A study of Recombination Operators for the Cyclic Bandwidth Problem

Jintong Ren[1], Jin-Kao Hao[1,2⋆], Eduardo Rodriguez-Tello[3]

[1] LERIA, Université d'Angers, 2 Boulevard Lavoisier, 49095 Angers, France
[2] Institut Universitaire de France, 1 rue Descartes, 75231 Paris, France
[3] CINVESTAV - Tamaulipas, Km. 5.5 Carretera Victoria - Soto La Marina, 87130 Victoria Tamps., Mexico

Abstract. This work is dedicated to a study of the NP-hard Cyclic Bandwidth Problem with the paradigm of memetic algorithms. To find out how to choose or design a suitable recombination operator for the problem, we study five classical permutation crossovers within a basic memetic algorithm integrating a simple descent local search procedure. We investigate the correlation between algorithmic performances and population diversity measured by the average population distance and entropy. This work invites more research to improve the two key components of the memetic algorithm: reinforcement of the local search and design of a meaningful recombination operator suitable for the problem.

Keywords: Recombination Operators · Memetic Algorithms · Cyclic Bandwidth · Population Diversity.

1 Introduction

The Cyclic Bandwidth Problem (CBP) is a typical graph labeling problem, which was introduced in [14] in the context of designing a ring interconnection network. CBP involves finding a disposition of computers on a cycle to make sure that the intercommunication information reaches its destination within at most k steps. The decision version of the problem is known to be a NP-complete problem [15]. In addition to network design, CBP has other relevant applications in very-large-scale integration design [3] and data structure representation [25].

CBP can be stated formally as follows: let $G(V, E)$ be a finite undirected graph and C_n a cycle graph, where V ($|V| = n$) is the set of vertices (or nodes) and $E \subset V \times V$ is the set of edges. Given a bijection (or arrangement) $\varphi : V \rightarrow V$ which represents an embedding of G in C_n, the cyclic bandwidth (the cost) of φ for G is defined as,

$$B_C(G, \varphi) = \max_{(u,v)\in E} \{|\varphi(u) - \varphi(v)|_n\}, \quad (1)$$

⋆ Corresponding author: jin-kao.hao@univ-angers.fr

where $|x|_n = min\{|x|, n - |x|\}$ $(1 \leq |x| \leq n-1)$ is called the *cyclic distance*, and $\varphi(u)$ denotes the label associated to vertex u. The goal of CBP is to find an arrangement φ^* with minimal $B_C(G, \varphi^*)$.

As a well-known meta-heuristic framework [12,17], memetic algorithms (MAs) have been widely used to solve a large number of NP-hard problems [5, 11, 13, 28, 29]. For permutation problems, MAs have also reported good performances for the Traveling Salesman Problem (TSP) [8, 16], the Quadratic Assignment Problem [2], and other bandwidth problems [1, 20].

Despite the theoretical and practical relevance of CBP, few studies can be found in the literature for solving the problem. A branch and bound algorithm was presented [24] to handle small graphs ($n < 40$). A tabu search algorithm was proposed [23] to deal with standard and random graphs with 8 to 8192 vertices. Very recently, an iterated three-phase search approach [19] was introduced and improved a number of previous best results reported in [23]. To our knowledge, the memetic approach has never been experimented to solve CBP in the literature, though MAs have been applied to other labeling problems such as the cyclic bandwidth sum problem [22] and the antibandwidth problem [20]. This work fills the gap by investigating the memetic approach for CBP. In particular, we focus on the role of the recombination or crossover (used interchangeably in this paper) component and study the contributions of five permutation recombination operators which are conveniently applicable to CBP. To highlight the impacts of the studied recombination operators, we base our study on a canonical memetic algorithm which combines a recombination operator for solution generation and a simple descent local search for solution improvement.

