Scientific Research Publishing

Journal of Applied Mathematics and Physics, 2024, 12, 2711-2722
https://www.scirp.org/journal/jamp
ISSN Online: 2327-4379
ISSN Print: 2327-4352

On an Invariant of Tournament Digraphs

Boris F. Melnikov, Bowen Liu

Faculty of Computational Mathematics and Cybernetics, Shenzhen MSU-BIT University, Shenzhen, China
Email: bormel@mail.ru, bormel@smbu.edu.cn, 1120220081@smbu.edu.cn

How to cite this paper: Melnikov, B.F. and Liu, B.W. (2024) On an Invariant of Tournament Digraphs. *Journal of Applied Mathematics and Physics*, **12**, 2711-2722.
https://doi.org/10.4236/jamp.2024.127162

Received: June 18, 2024
Accepted: July 28, 2024
Published: July 31, 2024

Abstract

To date, it is unknown whether it is possible to construct a complete graph invariant in polynomial time, so fast algorithms for checking non-isomorphism are important, including heuristic algorithms, and for successful implementations of such heuristics, both the tasks of some modification of previously described graph invariants and the description of new invariants remain relevant. Many of the described invariants make it possible to distinguish a larger number of graphs in the real time of a computer program. In this paper, we propose an invariant for a special kind of directed graphs, namely, for tournaments. The last ones, from our point of view, are interesting because when fixing the order of vertices, the number of different tournaments is exactly equal to the number of undirected graphs, also with fixing the order of vertices. In the invariant we are considering, all possible tournaments consisting of a subset of vertices of a given digraph with the same set of arcs are iterated over. For such subset tournaments, the places are calculated in the usual way, which are summed up to obtain the final values of the points of the vertices; these points form the proposed invariant. As we expected, calculations of the new invariant showed that it does not coincide with the most natural invariant for tournaments, in which the number of points is calculated for each participant. So far, we have conducted a small number of computational experiments, and the minimum value of the pair correlation between the sequences representing these two invariants that we found is for dimension 15.

Keywords

Graph, Directed Graph, Tournament, İnvariant

1. Introduction and Preliminaries. On Some Invariants of Undirected Graphs

Usually, a graph invariant is a value or an ordered set of values that somehow characterizes the structure of the given graph and does not depend on the way of

ordering of the vertices. It is important for checking graph isomorphism. For this thing, let us mention some well-known simple examples for an undirected connected graph $G=(V,E)$, see [1]-[6] etc.:

- the number of vertices, $|V|$;
- the number of edges, $|E|$;
- the chromatic number of a graph is an important invariant, which is defined as the minimal number of colors you need to color the vertices such that no neighboring vertices have the same color;
- the vector of degrees of vertices and its obvious generalization, *i.e.*, the vector of degrees of the 2nd order, [7] [8];
- whether the graph is Eulerian or Hamiltonian;
- whether it is a planar graph;
- the diameter of a graph;
- the Wiener index, *i.e.*, the value $w=\sum_{\forall i,j} d(v_i,v_j)$, where $d(v_i,v_j)$ is the minimum distance between vertices v_i and v_j;
- the Randić index, *i.e.*, the value $r=\sum_{(v_i,v_j)\in V} d(v_i)^{-1/2} d(v_j)^{-1/2}$, where v_i and v_j are the vertices forming an edge, and $d(v_k)$ is the vertex degree;
- etc.

It is also important to note that practically the same invariants can be used for oriented graphs, which are the main subject of this paper. Also note that we especially mentioned Wiener and Randić indexes last, since both the Graovac-Ghorbani index for undirected graphs (see Section 2) and the proposed in this paper new invariant for oriented tournaments graphs can be considered as their special generalizations.

However, all these examples would not be so interesting if we did not specify a counterexample, *i.e.* something that is not invariant. For it, there is usual to mention the number of crossings we get if we draw the graph in a plane is not an invariant, because the number of crossings depends on where exactly we put the vertices, and how we put the edges. However, the smallest possible number of crossings we can get is an invariant.

An invariant is called complete if the coincidence of graph invariant is necessary and sufficient to establish isomorphism. There exists the following well-known example:

- each of the values $\mu_{\min}(G)$ and $\mu_{\max}(G)$ (mini-code and maxi-code) is a complete invariant for a graph that has a fixed number of vertices *n*.

