Comp. Appl. Math. (2018) 37:1412–1471
https://doi.org/10.1007/s40314-016-0394-9

An evaluation of low-cost heuristics for matrix bandwidth and profile reductions

Sanderson L. Gonzaga de Oliveira[1] · **Júnior A. B. Bernardes**[1] · **Guilherme O. Chagas**[1]

Received: 31 March 2016 / Revised: 25 July 2016 / Accepted: 31 July 2016 /
Published online: 7 December 2016

Abstract Hundreds of heuristics have been proposed to resolve the problems of bandwidth and profile reductions since the 1960s. We found 132 heuristics that have been applied to these problems in reviews of the literature. Among them, 14 were selected for which no other simulation or comparison revealed that the heuristic could be superseded by any other algorithm in the analyzed articles with respect to bandwidth or profile reduction. We also considered the computational costs of the heuristics during this process. Therefore, these 14 heuristics were selected as potentially being the best low-cost methods to solve the bandwidth and/or profile reduction problems. Results of the 14 selected heuristics are evaluated in this work. For evaluation on the set of test problems, a metric based on the relative percentage distance to the best possible bandwidth or profile is proposed. The most promising heuristics for several application areas are identified. Moreover, it was found that the FNCHC and GPS heuristics showed the best overall results in reducing the bandwidth of symmetric and asymmetric matrices among the evaluated heuristics, respectively. In addition, the NSloan and MPG heuristics showed the best overall results in reducing the profile of symmetric and asymmetric matrices among the heuristics among the evaluated heuristics, respectively.

Keywords Bandwidth reduction · Profile reduction · Combinatorial optimization · Envelope reduction problem · Heuristics · Metaheuristics · Reordering algorithms · Sparse matrices · Reordering algorithms · Renumbering · Ordering · Graph labeling · Bandwidth minimization

Mathematics Subject Classification 05C78 · 05C85 · 68R05 · 90C27

Communicated by Jose Alberto Cuminato.

✉ Sanderson L. Gonzaga de Oliveira
sanderson@dcc.ufla.br

Júnior A. B. Bernardes
jrassis@posgrad.ufla.br

Guilherme O. Chagas
guilherme.chagas@computacao.ufla.br

[1] Universidade Federal de Lavras, Lavras, Brazil

1 Introduction

The resolution of large sparse linear systems $Ax = b$, in which A is a sparse matrix, is fundamental in several science and engineering applications and is generally the part of the simulation that requires the highest computational cost. The principal origin of the problems with large-scale matrices arises from the discretization of elliptic or parabolic partial differential equations (PDEs) (Benzi 2002). The methods of finite elements, finite differences, and finite volumes are some of the most common numerical problem-solving methods related to physical phenomena that are modeled by PDEs. Large sparse linear systems are generated when these methods are applied. In addition, large sparse linear systems are also originated from problems that are not modeled by PDEs, such as chemical engineering processes, design and analysis of integrated circuits, and power system networks (Benzi 2002). A considerable amount of memory and a high processing cost are necessary to store and to solve these large-scale linear systems.

Modern hierarchical memory architecture and paging policies favor programs that consider locality of reference into account. Thus, cache coherence (that is, a sequence of recent memory references is clustered locally rather than randomly in the memory address space) should be considered important when designing a new algorithm. For the low-cost solution of large and sparse linear systems, and to reduce the memory space required, an adequate nodal renumbering is desirable to ensure that the corresponding coefficient matrix A will have narrow bandwidth and small profile. Thus, a way of designing an algorithm to return a sequence of graph vertices with cache coherence is through the use of heuristics for bandwidth reduction. Therefore, heuristics for bandwidth and profile reduction are used to achieve low computational and storage costs for solving large sparse linear systems (Gonzaga de Oliveira and Chagas 2015). In particular, the profile reduction is also important for reducing the storage cost of applications that use the skyline data structure (Felippa 1975) to represent large-scale matrices.

Let $A = [a_{ij}]$ be an $n \times n$ symmetric matrix (corresponding to an undirected graph $G = (V, E)$, composed of a set of vertices V and a set of edges E). The bandwidth of line i is $\beta_i(A) = i - \min(j : (1 \leq j < i)\ a_{ij} \neq 0)$. Bandwidth $\beta(A)$ is the largest distance between the non-null coefficient of the lower triangular matrix and the main diagonal considering all lines of the matrix, that is, $\beta(A) = \max((1 \leq i \leq n)\ \beta_i(A)) = \max((1 \leq i \leq n)\ (1 \leq j < i)\ (i - \min(j : (1 \leq j < i)) \mid a_{ij} \neq 0))$ (or the bandwidth of G for a labeling $S = \{s(v_1), s(v_2), \ldots, s(v_{|V|})\}$ (i.e., a bijective mapping from V to the set $\{1, 2, \ldots, |V|\}$) is $\beta(G) = \max(|s(v_i) - s(v_j)| : (v_i, v_j) \in E)$). The profile of A can be defined as $\text{profile}(A) = \sum_{i=1}^{n} \beta_i(A)$.

On the other hand, let $A_u = [a_{ij}]$ be an $n \times n$ asymmetric matrix. The bandwidth of line i is $\beta_i(A) = \max(\beta_{l_i}(A_u), \beta_{u_i}(A_u))$, where $\beta_{l_i}(A_u) = i - \min(j : (1 \leq j < i)\ a_{ij} \neq 0)$ and $\beta_{u_i}(A_u) = \max(j : (i < j \leq n)\ a_{ij} \neq 0) - i$. Bandwidth $\beta(A_u)$ is the largest distance between the non-null coefficient of the lower or the upper triangular matrix and the main diagonal, considering all lines of the matrix, that is, $\beta(A_u) = \max((1 \leq i \leq n)\ \beta_{l_i}(A_u), \beta_{u_i}(A_u))$. The profile of A_u can be defined as $\text{profile}(A_u) = \sum_{i=1}^{n}(\beta_{l_i}(A_u) + \beta_{u_i}(A_u))$.

The problems of bandwidth and profile minimizations are hard (Papadimitriou 1976; Lin and Yuan 1994). Thus, several heuristics have been proposed to solve the bandwidth and/or profile reduction problems since the mid-1960s. The large number of methods available means that the user has an arduous task to determine which method to employ. Various comparisons among methods are available in the literature, but only for a few of them. Additionally, there have been few reviews published on this field. Cuthill (1972) performed a comparative study among results of the heuristics known up to 1971. Gibbs et al. (1976) made comparisons between the results of six heuristics. Benzi (2002) reviewed preconditioning techniques for iterative solution of large sparse linear systems. This review focused on techniques to improve performance and reliability of Krylov subspace methods. However, this review was not strictly of heuristics for bandwidth or profile reduction (Gonzaga de Oliveira and Chagas 2015). Thus, a comparison of a large variety of heuristics proposed for bandwidth and profile reductions is needed in the literature.

Disregarding computational costs, the Variable Neighborhood Search for bandwidth reduction (VNS-band) heuristic (Mladenovic et al. 2010) may be the method that represents the state of the art with respect to the problem of bandwidth reduction. Mladenovic et al. (2010) established the VNS-band timeout at 500 s to solve the problem in 113 instances of the Harwell-Boeing sparse-matrix collection (http://math.nist.gov/MatrixMarket/data/Harwell-Boeing) (Duff et al. 1989a). Even with the short amount of time it takes to find the solution, 500 s as the VNS-band timeout to search for better solutions can be considered high, since this is the time that the user waits for the results. On the other hand, the heuristic based on a dual-representation simulated annealing (DRSA-band) of Torres-Jimenez et al. (2015) obtained, in tests conducted by these authors, results slightly better than results of the VNS-band heuristic (Mladenovic et al. 2010) in relation to bandwidth reduction. In general, the DRSA-band is slower than the VNS-band in the results presented by Torres-Jimenez et al. (2015). Thus, the DRSA-band heuristic was not considered as potentially the best low-cost heuristic with significant bandwidth reduction because its computational cost is higher than the computational cost of the VNS-band heuristic; apart from significantly reducing the bandwidth, a heuristic must also present low computational cost, i.e., it cannot be slow compared to other heuristics. Many papers in this field, where these are only two examples, evaluate their heuristics on 113 instances of Harwell-Boeing dataset, where the number of rows/columns of the matrices included in the dataset varies from 30 to 1104. Although the Harwell-Boeing sparse-matrix collection was widely used for testing heuristics for bandwidth reduction in the literature, the matrices in this dataset are too small by today's standards. The results of such papers, with the largest case only of 1104 dimensions, offer little insight as to how the tested heuristics compare on large examples that are of more practical interest. Hence, additional datasets including much larger matrices are covered in our evaluation.

Thus, one of the main objectives of this study is to verify whether, with a low timeout, the VNS-band heuristic still achieves better results than the possible best low-cost heuristics for bandwidth or profile reduction identified in systematic reviews. Therefore, the main contribution of this work is the comparison of results obtained using 14 heuristics for bandwidth and profile reductions in symmetric and asymmetric matrices (up to 100,196 vertices). These 14 heuristics were selected in systematic reviews as the possible best low-cost methods to solve the problems (Chagas and Gonzaga de Oliveira 2015; Bernardes et al. 2015; Gonzaga de Oliveira and Chagas 2015).

Only potential low-cost heuristics for bandwidth and profile reductions were selected in the systematic reviews. The reason for this decision was that the local reordering of vertices of the graph associated with the matrix of the linear system may contribute to reducing the computational cost of an iterative solver, such as the Conjugate Gradient Method (CGM) (Duff and Meurant 1989b). It should be noted that it is important to have an ordering which does not lead to an increase in the number of iterations when a preconditioner is applied. Additionally, such reduction in the execution cost is also achieved by improving the number of cache hits (Das et al. 1992; Burgess and Giles 1997). On the other hand, the bandwidth and profile reductions are not directly proportional to the computational cost reduction obtained when linear systems are solved using an iterative method. Clearly, what is to be minimized is the total computing time including the reordering time (at least when only a single linear system is to be solved). Thus, a reordering of vertices must be performed at low cost. To provide more specific detail, although linear systems of reduced order can be solved efficiently with a multifrontal direct method, a prominent method for solving large-scale sparse linear systems is the conjugate gradient method (Hestenes and Stiefel 1952; Lanczos 1952). One can reduce computational costs using this method by applying a local ordering of the vertices (Duff and Meurant 1989b) of the corresponding graph of A to improve cache hit rates. This local ordering can be reached by applying a heuristic for bandwidth reduction (Das et al. 1992; Burgess and Giles 1997). On the other hand, an important issue is to have an ordering which leads to a small number of iterations when a preconditioner is applied (and this depends on the structure of the instance). Additionally, Benzi et al. (1999) showed that heuristics for bandwidth reduction can have a positive effect on the computational cost of the generalized minimal residual (GMRES) method (Saad and Schultz 1986).

The remainder of this paper is organized as follows. Section 2 explains the systematic reviews performed to identify the potential best low-cost heuristics for bandwidth and profile reductions. Section 3 presents how the simulations were conducted in this study. Section 4 shows the results. Finally, Sect. 5 addresses the conclusions.

2 Systematic reviews

Systematic reviews (Chagas and Gonzaga de Oliveira 2015; Bernardes et al. 2015; Gonzaga de Oliveira and Chagas 2015) report 73 and 74 heuristics for bandwidth and profile reductions, respectively, that had been published in the period of time spanning the 1960s to the present. Most of the heuristics were considered surpassed by other heuristics in these systematic reviews. Consequently, eight heuristics in each case were selected as potentially being the best low-cost heuristics for bandwidth [RCM–GL (George and Liu 1981), Burgess–Lai (BL) (Burgess and Lai 1986), WBRA (Esposito et al. 1998), FNCHC (Lim et al. 2003, 2004, 2007), GGPS (Wang et al. 2009), VNS-band (Mladenovic et al. 2010), hGPHH (Koohestani and Poli 2011), CSS-band (Kaveh and Sharafi 2012)] or profile [Snay (1976), RCM–GL (George and Liu 1981), RCM–GL–FL (Fenves and Law 1983), Sloan (1989), MPG (Medeiros et al. 1993), NSloan (Kumfert and Pothen 1997), Sloan–MGPS (Reid and Scott 1999), Hu and Scott (2001)] reduction.

From the heuristics identified in the systematic reviews, 17 heuristics were applied to both bandwidth and profile reductions. In addition, the Reverse Cuthill–McKee method with pseudo-peripheral vertex given by the George–Liu algorithm (RCM–GL) (George and Liu 1981) was selected in both systematic reviews of heuristics for bandwidth and profile reductions. Thus, 130 heuristics for bandwidth and profile reductions were identified in both systematic reviews, and 15 heuristics were selected because no other simulation or comparison showed that these 15 heuristics could be superseded by any other heuristics in the articles analyzed, in terms of bandwidth or profile reduction when the computation costs of the given heuristic are also considered.

The Gibbs–Poole–Stockmeyer (GPS) algorithm (Gibbs et al. 1976), outperformed by metaheuristic-based heuristics, is one of the most classic low-cost heuristics tested in the field for both bandwidth and profile reductions. Hence, the GPS algorithm was implemented and its results were compared with the other heuristics implemented in this computational experiment.

Table 1 Low-cost heuristics for bandwidth and profile reductions that were selected from systematic reviews

Bandwidth	Burgess–Lai (BL) (1986), WBRA (Esposito et al. 1998), FNCHC (Lim et al. 2003, 2004, 2007),
	GGPS (Wang et al. 2009), VNS-band (Mladenovic et al. 2010), hGPHH (Koohestani and Poli 2011), CSS-band (Kaveh and Sharafi 2012)
Profile	Snay (1976), Sloan (1989), MPG (Medeiros et al. 1993), NSloan (Kumfert and Pothen 1997), Sloan–MGPS (Reid and Scott 1999)
Bandwidth and profile	GPS (Gibbs et al. 1976), RCM–GL (George and Liu 1981)

On the other hand, Fenves and Law's RCM–GL (RCM–GL–FL) method (Fenves and Law 1983), despite being selected in the systematic review of heuristics for profile reduction, was not implemented in this work because it is a specific application of the RCM–GL method for finite element discretizations. Additionally, although no computational costs were presented by its authors, the Hu and Scott's heuristic (Hu and Scott 2001) was selected for its results in profile reductions when compared with the results shown by the heuristics that were tested by Hu and Scott (2001). However, the Hu-Scott heuristic was not implemented here because it was considered a high-cost heuristic, especially when performing matrix multiplications.

Another version of the VNS-band was proposed by Wang et al. (2014). This heuristic is outperformed by the fast node centroid hill climbing (FNCHC) heuristic (Lim et al. 2007). This is verified both in approximating the solution as well as comparing the computational cost to present an approximate solution. One can realize this by examining the results of the original VNS-band (Mladenovic et al. 2010) heuristic and the VNS-band proposed by Wang et al. (2014) and comparing them with the results presented below. It should be noted that the dataset used in the tests shown by Wang et al. (2014) is a subset of a dataset presented below.

Thus, Table 1 shows 14 heuristics that can be considered as the most promising low-cost heuristics to solve the problems. These 14 heuristics were implemented and tested in this work.

3 Description of the tests

Appendix A (Appendices A–D are available at http://www.dcc.ufla.br/~sanderson/app_coam16) shows the testing and calibration performed to compare our implementations with the codes used by the original proposers of the 14 heuristics to ensure the codes we implemented were comparable to the algorithms that were originally proposed. To evaluate the bandwidth and profile reductions provided by the 14 selected heuristics, two datasets commonly used in this field were employed. Specifically, 113 (50 symmetric and 63 asymmetric) instances of the Harwell-Boeing sparse-matrix collection (http://math.nist.gov/MatrixMarket/data/Harwell-Boeing) (Duff et al. 1989a) and 22 (17 symmetric and 5 asymmetric) instances of the University of Florida sparse-matrix collection (http://www.cise.ufl.edu/research/sparse/matrices/index.html) (Davis and Hu 2011) were used in this work. These instances represent a wide spectrum of scientific and engineering applications matrices (i.e., standard-test matrices arising from problems in linear systems, least squares, and eigenvalue calculations) and have been employed here for comparisons and evaluations since many other researchers have employed these instances. Specifically, the 113 instances of the Harwell-Boeing sparse-matrix collection were divided into two subsets: (i) 33 instances, ranging from 30 to 237 vertices, accordingly as stated in the collection, with each of these 33 instances (used as counter-examples to hypotheses in sparse-matrix research) having less than 200 vertices when vertices without adjacencies were disregarded; and (ii) 80 instances, ranging from 207 to 1104 vertices. This subdivision of the 113 instances of the Harwell-Boeing sparse-matrix collection into two sets of instances is common in the field, for example, see Martí et al. (2001), Lim et al. (2004, 2007), Piñana et al. (2004), Rodriguez-Tello et al. (2008), Mladenovic et al. (2010), Torres-Jimenez et al. (2015). Additionally, each set of instances was divided into symmetric and asymmetric matrices, as shown in the tables below.

To apply the heuristics implemented in this work in asymmetric matrices, we used the strategy proposed by Reid and Scott (2006) to apply the heuristics in asymmetric instances. In this strategy, the asymmetric matrix A_u is added to its transpose matrix A_u^{T}, i.e., $A_u + A_u^{\mathrm{T}}$, thus resulting in a symmetric matrix. Subsequently, the heuristic is applied to the graph associated with the matrix obtained and the reordering attained using the heuristic is used to reorder the rows of the original asymmetric matrix. This strategy, according to the results presented by Reid and Scott (2006), had the poorest results in the bandwidth reduction of asymmetric matrices using the RCM heuristic (George 1971) when compared with the two other strategies proposed by Reid and Scott (2006). Nevertheless, the strategy to compute $A_u + A_u^T$ was applied in our approach because it is included along with the RCM–GL method (George and Liu 1981) in the MATLAB software (MATLAB 2016). In addition, it is the simplest strategy and the one that presents the lowest storage and computational costs among those proposed by Reid and Scott (2006).

The workstations used in the execution of the simulations with the instances of the Harwell-Boeing and University of Florida sparse-matrix collections contained an Intel® Core™ i3-2120 (3 MB Cache, CPU 3.30 GHz×4, 8 GB of main memory DDR3 1333 MHz) and Intel® Xeon™ E5620 (12 MB Cache, CPU 2.40 GHz×8, 24 GB of main memory DDR3 1333 MHz) (Intel; Santa Clara, CA, United States), respectively. The Ubuntu 14.04 LTS 64-bit operating system with Linux kernel-version 3.13.0-39-generic was used. It should be noted that we followed the recommendations given by Johnson (2002) for this experimental analysis of the 14 heuristics for bandwidth and profile reductions.

A metric used by some authors, where Kumfert and Pothen (1997), Mladenovic et al. (2010), Koohestani and Poli (2011) are examples, is to add, for each heuristic, the bandwidths or profiles obtained. A similar metric used by other authors, where Martí et al. (2001), Piñana et al. (2004), Lim et al. (2007), Rodriguez-Tello et al. (2008) are examples, is to calculate the average bandwidth over the instances in each set. However, these metrics are not reasonable when considering a set of instances with very different sizes, as is the case in the tests presented in this paper. This is because, for example, a heuristic with very bad results in a small instance would not be penalized appropriately.

In many works, including Burgess and Lai (1986), Medeiros et al. (1993), Esposito et al. (1998), Mladenovic et al. (2010), Koohestani and Poli (2011), the authors compared the results of the heuristics by counting the number of times that the heuristic obtained the lower bandwidth or profile on the instances of the Harwell-Boeing sparse-matrix collection. This kind of metric is also shown in the following tables. In addition, the heuristic may be appropriately considered the best heuristic in a set when it has the best bandwidth or profile reduction among all the heuristics in the set. One possible scenario, then, is that a heuristic could be considered the best of the entire set if it attains the second best status in many instances, while different heuristics are the best heuristic for different individual instances. Thus, to evaluate the quality of the results obtained using the heuristics tested, for each heuristic h in each set of instances, $\rho_p = \frac{\text{profile}_h - \text{profile}_{\min}}{\text{profile}_{\min}}$ was calculated for each instance, where profile_h is the profile obtained using a heuristic h and $\text{profile}_{\min}$ is the lowest profile obtained in an instance by a heuristic tested. The same is carried out in relation to both bandwidth reduction (ρ_β) and execution times (ρ_t). To the best of our knowledge, this paper is the first (published) instance of this approach being used in the field.

4 Results and analysis

This section presents the results obtained by the heuristics in relation to bandwidth and profile reductions in 113 (Sect. 4.1) and 22 (Sect. 4.2) instances of the Harwell-Boeing and University of Florida sparse-matrix collections, respectively. In the tables shown in this section, it can be seen that the instance name and size (n), the value of the initial bandwidth (β_0) or profile (profile_0) of the instance, and the average values of bandwidth, profile and runtime obtained using each heuristic in 10 executions carried out in each of the instances. Section 4.3 analyzes the results obtained.

4.1 Results of 14 heuristics applied to instances of the Harwell-Boeing sparse-matrix collection

This section presents the bandwidth and profile results obtained using 14 heuristics applied to 113 instances of the Harwell-Boeing sparse-matrix collection. Sections 4.1.1–4.1.4 present the results of the 14 heuristics for bandwidth and profile reductions applied to the sets composed of 15 and 35 symmetric matrices and 18 and 45 asymmetric matrices of the Harwell-Boeing sparse-matrix collection, respectively.

4.1.1 Results of 14 heuristics applied to the set composed of 15 symmetric instances of the Harwell-Boeing sparse-matrix collection

In this set composed of 15 symmetric instances, the FNCHC (Lim et al. 2003, 2004, 2007) and NSloan (Kumfert and Pothen 1997) heuristics were the best ones for reducing bandwidth and profile when considering the ρ_β and ρ_p metrics, respectively [see Fig. 1a, b, and Table 9 (Tables 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 are available in Appendix B)]. These two heuristics also obtained the largest number of best results of bandwidth (in 12 instances) and profile (in 8 instances) in this dataset (see Fig. 1c, d).

The VNS-band heuristic was tested with a timeout of 500 s because this was the time set by its authors (Mladenovic et al. 2010). It should be noted that 500 s on the machine used in this study achieves more computation than the machine on which the tests were conducted by Mladenovic et al. (2010). In spite of this, the VNS-band heuristic was dominated by the FNCHC and NSloan heuristics in relation to bandwidth and profile reductions, respectively, for this set of instances.

