix with special diagonal value N is called the **D**-matrix, as (2) and (3).

$$A_{ij} = \begin{cases} N, & i = j, \\ n_{ij}, & i \neq j. \end{cases} \tag{2}$$

Here, N is the attribute number that depends on the value of $n_{i,j}$, $N > n_{i,j}$. $n_{i,j}$ is the number of winning times or scores teams i beats j. $r_{r\times 1}$ is a general rating vector produced by the **D**-Matrix system. n is the number of items of A. $e_{n\times 1}$ is the right-hand side vector, where $e = [1, \ldots, 1]^T$.

$$D^T r = n \times e, \quad e = [1, \ldots, 1]^T. \tag{3}$$

D is the row normalized matrix of A. $e = [1, \ldots, 1]^T$ is a n length vector with all ones. The r is the rating vector. As the final ratings result can be written as $r = D^{-1} \times e$, the same as $r = D^+ \times e$, where a pseudoinverse D^+ of a matrix D is a generalization of the inverse matrix [22]. That means during the utilization of D-matrix, D-matrix always has a solution. The relation of matrix A and matrix D is as (4). And the matrix transformation figure is as Table 1.

$$
a_i = \sum_{j=1}^{n} a_{ij}, \quad i \in \{1, 2, \ldots, n\},
$$
$$
D = A \times \begin{bmatrix}
\frac{1}{a_1} & & & & & \\
& \frac{1}{a_2} & & & & \\
& & \cdots & & & \\
& & & \frac{1}{a_i} & & \\
& & & & \cdots & \\
& & & & & \frac{1}{a_n}
\end{bmatrix} \tag{4}
$$

D-matrix is initially designed for rating individuals in the population of an evolutionary algorithm adopted for solving multi-objective optimization problems. In this situation, each individual represents one candidate solution and will be assigned several fitness function values. The concept of dominance is used in multi-objective optimization for evaluating the relationship between two items. In comparison, it is still necessary to rate two non-dominated items (with equalized status), for example, how many objective values are better than the other non-dominated individual, and how many objective values lose to the other. If it considers each objective as an attribute of an individual, then the objective number is the number of attributes. When an individual is comparing to itself, we could consider that it has a full number of attributes better than itself, or lose to itself.

In matrix A, each row shows the number of winning attributes to the other items. The value of $a_{i,i}$ is the i_{th} item wins attributes number to itself. $a_{i,j}$ is number of the i_{th} item beats the j_{th} item in attributes, and a_i is a row sum. Consequently, for the matrix D^T in linear system $D^T r = 1$,

TABLE 2. The $\widehat{R}_{n\times n}$ matrix and its variant $\ddot{R}_{n\times n}$.

$$\widehat{R}_{n\times n}= \begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ \text{n-1} \\ \text{n} \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & \cdots & n \end{array} \\ \left[\begin{array}{ccccc} 0 & 1 & 2 & \cdots & \text{n-1} \\ & 0 & 1 & \cdots & \text{n-2} \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & 1 \\ & & & & 0 \end{array}\right] \end{array} \qquad \ddot{R}_{n\times n}= \begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ \text{n-1} \\ \text{n} \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & \cdots & n \end{array} \\ \left[\begin{array}{ccccc} N & 1 & 2 & \cdots & \text{n-1} \\ & \text{N} & 1 & \cdots & \text{n-2} \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & 1 \\ & & & & N \end{array}\right] \end{array}$$

the i_{th} row has the equation $\frac{a_{i,1}}{a_1}r_1 + \cdots + \frac{a_{i,i}}{a_i}r_i + \cdots + \frac{a_{j,i}}{a_j}r_j + \cdots + \frac{a_{n,i}}{a_n}r_n = m$, where $r_1 + r_2 + \cdots + r_n = 1$. m is a constant number, r_i is the i_{th} item' winning probability, $\frac{a_{i,i}}{a_i}r_i$ is the expected value commonly used in the Probability Theory [23].

The D-matrix ranking method can pertain to both, biased as well as bias-free ranking systems [3], [4]. Here, bias-free is referred to as the method's ability to avoid the potential rating problem created when strong teams run up the score against weak teams. Colley uses the win-lose times to create a matrix, which is considered as a bias-free method, yet the Massey method, that uses game scores can be subject to such bias.

Thus, the value of $n_{i,j}$ can be the winning times for item i to item j, which will generate a bias-free matrix, and if the value takes gained scores, that will end up a bias matrix. The diagonal value N is called an attribute number because the value of N depends on what kind of attribute are compared between items. For instance, if the $n_{i,j}$ is item i winning times than item j, then N will be the match-ups number of all items. If the $n_{i,j}$ takes item i gained scores more than item j, then N should be assumed as the biggest scores one item gains than itself in all match-ups. We will use five examples to explain that.

Our ranking idea extends to any set of items that need to be ranked. Typically, all the case studies of ranking used in this article are well-accepted and taken from other authors' works.

III. HOW DOES D-MATRIX WORK?

Before discussing the example case-studies,let us observe a very interesting matrix $\ddot{R}$, as shown in Table 2. This is another matrix pattern of *fundamental-rank-differential matrix* $\widehat{R}$, which is from Book [4] in section 'Ranking by Reordering Methods of P_{98}'. The author notes that each $n \times n$ rank-differential matrix of length n vector is a symmetric reordering (i.e., a row and column permutation) of matrix $\widehat{R}$. Hence, we create our matrix $\ddot{R}$, which has the special diagonal value N and use the (4) to see how the $\vec{r}$ vector change with the varied N, as shown in Table 3.

Suppose matrix $\ddot{R}$ is the matrix A (Table 3), with $n = 10$ items, we use the (4) to generate the $\boldsymbol{r}$ vectors of $\ddot{R}$. The following Table 4 shows the varied vector $\boldsymbol{r}$ results along with the different N value.

TABLE 3. The basic matrix A.

$$\text{A} = \begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ 9 \\ 10 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & \cdots & 10 \end{array} \\ \left[\begin{array}{ccccc} N & 1 & 2 & \cdots & 9 \\ & \text{N} & 1 & \cdots & 8 \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & 1 \\ & & & & N \end{array}\right] \end{array}$$

The Table 4 is the $\boldsymbol{r}$ vector results of matrix A with different N value. As mentioned in [4], that every ranking vector of length n is a permutation of the integers 1 through n that assigns a rank position to each item. Then it is reasonable to think that the ranking system should generate the same order of ratings as the input team's data order. For instance, the first row of the matrix A is full of data, and the last row has only one N elements. If we consider row 1 is the best item, the last row is the worst, then this matrix after running the ranking system should produce a descending order of ratings.

As shown in the Table 4, in the case when $N = 1$, r vector varies from a positive value to negative value, then change to positive. When $N = 10$, the first time the system produces a descending r vector values, but it continue produces the exactly consistent order r vector values when the N value gets larger. It is worthy to note that, there are two questions here: Question 1. If the $N = n$, can it generate a correct result? The answer is 'yes or no'. Because if n is a big value, N should be bigger than n, then it can generate a correct answer. For example, if $n = 61$, the $N = 100$ then it mostly generates the correct order. Question 2. Do I have to guess what the N value is, the answer is no. The N value is predictable when doing the ranking process. As explained in the main idea of this D-matrix system, the idea behind is $r_a + \cdots + r_n = 1$, so the N value could be taken as the total winning times or winning scores in all competitions. We will explain how to set N value in the examples.

IV. ILLUSTRATIVE APPLICATIONS

In this section, five different examples are used for illustrating the construction process to show the different applications of D-matrix. All the examples are solved by the D-matrix method by biased and bias-free ways. The first example is an easy one; all the teams beat only once to each other. The second one is a movie ranking, and any two movies may be rated more than once. The third example is to show generating

TABLE 4. How does r vector change with different N.

	$N=1$	$N=8$	$N=9$	$N=10$	$N=11$	$N=20$	$N=50$	$N=100$	$N=200$	$N=1000$
r_1	46.0000	6.6250	6.0000	5.5000	5.0909	3.2500	1.9000	1.4500	1.2250	1.0450
r_2	-0.0000	4.8125	4.4444	4.1400	3.8843	2.6600	1.6856	1.3464	1.1741	1.0350
r_3	-29.0000	2.8828	2.7915	2.6980	2.6078	2.0460	1.4670	1.2417	1.1229	1.0249
r_4	-22.0000	1.1824	1.3123	1.3919	1.4403	1.4604	1.2524	1.1380	1.0720	1.0149
r_5	-0.0000	-0.0835	0.1723	0.3578	0.4956	0.9391	1.0477	1.0367	1.0217	1.0049
r_6	11.0000	-0.8565	-0.5734	-0.3542	-0.1810	0.5011	0.8565	0.9388	0.9722	0.9949
r_7	7.0000	-1.1982	-0.9628	-0.7674	-0.6038	0.1492	0.6802	0.8447	0.9236	0.9849
r_8	0	-1.2417	-1.0928	-0.9544	-0.8286	-0.1268	0.5181	0.7544	0.8761	0.9750
r_9	-2.0000	-1.1347	-1.0782	-1.0068	-0.9297	-0.3462	0.3676	0.6671	0.8293	0.9652
r_{10}	-1.0000	-0.9881	-1.0134	-1.0049	-0.9759	-0.5328	0.2248	0.5821	0.7831	0.9553

a matrix in a result-merging way and result-separating way. The fourth is a web page ranking example, in this example, the way D-matrix accelerating the computation of PageRank ranking and the unimportance of N value setting are shown. The last example is to showing the stability of the D-matrix system, in which the links among items vary and do not affect the rank results largely.

The first two examples show us that the D-matrix ranking system can work without the effect of the data amount. It means that even there are only a few useful collected data to items, the D-matrix method can provide a ranking order for the items.

A. EXAMPLE 1

First example Table 5, the data is from the 2005 NCAA football season of [4] in P_{23}. In this simple example, all teams played once with each other. We use this data to show how to construct our A matrix. Then with matrix A, do the row normalizing to get a matrix $\mathbf{D}$. Next, use equation (3) to get $\vec{r}$, which are the teams' ratings. At last, the teams' ranks can be get by sorting the ratings.

TABLE 5. Game score data for a small 5-team example.

	Duke	Miami	UNC	UVA	VT	Record	Point Differential
Duke		7-52	21-24	7-38	0-45	0-4	-124
Miami	52-7		34-16	25-17	27-7	4-0	91
UNC	24-21	16-34		7-5	3-30	2-2	-40
UVA	38-7	17-25	5-7		14-52	1-3	-17
VT	45-0	7-27	30-3	52-14		3-1	90

The D-matrix method can be used as a biased ranking or a bias-free ranking method. First, create the A matrix, the bias-free matrix A_1 is created by winning-times data, and the biased matrix A_2 will use the gained scores to build (see Table 6).

For matrix A_1, the first row is $(N_1, 0, 0, 0, 0)$, that means Duke team wins zero times (with four '0') to all other teams. Miami team beats every other team once so, the second row is $(1, N_1, 1, 1, 1, 1)$ with four '1' for Miami. The rest of rows will be UNC $(1, 0, N_1, 1, 0)$ with twice winnings, UVC $(1, 0, 0, N_1, 0)$ and VT $(1, 0, 1, 1, N_1)$.

Then for matrix A_2 (Table 6), the first row is $(N_2, 0, 0, 0, 0)$, which marks the gained scores of Duke to other teams. Miami team beats every other team once. The first play is with Duke, and the gained scores to Duke are $(52 - 7 = 45)$, so the entry is 45 in the second row of the first column. Miami gains $(34 - 16 = 18)$ scores to the UNC team, so the entry in the second row of the third column is 18. The rest rows entries will be gained scores of UNC UVC and VT to other teams.

After creating the matrix A, their ratings and ranks can be get by D-matrix equations. The result of Table 6 is shown as in Table 7. $rank_1$ is the bias-free ranking result of the matrix A_1, it only uses the winning or loses data, and $rank_2$ is a ranking result with bias, it uses the gained scores to rank.

TABLE 6. The bias-free A_1 and biased A_2 matrices for Table 5.

$$A_1 = \begin{array}{c|ccccc}
 & \text{Duke} & \text{Miami} & \text{UNC} & \text{UVA} & \text{VT} \\
\hline
\text{Duke} & N_1 & 0 & 0 & 0 & 0 \\
\text{Miami} & 1 & N_1 & 1 & 1 & 1 \\
\text{UNC} & 1 & 0 & N_1 & 1 & 0 \\
\text{UVA} & 1 & 0 & 0 & N_1 & 0 \\
\text{VT} & 1 & 0 & 1 & 1 & N_1
\end{array}$$

$$A_2 = \begin{array}{c|ccccc}
 & \text{Duke} & \text{Miami} & \text{UNC} & \text{UVA} & \text{VT} \\
\hline
\text{Duke} & N_2 & 0 & 0 & 0 & 0 \\
\text{Miami} & 45 & N_2 & 18 & 8 & 20 \\
\text{UNC} & 3 & 0 & N_2 & 2 & 0 \\
\text{UVA} & 31 & 0 & 0 & N_2 & 0 \\
\text{VT} & 45 & 0 & 27 & 38 & N_2
\end{array}$$

TABLE 7. The bias-free ratings ($rank_1$) and biased ratings ($rank_2$) for Table 5.

	Duke	Miami	UNC	UVA	VT
$ratings$	0.6561	1.4	0.972	0.8019	1.17
$rank_1$	5	1	3	4	2
$ratings$	-0.015	1.91	0.6342	0.7911	1.68
$rank_2$	5	1	4	3	2
$Colley$	5	1	3	4	2
$Massey$	5	1	4	3	2

The value is set as: $N_1 = 10$ for A_1 matrix and $N_2 = 100$ for A_2 matrix. For the matrix A_1, there are 10 games played. It can be understood as a team beats itself 10 times. As for the matrix A_2, the biggest score one team gained is 52 by Miami in this game data, so set N a larger number than 52. As expressed above, the diagonal value does affect the final ratings but does not affect the ranking orders as long as the N is big enough for its matrix. These 100 scores can be taken as any team beats itself by 100 scores. In Table 7, we list the results of Colley and Massey's rankings. As shown, the bias-free result of A_1 is the same as Colley's ranking result, and the biased A_2 result is the same as the Massey's ranking result.

B. MOVIE EXAMPLE 2

In the football game example 1, each team played once to each other. In this movie example, each movie may be rated together with others by different users many times.

