Applied Intelligence (2023) 53:7988–8004
https://doi.org/10.1007/s10489-022-03802-y

On the solution of the graph bandwidth problem by means of search methods

Behrooz Koohestani[1]

Accepted: 23 May 2022 / Published online: 20 July 2022

Abstract
The Graph Bandwidth Problem is a well-known and important graph layout problem with a large number of applications in scientific and engineering fields. The problem is proved to be NP-complete, and so far, a variety of methods have been proposed for its solution. Among these methods, the most popular ones include search methods, in particular informed search methods. An informed search method normally requires a metric to guide the search toward high-quality solutions. The most frequently used metric in previous studies on the Graph Bandwidth Problem is simply the bandwidth itself, i.e., the most obvious quality measure. In this paper, it is shown that this metric is not always appropriate for comparing the quality of solutions produced by various search methods, and its use may result in a significant reduction in the performance of such methods. In order to address this issue, a new metric is presented, and its effectiveness is verified by a considerable number of numerical experiments on benchmark problems.

Keywords Combinatorial optimisation · Graph layout problems · Bandwidth problem · Search methodologies · Informed search methods

1 Introduction

Combinatorial optimisation is a field that aims to address discrete optimisation problems with the use of combinatorial techniques. In discrete optimisation problems, the main purpose is to determine an optimal solution from a finite set of possible solutions. The Travelling Salesman Problem (TSP) is a classical example of problems in combinatorial optimisation. Such problems are generally very hard to solve [1].

Graph layout problems are a category of combinatorial optimisation problems. In these problems, the goal is to discover a layout or a linear arrangement of a given graph in such a way as to optimise a specific objective function. A layout is generated by labeling the vertices of a graph with distinct integers (i.e., from the set $\{1, 2, \ldots, n\}$) [2].

Minimum Linear Arrangement, Bandwidth, Cutwidth, Sum Cut, Modified Cut, Envelope, Vertex Separation, Vertex Bisection and Edge Bisection problems are considered the most important graph layout problems. Most of these problems are known to be NP-complete, meaning that finding optimal solutions cannot be guaranteed in polynomial time. However, because feasible solutions with near-optimal cost are usually sufficient, approximation algorithms and heuristics can also be employed for solving them [2, 3].

The Graph Bandwidth Problem (GBP) is a well-known and important graph layout problem, that was originally proposed to speed up a number of computations on sparse matrices [4]. The GBP is to label the vertices of a graph with distinct integers such that the maximum absolute difference between the labels of adjacent vertices is reduced. The problem can also be defined in the context of a symmetric matrix. In that case, the rows and columns of the matrix are reordered such that its non-zero elements are as close as possible to the main diagonal [4].

The GBP has a large number of applications in scientific and engineering fields, e.g., numerical analysis, large linear systems, circuit design, large scale power transmission systems, chemical kinetics, VLSI design, numerical geophysics, data storage, saving large hypertext media, finite element methods, network survivability, industrial electromagnetics and topology compression of road networks [5–8].

✉ Behrooz Koohestani
b.koohestani@tabrizu.ac.ir

[1] Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran

Minimising the bandwidth size is an NP-complete problem [9]. The GBP is also NP-complete even for trees having a maximum degree of three, and finding the optimal ordering in polynomial time is possible only in very special cases [10].

Highly effective methods for addressing the GBP are mainly informed search methodologies (i.e., Heuristics and Metaheuristics) in which a metric is required to guide the search towards better areas of the search space. The most frequently used metric in previous studies is simply the bandwidth itself, which is in fact the most obvious quality measure (See Section 2.3).

In this study, we show that this metric is not always suitable for comparing the quality of solutions produced by different informed search methods, and its utilisation may cause a remarkable reduction in the performance and effectiveness of such methods. In order to deal with this issue, a new metric is proposed, and its effectiveness is tested and verified by a large set of numerical experiments on standard benchmark problems from the University of Florida Sparse Matrix Collection [11]. The results are very promising and indicate that further progress on search-based GBP solvers can be expected from incorporating the proposed new metric into these solvers.

This paper is organised as follows. In Section 2, some background information about the GBP, search methods and relevant existing algorithms is provided. In Section 3, the motivation for the present study is described. In Section 4, the new metric is defined, and its application is discussed. In Section 5, the numerical experiments carried out to assess the effectiveness of the proposed metric are presented and analysed, followed by a discussion. In Section 6, the conclusions of this research are summarised.

2 Background

2.1 Problem definition

The GBP can be formally defined as follows. Let $G(V, E)$ be a graph with vertex set V and edge set E. Further, let $|V| = n$. A labeling f of G assigns the integers $\{1, \ldots, n\}$ to the vertices of G. Let $f(v)$ be the label of vertex v. Note that each vertex of G has a different label.

The bandwidth of a vertex $v \in V$, $B_f(v)$, is the maximum of the differences between $f(v)$ and the labels of its adjacent vertices with respect to the following conditions:

$$B_f(v) = \begin{cases} max\left(f(v) - f(u)\right) & \text{if } \{\exists u : (u \in N(v)) \wedge (f(u) < f(v))\} \\ 0 & \text{otherwise} \end{cases}, \tag{1}$$

where $N(v)$ is the set of vertices adjacent to v. Considering (1), the bandwidth of G with respect to f can be expressed as:

$$B_f(G) = max\left\{B_f(v) : v \in V\right\}. \tag{2}$$

The GBP, therefore, consists of finding a labeling f that minimises $B_f(G)$. Note that a labeling is simply a renumbering of the vertices of G.

The notion of profile is a key concept regarding the bandwidth. Given a graph $G(V, E)$, its profile (or of the corresponding sparse matrix) is:

$$P_f(G) = \sum_{v=1}^{n} B_f(v). \tag{3}$$

Figure 1 shows an undirected graph having 6 vertices and 7 edges. The bandwidth and profile of this graph are calculated as follows:

$$\begin{array}{l} Vertex\ 1 : B_f(1) = 0\ ,\quad Vertex\ 2 : B_f(2) = 1\ ,\quad Vertex\ 3 : B_f(3) = 1 \\ Vertex\ 4 : B_f(4) = 1\ ,\quad Vertex\ 5 : B_f(5) = 4\ ,\quad Vertex\ 6 : B_f(6) = 2 \\ B_f(G) = 4 \\ P_f(G) = 9 \end{array}$$

2.2 Search methods

The solution to a large number of problems in science and engineering can be obtained by discovering a series of related actions leading to a particular goal. The state of a problem is altered by applying each action, and the main aim is obviously finding a series of states and actions by which reaching a goal state from an initial state is possible. The search space can be defined as the set of possible states with their associated operators (actions). Search can then be defined as systematically examining problem states with the aim of finding a path from an initial state to a goal state. Therefore, an algorithm with the ability of performing such a search is called search method [12].

There are generally two types of search methods, namely uninformed search methods and informed search methods. Breadth-First Search (BFS) and Depth-First Search (DFS) are examples of uninformed search methods. In these methods, the information provided in the problem definition is the only information that is available and can be used by the algorithm. Tabu search (TS) and Simulated Annealing (SA) are examples of informed search methods. In these methods, domain-specific information is available that can be used by the algorithm to decide which path is more promising for continuing the search. Informed search methods are much more effective compared to uninformed search methods and employed extensively for solving hard problems. Informed search methods use a heuristic function

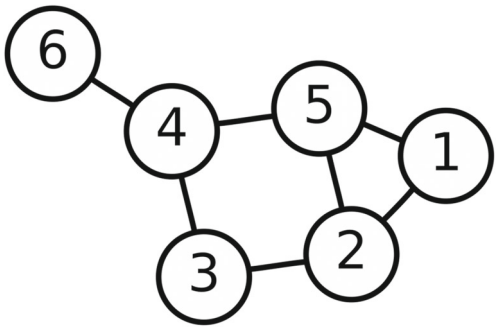

Fig. 1 An undirected graph

(also called a fitness function) for evaluating the fitness of a solution (i.e., the distance to a goal state). The heuristic function is responsible for guiding the search towards desirable areas of the search space. This is accomplished by returning a metric which indicates how effective a solution is. It is therefore clear that the quality of the metric returned by the heuristic function is a key factor in the performance of informed search methods [12].