The hGPHH-GL and RCM–GL methods were the fastest ones among the heuristics tested according to the ρ_t metric, and considering also that the RCM–GL method obtained the lowest computational costs for a larger number of instances (six instances) (see Fig. 2; Table 10). In particular, the WBRA (Esposito et al. 1998), FNCHC (Lim et al. 2003, 2004, 2007), VNS-band (Mladenovic et al. 2010), and CSS-band (Kaveh and Sharafi 2012) heuristics showed higher execution times than the 10 other heuristics tested. Moreover, the hGPHH-

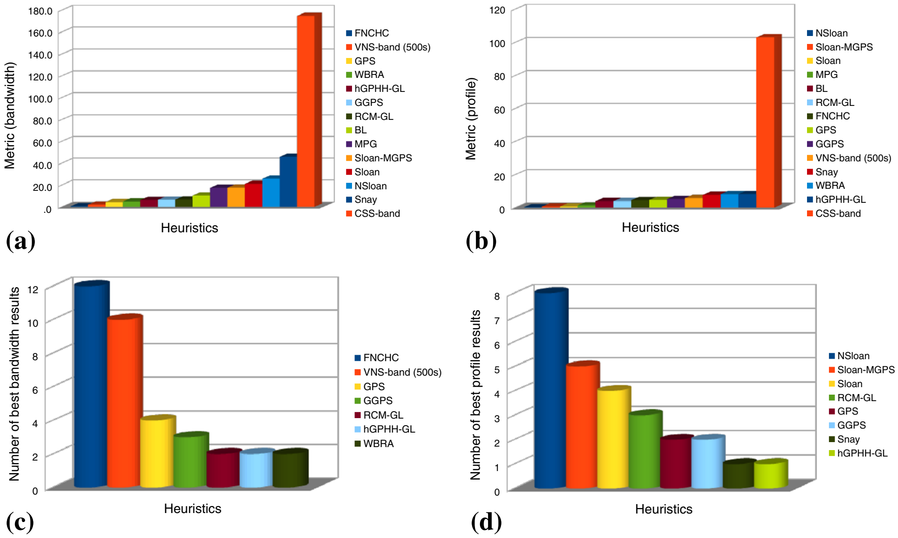

Fig. 1 Results (**a** $\sum \rho_b$, **b** $\sum \rho_p$) of 14 heuristics applied to reduce **a** bandwidth and **b** profile, and number of best results of several heuristics applied to reduce **c** bandwidth and **d** profile of 15 symmetric instances of the Harwell-Boeing sparse-matrix collection

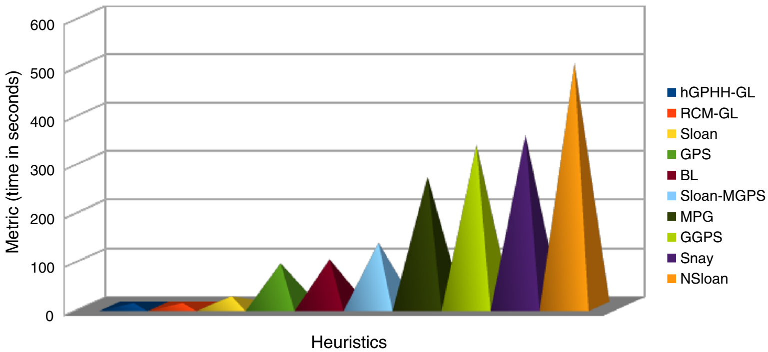

Fig. 2 Results ($\sum \rho_t$) of 10 heuristics applied to reduce bandwidth of 15 symmetric instances of the Harwell-Boeing sparse-matrix collection

GL method is as fast as the RCM–GL method because their difference is how to order the adjacent vertices (to be numbered) to the current vertex.

4.1.2 Results of 14 heuristics applied to the set composed of 35 symmetric instances of the Harwell-Boeing sparse-matrix collection

In this set composed of 35 symmetric instances, the VNS-band (8 s) (Mladenovic et al. 2010) and MPG (Medeiros et al. 1993) heuristics were the best ones for reducing bandwidth and profile when considering the ρ_b and ρ_p metrics, respectively (see Fig. 3a, b; Tables 11 and 12). These two heuristics also obtained the largest number of best results of bandwidth (in 24 instances) and profile (in 11 instances) in this set of instances, respectively (see Fig. 3c, d).

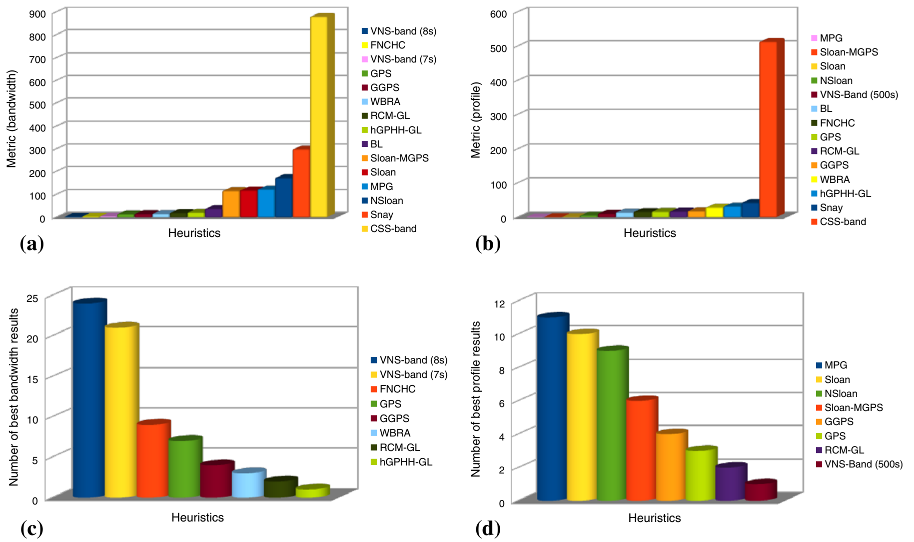

Fig. 3 Results (**a** $\sum \rho_b$, **b** $\sum \rho_p$) of 14 heuristics applied to reduce **a** bandwidth and **b** profile, and number of best results of several heuristics applied to reduce **c** bandwidth and **d** profile of 35 symmetric instances of the Harwell-Boeing sparse-matrix collection

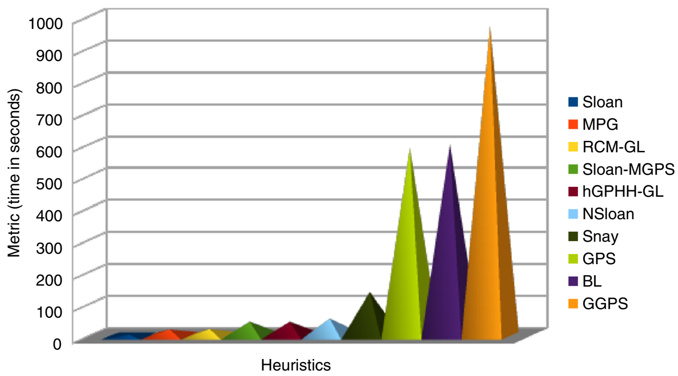

Fig. 4 Results ($\sum \rho_t$) of 10 heuristics applied to reduce bandwidth of 35 symmetric instances of the Harwell-Boeing sparse-matrix collection

In addition, Sloan's (1989) algorithm was the fastest method among the heuristics tested (see Fig. 4; Table 13).

4.1.3 Results of 14 heuristics applied to the set composed of 18 asymmetric instances of the Harwell-Boeing sparse-matrix collection

In this set composed of 18 asymmetric instances, the FNCHC and MPG heuristics were the best ones for reducing bandwidth and profile when considering the ρ_b and ρ_p metrics, respectively (see Fig. 5a, b; Table 14). The RCM–GL method was the fastest method among the heuristics tested (see Fig. 6; Table 15).

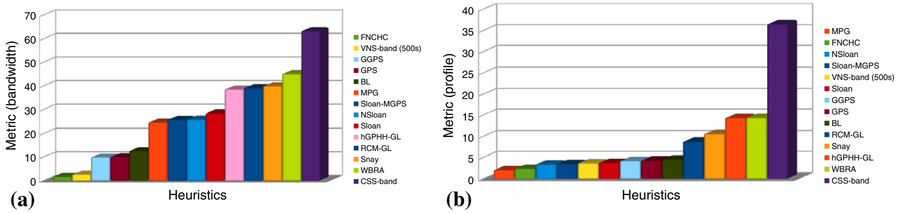

Fig. 5 Results (**a** $\sum \rho_b$, **b** $\sum \rho_p$) of 14 heuristics applied to reduce **a** bandwidth and **b** profile of 18 asymmetric instances of the Harwell-Boeing sparse-matrix collection

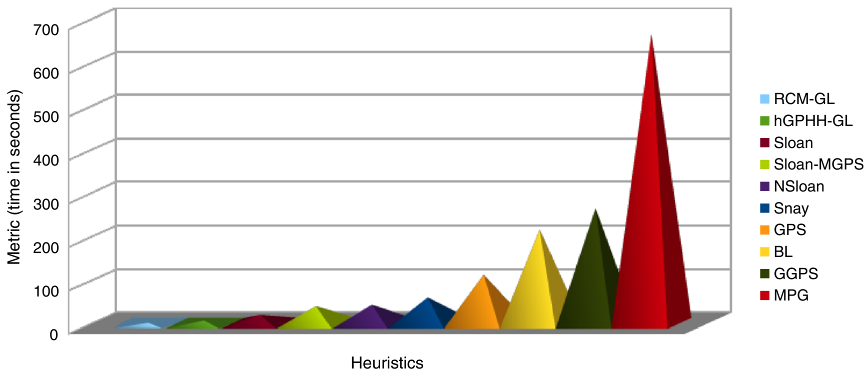

Fig. 6 Results ($\sum \rho_t$) of 10 heuristics applied to reduce bandwidth of 18 asymmetric instances of the Harwell-Boeing sparse-matrix collection

4.1.4 Results of 14 heuristics applied to the set composed of 45 asymmetric instances of the Harwell-Boeing sparse-matrix collection

In this set composed of 45 asymmetric instances, the VNS-band (12 s) (Mladenovic et al. 2010) and MPG (Medeiros et al. 1993) heuristics were the best ones for reducing bandwidth and profile when considering the ρ_b and ρ_p metrics, respectively (see Fig. 7a, b; Tables 16 and 17). These two heuristics also obtained the largest number of best results of bandwidth (in 34 instances) and profile (in 17 instances) in this set of instances, respectively (see Fig. 7c, d). The hGPHH-GL heuristic was the fastest method among the heuristics tested (see Fig. 8; Table 18).

4.2 Results of 10 heuristics applied to instances of the University of Florida sparse-matrix collection

Results of Snay's (1976), Burgess and Lai (1986), WBRA (Esposito et al. 1998), and CSS-band (Kaveh and Sharafi 2012) heuristics were dominated by the 10 other heuristics when applied to the 113 instances of the Harwell-Boeing sparse-matrix collection (see Sect. 4.1) so that these four heuristics were not applied to the 22 instances of the University of Florida sparse-matrix collection. Hence, Sects. 4.2.1 and 4.2.2 present the results of 10 heuristics for bandwidth and profile reductions applied to the sets composed of 17 symmetric and 5 asymmetric matrices of the University of Florida sparse-matrix collection, respectively.

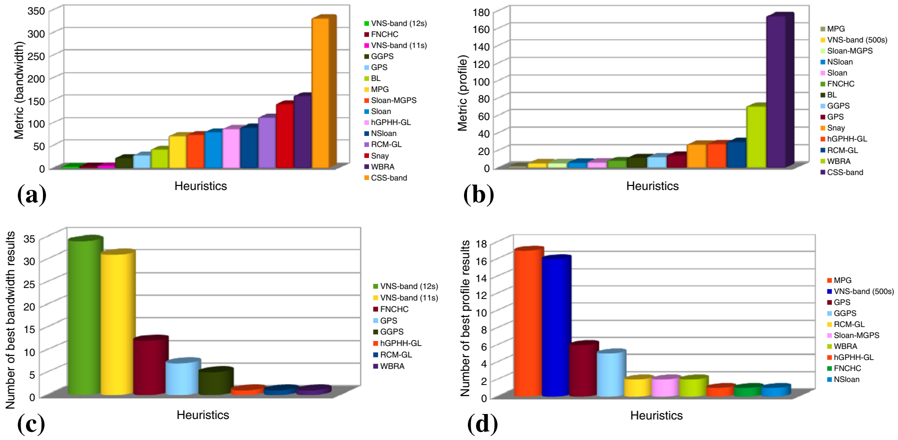

Fig. 7 Results (**a** $\sum \rho_b$, **b** $\sum \rho_p$) of 14 heuristics applied to reduce **a** bandwidth and **b** profile, and number of best results of several heuristics applied to reduce **c** bandwidth and **d** profile of 45 asymmetric instances of the Harwell-Boeing sparse-matrix collection

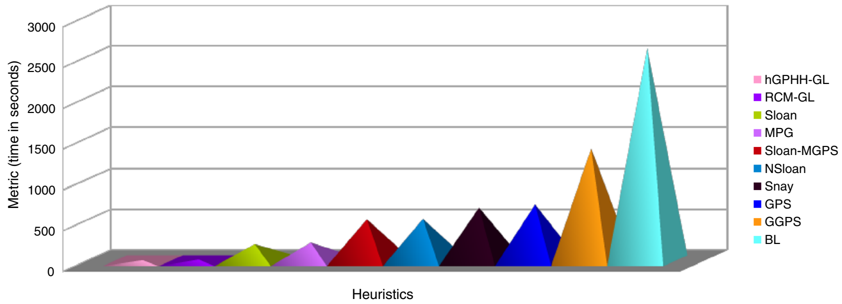

Fig. 8 Results ($\sum \rho_t$) of 10 heuristics applied to reduce bandwidth of 45 asymmetric instances of the Harwell-Boeing sparse-matrix collection

4.2.1 Results of 10 heuristics applied to the set composed of 17 symmetric instances of the University of Florida sparse-matrix collection

In this set composed of 17 symmetric instances, the FNCHC and NSloan heuristics were the best ones for reducing bandwidth and profile when considering the ρ_b and ρ_p metrics, respectively (see Fig. 9a, b; Table 19). These two heuristics also obtained the largest number of best results of bandwidth (in eight instances) and profile (in ten instances) in this set of instances, respectively (see Fig. 9c, d). The RCM–GL method was the fastest method among the heuristics tested (see Fig. 10; Table 20).

We established the VNS-band timeout at 500 s to solve the problem in these 17 symmetric instances of the University of Florida sparse-matrix collection. Probably, the VNS-band heuristic would achieve better results with a higher timeout, but our intention is to investigate low timeouts for the heuristics. In the tests shown in Table 20, one can note that the VNS-band heuristic achieved its results with a higher computing cost of 1 and 5 magnitudes in relation to the FNCHC and RCM–GL heuristics, respectively.

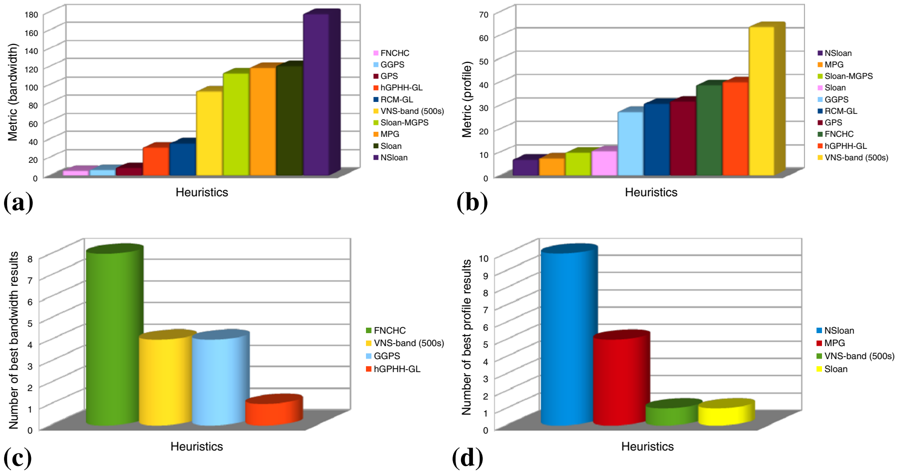

Fig. 9 Results (**a** $\sum \rho_b$, **b** $\sum \rho_p$) of 10 heuristics applied to reduce **a** bandwidth and **b** profile, and number of best results of several heuristics applied to reduce **c** bandwidth and **d** profile of 17 symmetric instances of the University of Florida sparse-matrix collection

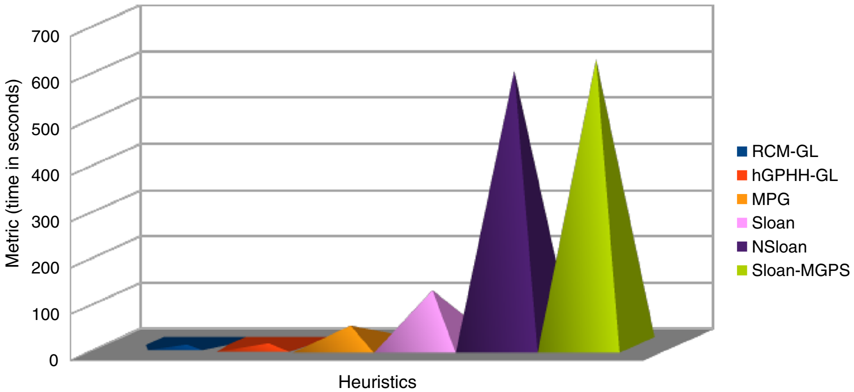

Fig. 10 Results ($\sum \rho_t$) of 10 heuristics applied to reduce bandwidth of 17 symmetric instances of the University of Florida sparse-matrix collection

4.2.2 Results of 10 heuristics applied to the set composed of five asymmetric instances of the University of Florida sparse-matrix collection

In this set composed of five asymmetric instances, one can verify that the GPS and MPG heuristics were the best ones for reducing bandwidth and profile when considering the ρ_b and ρ_p metrics, respectively (see Fig. 11a, b; Table 21). These two heuristics also obtained the largest number of best results of bandwidth (in two instances) and profile (in three instances) in this set of instances, respectively. The RCM–GL method was the fastest method among the heuristics tested (see Fig. 12; Table 21).

Similar to the executions performed with the 17 symmetric instances of the University of Florida sparse-matrix collection, we established the VNS-band timeout at 500 s to solve the problem in these five asymmetric instances of this dataset. Likewise, it is probable that the

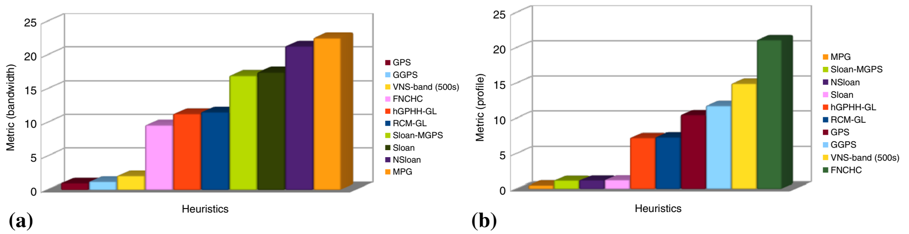

Fig. 11 Results (**a** $\sum \rho_b$, **b** $\sum \rho_p$) of 10 heuristics applied to reduce **a** bandwidth and **b** profile of five asymmetric instances of the University of Florida sparse-matrix collection

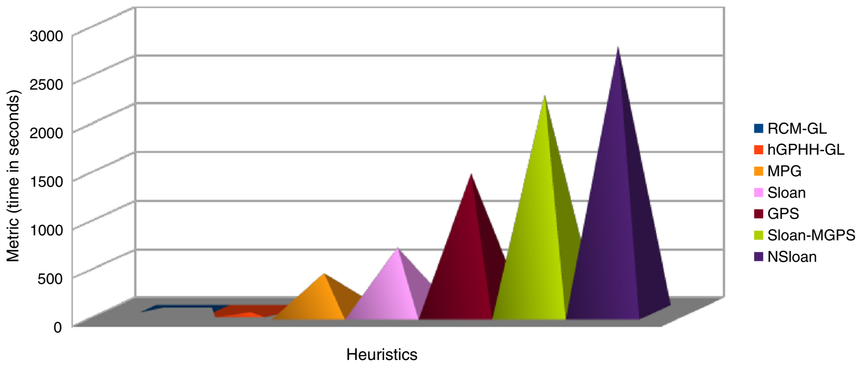

Fig. 12 Results ($\sum \rho_t$) of 6 heuristics applied to reduce bandwidth of five asymmetric instances of the University of Florida sparse-matrix collection

VNS-band heuristic would achieve better results with a higher timeout, but our intention is to investigate low timeouts for the heuristics. In the tests performed in these five asymmetric matrices, the VNS-band heuristic achieved its results with a higher computing cost of 1 and 5 magnitudes in relation to the GPS and RCM–GL algorithms, respectively (see Table 21).

4.3 Best heuristics applied to four sets of instances of the Harwell-Boeing and two sets of instances of the University of Florida sparse-matrix collections

The sets composed of 15 symmetric and 18 asymmetric instances contain much smaller matrices than the sets composed of 35 symmetric and 45 asymmetric instances, respectively. Moreover, the instances of the University of Florida sparse-matrix collection are larger than the instances of the Harwell-Boeing sparse-matrix collection. It may not be appropriate to consider a small instance with the same score of a much larger instance. Thus, the sets of instances of the University of Florida sparse-matrix collection were mainly considered to identify the best low-cost heuristics for bandwidth and profile reductions. On the other hand, the other sets composed of instances of the Harwell-Boeing sparse-matrix collection are also considered, as described below.

We divided the sets of test problems by application area and then the most promising heuristics for each specific application were identified. Tables 2 (symmetric matrices) and 3 (asymmetric matrices) show the most promising heuristics for application areas applied to the four sets of instances of the Harwell-Boeing (http://math.nist.gov/MatrixMarket/data/Harwell-Boeing (Duff et al. 1989a)) and the two sets of instances of the University of Florida

Table 2 The most promising heuristics (for ten application areas) for reducing bandwidth and profile of symmetric matrices

Heuristic	Application area	Reduction
FNCHC	Computational fluid dynamics problem	Bandwidth
	Dynamic analyses in structural engineering	
	Finite-element structures	
	problems in aircraft design	
	Oceanic modeling	
	Optimization problem	
	Power system networks	
	Partial differential equations	
	Structural problem	
RCM–GL	Finite-element model problem	
	Oil reservoir modeling	
MPG	Dynamic analyses in structural engineering	Profile
	Finite-element structures	
	problems in aircraft design	
NSloan	Computational fluid dynamics problem	
	Oil reservoir modeling	
	Optimization problem	
	Structural problem	
Sloan	Finite-element model problem	
	Partial differential equations	
	Power system networks	
	Oceanic modeling	

sparse-matrix (http://www.cise.ufl.edu/research/sparse/matrices/index.html (Davis and Hu 2011)) collections in relation to bandwidth and profile reductions (see Tables 22 and 23 in Appendix C). Appendix D analyzes the heuristics that showed the best bandwidth and profile reductions of symmetric and asymmetric matrices. Table 4 shows the heuristics that showed the best overall performance on these six sets of test matrices.