In this movie example see Table 8 ([4], from P_{25}), each user rates the movie with a valid integer 1 through 5, and 5 is the best score. A zero means the user did not rate the movie. To create a biased matrix with winning times and a bias-free matrix with winning scores, we must choose the pair-wise matchups between movies. For instance, the first user (*user 1*) rates *movie-1, movie-2* and *movie-3*, no rate for *movie-4*. Hence, there is no win or lose between *movie-4* and other movies for *user 1*.

TABLE 8. A sample data of movie stars ratings.

$$U = \begin{array}{c}\\ \text{User 1}\\ \text{User 2}\\ \text{User 3}\\ \text{User 4}\\ \text{User 5}\\ \text{User 6}\end{array}\begin{array}{cccc} \text{Movie 1} & \text{Movie 2} & \text{Movie 3} & \text{Movie 4}\\ 5 & 4 & 3 & 0\\ 5 & 5 & 3 & 1\\ 0 & 0 & 0 & 5\\ 0 & 0 & 2 & 0\\ 4 & 0 & 0 & 3\\ 1 & 0 & 0 & 4 \end{array}$$

In Table 8, the matchup between *movie 1* and *movie-4* has three valid rates for use, which are *user-2* rates a '5' and a '1' for *movie- 1* and *movie-4* respectively, *user-5* rates a '5' and a '3' and *user-6* rates a '1' and a '4'. In A_4 see Table 1 it uses a result-merging way. All the rates from three users for *movie-1* are $5 + 4 + 1 = 10$, and the rates for *movie-4* are $1 + 3 + 4 = 8$. Hence, gained extra rates for *movie-1* is $10 - 8 = 2$, it shows $A_4[1, 4] = 2$.

The A matrix of this movie example is shown as in Fig. 1. $N_3 = 6$ and $N_4 = 16$ are set for matrix A_3 and A_4. In matrix A_3, $A_3[i, j]$ uses the win-lose times, it is a bias-free pattern. We use $N_3 = 6$ because there are six users evaluating movies, it can be explained that movie-1 beats itself six times. And for matrix A_4, $A_4[i, j]$ uses how many stars one movie gains more than other movies. We set $N_4 = 16$ because one movie can get the biggest star gap is four, and there are four movies, it is expressed as that one movie gains 16 stars more than itself in all competitions.

$$A_3 = \begin{array}{c|cccc} & \text{Movie 1} & \text{Movie 2} & \text{Movie 3} & \text{Movie 4}\\ \hline 1 & N_3 & 1 & 2 & 1\\ 2 & 0 & N_3 & 2 & 1\\ 3 & 0 & 0 & N_3 & 1\\ 4 & 0 & 0 & 0 & N_3 \end{array}$$

$$A_4 = \begin{array}{c|cccc} & \text{Movie 1} & \text{Movie 2} & \text{Movie 3} & \text{Movie 4}\\ \hline 1 & N_4 & 1 & 4 & 2\\ 2 & 0 & N_4 & 3 & 4\\ 3 & 0 & 0 & N_4 & 2\\ 4 & 0 & 0 & 0 & N_4 \end{array}$$

FIGURE 1. The biased A_4 and bias-free A_3 matrices of Example Table 8.

The result of this movie example is shown in Table 9. In this table, the results from Colley and Massey ranking are also shown with two patterns for each, $Colley_{withtie}$ and $Colley_{notie}$, $Massey_{withtie}$ and $Messey_{notie}$. 'tie' happens when two users rate the two movies with the same score. Notice that the result of Colley's ranking for this example changes according to the handling of ties. In biased and bias-free cases, the final ranking orders of D-matrix are right.

TABLE 9. The ratings of biased A_4 and bias-free A_3 matrices of Table 8.

	Movie 1	Movie 2	Movie 3	Movie 4
$ratings$	1.6667	1.25	0.4537	0.6296
$rank_1$	1	2	4	3
$ratings$	1.4375	1.3477	0.646	0.5688
$rank_2$	1	2	3	4
$Colley\_withtie$	1	2	4	3
$Massey\_withtie$	2	1	3	4
$Colley\_notie$	2	1	3	4
$Massey\_notie$	2	1	3	4

From the two examples above, we could see that in biased A_2, A_4 matrices, and bias-free A_1, A_3 matrices, there are rows having only one non-zero diagonal entry, such as row number one in $A_{1,2}$ and $A_{3,4}$ of row number four. Even there are many rows with zero inputs except for the diagonal value N, this matrix can get ratings $\vec{r}$, those items with zero rows end up getting the same ratings in results.

C. NFL EXAMPLE 3

This example ([3] from P_{25}) as in Table 10 shows another application of D-matrix. In the movie example-2, it shows the way of creating the result-merging matrix. In this example, both result-merging (*MM*) and result-separating (*MP*) ways are introduced. As shown in the last movie example, the movie-1 gets 10 stars from users and *movie-4* gets 8 stars from users. In A_4 matrix, the final record is 2 points for *movie-1* as in entry $A_4[1, 4] = 2$ and $A_4[4, 1] = 0$, which is a result-merging way. If the entries in A_4 were shown as

TABLE 10. Select NFL games from the 2007-2008 season.

Team	Score	Team	Score
Carolina	16	New Orleans	13
Dallas	38	Philadelphia	17
Dallas	28	Washington	23
Houston	34	Carolina	21
Houston	23	New Orleans	10
New Orleans	31	Carolina	6
Philadelphia	33	Washington	25
Philadelphia	38	New Orleans	23
Washington	27	Dallas	6
Washington	20	Philadelphia	12

TABLE 11. The biased and bias-free *A* matrices of Example Table 10.

$$MP_1 = \begin{array}{c|cccccc} & Car & New & Dall & Phi & Hous & Wash \\ \hline 1 & N_5 & 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & N_5 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & N_5 & 1 & 0 & 1 \\ 4 & 0 & 1 & 0 & N_5 & 0 & 1 \\ 5 & 1 & 1 & 0 & 0 & N_5 & 0 \\ 6 & 0 & 0 & 1 & 1 & 0 & N_5 \end{array} \qquad MP_2 = \begin{array}{c|cccccc} & Car & New & Dall & Phi & Hous & Wash \\ \hline 1 & N_6 & 3 & 0 & 0 & 0 & 0 \\ 2 & 25 & N_6 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & N_6 & 21 & 0 & 5 \\ 4 & 0 & 15 & 0 & N_6 & 0 & 8 \\ 5 & 13 & 13 & 0 & 0 & N_6 & 0 \\ 6 & 0 & 0 & 21 & 8 & 0 & N_6 \end{array}$$

$$MM_1 = \begin{array}{c|cccccc} & Car & New & Dall & Phi & Hous & Wash \\ \hline 1 & N_5 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & N_5 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & N_5 & 1 & 0 & 0 \\ 4 & 0 & 1 & 0 & N_5 & 0 & 0 \\ 5 & 1 & 1 & 0 & 0 & N_5 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & N_5 \end{array} \qquad MM_2 = \begin{array}{c|cccccc} & Car & New & Dall & Phi & Hous & Wash \\ \hline 1 & N_6 & 0 & 0 & 0 & 0 & 0 \\ 2 & 22 & N_6 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & N_6 & 21 & 0 & 0 \\ 4 & 0 & 15 & 0 & N_6 & 0 & 0 \\ 5 & 13 & 13 & 0 & 0 & N_6 & 0 \\ 6 & 0 & 0 & 16 & 0 & 0 & N_6 \end{array}$$

$A_4[1,4] = 10$ and $A_4[4,1] = 8$, this is the result-separating (*MP*) way.

The third example is from P_{25} [3] see Table 10. There are six teams and ten games were played. Four types of A matrices are shown in Table 11, MP_1, MP_2, MM_1 and MM_2. As usual, the biased ranking method is labeled in odd number as MP_1, MM_1, the bias-free way is MP_2, MM_2. For the sake of simplification in matrices, it uses a short of names, *Car* is short for *Carolina*, *New* is short for *New Orleans*, and so on. And the rating results of these four matrices are shown in Table 12. The results from Colley, Keener, and Massey rankings are also listed in this table. The four types of D-matrix ranking results are consistent with them respectively.

TABLE 12. The results of biased and bias-free matrices for Table 10.

		Car	New	Dall	Phi	Hous	Wash	
ratings MM_2	(	0.5087	0.7570	0.9656	0.9287	1.52	1.32	)
rank MM_2	(	5	6	3	4	1	2	)
ratings MP_2	(	0.5017	0.8002	0.9836	0.8669	1.52	1.32	)
rank MP_2	(	5	6	3	4	1	2	)
ratings MM_1	(	0.9	0.81	1.1	0.99	1.2	1.0	)
rank MM_1	(	5	6	2	4	1	3	)
ratings MP_1	(	0.9092	0.8083	1.1009	0.9908	1.2	0.9908	)
rank MP_1	(	6	5	2	3	1	3	)
Colley	(	5	6	2	4	1	3	)
Keener	(	6	5	2	3	4	1	)
Massey	(	6	5	2	3	4	1	)

D. PAGE RANK EXAMPLE 4

A small web graph is pictured in Fig. 2 [15]. The arrow from page *i* to page *j* is called a backlinking of page *j*, and this link is an outlink of page *i*. The PageRank method starts with a simple summation equation, one page's PageRank equals the sum of PageRanks of all pages' outlinks to it. It is assumed that, in the beginning, all pages have equal $1/n$ PageRank, and all the pages' PageRank will converge to constant numbers. But there is a little problem here before getting the result, the original equation cannot deal with dangling nodes, those pages with no outlinks creating 0 rows in the matrix. So the

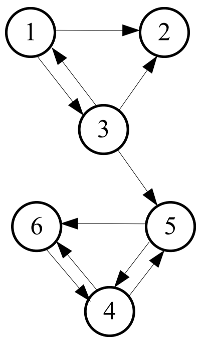


FIGURE 2. Directed graph for web of seven pages.

improvement for PageRank system is to create a stochastic and irreducible matrix for guaranteeing a positive PageRank vector existing, They use a *rank-one update* and *primitivity adjustment* to create a **G** matrix (Called Google Matrix), see (5) and (6).

Table 13 lists two methods to solve this page ranking example, Google Matrix method and D-matrix method. In table, matrix **H** is the fundamental matrix (raw hyperlink matrix) of PageRank system, **S** matrix (stochastic) is the row-normalized and one-rank updated of matrix **H**. Meanwhile A is the fundamental matrix of the D-matrix method, and **D** is a row-normalized matrix of A. The matrix **S** will be solved by Google Matrix way by (7) and (8). The matrix **D** will be solved by D-matrix by (3).

Table 14 shows the ratings and rank orders by Google's PageRank method and our D-Matrix method. The final π^T is generated by Google matrix **G** and $\vec{r}$ is generated by matrix **D** using (3). As shown in Table 14, *page-4* is ranked as the most important page by Google's PageRank method. By D-matrix method *page-2* is ranked as the most important page, because there is no outlink from *page-2*. Additionally, by Google's PageRank method, the process of constructing matrix **G** from matrix **H** and the computation of ratings of matrix **G** adding more complexity than the D-matrix ranking method. There are some quick computing algorithms for PageRank [24], which are fast. While here we use this traditional one to verify our correctness and show differences in computation

TABLE 13. The H-S-G matrices of PageRank and A-D matrices of D-Method.

$$H = \begin{array}{c|cccccc} & P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\ \hline P_1 & 0 & 1 & 1 & 0 & 0 & 0 \\ P_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ P_3 & 1 & 1 & 0 & 0 & 1 & 0 \\ P_4 & 0 & 0 & 0 & 0 & 1 & 1 \\ P_5 & 0 & 0 & 0 & 1 & 0 & 1 \\ P_6 & 0 & 0 & 0 & 1 & 0 & 0 \end{array} \qquad A = \begin{array}{c|cccccc} & P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\ \hline P_1 & 2 & 0 & 1 & 0 & 0 & 0 \\ P_2 & 1 & 2 & 1 & 0 & 0 & 0 \\ P_3 & 1 & 0 & 2 & 0 & 0 & 0 \\ P_4 & 0 & 0 & 0 & 2 & 1 & 1 \\ P_5 & 0 & 0 & 1 & 1 & 2 & 0 \\ P_6 & 0 & 0 & 0 & 1 & 1 & 2 \end{array}$$

$$S = \begin{array}{c|cccccc} & P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\ \hline P_1 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ P_2 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ P_3 & \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ P_4 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ P_5 & 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ P_6 & 0 & 0 & 0 & 1 & 0 & 0 \end{array} \qquad D = \begin{array}{c|cccccc} & P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\ \hline P_1 & \frac{2}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 \\ P_2 & \frac{1}{4} & \frac{2}{4} & \frac{1}{4} & 0 & 0 & 0 \\ P_3 & \frac{1}{3} & 0 & \frac{2}{3} & 0 & 0 & 0 \\ P_4 & 0 & 0 & 0 & \frac{2}{4} & \frac{1}{4} & \frac{1}{4} \\ P_5 & 0 & 0 & \frac{1}{4} & \frac{1}{4} & \frac{2}{4} & 0 \\ P_6 & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{4} & \frac{2}{4} \end{array}$$

TABLE 14. The final results of Google's PageRank and D-Matrix of Table 13.

		1	2	3	4	5	6	
$\pi^T =$	(	0.0372	0.05396	0.0415	0.375	0.206	0.286	)
$r =$	(	0.7	2.0	0.1	0.8	0.8	1.6	)
Page	(	P_1	P_2	P_3	P_4	P_5	P_6	)
PageRank	(	6	4	5	1	3	2	)
D-matrix	(	5	1	6	3	3	2	)

TABLE 15. The *r* results with different diagonal value *N* in D-Matrix of Table 13.