Heuristics are informed search methods. They are, in fact, intelligent search strategies, which are employed for addressing problems in different fields. One of the main characteristics of heuristics is that they are problem-specific. This means that heuristics are normally designed for the solution of a particular problem. Metaheuristics are also informed search methods. They are actually higher-level heuristics. One of the key differences between heuristics and metaheuristics is that metaheuristics are problem-independent, meaning that they can solve a wide range of problems [13].

2.3 Relevant existing algorithms

The first exact method for the GBP was proposed by Harary [14]. Gurari and Sudborough [15] and Corso and Manzini [16] also proposed three other exact methods for solving the problem. It should be noted that these methods have only been applied to relatively small-sized matrices, i.e., up to 100×100. Cygan and Pilipczuk [17] proposed an exact exponential-time algorithm for the GBP. Their algorithm consists of two phases, including generating segment assignments and depth-first search.

Cuthill and McKee [18] introduced one of the most well-known graph theoretic-based search methods for the reduction of the bandwidth of sparse symmetric matrices. Their algorithm (i.e., the CM) is heuristic in nature, and still widely used. In this algorithm, the vertices of a graph associated with a symmetric matrix are divided into classes with respect to their distances from a root vertex. This structure is called a *level structure*. The best candidates for the root vertex in the level structure are those vertices with the minimum degree. If the vertices in the level structure are visited in increasing-distance order, a permutation is generated. The permutation is then applied to the matrix or its associated graph with the aim of reducing the bandwidth. It was observed that renumbering the CM ordering in a reverse way (RCM) could produce even better solutions compared to the original ordering [19].

Gibbs et al. [20] developed another well-known graph-theoretic heuristic search method (i.e., the GPS). This algorithm also uses the concept of the level structure as described earlier. An important feature of the algorithm is that it incorporates a very effective heuristic for detecting the endpoints of a pseudo-diameter to be employed as a suitable starting vertex. The GPS algorithm can generally outperform the CM algorithm in terms of effectiveness [21]. The GPS has been very successful in reducing the bandwidth of finite-element stiffness matrices that typically arise from problems in structural engineering. Wang and Shi [22] proposed an improved version of the GPS. This algorithm uses a heuristic parameter for detecting appropriate pseudo-peripheral vertices.

Gonzaga de Oliveira et al. [23] presented a heuristic search method for addressing the problem. Their algorithm employs both the Wonder bandwidth reduction algorithm and George-Liu algorithm. The George-Liu algorithm is used to provide a starting vertex. Gonzaga de Oliveira et al. [24] also proposed an improved version of the George-Liu algorithm for obtaining pseudo-peripheral vertices.

Metaheuristics, which are higher-level informed search methods, have been comprehensively examined to see whether they can be effective alternatives for solving the GBP. Marti et al. [25] employed a Tabu Search method for this purpose. A Genetic Algorithm combined with a Hill-climbing algorithm was used by Lim et al. [26] in order to solve this problem. Another genetic algorithm-based approach was given by Pop et al. [27] as well. A greedy randomized adaptive search method combined with a path relinking strategy (GRASP-PR) for addressing the GBP was proposed by Piñana et al. [28]. Lim et al. also presented an Ant Colony Optimization algorithm combined with a Hill-climbing algorithm [29] and a Particle Swarm Optimization algorithm combined with a Hill-climbing algorithm [30].

Czibula et al. [31] presented a Reinforcement Learning approach for solving the matrix bandwidth minimisation problem. They also applied genetic algorithms and ant-based systems to the problem [32]. Koohestani and Poli introduced two Genetic Programming systems in which genetic programming was used as a meta-heuristic [33] and as a hyper-heuristic [34]. Pop and Matei [35] also introduced an improved heuristic based on genetic programming.

A dual representation simulated annealing (DRSA) for the bandwidth minimisation problem was developed

by Torres-Jimenez et al. [36]. In addition, in Chagas and Gonzaga de Oliveira [37] and Gonzaga de Oliveira et al. [38], metaheuristic-based approaches and low-cost heuristics for matrix bandwidth reduction were reviewed and evaluated, respectively. Mafteiu-Scai et al. [39] presented two hybrid methods based on Brain Storm Optimization, which is a swarm intelligence algorithm, for reducing the bandwidth.

Gonzaga de Oliveira and Silva [40] proposed a hyper-heuristic approach based on ant colony optimization (ACHH) for reducing the bandwidth of symmetric and non-symmetric matrices. The ACHH evolves graph-theoretic heuristic search methods for specific application areas. These heuristics are used to achieve low execution times for addressing the bandwidth reduction of large sparse matrices. The ACHH is provided with the main structure of the RCM-GL, KP-band heuristics, and a variation of King's algorithm. The algorithm is then asked to evolve low-cost bandwidth reduction heuristics. Finally, the resulting heuristics are evaluated and compared against the most promising low-cost heuristics available in the literature.

Gonzaga de Oliveira and Silva [41] also proposed another hyper-heuristic approach based on ant colony optimization. The structure of this hyper-heuristic is similar to their work described above. This ACHH evolves low-cost heuristics for bandwidth reduction with the aim of accelerating the convergence of the zero-fill incomplete Cholesky-preconditioned conjugate gradient method (ICCG method). In [42], a modified version of the ACHH is introduced. The modified version is combined with a Hill-Climbing algorithm and uses the components of the RCM, KP-band, RBFS-GL, and RLK heuristics. The heuristics generated by this hyper-heuristic are capable of producing better solutions than those delivered by low-cost bandwidth reduction heuristics.

The most recent metaheuristic approach to the GBP is probably that of Silva et al. [43] who presented a Biased Random-Key Genetic Algorithm (BRKGA). In BRKGA, each gene in a chromosome is a random real number (i.e., a key) uniformly generated from the interval [0,1]. During the selection process, one parent is always selected at random from elite individuals in the current population, whereas the other is a non-elite individual (i.e., the selection is biased). In crossover operation, a parameterized uniform crossover scheme is first applied to a chromosome. A decoder is then used to map the altered chromosome into a feasible solution (i.e., a permutation).

3 Motivation

As mentioned in the previous section, informed search methods require a metric by which the search is guided towards high-quality solutions. In the case of heuristics and metaheuristics applied to the GBP, the most obvious metric, normally used in previous studies, is the bandwidth (See Section 2.3).

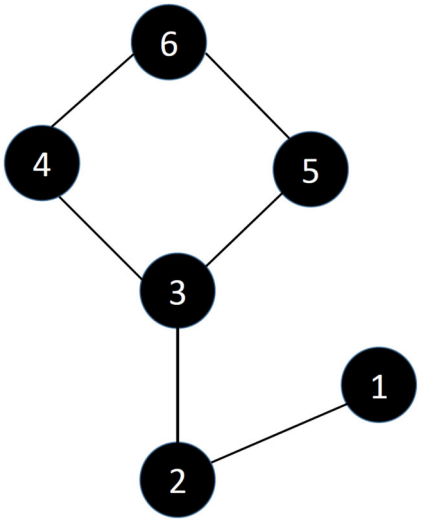

Fig. 2 Labeling 1, $B_f\,(G) = 2$

However, in spite of the fact that the bandwidth value is the quantity that should be minimised, it cannot be considered an ideal metric for a heuristic search. The most important reason for this issue is that if two or more candidate solutions have the same bandwidth (i.e., which is very likely), this does not necessarily mean that their qualities are the same. They may actually be quite different in terms of being closer to more promising solutions.

For instance, in Figs. 3 and 4, at first sight, it might seem that labeling 2 and labeling 3 have no priority over each other considering their bandwidth measures, which are both

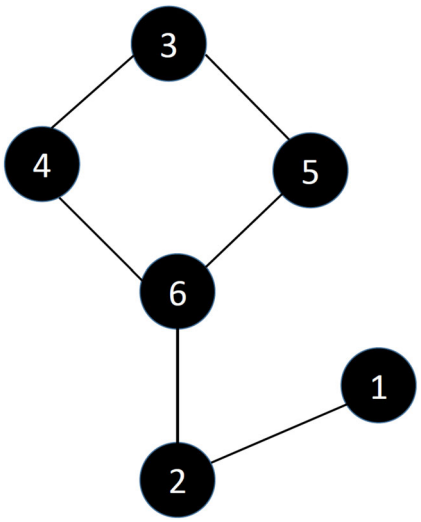

Fig. 3 Labeling 2, $B_f\,(G) = 4$

equal to 4. However, there is no doubt that labeling 2 is much closer to an ideal solution (i.e., labeling 1 in Fig. 2) than labeling 3. In fact, by a simple exchange of vertices labeled by 3 and 6, we can easily obtain labeling 1 from labeling 2 (Figs. 3 and 4).