Figure 13 [WBRA (Esposito et al. 1998), FNCHC (Lim et al. 2003, 2004, 2007), VNS-band (Mladenovic et al. 2010), and CSS-band (Kaveh and Sharafi 2012) heuristics], Figure 14 [GPS (Gibbs et al. 1976), Snay's (1976), Burgess and Lai (1986), and GGPS (Wang et al. 2009) heuristics], and Figure 15 [GPS (Gibbs et al. 1976), FNCHC (Lim et al. 2003, 2004, 2007), GGPS (Wang et al. 2009), and VNS-band (Mladenovic et al. 2010)], built from a wide variety of references that were part of this work, show graphs of execution times, of the highest time-consuming heuristics tested, applied to 113 and 22 instances of the Harwell-Boeing and University of Florida sparse-matrix collections, respectively.

As expected, the FNCHC (Lim et al. 2003, 2004, 2007), VNS-band (Mladenovic et al. 2010), and CSS-band (Kaveh and Sharafi 2012) heuristics showed higher computational costs than most of the methods tested here because these are metaheuristic-based heuristics.

The WBRA heuristic (Esposito et al. 1998) shows high computational cost because it builds a rooted level structure (RLS) for each vertex of the graph and these structures are renumbered based on a bottleneck linear assignment. This heuristic for bandwidth reduction

Table 3 The most promising heuristics (for 14 application areas) for reducing bandwidth and profile of asymmetric matrices

Heuristic	Application area	Best reduction
FNCHC	Chemical engineering	Bandwidth
	Chemical kinetics	
	Economic modeling	
	Flow in networks	
	Nuclear reactor modeling	
	Power network problem	
GPS	Circuit physics	
	Circuit simulation problem	
	Directed graph	
	Fluid flow modeling	
	Materials problem	
	Oil reservoir simulation	
	Simulation studies in computer systems	
hGPHH-GL	Astrophysics	
GPS	Materials problem	Profile
GGPS	Simulation studies in computer systems	
MPG	Astrophysics	
	Chemical engineering	
	Circuit simulation problem	
	Directed graph	
	Flow in networks	
	Fluid flow modeling	
	Nuclear reactor modeling	
	Oil reservoir simulation	
hGPHH-GL	Economic modeling	
NSloan	Circuit physics	
Sloan–MGPS	Chemical kinetics	
	Power network problem	

Table 4 Heuristics that showed the best overall performance on the six sets of test matrices

Type	β	Profile
Symmetric	FNCHC	GPS
Asymmetric	NSloan	MPG

of symmetric matrices did not show competitive results when compared to other heuristics, such as the VNS-band, FNCHC, and GPS heuristics. The WBRA heuristic employs a function to move vertices from the level with maximum width to the subsequent level (nearer the level 0) in the level structure (that may not be rooted at a vertex). This may result in a worse distribution of the vertices in relation to the RLS created by the GPS algorithm, for example.

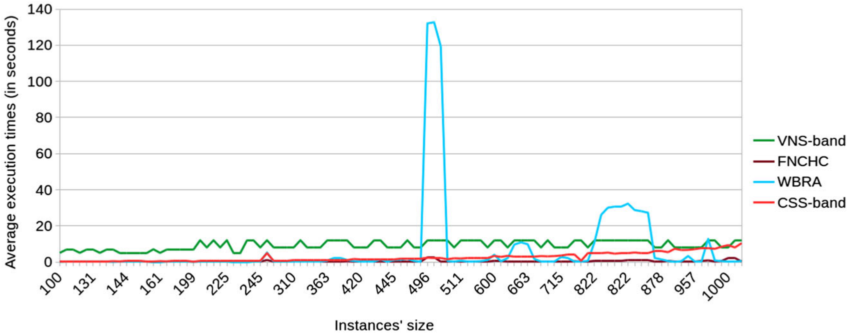

Fig. 13 Average execution times of the WBRA (Esposito et al. 1998), FNCHC (Lim et al. 2003, 2004, 2007), VNS-band (8 s) (Mladenovic et al. 2010), and CSS-band (Kaveh and Sharafi 2012) heuristics applied to 103 (ranging from 100 to 1104 vertices) instances of the Harwell-Boeing sparse-matrix collection

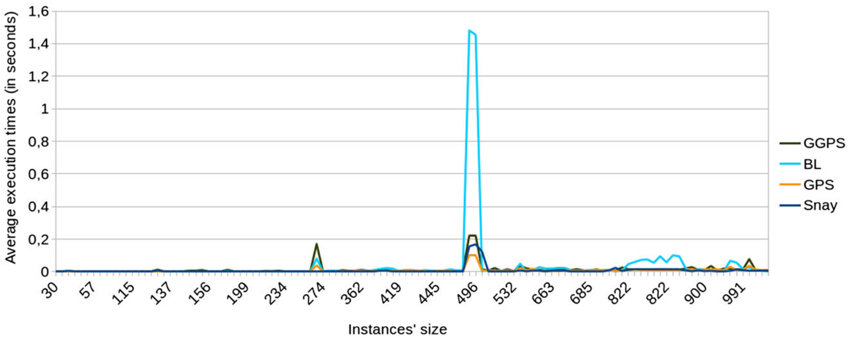

Fig. 14 Average execution times of the GPS (Gibbs et al. 1976), Snay's (1976), Burgess and Lai (1986), and GGPS (Wang et al. 2009) heuristics applied to 113 instances of the Harwell-Boeing sparse-matrix collection

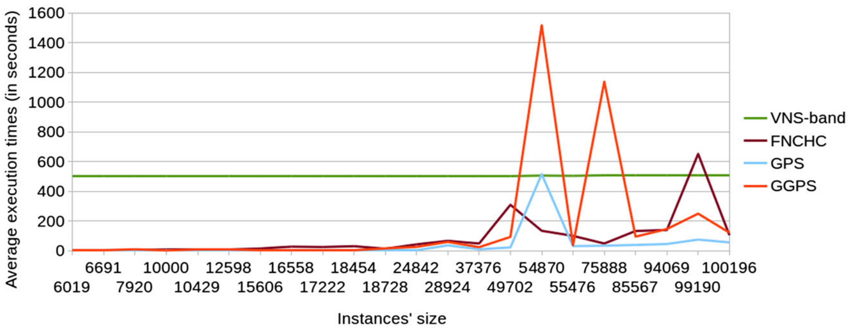

Fig. 15 Average execution times of the GPS (Gibbs et al. 1976), FNCHC (Lim et al. 2003, 2004, 2007), GGPS (Wang et al. 2009), and VNS-band (Mladenovic et al. 2010) heuristics applied to 17 symmetric and 5 asymmetric instances of the University of Florida sparse-matrix collection

The results of the CSS-band heuristic may be related to the random initial solution suggested by its authors (we strictly followed the algorithms proposed when implementing these heuristics). On the other hand, the VNS-band and FNCHC heuristics employ a variation of the breadth-first search procedure to provide an initial solution. Moreover, a slow convergence was observed within the CSS-band heuristic.

Although based on RCM and GPS (Gibbs et al. 1976) algorithms, Burgess and Lai's heuristic (Burgess and Lai 1986) contains a stage where three RLSs are modified and its results showed higher execution times than the GPS algorithm. To be more precise, Burgess and Lai's heuristic (Burgess and Lai 1986) attempts to reduce the width of each level of three RLSs built in the first step. This heuristic moves vertices from one level above or below until a minimum width is reached. Thus, this algorithm may compute a level for a long time until reaching this minimum width.

Snay's heuristic (Snay 1976) is similar to the RCM method, considering that both are based on breadth-first search procedure. The RCM method labels adjacent vertices to the current vertex (sorting them in ascending degree order) and Snay's heuristic (Snay 1976) takes into account vertices also in the second level (sorting them in ascending degree order, considering only adjacencies to vertices that are neither labeled nor are candidate vertices) of the vertices already labeled. Snay's heuristic (Snay 1976) performs this task for 10 pseudo-peripheral vertices, which explains its high computational cost. Even designed for profile reduction (by allowing vertices with small degree numbers being labeled before other vertices), the results of Snay's heuristic were not competitive with the results of the other heuristics for profile reduction, such as Sloan's algorithm (and its variations).

In addition to the GPS (Gibbs et al. 1976), MPG (Medeiros et al. 1993), NSloan (Kumfert and Pothen 1997), and FNCHC (Lim et al. 2003, 2004, 2007) heuristics for bandwidth and/or profile reductions, it should also be noted that the RCM–GL (George and Liu 1981) and hGPHH-GL (Koohestani and Poli 2011) methods demonstrated very low computational costs in the six datasets tested. Figure 16 shows the execution times of the RCM–GL (George and Liu 1981), Sloan's (1989), MPG (Medeiros et al. 1993), NSloan (Kumfert and Pothen 1997), Sloan–MGPS (Reid and Scott 1999), and hGPHH-GL (Koohestani and Poli 2011) heuristics, which are the lowest-cost heuristics tested, applied to 113 asymmetric instances of the Harwell-Boeing sparse-matrix collection. In particular, the 14 heuristics showed very high computational costs in the MBEACXC, MBEAFLW, and MBEAUSE instances (from economic modeling). These instances have many connected components and isolated vertices; the heuristics are applied to each connected component of a disconnected graph.

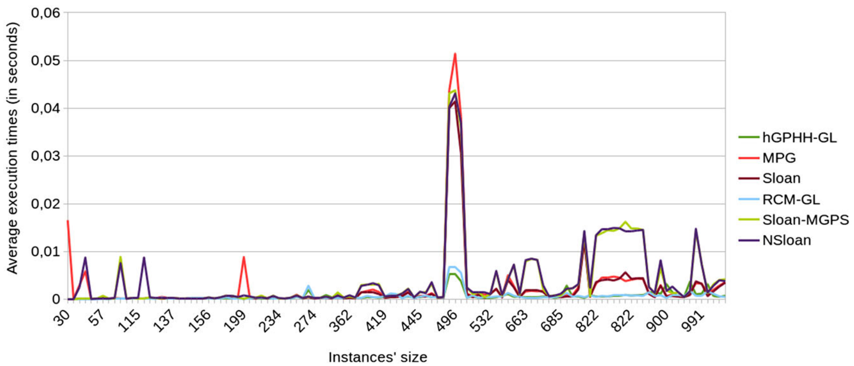

Fig. 16 Average execution times of the RCM–GL (George and Liu 1981), Sloan's (1989), MPG (Medeiros et al. 1993), NSloan (Kumfert and Pothen 1997), Sloan–MGPS (Reid and Scott 1999), and hGPHH-GL (Koohestani and Poli 2011) heuristics applied to 113 asymmetric instances of the Harwell-Boeing sparse-matrix collection

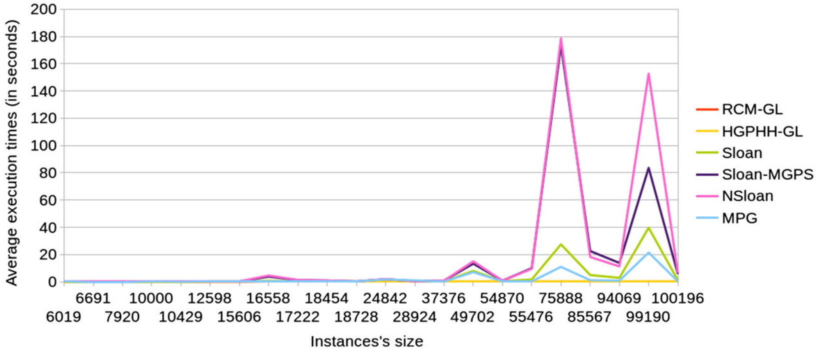

Fig. 17 Average execution times of the RCM–GL (George and Liu 1981), Sloan's (1989), MPG (Medeiros et al. 1993), NSloan (Kumfert and Pothen 1997), Sloan–MGPS (Reid and Scott 1999), and hGPHH-GL (Koohestani and Poli 2011) heuristics applied to 22 instances of the University of Florida sparse-matrix collection

Figure 17 shows the average execution times of the RCM–GL (George and Liu 1981), Sloan's (1989), MPG (Medeiros et al. 1993), NSloan (Kumfert and Pothen 1997), Sloan–MGPS (Reid and Scott 1999), and hGPHH-GL (Koohestani and Poli 2011) heuristics applied to 22 instances of the University of Florida sparse-matrix collection. In general, the MPG heuristic (Medeiros et al. 1993) showed lower computational cost than Sloan's (1989), NSloan (Kumfert and Pothen 1997), and Sloan–MGPS (Reid and Scott 1999). In particular, the MPG (Medeiros et al. 1993) limits the number of candidate vertices to be examined on each iteration.

5 Conclusions

Among 132 heuristics identified in the literature that were applied to bandwidth and/or profile reductions, 14 heuristics were selected as the most promising low-cost heuristics to solve these problems. In experiments with 113 and 22 instances (up to 100,196 vertices) of the Harwell-Boeing and University of Florida sparse-matrix collections, respectively, the FNCHC (Lim et al. 2003, 2004, 2007) and GPS (Gibbs et al. 1976) heuristics showed the best results with respect to bandwidth reduction of symmetric and asymmetric matrices, respectively. Additionally, the NSloan (Kumfert and Pothen 1997) and MPG (Medeiros et al. 1993) heuristics may be seen as the most promising low-cost algorithms for profile reduction of symmetric and asymmetric matrices, respectively.

Heuristics for reordering of vertices contribute to provide adequate memory location and, hence, improving cache hit rates (Das et al. 1992; Burgess and Giles 1997). Moreover, as exhibited by the computational experiment described in this paper, conclusively, the choice of a heuristic is highly dependent on the use of a specific instance. Thus, in addition to the GPS (Gibbs et al. 1976), MPG (Medeiros et al. 1993), NSloan (Kumfert and Pothen 1997), and FNCHC (Lim et al. 2003, 2004, 2007) heuristics, the RCM–GL (George and Liu 1981) and hGPHH-GL (Koohestani and Poli 2011) algorithms showed very low computational costs for bandwidth and/or profile reductions in the six sets of instances tested. These six heuristics were identified as the most promising low-cost heuristics for bandwidth and profile reductions so that the computational cost of solving large-scale linear systems by iterative methods can be reduced (depending on the preconditioner used to reduce the number of iterations of the solver applied). In addition, Sloan's, GGPS, and Sloan–MGPS heuristics can be applied to instances originating from specific application areas (see Tables 2, 3).

Additional datasets including much larger matrices shall be included in a future evaluation so that an insight is intended to be offered as to how the tested heuristics compare on very large examples that are of more practical interest. Thus, we intend to apply the 14 heuristics tested in this work and, in particular, those six heuristics selected, to perform reordering of vertices and reduce the computational cost of iterative methods for solving large-scale linear systems to verify the best method(s) in this context.

Acknowledgements This work was undertaken with the support of the FAPEMIG—Fundação de Amparo à Pesquisa do Estado de Minas Gerais (Minas Gerais Research Support Foundation, Brazil). We would like to thank Prof. Dr. Dragan Urosevic, from the Mathematical Institute SANU, for sending us the VNS-band executable programs. We would like also to thank Prof. Dr. Fei Xiao, from Beepi, for providing us the with the C programming language source code of the FNCHC heuristic. In addition, we would like to thank the reviewers for their valuable comments and suggestions.

Appendix A: Implementation of the heuristics, testing, and calibration

This appendix shows the testing and calibration performed to compare our implementations with the codes used by the original proposers of the 14 heuristics to ensure the codes we implemented were comparable to the algorithms that were originally proposed. Regarding the VNS-band heuristic, 32-bit and 64-bit executable programs of this heuristic (which were kindly provided by one of the heuristic's authors) were used. Mladenovic et al. (2010) developed a heuristic based on variable neighborhood search meta-heuristic to solve the bandwidth minimization problem. This meta-heuristic investigates increasingly distant neighbors of a solution. An initial solution is generated through applying a random breadth-first search procedure. Then, a main loop carries out three steps: shaking, local search, and neighborhood change. First, the neighborhood index k is initialized with $k_{\min}$, which is the size of the neighborhood. Second, in the shaking step, the meta-heuristic produces a solution S within the current neighborhood. Using the solution S, a local-search method produces a local-optimum solution O_l. Third, O_l replaces the current solution if the bandwidth of the solution O_l is narrower than the bandwidth of the current solution, and k is set to $k_{\min}$; otherwise, neighborhood is expanded incrementing k while $k < k_{\max}$. The VNS-band heuristic terminates when the execution time exceeds a timeout as previously established.

The FNCHC-heuristic source code was also kindly provided by one of the heuristic's authors. With this, the source code was converted in this present work to the C++ programming language. The FNCHC heuristic (Lim et al. 2003, 2004, 2007) is a variation of the NCHC heuristic (Lim et al. 2003, 2004, 2007). The FNCHC heuristic is based on a direct node adjustment method with hill climbing for solving the bandwidth minimization problem.

We asked all the other heuristics' authors for the sources and/or executables of their algorithms. However, some authors responded that they no longer had the code, some authors did not answer, and some authors explained that the programs could not be provided. Then, the remaining 12 heuristics were also implemented in the C++ programming language so that the computational costs of the heuristics could be compared appropriately. Specifically, the g++ version 4.8.2 compiler was used.

It should be noted that these heuristics are simple to implement:

- the RCM–GL (George and Liu 1981) (based on breadth-first search),
- Snay's heuristic (Snay 1976) (based on RCM method),

- Sloan's algorithm (Sloan 1989),
- the Medeiros–Pimenta–Goldenberg (MPG) heuristic (Medeiros et al. 1993) (based on Sloan's algorithm),
- the Normalized Sloan (NSloan) heuristic (Kumfert and Pothen 1997) (based on Sloan's algorithm),
- Sloan's algorithm with pseudo-peripheral vertex given by the modified GPS algorithm (Sloan–MGPS) (Reid and Scott 1999) (based on Sloan's algorithm),
- the heuristic based on genetic programming hyper-heuristic (hGPHH) (Koohestani and Poli 2011) (based on RCM method).

Details about the heuristics implemented are given below. Sections A.1 and A.2 show the testing and calibration performed with regard to the RCM–GL and hGPHH-GL methods, and the GPS algorithm, respectively.

Sloan's (1989), the NSloan (Kumfert and Pothen 1997), Sloan–MGPS (Reid and Scott 1999), and Charged System Search for bandwidth reduction (CSS-band) (Kaveh and Sharafi 2012) heuristics have important parameters that may change the results. Sections A.3 and A.4 show the testing and calibration performed with regard to Sloan's, Nsloan, and Sloan–MGPS heuristics, and the MPG heuristic, respectively. Section A.5 shows the testing and calibration performed with regard to the CSS-band heuristic. On the other hand, the other heuristics do not have parameters that affect the results.

To our knowledge, results of Snay's (1976), Burgess and Lai (1986), wonder bandwidth reduction algorithm (WBRA) (Esposito et al. 1998), Generic GPS (GGPS) (Wang et al. 2009), and CSS-band (Kaveh and Sharafi 2012) heuristics have only been published in their original papers and, unfortunately, we did not find the instances where these five heuristics were applied. On the other hand, since an efficient implementation comes at a cost in programming effort, we strictly observed the descriptions of the algorithms provided by their authors to obtain reasonably efficient implementations of (all) these heuristics.

It should be noted that it was not our intention that the results of the C++ programming language versions of the heuristics outperform results of the original implementations. Our intention was to implement reasonably efficient implementations of the heuristics tested to make it possible an appropriate comparison of the results of the 14 heuristics.

A.1: The RCM–GL and hGPHH-GL methods

In short, the RCM is a breadth-first search-based procedure and the vertices on each level are labeled in ascending order of degree number. Then, the final labeling is reverted. Similarly, the hGPHH is a breadth-first search-based procedure and the vertices on each level are labeled with a specific formula provided by the genetic programming hyper-heuristic (GPHH) (Koohestani and Poli 2011).

These two heuristics are highly dependent on the starting vertex and since Koohestani and Poli (2011) provided no description of the pseudo-peripheral vertex finder used, we employed the George–Liu algorithm (George and Liu 1979) to find a pseudo-peripheral vertex; hence, we named this method as hGPHH-GL, i.e., it is an RCM-based heuristic developed through a genetic programming hyper-heuristic. Thus, in both algorithms, the starting vertex is given here by the George–Liu algorithm.

Table 5 exhibits results of our implementation of the RCM–GL and hGPHH-GL methods applied to 11 instances of the Harwell-Boeing sparse-matrix collection. These results are compared to the results of the GHH (Koohestani and Poli 2011) and the RCM method contained in the MATLAB library (RCMM) (MATLAB 2016), which is a SPARSPAK-

Table 5 Results of the C++ programming language versions of the RCM–GL (George and Liu 1981) and hGPHH-GL methods and results of the RCMM (MATLAB 2016) and GHH (Koohestani and Poli 2011) methods in bandwidth reduction applied to 11 instances of the Harwell-Boeing sparse-matrix collection

Instance	RCM–GL	RCMM	hGPHH-GL	GHH
BCSPWR05	**59**	64	65	**58**
DWT_209	37	**33**	**34**	**34**
DWT_245	**43**	55	55	**43**
DWT_310	**15**	**15**	**13**	**13**
DWT_361	25	**15**	25	**15**
DWT_419	**34**	**34**	**33**	**33**
DWT_503	**64**	**64**	64	**51**
DWT_592	**42**	**42**	42	**41**
DWT_878	**37**	46	**37**	44
DWT_918	**57**	**57**	63	**44**
DWT_992	**65**	**65**	**63**	**63**
Number of best results	9	8	5	10
$\sum \rho_b$	0.79	0.61	1.78	0.19

based highly enhanced version of the RCM method described by George and Liu (1981). Specifically, Koohestani and Poli (2011) published these results. In general, one can consider that the results are similar, although differences can be explained by the use of the George–Liu algorithm (George and Liu 1979) to find pseudo-peripheral vertices.

A.2: The GPS algorithm

In short, the GPS algorithm (Gibbs et al. 1976) creates a level structure rooted at a pseudo-peripheral vertex. It also considers an end pseudo-peripheral vertex when building this structure. The final labeling is given starting by the end node of the rooted level structure (RLS) and the vertices on each level are labeled in decreasing order of degree number. Let $G = (V, E)$ be a connected and simple graph. Given a vertex $v \in V$, the level structure rooted at v, with depth (or the *eccentricity* of v) $\ell(v)$, is a partitioning $\mathscr{L}(v) = \{L_0(v), L_1(v), \dots, L_{\ell(v)}(v)\}$, where $L_0(v) = \{v\}$, $L_i(v) = \text{Adj}(L_{i-1}(v)) - \bigcup_{j=0}^{i-1} L_j(v)$, for $i = 1, 2, 3, \dots, \ell(v)$ and Adj(.) returns the adjacent vertices of the argument. The *width* $b(\mathscr{L}(v))$ of $\mathscr{L}(v)$ is defined as $b(\mathscr{L}(u)) = \max_{0 \le i \le \ell(u)} |L_i(u)|$.

