RAIRO-Oper. Res. 55 (2021) 2247–2264
https://doi.org/10.1051/ro/2021102

RAIRO Operations Research
www.rairo-ro.org

LOW-COST HEURISTICS FOR MATRIX BANDWIDTH REDUCTION COMBINED WITH A HILL-CLIMBING STRATEGY

Sanderson L. Gonzaga de Oliveira* and Libério M. Silva

Abstract. This paper studies heuristics for the bandwidth reduction of large-scale matrices in serial computations. Bandwidth optimization is a demanding subject for a large number of scientific and engineering applications. A heuristic for bandwidth reduction labels the rows and columns of a given sparse matrix. The algorithm arranges entries with a nonzero coefficient as close to the main diagonal as possible. This paper modifies an ant colony hyper-heuristic approach to generate expert-level heuristics for bandwidth reduction combined with a Hill-Climbing strategy when applied to matrices arising from specific application areas. Specifically, this paper uses low-cost state-of-the-art heuristics for bandwidth reduction in tandem with a Hill-Climbing procedure. The results yielded on a wide-ranging set of standard benchmark matrices showed that the proposed strategy outperformed low-cost state-of-the-art heuristics for bandwidth reduction when applied to matrices with symmetric sparsity patterns.

Mathematics Subject Classification. 05C78, 90C27.

Received February 19, 2021. Accepted July 9, 2021.

1. Introduction

The solution of large-scale sparse linear systems $Ax = b$, where $A = [a_{ij}]$ is an $n \times n$ large-scale sparse matrix, x is the unknown n-vector solution, and b is a known n-vector, is essential in various application areas in science and engineering, such as computational fluid dynamics (CFD), electromagnetics, structural, thermal, and elsewhere. The solution of large-scale sparse linear systems is commonly the simulation step that requires the highest running times.

A satisfactory ordering of rows and columns is beneficial for the low-cost solution of large and sparse linear systems by iterative and direct methods. Hence, the reordered coefficient matrix A will have a narrow bandwidth. Modern hierarchical memory architecture and paging policies favor programs that consider the locality of reference when using iterative methods for solving linear systems. Using a heuristic for matrix bandwidth reduction is an important choice to produce a sequence of graph (of the underlying matrix A) vertices with the spatial locality. Therefore, practitioners employ heuristics for bandwidth reduction to provide low processing costs for solving large sparse linear systems by iterative methods [19,20]. Bandwidth reduction is also employed

Keywords. Bandwidth reduction, sparse matrix, ant colony optimization, hyper-heuristic, reordering algorithms, renumbering, ordering, graph labeling, graph algorithm, local search procedure, Hill-Climbing procedure.

Universidade Federal de Lavras, Lavras, Brazil.
*Corresponding author: sandersongonzaga@gmail.com

in other fields, such as in the context of browsing hypertext [3], small world networks [21], visual analysis of data sets using visual similarity matrices [25], graph entropy rate minimization [4], symbolic model checking [1], mesh layout optimization [6] and the seriation problem [7]. In the present study, we concentrated on reducing the execution costs of solving systems of linear equations. Thus, the reordering time (*i.e.*, a preprocessing step of the matrix A) added to the running time of the linear system solver must be shorter than the running time of the solver without reordering, at least when using only a single vector b. Thus, the practical reordering algorithms in this context are fast approximate algorithms.

Heuristics for bandwidth reduction arrange nonzero coefficients of a sparse matrix as close to the main diagonal as possible. Let $A = [a_{ij}]$ be an $n \times n$ matrix corresponding to a graph $G = (V, E)$ composed of a set of vertices V and a set of edges E. The bandwidth of row i is $\beta_i(A) = i - \min_{1 \leq j \leq i}[j \mid a_{ij} \neq 0]$. The overall bandwidth of a matrix A is $\beta(A) = \max_{1 \leq i \leq n}[\beta_i]$. Equivalently, the bandwidth of G for a vertex labeling $S = \{s(v_1), s(v_2), \cdots, s(v_{|V|})\}$ (*i.e.*, a bijective mapping from V to the set $\{1, 2, \cdots, |V|\}$) is $\beta_S(G) = \max_{\{v,u\} \in E}[|s(v) - s(u)|]$, where $s(v)$ and $s(u)$ are labels of vertices v and u, respectively. The bandwidth minimization problem is a well-known $\mathcal{NP}$-hard problem [26]. A typical algorithm for the problem is heuristic in the sense that it attempts to attain a labeling that deliver a small bandwidth at low cost even for large-scale matrices.

An efficient solution of linear systems minimizes the total computational cost, including the reordering time, at least when only a single linear system is to be solved. Consequently, heuristics for bandwidth reduction must be capable of delivering high-quality solutions at a low cost. Since the mid-1960s, practitioners have proposed heuristics for bandwidth reduction [5,19]. Previous publications [5,15,17,19,20] have reviewed various heuristics and have indicated the most promising low-cost heuristics for bandwidth reduction so that they can be used in a preprocessing step when solving linear systems [17]. In this context, practical heuristics for bandwidth reduction compute at low cost and yield reasonable bandwidth results [17,19,20].

Our motivation in the present study is to obtain even better results than those delivered by low-cost state-of-the-art heuristics for bandwidth reduction. Additionally, the resulting method must still compute at a low cost. Thus, this paper applies a low-cost local search that delivers high-quality solutions to improve the results returned by up-to-date low-cost heuristics. Specifically, this article proposes an approach to reduce matrix bandwidth through low-cost heuristics, which are practical for large-scale problems, in tandem with an improved Hill-Climbing local search procedure. A hyper-heuristic based on ant colony optimization, modified in the present study, generates the heuristic and defines a parameter of the Hill-Climbing algorithm. Hyper-heuristics operate on a search space of heuristics instead of searching on the domain of problem solutions. Specifically, this paper modifies an ant colony hyper-heuristic (ACHH) approach for evolving heuristics for the bandwidth reduction of matrices [16,17]. We applied the Hill-Climbing algorithm proposed by Lim *et al.* [23]. The authors [23] specially designed this procedure for the bandwidth reduction problem.

We included four low-level heuristics for bandwidth reduction with the Hill-Climbing algorithm in the ant colony hyper-heuristic (ACHH) system. As a result, the modified ACHH system in the present study generates an expert-level heuristic for bandwidth reduction for each specific application domain studied. Additionally, the modified ACHH algorithm determines a specialized Hill-Climbing algorithm for the application area. To the best of our knowledge, no other research explored this approach, and thus this paper is the first (published) instance of this approach used in the field. Furthermore, this paper evaluates the resulting heuristics together with the Hill-Climbing algorithm evolved by the modified ACHH system in each application area against the most promising low-cost heuristics for bandwidth reduction.

The remainder of this paper is structured as follows. In Section 2, the related work is described. In Section 3, the modified hyper-heuristic method for bandwidth reduction is introduced. In Section 4, we describe the methodology to conduct the test. In Section 5, we describe the training stage and report the resulting heuristics from the modified ACHH system. In Section 6, the results are discussed. Finally, we address the conclusions in Section 7.

2. Related work

Practitioners have proposed an extensive amount of heuristics for bandwidth reduction since the 1960s. Previous publications [5,15,17,19,20,29] reviewed the literature. These studies, described below, indicated the heuristics for bandwidth reduction with the best performance proposed so far. We divide the design of heuristics for this problem into two main types. The first type, described in Section 2.1, comprises the design of low-cost heuristics based on graph theory concepts, and the second type, reported in Section 2.2, includes the design of metaheuristic-based algorithms for bandwidth reduction.

2.1. Graph-theory algorithms

A previous publication analyzed the results of 44 heuristics for bandwidth reduction based on graph theory concepts [15]. In short, the most successful graph-theory algorithms in this field is the Reverse Cuthill–McKee (RCM) method [14].

Koohestani and Poli [22] employed genetic programming (GP) metaheuristic to develop the KP-band heuristic. This GPHH approach for bandwidth reduction evolved the RCM method [14].

Graph-theory algorithms for bandwidth reduction are fast heuristics mainly applied to accelerate linear system solvers. A previous publication [20] verified that a reasonable bandwidth reduction is sufficient to accelerate linear system solvers. The principal characteristic is that the reordering algorithm must be fast. Thus, the publication indicated a reverse breadth-first search procedure with the starting vertex given by the George-Liu algorithm [14] (RBFS-GL for short) as one of the best heuristics to be used by practitioners in a preprocessing step of linear system solvers.

Recently, the authors of the present study showed that the resulting heuristics from an ant colony hyper-heuristic approach yielded better results than state-of-the-art reordering algorithms did when applied to symmetric positive definitive matrices. The objective was to reduce the execution costs of a linear system solver [16]. This research followed the current propensity for using a hyper-heuristic approach in challenging problems, which are difficult to be solved through traditional strategies. Thus, previous publications [16,17] proposed an ant colony hyper-heuristic (ACHH) approach for the bandwidth reduction aiming at accelerating a linear system solver. The hyper-heuristic system was provided as input a small set of matrices arising from an application domain. Then, the system generated a heuristic for the application area. The strategy yielded better results than previous low-cost state-of-the-art heuristics for bandwidth reduction (RCM, RBFS-GL, KP-band). Therefore, when considering the execution times as a relevant aspect of the analysis, the resulting heuristics from the ACHH approach are the current state-of-the-art heuristics for bandwidth reduction. The present study extends the previous investigation [17]. Specifically, the hyper-heuristic approach proposed in the present study employs a local search procedure.

2.2. Metaheuristic algorithms

The bandwidth reduction of matrices became a crucial problem because the solution of linear system solvers has connections with a wide range of other relevant problems in engineering fields. Thus, application programmers have employed the best-known metaheuristics to design heuristics for bandwidth reduction since the 1990s [5]. A previous study analyzed the results of 29 metaheuristic-based algorithms for bandwidth reduction [5].