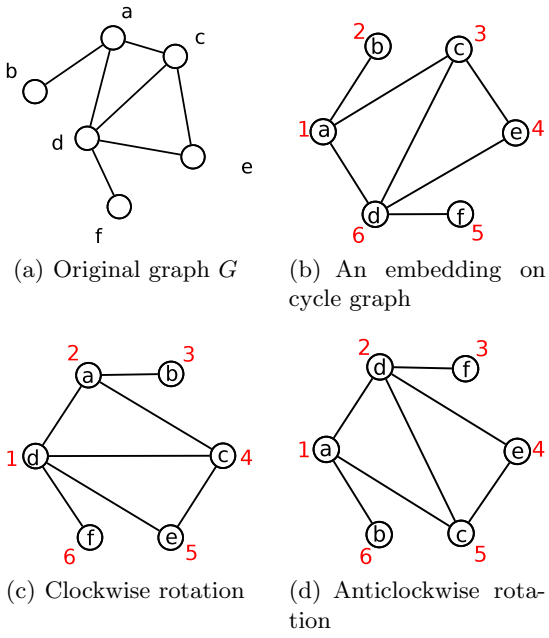
2 Memetic Algorithm for CBP

2.1 Search Space, Representation, Fitness Function

Given a graph $G = (V, E)$ of order $|V| = n$ and a cycle graph C_n, the search space Ω for the CBP is composed of all possible embeddings (labellings, solutions or arrangements) of G in C_n, $\varphi : V \rightarrow V$. Considering the symmetry characteristic of solutions, there exist $(n-1)!/2$ possible embeddings for G [23].

Figure 1 shows a graph with 6 vertices named from 'a' to 'f' (Fig. 1(a)). According to Equation (1), the objective value of Fig. 1(b) is 3 (decided by the longest edge 'dc' in the example). An embedding arranged in a cycle graph (Fig. 1(b)) where the numbers in red are the labels assigned to the vertices, and two embeddings where the vertices are rearranged in the cycle graph in clockwise direction (Fig. 1(c)) and in anticlockwise direction (Fig. 1(d)). Notice that the relative position of each pair of nodes in Fig. 1(b)-1(d) is not changed. So according to Equation (1), these three embeddings have the same objective value, and in fact they correspond to the same solution.

In practice, we represent an embedding φ by permutations $l = \{1, 2, \ldots, n\}$ such that the i-th element $l[i]$ denotes the label assigned to vertex i of V. Another representation of an embedding is proposed in [21], which maps a permutation



(a) Original graph G (b) An embedding on cycle graph

(c) Clockwise rotation (d) Anticlockwise rotation

Fig. 1. Illustration of a graph (a) with an embedding (b) and two equivalent symmetric embeddings (c) and (d)

φ to an array γ indexed by the labels. The i-th value of $\gamma[i]$ indicates the vertex whose label is i. We illustrate these representations with an example. For the embedding of Fig. 1(b), we have φ=(1 2 3 6 4 5) for the vertices from 'a' to 'f', and the corresponding γ representation is γ=(a b c e f d). In our algorithm, the φ representation is used in the local search procedure, because it eases the implementation of the *swap* operation, while the γ representation is adopted for the recombination operators, as well as the distance calculation presented in Section 5. The fitness of a candidate embedding φ in the search space is evaluated by Equation (1).

2.2 General procedure

The studied MA follows the general MA framework in discrete optimization [10]. Staring with an initial population (Section 2.3), it alternates between a local search procedure (Section 2.4) and a recombination procedure (Section 2.5). The pseudo-code of the proposed MA is presented in Algorithm 1. The algorithm first fills the population P with $|P|$ local optimal solutions provided by the local search procedure and then performs a series of generations. At each generation, two parent solutions φ_F and φ_M are selected at random from the population and are recombined to generate an offspring solution φ_C. Then, the local search is used to improve the offspring solution to attain a new local optimal solution. Finally, the

improved solution is used to update the population (Section 2.6). This process is repeated until a fixed number of generations ($MaxGene$) is reached.

Algorithm 1 Pseudo-code of general procedure

1: **Input**: Finite undirected graph $G(V, E)$, fitness function B_C, fixed size of population $|P|$ and maximum generations $MaxGene$
2: **Output**: The best solution found φ^*
3: $P = \{\varphi^1, \varphi^2, ...\varphi^{|P|}\} \leftarrow Init\_Population()$
4: $\varphi^* \leftarrow Best(P)$
5: **for** $i = 1$ to $|P|$ **do**
6: $\quad \varphi^i \leftarrow Local\_Search(\varphi^i)$
7: $\quad$ **if** $B_C(G, \varphi^i) < B_C(G, \varphi^*)$ **then**
8: $\quad\quad \varphi^* \leftarrow \varphi^i$
9: $\quad$ **end if**
10: **end for**
11: **for** $j = 1$ to $MaxGene$ **do**
12: $\quad \varphi_F, \varphi_M \leftarrow Parent\_Selection(P)$
13: $\quad \varphi_C \leftarrow Recombination\_Sol(\varphi_F, \varphi_M)$
14: $\quad \varphi_C \leftarrow Local\_Search(\varphi_C)$
15: $\quad$ **if** $B_C(G, \varphi_C) < B_C(G, \varphi^*)$ **then**
16: $\quad\quad \varphi^* \leftarrow \varphi_C$
17: $\quad$ **end if**
18: $\quad P \leftarrow Update\_Pop(\varphi_C, P)$
19: $\quad j \leftarrow j + 1$
20: **end for**
21: **return** φ^*

2.3 Initialization

In the initialization procedure ($Ini\_Population$), $|P|$ embeddings are generated randomly and independently at first. And then each embedding is improved by the local search procedure of Section 2.4 to attain a local optimum (lines 5-10, Alg. 1). The best solution φ^* in P is also recorded, which is updated during the subsequent search, each time an improved best solution is encountered.