Results of the GPS algorithm applied to the Harwell-Boeing sparse-matrix collection were provided by Martí et al. (2001), Piñana et al. (2004), Lim et al. (2007), and Rodriguez-Tello et al. (2008). These researchers compared the results of their heuristics with the results of the GPS algorithm applied to instances of this sparse-matrix collection. Their papers show the average bandwidth over the instances in each dataset. Nevertheless, a comparison of the average bandwidth (over the instances in each set) of the results obtained using the (C++ programming language version of the) GPS algorithm to the results shown in these publications is not adequate because the datasets used here contain instances with very different sizes, i.e., a heuristic that returns a very large bandwidth in a small instance would not be penalized appropriately.

Table 6 Results with respect to bandwidth (Lewis 1982) and profile (Everstine 1979; Lewis 1982) reductions obtained using the Fortran programming language version of GPS algorithm and results concerning bandwidth and profile reductions achieved by the C++ programming language version of the GPS algorithm applied to 13 instances of the Harwell-Boeing sparse-matrix collection

Instance	β		Profile	
	C++	Lewis (1982)	C++	Lewis (1982)
BCSPWR01	9	**6**	123	**100**
DWT_209	**37**	48	4816	**4744**
DWT_221	**19**	20	**1997**	2188
DWT_234	**18**	**18**	**1274**	1497
DWT_245	50	**48**	5054	**3568**
DWT_310	**13**	28	**2695**	2696
DWT_361	**14**	**14**	**4699**	5054
DWT_419	**34**	41	**8276**	8967
DWT_503	**64**	69	**14,783**	16,096
DWT_592	**36**	47	11,703	**11,284**
DWT_878	**27**	40	**19,120**	19,930
DWT_918	**58**	64	21,940	**21,297**
DWT_992	36	**35**	**33,076**	34,025
Number of best results	10	5	8	5
–	$\sum \rho_\beta = 0.57$	$\sum \rho_\beta = 2.68$	$\sum \rho_p = 0.73$	$\sum \rho_p = 0.59$

Then, we closely followed the recommendations given by Lewis (1982) in a Fortran programming language code to implement the GPS algorithm (Gibbs et al. 1976) in the C++ programming language. One of the main results of Lewis (1982) was to present a low-cost implementation of the GPS algorithm (Gibbs et al. 1976). Lewis (1982) did not explicitly present results of the GPS algorithm with respect to bandwidth and profile reductions. However, this author showed results of the Gibbs–King heuristic with regard to bandwidth and profile reductions and stated that the results of the GPS algorithm were as good as the results of the Gibbs–King algorithm. On the other hand, Lewis (1982) presented the GPS algorithm computational cost and showed that the Gibbs–King heuristic is much slower than the GPS algorithm. This author presented these results because the bandwidth and profile reductions obtained using the GPS algorithm were the same results obtained by Everstine (1979). On the other hand, Everstine (1979) showed only results concerning wavefront and profile reductions. Thus, Table 6 shows results with respect to bandwidth reductions obtained using the Gibbs–King heuristic (the same results obtained using the GPS algorithm) (Lewis 1982), results with regard to profile reductions attained by the GPS algorithm (Everstine 1979; Lewis 1982), and results concerning bandwidth and profile reductions achieved by the C++ programming language version of the GPS algorithm applied to 13 instances of the Harwell-Boeing sparse-matrix collection.

In general, the C++ programming language version of the GPS algorithm obtained better results than the Fortran programming language version of this algorithm (Lewis 1982) in relation to bandwidth reductions. On the other hand, although the C++ programming language version of the GPS algorithm obtained the largest number of best results (in eight instances) in this set of instances, the Fortran version of this algorithm (Lewis 1982) was the best one for reducing profile when considering the ρ_p metric. Nevertheless, the difference in the ρ_p metric is small and it can be explained by choices of the pseudo-peripheral vertices in these runs. Therefore, one can consider that the C++ programming language version used here is a reasonably efficient implementation of the GPS algorithm (Gibbs et al. 1976).

A.3: The Sloan, NSloan, and Sloan–MGPS heuristics

Similar to the GPS algorithm (Gibbs et al. 1976), the first step of Sloan's algorithm (Sloan 1989) returns a starting (s) and an end (e) pseudo-peripheral vertices. Then, Sloan's algorithm (Sloan 1989) creates a level structure rooted at s. The final labeling is given starting by the vertex s and the vertices are labeled using a max-priority queue according to the priority $p(v) = w_1 d(v, e) - w_2(\deg(v) + 1)$, where w_1 and w_2 are integer weights, $d(v, e)$ is the distance of vertex v to the vertex e, and $\deg(v)$ is the degree of vertex v. Sloan (1989) suggested to assign these two weights as $w_1 = 1$ and $w_2 = 2$. Setting $w_1 \gg w_2 \geq 1$ places more emphasis on the global criterion of distance from the target end vertex and forces the vertices to be labeled level by level, and the vertex-labeling algorithm is the method of King (King 1970). When setting $w_1 = 0$ and $w_2 = 1$, the vertex-labeling algorithm is similar to Snay's heuristic (Snay 1976).

One can realize that the RCM method (George 1971) obtains better bandwidth results than Sloan's algorithm (Sloan 1989) (and variations) because, in the RCM method, if $(u, v) \in E$, then $|s(u) - s(v)|$ is small, since the vertices are labeled level by level in relation to the level structure rooted at the starting vertex. In turn, Sloan's algorithm (Sloan 1989) may provide a labeling with small $|s(u) - s(v)|$, but large $d(u, v)$ when placing more emphasis on the current degree as the principal measure of priority (i.e., $w_2 > w_1$), which is a local criterion. On the other hand, one can realize that Sloan's algorithm (Sloan 1989) (and variations) obtains better profile results than the RCM method (George 1971), because the RCM method may produce β_i $(1 \leq i \leq |V|)$ similar to $\beta(A)$ and, consequently, producing a labeling with large profile. By its turn, Sloan's algorithm (Sloan 1989) (and variations) produces small β_i $(1 \leq i \leq |V|)$ by choosing to label firstly vertices with smaller degrees so that it produces a labeling with small profile [although in each iteration Sloan's algorithm (Sloan 1989) (and variations) forces to label vertices level by level—similarly to the RCM method—by increasing w_2 to the priority of the candidate vertices to be labeled].

We strictly observed the recommendations given by Sloan (Sloan 1989) and also the recommendations described in http://www.hsl.rl.ac.uk/archive/specs/mc40 (STFC 2015) in a Fortran programming language code to implement this heuristic in the C++ programming language. Likewise, we studied the Fortran programming language code of Sloan–MGPS available at http://www.hsl.rl.ac.uk/catalogue/mc60.html (STFC 2015) to implement this heuristic in the C++ programming language.

Sloan's algorithm (Sloan 1989) for finding pseudo-peripheral vertices follows three main steps. First, Sloan's algorithm (Sloan 1989) for finding pseudo-peripheral vertices selects a vertex of minimum degree v in the graph. Second, it scans $L_{\ell(v)}(v)$ and then observes a set containing only one vertex of each degree. Finally, considering the eccentricity and width of RLSs, this algorithm returns a starting (s) and an end (e) pseudo-peripheral vertices. This algorithm is a modified version of the algorithm of Gibbs et al. (1976) (which considers only the eccentricity of vertices) and is also applied together with the MPG (Medeiros et al. 1993) (see Sect. A.4) and NSloan (Kumfert and Pothen 1997) heuristics.

Reid and Scott (1999) proposed two modifications in Sloan's algorithm (Sloan 1989) for finding pseudo-peripheral vertices: it scans $L_{\ell(v)}(v)$ and observes a set containing only five vertices which are not adjacent to each other (instead of observing a set containing only one vertex of each degree) and, before returning the pseudo-peripheral vertices, it changes s by e if $\ell(e) > \ell(s)$. Therefore, the Sloan–MGPS heuristic (Reid and Scott 1999) is Sloan's vertex-labeling algorithm with pseudo-peripheral vertices given by this modified GPS algorithm.

Sloan (1989) proposed to employ two weights to label the vertices of the instance: w_1, associated with the distance $d(v, e)$ of the vertex v to the target end vertex e in $\mathscr{L}(s)$, and w_2, associated with the degree of each vertex. To provide specific detail, the priority $p(v) = w_1 d(v, e) - w_2(\deg(v) + 1)$ employed in Sloan's algorithm (Sloan 1989) presents different scales for both criteria. The value of $\deg(v) + 1$ ranges from 1 to $m + 1$ (where m is the maximum degree of the graph $G = (V, E)$), and $d(v, e)$ ranges from 0 (when $v = e$, and $d(v, e) > 0$ for $d \neq e$) to the eccentricity $\ell(e)$ (of the target end vertex e). Regarding Sloan's (1989), NSloan (Kumfert and Pothen 1997), and Sloan–MGPS (Reid and Scott 1999) heuristics, we established the two weights as these parameters were described in the original papers. When more than one pair of values have been suggested in the original papers, we performed exploratory investigations to determine the pair of values that obtains the best profile results. Then, the two weights are assigned as $w_1 = 1$ and $w_2 = 2$ for the Sloan–MGPS heuristic (Reid and Scott 1999), and as $w_1 = 2$ and $w_2 = 1$ for the NSloan heuristic (Kumfert and Pothen 1997). To give more specific detail, although assigning these two weights as $w_1 = 1$ and $w_2 = 2$, Sloan's (1989), MPG (Medeiros et al. 1993) (see Sect. A.4), and Sloan–MGPS (Reid and Scott 1999) heuristics place more emphasis on the distance $d(v, e)$ (related to $\mathscr{L}(s)$) when applied to sparse graphs because the scale of this criterion is larger than the scale of the other criterion. One can realize that, in sparse graphs, $\ell(e)$ is larger than m so that Kumfert and Pothen (1997) normalized these two criteria to propose the Normalized Sloan (NSloan) heuristic.

Results of Sloan's (Sloan 1989) and Sloan–MGPS (Reid and Scott 1999) heuristics applied to 17 symmetric instances of the University of Florida sparse-matrix collection were provided by Reid and Scott (1999). Table 7 exhibits results of the C++ programming language versions of Sloan's algorithm and the Sloan–MGPS heuristic, and the results of these heuristics provided by Reid and Scott (1999) applied to 17 symmetric instances of the University of Florida sparse-matrix collection. In this dataset composed of 17 symmetric instances, one can verify that the C++ programming language versions of the two heuristics obtained better profile results when considering the ρ_p metric. Disregarding the skirt instance (which Reid and Scott (1999) show larger profiles of 2 magnitudes in relation to the C++ programming language versions), one finds $\sum \rho_p = 0.84$ for Sloan's algorithm (Sloan 1989), against $\sum \rho_p = 0.58$ resulted by the C++ programming language version of this heuristic, and the ρ_p metric in both versions of the Sloan–MGPS is also similar when disregarding this instance.

Kumfert and Pothen (1997) provided results of the NSloan heuristic applied to instances of the University of Florida sparse-matrix collection. Kumfert and Pothen (1997) normalized the results of the NSloan heuristic in relation to the RCM method (George 1971) with pseudo-peripheral vertex given by the method of Duff et al. (1989a), which is similar to Sloan's algorithm for finding pseudo-peripheral vertices (Sloan 1989). Table 7 replicates these results. In addition, Table 7 shows results of the C++ programming language version of the NSloan heuristic normalized in relation to the results obtained using the C++ programming language version of the RCM–GL method (George and Liu 1981) applied to 17 symmetric instances of the University of Florida sparse-matrix collection. In this dataset composed of 17 symmetric instances, one finds $\sum \rho_p = 0.13$ for the original NSloan heuristic (Kumfert and Pothen 1997), against $\sum \rho_p = 1.64$ resulted by the C++ programming language version of this heuristic. One can observe that results of both implementations are similar and differences in the results can be explained by the use of different algorithms for finding pseudo-peripheral vertices.

Table 7 Results of implementations in the C++ programming language of Sloan's (1989), NSloan (Kumfert and Pothen 1997), and Sloan–MGPS (Reid and Scott 1999) heuristics and results of these heuristics provided by Kumfert and Pothen (1997) (KP97) and Reid and Scott (1999) (RS99) in profile reductions applied to 17 symmetric instances of the University of Florida sparse-matrix collection

Instance	Size	Sloan		Sloan–MGPS		NSloan	
		C++	RS99	C++	RS99	C++	KP97
barth	6691	**488,502**	490,000	485,788	**470,000**	**0.66**	**0.66**
barth4	6019	674,980	**450,000**	658,381	**330,000**	0.47	**0.27**
barth5	15,606	**2,315,620**	2,430,000	2,320,292	**1,440,000**	0.43	**0.41**
bcsstk30	28,924	**10,655,704**	15,720,000	**10,080,145**	16,150,000	0.53	**0.45**
commanche_dual	7920	**425,818**	440,000	397,955	**330,000**	0.59	**0.54**
copter1	17,222	7,128,590	**7,090,000**	7,127,731	**6,050,000**	**0.68**	0.70
copter2	55,476	43,783,488	**43,240,000**	43,208,117	**37,960,000**	**0.53**	0.56
finance256	37,376	**6,183,691**	6,570,000	**6,319,706**	6,350,000	0.22	**0.19**
ford1	18,728	2,399,069	**2,340,000**	**2,329,436**	2,350,000	**0.80**	0.81
ford2	100,196	**40,458,534**	40,630,000	**40,061,732**	41,050,000	**0.71**	**0.71**
nasasrb	54,870	18,486,625	**18,350,000**	**18,559,413**	19,010,000	**0.88**	0.91
onera_dual	85,567	114,445,935	**113,670,000**	112,617,638	**87,750,000**	0.46	**0.41**
pds10	16,558	**11,401,308**	13,680,000	10,967,390	**9,360,000**	0.34	**0.31**
shuttle_eddy	10,429	**583,595**	590,000	**579,306**	620,000	0.82	**0.81**
skirt	12,598	**728,955**	34,120,000	**699,284**	36,600,000	0.72	**0.67**
tandem_dual	94,069	90,000,767	**87,790,000**	87,639,361	**66,210,000**	0.54	**0.51**
tandem_vtx	18,454	6,300,173	**6,290,000**	5,983,094	**5,720,000**	0.37	**0.35**
No. of best results	–	9	8	7	10	6	13
$\sum \rho_p$	–	**0.58**	46.65	**2.99**	52.07	1.64	**0.13**

A.4: The MPG heuristic

The MPG heuristic (Medeiros et al. 1993) employs two max-priority queues: t contains vertices that are candidate vertices to be labeled, and q contains vertices belonging to t and also vertices that can be inserted to t. Similar to Sloan's algorithm (Sloan 1989) (and variations), the current degree of a vertex is the number of adjacencies to vertices that neither have been labeled nor belong to q. A main loop performs three steps. First, a vertex v is inserted into q to maximize a specific priority function. Second, the current degree currdeg_v of each vertex $v \in t$ is observed: the algorithm labels a vertex v if $\text{currdeg}_v = 0$, and the algorithm removes from t a vertex v (i.e., $t \leftarrow t - \{v\}$) if $\text{currdeg}_v > 1$. Third, if t is empty, the algorithm inserts into t each vertex $u \in q$ with priority $p_u \geq p_{\max}(q) - 1$, where $p_{\max}(q)$ returns the maximum priority among the vertices in q.

Table 8 exhibits results of the C++ programming language version of the MPG heuristic and the results of this heuristic provided by Medeiros et al. (1993) applied to 30 symmetric instances of the Harwell-Boeing sparse-matrix collection. In this dataset composed of 30 instances, one can verify that the C++ programming language version of the MPG heuristic obtained better profile results than the original version of this heuristic.

Table 8 Results of an implementation in the C++ programming language of the MPG heuristic (Medeiros et al. 1993) and results of this heuristic provided by Medeiros et al. (1993) in profile reductions applied to 30 instances of the Harwell-Boeing sparse-matrix collection

Instance	Size	MPG	
		C++	Medeiros et al. (1993)
DWT_59	59	**236**	284
DWT_66	66	**128**	193
DWT_72	72	**205**	241
DWT_87	87	**470**	534
DWT_162	162	**1354**	1546
DWT_193	193	**4401**	4553
DWT_198	198	**1160**	1301
DWT_209	209	**3003**	3078
DWT_221	221	**1818**	2005
DWT_234	234	**896**	1064
DWT_245	245	**2617**	2682
DWT_307	307	**6742**	7185
DWT_310	310	**2657**	2951
DWT_346	346	**6309**	6556
DWT_361	361	**4765**	5067
DWT_419	419	**6852**	7194
DWT_492	492	**2958**	3359
DWT_503	503	**14,164**	14,439
DWT_512	512	**4461**	4727
DWT_592	592	**9526**	10,031
DWT_607	607	**13,741**	14,570
DWT_758	758	**6564**	7249
DWT_869	869	**13,108**	13,874
DWT_878	878	**18,693**	19,782
DWT_918	918	**16,139**	16,984
DWT_992	992	**33,376**	34,524
DWT_1005	1005	**33,413**	33,490
DWT_1007	1007	**19,944**	22,808
DWT_1242	1242	**36,102**	37,842
DWT_2680	2680	**87,275**	89,193
Number of best results	–	30	0
$\sum \rho_p$	–	**0.00**	2.84

A.5: The CSS-band heuristic

Regarding the charged system search for bandwidth reduction (CSS-band) heuristic (Kaveh and Sharafi 2012), 500 iterations were used as a termination criterion, and 1 charged particle per 100 vertices was used in the tests performed. These values are the same used by Kaveh and Sharafi (2012). This heuristic was also tested with less iterations and less charged particles. This considerably decreased the computational cost of the heuristic; however, the bandwidth and profile reductions also decreased substantially. In addition, tests with more iterations and more charged particles were performed. Although it slightly improved the bandwidth and profile reductions, the computational cost increased considerably.

Appendix B: Results

The first and second parts of Table 9 contain the results of 14 heuristics applied to reduce the bandwidth and profile of 15 symmetric instances of the Harwell-Boeing sparse-matrix collection, respectively. Numbers in bold face are the best results. Table 10 shows the average execution times of the 14 heuristics applied to 15 symmetric instances of the Harwell-Boeing sparse-matrix collection. Similar to the other tables related to average execution times shown below, Table 10 also shows the number of best results, the rank order according to the $\sum \rho_t$ metric, and the mode of the order of magnitude of these 14 heuristics. In particular, to achieve the maximal precision of the metrics, we set up several decimal cases to avoid draws.

Tables 11 and 12 contain the results of 14 heuristics applied to reduce the bandwidth and profile of 35 symmetric instances of the Harwell-Boeing sparse-matrix collection, respectively. In particular, Table 11 contains the results of the VNS-band heuristic set at 7 and 8 s. This was done to show that 8 s was the lowest timeout required for the VNS-band heuristic to obtain better results than the FNCHC heuristic, which was the second best in reducing bandwidth, according to the ρ_b metric. Table 12 also contains the results of the VNS-band heuristic set with 500 s. Even so, when considering profile reductions, the VNS-band heuristic was dominated by the MPG (Medeiros et al. 1993), Sloan–MGPS (Reid and Scott 1999), Sloan's (1989), and NSloan (Kumfert and Pothen 1997) heuristics in this set of 35 symmetric instances. Table 13 shows the average execution times of the 14 heuristics applied to 35 symmetric instances of the Harwell-Boeing sparse-matrix collection.

Table 14 contains the results of 14 heuristics applied to reduce the bandwidth and profile of 18 asymmetric instances of the Harwell-Boeing sparse-matrix collection. In particular, in the GENT113 instance, it was only possible to apply the 64-bit executable program of the VNS-band heuristic with a limit of 6 s. Therefore, the bandwidth and profile values presented for this instance are obtained using the VNS-band (6 s) heuristic. Table 15 shows the average execution times of the 14 heuristics applied to 18 asymmetric instances of the Harwell-Boeing sparse-matrix collection.

Tables 16 and 17 contain the results of 14 heuristics applied to reduce the bandwidth and profile of 45 asymmetric instances of the Harwell-Boeing sparse-matrix collection, respectively. In particular, it was not possible to apply the 64-bit executable program of the VNS-band heuristic in the SHERMAN4 instance. In this instance, the values are shown after being obtained using the 32-bit executable program of the VNS-band heuristic, considering the same runtime limit. Table 18 shows the average execution times of the 14 heuristics applied to 45 asymmetric instances of the Harwell-Boeing sparse-matrix collection.

Table 19 contains the results of 10 heuristics applied to reduce the bandwidth and profile of 17 symmetric instances of the University of Florida sparse-matrix collection. Table 20 shows the average execution times of the 10 heuristics applied to 17 symmetric instances of the University of Florida sparse-matrix collection.

Table 21 contains the results of 10 heuristics applied to reduce the bandwidth and profile of five asymmetric instances of the University of Florida sparse-matrix collection. Table 21 also shows the average execution times of the 10 heuristics applied to five asymmetric instances of the University of Florida sparse-matrix collection.

