

A Biased Random-Key Genetic Algorithm for Bandwidth Reduction

P. H. G. Silva[1], D. N. Brandão[1], I. S. Morais[1(✉)], and S. L. Gonzaga de Oliveira[2]

[1] Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, Rio de Janeiro, RJ, Brazil
{pegonzalez,diego.brandao,igor.morais}@eic.cefet-rj.br
[2] Universidade Federal de Lavras, Lavras, MG, Brazil
sanderson@ufla.br

Abstract. The bandwidth minimization problem is a well-known $\mathcal{NP}$-hard problem. This paper describes our experience in implementing a biased random-key genetic algorithm for the bandwidth reduction problem. Specifically, this paper compares the results of the new algorithm with the results yielded by four approaches. The results obtained on a set of standard benchmark matrices taken from the SuiteSparse sparse matrix collection indicated that the novel approach did not compare favorably with the state-of-the-art metaheuristic algorithm for bandwidth reduction. The former seems to be faster than the latter. On the other hand, the design of heuristics for bandwidth reduction is a very consolidated research area. Thus, a paradigm shift seems necessary to design a heuristic with better results than the state-of-the-art metaheuristic algorithm at shorter execution times.

Keywords: Bandwidth reduction · Biased random-key genetic algorithms · Ordering · Reordering algorithms · Renumbering · Sparse matrices

1 Introduction

The bandwidth minimization problem consists of labeling the vertices of a graph $G = (V, E)$ (composed of a set of vertices V and a set of edges E) with integer numbers from 1 to n, where $n = |V|$ is the number of vertices such that the maximum absolute difference between labels of adjacent vertices is minimum. The bandwidth minimization problem is a well-known $\mathcal{NP}$-hard problem [1]. Prevalent algorithms for bandwidth reduction are heuristic methods since they attempt to find a labeling that provides a small bandwidth in a reasonable amount of time. Thus, heuristics for bandwidth reduction place nonzero coefficients of a sparse matrix as close to the main diagonal as possible. Figure 1 illustrates an example of applying a heuristic for bandwidth reduction to a matrix.

Let $A = [a_{ij}]$ be an $n \times n$ symmetric adjacency matrix associated with an undirected graph $G = (V, E)$. The bandwidth of matrix A is defined as $\beta(A) =$

© Springer Nature Switzerland AG 2020
O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12249, pp. 312–321, 2020.
https://doi.org/10.1007/978-3-030-58799-4_23

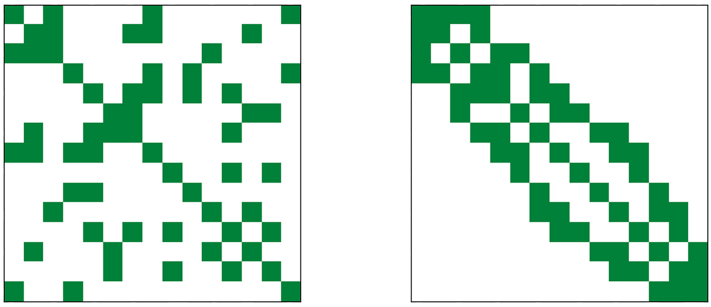

Fig. 1. An example of applying a heuristic for bandwidth reduction to a matrix. The nonzero coefficients (represented in green) of the matrix on the left are placed along the main diagonal in the matrix on the right after executing a heuristic for bandwidth reduction. (color figure online)

$\max_{1 \leq i \leq n} [\beta_i(A)]$, where $\beta_i(A) = i - \min_{1 \leq j \leq i} [j : a_{ij} \neq 0]$. Equivalently, the bandwidth of G for a vertex labeling $S = \{s(v_1), s(v_2), \cdots, s(v_{|V|})\}$ (i.e. a bijective mapping from V to the set $\{1, 2, \cdots, |V|\}$) is $\beta(G) = \max_{\{v,u\} \in E} [|s(v) - s(u)|]$.