To date, it is unknown whether it is possible to construct a complete graph invariant in polynomial time (some published works with corresponding titles are considered either untested or contain errors in the constructions), so fast algorithms for checking non-isomorphism are important. At the same time, according to the authors, heuristic algorithms for checking non-isomorphism of graphs are important, and for successful implementations of such heuristics, both the tasks of some modification of previously described graph invariants and the description of new invariants remain relevant. Many of the described

invariants make it possible to distinguish a larger number of graphs in the real time of a computer program. Approximately the same options apply to oriented graphs, in particular, for tournaments, which are one of the main subjects of this article.

The computational complexity of invariants varies: some invariants are trivial computations, while computations of other invariants are very "hard-to-compute". Probabilistic algorithms exist to determine the values of some rigid invariants, but those algorithms are not always allowed to be used.

Therefore, the complete graph invariant, which can be calculated in polynomial time, is unknown, but it has not been proven that it does not exist.

In this paper, we propose an invariant for a special kind of directed graphs, namely, for tournaments. The last ones, from our point of view, are interesting because when fixing the order of vertices, the number of different tournaments is exactly equal to the number of undirected graphs, also with fixing the order of vertices. In the invariant we are considering, all possible tournaments consisting of a subset of vertices of a given digraph with the same set of arcs are iterated over. For such subset tournaments, the places are calculated in the usual way, which are summed up to obtain the final values of the points of the vertices; these points form the proposed invariant.

2. On the Graovac-Ghorbani Index

Graph theory is a branch of discrete mathematics that focuses on the study of graphs, which are mathematical structures used to represent relationships between objects. Chemical graph theory applies graph theory to solve complex molecular problems. Its main objective is to use numerical measures to the topological structure of a molecule into a single value that characterizes its properties. Topological indices are numerical measures associated with the chemical constitution, used to correlate chemical structures with various physical properties, chemical reactivity, biological activity. These indices have proven useful in predicting the behavior of chemical substances. The Graovac-Ghorbani (ABC_{GG}) index is a topological descriptor that offers improved predictive power compared to similar descriptors. It is employed to model the boiling and melting point of molecules and finds applications in the pharmaceutical industry. In recent years, there has been a rise in publications discussing its mathematical properties.

In 2010, Graovac and Ghorbani introduced a new version of the atom-bond connectivity index, which is a distance-based topological descriptor named as the Graovac-Ghorbani (ABC_{GG}) index. The index is defined by Equation (1):

$$\mathrm{ABC}_{\mathrm{GG}}(G)=\sum_{u,v\in E(G)}\sqrt{\frac{n_v+n_u-2}{n_v n_u}}. \tag{1}$$

where n_u represents the count of vertices that are closer to vertex u than to vertex v, and n_v represents the count of vertices that are closer to vertex v than to vertex u.

The research conducted in [9] highlighted that the (ABC_{GG}) index has significantly improved correlations compared to the atom-bond connectivity index for certain physico-chemical properties. Over the past few years, there has been an extensive investigation of the mathematical characteristics of the (ABC_{GG}) index in various studies [10]-[13]. A comprehensive bibliography for future research was also provided in a recent presentation of the (ABC_{GG}) index survey in [13]. Given its recentness and the existing knowledge on the (ABC_{GG}) index, it is evident that there are numerous opportunities for further exploration of its properties.

3. Research Purpose

Let us start with a question. Suppose there is a game and n players participate in it. The rules of the game are as follows:

- A two-on-two duel between the participating players;
- The result of each duel is only the winner, there is no draw;
- After each duel, the points of the winning player will increase by one point, and the points of the losing player will remain the same;
- Every player has to fight against all players except himself;

After the game is over, each player needs to be given corresponding ranking points according to the points situation. The rules for granting ranking points are as follows:

- The player with the most points will get n ranking points;
- The player who gets the second most points will get $n-1$ ranking points, and so on;
- If there are two or more players with the same number of points, each player is given the average number of ranking points in their ranking range;

An example is given below:

- Suppose there are 5 players in such a game, their numbers are 1, 2, 3, 4, 5;
- Their scoring situation is 4, 2, 2, 2, 0;
- Then their ranking points will be 5, 3, 3, 3, 1.