This clearly shows that the bandwidth is not always a suitable metric for comparing the quality of solutions produced by different search methods. If the comparison of the solution quality is not done properly, it is not possible to guide the search towards promising solutions, which in turn results in a remarkable reduction in the performance of such methods. In order to deal with this issue, in this study, a new metric is proposed with the aim of increasing the effectiveness of heuristic search methods for the GBP, as described in the next section.

4 Proposed new metric

Considering the issues mentioned in Section 3, the development of a new metric seems to be necessary for addressing the GBP using informed search methods. This metric should be more informative and able to clearly differentiate between the quality of candidate solutions in comparison with the standard metric. In order to reach this aim, a new metric is proposed in this study for the GBP, called Modified Profile (MP) as follows:

$$MP_f(G) = \sum_{v=1}^{n} B_f(v)^k . \tag{4}$$

The reason behind the name chosen is that there is a relation between the profile and this new metric. For calculating the MP of a given graph G , the bandwidth of each vertex is raised to the power of k, where k is a positive integer, and then all of these terms are summed. Now, if k is equal to 1, the MP will be exactly the same as the profile. Figure 5 illustrates the process of determining the bandwidth, profile and modified profile. The MP has an interesting property as described below.

If vertex m is a single vertex with maximum bandwidth, the following relations are valid:

$$B_f(m) = B_f(G) . \tag{5}$$

$$MP_f(G) = B_f(1)^k + B_f(2)^k + \ldots + B_f(m)^k + \ldots + B_f(n)^k . \tag{6}$$

$$MP_f(G) = B_f(m)^k \left[\left(\frac{B_f(1)}{B_f(m)} \right)^k + \left(\frac{B_f(2)}{B_f(m)} \right)^k + \ldots + 1 + \ldots + \left(\frac{B_f(n)}{B_f(m)} \right)^k \right] . \tag{7}$$