Since bandwidth reduction is related to a vast range of important scientific and engineering applications, such as computational fluid dynamics and structural problems, researchers have proposed a large number of heuristics to reduce the bandwidth of matrices since the 1960s [2]. The Reverse Cuthill-McKee method [3] is the most classical heuristic for bandwidth reduction [4–7]. For instance, this method is available on MATLAB [8,9] and GNU Octave [10] mathematical softwares as the function *symrcm*[1], and on Boost C++ Library [11][2].

Metaheuristic-based algorithms for bandwidth reduction began to be proposed mainly in the 1990s [12]. As previously mentioned, since bandwidth reduction is a relevant problem in various scientific and engineering fields, practitioners have applied the best-known metaheuristics in the design of heuristics for bandwidth reduction [12], including tabu search [13,14], GRASP [15], genetic algorithm [16,17], ant colony optimization [6,7,18–21], particle swarm optimization [22], simulated annealing [23,24], variable neighbourhood search [25], genetic programming [26–28] and self-organizing maps [29,30].

The VNS-band heuristic [25] yielded better bandwidth results than a heuristic based on the simulated annealing metaheuristic [23], which was considered the previous state-of-the-art heuristic for bandwidth reduction. Torres-Jimenez et al. [24] proposed a Dual Representation Simulated Annealing (DRSA) heuristic for bandwidth reduction. The DRSA heuristic delivered better bandwidth

[1] https://www.mathworks.com/help/matlab/ref/symrcm.html?requestedDomain=www.mathworks.com, https://octave.sourceforge.io/octave/function/symrcm.html.

[2] http://www.boost.org/doc/libs/1_38_0/libs/graph/doc/cuthill_mckee_ordering.html.

results than the VNS-band heuristic. Based on the results reported [24], the DRSA heuristic is the state-of-the-art metaheuristic-based algorithm for bandwidth reduction.

Researchers have successfully applied the biased random-key genetic algorithms (BRKGA) to several optimization problems such as routing in OSPF networks [31], cell formation in manufacturing [32], and telecommunications [33]. This paper implements a BRKGA [34] for bandwidth reduction. To the best of our knowledge, this is the first heuristic for bandwidth reduction based on BRKGA. This paper compares the results yielded by the novel algorithm with the results obtained by the DRSA heuristic [24], a GRASP heuristic, the Reverse Cuthill-McKee method [3], and a variant of the breadth-first search procedure.

The remainder of this paper is organized as follows. In Sect. 2, a BRKGA for bandwidth reduction is introduced. In Sect. 3, we describe how the experiments were conducted. In Sect. 4, we discuss the results. Finally, in Sect. 5, the main conclusions are presented.

2 A BRKGA for Bandwidth Reduction

A BRKGA is similar to a genetic algorithm, but the former employs different strategies aiming at surpassing some disadvantages related to the latter. Bean [35] introduced the random-key genetic algorithms (RKGA) for combinatorial optimization problems. In RKGA, a permutation vector represents a solution to the problem in context. Vectors of random real numbers, called keys, represent the solutions. A deterministic algorithm, called a decoder, associates a feasible solution of the combinatorial optimization problem with a solution vector. Thus, the algorithm can compute an objective value or fitness function to the solution of the combinatorial optimization problem.

RKGA randomly selects two parents from the entire population to implement the crossover operation. Parents are allowed to be selected for mating more than once in a given generation. The principal objective of an RKGA is to mitigate the difficulty of genetic algorithm operators in dealing with feasible solutions.

A BRKGA selects parents for crossover differently from RKGA (see [34] for a review). BRKGA generates each element combining one individual selected at random from the elite solutions in the current population, whereas the other is a non-elite solution. The selection is said biased because one parent is always an elite individual. Therefore, elite solutions have a high probability of passing their genes to individuals of the new generation.