Table 9 Results of 14 heuristics applied to reduce bandwidth and profile of 15 symmetric instances of the Harwell-Boeing sparse-matrix collection

Result	Instance	n	β_0	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
Bandwidth	ASH85	85	39	14	**9**	15	10	21	10	10	19	34	10	23	10	47	76
	BCSPWR01	39	38	7	**5**	9	8	12	8	8	15	12	9	11	6	17	31
	BCSPWR02	49	34	10	**7**	12	13	32	13	14	23	28	10	26	11	35	43
	BCSPWR03	118	115	**10**	11	23	19	42	17	19	32	42	15	32	19	94	104
	BCSSTK01	48	35	24	**16**	33	27	34	27	26	33	40	26	30	24	33	42
	BCSSTK04	132	47	**37**	**37**	57	54	74	54	54	72	63	46	60	47	131	127
	BCSSTK05	153	28	**20**	**20**	32	24	31	26	28	29	33	23	26	28	36	146
	BCSSTK22	138	111	**10**	**10**	16	14	26	15	14	23	55	14	25	12	66	125
	CAN_144	144	142	**13**	**13**	22	15	26	18	14	24	24	15	21	19	28	134
	CAN_161	161	79	**18**	19	30	30	31	30	**18**	31	34	**18**	30	24	43	152
	DWT_234	234	48	12	**11**	23	17	39	19	24	35	55	17	36	17	90	212
	LUND_A	147	23	**23**	**23**	28	**23**	38	**23**	**23**	31	35	**23**	35	**23**	89	140
	LUND_B	147	23	**23**	**23**	31	23	35	23	23	32	33	23	35	23	73	140
	NOS1	237	4	**3**	4	4	4	4	4	4	4	4	**3**	4	4	5	220
	NOS4	100	13	**10**	**10**	17	13	24	12	12	24	19	13	25	11	22	91
	Number of best results	–	–	10	**12**	0	2	0	2	3	0	0	4	0	2	0	0
	$\sum \rho_b$	–	–	2.0	**0.5**	10.3	6.2	21.0	6.5	6.3	17.6	25.7	4.3	17.4	4.9	45.7	174.7
	Ranking	–	–	2	**1**	8	5	11	7	6	10	12	3	9	4	13	14

Table 9 continued

Result	Instance	n	$profile_0$	VNS-band (500s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
Profile	ASH85	85	1153	686	593	603	608	497	589	638	501	**493**	589	535	619	875	2434
	BCSPWR01	39	292	159	126	121	171	97	123	123	102	**88**	123	114	138	106	367
	BCSPWR02	49	377	306	224	223	344	164	214	356	149	**126**	212	160	296	172	565
	BCSPWR03	118	1288	740	841	736	1150	**438**	774	673	454	473	1073	521	1128	1111	4063
	BCSSTK01	48	851	759	554	618	749	491	654	676	**479**	530	676	484	729	661	923
	BCSSTK04	132	3631	3576	3555	3398	4079	3202	3585	3555	3238	**3167**	3493	3239	3631	5246	8077
	BCSSTK05	153	2449	2353	2360	2360	2540	2253	2313	2414	**2218**	**2218**	2465	2229	2449	2341	10002
	BCSSTK22	138	2124	883	882	878	948	823	891	873	826	**709**	870	832	968	919	5820
	CAN_144	144	7355	1295	1321	1323	1572	**972**	1071	1308	**972**	**972**	1096	973	1859	1239	8012
	CAN_161	161	3378	2610	2670	2753	3233	2610	3079	2610	2610	2610	2610	2576	2796	**2562**	10199
	DWT_234	234	1765	1460	1472	1155	1801	**856**	1363	1295	866	867	1274	921	1838	1458	14372
	LUND_A	147	2870	2471	2507	2363	**2303**	2316	**2303**	**2303**	2325	2370	**2303**	2482	2870	4934	9510
	LUND_B	147	2870	2532	2492	2366	2299	2313	**2229**	2303	2321	2374	**2229**	2509	2870	4951	9346
	NOS1	237	780	468	507	469	472	**467**	**467**	**467**	**467**	**467**	468	470	624	470	16663
	NOS4	100	766	742	781	852	892	704	761	900	**690**	691	936	725	845	737	3413
	Number of best results	–	–	0	0	0	1	4	3	2	5	**8**	2	0	0	1	0
	$\sum \rho_p$	–	–	5.68	4.46	3.83	8.18	0.71	3.85	5.12	0.66	**0.31**	4.57	1.39	8.13	7.69	103.03
	Ranking	–	–	10	7	5	13	3	6	9	2	**1**	8	4	12	11	14

Table 10 Average execution times, in seconds, of 14 heuristics applied to 15 symmetric instances of the Harwell-Boeing sparse-matrix collection

Instance	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
ASH85	503.84300	0.02657	0.00025	**0.00008**	0.00013	**0.00008**	0.00038	0.00885	0.00757	0.00022	0.00009	0.00360	0.00027	0.17278
BCSPWR01	527.31500	0.00745	0.00012	0.00003	0.00014	**0.00002**	0.00011	0.00014	0.00248	0.00008	0.00291	0.00070	0.00462	0.10229
BCSPWR02	635.63000	0.00900	0.00019	0.00004	0.00016	**0.00002**	0.00015	0.00016	0.00004	0.00011	0.00004	0.00307	0.00016	0.11034
BCSPWR03	500.57900	0.02613	0.00089	0.00011	0.00018	**0.00008**	0.00061	0.00018	0.00872	0.00034	0.00010	0.02567	0.00085	0.26830
BCSSTK01	621.28400	0.02829	0.00022	**0.00005**	0.00018	**0.00005**	0.00020	0.00018	0.00873	0.00019	0.00582	0.00880	0.00022	0.11412
BCSSTK04	500.28800	0.10722	0.00186	0.00036	**0.00018**	0.00041	0.01205	0.00018	0.00032	0.00178	0.00050	0.02050	0.00671	0.30567
BCSSTK05	500.63300	0.06697	0.00138	0.00029	0.00017	0.00029	0.00956	0.00017	**0.00015**	0.00134	0.00022	0.03248	0.00230	0.32391
BCSSTK22	500.37100	0.02796	0.00056	**0.00010**	0.00017	0.00011	0.00178	0.00017	0.00015	0.00063	0.00016	0.02015	0.00049	0.27752
CAN_144	500.47800	0.03813	0.00063	0.00014	0.00017	0.00018	0.00119	0.00017	**0.00008**	0.00059	0.00012	0.08453	0.00114	0.42721
CAN_161	500.43200	0.07352	0.00145	**0.00013**	0.00017	0.00014	0.00170	0.00018	0.00017	0.00076	0.00014	0.01367	0.00112	0.41429
DWT_234	500.22600	0.04197	0.00133	0.00014	0.00017	**0.00010**	0.00142	0.00018	0.00023	0.00079	0.00019	0.01059	0.00060	0.59239
LUND_A	501.48600	0.07169	0.00059	0.00021	0.00016	0.00028	0.00627	0.00016	**0.00013**	0.00171	0.00022	0.02714	0.00119	0.42032
LUND_B	501.53500	0.07329	0.00069	0.00021	0.00014	0.00030	0.00620	0.00016	**0.00013**	0.00118	0.00021	0.02720	0.00121	0.39847
NOS1	500.41400	0.02006	0.00043	0.00013	**0.00012**	0.00015	0.00277	0.00014	0.00014	0.00118	0.00016	0.00867	0.00033	0.44627
NOS4	500.51900	0.03208	0.00041	0.00007	0.00008	0.00009	0.00035	0.00011	0.00007	0.00027	**0.00006**	0.00392	0.00016	0.31988
Number of best results	0	0	0	4	2	**6**	0	1	4	0	1	0	0	0
$\sum \rho_t$	129,478,152.65	6645.17	97.15	**6.54**	21.15	6.97	333.37	131.12	501.96	88.77	267.21	3129.08	353.85	54,865.05
Ranking	14	12	5	**1**	3	2	8	6	10	4	7	11	9	13
Magnitude	3	−2	−4	−4	−4	−4	−3	−4	−3	−4	−4	−2	−4	−1

Table 11 Results of 14 heuristics applied to reduce bandwidth of 35 symmetric instances of the Harwell-Boeing sparse-matrix collection

Instance	n	β_0	VNS-band (7 s)	VNS-band (8 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
494_BUS	494	428	**35**	**35**	41	107	87	225	62	67	227	308	56	252	56	344	458
662_BUS	662	335	**51**	**51**	53	101	125	325	135	88	308	502	85	431	120	591	622
685_BUS	685	550	**39**	**39**	44	115	82	161	102	72	172	530	84	308	57	678	647
ASH292	292	24	**19**	**19**	21	53	34	108	28	25	106	85	34	106	24	212	278
BCSPWR04	274	265	**25**	**25**	26	72	77	206	65	66	199	201	48	158	44	252	255
BCSPWR05	443	435	**29**	**29**	36	65	64	154	59	47	169	153	49	172	50	424	415
BCSSTK06	420	47	**45**	**45**	47	81	49	131	50	52	141	155	47	147	47	284	409
BCSSTK19	817	567	**15**	**15**	17	22	22	69	22	22	76	308	18	72	22	296	790
BCSSTK20	485	20	**13**	**13**	16	25	20	307	20	20	304	350	19	298	18	191	467
BCSSTM07	420	47	49	**45**	47	81	52	128	54	54	153	142	50	143	47	322	408
CAN_292	292	282	**39**	**39**	44	121	67	184	74	74	173	199	78	197	66	251	276
CAN_445	445	403	**58**	**58**	62	109	94	294	89	85	262	340	88	250	105	307	426
CAN_715	715	611	165	108	**90**	126	133	198	133	133	186	259	126	183	121	367	691
CAN_838	838	837	93	**91**	94	217	138	220	124	120	190	176	136	217	128	784	816
DWT_209	209	184	**23**	**23**	24	39	34	104	37	30	125	99	37	99	37	179	197
DWT_221	221	187	**13**	**13**	**13**	22	17	34	16	16	25	39	19	48	19	70	207
DWT_245	245	115	25	**22**	**22**	60	55	117	43	43	117	88	50	130	44	115	230
DWT_310	310	28	**12**	**12**	**12**	20	13	21	15	13	21	27	13	23	17	49	296
DWT_361	361	50	16	16	16	24	25	41	25	**14**	42	25	**14**	44	19	50	344
DWT_419	419	356	**28**	**28**	**28**	47	33	102	34	33	116	137	34	106	44	226	402
DWT_503	503	452	48	48	**47**	88	64	257	64	64	226	232	64	194	54	355	487
DWT_592	592	259	33	33	**31**	65	42	153	42	42	153	169	36	155	52	340	573
DWT_878	878	519	**27**	**27**	37	70	37	86	37	37	67	538	**27**	183	49	463	850
DWT_918	918	839	**38**	**38**	41	73	63	137	57	52	176	330	58	106	51	655	885
DWT_992	992	513	73	73	52	63	63	127	65	63	127	68	**36**	132	57	991	969

Table 11 continued

Instance	n	β_0	VNS-band (7 s)	VNS-band (8 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
GR_30_30	900	31	59	59	47	101	59	95	59	49	85	62	49	59	**31**	74	872
JAGMESH1	936	778	31	31	28	48	28	51	**27**	**27**	50	50	**27**	49	43	725	903
NOS2	957	4	**3**	**3**	4	4	4	4	4	4	4	4	**3**	4	4	5	911
NOS3	960	43	79	79	64	80	79	165	81	65	115	85	83	130	**43**	515	939
NOS5	468	178	**64**	**64**	67	134	88	232	92	90	230	226	88	221	122	459	452
NOS6	675	30	20	20	19	29	**16**	29	17	**16**	29	29	**16**	29	21	39	641
NOS7	729	81	81	81	67	76	67	120	**65**	**65**	114	114	**65**	98	81	83	701
PLAT362	362	249	**36**	**36**	**36**	87	53	110	52	53	108	152	52	91	45	204	350
PLSKZ362	362	248	**20**	**20**	**20**	50	34	69	36	37	58	82	33	68	25	84	341
SHERMAN1	1000	100	56	56	57	96	57	244	61	57	225	326	57	176	**55**	542	947
Number of best results	–	–	21	**24**	9	0	1	0	2	4	0	0	7	0	3	0	0
$\sum \rho_b$	–	–	4.7	**3.9**	4.2	38.0	21.3	119.4	20.1	15.7	117.1	174.5	14.8	123.4	16.2	300.0	881.9
Ranking	–	–	3	**1**	2	9	8	11	7	5	10	13	4	12	6	14	15

Table 12 Results of 14 heuristics applied to reduce profile of 35 symmetric instances of the Harwell-Boeing sparse-matrix collection

Instance	n	$profile_0$	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
494_BUS	494	40,975	8630	12,201	11,215	22,601	**3669**	10,566	12,048	3734	3788	12,569	4738	17,776	5938	61,693
662_BUS	662	45,165	18,011	21,663	17,233	42,965	9275	32,903	30,699	9180	10,218	18,616	**8801**	34,954	19,944	118,988
685_BUS	685	28,261	15,107	20,397	17,865	25,787	**8560**	25,606	16,532	8581	9254	17,715	8878	25,249	40,690	145,227
ASH292	292	4224	3832	4100	4215	5433	3112	3898	4266	3035	2933	4000	**2796**	4224	5817	31,572
BCSPWR04	274	21,015	4039	4482	5269	9327	**2723**	4825	4909	**2723**	2845	7345	2756	8250	6702	23,186
BCSPWR05	443	36,248	8207	10,013	7503	16,041	3419	8825	8656	3430	**3310**	10,188	3823	14,993	11,626	52,418
BCSSTK06	420	14,691	14,380	15,062	15,056	15,086	13,408	13,238	**13,200**	13,555	13,404	13,287	13,503	14,691	31,996	76,841
BCSSTK19	817	74,051	**8372**	10,413	8828	10,950	9250	9594	9598	8613	26,015	8721	8450	10,678	51,345	255,853
BCSSTK20	485	4309	3603	4234	4006	5303	3246	5069	4378	3244	3177	4416	**3124**	3869	3311	81,118
BCSSTM07	420	14,691	14,131	15,047	14,923	15,323	13,699	13,816	13,786	13,769	13,895	**13,411**	13,630	14,691	38,004	75,258
CAN_292	292	23,170	8807	9313	10,161	11,636	5219	8589	9113	5210	5036	9414	**4396**	12,402	13,588	31,862
CAN_445	445	22,321	19,105	21,298	20,243	27,248	**16,435**	22,445	21,264	17,637	18,188	20,686	16,511	23,932	25,727	77,494
CAN_715	715	72,423	39,525	43,445	36,709	47,761	30,834	38,449	38,540	30,007	30,322	40,364	**29,810**	57,989	32,537	200,931
CAN_838	838	207,200	44,356	54,838	45,165	82,204	34,152	34,295	73,168	32,607	31,569	52,107	**31,182**	84,109	131,530	283,028
DWT_209	209	9503	3617	3680	3379	4433	3409	3980	3414	3434	**3122**	4816	3313	4554	3950	16,770
DWT_221	221	9910	2002	2077	2063	2210	1861	2011	2019	1842	1787	1997	**1709**	2225	3458	18,355
DWT_245	245	3934	3467	3558	4091	7270	2530	4620	6309	2471	2393	5054	**2290**	6988	4408	19,668
DWT_310	310	2696	2773	2795	2814	2759	2658	2836	2754	2661	**2641**	2695	2658	2764	4522	36,499
DWT_361	361	5084	4785	4902	4940	5337	4891	5139	**4699**	4907	4706	**4699**	4755	5030	9147	50,067
DWT_419	419	39,726	8525	8936	8572	9447	6955	8221	9684	**6822**	6908	8276	6834	11,172	16,058	68,002
DWT_503	503	35,914	15,737	16,862	15,381	17,101	**13,608**	14,816	14,802	14,259	19,461	14,783	14,164	17,441	23,803	102,643
DWT_592	592	28,805	12,533	12,534	11,088	12,043	9825	10,983	10,984	9818	12,423	11,703	**9580**	20,918	27,267	136,471
DWT_878	878	26,055	20,318	21,543	21,856	22,026	18,815	21,034	21,161	18,978	**17,981**	19,120	18,693	21,903	30,607	29,7043
DWT_918	918	108,355	21,935	24,562	21,760	27,339	17,533	22,187	20,983	16,771	35,185	21,940	**15,768**	24,671	61,037	32,5236

Table 12 continued

Instance	n	profile$_0$	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
DWT_992	992	262,306	34,588	36,112	36,894	37,980	33,000	37,136	36,926	33,064	**32,656**	33,076	33,376	36,106	60,400	434,021
GR_30_30	900	26,970	28,014	31,328	36,049	35,496	28,664	33,872	29,520	28,750	**26,564**	27,916	26,970	26,970	41,594	312,960
JAGMESH1	936	37,240	22,878	22,181	22,136	22,150	**21,541**	21,817	21,817	21,588	21,588	21,817	21,587	22,458	35,303	318,013
NOS2	957	3180	1908	2082	1909	1912	**1907**	**1907**	**1907**	**1907**	**1907**	1908	1910	2544	1910	277,961
NOS3	960	39,101	40,223	44,862	40,306	48,938	39,465	47,536	41,276	**38,386**	38,416	42,278	38,387	39,101	71,121	405,513
NOS5	468	27,286	23,308	24,232	24,732	28,509	20,273	25,228	28,180	**20,243**	20,475	27,632	20,494	30,619	44,648	89,723
NOS6	675	16,229	9729	9821	9301	9357	**9095**	9305	9305	**9095**	**9095**	9305	**9095**	9523	9513	148,014
NOS7	729	53,144	35,729	35,296	34,826	34,704	34,473	**34,110**	**34,110**	34,698	34,690	**34,110**	35,202	53,144	37,201	190,627
PLAT362	362	45,261	9897	10,184	14,062	14,363	**8511**	13,366	14,936	8517	12,171	13,366	8517	11,765	15,237	57,533
PLSKZ362	362	43,090	4833	5123	6318	7265	**3286**	6643	6948	3363	4350	6522	3324	5325	5057	45,268
SHERMAN1	1000	34,740	26,017	27,431	28,167	27,054	24,859	26,673	26,997	24,725	**21,831**	26,109	22,768	27,536	53,571	218,119
Number of best results	–	–	1	0	0	0	10	2	4	6	9	3	**11**	0	0	0
$\sum \rho_p$	–	–	11.2	16.4	14.7	32.4	1.5	17.4	18.3	1.4	5.7	16.9	**0.9**	29.2	43.5	514.7
Ranking	–	–	5	7	6	12	3	9	10	2	4	8	**1**	11	13	14

Table 13 Average execution times, in seconds, of 14 heuristics applied to 35 symmetric instances of the Harwell-Boeing sparse-matrix collection

Instance	VNS-band (7 s)	VNS-band (8 s)	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
494_BUS	7.11855	8.01115	500.11500	0.11930	0.00871	0.00030	**0.00024**	0.00030	0.00532	0.00043	0.00047	0.00184	0.00033	0.08981	0.00075	1.59015
662_BUS	7.11887	8.01106	500.12000	0.19709	0.01813	0.00054	**0.00042**	0.00049	0.00516	0.00094	0.00115	0.00354	0.00048	2.04695	0.00178	3.35895
685_BUS	7.11922	8.00967	500.09900	0.20994	0.00878	0.00054	**0.00039**	0.00065	0.00792	0.00077	0.00114	0.00431	0.00063	0.31447	0.00303	3.23729
ASH292	7.11108	8.01350	500.33600	0.09472	0.00360	0.00031	**0.00021**	0.00026	0.00193	0.00031	0.00034	0.00133	0.00032	0.01937	0.00062	0.73219
BCSPWR04	7.12557	8.01258	500.31300	0.08774	0.00189	0.00024	**0.00016**	0.00020	0.00245	0.00025	0.00027	0.00093	0.00025	0.10597	0.00055	0.74097
BCSPWR05	7.11811	8.01152	500.18600	0.11853	0.00594	0.00046	0.00025	**0.00022**	0.00364	0.00038	0.00038	0.00151	0.00030	0.04451	0.00107	1.31506
BCSSTK06	7.11770	8.01484	500.12700	0.24036	0.00364	0.00080	**0.00040**	0.00121	0.00514	0.00071	0.00067	0.00992	0.00075	0.25516	0.00248	1.48727
BCSSTK19	7.12342	8.01750	500.17700	0.18535	0.00663	0.00085	**0.00050**	0.00079	0.02591	0.00065	0.00241	0.00941	0.00080	0.53780	0.00377	5.03361
BCSSTK20	7.12066	8.01318	500.29200	0.08821	0.00714	0.00043	**0.00027**	0.00039	0.00391	0.00032	0.00035	0.00350	0.00043	0.53863	0.00053	1.66873
BCSSTM07	7.11651	8.01383	500.13500	0.22470	0.00312	0.00073	**0.00039**	0.00106	0.00338	0.00068	0.00069	0.00789	0.00076	0.24463	0.00261	1.55799
CAN_292	7.11619	8.01043	500.23800	0.11556	0.00416	0.00087	**0.00024**	0.00030	0.00262	0.00055	0.00055	0.00205	0.00033	0.25688	0.00113	0.55259
CAN_445	7.11684	8.00986	500.18400	0.21571	0.00725	**0.00042**	0.00050	0.00054	0.00296	0.00150	0.00156	0.00376	0.00065	1.07809	0.00195	1.49066
CAN_715	7.12165	8.00848	500.07400	0.25288	0.00716	0.00285	**0.00054**	0.00133	0.01361	0.00206	0.00220	0.00854	0.00082	2.70695	0.00290	3.54888
CAN_838	7.12106	8.01337	500.10800	0.36517	0.02074	0.00139	**0.00112**	0.00145	0.01964	0.00266	0.00254	0.01118	0.00140	2.13967	0.00793	5.88495
DWT_209	7.11117	8.01087	500.42400	0.08045	0.00215	0.00027	**0.00014**	0.00020	0.00164	0.00031	0.00030	0.00091	0.00021	0.06519	0.00050	0.47427
DWT_221	7.11641	8.01254	500.47200	0.06564	0.00156	0.00025	**0.00014**	0.00028	0.00247	0.00018	0.00019	0.00156	0.00024	0.07156	0.00041	0.67145
DWT_245	7.12919	8.01335	500.35300	0.08183	0.00273	0.00024	**0.00016**	0.00018	0.00146	0.00028	0.00026	0.00083	0.00022	0.11240	0.00046	0.66980
DWT_310	7.12016	8.01318	500.41400	0.08185	0.00115	0.00038	**0.00017**	0.00025	0.00908	0.00024	0.00023	0.00383	0.00023	0.06047	0.00049	0.92464
DWT_361	7.11770	8.01045	500.27600	0.11941	0.00258	0.00030	**0.00019**	0.00029	0.00684	0.00036	0.00033	0.00363	0.00024	0.03272	0.00074	1.02529
DWT_419	7.11792	8.01108	500.21900	0.15123	0.00209	0.00047	**0.00028**	0.00057	0.00486	0.00053	0.00056	0.00402	0.00049	0.50116	0.00129	1.63497
DWT_503	7.10700	8.00890	500.13600	0.25259	0.00321	0.00061	**0.00039**	0.00079	0.02181	0.00105	0.00149	0.00648	0.00060	0.16225	0.00231	2.02977
DWT_592	7.11370	8.00978	500.18700	0.20212	0.00220	0.00076	**0.00037**	0.00063	0.01973	0.00074	0.00091	0.00768	0.00049	1.06711	0.00198	2.02271