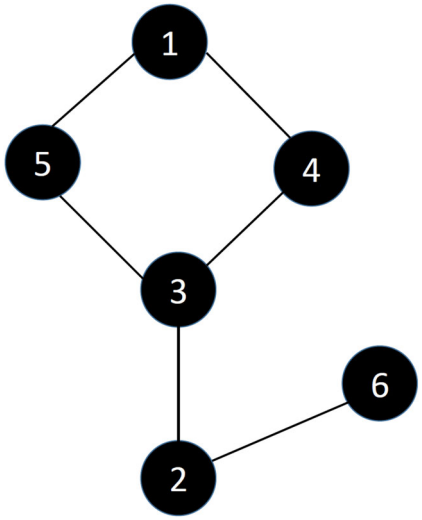

Fig. 4 Labeling 3, $B_f(G) = 4$

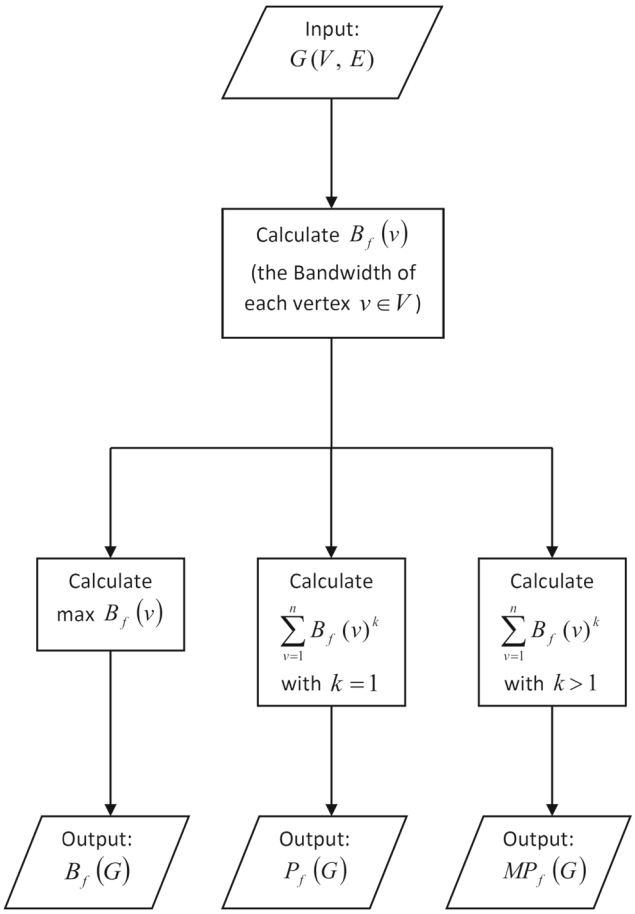

Fig. 5 Determination of bandwidth, profile and modified profile

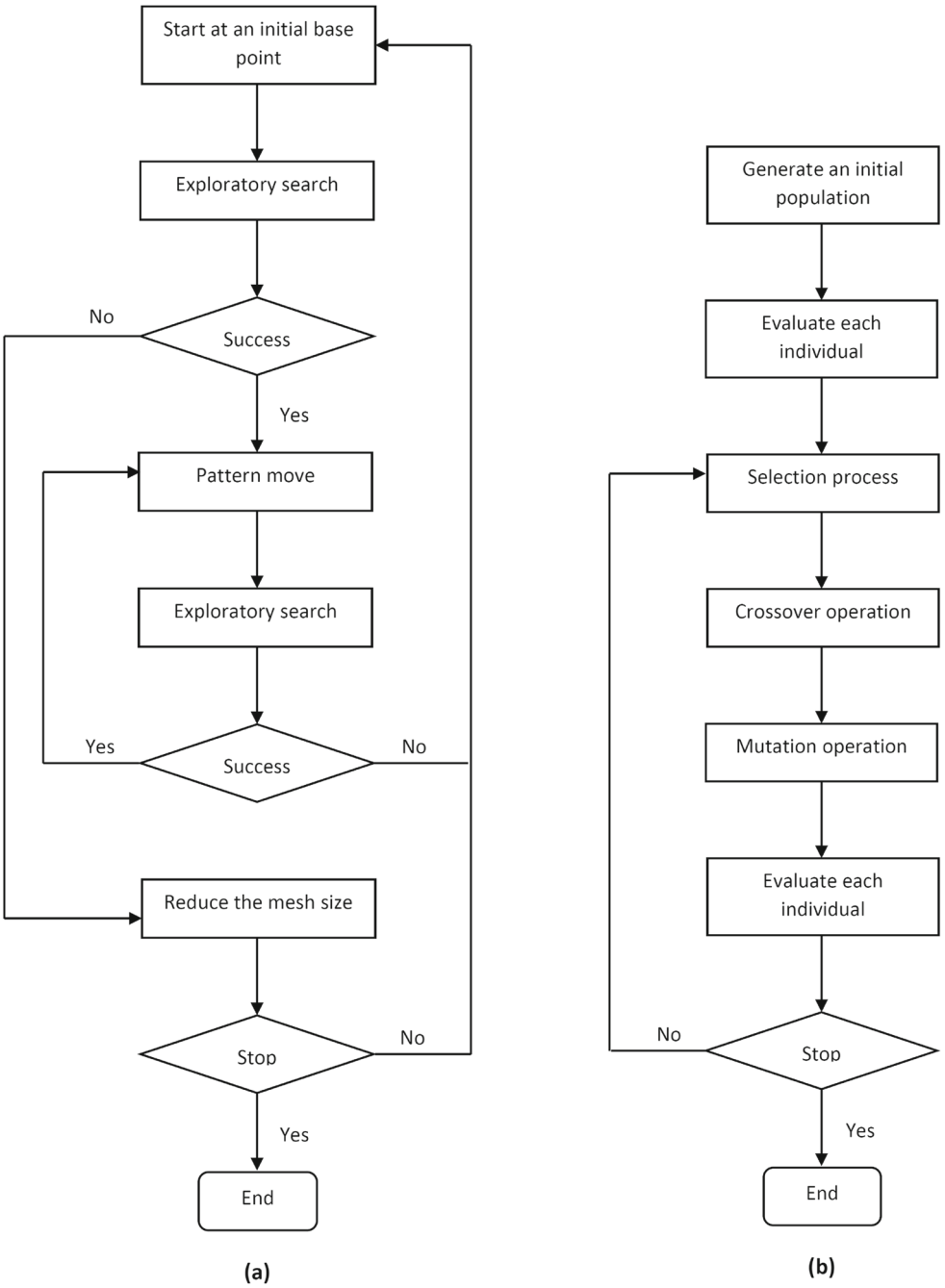

Fig. 6 (a) Flowchart of GPS-GBP. (b) Flowchart of GA-GBP

$$\lim_{k\to\infty} MP_f(G) = \lim_{k\to\infty} B_f(m)^k \left[\left(\frac{B_f(1)}{B_f(m)} \right)^k + \left(\frac{B_f(2)}{B_f(m)} \right)^k + \ldots + 1 + \ldots + \left(\frac{B_f(n)}{B_f(m)} \right)^k \right]. \quad (8)$$

Now, for $i \neq m$ & $i = 1 \ldots n$, we have:

$$\frac{B_f(i)}{B_f(m)} < 1 \quad , \quad \lim_{k\to\infty} \left(\frac{B_f(i)}{B_f(m)} \right)^k = 0. \quad (9)$$

Thus, we can conclude that

$$\lim_{k\to\infty} MP_f(G) = B_f(m)^k = B_f(G)^k. \quad (10)$$

For a general case in which there is a set of vertices with maximum bandwidth (take its cardinality as α), the equation above reads as follows:

$$\lim_{k\to\infty} MP_f(G) = \alpha B_f(m)^k = \alpha B_f(G)^k. \quad (11)$$

$$MP_f(G) = \begin{cases} P_f(G) & \text{if} k = 1 \ , \\ \alpha B_f(G)^k & \text{if} k \to \infty \ . \end{cases} \quad (12)$$

As it is clear, the limit of $MP_f(G)$ as k approaches infinity is the bandwidth of G raised to the power of k. Also, if k is equal to 1, then $MP_f(G)$ will be the profile of G. This is interesting because, in fact, the proposed new metric (i.e., the MP) appropriately incorporates information about both the bandwidth and profile at the same time, enabling us to control the dominance of the bandwidth over the profile and vice versa by increasing and decreasing the power k. Therefore, if the aim is to optimize the bandwidth and profile simultaneously, there is no need to define the problem as a multi-objective optimization problem, which is highly desirable and advantageous.

Also, in the case of using the MP as a metric returned by a fitness function for a heuristic search applied to the GBP, it is more informative than the bandwidth itself. The reason is that the MP indicates the quality of solutions and also encapsulates an indication of how far a particular solution is from better solutions in the search space.

In Section 3, the problem with the use of the bandwidth as a fitness measure was shown by an illustrative example. In this example, if the MP (with $k = 2$) is used instead, then for labeling 1 in Fig. 2, $MP_f(G) = 11$; for labeling 2 in Fig. 3, $MP_f(G) = 22$; and for labeling 3 in Fig. 4, $MP_f(G) = 42$. Here, we see that labeling 1 is still the best solution as before. In terms of labeling 2 and labeling 3, we can easily choose labeling 2 (the second best solution) over labeling 3 with respect to their MP measures, which is impossible to do by considering their bandwidth measures. This is very important for heuristic search methods applied to the GBP because of enabling such methods to more precisely assess the quality of solutions and choose the most promising ones, resulting in a considerable increase in their effectiveness.

5 Numerical experiments

In this section, we describe the experiments performed on the proposed new metric (i.e., the MP) for the GBP. In order to evaluate the effectiveness of the MP, it was employed within a Genetic Algorithm (GA) [44] and a Generalized Pattern Search algorithm (GPS) [45] as their objective functions. These two algorithms are well-known heuristic search methods, differing in several respects, some of which are as follows.

The GPS algorithm does not explore the global structure of the objective function. Therefore, it can be attracted by a local or global minimum. The GPS algorithm forms a sequence of iterates, converging to a stationary point if the objective function is smooth. If the objective function is discontinuous, the GPS algorithm may fail at a discontinuity. For the GPS algorithm, the parameters are generally continuous. However, it is also possible to have discrete parameters.

The GA starts with a population of candidate solutions randomly distributed in the search space. This decreases the possibility of being trapped in a local minimum which is not global. However, it is impossible to know for certain about the convergence of the GA even on smooth objective functions. In fact, during the optimization process, the population of candidate solutions may collapse to a small subset of the search space. In such a situation, the GA can fail to find a minimum.

The reason we used two informed search methods, each of which belongs to a different class of search methodologies, was to verify the effectiveness of the proposed new metric in a more conclusive manner.

Our GA, called GA-GBP, is an implementation of the standard Genetic Algorithm described in [44], but we used permutation encoding to represent candidate solutions and adapt the algorithm to cope with the GBP. In GA-GBP, we also used the graph representation, introduced in [46], and the crossover operator, introduced in [47]. Our GPS algorithm, called GPS-GBP, is an implementation of the Generalized Pattern Search algorithm presented in [45] that we adapted it to deal with the GBP. Algorithms 1 and 2 provide a high-level description of these two algorithms, and Fig. 6 illustrates their flowcharts. For more details on the implementations, the interested reader is referred to the references given here.

Algorithm 1 GA-GBP.

1: Randomly generate an initial population of chromosomes, each representing a permutation of the vertices of a given graph.
2: **repeat**
3: Apply each permutation to the edge list (or any other graph data structure) of the graph and generate new edge lists.
4: Compute the fitness value of each chromosome using a fitness function.
5: Select chromosomes from the population with respect to their fitness values for participating in genetic operations.
6: Produce a new generation of chromosomes using crossover and mutation operators (i.e., genetic operators).
7: **until** a predetermined stopping criterion is reached.
8: Return the permutation represented by the best chromosome in the last generation.

Algorithm 2 GPS-GBP.

1: Start at an initial base point, which is an array of randomly generated floating point values between -1 and 1, representing a permutation of the vertices of a given graph.
2: Produce a pattern of points as plus and minus the coordinate directions, multiplied by a mesh size.
3: Center the produced pattern on the current point.
4: Compute the objective function for each point in the pattern.
5: **if** the minimum value obtained in the previous step is less than the value at the current point **then**
6: Choose the minimum point discovered to be the current point.
7: Multiply the mesh size by 2.
8: Go to step 2.
9: **else**
10: Divide the mesh size by 2.
11: **if** the mesh size is less than a threshold **then**
12: Stop the algorithm.
13: Return the permutation represented by the current point.
14: **else**
15: Retain the current point and go to step 2.
16: **end if**
17: **end if**

In the present study, we performed three sets of experiments to examine the effect of employing the MP as an objective function on the performance of the GA-GBP and GPS-GBP algorithms, as follows:

5.1 Initial testing

In this first stage, we used the Buckyball graph [48] (better known as a soccer ball, a Buckminster Fuller geodesic dome, or a 60-atom carbon molecule) as a test problem. This graph has 60 vertices and 90 edges as shown in Fig. 7.

To compare the proposed new metric (i.e., $MP_f(G)$) with the standard metric (i.e., $B_f(G)$), they were incorporated into the GA-GBP and GPS-GBP algorithms as their objective functions. In terms of the MP, three different values of the parameter k (i.e., from 1 to 3) were used. For each of the two algorithms, 30 independent runs were executed for each of the four objective functions employed (i.e., $MP_f(G)$ with $k = 1, 2, 3$ and $B_f(G)$), and the bandwidth and profile were calculated for every run.

The parameters of the runs related to the GA-GBP were as follows: Number of generations = 1000, Population size = 100, Crossover rate = 80%, Mutation rate = 10%, Reproduction rate = 9%, Elitism rate = 1%. Also, the parameters of the runs related to the GPS-GBP were as follows: Mesh size = 1, Expansion factor = 2, Contraction factor = 0.5, Max iterations = 100 * number of variables. Tables 1 and 2 show the results obtained for the experiments mentioned above.