We followed the mains steps of the metaheuristic approach to design the BRKGA for bandwidth reduction. The BRKGA for bandwidth reduction evolves a population of chromosomes that consists of vectors of real numbers (called keys). Thus, a vector represents each solution. Each component in this vector is a real number in the range $[0, 1]$. Each solution represented by a chromosome is decoded by a decoding heuristic that receives the vector of keys and builds a solution.

We used a BRKGA framework provided by Toso and Resende [36]. In this framework, the user implements only problem-dependent procedures, such as

the decoding method, which is responsible for decoding a vector of random keys to a solution of the problem in context. The decoding method is also responsible for evaluating the solution and treating penalties if the algorithm generates an infeasible solution. Thus, the decoding method interprets a vector of random keys of the BRKGA.

BRKGA obtains each new individual by combining a randomly selected individual from the elite group (p_e) of the current population (p), and another one from the non-elite set ($p \backslash p_e$) of individuals. Thus, a single individual can be selected multiple times, and therefore can produce more than one offspring. Since BRKGA requires that $|p_e| < |p \backslash p_e|$, the probability of BRKGA to select for reproduction ($1/|p_e|$) an individual from the elite group is greater than the probability of the algorithm to select a non-elite individual ($1/|p \backslash p_e|$).

BRKGA does not use the standard mutation operator. This operator changes parts of chromosomes with a small probability. Instead, BRKGA employs the concept of mutants. Specifically, BRKGA introduces a fixed number of mutant solutions in the population in each generation. The algorithm randomly generates mutants solutions, equivalently as performed to construct the initial population. Mutants are equivalent to the mutation operator in traditional genetic algorithms, diversifying the search and helping the procedure to escape from local optimal solutions.

A constructive heuristic generates the initial population in the BRKGA for bandwidth reduction. The method is a random breadth-first search procedure. The procedure randomly labels vertices belonging to an adjacency list.

We used the parametrized uniform crossover scheme proposed by Spears and De Jong [37] to combine two parent solutions and produce an offspring. The offspring inherits each key from the best fit of the two parents with probability $\phi_a > 0.5$ and from the other parent with probability $1 - \phi_a$.

To accelerate convergence, each time the approach calls the decoder, if the solution is feasible, the BRKGA applies a built-in local search procedure. Specifically, the BRKGA applies a decoding method that takes a vector of random keys as input and returns the difference in bandwidth reduction between the new solution and the original bandwidth of the matrix. After executing the decoding method, the BRKGA performs a local search procedure proposed by Limpg et al. [16]. After applying the local search, the BRKGA converts the newly found solution to a chromosome representation and returns its cost.

3 Description of the Tests

We used the C++ programming language to implement the BRKGA for bandwidth reduction. We employed the same programming language to code the Reverse Cuthill-McKee method, GRASP approach, and a variant of the breadth-first search procedure (i.e., the constructive approach used to generate the initial population in the BRKGA). The workstation used in the execution of the simulations featured an Intel® Core™ i7 (CPU 4.2 GHz, 32 GB of main memory) (Intel; Santa Clara, CA, United States).

The BRKGA for bandwidth reduction uses a specific seed for random number generation at each run. After an exploratory investigation, we set the parameters of the serial BRKGA as follows:

- size of chromosomes : n;
- population size : 1,000;
- elite set fraction : 0.5;
- fraction of population to be replaced by mutants : 0.5;
- probability that offspring inherits an allele from elite parent : 0.5;
- number of independent populations : 1;
- current generation : 0 (so that the BRKGA restarts from the beginning of the approach);
- the best two individuals are combined every 100 generations;
- stopping criterion : 1,000 generations.

The results yielded by the DRSA method were taken from the original publication [24]. The DRSA heuristic was executed in a processor AMD Opteron^{TM} 6274 running at 2.2 GHz [24]. The authors of the DRSA algorithm [24] presented the best bandwidth achieved by their method and the running times in seconds required to yield the best bandwidth. However, we were not able to find the timeout set for the heuristic.