Now there is another way to calculate ranking points:

- Think of the players in the entire game as a set, then any k players in it form a subset;
- For each possible subset (except for a subset that contains only one player or does not contain players), corresponding points are awarded according to the duel situation between the contestants in the subset (the duel situation comes from the complete competition), the points rules are the same as the previous points rules, and according to the newly obtained points, each participant is awarded the corresponding ranking points according to the previous rules for granting ranking points;
- Add the ranking points obtained from each subset as new points to the corresponding players;
- Finally, each player is given the ranking points obtained by processing the

newly obtained points in the same way as the old ranking points.

An example is given below.

- Suppose there are 6 players in such a game, their numbers are 1, 2, 3, 4, 5, 6.
- The competition results among the players are as follows: Player 1 won against all other players. Player 2 won against players 4, 5, and 6. Player 3 won against players 2, 5, and 6. Player 4 won against players 3, 5, and 6. Player 5 won against player 6. Player 6 did not win any duel.
- Suppose there is subset 1, 2, 3.
- Their scoring situation is 2, 0, 1.
- Then their ranking points will be: 3, 1, 2.
- The ranking points obtained from this subset as new points to the corresponding players.
- After performing this operation on all possible subsets (subset size greater than 1), the new scoring situations obtained are as follows: 11, 79, 79, 79, 47, 31.
- Then their new ranking points will be: 6, 4, 4, 4, 2, 1.

Here, if we calculate the correlation using the Spearman's rank correlation coefficient formula (2),

$$\rho s = \frac{\sum_{i=1}^{n}\left(R_i - \bar{R}\right)\left(S_i - \bar{S}\right)}{\sqrt{\sum_{i=1}^{n}\left(R_i - \bar{R}\right)^2 \sum_{i=1}^{n}\left(S_i - \bar{S}\right)^2}} \tag{2}$$

We will get 1. In the formula, R_i represents the old points of player i, and $\bar{R}$ is the average of the old points. S_i represents the old points of player i, and $\bar{S}$ is the average of the old points.

What will be the difference in the results of the two different ways of awarding ranking points? Will the two results retain the relative relationship between different points?

4. Research Method

In order to get inspiration about the research direction, we wrote a program to calculate the results obtained by two different ways of awarding ranking points.

First of all, we wrote a tour class to store the information of the map. This class includes:

- the number of players;
- the duel situation of each player;
- the old points and old ranking points of each player;
- the new points and new ranking points of each player.

Then we used Gray code to generate all possible subsets of players. And in accordance with the prescribed rules, the players are awarded points and ranking points. Finally output the result to the file.

Some of the results obtained will be shown below, see **Figures 1-8**.

The first row of the data in the figure is the number of players in the game and

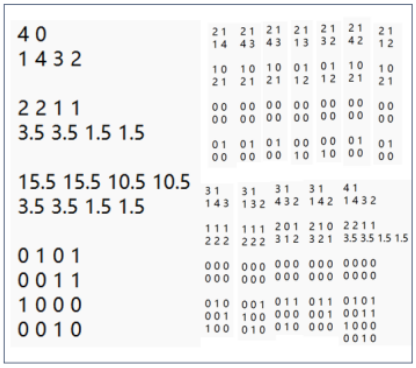

Figure 1. Tournament No. 1 study.

```
5 0 The size of the tournaments; is it complete?
2 5 3 4 1 participant numbers#

3 3 2 1 1 old points#
4.5 4.5 3 1.5 1.5 old scores#

41 40.5 31 21.5 21 new points#
5 4 3 2 1 new scores#

tournament matrix#
0 1 1 1 0
0 0 1 1 1
0 0 0 1 1
0 0 0 0 1
1 0 0 0 0
```

Figure 2. Tournament No. 2 study.

```
6 0 The size of the tournaments; is it complete?
1 4 2 3 6 5 participant numbers#

4 4 3 2 1 1 old points#
5.5 5.5 4 3 1.5 1.5 old scores#

100.5 98.5 80 62 41.5 43.5 new points#
6 5 4 3 1 2 new scores#

tournament matrix#
0 1 1 1 0 1
0 0 1 1 1 1
0 0 0 1 1 1
0 0 0 0 1 1
1 0 0 0 0 0
0 0 0 0 1 0
```