	(	P_1	P_2	P_3	P_4	P_5	P_6	)
N=2	(	0.7	2.0	0.1	0.8	0.8	1.6	)
D-matrix	(	5	1	6	3	3	2	)
N=3	(	0.7576	1.6667	0.3939	0.9091	0.9091	1.3636	)
D-matrix	(	5	1	6	3	3	2	)
N=5	(	0.8345	1.4	0.6276	0.9655	0.9655	1.2069	)
D-matrix	(	5	1	6	3	3	2	)
N=10	(	0.9092	1.2	0.8083	0.9908	0.9908	1.1	)
D-matrix	(	5	1	6	3	3	2	)
N=100	(	0.9901	1.02	0.9801	0.9999	0.9999	1.01	)
D-matrix	(	5	1	6	3	3	2	)

procedures.

$$a_i = \begin{cases} 1, & \textit{if } \mathbf{H}_i^T = 0 \\ 0, & \textit{otherwise} \end{cases} \tag{5}$$

$$\mathbf{S} = \mathbf{H} + (1/n)\mathbf{a}\mathbf{e}^T$$
$$\mathbf{G} = \alpha\mathbf{S} + (1-\alpha)\mathbf{e}\mathbf{v}^T \tag{6}$$

for $\alpha = 0.9$, the stochastic, primitive matrix $\mathbf{G}$ is

$$G = 0.9H + (0.9\begin{bmatrix}0\\1\\0\\0\\0\\0\end{bmatrix} + 0.1\begin{bmatrix}1\\1\\1\\1\\1\\1\end{bmatrix})\frac{1}{6}(1\;1\;1\;1\;1\;1)$$

$$= \begin{bmatrix} 1/60 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 19/60 & 19/60 & 1/60 & 1/60 & 19/60 & 1/60 \\ 1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15 \\ 1/60 & 1/60 & 1/60 & 7/15 & 1/60 & 7/15 \\ 1/60 & 1/60 & 1/60 & 11/12 & 1/60 & 1/60 \end{bmatrix} \tag{7}$$

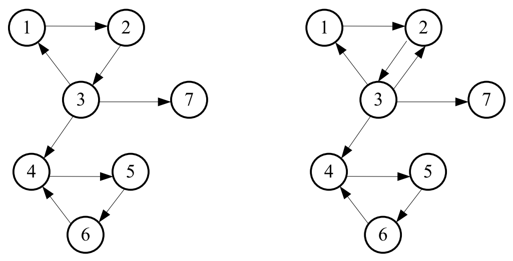
$$\pi^{(k+1)T} = \pi^{(k)T}\mathbf{G}$$

$$\mathbf{D}^T r = \mathbf{e} \quad \mathbf{e} = [1\;1\;\ldots\;1]_{n\times 1} \tag{8}$$

Another experiment is made for testing the impact of different diagonal N values on the result, as shown in Table 15. In this table, the $\vec{r}$ vector with different diagonal values as $\{N = 2, N = 3, N = 5, N = 10 \text{ and } N = 100\}$ are shown. From this experiment, it concludes that the D-matrix ranking system does change the ratings of each item when the diagonal value N varies, but it does not affect the final ranking orders.

E. PAGE RANK EXAMPLE 5

Another small web graph is pictured in Fig. 3 of P_{60} from [15]. In this figure, there is only one small difference between Fig. 3(a) and Fig. 3(b), that is the outlink from *page-3* to *page-2*.



(a) Web graph -1 (b) Web graph -2

FIGURE 3. Directed graph for web of seven pages.

In Table 16, W_1 is the A matrix for Fig. 3(a), and W_2 is the A matrix for Fig. 3(b). The W_2 and W_1 are quite similar, only one more '1' in large black size in W_2 than W_1, which is the outlink from *page-3* to *page-2*. It interprets as *page-2* wins ending up having this '1'.

TABLE 16. The D-matrix of seven pages graph of Fig. 3.

$$W_1 = \begin{array}{c|ccccccc} & P_1 & P_2 & P_3 & P_4 & P_5 & P_6 & P_7 \\ \hline P_1 & N & 0 & 1 & 0 & 0 & 0 & 0 \\ P_2 & 1 & N & 0 & 0 & 0 & 0 & 0 \\ P_3 & 0 & 1 & N & 0 & 0 & 0 & 0 \\ P_4 & 0 & 0 & 1 & N & 0 & 1 & 0 \\ P_5 & 0 & 0 & 0 & 1 & N & 0 & 0 \\ P_6 & 0 & 0 & 0 & 0 & 1 & N & 0 \\ P_7 & 0 & 0 & 1 & 0 & 0 & 0 & N \end{array}
\qquad
W_2 = \begin{array}{c|ccccccc} & P_1 & P_2 & P_3 & P_4 & P_5 & P_6 & P_7 \\ \hline P_1 & N & 0 & 1 & 0 & 0 & 0 & 0 \\ P_2 & 1 & N & \mathbf{1} & 0 & 0 & 0 & 0 \\ P_3 & 0 & 1 & N & 0 & 0 & 0 & 0 \\ P_4 & 0 & 0 & 1 & N & 0 & 1 & 0 \\ P_5 & 0 & 0 & 0 & 1 & N & 0 & 0 \\ P_6 & 0 & 0 & 0 & 0 & 1 & N & 0 \\ P_7 & 0 & 0 & 1 & 0 & 0 & 0 & N \end{array}$$

The $\vec{r}$ vector results with diagonal $\mathbf{N} = \mathbf{2}$ by D-matrix method for W_1 and W_2 are shown in Table 17. And it also lists the result from Google matrix **G** (the result is from book [15]) with $\alpha = 0.8$. Important note, there are only seven pages here, so the setting $N = 2$ is fine. If it is necessary for many webpages, N is better to be a larger number (it interprets one page wins itself in all links). In table, the first row shows the ratings for all pages from *page-1* (as P_1) to *page-7*, in which 0.7222 is the rating of *page-1*, 1.5556 is the rating of *page-2* (as P_2) for *web-1*; The second *rank* row shows the page rank order of pages, in which *page-2* ranks the first, *page-7* ranks the second; The third *order* row is the order of correct page order in ranks respectively, where 0.7222 for *page-1* is ranked in the sixth position, 1.5556 for *page-2* ranks in the first position.

Table 17 shows both the PageRank ranking method and the D-matrix method produce the same rank result for Fig. 3(a) and Fig. 3(b). But the rating variation is obvious for PageRank, as can be seen, the PageRank method produces totally different ratings to all pages for seven pages of Fig. 3(a) and Fig. 3(b), even if there is only one outlink adding to Fig. 3(b). However, as the D-matrix method result shows, adding a link only affects the ratings of the relative pages, as for *page-1*, *page-2*, and *page-3*, and the ratings of others stay the same.

TABLE 17. The *r* result with *N* = 2 of Table 16.

		P_1	P_2	P_3	P_4	P_5	P_6	P_7	
D-matrix results for webpage Fig. 3(a)									
web-1 =	(	0.7222	1.5556	-0.1111	1.3333	1.	1.	1.5	)
rank-1	(	2	7	4	$\frac{5}{6}$		1	3	)
order	(	6	1	7	3	4	4	2	)
D-matrix results for webpage Fig. 3(b)									
web-2 =	(	0.5	2.6667	-1	1.333	1.	1.	1.5	)
rank-2	(	2	7	4	$\frac{5}{6}$		1	3	)
order	(	6	1	7	3	4	4	2	)
PageRank results for webpage Fig. 3(a) [15]									
web-1 =	(	0.064	0.087	0.1056	0.237	0.2256	0.216	0.064	)
rank-1	(	6	5	4	1	2	3	6	)
order	(	4	5	6	3	2	$\frac{1}{7}$		)
PageRank results for webpage Fig. 3(b) [15]									
web-2 =	(	0.0736	0.1324	0.1429	0.194	0.1924	0.1909	.0736	)
rank-2	(	6	5	4	1	2	3	6	)
order	(	4	5	6	3	2	$\frac{1}{7}$		)

V. NFL GAMES FROM THE 2017 SEASON

D-matrix is also tested in this National Football League (NFL) game, in which the data is 2017 season scores combining from week 1 to week 17. There are 32 teams, each team played 16 times http://www.jt-sw.com/football/boxes/index.nsf/By/Season?OpenDocument&Season=2017. The result is compared with the famous used Colley and Massey ranking methods. The teams' full names and their abbreviation are shown as 'names' and 'abbr'.

names = ("Arizona Cardinals", "Atlanta Falcons", "Baltimore Ravens", "Buffalo Bills", "Carolina Panthers", "Chicago Bears", "Cincinnati Bengals", "Cleveland Browns", "Dallas Cowboys", "Denver Broncos", "Detroit Lions", "Green Bay Packers", "Houston Texans", "Indianapolis Colts", "Jacksonville Jaguars", "Kansas City Chiefs", "Los Angeles Chargers", "Los Angeles Rams", "Miami Dolphins", "Minnesota Vikings", "New England Patriots", "New Orleans Saints", "New York Giants", "New York Jets", "Oakland Raiders", "Philadelphia Eagles", "Pittsburgh Steelers", "San Francisco 49ers", "Seattle Seahawks", "Tampa Bay Buccaneers", "Tennessee Titans", "Washington Redskins");

abbr = ("ARZ", "ATL", "BAL", "BUF", "CAR", "CHI", "CIN", "CLE", "DAL", "DEN", "DET", "GB", "HOU", "IND", "JAC", "KC", "LAC", "LAR", "MIA", "MIN", "NE", "NO", "NYG", "NYJ", "OAK", "PHI", "PIT", "SF", "SEA", "TB", "TEN", "WAS");

Since the A matrix for this game data is large, only the final rank result is shown as in Table 18.

In Table 18, it lists six matrices ranking results including Colley, Massey (as the matrix of Massey is close to singular or badly scaled, the result may be inaccurate) and D-matrix in four types (D-1, biased and result-separating), (D-2, bias-free, and result-separating), (D-3, biased, and result-merging) and D-matrix (D-4, bias-free, and result-merging).

For simplicity and convenience, the team appearance order in The 'abbr' labeled as the digit orders, such as "ARZ-1","ATL-2","BAL-3","BUF-4", ..., "TB-30","TEN-31","WAS-32". The digit number in Table 18 records the rank order of each team. And the orders are from the lower rank to higher rank. As in the table, for Colley ranking, 8, 14, and 23 represented teams "CLE", "IND" and "NYG" correspondingly have the lowest ranks. While digits 26, 21, and 20 represented by teams "PHI", "NE" and "MIN" have the highest ranks. Though D-matrix and Colley methods get different ranking orders, there is slight vibration in results. The higher ranks' teams remain in higher ranks period and the lower ranks' teams are still controlled ahead of the middle ranks' teams. For example, teams ranked in the

TABLE 18. The rank results of NFL game of year 2017.

Colley ranking																	
order =	(	8	14	23	13	10	24	25	28	6	7	30	19	32	31	1	12
		3	17	15	29	9	4	11	16	18	2	27	22	5	26	21	20)
Massey ranking- (singular problem)																	
order =	(	5	6	31	25	26	28	13	16	22	20	3	19	2	32	12	18
		15	17	27	21	9	29	14	10	8	24	7	4	30	11	23	1)
D-1 ranking																	
order =	(	8	14	13	23	19	10	24	12	25	32	28	30	4	1	7	6
		31	11	9	5	2	29	16	3	27	17	22	18	20	26	15	21)
D-2 ranking																	
order =	(	8	23	14	13	30	6	24	10	25	28	19	12	32	7	1	11
		3	4	17	9	29	31	2	15	16	22	5	18	20	26	27	21)
D-3 ranking																	
order =	(	8	14	23	19	13	10	24	25	12	1	4	7	6	28	30	32
		31	9	11	5	2	29	16	17	3	27	22	20	18	15	26	21)
D-4 ranking																	
order =	(	8	23	14	13	30	6	24	10	25	28	19	12	32	7	1	4
		17	11	3	9	31	29	15	2	16	22	5	18	27	20	21	26)

last three from Colley are "CLE-8", "IND-14" and "NYG-23", those are in the last one, two, and four positions in the *D-1* ranking respectively. Similarly, teams ranked in first three from Colley of "PHI-26", "NE-21" and "MIN-20" get the former fourth, third, and first ranks in the D-matrix result.

VI. MULTI-OBJECTIVE OPTIMIZATION APPLICATION

Multiple objective optimization problem [25], [26] generates a set of optimal solutions representing the best trade-off among multiple objectives under various user preferences, instead of a single optimal solution. Usually, these optimal solutions are called Pareto-optimal solutions [27], in which no one solution can be noted to be better than the other [28]. A good method to solve a multi-objective problem is considered as it can be able to find as many optimal solutions as possible. Since evolutionary algorithms (EAs) operate with a population of individuals, a modified EA can be adapted to get diverse solutions [29]. While the selection of parents for producing the offspring is one of the main criticisms. Over the years, a lot of algorithms devoted to optimizing the points, such as high computational complexity of non-dominated sorting, the necessity of adjustment with relative parameters, etc.

In this section, we focus on the candidates' selection issues and propose an approximate non-dominated sorting (ANS) method based on D-Matrix. A multi-objective optimization method based on a genetic algorithm and a novel candidate selection operator is proposed.

A. D-ANS SORTING METHOD

In this section, we will introduce a new approximate non-dominated sorting method, called D-ANS.

The non-dominated sorting [30]–[34] strategy plays an important role in many multi-objective optimization methods, such as NSGAII [35], [36], SPEA [29], [37], and so on. Here, we introduce an approximate non-dominated sorting method that can separate most but not all non-dominated solutions from the population at each generation.

The A-ENS is one of the approximate non-dominated sorting methods proposed by Zhang *et al.* [38]. The motivation of A-ENS is to enhance the computational efficiency of dominance. It only performs at most three objective comparisons in determining the Pareto dominance relationship between two solutions. And it is embedded in three popular dominance-based evolutionary many-objective optimization algorithms. The results show a good performance. While, two types of sorting errors may happen in A-ENS, which decreases large accuracy of sorting. More information can be read from [38].

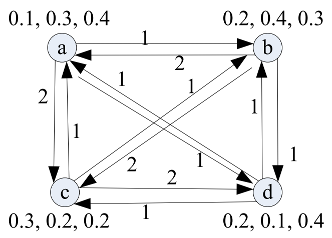
Note that D-ANS is a sorting method to sort a population of size N, the result of this sorting is to generate an order with exactly 1 to N rank number for solutions in population, not a front set result. It means the D-ANS gives importance (a rating value corresponds to a rank number) to each solution, the lower-ranked solution with a high rating, the more possibility it is a non-dominated solution.

In traditional non-dominated sorting approaches, a solution is considered as a non-dominated one even when it has only one best objective and all the rest of the objectives are worst [39]. If this kind of solution is chosen as the parents for producing the offsprings, the bad properties from parents will inevitably pass to the next generation. Consequently, this lowers the speed of convergence to elite solutions. The good solution sets should hold the ability to converge to the Pareto-optimal set, it is also desired that this set of solutions can be as spread as possible. This increases the requirement of maintaining the solutions with no bad objective values or with as much better objective values as possible. Hence, we propose an integrated sorting approach (called D-ANS) which focuses on integrating all objectives' competence rather than the final result of dominating.