4 Results

In this section, we evaluate the results obtained by the BRKGA for bandwidth reductions against the DRSA algorithm, GRASP approach, the Reverse Cuthill-McKee method, and a variant of the breadth-first search (Rand-BFS) procedure when applied to 43 matrices taken from the SuiteSparse matrix collection [38]. Table 1 shows the name, size, and original bandwidth (β_0) of the matrices used in this computational experiment. Additionally, the table shows the bandwidth results and CPU times obtained by the DRSA (taken from the original publication [24]), BRKGA (using the evolutionary algorithm in one execution carried out in each matrix), and GRASP approaches. The Reverse Cuthill-McKee method (RCM) took less than 0.01 seconds to compute each of the matrices used in this study. The variant of the breadth-first search procedure is even faster than the Reverse Cuthill-McKee method.

The last line of Table 1 shows $\sum \beta$, i.e., the sum of bandwidth results yielded by the algorithms. The same line shows the average time of the heuristics evaluated.

As expected, the other algorithms analyzed in this study dominated the Rand-BFS procedure in terms of bandwidth reduction. The metaheuristic algorithms returned better bandwidth results than did the Rand-BFS procedure in 42 matrices. The Rand-BFS procedure yielded the same bandwidth result as the BRKGA and GRASP algorithms when applied to the matrix gr_30_30. In particular, the Rand-BFS procedure delivered a better bandwidth result than the RCM method in this matrix. On the other hand, the RCM method yielded

Table 1. Bandwidth results yielded by several methods when applied to 43 matrices.

Matrix	n	β_0	DRSA(β)	t(s)	BRKGA(β)	t(s)	GRASP(β)	t(s)	RCM	Rnd-BFS
dwt_209	209	184	23	2	33	16	33	1	33	47
gre_216a	216	36	21	5	21	13	21	1	21	44
dwt_221	221	187	13	7	15	14	15	5	15	27
impcol_e	225	92	42	9	63	36	67	2	75	106
dwt_245	245	115	21	9	33	13	34	1	55	56
bcspwr04	274	265	24	3	37	15	37	1	49	70
ash292	292	24	19	4	24	18	24	1	32	37
can_292	292	282	38	6	57	27	60	1	72	81
dwt_310	310	28	12	11	13	20	13	1	15	19
gre_343	343	49	28	4	28	22	28	1	28	81
dwt_361	361	50	14	23	15	24	15	1	15	44
dwt_419	419	356	25	45	32	31	32	1	34	67
bcsstk06	420	47	45	15	48	116	49	3	50	55
bcsstm07	420	47	45	14	49	102	49	3	56	73
impcol_d	425	406	39	844	56	25	55	1	80	65
hor_131	434	421	54	26	66	143	65	5	84	119
bcspwr05	443	435	27	7	49	17	49	1	68	57
can_445	445	403	52	11	74	32	74	1	78	119
494_bus	494	428	28	18	55	17	55	1	82	71
dwt_503	503	452	41	50	51	66	52	2	64	77
gre_512	512	64	36	9	56	25	36	1	36	69
fs_541_1	541	540	270	1	465	181	433	117	533	538
dwt_592	592	259	29	53	40	45	42	1	42	65
662_bus	662	335	39	14	77	26	78	1	118	103
fs_680_1	680	600	17	27	17	50	20	1	20	41
685_bus	685	550	32	145	57	31	57	1	102	77
can_715	715	611	71	182	116	70	116	4	133	155
fs_760_1	760	740	37	39	39	146	41	4	43	81
bcsstk19	817	567	14	97	20	59	20	2	22	31
bp_0	822	820	234	1219	386	92	417	2	422	593
bp_1000	822	820	283	4988	457	140	496	5	479	642
bp_1200	822	820	287	1165	467	142	488	5	506	607
bp_1400	822	820	291	2140	475	141	484	5	536	632
bp_1600	822	820	292	2261	469	144	488	5	539	624
bp_200	822	820	257	1639	471	108	459	3	508	649
bp_400	822	820	267	2341	480	115	479	5	530	578
bp_600	822	820	272	1315	488	121	485	4	541	614
bp_800	822	820	278	1065	454	131	492	5	542	552
can_838	838	837	86	70	111	118	119	4	151	164
dwt_878	878	519	25	85	33	64	35	3	46	77
gr_30_30	900	31	33	57	48	65	48	4	59	48
dwt_918	918	839	32	62	46	65	46	2	57	61
dwt_992	992	513	35	107	52	194	52	5	65	70
$\sum \beta$ and average time			3828	470	6143	71	6258	5	7036	8386