Torres-Jimenez *et al.* [29] considered several metaheuristic algorithms for the problem. In this publication, the Dual Representation Simulated Annealing (DRSA) heuristic [29] outperformed the previous state-of-the-art metaheuristic algorithm concerning the solution quality when applied to 113 very small matrices with sizes ranging from 30 to 1104. Thus, currently, the state-of-the-art algorithm for bandwidth reduction is the DRSA heuristic [29].

The metaheuristic algorithms for the problem are slow, however. One can only apply these heuristics in a reasonable amount of time to rather small-sized matrices [20], *i.e.*, matrices with sizes of approximately 1000.

A relevant metaheuristic algorithm is the FNCHC heuristic [23]. The algorithm represents a fast implementation of a centroid-based approach combined with Hill Climbing to solve the bandwidth reduction problem.

The most promising metaheuristic algorithms [29] yielded better bandwidth results than the FNCHC heuristic did when applied to very small-sized matrices (with sizes of approximated 1000). Nevertheless, the FNCHC heuristic yielded consistent bandwidth results at shorter execution times than that metaheuristic algorithm [29]. Therefore, one can apply the FNCHC heuristic to medium-sized matrices (*i.e.*, matrices with sizes ranging from 100 000 to 1 000 000). FNCHC and NCHC use a parameter λ to determine a set of candidate vertices to have labels changed. The FNCHC heuristic used the parameter $\lambda = 0.95$, whereas the NCHC algorithm used this parameter with 1.0 when applied to all matrices [23].

Many papers justified designing a new high-cost heuristic for reducing the bandwidth of matrices because it is relevant in engineering applications. However, application programmers, who proposed slow algorithms for the problem, were unfamiliar with the need for the heuristic to have a low computational cost to accelerate linear system solvers [17]. A previous publication [20] evaluated several heuristics for bandwidth reduction. The study demonstrated that a metaheuristic algorithm is a noncontender for sparse matrix factorization and related problems. The reason was the long execution times of metaheuristic algorithms for the problem. Specifically, the study [20] evaluated several heuristics for bandwidth reduction to reduce the execution times of a solver of linear systems comprised of symmetric positive definite matrices. In this context, the heuristics for bandwidth reduction must compute at a low cost. Using a low-cost heuristic that reasonably reduces the bandwidth of the matrix is more practical than using a heuristic that highly reduces the bandwidth of the matrix but with long execution times and high memory requirements [20]. Thus, applications in this context require low-cost procedures. Otherwise, the linear system solvers are faster than the metaheuristic algorithm used to reduce the bandwidth of the matrix. The previous study [20] showed that a standard metaheuristic algorithm (that expands the search for better solutions, employs local search, etc.) for bandwidth reduction was not better than a low-cost method when the objective was to reduce the execution times of a linear system solver. Moreover, a metaheuristic algorithm may deliver better bandwidth reductions and, therefore, reduces the memory requirements of the linear system (depending on the data structure used). However, the potentially high memory consumption of such a metaheuristic algorithm may impede its application to workstations with a small main memory size [20], *i.e.*, only quite small-sized matrices can be used [17].

3. A modified ant colony hyper-heuristic approach for the matrix bandwidth reduction

In this section, we extend an ant colony hyper-heuristic approach for reordering the rows and columns of matrices [17]. Specifically, in this section, we incorporate a local search procedure into an ant colony hyper-heuristic method that generates and selects heuristics for the bandwidth reduction of symmetric and nonsymmetric matrices. Thus, we reproduce the ACHH approach in this section [17]. We refer to the modified ACHH system proposed in the present study as ACHH_{HC}.

We provided the ACHH_{HC} algorithm with the RCM [14], KP-band [22], and RLK heuristics [16, 17]. The ACHH_{HC} approach selects the RBFS heuristic as the resulting algorithm when the final formula generated is a constant real value. The RCM, KP-band, RBFS, and RLK heuristics differ by employing a priority formula to label vertices adjacent to the current vertex. The ACHH_{HC} approach uses the George-Liu (GL) [14] and Diagonal Dominance (DD) [31] algorithms for finding pseudoperipheral vertices.

In Sections 3.1 and 3.2, we briefly describe the ant colony optimization metaheuristic and the hyper-heuristic concept, respectively. In Section 3.3, we describe the structure and the pseudocode of the ACHH_{HC} algorithm.

3.1. The ant colony optimization metaheuristic

Ant colony optimization (ACO) is a metaheuristic in which a colony of articial ants collaborates on identifying satisfactory solutions to challenging discrete optimization problems. The technique uses relatively simple agents (called artificial ants) that communicate indirectly by stigmergy. An artificial ant in ACO is a stochastic constructive procedure that incrementally produces an approximate solution by adding opportunely defined solution components to a partial solution under development [10].

A model $P = (\mathbf{S}, \Omega, f)$ of a combinatorial optimization problem consists of a search space $\mathbf{S}$ defined over a finite set of discrete decision variables X_i, $i = i, \ldots, n$, a set Ω of constraints among the variables, and an objective function $f : \mathbf{S} \to \mathbb{R}_0^+$ to be minimized (maximizing a given objective function g from a maximization problem is equivalent to minimizing $f = -g$). The generic variable X_i takes values in $\mathbf{D}_i = \{v_i^1, \ldots, v_i^{|\mathbf{D}_i|}\}$. A feasible solution $\mathrm{s} \in \mathbf{S}$ is a complete assignment of values to variables that satisfies all constraints in Ω. A solution $\mathrm{s}^* \in \mathbf{S}$ is a global optimun iff $f(\mathrm{s}^*) \leq f(\mathrm{s})\ \forall \mathrm{s} \in \mathbf{S}$ [11].

Dorigo *et al.* [11] used the model of a combinatorial optimization problem to represent the pheromone model of ACO. A pheromone value is associated with each possible solution component. The authors denoted C as a set of all possible solution components.

An ant traverses the connected graph $G_C(V, E)$, where V is a set of vertices, E is a set of edges, and then provides a solution. Ants traverse vertices along edges, incrementally producing a partial solution. Ants deposit a certain amount of pheromone on vertices or edges. The amount of pheromone deposited may depend on the quality of the solution found. Subsequent ants employ the pheromone information as a guide toward promising regions of the search space [11].

ACO is a successful metaheuristic. There are an enormous amount of heuristic methods designed by this metaheuristic. Recently, Yang *et al.* [30] employed ACO to control a scattered field output of light passing through a turbid medium. Gan *et al.* [13] used ACO to propagate path optimization of product attribute design changes. Ma *et al.* [24] applied ACO for high-dimensional feature selection.

3.2. Hyper-heuristics

A hyper-heuristic is a high-level search approach that automates the process of selecting, combining, generating, or adapting one or several simpler heuristics or heuristic components. A hyper-heuristic explores a search space of low-level heuristics or heuristic components to solve computationally challenging problems [12]. A hyper-heuristic can also be defined to broadly describe the process of using (meta)heuristics to select (meta)heuristics or heuristic components to solve classes of instances of the problem at hand.

The resulting heuristic produced by a hyper-heuristic should efficiently yield excellent approximate solutions to the challenging computational search problem in context. Usually, a researcher designs a hyper-heuristic to build systems that produce satisfactory results to classes of instances of a given problem rather than solving just one occurrence of the problem (see [12] and references therein).

The number of hyper-heuristics published in recent years is enormous. Recently, Zhu *et al.* [32] proposed a genetic programming hyper-heuristic approach for the multi-skill resource-constrained project scheduling problem. Along the same lines, the authors of the present study proposed a genetic programming hyper-heuristic strategy for evolving low-cost heuristics for profile reductions [27]. Tian *et al.* [28] proposed an ACO-based hyper-heuristic with dynamic decision blocks for intercell scheduling.

3.3. Structure of ACHH$_{\mathrm{HC}}$

The ACHH$_{\mathrm{HC}}$ algorithm initializes a graph $C = (V_C, E_C)$ consisting of components of heuristics for bandwidth reduction. Figure 1 illustrates how we arranged the components (nodes) of the ACHH$_{\mathrm{HC}}$ algorithm. We show this graph in columns for clarity. The graph holds an initial node i, two nodes in column 1 that contain pseudoperipheral vertex finders, nodes in column 2, and nodes in column 3 that contain local search procedures. Using a local search procedure is a relevant difference in the implementation regarding previous publications [16, 17]. Nodes in column 2 refer to reordering algorithms to be used to label the vertices of a graph associated with a training matrix [17].

The ACHH$_{\mathrm{HC}}$ algorithm begins with the priority formulas of the RCM, KP-band, and RLK algorithms in nodes **f** in graph C. The algorithm builds new priority formulas randomly. The ACHH$_{\mathrm{HC}}$ system may either evolve or even again recover the priority formulas of the three original heuristics [17].