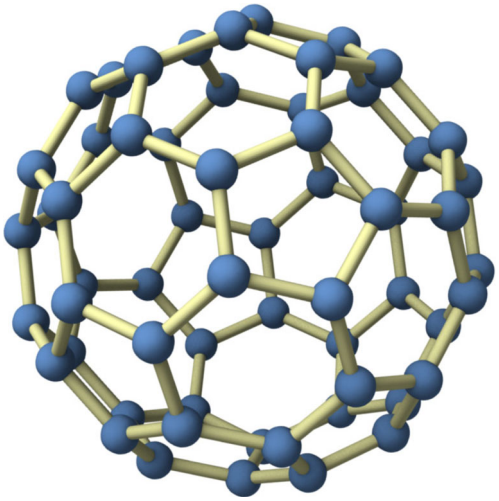

Fig. 7 The Buckyball graph

As described in Section 4, if $k = 1$, then the MP is the same as the profile. Also, by increasing the value of the parameter k (up to infinity), the MP gradually approaches the bandwidth of G raised to the power of k. However, it should be noted that by increasing the value of k, the term (or terms) related to a vertex (or vertices) with maximum bandwidth in (6) grows considerably, which may not be desirable when the MP is used as a fitness function. The reason is that this will result in a strong bias towards such a term (or terms), and the search will mostly focus on reducing the bandwidth of such a vertex (or vertices) and ignore other vertices. Therefore, it is important to find an appropriate value for the power k.

Considering the results of the experiments reported in Tables 1 and 2, in particular the mean of the bandwidth values and the best results obtained (marked in bold) by both algorithms under test, it is clear that the proposed new metric is superior to the standard metric.

5.2 Advanced testing 1

This stage deepens the former by employing the **HB/can**, **HB/dwt** and **HB/bcspwr** sets from University of Florida Sparse Matrix Collection (available at: https://www.cise.ufl.edu/research/sparse/matrices/HB/index.html) as test problems on which to compare the $MP_f(G)$ against the $B_f(G)$.

The **HB/can** set consists of 18 sparse matrices arising from finite-element applications with sizes ranging from 24×24 to 1072×1072 (the complete set was used). The **HB/dwt** set consists of 30 sparse matrices from NASTRAN (i.e., a finite element analysis program) users working in

Table 1 A comparison of $MP_f(G)$ with $B_f(G)$, both incorporated into the GA-GBP as its objective function

	GA-GBP							
	SM		NM1		NM2		NM3	
Iteration	Bandwidth	Profile	Bandwidth	Profile	Bandwidth	Profile	Bandwidth	Profile
1	30	777	42	453	13	452	11	455
2	36	881	**27**	438	**11**	445	**10**	449
3	42	949	43	441	12	443	11	455
4	31	824	42	453	13	451	11	465
5	**29**	692	39	476	14	445	11	460
6	42	1002	32	441	**11**	445	**10**	449
7	46	1009	38	438	13	453	11	448
8	37	929	35	434	12	462	12	460
9	36	825	37	457	12	455	11	463
10	**29**	705	30	438	12	454	**10**	457
11	35	858	48	437	12	461	11	464
12	33	946	45	504	12	445	11	456
13	35	792	48	444	14	447	11	462
14	46	1033	31	438	12	450	11	466
15	43	964	28	435	12	450	**10**	451
16	35	893	40	470	13	447	11	458
17	40	979	36	445	**11**	447	11	467
18	34	851	48	444	14	467	11	461
19	**29**	769	35	496	13	455	11	458
20	40	961	39	438	12	461	12	456
21	35	792	31	433	12	454	11	451
22	36	858	35	442	12	448	12	455
23	38	1012	47	459	**11**	444	11	460
24	40	966	**27**	438	**11**	444	11	450
25	37	904	48	457	13	444	11	472
26	32	915	29	434	12	448	11	458
27	36	867	55	446	12	450	12	467
28	39	957	48	442	13	455	11	452
29	34	844	34	440	12	448	12	472
30	47	1119	31	446	12	448	11	453
Mean	36.73	895.77	38.26	448.57	12.27	450.60	**11.03**	458.33
Best	29		27		11		**10**	

SM: Standard Metric, NM: New Metric

SM: $B_f(G)$.

NM1: $MP_f(G)$ (with $k = 1$).

NM2: $MP_f(G)$ (with $k = 2$).

NM3: $MP_f(G)$ (with $k = 3$).

The best results are shown in bold

U.S. Navy, Army, Air Force and NASA laboratories. We employed two largest sparse matrices from this set (i.e., dwt_1242 and dwt_2680 with sizes 1242×1242 and 2680×2680 respectively). The **HB/bcspwr** set which includes matrices related to power networks, consists of 10 sparse matrices. We employed five largest sparse matrices from this set (i.e., bcspwr06, bcspwr07, bcspwr08, bcspwr09 and bcspwr10 with sizes 1454×1454, 1612×1612, 1624×1624,

Table 2 A comparison of $MP_f(G)$ with $B_f(G)$, both incorporated into the GPS-GBP as its objective function

	GPS-GBP							
	SM		NM1		NM2		NM3	
Iteration	Bandwidth	Profile	Bandwidth	Profile	Bandwidth	Profile	Bandwidth	Profile
1	38	982	37	453	14	484	12	457
2	**27**	712	35	496	14	470	15	556
3	32	875	27	468	18	455	13	508
4	36	1063	28	446	14	461	13	460
5	34	957	**26**	446	15	465	13	463
6	37	941	39	476	14	460	12	474
7	**27**	850	31	460	15	462	13	483
8	29	865	29	473	16	458	13	495
9	32	890	57	479	**12**	465	13	473
10	35	950	37	457	14	469	**11**	475
11	33	935	38	491	15	459	**11**	468
12	35	1046	29	449	13	460	12	464
13	32	978	29	451	14	453	14	476
14	34	1018	34	464	13	460	12	482
15	35	1021	31	433	15	464	12	472
16	33	940	48	457	15	468	14	479
17	31	961	33	478	17	484	13	477
18	34	854	**26**	462	14	462	13	487
19	33	831	40	470	13	466	13	472
20	31	929	33	501	17	459	21	676
21	33	875	45	503	16	457	12	470
22	35	980	46	456	15	496	12	475
23	32	891	36	448	14	453	13	483
24	34	1037	47	459	13	459	18	577
25	34	1034	39	447	16	507	14	518
26	33	920	45	475	15	471	13	478
27	33	919	30	456	13	454	13	464
28	36	1033	42	530	19	532	13	470
29	33	981	35	457	15	458	16	556
30	34	998	27	449	**12**	455	12	458
Mean	33.17	942.20	35.96	466.33	14.67	467.53	**13.30**	491.53
Best	27		26		12		**11**	

SM: Standard Metric, NM: New Metric

SM: $B_f(G)$

NM1: $MP_f(G)$ (with $k = 1$)

NM2: $MP_f(G)$ (with $k = 2$)

NM3: $MP_f(G)$ (with $k = 3$)

The best results are shown in bold

1723×1723 and 5300×5300, respectively). These sets are subsets of the Harwell–Boeing sparse matrix collection, and have been used extensively by researchers.

Identical to the initial testing stage, first, the $MP_f(G)$ and the $B_f(G)$ were incorporated into the GA-GBP and GPS-GBP algorithms as their objective functions. 30 independent runs (with the same parameter setting as before) were then carried out per algorithm per objective function per test problem, and the bandwidth were calculated for every run.