better bandwidth results than did the Rand-BFS procedure in 37 matrices. The latter returned better bandwidth results than did the former in five matrices.

The metaheuristic algorithms dominated the RCM method. The DRSA algorithm yielded better bandwidth results than did the RCM method in 41 matrices. Moreover, the latter delivered the same bandwidth results as the former in two cases. The BRKGA yielded better bandwidth results than did the RCM method in 36 matrices. The latter delivered the same bandwidth results as the former in six test problems. The RCM method delivered a better bandwidth result than did the BRKGA when applied to the matrix gre_512. The GRASP algorithm yielded better bandwidth results than did the RCM method in 35 matrices. The latter provided the same bandwidth results as the former when applied to seven matrices. The RCM method delivered better bandwidth results than did the GRASP algorithm when applied to the matrix bp_1000.

In general, the BRKGA returned better bandwidth results than did the GRASP algorithm at longer running times. The BRKGA yielded better bandwidth results than did the GRASP algorithm in 17 matrices. Furthermore, the latter provided better bandwidth results than did the former in seven test problems. Both algorithms returned the same bandwidth in 19 matrices.

The DRSA algorithm yielded better bandwidth results than did the four other heuristics evaluated in 40 test problems. The DRSA and BRKGA approaches provided the same results in three cases (matrices gre_216a, gre_343, and fs_680_1). We could not compare the running times of the algorithms because the studies used different machines.

We performed statistical tests for pairwise comparison. Specifically, we used the Wilcoxon matched-pairs signed-rank test to analyze the bandwidth reduction yielded by the DRSA and BRKGA approaches. The null hypothesis for the test was that the algorithms yielded identical results, i.e., the median difference was zero. The alternative hypothesis was that the median difference is positive. Thus, we used a two-tailed test because we wanted to determine if there was any difference between the groups we were comparing (i.e., the possibility of positive or negative differences). In this case, $R_+ = 820$ and $R_- = 0$. The test statistic is, therefore, $T = 0$. Thus, $T \leq T_{crit(\alpha=0.01,40)} = 247$. We have statistically significant evidence at $\alpha = 0.01$ (i.e., the level of significance 0.01 is related to the 99% confidence level), to show that the median difference is positive, i.e., that the DRSA algorithm dominated the BRKGA. In this case, p-value equals 3.56056e−8. The use of a corrected significance level of $1 - (1 - \alpha)^{1/2} = 0.005$ (Dunn-Šidák correction) or $\alpha/2 \approx 0.005$ (Bonferroni correction) confirms the previous statement.

5 Conclusions

This paper describes our experience in implementing a BRKGA for bandwidth reduction. The algorithm did not compare favorably with the state-of-the-art metaheuristic-based heuristic for bandwidth reduction. Although the novel approach yielded better bandwidth results than a GRASP approach and classical

methods for bandwidth reduction, namely the Reverse Cuthill-McKee and a variant of the breadth-first search procedure, the DRSA algorithm dominated the BRKGA approach. The design of a parallel BRKGA for bandwidth reduction is a future step in this investigation.