2.4 Local search

Local search (LS) is an important component of the memetic algorithm, which aims to improve the input solution by searching a given neighborhood. In this work, we apply a simple Descent Local Search (DLS) in order to highlight the contributions of the recombination component.

DLS adopts the swap-based neighborhood of [23], where a neighboring solution of a given solution φ is obtained by simply swapping the labels of two vertices of φ. To specify the neighborhood, we first define, for a vertex u, its cyclic bandwidth $B_C(u, \varphi)$ with respect to the embedding φ as follows:

$$B_C(u,\varphi) = \max_{v \in A(u)} \{|l(u) - l(v)|_n\}, \tag{2}$$

where $A(u)$ denotes the set of vertices adjacent to u of cardinality $deg(u)$. Then the set of critical vertices is given by:

$$C(\varphi) = \{u \in V : B_C(u,\varphi) = B_C(G,\varphi)\}. \tag{3}$$

The neighborhood is defined as follows:

$$N(\varphi) = \{\varphi' = \varphi \oplus swap(u,v) : u \in C(\varphi), v \in V\}. \tag{4}$$

DLS starts with an input embedding, then it iteratively visits a series of neighboring solutions of increasing quality according to the given neighborhood. At each iteration, only solutions with a better objective value are considered and the best one is used to replace the incumbent solution. If there exist multiple best solutions, the first one encountered is adopted. We repeat this process until no better solution exists in the neighborhood. In this case, DLS attains a local optimum and the procedure of recombination is triggered to escape from the local optimum.

2.5 Recombination

Recombination is another important part of the MA, which aims to generate new diversified and potentially improving solutions. In our case, only one offspring solution is generated at each generation by each recombination application. In Section 3, we present five permutation recombination operators applied to CBP.

2.6 Updating population

Each new offspring solution improved by the local search procedure is used to update the population. In the proposed MA, we apply a simple strategy: we insert the new offspring into P, and remove the "worst" solution in terms of the objective value.

3 Recombination operators

There are several recombination operators that are already applied to permutation problems [6, 8, 9, 18, 26]. We consider five crossover operators introduced below. It is worth noting that all the recombination operations work with the γ representation mentioned in Section 2.1.

3.1 Order Crossover

The Order Crossover operator (OX) [6] generates an offspring solution with a substring of one parent solution and conserves the relative order of the numbers of the other parent solution. Let's consider an example with two parent solutions φ_F=(1 2 3 4 5 6 7 8) and φ_M=(2 4 6 8 7 5 3 1) (each number here denotes the index of a node). Given two random cut points (in this case, the first cut point is between second and third positions and the second cut point is between fifth and sixth positions, i.e., φ_F=(1 2 | 3 4 5 | 6 7 8) and φ_M=(2 4 | 6 8 7 | 5 3 1), two offspring solutions first inherit the substring between the two cut points: φ_{C1}=(+ + | 3 4 5 | + + +) and φ_{C2}=(+ + | 6 8 7 | + + +). Then, we copy the permutation starting from the second cut point of φ_M to the end, as well as from the beginning to the second cut point: (5 3 1 2 4 6 8 7). At last, the new obtained permutation is used to insert into φ_{C1} from the second cut point. The repeated numbers are skipped and we get φ_{C1}=(8 7 | 3 4 5 | 1 2 6). The same operations are performed on φ_{C2} with φ_F to get φ_{C2}=(4 5 | 6 8 7 | 1 2 3).

3.2 Order-based Crossover

The Order-based Crossover operator (OX2) [26] is a modified version of OX. Instead of choosing two cut points, OX2 chooses several random positions of one parent solution, and then the order of the selected positions is imposed on the other parent solution. For instance, we have two parent solutions φ_F=(1 2 3 4 5 6 7 8) and φ_M=(2 4 6 8 7 5 3 1), and the second, third and sixth positions are picked for φ_M. So the order of "4 6 5" is kept. For solution φ_F, we remove the corresponding numbers of these positions to get (1 2 3 + + + 7 8). Then we insert the numbers in the order "4 6 5" into φ_F and we get the offspring solution φ_{C1}=(1 2 3 4 6 5 7 8). The same operation can be performed for φ_M to obtain the other offspring solution φ_{C2}=(2 4 3 8 7 5 6 1).