Table 3 A comparison of $MP_f(G)$ against $B_f(G)$ (both included in the GA-GBP and GPS-GBP algorithms) on a set of 25 instances from the University of Florida Sparse Matrix Collection

	GA-GBP(SM) Bandwidth		GA-GBP(NM) Bandwidth		GPS-GBP(SM) Bandwidth		GPS-GBP(NM) Bandwidth	
Instance	Mean	Best	Mean	Best	Mean	Best	Mean	Best
can_24	11.83	7	**6.10**	**5**	15.07	13	**6.86**	**5**
can_61	38.53	29	**21.17**	**16**	40.97	33	**24.26**	**17**
can_62	30.66	24	**11.57**	**8**	32.83	27	**13.60**	**10**
can_73	43.57	37	**24.97**	**22**	46.20	38	**25.70**	**22**
can_96	66.83	49	**24.67**	**20**	72.73	66	**28.17**	**23**
can_144	105.30	83	**55.80**	**38**	107.90	99	**60.80**	**47**
can_161	118.03	103	**70.20**	**42**	133.23	120	**74.20**	**59**
can_187	135.50	120	**61.87**	**44**	158.53	134	**82.57**	**67**
can_229	163.53	134	**108.83**	**85**	194.33	174	**118.20**	**87**
can_256	202.67	174	**157.17**	**137**	217.33	202	**171.77**	**154**
can_268	218.17	194	**162.37**	**125**	232.70	212	**170.20**	**135**
can_292	229.13	202	**182.00**	**156**	236.50	216	**214.33**	**159**
can_445	354.60	288	**257.43**	**217**	387.97	366	**287.82**	**252**
can_634	540.16	493	**475.50**	**407**	570.97	538	**510.20**	**454**
can_715	592.13	543	**547.46**	**405**	636.27	615	**548.73**	**435**
can_838	727.63	650	**615.96**	**520**	764.80	726	**670.30**	**575**
can_1054	911.17	836	**880.10**	**736**	967.37	933	**923.63**	**786**
can_1072	932.37	853	**863.20**	**794**	979.30	946	**924.00**	**829**
dwt_1242	1103.26	1065	**1067.63**	**949**	1158.03	1132	**1069.46**	**995**
dwt_2680	**2579.56**	2461	2582.83	**2428**	**2649.93**	**2606**	2659.83	2619
bcspwr06	1288.50	1241	**1159.86**	**1032**	1429.33	1402	**1409.63**	**1339**
bcspwr07	1438.23	1389	**1335.76**	**1197**	1554.40	1505	**1516.56**	**1488**
bcspwr08	1460.07	1409	**1390.67**	**1209**	1572.83	1525	**1499.46**	**1389**
bcspwr09	1553.10	1498	**1469.90**	**1273**	1632.16	1596	**1585.91**	**1524**
bcspwr10	**5064.83**	**4952**	5069.33	5009	**5184.93**	**5082**	5243.30	5191
Sum		18834		**16874**		20306		**18661**
Wins/Draws	2/0	1/0	**23/0**	**24/0**	2/0	2/0	**23/0**	**23/0**

SM: Standard Metric, $B_f(G)$

NM: New Metric, $MP_f(G)$

The best results are shown in bold

The results associated with the tests are summarised in Table 3. The results reported clearly reveal that the GA-GBP and GPS-GBP algorithms with the proposed new metric significantly outperform the GA-GBP and GPS-GBP algorithms with the standard metric, considering the sum of the best bandwidth values, the number of the best bandwidth values and the number of the best mean bandwidth values (shown in the "Wins/Draws" row) obtained from 30 runs for each benchmark problem.

In order to examine whether or not the relative performance differences observed in this set of experiments were statistically significant, the Wilcoxon signed-rank

Table 4 Statistical analysis of the results summarised in Table 3

Algorithm	Asymp. Sig. (2-tailed)	Exact Sig. (2-tailed)	Exact Sig. (1-tailed)
GA-GBP (SM) — GA-GBP (NM)	.000	.000	.000
GPS-GBP (SM) — GPS-GBP (NM)	.000	.000	.000

Table 5 A comparison of the GA-GBP against BRKGA, GRASP and RCM algorithms

Instance	Vertices	Original Bandwidth	GA-GBP		BRKGA		GRASP		RCM	
			Bandwidth	CPU time	Bandwidth	CPU time	Bandwidth	CPU time	Bandwidth	CPU time
dwt_209	209	184	**30**	**0.8132**	33	16	33	1	33	<0.01
gre_216a	216	36	**21**	**1.0702**	**21**	13	**21**	1	**21**	〃
dwt_221	221	187	**15**	**1.0443**	**15**	14	**15**	5	**15**	〃
impcol_e	225	92	**58**	**1.3734**	63	36	67	2	75	〃
dwt_245	245	115	**31**	**1.0631**	33	13	34	1	55	〃
bcspwr04	274	265	**34**	**1.2581**	37	15	37	1	49	〃
ash292	292	24	**24**	**1.3587**	**24**	18	**24**	1	32	〃
can_292	292	282	**49**	**1.4575**	57	27	60	1	72	〃
dwt_310	310	28	**13**	**1.5006**	**13**	20	**13**	1	15	〃
gre_343	343	49	**28**	**1.6592**	**28**	22	**28**	1	**28**	〃
dwt_361	361	50	**15**	**1.6677**	**15**	24	**15**	1	**15**	〃
dwt_419	419	356	**32**	**2.1130**	**32**	31	**32**	1	34	〃
bcsstk06	420	47	**48**	**3.0952**	**48**	116	49	3	50	〃
bcsstm07	420	47	**48**	**2.9491**	49	102	49	3	56	〃
impcol_d	425	406	**54**	**2.0038**	56	25	55	1	80	〃
hor_131	434	421	**64**	**3.5116**	66	143	65	5	84	〃
bcspwr05	443	435	**48**	**1.6439**	49	17	49	1	68	〃
can_445	445	403	**64**	**2.2644**	74	32	74	1	78	〃
494_bus	494	428	**54**	**1.7861**	55	17	55	1	82	〃
dwt_503	503	452	**48**	**3.0796**	51	66	52	2	64	〃
gre_512	512	64	**36**	**2.5944**	56	25	**36**	1	**36**	〃
fs_541_1	541	540	**376**	**2.6787**	465	181	433	117	533	〃
dwt_592	592	259	**40**	**3.0258**	**40**	45	42	1	42	〃
662_bus	662	335	**76**	**2.7446**	77	26	78	1	118	〃
fs_680_1	680	600	**17**	**4.1504**	**17**	50	20	1	20	〃
685_bus	685	550	**57**	**3.0404**	**57**	31	**57**	1	102	〃
can_715	715	611	**110**	**4.1374**	116	70	116	4	133	〃
fs_760_1	760	740	**39**	**4.6488**	**39**	146	41	4	43	〃
bcsstk19	817	567	**20**	**4.7696**	**20**	59	**20**	2	22	〃

Table 5 (continued)

Instance	Vertices	Original Bandwidth	GA-GBP		BRKGA		GRASP		RCM	
			Bandwidth	CPU time	Bandwidth	CPU time	Bandwidth	CPU time	Bandwidth	CPU time
bp_0	822	820	**384**	**4.9666**	386	92	417	2	422	″
bp_1000	822	820	458	**5.7741**	**457**	140	496	5	479	″
bp_1200	822	820	**466**	**5.8592**	467	142	488	5	506	″
bp_1400	822	820	476	**5.7947**	**475**	141	484	5	536	″
bp_1600	822	820	**469**	**5.9333**	**469**	144	488	5	539	″
bp_200	822	820	**426**	**5.2176**	471	108	459	3	508 ″	
bp_400	822	820	**435**	**5.3952**	480	115	479	5	530	″
bp_600	822	820	**443**	**5.4269**	488	121	485	4	541	″
bp_800	822	820	456	**5.8063**	**454**	131	492	5	542	″
can_838	838	837	**111**	**5.5924**	**111**	118	119	4	151	″
dwt_878	878	519	**33**	**5.1456**	**33**	64	35	3	46	″
gr_30_30	900	31	**48**	**5.5802**	**48**	65	**48**	4	59	″
dwt_918	918	839	**46**	**5.5383**	**46**	65	**46**	2	57	″
dwt_992	992	513	**52**	**8.3091**	**52**	194	**52**	5	65	″
Sum			**5852**		6143		6258		7036	
Wins/Draws			**21/19**		3/19		0/13		0/5	
Mean				**3.4614**		71		5		
Wins/Draws				**43/0**		0/0		0/0		

All CPU times are in seconds

Numbers in bold face are the best results

Table 6 Statistical analysis of the results summarised in Table 5

Algorithm	Asymp. Sig. (2-tailed)	Exact Sig. (2-tailed)	Exact Sig. (1-tailed)
GA-GBP — BRKGA	.000	.000	.000
GA-GBP — GRASP	.000	.000	.000
GA-GBP — RCM	.000	.000	.000

test was used for performing nonparametric statistical tests. The tests were carried out using the SPSS 16.0 software package, and the Exact method was used for calculating the significance levels of the statistics. The *p*-values obtained from the statistical analysis, summarised in Table 4, demonstrate that the performance differences caused by the use of the new metric are statistically highly significant.