Table 13 continued

Instance	VNS-band (7 s)	VNS-band (8 s)	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
DWT_878	7.11298	8.01239	500.11600	0.32332	0.00663	0.00154	**0.00042**	0.00068	0.02766	0.00114	0.00129	0.01528	0.00062	1.38669	0.00230	6.16999
DWT_918	7.11517	8.01000	500.10500	0.34543	0.00927	0.00147	**0.00067**	0.00084	0.03426	0.00125	0.00265	0.01241	0.00101	0.40884	0.00467	6.58047
DWT_992	7.11654	8.01118	500.07600	0.52678	0.02726	0.00314	**0.00066**	0.00163	0.07698	0.00126	0.00139	0.03642	0.00115	0.44736	0.00494	8.43124
GR_30_30	7.11949	8.01269	500.14000	0.36400	0.01476	0.00316	0.00058	**0.00025**	0.01141	0.00186	0.00173	0.01394	0.00063	0.03762	0.00283	7.29861
JAGMESH1	7.11092	8.00928	500.16800	0.27140	0.00146	0.00068	**0.00053**	0.00082	0.00625	0.00143	0.00148	0.01456	0.00075	3.29223	0.00255	6.66841
NOS2	7.11763	8.01091	500.09800	0.07614	0.00190	0.00053	0.00043	0.00067	0.02009	**0.00042**	0.00043	0.00722	0.00060	0.07515	0.00070	7.06462
NOS3	7.12703	8.01534	500.11400	0.54996	0.06605	0.00366	**0.00089**	0.00174	0.01873	0.00166	0.00169	0.03008	0.00128	0.05944	0.00532	7.56294
NOS5	7.11101	8.01442	500.18000	0.30422	0.00614	0.00071	**0.00058**	0.00077	0.00636	0.00141	0.00126	0.00483	0.00073	1.20624	0.00278	1.68894
NOS6	7.11187	8.01380	500.23000	0.17752	0.00078	0.00049	**0.00035**	0.00038	0.01452	0.00067	0.00059	0.00630	0.00046	0.18980	0.00086	3.23702
NOS7	7.11345	8.01123	500.19100	0.34139	0.00217	**0.00053**	0.00055	**0.00053**	0.00450	0.00229	0.00228	0.00925	0.00081	2.15392	0.00256	4.29754
PLAT362	7.11487	8.01369	500.19100	0.22143	0.00483	0.00075	**0.00028**	0.00074	0.01049	0.00055	0.00082	0.00567	0.00050	0.21445	0.00151	1.02139
PLSKZ362	7.10801	8.01281	500.32100	0.09231	0.00227	0.00034	**0.00019**	0.00030	0.00684	0.00029	0.00034	0.00283	0.00028	0.14088	0.00055	1.01867
SHERMAN1	7.11647	8.01266	500.09100	2.23921	0.00698	**0.00068**	0.00164	0.00114	0.00923	0.00314	0.00283	0.01246	0.00198	0.15679	0.00454	9.30472
Number of best results	0	0	0	0	0	3	**29**	3	0	1	0	0	0	0	0	0
$\sum \rho_t$	830,340	934,755	58,365,332	22,151	599	46	**3**	22	970	45	55	588	20	48,871	138	265,316
Ranking	14	15	16	11	9	5	**1**	3	10	4	6	8	2	12	7	13
Magnitude	1	1	3	−1	−3	−4	−4	−4	−3	−4	−3	−3	−4	−1	−3	1

Table 14 Results of 14 heuristics applied to reduce bandwidth and profile of 18 asymmetric instances of the Harwell-Boeing sparse-matrix collection

Result	Instance	n	β_0	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
Bandwidth	ARC130	130	125	**63**	**63**	78	129	128	129	100	128	128	101	126	125	127	98
	CURTIS54	54	44	16	**10**	16	44	25	50	13	23	25	12	36	44	40	44
	FS_183_1	183	181	**60**	63	117	141	173	175	113	175	177	149	174	181	177	167
	GENT113	113	101	**27**[a]	39	38	125	60	111	44	59	62	34	70	94	103	104
	GRE_115	115	101	**23**	24	39	98	72	96	33	65	83	35	70	101	83	106
	GRE_185	185	60	**21**	22	33	41	39	39	23	37	37	**21**	37	60	65	172
	IBM32	32	26	14	**11**	17	27	17	25	18	20	18	18	21	26	28	26
	IMPCOL_B	59	43	24	**20**	28	53	51	54	28	49	41	41	44	43	57	54
	IMPCOL_C	137	91	**30**	32	65	109	89	110	58	81	84	45	87	91	130	126
	LNS_131	131	111	**20**	37	41	98	76	93	31	55	70	29	54	111	108	113
	MCCA	180	65	**37**	38	73	78	67	121	52	67	67	54	67	65	140	170
	PORES1	30	11	**7**	**7**	9	9	11	14	8	9	9	9	10	11	13	25
	STEAM3	80	43	11	**7**	8	**7**	10	**7**	10	10	10	**7**	9	43	11	75
	WEST0132	132	94	**32**	35	57	125	92	91	47	92	93	62	73	94	114	122
	WEST0156	156	147	**36**	38	80	148	132	151	54	118	121	65	97	147	128	144
	WEST0167	167	158	**34**	35	84	165	99	166	65	88	95	69	119	158	124	155
	WILL199	199	169	**65**	68	137	189	189	186	108	187	186	104	149	169	183	182
	WILL57	57	44	14	**7**	8	23	28	20	14	28	17	9	13	44	19	49
	Number of best results	–	–	**13**	7	0	1	0	1	0	0	0	2	0	0	0	0
	$\sum \rho_b$	–	–	2.6	**1.8**	12.8	38.7	28.7	39.3	10.0	25.9	26.0	10.1	24.8	45.3	40.1	63.4
	Ranking	–	–	2	**1**	5	10	9	11	3	7	8	4	6	13	12	14

Table 14 continued

Result	Instance	n	profile$_0$	VNS-band (500s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
Profile	ARC130	130	9624	7718	6085	**5595**	9624	8987	9175	6569	9018	9013	6657	8635	9624	12,947	9873
	CURTIS54	54	715	995	672	710	715	611	978	662	618	**593**	701	734	715	1019	1829
	FS_183_1	183	23,064	**10,079**	11,445	13,791	23,064	17,299	17,259	17,047	17,172	17,436	14,677	14,530	23,064	15,200	18,540
	GENT113	113	4419	2820[a]	3230	3014	4419	2776	3289	2917	2728	2753	2846	**2267**	4419	4830	6867
	GRE_115	115	5960	2981	3066	3472	5960	2827	4211	3350	2943	3203	3344	**2721**	5960	3286	7387
	GRE_185	185	9698	5541	5656	5732	9698	5591	7039	5700	5515	5515	5560	**5501**	9698	9414	23,366
	IBM32	32	599	469	400	429	599	391	450	**386**	404	396	488	419	599	452	574
	IMPCOL_B	59	**1192**	1259	1232	1555	**1192**	1367	1618	1349	1359	1207	1683	1361	**1192**	1415	2146
	IMPCOL_C	137	3975	**3921**	4486	4965	3975	4287	6217	4684	4346	4197	5132	4094	3975	5364	9967
	LNS_131	131	6090	2894	3107	3015	6090	3122	3375	2704	2891	3168	**2473**	2522	6090	4798	6997
	MCCA	180	4015	6535	6008	7941	**4015**	6891	6963	5684	6607	6607	5817	4618	**4015**	13,836	23,331
	PORES1	30	434	**276**	311	308	434	314	359	353	310	310	330	300	434	347	639
	STEAM3	80	3440	1216	**848**	852	3440	860	**848**	860	860	860	**848**	856	3440	864	5336
	WEST0132	132	5329	**4129**	4340	4769	5329	5081	7884	4504	4736	4350	5446	4390	5329	4786	9837
	WEST0156	156	12,197	5939	5888	7324	12,197	5984	7704	6449	6269	6505	8161	**5594**	12,197	6823	13,165
	WEST0167	167	8390	5597	5893	8599	8390	5757	10,080	8351	5531	**5350**	7526	5838	8390	8642	14,847
	WILL199	57	25,398	**14,801**	15,528	17,591	25,398	17,808	18,990	18,013	16,816	15,780	16,287	17,211	25,398	18,060	23,957
	WILL57	199	714	639	432	443	714	470	612	590	474	485	437	**379**	714	545	2139
	Number of best results	–	–	**5**	1	1	2	0	1	1	0	2	2	**5**	2	0	0
	$\sum \rho_p$	–	–	3.70	2.42	4.64	14.41	3.84	8.86	4.24	3.58	3.47	4.39	**2.08**	14.41	10.63	36.53
	Ranking	–	–	5	2	9	12	6	10	7	4	3	8	**1**	12	11	14

[a] Values obtained using the VNS-band (6 s) heuristic, which was the time limit for the heuristic to find a valid solution

Table 15 Average execution times, in seconds, of 14 heuristics applied to 18 asymmetric instances of the Harwell-Boeing sparse-matrix collection

Instance	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
ARC130	500.36700	0.04518	0.00329	**0.00013**	0.00025	0.00014	0.00228	0.00039	0.00038	0.00085	0.00030	0.01619	0.00062	0.27576
CURTIS54	512.93500	0.02045	0.00026	0.00007	0.00006	**0.00004**	0.00023	0.00007	0.00007	0.00016	0.00007	0.00323	0.00009	0.10114
FS_183_1	500.17900	0.07077	0.00349	0.00013	0.00032	**0.00011**	0.00318	0.00069	0.00068	0.00079	0.00035	0.33894	0.00084	0.42969
GENT113	500.55500	0.18503	0.00066	0.00011	0.00022	**0.00007**	0.00192	0.00026	0.00027	0.00061	0.00025	0.04220	0.00040	0.31037
GRE_115	500.53800	0.05219	0.00056	0.00013	0.00013	**0.00008**	0.00049	0.00020	0.00022	0.00041	0.00016	0.05500	0.00024	0.22651
GRE_185	500.99500	0.09578	0.00123	0.00016	0.00028	**0.00014**	0.00221	0.00039	0.00045	0.00117	0.00045	0.02162	0.00052	0.43835
IBM32	554.71500	0.01514	0.00012	0.00003	0.00003	**0.00002**	0.00012	0.00003	0.00004	0.00010	0.00006	0.00306	0.00006	0.08609
IMPCOL_B	552.06900	0.02528	0.00042	0.00005	0.00009	**0.00004**	0.00032	0.00011	0.00012	0.00024	0.00012	0.01656	0.00014	0.09577
IMPCOL_C	500.36000	0.06834	0.00110	0.00013	0.00016	**0.00006**	0.00076	0.00027	0.00026	0.00075	0.00025	0.00913	0.00037	0.40642
LNS_131	500.51500	0.28147	0.00096	0.00007	0.00017	**0.00005**	0.00056	0.00021	0.00021	0.00038	0.00020	0.06512	0.00030	0.33030
MCCA	500.27000	0.11936	0.00360	**0.00021**	0.00066	0.00039	0.01136	0.00076	0.00074	0.00248	0.00077	0.08900	0.00138	0.44040
PORES1	517.13500	0.01390	0.00012	**0.00003**	**0.00003**	**0.00003**	0.00020	0.00003	0.00003	0.00015	0.01656	0.00151	0.00005	0.07414
STEAM3	716.82600	0.03620	0.00024	0.00020	**0.00012**	0.00015	0.00310	0.00013	0.00026	0.00059	0.00026	0.00308	0.00032	0.12844
WEST0132	500.39000	0.05658	0.00104	**0.00009**	0.00016	0.00010	0.00092	0.00026	0.00029	0.00078	0.00025	0.02160	0.00034	0.31351
WEST0156	500.39500	0.08212	0.00132	0.00012	0.00020	**0.00008**	0.00084	0.00038	0.00036	0.00055	0.00025	0.01444	0.00042	0.28191
WEST0167	500.31900	0.07043	0.00087	0.00013	0.00023	**0.00012**	0.00138	0.00040	0.00039	0.00087	0.00028	0.01572	0.00050	0.29481
WILL57	536.42800	0.01585	0.00036	0.00007	0.00007	**0.00005**	0.00057	0.00007	0.00007	0.00026	0.00039	0.00515	0.00009	0.10329
WILL199	500.21500	0.14719	0.00278	0.00016	0.00029	**0.00011**	0.00154	0.00073	0.00080	0.00104	0.00882	0.20027	0.00074	0.26639
Number of best results	0	0	0	4	2	**14**	0	0	0	0	0	0	0	0
$\sum \rho_t$	154,759,581	18,842	216	7	20	**1**	265	40	43	112	666	9709	60	58,570
Ranking	14	12	8	2	3	**1**	9	4	5	7	10	11	6	13
Magnitude	3	−2	−4	−4	−4	−5	−4	−4	−4	−4	−4	−2	−4	−1

Table 16 Results of 14 heuristics applied to reduce bandwidth of 45 asymmetric instances of the Harwell-Boeing sparse-matrix collection

Instance	n	β_0	VNS-band (11 s)	VNS-band (12 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
BP_0	822	820	**250**	**250**	259	541	815	761	819	439	764	785	422	739	820	815	789
BP_1000	822	820	**271**	**271**	315	679	820	774	818	489	750	784	603	668	820	812	799
BP_1200	822	820	**279**	**279**	317	630	815	774	820	491	780	791	554	698	820	812	795
BP_1400	822	820	**286**	**286**	324	681	816	776	816	475	756	811	636	707	820	820	796
BP_1600	822	820	**299**	**299**	322	673	813	784	815	476	788	802	609	805	820	811	799
BP_200	822	820	371	371	**285**	609	811	794	814	483	757	792	508	721	820	789	796
BP_400	822	820	356	297	**296**	620	806	773	814	530	778	818	530	668	820	819	795
BP_600	822	820	358	358	**304**	630	811	766	813	523	753	806	541	803	820	819	798
BP_800	822	820	**301**	**301**	309	600	817	801	818	523	800	776.1	523	694	820	811	800
FS_541_1	541	540	**270**	**270**	**270**	354	540	540	540	529	540	540	529	530	540	539	518
FS_680_1	680	600	25	25	**17**	29	30	25	404	20	33	24	25	25	600	38	629
FS_760_1	760	740	52	**38**	**38**	71	72	69	175	40	51	67	62	110	740	670	731
GRE_216A	216	36	28	28	**21**	33	41	33	36	**21**	35	35	**21**	33	36	67	203
GRE_343	343	49	57	57	**28**	40	53	48	49	**28**	47	48	**28**	43	49	99	327
GRE_512	512	64	**21**	**21**	36	61	66	60	64	36	63	62	36	54	64	140	490
HOR_131	434	421	**55**	**55**	60	121	101	218	93	73	223	244	104	227	421	425	417
IMPCOL_A	207	167	**33**	**33**	53	66	127	159	165	48	111	144	58	118	167	144	194
IMPCOL_D	425	406	42	42	**41**	81	418	143	416	61	137	118	85	104	406	249	404
IMPCOL_E	225	92	**42**	**42**	43	109	219	138	190	75	147	126	65	146	92	185	214
JPWH_991	991	197	232	232	163	192	921	433	928	**152**	367	397	**152**	403	197	864	950
LNSP_511	511	57	**46**	**46**	57	97	447	119	450	55	101	138	55	150	57	472	486

Table 16 continued

Instance	n	β_0	VNS-band (11 s)	VNS-band (12 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
MBEACXC	496	490	**316**	**316**	427	465	484	478	484	481	484	482	481	462	490	486	491
MBEAFLW	496	490	**316**	**316**	426	465	484	478	484	481	484	482	481	462	490	486	492
MBEAUSE	496	490	**285**	**285**	429	419	490	488	490	480	488	489	482	463	490	490	490
MCFE	765	187	**126**	**126**	133	258	247	697	734	179	696	696	178	190	187	244	754
NNC261	261	64	**24**	**24**	26	49	37	66	40	36	71	165	36	57	64	177	246
NNC666	666	262	52	52	**46**	88	69	211	67	63	156	309	66	158	262	283	642
ORSIRR_2	886	554	**95**	**95**	99	206	157	533	160	111	429	740	132	573	554	613	854
PORES_3	532	77	23	**13**	14	24	16	36	14	15	33	32	**13**	35	77	144	510
SAYLR1	238	**14**	**14**	**14**	**14**	15	15	27	15	15	22	22	**14**	23	**14**	16	224
SAYLR3	1000	100	56	56	57	104	57	193	59	**55**	152	214	57	164	100	460	941
SHERMAN4	1104	368	36[a]	36[a]	31	46	29	52	41	28	52	53	**27**	34	368	64	1027
SHL_0	663	661	**232**	**232**	252	374	658	633	660	412	623	630	425	652	661	660	623
SHL_200	663	661	**239**	**239**	256	360	656	649	661	426	648	649	437	642	661	658	628
SHL_400	663	662	**234**	**234**	250	385	658	631	660	425	637	635	465	639	662	660	635
STEAM1	240	146	**44**	**44**	46	63	50	86	50	50	93	137	50	146	146	151	231
STEAM2	600	331	**63**	**63**	**63**	67	**63**	111	**63**	**63**	85	91	**63**	127	331	107	588
STR_0	363	359	**122**	**122**	127	256	346	333	357	173	329	345	192	301	359	360	351
STR_200	363	359	**129**	**129**	137	267	815	351	348	199	345	350	233	326	820	352	351
STR_600	363	359	**138**	**138**	142	288	352	354	351	221	343	341	262	328	359	354	352

Table 16 continued

Instance	n	β_0	VNS-band (11 s)	VNS-band (12 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
WEST0381	381	363	**153**	**153**	162	295	372	344	370	269	360	351	328	294	363	376	367
WEST0479	479	388	**125**	**125**	133	313	473	401	494	237	400	440	304	387	388	400	459
WEST0497	497	416	**88**	**88**	96	221	494	305	465	269	282	315	189	299	416	348	476
WEST0655	655	564	**169**	**169**	173	408	649	512	561	257	505	541	382	470	564	564	630
WEST0989	989	855	379	**236**	237	560	841	815	903	323	815	860	403	841	855	832	953
Number of best results	–	–	31	**34**	12	0	1	0	1	5	0	0	7	0	1	0	0
$\sum \rho_b$	–	–	5.3	**3.4**	4.6	42.1	87.6	80.6	112.6	22.8	74.4	90.5	28.8	71.6	160.5	142.4	332.6
Ranking	–	–	3	**1**	2	6	10	9	12	4	8	11	5	7	14	13	15

[a] Bandwidth obtained using the 32-bit executable program

Table 17 Results of 14 heuristics applied to reduce profile of 45 asymmetric instances of the Harwell-Boeing sparse-matrix collection

Instance	n	β_0	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
BP_0	822	313,542	199,627	215,330	249,988	235,809	221,904	232,492	248,221	216,203	218,189	272,394	**199,248**	313,542	239,321	367,471
BP_1000	822	411,136	**252,634**	271,438	310,747	301,295	281,488	301,913	313,475	274,726	268,901	312,766	260,666	411,136	302,412	426,792
BP_1200	822	419,750	**255,182**	268,572	311,502	303,672	291,796	302,688	307,522	272,977	278,526	314,490	274,486	419,750	322,898	430,840
BP_1400	822	416,507	**263,095**	279,842	313,794	309,504	282,287	303,342	316,360	274,946	290,421	322,162	267,449	416,507	325,486	433,218
BP_1600	822	418,320	**257,411**	276,168	317,037	308,094	300,270	301,596	322,252	285,558	292,966	307,868	264,877	418,320	321,015	435,456
BP_200	822	356,678	223,858	242,323	280,548	276,640	269,264	276,452	273,874	252,176	246,959	290,703	**220,987**	356,678	281,421	402,470
BP_400	822	372,881	**230,051**	250,851	283,621	277,420	252,919	287,618	303,001	251,589	261,331	335,155	245,956	372,881	291,338	405,921
BP_600	822	390,312	**239,290**	255,028	295,578	293,314	271,259	291,410	337,004	267,101	264,875	336,451	254,504	390,312	295,740	410,346
BP_800	822	400,827	**249,980**	264,568	291,020	284,372	273,252	311,978	290,406	275,208	270,866	311,143	262,635	400,827	300,408	418,244
FS_541_1	541	231,254	157,668	158,608	164,501	161,360	163,110	157,999	160,727	**156,260**	163,874	220,229	179,121	231,254	206,669	222,494
FS_680_1	680	230,957	13,842	13,727	16,084	157,707	14,347	154,209	152,96	13,738	13,597	16,817	**10,263**	230,957	11,570	223,733
FS_760_1	760	419,828	45,288	44,868	49,130	59,194	39,503	56,918	47,978	**38,527**	39,157	60,737	39,187	419,828	276,529	419,364
GRE_216A	216	7130	6369	6378	6332	7353	6315	7388	**6252**	6303	6318	**6252**	6281	7130	7297	28,293
GRE_343	343	15,608	14,010	13,932	13,681	16,031	13,631	16,104	**13,494**	13,524	13,500	**13,494**	13,544	15,608	16,097	73,448
GRE_512	512	30,715	27,793	27,490	26,672	31,763	26,542	31,586	**26,250**	26,388	26,429	**26,250**	26,357	30,715	31,870	161,253
HOR_131	434	108,080	37,611	38,024	39,140	41,452	33,840	46,568	44,934	32,308	32,246	42,731	**31,842**	108,080	74,916	155,563
IMPCOL_A	207	**5291**	7023	8982	8356	5851	6821	9706	8132	6051	6636	7301	6304	**5291**	7719	22,582
IMPCOL_D	425	19,078	18,655	19,363	21,928	18,868	16,752	21,433	20,916	16,086	**15,957**	21,980	17,084	19,078	18,780	108,628
IMPCOL_E	225	8971	8774	**8638**	10,510	13,970	14,879	11,244	8757	14,696	9507	8870	9711	8971	19,315	32,312
JPWH_991	991	139,359	**130,588**	178,925	158,827	255,225	139,019	247,683	172,468	135,791	131,928	153,998	135,695	139,359	252,702	617,513
LNSP_511	511	33,577	26,425	27,493	30,979	28,029	27,631	49,094	27,097	28,095	27,671	26,776	**26,180**	33,577	64,737	140,758
MBEACXC	496	173,863	**153,407**	183,140	167,115	181,029	179,594	164,128	184,889	176,385	176,463	185,763	172,636	173,863	192,989	201,549

Table 17 continued

Instance	n	β_0	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
MBEAFLW	496	173,863	**153,407**	183,759	167,115	171,448	179,594	172,341	184,889	176,385	176,463	185,763	172,636	173,863	192,989	203,572
MBEAUSE	496	161,473	**141,037**	169,708	161,272	153,244	169,902	165,406	177,858	166,395	166,186	176,638	159,595	161,473	183,911	184,658
MCFE	765	78,957	118,093	115,725	137,029	89,225	137,268	160,071	116,427	153,285	154,012	115,859	**76,328**	78,957	95,606	534,213
NNC261	261	9383	8501	9040	10,629	10,719	7879	11,034	9558	7543	8523	11,068	**7044**	9383	18,630	46,056
NNC666	666	37,389	38,208	41,677	46,137	46,248	37,268	48,829	45,891	34,931	53,719	47,857	**31,615**	37,389	70,067	300,427
ORSIRR_2	886	211,572	125,336	131,915	137,841	144,929	99,816	156,553	135,201	99,034	117,165	149,219	**91,441**	211,572	199,830	577,745
PORES_3	532	55,932	10,610	11,104	11,202	10,995	10,652	10,690	10,737	10,454	10,467	10,777	**10,253**	55,932	23,112	200,447
SAYLR1	238	6298	5957	5520	5010	5139	5011	**4998**	**4998**	5010	5010	**4998**	5010	6298	5240	36,737
SAYLR3	1000	69,490	52,018	53,004	57,663	52,575	48,714	52,843	52,213	48,869	49,730	53,150	**47,944**	69,490	120,108	441,649
SHERMAN4	1104	269,743	33,267	22,295	21,203	34,316	20,994	33,748	22,049	20,845	20,781	21,181	**20,709**	269,743	27,752	419,220
SHL_0	663	148,668	109,089	127,217	117,021	96,751	71,616	108,431	139,926	73,825	74,453	141,750	**68,143**	148,668	81,619	195,984
SHL_200	663	153,482	110,244	128,214	117,823	103,117	74,555	112,165	136,033	74,754	73,213	165,441	**72,867**	153,482	83,618	196,907
SHL_400	663	159,588	115,808	129,004	127,843	114,794	90,498	109,703	148,933	77,981	83,147	159,558	**72,895**	159,588	95,458	201,811
STEAM1	240	33,051	16,539	16,212	15,864	16,257	15,891	**15,060**	15,084	15,996	15,585	**15,060**	15,798	33,051	23,268	50,325
STEAM2	600	196,168	64,328	63,685	62,312	**61,704**	62,672	69,416	**61,704**	62,292	62,268	**61,704**	62,916	196,168	73,848	328,472
STR_0	363	67,460	**45,874**	47,301	53,681	47,374	58,130	47,623	52,381	57,260	55,860	49,611	48,400	67,460	50,388	80,962
STR_200	363	80,640	**48,341**	50,375	59,162	59,215	61,814	58,220	59,332	61,161	62,840	58,039	56,639	80,640	66,951	90,055
STR_600	363	87,176	**50,756**	55,059	61,994	63,831	58,755	64,123	65,187	58,323	63,830	58,628	59,949	87,176	71,414	90,508
WEST0381	381	**70,091**	70,322	73,202	82,995	83,604	86,467	86,994	84,393	90,096	82,148	81,202	78,202	**70,091**	84,462	97,921
WEST0479	479	84,283	**60,163**	63,125	81,677	98,934	62,424	95,668	88,014	62,643	70,385	88,591	60,410	84,283	71,039	138,152
WEST0497	497	69,403	41,807	42,426	46,587	92,078	50,145	90,686	47,002	45,649	44,686	51,786	**40,851**	69,403	51,592	129,716