3.3 Cycle Crossover

The Cycle Crossover operator (CX) [18] seeks a way to preserve the common information in both parent solutions. Two new offspring solutions φ_{C1} and φ_{C2} are created from two parents φ_F and φ_M where the number of each position in φ_{C1} and φ_{C2} is decided by the number of the corresponding position of one parent. For example, we consider two parent solutions φ_F=(1 2 3 4 5 6 7 8) and φ_M=(2 4 6 8 7 5 3 1). Firstly, the number of the first position of φ_{C1} could be 1 or 2, Supposing that we pick 1 here (1 + + + + + + +). Then, the number of the eighth position could not be 1 because it is already assigned to the first position, hence we allocate it with a number from φ_F to get (1 + + + + + + 8). After that, we find the position of φ_M whose number is 8 and assign the number of φ_F to the corresponding position of φ_{C1}. We repeat the same operation and find that the forth and the second number of φ_{C1} come from φ_F, which leads to (1 2 + 4 + + + 8). For the rest of the positions, we fill them with the numbers from φ_M to obtain a complete offspring solution φ_{C1}=(1 2 6 4 7 5 3 8). Similarly, we could get the other offspring solution φ_{C2}=(2 4 3 8 5 6 7 1).

3.4 Partially Mapped Crossover

The Partially Mapped Crossover operator (PMX) [9] passes the absolute position information from the parent solutions to the offspring solutions. An offspring solution gets a substring from one parent and its remaining positions take the values of the other parent. For example, we consider again φ_F=(1 2 3 4 5 6 7 8) and φ_M=(2 4 6 8 7 5 3 1). At the beginning, two random cut points are chosen for both parent solutions: φ_F=(1 2 3 | 4 5 6 | 7 8) and φ_M=(2 4 6 | 8 7 5 | 3 1). Then we pass the information between the two cut points to the offspring solutions: φ_{C1}=(+ + + | 4 5 6 | + +) and φ_{C2}=(+ + + | 8 7 5 | + +). Also, we get the mapping for the substrings between the two cut points: 4↔8, 5↔7, 6↔5. After that, the other positions of the offspring solutions are filled with the other parent solution, hence we get φ_{C1}=(2 4 6 | 4 5 6 | 3 1) and φ_{C2} =(1 2 3 | 8 7 5 | 7 8). For the duplicate labels in the solution, we use the mapping of substrings to replace the repeated numbers. In this case, 5↔7 and 6↔5 result in 6↔7. Therefore, the offspring solutions are φ_{C1}=(2 8 7 | 4 5 6 | 3 1) and φ_{C2}=(1 2 3 | 8 7 5 | 6 4).

3.5 Distance Preserved Crossover

The Distance Preserved Crossover operator (DPX) [8], designed for solving the Traveling Salesman Problem (TSP), aims to produce an offspring solution which has the same distance to each of its parents. It is noteworthy that the distance here is the distance based on the common connections between two solutions, instead of the Hamming distance. We come back to this issue in Section 5. For DPX, we firstly delete the uncommon connections of two neighboring numbers for both parent solutions. Then, the parent solutions are separated into different substrings. Finally, we reconnect all the substrings without using any of the connections which are contained in only one of the parent solutions. For more detailed explanations and examples, please refer to [8].

4 Experimental results

4.1 Instances and settings

In this section, we report experimental results of the MA using the 5 different recombination operators introduced in Section 3. The study was based on 20 representative graphs with 59 to 2048 vertices, selected from a test-suite of 113 benchmark instances (`https://www.tamps.cinvestav.mx/~ertello/cbmp.php`). 14 of the chosen graphs are standard graphs covering 7 dissimilar categories (path, cycle, complete tree, 2-dimension mesh, 3-dimension mesh, caterpillar and hypercube) and the other 6 graphs (called Harwell-Boeing graphs) come from real-world scientific and engineering applications and are part of the Harwell-Boeing Sparse Matrix Collection. Considering the stochastic nature of the algorithm, each instance was independently solved 50 times under the environment of Linux using an Intel Xeon E5-2695 2.1 GHz CPU and 2GB RAM. Each execution was

limited to 20000 generations ($MaxGene = 20000$) and the population size $|P|$ was set to 20.