Figure 3. Tournament No. 3 study.

```
7 0 The size of the tournaments; is it complete?
2 6 4 5 3 1 7 participant numbers#

4 4 4 3 3 2 1 old points#
6 6 6 3.5 3.5 2 1 old scores#

199 201 202 156 160 115 80 new points#
5 6 7 3 4 2 1 new scores#

tournament matrix#
0 0 1 0 1 1 1
1 0 0 0 1 1 1
0 1 0 1 0 1 1
1 1 0 0 0 0 1
0 0 1 1 0 1 0
0 0 0 1 0 0 1
0 0 0 0 1 0 0
```

Figure 4. Tournament No. 4 study.

```
7 0 The size of the tournaments; is it complete?
1 4 2 6 5 3 7 participant numbers#

5 5 4 3 2 2 0 old points#
6.5 6.5 5 4 2.5 2.5 1 old scores#

232 229 191 155 120 123 63 new points#
7 6 5 4 2 3 1 new scores#

tournament matrix#
0 1 1 0 1 1 1
0 0 1 1 1 1 1
0 0 0 1 1 1 1
1 0 0 0 0 1 1
0 0 0 1 0 0 1
0 0 0 0 1 0 1
0 0 0 0 0 0 0
```

Figure 5. Tournament No. 5 study.

```
10 0 The size of the tournaments; is it complete?
1 8 10 6 5 7 4 2 3 9 participant numbers#

8 6 5 5 4 4 4 4 3 2 old points#
10 9 7.5 7.5 4.5 4.5 4.5 4.5 2 1 old scores#

2759 2263 1891 1885 1493 1527 1491 1491 1059 771 new points#
10 9 8 7 5 6 3.5 3.5 2 1 new scores#

tournament matrix#
0 1 1 1 1 1 1 1 0 1
0 0 1 0 1 1 0 1 1 1
0 0 0 0 1 0 1 1 1 1
0 1 1 0 1 0 0 1 0 1
0 0 0 0 0 1 1 0 1 1
0 0 1 1 0 0 1 0 1 0
0 1 0 1 0 0 0 1 1 0
0 0 0 0 1 1 0 0 1 1
1 0 0 1 0 0 0 0 0 1
0 0 0 0 0 1 1 0 0 0
```

Figure 6. Tournament No. 6 study.

```
15 0 The size of the tournaments; is it complete?
1 3 4 12 9 2 6 11 8 10 7 13 14 15 5 participant numbers#

10 10 10 9 9 8 7 7 6 5 5 5 5 5 4 old points#
14 14 14 11.5 11.5 10 8.5 8.5 7 4 4 4 4 4 1 old scores#

112199 111795 113159 101699 99615 87359 75391 73567 59127 47935 45387 47503 47691 46931 36547 new points#
14 13 15 12 11 10 9 8 7 6 2 4 5 3 1 new scores#

tournament matrix#
0 1 0 0 1 1 1 0 1 1 0 1 1 1 1
0 0 0 1 1 1 1 0 0 1 1 1 1 1 1
1 1 0 0 1 1 1 0 1 1 1 1 0 0 1
1 0 1 0 1 1 1 0 1 0 0 1 1 0 1
0 0 0 0 0 1 1 1 0 1 1 1 1 1 1
0 0 0 0 0 0 1 1 1 1 0 1 1 1 1
0 0 0 0 0 0 0 1 1 1 1 0 1 1 1
1 1 1 1 0 0 0 0 0 1 0 0 0 1 1
0 1 0 0 1 0 0 1 0 0 0 0 1 1 1
0 0 0 1 0 0 0 0 1 0 1 1 1 0 0
1 0 0 1 0 1 0 1 1 0 0 0 0 0 0
0 0 0 0 0 0 1 1 1 0 1 0 0 0 1
0 0 1 0 0 0 0 1 0 0 1 1 0 1 0
0 0 1 1 0 0 0 0 0 1 1 1 0 0 0
0 0 0 0 0 0 0 0 0 1 1 0 1 1 0
```