An ant determines a path to traverse graph C from node **i** to node **hc** using the probability function $\mathfrak{p}_{wj}^a(k) = \frac{\tau_{wj}}{\sum_{c \in \mathcal{N}_w} \tau_{wc}}$, where τ_{wj} is the pheromone trail of the node (component) w to node j and $\mathcal{N}_w$ is the

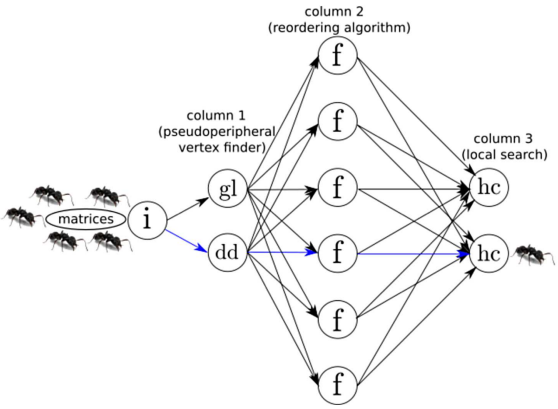

FIGURE 1. A graph that illustrates how the ACHH_{HC} algorithm organizes the components (nodes). Node **i** is the initial vertex. Nodes **gl** and **dd** (in column 1) consist of pseudoperipheral vertex finders. Nodes **f** (in column 2) comprise reordering algorithms. Nodes **hc** (in column 3) consist of local search procedures. Each edge stores a pheromone value. As an illustration, the ant next to node **hc** deposited pheromones on the blue edges of its traced path. The five other ants are ready to transport training matrices ("food") from node **i** to the nest located at nodes **hc**.

feasible neighborhood of node w. Moreover, $\mathcal{N}_w$ is a set comprised of components in the same column of node j (including j). The probability function is the ratio between the pheromone amount deposited on the edge that connects node (component) w to node j and the sum of the pheromone amount deposited on the edges between nodes w and nodes contained in $\mathcal{N}_w$. An ant a employs the probability function to decide which component of a heuristic for bandwidth reduction to include in its partial solution. Thus, the function is the probability that ant includes a node (component) to its trail from node **i** to node **hc** in iteration k. An ant deposits pheromones on this traversed path conforming to the quality solution and the running time taken by the individual program defined in the nodes of the path when applied to each training matrix [16, 17].

The ACHH_{HC} algorithm employs an evaporation rate to avoid a very fast convergence of the algorithm in a path from node **i** to node **hc** in graph C. The pheromone update is given by $\tau_{ij} = (1 - P) \cdot \tau_{ij} + g(\varrho)$, where $P \in (0, 1]$ is a parameter of the evaporation rate, $g()$ is the quality function of the pheromone deposition on edges, and ϱ is either the execution time of a reordering algorithm or the bandwidth reduction produced in a solution. We employed $P = 0.3$ [2].

The ACHH_{HC} algorithm employs two functions to indicate the quality of an individual heuristic and, hence, to update the pheromone amount in the edges of graph C. The first function is $1/t$, where t is the time (in milliseconds) that a component takes, and the second uses the bandwidth reduction yielded in a solution. The ACHH_{HC} algorithm uses $1/t$ to update the pheromone amount of the edge just traversed by an ant in graph C. The algorithm uses the quality solution to update the pheromone amount on the entire path walked by an ant. The objective was to generate low-cost heuristics that yield excellent bandwidth results. Choosing a high value to multiply the bandwidth found would produce high-cost heuristics. After some experiments, we empirically defined that the variable bandwidth would have a weight five times greater than the execution time to generate the resulting heuristic. Thus, the quality solution is five times more important than the running time. Therefore, the ACHH_{HC} system adds $5 \cdot \beta + 1/t$ to the pheromone amount on the entire path traversed by an ant [17].

The running times of the ACHH_{HC} depend on the number and size of the matrices used in the training phase. We decided that the running times of the ACHH_{HC} algorithm were up to two days for each application domain. After some experiments, we empirically defined the number of iterations. The ACHH_{HC} algorithm executed 100 000 iterations to generate or select new heuristics for bandwidth reduction for specific application areas.

The algorithm replaces half of the nodes in columns 2 and 3 with new components every ten iterations. The algorithm changes the content of the nodes that have incident edges with the smaller pheromone amount. We describe this characteristic in detail below. The ants choose nodes recently generated as components because the old nodes already have values of pheromone in their incident edges [17].

Algorithm 1 shows the ACHH_{HC} system. The ACHH_{HC} algorithm is provided as input the training set; the number of nodes in columns 1, 2, and 3 (see Fig. 1); the number of iterations; and a set of ants. The ACHH_{HC} algorithm creates graph C in line 1. The loop in lines 3 through 24 iterates the inner repetition loop in lines 4 through 19 for each ant. An ant defines its route in graph C using the probability function described above. At line 5, Algorithm 1 determines which nodes in columns 1, 2, and 3 the ant will use to traverse graph C.

Algorithm 1: An ant colony hyper-heuristic in tandem with a local search procedure for bandwidth reduction.

Input: a set of *matrices*, the number of nodes in columns 1, 2, and 3 in the graph of components, the number of iterations $NrIter$, a set of *ants*.
Output: an evolved heuristic for bandwidth reduction.

```
1  build a graph of components that contains the candidate nodes (components) of a resulting heuristic;
2  k ← 0;
3  while (k < NrIter) do
4      foreach (ant a ∈ ants) do
5          use the probability function to determine a path (nodes in columns 1, 2, and 3) for ant a to traverse graph C from
           node i to node hc;
6          foreach (matrix i ∈ matrices) do
7              compute the original bandwidth of matrix i;
8              find a pseudoperipheral vertex u using the node in column 1 determined at line 5;
9              build the level structure rooted at vertex u;
10             update the pheromone quantity of the edge that connects node i to the node in column 1 defined at line 5
               considering the time took by the pseudoperipheral vertex finder used in column 1;
11             compute the priority of every vertex employing the node f in column 2 determined at line 5;
12             label the vertices of matrix i according to the reordering heuristic contained in node f just traversed;
13             update the pheromone amount of the edge that connects the nodes in columns 1 and 2 considering the time to
               label matrix i;
14             execute the local search procedure stored in node hc defined at line 5;
15             update the pheromone amount in the edge that connects node f to node hc considering the time took by the
               Hill-Climbing algorithm;
16             compute the bandwidth of the current solution;
17             update the pheromone amount of edges of the path just traversed considering the bandwidth of the current
               solution;
18         end foreach
19     end foreach
20     k ← k + 1;
21     if (mod(k, 10) = 0) then
22         generate new formulas for half of the nodes in column 2 that have incident edges with less pheromone than the other
           nodes in column 2 and a new value for λ for the node in column 3 that has incident edges with the smallest
           pheromone amount;
23     end if
24 end while
25 return the pseudoperipheral vertex finder in column 1 with incident edges (considering node i and nodes in column 2) with
   the highest pheromone amount, the formula in node f in column 2 with incident edges (considering columns 1 and 3) with
   the highest pheromone amount, and value λ stored in node hc (in column 3) with incident edges with the highest pheromone
   amount.
```

The foreach loop in lines 6 through 18 processes the set of training matrices for each ant. The algorithm computes the original bandwidth of each matrix in line 7. After defined the pseudoperipheral vertex finder at line 8, Algorithm 1 creates the level structure rooted at u in line 9. Afterward, Algorithm 1 (at line 10) updates the pheromone quantity in the edge that connects the initial node **i** to a node in column 1 that contains a pseudoperipheral vertex finder considering $1/t$, where t (in ms) is the running time of the pseudoperipheral vertex finder. Algorithm 1 updates (at line 11) the priority and labels (at line 12) each vertex of graph G

employing the formula contained in the node of column 2 (selected at line 5). The algorithm (at line 13) then updates the pheromone quantity in the edge that connects a node in column 1 that contains a pseudoperipheral vertex finder to the current node in column 2 using the time took to compute the reordering algorithm [17].

Then, Algorithm 1 (at line 14) executes the local search procedure. Specifically, the algorithm employs a Hill-Climbing procedure [23]. Given a vertex labeling S, a pair of vertices are λ-critical if $|s(v) - s(u)| \geq \lambda \cdot \beta_S(G)$, where $\{v, u\} \in E$, $0 < \lambda \leq 1$, and $s(v)$ and $s(u)$ are labels of vertices v and u, respectively. Hence, the strategy defines a set $\Lambda = \{v \in V : |s(v) - s(u)| \geq \lambda \cdot \beta_S(G)\}$ for $\{v, u\} \in E$. For each vertex $v \in V$, the procedure defines $\text{mid}(v) = \frac{s_{\max}(v)+s_{\min}(v)}{2}$, where $s_{\max}(v) = \max_{\{v,u\}\in E}[s(v), s(u)]$ and $s_{\min}(v) = \min_{\{v,u\}\in E}[s(v), s(u)]$ are respectively the maximum and minimum labels of adjacent vertices to the vertex v, including v. The attribute $\text{mid}(v)$ is an approximated label to v that minimizes the difference between its label and the labels of its adjacent vertices [23]. The set $\mathcal{C}(v) = \{u : |\text{mid}(v) - s(u)| < |\text{mid}(v) - s(v)|\}$ contains the candidate vertices to have labels changed with vertex v. For each critical vertex $v \in \Lambda$, the procedure swaps $s(v)$ and $s(u)$ for each vertex u in $\mathcal{C}(v)$ in increasing order of $|\text{mid}(v) - s(v)|$. The process continues while the swap reduces $\beta_S(G)$. Thus, the ACHH_{HC} system seeks for a satisfactory λ value. After executed the local search procedure, Algorithm 1 (at line 15) updates the pheromone amount in the edge that connects node **f** to node **hc** considering the time took by the local search procedure.

The ACHH_{HC} algorithm computes the bandwidth of the solution at line 16. Algorithm 1 (at line 17) updates the pheromone amount of the entire path (defined at line 5) traversed by the ant, considering the bandwidth reduction of the solution; that is, the difference in bandwidth between the current solution and the original bandwidth of the training matrix (calculated at line 7). As previously mentioned, this update is multiplied by five [17]. Algorithm 1 increments variable k at line 20.