The process of D-ANS includes: First, for each solution, we calculate superior objective value number than every solution. At last, create a matrix and calculate items' ratings by D-matrix ranking method. Suppose two solutions a and b of a problem with t objectives. Each entry $(AM_{a,b})$ in a matrix (AM) represents the superior number of objective values that solution a has compared to b. If solution a has i better objective values than solution b and solution b has $(t-i)$ better objective values than solution a, we have $AM_{ab} = i, AM_{ba} = t - i$. For entries AM_{aa} or AM_{bb}, it can be considered that solution a or b have t better objective values than itself, so $AM_{aa} = t$ or $AM_{bb} = t$, or as suggested in D-Matrix, those diagonal values could be set a larger number to distinguish rating values easily.

A n-size population with m objective number constructs a $n \times n$ level attribute matrix. We mentioned earlier that, as soon as a matrix is constructed, the unique ratings of solutions(items) can be generated by the D-Matrix ranking method. The population of solutions gets a rank order by sorting the ratings.

Here is a small example, Suppose a population of four solutions with three objectives (attributes), every objective is for minimization. The population are $\{a, b, c, d\}$, and the fitness values are: $a = (0.1, 0.3, 0.4), b = (0.2, 0.4, 0.3), c = (0.3, 0.2, 0.2), d = (0.2, 0.1, 0.4)$. The D-matrix for these four individuals in result-separating way would be like in Fig. 4. The left image is the links relations, the arrows point to the better one, and the weights are the amounts of better objectives. Matrix $P_{ab} = 2$ equals to an arrow b to a with weight value 2. There are only three objectives, so $P_{aa} = 3$. After creating this matrix P, the ratings of solutions *a, b, c, d* can be obtained by D-matrix ranking method. The order of ratings is the rank of solutions. To this example, the ratings are $a = 1.1351, b = 0.4865, c = 0.8649, d = 1.5135$. The order of importance of four solutions is d, a, c then b.



$$P = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \left[\begin{array}{cccc} 3 & 2 & 1 & 1 \\ 1 & 3 & 1 & 1 \\ 2 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \end{array} \right] \end{array}$$

FIGURE 4. An example for D-ANS process of four solutions.

This approximate non-dominated sorting method tries to find and rank most of the non-dominated solutions in front orders instead of finding an accurate non-dominated solution set.

1) ANALYSIS OF D-ANS COMPLEXITY

The D-ANS can be divided into the following two main steps:

- Construct a matrix with the information among solutions of better objective numbers.
- Use the D-matrix ranking method to generate ratings for solutions by solving this matrix.

In step one, each entry in a matrix (a) requires a comparison between two solutions. While one comparison, we get $a_{i,j}$ and $a_{j,i}$ two values simultaneously. As for $a_{i,i}$, it is a constant value, which is the objective number m. Hence the total computations are $\frac{(n-1)n}{2}$, the computation complexity is $\mathcal{O}(mn^2)$ where m is the number of objectives and n is the population size. And it requires $\mathcal{O}(n^2)$ space. The second step is the process of getting an inverse of a matrix and a multiplication with a matrix.

2) ANALYSIS OF D-ANS SORTING ACCURACY

As mentioned, this D-ANS approximate non-dominated sorting method tries to find solutions that are good in a specific definition. This is not an accurate sorting, which means a solution with a lower rank (better position) might not be a non-dominated solution or a higher ranked solution might be a non-dominated solution.

Remark 1: The following statements hold for D-ANS in ranking solutions with multi-objectives.

1. One solution with all the best objective values will get the highest rank. 2. One solution with all the worst objective values will get the lowest rank. 3. The more good objective values one solution gets, the lower rank this solution is sorted. In the same way, the higher the rank one solution sorted, the fewer good objective values one solution owns.

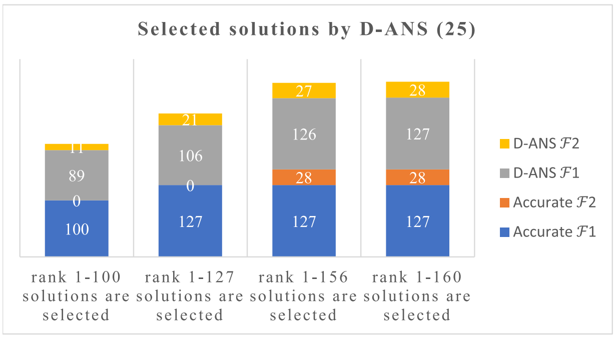
Since D-ANS is not an accurate sorting method, one might wonder how good is the sorting result. Two kinds of errors may be happening by D-ANS in performing non-dominated sorting.

1. One solution with only a few best objective values and many worst objective values (supposed to be a non-dominated solution) may end up in rear rank.

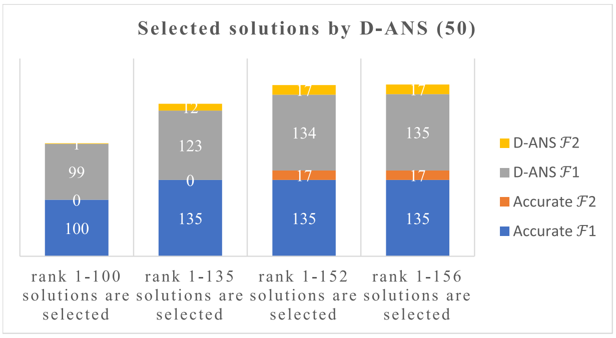
For example. Four solutions $a = (0.1, 0.4, 0.4, 0.6, 0.6)$, $b = (0.2, 0.2, 0.2, 0.2, 0.2)$, $c = (0.3, 0.3, 0.3, 0.3, 0.3)$ and solution $d = (0.2, 0.3, 0.3, 0.4, 0.3)$, then solution a is supposed to be non-dominated solution, but its rank is the highest one. Their rating values are $a = 0.2826, b = 2.2500, c = 0.6522, d = 0.8152$, the rank order is $[b\ d\ c\ a]$. This is because solution a has worse four objectives than other solutions.

2. One solution with many better objective values but each objective value is a bit worse than the best one may end up in front rank.

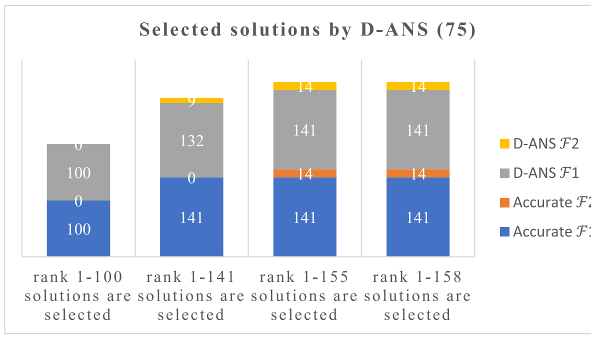
For example. Four solutions $a = (0.3, 0.4, 0.4, 0.6, 0.6)$, $b = (0.2, 0.2, 0.2, 0.2, 0.2)$, $c = (0.3, 0.3, 0.3, 0.3, 0.3)$ and solution $d = (0.6, 0.3, 0.5, 0.6, 0.1)$, then solution c is supposed to be dominated by solution b, however, the rank of solution c is second, ahead of solution d, which is actually a non-dominated solution. Their rating values are



(a) Population with 25 iteration



(b) Population with 50 iteration



(c) Population with 75 iteration

FIGURE 5. Analysis of sorting accuracy for Kursawe problem.

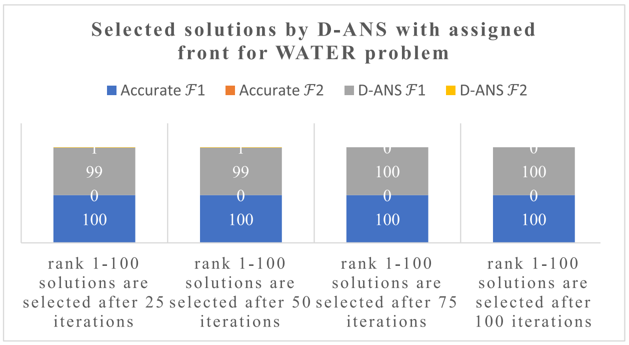


FIGURE 6. Analysis of sorting accuracy for WATER problem.

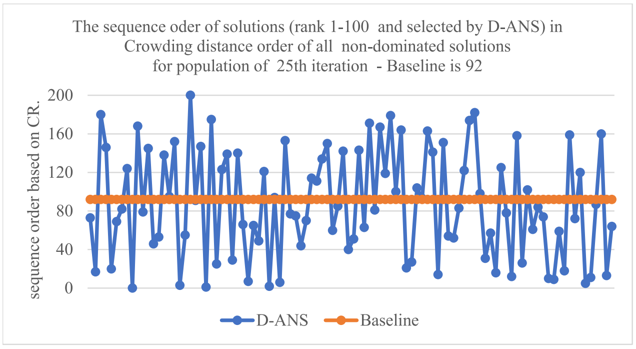
$a = 0.2938, b = 2.2982, c = 0.8139, d = 0.5941$. This is because solution c is better than solution d in average number of objectives.

Fig. 5 illustrates the accuracy of D-ANS sorting for test data from the Kursawe problem [40] (three variables, two objectives and real-coded) and Water problem [41] (five objectives and real-coded) of Fig. 6. The data (solutions) is randomly generated by jMetal framework and updated generation after generations. It means the data will include more non-dominated solutions with the iteration number increases.

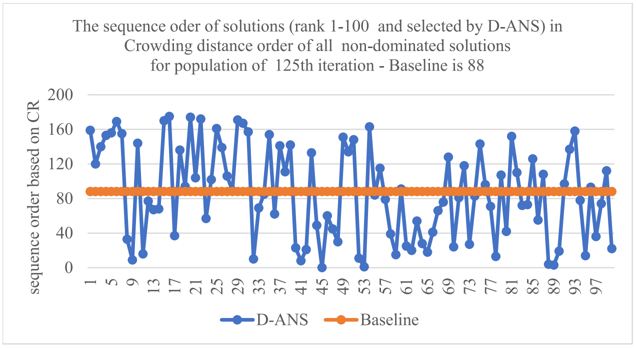
As mentioned, D-ANS is to give rank numbers to solutions to convey their importance. Hence, one way to evaluate the accuracy of D-ANS is to count the number of corresponding solutions in each $\mathcal{F}$ before the appearance of a fixed ranked solution. It can be illustrated as, suppose there are ten solutions sorted by D-ANS, then in the first half solutions ranked from the first to the fivth, calculate the number of non-dominated solutions in it, and the number of $\mathcal{F}_2$ solutions in it. In Fig. 5(a), $\mathcal{F}_1$ stands for the non-dominated solutions. The title 'Selected solution by D-ANS (25)' is to show a result of a population after 25 iteration times' updating from the initialization. The histogram with values is explained as: (i) the actual selecting result by D-ANS. (ii) the ideal selection result. For example, values with [11-89/ 0-100] are that it selects 100 [0 + 100] solutions which ranked from 1 to 100 by D-ANS from a population, in these selections by D-ANS, there are 89 solutions are non-dominated solutions and 11 solutions are belongs to $\mathcal{F}_2$. (ii) 0-100 means that the ideal result is a selection of 100 non-dominated solutions and zero $\mathcal{F}_2$ solution. The third histogram in Fig. 5(a) with values [27-126 / 28- 127]: (i) the total number selected solutions is 156 (28 + 127), and in them there are 127 non-dominated solutions and 28 $\mathcal{F}_2$ solutions as a true number. (ii) the number 27-126 means it selects 156 solutions which ranked from 1 to 156 by D-ANS, but there are only 126 solutions are non-dominated solutions and 27 solutions are actually from $\mathcal{F}_2$. The other two solutions are ranked in 156 but they belong to $\mathcal{F}_3$ or another front set. As shown in Fig. 5(a) the fourth histogram, all the non-dominated solutions can be selected when the selected solution number increases to 160.

In Fig. 5(b), the title 'Selected solution by D-ANS (50)' is to select several solutions from a population that have been updated 50 iterations after initialization. From the first to the fourth histogram, it shows that more and more non-dominated solutions in the population and it gets easier and accurate to select out of $\mathcal{F}_1$ solutions by D-ANS. When the iterations turn to 75 as Fig. 5(c), the first 100 ranked solutions are all non-dominated.

In Fig. 6, the title 'Selected solution by D-ANS with the assigned front for WATER problem' is also to select several solutions from the WATER problem population which population is updated with iterations. The WATER problem has five objectives, it means a solution is more likely to be a non-dominated solution. In Fig. 6, the solutions with lower ranks are nearly all non-dominated solutions even only 100 solutions are selected. But for a multi-objective



(a) Population with 25 iteration



(b) Population with 125 iteration

FIGURE 7. **Analysis of sorting diversity for WATER problem.**

optimization problem, the quality of a solution not only lies in the closeness to the Pareto front but in the degree of dispersion (diversity).

Fig. 7 is to show the diversity property of selected solutions. Firstly, count all the non-dominated solutions numbers from the population, and a baseline is a number of a half non-dominated solutions size. Secondly, calculate the crowding distance [35] value for all the non-dominated solutions, then sort these solutions by crowding distance (CR) value. Thirdly, each non-dominated solutions has a CR order. To be fair, if a solution is dominated one and it is selected by D-ANS, then it got the biggest order in CR. Fig. 7(a) shows an example of a population that has been updated 25 iterations from initialization. 100 solutions are selected with rank 1 to 100 by D-ANS from this population. The figure shows that almost half of the selected solutions by D-ANS have a better CR order than the baseline 92, and the other half of solutions have a worse CR order. It also shows that some of the CR orders of solutions are touching the zero line, which means those solutions having the biggest crowding distance values are the best solutions. Fig. 7(b) shows the same experiment with a different population. The baseline is 88, which means there are 176 non-dominated solutions in this population. As shown, the CR orders of selected solutions are scattered in a large space, which means the selected solutions by D-ANS have good diversity.

B. INTEGRATED SORTING D-MATRIX GENETIC ALGORITHM

In this section, we will describe a multi-objective optimization method based on D-ANS and genetic algorithm, which is called ISDGA.

1) APPROXIMATE SORTING PROCESS

The basic process of the Integrated Sorting D-matrix Genetic Algorithm (ISDGA) is based on a genetic algorithm, so the selection and crossover operation is inevitable While we adopt the approximate non-dominated sorting process instead of an accurate non-dominated sorting process to do the selection procedure. In this process, D-ANS will be used to select parents for producing offsprings.

2) ELITE SOLUTION PRESERVATION

In this section, we adopt an elite solution preservation strategy from OMOPSO [42], in which it utilizes a crowding distance parameter [35]. The crowding distance parameter is for density estimation, it can get an evaluation of the density surrounding a specific solution in the population. The crowding distance computation of one solution is a sum of all objectives' crowding extent. For each objective function value, the population requires a sorting order in this objective, and it is designed to assign an infinite distance value to the boundary solutions that hold the minimum and maximum objective values. All other intermediate solutions calculate their crowding extent equal to the absolute normalized difference of two adjacent solutions in this objective.