The results of the numerical experiments reported in this section also support the theoretical conclusion previously presented in Section 4 and confirm that the proposed metric is capable of increasing the effectiveness of informed search methods designed to solve the GBP.

5.3 Advanced testing 2

In the second round of the experiments described in the previous section, we observed that the GA-GBP significantly outperforms the GPS-GBP in all cases. Therefore, we selected the GA-GBP to conduct more experiments and make comparisons with other GBP solvers as follows.

Unlike the previous experiments, in this set of experiments, the initial population in the GA-GBP was generated by performing a Breadth-First Search (BFS) algorithm on a given graph from randomly selected start vertices. We remind that the BFS is a very well-known uninformed search method for traversing a graph. This non-random population initialisation mechanism has been used in several bandwidth reduction studies mentioned in this paper (See Section 1). Based on these studies and our own, it might be unlikely that by using a random initialisation, optimal or near optimal solutions could be found. It should be noted that the first and second experiments described before were not designed to find optimal or near optimal bandwidth values for the test problems, but rather to disclose the weakness of the standard metric and to demonstrate the effectiveness of the proposed alternative metric.

In order to further assess the performance of the GA-GBP, including the proposed new metric as its objective function, we compared it against three different algorithms designed for bandwidth reduction, i.e., BRKGA (the most recent metaheuristic approach), GRASP (a high-performance multistart metaheuristic) and RCM (the most well-known graph theoretic-based search method). Each algorithm was tested on a set of 43 standard benchmark problems from the SuiteSparse matrix collection (available at: https://sparse.tamu.edu). This set was used in [43] for performance comparison.

In this series of experiments, an AMD Athlon(TM) processor operating at 2.2 GHz was used to run the GA-GBP. The parameters of the runs were as follows: Number of generations = 100, Population size = 100, Crossover rate = 90%, Mutation rate = 8%, Reproduction rate = 1%, Elitism rate = 1%. We report the best bandwidth obtained for each benchmark problem as well as the time needed to find the best bandwidth for each problem instance on this computer. It should be emphasised that all the results associated with the BRKGA, GRASP and RCM were taken from [43].

In Table 5, we report the bandwidth values resulting from the permutations generated by the GA-GBP, BRKGA, GRASP and RCM algorithms as well as their corresponding CPU times for the 43 test problems. With respect to the sum of the bandwidth values and the number of the best bandwidth values obtained, it is clear that the GA-GBP outperforms all three algorithms under test in terms of solution quality. The results of the Wilcoxon signed-rank tests, carried out using the SPSS 16.0 software package and summarised in Table 6, also confirms that there is a statistically significant difference between the performance of GA-GBP and the other three algorithms.

Considering the fact that BRKGA, GRASP and RCM were executed on an Intel(R) Core(TM) i7 processor operating at 4.2 GHz (as reported in [43]), we could not directly compare the CPU times of these algorithms (all rounded to the nearest integer) with the GA-GBP. However, since the processor used in [43] is substantially faster than the processor we used, by a simple inspection of the CPU times reported in Table 5, it can be concluded that the GA-GBP outperforms both metaheuristics under consideration (i.e., the BRKGA and GRASP), especially the BRKGA, in terms of execution speed. Note that the RCM is a graph theoretic-based search method. Such algorithms typically have a simple structure and can run hundreds to even hundreds of thousands of times faster than metaheuristics (e.g., the GA-GBP). Therefore, a runtime comparison between them is not reasonable.

The results obtained by the DRSA, a metaheuristic approach based on simulated annealing, is also presented in [43]. According to these results, the DRSA produces high quality solutions on the 43 benchmark problems used and outperforms the BRKGA, GRASP and RCM algorithms. The DRSA also outperforms the GA-GBP, our proposed algorithm. However, the disadvantage of DRSA is that it is very slow in execution. As reported in [43], the mean of the CPU times obtained by the DRSA is 470 seconds (rounded to the nearest integer) on an AMD Opteron(TM) operating at 2.2 GHz, while the mean of the CPU times obtained by the GA-GBP is 3 seconds (rounded to the nearest integer) on an AMD Athlon(TM) operating at 2.2 GHz. Since the two processors used are almost identical, we can reasonably compare the runtimes obtained by the DRSA and GA-GBP algorithms and conclude that the DRSA is approximately 156 times slower than the GA-GBP when applied to these 43 standard benchmark problems.

5.4 Discussion

In this work, we show that the standard metric used in previous studies on the GBP is not ideal for comparing the quality of solutions produced by search-based solvers and thus for guiding the search. We address this issue by proposing an alternative metric (i.e., the MP). The results of experiments reported here give clear evidence on the validity of the theoretical conclusions presented earlier and on the effectiveness of the MP in improving the performance of search-based GBP solvers.

The MP offers some advantages as follows: being more informative and providing better guidance for the search, incorporating information about both the bandwidth and profile at the same time, controlling the dominance of the bandwidth over the profile and vice versa by increasing and decreasing the power k, providing the possibility of optimizing the bandwidth and profile simultaneously without the need to define the problem as a multi-objective optimization problem, and encapsulating an indication of how far a particular solution is from better solutions in the search space.

The main disadvantage of using the MP for a heuristic search applied to the GBP is that an appropriate value for the power k should be found, otherwise this will result in a strong bias towards the term (or terms) related to a vertex (or vertices) with maximum bandwidth, and the search will mostly focus on reducing the bandwidth of such a vertex (or vertices) and ignore other vertices, which is not desirable.

In future work, we will investigate the possibility of improving the MP as well as developing a more robust metric for the GBP. With respect to the findings of the present study, we will also examine the standard metrics associated with other graph layout problems to see whether more effective alternatives can be introduced for solving these problems by means of informed search methods.

6 Conclusions

In this paper, a new metric for the Graph Bandwidth Problem has been introduced. This problem is proved to be NP-complete and has a significant number of applications in scientific and engineering domains. Therefore, many different types of algorithms have been developed for its solution. These algorithms are generally informed search methodologies in which a metric is needed for guiding the search towards good quality solutions. The most obvious and frequently used metric in earlier studies on the Graph Bandwidth Problem is the bandwidth itself. In this work, we have shown that this standard metric is not always appropriate for comparing the quality of solutions generated by different heuristic search methods, and its use may significantly reduce the performance of such methods. In order to tackle this issue, a new metric was proposed, and its effectiveness was tested and verified by a large number of numerical experiments on 68 standard benchmark problems from well-known sparse matrix collections. The results obtained were promising and showed that the proposed new metric was highly effective. Therefore, we believe that its incorporation into more sophisticated informed search methodologies previously used for addressing the Graph Bandwidth Problem may lead to further progress on these methods.