References

1. Papadimitriou, C.H.: The NP-completeness of bandwidth minimization problem. Comput. J. **16**, 177–192 (1976)
2. Gonzaga de Oliveira, S.L., Chagas, G.O.: A systematic review of heuristics for symmetric-matrix bandwidth reduction: methods not based on metaheuristics. In: The XLVII Brazilian Symposium of Operational Research (SBPO), Ipojuca-PE, Brazil, Sobrapo, August 2015
3. George, A., Liu, J.W.: Computer Solution of Large Sparse Positive Definite Systems. Prentice-Hall, Englewood Cliffs (1981)
4. Gonzaga de Oliveira, S.L., Bernardes, J.A.B., Chagas, G.O.: An evaluation of low-cost heuristics for matrix bandwidth and profile reductions. Comput. Appl. Math. **37**(2), 1412–1471 (2016). https://doi.org/10.1007/s40314-016-0394-9
5. Gonzaga de Oliveira, S.L., Abreu, A.A.A.M.: An evaluation of pseudoperipheral vertex finders for the reverse Cuthill-Mckee method for bandwidth and profile reductions of symmetric matrices. In: Proceedings of the 37th International Conference of the Chilean Computer Science Society (SCCC), Santiago, Chile, November 2018, pp. 1–9. IEEE (2018). https://doi.org/10.1109/SCCC.2018.8705263
6. Gonzaga de Oliveira, S. L., Silva, L.M.: Evolving reordering algorithms using an ant colony hyperheuristic approach for accelerating the convergence of the ICCG method. Eng. Comput. (2019). https://doi.org/10.1007/s00366-019-00801-5)
7. Gonzaga de Oliveira, S., Silva, L.: An ant colony hyperheuristic approach for matrix bandwidth reduction. Appl. Soft Comput. **94**, 106434 (2020)
8. Gilbert, J.R., Moler, C., Schreiber, R.: Sparse matrices in MATLAB: design and implementation. SIAM J. Matrix Anal. **3**(1), 333–356 (1992)
9. The MathWorks Inc.: MATLAB (1994–2018). http://www.mathworks.com/products/matlab/
10. Eaton, J.W., Bateman, D., Hauberg, S., Wehbring, R.: GNU Octave version 4.0.0 manual: a high-level interactive language for numerical computations (2015)
11. Boost: Boost C++ libraries (2017). http://www.boost.org/. Accessed 28 Jun 2017
12. Chagas, G.O., Gonzaga de Oliveira, S.L.: Metaheuristic-based heuristics for symmetric-matrix bandwidth reduction: a systematic review. Procedia Comput. Sci. 51, 211–220 (2015)
13. Martí, R., Laguna, M., Glover, F., Campos, V.: Reducing the bandwidth of a sparse matrix with tabu search. Eur. J. Oper. Res. **135**(2), 450–459 (2001)
14. Campos, V., Piñana, E., Martí, R.: Adaptive memory programming for matrix bandwidth minimization. Ann. Oper. Res. **183**, 7–23 (2011)
15. Piñana, E., Plana, I., Campos, V., Martí, R.: GRASP and path relinking for the matrix bandwidth minimization. Eur. J. Oper. Res. **153**(1), 200–210 (2004)
16. Lim, A., Rodrigues, B., Xiao, F.: Heuristics for matrix bandwidth reduction. Eur. J. Oper. Res. **174**(1), 69–91 (2006)
17. Czibula, G., Crişan, G.C., Pintea, C.M., Czibula, I.G.: Soft computing approaches on the bandwidth problem. Informatica **24**(2), 169–180 (2013)