The elite solution preservation is executed after an integrated sorting to the population. So this procedure starts with the first individual with the highest ratings to at most the population size n end. This procedure conduces to a less number of solutions adding into and removing out of leaders. Each preserved solution is a non-dominated solution. It needed to be noted that, the number of individuals in leaders may be not equal to n, the size of the leader grows gradually with the simulation runs.

3) MAIN LOOP

Initially, a random first-generation population P is created as the basic genetic algorithm. Each individual presents a solution for the multi-objective problem, and each solution is evaluated by the objective functions. At first, the leaders set is empty, the parent population is selected from P to create offsprings with the crossover and mutation processes. Once there is a new-generation population, the D-ANS is used to make the population (new and old) a queue list. The leaders start to add an element from the beginning of the queue to the end. Since the leaders are not empty by then, the parent selection for creating offsprings will be picked from the leaders or population P in a fifty percent chance of each.

The procedure of the ISDGA algorithm is simple and straightforward. The following rules should be done during

The process of ISDGA

```
$P = \{p_1, \ldots, p_n\}, R = \{\}, leaders = \{\}.$   initialization status
for it =0 $\to$ $(iterationthreshold)$
    if leaders = null $||ran < \rho$
        $R = P \cup (P \to offsprings)$               offsprings are generated by $P$
    else
        $R = P \cup (leaders \to offsprings)$         offsprings are generated by $leaders$
    $R$ = D-ANS($R$)                                  rank $R$ using D-ANS
    for each solution in R
        $leaders = leaders.add(R_i)$                  if $R_i$ is nondominated status to solutions in leaders
    if size(leaders) $> n$
        crowding-distance-assignment(leaders)         assign crowding distance values to solutions in leaders.
        update(leaders)                               remove solutions with worst crowding values
    $emptyP, P = P \cup R_{[1-n]}$                    $P$ copy the first-half elements of $R$ to P.
for-end
```

TABLE 19. Properties of the 16 test problems. *M* and *D* denote the number of objectives and the number of decision variables, respectively. The type of design variables, the shape of the Pareto front, and other information are also described [45].

Name	Original name	M	D	Variables
RE2-4-1	Four bar truss design	2	4	Continuous
RE2-3-2	Reinforced concrete beam design	2	3	Mixed
RE2-4-3	Pressure vessel design	2	4	Mixed
RE2-2-4	Hatch cover design	2	2	Continuous Convex
RE2-3-5	Coil compression spring design	2	3	Mixed Mixed Disconnected
RE3-3-1	Two bar truss design	3	3	Continuous
RE3-4-2	Welded beam design	3	4	Continuous

TABLE 20. Properties of eight constrained test problems problems [45].

Name	Original name	M	D	N	Variables
CRE2-3-1	Two bar truss design		2	3 3	Continuous
CRE2-4-2	Welded beam design	2	4	4	Continuous
CRE2-4-3	Disk brake design	2	4	4	Continuous
CRE2-7-4	Speed reducer design	2	7	11	Mixed
CRE2-4-5	Gear train design	2	4	1	Integer

the whole process. Maximum space is assigned to the leaders, if there is a non-dominated solution and the leaders set has space, this solution should be added into the leaders. If a solution is dominated then it should be deleted from the leaders. If the non-dominated solutions' size exceeds the maximum threshold, a crowding distance parameter will be assigned to all non-dominated solutions, then the solutions that hold the minimum crowding distance will be discarded. The pseudocode of the ISDGA process is shown in "The process of ISDGA".

C. SIMULATION RESULTS

In this section, we use a Java framework for multi-objective optimization tool, which is called jMetal [43], to simulate the proposed ISDGA algorithm. jMetal contains several classic and effective optimizers, a quite wide set of benchmark problems, and well-known performance assessment indicators. The framework also includes support to carry out comprehensive and one's own experimental studies.

The test problems used to compare the performance are ZDT [44] and WFG citeWFG. The experiments also test UF4, Golinski, Srinivas, Tanaka, Osyczka2, Binh, along with Kursawe, Water Vinnnet3 ConstrEx, and Fonseca, all those test functions are embedded in Jmetal Framework. Except for this, we also test our algorithms on some real-world problems suite consisting of bound-constrained problems and multi-objective mixed-integer optimization problems. Those problems are available in an off-the-shelf manner provided by [45].

We use the four classic and well-used algorithms NSGAII [35], SPEA2 [29], MOEA/D [46], [47], Randomsearch as the comparison targets, actually these four comparison algorithms are all very good at some certain points and perform very well. The same parameter settings are set for all algorithms as suggested in jMetal. We set a basic maximum of 25000 solution evaluation, 100 population size, 1/ l mutation probability (l is the number of decision variables), and 0.9 as crossover probability. A ρ is set to 0.3 in the experiment for ISDGA.

The performance measure [43] used in this article includes Epsilon, Spread, IGD, and deltaP, for each is in two aspects: mean & standard deviation and median & IQR, more information about the evaluation could be seen in jMetal. Hypervolume(HV) calculates the volume covered by optimal solutions, which is not measured because HV is not suitable for evaluating multi-objective problems. Epsilon computes the distance-related measure between one solution to the Pareto front. Spread is also noted as $\triangle$, which measures the spread extent of the obtained solutions. IGD calculates an average minimum distance from each point in optimal points (reference) to those in obtained solutions, which measures both convergence and diversity of solution set. A smaller IGD value indicates a better convergence as well as diversity [48]. deltaP is also named deltaIGD, which is an inverted generational distance metric [43].

Table 21 to Table 30 shows the experimental results of five algorithms. To easily analyze these results, a gray-colored background has been used to identify outstanding solutions. Particularly, in each row, a darker gray colors the best value in a specific indicator, and the second-best value is highlighted by a lighter gray background in the same indicator. For EPSILON and SPREAD indicators (parameter), the lower of one value is, the better it performs. And

TABLE 21. SPREAD. Mean and standard deviation.

	ISDGA	NSGAII	MOEAD	RandomSearch	SPEA2
WFG3	$3.69e-01_{4.9e-03}$	$5.74e-01_{1.4e-02}$	$5.63e-01_{5.3e-04}$	$6.61e-01_{2.8e-02}$	$4.34e-01_{6.2e-03}$
WFG4	$1.37e-01_{7.5e-03}$	$4.04e-01_{3.2e-02}$	$5.30e-01_{2.4e-02}$	$6.16e-01_{4.4e-02}$	$2.73e-01_{1.1e-02}$
WFG5	$1.38e-01_{1.8e-02}$	$4.20e-01_{3.4e-02}$	$4.64e-01_{2.9e-03}$	$6.33e-01_{5.1e-02}$	$2.73e-01_{2.2e-02}$
WFG6	$1.08e-01_{1.9e-02}$	$3.82e-01_{2.8e-02}$	$4.12e-01_{8.7e-04}$	$6.53e-01_{3.2e-02}$	$2.59e-01_{1.3e-02}$
ZDT1	$7.24e-02_{6.4e-03}$	$3.63e-01_{2.6e-02}$	$3.85e-01_{5.5e-02}$	$8.47e-01_{3.1e-02}$	$1.55e-01_{6.5e-03}$
ZDT2	$1.11e-01_{1.3e-02}$	$4.04e-01_{3.8e-02}$	$2.75e-01_{6.0e-02}$	$9.08e-01_{1.7e-02}$	$1.61e-01_{1.1e-02}$
ZDT3	$7.07e-01_{3.8e-03}$	$7.44e-01_{1.4e-02}$	$9.57e-01_{2.4e-02}$	$8.68e-01_{2.2e-02}$	$7.06e-01_{4.3e-03}$
ZDT4	$3.02e-01_{7.6e-02}$	$3.91e-01_{3.5e-02}$	$9.60e-01_{5.5e-02}$	$9.40e-01_{4.1e-02}$	$2.58e-01_{1.1e-01}$
ZDT6	$1.50e-01_{8.6e-03}$	$3.55e-01_{2.8e-02}$	$1.51e-01_{8.1e-04}$	$9.38e-01_{2.1e-02}$	$2.29e-01_{1.3e-02}$
UF4	$5.00e-01_{8.1e-02}$	$5.01e-01_{2.3e-02}$	$4.21e-01_{5.8e-02}$	$6.20e-01_{5.0e-02}$	$5.05e-01_{3.5e-02}$
Golinski	$2.00e-01_{4.3e-02}$	$4.67e-01_{3.7e-02}$	$1.05e+00_{2.4e-02}$	$7.50e-01_{1.2e-01}$	$7.23e-01_{2.1e-02}$
Srinivas	$7.96e-02_{9.5e-03}$	$3.75e-01_{3.5e-02}$	$6.37e-01_{6.4e-04}$	$6.03e-01_{1.5e-02}$	$1.76e-01_{1.5e-02}$
Tanaka	$7.94e-01_{4.4e-02}$	$8.14e-01_{7.4e-03}$	$1.00e+00_{0.0e+00}$	$7.05e-01_{4.6e-02}$	$7.61e-01_{2.7e-02}$
Osyczka2	$6.19e-01_{1.2e-01}$	$5.76e-01_{4.6e-02}$	$9.48e-01_{2.1e-03}$	$7.05e-01_{1.5e-01}$	$6.74e-01_{7.8e-02}$
Binh2	$4.28e-01_{1.1e-02}$	$5.86e-01_{1.3e-02}$	$8.34e-01_{6.9e-04}$	$8.17e-01_{5.6e-03}$	$4.23e-01_{3.8e-03}$
Kursawe	$4.16e-01_{6.8e-03}$	$5.81e-01_{2.4e-02}$	$7.29e-01_{4.5e-03}$	$5.63e-01_{5.1e-02}$	$4.41e-01_{8.9e-03}$
Water	$5.56e-01_{1.3e-02}$	$5.47e-01_{2.3e-02}$	$1.19e+00_{7.5e-02}$	$5.38e-01_{4.4e-03}$	$5.18e-01_{2.1e-02}$
Viennet3	$6.41e-01_{6.6e-02}$	$7.64e-01_{2.2e-02}$	$1.16e+00_{5.0e-03}$	$1.18e+00_{3.4e-02}$	$7.93e-01_{3.2e-02}$
ConstrEx	$1.44e-01_{1.3e-02}$	$4.48e-01_{2.5e-02}$	$7.73e-01_{1.1e-03}$	$5.37e-01_{3.6e-02}$	$5.13e-01_{3.2e-02}$
Viennet4	$8.87e-01_{5.1e-02}$	$9.83e-01_{1.4e-02}$	$9.99e-01_{1.4e-02}$	$7.65e-01_{1.9e-02}$	$7.91e-01_{3.7e-02}$
Fonseca	$8.49e-02_{8.1e-03}$	$3.96e-01_{1.7e-02}$	$1.47e-01_{5.5e-04}$	$5.47e-01_{7.5e-02}$	$1.44e-01_{1.4e-02}$

TABLE 22. SPREAD. Median and IQR.

	ISDGA	NSGAII	MOEAD	RandomSearch	SPEA2
WFG3	$3.66e-01_{1.0e-02}$	$5.66e-01_{2.2e-02}$	$5.63e-01_{1.1e-03}$	$6.61e-01_{4.8e-02}$	$4.31e-01_{1.3e-02}$
WFG4	$1.39e-01_{1.4e-02}$	$4.18e-01_{6.7e-02}$	$5.34e-01_{5.0e-02}$	$6.13e-01_{8.5e-02}$	$2.75e-01_{2.3e-02}$
WFG5	$1.32e-01_{3.7e-02}$	$4.14e-01_{6.5e-02}$	$4.64e-01_{6.4e-03}$	$6.15e-01_{1.1e-01}$	$2.60e-01_{4.4e-02}$
WFG6	$1.00e-01_{2.8e-02}$	$3.77e-01_{5.8e-02}$	$4.12e-01_{1.8e-03}$	$6.60e-01_{6.3e-02}$	$2.62e-01_{2.7e-02}$
ZDT1	$7.55e-02_{1.3e-02}$	$3.63e-01_{4.7e-02}$	$3.80e-01_{9.9e-02}$	$8.57e-01_{6.2e-02}$	$1.53e-01_{1.4e-02}$
ZDT2	$1.15e-01_{2.3e-02}$	$3.83e-01_{7.9e-02}$	$2.67e-01_{1.2e-01}$	$9.11e-01_{3.5e-02}$	$1.58e-01_{2.3e-02}$
ZDT3	$7.04e-01_{7.1e-03}$	$7.46e-01_{2.6e-02}$	$9.57e-01_{4.6e-02}$	$8.59e-01_{3.6e-02}$	$7.05e-01_{7.4e-03}$
ZDT4	$3.27e-01_{1.2e-01}$	$3.82e-01_{5.5e-02}$	$9.49e-01_{9.6e-02}$	$9.57e-01_{7.7e-02}$	$2.09e-01_{2.3e-01}$
ZDT6	$1.56e-01_{1.8e-02}$	$3.53e-01_{5.5e-02}$	$1.51e-01_{1.6e-03}$	$9.26e-01_{4.4e-02}$	$2.25e-01_{2.9e-02}$
UF4	$5.26e-01_{1.6e-01}$	$5.05e-01_{5.0e-02}$	$4.36e-01_{1.3e-01}$	$6.12e-01_{9.1e-02}$	$5.18e-01_{6.6e-02}$
Golinski	$1.88e-01_{7.8e-02}$	$4.63e-01_{7.9e-02}$	$1.06e+00_{4.1e-02}$	$7.94e-01_{2.0e-01}$	$7.22e-01_{4.5e-02}$
Srinivas	$8.23e-02_{2.0e-02}$	$3.67e-01_{7.6e-02}$	$6.37e-01_{1.3e-03}$	$6.08e-01_{3.0e-02}$	$1.82e-01_{2.4e-02}$
Tanaka	$7.93e-01_{9.5e-02}$	$8.14e-01_{1.4e-02}$	$1.00e+00_{0.0e+00}$	$7.38e-01_{9.6e-02}$	$7.57e-01_{5.3e-02}$
Osyczka2	$6.95e-01_{2.1e-01}$	$5.92e-01_{9.6e-02}$	$9.49e-01_{4.1e-03}$	$7.02e-01_{2.8e-01}$	$6.74e-01_{1.6e-01}$
Binh2	$4.32e-01_{2.0e-02}$	$5.92e-01_{2.8e-02}$	$8.34e-01_{1.1e-03}$	$8.14e-01_{1.2e-02}$	$4.21e-01_{7.4e-03}$
Kursawe	$4.17e-01_{1.5e-02}$	$5.71e-01_{4.8e-02}$	$7.27e-01_{9.2e-03}$	$5.60e-01_{1.1e-01}$	$4.36e-01_{1.6e-02}$
Water	$5.58e-01_{2.8e-02}$	$5.51e-01_{3.9e-02}$	$1.21e+00_{1.1e-01}$	$5.40e-01_{8.5e-03}$	$5.16e-01_{3.9e-02}$
Viennet3	$6.04e-01_{1.4e-01}$	$7.67e-01_{4.7e-02}$	$1.16e+00_{1.1e-02}$	$1.17e+00_{6.9e-02}$	$7.98e-01_{5.8e-02}$
ConstrEx	$1.51e-01_{2.7e-02}$	$4.60e-01_{4.9e-02}$	$7.72e-01_{1.9e-03}$	$5.26e-01_{5.6e-02}$	$5.00e-01_{6.7e-02}$
Viennet4	$9.00e-01_{1.0e-01}$	$9.75e-01_{2.6e-02}$	$1.00e+00_{2.7e-02}$	$7.53e-01_{3.6e-02}$	$7.92e-01_{7.5e-02}$
Fonseca	$8.56e-02_{1.7e-02}$	$3.99e-01_{3.2e-02}$	$1.46e-01_{9.0e-04}$	$5.81e-01_{1.5e-01}$	$1.45e-01_{2.9e-02}$

another thing needed to be pointed, in the jMetal framework of version 4.5, the random seed starts with a systematic time, this causes little disparity in the same experiment. Differently, the seed with one is set in all studies in our experiment.