Priority formulas in the low-level heuristics in tandem with Hill Climbing need some iterations to be handled by artificial ants, depending on whether ants traverse them. The ACHH_{HC} system would have a high computational cost if it allowed low-level heuristics that do not produce satisfactory results to be evaluated by the system in a large number of iterations. After some experiments, we empirically decided to replace low-level heuristics and the parameter λ in Hill-Climbing every ten iterations. The reason was to adjust the number of iterations to evaluate the performance of the low-level heuristics concerning their output solution and the computational time of the ACHH_{HC} system. This choice of replacing heuristics in tandem with Hill Climbing every ten iterations allows the system to compute more low-level heuristics in tandem with Hill Climbing in the time required for the system to perform the number of iterations previously defined. Thus, every ten iterations (see line 21), the algorithm replaces the content of the three (half) nodes that have incident edges with less pheromone than the other three nodes in column 2 (at line 22). The algorithm also replaces the content of the node **hc** that has incident edges with less pheromone than the other node **hc**.

The algorithm returns at line 25 the pseudoperipheral vertex finder with incident edges with the highest pheromone amount. The same algorithm also returns the formula in node **f** (in column 2) with incident edges with the highest pheromone amount. Additionally, the algorithm returns the value λ contained in node **hc** in column 3 with incident edges with the highest pheromone amount.

4. Description of the tests

We implemented the algorithms using the C++ programming language. The g++ version 5.4.0 compiler was used, with the optimization flag −O3. We applied the RCM, RBFS-GL, and KP-band combined with Hill Climbing using $\lambda = 0.5$.

The workstation used in the executions of the simulations with matrices taken from the SuiteSparse matrix collection [8] featured an Intel® Core™ i7-4770 with 8 GB of main memory DDR3 1.333 GHz (Intel; Santa Clara, CA, United States). This machine used the Ubuntu 16.04.4 LTS 64-bit operating system with Linux kernel-version 4.2.0-36-generic.

Sections 5 and 6 use 27 and 30 matrices taken from the SuiteSparse matrix collection [8], respectively. The collection is an extensive and actively expanding set of sparse matrices arising from real applications. Furthermore,

the collection is extensively employed by the numerical linear algebra community for the design and performance evaluation of sparse matrix algorithms, allowing for robust and repeatable experimentation since performance results with artificially-generated matrices can be inaccurate. Moreover, the collection covers a broad spectrum of domains, including 2D or 3D geometric domains (structural engineering, CFD, electromagnetics, thermodynamics, etc.) [8].

Section 6 uses matrices from application areas with a sufficient number of matrices to train the ACHH_{HC} algorithm (see Sect. 5). We selected the four, three, six, eight, and five largest symmetric real matrices arising from 2D/3D, CFD, electromagnetics, structural, and thermal problems with sizes greater than 380 000, 120 000, 100 000, 700 000, and 80 000, respectively, to compare the results with state-of-the-art heuristics for bandwidth reduction. We also used the four largest real directed weighted graphs (with sizes smaller than three million vertices) in the SuiteSparse matrix collection.

In this field, a common method for evaluating the heuristic results is to compare its bandwidth results by counting the number of times that the heuristic yielded the narrower bandwidth on the matrices used. We also use this metric. Additionally, to analyze the quality of the bandwidth results yielded by the algorithms evaluated, for each algorithm H applied to each set of matrices, we calculate $\rho_\beta = \sum_{i=1}^{N} \frac{\beta_H(i) - \beta_{min}(i)}{\beta_{min}(i)}$, where $\beta_H(i)$ is the bandwidth yielded by algorithm H when applied to matrix i, $\beta_{min}(i)$ is the narrowest bandwidth delivered in matrix i (using the algorithms evaluated and the original bandwidth), and N is the number of matrices [19]. We also use ρ_t to evaluate the running times of the heuristics.

In addition to metric ρ_β, for each algorithm H applied to each dataset, we calculate $\upsilon_\beta = \sum_{i=1}^{N} \frac{\beta_H(i) - \beta_{min}(i)}{\beta_{max}(i)}$ for each matrix i, where $\beta_{max}(i)$ is the widest bandwidth yielded by the algorithms evaluated when applied to matrix i [18]. Metrics ρ_β and ρ_t employ function min because bandwidth reduction is a minimization problem. The metrics can use function max for maximization problems. Furthermore, the metric υ_β can also change the functions accordingly for maximization problems. In addition to these metrics, we also use Friedman (Friedman two-way analysis of variances by ranks), Iman-Davenport, Quade, and Friedman aligned ranks tests to analyze the results[1].

We also used the contrast estimation based on medians (CEBM) to interpret the results. CEBM contains contrast information of the medians of differences between performances of algorithms considering all pairwise comparisons. On the other hand, metric ρ_β contains detailed information from all instances and uses the best result from each instance among all algorithms evaluated. We used the structure of $A + A^T$ in the experiments with nonsymmetric matrices.

5. Training sets and the resulting heuristics

A previous publication [16] used matrices contained in the SuiteSparse matrix collection [8] with sizes greater than 1 000 000 to apply the resulting heuristics such that we could use a sufficient number of matrices from the same application area in the training stage of the ACHH_{HC} algorithm and with sizes greater than 2500. Thus, this present paper used (non)symmetric matrices from (two) four application areas in this matrix collection. Table 1 shows the matrices used in the learning process for the ACHH_{HC} system. We executed the ACHH_{HC} system with the training sets listed in Table 1 and recorded the best-of-run individual heuristic for each application domain. As a result, the ACHH_{HC} system selected the best-evolved heuristic for each training set. In all cases, the ACHH_{HC} algorithm selected the George-Liu algorithm to provide pseudoperipheral vertices to the reordering procedure.

The ACHH_{HC} system took two days to generate a heuristic for the bandwidth reduction of matrices arising from structural problems. The ACHH_{HC} system took one day in each case to produce heuristics for the bandwidth of matrices arising from the other application domains studied.

[1]Nonparametric statistical tests rely on a random sample. The reason is that a random sample is probably to be representative of the sampled population. We selected the highest matrices arising from specific application areas contained in the SuiteSparse matrix collection. Although forming a non-random sample, the sample represents the application domain matrix set. Then, the results of applying the nonparametric statistical tests are acceptable.

TABLE 1. Matrices used in the learning process for the ACHH_{HC} system.

Sparsity pattern	Kind	Matrix	n	$\lvert E\rvert$
Symmetric	Computational fluid dynamics	*Press_Poisson*	14 822	715 804
		ramage02	16 830	2 866 352
		copter1	17 222	211 064
		3dtube	45 330	3 213 618
		stokes128	49 666	558 594
		copter2	55 476	759 952
	Electromagnetics	*pli*	22 695	1 350 309
		mhd4800b	4800	27 520
		mhd3200b	3200	18 316
		qc2534	2534	463 360
	Thermal	*ted_B*	10 605	144 579
		lshp3466	3466	23 896
		lshp3025	3025	20 833
	Structural	*pkustk05*	37 164	2 205 144
		pwt	36 519	326 107
		bcsstk31	35 588	1 181 416
		Ship_001	34 920	3 896 496
		pkustk09	33 960	1 583 640
		bcsstk35	30 237	1 450 163
		bcsstm35	30 237	20 619
		thread	29 736	4 444 880
Nonsymmetric	2D/3D	*kim1*	38 415	933 195
		2D_27628_bjtcai	27 628	206 670
		shermanACb	18 510	145 149
	Directed weighted graph	*cage11*	39 082	559 722
		EAT_RS	23 219	325 592
		foldoc	13 356	120 238

The ACHH_{HC} algorithm developed new heuristics for bandwidth reduction since the priority formulas generated by the system were different from the priority formulas present in the literature. The simplified versions of the priority formulas evolved in the resulting heuristics for bandwidth reduction are as follows.

The ACHH_{HC} system selected, as the best individual program, a method consisting of a priority formula evolved from the RCM method for symmetric matrices originating from structural problems. In this case, the heuristic sorts the vertices adjacent to the current vertex in increasing order of degree using the formula $0.2084 \cdot (\text{Adj}(w))^6 + 0.86361 \cdot (\text{Adj}(w))^4 + 1.576708 \cdot n - 0.246786 \cdot (\text{Adj}(w))^3$. Afterward, the Hill-Climbing procedure computes the matrix employing $\lambda = 0.115115$.

For symmetric matrices arising from 2D/3D problems, the ACHH_{HC} system selected, as the best individual program, a method comprised of priority formulas evolved from the RLK heuristic. In this situation, the heuristic sorts the vertices adjacent to the current vertex in increasing order of degree using the formula $0.5417 \cdot (\mathfrak{d}(w))^2 - 0.94314 \cdot \mathfrak{d}(w) + 0.0134 \cdot n$, where $\mathfrak{d}(w)$ is the degree of vertex w considering only adjacencies to unlabeled vertices and n is the dimension of the matrix. The Hill-Climbing procedure then computes the matrix using $\lambda = 0.2455$.

The ACHH_{HC} algorithm selected as the best individual program a method comprised of a priority formula evolved from the KP-band heuristic such that the heuristic sorts the vertices adjacent to the current vertex in increasing order of degree using the formula:

– $(\varsigma(w))^4 - 0.23401 \cdot (\varsigma(w))^2 + 2.304308 \cdot n - \varsigma(w)$ and the Hill-Climbing procedure then computes the matrix employing $\lambda = 0.275483$ when applied to symmetric matrices arising from CFD problems,

- $(\varsigma(w))^4 - 0.6487 \cdot (\varsigma(w))^2 - 1.31478 \cdot (\varsigma(w))^4 \cdot n$ and the Hill-Climbing procedure then computes the matrix using $\lambda = 0.353461$ for symmetric matrices arising from electromagnetics problems,
- $0.26454 \cdot (\varsigma(w))^4 + 0.84624 \cdot (\varsigma(w))^2 - \varsigma + 0.3497$ and the Hill-Climbing procedure then computes the matrix employing $\lambda = 0.62537$ for symmetric matrices arising from thermal problems,
- ς and the Hill-Climbing procedure then computes the matrix using $\lambda = 0.4899$ for directed weighted graphs,

where $\varsigma(w)$ is the sum of degrees of vertices connected to vertex w.