18. Lim, A., Lin, J., Rodrigues, B., Xiao, F.: Ant colony optimization with hill climbing for the bandwidth minimization problem. Appl. Soft Comput. **6**(2), 180–188 (2006)
19. Kaveh, A., Sharafi, P.: Nodal ordering for bandwidth reduction using ant system algorithm. Eng. Comput. **26**, 313–323 (2009)
20. Pintea, C.-M., Crişan, G.-C., Chira, C.: A hybrid ACO approach to the matrix bandwidth minimization problem. In: Graña Romay, M., Corchado, E., Garcia Sebastian, M.T. (eds.) HAIS 2010. LNCS (LNAI), vol. 6076, pp. 405–412. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13769-3_49
21. Pintea, C.M., Crişan, G.C., Shira, C.: Hybrid ant models with a transition policy for solving a complex problem. Logic J. IGPL **20**(3), 560–569 (2012)
22. Lim, A., Lin, J., Xiao, F.: Particle swarm optimization and hill climbing for the bandwidth minimization problem. Appl. Intell. **3**(26), 175–182 (2007)
23. Rodriguez-Tello, E., Jin-Kao, H., Torres-Jimenez, J.: An improved simulated annealing algorithm for bandwidth minimization. Eur. J. Oper. Res. **185**, 1319–1335 (2008)
24. Torres-Jimenez, J., Izquierdo-Marquez, I., Garcia-Robledo, A., Gonzalez-Gomez, A., Bernal, J., Kacker, R.N.: A dual representation simulated annealing algorithm for the bandwidth minimization problem on graphs. Inf. Sci. **303**, 33–49 (2015)
25. Mladenovic, N., Urosevic, D., Pérez-Brito, D., García-González, C.G.: Variable neighbourhood search for bandwidth reduction. Eur. J. Oper. Res. **1**(200), 14–27 (2010)
26. Koohestani, B., Poli, R.: A hyper-heuristic approach to evolving algorithms for bandwidth reduction based on genetic programming. In: Bramer, M., Petridis, M., Nolle, L. (eds.) Research and Development in Intelligent Systems XXVIII, pp. 93–106. Springer, London (2011). https://doi.org/10.1007/978-1-4471-2318-7_7
27. Pop, P.C., Matei, O.: An improved heuristic for the bandwidth minimization based on genetic programming. In: Corchado, E., Kurzyński, M., Woźniak, M. (eds.) HAIS 2011. LNCS (LNAI), vol. 6679, pp. 67–74. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21222-2_9
28. Pop, P., Matei, O., Comes, C.A.: Reducing the bandwidth of a sparse matrix with a genetic algorithm. Optim. J. Math. Prog. Oper. Res. **63**(12), 1851–1876 (2013)
29. Gonzaga de Oliveira, S.L., de Abreu, A.A.A.M., Robaina, D., Kischinhevsky, M.: A new heuristic for bandwidth and profile reductions of matrices using a self-organizing map. In: Gervasi, O., et al. (eds.) ICCSA 2016. LNCS, vol. 9786, pp. 54–70. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42085-1_5
30. Gonzaga de Oliveira, S.L., Abreu, A.A.A.M., Robaina, D.T., Kischnhevsky, M.: An evaluation of four reordering algorithms to reduce the computational cost of the Jacobi-preconditioned conjugate gradient method using high-precision arithmetic. Int. J. Bus. Intell. Data Min. **12**(2), 190–209 (2017)
31. Ericsson, M., Resende, M.G.C., Pardalos, P.M.: A genetic algorithm for the weight setting problem in OSPF routing. J. Comb. Optim. **6**, 299–333 (2002)
32. Gonçalves, J.F., Resende, M.G.C.: An evolutionary algorithm for manufacturing cell formation. Comput. Ind. Eng. **47**, 247–273 (2004)
33. Resende, M.G.C.: Biased random-key genetic algorithms with applications in telecommunications. TOP **20**, 130–153 (2012)
34. Gonçalves, J.F., Resende, M.G.C.: Biased random-key genetic algorithms for combinatorial optimization. J. Heuristics **17**, 487–525 (2011)
35. Bean, J.C.: Genetic algorithms and random keys for sequencing and optimization. ORSA J. Comput. **6**, 154–160 (1994)
36. Toso, R.F., Resende, M.G.C.: A C++ application programming interface for biased random-key genetic algorithms. Optim. Methods Softw. **30**, 81–93 (2015)

37. Spears, W.M., De Jong, K.D.: On the virtues of parameterized uniform crossover. Technical report, DTIC Document (1995)
38. Davis, T.A., Hu, Y.: The University of Florida sparse matrix collection. ACM Trans. Math. Softw. **38**(1), 1–25 (2011)