Figure 7. Tournament No. 7 study.

```
15 0 The size of the tournaments; is it complete?
1 12 10 5 4 2 9 8 3 11 6 13 14 15 7 participant numbers#

10 10 9 8 8 8 8 7 7 6 6 5 5 4 4 old points#
14.5 14.5 13 10.5 10.5 10.5 10.5 7.5 7.5 5.5 5.5 3.5 3.5 1.5 1.5 old scores#

115283 113375 102347 87735 88927 87423 87327 73959 74463 59783 60159 44267 44199 33067 33591 new points#
15 14 13 11 12 10 9 7 8 5 6 4 3 1 2 new scores#

tournament matrix#
0 1 1 1 1 0 1 0 1 1 1 1 0 0 1
0 0 0 1 1 0 1 1 1 1 1 0 1 1 1
0 1 0 1 0 1 1 1 0 0 1 1 0 1 1
0 0 0 0 1 1 0 1 0 1 1 1 0 1 1
0 0 1 0 0 1 1 1 0 0 1 1 1 0 1
1 1 0 0 0 0 1 0 0 1 1 0 1 1 1
0 0 0 1 0 0 0 0 1 1 1 1 1 1 1
1 0 0 0 0 1 1 0 0 1 0 1 1 1 0
0 0 1 1 1 1 0 1 0 0 0 1 0 1 0
0 0 1 0 1 0 0 0 1 0 0 1 1 1 0
0 0 0 0 0 0 0 1 1 1 0 1 1 0 1
0 1 0 0 0 1 0 0 0 0 0 0 1 1 1
1 0 1 1 0 0 0 0 1 0 0 0 0 0 1
1 0 0 0 1 0 0 0 0 0 1 0 1 0 0
0 0 0 0 0 0 0 1 1 1 0 0 0 1 0
```

Figure 8. Tournament No. 8 study.

The following are two functions of the program **Subset_kid()**, **calculatesco()** and **calculatesco_part()**, see **Figures 9-11**.

Whether it is a complete game (0 is a complete game). The second row is the number of players included in the subset, the third row is the old points, the fourth row is the old ranking points, the fifth row is the new points, and the sixth

```
void tour::Subset_kid(const tour& parent,Linarr* number) {
    this->nDim=number->GetnDim();
    InitMemory();
    int* num = new int[this->nDim];
    number->Writearr(num, this->nDim);
    sub = 1; number->Get1(); number->Getuser();
    for (int i = 1; i <= this->nDim; i++) {
        Set_p(i, parent.Get_p(number->user->value));
        number->movebyrelativeplace(1); number->Getuser();
    }
    for(int i=1;i<=nDim;i++){
        Set_t_unnormal(i,i,0);
        for(int j=i+1;j<=nDim;j++)
            Set_t(i,j,parent.Get_t(num[i-1], num[j-1]));
    }
    delete[] num;
}
```

Figure 9. Subset_kid().

```
double tour::calculatesco_part(double* sco,Linarr2* a,double sum,int count,int* score) {
    double tmp = a->user->value; int i = a->user->number; a->movebyrelativeplace(1); a->Getuser(); score[0] += 1;
    double Return = 0;
    if (a->user == nullptr || a->user->value != tmp) {
        Return = (score[0] - 1 + sum) / (count + 1);
        sco[MakeIndex_s(i)] = Return;
    }
    else {
        Return = calculatesco_part(sco, a, sum + score[0] - 1, count + 1, score);
        sco[MakeIndex_s(i)] = Return;
    }
    return Return;
}
```

Figure 10. Calculatesco().

```
void tour::calculatesco(double* sco,double* point) {
    Linarr2 Forcal(0, 0);
    int* number = new int[nDim]; for (int i = 1; i <= nDim; i++)number[i - 1] = i;
    Forcal.Readarr(point, number, nDim);
    Forcal.Get1(); Forcal.Getuser();
    int score[1] = { 1 };
    for (; score[0] <= nDim; ) {
        double tmp_value = Forcal.user->value; int tmp_number = Forcal.user->number;
        Forcal.movebyrelativeplace(1); Forcal.Getuser(); score[0] += 1;
        if (Forcal.user == nullptr) {
            sco[MakeIndex_s(tmp_number)] = score[0] - 1; break;
        }
        if (Forcal.user->value == tmp_value) {
            sco[MakeIndex_s(tmp_number)] = calculatesco_part(sco, &Forcal, score[0] - 1, 1, score);
            if (Forcal.user == nullptr)return;
            tmp_value = Forcal.user->value; tmp_number = Forcal.user->number;
        }
        if(Forcal.user->value !=tmp_value) {
            sco[MakeIndex_s(tmp_number)] = score[0] - 1;
        }
    }
    delete[]number;
}
```