With the help of the gray-colored elaboration in some cells, it is easy to tell that the ISDGA algorithm performs better in SPREAD and EPSILON indicators, especially on ZDT, WFG, and other real-world application test functions. Though In other cases, IGDGA might not be in the first place of performance, IGDGA gets more colored areas than compared algorithms in all. MOEAD gets most of the dark grayed areas in EPSILON indicators on traditional test functions but it fails in real-world application function at SPREAD indicators and IGD, deltaP indicators. SPEA2 performs very well at deltaP and IGD indicators on real-world application test functions but fails in SPREAD performance. Meanwhile, SPEA2 does not work very well on some traditional test functions on SPREAD and EPSILON indicators.

The experiments above are implemented under the same maximum evaluation for each algorithm, in which the algorithm will give back the result once the algorithm reaches this threshold. While as the process for generating an offspring differs from varied algorithms, the evaluation costing time for each iteration might be in a great difference. Actually, SPEA2 and our ISGDA algorithms take more time than others. Here we provide a rough time comparison among algorithms in one problem Kursawe for one run. Total execution time: ISGDA is 3112ms; SPEA2 is 5164ms; MOEAD is 578ms; RandomSearch is 204ms; NSGAII is 1231ms; SPEA2 takes a complex computation process for a solution. ISGDA costs more time on matrix computation,

TABLE 23. EPSILON. Mean and standard deviation.

	ISDGA	NSGAII	MOEAD	RandomSearch	SPEA2
WFG3	$2.00e+00_{2.0e-03}$	$2.00e+00_{9.2e-04}$	$2.00e+00_{1.0e-04}$	$2.29e+00_{2.0e-02}$	$2.00e+00_{1.9e-03}$
WFG4	$1.52e-02_{4.0e-04}$	$3.51e-02_{6.5e-03}$	$4.97e-02_{4.4e-03}$	$4.55e-01_{1.5e-01}$	$2.61e-02_{2.4e-03}$
WFG5	$6.32e-02_{4.9e-04}$	$8.33e-02_{1.8e-03}$	$7.29e-02_{2.2e-04}$	$2.89e-01_{2.0e-02}$	$7.25e-02_{1.2e-03}$
WFG6	$3.43e-02_{2.2e-02}$	$5.38e-02_{1.3e-02}$	$2.42e-02_{2.8e-04}$	$3.61e-01_{3.8e-02}$	$2.71e-02_{3.8e-03}$
ZDT1	$6.63e-03_{2.1e-04}$	$1.20e-02_{9.2e-04}$	$2.60e-02_{2.4e-03}$	$1.78e+00_{1.4e-01}$	$9.20e-03_{9.9e-04}$
ZDT2	$7.00e-03_{5.1e-04}$	$1.30e-02_{2.3e-03}$	$3.06e-02_{9.6e-03}$	$3.41e+00_{1.2e-01}$	$9.09e-03_{4.7e-04}$
ZDT3	$6.14e-03_{7.6e-04}$	$8.57e-03_{1.2e-03}$	$1.31e-01_{1.3e-02}$	$2.10e+00_{4.3e-02}$	$7.16e-02_{1.2e-01}$
ZDT4	$8.74e-02_{5.8e-02}$	$1.63e-02_{2.4e-03}$	$4.30e-01_{2.2e-01}$	$4.83e+01_{3.3e+00}$	$5.18e-02_{5.3e-02}$
ZDT6	$1.40e-02_{5.8e-04}$	$1.48e-02_{1.2e-03}$	$4.94e-03_{1.4e-04}$	$6.33e+00_{2.2e-01}$	$2.76e-02_{4.2e-03}$
UF4	$8.25e-02_{3.4e-03}$	$6.63e-02_{3.6e-03}$	$9.44e-02_{4.2e-03}$	$1.37e-01_{1.1e-03}$	$7.78e-02_{4.7e-03}$
Golinski	$6.13e+00_{7.1e-01}$	$9.09e+00_{1.8e+00}$	$2.42e+00_{1.3e-01}$	$2.16e+02_{4.3e+01}$	$1.79e+01_{6.6e+00}$
Srinivas	$1.27e+00_{3.3e-02}$	$3.10e+00_{6.3e-01}$	$3.25e+00_{3.9e-02}$	$2.44e+00_{2.7e-01}$	$1.85e+00_{1.1e-01}$
Tanaka	$1.08e-02_{2.2e-03}$	$8.88e-03_{7.7e-04}$	$-4.40e-02_{4.7e-10}$	$2.85e-02_{4.6e-03}$	$8.27e-03_{7.7e-04}$
Osyczka2	$2.24e+01_{1.2e+01}$	$7.22e+00_{7.0e+00}$	$1.07e+01_{2.2e-01}$	$5.23e+01_{8.0e+00}$	$3.94e+01_{4.4e+01}$
Binh2	$4.06e-01_{1.1e-02}$	$9.13e-01_{1.0e-01}$	$4.79e-01_{2.2e-03}$	$9.22e-02_{9.2e-03}$	$8.64e-01_{7.6e-02}$
Kursawe	$5.03e-02_{3.3e-03}$	$8.11e-02_{1.3e-02}$	$8.10e-02_{7.6e-03}$	$9.87e-01_{1.6e-01}$	$6.68e-02_{5.4e-03}$
Water	$3.73e+04_{7.1e+03}$	$7.08e+04_{1.6e+04}$	$1.16e+06_{2.0e+04}$	$1.97e+04_{2.6e+03}$	$6.73e+04_{9.4e+03}$
Viennet3	$3.85e-02_{5.7e-03}$	$4.63e-02_{1.2e-02}$	$5.22e-02_{2.7e-04}$	$2.72e-02_{4.9e-03}$	$4.67e-02_{4.6e-03}$
ConstrEx	$8.07e-03_{1.1e-03}$	$1.66e-02_{4.3e-03}$	$4.45e-03_{1.9e-05}$	$2.64e-02_{4.9e-03}$	$2.86e-02_{3.9e-03}$
Viennet4	$3.52e-01_{1.3e-02}$	$4.60e-01_{3.0e-02}$	$2.19e-01_{5.3e-03}$	$3.43e-01_{1.0e-02}$	$3.49e-01_{1.9e-02}$
Fonseca	$6.70e-03_{7.7e-04}$	$1.43e-02_{1.6e-03}$	$6.35e-03_{2.1e-05}$	$7.58e-02_{6.2e-03}$	$8.86e-03_{1.4e-03}$

TABLE 24. EPSILON. Median and IQR.

	ISDGA	NSGAII	MOEAD	RandomSearch	SPEA2
WFG3	$2.00e+00_{4.4e-03}$	$2.00e+00_{2.0e-03}$	$2.00e+00_{1.7e-04}$	$2.28e+00_{3.9e-02}$	$2.00e+00_{3.5e-03}$
WFG4	$1.50e-02_{8.1e-04}$	$3.49e-02_{1.1e-02}$	$5.13e-02_{8.5e-03}$	$5.67e-01_{2.9e-01}$	$2.54e-02_{5.2e-03}$
WFG5	$6.33e-02_{9.1e-04}$	$8.45e-02_{3.4e-03}$	$7.30e-02_{4.0e-04}$	$2.99e-01_{4.1e-02}$	$7.32e-02_{2.1e-03}$
WFG6	$2.76e-02_{3.3e-02}$	$4.99e-02_{2.5e-02}$	$2.41e-02_{5.8e-04}$	$3.62e-01_{8.3e-02}$	$2.60e-02_{7.3e-03}$
ZDT1	$6.48e-03_{4.3e-04}$	$1.22e-02_{1.9e-03}$	$2.65e-02_{5.1e-03}$	$1.74e+00_{2.8e-01}$	$9.36e-03_{1.8e-03}$
ZDT2	$6.69e-03_{1.0e-03}$	$1.38e-02_{4.6e-03}$	$2.98e-02_{1.7e-02}$	$3.40e+00_{2.3e-01}$	$9.04e-03_{9.3e-04}$
ZDT3	$5.90e-03_{1.3e-03}$	$8.34e-03_{2.5e-03}$	$1.26e-01_{2.1e-02}$	$2.09e+00_{8.6e-02}$	$1.01e-02_{1.5e-01}$
ZDT4	$6.95e-02_{1.1e-01}$	$1.54e-02_{5.1e-03}$	$2.91e-01_{4.5e-01}$	$4.98e+01_{5.8e+00}$	$2.44e-02_{9.7e-02}$
ZDT6	$1.38e-02_{1.2e-03}$	$1.54e-02_{1.8e-03}$	$4.87e-03_{2.2e-04}$	$6.37e+00_{4.8e-01}$	$2.66e-02_{9.3e-03}$
UF4	$8.28e-02_{6.0e-03}$	$6.69e-02_{7.4e-03}$	$9.49e-02_{7.4e-03}$	$1.37e-01_{2.3e-03}$	$7.57e-02_{9.1e-03}$
Golinski	$6.36e+00_{1.1e+00}$	$8.91e+00_{3.6e+00}$	$2.45e+00_{2.1e-01}$	$1.93e+02_{8.1e+01}$	$1.79e+01_{1.2e+01}$
Srinivas	$1.26e+00_{6.5e-02}$	$2.89e+00_{1.3e+00}$	$3.25e+00_{8.2e-02}$	$2.41e+00_{5.4e-01}$	$1.86e+00_{2.0e-01}$
Tanaka	$1.04e-02_{4.8e-03}$	$8.79e-03_{1.4e-03}$	$-4.40e-02_{0.0e+00}$	$3.11e-02_{9.5e-03}$	$8.50e-03_{1.3e-03}$
Osyczka2	$3.05e+01_{2.4e+01}$	$1.87e+00_{1.4e+01}$	$1.08e+01_{3.6e-01}$	$5.13e+01_{1.6e+01}$	$1.98e+01_{6.9e+01}$
Binh2	$4.08e-01_{2.0e-02}$	$8.45e-01_{2.0e-01}$	$4.78e-01_{4.9e-03}$	$8.76e-02_{1.9e-02}$	$8.48e-01_{1.6e-01}$
Kursawe	$4.99e-02_{6.5e-03}$	$7.76e-02_{2.2e-02}$	$7.86e-02_{1.6e-02}$	$9.70e-01_{3.0e-01}$	$6.68e-02_{1.0e-02}$
Water	$3.58e+04_{1.4e+04}$	$6.57e+04_{3.0e+04}$	$1.16e+06_{3.5e+04}$	$1.96e+04_{5.1e+03}$	$6.47e+04_{2.1e+04}$
Viennet3	$3.97e-02_{1.2e-02}$	$4.52e-02_{2.6e-02}$	$5.22e-02_{5.9e-04}$	$2.56e-02_{1.0e-02}$	$4.65e-02_{9.1e-03}$
ConstrEx	$8.38e-03_{2.4e-03}$	$1.52e-02_{8.7e-03}$	$4.45e-03_{3.2e-05}$	$2.53e-02_{7.8e-03}$	$2.80e-02_{8.5e-03}$
Viennet4	$3.48e-01_{2.6e-02}$	$4.67e-01_{5.3e-02}$	$2.20e-01_{1.1e-02}$	$3.38e-01_{2.0e-02}$	$3.42e-01_{2.8e-02}$
Fonseca	$6.85e-03_{1.7e-03}$	$1.47e-02_{3.3e-03}$	$6.36e-03_{3.5e-05}$	$7.37e-02_{1.0e-02}$	$8.15e-03_{2.9e-03}$

TABLE 25. SPREAD. Mean and standard deviation.

	ISDGA	NSGAII	MOEAD	RandomSearch	SPEA2
RE21	$7.20e-02_{1.2e-02}$	$3.95e-01_{4.9e-02}$	$1.06e+00_{1.7e-03}$	$6.35e-01_{8.3e-02}$	$2.96e-01_{1.4e-02}$
RE22	$9.23e-01_{6.5e-01}$	$5.19e-01_{1.7e-01}$	$3.68e-01_{1.5e-02}$	$1.07e+00_{2.6e-01}$	$6.19e-01_{4.7e-01}$
RE23	$3.51e-01_{2.3e-02}$	$5.72e-01_{2.8e-02}$	$1.70e+00_{3.4e-02}$	$8.34e-01_{5.7e-02}$	$9.89e-01_{2.1e-02}$
RE24	$1.37e-01_{1.3e-02}$	$4.17e-01_{1.6e-02}$	$1.39e+00_{1.1e-03}$	$9.37e-01_{6.6e-02}$	$7.07e-01_{1.5e-02}$
RE25	$1.56e+00_{1.1e-04}$	$1.56e+00_{7.7e-05}$	$1.02e+00_{2.2e-03}$	$5.89e-01_{1.3e-01}$	$1.56e+00_{9.6e-04}$
RE31	$8.44e-01_{3.0e-02}$	$8.87e-01_{4.3e-02}$	$1.32e+00_{2.9e-02}$	$8.91e-01_{2.4e-02}$	$5.33e-01_{2.4e-02}$
RE32	$2.83e-01_{1.8e-01}$	$5.54e-01_{4.2e-02}$	$1.09e+00_{1.3e-03}$	$1.18e+00_{1.1e-01}$	$7.25e-01_{1.7e-01}$
CRE21	$2.94e-01_{1.2e-01}$	$3.95e-01_{3.2e-02}$	$1.30e+00_{1.4e-02}$	$9.11e-01_{4.1e-02}$	$9.86e-01_{2.3e-03}$
CRE22	$8.86e-02_{1.7e-02}$	$4.23e-01_{2.6e-02}$	$1.13e+00_{7.1e-03}$	$1.01e+00_{7.0e-02}$	$9.85e-01_{2.1e-03}$
CRE23	$1.11e-01_{1.2e-01}$	$4.12e-01_{2.0e-02}$	$1.26e+00_{5.0e-04}$	$8.99e-01_{1.7e-01}$	$2.92e-01_{1.6e-01}$
CRE24	$1.70e-01_{4.3e-02}$	$4.18e-01_{2.2e-02}$	$1.11e+00_{2.5e-02}$	$8.48e-01_{1.3e-01}$	$5.55e-01_{1.7e-02}$
CRE25	$4.05e-01_{1.3e-02}$	$1.45e+00_{4.8e-02}$	$1.29e+00_{6.8e-03}$	$5.59e-01_{1.0e-01}$	$1.48e+00_{4.4e-02}$

which could be optimized by Intel's special Advanced Vector Extensions instructions [49] or other well-designed vector coding programs.