4.2 Computional results

Table 1 outlines the computational results of our MA variants with the 5 different recombination operators. The columns"Best" and "Avg" list the best and average objective values found. According to the definition introduced in Section 1, a smaller objective value indicates a better result. Table 1 shows that the algorithm with OX2 obtains the best results not only in terms of "Best" but also in terms of "Avg" over the 20 test instances. From the average values listed in the last row, we find that OX2 is a much more suitable operator than the other operators for CBP. Also, the non-parametric Friedman test on the 5 groups of best results leads to a p-value=6.71e-14 < 0.05, confirming that there exists a statistically significant difference among the compared results.

	CX		DPX		OX		OX2		PMX	
Graph	Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg
nos6	327	331.28	327	329.74	266	287.98	**216**	**227.84**	327	331.98
path1000	461	475.42	462	474.02	254	301.04	**226**	**247.54**	468	482.68
nos4	44	46.12	43	45.24	32	39.32	**28**	**34.48**	42	45.78
tree10x2	39	42.72	35	40.72	**28**	32.50	**28**	**29.26**	36	41.56
cycle1000	457	476.66	466	473.38	252	296.98	**226**	**246.94**	459	480.86
mesh2D8x25	88	93.04	89	91.82	59	75.18	**57**	**62.94**	87	93.28
caterpillar29	203	211.48	203	208.70	138	162.98	**100**	**127.32**	198	210.14
mesh3D6	102	103.88	101	102.96	86	93.08	**73**	**78.26**	102	104.28
hypercube11	1022	1022.76	1022	1022.14	1019	1021.26	**952**	**1010.48**	1022	1022.54
cycle475	200	215.16	206	213.36	105	128.36	**99**	**110.76**	192	217.30
mesh2D28x30	409	413.40	410	412.06	336	371.76	**270**	**287.46**	406	414.06
mesh3D11	660	662.04	660	661.28	625	650.30	**507**	**522.82**	660	662.40
can__715	354	355.80	355	355.14	347	353.92	**293**	**316.70**	354	355.74
impcol_b	28	28.46	27	27.96	25	27.22	**20**	**26.72**	28	28.00
path475	202	214.50	206	212.86	112	132.24	**102**	**112.94**	189	217.56
494_bus	220	230.76	222	228.72	135	165.74	**128**	**138.62**	216	233.38
tree21x2	199	212.08	203	208.96	139	171.34	**124**	**140.84**	200	210.68
caterpillar44	481	493.28	479	491.24	340	400.78	**281**	**321.70**	480	495.60
impcol_d	207	209.60	207	208.80	190	202.98	**159**	**169.74**	208	209.80
tree2x9	475	489.08	478	485.86	296	330.14	**257**	**276.60**	472	491.84
Average	308.90	316.38	310.50	315.75	239.20	262.26	**207.30**	**224.50**	307.30	317.47
p-value	6.71e-14									

Table 1. Experimental results of MA using 5 different recombination operators.

Table 2 reports the comparative results between the best MA with OX2 (called MA_{OX2}) and TS_{CB}, which is the state-of-art algorithm for CBP presented in [23]. Table 2 shows the same information as in Table 1, except for the column "CC" which represents the difference between the best values found by TS_{CB} and MA_{OX2}. A negative "CC" indicates a worse result of MA_{OX2} compared to TS_{CB}. It is clear that for the 20 test graphs, MA_{OX2} does not compete well with TS_{CB}. Indeed, TS_{CB} is a powerful iterated tabu search algorithm which uses three dedicated neighborhoods to effectively explore the search

space. Also, the Wilcoxon signed-rank test with the two groups of best values leads to a p-value=1.31e-4 < 0.05, confirming the statistical significance between the compared results. This comparison tends to indicate that in practice, it is not enough for the MA to apply a recombination operator and a simple local search. In addition to a suitable recombination operator, the MA needs a powerful local optimization procedure to ensure an effective exploitation.

	MA_{OX2}		TS_{CB}		
Graph	Best	Avg	Best	Avg	CC
nos6	216	227.84	22	23.50	-194
path1000	226	247.54	8	8.90	-218
nos4	28	34.48	10	10.00	-18
tree10x2	28	29.26	28	28.00	0
cycle1000	226	246.94	8	8.50	-218
mesh2D8x25	57	62.94	8	8.20	-49
caterpillar29	100	127.32	24	25.80	-76
mesh3D6	73	78.26	31	31.00	-42
hypercube11	952	1010.48	570	582.20	-382
cycle475	99	110.76	5	5.80	-94
mesh2D28x30	270	287.46	30	174.00	-240
mesh3D11	507	522.82	336	336.80	-171
can_715	293	316.70	60	65.80	-233
impcol_b	20	26.72	17	17.00	-3
path475	102	112.94	5	5.60	-97
494_bus	128	138.62	46	56.10	-82
tree21x2	124	140.84	116	116.00	-8
caterpillar44	281	321.70	39	54.00	-242
impcol_d	159	169.74	38	43.10	-121
tree2x9	257	276.60	63	64.20	-194
Average	207.30	224.50	**73.20**	**83.23**	
p-value	1.31e-4				

Table 2. Comparison between MA_{OX2} and TS_{CB} [23].