6. Results and analysis

In this section, we show the bandwidth results yielded by the new heuristics for bandwidth reduction in tandem with the specialized Hill Climbing procedure for specific application domains, as described in Section 5. The ACHH$_{\text{HC}}$ system proposed in Section 3 generated the approaches. We compare the results yielded by the strategies with low-cost state-of-the-art heuristics for the bandwidth reduction of symmetric and nonsymmetric matrices arising from six application areas. Specifically, we evaluate the results with the RCM, RBFS-GL, and KP-band heuristics and the resulting heuristics from the ACHH algorithm [17] (*i.e.*, without employing the local search procedure). Additionally, we compare the results yielded by the new approaches with the RCM, RBFS-GL, and KP-band together with the Hill Climbing procedure. We refer to the RCM method combined with Hill Climbing as RCMHC. Along the same lines, we refer to the KP-band and RBFS-GL heuristics as KPHC and RBFSHC, respectively.

Table 2 shows the bandwidth results yielded by the resulting heuristics from ACHH$_{\text{HC}}$. Additionally, the table shows the bandwidth results delivered by the RCMHC, RBFSHC, and KPHC. The same table also reproduces the bandwidth solutions provided by the RCM, RBFS-GL, and KP-band heuristics and the resulting heuristics from the ACHH algorithm [17] (*i.e.*, without using Hill Climbing). Table 2 highlights in boldface the best results.

A previous publication [17] showed that the resulting heuristic from the ACHH system dominated the RCM, RBFS-GL, and KP-band heuristics (without using the local search procedure). Additionally, Table 2 shows that different approaches dominated the RCM, RBFS-GL, and KP-band heuristics without employing the local search procedure in each application area. Then, we concentrate the analysis to evaluate the results yielded by five approaches. We compare the resulting method generated by the ACHH$_{\text{HC}}$ system with four approaches: the resulting heuristics from the ACHH algorithm [17] (*i.e.*, without employing the local search procedure), and the RCM, RBFS-GL, and KP-band together with the Hill Climbing procedure. In Sections 6.1 and 6.2, we report the heuristic results when applied to symmetric and nonsymmetric matrices, respectively.

6.1. Symmetric matrices

We analyze the results yielded by five heuristics applied to symmetric matrices arising from four application domains in this section. In Sections 6.1.1 through 6.1.4, we report the heuristic results when applied to matrices arising from CFD, thermal, electromagnetics, and structural problems, respectively.

6.1.1. Matrices arising from CFD problems

Metrics ρ_β and υ_β in Table 2 show that the approach generated (from the priority formula employed in the KP-band heuristic) by the ACHH$_{\text{HC}}$ algorithm described in Section 3 yielded, in general, better bandwidth results than did the other approaches evaluated when applied to symmetric matrices arising from CFD problems. We ran the Friedman test to compare the results yielded by five methods: the resulting method from the ACHH$_{\text{HC}}$ system, RCMHC, RBFSHC, KPHC, and the resulting heuristic evolved by the ACHH algorithm [17] when applied to three matrices arising from CFD problems. The null hypothesis for the test was that the methods all provided identical results or that the samples differed in some means. The alternative hypothesis was that at least one method did yield different results. The total ranks for each of the five algorithms were 3 (ACHH$_{\text{HC}}$), 7.5 (ACHH), 8.5 (RBFSHC), 11.5 (KPHC), and 14.5 (RCMHC). Therefore, $\chi_F^2 = 10$. The nonparametric statistical test rejected the null hypothesis because $\chi_F^2 \geq 9.6$, which is the critical value when using $\alpha = 0.025$ and the $k = 5$ table, as that was how many methods we had.

TABLE 2. Results yielded by heuristics for bandwidth reduction applied to 30 matrices arising from six application areas contained in the SuiteSparse matrix collection [8].

Matrix	n	$\|E\|$	β_0	With Hill Climbing								Without Hill Climbing							
				ACHH		RCM		RBFS-GL		KP-band		ACHH		RCM		RBFS-GL		KP-band	
				β	t(s)	β	t(s)	β	t(s)	β	t(s)	β	t(s)	β	t(s)	β	t(s)	β	t(s)
Computational fluid dynamics problems																			
parabolic_fem	525 825	3 674 625	525 820	**510**	1.2	514	1.4	513	1.2	514	1.3	513	0.8	513	0.6	514	0.3	514	0.7
cfd2	123 440	3 085 406	4501	**2253**	1.6	2374	1.5	2353	1.6	2338	1.5	2270	0.3	2364	0.3	2386	0.2	2344	0.3
StocF	1 465 137	21 005 389	**36 616**	40 171	2.2	42 770	2.6	40 178	2.4	40 768	2.7	40 571	3.9	40 178	3.6	42 770	2.2	40 768	3.6
	ρ_β and ρ_t		1031.02	**0.01**	10	0.23	10	0.15	10	0.16	10	0.12	3	0.15	2	0.23	0	0.16	2
	υ_β		1.50	**0.08**		0.17		0.11		0.12		0.10		0.11		0.17		0.12	
Thermal problems																			
thermal1	82 654	574 458	80 916	**220**	2.0	**220**	2.2	233	2.1	232	2.3	222	0.2	228	0.1	240	0.1	238	0.2
thermomech_TC	102 158	711 558	102 138	**251**	1.2	256	1.4	**251**	1.2	262	1.4	**251**	0.3	256	0.2	**251**	0.1	262	0.2
thermomech_TK	102 158	711 558	102 138	**263**	1.6	**263**	1.5	264	1.6	266	1.4	279	0.3	**263**	0.1	268	0.1	271	0.2
thermomech_dM	204 316	1 423 116	204 276	**251**	1.6	261	1.5	277	1.6	271	1.4	**251**	0.5	261	0.2	277	0.1	271	0.3
thermal2	1 228 045	8 580 313	1 226 000	**857**	2.2	865	2.6	907	2.4	920	2.6	859	2.4	915	1.7	919	0.9	924	1.6
	ρ_β and ρ_t		3402.50	**0.00**	85	0.07	89	0.22	87	0.26	88	0.07	12	0.16	5	0.29	0	0.31	6
	υ_β		4.9903	**0.0000**		0.0001		0.0003		0.0004		0.0002		0.0002		0.0005		0.0006	
Electromagnetics problems																			
2cubes_sphere	101 492	1 647 264	100 407	**4608**	0.4	4699	0.2	4649	0.2	4811	0.4	4719	0.3	4812	0.2	4760	0.2	4927	0.3
offshore	259 789	4 242 673	19 627	**18 694**	1.0	19 580	1.0	20 763	0.5	18 714	1.0	19 606	0.8	20 535	0.8	21 776	0.4	19 627	0.8
gsm_106857	589 446	21 758 924	588 744	**16 933**	4.2	17 470	3.8	17 478	1.8	17 591	4.2	17 197	3.5	17 742	3.2	17 750	1.5	17 865	3.5
tmt_sym	726 713	5 080 961	1921	1153	0.8	**1129**	1.0	**1129**	0.6	1131	1.0	1142	0.7	1141	0.8	1141	0.5	1143	0.8
dielFilterV3real	1 102 824	89 306 020	1036475	27 140	10.2	**21802**	11.3	22467	4.2	23 585	11.3	25 102	8.5	23 729	9.4	24 453	3.5	25 670	9.4
dielFilterV2real	1 157 456	48 538 952	948 032	19 778	6.6	17 676	6.5	**16 265**	3.0	17 649	6.6	17 995	5.5	19 621	5.4	18 054	2.5	19 591	5.5
	ρ_β and ρ_t		159.14	0.48	8	0.19	8	**0.18**	1	0.25	9	0.36	6	0.50	6	0.49	0	0.57	6
	υ_β		4.34	0.02		0.04		0.10		**0.01**		0.06		0.10		0.16		0.06	
Structural problems																			
apache2	715 176	4 817 870	65 837	**3092**	2.0	3134	1.1	3405	0.7	3292	1.1	3367	1.7	3413	0.9	3707	0.6	3584	0.9
Emilia_923	923 136	40 373 538	17 279	15 873	3.6	15 497	3.6	**13 467**	2.4	14 962	3.6	14 669	3.0	16 883	3.0	14 672	2.0	16 300	3.0
Audikw_1	943 695	77 651 847	925 946	**34 550**	8.4	38 571	18.0	38 837	8.2	38 556	9.5	35 087	7.0	39 170	15.0	39 440	6.8	39 155	7.9
ldoor	952 203	42 493 817	686 979	10 077	4.8	10 221	4.8	**9698**	2.4	9766	4.8	10 318	4.0	10 466	4.0	9930	2.0	10 000	4.0
Serena	1 391 349	64 131 971	81 578	81 709	7.4	**80 997**	7.4	**80 997**	3.7	81 709	9.8	85 694	6.2	84 947	6.2	84 746	3.1	85 694	8.2
Geo_1438	1 437 960	60 236 322	26 018	24 382	7.2	24 451	7.2	29 395	3.6	27 521	7.2	**24 138**	6.0	24 700	6.0	29 695	3.0	27 802	6.0
Hook_1498	1 498 023	59 374 451	29 036	31 811	7.2	26 262	7.2	26 254	3.6	**26 075**	7.2	28 943	6.0	29 151	6.0	29 142	3.0	28 943	6.0
af_shell10	1 508 065	52 259 885	**2634**	4264	4.6	3624	4.8	3624	2.2	3624	4.6	3944	3.8	3944	4.0	3944	1.8	3944	3.8
	ρ_β and ρ_t		116.4	1.1	11	**0.7**	11	0.8	2	0.8	10	0.9	7	1.3	7	1.3	0	1.4	7
	υ_β		3.29	0.72		**0.37**		0.42		0.45		0.53		0.68		0.72		0.75	
2D / 3D problems																			
torso2	115 967	1 033 473	103 933	**595**	0.2	600	0.2	617	0.1	710	0.2	597	0.2	602	0.2	619	0.1	712	0.2
torso1	116 158	8 516 500	112 127	**3168**	1.4	3707	0.5	4070	0.4	3451	1.4	3181	1.2	3723	0.4	4087	0.3	3466	1.2
stomach	213 360	3 021 648	20 021	1135	0.7	**1131**	0.7	1132	0.4	**1131**	0.7	1133	0.6	1133	0.6	1134	0.3	1133	0.6
torso3	259 156	4 429 042	212262	6659	1.1	7074	1.1	**6516**	0.6	7395	1.1	6599	0.9	7139	0.9	6576	0.5	7463	0.9
	ρ_β and ρ_t		256.35	0.03	7	0.26	4	0.32	1	0.42	7	**0.02**	6	0.28	3	0.34	0	0.44	6
	υ_β		3.8788	0.0009		0.0075		0.0083		0.0078		**0.0006**		0.0081		0.0089		0.0083	
Directed weighted graphs																			
language	399 130	1 216 334	398 620	**141 159**	0.8	141 862	0.8	141 288	0.5	148 920	0.8	141 570	0.7	142 275	0.7	141 699	0.4	149 354	0.7
Cage13	445 315	7 479 343	206 333	43 637	2.4	43 892	2.4	43 741	1.3	**43 473**	2.4	43 821	2.0	44 078	2.0	43 926	1.1	43 657	2.0
webbase-1M	1 000 005	3 105 536	987 649	412 116	1.1	415 051	1.1	445 472	0.7	447 971	1.1	**408 400**	0.9	418 863	0.9	449 564	0.6	452 085	0.9
Cage14	1 505 785	27 130 349	676 026	206 726	8.4	**201 534**	8.4	204 363	4.8	201 534	8.4	206 416	7.0	201 837	7.0	204 671	4.0	211 378	7.0
	ρ_β and ρ_t		9.34	0.04	4	**0.03**	4	0.11	1	0.15	4	0.04	3	0.05	3	0.13	0	0.22	3
	υ_β		2.724	0.012		0.011		0.043		0.060		**0.010**		0.017		0.050		0.080	