Figure 11. Calculatesco_part().

row is the new ranking points. The matrix in the figure is the game situation between the players in the case of this subset. For example, the first row is the duel situation of the first player in the player row, and each column represents the duel situation with the player in the corresponding position in the player row (1 as the winner).

In function **Subset_kid()**, we used a linear array **Liarr** to store information about myself, such as {1, 2}, meaning that the information of the players at the first and second positions should be taken. The array **num** is used to simplify the subsequent operations. The first loop stores the player's numbers corresponding to the positions in the subset. The second loop retrieves the duel information related to the selected players from the original matrix (as introduced above).

In function **calculatesco()**, we used the array **point** to represent the point, and the array **sco** to represent the ranking point. Here because the number of players has nothing to do with their position in the player array, in order to facilitate the calculation, we use the array **num** to store the position of each player in the player array. Then we used the two-variable storage of the array **Linarr2** which can bundle the player's position information with point and used its quick sort feature for a single variable to arrange it in the order of point increment. Then we use a loop to assign a corresponding ranking point to each player. The specific implementation process is as follows:

- the value of **score[0]** is always equal to the ranking point of the current point ranking;
- if the player's point is unique, then give him the value of **score[0]** at this time, because the player's position information is bundled with the point, so the program can directly use the position information to assign a ranking point to the player in the corresponding position;
- if the player's point is not unique, that is, there are other players who have the same point, then recursively find out how many players have the same

point through the **calculatesco_part()** function, After finding the specific number, the function will automatically assign the ranking point that each player with the same point should have based on the location information.

So the function **calculatesco()** successfully completed the function of assigning ranking points.

5. Assumptions about Invariant and Current Proof Progress

From the example, we can find that the person who has a larger number of points in the complete game will also have a larger ranking point (greater than the person whose points are lower than him). And the person with smaller points will also have smaller ranking points (less than the person with higher points than him).

Based on this interesting fact, we assume that under the two ranking methods, the ranking order of different points is an invariant of the game situation figure.

The following will give the current progress we have made in the process of proving this invariant.

In the beginning, suppose that in any duel of n contestants, the person who gets the points K (K > 1), on average, the new points will be at least 2^{n-2} points more than the new points of the person who gets the points K − 1.

This is easy to prove through the following steps:

- Add a point to any person whose points are K − 1, then a point must be deducted from someone;
- In the process of obtaining new points, this exchange of points only affects the situation where both parties to the exchange exist in a subset, and it will not have an impact on other subsets, because other subsets do not contain duels that exist in the exchange;
- Then there are 2^{n-2} subsets of both sides of the exchange at the same time;
- The ranking points of the players who earn points in each such subset will increase by at least 1 point, because the points of the players who earn points in this case will definitely increase by 1 in each such subset, so the player's points ranking in the case of this subset will also increase, and according to the old rules of granting ranking points, it can be learned that the player's old ranking points in the case of this subset will also increase by at least 1;

So, in the average case, the person who gets the points K (K > 1), the new points will be at least 2^{n-2} points more than the new points of the person who gets the points K − 1.

In the next step, if it is proved that in different competitions with n players, people who have obtained the same points, the range of changes in their new points will not exceed a certain value, it can be proved that under the two ranking methods, the ranking order of different points is an invariant of the game situation figure.

6. Conclusions

Thus, many invariants are used, for example, in computational chemistry to

solve a wide range of general and special problems. These tasks include:

- search for substances with predetermined properties (search for dependencies such as "structure-property", "structure-pharmacological activity"),
- primary filtering of structural information for the repeated generation of molecular graphs of a given type,
- and a number of others.