Table 17 continued

Instance	n	β_0	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
WEST0655	655	159,257	112,843	112,860	144,307	135,904	123,381	139,047	159,705	125,255	122,698	165,510	**110,823**	159,257	147,729	258,930
WEST0989	989	250,490	**220,848**	233,135	296,197	376,481	261,928	375,258	285,486	253,005	241,153	330,835	252,833	250,490	308,086	573,230
Number of best results	–	–	16	1	0	1	0	2	5	2	1	6	**17**	2	0	0
$\sum \rho_p$	–	–	5.5	8.2	11.8	27.5	6.6	30.0	12.5	5.6	6.2	14.6	**2.1**	70.7	26.9	174.4
Ranking	–	–	2	6	7	11	5	12	8	3	4	9	**1**	13	10	14

Profile obtained using the 32-bit executable program

Table 18 Average execution times, in seconds, of 14 heuristics applied to 45 asymmetric instances of the Harwell-Boeing sparse-matrix collection

Instance	VNS-band (11 s)	VNS-band (12 s)	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
BP_0	11.01300	12.00800	500.07810	0.72333	0.04619	0.00059	0.00360	**0.00046**	0.01394	0.01333	0.01342	0.00805	0.00326	10.97370	0.01185	5.04834
BP_1000	11.02020	12.01230	500.06160	0.77538	0.05913	**0.00060**	0.00399	0.00070	0.01540	0.01385	0.01474	0.01258	0.00449	26.14210	0.01250	4.85287
BP_1200	11.01280	12.02590	500.05900	0.76745	0.07177	0.00063	0.00414	**0.00057**	0.01399	0.01447	0.01469	0.01019	0.00446	30.03660	0.01303	5.16874
BP_1400	11.00950	12.00980	500.06800	0.74581	0.07460	**0.00073**	0.00393	0.00082	0.01145	0.01433	0.01493	0.01029	0.00474	30.64940	0.01298	4.59449
BP_1600	11.01170	12.01150	500.06200	0.77311	0.05441	**0.00078**	0.00432	0.00079	0.01327	0.01485	0.01486	0.01045	0.00452	30.43420	0.01326	4.89230
BP_200	11.01290	12.01220	500.06600	0.80888	0.09430	0.00090	0.00563	**0.00083**	0.01566	0.01622	0.01428	0.01014	0.00380	32.27750	0.01342	4.93832
BP_400	11.01060	12.01060	500.05650	0.80673	0.05735	**0.00076**	0.00426	0.00077	0.01334	0.01476	0.01429	0.00934	0.00408	28.71400	0.01372	5.21761
BP_600	11.01250	12.01620	500.05740	0.82667	0.09985	**0.00081**	0.00443	**0.00081**	0.01141	0.01480	0.01440	0.00941	0.00432	28.08100	0.01381	4.98693
BP_800	11.01750	12.01680	500.04850	0.80806	0.09464	0.00081	0.00432	**0.00068**	0.01174	0.01453	0.01453	0.01065	0.00423	27.24180	0.01364	5.08025
FS_541_1	11.01180	12.01140	500.09950	0.20655	0.04804	**0.00040**	0.00220	0.00051	0.03533	0.00576	0.00592	0.01954	0.00206	0.41081	0.00680	2.16609
FS_680_1	11.01110	12.01170	500.12270	0.17512	0.00405	**0.00027**	0.00063	0.00037	0.00662	0.00081	0.00077	0.00426	0.00076	0.12872	0.00111	3.07982
FS_760_1	11.01540	12.01160	500.11790	0.40112	0.00697	**0.00064**	0.00207	0.00070	0.00779	0.00308	0.00317	0.00830	0.00250	0.36369	0.00853	3.92669
GRE_216A	11.02480	12.00490	500.78930	0.08698	0.00074	**0.00014**	0.00048	0.00022	0.00386	0.00076	0.00037	0.00175	0.00033	0.03562	0.00102	0.47035
GRE_343	11.02450	12.01620	500.45700	0.15027	0.00092	**0.00022**	0.00082	0.00028	0.00712	0.00141	0.00074	0.00332	0.00058	0.17124	0.00181	0.95727
GRE_512	11.01390	12.03380	500.22160	0.23382	0.00196	0.00033	**0.00027**	0.00035	0.01413	0.00038	0.00139	0.00638	0.00095	0.30448	0.00053	1.97609
HOR_131	11.01890	12.01690	500.18760	0.33836	0.00845	**0.00054**	0.00140	0.00070	0.00390	0.00221	0.00215	0.00372	0.00156	1.38752	0.00310	1.51304
IMPCOL_A	11.01100	12.00590	500.26420	0.07314	0.00168	**0.00014**	0.00029	**0.00014**	0.00130	0.00048	0.00057	0.00105	0.00045	0.02356	0.00064	0.51433
IMPCOL_D	11.01810	12.01400	500.13360	0.16637	0.00452	0.00040	0.00070	**0.00019**	0.00420	0.00126	0.00121	0.00362	0.00080	0.13316	0.00155	1.38153
IMPCOL_E	11.01040	12.01080	500.22410	0.09095	0.00356	0.00016	0.00045	**0.00014**	0.00551	0.00075	0.00068	0.00174	0.00056	0.33301	0.00102	0.60196
JPWH_991	11.01660	12.01880	500.07920	0.25023	0.01624	0.00116	0.00322	**0.00074**	0.01212	0.00737	0.00728	0.01028	0.00328	0.81226	0.00895	7.26773
LNSP_511	11.01650	12.02120	500.14660	0.24249	0.00394	0.00031	0.00094	**0.00028**	0.00448	0.00147	0.00146	0.00425	0.00110	0.67530	0.00282	1.90546

Table 18 continued

Instance	VNS-band (11 s)	VNS-band (12 s)	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
MBEACXC	11.06630	12.06590	500.09600	2.71791	1.48127	**0.00525**	0.04002	0.00668	0.22174	0.04309	0.04027	0.10215	0.04373	132.09500	0.15521	2.34597
MBEAFLW	11.06070	12.06120	500.08630	2.71398	1.45529	**0.00528**	0.04139	0.00679	0.22022	0.04372	0.04303	0.10133	0.05138	132.72100	0.16811	2.31394
MBEAUSE	11.04680	12.05300	500.08960	0.23382	**0.00196**	0.00379	0.03059	0.00555	0.01413	0.03579	0.03694	0.00638	0.03820	119.40900	0.12306	1.97609
NNC261	11.03070	12.02740	500.11570	0.10887	0.00218	**0.00027**	0.01222	0.00042	0.00428	0.01392	0.01426	0.00229	0.01244	0.06117	0.02338	0.66376
MCFE	11.01060	12.00890	500.25650	1.03463	0.07967	0.00191	**0.00039**	0.00282	0.16950	0.00048	0.00054	0.03807	0.00053	5.17640	0.00085	4.85498
NNC666	11.01240	12.00760	500.12100	0.29576	0.00780	0.00056	0.00151	**0.00049**	0.01064	0.00205	0.00289	0.00565	0.00177	1.60060	0.00328	2.93249
ORSIRR_2	11.00970	12.02030	500.10410	0.57990	0.01281	0.00113	0.00282	**0.00082**	0.01107	0.00647	0.00812	0.00769	0.00293	0.50638	0.00778	5.38558
PORES_3	11.01130	12.01900	500.23900	0.17307	0.00216	**0.00034**	0.00107	0.00048	0.00432	0.00115	0.00107	0.00650	0.00125	0.11005	0.00176	2.06139
SAYLR1	11.00580	12.02200	500.57470	0.08067	0.00039	0.00015	0.00025	**0.00013**	0.00079	0.00033	0.00032	0.00107	0.00029	0.00306	0.00039	0.59436
SAYLR3	11.01720	12.02080	500.09790	2.13638	0.00593	**0.00051**	0.00264	0.00060	0.00735	0.00402	0.00397	0.00800	0.00284	0.14012	0.00522	7.99298
SHERMAN4	11.00930	12.02110	500.18840	0.32052	0.00319	0.00072	0.00348	**0.00046**	0.00524	0.00418	0.00379	0.00800	0.00359	0.50591	0.00419	10.39652
SHL_0	11.00810	12.01180	500.08500	0.19918	0.01906	0.00038	0.00182	**0.00032**	0.01276	0.00791	0.00819	0.00393	0.00162	9.62097	0.00706	2.88522
SHL_200	11.00800	12.00840	500.08820	0.19452	0.02265	0.00039	0.00189	**0.00032**	0.01039	0.00847	0.00850	0.00323	0.00176	10.85220	0.00723	3.01915
SHL_400	11.00650	12.00640	500.07010	0.21131	0.02254	0.00038	0.00186	**0.00030**	0.01037	0.00822	0.00823	0.00398	0.00166	9.79234	0.00721	2.79648
STEAM1	11.01200	12.01310	500.48220	0.22305	0.00265	**0.00047**	0.00074	0.00053	0.00242	0.00082	0.00077	0.00292	0.00094	0.24201	0.00133	0.58608
STEAM2	11.03370	12.03460	500.16450	0.70515	0.00928	**0.00108**	0.00400	0.00133	0.01245	0.00426	0.00418	0.01826	0.00499	3.95201	0.00673	3.31327
STR_0	11.01310	12.01280	500.09830	0.23600	0.00940	0.00029	0.00138	**0.00028**	0.00546	0.00295	0.00279	0.00489	0.00141	0.69059	0.00317	1.13044
STR_200	11.00870	12.00890	500.09140	0.30776	0.01661	**0.00043**	0.00143	0.00069	0.00612	0.00304	0.00303	0.00434	0.00175	2.03940	0.00372	1.04132
STR_600	11.00940	12.01300	500.10250	0.32812	0.02201	**0.00046**	0.00148	0.00049	0.00940	0.00309	0.00335	0.00581	0.00201	2.27952	0.00383	1.15622
WEST0381	11.00720	12.00740	500.10780	0.41265	0.01805	**0.00032**	0.00108	0.00039	0.00452	0.00318	0.00294	0.00753	0.00153	1.04006	0.00312	0.93019
WEST0479	11.00870	12.01280	500.08460	0.33187	0.01417	**0.00035**	0.00109	0.00037	0.00793	0.00323	0.00357	0.00384	0.00125	1.76949	0.00339	1.56818

Table 18 continued

Instance	VNS-band (11 s)	VNS-band (12 s)	VNS-band (500 s)	FNCHC	BL	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG	WBRA	Snay	CSS-band
WEST0497	11.00560	12.01070	500.11030	0.21657	0.00985	0.00034	0.00102	**0.00031**	0.00646	0.00250	0.00239	0.00376	0.00116	0.48501	0.00313	1.49981
WEST0655	11.00750	12.02230	500.08120	0.56168	0.02734	**0.00049**	0.00248	0.00071	0.01090	0.00716	0.00724	0.00534	0.00276	0.40966	0.00655	2.79211
WEST0989	11.01630	12.01540	500.05150	0.84681	0.05563	0.00103	0.00374	**0.00076**	0.01385	0.01434	0.01474	0.00960	0.00333	12.74550	0.01340	7.45910
Number of best results	0	0	0	0	1	**24**	2	20	0	0	0	0	0	0	0	0
$\sum \rho_t$	1,337,324	1,458,855	60,740,694	40,330	2607	**10**	206	14	1381	512	514	693	224	636,792	654	286,837
Ranking	14	15	16	11	10	**1**	3	2	9	5	6	8	4	13	7	12
Magnitude	2	2	3	−1	−2	−4	−3	−4	−2	−3	−3	−3	−3	−1	−3	1

Table 19 Results of 10 heuristics applied to reduce bandwidth and profile of 17 symmetric instances of the University of Florida sparse-matrix collection

Result	Instance	n	β_0	VNS-band (500 s)	FNCHC	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG
Bandwidth	BARTH	6691	6523	**155**	172	195	660	197	186	546	985	185	874
	BARTH4	6019	5305	1147	175	4250	703	4927	**171**	780	1840	216	1372
	BARTH5	15,606	15,080	5197	352	362	1234	354	**350**	1079	5770	374	3000
	BCSSTK30	28,924	16,947	2203	**1684**	2514	15,774	2512	2510	15,229	17,733	2512	13,132
	COMMANCHE_DUAL	7920	7493	**94**	110	134	1528	155	130	1516	1985	134	1458
	COPTER1	17,222	16,864	4940	**783**	917	2712	915	915	2617	4770	965	2661
	COPTER2	55,476	55,279	5637	**2065**	2106	9590	2304	2138	9690	25,271	2705	8915
	FINANCE256	37,376	37,348	2025	**1968**	2076	8915	2053	2026	8884	8964	2313	8659
	FORD1	18,728	18,704	3458	248	304	2147	273	**247**	1986	3184	293	2227
	FORD2	100,196	96,272	20,749	**902**	980	6794	982	936	6525	10,470	983	8659
	NASASRB	54,870	893	3992	712	**585**	4918	586	589	4902	5134	739	4632
	ONERA_DUAL	85,567	78,363	27,165	2729	3441	21,070	3986	**2714**	16,901	47,506	2947	26,240
	PDS10	16,558	15,451	5341	**2367**	2930	13,743	3678	2502	13,848	16,012	3902	16,328
	SHUTTLE_EDDY	10,429	9597	**159**	169	176	805	176	176	809	840	176	767
	SKIRT	12,598	1046	**55**	316	317	2056	314	309	1861	1759	314	1176
	TANDEM_DUAL	94,069	70,502	26,476	**2111**	2197	8695	2149	2120	7173	16,719	2280	11,581
	TANDEM_VTX	18,454	18,411	5843	**1327**	1508	4235	1587	1427	4048	5863	1912	4827
	Number of best results	–	–	4	**8**	1	0	0	4	0	0	0	0
	$\sum \rho_b$	–	–	92.97	**5.34**	31.18	121.11	35.81	6.21	112.99	178.28	8.77	119.09
	Ranking	–	–	6	**1**	4	9	5	2	7	10	3	8

Table 19 continued

Result	Instance	n	profile$_0$	VNS-band (500 s)	FNCHC	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG
Profile	BARTH	6691	15,855,195	730,380	836,489	822,032	471,415	686,506	686,681	469,925	456,304	653,727	**444,424**
	BARTH4	6019	3,315,889	3,084,910	979,795	1,723,759	674,980	1,549,659	891,330	653,813	**414,700**	947,143	419,139
	BARTH5	15,606	4,058,103	10,076,311	3,813,333	3,437,767	2,315,620	3,289,486	3,268,534	2,320,292	**1,338,429**	3,339,755	1,375,684
	BCSSTK30	28,924	15,686,968	43,230,410	36,577,745	41,565,277	10,655,704	22,427,151	28,670,391	10,080,145	10,080,145	41,007,195	**8,897,060**
	COMMANCHE_DUAL	7920	16,543,294	522,659	570,784	570,687	403,904	590,636	529,624	397,955	**320,369**	529,607	371,290
	COPTER1	17,222	18,983,170	35,170,713	9,877,267	10,529,454	7,128,590	8,345,238	8,423,654	7,193,200	**5,882,278**	8,495,921	7,061,783
	COPTER2	55,476	1,084,047,722	151,710,188	87,476,525	85,279,578	43,783,488	69,340,196	69,707,456	46,722,171	**38,741,576**	74,416,281	40,940,860
	FINANCE256	37,376	241,399,968	52,283,288	48,500,421	48,151,566	**6,183,693**	33,641,884	35,306,008	6,319,706	6,319,706	39,514,304	7,056,838
	FORD1	18,728	35,197,733	7,723,399	3,326,349	3,516,591	2,341,828	2,861,301	2,846,891	2,329,436	2,318,508	3,034,119	**2,173,629**
	FORD2	100,196	372,184,287	183,252,463	65,656,181	61,611,796	40,458,534	55,222,494	55,690,206	39,916,921	39,209,314	56,762,866	**35,804,553**
	NASASRB	54,870	20,311,330	44,771,390	20,713,854	19,701,681	18,486,625	19,448,635	19,448,680	18,559,413	**17,724,407**	19,270,542	18,819,718
	ONERA_DUAL	85,567	709,077,813	553,398,262	177,554,480	191,011,865	114,445,935	189,351,468	145,256,651	112,617,638	**78,573,231**	157,984,330	97,521,955
	PDS10	16,558	18,035,808	46,804,896	29,129,946	30,448,184	13,474,213	30,686,304	21,966,264	10,967,390	**9,433,197**	31,521,682	9,689,476
	SHUTTLE_EDDY	10,429	11,656,811	894,056	834,502	813,152	583,595	708,216	708,232	579,306	**573,871**	708,216	574,981
	SKIRT	12,598	3,661,041	**95,044**	1,330,382	1,245,815	726,891	1,077,159	1,031,686	699,293	720,644	1,080,639	704,518
	TANDEM_DUAL	94,069	487,466,755	276,139,384	144,129,745	132,980,894	90,000,767	125,423,530	115,371,267	87,639,361	**63,587,744**	119,020,989	74,063,586
	TANDEM_VTX	18,454	81,008,429	48,357,187	18,176,114	18,993,998	6,300,173	15,415,989	12,699,112	5,983,094	5,406,302	14,416,158	**5,257,906**
	Number of best results	–	–	1	0	0	1	0	0	0	**10**	0	5
	$\sum \rho_p$	–	–	63.75	38.82	40.14	10.63	30.84	27.41	9.91	**6.95**	31.91	7.51
	Ranking	–	–	10	8	9	4	6	5	3	**1**	7	2

Table 20 Average execution times, in seconds, of 10 heuristics applied to 17 symmetric instances of the University of Florida sparse-matrix collection

Instance	VNS-band (500 s)	FNCHC	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG
BARTH	500.144	4.491	0.013	0.014	**0.009**	0.710	0.060	0.061	0.363	0.016
BARTH4	500.156	4.202	0.007	0.060	**0.005**	0.881	0.095	0.086	0.383	0.058
BARTH5	500.344	13.995	0.026	0.047	**0.020**	4.289	0.262	0.160	1.582	0.026
BCSSTK30	501.604	65.837	**0.271**	0.630	0.310	58.617	0.491	0.464	35.345	1.021
COMMANCHE_DUAL	500.177	3.925	0.010	0.010	**0.007**	9.078	0.039	0.036	0.345	0.012
COPTER1	500.428	24.196	0.045	0.174	**0.037**	3.906	1.225	1.298	5.090	0.191
COPTER2	503.331	99.414	0.191	1.692	**0.182**	36.964	9.950	9.490	30.209	0.515
FINANCE256	501.383	47.697	0.078	0.206	**0.065**	23.168	0.832	0.827	8.983	0.130
FORD1	500.387	12.408	0.026	0.049	**0.021**	14.777	0.268	0.308	1.854	0.037
FORD2	507.843	103.294	0.196	1.095	**0.177**	120.601	5.723	6.461	55.416	0.406
NASASRB	504.800	133.482	**0.365**	0.429	0.394	1515.449	0.781	0.743	514.690	0.435
ONERA_DUAL	505.970	132.203	0.158	4.920	**0.125**	94.083	22.530	18.166	38.393	1.163
PDS10	500.358	26.839	0.030	0.662	**0.027**	4.384	3.789	4.592	3.065	0.118
SHUTTLE_EDDY	500.185	8.054	0.017	**0.014**	0.015	9.189	0.053	0.052	3.398	0.019
SKIRT	500.325	6.309	0.029	**0.026**	0.027	7.604	0.065	0.064	2.570	0.032
TANDEM_DUAL	506.947	138.572	0.165	2.873	**0.145**	145.256	13.816	11.260	44.268	0.824
TANDEM_VTX	500.515	29.721	0.058	0.177	**0.049**	4.864	0.896	0.947	3.408	0.096
Number of best results	0	0	2	2	**13**	0	0	0	0	0
$\sum \rho_t$	388,737.49	10,647.87	3.34	118.39	**0.33**	11,070.24	616.20	592.42	3739.04	41.76
Ranking	10	8	2	4	**1**	9	6	5	7	3
Magnitude	3	2	−2	−2	−2	1	−1	−1	1	−2

Table 21 Results of 10 heuristics applied to reduce bandwidth and profile of five asymmetric instances of the University of Florida sparse-matrix collection

Result	Instance	n	β_0	VNS-band (500 s)	FNCHC	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG
Band width	ASIC_100KS	99,190	98,439	31,187	99,155	20,584	88,298	21,540	20,195	85,965	96,004	**17,160**	91,392
	CKT11752_TR_0	49,702	49,452	13,914	45,430	49,251	39,662	49,382	**8959**	41,287	39,838	10,474	40,142
	CRYG10000	10,000	9900	102	102	201	198	199	101	174	175	**100**	196
	HVDC1	24,842	24,762	3096	**1814**	8158	14,182	8535	2159	13,643	208,09	2107	22,997
	SOC-EPINIONS1	75,888	75,416	**24,450**	43,317	75,681	75,547	75,713	45,376	75,108	75,240	40,933	75,836
	No. of best r.	–	–	1	1	0	0	0	1	0	0	**2**	0
	$\sum \rho_b$	–	–	2.1	9.6	11.3	17.5	11.6	1.2	17.0	21.3	**1.0**	22.5
	Ranking	–	–	3	4	5	8	6	2	7	9	**1**	10

Result	Instance	n	profile$_0$	VNS-band (500 s)	FNCHC	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG
Profile	ASIC_100KS	99,190	478,531,931	3,032,996,878	3,591,894,759	1,628,860,388	888,465,618	1,640,771,236	1,318,944,816	801,752,580	886,201,276	1,743,037,215	**671,053,900**
	CKT11752_TR_0	49,702	925,525,268	357,719,615	846,783,610	163,572,689	152,265,161	167,516,073	343,271,483	163,144,194	109,774,205	327,547,878	**99,470,872**
	CRYG10000	10,000	3,930,296	1,536,048	1,494,497	1,348,635	1,343,475	1,348,231	1,343,930	1,343,566	1,343,538	**1,343,391**	1,343,551
	HVDC1	24,842	179,567,305	56,337,346	55,241,949	31,875,404	5,968,346	32,081,783	51,024,261	**5,825,207**	8,389,064	42,293,356	9,003,647
	SOC-EPINIONS1	75,888	1,952,507,969	1,244,690,543	1,995,046,465	2,047,931,756	1,658,682,765	2,050,741,356	1,954,290,225	1,635,597,145	1,620,365,249	1,604,442,779	**1,170,481,266**
	No. of best results	–	–	0	0	0	0	0	0	1	0	1	**3**
	$\sum \rho_p$	–	–	14.99	21.17	7.30	1.30	7.39	11.85	1.23	1.25	10.52	**0.55**
	Ranking	–	–	9	10	5	4	6	8	2	3	7	**1**

Table 21 continued

Result	Instance	n	VNS-band (500 s)	FNCHC	hGPHH-GL	Sloan	RCM–GL	GGPS	Sloan–MGPS	NSloan	GPS	MPG
Average time	ASIC_100KS	99,190	508.665	650.950	0.186	39.633	**0.162**	248.853	83.549	152.652	74.284	21.435
	CKT11752_TR_0	49,702	502.572	308.154	0.075	7.966	**0.061**	92.201	13.257	14.996	21.515	7.033
	CRYG10000	10,000	500.232	7.844	0.012	0.182	**0.010**	1.018	0.232	0.232	1.929	0.181
	HVDC1	24,842	500.844	42.160	0.038	1.712	**0.033**	25.074	1.854	1.957	5.148	1.699
	SOC-EPINIONS1	75,888	505.132	47.729	0.124	27.419	**0.120**	1135.952	173.569	178.549	33.083	10.916
	Number of best results	–	0	0	0	0	**5**	0	0	0	0	0
	$\sum \rho_t$	–	82,531	11,514	1	6677	**0**	13,333	2249	2748	1434	403
	Ranking	–	10	8	2	4	**1**	9	6	7	5	3
	Magnitude	–	3	3	–2	2	–2	2	2	3	2	2

Appendix C: Application areas

Tables 22 and 23 show several instances of the University of Florida and Harwell-Boeing sparse-matrix collections divided by application area, respectively.