TABLE 3. Multiple comparisons with the resulting method from the ACHH_{HC} system through Conover post-hoc test and FWER adjusted using Holm–Bonferroni (HB) method when using matrices arising from CFD problems.

Hyp.	Method	T-stat.	p-value	HB
H_1	RCMHC	6.6	0.0002	✓
H_2	KPHC	4.9	0.0012	✓
H_3	RBFSHC	3.2	0.0131	✓
H_4	ACHH	2.6	0.0317	✓

We also ran the Iman-Davenport test. The nonparametric statistical test returns $F_F = 10$. Thus, $F_F \geq F(4, 2\times 8) = 7$, which is the critical value at 1% significance level. In this case, p-value $= 0.003344$. Consequently, the Iman-Davenport test also rejected the null hypothesis of the equivalence of rankings at a high level of significance.

The Friedman and Iman-Davenport tests produced significant results. Subsequently, we conducted the Conover post-hoc test to pinpoint whether the pairwise groups have significant differences. The null hypothesis was that the algorithms yielded the same results. The alternative hypothesis was that the algorithms yielded different results. Afterward, we adjusted FWER applying the Holm–Bonferroni method (Holm's sequential Bonferroni procedure) at $\alpha = 0.05$. Table 3 shows that the method rejected the null hypothesis for H_1 (RCMHC), H_2 (KPHC), H_3 (RBFSHC), and H_4 (ACHH). Thus, the table shows that the results are significant for the four cases.

Section 5 proposed a new approach to reduce the bandwidth of symmetric matrices arising from CFD problems. The new heuristic is a variant of the KP-band heuristic executed in conjunction with the Hill-Climbing procedure with parameter λ specific to matrices arising from CFD problems. The ACHH_{HC} system proposed in the present study generated the new method. Thus, Table 3 shows that the new heuristic produced by the ACHH_{HC} approach dominated the resulting heuristic from the ACHH system [17] without using Hill Climbing. The same table shows that the new heuristic also dominated RCMHC, KPHC, and RBFSHC.

6.1.2. Matrices arising from thermal problems

Metrics ρ_β and υ_β in Table 2 show that the method evolved (also from the priority formula employed in the KP-band heuristic) by the ACHH_{HC} system delivered better bandwidth results than the other approaches evaluated when applied to all symmetric matrices originating from thermal problems. We also ran the Friedman test to compare the results yielded by five methods: the resulting heuristic from the ACHH_{HC} approach, RCMHC, RBFSHC, KPHC, and the resulting heuristic evolved by the ACHH algorithm [17] when applied to five matrices arising from thermal problems. The null hypothesis for the test was that the methods all provided identical results or that the samples differed in some means. The alternative hypothesis was that at least one method did yield different results. The total ranks for each of the five algorithms were 7.5 (ACHH_{HC}), 13 (RCMHC), 13.5 (ACHH), 19 (RBFSHC), and 22 (KPHC). Therefore, $\chi_F^2 = 11$. The nonparametric statistical test rejected the null hypothesis. The reason was that $\chi_F^2 \geq 9.0$, which is the critical value when using $\alpha = 0.05$ and the $k = 5$ table.

We also ran the Iman-Davenport test. The nonparametric statistical test returns $F_F = 4.163$, so p-value is 0.01688. Quade test returned $T_3 = 4.167$ and, therefore, $T_3 \geq F_F \geq F(4, 4\times 4) = 3$, which is the critical value at 5% significance level. Therefore, the Iman-Davenport and Quade tests also rejected the null hypothesis.

In this case, we also ran the Friedman aligned ranks test using the data sets delivered by metric ρ_β. Thus, all matrices are on the same scale. The total ranks for each of the five algorithms were 31.5 (ACHH_{HC}), 49.5 (ACHH), 54 (RCMHC), 90 (RBFSHC), and 100 (KPHC). Therefore, $T = 10.7$, so p-value is 0.03015 and $T \geq 9.488$, which is the critical value when using $\alpha = 0.05$ and four degrees of freedom. Consequently, the test also rejected the null hypothesis. The nonparametric statistical test produced significant results. Afterward, we

TABLE 4. Multiple comparisons with the resulting heuristic from the ACHH_{HC} system through a post-hoc test [9] and FWER adjusted using Li procedure at $\alpha = 0.05$ when using matrices arising from thermal problems.

Hyp.	Method	z	p-value	Li
H_1	KPHC	6.1	0.00001	✓
H_2	RBFSHC	5.2	0.00001	✓
H_4	RCMHC	2.0	0.04550	✓
H_3	ACHH	1.6	0.10960	✗

applied a post hoc procedure [9] to compare the resulting heuristic from ACHH_{HC} with the set of algorithms evaluated in the study. The null hypothesis was that the heuristics yielded the same results. The alternative hypothesis was that the heuristics yielded different results. Then, we adjusted FWER applying the Li procedure. Table 4 shows that the method rejected the null hypothesis for H_1 (KPHC), H_2 (RBFSHC), and H_3 (RCMHC). Thus, the table shows that the results are significant for three cases. Furthermore, the same table shows that the heuristic for bandwidth reduction generated by the ACHH_{HC} system dominated KPHC, RBFSHC, and RCMHC.

The statistical test did not reject the null hypothesis when determining whether the (ACHH_{HC}, ACHH) pairwise group has significant differences. Nevertheless, we can observe the results of the approaches in detail. The heuristic for bandwidth reduction developed by the ACHH_{HC} system yielded three better bandwidth results than the heuristic generated by the ACHH algorithm [17] without using Hill Climbing. The resulting heuristic from the ACHH_{HC} and the resulting heuristic from the ACHH algorithm [17] delivered the same bandwidth results when applied to the matrices thermomech_TC and thermomech_dM. Thus, metrics ρ_β and υ_β indicate that the heuristic for bandwidth reduction generated by the ACHH_{HC} approach proposed in the present study yielded better overall results than did the resulting heuristic from the ACHH algorithm introduced in our previous publication [17].

We also used the partial results from metric ρ_β to calculate CEBM, which found averages $M_{\text{ACHH}_{\text{HC}}} = -0.025$, $M_{\text{KPHC}} = -0.011$, $M_{\text{RCMHC}} = 0.001$, $M_{\text{RBFSHC}} = 0.007$, and $M_{\text{ACHH}} = 0.020$. Thus, CEBM confirms our previous analysis and also indicates that ACHH_{HC} is the best performance approach.

6.1.3. Matrices arising from electromagnetics problems

The total ranks calculated by the Friedman aligned ranks tests for each of the five algorithms analyzed in this section were 80.5 (RBFSHC), 81.5 (RCMHC), 87 (KPHC), 101 (ACHH_{HC}), and 115 (ACHH). The nonparametric statistical test showed slight differences between total ranks. Friedman, Iman-Davenport, Quade, and Friedman aligned ranks tests did not reject the null hypothesis in this case. Nevertheless, we can analyze the results of the approaches in detail.