References

1. Papadimitriou CH, Steiglitz K (1982) Combinatorial optimization: Algorithms and complexity. Inc, Prentice-Hall
2. Díaz J, Petit J, Serna M (2002) A survey of graph layout problems. Comput Surv 34:313–356
3. Díaz J (1992) Graph layout problems. In: Proceedings of the 17th International Symposium on Mathematical Foundations of Computer Science. MFCS '92, pp. 14–23. Springer
4. Pissanetskey S (1984) Sparse matrix technology academic press
5. Chinn PZ, Chvátalová J, Dewdney AK, Gibbs NE (1982) The bandwidth problem for graphs and matrices — a survey. J Graph Theory 6(3):223–254
6. Davis TA Direct Methods for Sparse Linear Systems. Fundamentals of Algorithms. SIAM, Philadelphia (2006)
7. Deo N (2004) Graph theory with applications to engineering and computer science. PHI learning, Delhi, India
8. Golumbic MC, Hartman IBA (2005) Graph theory, Combinatorics and Algorithms: Interdisciplinary Applications. Operations Research/Computer Science Interfaces Series. Springer, USA

9. Papadimitriou CH (1976) The NP-completeness of the bandwidth minimization problem. Computing 16(3):263–270
10. Garey M, Graham R, Johnson D, Knuth D (1978) Complexity results for bandwidth minimization. SIAM J Appl Math 34(3):477–495
11. Davis TA, Hu Y (2011) The university of florida sparse matrix collection. ACM Trans Math Softw 38(1):1–1125
12. Russell S, Norvig P (2016) Artificial Intelligence: A Modern Approach. Always learning. Pearson, USA
13. Eiben AE, Smith JE (2015) Introduction to evolutionary computing. Springer, Berlin Heidelberg
14. Harary F Graph Theory. Addison-Wesley, Reading, Mass (1969)
15. Gurari E, Sudborough I (1984) Improved dynamic programming algorithms for bandwidth minimization and the min-cut linear arrangement problem. J Algorithm 5:531–546
16. Corso GD, Manzini G (1999) Finding exact solutions to the bandwidth minimization problem. Computing 62(3):189–203
17. Cygan M, Pilipczuk M (2012) Even faster exact bandwidth. ACM Trans Algorithm 8(1):8–1814
18. Cuthill E, McKee J (1969) Reducing the bandwidth of sparse symmetric matrices. In: ACM National conference. Association for computing machinery, New York, pp 157–172
19. George JA Computer implementation of the finite element method. PhD thesis, Stanford, CA, USA (1971)
20. Gibbs NE, Poole WG, Stockmeyer PK (1976) An algorithm for reducing the bandwidth and profile of a sparse matrix. SIAM J Numer Anal 13(2):236–250
21. Everstine GC (1979) A comparison of three resequencing algorithms for the reduction of matrix profile and wavefront. Int J Numer Methods Eng 14:837–853
22. Wang Q, Shi X-W (2009) An improved algorithm for matrix bandwidth and profile reduction in finite element analysis. Prog Electromagn Res Lett 9:29–38
23. Gonzaga de Oliveira SL, Chagas GO, Robaina DT, Brandão DN, Kischinhevsky M (2018) A modified bandwidth reduction heuristic based on the wbra and george-liu algorithm. In: Shi Y, Fu H, Tian Y, Krzhizhanovskaya VV, Lees MH, Dongarra J, Sloot PMA (eds) Computational science – ICCS 2018. Springer, Cham, pp 416–424
24. Gonzaga de Oliveira SL, Abreu AAAM, Osthoff C, Henderson Guedes de Oliveira LN (2019) A variant of the george-liu algorithm. In: Misra S, Gervasi O, Murgante B, Stankova E, Korkhov V, Torre C, Rocha AMAC, Taniar D, Apduhan BO, Tarantino E (eds) Computational science and its applications – ICCSA 2019. Springer, Cham, pp 3–12
25. Marti R, Laguna M, Glover F, Campos V (2001) Reducing the bandwidth of a sparse matrix with tabu search. Eur J Oper Res 135(2):450–459
26. Lim A, Rodrigues B, Xiao F (2003) Integrated genetic algorithm with hill climbing for bandwidth minimization problem. In: Cantú-Paz E et al (eds) Proceedings of the Genetic and Evolutionary Computation Conference (GECCO). LNCS, vol. 2724, pp. 1594–1595. Springer, Berlin, Heidelberg
27. Pop P, Matei O, Comes C-A (2014) Reducing the bandwidth of a sparse matrix with a genetic algorithm. Optimization 63(12):1851–1876
28. Piñana E, Plana I, Campos V, Martí R (2004) GRASP And path relinking for the matrix bandwidth minimization. Eur J Oper Res 153(1):200–210
29. Lim A, Lin J, Rodrigues B, Xiao F (2006) Ant colony optimization with hill climbing for the bandwidth minimization problem. Appl Soft Comput 6(2):180–188
30. Lim A, Lin J, Xiao F (2007) Particle swarm optimization and hill climbing for the bandwidth minimization problem. Appl Intell 26(3):175–182
31. Czibula G, Czibula I-G, Pintea C-M A reinforcement learning approach for solving the matrix bandwidth minimization problem. Studia Universitatis Babes-Bolyai Informatica **55**(2) (2010)
32. Czibula G, Crişan G-C, Pintea C-M, Czibula I-G (2013) Soft computing approaches on the bandwidth problem. Informatica 24(2):169–180
33. Koohestani B, Poli R (2010) A genetic programming approach to the matrix bandwidth-minimization problem. In: Schaefer R, Cotta C, Kolodziej J, Rudolph G (eds) parallel problem solving from nature, PPSN XI. Lecture notes in computer science, vol. 6239. Springer, Berlin, Heidelberg, pp 482–491
34. Koohestani B, Poli R (2011) A hyper-heuristic approach to evolving algorithms for bandwidth reduction based on genetic programming. In: Bramer M, Petridis M, Nolle L (eds) Research and development in intelligent systems XXVIII. Springer, Cambridge, England, pp 93–106
35. Pop PC, Matei O (2011) An improved heuristic for the bandwidth minimization based on genetic programming. In: Proceedings of the 6th International Conference on Hybrid Artificial Intelligent Systems - Volume Part II. HAIS'11. Springer, Berlin, Heidelberg, pp 67–74
36. Torres-Jimenez J, Izquierdo-Marquez I, Garcia-Robledo A, Gonzalez-Gomez A, Bernal J, Kacker RN (2015) A dual representation simulated annealing algorithm for the bandwidth minimization problem on graphs. Inf Sci 303:33–49
37. Chagas GO, de Oliveira SLG (2015) Metaheuristic-based heuristics for symmetric-matrix bandwidth reduction: A systematic review International Conference On Computational Science, ICCS 2015, vol 51. Procedia Computer Science, pp 211–220
38. Gonzaga de Oliveira SL, Bernardes JAB, Chagas GO An evaluation of low-cost heuristics for matrix bandwidth and profile reductions. Computational and Applied Mathematics (2016)
39. Mafteiu-Scai LO, Mafteiu-Scai E, Voina T Bandwidths optimization on sparse matrices using brain storm optimization. In: 2017 19th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), pp 219–224 (2017). IEEE
40. Gonzaga de Oliveira SL, Silva LM (2020) An ant colony hyperheuristic approach for matrix bandwidth reduction. Appl Soft Comput 94:106434
41. Gonzaga de Oliveira SL, Silva LM (2020) Evolving reordering algorithms using an ant colony hyperheuristic approach for accelerating the convergence of the iccg method. Eng Comput 36(4):1857–1873
42. Gonzaga de Oliveira SL, Silva LM (2021) Low-cost heuristics for matrix bandwidth reduction combined with a hill-climbing strategy. RAIRO-Oper Res 55(4):2247–2264
43. Silva PHG, Brandão DN, Morais IS, Gonzaga de Oliveira SL (2020) A biased random-key genetic algorithm for bandwidth reduction. In: Gervasi O, Murgante B, Misra S, Garau C, Blečić I, Taniar D, Apduhan BO, Rocha AMAC, Tarantino E, Torre CM, Karaca Y (eds) Computational Science and Its Applications – ICCSA 2020. Springer, Cham, pp 312–321
44. Goldberg DE (1989) Genetic Algorithms in Search, Optimization and Machine Learning, 1st edn. Addison-wesley Longman Publishing Co. Inc., Boston, MA, USA
45. Hooke R., Jeeves TA (1961) "direct search" solution of numerical and statistical problems. J ACM 8(2):212–229

46. Koohestani B (2020) A graph representation for search-based approaches to graph layout problems. Int J Comput Sci Eng 21(3):429–436
47. Koohestani B (2020) A crossover operator for improving the efficiency of permutation-based genetic algorithms. Expert Syst Appl 113381:151
48. Chung F, Sternberg S (1993) Mathematics and the buckyball. Am Sci 81(1):56–71

Behrooz Koohestani received the M.Sc. degree in information technology from Heriot-Watt University, Edinburgh, U.K., in 2008, and the Ph.D. degree in computer science from the University of Essex, Colchester, U.K., in 2013. He is currently an Assistant Professor with the Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran. His research interests include artificial intelligence, evolutionary computation and its applications to significant scientific, engineering, and commercial problems.