5 Understanding the performance differences of the compared crossovers

In Section 4, we observe that OX2 excels compared to the other crossover operators. In this section, we investigate the reasons why OX2 has a better performance than the other crossovers. For this, we follow [27] and study the evolution of the population diversity. To this end, we consider two diversity indicators: average solution distance $D_{avg}(P)$ and population entropy $E_p(P)$.

5.1 Distance and Population Entropy

We first introduce the average solution distance $D_{avg}(P)$ of the population.

$$D_{avg}(P) = \frac{2}{|P|(|P|-1)} \sum_{i=1}^{|P|} \sum_{j=i+1}^{|P|} d_{ij} \tag{5}$$

where d_{ij} is the distance between two solutions γ_i and γ_j of P, which is defined as the number of the adjacent connections that are contained in γ_i but not in γ_j. For example, given two solutions γ_1={h a b d e f c g} and γ_2={b a c h g d f e}. The set of adjacent connections is {ha, ab, bd, de, ef, fc, cg, gh} for γ_1 and {ba, ac, ch, hg, gd, df, fe, eb} for γ_2. The common adjacent connections are {ab, ef, gh} (ba and ab are the same connections). The distance d_{12} equals thus 8-3=5. This distance is used in [8] to deal with TSP whose solutions have the symmetry feature. As shown in Fig. 1, CBP solutions have the feature of symmetry, so the use of this distance measure is very important for CBP.

Another indicator to describe the population diversity is the population entropy $E_p(P)$ [7].

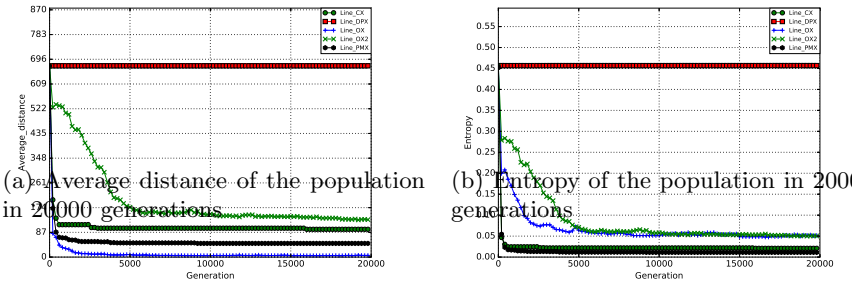
$$E_p(P) = \frac{-\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{n_{ij}}{|P|}\right)\log\left(\frac{n_{ij}}{|P|}\right)}{n\log n} \tag{6}$$

where n_{ij} represents the number of times that variable i is set to value j in all solutions in P. One notices that $E_p(P)$ varies in the interval [0,1]. When $E_p(P)$ equals 0, all the solutions of P are identical. A large $E_p(P)$ value indicates a more diverse population.

The instance 'nos6' is a representative large graph with 675 nodes from practical application and rather difficult, so we use it as a working example. Figure 2 shows the average distance, average entropy and average best objective value found in 50 independent executions over the graph 'nos6'. Under 5000 generations, the population of the MA with OX2 has a high average distance and entropy, leading to much better solutions. From generations 5000 to 20000, the entropy is identical to that of OX, and the best average objective found stops decreasing. These observations remain valid for all test graphs except the graph 'impcol_b' (even if the MA with OX2 does not have a large population distance and entropy, it gets good results comparing to others). Therefore, for the operators CX, OX, OX2 and PMX, a higher entropy and average distance of population leads to a good quality solution. However, what is surprising is that the average distance and entropy with DPX always stay at a high level for all test graphs, yet the quality of solutions found is not as good as that of the other operators. To shed light on this behavior, we show a deeper analysis of the interaction between the crossover mechanism and the characteristics of problem in the next section.

5.2 Interaction between crossover and problem characteristics

In Section 5.1, we find that the recombination operator with a higher entropy and average distance of the population generally helps to find solutions of good quality. However, the DPX operator fails to reach good solutions even if the entropy and average distance of population under the MA with DPX always stay at a high level. From Figure 3, which presents the average objective value of the offspring solutions of instance nos6 using the average data of 50 independent



(a) Average distance of the population in 20000 generations

(b) Entropy of the population in 20000 generations

(c) Best objective value found in 20000 generations

Fig. 2. Distance and population entropy applied to the instance nos6.