The heuristic generated by the ACHH_{HC} algorithm returned the highest number of best results (in three out of six test problems). Metric $\upsilon_\beta = 0.02$ calculated with the results provided by the heuristic for bandwidth reduction created by the ACHH_{HC} system indicates that the method yielded overall bandwidth results almost as small as the best results, recalling that the highest bandwidth results returned by the algorithms attenuate this metric. However, the approach, applied to the matrices dielFilterV2real and dielFilterV3real, delivered higher bandwidth results than the other approaches. These unsatisfactory results affected metric $\rho_\beta = 0.48$. Thus, metrics ρ_β and υ_β in Table 2 show that the heuristic for bandwidth reduction (from the priority formula employed in the KP-band heuristic) produced by the ACHH_{HC} algorithm did not provide the best bandwidth results when applied to symmetric matrices deriving from electromagnetics problems.

Metric $\upsilon_\beta = 0.01$ in Table 2 suggests that KPHC yielded the most consistent bandwidth results. However, the heuristic returned unsatisfactory results when applied to the matrices dielFilterV2real, dielFilterV3real, 2cubes_sphere, and gsm_106857. The highest bandwidth results yielded by the algorithms attenuated the metric.

On the other hand, metric ρ_β is not bounded. Hence, the four unsatisfactory results influenced metric $\rho_\beta = 0.25$ calculated with the results delivered by KPHC.

Table 2 shows that RBFSHC, applied to the matrix offshore, delivered an unsatisfactory bandwidth result. In this case, the small original bandwidth did not attenuate metric $\upsilon_\beta = 0.1$. Metric $\rho_\beta = 0.18$ in the same table, on the other hand, shows that RBFSHC returned overall better bandwidth results than did the other methods evaluated. As previously mentioned, RBFSHC, applied to the matrix offshore, yielded an unsatisfactory result. Despite that, the strategy returned two out of six best bandwidth results. In particular, RBFSHC yielded five better bandwidth results than KPHC. Thus, we considered that the former delivered, in general, better bandwidth results than the latter at lower running times (see metric ρ_t in Tab. 2). Furthermore, RBFSHC yielded four out of six better results than the resulting heuristic generated from the ACHH system [17] without using Hill Climbing at lower running times (see metric ρ_t in Tab. 2).

We also calculated CEBM for this case. The averages were $M_{\text{RBFSHC}} = -0.008$, $M_{\text{RCMHC}} = -0.003$, $M_{\text{KPHC}} = 0.000$, $M_{\text{ACHH}_{\text{HC}}} = 0.004$, and $M_{\text{ACHH}} = 0.008$. CEBM confirms our previous analysis and also indicates that RBFSHC is the best performance approach. Thus, the local search procedure benefited the heuristic for bandwidth reduction.

6.1.4. Matrices arising from structural problems

The heuristic (from the priority formula employed in the RCM method) generated by the ACHH_{HC} algorithm returned the best results in two out of eight test cases when applied to symmetric matrices from structural problems. However, metrics ρ_β and υ_β in Table 2 show that the approach did not provide the best bandwidth results.

Metrics ρ_β and υ_β in Table 2 show that RCMHC provided overall better bandwidth results than did the other approaches evaluated when applied to matrices arising from symmetric structural problems. RCMHC yielded five out of eight better results than the resulting heuristic generated from the ACHH algorithm [17] without using Hill Climbing. Again, the local search procedure helped the heuristic for bandwidth reduction to find small bandwidth results.

Friedman, Iman-Davenport, Quade, and Friedman aligned ranks tests did not reject the null hypothesis also for this case. Thus, we calculated CEBM. The averages were $M_{\text{RBFSHC}} = -0.042$, $M_{\text{KPHC}} = -0.035$, $M_{\text{RCMHC}} = -0.028$, $M_{\text{ACHH}} = 0.007$, and $M_{\text{ACHH}_{\text{HC}}} = 0.097$. In this case, CEBM indicates that RBFSHC and KPHC yielded better results than RCMHC. However, RBFSHC and KPHC yielded an unsatisfactory solution when applied to the matrix Geo_1438, whereas RCMHC delivered a satisfactory solution when applied to the matrix. Metrics ρ_β and υ_β detect this type of result, but CEBM does not consider it. On the other hand, CEBM emphasizes that the Hill Climbing procedure benefited the heuristics for bandwidth reduction.

6.2. Nonsymmetric matrices

We analyze the results yielded by five heuristics applied to nonsymmetric matrices arising from two application domains in this section. Friedman, Iman-Davenport, Quade, and Friedman aligned ranks tests did not reject the null hypotheses when applied to the two cases. Nevertheless, we examine the results of the approaches in detail. In Sections 6.2.1 and 6.2.2, we report the heuristic results when applied to matrices arising from 2D/3D problems and direct weighted graphs, respectively.

6.2.1. Matrices arising from 2D/3D problems

The total ranks calculated by the Friedman aligned ranks tests for each of the five algorithms analyzed in this section were 28 (ACHH_{HC} and ACHH), 46 (RBFSHC), 50.5 (RCMHC), and 57.5 (KPHC). The test revealed slight differences between total ranks.

Metrics $\rho_\beta = 0.03$ and $\upsilon_\beta = 0.0009$ in Table 2 show that the heuristic (from the priority formula employed in the RLK heuristic) evolved by the ACHH_{HC} algorithm yielded small bandwidth results. Additionally, the approach returned the highest number of best results (in two out of four test problems).

Table 2 also shows that the heuristic created by the ACHH system in the previous publication [17] (without using of the Hill-Climbing procedure) returned no best bandwidth result. Despite that, metrics $\rho_\beta = 0.02$ and $\upsilon_\beta = 0.0006$ in the same table show that the heuristic yielded, in general, the most consistent results when applied to symmetric matrices arising from 2D/3D problems. Additionally, the heuristic yielded, in general, better bandwidth results than the heuristic generated by the ACHH_{HC} system at lower running times (see metric ρ_t in Tab. 2). Thus, we consider that the resulting heuristic from the ACHH algorithm proposed in the previous study [17] remains the most promising approach to reduce the bandwidth of matrices arising from 2D/3D problems.

We also calculated CEBM for this case. The averages were $M_{\text{ACHH}} = -0.032$, $M_{\text{ACHH}_{\text{HC}}} = -0.031$, $M_{\text{RBFSHC}} = -0.004$, $M_{\text{RCMHC}} = 0.007$, $M_{\text{KPHC}} = 0.059$. CEBM confirms our previous analysis and also indicates that the heuristic generated by the ACHH system [17] (without using the Hill-Climbing procedure) returned slightly better performance than the resulting method generated by ACHH_{HC}.

6.2.2. Directed weighted graphs

In the Friedman test, the total ranks for each of the five algorithms were 10 (ACHH_{HC}), 12 (ACHH), 12 (RBFSHC), 12.5 (KPHC), and 13.5 (RCMHC). The test showed tiny differences between total ranks. Then, we calculated CEBM. The averages were $M_{\text{ACHH}_{\text{HC}}} = -0.007$, $M_{\text{RCMHC}} = -0.005$, $M_{\text{ACHH}} = -0.004$, $M_{\text{RBFSHC}} = 0.001$, and $M_{\text{KPHC}} = 0.015$. CEBM indicates that the heuristic generated by the ACHH_{HC} system is the best performance approach.

Metrics $\rho_\beta = 0.04$ and $\upsilon_\beta = 0.012$ in Table 2 also show that the heuristic (from the priority formula employed in the KP-band heuristic) evolved by the ACHH_{HC} algorithm yielded small bandwidth results. Additionally, the approach returned one out of four best results. However, the same metrics show that RCMHC and the resulting heuristic from the ACHH system proposed in the previous study [17] produced slightly better bandwidth results than the heuristic generated by the ACHH_{HC} algorithm. The differences are very tiny, however. On the other hand, ACHH_{HC} yielded three out of four better results than RCMHC. Furthermore, ACHH_{HC} and the resulting heuristic from the ACHH system proposed in the previous study delivered the same number of best results.

Metric ρ_β in Table 2 shows that RCMHC produced overall better bandwidth results than did the other approaches evaluated when applied to directed weighted graphs. On the other hand, metric υ_β in the same table shows that the heuristic created by the ACHH algorithm in the previous publication [17] (without using the Hill-Climbing procedure) yielded, in general, better bandwidth results than did the other approaches evaluated. As previously mentioned, the latter returned three out of four better bandwidth results than the former. Additionally, the heuristic evolved by the ACHH algorithm in the previous publication [17] took slightly shorter times than the heuristics generated by the ACHH_{HC} system and RCMHC (see metric ρ_t in Tab. 2).

We conclude that employing Hill Climbing, applied to directed weighted graphs, did not sufficiently benefit the heuristics. Thus, we consider that the heuristic generated by the ACHH algorithm in the previous publication [17] (without using the Hill-Climbing procedure) also remains the most promising approach when considering directed weighted graphs.

7. Conclusions

This paper concentrated on the bandwidth reduction problem for large-scale sparse matrices in serial computations. Specifically, this paper extended a hyper-heuristic based on the ant colony optimization metaheuristic. The hyper-heuristic generates or selects heuristics for the bandwidth reduction of symmetric and nonsymmetric matrices. Specifically, we included a local search procedure in the approach. The investigation integrated low-cost state-of-the-art heuristics for bandwidth reduction with a specific Hill-Climbing algorithm. The experiments compared the resulting heuristics for bandwidth reduction generated by the ACHH_{HC} algorithm with low-cost state-of-the-art heuristics for this problem.

As in our previous approaches [16, 17], we provided the ACHH_{HC} algorithm with components of the central structure of the RCM, KP-band, RBFS-GL, and RLK heuristics. The ACHH_{HC} system develops a specific-purpose heuristic for the application problem.