The integrated sorting form ISDGA is a step to find solutions with high ratings, those high ratings are interpreted as their objective values are much better than others

TABLE 26. SPREAD. Median and IQR.

	ISDGA	NSGAII	MOEAD	RandomSearch	SPEA2
RE21	$6.78e-02_{1.9e-02}$	$4.06e-01_{8.2e-02}$	$1.06e+00_{3.5e-03}$	$6.21e-01_{1.5e-01}$	$2.92e-01_{2.5e-02}$
RE22	$1.44e+00_{1.3e+00}$	$4.38e-01_{3.0e-01}$	$3.58e-01_{3.1e-02}$	$9.42e-01_{5.1e-01}$	$5.64e-01_{8.7e-01}$
RE23	$3.56e-01_{4.8e-02}$	$5.63e-01_{5.7e-02}$	$1.71e+00_{6.5e-02}$	$8.24e-01_{9.9e-02}$	$9.92e-01_{4.0e-02}$
RE24	$1.46e-01_{2.3e-02}$	$4.16e-01_{3.3e-02}$	$1.39e+00_{2.4e-03}$	$9.65e-01_{1.4e-01}$	$7.13e-01_{2.9e-02}$
RE25	$1.56e+00_{2.0e-04}$	$1.56e+00_{1.6e-04}$	$1.02e+00_{3.8e-03}$	$5.69e-01_{2.7e-01}$	$1.56e+00_{1.3e-03}$
RE31	$8.37e-01_{6.3e-02}$	$8.88e-01_{8.7e-02}$	$1.30e+00_{5.9e-02}$	$8.79e-01_{5.2e-02}$	$5.47e-01_{4.0e-02}$
RE32	$1.97e-01_{2.4e-01}$	$5.53e-01_{9.3e-02}$	$1.09e+00_{2.6e-03}$	$1.19e+00_{1.9e-01}$	$7.76e-01_{3.5e-01}$
CRE21	$3.09e-01_{2.0e-01}$	$3.90e-01_{6.5e-02}$	$1.30e+00_{3.0e-02}$	$8.97e-01_{8.1e-02}$	$9.86e-01_{4.8e-03}$
CRE22	$7.99e-02_{2.9e-02}$	$4.28e-01_{5.3e-02}$	$1.12e+00_{1.2e-02}$	$1.04e+00_{1.5e-01}$	$9.85e-01_{3.8e-03}$
CRE23	$5.72e-02_{1.6e-01}$	$4.16e-01_{4.1e-02}$	$1.26e+00_{1.1e-03}$	$9.25e-01_{3.7e-01}$	$1.98e-01_{3.0e-01}$
CRE24	$1.83e-01_{9.1e-02}$	$4.23e-01_{4.8e-02}$	$1.10e+00_{4.4e-02}$	$8.47e-01_{2.3e-01}$	$5.50e-01_{3.1e-02}$
CRE25	$4.12e-01_{1.6e-02}$	$1.46e+00_{9.2e-02}$	$1.29e+00_{1.5e-02}$	$5.61e-01_{2.2e-01}$	$1.49e+00_{8.6e-02}$

TABLE 27. IGD. Mean and standard deviation.

	ISDGA	NSGAII	MOEAD	RandomSearch	SPEA2
RE21	$8.19e-02_{7.3e-04}$	$1.15e-01_{1.2e-02}$	$1.32e+01_{2.4e-02}$	$4.88e-01_{4.3e-02}$	$8.34e-02_{7.7e-04}$
RE22	$4.12e-02_{2.0e-03}$	$5.90e-02_{2.7e-03}$	$1.65e-01_{1.5e-02}$	$1.20e-01_{8.5e-03}$	$4.25e-02_{9.0e-04}$
RE23	$1.76e+02_{5.1e+00}$	$2.94e+02_{8.6e+00}$	$8.00e+03_{1.3e+02}$	$6.08e+03_{1.4e+03}$	$1.22e+02_{1.4e+00}$
RE24	$5.03e-02_{1.2e-03}$	$7.46e-02_{1.2e-02}$	$1.42e+00_{1.8e-03}$	$1.81e-01_{1.1e-01}$	$4.44e-02_{1.5e-03}$
RE25	$5.66e-05_{3.6e-07}$	$9.20e-05_{2.7e-06}$	$2.99e+03_{0.0e+00}$	$2.67e+02_{4.7e+01}$	$5.86e-05_{6.0e-07}$
RE31	$1.29e+04_{1.3e+03}$	$1.13e+04_{3.6e+02}$	$2.14e+05_{1.9e+02}$	$2.14e+05_{1.2e+00}$	$7.04e+03_{1.8e+02}$
RE32	$4.36e+04_{2.3e+03}$	$7.30e+04_{1.7e+04}$	$4.99e+06_{1.3e+01}$	$4.48e+06_{4.0e+05}$	$2.83e+04_{4.1e+02}$
CRE21	$1.09e+04_{9.3e+03}$	$1.13e+03_{4.7e+01}$	$8.50e+04_{1.4e-01}$	$8.50e+04_{9.9e+00}$	$4.36e+02_{8.7e+00}$
CRE22	$1.87e+00_{3.0e-02}$	$2.47e+00_{1.3e-01}$	$1.81e+02_{1.3e-01}$	$1.77e+02_{6.0e+00}$	$1.05e+00_{4.8e-03}$
CRE23	$3.96e-04_{6.3e-06}$	$5.63e-04_{1.5e-05}$	$4.52e-03_{1.5e-05}$	$7.01e-03_{7.1e-04}$	$4.34e-04_{1.2e-05}$
CRE24	$1.19e+00_{1.0e+00}$	$4.41e-01_{4.8e-02}$	$1.11e+01_{1.4e+00}$	$3.61e+00_{6.0e-01}$	$3.53e-01_{7.2e-02}$
CRE25	$2.15e-02_{4.0e-03}$	$1.51e-02_{6.4e-03}$	$2.97e-02_{6.9e-03}$	$4.61e-02_{9.2e-03}$	$1.38e-02_{4.9e-03}$

TABLE 28. IGD. Median and IQR.

	ISDGA	NSGAII	MOEAD	RandomSearch	SPEA2
RE21	$8.24e-02_{1.5e-03}$	$1.13e-01_{2.4e-02}$	$1.32e+01_{4.6e-02}$	$4.97e-01_{7.3e-02}$	$8.33e-02_{1.4e-03}$
RE22	$4.17e-02_{4.4e-03}$	$6.05e-02_{5.0e-03}$	$1.56e-01_{3.0e-02}$	$1.20e-01_{1.6e-02}$	$4.22e-02_{1.7e-03}$
RE23	$1.75e+02_{1.0e+01}$	$2.96e+02_{1.8e+01}$	$7.96e+03_{2.6e+02}$	$6.74e+03_{3.0e+03}$	$1.22e+02_{2.2e+00}$
RE24	$4.99e-02_{2.2e-03}$	$6.99e-02_{1.8e-02}$	$1.42e+00_{3.9e-03}$	$9.91e-02_{2.2e-01}$	$4.45e-02_{3.3e-03}$
RE25	$5.64e-05_{8.0e-07}$	$9.18e-05_{5.8e-06}$	$2.99e+03_{0.0e+00}$	$2.47e+02_{8.8e+01}$	$5.88e-05_{9.8e-07}$
RE31	$1.28e+04_{2.6e+03}$	$1.12e+04_{6.8e+02}$	$2.14e+05_{4.2e+02}$	$2.14e+05_{2.3e+00}$	$7.10e+03_{3.3e+02}$
RE32	$4.41e+04_{4.0e+03}$	$6.98e+04_{3.1e+04}$	$4.99e+06_{2.7e+01}$	$4.31e+06_{8.6e+05}$	$2.83e+04_{8.5e+02}$
CRE21	$1.18e+04_{1.9e+04}$	$1.15e+03_{1.0e+02}$	$8.50e+04_{3.1e-01}$	$8.50e+04_{1.5e+01}$	$4.32e+02_{1.7e+01}$
CRE22	$1.86e+00_{5.8e-02}$	$2.45e+00_{2.7e-01}$	$1.81e+02_{2.7e-01}$	$1.79e+02_{1.2e+01}$	$1.05e+00_{9.5e-03}$
CRE23	$3.98e-04_{1.3e-05}$	$5.59e-04_{2.6e-05}$	$4.53e-03_{3.0e-05}$	$7.07e-03_{1.4e-03}$	$4.37e-04_{2.6e-05}$
CRE24	$6.03e-01_{2.0e+00}$	$4.24e-01_{7.2e-02}$	$1.03e+01_{2.3e+00}$	$3.78e+00_{1.2e+00}$	$3.17e-01_{1.0e-01}$
CRE25	$2.35e-02_{5.1e-03}$	$1.36e-02_{1.4e-02}$	$2.63e-02_{8.6e-03}$	$4.85e-02_{1.8e-02}$	$1.22e-02_{8.0e-03}$

TABLE 29. deltaP. Mean and standard deviation.

	ISDGA	NSGAII	MOEAD	RandomSearch	SPEA2
RE21	$2.16e+00_{1.2e-02}$	$2.80e+00_{2.1e-01}$	$3.39e+02_{8.7e-01}$	$1.96e+01_{1.1e+01}$	$2.24e+00_{2.0e-02}$
RE22	$2.51e+02_{4.0e+02}$	$2.00e+00_{8.7e-01}$	$2.08e+00_{1.2e-01}$	$6.44e+01_{1.2e+02}$	$4.78e+03_{9.6e+03}$
RE23	$4.44e+03_{1.3e+02}$	$6.91e+03_{1.2e+02}$	$1.93e+05_{2.7e+03}$	$1.41e+05_{3.3e+04}$	$3.25e+03_{4.0e+01}$
RE24	$1.12e+00_{2.5e-02}$	$1.49e+00_{1.0e-01}$	$2.47e+01_{3.6e-02}$	$3.34e+00_{1.9e+00}$	$1.19e+00_{4.3e-02}$
RE25	$1.53e-03_{1.4e-05}$	$2.35e-03_{5.4e-05}$	$6.62e+03_{0.0e+00}$	$6.18e+04_{2.0e+04}$	$1.58e-03_{1.6e-05}$
RE31	$2.83e+05_{1.9e+04}$	$2.88e+05_{1.3e+04}$	$6.50e+06_{9.0e+03}$	$6.51e+06_{6.0e+01}$	$2.34e+05_{5.5e+03}$
RE32	$1.18e+06_{6.2e+04}$	$1.79e+06_{3.2e+05}$	$1.32e+08_{6.8e+02}$	$1.13e+08_{1.4e+07}$	$8.29e+05_{9.3e+03}$
CRE21	$1.61e+05_{1.4e+05}$	$2.39e+04_{7.4e+02}$	$2.00e+06_{4.3e+00}$	$2.00e+06_{3.1e+02}$	$9.74e+03_{1.9e+02}$
CRE22	$4.62e+01_{9.2e-01}$	$5.82e+01_{2.6e+00}$	$4.55e+03_{4.3e+00}$	$4.40e+03_{2.0e+02}$	$3.39e+01_{3.2e-01}$
CRE23	$1.37e-02_{2.1e-04}$	$1.82e-02_{4.7e-04}$	$1.06e-01_{2.6e-04}$	$2.33e-01_{1.9e-02}$	$1.49e-02_{4.0e-04}$
CRE24	$1.64e+01_{9.6e+00}$	$1.13e+01_{5.3e-01}$	$2.38e+02_{3.1e+01}$	$1.13e+02_{1.5e+01}$	$9.63e+00_{2.2e-01}$
CRE25	$6.35e-02_{1.6e-02}$	$1.20e-01_{4.5e-02}$	$1.84e-01_{9.0e-02}$	$1.54e+00_{2.7e-01}$	$6.33e-02_{2.2e-02}$

or their some certain objective values are the best. When those solutions with high ratings are selected as parents, their offsprings under the crossover and mutation operations become optimal front solutions more likely. This proposed strategy ISDGA, combined integrated sorting using D-matrix and Genetic algorithm, a simple yet efficient idea, has proved

TABLE 30. deltaP. Median and IQR.