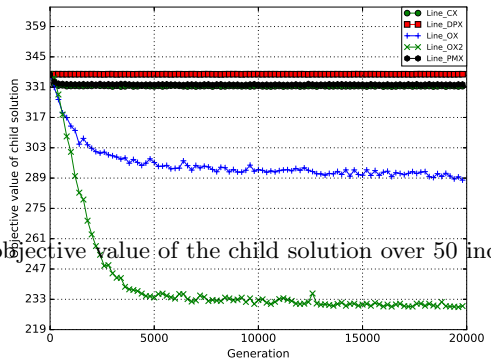


Fig. 3. Average objective value of the child solution over 50 independent executions.

executions, we find that DPX does not generate high quality offspring solutions during the search.

To understand why DPX does not help the MA to find good quality solutions, we first recall that DPX is designed for TSP, which is a quite different problem compared to CBP considered in this work. In [4], it is observed that for TSP, the average distance between local optima is similar to the average distance between a local optimum and the global optimum and common substrings in the local optima also appear in the global optimum. DPX explores this particular feature of TSP and is thus suitable to TSP. However, CBP has a totally different objective function and does not share the above characteristic.

Indeed, to calculate the objective value of a solution of TSP, we only need to consider, for each vertex, its two linked edges and sum up the edge distances of the tour. In this case, solution sub-tours (substrings) are clearly a key component which characterizes the solutions. Yet in a solution of CBP, we need to consider for each vertex all the edges linked to the vertex in the graph, such that the objective value (see Equation (1)) relies on the largest cyclic bandwidth. In the case of CBP, the key point is the relative position for the pairs of nodes which are linked by an edge. Therefore given that TSP and CBP have very different characteristics, a good crossover operator designed for TSP (in our case, DPX) may fail when it is applied to CBP.

This inspires us that the choice and design of recombination operators are not only relied on the entropy and average distance of population, but also on the characteristics of the considered problem.

6 Conclusion

In this paper, we have investigated the memetic framework for solving the NP-hard Cyclic Bandwidth problem. We have compared five permutation recombination operators (CX, OX, OX2, PMX and DPX) within a basic memetic algorithm which uses a simple descent procedure for local optimization. The experimental results indicate that OX2 achieves the best performance for the test instances. We have studied the population diversity measured by the average distance and entropy of the MA variants using different recombination operators. We have also explored the correlation between the population diversity and the performance of the studies MA variants. This study indicates that the basic memetic algorithm combining an existing recombination operator and a simple descent local search procedure is not competitive compared to the state-of-the-art CBP algorithms. Additional (preliminary) experiments with MAs using an enforced local optimization procedure (such as the powerful local search algorithms presented in [19, 23]) have not led to more convincing results. Meanwhile, given the excellent performances achieved by MAs on many difficult optimization problems, this work invites more research effort on seeking meaningful recombination operators suitable for CBP. It is then expected that a MA integrating such a recombination operator and a powerful local optimization procedure would achieve state-of-the-art performances.

Acknowledgments

We are grateful to the referees for their valuable suggestions and comments which helped us to improve the paper. Support from the China Scholarship Council (CSC, Grant 201608070103) for the first author and support from the Mexican Secretariat of Public Education through SEP-CINVESTAV (2019–2020, Grant 00114) for the third author are also acknowledged.