We trained the ACHH_{HC} algorithm for classes of matrices arising from (non)symmetric matrices originating from (two) four application areas in the learning process. As a result of the learning process, the ACHH_{HC} system generated an expert-level method combined with the Hill-Climbing procedure with a specially defined parameter λ for each application area. Thus, the ACHH_{HC} algorithm in the present study created new expert-level set reordering algorithms in specific application domains. Although the training stage required approximately one day on a workstation, the resulting reordering algorithm is fast, even using the Hill-Climbing procedure. The resulting approach computes almost 90 million nonzero coefficients (in a matrix with a size larger than one million) in approximately 10 s on a workstation in single-core computations (see Tab. 2).

The experiments conducted in this paper showed that the specific Hill-Climbing procedure benefited the low-cost state-of-the-art heuristics for bandwidth reduction when applied to matrices with symmetric sparsity patterns. The RBFS-GL (RCM) heuristic combined with Hill Climbing delivered the best bandwidth results when applied to matrices arising from electromagnetics (structural) problems. Furthermore, the heuristics generated by the ACHH_{HC} algorithm yielded overall better bandwidth results when applied to matrices arising from two application areas (CFD and thermal problems). The resulting heuristics combined with the Hill-Climbing procedure evolved by the ACHH_{HC} algorithm are cheap and fast to implement, requiring less expertise in both the problem domain and state-of-the-art heuristic methods. Similarly, the RBFS-GL and RCM heuristics in tandem with the Hill-Climbing procedure are also fast and easy to implement. In particular, RBFS-GL with Hill-Climbing computed almost 90 million nonzero coefficients (in a matrix with a size larger than one million) in approximately four seconds on a workstation.

The results showed that the local search procedure, applied to nonsymmetric 2D/3D problems and direct weighted graphs, did not help the heuristics for bandwidth reduction. We plan to design a local search procedure that improves the bandwidth results yielded by the approach when applied to nonsymmetric matrices.

Executions in parallel can shift the balance between the time required to compute the labeling and the time taken by the linear system solver. We also intend to investigate the effects of orderings in parallel implementations of linear system solvers using OpenMP, Galois, and message passing interface systems. A systematic review of parallel heuristics for bandwidth and profile reductions is another future step of this investigation. We also plan to evaluate the new heuristics for bandwidth reduction implemented within the Intel® Math Kernel Library running on Intel® Scalable processors.

Acknowledgements. The CAPES - Coordenação de Aperfeicoamento de Pessoal de Nível Superior supported this work. We thank the anonymous referees for their careful reading and their many insightful comments and suggestions.

References

[1] E.G. Amparore, M. Beccuti and S. Donatelli, Gradient-based variable ordering of decision diagrams for systems with structural units, edited by D. D'Souza and K. Narayan Kumar. In: Vol. 10482 of *ATVA 2017. Lecture Notes in Computer Science*. Springer, Cham (2017) 184–200.

[2] Z.A. Aziz, Ant colony hyper-heuristics for travelling salesman problem. *Proc. Comput. Sci.* **76** (2015) 534–538. 2015 IEEE International Symposium on Robotics and Intelligent Sensors (IEEE IRIS2015).

[3] M.W. Berry, B. Hendrickson and P. Raghavan, Sparse matrix reordering schemes for browsing hypertext, edited by J. Renegar, M. Shub and S. Smale. In: Vol. 32 of *Lectures in Applied Mathematics. The Mathematics of Numerical Analysis*. American Mathematical Society Press, Park City, Utah, USA (1996) 99–123.

[4] M.E. Bolanos, S. Aviyente and H. Radha, Graph entropy rate minimization and the compressibility of undirected binary graphs. In: *Proceedings of IEEE Statistical Signal Processing Workshop (SSP)*. IEEE, Ann Arbor, MI (2012) 109–112.

[5] G.O. Chagas and S.L. Gonzaga de Oliveira, Metaheuristic-based heuristics for symmetric-matrix bandwidth reduction: a systematic review. *Proc. Comput. Sci. (ICCS 2015 Int. Conf. Comput. Sci.)* **51** (2015) 211–220.

[6] H.-K. Chen, Evaluation of triangular mesh layout techniques using large mesh simplification. *Multimedia Tools App.* **76** (2017) 25391–25419.

[7] A. Concas, C. Fenu and G. Rodriguez, PQser: a Matlab package for spectral seriation. *Numer. Algorithms* **80** (2019) 879–902.

[8] T.A. Davis and Y. Hu, The University of Florida sparse matrix collection. *ACM Trans. Math. Softw.* **38** (2011) 1–25.

[9] J. Derrac, S. García, D. Molina and F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **1** (2011) 3–18.

[10] M. Dorigo and T. Stützle, Ant Colony Optimization. The MIT Press, Cambridge, MA (2004).

[11] M. Dorigo, M. Birattari and T. Stützle, Ant colony optimization: artificial ants as a computational intelligence technique. *IEEE Comput. Intell. Mag.* **1** (2006) 28–39.

[12] J.H. Drake, A. Kheiri, E. Özcan and E.K. Burke, Recent advances in selection hyper-heuristics. *Eur. J. Oper. Res.* **2852** (2020) 405–428.

[13] Y. Gan, Y. He, L. Gao and W. He, Propagation path optimization of product attribute design changes based on petri net fusion ant colony algorithm. *Expert Syst. App.* **173** (2021) 114664.

[14] A. George and J.W. Liu, Computer Solution of Large Sparse Positive Definite Systems. Prentice-Hall, Englewood Cliffs (1981).

[15] S.L. Gonzaga de Oliveira and G.O. Chagas, A systematic review of heuristics for symmetric-matrix bandwidth reduction: methods not based on metaheuristics. In: Proceedings of the Brazilian Symposium on Operations Research (SBPO 2015). Sobrapo, Pernambuco, Brazil (August 2015).

[16] S.L. Gonzaga de Oliveira and L.M. Silva, Evolving reordering algorithms using an ant colony hyperheuristic approach for accelerating the convergence of the ICCG method. *Eng. Comput.* **36** (2019) 1857–1873.

[17] S.L. Gonzaga de Oliveira and L.M. Silva, An ant colony hyperheuristic approach for matrix bandwidth reduction. *Appl. Soft Comput.* **94** (2020) 106434.

[18] S.L. Gonzaga De Oliveira, A.A.A.M. Abreu, D.T. Robaina and M. Kischnhevsky, Finding a starting vertex for the reverse Cuthill–Mckee method for bandwidth reduction: a comparative analysis using asymmetric matrices, edited by O. Gervasi, et al. In: Vol. 10960 of *The 18th International Conference on Computational Science and Its Applications (ICCSA), Lecture Notes in Computer Science*. Springer International Publishing, Cham (2018) 123–137.

[19] S.L. Gonzaga de Oliveira, J.A.B. Bernardes and G.O. Chagas, An evaluation of low-cost heuristics for matrix bandwidth and profile reductions. *Comput. Appl. Math.* **37** (2018) 1412–1471.

[20] S.L. Gonzaga de Oliveira, J.A.B. Bernardes and G.O. Chagas, An evaluation of reordering algorithms to reduce the computational cost of the incomplete Cholesky-conjugate gradient method. *Comput. Appl. Math.* **37** (2018) 2965–3004.

[21] D.J. Higham, Unravelling small world networks. *J. Comput. Appl. Math.* **158** (2003) 61–74.

[22] B. Koohestani and R. Poli, A hyper-heuristic approach to evolving algorithms for bandwidth reduction based on genetic programming. In: Research and Development in Intelligent Systems XXVIII. Springer London, London, UK (2011) 93–106.

[23] A. Lim, B. Rodrigues and F. Xiao, A fast algorithm for bandwidth minimization. *Int. J. Artif. Intell. Tools* **16** (2007) 537–544.

[24] W. Ma, X. Zhou, H. Zhu, L. Li and L. Jiao, A two-stage hybrid ant colony optimization for high-dimensional feature selection. *Pattern Recognit.* **116** (2021) 107933.

[25] C. Mueller, B. Martin and A. Lumsdaine, A comparison of vertex ordering algorithms for large graph visualization. In: Proceedings of the 6th International Asia-Pacific Symposium on Visualization (APVIS'07). Sydney, Australia (2007) 141–148.

[26] C.H. Papadimitriou, The NP-completeness of bandwidth minimization problem. *Comput. J.* **16** (1976) 177–192.

[27] L.M. Silva and S.L. Gonzaga de Oliveira, An experimental analysis of a GP hyperheuristic approach for evolving low-cost heuristics for profile reductions. In: Anais do SEMISH – Seminário Integrado de Software e Hardware. Cuiabá, MT (2020).

[28] Y. Tian, D. Li, P. Zhou, R. Guo and Z. Liu, An aco-based hyperheuristic with dynamic decision blocks for intercell scheduling. *J. Intell. Manuf.* **29** (2018) 1905–1921.

[29] J. Torres-Jimenez, I. Izquierdo-Marquez, A. Garcia-Robledo, A. Gonzalez-Gomez, J. Bernal and R.N. Kacker, A dual representation simulated annealing algorithm for the bandwidth minimization problem on graphs. *Inf. Sci.* **303** (2015) 33–49.

[30] Z. Yang, L. Fang, X. Zhang and H. Zuo, Controlling a scattered field output of light passing through turbid medium using an improved ant colony optimization algorithm. *Optics Lasers Eng.* **144** (2021) 106646.

[31] Y. Zhang, W. Shao and S.-J. Lai, Some new strategies for RCM ordering in solving electromagnetic stattering problems. *Comput. Phys. Commun.* **184** (2013) 1161–1164.

[32] L. Zhu, J. Lin, Y.-Y. Li and Z.-J. Wang, A decomposition-based multi-objective genetic programming hyper-heuristic approach for the multi-skill resource constrained project scheduling problem. *Knowl.-Based Syst.* **225** (2021) 107099.