	ISDGA	NSGAII	MOEAD	RandomSearch	SPEA2
RE21	$2.16e+00_{2.6e-02}$	$2.78e+00_{3.9e-01}$	$3.38e+02_{1.7e+00}$	$1.62e+01_{1.9e+01}$	$2.23e+00_{3.3e-02}$
RE22	$1.03e+02_{5.7e+02}$	$1.61e+00_{1.2e+00}$	$2.00e+00_{2.3e-01}$	$3.19e+00_{1.5e+02}$	$1.17e+00_{1.2e+04}$
RE23	$4.40e+03_{2.5e+02}$	$6.95e+03_{2.5e+02}$	$1.93e+05_{5.5e+03}$	$1.54e+05_{6.5e+04}$	$3.22e+03_{7.7e+01}$
RE24	$1.11e+00_{4.4e-02}$	$1.46e+00_{1.6e-01}$	$2.47e+01_{7.5e-02}$	$1.97e+00_{3.7e+00}$	$1.19e+00_{9.0e-02}$
RE25	$1.53e-03_{3.0e-05}$	$2.35e-03_{1.1e-04}$	$6.62e+03_{1.2e-04}$	$5.93e+04_{4.4e+04}$	$1.57e-03_{3.4e-05}$
RE31	$2.79e+05_{4.0e+04}$	$2.84e+05_{2.6e+04}$	$6.51e+06_{2.0e+04}$	$6.51e+06_{1.2e+02}$	$2.31e+05_{1.2e+04}$
RE32	$1.19e+06_{1.1e+05}$	$1.75e+06_{5.8e+05}$	$1.32e+08_{1.4e+03}$	$1.07e+08_{3.0e+07}$	$8.28e+05_{1.9e+04}$
CRE21	$1.53e+05_{2.8e+05}$	$2.34e+04_{1.5e+03}$	$2.00e+06_{9.5e+00}$	$2.00e+06_{4.9e+02}$	$9.65e+03_{3.9e+02}$
CRE22	$4.61e+01_{1.8e+00}$	$5.90e+01_{4.4e+00}$	$4.55e+03_{9.0e+00}$	$4.47e+03_{4.1e+02}$	$3.39e+01_{6.2e-01}$
CRE23	$1.38e-02_{4.4e-04}$	$1.82e-02_{7.9e-04}$	$1.06e-01_{4.7e-04}$	$2.35e-01_{3.4e-02}$	$1.50e-02_{8.7e-04}$
CRE24	$1.05e+01_{1.8e+01}$	$1.12e+01_{9.1e-01}$	$2.21e+02_{5.1e+01}$	$1.18e+02_{3.0e+01}$	$9.64e+00_{4.7e-01}$
CRE25	$7.14e-02_{2.0e-02}$	$1.50e-01_{9.4e-02}$	$1.41e-01_{1.2e-01}$	$1.60e+00_{5.4e-01}$	$7.14e-02_{4.7e-02}$

its advantages. ISDGA could find increasing attention and applications in the following works.

VII. CONCLUSION

In this article, we proposed a ranking system called the D-matrix method, which has a special identical diagonal value. D-matrix ranking method can be biased and bias-free ranking method and also can be used in result-separating and result-merging ways. Meanwhile, five examples are used to show the five outstanding characteristics of our proposal. Besides, due to the preference of D-matrix which is about mutual relation among data not about the matchup score, D-matrix can also be used to do the webpage ranking and get a good calculation speed. The special and well-designed diagonal value eases the difficulties of constructing a stochastic and irreducible matrix. Moreover, a pseudo-inverse matrix guarantees D-matrix a solution anytime. The rating of one item is only related to those items that are connected to it, if some irrelevant item changes, the rating of this item will get zero effect. Except for that, in this article, it gives the supporting matrix which explains the importance of the selection of diagonal value. Most importantly, this D-matrix ranking system can be adapted to any ranking field.

We have also tested the proposed D-matrix ranking system on the National Football League data of the 2017 season. And four different ways are to show the application of D-matrix, and they all get reasonable ranking results. In the last, an approximate non-dominated sorting method D-ANS is proposed based on D-matrix to do selection operation of evolutionary algorithm, which selects solutions with more of better objectives on average. This selection process boosts a better evolution of the population, which might be useful for developing new multi-objective optimization algorithms.

REFERENCES

[1] Y. Sprumont, "Ranking by rating," *Theor. Econ.*, vol. 13, no. 1, pp. 1–18, Jan. 2018.

[2] S. Devlin and T. Treloar, "A network diffusion ranking family that includes the methods of Markov, massey, and colley," *J. Quant. Anal. Sports*, vol. 14, no. 3, pp. 91–101, Sep. 2018.

[3] A. Y. Govan "Ranking theory with application to popular sports," Ph.D. dissertation, Dept. Appl. Math., North Carolina State Univ., Raleigh, NC, USA, 2008. [Online]. Available: https://repository.lib.ncsu.edu/handle/1840.16/3102

[4] A. N. Langville and C. D. Meyer, *Who's# 1?: The Science of Rating and Ranking*. Princeton, NJ, USA: Princeton Univ. Press, 2012.

[5] T. P. Chartier, E. Kreutzer, A. N. Langville, and K. E. Pedings, "Sensitivity and stability of ranking vectors," *SIAM J. Sci. Comput.*, vol. 33, no. 3, pp. 1077–1102, Jan. 2011.

[6] R. Song, Y. Li, Y. Jia, Y. Wang, and P. Rao, "Efficient, robust and divisible paired comparison for subjective quality assessment," *Multimedia Tools Appl.*, vol. 77, no. 11, pp. 13597–13613, 2018.

[7] D. Prajapati, "Paired comparison method to reduce the rejection in automobile industry," in *Proc. Nat. Conf. Adv. Futuristic Trends Aerosp. Eng.*, 2015, pp. 143–149.

[8] S. Chakravarthy and A. S. Alfa, *Matrix-Analytic Methods Stochastic Models*. Boca Raton, FL, USA: CRC Press, 2016.

[9] K. Massey. (1997). *Statistical Models Applied to the Rating of Sports Teams*. Bluefield College, Bluefield, VA, USA. [Online]. Available: https://www.masseyratings.com/theory/massey97.pdf

[10] M. Franceschet and E. Bozzo, "The Massey's method for sport rating: A network science perspective," 2017, *arXiv:1701.03363*. [Online]. Available: http://arxiv.org/abs/1701.03363

[11] W. Colley. (2002). *Colley's Bias Free College Football Ranking Method*. [Online]. Available: https://www.colleyrankings.com/matrate.pdf

[12] C. V. Lutzer, "Colley's coin: Ranking sports teams with laplace's rule of succession," *Math. Mag.*, vol. 90, no. 5, pp. 365–370, 2017.

[13] J. P. Keener, "The Perron–Frobenius theorem and the ranking of football teams," *SIAM Rev.*, vol. 35, no. 1, pp. 80–93, Mar. 1993.

[14] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Stanford InfoLab, Stanford, CA, USA, Tech. Rep., 1999.

[15] A. N. Langville and C. D. Meyer, *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton, NJ, USA: Princeton Univ. Press, 2011.

[16] R. Mattingly and A. Murphy, "A Markov method for ranking college football conferences," SUNY Cortland, NY, USA, 2010.

[17] J. Gao, Y.-W. Dong, M.-S. Shang, S.-M. Cai, and T. Zhou, "Group-based ranking method for online rating systems with spamming attacks," *EPL (Europhys. Lett.)*, vol. 110, no. 2, p. 28003, Apr. 2015.

[18] N. Veček, M. Mernik, and M. Črepinšek, "A chess rating system for evolutionary algorithms: A new method for the comparison and ranking of evolutionary algorithms," *Inf. Sci.*, vol. 277, pp. 656–679, Sep. 2014.

[19] A. E. Elo, *The Rating of Chessplayers, Past and Present*. New York, NY, USA: Arco Pub., 1978.

[20] G. P. Basharin, A. N. Langville, and V. A. Naumov, "The life and work of aa markov," *Linear algebra Its Appl.*, vol. 386, pp. 3–26, 2004.

[21] A. Y. Govan, A. N. Langville, and C. D. Meyer, "Offense-defense approach to ranking team sports," *J. Quant. Anal. Sports*, vol. 5, no. 1, Jan. 2009.

[22] A. Ben-Israel and T. N. Greville, *Generalized Inverses: Theory and Applications*, vol. 15. Ottawa, ON, Canada: Springer, 2003.

[23] B. V. Gnedenko, *Theory of Probability*. Evanston, IL, USA: Routledge, 2018.

[24] G. Wu and Y. Wei, "Arnoldi versus GMRES for computing pageRank: A theoretical contribution to Google's pageRank problem," *ACM Trans. Inf. Syst.*, vol. 28, no. 3, pp. 1–28, Jun. 2010.

[25] K. Deb, "Multi-objective optimization," in *Search Methodologies*. Berlin, Germany: Springer-Verlag, 2014, pp. 403–449.

[26] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. Hoboken, NJ, USA: Wiley, 2001, vol. 16.

[27] Y. Wang, H.-X. Li, G. G. Yen, and W. Song, "MOMMOP: Multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 830–843, Apr. 2015.

[28] J. Cheng, G. G. Yen, and G. Zhang, "A many-objective evolutionary algorithm with enhanced mating and environmental selections," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 592–605, Aug. 2015.

[29] E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the strength Pareto evolutionary algorithm," TIK-Rep., 2001, vol. 103.

[30] K. McClymont and E. Keedwell, "Deductive sort and climbing sort: New methods for non-dominated sorting," *Evol. Comput.*, vol. 20, no. 1, pp. 1–26, Mar. 2012.

[31] P. C. Roy, K. Deb, and M. M. Islam, "An efficient nondominated sorting algorithm for large number of fronts," *IEEE Trans. Cybern.*, vol. 49, no. 3, pp. 859–869, Mar. 2019.

[32] H. Wang and X. Yao, "Corner sort for Pareto-based many-objective optimization," *IEEE Trans. Cybern.*, vol. 44, no. 1, pp. 92–102, Jan. 2014.

[33] S. Mishra, S. Saha, and S. Mondal, "Divide and conquer based non-dominated sorting for parallel environment," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2016, pp. 4297–4304.

[34] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "An efficient approach to nondominated sorting for evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 201–213, Apr. 2015.

[35] 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.

[36] N. Srinivas and K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," *Evol. Comput.*, vol. 2, no. 3, pp. 221–248, Sep. 1994.

[37] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, 1999.

[38] X. Zhang, Y. Tian, and Y. Jin, "Approximate non-dominated sorting for evolutionary many-objective optimization," *Inf. Sci.*, vol. 369, pp. 14–33, Nov. 2016.

[39] L. M. Antonio and C. A. C. Coello, "Coevolutionary multiobjective evolutionary algorithms: Survey of the State-of-the-Art," *IEEE Trans. Evol. Comput.*, vol. 22, no. 6, pp. 851–865, Dec. 2018.

[40] F. Kursawe, "Evolution strategies for vector optimization," in *Proc. Preliminary 10th Int. Conf. Multiple Criteria Decis. Making*, 1992, pp. 187–193.

[41] T. Ray, K. Tai, and C. Seow, "An evolutionary algorithm for multiobjective optimization," *Eng. Optim.*, vol. 33, no. 3, pp. 399–424, 2001.

[42] M. R. Sierra and C. A. C. Coello, "Improving PSO-based multi-objective optimization using crowding, mutation and 2-dominance," in *Proc. Int. Conf. Evol. Multi-Criterion Optim.* Berlin, Germany: Springer-Verlag, 2005, pp. 505–519.

[43] J. J. Durillo and A. J. Nebro, "JMetal: A java framework for multi-objective optimization," *Adv. Eng. Softw.*, vol. 42, no. 10, pp. 760–771, Oct. 2011.

[44] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, Jun. 2000.

[45] R. Tanabe and H. Ishibuchi, "An easy-to-use real-world multi-objective optimization problem suite," *Appl. Soft Comput.*, vol. 89, Apr. 2020, Art. no. 106078.

[46] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[47] Z. Wang, Q. Zhang, A. Zhou, M. Gong, and L. Jiao, "Adaptive replacement strategies for MOEA/D," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 474–486, Feb. 2016.

[48] Y. Tian, X. Zhang, R. Cheng, and Y. Jin, "A multi-objective evolutionary algorithm based on an enhanced inverted generational distance metric," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2016, pp. 5222–5229.

[49] C. E. Leiserson, N. C. Thompson, J. S. Emer, B. C. Kuszmaul, B. W. Lampson, D. Sanchez, and T. B. Schardl, "There's plenty of room at the top: What will drive computer performance after Moore's law?" *Science*, vol. 368, no. 6495, Jun. 2020, Art. no. eaam9744.

LINGPING KONG received the master's and Ph.D. degrees in computer applied technology from the Harbin Institute of Technology, Shenzhen, China, in 2013 and 2018, respectively. She is currently with the VŠB-Technical University of Ostrava, Czech Republic. Her research interest includes multiobjective optimization and its applications.

VÁCLAV SNÁŠEL (Senior Member, IEEE) received the master's degree in numerical mathematics from the Faculty of Science, Palacký University, Olomouc, Czech Republic, in 1981, and the Ph.D. degree in algebra and number theory from Masaryk University, Brno, Czech Republic, in 1991.

He is currently a Full Professor with the VŠB-Technical University of Ostrava, Czech Republic. His research and development experience includes more than 30 years in the industry and academia. He works in a multidisciplinary environment involving artificial intelligence, social networks, conceptual lattice, information retrieval, semantic web, knowledge management, data compression, machine intelligence, and nature and bioinspired computing applied to various real-world problems. He has authored or coauthored several refereed journal articles and conference papers, books, and book chapters. He is the Chair of the IEEE International Conference on Systems, Man, and Cybernetics, Czechoslovak Chapter. He also served as an Editor/Guest Editor for several journals, such as *Engineering Applications of Artificial Intelligence* (Elsevier), *Neurocomputing* (Elsevier), and the *Journal of Applied Logic* (Elsevier).

SWAGATAM DAS (Senior Member, IEEE) received the B.E.Tel.E. and M.E.Tel.E. degrees in control engineering specialization and the Ph.D. degree from Jadavpur University, India, in 2003, 2005, and 2009, respectively.

He is currently an Associate Professor with the Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata, India. His research interests include evolutionary computing, pattern recognition, multiagent systems, and wireless communications. He has published more than 300 research papers in peer-reviewed journals and international conferences. He has more than 20 000 Google Scholar citations and an H-index of 66 to date. He is an Editorial Board Member of *Progress in Artificial Intelligence* (Springer), *Applied Soft Computing* (Elsevier), *Engineering Applications of Artificial Intelligence* (Elsevier), and *Artificial Intelligence Review* (Springer). He was a recipient of the 2012 Young Engineer Award from the Indian National Academy of Engineering (INAE) and the 2015 Thomson Reuters Research Excellence India Citation Award as the Highest Cited Researcher from India in the Engineering and Computer Science Category, from 2010 to 2014. He has been associated with the international program committees and organizing committees of several regular international conferences, including the IEEE CEC, the IEEE SSCI, SEAL, GECCO, and SEMCCO. He is the Founding Co-Editor-in-Chief of *Swarm and Evolutionary Computation* (Elsevier). He has also served as or is serving as an Associate Editor for *Pattern Recognition* (Elsevier), *Neurocomputing* (Elsevier), *Information Sciences* (Elsevier), the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, *IEEE Computational Intelligence Magazine*, IEEE ACCESS, and so on. He has acted as a Guest Editor of special issues in journals, such as the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS

• • •