References

1. Bansal, R., Srivastava, K.: A memetic algorithm for the cyclic antibandwidth maximization problem. Soft Computing **15**(2), 397–412 (2011)
2. Benlic, U., Hao, J.K.: Memetic search for the quadratic assignment problem. Expert Systems with Applications **42**(1), 584–595 (2015)
3. Bhatt, S.N., Leighton, F.T.: A framework for solving VLSI graph layout problems. Journal of Computer and System Sciences **28**(2), 300–343 (1984)
4. Boese, K.D.: Cost versus distance in the traveling salesman problem. UCLA Computer Science Department Los Angeles (1995)
5. Chen, Y., Hao, J.K.: Memetic search for the generalized quadratic multiple knapsack problem. IEEE Transactions on Evolutionary Computation **20**(6), 908–923 (2016)
6. Davis, L.: Applying adaptive algorithms to epistatic domains. In: International Joint Conference on Artificial Intelligence. vol. 85, pp. 162–164 (1985)
7. Fleurent, C., A. Ferland, J.: Object-oriented implementation of heuristic search methods for graph coloring. Cliques, Coloring, and Satisfiability. DIMACS Series in Discrete Mathematics and Theoretical Computer Science **26**, 619–652 (1996)
8. Freisleben, B., Merz, P.: A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems. In: Proceedings of IEEE International Conference on Evolutionary Computation. pp. 616–621. IEEE (1996)
9. Goldberg, D.E., Lingle, R., et al.: Alleles, loci, and the traveling salesman problem. In: Proceedings of International Conference on Genetic Algorithms and Their Applications. Lawrence Erlbaum, Hillsdale, NJ, vol. 154, pp. 154–159 (1985)
10. Hao, J.K.: Memetic Algorithms in Discrete Optimization. In F. Neri, C. Cotta, P. Moscato (Eds.) Handbook of Memetic Algorithms. Studies in Computational Intelligence 379, Chapter 6, pp. 73–94 (2012)
11. Jin, Y., Hao, J.K., Hamiez, J.P.: A memetic algorithm for the minimum sum coloring problem. Computers & Operations Research **43**, 318–327 (2014)
12. Krasnogor, N., Smith, J.: A tutorial for competent memetic algorithms: model, taxonomy, and design issues. IEEE Transactions on Evolutionary Computation **9**(5), 474–488 (2005)
13. Lai, X., Hao, J.K.: A tabu search based memetic algorithm for the max-mean dispersion problem. Computers & Operations Research **72**, 118–127 (2016)
14. Leung, J.Y., Vornberger, O., Witthoff, J.D.: On some variants of the bandwidth minimization problem. SIAM Journal on Computing **13**(3), 650–667 (1984)
15. Lin, Y.: The cyclic bandwidth problem. Chinese Science Abstracts Series A **14**(2 Part A), 14 (1995)
16. Merz, P., Freisleben, B.: Memetic algorithms for the traveling salesman problem. Complex Systems **13**, 297–345 (1997)

17. Moscato, P., Cotta, C.: A Gentle Introduction to Memetic Algorithms, pp. 105–144. Springer US, Boston, MA (2003)
18. Oliver, I., Smith, D., Holland, J.: A study of permutation crossover operators on the travelling salesman problem, genetic algorithms and their applications. In: Proceedings of theupdating pop Second International Conference on Genetic Algorithms. pp. 224–230 (1987)
19. Ren, J, Hao, J.K., Rodriguez-Tello, E.: An iterated three-phase search approach for solving the Cyclic Bandwidth Problem. IEEE Access **7**, 98436-9845 (2019)
20. Rodriguez-Tello, E., Betancourt, L.C.: An improved memetic algorithm for the antibandwidth problem. In: International Conference on Artificial Evolution (Evolution Artificielle). pp. 121–132. Springer (2011)
21. Rodriguez-Tello, E., Hao, J.K., Torres-Jimenez, J.: An improved simulated annealing algorithm for bandwidth minimization. European Journal of Operational Research **185**(3), 1319–1335 (2008)
22. Rodriguez-Tello, E., Narvaez-Teran, V., Lardeux, F.: Comparative study of different memetic algorithm configurations for the cyclic bandwidth sum problem. Parallel Problem Solving from Nature, PPSN XV. pp. 82–94. Springer, Cham (2018)
23. Rodriguez-Tello, E., Romero-Monsivais, H., Ramirez-Torres, G., Lardeux, F.: Tabu search for the cyclic bandwidth problem. Computers & Operations Research **57**, 17–32 (2015)
24. Romero-Monsivais, H., Rodriguez-Tello, E., Ramírez, G.: A new branch and bound algorithm for the cyclic bandwidth problem. Advances in Computational Intelligence. MICAI 2012. Lecture Notes in Computer Science, vol 7630. Springer, Berlin, Heidelberg. pp. 139–150 (2012)
25. Rosenberg, A.L., Snyder, L.: Bounds on the costs of data encodings. Mathematical Systems Theory **12**(1), 9–39 (1978)
26. Syswerda, G.: Scheduling optimization using genetic algorithms. Handbook of Genetic Algorithms. pp. 322–349 (1991)
27. Wang, Y., Lü, Z., Hao, J.K.: A study of multi-parent crossover operators in a memetic algorithm. Parallel Problem Solving from Nature, PPSN XI. LNCS, vol 6238. Springer, Berlin, Heidelberg, pp. 556–565 Springer (2010)
28. Wu, Q., Hao, J.K.: Memetic search for the max-bisection problem. Computers & Operations Research **40**(1), 166–179 (2013)
29. Zhou, Y., Hao, J., Glover, F.: Memetic search for identifying critical nodes in sparse graphs. IEEE Transactions on Cybernetics **49**(10): 3699-3712 (2019)