Often, along with topological indices (depending only on the structure of the molecule), information about the chemical composition of the compound is also used.

As an example of considering the invariant described in the paper and the directions of further work, let us consider the following. A heuristic approach to the verification of isomorphic graphs is described in this work. The approach is a sequential verification of the graph characteristics which are invariants. The results of computational experiments are described. The aim of experiments is to determine which of the comparative sequences (for graph invariants values) are more efficient for graph isomorphism verification, see [7] [8] [14] etc.

It will be important to demonstrate the application to counting and enumeration of tournaments, as well as to checking two graphs for isomorphism, including heuristic verification; the connectivity with the invariant considered in the paper is obvious.

In the future, we propose to take a deeper look at the relationship with the Graovac-Ghorbani index; the connectivity is less obvious here, but when we think about it, it is quite visible.

Acknowledgements

This work was partially supported by a grant from the scientific program of Chinese universities "Program to support the stability of Higher Education" (section "Shenzhen 2022 Commission on Science, Technology and Innovation of the Shenzhen Municipality").

Founding

This work was partially supported by a grant from the scientific program of Chinese universities "Program to support the stability of Higher Education" (section "Shenzhen 2022 Commission on Science, Technology and Innovation of the Shenzhen Municipality").

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Harary, F. (1969) Graph Theory. Addison-Wesley. https://doi.org/10.21236/AD0705364

[2] Trudeau, R.J. (1994) Introduction to Graph Theory. Dower Publications.

[3] Chartrand, G. and Zhang, P. (2012) Introduction to Graph Theory. Dover Publications.

[4] Gera, R., Hedetniemi, S. and Larson, C. (2016) Graph Theory. Favorite Conjectures and Open Problems—1. Springer-Verlag. https://doi.org/10.1007/978-3-319-31940-7

[5] Diestel, R. (2017) Graph Theory. Springer-Verlag. https://doi.org/10.1007/978-3-662-53622-3

[6] Gera, R., Hedetniemi, S. and Larson, C. (2018). Graph Theory. Favorite Conjectures and Open Problems—2. Springer-Verlag. https://doi.org/10.1007/978-3-319-97686-0

[7] Sayfullina, E.F. (2016). A Heuristic Approach to the Verification of Isomorphic Graphs. *Collection of selected papers of the II International Conference on Information Technology and Nanotechnology*, 838-842. https://doi.org/10.18287/1613-0073-2016-1638-838-842

[8] Melnikov, B. and Sayfullina, E. (2013) Application of a Multiheuristic Approach for Random Generation of a Graph with a Given Degree Vector. News of Higher Educational Institutions. Volga Region. *Physical and Mathematical Sciences*, **27**, 70-83. (In Russian)

[9] Furtula, B. (2016) Atom-Bond Connectivity Index versus Graovac—Ghorbani Analog. *MATCH Communications in Mathematical and in Computer Chemistry*, **75**, 233-242.

[10] Pacheco, D., de Lima, L. and Oliveira, C.S. (2021) On the Graovac—Ghorbani Index for Bicyclic Graphs with No Pendent Vertices. *MATCH Communications in Mathematical and in Computer Chemistry*, **86**, 429-448.

[11] Filipovski, S. (2022) Connected Graphs with Maximal Graovac-Ghorbani Index. *Match Communications in Mathematical and in Computer Chemistry*, **89**, 517-525. https://doi.org/10.46793/match.89-2.517f

[12] Majstorović, S. (2023) Maximizing Graovac-Ghorbani Index of Trees with Fixed Maximum Degree. *MATCH Communications in Mathematical and in Computer Chemistry*, **90**, 673-684. https://doi.org/10.46793/match.90-3.673m

[13] Pacheco, D., Oliveira, C. and Novanta, A. (2023) A Survey on Graovac-Ghorbani Index. *Match Communications in Mathematical and in Computer Chemistry*, **90**, 301-312. https://doi.org/10.46793/match.90-2.301p

[14] Melnikov, B., Sayfullina, E., Terentyeva, Y. and Churikova, N. (2018) Application of Random Graph Generation Algorithms to Study the Reliability of Communication Networks. *Informatization and Communication*, No. 1, 71-80. (In Russian)