Table 22 Twenty two instances of the University of Florida sparse-matrix collection divided by application area

Application area	Instances
Structural problem	BARTH, BARTH4, BARTH5, BCSSTK30, COMMANCHE_DUAL, FORD1, FORD2, NASASRB, ONERA_DUAL, SHUTTLE_EDDY, SKIRT, TANDEM_DUAL, TANDEM_VTX
Computational fluid dynamics problem	COPTER_1, COPTER_2
Optimization problem	FINANCE256, PDS10
Circuit simulation problem	ASIC_100KS, CKT11752_TR_0
Materials problem	CRYG10000
Power network problem	HVDC1
Directed graph	SOC-EPINIONS1

Table 23 One-hundred and thirteen instances of the Harwell-Boeing sparse-matrix collection divided by application area

Application area	Instances
Astrophysics	MCCA, MCFE
Chemical engineering	IMPCOL_A, IMPCOL_B, IMPCOL_C, IMPCOL_D, IMPCOL_E, WEST0132, WEST0156, WEST0167, WEST0381, WEST0479, WEST0497, WEST0655, WEST0989
Chemical kinetics	FS_183_1, FS_541_1, FS_680_1, FS_760_1
Circuit physics	JPWH_991
Dynamic analyses in structural	BCSSTK01, BCSSTK04, BCSSTK05, BCSSTK06,
engineering	BCSSTK19, BCSSTK20, BCSSTK22, BCSSTM07
Economic modeling	MBEACXC, MBEAFLW, MBEAUSE
Finite-element model problem	JAGMESH1
Finite-element structures	CAN_144, CAN_161, CAN_292, CAN_445,
problems in aircraft design	CAN_715, CAN_838
Flow in networks	HOR_131
Fluid flow modeling	LNS_131, LNSP_511
Nuclear reactor modeling	NNC261, NNC666
Oceanic modeling	PLAT362, PLSKZ362
Oil recovery	STEAM1, STEAM2, STEAM3
Oil reservoir modeling/simulation	ORSIRR_2, SAYLR1, SAYLR3, SHERMAN1, SHERMAN4
Partial differential equations	GR_30_30
Power systems networks/	494_BUS, 662_BUS, 685_BUS, BCSPWR01,

Table 23 continued

power networks	BCSPWR02, BCSPWR03, BCSPWR04, BCSPWR05
Reservoir modeling	PORES_1, PORES_3
Simulation studies in computer systems	GRE_115, GRE_185, GRE_216A, GRE_343, GRE_512
Structural engineering/linear equations in structural engineering	DWT_209, DWT_221, DWT_234, DWT_245, DWT_310, DWT_361, DWT_419, DWT_503, DWT_592, DWT_878, DWT_918, DWT_992, NOS1, NOS2, NOS3, NOS4, NOS5, NOS6, NOS7
Variety of disciplines	ARC130, ASH85, ASH292, BP_0, BP_200, BP_400, BP_600, BP_800, BP_1000, BP_1200, BP_1400, BP_1600, CURTIS54, GENT113, IBM32, LUND_A, LUND_B, SHL_0, SHL_200, SHL_400, STR_0, STR_200, STR_600, WILL57, WILL199

Appendix D: The most promising low-cost heuristics for bandwidth and profile reductions of symmetric and asymmetric matrices

Table 24 summarizes the results of the heuristics that showed the best overall performance on the six sets of test matrices. Additionally, Table 24 shows the lowest-cost heuristics tested in these sets of instances. Furthermore, in spite of the small number of executions for each heuristic in each instance, Table 24 shows the largest standard deviation σ and coefficient of variation attained in relation to the execution times of the 14 heuristics for these six datasets tested.

It should be noted that when execution times are crucial for the application, a reasonable bandwidth or profile reduction by a very fast heuristic may be better than a very large bandwidth or profile reduction provided by a heuristic with high computational cost. Then, Table 24 shows the rank order of heuristics and their order of magnitude regarding their computational costs.

Unlike heuristics such as the RCM–GL, hGPHH-GL, GPS, and GGPS, in the heuristics designed for profile reduction tested here, vertices are not exclusively labeled level by level in relation to the level structure rooted at a starting vertex. This favors a profile reduction by selecting a vertex of small degree number to be labeled before other vertices. This is carried out in Snay's (1976), Sloan's (1989), MPG (Medeiros et al. 1993), NSloan (Kumfert and Pothen 1997), and Sloan–MGPS (Reid and Scott 1999) heuristics.

Sloan's algorithm (Sloan 1989) showed lower computational cost than the Sloan–MGPS heuristic (Reid and Scott 1999) in all tests performed [e.g., Sloan's algorithm (Sloan 1989) achieved its results with a lower computing cost of 1 magnitude in relation to the Sloan–MGPS heuristic (Reid and Scott 1999) when applied to the 17 symmetric instances of the University of Florida sparse-matrix collection (see Table 20)]. It should be noted that Sloan's (1989) and MPG (Medeiros et al. 1993) heuristics were implemented with a linked list and the NSloan (Kumfert and Pothen 1997) and Sloan–MGPS (Reid and Scott 1999) heuristics were implemented with a binary heap (as proposed in the original algorithms). Specifically, a version of Sloan's algorithm (Sloan 1989) implemented here with a binary heap was more

Table 24 Better heuristics in four sets of instances of the Harwell-Boeing and in two sets of instances of the University of Florida sparse-matrix collections

Dataset	Type	Best reduction (average order of magnitude of execution times)		Best times (average order of magnitude)	Largest	
		β	Profile		σ	Coefficient of variation
Harwell-Boeing	Symmetric	1°: VNS-band (1)	1°: MPG (−4)	Sloan	0.10	88.36%
		2°: FNCHC (−2)		(−4)	(CSS-band)	(VNS-band)
		4°: GPS (−3)				
		7°: RCM–GL (−4)				
	Asymmetric	1°: VNS-band (2)	1°: MPG (−3)	hGPHH-GL	12.80	42.38%
		2°: FNCHC (−1)	11°: hGPHH-GL (−4)	(−4)	(WBRA)	(CSS-band)
		4°: GGPS (−2)				
		5°: GPS (−3)				
		10°: hGPHH-GL (−4)				
Univ. of Florida	Symmetric	1°: FNCHC (2)	1°: NSloan (1)	RCM–GL	1.18	25.77%
		4°: hGPHH-GL (−2)	2°: MPG (−1)	(−2)	(FNCHC)	(NSloan)
			6°: RCM–GL (−2)			
	Asymmetric	1°: GPS (2)	1°: MPG (1)	RCM–GL	68.61	22.27%
		5°: hGPHH-GL (−2)	5°: hGPHH-GL (−2)	(−2)	(FNCHC)	(FNCHC)

expensive than the version implemented with a linked list. The number of vertices in the max-priority queue in each iteration may be small so that the operations within the linked list are less expensive than the operations within the max heap (which would show a different asymptotic behavior for a sufficiently large number of vertices). On the other hand, Sloan's algorithm (Sloan 1989) was dominated by the Sloan–MGPS heuristic (Reid and Scott 1999) in relation to bandwidth and profile reductions in all tests performed: the MGPS method (Reid and Scott 1999) may have selected pseudo-peripheral vertices with larger eccentricity (or a smaller width of the corresponding RLS was generated) than the pseudo-peripheral vertices given by Sloan's algorithm (Sloan 1989) (for this task).

Sections D.1 and D.2 describe the best heuristics for bandwidth reduction of symmetric and asymmetric matrices, respectively. Finally, Sect. D.3 addresses the best low-cost heuristic for profile reduction of symmetric and asymmetric matrices.

D.1: Best heuristic for bandwidth reduction of symmetric matrices

When setting low VNS-band timeouts and their results compared with low-cost heuristics, the VNS-band (8 s) heuristic provided better bandwidth results in the set composed of 35 symmetric instances of the Harwell-Boeing sparse-matrix collection (see Fig. 3c). The FNCHC heuristic was the second best in reducing bandwidth in this dataset, according to the ρ_b metric (see Fig. 3a and Table 11). The VNS-band heuristic achieved these results with a higher computing cost of two magnitudes in relation to the FNCHC heuristic (see Table 13).

Additionally, in the tests presented in this paper, the FNCHC heuristic obtained the best results in 12 instances in the set composed of 15 symmetric instances (see Fig. 1c). Moreover, the FNCHC heuristic obtained the best results in the 17 symmetric instances of the University of Florida sparse-matrix collection (see Figs. 9a, c in Sect. 4.2.1 and the first part of Table 19). Therefore, the *FNCHC* (Lim et al. 2003, 2004, 2007) heuristic was considered the algorithm that achieved the most promising (overall) results for reducing bandwidth of symmetric matrices.

D.2: Best heuristic for bandwidth reduction of asymmetric matrices

Although the FNCHC and VNS-band heuristics obtained the best results in relation to bandwidth reduction in the sets composed of 18 and 45 asymmetric instances of the Harwell-Boeing sparse-matrix collection (see Figs. 5a, 7a, c, the first part of Tables 14, and 16), respectively, when applied to the dataset composed of five asymmetric instances of the University of Florida sparse-matrix collection, the GPS algorithm obtained the best results according to the ρ_β metric (see Fig. 11a, the first part of Table 21, and Sect. 4.2.2). Moreover, in general, the GPS algorithm achieved the third (see Sects. 4.1.1 and 4.1.2) and forth (see Sects. 4.1.3 and 4.1.4) best results according to the ρ_β metric when applied to the four datasets composed of symmetric and asymmetric instances of the Harwell-Boeing sparse-matrix collection. The GPS algorithm was outperformed by the FNCHC and VNS-band (with its timeout set at 8, 12, and 500 s) heuristics when applied to the 113 instances of the Harwell-Boeing sparse-matrix collection (see Figures 1a–c, 3a–c, 5a, and 7a–c, and Tables 11, 14, and 16). On the other hand, the GPS algorithm outperformed the VNS-band (500 s) when applied to the 22 instances of the University of Florida sparse-matrix collection (see Figs. 9a and 11a; Tables 19, 21).

It should be noted that the FNCHC and VNS-band heuristics generate several solutions and employ local-search procedures to provide local-optimum solutions. The GPS algorithm generates only one solution. Moreover, a local-search procedure is not employed within the GPS algorithm.

The FNCHC and VNS-band heuristics achieved their results with a higher computing cost of 1 magnitude in relation to the GPS heuristic in the set composed of five asymmetric instances of the University of Florida sparse-matrix collection (see Table 21). Thus, the *GPS* algorithm was considered the most promising low-cost heuristic for bandwidth reduction of asymmetric matrices.

D.3: Best low-cost heuristic for profile reduction of symmetric and asymmetric matrices

The MPG (Medeiros et al. 1993) and NSloan (Kumfert and Pothen 1997) heuristics obtained the best profile results in 11 and 9 instances in the set composed of 35 symmetric instances of the Harwell-Boeing sparse-matrix collection, respectively (see Fig. 3d; Table 12). Moreover, the NSloan (Kumfert and Pothen 1997) obtained the best profile results in the set composed of 15 symmetric instances of the Harwell-Boeing sparse-matrix collection (see Fig. 1b, d and the second part of Table 9) and in the set composed of 17 symmetric instances of the University of Florida sparse-matrix collection (see Fig. 9b, d and Table 19). Therefore, the *NSloan* (Kumfert and Pothen 1997) heuristic was considered the best low-cost heuristic for profile reduction of symmetric matrices.

In relation to asymmetric instances, the MPG (Medeiros et al. 1993) heuristic obtained the best results in the sets composed of 18 and 45 asymmetric instances of the Harwell-Boeing sparse-matrix collection (see Figs 5b, 7b, and 7d; Tables 14, 17) and in the set composed of five asymmetric instances of the University of Florida sparse-matrix collection (see Fig. 11b and the second part of Table 21). Therefore, the *MPG* heuristic (Medeiros et al. 1993) was considered the best heuristic for profile reduction of asymmetric matrices.

Usually, in sparse graphs, $\ell(e)$ is larger than m. Thus, Sloan's (1989) and Sloan–MGPS (Reid and Scott 1999) heuristics place more emphasis on the distance of a vertex v to the target end vertex (a global criterion) than the degree of v (a local criterion). Thus, these two heuristics tend to label vertices level by level in a level structure rooted at a starting vertex. By normalizing the priority of these two criteria, the NSloan heuristic (Kumfert and Pothen 1997) places similar emphasis on these two criteria. On the other hand, a vertex-labeling algorithm that labels vertices in an ascending degree number produces poor bandwidth reduction at a high computational cost because it considers many candidate vertices for each iteration (we tested it here within exploratory investigations). Thus, the MPG heuristic (Medeiros et al. 1993) controls the number of candidate vertices and (although assigning the two weights as $w_1 = 1$ and $w_2 = 2$) offers a trade-off between the two criteria mentioned. These characteristics of the NSloan (Kumfert and Pothen 1997) and MPG (Medeiros et al. 1993) heuristics can explain their best results in profile reduction of symmetric and asymmetric instances when compared to the other heuristics for this task, respectively.

References

Benzi M (2002) Preconditioning techniques for large linear systems: a survey. J Comput Phys 182:418–477

Benzi M, Szyld DB, Van Duin A (1999) Orderings for incomplete factorization preconditioning of nonsymmetric problems. SIAM J Sci Comput 20(5):1652–1670

Bernardes JAB (2015) Gonzaga de Oliveira, S.L.: A systematic review of heuristics for profile reduction of symmetric matrices. In: Koziel S, Leifsson L, Lees M, Krzhizhanovskaya VV, Dongarra J, Sloot PM (eds)

International conference on computational science, ICCS 2015, vol 51, pp 221–230. Elsevier, Reykjavik, Iceland
Burgess DA, Giles M (1997) Renumbering unstructured grids to improve the performance of codes on hierarchial memory machines. Adv Eng Softw 28(3):189–201
Burgess IW, Lai PKF (1986) A new node renumbering algorithm for bandwidth reduction. Int J Numer Methods Eng 23:1693–1704
Chagas GO, Gonzaga de Oliveira SL (2015) Metaheuristic-based heuristics for symmetric-matrix bandwidth reduction: a systematic review. In: Koziel S, Leifsson L, Lees M, Krzhizhanovskaya VV, Dongarra J, Sloot PM (eds) International conference on computational science, ICCS 2015, vol 51, pp 211–220. Elsevier, Reykjavik, Iceland
Cuthill EH (1972) Several strategies for reducing the bandwidth of matrices. In: Proceedings of a symposium on sparse matrices and their applications, pp 157–166. The IBM Research Symposia Series, New York, USA
Das R, Mavriplis DJ, Saltz JH, Gupta SK, Ponnusamy R (1992) The design and implementation of a parallel unstructured euler solver using software primitives. Tech. Rep. AD-A249 437, Institute for Computer Applications in Science and Engineering—NASA, Virgina, USA
Davis TA, Hu Y (2011) The university of florida sparse matrix collection. ACM Trans Math Softw 38(1):1–25
Duff IS, Grimes RG, Lewis JG (1989) Sparse matrix test problems. ACM Trans Math Softw 15(1):1–14
Duff IS, Meurant GA (1989) The effect of ordering on preconditioned conjugate gradients. BIT Numer Math 29(4):635–657
Esposito A, Catalano MSF, Malucelli F, Tarricone L (1998) A new matrix bandwidth reduction algorithm. Oper Res Lett 23:99–107
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
Felippa CA (1975) Solution of linear equations with skyline-stored symmetric matrix. Comput Struct 5(1):13–29
Fenves SJ, Law KH (1983) A two-step approach to finite element ordering. Int J Numer Methods Eng 19(6):891–911
George, A.: Computer implementation of the finite element method. Ph.D. thesis, Stanford University, Stanford, USA (1971)
George A, Liu JW (1981) Computer solution of large sparse positive definite systems. Prentice-Hall, Englewood Cliffs
George A, Liu JWH (1979) An implementation of a pseudoperipheral node finder. ACM Trans Math Softw 5(3):284–295
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
Gibbs NE, Poole WG, Stockmeyer PK (1976) Comparison of several bandwidth and profile reduction algorithms. ACM Trans Math Softw 4(1):322–330
Gonzaga de Oliveira, S.L., Chagas, G.O.: 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 (2015)
Hestenes MR, Stiefel E (1952) Methods of conjugate gradients for solving linear systems. J Res Natl Bur Stand 49(36):409–436
Hu Y, Scott JA (2001) A multilevel algorithm for wavefront reduction. SIAM J Sci Comput 23(4):1352–1375
Johnson D (2002) A Theoretician's guide to the experimental analysis of algorithms. In: Goldwasser M, Johnson DS, McGeoch CC (eds) Proceedings of the 5th and 6th DIMACS implementation challenges. American Mathematical Society, Providence
Kaveh A, Sharafi P (2012) Ordering for bandwidth and profile minimization problems via charged system search algorithm. Iran J Sci Technol Trans Civil Eng 36(2):39–52
King IP (1970) An automatic reordering scheme for simultaneous equations derived from network systems. Int J Numer Methods Eng 2(4):523–533
Koohestani, B., Poli, R.: A hyper-heuristic approach to evolving algorithms for bandwidth reduction based on genetic programming. In: Research and development in intelligent systems XXVIII, pp 93–106. Springer London, London, UK (2011)
Kumfert G, Pothen A (1997) Two improved algorithms for envelope and wavefront reduction. BIT Numer Math 37(3):559–590
Lanczos C (1952) Solutions of systems of linear equations by minimized iterations. J Res Natl Bur Stand 49(1):33–53
Lewis JG (1982) Implementations of the Gibbs–Poole–Stockmeyer algorithms and Gibbs–King algorithms. ACM Trans Math Softw 8:180–189

Lim A, Rodrigues B, Xiao F (2003) A new node centroid algorithm for bandwidth minimization. In: Proceedings of the 18th international joint conference on artificial intelligence, IJCAI'03Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp 1544–1545

Lim A, Rodrigues B, Xiao F (2004) A centroid-based approach to solve the bandwidth minimization problem. In: Proceedings of the Hawaii international conference on system sciences. Institute of electrical and electronics engineers Inc., Big Island, HI, United States, vol 37, pp 1203–1208

Lim A, Rodrigues B, Xiao F (2007) A fast algorithm for bandwidth minimization. Int J Artif Intell Tools 3:537–544

Lin YX, Yuan JJ (1994) Profile minimization problem for matrices and graphs. Acta Math Appl Sin 10(1):107–122

Martí 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

Medeiros SRP, Pimenta PM, Goldenberg P (1993) Algorithm for profile and wavefront reduction of sparse matrices with a symmetric structure. Eng Comput 10(3):257–266

Mladenovic N, Urosevic D, Pérez-Brito D, García-González CG (2010) Variable neighbourhood search for bandwidth reduction. Eur J Oper Res 200:14–27

Papadimitriou CH (1976) The NP-completeness of bandwidth minimization problem. Comput J 16:177–192

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

Reid JK, Scott JA (1999) Ordering symmetric sparse matrices for small profile and wavefront. Int J Numer Methods Eng 45(12):1737–1755

Reid JK, Scott JA (2006) Reducing the total bandwidth of a sparse unsymmetric matrix. SIAM J Matrix Anal Appl 28(3):805–821

Rodriguez-Tello E, Kao HJ, Torres-Jimenez J (2008) An improved simulated annealing algorithm for bandwidth minimization. Eur J Oper Res 185:1319–1335

Saad Y, Schultz MH (1986) GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Stat Comput 7:856–869

Sloan SW (1989) A Fortran program for profile and wavefront reduction. Int J Numer Methods Eng 28(11):2651–2679

Snay RA (1976) Reducing the profile of sparse symmetric matrices. Bull Geod 50(4):341–352

STFC: The Science and Technology Facilities Council. HSL. A collection of Fortran codes for large scale scientific computation. http://www.hsl.rl.ac.uk. Accessed Dec 2015

The MathWorks, I.: MATLAB. http://www.mathworks.com/products/matlab/index.html (1994–2016). Accessed Mar 2016

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 33:33–49

Wang, C., Xu, C., Lisser, A.: Bandwidth minimization problem. In: 10th International conference on modeling, optimization and simulation (Mosim14). Nancy, France (2014)

Wang Q, Guo YC, Shi XW (2009) A generalized GPS algorithm for reducing the bandwidth and profile of a sparse matrix. Prog Electromagn Res 90:121–136